A QoS-based Resource Selection Approach for Virtual Networks

Saba Behrouznia

A thesis in the Concordia Institute for Information Systems Engineering presented in partial fulfillment of the requirements for the degree of

MASTER OF APPLIED SCIENCE

in Quality Systems Engineering

Concordia University

Montreal, Canada

April 2015

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Saba Behrouznia

Entitled: A QoS-bases Resource Selection for Virtual Networks

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Quality Systems Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_Dr. Chun Wang_____ Chair

_Dr.Jamal Bentahar _____ Examiner

_Dr. Anjali Agarwal_____ Examiner

_Dr Rachida Dssouli and Dr May El Barachi __ Supervisor

Approved by _____
Chair of Department or Graduate Program Director

_____
Dean of Faculty

Date April 20th 2015

Abstract

A QoS-based Resource Selection Approach for Virtual Networks

Saba Behrouznia

The Internet has gained an outstanding success in a short amount of time and it became a critical infrastructure for accessing information and global commerce. With the help of the Internet and its new channels for connecting people a new way of communication has been established. Meanwhile, its great success leads to new limitations. The Internet consists of various network infrastructure providers with different objectives which makes emerging of new technologies or major architectural changes that require cooperative agreements, relatively impractical. While the current Internet architecture is not suitable for supporting many types of applications, network virtualization is considered as promising, yet challenging solution of these limitations. Network virtualization separates the role of traditional internet service providers (ISPs) into physical infrastructure providers (PIPs) responsible for deploying the physical infrastructure and service providers (SPs) offering end-to-end services to end users. Another motivation for network virtualization is the possibility to add value in the virtualization layer aiming to make use of new technologies (e.g. QoS schemes) and customizing existing technologies to adapt specific services (i.e. customizable networks). This provides the means to run multiple virtual networks on a shared substrate network simultaneously while each virtual network is customized for a specific use.

The key challenge in virtual networks is the problem of assigning virtual nodes and links to physical resources. Virtual network mapping/embedding consists in finding the most suitable physical nodes and links in the physical network in order to map virtual network requests with certain constraints on virtual nodes and links. The goal of this thesis is to design and implement substrate network resource selection scheme to increase the overall efficiency of the virtual network embedding process and satisfy the set of predefined resource constraints. This work assumes the existence of a virtual infrastructure provider requesting virtual networks from physical infrastructure providers and proposes a selection algorithm based on service-oriented architecture. Our proposed virtual network embedding algorithm is a heuristic algorithm that considers static attributes along with dynamic attributes of nodes and links as well as end-to-end QoS constraints.

Acknowledgments

First and foremost, I would like to express my gratitude to my dear supervisor, Professor Rachida Dssouli, for her valuable support and guidance during my studies at Concordia. Her technical help along with encouragements and constructive advices were a real source of motivation and inspiration to me. Her advice and friendship eased many tasks during times of difficulty.

Secondly, I am really grateful to Dr. May El Brachi who has helped me throughout this research and enriched my knowledge. She has supported me incredibly with her valuable ideas and comments during this work. I am also thankful to Dr. Nadjia Kara for her guidance and help in this work. I would like to thank my fellow coworkers specially Sleiman Rabah for his great teamwork and support.

I would like to thank my dear parents who have given me the opportunity of a good education. Their support and love during my studies was a great source of motivation for me.

Contents

List of Figures

List of Abbreviations

InP Infrastructure Provider

IPTV Internet Protocol Television

ISP Internet Service Provider

PIP Physical Infrastructure Provider

QoS Quality of Service

SP Service Provider

VIP Virtual Infrastructure Provider

VNet Virtual Network

VNP Virtual Network Provider

VNE Virtual Network Embedding

VOIP Voice Over Internet Protocol

Chapter 1

Introduction

1.1 Motivation

In short period of time, the Internet has gained a phenomenal success and it became a critical infrastructure for accessing information and global commerce. Internet has established a new way of communication by providing new ways of connecting people but its stunning success caused new limitation. Internet consists of various network infrastructure providers with different objectives which makes emerging of new technologies or major architectural changes that require cooperative agreement, relatively impractical (1). While the current Internet architecture is not suitable for supporting many types of applications, network virtualization is considered as promising, yet challenging solution of these limitations.

Network virtualization refers to creation of several isolated logical networks that can coexist on the same infrastructure. Network virtualization leads to better flexibility, security and administration by allowing multiple heterogeneous network architecture to share a physical infrastructure. It separates the role of traditional internet service providers (ISPs) into infrastructure providers and service providers. The Infrastructure providers is responsible for deploying the physical infrastructure while the service providers offer end-to-end services such as layer 3 VPNS and VoIP to end users.

From the business point of view, due to the high cost of deployment of physical networking, many service providers that could potentially offer different services may not be able to even enter the market. The network virtualization technology can be deployed and the physical resources could be partitioned into separate sections and form different virtual networks operated by different service providers. This is helpful for those small players to enter the market and lease the resources of an existing network infrastructure and offer their services. Another

motivation for network virtualization is the possibility to add value in the virtualization layer aiming to make use of new technologies (e.g. QoS schemes) and customizing existing technologies to adapt specific services (i.e. customizable networks). This provides the means to run multiple virtual networks on a shared substrate network simultaneously while each virtual network is customized for a specific use (2).

One of the main challenges for network virtualization is the dynamic discovery and selection of virtual resources that can be performed to form virtual networks. The virtual network (VNet) is a logical entity that is composed of virtual nodes that are connected by virtual links. VNet provisioning is mapping or embedding virtual nodes onto physical nodes and virtual links onto physical links or paths in order to create a virtual network. In a virtual network, the virtual infrastructure provider has to select and aggregate the virtual resources offered by physical infrastructure provider in order to create a virtual network on top of the existing physical network. To achieve this approach, there is a need for an efficient resource selection approach which considers both static and dynamic attributes in order to efficiently allocate resources from physical network to virtual network.

The virtual network embedding problem is known to be NP-hard and most of the past researches are proposing either heuristic algorithms or unrealistically restricting the problem space. Unlike VPN (Virtual Private Network) which have standard topology, such as full mesh and hub-and-spoke (3) virtual networks have diverse topologies. Moreover, in VNet provisioning problem, the VNet request has resource constraints such as CPU for nodes and bandwidth for links that the embedding has to satisfy. The other property that makes the embedding problem difficult is considering the dynamic arrival of VNet requests versus assuming that we know all the requests in advance (offline). After all, since the physical resources are limited, the embedding algorithm must perform admission control to manage the requests and reject those that can’t be embedded (4).

1.2 Thesis Objective

The key challenge in virtual networks is the problem of assigning virtual nodes and links to physical resources. Virtual network mapping/embedding is to find the most suitable physical nodes and links in the physical network in order to map VNets with certain constraints on virtual nodes and links. The goal of this thesis is to design and implement substrate network resource selection scheme to increase the overall efficiency of the virtual network embedding process.

Each VNet request contains certain static characteristic requirements which must be described for nodes and links. The resource selection scheme should be capable of matching static attribute requirements of the requests to each PIP network and select the best matching physical infrastructure provider (PIP). The network element can be a router, switch, access point or link. Static attributes of an element describes characteristics, properties and functions of an element such as node type (router, switch, access point, base station), interface type (Ethernet, optical fiber, radio, ATM) and link type (Optical, Coaxial, Microwave).

Considering only static attributes of the elements in virtual network embedding is not enough for an efficient performance of the algorithm. Thus a cost-effective utilization of dynamic attributes is necessary in resource selection procedure. However, there are certain obstacles in order to acquire the precise values of those dynamic network attributes due to the instability nature of these attributes. Monitoring the dynamic attributes requires a significant monitoring cost and on the other hand the infrastructure providers are not willing to reveal their information to another business entity such as virtual infrastructure provider (VIP). Therefore, a resource discovery procedure should be able to obtain the most vital dynamic attribute information in order to deliver an efficient resource selection solution.

The main Objective of solving the virtual network mapping problem is to both make efficient use of the underlying resources while satisfying the set of predefined resource constraints.

This work assumes the existence of VIP requesting VNets from PIPs (2) and proposes a selection algorithm based on service-oriented architecture proposed in (5). Our proposed virtual network embedding algorithm is heuristic algorithm which considers static attributes along with dynamic attributes of nodes and links and end-to-end QoS constraints.

1.3 Thesis Contribution

In this work we proposed an algorithm well-suited for the service-oriented network virtualization architecture. One of the contributions of our work is that a new class of global constraints has been proposed for the virtual network mapping problem, however, we assume that the requests are known in advance. Besides, we propose a node power-degree in our algorithm and consider the capacity of the substrate nodes in order to make the best use of the substrate network. We have also taken into consideration the end-to-end QoS attributes in the network and the path selection is performed base on the path with highest ranks. For better performance, we consider using path splitting in link mapping stage and compare the results with the algorithm that only utilize k shortest path for link mapping.

1.4 Thesis organization

The remainder of this thesis is as follows:

In chapter 2, background and related work in virtualization, network virtualization and related business models are presented. The detailed description of service-oriented network virtualization business model and architecture is discussed. In chapter 3, we present a complete survey of the current existing virtual network embedding algorithms and we compare them based on different metrics. In chapter 4, we proposed a QoS-based resource selection approach for virtual network in details. In chapter 5, the simulation of the algorithm is given along with the evaluation of the results. In chapter 6, we make conclusions and discuss the potential future work.

Chapter 2

Background and Related Work

2.1 Overview

Network virtualization has emerged rapidly in order to become the alternative to the current Internet architecture. Network virtualization offers promising benefits and challenges and it has gained extensive popularity among researchers and commercial market. While network virtualization has been considered to be a relatively new concept, the original concept has been evolved throughout different applications such as VLANS, VPNS and overlay networks.

2.2 Chapter Organization

In this chapter we start by presenting the network virtualization concept and applications and then we discuss different business roles in network virtualization. We will also present the service-oriented network virtualization architecture that we have proposed our selection algorithm based on.

2.3 Virtualization Concepts

Virtualization has been used in different forms for many years. Virtualization refers to the abstraction between physical resources and their logical representation and may refer to different layers of a computer system or network.

There are different forms of virtualization such as operating system virtualization, storage virtualization and network components virtualization. Virtual machines is a good example of OS virtualization while cloud computing is an instance of computational resource virtualization and MPLS technology is a type of links virtualization. One of the biggest challenges in virtualization is to provide the capability of pooling computing resources from clusters of servers and assigning virtual resources to applications on-demand. Ultimately, the goal of virtualization is to reduce the capital and operational costs.

Virtualization can be applied to operating system in which a virtual machine creates isolated execution environments and it could also refer to storage virtualization which is decoupling logical storage from physical storage. Virtualization could also be applied to network which allows the coexistence of multiple virtual networks on top of a shared physical network. Here we summarize the major applications of network virtualization

2.3.1 Virtual Local Area Network (VLAN)

VLANs consist of group of devices on one or more LANs that are configured (using management software) so that they can communicate as if they were attached to the same network, when in fact they are located on a number of different LAN segments. VLANs are based on logical connection instead of physical and they are extremely flexible. (6).

2.3.2 Virtual Private Network (VPN)

Virtual private networks are basic form of network virtualization which is limited to traffic isolation comparing to virtualized networks that offer administrative and customization of the network. A virtual private network is creating a private network which contains links over public or shared networks. VPN enables the users to establish a secure connection to a remote computer while using the routing infrastructure provided by the Internet (7).

2.3.3 Programmable Network

To enable on-demand services, the separation of communications hardware from control software is required. If that separation is in place, software can be programmed irrespective of the underlying hardware to deliver necessary functionalities.

2.3.4 Overlay Network

Overlay network is a network built on top of one or more existing networks. It adds an additional layer of virtualization. Best known example is Internet which connects local area networks, built on local area networks (phone line) and add an IP header to all packets.

2.4 Network Virtualization

Network virtualization is a logical topology which consists of virtual nodes that are connected via virtual links. The highly diverse network application requirements have been a motivation for new networking technologies and also alternative architecture for the next generation Internet (8). This is expected to enhance the flexibility of subsequent network, diversity and manageability of the offered services and quality of services capabilities. Some of the virtual network applications are IPTV, online game and content distribution systems. Network virtualization separates control plane form data plane and as a result decoupling the role of traditional Internet Service Providers (ISPs) into two entities: Service Providers (SPs) and Infrastructure Providers (InPs).

Virtual network is composed of two different components which are node virtualization and link virtualization. Node virtualization is virtualization of hardware resources and a single substrate node may contain a number of virtual routers. Link virtualization allows the transport of multiple separate virtual links over a shared physical link.

2.5 Network Virtualization Design Objective

First idea of network virtualization was developing virtual test-beds for evaluating new network architecture. Then a need for a fundamental change was necessary to allow networking systems with alternative architecture and different implementation coexist on a shared infrastructure platform and to make each VNet appear to a user as a dedicated physical network with dedicated resources, networking services and security policies.

VNets can simultaneously support multiple network architectures, experiments and services (e.g. IPTV, VPNs, etc.) and they can be supplied as a service by VNet Provider(s) and delivered to VNet users to support their infrastructures and services. Virtualization layer allow forming customized networks without requiring for the internet service providers to reach an agreement to deploy new protocols. Network virtualization is relatively new research area and one of the challenges is how to allocate resources in infrastructure network to virtualized network.

2.6 Reference Business Model

Network virtualization enables various networks sharing the same physical resources. Virtual networks can deliver different kinds of applications such as IPTV, live stream and online banking. The players in the network virtualization model are different from the traditional network model. In this section we present the business roles in the service-oriented network virtualization and their relations; further details on the business model are presented in (2).

2.6.1 Roles in the reference business model

There are five various roles involved in this business model and here we summarize these roles.

Physical Infrastructure Provider (PIP): The PIP owns and manages the physical network infrastructure and is responsible for deploying and maintaining physical network resources. The PIP is involved in the low level services (i.e. routing) and by using virtualization technology they can divide their physical resources into isolated virtual networks. Based on the customer's requirements (i.e. topology, CPU and link bandwidth) the PIP set up the VNet from its resources.

Service Provider (SP): The SP implements the network protocols on virtual networks to offer end-to-end services to the end users. The SP is no longer tied to setting up a costly infrastructure to offer his services. Instead, he can use service building blocks offered by Virtual Infrastructure provider (VIP) and offer value added services to the subscribers.

Virtual Infrastructure Provider (VIP): The main role in this model is the virtual infrastructure provider (VIP) who finds and brings together virtual resources offered by the PIP and creates the virtual network. The VIP offers service building blocks and support SPs or other VIPs with these services and he doesn't deal with subscribers. The services offered by VIP are used by SPs to compose new value added services and offer them to the subscribers.

End Users: They have an agreement with SPs and they are the ultimate users of the value added services.

Service and Resources Registry (SRR): This module acts as the mediators among PIPs and SPs and it collects all the information required for finding other parties. The job of this entity is the focus of our work that is modeled as a description and discovery framework.

The other advantage of this model is that it offers hierarchy of roles. For example, SP who offers value added services to the end user can further offer its resources to the other SPs and so on. As shown in Figure 1 Hierarchical relationship among different roles in service-oriented architecture, each instantiated virtual network could add components on the network below it and establishing a vertical hierarchy. VNets could also act as merging layer and give the ability of making use of heterogeneous networking infrastructures. This cooperation among PIPs that enable the interworking functionalities could be considered as a horizontal dimension.

