P-Cycle-based Protection in Network Virtualization

Yihong Song

A thesis submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

MASTER OF APPLIED SCIENCE

in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
University of Ottawa
Ottawa, Canada

2013

Abstract

As the “network of network”, the Internet has been playing a central and crucial role in modern society, culture, knowledge, businesses and so on in a period of over two decades by supporting a wide variety of network technologies and applications. However, due to its popularity and multi-provider nature, the future development of the Internet is limited to simple incremental updates.

To address this challenge, network virtualization has been propounded as a potential candidate to provide the essential basis for the future Internet architecture. Network virtualization is capable of providing an open and flexible networking environment in which service providers are allowed to dynamically compose multiple coexisting heterogeneous virtual networks on a shared substrate network. Such a flexible environment will foster the deployment of diversified services and applications.

A major challenge in network virtualization area is the Virtual Network Embedding (VNE), which aims to statically or dynamically allocate virtual nodes and virtual links on substrate resources, physical nodes and paths. Making effective use of substrate resources requires high-efficient and survivable VNE techniques. The main contribution of this thesis is two high-performance *p*-Cycle-based survivable virtual network embedding approaches. These approaches take advantage of *p*-Cycle-based protection techniques that minimize the backup resources while providing a full VN protection scheme against link and node failures.

Acknowledgements

I must first express my deepest gratitude to my supervisor, Professor Ahmed Karmouch, for his patient guidance, support and confidence in me throughout my studies. His valuable technical helps, encouragements and constructive suggestions were a source of inspiration and motivation all along my way.

Secondly I am really grateful to Dr. Abdallah Jarray, who has enriched and refined my knowledge with his valuable ideas and comments. I would also like to thank Yousif Al Ridhawi, Imad Abdeljaouad, Ismaeel Al Ridhawi, Bassem Wanis, Heli Dimuth Amarasinghe and other members of the Intelligence for Mobile Autonomic and Cognitive Networks Laboratory for their help and support during my research. Their companionship and friendship eased many tasks during times of difficulty. Finally, I want to dedicate this thesis to my parents who have given me the chance of a good education, and so much love and support over the years. I probably owe them much more then I think. To you I dedicate this thesis, for teaching me that nothing in life is impossible.

Table of Contents

List of Figures

List of Tables

Acronyms

BLSR	Bi-directional Line Switched Ring
CDMA	Code Division Multiple Access
CG	Column Generation
FCFS	First Come First Served
FDMA	Frequency Division Multiple Access
FIPP	Failure-Independent Path-Protecting *p*-Cycles
IECs	Independent Embedding Configurations
InP	Infrastructure Provider
IP	Internet Protocol
ISP	Internet Service Providers
ILP	Integer Linear Programming
JNLE-CG	Join Node and Link Embedding using CG
MIP	Mixed Integer Programming
MPLS	Multiprotocol Label Switching
NEPC	Node-encircling *p*-Cycle
p-Cycle	pre-planned, preconfigured Cycles
p-Cycle-SPP	*p*-Cycle-based Span-and-path Protection Technique
p-Cycle-VSP	*p*-Cycle-based Virtual Span Protection Technique
p-Cycle-EPP	*p*-Cycle Embedding Path Protection
QoS	Quality of Service
RVNE	Resilient Virtual Network Embedding
SBPP	Shared Backup Path Protection
SDMA	Space Division Multiple Access
SFE-DPP	Stress-Function based Embedding Combined Disjoint Path Protection
SLA	Service Level Agreement

SLP-2PE	Substrate Link Protection combined with Two-Phases embedding
SMS	Short Message Service
SP	Service Providers
TDMA	Time Division Multiple Access
VN	Virtual Network
VNE	Virtual Network Embedding
VNM-CNLP	Virtual Network Mapping with Combined Node and Logical Links Failures Protection
VNP	Virtual Network Provider
WDM	Wavelength-Division Multiplexing

1. Introduction

1.1 Motivation

In the past few decades, the Internet has created a great revolution in people's lives. As the main enabler of our communication and information era, the Internet can support a multitude of distributed applications and a wide variety of network technologies. In addition, the Internet plays a critical role in the business, education, entertainment and social lives of people. The amazing rapid growth and wide deployment of network technology indicates the success of the Internet, but at the same time it also creates obstacles to its future innovations. Due to the huge size and scale of the Internet, and the conflicting goals and policies among multiple existing stakeholders, any deployment of novel network technology and architecture, or any modification of existing ones is nearly impossible under the current Internet situation.

To mitigate this impasse, network virtualization has been widely propounded as a fundamental diversifying attribute of the future Internetworking paradigm. This allows multiple heterogeneous network architectures, Virtual Networks (VNs), to coexist on a shared substrate [1], [2], [3]. Each virtual network in a network virtualization environment is a collection of virtual nodes and virtual links, which essentially are a subset of the substrate network resources. In the network virtualization environment, every VN is isolated from each other and it is free to implement its own naming, addressing and transporting mechanism. People believe that such a flexible environment

will foster the deployment of diversified services and applications, and eradicate the ossifying forces of the Internet, and stimulate innovation [1], [2].

In existing literature, proposals for network virtualization architecture have been presented in several contexts. For convenience, in this work I will adopt 4WARD architecture[4], where the role of traditional Internet Service Providers (ISPs) is divided into three main roles: 1) Infrastructure Providers (InPs), which provide and manage the physical infrastructure (the substrate), 2) and Virtual Network Providers (VNPs), which are responsible for assembling virtual resources from one or multiple InPs into a virtual topology, 3) Virtual Network Operators (VNOs), which are responsible for the installation and operation of a VN over the virtual topology provided by the VNP, according to the needs of the service providers (SPs). These operations are realized in a tailored connectivity service. With support of network virtualization, multiple service providers can dynamically deploy and manage customized end-to-end services for end users by effectively sharing and utilizing underlying network resources.

As one of the key research directions in the network virtualization field, virtual network embedding is the resource allocation of the underlying physical network for each virtual network. Obviously, effective network virtualization needs high-efficient virtual network embedding algorithm, which is used to map a virtual network onto the physical substrate network while respecting the constraints on its nodes and links. However, when mapping virtual network onto substrate network, many factors, such as node and link constraints, online requests and diverse topologies, have to be taken into consideration and these factors make the VNE problem extremely difficult. Even if only node and link constraints are considered, and other factors are ignored, VNE is still a

NP-hard problem [5]. Previously proposed approaches in relevant literature [5]-[9] have focused on restricting the problem in particular scenarios that can be summarized as the following: (i) some approaches split the problem into two separates node and link mapping phases. Non-join node and link embedding may result in a high number of VN requests being blocked and resources being under-used, with a resulting reduced profit for InP. In addition, (ii) other approaches relax resources availability constraints, which may ensue in non-respect of Quality of Service (QoS) requirements of VNs over time.

Moreover, all these studies are done under the normal operation of substrate network and none of them provide the backup mechanisms to combat potential node/link failures. In fact, the network failures are common in both IP backbone and autonomous system. Resiliency of VN service is an urgent requirement when the service providers provide general VN services. Especially, in recent years, the data transfer rate is considerably large and more and more critical business users rely on communication network. Thus an interruption of service, even for a short period of time, can cause a large amount of data loss. This is not acceptable by business users.

To provide survivable VN service, a few works, such as [35] [36], have been dedicated to provide backup mapping mechanisms for failing physical links. Nevertheless, proposals in this respect are suffering from the following limitations: (i) are based on protection mechanism with a significant bandwidth consumption, resulting in worse resource usage and reduced profit for InPs, and (ii) offer no mechanism to optimize the link division between primary and re-routed traffic, which may reduce the room for embedding new VNs. Furthermore, compared to link, the protection for node failure is more desirable because the impact of node failure is more severe, especially in the

network virtualization environment, as it can prune several VNs simultaneously. Existing research focuses on providing node protection mechanism in general cases, such as optical and Multiprotocol Label Switching (MPLS) networks. It is apparent the unique challenges in network virtualization environment need more attention.

The contribution of this thesis is that we apply a popular and relative new protection technique — *p*-Cycle in the network virtualization environment and propose high cost-efficient and survivable virtual network embedding approaches for node and link failure.

1.2 Definition of Survivable Virtual Network Embedding

In this Section, the key terms “virtual network” and “survivable” are defined separately, and then a complete definition for survivable virtual network embedding is given.

Virtual network is a collection of virtual nodes interconnected by virtual links. A virtual node is a logical entity, which is constructed by partitioning substrate network equipment (e.g. router) resources such as CPU capacity and memory. By independently controlling its correspondent source subset, a virtual node is able to be isolated from other coexisting virtual nodes in the same substrate node. In other words, a particular substrate node may be hosting none or any number of virtual nodes. Similarly, a virtual link is constructed by partitioning interconnecting link resource (e.g. copper wire, optical, microwave) such as bandwidth, and guaranteeing the isolation from coexisting logical links, to function independently. As a result, virtual link is capable of spanning over multiple substrate links to interconnect an adjacent virtual node hosted by non-

adjacent substrate node. Moreover, the process of mapping a virtual network onto the physical substrate network while respecting the constraints on its nodes and links is called virtual network embedding.

In addition to the explanation of virtual network, the meaning of the terms “survivable” or “survivability” needs to be explained. There are a number of definitions of survivability. The one I use here emphasizes timeliness, survivability under failure and attack. According to [10], “survivability is the capability of a system to fulfill its mission in a timely manner, even in the presence of attacks or failures.” When it comes to virtual network, the survivability of a virtual network can be defined as the ability to continue providing full service or recovering full service rapidly in the presence of failures.

In conclusion, according to the meaning and explanation given above, the survivable virtual network embedding is a process of allocating a virtual network onto the physical substrate network. This process is carried out not only with respecting constraints on its nodes and links, but also with providing the capability to continue giving full service or recovering full service rapidly in the presence of failures.

1.3 Thesis Objectives

The goal of this thesis is to design and simulate a cost-efficient and survivable virtual network embedding scheme for the purpose of mapping virtual networks effectively and resiliently. This is implemented by using pre-configured protection cycle (*p*-Cycle) techniques. According to current literature, survivability of VN service is an urgent requirement for InP and a need exists for further research in this field. Therefore, a sophistic survivable virtual network embedding scheme must consist of the following

characteristics:

Solving VNE problem efficiently: VNE problem is an extremely challenging problem, because of its unique features, such as node and link constraints, admission control, online requests and diverse topologies. Even some of these properties are ignored and VNE problem can be reduced to multiway separator problem, which is still NP-hard. A good survivable VNE scheme should be a good VNE algorithm. On the one hand, it should be able to effectively solve multiple technical problems mentioned above. On the other hand, from the point of view of business, survivable VNE schemes should maximize the long-term average revenue, to obtain the greatest economic benefit.

Full Node and link protection in network virtualization environment: Differing from the protection for general cases, which have been investigated and discussed extensively, the VN protection problem is much more intractable. The key challenge is that the VN service request is online. Accordingly, a VN protection scheme should be capable of establishing backup paths and assigning enough backup bandwidth on these paths, without providing any information about future requests. In addition, a VN protection scheme needs to ensure that all virtual links are intact in the presence of failures. Also, a VN protection scheme should be able to fully recover single link failures as well as single node failures which may cause a multiple VN links failure in network virtualization environment.

Resource usage vs. service resiliency tradeoff: In almost every scenario that involves establishing a protection mechanism for a virtual network, the survivability of service is proportional to the amount of resource used for backup path. Invariably, strong survivability always results in high a cost and there is a tradeoff between the

survivability and the cost. The goal of this thesis is to provide full survivability while minimizing the cost used for protection. In order to reduce the resource cost of VN protection, the sharing of a backup path among multiple embedding paths is encouraged when building a protection scheme. However, on certain links, this could result in more bandwidth being reserved for protection, and limiting the room for embedding new VN requests. Moreover, a traffic congestion on the backup path could result from large amount of working load being designed for recovery on the span backup path whose resource cost is lowest. As a consequence, rules and restrictions related to QoS requirements should be set up to control the tradeoff between sharing and load balance.

1.4 Thesis Contributions

According to the characteristics highlighted in Section 1.3, an efficient survivable virtual network embedding approach has to embed and protect VN effectively while respecting QoS requirements on node and link. In this thesis, I propose cost-effective VN mapping and protection schemes while simultaneously considering main QoS requirements.

This thesis presents a comprehensive survey of recent virtual network embedding schemes by classifying them into different categories and analysing the features of each category. Also the thesis provides a literature review of previous node/link protection and restoration approaches proposed in network virtualization environment as a new category, i.e., survivable VNE, of VNE scheme.

Furthermore, in this thesis, a large scale optimization technique, Column Generation technique, proposed in [11] is utilized to handle the complexity issue of the VN embedding problem. In the stage of VN embedding, the effects of potential network

failures are taken into consideration.

To address the concerns about backup resource utilization and survivability, *p*-Cycle technique is introduced into VN protection and restoration and multiple *p*-Cycle–based VN protection mechanisms are proposed. These mechanisms focus separately on node protection and combined node and multiple virtual links protection. *P*-Cycle techniques can not only improve the efficiency of backup resource cost and recovery speed, but also optimize the tradeoff between sharing and load balance by setting pre-requests on those embedding paths which share the same backup path.

A link protection approach and a node protection approach are proposed separately because the challenges that need to be resolved are distinct from each other. In the scenario of link protection, concentration is placed on survivability guarantee of all the VN links simultaneously. While in the node protection approach, in order to guarantee a full VN protection against a single physical node failure and a multiple logical links failure, the main focus is on the physical layer protection mechanism while taking into account constraints from the logical layer (virtual networks).

1.5 Thesis Organization

The remainder of this thesis is structured as follows:

In Chapter 2, the background information and related work are presented which motivated us to investigate the survivable network virtualization embedding. First, some basic concepts are introduced in the research area of virtual network embedding. And then the classification and analysis for some existing approaches to the VNE problem are presented. Lastly, this Chapter presents a relatively new category in virtual network

embedding approaches – resilient virtual network embedding. An introduction of pre-configured cycle, which is used in our virtual network protection approach, is also given in this part.

Chapter 3 presents the small-batch provisioning initially and then performs a detailed analysis of the Resilient VNE approach. This approach consists of combining a cost-efficient join node and link mapping using Column Generation (CG) technique [11] and a VN protection (backup) mechanism based on *p*-Cycle concept. Two benchmarks which will be used to make a comparison with the approach in the performance evaluation are also presented.

In Chapter 4, another VN mapping with combined physical node and multiple logical links protection mechanism which also consist of VN embedding and VN protection is presented. Prior to that, the features of a node protection mechanism which is different from a link protection are presented. To conclude, a benchmark for comparison purpose is also presented.

In Chapter 5, the performances of both proposed approaches are evaluated by carrying out simulations. The objectives of the simulations are discussed and indispensable information related to simulation setup, configurations and results is provided. A comparison of the approaches with benchmarks is also given.

In Chapter 6, the thesis concludes with stating future research plans and potential enhancements to the current design.

2. Background and Related Work

2.1 Introduction of Virtual Network Embedding (VNE) Problem

2.1.1 Substrate Network

The physical infrastructure network S is represented by an undirected graph $G_s = (W_s , L_s)$, where W_s denotes the set of substrate nodes and L_s the set of bidirectional links. Figure 2.1 shows an example of a physical infrastructure network, where each physical link $l \in L_s$ offers a bandwidth capacity b_l (number over link) and each substrate node $u \in W_s$ has a CPU capacity p_u (number over node). We introduce a bandwidth unit cost c_l for each substrate link $l \in L_s$, for load balancing purpose. Similarly, we associate a CPU unit cost for each substrate node $u \in W_s$. We denote by Π^s the set of paths in graph G_s.

2.1.2 Virtual Network Request

Similarly, a Virtual Network $n \in N$ is represented by a directed graph $G_n(A_n, E_n)$. The QoS requirements of each virtual link $e \in E_n$ belonging to QoS class $j \in Q_1$ are defined by the couple (b_j , d_j), where b_j is the required bandwidth and d_j is the maximum number of switching nodes as an indirect way to upper-bound the end-to-end delay. We assume that the number of used links has a neglect effect on the end-to-end delay.

Similarly, QoS requirements of each virtual node $a \in A_n$ belonging to QoS class $j \in Q_2$ are defined by the couple (p_j, t_j), where p_j is the required CPU and t_j is the potential nodal mapping location. We denote by *c (a)* (resp. *c (e)*) the QoS class of virtual node *a* (resp. link *e*). Π_{uv}^{e} is the set of all shortest paths between embedding nodes (u, v) assigned to virtual link $e \in E_n$. Figure 2.1 shows an example of two VN requests VN_1 and VN_2. We assume that embedding of each VN request *n* provides the revenue P_n.

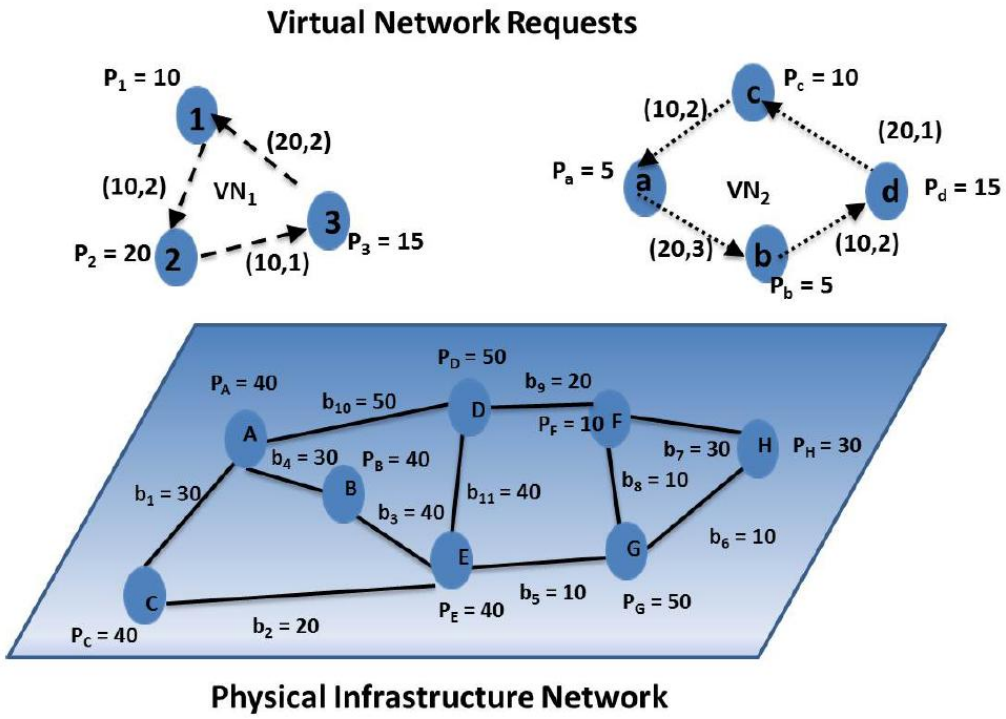


Figure 2.1: Substrate Network and Virtual Request [11]

2.1.3 Virtual Network Embedding

A virtual network embedding (called also mapping, I will exchange their use through the document) for a VN request is the process to assign the substrate network resources to the VN request, which can be decomposed into node and link mapping as follows.

1) Node mapping: Each virtual node $a \in A_n$ from the same VN request n is embedded to different substrate node $u \in W_s$ by mapping: $M_N : A_n \rightarrow W_s$

2) Link mapping: Similarly, each virtual link $e \in E_n$ from the same VN request n is embedded to different substrate path $\pi_{uv}^{e} \in \Pi^s$ by mapping: $M_L: E_n \rightarrow \Pi^s$, where (u, v) are substrate nodes assigned to virtual nodes (s, d) source and destination nodes of virtual link e respectively.

