Utilization of Dynamic Attributes in Resource Discovery for Network Virtualization

Heli Amarasinghe

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

MASTER OF APPLIED SCIENCE

in Electrical and Computer Engineering

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

May 2012

Table of Contents

Table of Contents i
List of Figures iii
List of Tables iv
Abstract v
Acknowledgements vi
Acronyms vii
1 Introduction 1
1.1 Motivation 1
1.2 Definition of Key-words 3
1.3 Thesis Objectives 5
1.4 Thesis Contributions 7
1.5 Thesis Organization 8
2 Background and Related work 10
2.1 Overview 10
2.2 Evolution of Virtualization concepts 10
2.2.1 Virtual LAN (VLAN) 11
2.2.2 Virtual Private Network (VPN) 11
2.3 Network Overlaying 12
2.4 Network Virtualization 16
2.4.1 Business Model 16
2.4.2 Architecture 20
2.5 Issues addressed by Network Virtualization 22
2.6 Resent Virtualization Projects 24
2.6.1 Projects categorized based on the OSI model 24
2.6.2 Projects categorized based on Communication Technology 27
2.6.3 Projects categorized based on granularity of virtualization 29
2.7 Summary 32
3 Aggregation Based Virtual Network Resource Discovery 33
3.1 Objective 33
3.2 Importance of Resource Discovery in the process of VN Embedding 33
3.3 Existing Resource Discovery Techniques 35
3.4 Limitations of existing techniques 42
3.5 Improvements of proposed method over existing techniques 44
3.6 System Architecture 46
3.7 Continuous Monitoring of Dynamic Attributes 51

3.8 Initial Request Filtering 55
3.9 Sorting Selected Infrastructure Providers 56
3.10 Summary 57
4 Vector Based Resource Discovery for Network Virtualization 59
4.1 Objective 59
4.2 Weaknesses of the basic aggregation technique 60
4.3 Vector Based Resource Discovery Framework 61
4.4 Vector Based Aggregation Technique 65
4.5 Enhanced selection and Initial Filtering 73
4.6 Substrate Sorting 75
4.7 Summary 77
5 Performance Evaluation 78
5.1 Objective 78
5.2 Evaluation Strategy 78
5.3 Simulation Setup 79
5.4 Substrate and VN Request Generation 79
5.5 Performance Evaluation of Basic Aggregation Approach 81
5.6 Performance Evaluation of Vector Based Aggregation Approach 84
5.7 Summary 90
6 Conclusion and Future Work 91
6.1 Conclusion 91
6.2 Future Work 93
6.2.1 Study the relationship between cost, accuracy and efficiency 93
6.2.2 Dynamically changing the normalizing constant at runtime 93
6.2.3 Provisions for multiple VNPs, VN splitting and path splitting 94
7 References 95

List of Figures

Figure 1.1 Definition of Key-words 4
Figure 2.1 VPN Tunnel connecting two remote sites [16] 12
Figure 2.2 Failure detection and re-routing in RON [24] 14
Figure 2.3 Main roles in (a) Traditional and (b) Network virtualization business models [28]-[33] 17
Figure 2.4 Hierarchy of Roles [31], [32] 19
Figure 2.5 High-level Architecture of Network Virtualization [28]-[33] 21
Figure 3.1 Components of Topology Discovery Platform [8] 36
Figure 3.2 Virtual Resource Description and Clustering Architecture [6] 38
Figure 3.3 Dendrogram Representation of Static Attributes [6] 40
Figure 3.4 System architecture of the framework; Virtual Resource Organization and Virtual Network Embedding Across Multiple Domains [7] 41
Figure 3.5 Aggregation-based VN Resource Discovery (ADVNE) System Architecture 47
Figure 3.6 Incremental Aggregation over Network Spanning Tree [60] 53
Figure 4.1 Enhanced Resource Discovery Framework with Vector based Aggregation 62
Figure 4.2 Operational Sequence of Vector-based Resource Discovery Framework 64
Figure 4.3 Vector-based Incremental Aggregation over Network Spanning Tree 69
Figure 4.4 Element-wise Matching of Request and Aggregation Vectors 74
Figure 5.1 Performance and Behavior of Tree and Aggregation Algorithms 81
Figure 5.2 Number messages exchanged vs. Number of physical nodes in Aggregation and centralized approaches 82
Figure 5.3 Average time, to take final decision of acceptance or rejection due to different factors 83
Figure 5.4 Number of rejections in the resource discovery phase vs. number of substrate nodes (Accuracy of pre-matching) 85
Figure 5.5 Average VN rejection time vs. number of substrate nodes (cost of pre-matching) 86
Figure 5.6 Total processing time taken for 5000 VN requests with different resource discovery approaches 88
Figure 5.7 Efficiency of VN embedding over the size of vector 89

List of Tables

Table 4.1 Vector-based Aggregation Results Analysis 71
Table 5.1 Composition of VN requests..... 80

Abstract

The success of the internet over last few decades has mainly depended on various infrastructure technologies to run distributed applications. Due to diversification and multi-provider nature of the internet, radical architectural improvements which require mutual agreement between infrastructure providers have become highly impractical. This escalating resistance towards the further growth has created a rising demand for new approaches to address this challenge. Network virtualization is regarded as a prominent solution to surmount these limitations. It decouples the conventional Internet service provider's role into infrastructure provider (InP) and service provider (SP) and introduce a third player known as virtual network Provider (VNP) which creates virtual networks (VNs). Resource discovery aims to assist the VNP in selecting the best InP that has the best matching resources for a particular VN request. In the current literature, resource discovery focuses mainly on static attributes of network resources highlighting the fact that utilization on dynamic attributes imposes significant overhead on the network itself.

In this thesis we propose a resource discovery approach that is capable of utilizing the dynamic resource attributes to enhance the resource discovery and increase the overall efficiency of VN creation. We realize that recourse discovery techniques should be fast and cost efficient, enough to not to impose any significant load. Hence our proposed scheme calculates aggregation values of the dynamic attributes of the substrate resources. By comparing aggregation values to VN requirements, a set of potential InPs is selected. The potential InPs satisfy basic VN embedding requirements. Moreover, we propose further enhancements to the dynamic attribute monitoring process using a vector based aggregation approach.

Acknowledgements

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

Secondly I am really grateful to Dr. Abdel Belbekkouche, who has enriched and refined my knowledge with his valuable ideas and comments. I would also like to thank Mahmud Hasan, Dr. Jerry Abdellah, Yousif Al Ridhawi, Imad Abdeljaouad, Ismaeel Al Ridhawi, Bassem Wanis and other members of the Intelligence for Mobile Autonomic and Cognitive Networks Laboratory for their help and support during my research. Their companionship and friendship eased many tasks during times of difficulty. I am also thankful to Dr. Kapila Zoysa for spending his valuable time, reading the initial draft of the thesis and helping me to make it more reader friendly.

Finally, I want to dedicate this thesis to my parents who have given me the chance of a good education, and so much love and support over the years. I probably owe them much more than I think. I am also grateful to my two brothers Nethu and Dinu, who have always been a source of encouragement, support and help. I am also very grateful and indebt to my wife Apsara who has helped me since I started writing this thesis, and for her patience, love and encouragement.

Acronyms

ADVNE	Aggregation-based Discovery for Virtual Network Environments
ATM	Asynchronous Transfer Mode
BFS	Breadth First Search
BGP	Border Gateway Protocol
CPU	Central Processing Unit
HDSL	High-bit-rate Digital Subscriber Line
InP	Infrastructure Provider
IP	Internet Protocol
IPTV	Internet Protocol Television
ISP	Internet Service Provider
LAN	Local Area Network
MPLS	Multi Protocol Label Switching
OS	Operating System
OSI	Open Systems Interconnection
VDSL	Very-high-bit-rate Digital Subscriber Line
VLAN	Virtual Local Area Network
VN	Virtual Network
VNP	Virtual Network Provider
VOIP	Voice Over Internet Protocol
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network
WiMAX	Worldwide Interoperability for Microwave Access

1 Introduction

1.1 Motivation

The success of the internet as a wide area network has become the most influential driving force towards the growth of various network based services during past three decades. It has instigated a new era of communication by providing new ways to connect people and greatly increasing the accessibility to information. However, its own popularity has created obstacles for further growth and innovations in the underlying computer network technologies [1]-[5].

Public internet consists of multiple infrastructure provider networks with different organizational and financial objectives. Owing to multi-provider nature, deployments of new technologies or major architectural changes that entail cooperative agreement between those competing infrastructure providers have proved highly impractical. Number of favorable architectural and fundamental improvements (e.g. IP multicast, differentiated services and secure routing protocols) were unsuccessful in gaining expected popularity since they failed to get the required mutual agreement between internet stakeholders in order to provide expected benefits. This high resistance towards further growth has resulted in a stagnation state which is widely known as ossification problem [1], [2]. Network virtualization is regarded as a prominent solution to surmount these ossifying forces allowing the deployment of new powerful network services [1]-[5].

Network virtualization allows multiple logical networks to coexist in a shared physical infrastructure composed of one or more infrastructure provider networks. It decouples the

role of conventional internet service providers into *infrastructure providers* and *service providers*. The task of infrastructure Providers (InPs) is to provide and manage physical infrastructure while Virtual Network Providers (VNPs) create virtual networks on top of InP networks to satisfy Virtual Network (VN) user requirements. Besides this main function, VNP perform the duty of identifying and selecting suitable InP or InPs for each VN request when multiple InPs exist in the system associated by service agreements with VNP. These selection decisions are made by VNP based on the potential of each InP to successfully host (or embed) the VN request. The embedding potential is determined based on network attributes such as network element type, virtualized environment, operating system, network stack, available processing power, available memory and link bandwidth [9].

The network attributes are mainly divided in to two categories; static and dynamic based on the propensity to change with time dimension [6]-[10]. Static network attributes (also known as non-functional attributes) are the characteristics of network elements that do not change with time. Some examples of static network attributes are type of element, execution environment, virtualization environment and operating systems. On the other hand, dynamic (functional) network attributes are the changing parameters of network elements. Examples of dynamic network attributes are available processing power and memory of routing and switching equipments and available bandwidth of network links. Due to their inherent nature of variability with time, dynamic network attributes demand continuous monitoring to obtain accurate values at a given instant [6].

When selecting best candidate substrate networks to provide requested virtual networks, values of the substrate network static and dynamic attributes need to be accurately measured or estimated. The available research works in the literature [6]-[10] proposes techniques of utilizing static network attributes when selecting most appropriate substrate network or set of substrates for a given virtual network request. However, those research works circumvent the obligation of utilizing the dynamic attribute values by highlighting the fact that continuous monitoring of dynamic attribute imposes significant overhead over the network which is being monitored. Despite the negative comments of previous research work on the use of dynamic attributes for selecting appropriate substrate network, it is still worthy of investigating the possibility of utilizing the capacity of dynamic attributes owing its immense benefits attached to its character. Therefore in this thesis, a technique is proposed to utilize dynamic attributes to implement efficient resource discovery scheme for network virtualization, recognizing the tradeoff between cost and accuracy of dynamic attribute monitoring.

1.2 Definition of Keywords

In this section we define the keywords; “virtual network” and “network virtualization”. We also define virtual nodes and virtual links which are the main components of virtual network. Virtual node is a logical entity, constructed by partitioning substrate network equipment (e.g. router) resources such as processing power and memory. The virtual node is capable of controlling the allocated subset of resources independently, which guarantees isolation from other coexisting virtual nodes in the same substrate node. Similarly, virtual link is constructed by partitioning interconnecting link (e.g. copper

wire, optical, microwave) resources such as bandwidth, and guaranteeing the isolation from coexisting logical links, to function independently [1]-[5]. Concept of coexistences of multiple virtual nodes on single physical node and multiple virtual links on single physical link can be illustrated as shown in Figure 1.1.