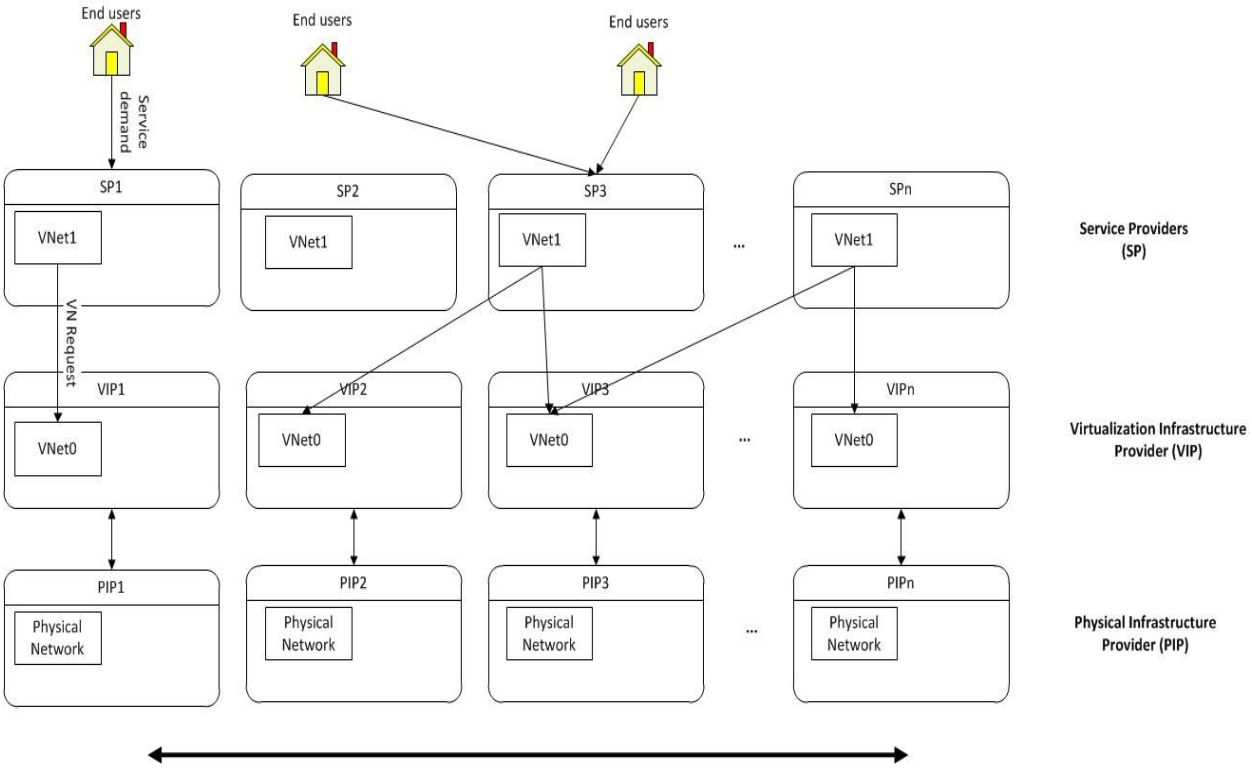


Figure 1 Hierarchical relationship among different roles in service-oriented architecture

2.6.2 Overview of service-oriented business model for virtual network

Several factors such as the high cost of deploying infrastructure resources, economies of scale and the emergence of utility computing were the motivation behind proposing this architecture. This service-oriented architecture is a hierarchical model in which different levels of services are offered by various players. Service in this model comprises of value-added services offered to end users and also functionalities ranging from low level network resources functionalities to high level ones. Four levels of services offered in this model, *essential services* which refers to services delivered by physical Infrastructure provider such as routing and transport services, *service enablers* which consist of functions used for end-user services (e.g. session management and security) along with *service building blocks* are offered by virtual infrastructure provider

This model is presented in Figure 2 and it proposes five different business roles; physical infrastructure provider (PIP), service provider (SP), virtual infrastructure provider (VIP) and consumer and services and resources registry (SRR). Physical infrastructure provider (PIP) owns the physical network infrastructure and by using the virtualization technique it can isolate its resources into different partitions. The SP offers the value-added services and holds the agreement with the consumer while the VIP is responsible for finding virtual resources offered by PIPs and instantiate VNet.

In a sample business scenario, a SP who wishes to offer QoS-enabled services to the end users requires a scalable virtual network in order to deploy its services. The offered services are designed to be consumed on-demand meaning the additional resources should be easily requested and added to improve the performance of the network. The instantiated virtual network would advantage from having value added properties such as security and QoS schemes. The SP sends the VNet request with all the details related to the resources it requires. The VIP evaluates the VNet request and allocates the related required resources with the participation of the PIP. Finally, the SP has access to the VNet created in order to offer its services to the subscribers. In the next section we discuss the architecture of the proposed framework.

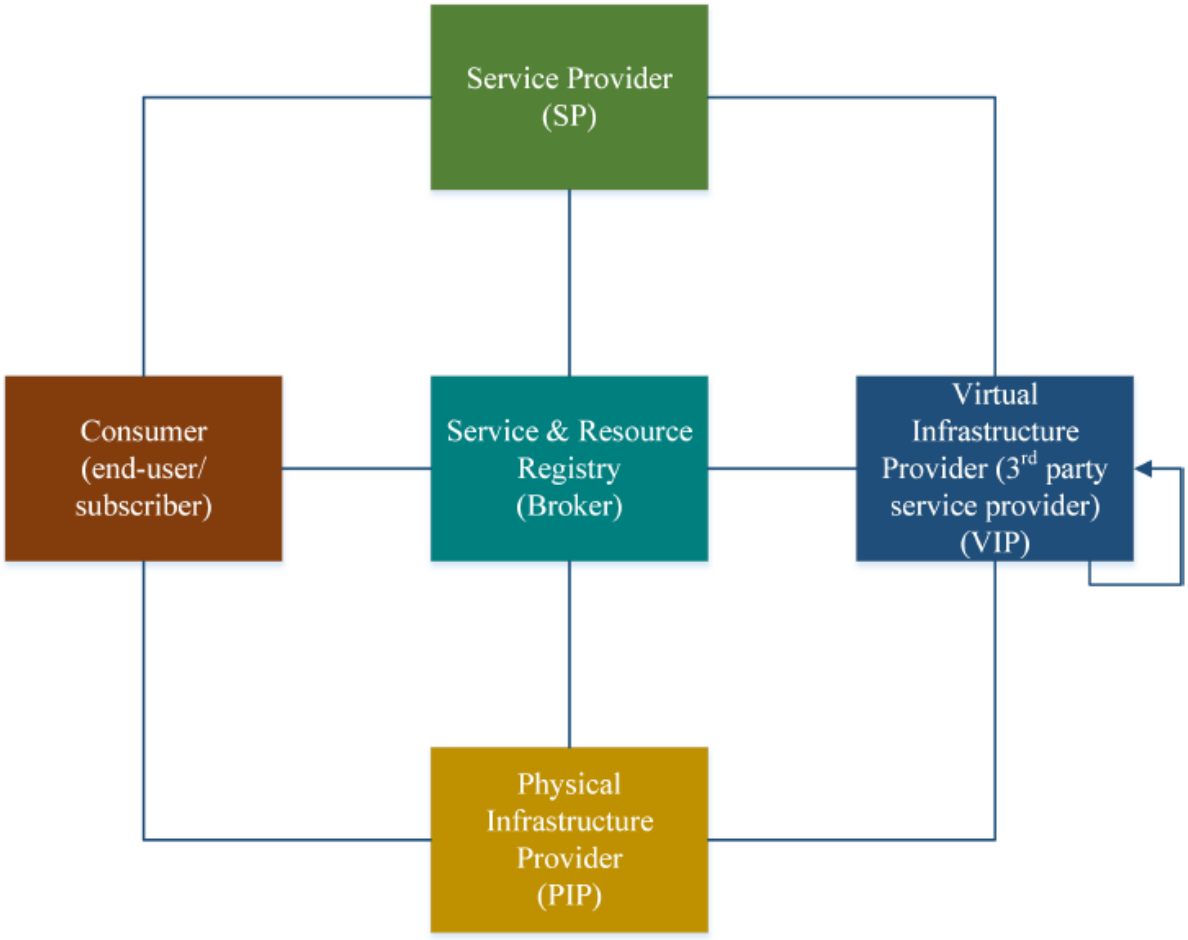


Figure 2 Business model for service-oriented architecture as proposed in (5)

2.6.3 Reference Architecture

Figure 3 Service-oriented architecture shows the high level of the service-oriented architecture which composed of two levels of VNets. In this layered architecture the PIP is represented in the physical level, the VIP is represented at the first virtual level and the SP is represented at the second virtual level. The VIP who wishes to offer service building blocks sends his request to the PIP and after resource discovery the VIP is able to make its instantiated virtual network. The Service Provider who wants to offer end to end services to the subscribers negotiate with the VIP and build its service over the scalable virtual network so that if there is a need for additional

resources it can be easily requested and added to satisfy the requirements of the SP and improve the overall network performance.

For managing the information transactions and organization a resource-broker approach has been offered. The introduced resource and service broker organizes the communications among different roles. The resource broker is the entity through which the participant parties are able to publish their resources and discover other parties' resources. It also provides the ability to select the best resources based on their characteristics and properties through the five-step embedding algorithm which is proposed in this work.

In this thesis we propose a new algorithm for resource selection and embedding for virtual networking environments based on a service-oriented network virtualization architecture proposed in (2). In the following section we give a descriptive review on network virtualization mapping problem in general.

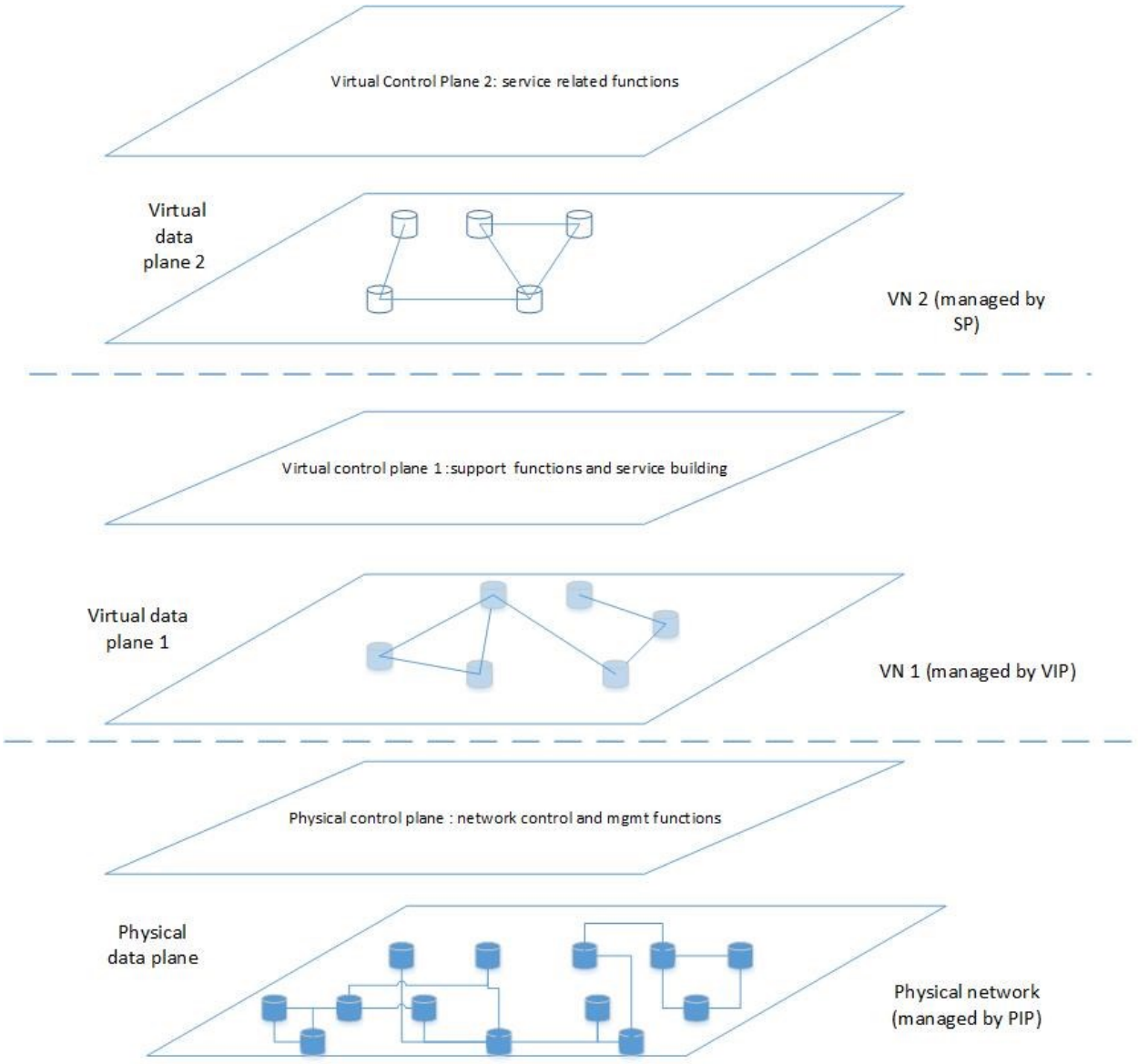


Figure 3 Service-oriented architecture (2)

2.7 Summary

In this chapter we presented related works in VNet embedding problem. VNet embedding problem aims to utilize the substrate network resources in the most efficient way. A detailed explanation of the VNet business model and architecture was presented. We realized that five roles are involved in the studied VNet business model. Service provider who offers the end-to-end services to the end user, physical infrastructure provider who offers the physical resources to the virtual infrastructure provider and the virtual infrastructure provider who wishes to instantiate the virtual network selects the best resources from PIPs and therefore build their virtual network. There is another role called broker who acts as the mediators among PIPs and SPs and collects all the information required for finding other parties. In the next chapter we present a complete survey in virtual network embedding problem and compared existing algorithms based on their objectives and the metrics they take into consideration.

Chapter 3

Literature Review

In this chapter we performed a complete survey for existing solutions for virtual network embedding problem and we compare different approaches based on different metrics.

3.1 Network Virtualization mapping problem

One of the main issues in network virtualization is an efficient utilization of substrate network resources which is dependent on efficient approach for virtual network embedding. Virtual network embedding refers to mapping virtual networks on physical substrate network resources.

For each request made for creating virtual network the virtual network embedding is responsible for mapping virtual nodes on physical nodes and virtual edges onto one or more physical paths. The Virtual Network embedding problem is a NP-hard problem even considering the offline mode in which the requests are known in advance and most of the existing solutions are heuristic algorithms. Previous researches have certain limitations, some of them only consider specific constraints such as bandwidth (9) or they consider offline version of VNet requests (10) or their work is limited to specific type of topologies such as hub-and-spoke (11).

This problem has gained a lot of attention in recent years (10, 12, 13, 14) due to the fact that the efficiency of the mapping algorithm can profoundly improve the number of VNets that can be placed on the physical resources and to reduce the consuming time.

The objective of virtual network mapping is to accept more virtual requests and therefore make efficient use of the substrate network while satisfying the nodes and links' resource constraints and reduce the embedding cost and increase the revenue for providers. Figure 4 Concept of virtual network mapping illustrates the concept of the virtual network mapping where a virtual network is embedded in a physical network. As seen in the picture, two virtual nodes from the same VNet are generally embedded in different physical nodes, while two virtual nodes from different virtual network can be embedded in the same physical node.

In the next part we will review resource discovery and allocation in network virtualization and we present three of the current existing algorithms and comparing them in terms of the technique and their approach along with their limitations.

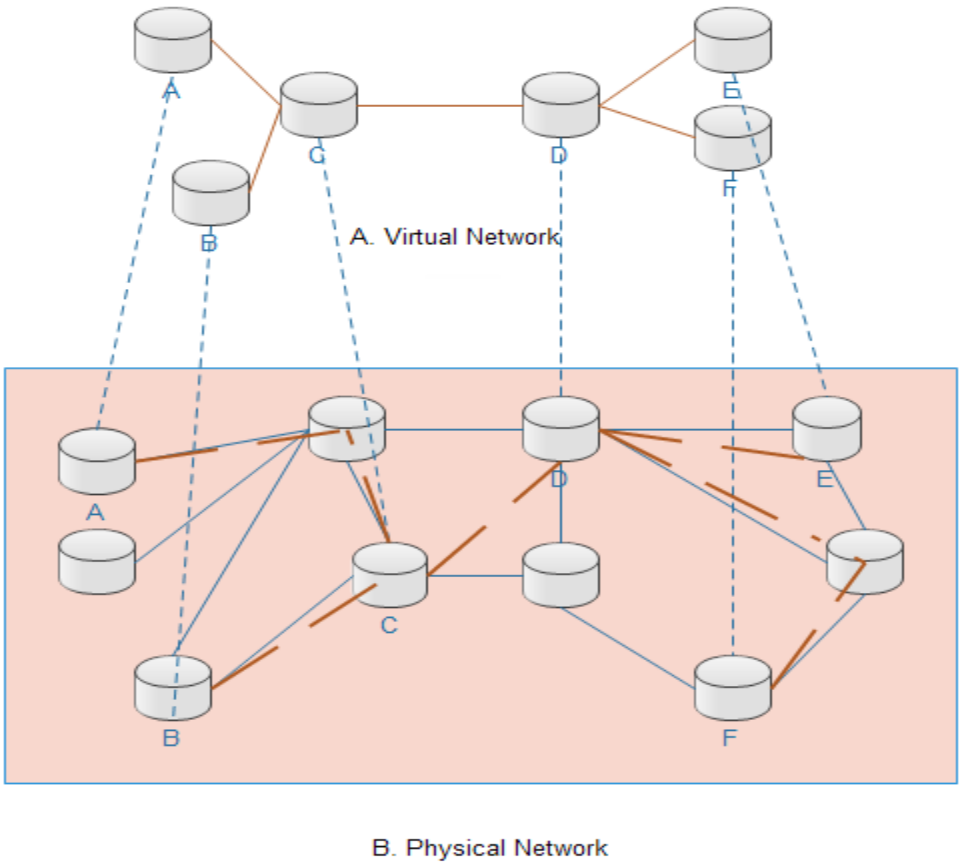


Figure 4 Concept of virtual network mapping

3.2 Resource discovery and allocation in network virtualization

As discussed earlier in this work, various roles are involved in the network virtualization environment. The PIP (Physical Infrastructure Provider) who manages the physical network and offers its resources in order to create isolated slices using virtualized techniques. The SP (Service Provider) offers services directly to the end uses which could be either simple or the

combination of service building blocks. The VIP (Virtual Infrastructure Provider) searches for virtual resources and put them together to create virtual networks. The VIP works with SPs or other VIPs with services but he doesn't have any direct agreement with end-users.

In the service-oriented business model, PIPs are responsible for allocating a section of their resources to VIPs and VIPs allocate their VNets (offering service building blocks) to SPs. This whole procedure is achieved via virtual network (virtual nodes connected by virtual links) embedding on one or more physical networks (physical nodes connected by physical links). This separation of roles allows each PIP only manages its physical infrastructure while each SP focuses on delivering the end-to-end services (requested resources from VIP) to the customers. VIP is responsible for creating virtual network based on SP request (15).