2.1.4 Objectives

The main objective of VNE is to use limited substrate network resources to provide services for as many as possible virtual network requests. From the point of view of InP, a natural objective of VNE algorithm would be to maximize the revenue [5]. Similar with previous researches [5] and [8], I denote by $P_n(t)$ the revenue of serving the VN request n at time t. Then, the objective is to maximize the long-term average revenue, given by the following:

$$\lim_{T \to \infty} \frac{\sum_{t=0}^{T} P_n(t)}{T} \tag{2.1}$$

The revenue can be defined in a variety of ways according to different economic models. Generally, bandwidth and CPU are regarded as the main substrate network resources. Hence the revenue for a VN request could be presented as the weighted sum of revenues for bandwidth and CPU, each of which is proportional to the amount of the requested resources. Similar to the work in [5], [8], we can introduce a tunable weight α that allows the substrate provider to strike a balance between the relative costs of the two classes of resources. Thus, for a VN n, we define its revenue $P[G_n(t)]$ at any particular time t that the virtual network G_n is running as:

$$P[G_n(t)] = \sum_{e \in E_n} bw(e) + \alpha \sum_{a \in A_n} CPU(a) \quad (2.2)$$

Where $bw(e)$ and $CPU(a)$ are the bandwidth and CPU requirements for the virtual link *e* and the virtual node *a* respectively. α is an adjustable weight to balance the proportion of resource allocation between node and link. From this equation, we can notice that the revenue is determined only by bandwidth and CPU but not affected by substrate path distance or some other factors, because VN requests only care about whether their constraints in substrate network can be satisfied or not. To achieve high revenue, the crucial part is to embed incoming VN requests efficiently, in which way the substrate resources is minimally occupied. An inefficient embedding of a virtual network at time *t*, which may restrict the substrate's ability to accept future requests, is not what we expect. In this work, we focus on CPU and bandwidth as the main substrate resources. Therefore the InP profit of each VN request *n* is calculated as follows.

$$REV[G_n] = P_n - COST\ [M_N(A_n), M_L(E_n)] \quad (2.3)$$

Where the first term calculates the offered VN revenue regards VN request *n* and the second term calculates the cost of assigned resources. We note that resources used for protection of VN primary flow do not incur a direct profit. Backup paths are considered as QoS requirements. In other words, we assume that are implicitly included in the Service Level Agreement (SLA) contracted between VNPs and InPs.

2.1.5 Parameters

In VNE research, there are many parameters that need to be considered and Figure 2.2 illustrates some common parameters considered in previous researches. Based on the

mutability, accessibility, and interdependency with others, parameters in VNE can be divided into three categories: primary parameters, secondary parameters and indirect parameters [12].

Primary parameters usually are inherent or fixed properties of a node or a link itself. These parameters are the capacities that don't depend on the state of another node or link, but only its utilization, such as CPU or memory of a node or bandwidth of a link. Primary parameters are directly specified in the VN request.

Parameters			
	Node	Primary	Geography location CPU Memory
		Secondary	Processing delay Energy consumption
	Link	Primary	Capacity Propagation delay
		Secondary	Transmission delay
	Node and Link	Primary	Unique Identifier
		Secondary	Packet loss probability
		Indirect	Resilience Shared risk group

Figure 2.2: A categorization of parameters can be considered in VNE research [12]

Differing from primary parameters, secondary parameters are variable since they are derived from primary parameters or other secondary parameters. For instance, the transmission delay is the sum of the processing delay on substrate nodes and the propagation and transmission delay on substrate links. Secondary parameters can also be

requested directly in a VN request.

Indirect parameters are requested potentially for a whole substrate path and are not directly requested by a VN. For example, the demand for a survivable VN comprises that primary path is disjoint from the backup path in the physical network and that the physical nodes and links fulfill certain failure probabilities.

Two important things should be noted. First, some parameters may be considered different in substrate network and virtual network. For example, CPU resource of a substrate node is a primary parameter while it is regarded as a secondary parameter after it is virtualized to be a part of a virtual link. Second, the constraints on these parameters cause challenging problems in VNE research area and the combination of these constraints make the VNE problem computationally difficult to solve.

2.1.6 Summary

Virtual network embedding involves the effective and efficient usage of limited substrate resources to map virtual network requests with as many constraints as possible. To conclude, this problem is simple to define but difficult to solve. The difficulty comes from four main particular reasons [5].

1) Node and link constraints of VN request. In order to satisfy a certain level of service, the VN usually has multiple specific constraints for the physical resources that embedding has to satisfy. These constraints are not only imposed on link such as bandwidth and transmit delay, but also constructed on nodes such as geography location and CPU. As an example, in some online games, the service may need virtual nodes to locate in several big cities and require 2 GHz of CPU for each virtual node as well as 4

Mbps for each virtual link, so that propagation delay will be less than 50 msec.

2) Admission control. Because the substrate resources are limited, not all the VN requests are able to be embedded in the same period of time, thus, some of them have to be postponed or dropped. For example, the bandwidth of one substrate link is 10 MB but 20 MB virtual link bandwidth is requested from different VN requests. To solve this problem, a higher priority must be given to some of them and others must be put into a waiting queue. Otherwise, the virtual links which don't need exclusive service can be put together and allowed to share one substrate link. Another solution is to use the First Come First Served (FCFS) strategy and drop the VN requests that cannot be served. The details of these strategies will be addressed in a subsequent Section.

3) Online request. In practice, we cannot predict the information of future VN requests, such as the time it arrives and departs, the period of time it will stay in the network or the constraints it requests. For example, a service provider will deploy a new service at any time and update or stop this service when it is no longer profitable. As a result, embedding algorithm has to handle online VN requests. Online problems are typically much more difficult to solve, because the uncertainty of VN requests makes it impossible to achieve an optimal resource allocation.

4) Diverse topologies. Different algorithms have different performance under various topology conditions. For instance, centralized algorithms are well suited to the hub-and-spoke topology, while they don't work well in some other topologies. No algorithm can guarantee that it supports arbitrary topology efficiently in the huge and complex world of the Internet today.

The four problems introduced above are so challenging that researchers usually only

focus on a few of them and ignore others, when they are developing VNE algorithms. However, even if only the node and link constraints are considered, the VNE problem is still a NP-hard problem. In previous literature, a variety of schemes have been proposed focusing on different aspects of this problem. In the following Section I will make a classification for current VNE approaches based on different standards and analyse the features of each classification.

2.2 Recent VNE Algorithm Classification and Analysis

Recently, many researchers have proposed algorithms for VNE problem from a variety of view of points. These can be classified in the following ways.

2.2.1 Centralization VNE Algorithm VS. Distribution VNE Algorithm

From the respect of computing manners, the VNE algorithms can be divided into centralized VNE algorithms and distribution VNE algorithms. Most existing VNE algorithms are based on centralized computing, such as [6],[13]. In these proposals, there is a central entity, usually named admission control, which is responsible for acknowledging and updating the information of physical network resources, and allocating virtual nodes to these resources when there is a request. While in distribute algorithms, the physical network is divided into multiple partitions first, and then in each partition, the mapping decision is done independently or cooperatively by distributed nodes [14] (hub nodes [15] or broker intermediaries [16]).

By using the global view and knowledge of substrate network, admission control can optimize the resource mapping and reduce the occurrence probability of conflicts and instability. However, maintaining a real-time global view of a whole physical network is not an easy job because it needs a high-efficient communication mechanism which will use a large percentage of overhead. In addition, the centralized VNE algorithms have limitations in process speed, scalability and fault tolerance, especially when the underlying physical network is highly dynamic. Examples include when demands might change on-line, the network environment state varies frequently and topology change happens from time to time.

The problems with centralized algorithms are main motivations for researchers to develop some VNE mapping methods in a decentralized way. Compared with a centralized way, distributed algorithm has an obvious advantage with process speed and failure recovery ability. Because the resource is delegated to distributed nodes, such distributed nodes can be regarded as “admission agents”, and these “admission agents” make their decisions in a parallel manner. However, to coordinate with other nodes, a node has to signal to other nodes, which also cause a huge overhead and low performance when the network is very large. Additionally, distributed algorithms allow nodes to make decisions using different algorithms respectively. This feature can be leveraged to build testbed for deploying different algorithms, e.g. ORCA (Open Resource Control Architecture). The following is a comparison table between centralization algorithm and distribution algorithm.

	Centralization	Distribution
Embedding Control	Admission control center	Distributed nodes(named hub or broker intermediaries)
Embedding mechanism	1) Acknowledging and updating information of physical network resource 2) allocating VN request	1) Divide physical network into multiple partitions 2) Mapping decision is done independently or cooperatively by distributed nodes
Advantages	1) Optimize the resource mapping 2) Low the occurrence probability of conflicts	1) High process speed; 2) Strong failure recovery ability
Disadvantages	1) Overhead is large when network is highly dynamic; 2) Limitation in fault tolerance, scalability and process speed;	Overhead is huge and performance is low when network is very large

Table 2.1: Centralization VNE algorithm vs. Distribution VNE Algorithm

2.2.2 Offline VNE Algorithm VS. Online VNE Algorithm

In this Section, the taxonomy is to classify VNE algorithms as offline and online. The differentiation between offline and online VNE algorithm is mainly based on the virtual network request. If all the information about virtual network requests, such as arrival time, duration and virtual network topology, are defined and specified in advance, this kind of algorithm is offline like [7],[8]; otherwise, it is online [17],[18]. In practice, a VN request cannot be predicted, and may arrive dynamically and stay for a random period of time. For example, a service provider may want to deploy a new service at any time and continue this service until it is no longer beneficial. In addition, the topology of virtual network may vary, which causes difficulty for researchers to efficiently map the

requests to substrate network.

To solve offline VNE problem, the main idea is to take a provided set of VN requests together with acknowledgement of substrate network and compute a near optimal mapping for these requests. However, developing a qualified online VNE algorithm is usually more complex than providing an offline one, because of the unique features of online problem. The common solution to designing online VNE algorithm is to redistribute resources on a FIFO-basis as the VN requests arrive.

Embedding VN requests by offline small batch is more profitable than serving VN requests one by one, i.e., online embedding. As a result, offline algorithm is more preferable for InPs.

2.2.3 VNE in Wired Network VS. Wireless Network

So far, most of the papers have focused on wired networks rather than wireless due to its popularity and precedence in deployment. However, the VN embedding problem in wireless substrate networks was not investigated until recent years and the research around this problem is still limited. This is due to various challenges that are unique in wireless networks and are not observable in wire network. The first challenge which is also the biggest difficulty in wireless network embedding is that wireless communication link has broadcast nature. As a result, in order to avoid affecting the virtual links and nodes of different VNs, a wireless node has to be divided into distinct wireless communication dimensions. For example, FDMA virtualizes a node in frequencies [19], and similarly, CDMA, SDMA, TDMA adopt node virtualization in code, space (e.g. [20]) and time (e.g. [21]) respectively. Apart from these wireless spectrum virtualization

methods, there is also an alternative for virtualization of the physical devices in the wireless network. The physical devices that usually are virtualized include mobile/cellular node [22]-[26], access point [27]-[29] and interfaces [30]-[32]. Second, when establishing a wireless link, i.e., a transmitter-receiver pair, in one channel which is mutually isolated with other channels, two important requirements have to be considered: 1) coherence between transmitter and receiver; and 2) isolation between different virtual nodes within one communication range. In addition, the bandwidth allocation problem, the considerable overhead produced by signalling and retransmission, and the tradeoff between flexibility and complexity also need to be solved.

However, undeniably, virtualization in wireless networks has a bright future, especially under the condition of the rapid development of mobile cloud, which "is currently one of the most over-hyped technology trends as it is considered that will pave the way for the Internet of Things"[33] .

2.2.4 QoS-aware VNE Algorithm

To achieve a high level of QoS, two difficult problems need to be solved: one is how to satisfy the nodes and link constraints of VN request, so that certain kinds of service can be applied in this VN; the other is a routing problem, which is related to traffic engineering.

In conventional mapping approaches, each VN request is provided with an exclusive service which allocates the whole bandwidth with full availability to the VN request. In reality, this kind of service is lacks flexible and efficient of serving all kinds of VN

demands. Some VNs don't require the exclusive service continuously, such as the VN for email, fax, SMS service. These services don't need to use all the reserved resources all the time and the subscriber of these services will not be happy if they are required to pay for the unnecessary service. Based on this motivation, paper [34] analyzes and applies the careful overbooking concept, the main idea of which is to provide Service Level Agreement (SLA) with flexible levels of availability, in VN embedding and proposes a SLA module. With the assumption that substrate network know the number of users connected to each link, many parts of VN demands which have the same source and destination address can be multiplexed into an aggregated demand. When this demand arrives at system with QoS parameters, the SLA module will calculate the level of availability, full availability or limited availability, and allocate necessary capacity to host that demand.

Furthermore, in order to find the best rout with minimum cost, a Mixed Integer Programming (MIP) module is developed based on Shortest Path First algorithm. At first, researchers use multiple constraints to determine a searching space. Then the Shortest Path First algorithm is applied to find out m least cost paths from the requested source and destination. Lastly, the traffic is routed through one of the best paths, but if an eligible path cannot be found, traffic will be routed in parallel. The multiple routing has advantages in capacity utilization, VN robustness and traffic diversity. However, it cannot guarantee the QoS in aggregated demands, so in this approach one path routing is preferable than multipath routing.

This approach can help infrastructure providers reduce the operation cost and lower the service price. As a result, more customers are attracted to use these services. However

the limitations of this approach also need to be solved. First of all, this approach focuses on those services that don't need exclusive resources. Some other approaches are needed to solve general cases. Additionally, if the network is highly dynamic and VN demands vary frequently, the overhead on substrate network multiplex will be quite huge.

2.2.5 VNE Algorithms with Load-balancing

In order to balance the load of substrate network, paper [8] proposes two kinds of virtual network embedding algorithms: VN embedding algorithm without reconfiguration (VNA-I) and VN embedding algorithm with configuration (VNA-II).

For the VNE algorithm (VNA-I), this paper proposes a baseline embedding algorithm. Instead of treating the node stress optimization and link stress optimization as two independent sub-problems, this algorithm considers both node and link stresses throughout the virtual network embedding process. The key idea of this baseline algorithm is to select a cluster of substrate nodes that are not only lightly loaded but also will probably to result in low substrate link stresses when they are connected. This could be achieved by identifying a cluster center in the substrate network based on the stress level in its neighbourhood. After identifying the cluster center, the rest of the substrate nodes for the VN request would be selected by considering not only the node stress but also the link stress weighted by its distance to other selected substrate nodes. After all substrate nodes are determined, the virtual network links would be embedded by using the shortest-distance path algorithm. Moreover, two methods are proposed to optimize this baseline algorithm. One is to subdivide the virtual network into several star topologies, each of which connect to each other, and then embed these sub-virtual

networks one by one. The other method is to use an adaptive optimization strategy that would test the stress ratio of nodes and links. If the stress ratio of nodes is larger than the links, which means the substrate node stress is more unbalanced, the adaptive scheme performs the node-opt strategy upon the arrival of a new VN request. Otherwise the link-opt strategy would be performed.

In the dynamic process of VN assignment, network conditions change over time due to the arrival and departure of virtual networks, which leads to inefficient resource utilization and load un-balancing. Thus VNE algorithm with configuration (VNA-II) is proposed to periodically check the stress ratio of substrate nodes and links. If the stress ratio of nodes or links exceeds the pre-configured threshold value, all the virtual networks embedded on these nodes and links will be reconfigured.

The two algorithms proposed in this paper have the potential to improve the substrate resources utilization efficiency. However, constrains of nodes and links haven't been considered. For the VNE algorithm with configuration (VNA-II), although the traffic load of substrate network will be more balanced through this process, reconfiguring the existing VN networks may interrupt the running network service, which is an issue of concern.

2.2.6 Summary

Due to four main computational challenges, VNE problem is known to be NP-hard. The proposed approaches that were previously introduced focused on restricting the problem space in one or more dimensions to address these challenges. This was done at the expense of limiting the practical applicability of the solutions. Some approaches split the

VN embedding problem into two separates node and link mapping phases. Non-join node and link embedding may result in a high number of VN requests being blocked and resources being under-used, which reduces the profit for InP, and other approaches relax resource availability constraints, which may ensue in non-respect of QoS requirements of VNs over time.

Moreover, all these studies are done under the normal operation of a substrate network. They assume the network to be continuously operational, which is a non-realistic assumption.

2.3 Resilient Virtual Network Embedding

2.3.1 Protection and Restoration for VNE

Many papers have extensively investigated and discussed the protection and restoration for general cases. As an example, the survivability for MPLS networks [51] and the protection of optical multicast traffic in WDM networks [37] [38] have been studied. The situation in network virtualization is different from general cases in some aspects, because of the challenges introduced by unique features of the network virtualization environment. First of all, the problems studied in MPLS are an offline version and, i.e., assume the traffic demand matrix has been available in advance. While in virtual network, because the VN request is online and we cannot predict its demand pattern, the guarantee for service resiliency is different and the problem of provisioning the backup networks is more complicated than that in MPLS. Second, we have to balance the trade-off between resource usage and survivability of the substrate network. Since the strong

survivability is achieved at the expense of a large amount of utilization of substrate resource, we have to optimize the resource usage when ensuring a specific level of survivability. Last but not least, in network virtualization environment, one of the basic requirements for survivability is that all the virtual links should be intact and connected when failure happens, which is different from that in other network environments. For example, in optical network, the goal is to only ensure that all the virtual nodes are still connected in the presence of failures, whether directly or indirectly.

In literature, very few papers provide protection and restoration mechanism for VN environment in their algorithms. All of them assume that the substrate network is a non-failure environment and can be operational all the time. This is not realistic. A variety of reasons, such as fibre cut, maintenance, mis-configuration and policy change, can cause disruption in substrate nodes and links. In recent year, particularly, the data transfer rate is considerably large and more and more critical business users rely on communication network. Thus an interruption of service, even for a short period of time, can cause a large amount of data loss and the violation of SLAs contracted between InPs and VNs owners, resulting in economic penalties and revenue loss for InPs. This is definitely unacceptable for business users.

To solve this challenge, researchers have introduced the survivability into VNE research and developed a “so called” survivable virtual network mapping (SVNM). While having some similarities to general cases, there are also two different strategies for the SVNM problem: protection and restoration. The protection mechanism can be divided into two phases which are done before failure happens. The first one is pre-plan or pre-configuration. When designing substrate network or embedding VN requests, we pre-

configure a set of candidate backup bandwidth for each substrate link. Second, we reserve some resources for a specific VN for recovery purposes. In the event of failures, we just need to switch primary flow to backup flow. The advantages of this mechanism are that the restoration time is shorter and 100% recovery can be guaranteed. But some bandwidth may be wasted when there is no failure. Differing from protection, restoration mechanism neither pre-computes nor reserves resources before failure happens. Instead, when a link or node fails, it will dynamically reroute the affected flows to the backup detours. Restoration overcomes the defect of protection but it spends more time in recovery and it cannot guarantee 100% recovery. The two mechanisms will be compared and summarized in Table 2.2.

	Protection	**Restoration**
Type	Proactive	Reactive
Characteristic	1) Pre-plan & pre-configure when designing substrate network or embedding VN 2) Reserve resource for recovery	1) No pre-configure and reservation for recovery 2) Dynamically direct to backup detours to recover the affected VN Embedding
Advantage	1) 100% recovery 2) Restore time is shorter	Efficient in capacity
Disadvantage	Inefficient in capacity	1) Restore time is longer 2) 100% recovery cannot be guaranteed

Table 2.2: Protection VS. Restoration

As previously mentioned, a great number of proposals have been focused on VN embedding problems, but most of them are done under the normal operation of substrate. A few works have been dedicated to provide resiliency for VN mapping. To address

resiliency concerns, one solution could be the allocation of a dedicated backup for each VN mapping. However, it is not efficient in terms of resource usage from the InPs point of view. Indeed, it is expected that backup resources can be shared for the protection of several VNs, keeping room for the mapping of more VN requests.

To the best of our knowledge, paper [35] is the first research document that attempts to add survivability to VNE algorithm and proposes a hybrid policy heuristic for solving SVNM. The heuristic consists of three steps. In the first phase, each substrate link is divided into two parts: one reserved for primary flow and the other reserved for backup flow. Additionally, a set of possible backup detours are pro-computed for each substrate link before any VN request arrives. The backup detours are selected by using a path selection algorithm, i.e., k-shortest path algorithm. The second phase embeds a virtual node using an algorithm with coordinated node and link mapping and embeds virtual link using a multi-commodity flow based algorithm. If a link failure happens, the final phase, a reactive backup detour optimization, is invoked to allocate resource from available candidate backup detours selected in the first phase to direct the affected traffic. If the available backup bandwidth is not enough for all affected flow, backup resource allocation will give high priority to the virtual link with higher penalty values.