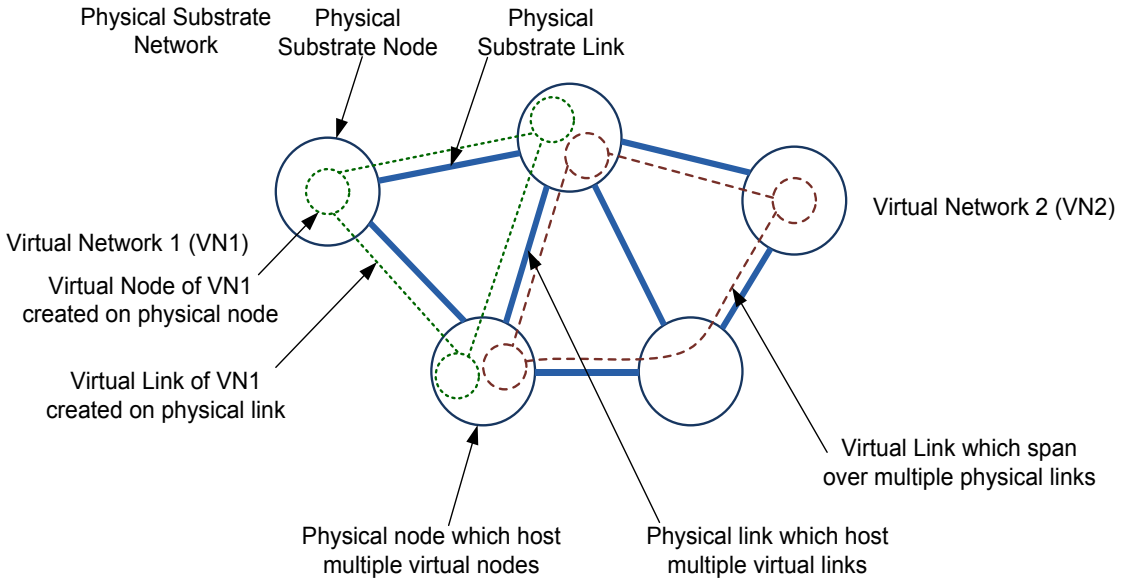

Figure 1.1 Definition of Key-words

AVN is a set of virtual nodes interconnected by virtual links. Any substrate node can host multiple virtual nodes belong to one or more virtual networks. In other words, a particular substrate node may be hosting none or any number of virtual nodes based on location processing and node constraints defined by virtual network provider. Similarly, a virtual link can span over multiple substrate links when it requires to interconnect virtual nodes hosted by non-neighboring substrate nodes. In this research, we stay with the pluralist approach of network virtualization, which defines coexistence of architecturally diversified and fully virtualized, multiple virtual networks in a single substrate network. The pluralist philosophy allows virtual networks to offer network services and maintain a

clear separation from their underlying infrastructure [2]. Thus, we do not limit the scope of virtual networks discussed in this thesis to a single layer of virtualization (e.g. layer 2 VPN, layer 3 VPN, application layer overlay) or a single underlying communication technology (e.g. copper-wire, optical, wireless).

For this reason, we identify the term "Network virtualization" as the process of setting-up multiple separate virtual networks that coexist on a shared substrate. This paradigm is expected to be the foundation of the future internet architecture where multiple coexisting heterogeneous network architectures from different virtual network providers, sharing a common physical substrate managed by multiple infrastructure providers [1]-[11].

1.3 Thesis Objectives

The goal of this thesis is to design and simulate an efficient substrate network resource discovery scheme to increase the overall efficiency of the virtual network embedding process. According to literature, the research work that has given due attention to resource discovery in network virtualization is very scanty; the importance of identifying suitable substrate resources for a particular VN request is significant. Therefore, a sophisticated VN resource discovery scheme must consist of following key characteristics:

Static attribute description and matching: Proficient resource discovery scheme should be capable of matching static attribute requirements of requests to each InP network and select best matching InP or set of InPs. The network element can be a router, switch, link, base station or a managing node. Static attributes describe characteristics, properties and functions of an element such as OS (Cisco, Linux, Solaris, Juniper), Type

(Router, Switch, link, host), Access Technology (ATM, FR, T1/E1, HDSL, VDSL) and Link type (Optical, Coaxial, Microwave). Each VN request comprises of its own set of static attribute requirements. To achieve the objective of efficient static attribute matching, they need to be described in terms of properties and functionalities. Well known network description specifications such as cNIS [12] and NDL [13], provides schema based resource description techniques, which can be used for this purpose. Then the properly described static attributes need to be classified to allow fast matching of resources. Cluster based classification techniques proposed by [6] and [7] can be used to perform static attribute matching efficiently.

Cost-effective utilization of dynamic attributes: Taking resource discovery decisions exclusively based on static attributes is irresolute, since dynamic attributes play a major role in the embedding process. Substrate resources selected only based on static attribute matching has higher tendency of rejection at latter stages of the VN embedding process than the resources selected by both static and dynamic attribute matching. Therefore, efficient utilization of dynamic attribute utilization is important for the overall performance of the embedding process [5]. However there are several impediments associated with acquiring accurate values of those dynamic network attributes. The fluctuation of attribute values with time raises major challenges, demanding continuous monitoring, delays and inevitably intense monitoring costs. Moreover, from the security perspective, InPs will be highly reluctant to reveal detailed information of their infrastructure to an external business entity such as VNP. Hence sophisticated resource discovery scheme should be capable of extracting most crucial dynamic attribute information required to perform efficient resource discovery.

Recognizing and controlling cost vs. accuracy tradeoff: In almost every scenario that involves measuring or monitoring process to obtain required data or information, accuracy of the final output is proportional to the amount of resources allocated to the process. Invariably, accurate results always demand high costs and there is a tradeoff between the accuracy and cost. As the accuracy of the monitored dynamic attribute values increases, the associated cost of monitoring will inevitably increases since monitoring requires additional bandwidth from the network links and processing power and memory from the nodes. Since cost of monitoring is an overhead to the network which is being monitored, employed techniques should be able to extract only most crucial information minimizing the monitoring overhead. Hence ability of the monitoring software to conveniently controlling this trade-off is highly important. Therefore we attempt to fortify our proposed virtual network resource discovery framework, reflecting the importance of the key characteristics.

1.4 Thesis Contributions

According to the characteristics highlighted in section 1.3, an efficient network virtualization resource discovery framework has to define and match the static attributes and dynamic attributes efficiently, when selecting most suitable resources and InPs for the VN request in hand. However in our research work, we mainly concentrate our attention towards dynamic network attributes since state of the art is sufficiently rich in static attribute description and matching techniques as we discussed previously [6], [7].

We have focused our attention on resource discovery for network virtualization, distinguishing it as an inadequately explored branch of the network virtualization. We

introduce aggregation-based resource discovery for network virtualization [14]; a VN embedding framework which increases the overall efficiency of VN embedding process by utilizing dynamic attributes of the network resources at the initial stage of embedding to filter InPs before forwarding the VN binding request. For this purpose, we calculate aggregation values of dynamic attributes for each InP which was selected by static attribute matching process. Based on calculated dynamic attribute values, we filter InP networks in order to shortlist a set of InPs which satisfy basic embedding requirements. We sort these InPs in a descending order of embedding potential and sequentially forwarding VN resource requirement specifications to select the best candidate InP. Moreover, we further enhance the monitoring and aggregation process by employing vector-based Aggregation technique by increasing the controllability over the well-known tradeoff between costs vs. accuracy. Finally, we have performed simulations and analyzed the results to verify the performance of our approach.

1.5 Thesis Organization

The remainder of this thesis is structured as follows:

In Chapter 2, the background information and related work is presented which motivated us to investigate the network virtualization paradigm. We put our efforts to illustrate the evolution of virtualization concepts from Virtual Local Area Networks (VLANs) and Virtual Private Networks (VPNs) to current network virtualization notions. Then the detailed description of network virtualization business model and architecture and issues of the current wide area networks are successfully addressed by network virtualization.

Finally a summary is included on the state of the art of network virtualization including similar research projects available in the literature.

In Chapter 3, we accentuate the importance of a sophisticated resource discovery scheme in VN embedding process and perform a detailed analysis of the closely related resource discovery approaches available in the literature. We point out several limitations of those closely related research works in the literature and highlighted the advancements in our proposed architecture. We also illustrate our enhanced virtualization resource discovery approach by providing the system architecture and main functions of the proposed framework.

In Chapter 4, we highlight some noticeable impediments of the basic aggregation technique proposed in chapter 3, and strive to address them by introducing a vector based aggregation technique. We provide mathematical formulation to demonstrate the ability of the proposed vector based discovery scheme to control the accuracy and cost of the dynamic attribute utilization at the resource discovery phase.

In Chapter 5, we evaluate the performance of the both proposed approaches by carrying out simulations. We discuss the objectives of the simulations and provide indispensable information related to simulation setup and configurations. We also compare simulation results and perform a detailed analysis to reach important conclusions.

In Chapter 6, we make the concluding remarks and emphasize on potential future enhancements to the proposed resource discovery frameworks.

2 Background and Related work

2.1 Overview

In recent years, network virtualization has attracted a significant attention of the researchers who work on future internet paradigm However, the underlying concept of virtualization has started evolving few decades ago [1]-[5]. Throughout the course of evaluation, some concepts were successfully adopted by industrial applications such as VLANs, VPNs and Overlay Networks [4].

In this chapter, we summarize several prominent applications of network virtualization theories and their characteristic relations. Among these technologies, we discuss network overlaying techniques in-depth, aiming at providing the key applications and conceptual differences between network overlays and virtual networks. Then we introduce the network virtualization business model, players and system architecture. Finally, we categorize the state of the art of network virtualization based on recognizable characteristics of noticeable research projects.

2.2 Evolution of Virtualization Concepts

The term virtual networks has been initially introduced by telecommunication companies in late 1980's, as a method of logically partitioning large telecom networks into smaller networks with the motive of achieving flexibility and control escalating costs associated with the large-scale telecommunication networks. Later with the popularity of data networks and the internet, the concept of existence of multiple, logically separated

networks on shared substrates have been widely highlighted in the networking literature in different capacities. From those technologies, Virtual Local Area Networks (VLANs), Virtual Private Networks (VPNs) and Virtual Private LAN Services (VPLSs) were widely accepted by industrial community as successful virtual network technologies to achieve logical separation among communication links co-exist in the same physical substrate. Extending the functionalities of the substrate internet is a distinguishable characteristic of all those technologies, even though they were introduced aiming diverse objectives [3], [4].

2.2.1 Virtual LAN (VLAN)

VLANs are logical networks that run on top of physical LAN to allow hosts with common functional or organizational requirements to communicate with each other although they are not physically connected to the same physical network switch or hub. VLANs can be used to create multiple logical networks on a single physical LAN or even they can be used to logically combine several physical LAN segments to function as a single LAN. Several VLANs can coexist in a single routing domain and even they can be configured to span over multiple routing domains [15].

2.2.2 Virtual Private Network (VPN)

VPN is a private network that uses a public network (usually the internet) to connect remote sites or users together as shown in Figure 2.1. VPN uses "virtual" connections routed through the internet from the business's private network to the remote site or employee ensuring data security. Based on the protocols used in the data plane, VPNs can be classified into Layer 1 VPN, Layer 2 VPN, Layer 3 VPN and higher layer VPNs.

Layer 4 and other higher layer VPNs are not very common, since more controllability can be achieved by VPNs implemented at the transport and application layers of the protocol stack. The most prevalent method of providing virtualization at these layers is to use encryption services at either layer [16], [17].