In service-oriented network virtualization architecture virtual network embedding involves different steps and the cooperation of the business roles. We present the steps as follows:

1. **Resource Description:** The PIPs offer their resources and the VIPs lease those that satisfy their requirements. PIPs require describing their resources (node and links characteristics) in order to facilitate the discovery step to retrieve information.
2. **Resource Publication:** PIPs advertise their resources through the broker. Generally the description of the resources is stored in the repositories or a discovery framework so that they can be acquired by the VNet requestors.
3. **Resource Discovery:** resource discovery is the phase when VIPs try to find the available resources in the underlying physical network. Resource discovery becomes ongoing step when various PIPs are involved and is delivered through a VNet request. A VNet requests delivers all functional and non-functional requirements of the VIPs to the broker who has the information of all the PIPs.
4. **Resource Selection:** This phase is about finding a set of appropriate nodes and links in a way that the characteristics of the nodes and links in the VNet request could be fulfilled.
5. **Resource Negotiation:** In this phase the SPs negotiate with various providers in order to select the most suitable one. They consider both the properties of the resources and also the QoS attributes.

6. **Resource Embedding:** this phase is the process of allocating physical resources to VNet requester (16).

In this work our focus is to propose an algorithm for resource discovery and selection. In the next part we will review the existing algorithms on resource discovery and mapping and in the next chapter we present our model.

3.3 Current Existing Mapping Algorithms

The main resource allocation challenge in network virtualization is the question of how to map virtual networks in physical networks and in the researches this problem is considered as virtual network embedding problem. The optimal resource allocation approach is necessary to provide customized end-to-end services to the customers. The performance of different algorithms can be compared based on various objectives (i.e. QoS, business revenue and acceptance ratio) (17).

There are many different mapping algorithms proposed in the literature to solve the virtual network mapping problem. The main objective of these algorithms are making an efficient use of the substrate network, improving the business profit by increasing the number of successful mapping and making a fast and responsive framework. However, those applications that have specific constraints such as packet loss rate or delay (end-to-end QoS attributes) could suffer from the poor performance if paths with high packet loss and end-to-end delay are utilized. Moreover, in the service-oriented architecture many scenarios exist in which applications have different network constraints and have specific preferences for each of them. For example, one application may require shorter paths for fast packet forwarding and low packet loss for better throughput.

Hence, there are some challenges in this topic that makes it difficult such as constraint on link and nodes, which is mainly CPU for nodes and bandwidth for the links. There are some additional constraints such as node location, link delay and QoS attributes which hasn't been addressed in most of the proposed algorithms. The other obstacle in network virtualization mapping is the online request, in most conducted researches the arrival of virtual requests are based on a distribution instead of arriving all at large mixture. It is also important for the method to cover various types of topologies instead of only consider some specific topologies (such as hub-and-spoke). A survey of network virtualization mapping algorithms could be found in (18).

3.4 Static vs. dynamic approaches

In static approaches once the VNet request has been embedded to the substrate network, the selected physical resources will be fixed regardless of the actual consumption during the VNet lifetime. In dynamic approaches, the virtual network embedding is dynamically adjusted to adapt to some external conditions such as path splitting.

3.5 Requirements for our virtual network embedding algorithm

Based on the performed literature survey, our VNE algorithm has to cover certain requirements. The requirements for VNE algorithm are as follows:

1. The VNE algorithm should support diverse topologies. Since virtual networks have diverse topologies, some of them (tree or hub-and-spoke) may be more common.
2. The VNE algorithm should consider both functional and non-functional attributes of a node. Functional attributes are node type, interface type, operating system, virtual environment type, network stack type, storage type, lick virtual technique and link type. Non-functional attributes which could be called as time-variant ones are those that change over time such as bandwidth, delay, CPU and packet loss.
3. In order to be practical, the VNE algorithm should handle requests as they arrive (dynamic requests)
4. The algorithm should support ranking and reputation of resources based on their QoS attributes to ensure efficiency of the selection process.

For the literature review, we surveyed 9 existing algorithm and compare them in the Table 1. Previous studies in virtual network embedding were limiting the problem scope in different ways. Some assumed that all the requests are known in advance (19) and (20), some works don't consider the node constraints (21) and (22) while some don't consider admission control (19, 20, 11) or they only consider a specific topology (19). The VNE approaches can be categorized to static or dynamic approach, one-stage VNE algorithms and two-stage VNE algorithms.

In (10) the authors proposed two types of virtual network assignments. Their objective was to provide a balanced link and node stress in the substrate network. The first one is the VNet assignment without reconfiguration (fixed through VNet lifetime) which is a basic scheme to get

near optimal substrate node performance and use it as a building block for other algorithm. The second algorithm is the VNet assignment with reconfiguration in which they developed a selective VNet reconfiguration scheme that prioritizes the reconfiguration for the most critical virtual networks (those on highly stressed nodes and links). In their work the main resource used by VNet nodes and links are CPU and bandwidth. The results show that reconfiguration only a subset of the most critical virtual networks reaches most of the benefits of dynamic reconfiguration while keeping the cost low.

In (11), a topology-based approach is proposed which is a combination of backbone-star substrate network topologies based on traffic constraint virtual network design. Directional graphs make this solution suitable for geographical spare networks however they consider the offline problem in which the requests are already known in advance. Their objective is to minimize the cost and they used the Mixed Integer Quadratic Programming to model their problem. The backbone-star topology consists of a backbone-node (single, centrally located node) and access node (has a single edge connecting it to backbone-nodes). Traffic constraints include termination constraint (describe the total traffic terminating at the virtual network's access node), pairwise traffic constraint and distance constraint. The backbone node mapping is performed through these steps:

1. Select an arbitrary mapping of backbone nodes
2. Connect access nodes to backbone nodes (in each iteration)
3. Find shortest paths between backbone node and access node
4. Determine link capacities (linear programing)
5. Find the best backbone node mapping.

Constructing a complete VNet topology with defined link capacity from previous steps, now we explore alternative backbone node mapping with the same VNet topology and link capacity. The process continues from the second step using the best backbone node mapping and will stop when there's no further improvement or a maximum number of iteration is reached. The limitation of this approach is that it assumes link capacities are sufficient to relax some constraints.

In (4), instead of restricting the problem like (10) and (11) they made the substrate network more supportive of the VNet embedding problem. They propose a simpler algorithm that make more

efficient use of the substrate resources and include path splitting and path migration algorithms. They also consider online approach in which the requests arrive randomly and leave the substrate network after certain time. They classify four main properties of the virtual network embedding problem. 1. Online vs. offline request 2. Limitation of node and link constraints 3. Admission control (to reject the requests if the resources are not efficient) 4. Variation in virtual network topologies. For node mapping they consider the CPU and location constraints while for link mapping the bandwidth has been considered. They use a greedy algorithm to assign nodes in which they sort and prioritize requests based on their potential revenues and then process them in order of priority. Candidate substrate node for each virtual node is determined based on available resources and requirements. For each virtual node the substrate node with "maximum available resources" is allocated. To assign links, for each virtual node, the k-shortest paths are computed and bandwidth requirements are considered. If none of the path could satisfy the bandwidth request, the request will be put in the queue. In order to make more efficient link mapping, it's better to make the physical paths shared by different virtual links rather than mapping of each virtual path completely to a single substrate path. In this approach, if a request arrives with a required bandwidth, but none of the substrate links have that specific bandwidth, the request will be rejected. While using path splitting allow the links with different capacities to serve the required bandwidth and support the VNet request which leads to the better resource utilization, reliability and load balancing.

In (23), authors propose an approach to mapping the virtual nodes as closely as possible in the substrate network and then assigning virtual links to the shortest paths which satisfy their demands. To embed a new VNet, virtual nodes with bigger resource requirements are mapped first. They used k-shortest paths algorithm to find paths between each pair of nodes and the effect of varying k for k-shortest paths on different substrate networks is analyzed. If a VNet request is rejected, it is stored at the end of the requests queue so that it may possibly be mapped later. They consider admission control and online VNet request and they don't limit the solution to any specific topology. The proposed approach in (4) utilizes node resources effectively but can make the nodes places far apart from each other and result in relatively longer paths. However, in (23) the solution focus on placing the virtual nodes as closely in the substrate as possible so the paths found for the virtual links mapping are the shortest in length.

The first proposal of a distributed approach was in (22) where two main entities, the substrate network and virtual network, are considered. The objective is to map virtual network requests on substrate networks efficiently while minimizing the cost. Unlike previous solutions the approach in this work is distributed on all the substrate nodes. The advantages of this multi-agent based approach are making the framework more robust by avoiding the single point of failure and allowing parallel high-speed processing of multiple VNet requests. The limitation of this work is the assumption of unlimited substrate network resource (CPU and bandwidth) and the offline assignment. A VNet mapping protocol is proposed to communicate and exchange messages between agent-based substrate nodes to achieve the mapping. Nodes handling the intelligent agents should be able to decide which mapping action to undertake. The solution is based on a VNet decomposition into elementary clusters (hub-and-spoke). Virtual nodes with higher capacities are selected as hubs and others as spokes. For mapping each hub-and-spoke cluster to substrate network, the substrate node with the highest capacity is selected as a root node and the hub node is assigned to it. Spokes nodes are then assigned by root substrate nodes based on a shortest path algorithm. After mapping all of the nodes and links, the cluster is removed (logically) and the same is repeated for other clusters. Selected substrate nodes need to communicate, collaborate and interact through a communication protocol to perform the above algorithm.

PolyViNE (24) is a policy-based distributed inter-domain embedding framework. The authors deploy a distributed protocol that coordinated participant InPs and ensure competitive pricing at every step of the process. Each InP uses its own policies to take decisions without any external restrictions, however the framework creates a high level competition among all the InPs by introducing competitive bidding. The substrate network contains multiple InPs, each one contains various substrate nodes and links and the border nodes in each InP connect it to other InPs through inter-domain links. In their model the service provider advertises its request to a number of InPs and their response indicates the ability of each InP to embed the requested virtual network and related cost. The service provider chose the offer with the lowest cost and meanwhile, InPs can communicate with each other during the matching process, if one InP does not have enough resource it distribute the request to others (by using the location constraints to forward the request). InPs can jointly participate in the bidding process and together provide resource for a single VNet request.

In (25) the authors proposed an auction based approach in which a partitioning algorithm with the goal of minimizing the cost and satisfying the SP's location constraints has been proposed. Their model is an open market model for automated service negotiation and contracting in network virtual embedding. The InPs in this model participate in a two-stage Vickery auction model. V-Mart offer an environment in which InPs can participate in a faithful and fair competition through auctioning over the VNet resources and for the SPs, it offers a customer-driven (considering SP as customer) virtual resource partitioning and contracting engine. Their model is flexible and the SP has the freedom to choose the best contracting strategy among multiple InPs instead of choosing a set of service offerings with pre-specified configurations and levels of performance from the InPs. They also propose a fair Market and the negotiation between SP and InP is being performed in a way that the SP is protected from pricing manipulation of the InPs and the InPs are able to compete with each other in an open and secure manner. The model is automated in a way that the negotiation and contracting process is automated to handle quick and on-demand VNet request. They also claim to have an efficient model both in terms of running time and the number of auctioning iterations. In this model The SP send a Request including the virtual topology and the requirements on the virtual resources. Then, each VNet bidder (InP) has to indicate the virtual resources it is willing to host and the related cost done in 2 steps. First the VNet bidder performs embedding of the VNet. Second, the VNet bidder provides a price quote. Afterwards, the SP obtains a set of price quotes for each virtual resource from the VNet bidders and partitions the VN into multiple segments and attaches them to specific VNet bidders. When all the list of segments is sent to all of the VN bidders as well as the winning VNet bidder of each segment and the winning Vickery price, the InPs make one last sealed bid that is upper bounded by the winning quote. Finally, the final winner of each segment is determined by the SP and they make the contract. The two recent auction approaches, the auctioneer plays the role of a central entity leading to resemblance to a centralized architecture.

In (26) the authors propose an aggregation-based discovery for virtual network environment. In this model, each InP has a monitoring agent which performs the task of monitoring aggregation values of certain dynamic attributes. InPs advertise their static attributes in a repository that can be reached by VnP. The aggregation values of certain dynamic attributes are monitored by a

Monitoring-Agent. They perform an initial filtering VNet requests over the set of the conditions that assure each VNet request satisfies certain set of basic requirements such as CPU and bandwidth. For each VNet request, max CPU requirement, total CPU, Max BW requirement and min total BW will be calculated and compared with their aggregation peers obtained from the substrate. Those substrates that cannot satisfy the requirement will be ignored by VNP and next one will be considered. Although Monitoring dynamic attributes make a heavy overhead on the network, it is highly needed in order to make an accurate embedding decision. To reduce the overhead instead of monitoring each single attribute individually they do the monitoring by calculating the aggregation of the monitored parameter. Aggregation values are calculated for individual substrate only after a change occur (successful VNet embedding, VNet departure); therefore, resources used for calculating aggregated do not impose a significant overhead. This approach gets more complicated when many VNPs are interacting with many InPs and the failure detection and capability of splitting VNets over multiple InPs are some of the challenges.

Table 1 Comparioson among existin VNE algorithms

Work	**Approach/ technique**	**Business Roles/Entities**	**Objective**	**Dynamic Resource monitoring**	**CPU/BW**	**Topology**	**On/Offline**	**Ranking/Reputation of resource**
(10)**. Zhu et al.**	Heuristic (greedy) Centralize	-Substrate Network Provider -Virtual Network	Load balancing	No	Both	General	Offline	No
(11) Jin Lu et al	Exact formulation Heuristic (iterative method) Centralize	-Substrate Network -Virtual Network	Cost	No	Both	Backbone Star	Offline	No
(4) M.Yu	Heuristic (greedy) Centralize	-Physical Network -Virtual Network	Revenue & Cost	No	Both	General	Online	No

(23) A. Razzaq	Heuristic (greedy+ closest node mapping)	Substrate Network Virtual network	Revenue & Cost	No	Both	General	online	No
(22) I. Houdi	Heuristic Distributed Multi-agent	Substrate Network Virtual network	Load balance, cost	No	Both	General	Offline	No
(24). Chowdhury et al.	Distributed Policy-based Aucrion-baesed Inter-domain VN embedding	Infrastructure Provider Service Provider	Revenue & Cost	No	Both	General	Offline	No
(25). Zaheer et al	Auction-based	Physical Infrastructure provider Service Provider	Revenue & Cost	No	NA	NA	NA	No
(26). Heli Amarasinghe	Broker-based approach	Physical Infrastructure provider Service Provider	Revenue & Cost	Yes	Both	General	Offline	No

In (27) a discovery framework is proposed for 4WARD (28) model in which they deploy a clustering approach for resource discovery. Each element is described by two types of attributes: functional such as type of node, virtualization tool and the OS and non-functional attributes such as bandwidth, capacity, cost or QoS. The framework deploy an approach in which they cluster the resources information and demonstrate them in a tree structure called dendrogram, a sample is given in Figure 5. They perform the selection phase based on this approach. First they compare the root of the tree with the receiving resource requirement; if they match they move forward to the leaves of the tree. Otherwise, they reject the request and evaluate the next coming request.

Figure 5 Sample dendrogram (27)

This approach facilitates the matching process by clustering the resources in a tree structure but it has its limitations. This framework doesn't consider the non-functional attributes which vary with time due to the overhead that monitoring creates on the network. The other limitations of this framework is if multiple PIPs have the same attributes then there is not additional constraints to distinguish the resources such as location and cost and the VIP should deal with all of them.

In (29) the authors apply Markov Random Walk (RW) model to rank a network node based on its available resource and topology attributes. For their first algorithm, they propose node mapping and link mapping as two steps of the network embedding as most of the researches in the literature. In node mapping stage they consider rank of the nodes and then use the shortest paths with un-splittable paths and solve the multi-commodity flow problem for splittable paths. They propose another algorithm which embeds both nodes and links in the same stage and use breadth-first search.

As shown in the Table 1, the existing algorithms for virtual network embedding are not satisfying our requirements and in this work we propose and evaluate our QoS-based resource selection approach for virtual network based on the given requirements.

3.4 Summary

In this chapter we presented a survey for existing virtual network embedding approaches. We presented the analyses and applications available in literature. It gives a brief and clear classification of virtual network embedding algorithm and explained approaches in detail. We have also compared the existing algorithms in terms of different approaches and the metrics they consider in their approaches. In the next chapter we present our QoS-based algorithms in detail along with the formulas used for node and link mapping.

Chapter 4

QoS-based Resource Selection Approach for Virtual Networks

4.1 Introduction

Our proposed selection and embedding algorithm for virtual networking environments is based on the service-oriented network virtualization architecture presented in (5) and is a heuristic algorithm which considers static attributes along with dynamic attributes of nodes and links and end-to-end QoS constraints. The resource broker proposed as part of this architecture allows the different roles to publish their resources and discover other role's resources. The broker has to make use of an efficient mapping algorithm in order to discover the most suitable resources based on properties requested in the VNet request and select the resources accordingly. The selection algorithm is used in two levels within the network virtualization hierarchy. At the first hierarchical level, the VIP wishes to instantiate the VNet and sends the request to the broker and the broker who has the information of the PIPs selects the best resources using the algorithm. At the second level, the SP wishes to build VNet and use the service building blocks offer by the VIP. In this level the QoS attributes should be considered. For the second level, the algorithm ranks the services based on their QoS parameters and the selection is performed depending on the type of the requested service. For example some services need low delay while some services require high throughput. In this section we propose our embedding algorithm in service-oriented virtual network environment.