In this algorithm, researchers leverage characteristics from both protection and restoration mechanisms. The pre-computation in the first phase is protection mechanism while the backup resource allocation is restoration. This has result in this algorithm being called hybrid policy heuristic. There are some areas in the algorithm that need to be improved. First, since this approach is based on a fast re-route strategy and utilizes a pre-reserved quota on each physical link, the efficiency is not high in terms of both

restore time and substrate bandwidth utilization. Additionally, the proposed backup strategy offers no mechanism to optimize the link division between primary and re-routed VN embedding paths, which may result in worse substrate resource usage.

In order to achieve high-efficient bandwidth utilization, T. Guo et al. [36] investigated the problem of shared backup network provision for VN embedding and presented two schemes: shared on-demand approach and shared pre-allocate approach. By sharing the bandwidth used by restoration flow from different VNs, the required backup bandwidth is greatly reduced. Whereas, some defects of the approaches in this paper need to be improved. For example, the first approach needs to be implemented during each VN embedding process and the second approach has to maintain enough backup bandwidth to protect maximum allowed primary flow on each link, which may not be effective at low request load.

The main drawback of these existing approaches is that resources are not optimally partitioned between primary and bypass paths. Consequently, substrate resources are not efficiently utilized, resulting in a great number of rejected VN requests and consequently in less InP revenue. In order to solve this drawback, a relatively new protection technique, which is called pre-configured protection cycle (*p*-Cycle), can be used to protect single element failures in virtual networks. *P*-Cycle technique is capable of achieving ring-like high speed protection with mesh-like high efficiency in use of spare capacity. The backup resource utilization of protection mechanism based on *p*-Cycle technique is more efficient and we can optimize the resource utilizations for offering full protection.

Two *p*-Cycle based approaches for the resilient virtual network embedding will be

introduced in the following Section. But before that, an introduction for *p*-Cycle techniques is necessary.

2.3.2 Introduction of *p*-Cycle Protection techniques

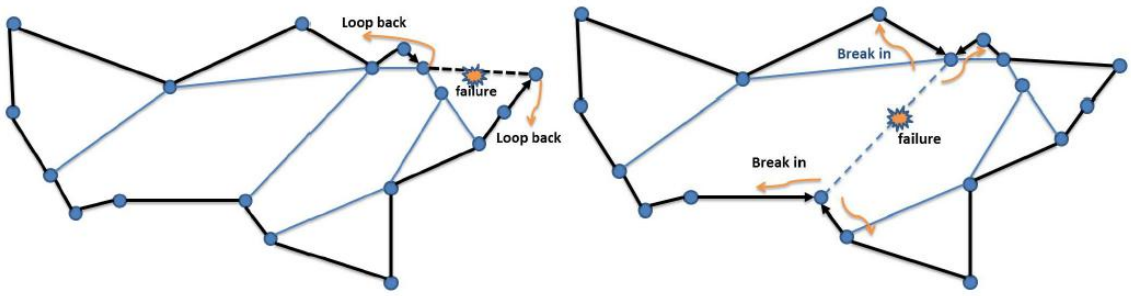
2.3.2.1 Overview

The pre-configured protection cycle, known as *p*-Cycle, is an efficient method for the design and operation of mesh restorable networks. The fundamental idea of *p*-Cycle is based on Bi-directional Line Switched Ring (BLSR) protection: proactively builds the protection paths by utilizing the concept of fully pre-cross-connected linear segments. When failure happens, only the two end nodes of the failed span need to switch to the pre-cross-connected protection path. As a result, the real time restoration speed is fast, which is a main property of *p*-Cycle. Additionally, *p*-Cycle not only supports protection for on-cycle span like ring protection but it also can protect straddling span, two end nodes of which are on cycles, with two alternative protection paths (see Figure 2.3). Consequently, the straddling links can have working capacity but no spare capacity, which is another important characteristic of *p*-Cycle. In conclusion, *p*-Cycle achieves ring-like high speed as well as mesh-like high efficiency in terms of spare capacity usage.

2.3.2.2 Link-protecting *p*-Cycles

Initially, *p*-Cycle mainly focuses on link protection. As mentioned above, *p*-Cycle can protect two types of links in a network, one is on-cycle link and the other is straddling link, and it combines the desirable properties of ring-like restoration and mesh-like

restoration.



(a) Protection "on-cycle" failure

(b) Protection a "straddling" failure

Figure 2.3: *p*-Cycle protection technique [11]

2.3.2.3 Path-protecting *p*-Cycles

Path-protecting *p*-Cycle is a path-based protection scheme. There are several kinds of path-oriented *p*-Cycle. Initially, most path protecting *p*-Cycles are based on shared backup path protection (SBPP) which is a popular pre-planned path restoration scheme. The basic concept of SBPP is to pre-configure a backup route for each working path, and when failure happens, a backup path is formed by sizing and cross connecting spare channels on all the intermediate nodes of the backup route. In SBPP, working paths have to satisfy the disjoint restriction, i.e., backup routes are disjoint from the working paths and only mutual-disjoint working paths are allowed to share backup paths, so the SBPP has good capacity efficiency. Figure 2.4 illustrates a set of mutually disjoint working paths and their corresponding protection paths. The maximum sharing in this example happens on the link AD where three separate working paths share a single unit of spare capacity along the backup route. SBPP-based *p*-Cycle inherits all the desirable characteristics and makes some improvements. Unlike SBPP, SBPP-based *p*-Cycle

doesn't need cross connection on the immediate nodes and only the two end nodes of the cycle perform failure detection and cross connection on demand in real time.

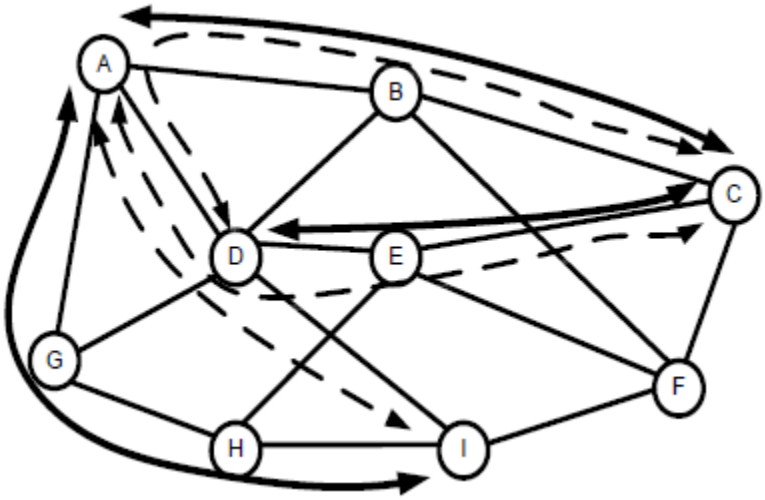


Figure 2.4: A set of three mutually disjoint working routes and their corresponding shared backup paths [46]

To eliminate the time spent on cross connection, researchers provide Failure-Independent Path-Protecting *p*-Cycles (FIPP). In FIPP, on one hand, all the protection paths are pre-connected, under which condition end nodes only need to switch the affected traffic to protection path upon failure. On the other hand, the working path in FIPP can have path segment in common with its backup path. As a consequence, FIPP *p*-Cycle has potentially higher capacity efficiency than SBPP. Figure 2.5 illustrates a set of five mutually disjoint working paths that are protected by one FIPP *p*-Cycle. As it is evident, no failure can affect two compatible demands which are protected by the same FIPP *p*-Cycle.

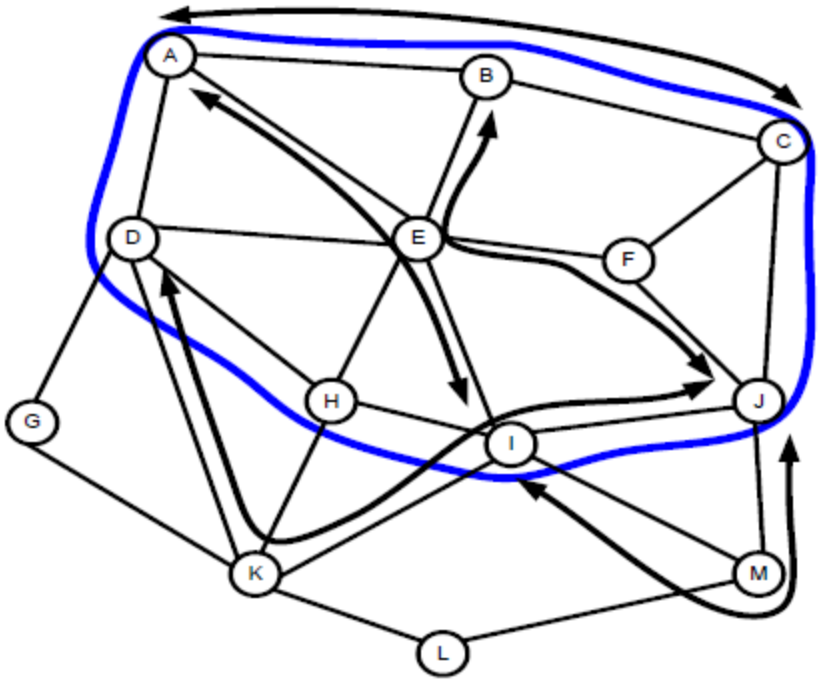


Figure 2.5: A FIPP *p*-Cycle is protecting a set of five mutually disjoint working paths [46]

2.3.2.4 Flow-protecting (Segment) *p*-Cycle

By their respective natures, Link *p*-Cycle and path *p*-Cycle only protect spans that are part of themselves or that directly straddle the respective *p*-Cycle, and partially protect against node failure. Motivated by these factors, G. Shen et al. [37] introduced the concept of "flow *p*-Cycle". *"A flow can be defined as any single contiguous segment of a working end-to-end path."* According to this definition, the entire path, a set of links along the path and single link can be called flow and can be protected by flow *p*-Cycle. Because segment *p*-Cycle can access more opportunities for span-capacity-share than link *p*-Cycle, it has higher capacity efficiency than link *p*-Cycle. For example, in Figure 2.6(a), there is a link protection *p*-Cycle scheme with a *p*-Cycle (A-B-C-D-E-A) for protecting demand B-A-E-D. Given that the required bandwidth by demand on each link

is unit capacity, the used capacity for primary flow is three units and the used spare capacity for protection is five units. Therefore, capacity redundancy is 5 ÷ 3 (3 working units) = 167%. Figure 2.6(b) shows that segment *p*-Cycle B-C-D-E-B, which consists of three protection segments, is able to protect demand B-D. This is divided into two working segments, s_1 and s_2. When the failure of the on-cycle link associated with segment E-D happens, as shown in Figure 2.6(c), the end nodes of the working segment E-D are switched to the protection segments E-B and B-C-D. A failure of link A-B belonging to the straddling working segment B-A-E, as shown in Figure 2.6(d), can be recovered through protection segment B-E (or the two protection segments, B-E and B-C-D-E). The segment *p*-Cycle needs four units of spare capacity, and the capacity redundancy thus equals 4 ÷ 3 = 133%. As a result, the segment *p*-Cycle solution is more capacity efficient than the link *p*-Cycle protection solution used in this example.

In terms of recovery speed, the restoration speed of pure link *p*-Cycle is faster than that of pure path protection *p*-Cycle, and flow *p*-Cycle would lie between them.

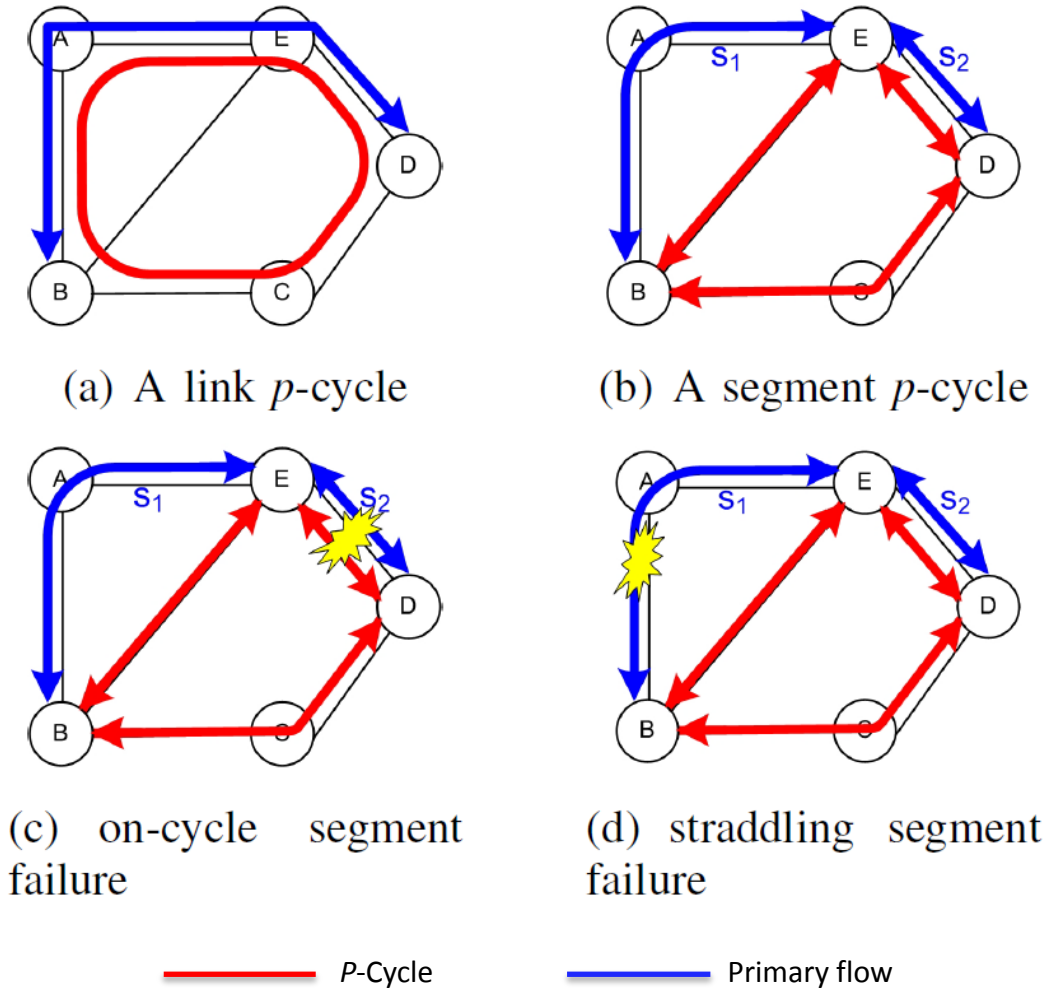


Figure 2.6: Link *p*-Cycles vs. segment *p*-Cycles [48]

2.3.2.5 Node-encircling *p*-Cycle (NEPC)

A node-encircling *p*-Cycle is a *p*-Cycle for a particular node if it contains all of the node's neighbouring nodes (i.e., those nodes connected directly to it with a span) but not the node itself. In this way, the NEPC is able to substitute the failure node for all possible flows. Generally, there are two kinds of NEPC： simple and non-simple. A simple node-encircling cycle is one that does not cross any node or span more than once, as shown in Figure 2.7(a). However, in some cases, such a cycle is difficult to be

recognized or even doesn't exist. Nonetheless, as long as the pre-failure network graph is at least two-connected, we can always draw at least one logically encircling non-simple cycle that crosses at least one node or span more than once, as shown in Figure 2.7(b)-(c). Non-simple cycles can be divided in several sub-types, primarily distinguished by the maximum number of traversals through a node or a span (in one direction). In the following Table 2.3, different types of non-simple cycles are represented. In this thesis, only 2-1 non-simple NEPC is used.

Subtypes of non-simple cycles	Characteristic
2-1:	**Nodes can be traversed twice, links only once per direction**
2-2:	**Nodes can be traversed twice, links can be traversed twice in each direction**
3-1:	**Nodes can be traversed three times, links only once per direction**

Table 2.3: Subtypes of non-simple cycles [49]

In some cases, in order to include all adjacent nodes, a NEPC has to include some nodes that are not adjacent to the protected node, which causes low-efficient resource utilization.

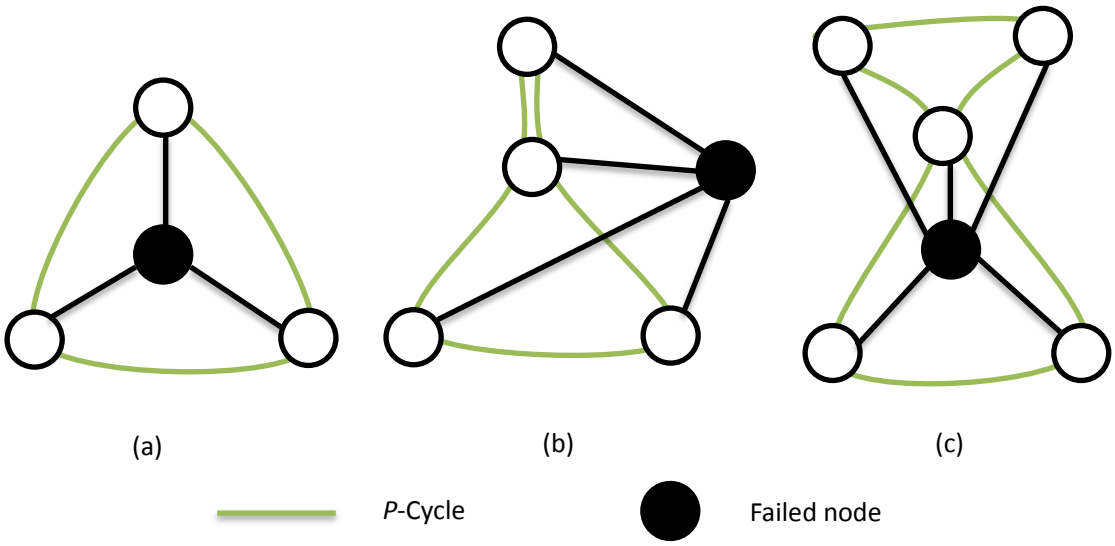


Figure 2.7: (a) A simple node-encircling *p*-Cycle, and (b)-(c) two non-simple node-encircling *p*-Cycles [40]

2.3.2.6 *p*-Cycle Application in General Cases

In general cases, the existing protection approaches may be classified into five major categories: (i) tree-based protection; (ii) ring-based protection; (iii) path-based protection; (iv) segment-based protection; (v) *p*-Cycle-based protection. They have their own advantages and disadvantages, but in [36], researchers showed that the link-protecting *p*-Cycle-based approach outperforms the other existing protection approaches for link failure recovery.

In [39], authors proposed a node and link failure protection approach in the context of optical networks. They firstly extend the node protection concept of the *p*-Cycle approach to achieve more efficient resource utilization. Then propose a novel algorithm that integrates their concept for the node protection. In this paper, the main focus in this proposal is on the physical layer. There is no information about the upper logical layers.

Accordingly, a physical failing node or link can have a severe effect on the logical layer which is the case in the network virtualization environment. Indeed, a single physical node failure can result in multiple logical links failure as virtual links belonging to the same VN can be mapped into physical paths that span over the failing node.

In [40], a combined node and span protection approach with a single or shared set of node-encircling *p*-Cycles (NPECs) is proposed. In other words, each node is protected by a single dedicated or a shared NPECs. The main concern as aforementioned in [39] is related to fact that the node protection proposal omit to consider any logical linking among protected paths. Accordingly, the proposal is not well adapted to the context network virtualization under consideration in the current paper.

In [41], authors proposed a node failure protection scheme. Again, the main focus in this node failure protection proposal is on providing a mechanism that can handle physical node failure omitting any consideration related to virtual networks in the upper layers. The main contribution of this proposal is an extension of ordinary span-protecting *p*-Cycles to include more candidates in order to increase the protection efficiency in terms of resources usage. The main idea consists on that in addition to that each *p*-Cycle has its own spans for which it provides intended span failure protection, every *p*-Cycles will also happen to intercept a number of working flows upstream and downstream of nodes on *p*-Cycles.