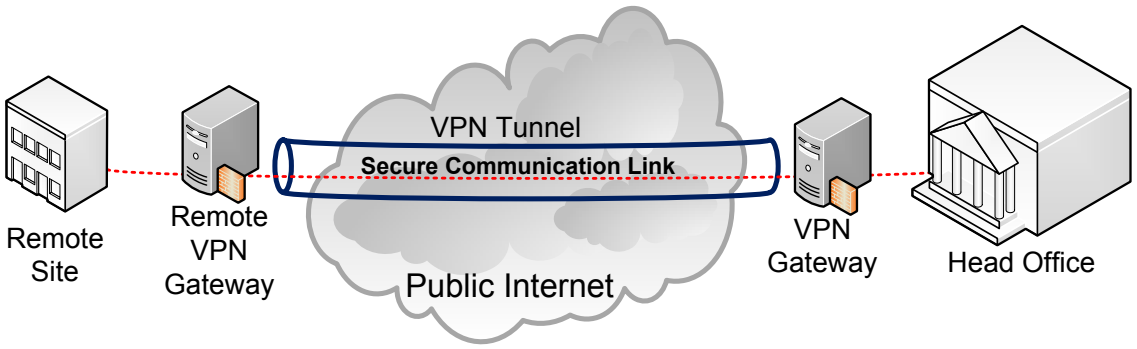

Figure 2.1 VPN Tunnel connecting two remote sites [16]

However, VPNs have some inherent limitations that make them inefficient in certain situations. Tunneling technique employed in VPNs impose constraints when deploying network services, since it only offer a short-term ad-hoc solution instead of providing long-term global solutions. Lack of controllability over the availability and performance is another significant disadvantage in VPNs. Moreover, link speed can be typically slower due to high processing power demanded by encryption algorithms to maintain the obligatory VPN security [19].

2.3 Network Overlaying

Network overlaying can be considered as one of the widely discussed technologies that attempt to address internet ossification problem. They can be used to deploy new network services exclusively on the application layer and thus can be implemented without the

necessity of modifying the underlying network core. Hence, designers are capable of implementing routing and traffic management techniques, tailored to execute application services on top of the underlying wide area networks [20], [21]. The public internet was also initiated as an overlay network, since it was created on top of the public telephone network that connect internet routing and switching equipments by long distance telephone cables. Similarly, modern overlays run on top of the internet infrastructure, building application layer network and managed packets by routing algorithms specified by the overlay network designers. Thus new functionalities and new services can be deployed using overlay networks instantly without the necessity of reconfiguring routers and hardware upgrades which usually take massive amount of resources. They also provide researches a powerful platform to conduct experiments and create new services.

Network Overlaying does not require the need of deploying new equipments or modifying existing routing protocols or networking software. Instead, overlaying deploys new software on top of the existing one. Another specialty of overlaying is freedom to select nodes to place overlay nodes. When deploying an overlay network on top of a substrate, most substrate nodes only need to perform packet forwarding function, and specific substrate nodes are required to execute the overlay routing and switching software on them. In this way, overlay network operator can select substrate nodes bypassing bottleneck nodes and weak nodes in terms of security and processing power. Additionally, overlaying combines packet switching and circuit switching advantages and decides paths optimized for requirements of the application services. In addition, overlay networks can dynamically get adjusted to congestions by selecting network routes through non-congested segments of the underlying network. Finally, overlay obtain the

advantage of the surplus of processing memory and permanent storage available in the internet working hardware to perform tasks that are usually unable to perform using conventional routers, such as resource intensive cryptographic operations, database lookups or file caching. Performing the slow and resource intensive tasks by overlay software helps to prevent critical routers which usually perform those operations, being overloaded. The possibility of performing the tasks enables to implement powerful new facilities such as distributed, scalable content distribution networks.

Overlay networks can be divided into two types based on their design architecture as routing overlays and peer-to-peer overlays. Virtual private networks (VPNs) can be considered as a simple routing overlay model designed to allow secure communication over public internet. Other than that, routing overlays were used to rout IP version 6 packets over IP version 4 networks, IP multicasting [21], [22] and design robust and resilient routing architectures [23]. Resilient Overlay networks (RON) [24] is a routing overlay technology designed to increase the robustness of the network routing.

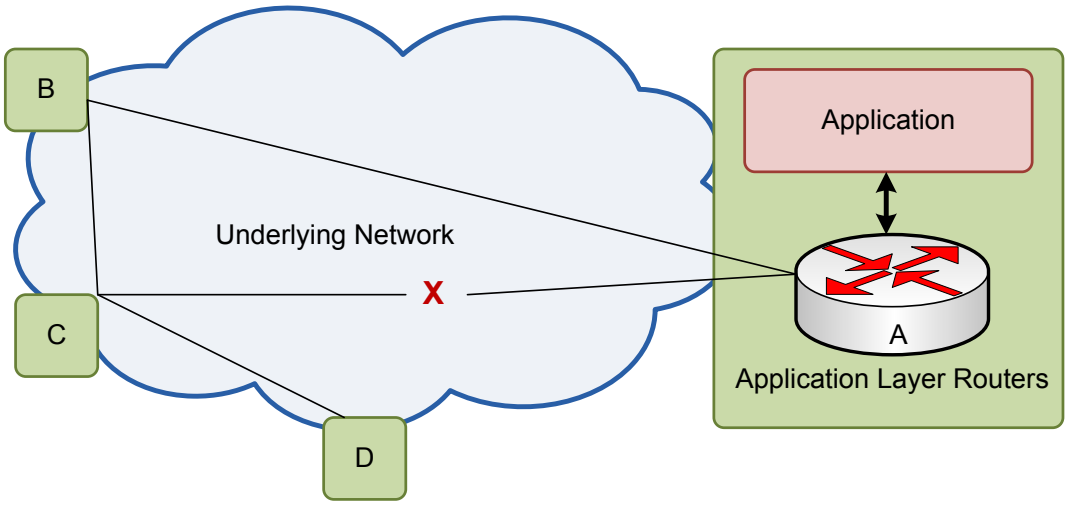

Figure 2.2 Failure detection and re-routing in RON [24]

RON builds an application overlay network with the intention of increasing the performance and reliability of routing. RON addresses certain limitations visible in IP routing such as adapting to congestions by detecting congestions and rerouting, fast convergence by quickly directing traffic through intermediary and dynamically selecting network paths that can circumvent autonomous system routing policies as shown in Figure 2.2.

Beyond these, p2p overlays are capable of offering several unique set of services, since each node in the network simultaneously perform functions of the server and client executing functionally equivalent software in both nodes. This symmetry of roles allows sharing of bandwidth, storage and computation power resources. Hence the total capacity and scalability increases with the demand and number of peers joining the system [20], [25]. However, despite those advantages, there are considerable amount of overhead costs associated with overlay path selection, path splitting, additional redundancy requirements and additional security requirements which demand significant level of processing overhead. Overlay networks do not provide general solutions for efficient deployment of new network services over large scale networks, although they allow application layer services with limited scalability. Moreover, for the reason they are implemented mostly in application layer, traffic management schemes in lower layers are barely improved. Due to such reasons, network overlaying is not widely recognized as a credible technology, though it is capable of addressing the ossifying forces of the internet, which we have discussed previously [1]-[5].

2.4 Network Virtualization

Virtualization can be used as a tool for optimum and efficient utilization of resources as it provides benefits by hardware platform independency, scalability, optimization and ability to allocate resources dynamically. It uses a layer of software that provides the illusion of a real hardware (physical memory, physical storage, physical network etc.) to multiple instances of virtual hardware (virtual memory, virtual storage, virtual network etc.). Concepts of virtualization have evolved in computing under several fields including hardware, software, memory, storage, data and network [1]. Virtual networking is recognized as a promising solution to limitations in the existing internet.

2.4.1 Business Model

Throughout the history of computer networking, decoupling of roles is recognized as a means of achieving flexibility and increased controllability and assists defining roles of each player and their interactions [26], [27]. In the traditional internet architecture, application service providers have to deploy their network services by entering into direct agreements with InPs in order to achieve end-to-end networking connectivity, required for their services as shown in Figure 2.3(a). If service destination points cannot be connected by a single infrastructure provider network, service provider is obliged to deal with multiple InPs to achieve the end-to-end connectivity. As we have mentioned earlier, network virtualization is defined by decoupling the role of the traditional internet service provider (ISP) into two independent entities; Infrastructure Provider (InP) and Virtual Network service Providers (VNP). InP's own and manage the physical infrastructure and

VNP's create and manage virtual networks by aggregating resources from multiple Infrastructure providers and offer end-to-end services.

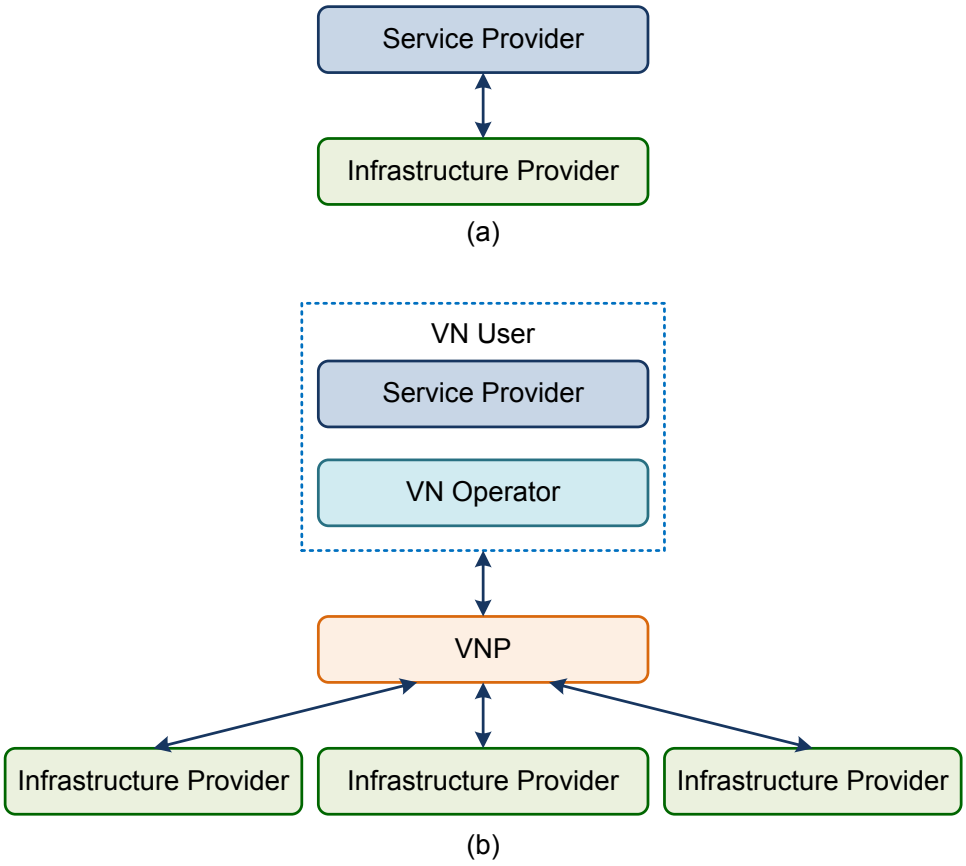

Figure 2.3 Main roles in (a) Traditional and (b) Network virtualization business models [28]-[33]

However, these business roles are not necessarily need to be handled by distinct business entities. Any business entity can perform one or more of these roles. The aspiration behind the introduction of these roles is to minimize the associated fix costs of maintaining physical resources.

4WARD [28]-[30] network virtualization business model defines four major business roles as shown in Figure 2.3(b). According to the 4WARD model, service providers do not directly deal with InPs. Service providers deal with VN Operators to deploy their

application specific services. VN Operators install network protocols and maintain VN slice created by VNP. However, for the architectural simplicity, we combine the roles of service provider and VN Operator in to a single business entity, and identify it as VN user.