4.2 Proposed Requirements

Candidate selection is the process of selecting the best resources or in case of multiple PIPs, the best PIP by analyzing all physical node and links to find the optimum matching substrate resources to host the virtual ones. Based on the service-oriented architecture our algorithm has to cover following requirements:

- The Selection Algorithm needs to consider both functional and non-functional attributes of a node. Functional attributes are node type, interface type, operating system, virtual

environment type, network stack type, storage type, link virtual technique and link type. Non-functional attributes which could also be called as time-variant ones are those that change over time such as bandwidth, delay, CPU and packet loss.

- The algorithm has to rank the PIPs and VIPs based on the QoS of their services.
- The algorithm should be dynamic and performs remapping if required. The example of dynamic nature of the model is a service provider who might deploy a new service at any time and maybe decides to terminate the service when it's not as profitable.
- It is desired to propose a simple algorithm as much as possible in order to avoid heavy computations.

4.3 Resource Selection and Monitoring in Virtualized Network

Candidate selection is the process of selecting the most suitable PIP by analyzing all the link and node resources to find the optimum matching substrate resources to host virtual resources. Upon receiving a resource discovery request, the broker selects the most appropriate resources that are aligned with the requirements as specified in the request and in return it identifies the list of the candidate resources. In the last stage of the algorithm (depending on the level of the virtualization) the QoS attributes of the offered services are ranked and considered in the selection procedure. The flowchart of the algorithm is given in Figure 6 Flowchart of the QoS-Based Approach for

1. Input set for virtual network mapping algorithm
2. Matching step which considers functional attributes of nodes and links
3. Filtering step which considers non-functional attributes of nodes and links
4. Aggregation algorithm in which the link mapping occurs using K-shortest path and the characteristics of the paths are computed using specified function of the application
5. Ranking which is related to the ranking of the links in PIPs based on their QoS constraints and resource allocation based on the requested service
6. Selection which is the final stage of the algorithm and the selected substrate nodes and links will be identified

The algorithm in this work, models requests that are known in advance (offline requests) for virtual network instantiation on the substrate network. Each request specifies the topology of the virtual network, along with functional and non-functional attributes as well as the QoS requirements. The algorithm assumes that the customer will specify the characteristics of their resource requirements very clearly and the broker is responsible to advertise the available resources. The input consists of different scenarios and network topologies and we evaluate the algorithm for each of the scenarios. Certain metrics used for evaluation of the algorithm in terms of the efficiency and execution time. In the following, first we describe the requirements for the algorithm, and then we present the operation of the algorithm. At the end, different input sets for various scenarios and the metrics for evaluation purposes are discussed.

The algorithm requires two types of inputs. Input for substrate network, which represents the physical infrastructures which host the virtual network. The other input consists of multiple virtual network requests which have to be embedded onto the physical network. In the request, all the functional and non-functional attributes of the nodes and links are given. There are also various constraints on both links and nodes. The substrate links have a limited bandwidth and the nodes have also limited processing power resources which make it hard to keep track of them all the time.

After the arrival of the request, in matching phase the algorithms searches for functional attributes of the nodes and links and selects the nodes and links accordingly. Functional attributes are those attributes that are static and will not change during the time. The example of functional attributes is node type, OS type, virtualization environment and link type.

After the matching phase, the algorithm filters the process by considering non-functional attributes such as CPU, BW, end-to-end delay and packet loss. In the filtering stage the node selection is performed based on the nodes that have the maximum available capacity among the substrate nodes to allocate the virtual nodes that require the largest CPU. For mapping the virtual links, the k-shortest path algorithm has been used and the substrate paths will be selected accordingly. The detailed description of the k-shortest path algorithm is given in the Appendix A.

In our algorithm, we introduce a ranking mechanism through which we rank the links in PIPs based on their QoS constraints and allocate the resources based on the ranking. For different applications offering different services to the end-users the QoS constraints are different. Based on the required service in the VNet request the algorithm perform the mapping to ensure the QoS requirement are satisfied and each VNet request benefits from whatever QoS attribute it requires the most. For example VoIP which represents voice over chat requires minimum delay while web which refers to the general web traffic requires low bandwidth, small delay and for online streaming services, we require a high bandwidth. Based on these assumptions the algorithm performs the ranking paths based on QoS attributes.

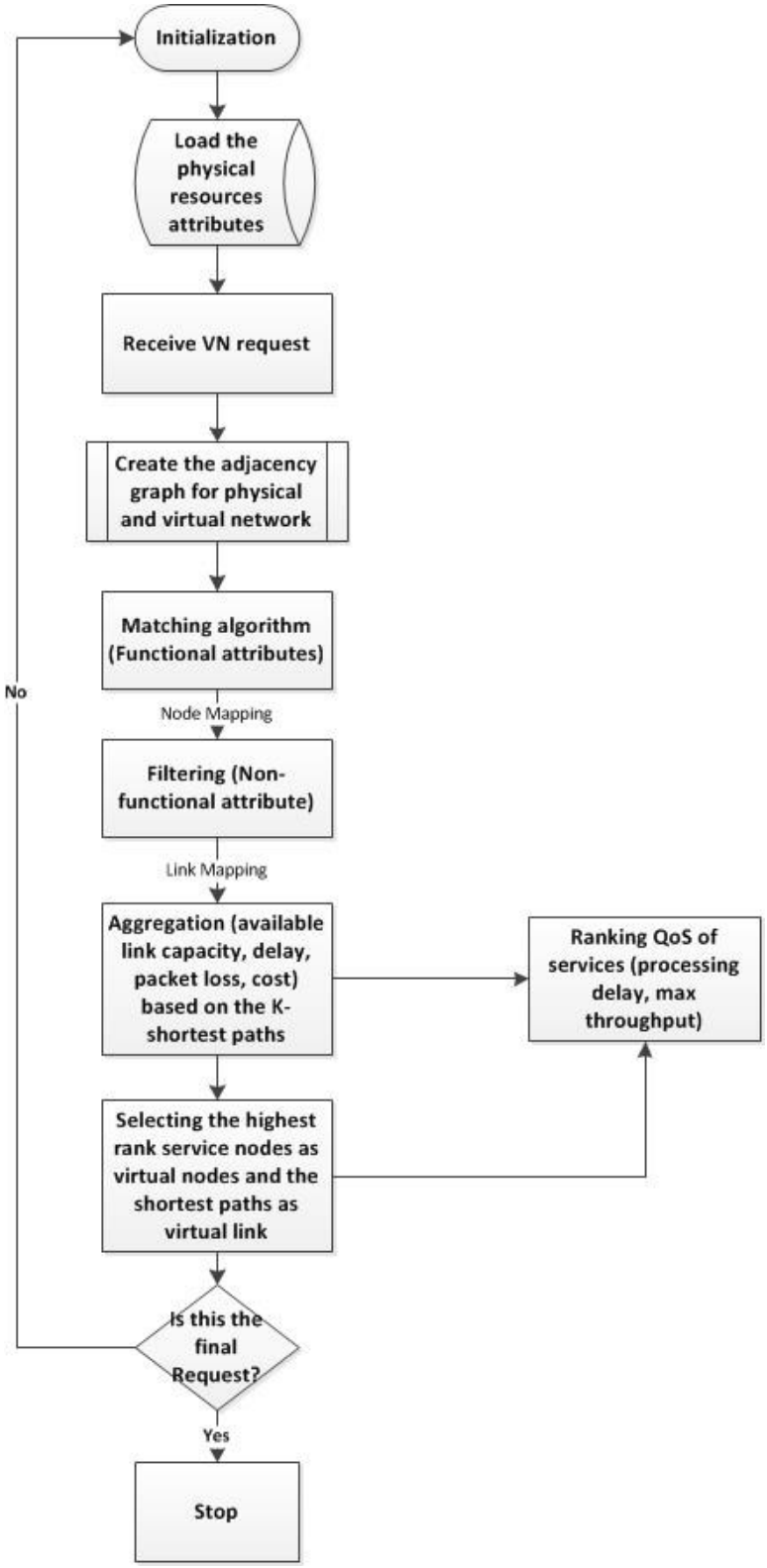


Figure 6 Flowchart of the QoS-Based Approach for Resource Selection in Virtual Networks

4.4 Modeling physical and virtual network

The algorithm in this work models VNet requests that are already known in advance. All scenarios assume a complete knowledge of all virtual network resources that should be mapped into substrate resources. Each request specifies the topology of the virtual network, along with functional attributes and non-functional attributes and the QoS requirements.

4.4.1 Substrate network model

The substrate network is represented by an undirected graph denoted by $G^s(N^s, L^s, C_s^n, C_s^l)$ in which the substrate nodes (N^s) are modeled as the vertices of the graph and the substrate links (L^s) as the edges. Each node in the substrate graph has attributes denoted as C_s^n as follows: CPU $C(n^s)$ which denotes the available capacity of the physical node ns, location loc (n^s) which is a coordinate of the physical nodes. The link attributes denoted as C_s^l are as following: bandwidth b (l^s) which denotes the available bandwidth capacity of the physical link l^s, delay d (l^s) and packet loss pl (l^s) denote the delay and packet loss percentage of the physical link respectively. Consider P^s a set of substrate paths in the substrate network G^s, the available bandwidth capacity AC (P) related to a substrate path $P \in P^S$ between two substrate nodes is calculated as the minimum residual capacity of all the links in the substrate path:

$$AC(l^S) = b(l^s) - \sum_{l^v | L^s} b(l^v) \quad (1)$$

$$AC(P) = \min AC(l^S), \quad l^S \in P \quad (2)$$

4.4.2 Virtual network model

The virtual network is also represented as $G^v(N^V, L^V, C_v^n, C_v^l)$ in which virtual nodes are presented as N^V and virtual links as L^V .Each node in the VNet requests is associated with certain characteristics as follows: CPU C (n^V), Bandwidth b (l^V), the maximum delay allowed in the virtual links d (l^V) and location of each node loc (n^V). All the node attributes and link attributes in the virtual network request are denoted as C_v^n, C_v^l respectively.

Location in our model can be considered as specific policy for VIPs that may require the physical node to be in specified area. For this element we have taken into consideration the location of physical nodes and the desired location of the virtual nodes. We define a *Distance* denoted by D^V in the request which specifies how far a virtual node can be located from the

given location. Each virtual link $l^v \in L^V$, is associated with bandwidth b (l^V), delay d (l^V) and packet loss pl (l^V). The adjacency matrix 3 and 4 represent the connectivity of the physical network and the virtual network. The rows and columns of the matrix represent the vertices of the graph with 1 or 0 depending on whether the related nodes are adjacent or not.

(3) AM^s_{ij} 1, the physical node i is connected to j

0, else

(4) AM^v_{mn} 1, the virtual node m is connected to n

0, else

4.5 Virtual Network Embedding Model

Virtual network embedding is defined as a mapping denoting by *MAP* from G^V to a subset of G^S such that the constraints in G^V are covered. *MAP*: $(G^v, C_v^n, C_v^l) \rightarrow (G^s, C_s^n, C_s^l)$

The mapping consists of node mapping and link mapping stage as following:

Node mapping: MAP^N: $(N^v, C_v^n) \rightarrow (N^s, C_s^n)$

Link mapping: MAP^L: $(L^v, C_v^l) \rightarrow (L^s, C_s^l)$

The left side of Fig.7 shows an example of the VNet request and the right side of it shows the VNet mapping solution. As the picture shows the nodes of the request are mapped to substrate nodes 1, 3 and 5. The virtual links (a, b) are mapped to the physical paths (1, 5) and (a,c) are mapped to physical path (1, 2, 3) and (b, c) are mapped to (3, 4, 5) with respect to CPU and bandwidth.

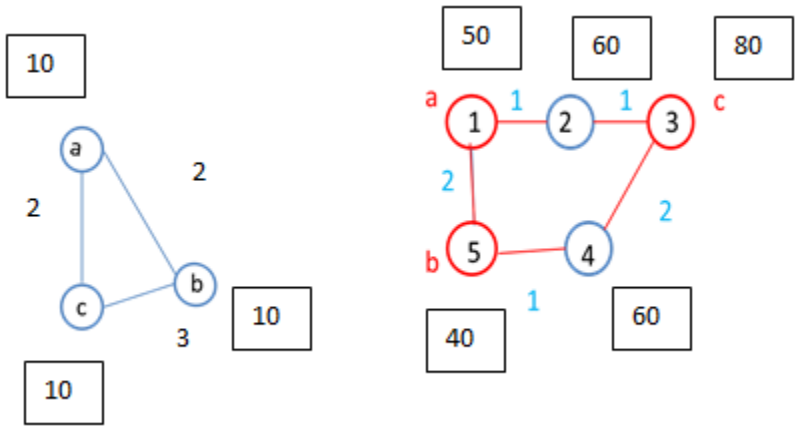


Figure 7. Example of VNet Mapping

4.6 Objective Description

The mapping algorithm may have multiple objectives. In this thesis, our objective is to find a set of substrate nodes subject to the requirements in the VNet request so that the total acceptance ratio and the revenue are maximized and the cost of the mapping is minimized. We have also taken the runtime into account and it is used in evaluation chapter (Chapter 5) to estimate the algorithm's efficiency.

4.7 End-to-end Virtual Resources' Selection and Mapping Algorithm

Our proposed mapping algorithm (30) consists of five different steps and in each step certain criteria have been considered. The five steps are as following: Matching, Filtering, Aggregation, Ranking and Selection that will be explained in detail in next section. Generally applications that the quality of the end-to-end services play an important role can benefit from the proposed algorithm such as customizable networks, layer 3 VPNs and VoIP.

4.7.1 Matching

Candidate matching is finding a set of available substrate resources based on functional attributes that comply with the requirements specified by the VNet request.

In a VNet request the virtual links and nodes topology and their requirements are given specifically**.** A virtual node could possibly be a virtual host or a virtual router/switch. A virtual link can map over multiple physical links forming a physical path. A virtual link is usually mapped to multiple physical paths so it can satisfy some of the constraints of the virtual link such as bandwidth constraints that cannot be satisfied using a single path. (31)

In this phase only functional attributes which include typically the static parameters like operating system type, network stack, node/link type and virtualization platform are considered. The non-functional attributes which include real-time parameters (e.g. processing power, memory, and actual capacity) will be considered in the next steps of embedding.

In the first step of the algorithm, we partition and organize the resource descriptions. We have divided the resource matching part into two sections, *substrate resource adder* and *substrate resource finder*. The substrate resource adder builds a tree based on the resource description by the PIP in such an order so it is known that which specification is being dealt in each layer of the tree. We can easily add or remove resources whenever the PIP add or remove its resources. First we need to import the PIP resource specifications in the framework and then search through the data to select the most appropriate ones.

Upon receiving a request the substrate resource finder search the tree based on the requested attributes. Then in each children of the tree it finds the nodes with the required specifications and stores the potential candidates. At the end, the intersection of all the potential candidates with required specifications is our output for this phase. The VIP uses this framework to discover and match available resources using the VNet request.

A. The Matching Algorithm (substrate resource adder)

1. Create the structure array with the specified resources and fill it according to their functional attribute as shown in Figure 8.
2. At any time that a resource has been added to the service and registry we have to add it so it will be added among the resources for the coming virtual network request.

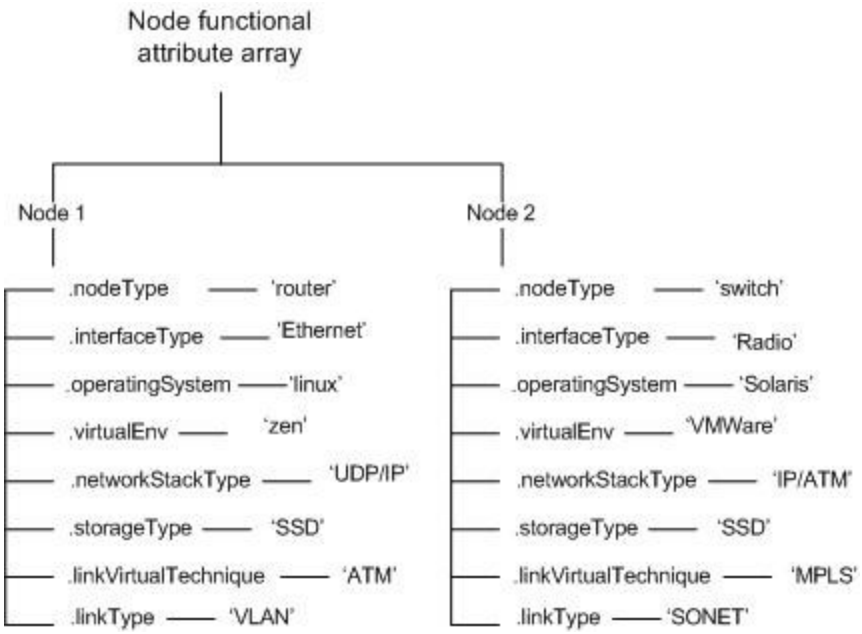


Figure 8 Substrate Resource Adder in Matching Step

B. The Matching Algorithm (virtual resource finder)

1. Upon receiving the request, *Virtual Resource Finder* start searching through the tree and for each specification it stores the candidate resources.
2. After searching the whole attributes, the intersection of all the candidate resources are the resources that have all the specifications that has been requested.
3. The output is all the resources with the requested functional attributes.