3. Link Protection Approach: Resilient Virtual Network Embedding (RVNE)

3.1 Overview

To overcome the drawbacks in terms of real time restoration speed and bandwidth utilization that exists in previous survivable virtual network embedding approaches, this thesis investigates a popular and interesting technology -- *p*-Cycle. This fast speed and highly efficient capacity utilization protection mechanism is applied into the VN embedding environment to protect against single link failure. *P*-Cycle is famous for its high capacity efficiency and fast restoration speed which comes from its inherent properties that *p*-Cycle can protect on-cycle span and straddling span simultaneously plus no spare capacity is needed in straddling span.

In this Chapter, the RVNE approach is presented. It involves: (a) calculating in the first stage a cost-efficient join VN node and link mapping using Column Generation technique to handle complexity issues, which is proposed by [11], and (b) proposing a *p*-Cycle-based VN protection approach that minimizes the backup resources while providing a full link protection scheme. My contribution focuses on the second phase.

In the first stage of the RNVE approach, the use of the Column Generation technique means that the VN embedding problem is decomposed into a master problem (which includes constraints related to the availability of substrate resources) and a pricing problem (which includes the constraints related to the embedding of VN resources). We

call this approach Join Node and Link Embedding using Column Generation (JNLE-CG). The master and pricing problem formulations will be presented in the following Section. The objective of the second stage is to propose a protection scheme for the working path of an embedding scheme in order to guarantee 100% protection in case of a single link failure, while minimizing the cost of used resources. One thing should be noted. Only single substrate link failure is taken into consideration since the multiple concurrent failures happen rarely in the real world. In this stage, first of all, the Hongbo Liu's algorithm [42] is proposed to be used to enumerate the set of eligible candidate *p*-Cycle-based on the substrate network topology. Then an Integer Linear Programming (ILP) model is proposed to select the optimal set *p*-Cycle that can satisfy the objective of 100% protection and minimizes the cost of used resource from the candidate *p*-Cycles.

3.2 Small-batch Provisioning

In a realistic virtualization scenario, VN requests may not usually arrive one after another [43] in regular interval of time. Thus, a realistic VN embedding scenario could be based on a periodical approach, where VN requests are queued and then processed in small-batches in order to optimize InP profit's over time. To do so, I discretize the provisioning time into a set of consecutive short periods. Hence, VN demand can be expressed more precisely, let *T* be the set of planning periods of time and *R(0)* the initial set of VN requests. The set of VN requests *R(t)* indexed by $t \geq 1$ is defined as:

$$R(t) = R(t-1) + R_{NEW}(t) - R_{DROP}(t) \qquad (3.1)$$

Where $R(t-1)$ is the set of accepted VN requests at the ending of period $t-1$. $R_{NEW}(t)$ is the set of new incoming VN requests and $R_{DROP}(t)$ is the set of ending VN

requests at the start of period *t*. Where NEW and DROP are randomly selected between for example, 10% and 40%, giving us a range of cases from slowly fluctuating (10%) to fast changing (40%) of VN demand.

3.3 VN Embedding

To overcome the complexity issue of VN embedding problem, a mathematical model which makes use of Column Generation (CG) technique [11] is used and the embedding problem is reformulated in terms of Independent Embedding Configurations (IECs), where each of which defines the embedding solution of one VN request. An IEC is defined as a set of substrate nodes and links used to handle resource requirements (bandwidth and CPU) of the granted VN request, see Figure 3.1. We represent an IEC $c \in C$ by a vector $(a_n^c)_{n \in N}$ such that a_n^c = 1 if the IEC *c* serves VN request *n* and 0 otherwise. We denote by $COST_c$ the cost of an IEC *c* that corresponds to the costs of used bandwidth and CPU for the embedding of the VN request granted by the IEC *c*. It is defined according to Equation (3.2) as follows.

$$COST_c = \sum_{l \in L_s} b^c(l) \times c_l + \sum_{u \in W_s} p^c(u) \times c_u \tag{3.2}$$

Where $b^c(l)$ and $p^c(u)$ are the used substrate bandwidth of link l and CPU resources of node u by IEC *c* respectively. As mentioned previously, each VN request *n* offers revenue for the embedding of its sets of virtual nodes and links. Accordingly, an EIC *c* serving VN request *n* generates an InP profit's defined as:

$$REV_c = P_n - COST_c \tag{3.3}$$

The use of the Column Generation technique means that the VN embedding problem is decomposed into a master problem (which includes constraints related to the availability of substrate resources) and a pricing problem (which includes the constraints related to the embedding of VN resources). We call this approach Join Node and Link Embedding using CG (JNLE-CG). I present the master and pricing problem formulations as following.

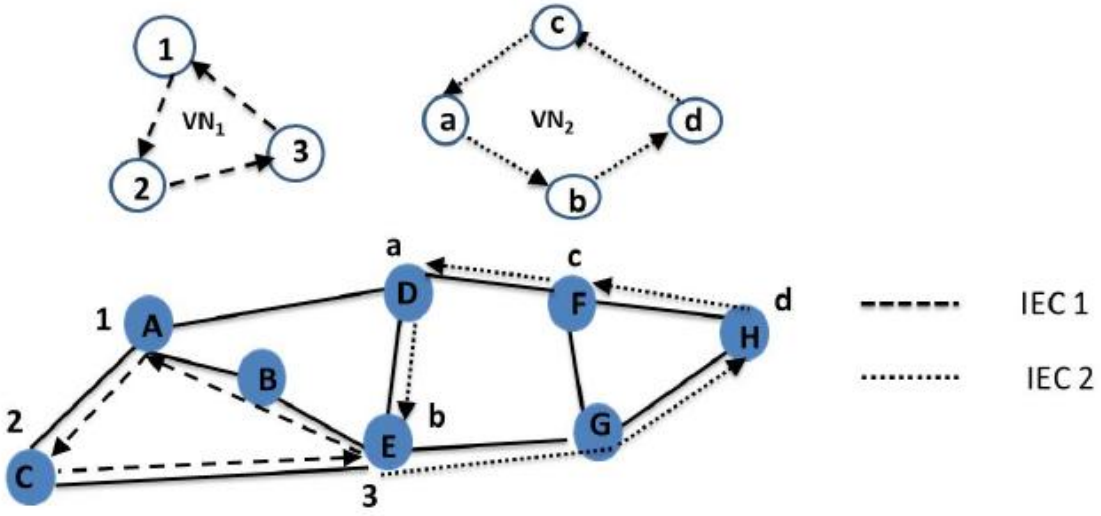


Figure 3.1 Independent Embedding Configuration [11]

1) *Master problem*: It corresponds to the choice of a maximum of *N* IECs in order to maximize the InP profit's. To so do, I use the following Integer Linear Program *ILP(M)* based on the decision binary variable λ_c that takes the value 1 if IEC *c* is used in embedding solution and 0 otherwise.

   a) Objective Function:

$$max \sum_{c \in C} REV_c \ \lambda_c \tag{3.4}$$

   b) Constraints:

$$\sum_{c \in C} \lambda_c \; b^c(l) \leq b_l; \quad l \in L_s \quad (\alpha_l) \tag{3.5}$$

$$\sum_{c \in C} \lambda_c \, p^c(u) \leq p_u; \quad u \in W_s \quad (\beta_u) \tag{3.6}$$

$$\sum_{c \in C} \lambda_c \, a_c^n \; \leq 1; \quad n \in N \quad (\gamma_n) \tag{3.7}$$

$$\sum_{c \in C} \lambda_c \leq N; \tag{3.8}$$

$$\lambda_c \; \in \{0,1\} \tag{3.9}$$

Equations (3.5) and (3.6) indicate that substrate link and node loads are no more than the residual bandwidth b_l and residual CPU capacity p_u respectively. Equation (3.7) indicates that a maximum of one IEC can be selected for the embedding of each accepted VN request $n \in N$. Equation (3.8) guarantees that a maximum of *N* IECs can be selected for the embedding of all VN requests, as optimally all VN requests are accepted where each VN request is granted by a distinguish IEC. Equation (3.9) expresses the integrality of the master variable λ_c.

2) *Pricing problem:* It corresponds to the problem of generating an additional column (IEC) for the constraint matrix of the current master problem. It is defined as follows. Let $\alpha_l, \beta_u, \gamma_n$ and η be the dual variables associated with constraints (3.5), (3.6), (3.7), and (3.8) respectively. Then, the reduced cost of variable λ_c can be written:

$$\overline{RVE} = RVE_c - \sum_{l \in L_s} \alpha_l b^c \, (l) - \sum_{u \in W_s} \beta_u p^c \, (u) - \sum_{n \in N} \gamma_n \, a_c^n - \eta \tag{3.10}$$

In order to linearize the expression of the reduced cost, and also to express the constraints of the pricing problem, the following decision variables are defined.

- $x_\pi^e = 1$ if virtual link $e \in E_n$is assigned to path π and 0 otherwise.

- $x_a^u = 1$ if virtual node $a \in A_n$ is assigned to node $u \in W_s$ and 0 otherwise.
- $z_n = 1$ if VN $n \in N$ is granted by IEC $c \in C$ and 0 otherwise.

We next derive the following relations between variables of the pricing problem and the coefficients of the master problem. For each $c \in C$ and $n \in N$, we have:

$$a_c^n = z_n = \frac{1}{|E_n|} \sum_{e \in E_n} \sum_{(u,v) \in W^2} \sum_{\pi \in \Pi_{uv}^e} x_\pi^e \tag{3.11}$$

For each link $l \in L_s$, we have:

$$b^c(l) = \sum_{n \in N} \sum_{e \in E_n} \sum_{(u,v) \in W^2} \sum_{\pi \in \Pi_{uv}^e} b^{c(e)} \delta_\pi^l x_\pi^e \tag{3.12}$$

For each node $\in W_s$, we have:

$$p^c(u) = \sum_{n \in N} \sum_{a \in A_n} P_{c(a)} x_a^u \tag{3.13}$$

Thus, the pricing objective function and constraints can be expressed as follows.

(a) *Objective*:

$$\begin{aligned} \overline{RVE} = & \sum_{n \in N} P_n z_n - \sum_{n \in N} \sum_{e \in E_n} \sum_{(u,v) \in W^2} \sum_{\pi \in \Pi_{uv}^e} COST(\pi_{uv}^e) \times x_\pi^e \\ & - \sum_{l \in L_s} \alpha_l \sum_{n \in N} \sum_{e \in E_n} \sum_{(u,v) \in W^2} \sum_{\pi \in \Pi_{uv}^e} b^{v(e)} \delta_\pi^l x_\pi^e - \sum_{u \in W_s} \beta_u \sum_{n \in N} \sum_{a \in A_n} P_{c(a)} x_a^u \\ & - \sum_{n \in N} \gamma_n \times z_n - \eta \end{aligned} \tag{3.14}$$

Where $COST(\pi_{uv}^e) = c_u p_u + c_v p_v + \sum_{l \in \pi} c_l b^{c(e)}$

(b) *Constraints*:

$$z_n \leq \sum_{(u,v) \in w_s^2} x_s^u \, x_d^v \,; \quad (s,d) = e \in E_n. \tag{3.15}$$

$$\sum_{u \in W_s} x_a^u \leq z_n; \quad \forall a \in A_n. \tag{3.16}$$

$$x_a^u = 0; \ \forall u \in W_s, \ u \notin t_{c(a)}, \ \forall a \in A_n. \tag{3.17}$$

$$x_s^u x_d^v \ \leq \ \sum_{\pi \in \Pi_{uv}^e} x_\pi^e \, ; \ (u, w) \in W_s^2, (s, d) = e \in E_n. \tag{3.18}$$

$$\sum_{(u,v) \in W_s^2} \sum_{\pi \in \Pi_{uv}^e} x_\pi^e \ \leq \ z_n; \quad e \in E_n. \tag{3.19}$$

$$L(\pi) - 1 \leq d^{c(e)}; \quad \pi \in \Pi_{uv}^e, \ (u, v) \in W_s^2, e \in E_n. \tag{3.20}$$

$$\sum_{n \in N} z_n \leq 1. \tag{3.21}$$

Equations (3.15), (3.16), and (3.17) express QoS requirements for the embedding of virtual network nodes. Equations (3.18) and (3.19) indicate that only one embedding path is assigned for each virtual link. Equation (3.20) expresses the QoS requirement for the embedding of virtual network links in terms of maximum number of end-to-end switching nodes. Equation (3.21) guarantees that an IEC can embed a maximum of one VN request. By reducing the complexity of the pricing problem in this way, the Column Generation process is speed up.

3) *Solving JNLE-CG model*: To solve the JNLE-CG model developed in the previous Section, we denote by *LP(M)* the continuous relaxation of the master problem *ILP(M)* obtained by exchanging the integrality constraint (3.9) by $\lambda_c \in [0, 1]$ for any $c \in C$, then the following algorithm is applied.

2. Solve Master problem *LP* (*M*) using CPLEX and go to Step 3.

3. Solve Pricing problem using CPLEX and go to Step 4.

4. If a column with a positive reduced cost has been found, add this column to

the current master problem, and re-iterate with Steps 2 and 3. Otherwise, *LP* (*M*) is optimally solved, then goes to Step 5.

5. To calculate an integer solution, we re-establish integrality constraint on variable λ_c and we call CPLEX solver.

3.4 *p*-Cycle Embedding Path Protection (*p*-Cycle-EPP) Approach

3.4.1 Eligible *p*-Cycles Enumeration

In this approach, Hongbo's algorithm is used to generate *p*-Cycles on a substrate network topology. If the vertices and edges of a graph G' are subset of those of another graph *G*, *G'* is a subgraph of *G*. A path can be presented by using alternating vertices and edges, beginning and ending with vertex. For example, $v_1 e_1 v_2 e_2$ ... $e_{n-1} v_n$, where the first vertex v_1 is called head; the last one v_n is called tail. The length of a path is the number of edges. A simple path is a path such that all the vertices are distinct. If the head and tail are identical, a simple path is called cycle, otherwise, it is called open path. A cycle consists of edges. However, besides cycle, the combination of edges could be open path or sub graph which is a union of cycles and open paths. After talking about the definition of some concepts, Hongbo et al. concluded that "if all the open path of length $k-1$, is obtained, we can generate all open path of length *k* and all k-cycle" [42]. Based on this factor, the main process this algorithm is to generate all k-cycles and *k* length open paths from $k-1$ length open paths and this process will be iterated as the value of *k*

increases progressively. The loop will be finished until there is no open path generated. Below are the details of the generation process:

Phase 1: To avoid the generation of duplicate cycles from different vertices of the same cycle, in the first step of the algorithm, the graph is represented as an adjacent list or an adjacent matrix. Each vertex is assigned a unique integer designator as their order, which is ascending from head to tail.

Phase 2: Second, all the vertices are put into a FIFO queue which is used to store the open paths. For convenience, a register is used to record the length of the path and it is 0 at the beginning.

Phase 3: Then the main loop of the algorithm starts to iterate. An open path is fetched from the queue and the adjacent situation of the tail of this open path is verified. If there is an edge which connects the head and tail of the open path, a cycle is generated and then outputted. In this case, the register has to be checked to see if it equals 0 to avoid a self-loop cycle. If a cycle cannot be enumerated, it must be checked whether there is an adjacent edge of the tail whose end does not occur in the open path and whether the order of its end is greater than that of its head. If this situation occurs, this edge and the k length open path will construct a new k+1 length open path and this new open path will be put into the queue.

Phase 4: After fetching and verifying all k length of open paths, the register will be set to k+1 and a new loop started. The main loop operates cyclically until the queue is empty, which is also the end of the algorithm.

The flowchart in Figure 3.2 shows the procedure of Hongbo's algorithm.

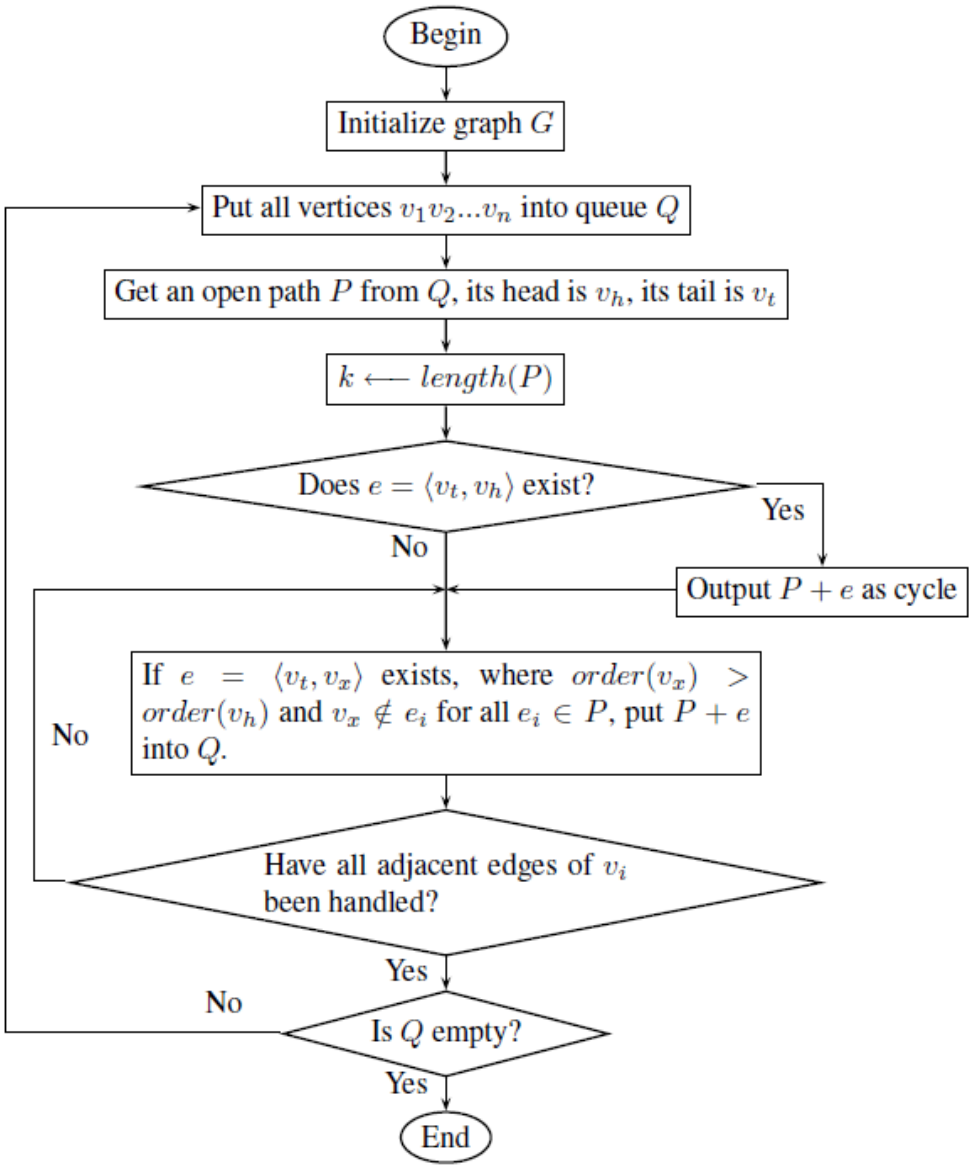


Figure 3.2: Flowchart of Hongbo's algorithm [42]

In the implementation, an upper bound was set for the length of generated *p*-Cycles, when using Hongbo's algorithm to generate *p*-Cycles. By doing so, two benefits result: 1) Max hop is one of important QoS requirements in network virtualization environment. By restricting the length of *p*-Cycle, a limitation for max hops of backup

path on *p*-Cycle can be set up in an indirect way. 2) Because the max length of *p*-Cycle is restricted, the number of candidate *p*-Cycle is limited and a scalability problem is avoided.