The composition of the underlying network is transparent to VN users who deploy services, using the VNs deployed on top of the virtualization layer. Hence, VN users in the virtualization business model are similar to the application service providers in the current internet architecture. The only difference is they experience more flexibility and availability when choosing node locations to deploy their services. By acquiring VN services from VNP, application service providers play the role of end user in the business model and VNP plays the role of the service provider.

In a single VNP scenario, VN users see only one virtualization business layer maintained by VN provider who provides single, which can be configured according to their service requirements. Hence, a VN user such as video streaming (ex IP-TV) service provider who wants to extend his services to a remote area, only needs to negotiate with the VN provider for specific hardware resources required for those service extension. VNP will then communicate with the infrastructure providers to obtain the required resources. InPs need to enable the creation of virtual nodes and links over their available resources and provide them to the VNP. VNP will create the VN slice which is a lifeless set of virtual resources as requested by VN User. VN Users will deploy customized protocols by programming the allocated network resources to obtain end-to-end connectivity to provide desired services to their customers (e.g. end users of the IPTV service).

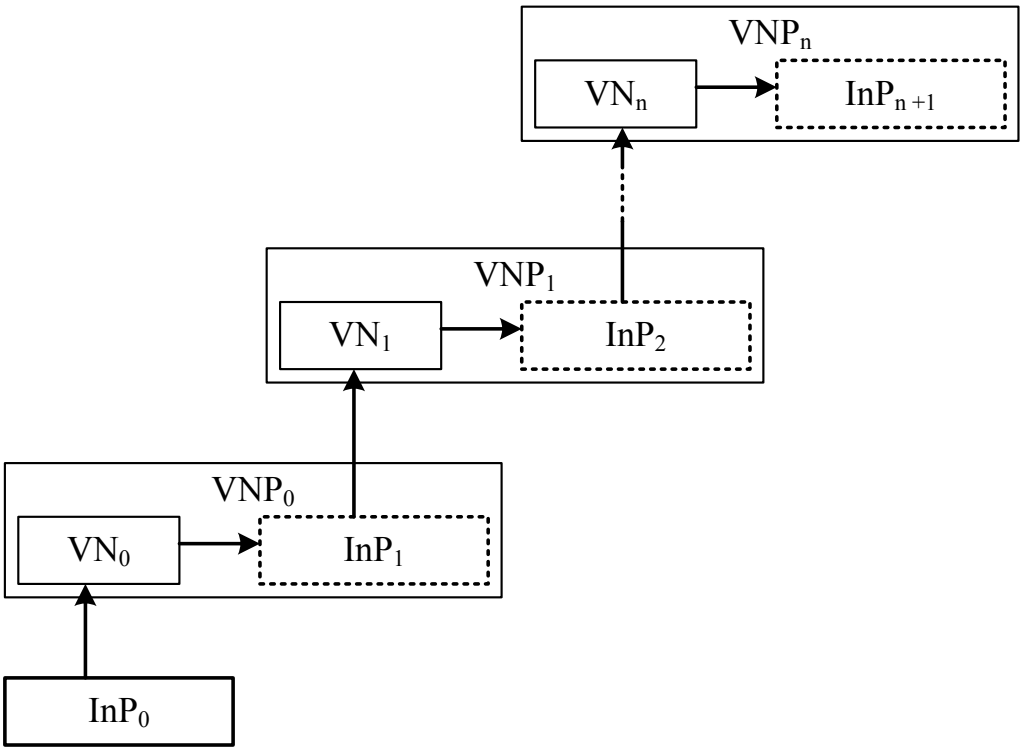

Figure 2.4 Hierarchy of Roles [31], [32]

If VN provider is capable of leasing more resources from InPs, they can use them to provide virtual network slices for multiple VN Users and VNs created on those slices will co-exist without interfering each other. Furthermore, one VNP can lease its excess resources to another VNP who can again lease the resources to VN users or more VNPs as shown in Figure 2.4. This is known as recursion and it creates hierarchy of roles and for a VNP in a top level can envision its underlying VNP as an infrastructure provider. However at recursion, constraints of the parent virtual network are automatically transferred to its child, which is called the constrain inheritance. For an example, in Figure 2.4, constraints imposed by VNP_0 on its VN_0 will be automatically transferred to all the child VNs built on top of VN_0 [31].

In certain circumstances such as when the system consists of multiple users, multiple virtual network providers and multiple infrastructure providers, a broker or a mediating fourth role can play a vital role in the system by matching VN user's requirements with the VNP and InP services. Moreover the broker will be able to improve communication between players and ensure reliability of the network virtualization process [31], [32].

2.4.2 Architecture

The fundamental idea behind the virtualization is the introduction of virtualization layer which decouples the role of conventional network service provider to VN provider and infrastructure provider. Virtualization layer is managed by VNP who lease resources from InPs, construct VN slices based on VN user requirements to deploy VNs to end to end connectivity and custom network solutions to VN users. The major architectural characteristic of these VNs is the coexistence, which allows multiple VNs deployed by the same or different peer VNPs, execute simultaneously and perform their intended networking functions on a shared substrate composed of multiple InPs. As an example, Figure 2.5 shows coexistence of VN_1 and VN_2 which are deployed by the same VNP, on top of the physical resources of InP-1 and InP-2.

VN is a collection of virtual nodes and a set of virtual links that connect virtual nodes. Virtual nodes are constructed by partitioning the processing power, memory and other node resources in pre-defined proportions from physical network devices such as routers. Two virtual nodes coexist in the same physical node do not have shared memory, shared registers or any other association, unless they are connected by a virtual link.

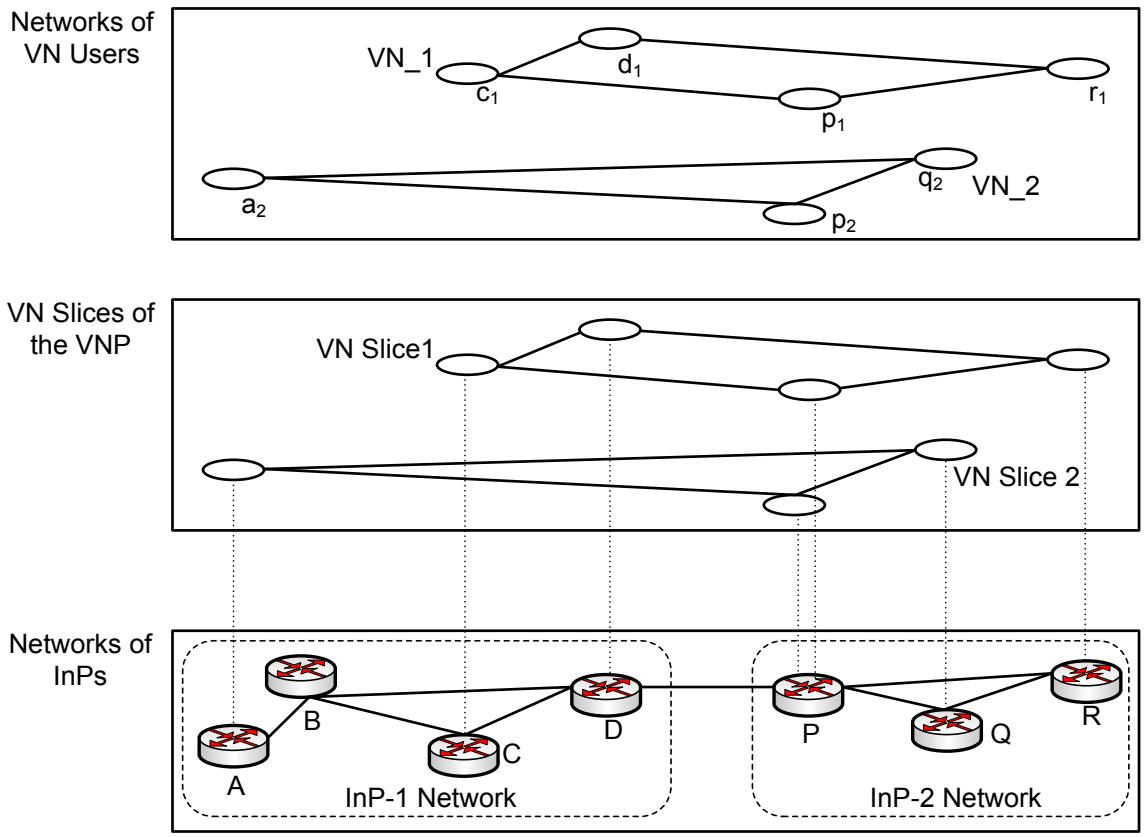

Figure 2.5 High-level Architecture of Network Virtualization [28]-[33]

As an example, in Figure 2.5, two virtual nodes p_1 and p_2 which respectively belong to VN_1 and VN_2 coexist on the same physical node P, with complete logical isolation between each other. If one virtual node is attacked, compromised or move to an un-stable state, it should not affect the other virtual nodes residing in the same physical node [32].

On the other hand, virtual links are constructed by partitioning the bandwidth and other link resources in defined proportions from physical wired or wireless network links. Similar to virtual nodes, two virtual links coexist in the same physical link should not have any interactions or interferences from each other. A virtual link can span over more than one physical link to connect virtual nodes that are not located in neighboring

physical nodes [30]-[33]. Let (x-y) represents the link between nodes x and y. According to Figure 2.5, the virtual link (r_1-d_1) spans over physical links (R-P) and (P-D) and virtual link (a_2-q_2) uses (A-B), (B-D), (D-P) and (P-Q) substrate links to connect virtual nodes a_2 and q_2 in VN-2. Also multiple virtual links either from the same virtual network or from different virtual networks, can use bandwidth and other link resources of same physical links such as virtual links (d_1-r_1) and (c_1-p_1) of VN-1 and (a_2-p_2) and (a_2-q_2) of VN-2 operated in inter substrate link (D-P) simultaneously to achieve end to end connectivity.

Underlying substrate network can be composed of networks owned by different Infrastructure providers (physical network owners). These physical infrastructure providers can offer network resources such as processing power, memory and bandwidth from selected network equipments. They can keep core network equipments and critical topology information hidden from VNPs to achieve traffic engineering and network security requirements [32]. For an example, router B of InP-1 in the Figure 2.5 is not revealed to VNP and hence VNP does not have the actual topology information of InP-1.

2.5 Issues Addressed by Network Virtualization

Whenever a new more efficient routing algorithm or protocol needs to be implemented within the current internet architecture, hundreds of millions of currently deployed internet routers and other devices need to be upgraded or replaced. Moreover, due to multi-provider nature of the internet, it is prerequisite for the internet service providers to enter into collaborative agreements to achieve expected benefits of most new protocols. For an example, internet service provider that deploys an advance secure routing protocol such as S-BGP will only be able to harness full benefits of it if all other internet service

providers deploy S-BGP on their networks [2]. Network virtualization attempts to provide strong solutions to most of these impasses as we discussed previously [1]-[5]. It does not require deploying new equipments or modify existing routing protocols or networking software. Instead, virtualization layer provides separation between application service providers and infrastructure providers and allow building customized virtual networks. Also, virtualization takes the benefit of the surplus of processing, memory, and permanent storage available in internetworking hardware. Another specialty of virtualization is freedom to select nodes to place virtual nodes. When deploying a virtual network on top of a substrate network, most substrate nodes only need to perform packet forwarding function, and only selected substrate nodes are required to host virtual nodes on them. Hence, VNP can select substrate nodes while bypassing bottleneck nodes and weak nodes in terms of security and processing power. On the other hand, InPs have the freedom to offer resources to virtual networks from selected network elements based on network traffic monitoring results or other requirements [30]-[33]. Other than that, virtualization combines packet switching and circuit switching advantages and allow optimized path selection, based on traffic patterns and requirements of the application service providers. In addition, virtual networks can be configured to adjust and avoid substrate network congestions by selecting network routes through non-congested segments of the underlying network or employing path splitting [34], [35] and re-optimization [36] techniques.