For our model we consider the following functional attributes for node. Node type: virtual router, virtual switch, virtual access point, virtual base station, interface type: Ethernet, Optical/Fiber, Radio (Wife, WiMax), ATM, frame relay, operating system: Linux, windows, Solaris, UNIX, virtual environment type: VMware, Zen, KVM, network stack type: TCL/IP, UDP/IP, IP/ATM, IP/Ethernet, storage type: SSD , HDD. The link functional attributes are as follows: link virtual technique: ATM, MPLS, Ethernet 802.1g, VLAN, link type: VLAN, SONET, 802.11.

4.7.2 Filtering

When a virtual network request arrives, the substrate network has to determine whether to accept the request or not. If the request is accepted, then the substrate network selects the suitable resources for the VNet and allocates network resources on the selected substrate nodes and paths. Substrate network resources need to be measured and compared with the VNet requirements.

Generally the physical nodes are used incompletely by some of the virtual requests and it is essential to calculate the remaining capacities of the substrate resources. AC (n^s) denotes the available CPU capacity of the substrate node $n^s \in N^s$

$$AC(n^S) = \boldsymbol{C}(\boldsymbol{n^s}) - \sum_{\boldsymbol{n^v|n^s}} \boldsymbol{c}(\boldsymbol{n^v}) \qquad (3)$$

Where $n^V|n^S$ denoted that virtual node n^v is hosted on the substrate node n^s. The aim of node mapping is to select the node which has more collective bandwidth of outgoing links among multiple similar nodes (in terms of resource availability). The amount of available resource for substrate node n^s is defined by:

$$AR(n^s) = AC(n^S) \sum_{\boldsymbol{l^s \in L(n^s)}} \mathbf{AC}(\boldsymbol{l^s}) \qquad (4)$$

where $L(n^s)$ is the set of all adjacent substrate links of ns, $AC(n^s)$ is the available capacity of ns and $AC(l^s)$ is the available BW resource for the substrate link l^s. Available capacity of a substrate link $l^s \in L^s$ could be derived from equation (5)

$$AC(l^S) = \boldsymbol{b}(\boldsymbol{l^s}) - \sum_{\boldsymbol{l^v|L^s}} \boldsymbol{b}(\boldsymbol{l^v}) \qquad (5)$$

where lv|Ls denoted that virtual link lV is hosted on the substrate link lS. The available bandwidth capacity of a substrate path $P \in P^s$ is

$$AC(P) = \min AC(l^S) \ , l^S \in P \qquad (6)$$

The VNet assignment has been decomposed into two major components node mapping and link mapping that will be explained in the next part.

A. Node mapping

In node mapping each virtual node from the VNet request should map to a different substrate node In other words, it's not possible to map two virtual nodes on the same physical node. The constraints for node mapping are given in the following equations:

$$C(n^V) =< AC(MN(n^V)) \quad (7)$$

$$dis(loc(n^V), loc(MN(n^V))) =< D^v \quad (8)$$

Where dis(x,y) measures the distance between the location of two substrate nodes x and y.

Algorithm 1: Virtual Network Node Mapping:

---

1. *Upon receiving a VNet request* // Read the VNet request attributes and substrate network attribute from database
2. *Extract* C_v^n, C_v^l and D^V *from the VNet request*
3. **While** $N^v(i) \neq 0$ **do** // while there are non-assigned nodes
4. $\forall n^v \in N^v(i)$ **do**
5. j=1
6. **Sort** ($N^s(j)$) *// sort the selected nodes from matching algorithm according to node available resource (4)*
7. **If** $C(n^V) =< AC(C(n^s(j)))$ & $dis(loc(n^V), loc(M_N(n^V))) =< D^V$
8. $\forall n^s \in N^s(j)$
9. K=0
10. **While** $(AC(C(n^s)) >= C(n^V))$
11. K=K+1
12. **End while**
13. $n^v \rightarrow n^s$ // n^v assign to k-th n^s
14. $AC(n^S) = C(nS) - \sum_{nV|nS} c(nV)$
15. $AR(n^s) = AC(n^S) \sum_{lS \in L(nS)} AC(l^s)$ *// 14 and 15 update the available capacity and resource for the node ns*
16. $N^v(i) = N^v(i) - n^v$ *// remove the node nv from list of requested nodes*
17. **Else**
18. j= j+1 *// go to the next node*
19. **go to** *6*
20. **end if**
21. **end while**

4.7.3 Aggregation

The next step is to map an edge ($l^V \in L^V$) on a substrate path (P^S) containing one or more than one links. For link mapping we have used k-shortest paths algorithm for finding suitable edges.

A. Link mapping

In edge mapping each virtual link is mapped to a substrate path (or multiple paths). The substrate paths are those that connect the substrate nodes that have been preselected to host the virtual nodes. The constraint for link mapping is given by the following equation:

$AC(P^s) >= b(l^v)$ (9)

Algorithm 2: Virtual Network Edge Mapping:

---

1. Take the request which has successfully passed the virtual node mapping stage.
2. Create the adjacency cost matrix for the substrate network graph
3. Search the k-shortest paths between the pair of nodes connected by the edge (mapped on nodes by the node mapping algorithm). The output of this algorithm is
4. a. [shortestpaths]: the list of K shortest paths
5. b. [totalcost]: cost of the K shortest paths
6. After obtaining the k shortest paths, the links of these paths along with their attributes are extracted from substrate network.
7. For all the links of the paths calculate the available BW and delay and packet loss
8. If AC(Pi S) >= $b(l^v)$ & $\sum$ d(li S) <= $d(l^v)$ & $\sum$pl (li S) <=pl (l^v)
9. Select the ith shortest path which satisfies the edge demand (according to the edge mapping function, SPM).
10. Call the Rank Function which ranks the paths based on end-to-end service level attribute (packet loss, end-to-end delay) and select the paths with the highest ranks
11. If the edge request is satisfied, update available link capacities of the selected path according to (6) and GOTO 12
12. If this was the last request, then, stop, else, call "node mapping algorithm".

In Aggregation, edge mapping solution assigns all edges to the substrate paths and each virtual link can be mapped on one or more physical paths. Edge mapping starts by finding edge disjoint k-shortest path for each edge of the VNet graph. Afterwards, resources as well as QoS attributes of links (delay, packet loss) on each of these paths are calculated and the paths having sufficient resources (satisfying the QoS according to the service in demand) are selected. At this point the algorithm calls the ranking algorithm and finally it selects the path which has the highest rank. Ranking is based on end-to-end service level attributes (max throughput, end-to-end delay) if

there are sufficient paths resources to satisfy the request of all the edges then the VNet is completely mapped and the request is satisfied.

4.7.4 Ranking:

In our work, we don't monitor the network to measure the network parameters. We assume that the application providing QoS will require a performance monitoring and that the data acquired by such monitoring system could be used in simulation of the algorithm.

We propose the ranking for end-to-end QoS attributes for our algorithm. Our purpose is to select paths with high quality; therefore, we assign related weight to the QoS parameters according to the requested service. After selecting the preliminary nodes and links we calculate the ranks for the links in instantiated virtual network and choose the ones with the highest rank to proceed with the embedding. We consider max throughput, end-to-end delay as the parameters in our simulation but here we give a general equation for the ranking concept based on (32).

An application with 'n' QoS constraints is represented as $P_1, P_2, \ldots, P_n$, and each is assigned to a specific weight, $w_1, w_2, .., w_n$ according to the importance of the constraints. Note that the addition of all the weights should be equal to 1. ($w_1+w_2+\ldots+w_n=1$)

By assigning weights to the end-to-end attributes, the intention is that each attributes related to the constraints contributes to the quality of the path in proportion to its weight. For example VoIP and online gaming require low end-to-end delay while bulk file transfer require high throughput.

According to the constraints (P_i) and the requested value (R_i) the equation 10 shows the quality of the path (Q).

$$Q = \sum_{i=1}^{i=n} \frac{Pi}{Ri} w_i \qquad (10)$$

Q represents the ratio of the QoS attributes received by an application over the QoS attributes requested. The purpose of the ranking stage is to ensure that the algorithm favors high values of the metrics that are more important for the requested application. For example minimum end-to-end delay is the most important QoS attribute for VoIP applications and should acquire the highest weight among other attributes.

4.8 End-to-end selection and mapping algorithm

Given the detailed description of each stage of the algorithm, here are the main steps of the algorithm:

Algorithm 3: End-to-end selection and mapping algorithm:

1. Match all the functional attributes of the VNet nodes and links and specify the candidates.
2. If no candidate is selected reject the request, go to step11
3. Filter all the nodes and links from the substrate topology that meet the application non-functional constraints
4. If there is no node reject the request and go to step11
5. Store the nodes and links of substrate nodes that fulfill the requirements
6. From the VNet request, select a node to perform the embedding
7. Find the shortest edges using k-shortest path among nodes discovered in step5 and compute the path characteristics using the aggregate functions
8. If no link exists reject the request and go to step 11
9. Compute the quality of the k-shortest path using equation (10) and based on the quality of the direct routes rank all the existing paths
10. Select the nodes with highest degree (if multiple nodes exist) and the path with highest rank in terms of QoS attributes.
11. If there is more request, go to step1, otherwise go to step 12
12. If no request left, stop and return the selected nodes and links.

4.9 Example Scenario

We model the substrate network with 10 nodes with mesh topology (45 links) as shown in Figure 9 Substrate network. Numbers in squares represent the CPU, and numbers along the links represent the bandwidth. VNet request contains 5 nodes with 6 links along with other attributes (CPU, location, BW, Packet loss, delay) as in 10.

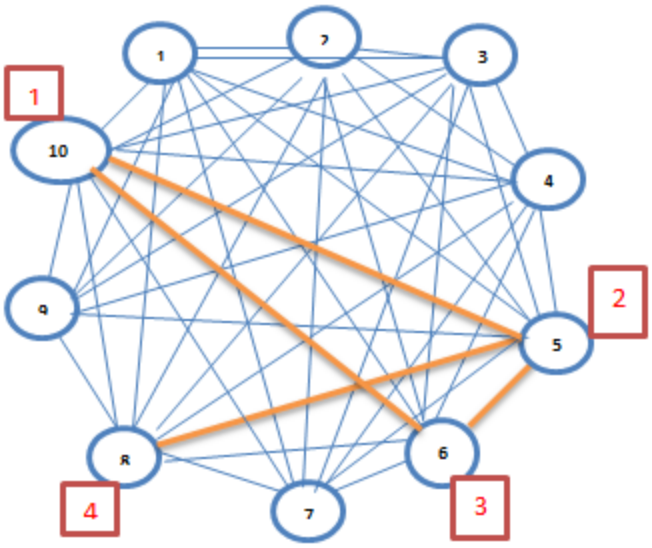


Figure 9 Substrate network

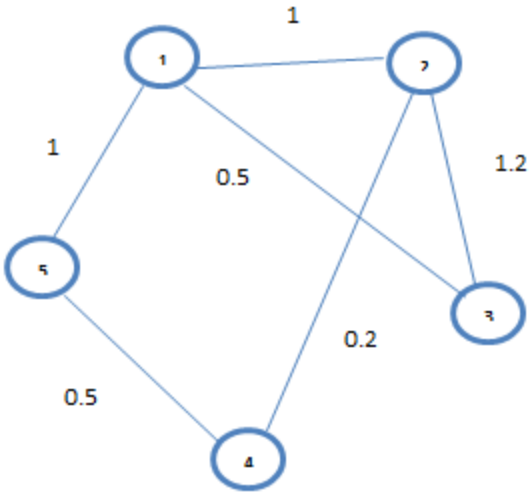


Figure 10 Virtual network request

After executing the algorithm, first functional attributes will be checked and selected candidates go to the next step witch is filtering. In filtering, according to the available capacity of the nodes and available resources (the nodes that are connected to more powerful links are considered with

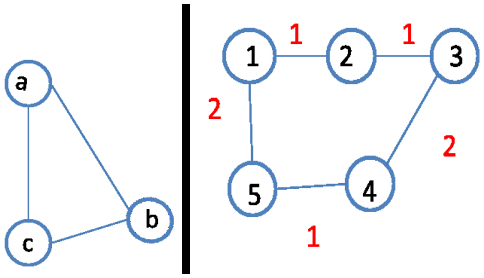
higher available resources) the nodes will be filtered. At first, the available capacity of the substrate nodes is equal to their actual CPU. After each allocation the available capacity of the nodes will be updated and the CPU amount that has been allocated will be deducted from the available capacity of the node in the repository. After node selection, the nodes in the VNet request will be embedded with the selected nodes from the substrate network. In this step, the candidate nodes to host the virtual nodes are selected.

In the next step which is aggregation, the paths are selected based on k-shortest path and then other attributes, such as BW, delay and packet loss, of the links will be checked and the paths with suitable attributes will be selected. In the ranking step, the paths are given ranks based on the end-to-end QoS attribute (end-to-end delay, max throughput) according to the QoS constraints in the requested service the links with the higher ranks will be selected.

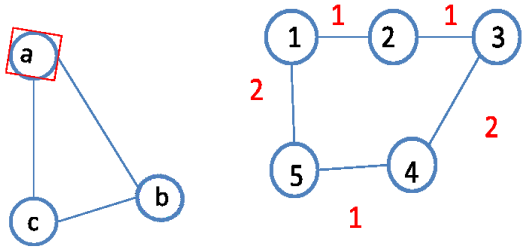
Figure 11 depicts the mapping process for a sample substrate network and virtual network. The substrate network consists of five nodes and five links. The request consists of three nodes and three links. As the request arrives, the algorithm will identify the nodes and start searching for the candidate node (matching algorithm). As mentioned earlier, only the functional attribute of the nodes are considered in this step. Matching Algorithm's output is all the candidate nodes considering only functional attributes. Then in the filtering algorithm, those nodes that possess enough capacity and higher bandwidth links attached to them, considering the location of the node, are selected for node mapping.

When all nodes have been selected, it's time to allocate the links between them. The aggregation algorithm performs this job by using the k-shortest path and other calculations regarding the available link capacity and QoS attributes such as delay and packet loss. If there is no substrate path that satisfies the requirement, the path splitting takes place. According to path splitting it will split the path into small bandwidth and then the mapping is performed. For example, if $bw(l^s)$ is the bandwidth of virtual link it will split into $bw(l^s) = bw(l^s)/2$. After splitting the bandwidth, virtual link will be mapped onto substrate path which satisfy the requirement. This process of link mapping continues until no virtual link is left.

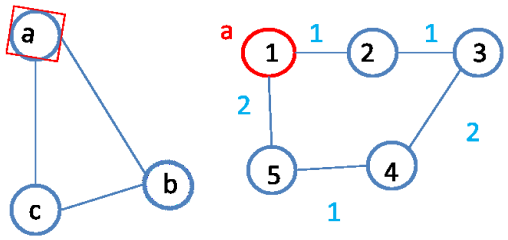
Finally, algorithm has a ranking function which ranks the links according to the service the VNet user has asked for. For example, if the user needs to establish a voice service the delay of the links became more important while for a mail service the packet loss rate became more crucial. The algorithm ranks the links based on the requested service and mapped the links with higher ranks.



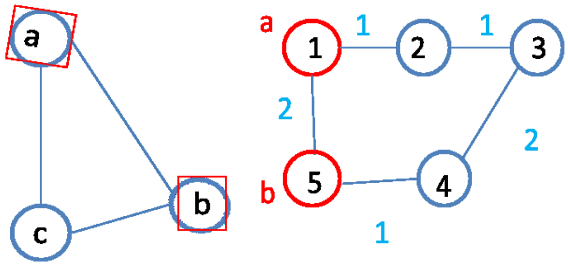
Virtual and substrate Network



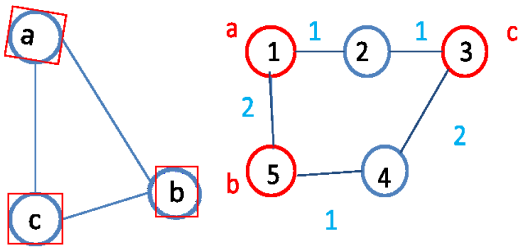
b. Request for node "a" received



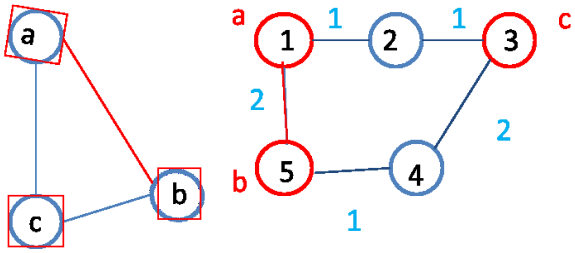
c. Request for node "a'' has been mapped to node 1.