3.4.2 *p*-Cycle-EPP Model

This Section of this thesis presents the proposed path protecting *p*-Cycle-based VN Protection approach called *p*-Cycle-EPP. In this model, the objective is to minimize the cost used for *p*-Cycle protection. This objective can be achieved through making use of two features of *p*-Cycle: 1) straddling link has two alternative protection paths on the cycle and it is twice as efficient in terms of bandwidth utilization; 2) FIPP *p*-Cycle allows multiple link-disjoint embedding paths to share the same *p*-Cycle. As a result, despite the bandwidth constraint and domain constraint, another constraint is added, which enforces a link-disjointness on the embedding paths that share any *p*-Cycle structure, into the ILP model.

a) Parameters:

- π_N and P are the set of embedding paths and candidates protection *p*-Cycles respectively.
- b_π represents the amount of bandwidth used by embedding path $\pi \in \pi_N$.
- δ_l^π= 1, if path π uses link $l \in L_s$ and 0 otherwise.
- α_p^l= 1, if *p*-Cycle p uses link l and 0 otherwise.
- σ_π^p represents the number of backup paths that a *p*-Cycle p offers. Thus σ_π^p= 2 if end-nodes of path π are in cycle p, σ_π^p = 1 if end-nodes of path π are in cycle p and at least one span uses cycle p, and σ_π^p= 0 otherwise.

- z_π^p = 1, if embedding path π is protected by *p*-Cycle *p* and 0 otherwise.

b) Decision variables:

- y^p= represents how much bandwidth is used on *p*-Cycle *p* for protection of embedding paths.
- y_π^p= represents how much bandwidth of *p*-Cycle *p* is used to protect embedding path.

c) *Objective function:*

$$Min \sum_{p \in P} y^p \sum_{l \in L_s} c_l \, \alpha_p^l \tag{3.22}$$

d) Constraints:

$$y_\pi^p \leq y^p ; \quad \forall p \in P \quad \forall \pi \in \pi_n \tag{3.23}$$

$$b_\pi \leq \sum_{p \in P} \sigma_\pi^p y_\pi^p \quad ; \quad \forall \pi \in \pi_n \tag{3.24}$$

$$\sum_{\pi \in \pi_n} \delta_\pi^l \, z_\pi^p \leq 1; \quad \forall p \in P \quad \forall l \in L_s \tag{3.25}$$

$$y_\pi^p \leq M * z_\pi^p \tag{3.26}$$

$$y_\pi^p \quad y^p \in Z_+ \tag{3.27}$$

Constraint (3.22) minimizes the total cost of all *p*-Cycles used for VNs protection. Constraint (3.23) ensures that when a *p*-Cycle is shared by multiple links, the link that requires most bandwidth can be restored through *p*-Cycle. Constraint (3.24) guarantees that all embedding paths must be protected by *p*-Cycle. Constraint (3.25) indicates that two embedding paths sharing the same *p*-Cycle protection must be link-disjoints. Constraint (3.26) expresses the relationship between decisions variables z_π^p and y_π^p. When embedding path π is protected by *p*-Cycle *p*, at least one bandwidth of *p*-Cycle *p*

should be used to protect embedding path π. Constraint (3.27) is the variables domain constraint.

To solve this model, first of all, I use Hongbo's algorithm introduced in 3.4.1 to enumerate candidates *p*-Cycles *P* for each embedding path π. Secondly, I call CPLEX to solve this mathematical problem.

3.5 Benchmarks

To assess the efficiency of the RVNE approach, two benchmarks are selected and implemented. The first benchmark is proposed in paper [35] and [36], and it is named SLP-2PE which is short for Substrate Link Protection Combined with Two-Phases Embedding (node embedding and link embedding) approach. The second one is Stress-Function based Embedding Combined Disjoint Path Protection (SFE-DPP) approach, proposed in [50]. This approach applies a greedy VN node mapping using stress function used in [8] then two disjoint paths are calculated for primary and backup of each virtual link.

Benchmark 1: SLP-2PE

SLP-2PE is based on a proactive policy that pre-allocates both primary and backup bandwidth then a two-phase embedding approach is applied for VNs mapping. One character of SLP-2PE is that this approach combines protection procedure and embedding procedure together and solves these two bandwidth allocation problems using one ILP model. As a result, this pre-allocation approach only needs to be implemented once before any VN request arrives and thus significantly reduce the computing load during VN embedding. What's more, SLP-2PE allows restoration flows

from different VNs to share the same backup bandwidth. In this way, the required backup bandwidth is able to be greatly reduced and more substrate resources can be left for embedding future VN requests.

In this benchmark, candidate *p*-Cycle set is also generated by Hongbo's algorithm and I use an ILP model in [36] to find the optimal result. The ILP model is as following.

Objective:

$$Maximize \sum_{s \in E^s} \alpha_s B(s) \tag{3.28}$$

Where $\alpha_s$is the percentage of the bandwidth allocated on a substrate link *s* for mapping primary flows. $B(s)$ is the bandwidth of substrate link *s*.

Subject to:

1) Restoration flow constraint

$$\sum_{r \in R(f)} y_r(f) = \alpha_f B(f), \ \forall f \in E^s \tag{3.29}$$

Where $y_r(f)$ is the restoration flow on the substrate path r in case of the substrate link f failure. $R(f)$ denotes the set of candidate bypass paths for a substrate link *f*.

2) Restoration bandwidth constraint

$$\sum_{r \in R(f)} I_s(r) y_r(f) \leq (1 - \alpha_s) B(s), \ \forall s \in E^s, \forall f \in E^s \tag{3.30}$$

Where $I_s(r)$ is 1 if path *r* uses link *s*; 0, otherwise.

The objective function (3.28) tries to maximize the total protected bandwidth of the substrate network for mapping primary flows. Constraint (3.29) represents that the bandwidth allocated on each substrate link for mapping primary flows should be fully

protected via the restoration flows over its bypass paths. Constraint (3.30) ensures that the bandwidth demand of the restoration flows can be satisfied by the pre-allocated backup bandwidth upon any substrate link fails.

Benchmark 2: SFE-DPP

The idea of benchmark 2 is relatively simple. After a VN request is embedded, a disjoint k-shortest path will be calculated for each embedding path as its backup path until all embedding paths are protected or there is not enough bandwidth for backup paths. This process is based on greedy algorithm. In order to maximize bandwidth sharing, multiple embedding paths are allowed to completely or partially share a k-shortest path for protection purpose.

3.6 Summary

To guarantee a VN provisioning with resiliency mechanism, the Resilient VN Embedding approach (RVNE) is proposed in this Section. This approach firstly performs a cost-efficient join VN node and link mapping using the CG technique, then provides a full VN protection scheme based on *p*-Cycle concept. To evaluate the performance of this approach, two existing survivable VNE schemes are used as benchmarks.

4. Node Protection Approach: VN Mapping with Combined Node and Logical Links Failures Protection (VNM-CNLP)

4.1 Overview

Node failure means intermediate node failure, and source/sink nodes failures are omitted. Because source flows and the sink flows cannot be restored by any protection approach. The objective of the node failure recovery is to restore the transiting flows that pass through a failed node. The most efficient protection method against physical node failure is to establish a set of backup paths for primary paths in a proactive manner on substrate network. This enables fast recovery from the failure and transport of data is almost undisturbed. But the shortcoming is resources and capacity usage are large, so researchers [45], [46] have adapted some alternative technologies, such as, ring, mesh and *p*-Cycle mechanisms to overcome this defect.

However, most of existing work focus on general cases, such as optical and MPLS networks. In fact, in optical networks the aim of protection is only to guarantee that physical nodes are still linked together in the presence of node failures (weak resiliency). While, in network virtualization environment, we need to ensure that all virtual links are intact in the presence of failures. In other words, for a VN request, we have to guarantee survivability of all the VN links simultaneously (strong resiliency).

Furthermore, in network virtualization environment, a physical node failure not only affects the physical layer, but also propagates to the virtual network layer. If the virtualization is recursive, i.e., the first level VNs can act as InPs to the second level of VNs [45], dropping a failing physical node will affect any number of upper VN layers. Accordingly, node failure protection mechanisms can be provided at different layers. In case of a physical layer protection mechanism, a node failure is detected directly at the physical layer. The affected VN mapping paths can be repaired by activating the backup paths or by rerouting affected VN traffic on the fly. This is in general transparent to the VN layers. On the other hand, in case of reactive failure protection at the VN layer, a physical failure propagates to the logical layer where it is detected. Routing paths in the VN topology are re-computed to avoid the failing virtual links, which is transparent to the physical layer.

It is well known that a single physical node failure is logically equivalent to multiple physical link failures in a network [12]. Moreover, in virtualization environment, a single physical node failure can also result in a multiple VN links failure (logical links failure) as virtual links belonging to the same VN can be mapped to physical paths that span over the failing physical node. As a result, the focus will be on the physical layer protection mechanism while taking constraints from the logical layer (virtual networks) into consideration in order to guarantee a full VN protection against a single physical node failure and a multiple logical links failure. In other word, the aim of this Section is to provide a VN mapping backup against physical node failure that takes into account constrains from the logical layer related to the VN topology and QoS requirements.

The node protection approach presented in this Chapter is named VN Mapping with

Combined Node and Logical Links Failures Protection, which involves two phases: VN embedding and VN protection. In first phase, the Join Node and Link Embedding approach is still used, except a new constraint is added. In the second phase, three *p*-Cycle-based protection techniques are proposed, which are my contribution.

4.2 VN Embedding

In the first stage of VNM-CNLP, I use Join Node and Link Embedding using CG (JNLE-CG) approach, developed in paper [11], to calculate a cost-efficient VN mapping while minimizing the effects of a single node failure in VN. In this stage, I still use small-batch VN provisioning and reformulate the VN embedding problem in terms of IECs. Because the first part of JNLE-CG approach, mater problem formulation, is the same with RVNE approach in Chapter 3, I present this approach starting from pricing problem formulation, as following.

1) *Pricing problem*: It corresponds to the problem of generating an additional column (IEC) for the constraint matrix of the current master problem. The same pricing model proposed in work [11] will be used, except the following constraint (4.1) is added in order to ensure that embedding paths for each VN request must be link-disjoint. Accordingly, we minimize the effect of a physical node failure into the VN layer and guarantee more resiliencies in the VN backup phase.

$$\sum_{e \in E_n} \sum_{(u,v) \in W_s^2} \sum_{\pi \in \Pi_{uv}^e} \delta_\pi^l x_\pi^e \leq 1;\ l \in L_s \tag{4.1}$$

Where δ_π^l= 1 if embedding path π uses substrate link l and 0 otherwise. x_π^e is the main decision variable of pricing problem and it is equal to 1 if embedding path π is

used for the mapping of virtual link *e* and 0 otherwise.

2) *Solving JNLE-CG model*: To solve the JNLE-CG model developed in the previous Section, I denote by *LP (M)* the continuous relaxation of the master problem *ILP(M)* obtained by exchanging the integrality constraint (3.9) by $\lambda_c \in [0,1]$ for any $c \in C$. Then the *LP(M)* is solved using any linear programming solver (such as CPLEX) until the optimal solution is reached. To check the effectiveness of this solution, we solve the pricing problem in order to identify variables with positive reduced cost. If such a variable exists, it is added to the master problem and solved again until the master problem has been solved to optimality. More details can be found in Chapter 3.

4.3 VN Survivability against Physical Node and Multiple Logical Links Failures

VN survivability can be seen as a generalization of the QoS guarantees that InP commits to VN users. In this approach, VN survivability means VN backup plan to recover from a single physical node failure, and as compared to physical link failure, its impact is more severe. Accordingly, the emphasis will be on the physical layer to provide a protection mechanism against a single physical node failure while taking into account constraints from the logical layer (virtual network layer) in order to guarantee a full VN protection against a multiple logical links failure. To do so, we proceed as follows: (a) VN mapping is done in a such way that virtual links belonging to the same VN are mapped to link-disjoint physical paths (see constraint (4.1)), and (b) input of VN

protection phase will be the mapping topology (set of selected embedding paths) for each VN request instead of a set of independent VN embedding paths. Doing so, VN backup will be restricted in such a way that embedding paths used for embedding of virtual links belonging to the same VN should be node-disjoint in order to guarantee a node failure independent path protection scheme. Additionally, for the purpose of minimizing the backup resources utilization, I maximize the number of straddling link structure in *p*-Cycle protection and the number of embedding paths that share the same backup path on cycle.

Unlike link protection, there is not such a kind of *p*-Cycle that specially designed for node protection except NEPC. However, since NEPC has to intercept any working path transiting the encircled node, both upstream and downstream of the protected node, NEPC-based approaches have limitations in improving the efficiency of resource utilization and finding eligible *p*-Cycles in some cases. Consequently, path protecting *p*-Cycle and segment (flow) protecting *p*-Cycle become the alternative options. I adopt FIPP *p*-Cycle, which is a type of path protecting *p*-Cycle, as candidate *p*-Cycle in the first VN protection approach and a combination of path and segment protecting *p*-Cycles in the second VN protection approach.

4.3.1 *p*-Cycle-based Virtual Span Protection Technique (*p*-Cycle-VSP)

The *p*-Cycle-VSP technique relies on the transfer of a node failure protection into a span failure protection, with Figure 4.1 illustrating this. To support this technique the following two phases are applied.

a) *Phase 1*: For each substrate node used in a VN embedding path, it is assumed that there exists a *virtual span* that crosses this node from one of adjacent nodes to another. Both of these two adjacent nodes should be transited by the same working flow. See Figure 4.1, the orange node represents the failed node and the green line represents the virtual span. After this transformation, virtual spans, which can be protected by path protecting *p*-Cycle, replace the nodes and become the protection targets. In other words, the VN embedding paths are protected against node failure by finding the equivalent protecting set of virtual spans. By this means, a higher resource utilization can be achieved while simultaneously guaranteeing 100% node protection. Next, finding candidate *p*-Cycles for protecting these "virtual span" is easy by using Hongbo's algorithm [42].

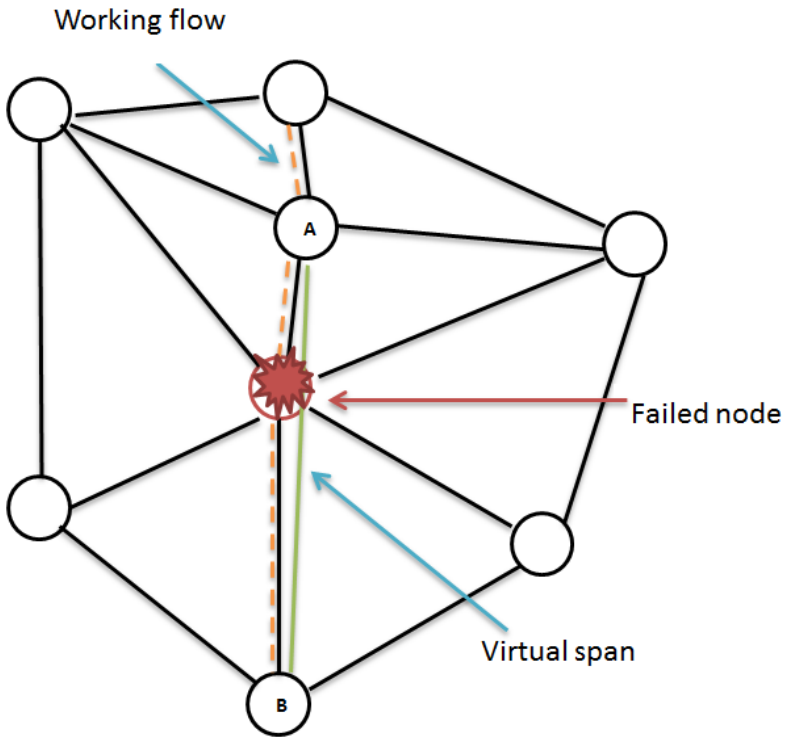


Figure 4.1: *p*-Cycle-based virtual span protection

b) *Phase 2:* The *p*-Cycle-VSP protection technique is formulated as an ILP and is defined as follows. This ILP model sets constraints in terms of link bandwidth, node-disjointness as well as max hops which is not taken into consideration in link protection.

Parameters:

- S and P are the set of virtual spans and candidates protection *p*-Cycles respectively.
- T_N denotes the set of all embedding topologies generated for N virtual networks. We note that an embedding topology of a VN n corresponds to the set of assigned embedding paths.
- S_n denotes the set of virtual spans generated for the embedding topology calculated for the VN n in embedding phase.
- b_s denotes the bandwidth requirement of span s.
- δ_s^u= 1, if physical node u is an intermediate node of span s and 0 otherwise.
- α_p^l= 1, if *p*-Cycle p uses link l and 0 otherwise.
- α_s^p represents the number of backup paths that a *p*-Cycle p offers to span s. Thus, α_s^p= 2 if end-nodes of span s are in *p*-Cycle p, α_s^p = 1 if end-nodes of span s are in *p*-Cycle p and at least one link uses cycle p, and α_s^p= 0 otherwise.

Decision Variables:

- y^p represents how much bandwidth is used on *p*-Cycle p for protection of VN embedding paths.
- y_s^p denotes how much bandwidth of *p*-Cycle p is used to protect virtual span s.
- z_s^p = 1 if span is protected by *p*-Cycle p and 0 otherwise.

Objective function:

$$\min \sum_{p \in P} y^p \sum_{l \in L_s} c_l \alpha_p^l \tag{4.2}$$

Subject to:

$$y_s^p \leq y^p; \quad \forall s \in S \quad \forall p \in P \tag{4.3}$$

$$y_s^p \leq M\, z_s^p ; \quad \forall s \in S \quad \forall p \in P \tag{4.4}$$

$$b_s \leq \sum_{p \in P} \sigma_s^p\, y_s^p ; \qquad \forall s \in S \tag{4.5}$$

$$\sum_{s \in S_n} z_s^p \delta_s^u \leq 1; \quad \forall p \in P \quad \forall u \in W_s \quad \forall n \in N \tag{4.6}$$

$$L(p,s) \leq H^{c(s)}; \quad c(s) \in Q_1; \ \forall s \in S; \forall p \in P \tag{4.7}$$

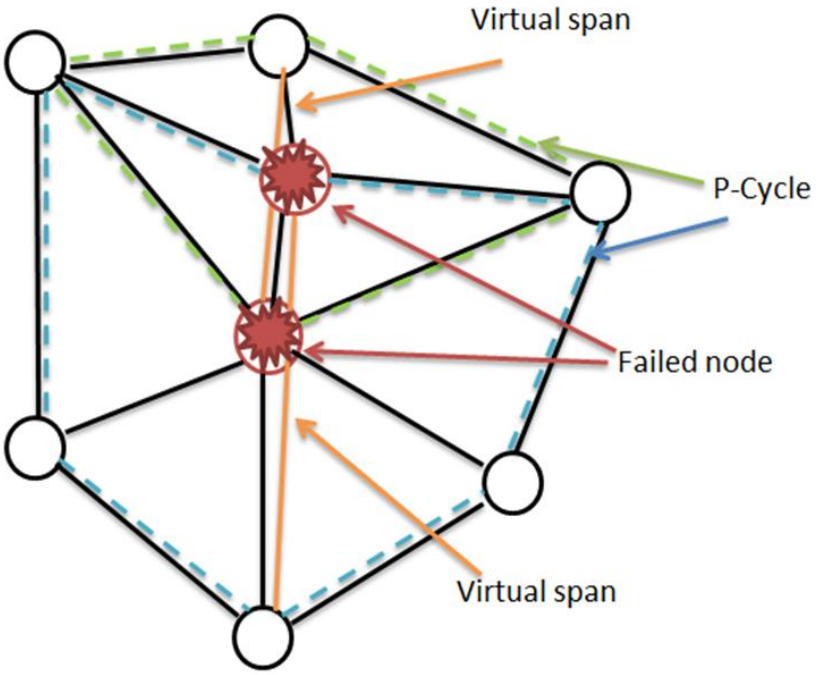
$$y^p , y_s^p \in Z_+, z_s^p \in \{0,1\}; \ p \in P, s \in S_n \tag{4.8}$$

Where $L(p,s)$ is expressed as the sum of number of hops of backup and embedding paths minus the number of hops of used span. *M* is a large number at least equal to the largest QoS class bandwidth requirement. The Objective function (4.2) minimizes the cost of *p*-Cycles used for VNs protection. Equation (4.3) ensures that when a *p*-Cycle is shared by multiple virtual spans, the span that requires most bandwidth is able to be restored through *p*-Cycle. Equation (4.4) expresses the relationship between decisions variables z_s^p and y_s^p. When span *s* is protected by *p*-Cycle *p*, at least one unit of bandwidth of *p* should be used to protect *s*. Equation (4.5) ensures that enough bandwidth should be provided by a *p*-Cycle that protects a virtual span. Equation (4.6) confirms spans relative to the same VN embedding topology sharing any protection *p*-Cycle should be node-disjoint. Constraint (4.7) expresses QoS requirement in terms of number of switching nodes. Constraint (4.8) is the variables domain constraint.