Furthermore, network virtualization can provide enhanced fault tolerance and damage recovery schemes to provide more reliable network services [37], [38]. Proactive VN provisioning with backup links and nodes can assure fast recovery in case of substrate

network failures. Single substrate link failure can affect all the network services provided to a particular sector of a network. Virtualization can address this issue effectively by dynamically routing the traffic through the backup VNs to provide un-interrupted service.

2.6 Resent Virtualization Projects

Literature on network virtualization is highly diverse since the terms "Network Virtualization" and "Virtual Networks" has given different interpretations by different research groups over two decades. The virtualization projects spans over wide range of networking technologies and moreover, different research groups have used their own set of terminologies to describe characteristics of their works. However, almost all those works attempted to introduce logical separation between networks in different levels, layers and technologies. Hence we categorize virtualization projects in the current literature based on their main virtualization characteristics, in to following categories;

1. Projects categorized based on the Open Systems Interconnection (OSI) model,
2. Projects categorized based on underlying communication technology, and
3. Projects categorized based on granularity of Virtualization.

2.6.1 Projects Categorized Based on the OSI Model

Most of the existing virtualization projects can be categorized based on OSI model layers, which were specifically considered to apply virtualization concepts and create virtual networks. In most of the research works, virtualization concepts are applied in lower layers in the OSI model, since these virtual networks grant more flexibility to the controller or the business entity who manage logical networks. *User Controlled Light*

Paths (UCLP – Ottawa Implementation) [39] attempt to establish virtualization in physical layer and allows implementation of multiple parallel virtual networks on CA*NET 4 network. Parallel virtual networks can have their own link and network layer protocols. As an example, one network can be configured to support IPv4 routed applications, while another parallel network running IPv6 or some other network layer protocol. *UCLP* also allows users to construct new virtualized networks, at any time, without requiring setup or permission of any centralized controlling entity allowing quick and convenient migration of new services and protocols. Furthermore, network resources, instruments and sensors can be integrated to virtual networks using workflow technologies such as BPEL or service oriented architecture (SOA) to create new powerful web services [40]. This device integration in to virtual networks permit users to create multiple workflows that can utilize the network resources conveniently and effectively, granting the applications more control over the network.

VNET [41], [42] is a data link layer operated virtual machine monitoring and overlay network system designed to establish communication links between virtual machines. It provides link layer connectivity to multiple logical networks using Layer 2 tunneling protocol to construct Virtual LAN over local area networks. Running a virtual machine on a remote site imposes number of constraints due to remote site networking security policies. The impact of this situation is intensified as the number of sites increases, and if the virtual machine migration is permitted from site to site. *VNET* provides a mechanism to project remote virtual machine network interface on to the virtual network. Hence, remote virtual machine will not have network access at the LAN in which it has physical

connectivity, but it will allow move network management problems to a virtual network, avoiding site specific policy constraints.

AGAVE [43] is a networking layer based virtual network project, which attempts to solve technical problems related to end-to-end QoS aware service provisions over IP networks. It has provided specifications for developing and validating an inter-domain architecture based on virtual networking concepts. *AGAVE* allows multiple service providers construct and simultaneously execute parallel networks specifically optimized to their end-to-end service requirements. It proposes some lightweight solutions that can be more easily deployed compared to some existing proposals to overcome the limitations of wide area networking. It also defines an open IP connectivity provisioning interface to allow Service Providers to communicate with underlying InPs, for the provision of their IP-based services.

Although most of the network virtualization projects focus on construction of coexisting logical networks at the lower layers of the OSI model, we can find some projects that attempt at addressing virtualization requirements using upper layers such as session and application layers can be found. The purpose of the *VIOLIN* [44], [45] project is to provide the ability to implement multiple logical networks, which has application level isolation, on top of an overlay infrastructure such as *PlanetLab*. The whole *VIOLIN* system is application level software hosted by end hosts. It is comprised of virtual routers, virtual links and virtual machines. The major difference between VIOLIN and application level overlay networks is the re-establishment of system (OS)-enforced boundary between applications and network services. Thus, the system is capable of

running value-added network-level protocols for routing, transport, and management. Each VIOLIN system is a complete computing and networking environment that can be configured to provide network and distributed computation servicers with its own IP address space. Hence, all the computations and communications are strictly restricted within the VIOLIN system. Since all the communication and computation entities are software based, high flexibility allows dynamic addition, deletion, migration and configuration of virtual networks and network entities. Moreover, it supports execution of legacy network services and applications conveniently with additional features and value added services, since software based VIOLIN network entities can be configured to provide tailored services for each network service.

2.6.2 Projects Categorized Based on Communication Technology

Several virtualization research works in the literature have been designed targeting specific networking media or communication technology. They attempted to create virtual networks on top of the targeted networking platform according to their platform specific characteristics and constraints. The networking technologies can be media dependent such as wireless LAN, wireless mobile telephony or protocol dependent such as IP, ATM. Among these platform specific approaches, we consider few sound research works, which attempted to create virtual networks targeting wireless, internet Protocol (IP) and Asynchronous Transfer Mode (ATM) based network technologies.

Although virtualization is not much evolved in the field if wireless networking, few sound research works could be found in the literature [46]-[48]. VN embedding framework for wireless networks was introduced by Park and Kim [47], which divide

wireless node into several wireless communication dimensions, which is almost analogous to multiple access ways in cellular networks such as TDMA, FDMA, CDMA, SDMA, etc. Their framework for wireless virtual network mapping is composed of 3 parts including substrate network, virtual network, mapping structure. First they identified communication dimensions of underlying substrate network, which were important for virtualization and capacity of each corresponding dimension for that particular substrate. Examples for substrate dimensions can be space, frequency, time etc. Then they determined the resource requirement of the virtual networks for the same set of dimensions. If the considered number of dimensions is equal to N, each request is represented by N-dimensional blocks. Finally the mapping of substrate dimensions to corresponding request dimensions is carried out by presenting N-dimensional empty space for substrates and filling it with N-dimensional blocks constructed for each virtual network. In this way the virtual network embedding problem was converted to a brick-work problem with heterogeneous N-dimensional bricks to minimize vacant hole as small as possible.

SplitAP [48] proposes an architecture that addresses the problem of sharing up-link airtime across groups of users by employing network virtualization concepts. It allows wireless network providers to use different algorithms to ensure up-link airtime to be divided fairly across wide range of client hardware. It also permits wireless InPs to use different broadcast domains, different levels of security, support for different protocols, ease of deployment and minimum bandwidth loss for resource partitioning. To provide these functionalities to InPs, *SplitAP* makes use of three key features of virtualization: Abstraction, Programmability and Isolation. Abstraction allows the users to use *SplitAP*

with minimal modifications to client software and hardware, and programmability allows allocating different up-link air time quotes for individual access points. Finally isolation provides logical separation between different wireless user groups allowing the wireless InP to use different airtime control strategies.

X-Bone [49]and Virtual internet [50] are two IP based virtual networking projects. *X-Bone* focuses on the ability of creating a virtual network by automating the deployment and management of overlay networks using encapsulation, and Virtual internet virtualizes all the components of the internet such as hosts, routers and links between them.

The *Tempest* [51] allows InPs to run both standard and non standard control architectures on a single ATM network. VN users can lease a virtual network from the Tempest network operator and customize and control it in the way best suited to their needs. The Tempest provides network programmability and allows new control architectures to be introduced dynamically into the network.

2.6.3 Projects Categorized Based on Granularity of Virtualization

We categorized several prominent research works in the current literature, which exploit virtualization concepts in different granularities. These projects are regarded as noticeable contributions to the state of the art of network virtualization. Some of these projects attempted to make use of virtualization concepts only partially on the node instances and others attempted to utilize the network virtualization concepts fully. *PlanetLab* [52], [53], *GENI* [54] and *VINI* [55] are three major experimental projects focused on node virtualization and has wide geographical span, to support for researches for test-bed experiments.

CABO [56], *CABERNET* [57] and previously discussed *4WARD* [28]-[30] are three recent projects which fully utilize the virtualization concept for separating the network service provider role in to VNP and InP and allow multiple heterogeneous VNs to be run on physical infrastructure. *CABO* is among the pioneering research projects to expose the advantages of decoupling conventional internet Service Provider's (ISP's) role in to infrastructure providers and service providers. Infrastructure providers deploy and maintain network equipment, and service providers deploy network protocols and offer end-to-end services. Its design adopts pluralist philosophy of supporting multiple simultaneous network architectures. To achieve this objective, *CABO* defines virtual network as a network that consists of virtual nodes and links which belongs to the same service provider. Virtual nodes and links are created by subdividing physical node and link resources. Also, virtual nodes can be subdivided into multiple virtual nodes and virtual links can be subdivided in to multiple virtual links. Service providers can install customized routing and other service related protocols on their virtual components. *CABO* is capable of providing number of benefits including better end to end network service, customized protocols, co-location for expanded network presence, testing and deploying new protocols, protection against configuration errors and accountability at every layer. It defines techniques to implement virtual routers, virtual links and scheduling required to construct virtual networks.

4WARD [28]-[30] project describes each player's role of their proposed network virtualization architecture, which slightly differs from CABO and CABERNET. According to the 4WARD model, Infrastructure Providers (InP) are the owners of the physical network resources, and they partition their physical resources in to isolated

virtual slices using a virtualization technology. Hence, InP has some knowledge of the resources allocated to each VN, but not the protocols running inside. The Virtual Network Provider (VNP) finds and aggregates virtual resources from one or more InPs and provides them to Virtual Network Operator (VNO). In other words, VNP does not provide any VN to VNOs, but an empty container which the VNO can install protocols to build VN [30]. VNO construct protocol stack and network architecture independently of substrate network technologies. VNO constructs the protocol stack and network architecture independent of the substrate network technologies and further it manages VN without experiencing any significant impact, as its underlying resources are not physical.

4WARD model also defines lifecycle of a VN up-to its operational stage, in four steps [29]; Design, Provision, Instantiation and Operation. At design stage, VNO needs to explain the required topology, resources and corresponding additional constraints. However VNO can later request to increase or reduce the allocated resources based on dynamic requirements. This initial requirement assessment is forwarded to the VNP at the VN provisioning stage. The main function of VNP is assembling VN according to the VNO requirement, by picking a set of resources that matches to the VN request. For that reason, VNP may forward VN description, completely or partially, to InPs who will setup their substrate resources accordingly and create slices. If the VN slices are created successfully, VNP gets access to the VN slices. At Instantiation stage, management access to the reserved VN slices will be provided to the VNO over an interface similar to serial console or remote control panel to install required operating systems and protocols that facilitate to bring it up to the operational state. At the final VN operation stage, VNO only needs to contact VNP to make modifications to resources and topology such as

expansion and contraction of VN. VNO is capable of performing other run-time operations such as connecting new end users, without any involvement of VNP.

2.7 Summary

In this chapter, we presented some of the successful applications of network virtualization concepts that have been widely accepted by the computer networking industry including VLAN, VPN and Overlay networks. Thereafter, the conceptual and architectural view over network virtualization including the players of the virtualization business model, and the roles of each player were adequately described. Then we discussed the issues addressed by network virtualization, and finally a categorization was made on the recent and well-known research projects in virtualization literature based on their characteristics.