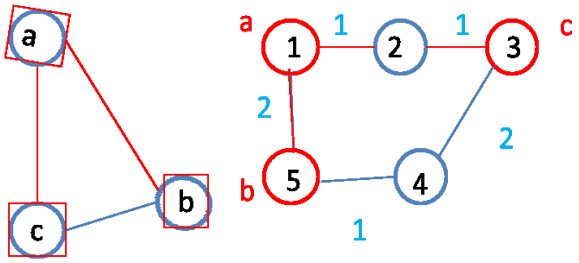
d. Request for node "b" arrived and mapped to node "5"



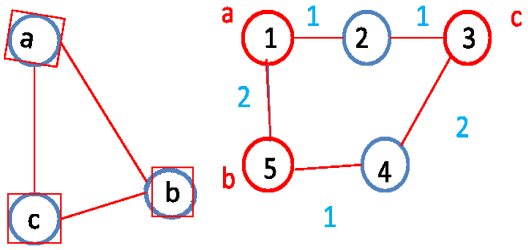
e. Request for node "c" arrived and mapped to node "3"



f. Now the links between the nodes are selected (a,b) using k-shortest path (1,5)



g. The second link from node a is (a,c) and it mapped to links(1,2) and (2,3) because it was shorter than (1,5)(5,4)(4,3) according to K-shortest paths



h. The third link in the request is (b,c) which is mapped to (5,4) and (4,3).

Figure 11 Example of mapping process

4.10 Improving the performance using path splitting

For improving the performance of the algorithm, we have adapted path splitting which helps to make a better use of the resources by allowing the substrate network to accept more VNet requests (12). By path splitting, the bandwidth of the path splits into multiple small bandwidths to satisfy the resource constraints. Fig 12 shows an example of path splitting (12). On the left side of the picture the virtual network request consisting of three nodes and two links arrives and the virtual nodes a, b and c, are mapped to physical nodes A, E and F. The links (a, b) and (a, c) are also mapped to the substrate paths (A, D, E) and (A, D, F). As seen on the right side of the picture, the next request arrives consisting two nodes d and e and a single link. The requested virtual link is 30 units and none of the existing path could satisfy the new requested bandwidth

and without using the path splitting it would get rejected. However, using path splitting the new request could be mapped to nodes D and E and the substrate network allocate 20 units of bandwidth on the path (D, E) and 10 on the path (D, G, H, I, E).

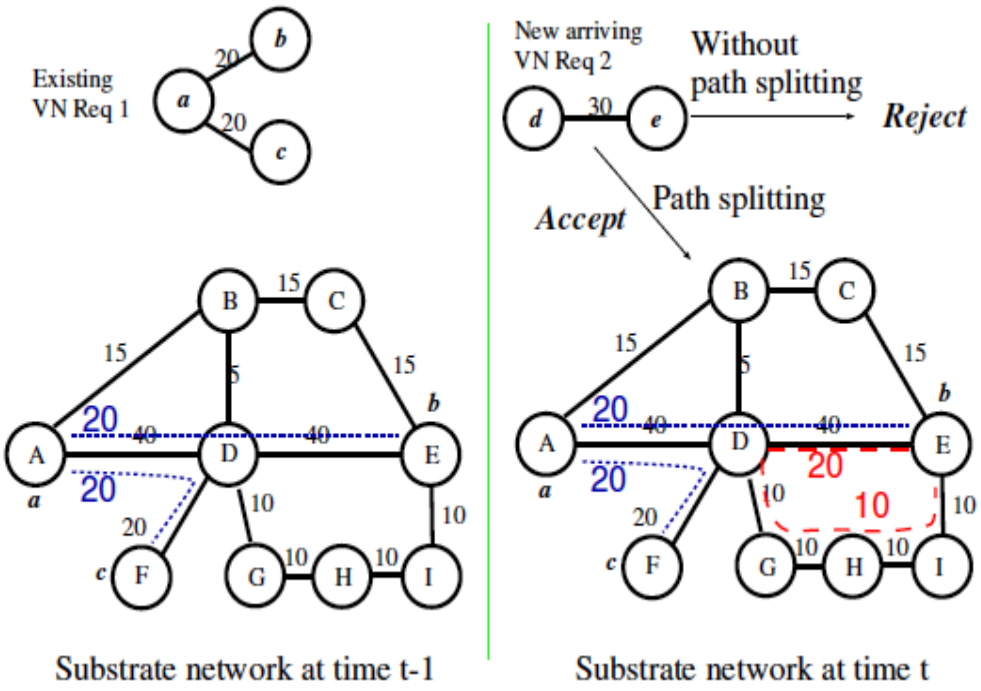


Figure 12 path splitting example (12)

With path splitting, we could get advantage of better resource utilization by connecting the small pieces of available bandwidth and let the substrate network to accept more VNet requests. It will also lead to better load balancing and reliability. A virtual link l with some capacity constraint, for example, Cl, is mapped into multiple paths in the substrate network such that the sum of reserved end-to-end bandwidth along the multiple paths is equal to Cl. The traffic is divided over the substrate paths as a splitting ratio, such as a ratio of 2:1 in the example in Figure 12. Under flexible splitting over multiple paths, the link-embedding problem can be reduced to the Multi-commodity Flow Problem (MFP) (33). In fact, even having just two paths can significantly reduce the maximum load on a network, compared to solutions that limit the traffic flow to a single path [24, 19]. Path splitting has another advantage and it's the existing of multiple paths that enable faster recovery from network failures. For example, if there is a failure in link or node, the network can easily switch the affected traffic to other paths by changing the splitting

ratios. In compare, in a single path setting, a failure requires establishing a new end-to-end path, leading to a more severe service disruption.

4.10.1 End-to-end algorithm using path splitting

Given the description for path splitting, in the link mapping stage, the k shortest paths are found by using the k-shortest paths algorithm using search on values of k until a path is found on the substrate network that satisfies the bandwidth requirement of the virtual link. If the substrate network allows the path splitting, we will use the multi-commodity flow algorithm to embed virtual links to the substrate links. It provides a multi-path routing solution for each virtual link using optimal linear programming. Algorithm 4 explains the end-to-end algorithm using path splitting in details.

Algorithm4: End-to-end algorithm using path splitting:

1. Match all the functional attributes of the VNet nodes and links and specify the candidates.
2. If no candidate is selected reject the request, go to step11
3. Filter all the nodes and links from the substrate topology that meet the application non-functional constraints
4. If there is no node, reject the request and go to step11
5. Store the nodes and links of substrate nodes that fulfill the requirements
6. From the VNet request, select a node to perform the embedding
7. Find the shortest edges using k-shortest path among nodes discovered in step5 and compute the path characteristics using the aggregate functions
8. If no link exists, then compute $bw(l^s)= bw(l^s)/2$ and repeat step 7
9. If no link exists reject the request and go to step 12
10. Compute the quality of the k-shortest path using equation (10) and based on the quality of the direct routes rank all the existing paths
11. Select the nodes with highest degree (if multiple nodes exist) and the path with highest rank in terms of QoS attributes.
12. If there is more request, go to step1, otherwise go to step 13
13. If no request left, stop and return the selected nodes and links.

4.11 Summary

In this chapter we present QoS-based resource selection approach for virtual networks. We explained node mapping and link mapping in details. For node mapping, we consider functional attributes and we perform a matching step based on that. In filtering step, the non-functional attributes have been taken into consideration and for link mapping bandwidth, delay and packet loss have been calculated. For link mapping step, k shortest path has been used and if no link is found we make use of path splitting in order to map the links more efficiently. In order to offer a better end-to-end service, we consider a ranking schema in which the paths are being ranked based on their QoS attributes and the service they offer and at the end of the selection process the links with highest ranks are selected. At the end, we proposed another algorithm using path splitting which makes the network more reliable and offers better load balancing. In fact, having two paths can significantly reduce the maximum load on a network compared to solutions that bound the traffic to a single path.

Chapter 5

Performance Evaluation

5.1 Overview

In the previous chapter, we have presented the QoS based resource selection algorithm along with detailed explanations of each step. The implemented prototype provides a mechanism for VIPs to select the best resources from PIPs and therefore build their virtual network. We have also presented an example of the whole mapping process. In this chapter, we evaluate the performance of the proposed algorithm using the Alevin framework (34) by introducing the related metrics and conducting simulation and we discuss the obtained results.

We performed extensive experiments to evaluate the performance of the algorithms in terms of revenue, runtime and acceptance ratio. Each of the existing VNet embedding algorithms considers specific aspects in their work but the ultimate goal is to acquire higher acceptance ration and increase the revenue. From the presented evaluation results, we show that QoS-bases resource discovery algorithm with path splitting improves the overall performance in comparison to the algorithm without using path splitting.

By conducting the experiments, we do not claim that we have the best implementation. However, since the response time of the algorithm depends on several factors (implementation platform, type of request, etc.) the purpose of these experiments is to demonstrate that the implemented algorithm is functional.

5.2 Simulation Setup

The algorithm has been developed using Java and Alevin framework. Alevin is an open source framework for the evaluation of various virtual network embedding algorithms. This simulation framework is built in order for researchers to add new VNE algorithms and it has a strong GUI. The workflow of the Alevin simulation framework is presented in Figure 13. The generated random topology from GTITM has been imported as the inputs. We have used a Linux operating

system in a virtual machine with 4.00 GB of RAM and quad core 3.1- GHz processor for simulations. For both substrate and virtual resource description (input scenarios) we opted to use XML. A sample of XML used in the algorithm is given in the appendix B.

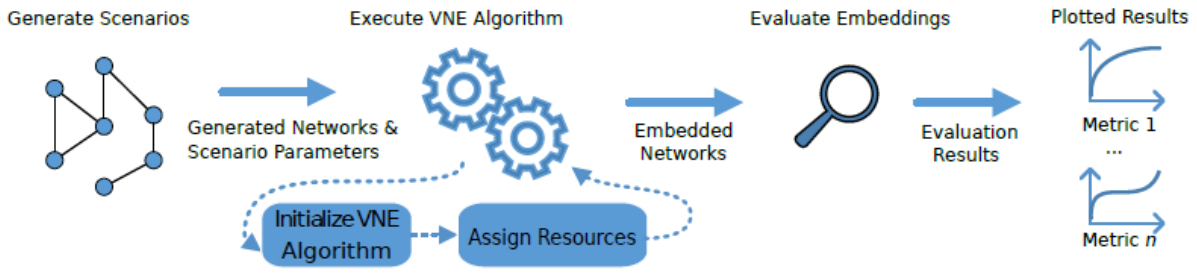


Figure 13 Workflow of the Alevin Simulation Framework (34)

5.3 Topology Modeling

One of the challenges in networks is to determine how to generate topologies for the virtual network requests and substrate network. Although networks are hard to model, the goal is to represent networks using graphs such that the key parameters are simulated closely with those of real networks. The graphs demonstrate all possible sources and destinations for data, usually presuming an endpoint is a source or destination at each node of the modeled graph. The edge of the graph represents the possible path from a node (source) to another node (destination).

For simulation purpose we choose GT-ITM (Georgia Tech Internetwork Topology Models) (35) simulator and NS2 (Network Simulator2) to generate the network topology. GT-ITM has been commonly used for research purposes and it uses TCL scripting. The substrate network is configured to have 10 nodes for a small network, 50 for a medium sized network and 100 for a large network. The CPU of the nodes and Bandwidths of the links is uniformly distributed (between 50 to 100) with the simulator tools.

The virtual network requests are considered in three sizes; small with 5 nodes, medium 10nodes and large 20 nodes. The CPU of all requests generated using the uniform distribution (from 20% to 80% of the substrate network) and link bandwidth requested are generated using the uniform distribution (from 20% to 80% of the substrate network). The settings of the substrate network

are shown in table 2 and the virtual network in shown in table 3. We consider an application with QoS requirements minimizing packet loss rate and end-to-end delay as the parameters in our simulation. Packet loss is the total number of packets sent minus the number of packets received. End-to-end delay is the average time taken by a data packet to reach the destination Σ (arrive time – send time) / Σ Number of connections.

Number of substrate nodes	10	50	100
Number of substrate links	45	200	500
Substrate CPU, link BW	A uniform distribution [50 100]	A uniform distribution [50 100]	A uniform distribution [50 100]

Table 2. Substrate network settings

Number of virtual nodes	5	10	20
Virtual node CPU and link BW	Uniform distribution from 20% to 80% of the substrate network	Uniform distribution from 20% to 80% of the substrate network	Uniform distribution from 20% to 80% of the substrate network
Arrival rate of virtual request	5 per 100 time unit	5 per 100 time unit	5 per 100 time unit
Duration time of virtual request	Average of 1000 time unit	Average of 1000 time unit	Average of 1000 time unit

Table 3. Virtual network settings

GT-ITM uses three different graphs to generate the network: regular, flat random and hierarchy. In the flat random topology, a group of nodes distribute randomly with specific edge

probabilities such as Waxman model. In the hierarchy model, the topology is extended by connecting the smaller groups together to form a larger network. In this work we used transit-stub method which is a type of hierarchy model. In this model, each routing domain is divided into transit or stub domain. Figure 14 shows a sample graph with 10 nodes and fully connected links created by GT-ITM. In order to visualize the graph create by GT-ITM we had to convert the output file (.gb) to NS2 format (.tcl) and display.

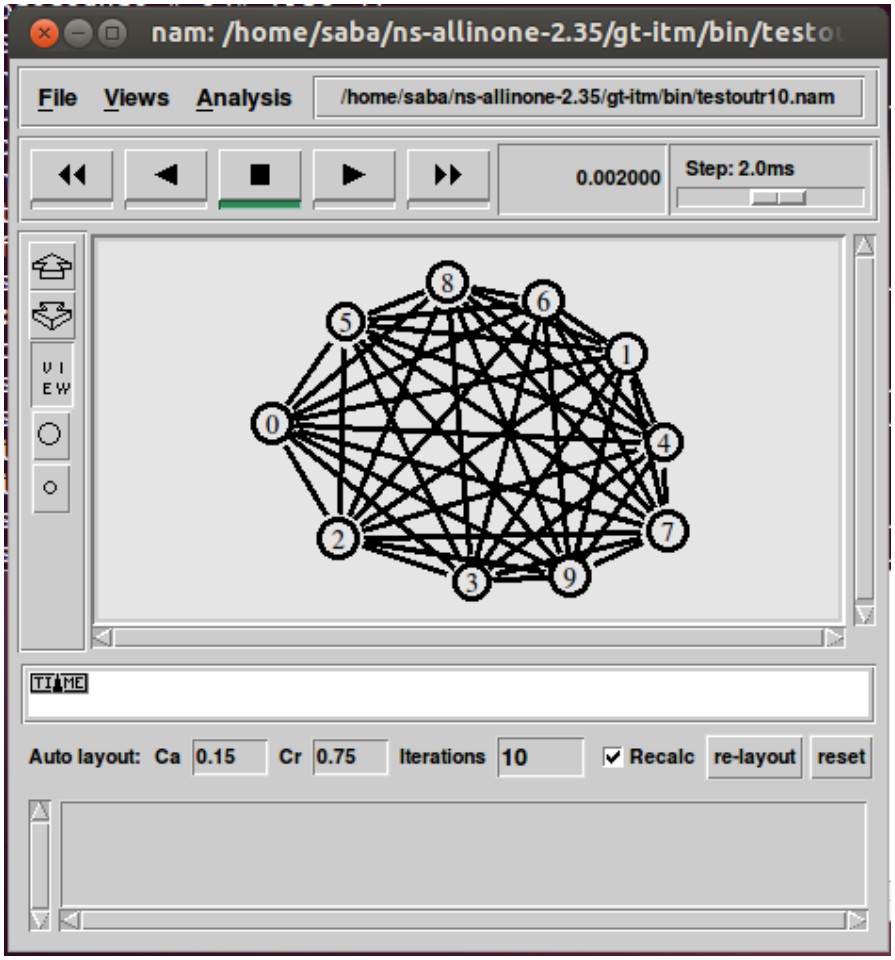


Figure 14. Sample graph with 10 nodes and fully connected links

5.4 Performance Analysis

After the creation of the network topologies, resources get assigned to the substrate nodes and links. For the evaluation of our algorithm, we have compared our two proposed algorithms in terms of acceptance ration, revenue and run-time and presented the results.

5.4.1 Acceptance Ratio:

Acceptance ratio is percentage of total VNet requests that have been accepted by the algorithm in a specific time. Every VNet request has its own resource requirements both in nodes and links and they have different topologies and added-value services but the ultimate goal of mapping algorithm is to maximize the number of the accepting VNet requests. The VNet request acceptance ratio calculation is shown in (11) and it states the overall performance of the embedding algorithm. It is calculated by the number of accepted VNet requests, L', over the number of all VNet requests, L.

$$AR^{VN} = L'/L \qquad (11)$$

5.4.2 Revenue:

The objective of our work is to maximize the revenue and minimize the cost of embedding in the long time. Revenue is important because it represents the profit of the embedding and it refers to the amount of money the customer pay for their network. As authors in (10) described, the revenue of a VNet request is calculated as equation (12).

$$R(G^V) = \sum_{l^V \in L^V} BW(l^V) + \sum_{n^V \in N^V} CPU(n^V) \qquad (12)$$

5.4.3 Cost:

The other factor important in VNet embedding is cost for a VNet request and it is calculated as equation (13).

$$C(G^V) = \sum_{l^S \in L^S} \sum_{l^V \in L^V} BW(f_{ls,}^{lv} lv) + \sum_{n^V \in N^V} CPU(n^V) \qquad (13)$$

Where BW(l^V) denotes the bandwidth of the virtual links, CPU(n^V) denotes the CPU capacity of the virtual nodes and BW(f_{ls}^{lv}) denotes the total amount of bandwidth allocated on physical link l^S for virtual link l^V. Cost of a mapping is the amount of substrate resources allocated to the virtual network request.