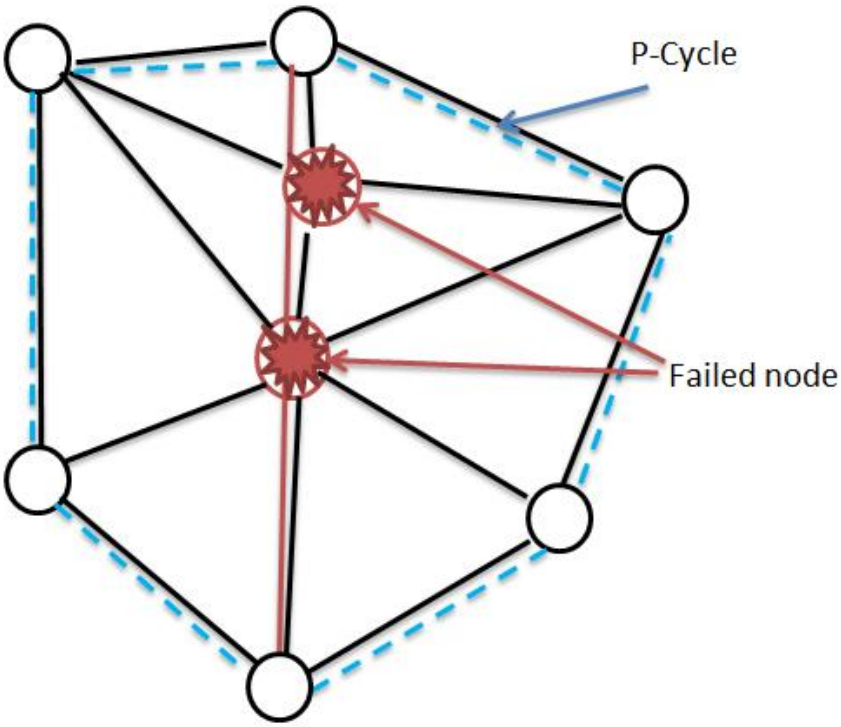
An improved version of *p*-Cycle-VSP technique that combines multiple virtual spans together into a new virtual span is proposed. Then, on the condition that the sharing of *p*-Cycle protection among multiple embedding paths increases and a better resource utilization performance can be obtained, a *p*-Cycle will be found to protect this new virtual span instead of previously multiple virtual spans. This improved approach is called *p*-Cycle-CVSP, for combined virtual span based *p*-Cycle protection.

However, here lies a problem. How virtual spans are to be combined. There are two possible ways to combine virtual span: one is to combine virtual spans that cross the same node; the other way is to combine two spans that share one same link in an iteration way. In this research, the second approach is taken.

By combining some virtual spans, nodes are able to share *p*-Cycle protection as much as possible, and backup path sharing efficiency increases. An example of this is illustrated in Figures 4.2(a) and 4.2(b). The orange line represents an embedding path and the blue and green dash lines denote two *p*-Cycles. We can assume that the traffic on this embedding path is 2 units and when failure happens on a node, the *p*-Cycle will split the traffic evenly into two directions. If the previous algorithm is used, i.e., without combination of virtual spans, 4 units of capacity are needed to recover the primary flow traversing node A. In addition, 6 units of capacity are needed to restore node B failure. In total, 10 units are required for protecting two nodes. If these two virtual spans are combined to a new one, then the black *p*-Cycle will be used to protect this new virtual span. Because only single node failure is considered, 8 units of capacity are needed when node A fails or 6 units of capacity are needed when node B fails. As a result, in average, 7 units of capacity are needed to protect these two nodes.



(a) Two virtual spans are protected by two p-Cycles



(b) Combination of two virtual spans

Figure 4.2: An example of improved version of p-Cycle-VSP

In fact, to obtain the optimal performance, the coexistence of both previous spans and new spans is allowed, i.e., the candidate *p*-Cycle set includes those *p*-Cycles that are used to protect original virtual spans and those *p*-Cycles that are used to protect virtual spans generated from combination. In order to avoid overlap of protection, the same ILP model is used as in the *p*-Cycle-VSP approach, except constraint (4.9) is modified as follows:

$$max_{\forall s \in S_u}(b_s) \leq \sum_{p \in P} \sigma_s^p \, y_s^p \,, \qquad \forall u \in W_s \tag{4.9}$$

Where S_u denotes the span set in which virtual spans cross node u.

Constraint (4.9) guarantees that enough bandwidth is provided by protecting *p*-Cycles for the span with the largest bandwidth requirement in S_u set denoted by $max_{\forall s \in S_u}(b_s)$.

Theoretically, the *p*-Cycle-CVSP approach has a better performance than the *p*-Cycle-VSP approach, in terms of resource utilization. However, two disadvantages of *p*-Cycle-CVSP approach are the complexity and scalability problems produced by the remarkable increase of candidate *p*-Cycles and virtual spans. Also, more computing time is required to find an optimal solution.

4.3.2 *p*-Cycle-based Span-and-path Protection Technique (*p*-Cycle-SPP)

This approach consists on enlarging the candidate protecting *p*-Cycles set compared to those used in *p*-Cycle-VSP technique. To do so, the two following phases are applied.

a) *Phase 1:* Compared to *p*-Cycle-CVSP and *p*-Cycle-VSP techniques, the range of candidate *p*-Cycles is extended. The new range includes *p*-Cycles that have one of the

following relationships with its respective protected embedding path against a given node failure.

1) Embedding path is partially on-span with used *p*-Cycle (Figure 4.3(a)).

2) Embedding path is partially on-span and on straddling with used *p*-Cycle Figure 4.3(b)).

3) Embedding path is partially on straddling with *p*-Cycle (Figure 4.3(c)).

For these three relationships, I will use Onguetou's [41] concept which consists on the following:

"The protecting p-Cycle must intercept the affected flow in at least two nodes, that is, one node upstream and one node downstream of the node failure."

b) *Phase 2*: I formulate the *p*-Cycle-SPP protection technique as an ILP defined as follows.

Parameters:

- P denotes the set of candidate protection *p*-Cycles.
- b_π denotes the bandwidth requirement of embedding path π.
- a_π^p represents the number of backup paths that a *p*-Cycle p offers. Thus, $a_\pi^p = 2$ if end-nodes of path π are in *p*-Cycle p, $a_\pi^p = 1$ if end-nodes of path π are in *p*-Cycle p and at least one link of path uses *p*-Cycle p, and $a_\pi^p = 0$ otherwise.

Decision Variables:

- y^prepresents how much bandwidth is used on *p*-Cycle p for protection of VN embedding paths.
- $y_{u\pi}^p$ denotes the amount of bandwidth used on *p*-Cycle p to protect embedding path π, when node u fails.

- $z_{u\pi}^{p}$= 1 if embedding path π is protected by *p*-Cycle *p* against node failure *u* and 0 otherwise.

Objective function:

$$min \sum_{p \in P} y^{p} \sum_{l \in L_s} c_l \alpha_l^{p} \tag{4.10}$$

Subject to:

$$\sum\nolimits_{n \in N} \sum_{\pi \in T_n} y_{u\,\pi}^{p} \leq\ y^{p}; \quad \forall p \in P\ \ \forall u \in W_s \tag{4.11}$$

$$y_{u\,\pi}^{p} \leq\ M\ z_{u\,\pi}^{p}\ ; \quad \forall u \in W_s \quad \forall p \in P\ \ \forall \pi \in T_n\ \forall n \in N \tag{4.12}$$

$$b_\pi \leq \sum_{p \in P} y_{u\,\pi}^{p}\ a_\pi^{p}\ ; \quad \forall \pi \in T_n \quad \forall n \in N \quad \forall u \in W_s \tag{4.13}$$

$$\sum_{\pi \in T_n} z_{u\,\pi}^{p} \leq\ 1; \quad \forall p \in P \quad \forall u \in W_s \ \ \forall n \in N \tag{4.14}$$

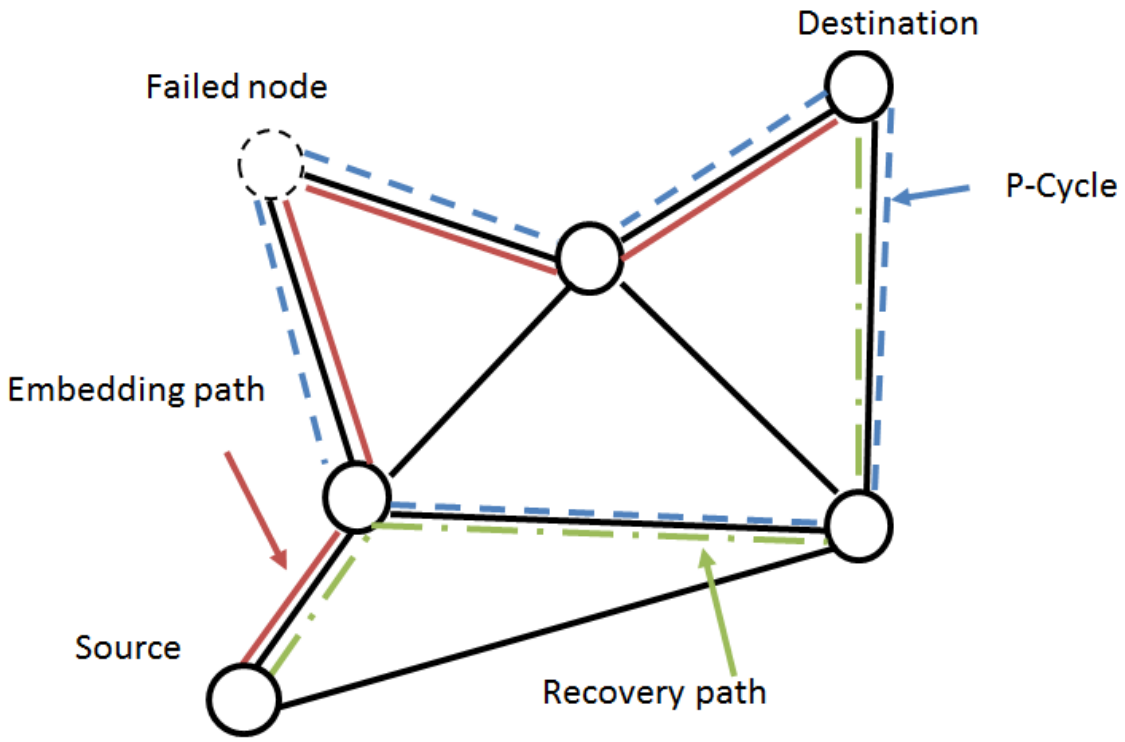
$$L(p,u,\pi) \leq H^{c(\pi)}\ ; \quad c(\pi) \in\ Q_1; \quad \forall \pi \in T_n\ \ \forall p \in P \ \ \forall u \in W_s \tag{4.15}$$

$$y^{p}\ , y_s^{p} \in\ Z_+; \ \ z_{u\,\pi}^{p} \in \{0,1\}; \quad p \in P, \forall u \in W_s\ , \forall \pi \in T_n \tag{4.16}$$

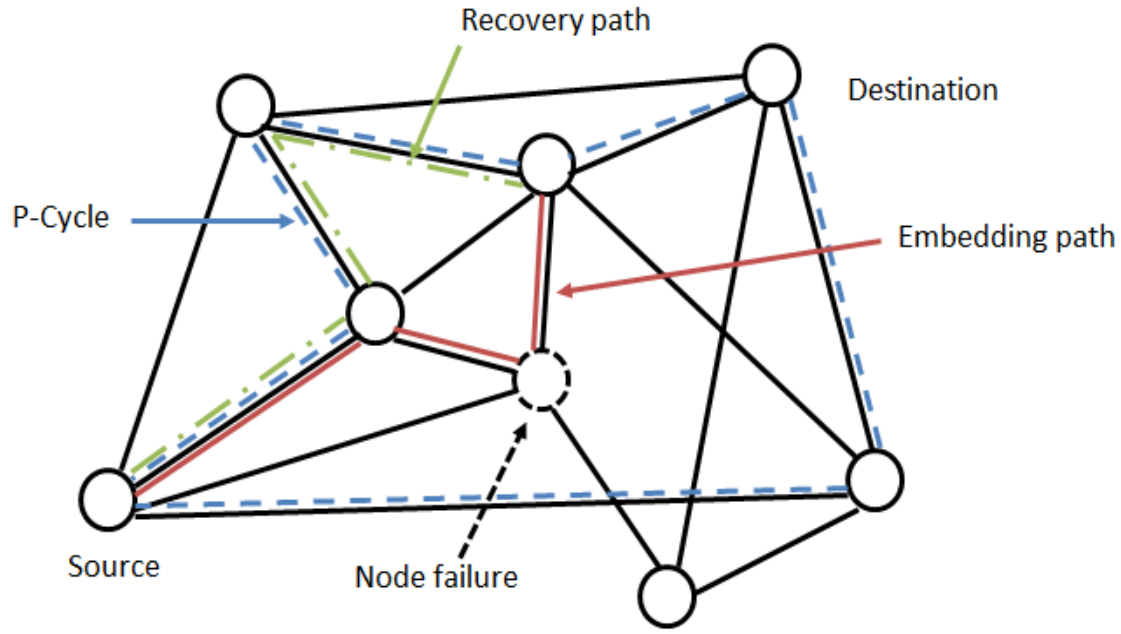
Where $L(p,u,\pi)$ is expressed as the sum of number of hops of backup and embedding paths minus the number of hops of used *p*-Cycle.

The objective function (4.10) minimizes the cost of *p*-Cycles used for VNs protection. Constraint (4.11) ensures that in one virtual network, when a *p*-Cycle is shared for the protection by multiple nodes, the node that requires most bandwidth can be restored through *p*-Cycle. Constraint (4.12) express linking among variables, $z_{u\,\pi}^{p}$ and $y_{u\,\pi}^{p}$. Constraint (4.13) ensures that enough bandwidth should be provided by *p*-Cycles that protect embedding paths. Constraint (4.14) is used to protect VN embedding path π

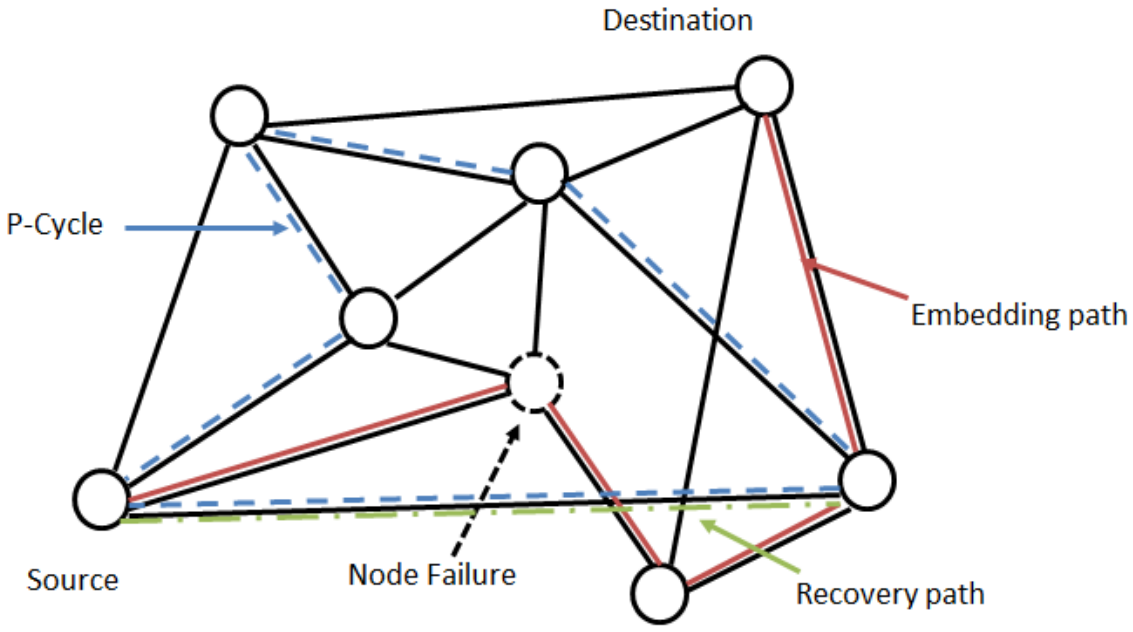
against a multiple logical links failure. Thus, backup paths of the same VN embedding topology sharing any protection *p*-Cycle should be node-disjoint. Constraint (4.15) expresses QoS requirement in terms of number of switching nodes. Constraint (4.16) is the variables domain constraint.



(a) Partially on-span



(b) Partially on-span and straddling



(c) Partially straddling

Figure 4.3: *p*-Cycle-based span-and-path protection candidates

4.4 Benchmark

Since the main contribution in this work is the proposition of VN protection techniques, accordingly I propose a benchmark VNM-NPEC approach that uses:

(a) A VN mapping based on JNLE-CG approach;

(b) A VN protection technique based on the node encircling p-Cycle (NPEC) that consists on finding a set of particular p-Cycles for each substrate failing node.

The model of benchmark is very similar with that of p-Cycle-SPP approach, except the constraint (4.13) should be modified to:

$$b_\pi \leq \sum_{p \in P} y^p_{u\,\pi} * 2 \ , \quad \forall \pi \in T_n \quad \forall n \in N \quad \forall u \in W_s \tag{4.17}$$

Since NPEC is defined for the protection of transiting flows through one specific node by containing all the immediate neighbour-nodes of the protected node but not the protected node itself, each node is protected by its corresponding NEPC in a straddling way.

In the benchmark approach, candidate NEPC set includes simple NEPCs and non-simple NEPCs. Generally speaking, as long as the pre-failure network graph is at least two connected, a simple NEPC or at least one logically encircling non-simple cycle can always be formed for a node. However, in case of there is no NEPC, I take a path that goes through a maximum number of neighbour-nodes is used for protection.

4.5 Summary

In this Chapter, the focus is on node protection using *p*-Cycle, path protecting *p*-Cycle and segment *p*-Cycle, mainly, in network virtualization environment. Three node protection approaches are proposed, *p*-Cycle-VSP, *p*-Cycle-CVSP and *p*-Cycle-SPP, to minimize total spare capacity cost while simultaneously maximizing the node failure restorability. The main idea of *p*-Cycle-VSP and *p*-Cycle-CVSP approaches is to transfer node failure protection to span failure protection, while the key direction of *p*-Cycle-SPP is to use Onguetou's concept to extend the range of candidate *p*-Cycle. All these approaches improve the capacity efficiency by enhancing the sharing of *p*-Cycle protection and the amount of straddling link.

5. Performance Evaluation

5.1 Objective

In this Chapter, the performances of the RVNE approach and VNM-CNLP approach are evaluated by conducting simulations. From the presented simulation results, we can see that the proposed survivable virtual network embedding schemes have better performances than benchmarks in terms of the evaluation metrics.

5.2 Evaluation Environment and Strategy

In the implementation, the computing platform is an Intel Quad-Core CPU (3.0 GHz) PC with 4G RAM. Visual Studio 2010 and C++ are used to simulate the topology of the substrate network and VN requests. To find an optimal solution for the ILP model, CPLEX 12.4 which is the last version of IBM CPLEX is called. Since small-batch VN provisioning which discretizes the provisioning time into a set of consecutive short periods is used, the performance of each method is tested in 10 different periods.

Additionally, to avoid scalability problems, at the most, 10 different *p*-Cycles are allowed to be used to protect one single node in VNM-CNLP algorithm implementation. For those nodes which have more than 10 eligible *p*-Cycles protection, the shortest 10 *p*-Cycles will be selected as preferable candidate *p*-Cycles for protection.