3 Basic Aggregation-based Resource Discovery

3.1 Objective

The main objective of this chapter is to introduce an aggregation-based resource discovery framework to improve the VN embedding. The importance of resource discovery in VN embedding process is discussed and detailed analysis is carried out on the existing resource discovery techniques. We highlight the challenges in the VN resource discovery, and how our current research work attempts to address them in a logical manner. Some of the noticeable limitations of our approach that may lead to inefficient resource discovery are also discussed. We propose an aggregation-based VN resource discovery technique and illustrate the how it meet the challenges.

3.2 Importance of Resource Discovery in the process of VN Embedding

In the network virtualization architecture, VNP is responsible of identifying and selecting suitable infrastructure providers i.e., it processes VN requests to satisfy the VN users. This whole process is known as VN Embedding, and it can be described as mapping of new virtual links and nodes. The VN includes a set of requirements and quality constraints that are considered in VN embedding. The role of embedding is to find and bind the substrate nodes and links that meet the requirements and constraints [34]. As we have discussed previously, VN embedding can be done in three phases [6];

1. Candidate discovery – Find set of suitable Infrastructure providers,

2. Selection – Choose best candidate using optimization, and

3. Binding – Allocate resources from substrate to set-up VN.

Prior to a VN provisioning, the InPs advertise their available resources (network elements such as routers, switches and links and their physical locations), they wish to offer incoming VN requests and also advertise certain dynamic attributes values to VNP. During the resource discovery phase of the embedding process, the VNP tries to determine the potential InPs, based on their available resources. Upon receiving a VN request from VN users, the VNP extract required static and dynamic attributes from the request. Initially, static parameters are compared to select infrastructure providers that satisfy most of the basic requirements. Thereafter, dynamic attribute values are retrieved from the selected InP networks to analyze the embedding potential. In our work, we have chosen available CPU (processing power) as the dynamic node attribute and available bandwidth as the dynamic link attribute. Following the identification of potential candidates, the best candidate is selected in the candidate selection phase and it is requested to allocate resources to the new virtual network in the binding phase.

The resource discovery is a continuous process and particularly crucial, when multiple InPs compete with each other to maximize the VN request acceptance ratio and minimize embedding time. In that case, the resource discovery starts prior to resource allocation to VN requests, in order to assists VNP to decide on the best InP that will satisfy the requirements of each request most appropriately.

3.3 Existing Resource Discovery Techniques

Although the VN discovery has a significant impact in the process of VN embedding, it is striking how it failed to attract much attention in the literature to date. Enhanced business models such as 4WARD [31], [33] considers the existence of a service provider (SP), a virtual network operator (VNO), a virtual network provider (VNP) and a number of infrastructure providers (InPs), and the last two categories are involved in the resource discovery and allocation. The 4WARD model has the merit of distributing tasks related to the separation of services and infrastructure on various players costing more interactions between different players, and this applies to resource allocation, requested by VNP instead of SP in other models.

Authors in [8] point out that, although there are multiple commercial substrate topology discovery applications, there is no adequate work conducted in the field of network virtualization to investigate topological information. They have provided an architecture to facilitate the creation and management of VN and a distributed algorithm that performs virtual and substrate topology discovery. They introduced a platform that consists of three modules as shown in Figure 3.1.

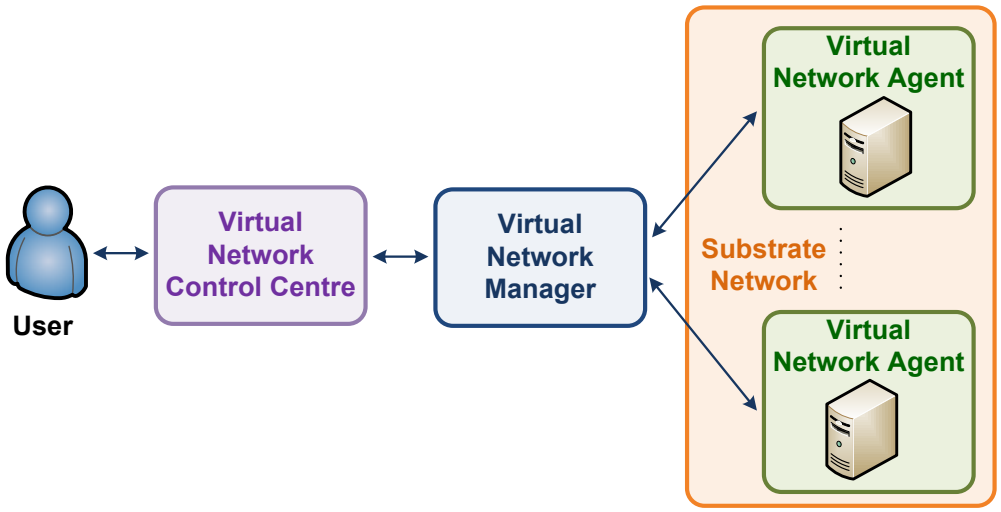

Figure 3.1 Components of Topology Discovery Platform [8]

VN agent module runs on each and every substrate node and sends their local resource information to the managing module. The manager module has three functions:

- Gather information from the agents and send instructions to them,
- Aggregate their information to build the substrate and virtual network topologies and manage static and dynamic resource database, and
- Provide up-to-date information to the control centre about virtual networks and information about the substrate and virtual resources.

Control center module is a graphical user interface which is used as an interface between the user and the managing module. It allows users to create, manage and monitor the VN resources and topologies. The platform performs four main functions. Physical and virtual resource and topology discovery function allows the nodes to discover their neighbors and build network map. Substrate and virtual network monitoring function provide periodic updates of resources' information. The VN mapping function is carried out by manually placing virtual links and nodes by drag and drop operations using the

graphical interface in the control centre. The VN management feature allows changing the virtual resource state, allocating CPU and RAM and deleting the resource or even the full VN.

Proposed distributed topology discovery reduces the processing overhead on manager and assists the user in monitoring physical and existing virtual network topologies. Agent modules communicate in a multicast group and use link-local messaging to discover neighbors in order to create both physical and virtual topology maps. One agent is selected as the designated root in any given network and topology discovery algorithm, which exchange messages between agents using the spanning tree algorithm. Agents exchange information about their local resources and resources advertized by their neighbors.

Nogueira et al. [8] evaluate the performance of their proposed system in a simulation environment by comparing the imposed overhead relative to flooding and probabilistic flooding methods. They also carry out experiments to analyze the performance of centralized topology discovery vs. distributed topology discovery and show that distributed approach provides better results with the higher number of VN requests.

A description and clustering technique for matching and discovering resources for virtual networks is proposed in [6] for the 4WARD model. Its basic function is to select an appropriate InP from a number of possibilities. They state that existing resource description specifications do not sufficiently provide schemas to describe virtual resources to provide network virtualization functions automatically. Therefore, the

motive of their introduced scheme, which is capable of describing virtual resources in terms of properties and functionalities, is to facilitate automated network virtualization.

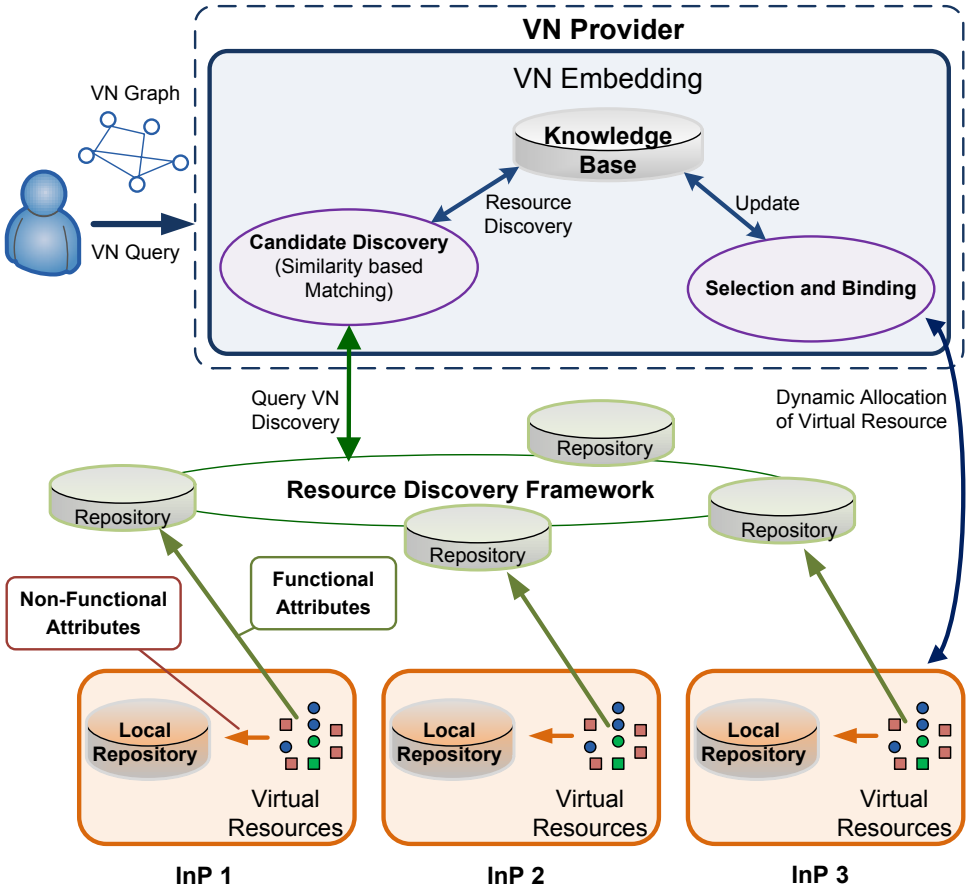

Figure 3.2 Virtual Resource Description and Clustering Architecture [6]

Other contributions of their work are the resource discovery and matching framework and a novel hierarchical conceptual clustering technique, which is used to select the most suitable one out of a set of infrastructure providers. The system architecture of the virtual network resource discovery and clustering framework is shown in Figure 3.2. Each network element is described by two types of attributes: functional and non-functional. Functional attributes are the relatively static attributes which include type of element, execution environment, virtualization tools, and operating system and network stack.

Non-functional attributes are the relatively dynamic attributes, which include performance parameters, capacity, location, cost/price and QoS. The InPs are responsible for advertising and registering their functional attributes in an external resource discovery framework. Due to highly dynamic nature of non-functional attributes, their values cannot be stored and updated in the discovery framework as it imposes significant communication overhead and delay. Therefore, non-functional attributes are updated and stored in repositories local to infrastructure providers. The VN provider has access to external repositories, through Resource Discovery Framework and matches requested resources with available resources. Hence, the discovery process will look only for functional attributes advertized in resource discovery framework repositories. Non-functional attributes will be utilized during selection and binding phases.

Houidi et al. also introduces a similarity based clustering approach to partition the resources in the repositories to minimize latency and overhead during matching. Clustering operation is carried out in four steps in the resource discovery framework;

1. Retrieve functional attributes of each InP network from repositories,
2. Based on similarities, classify them in to clusters,
3. Create descriptions for each cluster, and
4. Construct a tree structure to organize clusters (Dendrogram).

Resource clustering involves arranging resource information in a tree structure called a dendrogram (Figure 3.3). This facilitates the matching process using a simple matching algorithm, which starts at the root of the dendrogram. If the root does not satisfy the resource requirement, the request is rejected. The reverse is, if the root satisfies the

resource requirement, searching process progresses through the branches of the dendrogram until an InP is selected.