For better understanding the calculation of the revenue and cost we present an example to demonstrate the evaluation of this metric. As shown in the Figure 15, the left graph is the VNet request and the right one is the substrate network. After the embedding of the VNet in the substrate, the revenue gained by the virtual network is R= (10+12+14) + (9+8+10)= 63 and the cost is C= (10+12+14) + (8*2 +9 + 10)= 71

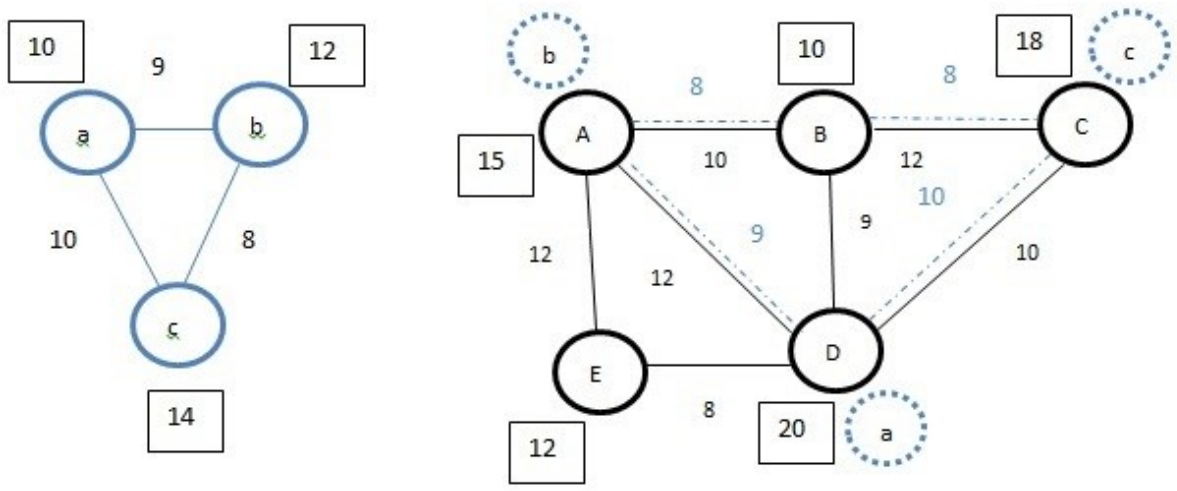


Figure 15 Example for calculating cost of the network

5.5 Performance Evaluation of the Algorithm

The Alevin framework supports the input scenarios as XML files into the VNE algorithm; therefore, it is easy to import different XML files into one VNE algorithm and compare the results. We have executed the algorithm using the sample XML file in appendix B and presented the results of the algorithm in Figure 16.

As previously described, the algorithm solves the VNE problem into two main stages; virtual node mapping which is performed in the first place and virtual link mapping which is performed after node mapping. In this way, the node and link mapping stages can be implemented independently of each other and the advantage is that the node and link mapping of different algorithms can be combined which might results in better performance. For our evaluation purpose, we implemented the node mapping based on algorithm1 and for link mapping we take advantage of both k shortest path and path splitting and we will compare the obtained results later in this chapter. In edge mapping using k shortest path, virtual link mapping is implemented by connecting each pair of mapped virtual nodes by shortest path in the substrate network calculated using the Dijkstra algorithm. In case of not finding a path to satisfy the request, the bandwidth is divided by two and then the algorithm attempts to accommodate the links with smaller bandwidth onto the substrate network. Path splitting continues until a path is found to satisfy the request.

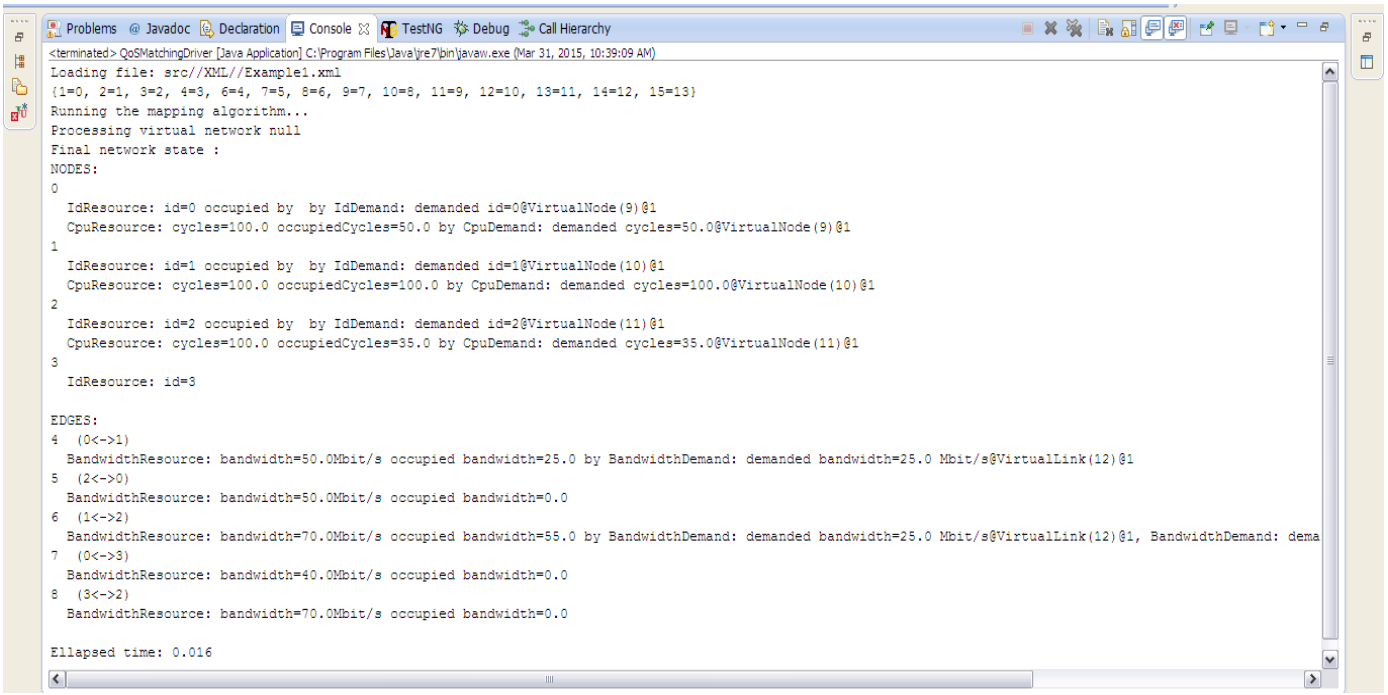


Figure 16 Sample output of the algorithm using K shortest path

5.6 Algorithm Evaluation

Proposed VNet embedding algorithm increases the overall efficiency of VNet embedding process by utilizing static attributes of the network resources at the initial stage of embedding to filter requests before forwarding the VNet request to the selection phase. In the aggregation part, we compare the values of dynamic attributes for each resource which was already selected by matching process. Based on static attribute values, we filter PIP networks in order to shortlist a set of PIPs which satisfy basic embedding requirements. Then we forward the VNet resource requirement specifications to aggregation to select the best candidate based on dynamic attributes and the shortest path among the selected nodes from PIP. The main functions in our algorithms are as follows:

- Retrieve information from the database
- Match the nodes with similar functional attributes
- Aggregate resource information to build the substrate and virtual network topologies and manage static and dynamic resource database
- Ranking the selected links and selecting the nodes and links that satisfy the request, and
- Provide up-to-date information to the repositories about virtual networks and information about the substrate and virtual resources.

The substrate and virtual resources are described in XML format in order to make the modifications easy. In VNE problem it is important to consider the dynamic attributes which play a vital role during the virtual resource embedding process. The cost of accurate monitoring of these dynamic variables is high and there is always a tradeoff between accuracy and cost while monitoring the required network parameters for resource discovery. As some other works, here we assume complete availability of this information.

For achieving the higher efficiency, we introduce the filtering phase to reject the VNet requests that cannot be satisfied while in most existing solutions this rejection is done in selection stage which is not efficient in terms of processing power and results. Initial filtering of VNet requests allow VIP to promptly inform the VNet user that certain requirements cannot be provided with the available resources of PIPs who are currently in contract with the VIP.

For the purpose of evaluation, we compare our two proposed algorithms. First, the QoS-base resource selection algorithm without path-splitting and QoS-base resource selection algorithm using path splitting as shown in Table 4.

Table 4 Compared Algorithms

ALGOROITHM	DESCRIPTION
1. A QoS-based Resource Selection Approach for Virtual Networks	VNet embedding algorithms that map multiple VNet requests with functional and non-functional attributes using K-shortest path
2. A QoS-based Resource Selection Approach for Virtual Networks using path splitting	VNet embedding algorithms that map multiple VNet requests with functional and non-functional attributes using path splitting

5.6 Observations

We plot the performance metrics based on time for both QoS base approach for resource selection for virtual network without path-splitting and QoS approach for resource selection in virtual networks using path splitting to illustrate how each of these algorithms perform. The key observation is as following.

1. Run-time is the average time of processing a VNet request using the mapping algorithm. While going through the aggregation the links among selected node have been found and selected to make the requested virtual network. For calculating the average run-time we measure the total running time for all VNet request and dividing that to the number of VNet requests. As shown in Figure 17, the average run time for the algorithm using path splitting is slightly larger than the one using k shortest path which was predictable. The reason is that algorithm using path splitting is more time consuming in terms of calculations since the bandwidth is recalculated (divided by a ratio) for the link mapping purpose.

2. As shown in Figure 18, QoS-based resource selection approach using path splitting leads to higher revenue. Using path splitting in algorithm, leads to the higher revenue in comparison with the algorithm without using path splitting since path splitting algorithm accepts and embeds more requests and therefore it will lead to larger revenue.
3. The other metric we used for performance evaluation is acceptance ratio. Experiments in Figure 19 show that the algorithm using path splitting leads to higher acceptance ratio. The reason is that in the algorithm using path splitting, the VNet request rejection is less comparing to the other algorithm. Algorithm using path splitting is capable of embedding the links that were rejected in other algorithm and therefore the algorithm embeds more VNet requests and the revenue is higher compare to the algorithm with using path splitting.

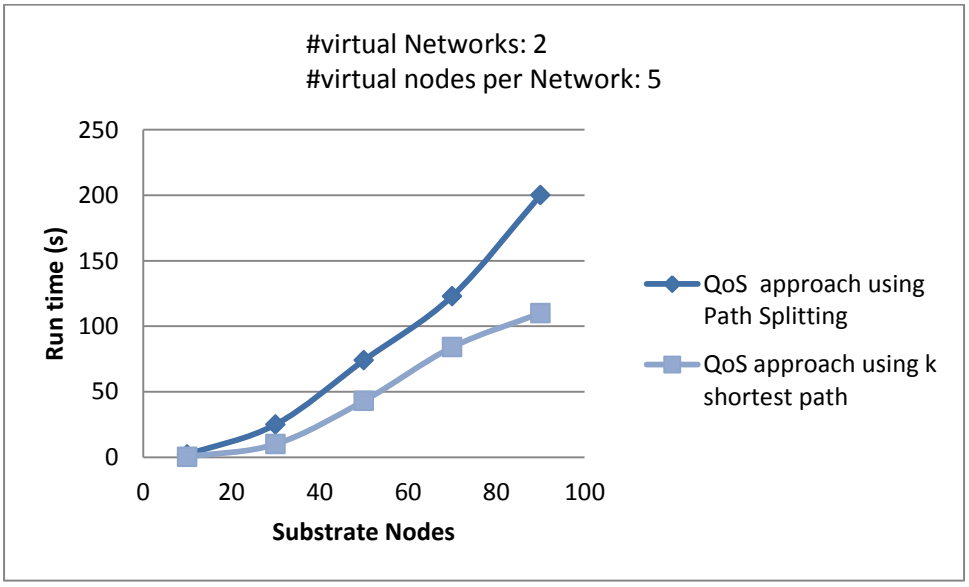


Figure 17 Comparison for average run time

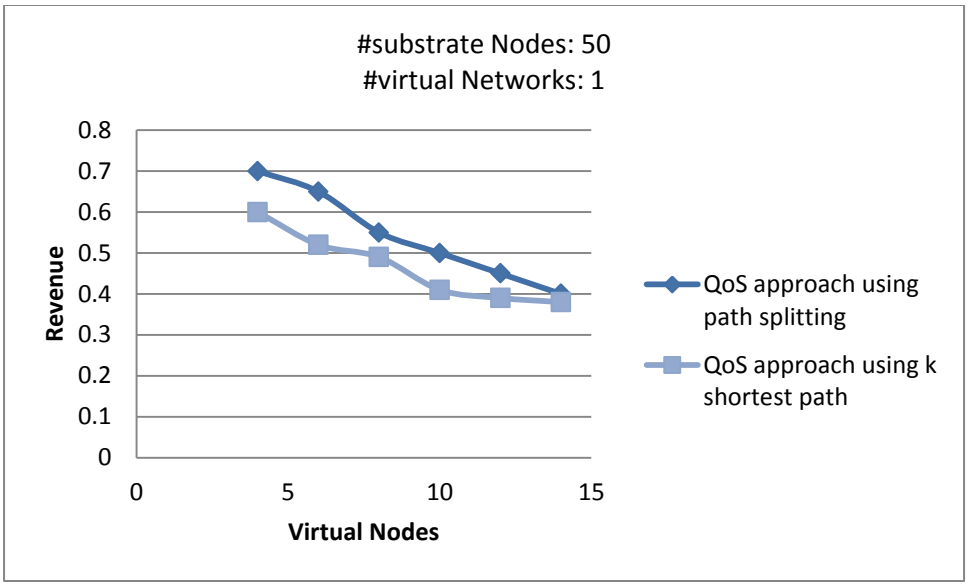


Figure 18 Comparison for revenue

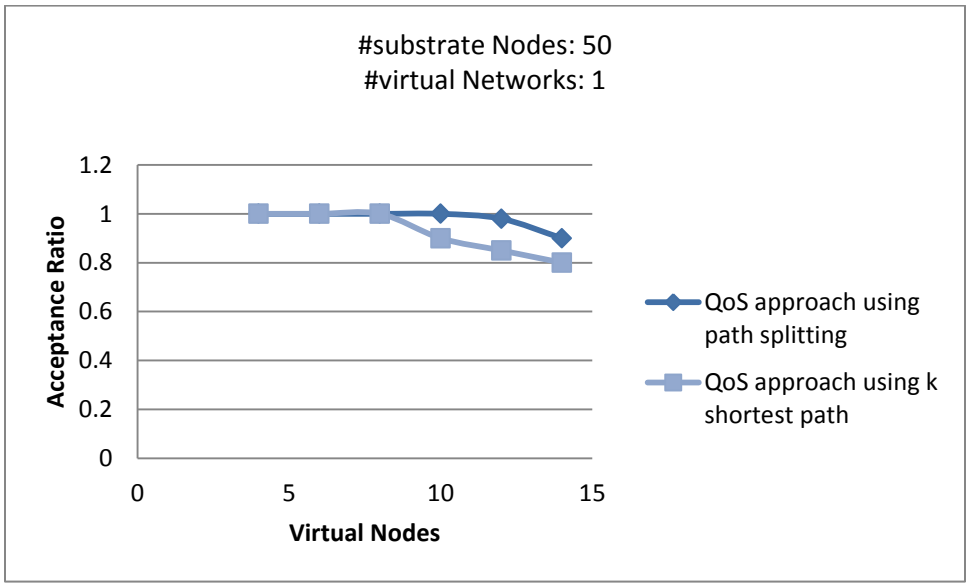


Figure 19 Comparison for acceptance ratio

In conclusion, by using path splitting the algorithm accepts more requests and if there is no substrate path to satisfy the bandwidth requirement then path splitting splits the links into smaller ones with smaller bandwidth and then the link mapping takes place. Thus path splitting provides

better resource utilization by accepting more requests. It increases the revenue and acceptance ratio of the algorithm by accepting and mapping of virtual network onto substrate network.

5.9 Summary

In this chapter, we presented the performance evaluation of the algorithms described in Table 4 and compared the results based on runtime, revenue and acceptance ratio. We realized that using path splitting leads to better VNet mapping and improves both revenue and acceptance ratio. Moreover by using path splitting, the network can benefit from load balancing and reliability since having two paths can significantly reduce the maximum load on a network compared to solutions that limit the traffic flow to a single path. Having multiple paths could also be helpful in case of recovery from failure which happens in network.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Network virtualization has been a popular subject of work for researches in recent years. The technology allows overcoming the limitations of the current Internet architecture. Meanwhile, the resource discovery and selection requires more attention from the researchers in order to expand the virtual network utilization.

Resource selection and embedding is one of the major problems in network virtualization and utilizing the substrate resources in the best possible way is the main challenge is this problem. The desired approach should consider different node and link characteristics and in case of service-oriented network, value the QoS attributes in the selection approach.

Performing the evaluation of our algorithm in Alvein framework, we realized this framework is a very flexible and strong in terms of extensibility and therefore provides several interfaces to extend existing packages in order to implement novel VNE algorithms and simulation scenarios.

6.2 Contributions of our work

Virtual resources' selection and embedding represents one of the key research issues in network virtualization and utilizing the substrate resources in the most efficient way is the main challenge related to this problem. The desired approach should consider resources' dynamic attributes along with their static characteristics. In this work we concentrated on these characteristics and also take into consideration the QoS constraints in order to be able to select the best PIP to satisfy a VNet request. The proposed algorithms are suitable for the applications that the quality of the end-to-end services plays an important role, for example customizable networks, layer 3 VPNs and VoIP.