5.3 Simulation Setup

To assess the efficiency of the proposed periodical approach, experimental assessments were carried out on an EU metro backbone network that spans the 20 largest metropolitan areas in Europe, connected by 44 bidirectional links [7]. Since VNs are still not well-known in the industry, synthetic VN topologies are generated using the generator inspired from GT-ITM tool proposed in [47]. QoS requirements of new VN requests are randomly determined by a uniform distribution among $|Q_1| = 5$ QoS classes for VN nodes and among $|Q_2| = 5$ QoS classes for VN links. For example, in link QoS class, bandwidth attribute and max hoops attribute are defined, and different values are set for those attributes in each class. Profit and unit costs are expressed in terms of $X, which represent the price of 1 Mb of bandwidth or 1 unit of CPU.

5.4 Performance Evaluation of RVNE Approach

5.4.1 Performance Evaluation Metrics

To evaluate the performances of the proposed survivable VNE approaches, the following metrics are measured.

1) InP Profits: measured as the revenue collected from accepted VN requests, minus the cost of used resources.
2) Acceptance ratio of VN requests: measured as the ratio of accepted VN requests to their overall number.

3) Unit backup cost: measured as the ratio of backup cost to bandwidth used by primary flow.
4) Backup bandwidth efficiency: measured as the ratio of required backup bandwidth to the one used by primary flow.

5.4.2 Evaluation Scenario

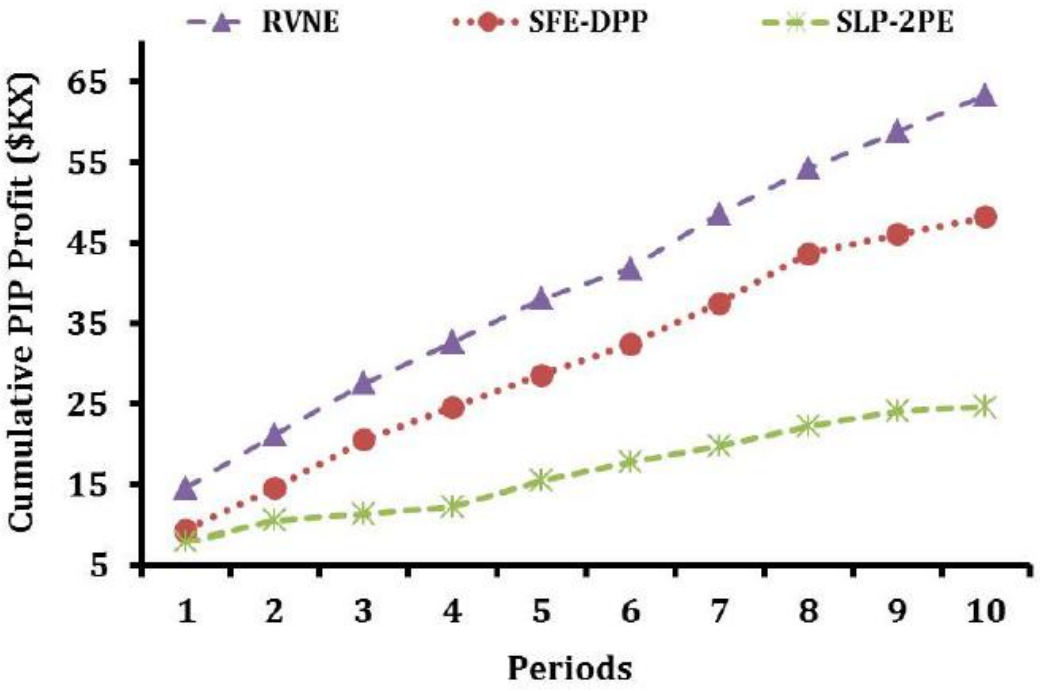
To evaluate the performance of the RVNE approach, the following scenarios will be evaluated:

1) RVNE: uses JNLE-CG approach for VN mapping and *p*-Cycle-EPP approach for VN protection.
2) Benchmark - 1: uses SLP-2PE approach for VN mapping and protection.
3) Benchmark - 2: uses SFE-DPP approach for VN mapping and protection.

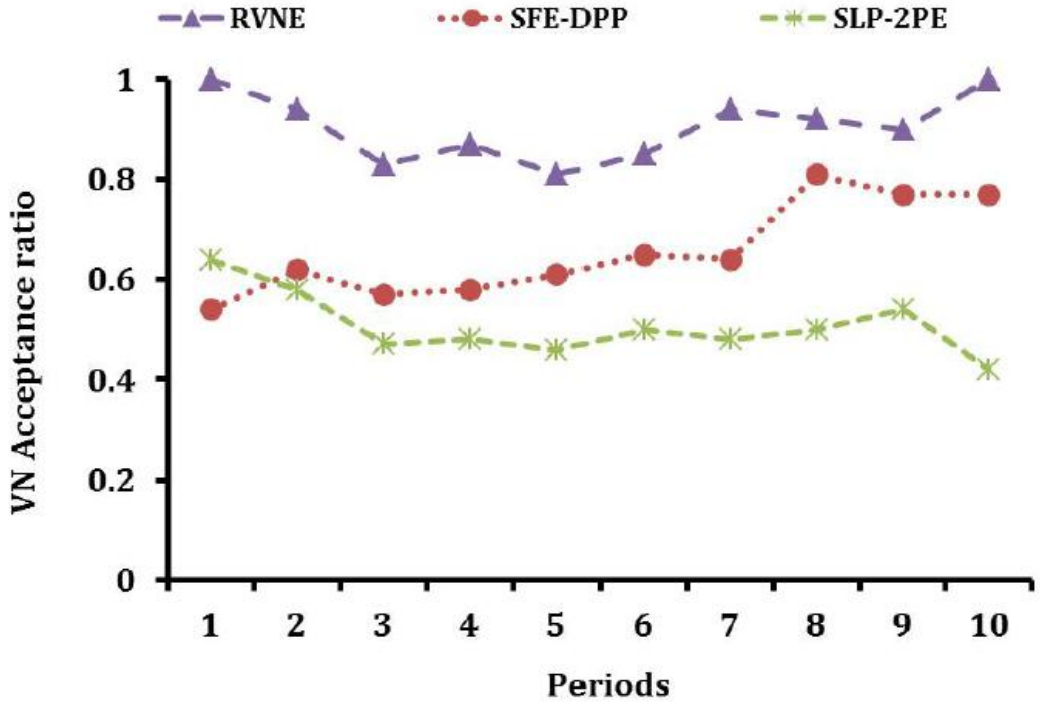
5.4.3 Evaluation Results

The emphasis in this Section is on quantifying the advantage of combining *p*-Cycle protection mechanism against single link failure and CG embedding approach that performs join node and link mapping in terms of InP profit, VN protection cost, and resources usage. To demonstrate the strength of RVNE approach, the benchmarks are implemented and a comparison between RVNE and benchmarks is made. Experiments on large mix of VN requests show a clear advantage of RVNE approach over benchmarks. Indeed, PIP profit's is increased up to 20%, VN embedding paths can be fully protected and bandwidth resources are used more efficiently.

1) Join node and link mapping using CG increases profit and acceptance ratio: In Figures 5.1(a) and 5.1(b), we observe that as expected RVNE approach outperforms benchmarks in terms of InP profit's and VN acceptance ratio over a large mix of QoS requirements of VN requests. I believe that the superiority of our RVNE approach is because of the join node and link mapping of VN resources and the CG modeling technique. This combination is able to provide an optimal or near-optimal embedding solution resulting in high InP profits and an efficient resources utilization, which increases VNs acceptance ratio. As expected, results showed in these curves confirm my expectation that two-phases embedding approach used in SLP-2PE may result in a lack of InP profits to gain. A myopic node embedding that considers only CPU capacities will ensue in scares of bandwidth in substrate links resulting in high blocking rate of VN requests. SFE-DPP is based on node and link stress function which may help on increasing the VN acceptance ratio sometimes closer to that provided by RVNE embedding approach, however, it is unable to provide more InP profits as it is based on greedy embedding approach.

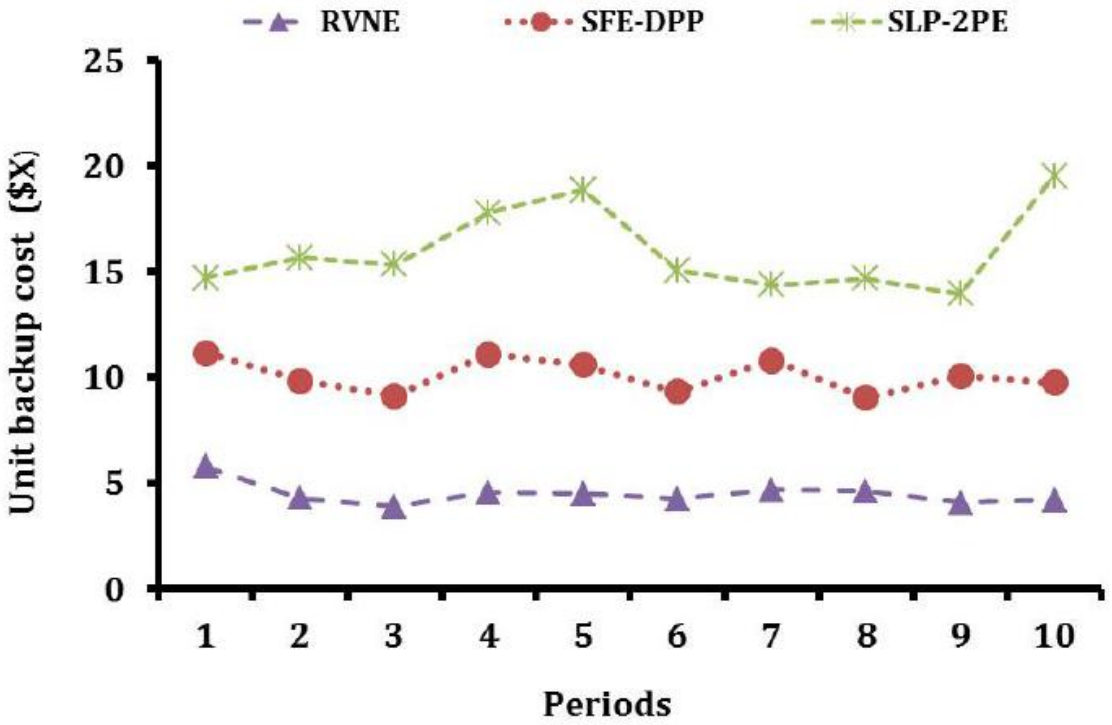


(a) Cumulative InP Profits

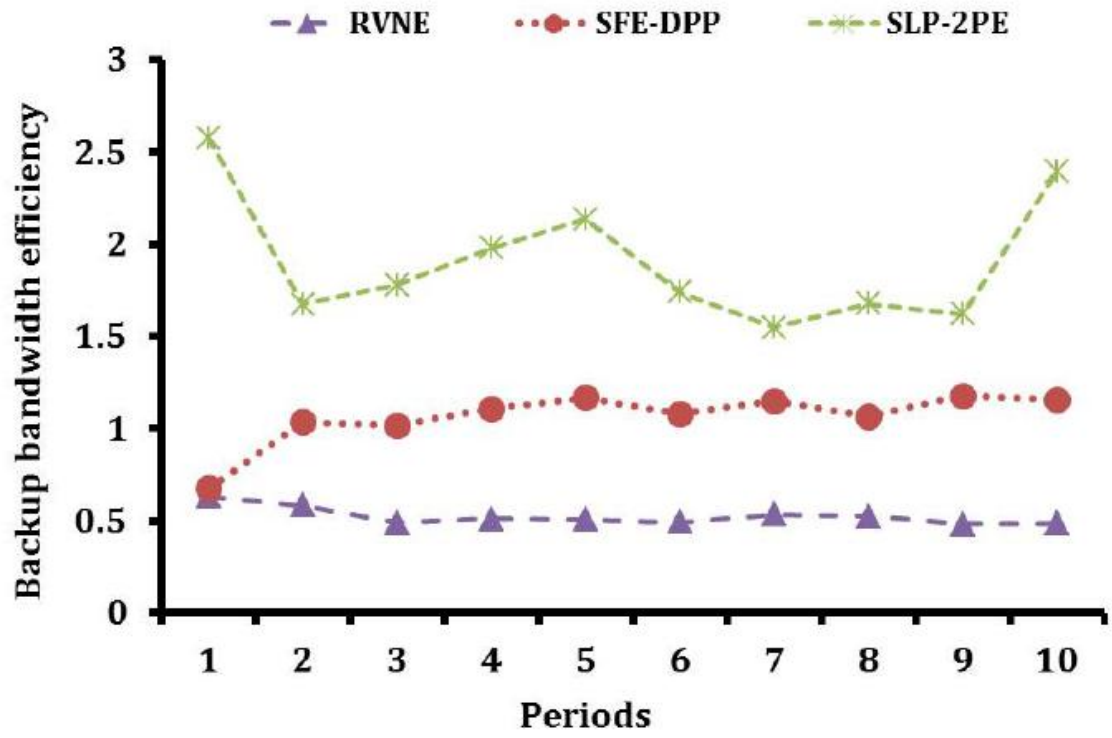


(b) VN acceptance ratio

Figure 5.1: Efficiency of embedding approach (JNLE-CG)



(a) Backup cost for 1 Mb of primary flow



(b) Backup bandwidth for 1 Mb of primary flow

Figure 5.2: Efficiency of protection approach (*p*-Cycle-EPP)

2) p-Cycle guarantees full protection with minimum cost: Figure 5.2(a) plots the backup cost per unit of protected bandwidth vs. the allocation time periods. This Figure shows that RVNE model provides the lowest backup cost per unit of protected bandwidth over a large mix of VN requests. For instance in period 2, SFE-DPP uses 9.02 unit cost to protect 1 unit of primary flow; while our approach only uses 4.6 unit cost. The reason behind this is that *p*-Cycle-based VN backup model requires less protection bandwidth resource as straddling link of each used *p*-Cycle requires no spare capacities. Indeed, Figure 5.2(b) shows that in period 2, SFE- DPP uses 1.06 unit of bandwidth to protect 1 unit of primary flow; while our approach RVNE only uses 0.53. Accordingly, the results showed in this curves confirm my expectation that protection against single link failures using disjoint working and bypass paths or pre-selected backup detours may result in high bandwidth consumption. These models are unable to optimize the link bandwidth division and the sharing between primary and backup paths.

5.5 Performance Evaluation of VNM-CNLP

5.5.1 Performance Evaluation Metrics

To evaluate the performance of the VNM-CNLP approach, the following metrics are measured in the simulation. Because the JNLE-CG approach has been evaluated in Section 5.4 and the emphasis of the VNM-CNLP scheme is on the protection part, we mainly evaluate the metrics in terms of VN protection.

1) *VN backup cost*: measured as the cost of the used bandwidth for VN protection.

2) *VN backup efficiency*: measured as the ratio of VN protection cost to the amount of protected bandwidth used by VN embedding paths.
3) *Bandwidth capacity redundancy*: measured as the ratio of bandwidth used for VN protection to the bandwidth used by VN embedding paths.

5.5.2 Evaluation Scenario

To evaluate the performance of our VNM-CNLP approach, the following scenarios will be assessed:

1) VNM-CNLP-1: uses JNLE-CG approach for VN mapping and *p*-Cycle-VSP for VN protection.
2) VNM-CNLP-2: uses JNLE-CG approach for VN mapping and *p*-Cycle-CVSP for VN protection.
3) VNM-CNLP-3: uses JNLE-CG approach for VN mapping and *p*-Cycle-SPP for VN protection.
4) Benchmark: uses JNLE-CG approach for VN mapping and NEPC for VN protection.

5.5.3 Evaluation Results

The focus in this Section is on quantifying the advantage of using the proposed *p*-Cycle protection mechanisms against single physical node failure in terms of VN protection cost, bandwidth redundancy and VN protection efficiency. Experiments on large mix of VN requests confirm the superiority of our protection techniques over benchmark.

1) Based segment p-Cycle protection approaches vs. benchmark: Figure 5.3 plots the backup cost of protected bandwidth used by embedding paths vs. the allocation time periods. This Figure shows that segment *p*-Cycle-based protection approaches provide the lowest VN backup cost over a large mix of VN requests. We observe the same tendency, in terms of VN backup efficiency, i.e., cost of protecting 1 unit of bandwidth redundancy as shown in Figures 5.4 and 5.5 respectively. The main reason behind this is that segment *p*-Cycle-based VN backup approaches compared to NEPC model use protection bandwidth resource with lower cost and maximizes the backup bandwidth sharing among protected VN embedding paths. Indeed, averaging over the backup efficiency obtained over 10 periods shown in Figure 5.4, I conclude that benchmark NPEC uses on average 5.48 unit cost to protect 1 unit of protected bandwidth; while VNM-CNLP-1, VNM-CNLP-2 and VNM-CNLP-3 use on average 3.84, 1.95 and 3.68 unit cost respectively. Accordingly, these results confirm our expectation that protection against a single node failure using node encircling *p*-Cycle may result in high VN protection cost as it is unable to maximize the sharing of used VN backup resources. For example, in period 6 we can note that NEPC provides the worst results in terms of backup cost, backup efficiency and bandwidth capacity redundancy, this is related to the fact that NPEC setups almost a distinguish backup for each VN request. As shown in Figure 5.5, the capacity redundancy at period 6 is equal to 0.84, so there is roughly no resource sharing among VNs backups.