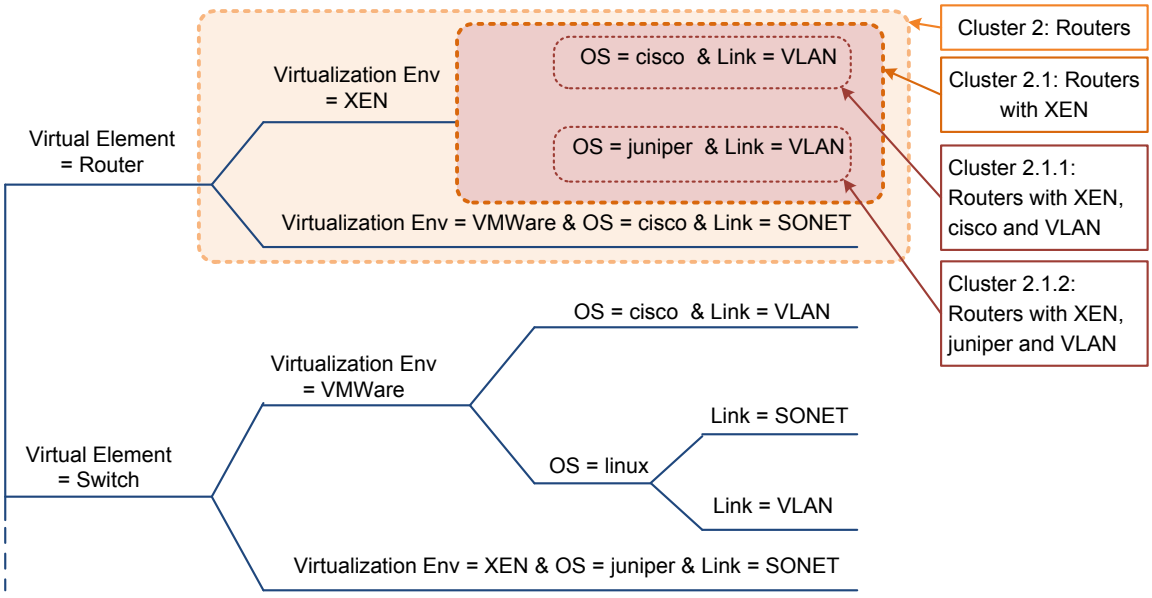

Figure 3.3 Dendrogram Representation of Static Attributes [6]

A local knowledge base is used to maintain the history to reduce load and latency during resource discovery. Finally they evaluated the time taken by matching algorithm to discover candidate resources, and compared the worst case scenario with balanced and unbalanced dendrogram tree structures.

In [9], the same authors propose to advertise the cost of allocating resources in a given InP, which is a non-functional attribute. The same technique is used for resource matching in [10]. The authors in [7] use the same framework as in [6], but propose a different way of organizing resource discovery data, anticipating reduction of searching range and cost, and improve the efficiency of resource discovery. Their proposed system architecture is illustrated in Figure 3.4.

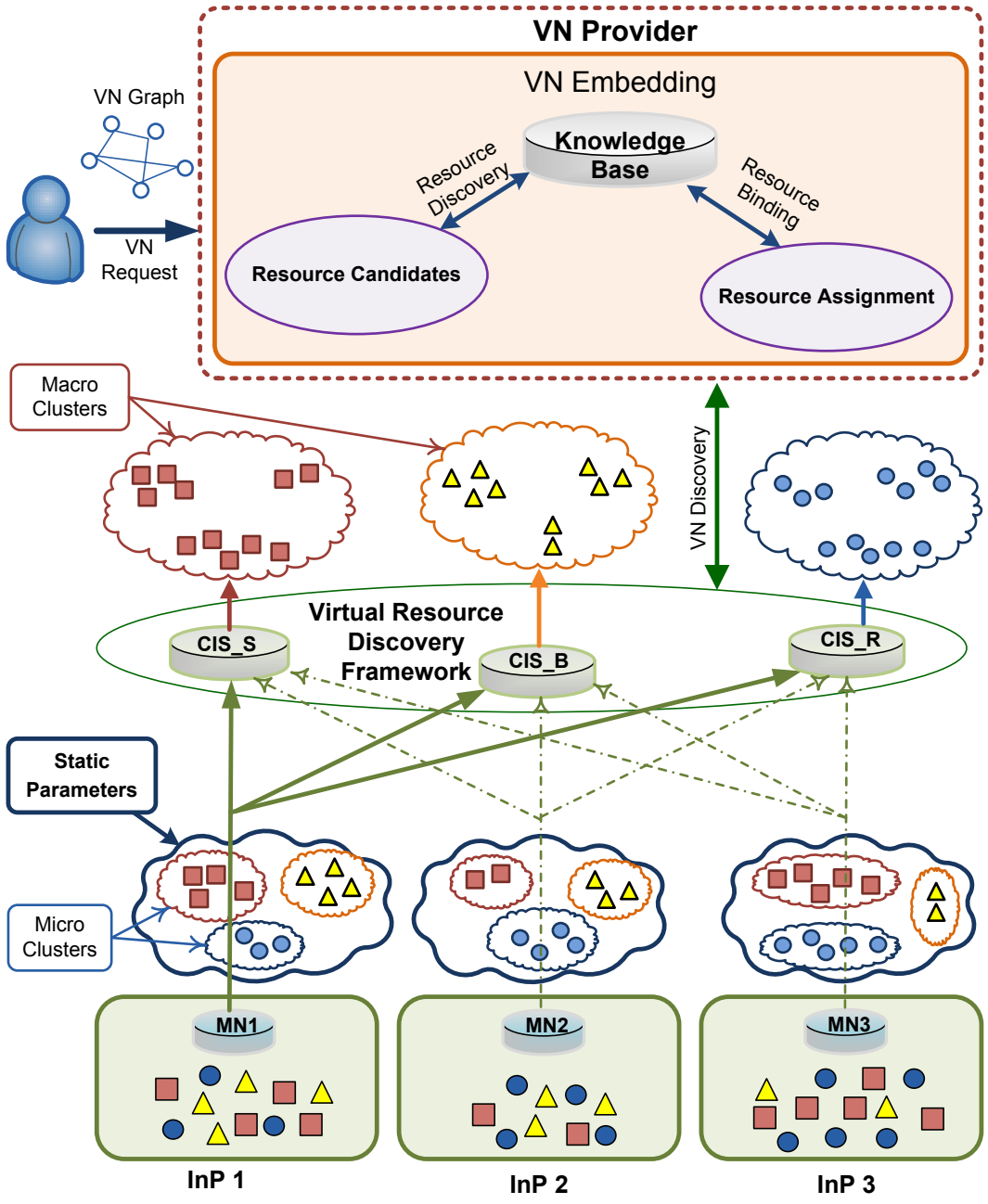

Figure 3.4 System architecture of the framework; Virtual Resource Organization and Virtual Network Embedding Across Multiple Domains [7]

Functional and non-functional attributes are referred to as static and time variant attributes. The authors emphasize that advertising time variant attributes is not reasonable, due to their heavy overhead in communication, operation and maintenance.

They proposed storing time variant attributes (such as the residual capacity of a substrate link) in the management node of each InP. Static attributes are stored in Micro Clusters (MiCs) at the InP level and the MiCs are grouped into Macro Clusters (MaCs) at the resource discovery level based on their root attribute. Cluster Index Servers (CISs) are used at the discovery framework level to improve the searching process in MaCs.

3.4 Limitations of existing techniques

Existing VN resource discovery approaches in literature (such as [6]-[10]) propose solid techniques to identify potential InP networks to a given set of VN requirements. However these approaches share few noticeable limitations. Virtual network creation and management platform which was designed and implemented by Nogueira et al. [8] facilitate manual selection of substrate nodes and links to the host virtual nodes and links. They highlight that the complexity of any embedding algorithm that is designed to perform this function will be highly complex and demand unnecessarily heavy computational power, since it needs to optimize node mapping and link mapping concurrently. Their VN creation platform allows VNP to manually drag and drop virtual elements on the substrate elements to map virtual networks on top of substrate networks. Instead of venturing into a resource demanding optimized solution, their approach was just for a sufficiently satisfactory solution. Eventually, this argument creates several constraints and limitations, as this option is highly inefficient in a scenario where VNP is required to process VN requests which comes rapidly with relatively short life time. Parallel VN mapping becomes impossible with this approach, since each VN is required

to be mapped manually with human intervention. Further, VN re-optimization is almost impossible to do manually as it causes significant breakdown times and delays.

Nogueira et al. [8] introduces a distributed topology discovery algorithm, which performs virtual and physical node discovery to construct virtual and physical topology diagrams. The objectives of the work is to discover virtual network topologies becomes questionable since after embedding, VNP should already possess the information required to construct VN topology in its local knowledge base. Generally, VN request appears as a set of node requirements and set of interconnecting link requirements. During the embedding process, the VN topology will not be altered, since the VN user is the entity that will typically decide the topology requirements based on its connectivity or business objectives. Therefore, the approach suggested by the authors can only be useful when the VNP or InP required to confirm the embedding or to find out any dynamic changes made to VN topology has taken place as planned.

Resource discovery approach introduced in [6] can be identified as a substantial contribution towards the network virtualization resource discovery literature. However, they solely consider functional attributes to determine the most suitable InP at the discovery phase disregarding the non-functional attributes, when the discovering resources become a noticeable limitation of the framework. The reason to disregard non-functional attributes is their time varying nature, which claims demand real-time monitoring cost including higher processing overhead and significant bandwidth overhead for monitoring messages. However, without information on the non-functional

aspects, satisfying requests becomes uncertain, since non-functional attributes play a critical role in selecting substrate resources during matching and binding phases.

Additionally, scenarios such as multiple substrate resources that belong to the same InP sharing similar functional characteristics and multiple InPs having resources with same functional attributes makes the proposed discovery approach incapable to provide useful a output. In that case, the VNP has to deal with all the InPs that have the same set of functional attributes (since pricing and location are not advertised) leading to additional delays and computational costs.

Virtual resource organization framework provided in [7] introduces an improvement to the same fundamental system architecture used in [6]. The main contribution of this research work is the introduction of a resource rearranging technique that divides static parameters of InP resources based on the type in the resource discovery framework.

3.5 Improvements of proposed method over existing techniques

One of the noticeable limitations of the existing network virtualization resource discovery techniques in the existing literature is circumvention of the necessity of utilizing dynamic network attributes, which plays a vital role during the virtual resource embedding process. The cost of accurate monitoring of these dynamic variables is notably high and there is always a tradeoff between accuracy and cost, when monitoring the required network parameters for resource discovery. Hence, most of the existing research works such as [6]-[10] either assumes complete availability of this information or consider relatively static network parameters. Generally, candidates discovered using static parameters have higher tendency for rejection during candidate selection and binding

phases, which also ultimately result in higher processing overhead and delays. The dynamic nature of certain crucial network attributes required for VN mapping and the overhead incurred by continuous monitoring of these attributes stands as one of the challenges in discovering substrate resources to embed virtual networks.

We propose an aggregation-based discovery approach, which can extract the values of crucial dynamic network attributes to use them to enhance the resource discovery process. From the network graph, we construct a spanning tree to perform incremental aggregation. Values of dynamic network attributes of nodes such as available processing power, memory, and link dynamic attributes such as available bandwidth are determined locally at each node and aggregated towards root of the aggregation tree. Each element in the tree needs to forward a single message to its parent node, to send both, its local node parameters and aggregation values it received from its children. This reduces the monitoring overhead considerably. With dynamic network attribute values, resource discovery decisions are more accurate.

We also introduce a VN filtering scheme to reject VN requests in the resource discovery phase. In the existing network virtualization literature, VN requests are rejected mostly during binding stage. This leads to waste of processing power and results in decision making delays. Initial filtering of VN requests allow VNP to promptly inform the VN user that certain requirements cannot be provided with the available resources of InPs who are currently in contract with the VNP. In addition, novel InP sorting technique is employed to queue the InPs based on their individual potential to embed the received VN request.

We further enhance the accuracy of resource monitoring by increasing the controllability over the cost verses accuracy tradeoff by employing a histogram based aggregation approach which we discuss in detail in Chapter 4.