In this thesis we proposed two algorithms based on the service-oriented network virtualization architecture. In proposed mapping algorithms, a new class of global constraints have been

proposed for the virtual network mapping problem. For node mapping, we propose a node power-degree in our algorithm and consider the capacity of the substrate nodes in order to make the best use of the substrate network. For better offering the end-to-end services, the end-to-end QoS attributes ranking has proposed in the network for selecting the paths with higher ranks in terms of QoS attributes. Moreover, for optimizing the selection procedure for resource allocation, we offer using path-splitting which results into better performance in terms of revenue and acceptance ratio.

6.3 Future work

In this work we don't monitor the network to measure the network parameters and we assumed that this ability already exist in the framework. However, for future work we can deploy a monitoring agent in order to measure the network parameters and perform the simulation with more real data.

The other aspect that could be worked on is considering the path migration in the VNet embedding problem that makes it more challenging. Moreover, our algorithm runs in a centralized manner, based on the assumption of only one infrastructure provider and required a central coordinator to maintain global substrate information which is not quite realistic in the context of the Internet.

Bibliography

1. *Overcoming the Internet Impass through Virtualization.* **Thomas Anderson, Larry Peterson, Scott Shenker and Jonathan Turner.** 2005, pp. 31-41.

2. *Towards a service-oriented network virtualization architexture.* **May El Barachi, Nadjia Kara, Rachida Dssouli.** Dec. 2010. 3rd ITU-T Kaleidoscope Event 2010. pp. 1-7.

3. *Measuremetn based characterization and provisioning of IP VPNs.* **S. Raghunath, K.K. Ramakrishnan, S. Kalyanaraman, C. Chase.** New York, YN, USA : s.n., 2004. Internet Measurement Conference.

4. *Rethinking virtual network embedding: substrate support for path splitting and migration.* **Minlan Yu, Yung Yi, Jennifer Rexford, Mung Chiang.** March 2008, SIGCOMM Comput. Commun. Rev., Vol. Vol. 38, pp. 17-29.

5. *Open Virtual Playground: Initial Architecture and Results.* **May El Barachi, Nadjia Kara, Rachida Dssouli.** 2012. in Procedding of the 9th IEEE Consumer Comminications and Networking Conference.

6. **Cisco Systems, Inc.** *Cisco IOS Software Configuration Guide.*

7. **Corporatin, Microsoft.** *Virtual Private Networking in Windows 2000: An Overview.* 1999.

8. **Duan, Qiang.** *Automatic network service discovery and selection in virtualization-based future Internet.* s.l. : GLOBECOM Workshops (GC Wkshps), IEEE, 2011.

9. *Dynamic topology configuration in service overlay networks: A study of reconfiguration policies.* **M.Ammar, J.Fan and.** s.l. : IEEE INFOCOM, 2006.

10. *Algorithms for Assigning Substrate Network Resources to Virtual.* **Ammar, Y.Zhu and M.** s.l. : IEEE INFOCOM, 2006.

11. *Efficient mapping of virtual networks onto a shared substrate.* **Lu, Jing, and Jonathan Turner.** s.l. : Washington University in St. Louis, Tech. Rep, 2006.

12. *Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration.* **Minlan Yu, Yung Yi, Jennifer Rexford.** 2008.

13. *A distributed virtual network mapping algorithm.* **Houidi, Ines, Wajdi Louati, and Djamal Zeghlache.** Beijing, China : Communications, 2008. ICC'08. IEEE International Conference on. IEEE, 2008.

14. *Virtual network embedding with coordinated node and link mapping.* **Chowdhury, NM Mosharaf Kabir, Muntasir Raihan Rahman, and Raouf Boutaba.** s.l. : INFOCOM 2009, IEEE, 2009.

15. *Resource discovery and allocation in network virtualization.* **Belbekkouche, Abdeltouab, Md Hasan, and Ahmed Karmouch.** s.l. : ommunications Surveys & Tutorials, IEEE 14.4, 2012. 1114-1128.

16. *Network virtualization: A viable path towards the future internet.* **N. Niebert, I. Khayat, S.Baucke, R.Keller, R.Rembarz,.** s.l. : wireless personal communication vol. 45, no.4, pp.511-520, 2008.

17. *Virtual Network Embedding: A Survey.* **Fischer, A., et al., et al.** s.l. : Communications Surveys & Tutorials, IEEE, 2013. vol.15, no.4, pp.1888,1906.

18. *A survey of network virtualization.* **Chowdhury, N. M., and Raouf Boutaba.** s.l. : Computer Networks 54.5 , 2010.

19. *Virtual Network Embedding: A Survey.* **Andreas Fischer, Juan Felipe Botero, Michael Till Beck, Hermann de Meer, Xavier.** 2013. IEEE Communications Surveys.

20. *A new algorithm based on the proximity principle for the virtual network embedding problem.* **J. Liu, T. Huang, J. ya Chen, and Y. Liu,.** s.l. : Journal of Zhejiang University, 2011, Vols. vol. 12, no. 11.

21. *Network virtualization: a view from the bottom.* **Jim'enez, Jorge Carapinha and Javier.** s.l. : In Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architecture.

22. *A distributed virtual network mapping.* **I. Houidi, W. Louati, and D. Zeghlache,.** s.l. : IEEE ICC, 2008.

23. *An approach towards resource efficient virtual network embedding.* **Rathore, A.Razzaq and M. S.** Richardson, TX, USA : International Conference on High Performance Switching and Routing, 2010.

24. *Polyvine: Polivy-based virtual network embedding across multiple domains.* **N. Chowdhury, F.Samuel, and R. Boutaba.** New Delhi, India : ACM SIGCOMM Workshop on virtualized Infrastructure Systems and Architectures, 2010.

25. *Multi-provider service negotiation and contracting in network virtualization in network virtualization.* **Zaheer, F.E, Jin Xiao, Boutaba, R.** s.l. : Network Operations and Management Symposium (NOMS) IEEE, 2010.

26. *Aggregation-based discovery for virtualized network environments.* **Heli Amarasinghe, Abdeltouab Belbekkouche, Ahmed Karmouch.** Ottawa, ON, Canada : Communication QoS, Reliability and Modeling Symposium (CQRM) IEEE ICC'12, 2012.

27. *Virtual resource description and clustering for virtual network discovery.* **Houidi, I. Louati, W. Zeghlache, and S. D Baucke.** s.l. : Communications, 2009.

28. *Network virtualization architecture: proposal and initial prototype.* **Schaffrath, Gregor, et al.** s.l. : Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures., 2009.

29. *Virtual network embedding through topology-aware node ranking.* **X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang.** s.l. : SIGCOMM Comput. Commun. Rev., vol. 41, pp. 38–47,, April 2011.

30. *A QoS-based Resource Selection Approach for Virtual Networks.* **Saba Behrouznia, May El Brachi, Rachida Dssouli.** Ottawa, Canada : International Conference on Computer and Information Science and Technology, 2015.

31. *Survivable Virtual Network Embedding.* **Munstasir Raihan Rahman, Raouf Boutaba.** s.l. : IEEE Transactions on network and service management, June 2013. VOL.10,NO2 .

32. *QoSMap: QoS aware Mapping of Virtual Networks for Resiliency and Enfficiency.* **Jawwad Shamsi, Monica Brockmeyer.**

33. *Network Flows: Theory, Algorithms, and Applications.* **R.K. Ahuja, T.L. Magnanti and J.B. Orlin.** . s.l. : Printice Hall, 1993.

34. *A Simulation Framework for Virtual Network Embedding Algorithms.* **Michael Till Beck, Andreas Fischer, Fabian Kokot, Claudia Linnhoff-Popien, Hermann De Meer.** s.l. : 16th International Telecommunications Network Strategy and Planning Symposium, 2014.

35. *How to model an internetwork* . **Zegura, Ellen W., Kenneth L. Calvert, and Samrat Bhattacharjee.** s.l. : INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE Computer Societies., 1996.

36. *A multi-service multi-role integrated information model for dynamic resource discovery in virtual networks.* **El Barachi, M., Rabah, S., Kara, N., Dssouli, R., & Paquet, J.** s.l. : Wireless Communications and Networking Conference (WCNC), 2013 IEEE, 2013.

37. **John D.C. Little, Stephen C. Graves.** *Little's law.* pp. 81-100.

38. *A Virtual Network Embedding Algorithm Based on Virtual Topology Connection Feature.* **Hongyan Cui, Wenjun Gao, Jiang Liu, Yunjie Liu.** 2013. ISSN:1882-5621/13.

Appendix

Appendix A

In this section we present the Dijkstra algorithm that can be generalized in order to find the K shortest path.

Definitions

- **G(V, E)**: weighted directed graph, with set of vertices **V** and set of directed edges **E**,
- **w(u, v)**: cost of directed edge from node **u** to node **v** (costs are non-negative).

Links that do not satisfy constraints on the shortest path are removed from the graph

- **s**: the source node
- **t**: the destination node
- **K**: the number of shortest paths to find
- $\mathbf{P_u}$: a path from **s** to **u**
- **B** is a heap data structure containing paths
- **P**: set of shortest paths from **s** to **t**
- $\mathbf{count_u}$: number of shortest paths found to node **u**

Algorithm:

***P** =empty,

***$\mathbf{count_u}$** = 0, for all u in V

insert path $\mathbf{P_s}$ = {s} into **B** with cost **0**

while **B** is not empty and $\mathbf{count_t} < \mathbf{K}$:

– let $\mathbf{P_u}$ be the shortest cost path in **B** with cost **C**

– $\mathbf{B} = \mathbf{B} - \{\mathbf{P_u}\}$, $\mathbf{count_u} = \mathbf{count_u} + 1$

– if **u** = **t** then $\mathbf{P} = \mathbf{P} \cup \mathbf{P_u}$

– if $\mathbf{count_u} \leq \mathbf{K}$ then

- for each vertex **v** adjacent to **u**:

– let $\mathbf{P_v}$ be a new path with cost **C** + **w(u, v)** formed by concatenating edge **(u, v)** to path $\mathbf{P_u}$

– insert $\mathbf{P_v}$ into **B**

Appendix B

XML file sample used as input for the algorithm

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Scenario
  xmlns="http://sourceforge.net/projects/alevin/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
>
  <SubstrateNetwork>
    <SubstrateNodes>
      <SubstrateNode coordinateX="-4.803" coordinateY="-2.969" id="1">
        <Resource type="IdResource">
          <Parameter name="Id" type="String" value="1"/>
        </Resource>
        <Resource type="CpuResource">
          <Parameter name="Cycles" type="Double" value="100"/>
        </Resource>
        <Resource type="NodeType">
          <Parameter name="Router" />
        </Resource>
        <Resource type="OS">
          <Parameter name="Linux" />
        </Resource>
        <Resource type="VirtualEnvType">
          <Parameter name="ZEN" />
        </Resource>
      </SubstrateNode>
      <SubstrateNode coordinateX="2.548" coordinateY="-6.104" id="2">
        <Resource type="IdResource">
          <Parameter name="Id" type="String" value="2"/>
        </Resource>
        <Resource type="CpuResource">
          <Parameter name="Cycles" type="Double" value="100"/>
        </Resource>
        </Resource>
        <Resource type="NodeType">
          <Parameter name="Switch" />
        </Resource>
        <Resource type="OS">
          <Parameter name="Linux" />
        </Resource>
        <Resource type="VirtualEnvType">
          <Parameter name="VMWare" />
      </SubstrateNode>
```

```
    </SubstrateNode>
    <SubstrateNode coordinateX="2.062" coordinateY="1.734" id="3">
      <Resource type="IdResource">
        <Parameter name="Id" type="String" value="3"/>
      </Resource>
      <Resource type="CpuResource">
        <Parameter name="Cycles" type="Double" value="100"/>
      </Resource>
      <Resource type="NodeType">
        <Parameter name="accessPoint" />
      </Resource>
      <Resource type="OS">
        <Parameter name="Windows" />
      </Resource>
      <Resource type="VirtualEnvType">
        <Parameter name="VMWare" />
    </SubstrateNode>
    <SubstrateNode coordinateX="-4.046" coordinateY="3.085" id="4">
      <Resource type="IdResource">
        <Parameter name="Id" type="String" value="4"/>
      </Resource>
      <Resource type="NodeType">
        <Parameter name="BaseStation" />
      </Resource>
      <Resource type="OS">
        <Parameter name="Unix" />
      </Resource>
      <Resource type="VirtualEnvType">
        <Parameter name="KVM" />
    </SubstrateNode>
  </SubstrateNodes>
  <SubstrateLinks>
    <SubstrateLink destination="2" id="6" source="1">
      <Resource type="BandwidthResource">
        <Parameter name="Bandwidth" type="Double" value="50"/>
      </Resource>
      <Resource type="InterfaceType">
        <Parameter name="Ethernet" />
      </Resource>
    </SubstrateLink>
    <SubstrateLink destination="1" id="7" source="3">
```

```
    <SubstrateLink destination="1" id="7" source="3">
      <Resource type="BandwidthResource">
        <Parameter name="Bandwidth" type="Double" value="50"/>
      </Resource>
      <Resource type="InterfaceType">
        <Parameter name="Optical" />
      </Resource>
    </SubstrateLink>
    <SubstrateLink destination="3" id="8" source="2">
      <Resource type="BandwidthResource">
        <Parameter name="Bandwidth" type="Double" value="70"/>
      </Resource>
      <Resource type="InterfaceType">
        <Parameter name="ATM" />
      </Resource>
    </SubstrateLink>
    <SubstrateLink destination="4" id="9" source="1">
      <Resource type="BandwidthResource">
        <Parameter name="Bandwidth" type="Double" value="40"/>
      </Resource>
      <Resource type="InterfaceType">
        <Parameter name="Wifi" />
      </Resource>
    </SubstrateLink>
    <SubstrateLink destination="3" id="10" source="4">
      <Resource type="BandwidthResource">
        <Parameter name="Bandwidth" type="Double" value="70"/>
      </Resource>
      <Resource type="InterfaceType">
        <Parameter name="Fiber" />
      </Resource>
    </SubstrateLink>
  </SubstrateLinks>
</SubstrateNetwork>
<VirtualNetworks>
  <VirtualNetwork layer="1">
    <VirtualNodes>
      <VirtualNode coordinateX="-0.757" coordinateY="-3.622" id="11">
        <Demand type="IdDemand">
          <Parameter name="DemandedId" type="String" value="1"/>
        </Demand>
```

```
      <Parameter name="DemandedId" type="String" value="1"/>
    </Demand>
    <Demand type="CpuDemand">
      <Parameter name="DemandedCycles" type="Double" value="50"/>
    </Demand>
     <Demand type="NodeType">
      <Parameter name="Switch" />
    </Demand>
  </VirtualNode>
  <VirtualNode coordinateX="4.216" coordinateY="-0.595" id="12">
    <Demand type="IdDemand">
      <Parameter name="DemandedId" type="String" value="2"/>
    </Demand>
    <Demand type="CpuDemand">
      <Parameter name="DemandedCycles" type="Double" value="100"/>
    </Demand>
     <Demand type="NodeType">
      <Parameter name="Router"/>
    </Demand>
  </VirtualNode>
  <VirtualNode coordinateX="-4.541" coordinateY="1.081" id="13">
    <Demand type="IdDemand">
      <Parameter name="DemandedId" type="String" value="3"/>
    </Demand>
    <Demand type="CpuDemand">
      <Parameter name="DemandedCycles" type="Double" value="35"/>
    </Demand>
    <Demand type="NodeType">
      <Parameter name="BaseStation"/>
    </Demand>
  </VirtualNode>
</VirtualNodes>
<VirtualLinks>
  <VirtualLink destination="13" id="14" source="11">
    <Demand type="BandwidthDemand">
      <Parameter name="DemandedBandwidth" type="Double" value="25"/>
    </Demand>
    <Demand type="InterfaceType">
      <Parameter name="Fiber" />
    </Demand>
  </VirtualLink>
```

```
        <Parameter name="DemandedCycles" type="Double" value="100"/>
      </Demand>
       <Demand type="NodeType">
        <Parameter name="Router"/>
      </Demand>
    </VirtualNode>
    <VirtualNode coordinateX="-4.541" coordinateY="1.081" id="13">
      <Demand type="IdDemand">
        <Parameter name="DemandedId" type="String" value="3"/>
      </Demand>
      <Demand type="CpuDemand">
        <Parameter name="DemandedCycles" type="Double" value="35"/>
      </Demand>
      <Demand type="NodeType">
        <Parameter name="BaseStation"/>
      </Demand>
    </VirtualNode>
  </VirtualNodes>
  <VirtualLinks>
    <VirtualLink destination="13" id="14" source="11">
      <Demand type="BandwidthDemand">
        <Parameter name="DemandedBandwidth" type="Double" value="25"/>
      </Demand>
      <Demand type="InterfaceType">
        <Parameter name="Fiber" />
      </Demand>
    </VirtualLink>
    <VirtualLink destination="13" id="15" source="12">
      <Demand type="BandwidthDemand">
        <Parameter name="DemandedBandwidth" type="Double" value="30"/>
      </Demand>
      <Demand type="InterfaceType">
        <Parameter name="Wifi"/>
      </Demand>
    </VirtualLink>
  </VirtualLinks>
 </VirtualNetwork>
</VirtualNetworks>
Scenario>
```