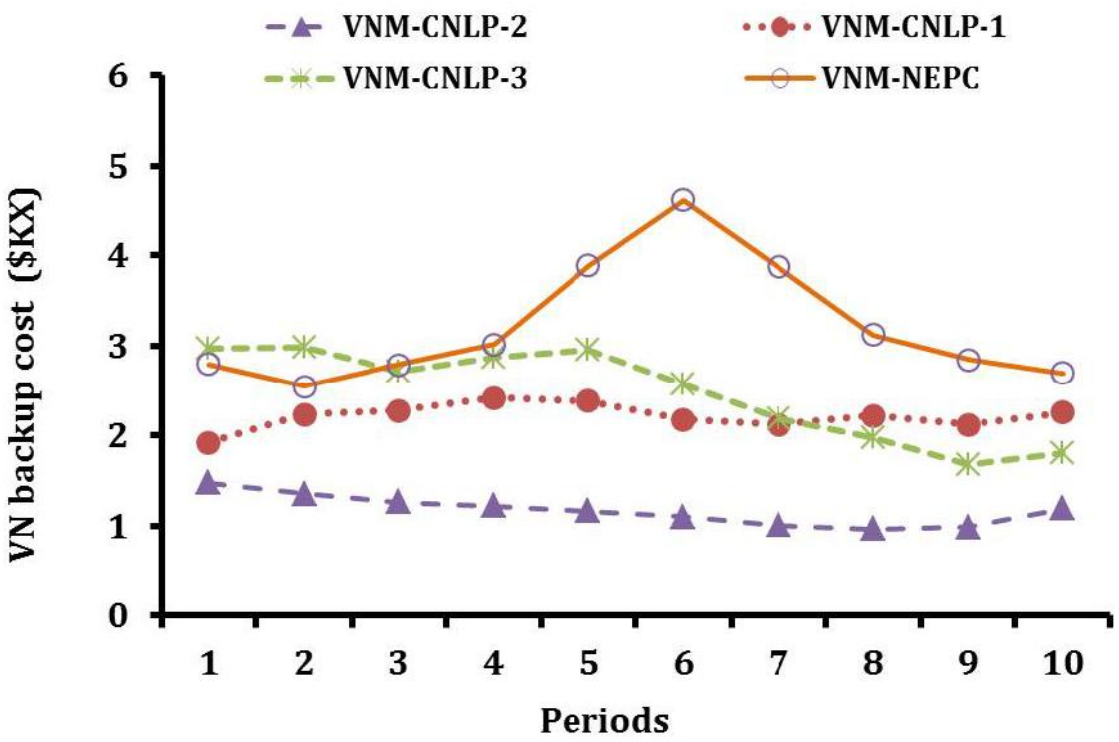


Figure 5.3: VN backup cost

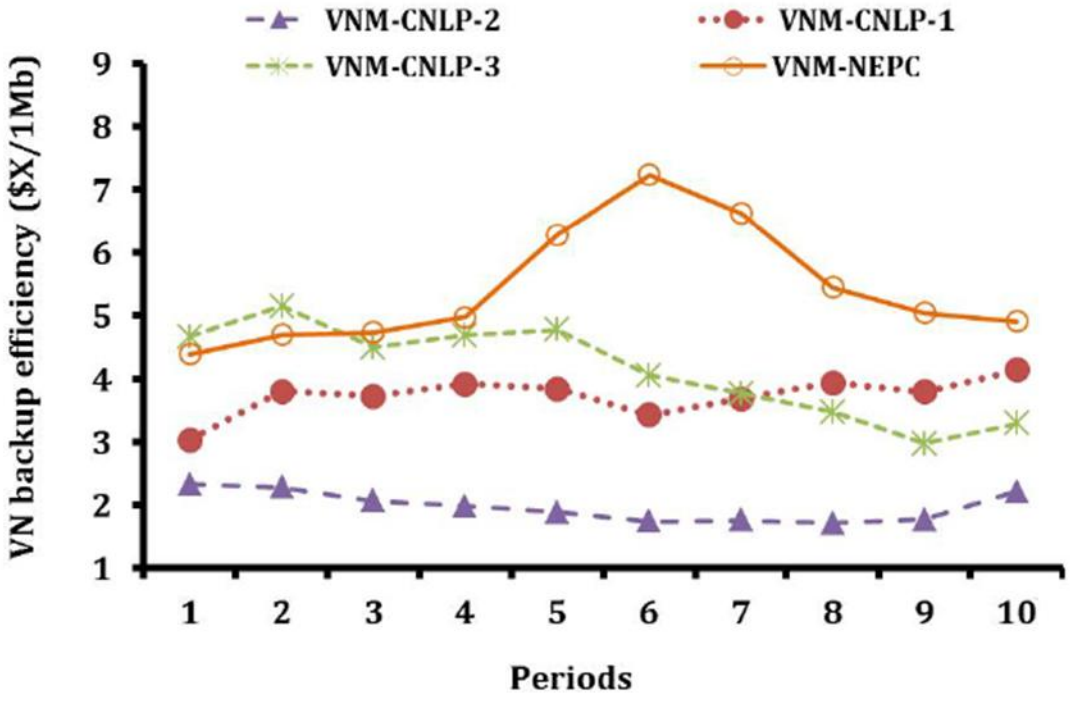


Figure 5.4: VN backup efficiency

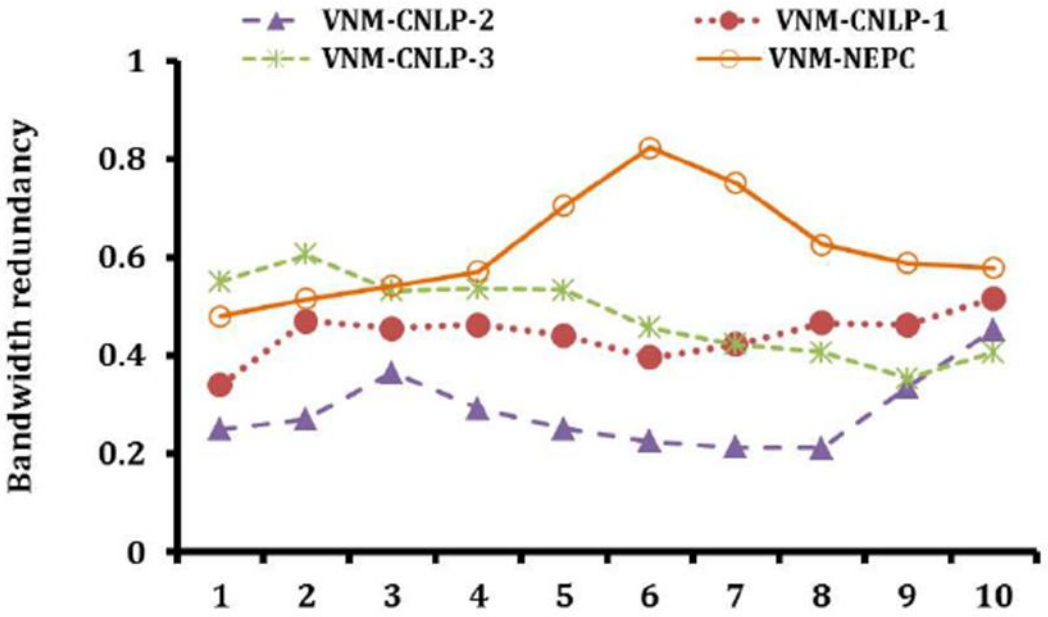


Figure 5.5: VN capacity efficiency

2) Why proposing three segment p-Cycle protection techniques: As aforementioned, the three based segment *p*-Cycle protection techniques provide on average better results than benchmark NPEC. However, each of them has its own advantages and shortcomings that can be summarized as following.

- *p*-Cycle-VSP has the lowest computation time, as it uses the lowest number of span *p*-Cycle candidates. However, since it provides protection by span, an embedding path can be protected by more than one *p*-Cycle. Accordingly, this technique has the largest failure recovery time.
- *p*-Cycle-CVSP has the lowest backup cost, the best efficiency and the lowest redundancy ratios, because it has the highest *p*-Cycle protection sharing ratio. However it has the largest computation time, since many new virtual spans which are generated by combination are added into protection and these virtual

spans use the largest number of span protection *p*-Cycles as candidates. Moreover, the failure recovery time is comparable to *p*-Cycle-VSP but worse than *p*-Cycle-SPP.

- *p*-Cycle-SPP has the lowest failure recovery time, as it proposes a protection *p*-Cycle per embedding path. However, it provides the worst results in terms of backup cost, backup efficiency compared to span *p*-Cycle-based techniques. This is related mainly to the fact that it uses a lower VN protection resources sharing ratio.

5.6 Summary

In this Chapter, the performances of *p*-Cycle technique based survivable VNE approaches and their corresponding benchmarks are evaluated. The scenario of substrate network and virtual network request are simulated by using C++ and all the data used by ILP model are processed by CPLEX. Analyses of the approaches have been evaluated in terms of embedding metrics and protection metrics. The strengths and weaknesses of each approach are explained. Also, comparisons of the proposed approaches with benchmarks have been made. Based on the result, it is evident that the approaches developed in this research do improve the efficiency of resource utilization and failure recovery time, and as a result will lead to a profit growth of InP. In addition, this research addresses the scalability problems during the implementation stage.

6. Conclusion and Future Work

6.1 Conclusion

The concept of network virtualization has been proposed to overcome Internet ossification by allowing the sharing of a common physical substrate by several providers running their own services in a fully isolated and protected way. In the research of network virtualization, the efficient embedding of Virtual Networks on the physical substrate represents one of the main challenges, since it is necessary to set up a coordinated procedure to provide the requested set of resources in the most beneficial way. Efficiently and quickly mapping a new virtual network, with constraints on the virtual nodes and links, onto specific physical nodes and links in the substrate network offers an InP competitive advantage over other providers.

In literature, a lot of research has been investigated in this area, but VN embedding still remains an insufficiently explored domain. Most of the previous proposals do not provide survivability guarantee for VN in their methodology. In this thesis, the survivability problem is explored in a network virtualization environment and two high-performance survivable VN embedding approaches are proposed for link and node protection by using various *p*-Cycle-based technologies.

Initially, a comprehensive survey of VN embedding is presented by categorizing and highlighting the features of each classification. This is followed by another analysis of previous survivability studies about node and link in both a general case and a network virtualization environment.

Subsequently, the RVNE approach, which focuses on the protection against single substrate link failure, is presented. This approach performs join VN node and link mapping using the CG technique which increases InP's profits because resources are utilized efficiently. In the second phase, this technique minimizes the backup resources while providing a full VN protection scheme using a *p*-Cycle protection mechanism. Experiments on a large mix of VN requests show a clear advantage of the RVNE approach over relevant benchmarks. Indeed, InP's profits are increased up to 20%, VN embedding paths can be fully protected and bandwidth resources are used more efficiently.

In conclusion, another survivable VN embedding approach is proposed that provides a protection mechanism for the physical layer, while taking into account constraints from the virtual network layer. This is done for the purpose of guaranteeing full VN protection against a single physical node failure and a multiple logical links failure. Experiments on a large mix of VN requests confirm the superiority of this research's protection techniques over existing benchmarks.

6.2 Future Work

The results of this research open up several avenues in which further research may be directed.

6.2.1 *p*-Cycle-based protection against combined link and node failure in VN

In this thesis, the proposed *p*-Cycle-based approaches focus on VN protection against

link failure and node failure respectively and a good performance has been achieved in each type of protection. However, in reality, both link and node are required to be protected. It is complicate, even conflictive in some cases, to deploy two kinds of *p*-Cycle-based protection on one substrate network. As a result, a protection approach for combined node and link failure recovery, which has been investigated extensively in general cases, is a field for future research.

6.2.2 Path and segment protecting *p*-Cycle-based Virtual Span Protection Technique

The VNM-CNLP approach is cost-effective against node failure protection, but it still has potential for improving resource utilization. In theory, a combination based on the *p*-Cycle-CVSP and *p*-Cycle-SPP approaches can obtain a more optimal solution than existing approaches. However, the scalability problem is a big challenge that needs to be resolved.

7. References

[1] T. Anderson, L. Peterson, S. Shenker, J. Turner, "Overcoming the Internet impasse through virtualization," *Computer*, vol.38, no.4, pp. 34- 41, Apr. 2005.

[2] J.S. Turner, D. E. Taylor, "Diversifying the Internet," *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, vol.2, Dec. 2005, pp.755-760.

[3] N.M. Mosharaf Kabir Chowdhury, R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862-876, Apr. 2010.

[4] S. Baucke et al., "Virtualization approach: concept, 4WARD project," Deliverable 3.1.0, 2009.

[5] M. Yu, Y. Yi, J. Rexford, M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration." *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 2, pp. 17-29, 2008.

[6] J. Fan, M.H. Ammar, "Dynamic topology configuration in service overlay networks: a study of reconfiguration policies," in *Proc. INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, Apr. 2006, pp.1-12.

[7] J. Lu, J. Turner, "Efficient mapping of virtual networks onto a shared substrate," *DCSE department, Washington University in St Louis, Technical Report*, vol. 35, pp.1-11, 2006.

[8] Y. Zhu, M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," *INFOCOM 2006. 25th IEEE International Conference on Computer Communications.* in *Proc*, Apr. 2006, pp.1-12.

[9] S. Zhang, Z. Qian. S. Guo, S. Lu, "FELL: A flexible virtual network embedding algorithm with guaranteed load balancing," *Communications (ICC), 2011 IEEE International Conference on*, Jun. 2011, pp.1-5.

[10] K.K. Guatam, Rai. Anurag, "A survivability strategy in route optimization mobile network by memetic algorithm," *(IJCSIS) International Journal of Computer Science and Information Security,* vol. 7, no. 1, pp.131-134, Jan. 2010.

[11] A. Jarray, A. Karmouch, "Column generation approach for one-shot virtual network embedding," presented at the *IEEE GlobeCom Workshop MENS*, Anaheim, USA, Dec. 2012.

[12] A. Fischer, J.F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, Hermann de Meer "ALEVIN-A framework to develop, compare, and analyze virtual network

embedding algorithms." *Workshops der wissenschaftlichen Konferenz Kommunikation in verteilten Systemen 2011,* vol. 37, pp. 1-12, 2011.

[13] J. Lischka, H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *Proc of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures*, Aug. 2009, pp.81-88.

[14] G. Hernando, S. Pérez, J.M. Cabero, "Mobility-Aware Distributed Embedding (MADE) of virtual networks," *Future Network and Mobile Summit, 2010*, Jun. 2010, pp.1-8.

[15] I. Houidi, W. Louati, D. Zeghlache, "A distributed virtual network mapping algorithm," *Communications, 2008. ICC '08. IEEE International Conference on*, May 2008, pp. 5634-5640.

[16] Y. Xin, I. Baldine, A. Mandal, C. Heermann, J. Chase, A. Yumerefendi. "Embedding virtual topologies in networked clouds." in *Proc of the 6th International Conference on Future Internet Technologies*. *ACM*, Jun. 2011, pp. 26-29.

[17] N.M.M.K. Chowdhury, M.R. Rahman, R. Boutaba, "Virtual network embedding with coordinated node and link mapping," *INFOCOM 2009, IEEE*, Apr. 2009, pp. 783-791.

[18] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, J. Wang, "Virtual network embedding through topology-aware node ranking," *ACM SIGCOMM Computer Communication Review*, vol.41, no.2, pp. 39-47, Apr 2011.

[19] S. Singhal, G. Hadjichristofi, I. Seskar, D. Raychaudhri, "Evaluation of UML based wireless network virtualization," *Next Generation Internet Networks, 2008. NGI 2008*, Apr. 2008, pp. 223-230.

[20] R. Mahindra, G.D. Bhanage, G. Hadjichristofi, I. Seskar, D. Raychaudhuri, Y.Y. Zhang, "Space versus time separation for wireless virtualization on an indoor grid," *Next Generation Internet Networks, 2008. NGI 2008*, Apr. 2008, pp. 215-222.

[21] S. Perez, J.M. Cabero, E. Miguel, "Virtualization of the wireless medium: a simulation-based study," *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, Apr. 2009, pp. 1-5.

[22] R. Kokku, R. Mahindra, H. Zhang, S. Rangarajan, "NVS: a virtualization substrate for WiMAX networks," in *Proc of the sixteenth annual international conference on Mobile computing and networking*, *ACM*, 2010, pp. 233-244.

[23] Y. Zaki, L. Zhao, C. Goerg, A. Timm-Giel, "LTE wireless virtualization and spectrum management," *Wireless and Mobile Networking Conference (WMNC), 2010 Third Joint IFIP* , Oct. 2010, pp. 1-6.

[24] G. Bhanage, R. Daya, I. Seskar, D. Raychaudhuri, "VNTS: A virtual network traffic shaper for air time fairness in 802.16e systems," *Communications (ICC), 2010 IEEE International Conference on*, May 2010, pp. 1-6.

[25] G. Bhanage, I. Seskar, R. Mahindra, D. Raychaudhuri, "Virtual basestation: architecture for an open shared wimax framework," in *Proc. of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*. ACM, Sep. 2010, pp. 1-8.

[26] E.K. Paik, S. Lee, C. Lee, J. Han, C. Park, T. Kwon, Y. Choi, "Service differentiation using mobile femtocell virtualization," *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, Jan. 2010, pp.1-4.

[27] F. Zdarsky, I. Martinovic, J.B. Schmitt, "The case for virtualized wireless access networks," in *Proc. of the International Workshop on Self-Organizing Systems*, *2006*, LNCS 4124, pp. 90-104.

[28] H. Coskun, I. Schieferdecker, Y. Hazmi, "Virtual WLAN: going beyond virtual access points," *Electronic Communications of the EASST*, vol. 17, pp. 1-12, 2009.

[29] B. Aboda, "Virtual access points." [Online]. Available: http://www. drizzle.com/?aboba/IEEE/11-03-154r1-I-Virtual-Access-Points.doc.[May 22, 2003].

[30] A.D. Rivera, W.L. Zucci, "Virtualization of wireless network interfaces Wi-Fi IEEE 802.11," in *Proc. of the 9th WSEAS International Conference on Telecommunications and Informatics*, 2010, pp. 46-51.

[31] A.J. Nicholson, S. Wolchok, B.D. Noble, "Juggler: virtual networks for fun and profit," *Mobile Computing, IEEE Transactions on*, vol.9, no.1, pp.31-43, Jan. 2010.

[32] Y. Al-Hazmi, H. de Meer, "Virtualization of 802.11 interfaces for wireless mesh networks," *Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International Conference on*, Jan. 2011, pp. 44-51.

[33] A. Leivadeas, C. Papagianni, E. Paraskevas, G. Androulidakis, S. Papavassiliou, "An architecture for virtual network embedding in wireless systems," *Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on*, Nov. 2011, pp. 62-68.

[34] T. Trinh, H. Esaki, C. Aswakul, "Quality of service using careful overbooking for optimal virtual network resource allocation," *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2011 8th International Conference on*, May 2011, pp. 296-299.

[35] M. Rahman, I. Aib, R. Boutaba, "Survivable virtual network embedding", *NETWORKING*, LNCS 6091, pp. 40-52, 2010.

[36] T. Guo, N. Wang, K. Moessner, R. Tafazolli, "Shared backup network provision for virtual network embedding," *Communications (ICC), 2011 IEEE International Conference on*, Jun. 2011, pp.1-5.

[37] G. Shen, W.D. Grover, "Extending the *p*-Cycle concept to path segment protection for span and node failure recovery," *Selected Areas in Communications, IEEE Journal on*, vol.21, no.8, pp. 1306-1319, Oct. 2003.

[38] F. Zhang, W. Zhong, "Performance evaluation of *p*-Cycle based protection methods for provisioning of dynamic multicast sessions in mesh WDM networks," *Photonic Network Communications 2008*, vol. 16, no.2, pp. 127-138, Sep. 2008.

[39] A. Frikha, B. Cousin, S. Lahoud, "Extending node protection concept of *p*-Cycles for an efficient resource utilization in multicast traffic," *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, Oct. 2011, pp. 175-178.

[40] J. Doucette, P.A. Giese, W.D. Grover, "Combined node and span protection strategies with node-encircling *p*-Cycles," *Design of Reliable Communication Networks, 2005. (DRCN 2005),* in *Proc.5th International Workshop on*, 16-19, Oct. 2005, pp. 213-221.

[41] D.P. Onguetou, W.D. Grover, "A new insight and approach to node failure protection with ordinary *p*-Cycles," *Communications, 2008. ICC '08. IEEE International Conference on*, May 2008, pp. 5145-5149.

[42] H. Liu, J. Wang, "A new way to enumerate cycles in graph," *Telecommunications, 2006. AICT-ICIW '06. International Conference on Internet and Web Applications and Services/Advanced International Conference on*, Feb. 2006, pp. 57.

[43] M. Chowdhury, M.R. Rahman, R. Boutaba, "ViNEYard: virtual network embedding algorithms with coordinated node and link mapping," *Networking, IEEE/ACM Transactions on*, vol.20, no.1, pp.206-219, Feb. 2012.

[44] M.E. Ubbecke, J. Desrosiers, "Selected Topics in Column Generation," *Operations Research*, vol. 53, pp. 1007-1023, 2005.

[45] W.D. Grover, D. Stamatelakis, "Cycle-oriented distributed preconfiguration: ring-like speed with mesh-like capacity for self-planning network restoration," *Communications, 1998. ICC 98. Conference Record. 1998 IEEE International Conference on*, vol.1, Jan. 1998, pp. 537-543.

[46] M.S. Kiaei, C. Assi, B. Jaumard, "A survey on the *p*-Cycle protection method," *Communications Surveys & Tutorials, IEEE*, vol.11, no.3, pp. 53-70, 3rd Quarter 2009.

[47] E.W. Zegura, K.L. Calvert, S. Bhattacharjee, "How to model an internetwork," *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation.* in *Proc. IEEE*, vol.2, 24-28 Mar. 1996,

pp. 594-602.

[48] B. Jaumard, H. Li, S. Sebbah, "Design of path-segment-protecting *p*-Cycles in survivable WDM mesh networks," *Telecommunications Network Strategy and Planning Symposium (NETWORKS), 2010 14th International*, 27-30 Sep. 2010, pp. 1-6.

[49] C.G. Gruber, "Resilient networks with non-simple *p*-Cycles," *Telecommunications, 2003. ICT 2003. 10th International Conference on*, vol.2, 23 Feb.-1 Mar. 2003, pp. 1027-1032.

[50] Y. Chen, J. Li, T. Wo, C. Hu, W. Liu, "Resilient virtual network service provision in network virtualization environments," *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on* , 8-10 Dec. 2010, pp. 51-58.

[51] J.A. Zubairi, "Current practices for MPLS protection," *High Capacity Optical Networks and Enabling Technologies, 2007. HONET 2007. International Symposium on*, 18-20 Nov. 2007, pp. 1-5.