3.6 System Architecture

In this section, we present our resource discovery approach termed as *ADVNE (Aggregation based Discovery for Virtual Network Environments)*. The aim of *ADVNE* is to address the noticeable issues that occur, when disregarding non-functional (non-static) attributes in selecting potential infrastructure providers from a pool. Static attributes will be used initially to select infrastructure providers who satisfy basic requirements of the VN request. We do not intend to discuss how to manipulate static attributes, since clustering and dendrogram based resource discovery techniques introduced in the previous contributions (e.g., [6], [7] and [9]) has provided appropriate solutions. In our research work, the main objective was to introduce an efficient and cost effective methodology to utilize dynamic attributes to augment the VN resource discovery process.

In simulations, we only considered available processing power and available bandwidth as dynamic node and link characteristics respectively. But, other node and link dynamic attributes such as available memory, data transmission delay etc. can also be monitored by following the same approach. The system architecture of the *ADVNE* can be illustrated as shown in Figure 3.5.

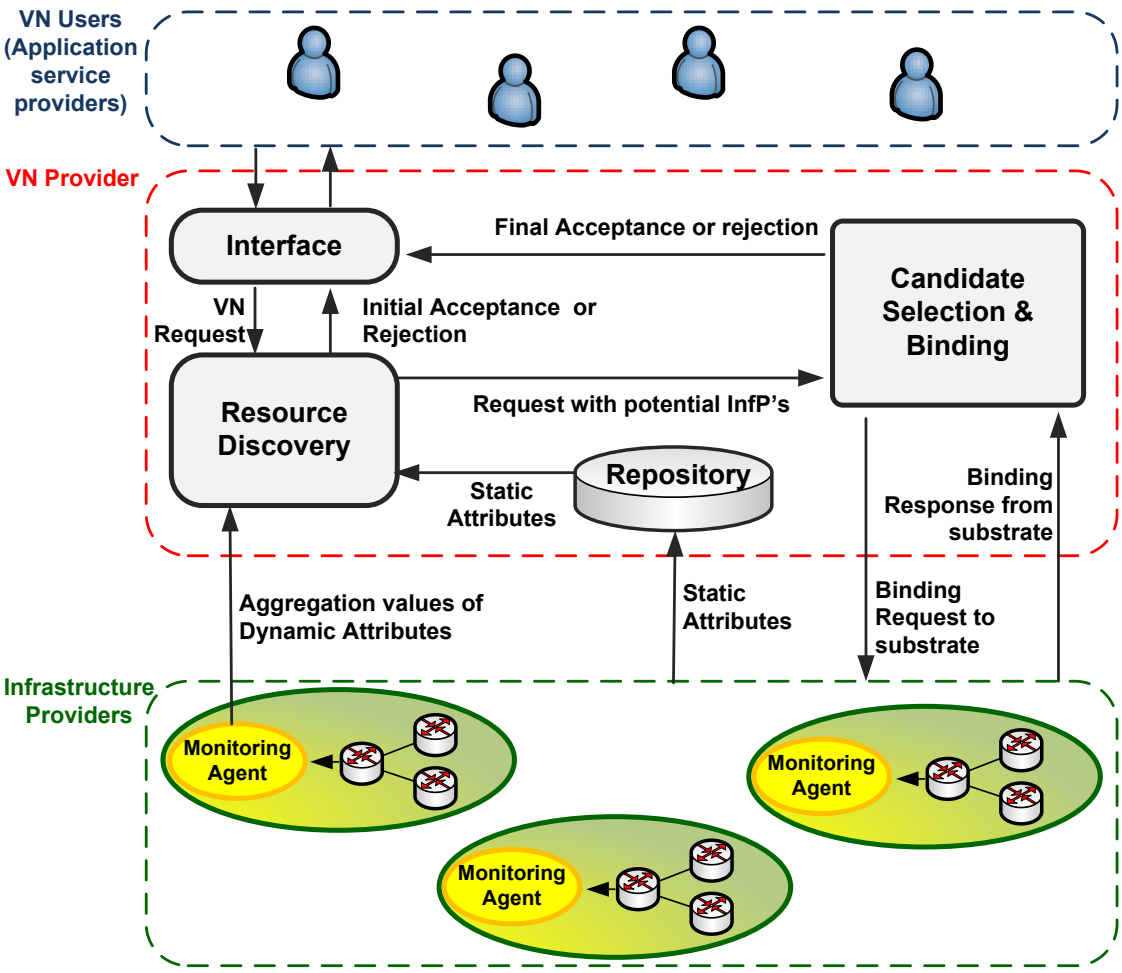

Figure 3.5 Aggregation-based VN Resource Discovery (ADVNE) System Architecture

Proposed network virtualization platform consist of three players;

- VN Users: Application service providers who require virtual networks to provide end-to-end business solutions or services such as *IPTV*, *VOIP* etc. VN users enter in to contracts with VNP requesting for virtual networks to have end-to-end connectivity with certain amount of processing power and bandwidth resources. They have to provide their required VN in the form of Cartesian coordinates of virtual node locations, required processing power in each location,

required connections between nodes and bandwidth requirements in each virtual links.

- Virtual Network Provider (VNP): Intermediate business layer that communicate with infrastructure providers and VN users in order to provide VNs and holds the responsibility of coordination and management of *ADVNE*. We have merged the functions of VNP and Virtual Network Operator (VNO) in conventional network virtualization business model introduced in [31], [33]. VNP accepts requests coming from users based on their static attribute values available in the repository and dynamic attribute values acquired from InPs, then selects and sorts InPs based on their embedding potential and forward VN requests to InPs.
- Infrastructure Providers (InP): Network resource suppliers in the system provides VNs. InPs in *ADVNE* need to install monitoring agent which allows VNP to calculate aggregates of dynamic attributes of resources. Moreover, they are required to disclose the static attributes of network resources that are available to host virtual resources in the VNP's repository. According to the conceptual definition of network virtualization, InP can be either a physical network owner or another virtual network owner.

Functional components of the above roles in *ADVNE* platform are used to perform resource discovery and embedding decision making functions are explained below;

- Monitoring Agent: Each infrastructure provider is required to execute an instance of monitoring program on top of it called Monitoring Agent, which

performs the task of aggregating values of required dynamic attributes. Monitoring agent construct an optimum spanning tree for the InP network graph to perform aggregation function. It also forwards aggregation values to the resource discovery module in VNP at its request.

- Repository: Infrastructure providers bear responsibility of advertising static network attributes in a common repository that can be accessed by VNP. Therefore, sophisticated description of static attributes is assumed to be readily available for VNP to perform static resource matching. As emphasized in [6], infrastructure providers can disclose information of their network resources up to a preferred degree by maintaining their privacy and security requirements. Static attributes in the repository can be structured following the description and clustering techniques introduced in [6], [7] and [9] making it possible for the VNP to carry-out static attribute matching efficiently.
- Interface: According to the proposed architecture, VN requests from users arrive at the VNP's interface with a description of node/edge requirements /constraints such as CPU capacity and link bandwidth. Interface time-stamp the request and forward request to resource discovery module. Conversion of the request in to a readable format (node coordinate and interconnecting link format) is also a task of the interface. Moreover, it transfer the final VN acceptance or rejection decision coming from candidate selection and binding module to the VN users.
- Resource Discovery: Main functional component in the *ADVNE* system is the resource discovery module. Based on static and dynamic network attributes,

discovery module seeks the best matching InP for a VN request coming from the VN users through the interface. Resource discovery functions will be discussed in detail in the next section.

- Candidate Selection and Binding: Selecting the best candidate and sending a resource allocation request to InP are the functions of this module. Selecting the infrastructure providers with highest embedding potential is different from the candidate selection function carried out by this module. The process of candidate selection is highly resource consuming, since it selects only one InP network by analyzing all the link and node resources to find the optimum matching substrate resources to host virtual resources.

When a VN request is arrived to the interface of the VNP, it is forwarded to the ***Resource Discovery*** module. The resource discovery module retrieves static attribute values of resources from ***repository*** and aggregation values of dynamic attributes from monitoring agents in InPs. Static attributes will be matched initially and successful InPs will be considered for dynamic attribute matching.

Based on the available InP resources, VNP makes a decision whether to accept or reject incoming requests. This operation is known as initial filtering and it will be described later in this chapter. After an initial filtering, InPs that are potentially able to embed the incoming VN request are selected and sorted in descending order based on their embedding potential. This sorted order of InPs will be forwarded along with the request description, to the ***Candidate Selection and Binding module***. Candidate selection and binding module will forward the request description to the first infrastructure provider in

the sorted order. If embedding is returned with a failure from the first substrate, request will be forwarded to the second InP in the sorted list. This process iterates until the VN request is embedded or none of the selected InPs is embed, resulting VN request is rejected.

3.7 Continuous Monitoring of Dynamic Attributes

Continuously monitoring values of network attributes such as bandwidth, delay, CPU and memory of network links and nodes are significantly resource consuming. Since these resources are taken from the network itself, monitoring cost appears as an overhead burden to the network. On the other hand, accurate embedding decisions demand detailed information of network resources including highly dynamic attributes which require continuous monitoring. An existence of tradeoff between accuracy and cost is obvious, since high accuracy causes a high overhead cost, while low estimation accuracy yields less overhead.

Dynamic Network resource monitoring for virtualization can be carried out employing either centralized, decentralized or distributed schemes. Centralized network monitoring has been highly criticized for its weak responsiveness and accuracy. In addition, the static centralized monitoring is poor in scalability, since centralized management results in bottlenecks due to higher monitoring packet concentration around the managing station. Moreover, management station is a single point of failure in the whole monitoring system.

Echo protocol [58] is an alternative distributed network monitoring approach which can be also used for distributed configuration and resource discovery. Its execution pattern

resembles an expansion and contraction of water ripple, starting and ending at the root node of the network spanning tree. During expansion, explorer messages are forwarded and during contraction, echo messages are forwarded towards the root with aggregation values of the monitored network variable. Finally, the total aggregation value of the monitored variable is available at the root.

Generic Aggregation Protocol (*GAP*) [59], [60] is an extension to echo protocol which addresses few weak aspects of echo model. Echo protocol is incapable of running several echo operations simultaneously to monitor multiple network variables concurrently in the management plane. Also, the echo protocol is not robust for the changes to the network graph such as node churn and failures. *GAP* executes on a connected, bidirectional network graph and capable of providing continuous estimates of network variables while dynamically adapting to node churn and failures. Main functional improvement of *GAP* over echo model is an event driven, push upward mechanism, which replaces ripple expansion and contraction mechanism in echo protocol. It provides distributed computation and management by running management process on each network node to locally compute aggregation values and communicate with the parent and child management processes over management overlay network.

If local aggregation value of a monitored variable exceed filter width (go over or below than a pre-defined upper and lower limits) in any network node, it will forward the aggregation value to its parent node in the network spanning tree. Figure 3.6 shows an example spanning tree constructed by *GAP* to continuously monitor values of a dynamic network attribute. Local values of the dynamic network attribute corresponding to nodes

a, b, c, d and *e* are *V(a), V(b), V(c) V(d)* and *V(e)* and the aggregation values are *A(a), A(b), A(c), A(d)* and *A(e)* respectively. If the value of the monitored variable in node *a* exceeded the filter width, aggregation trigger message will be forwarded to node ***c*** along with the value *A(a)***.** In leaf nodes such as *a* and *b*, partial aggregation value will be equal to local attribute values and in intermediary nodes such as *c* and *d***,** partial aggregate value is the sum of aggregate values coming from children and its own local attribute value. Hence, partial aggregate value of node *c* will be; *A(c) = A(a) + A(b) + V(c).* Finally, the sum of all the local values of the considered network variable will be available at the root of the spanning tree.

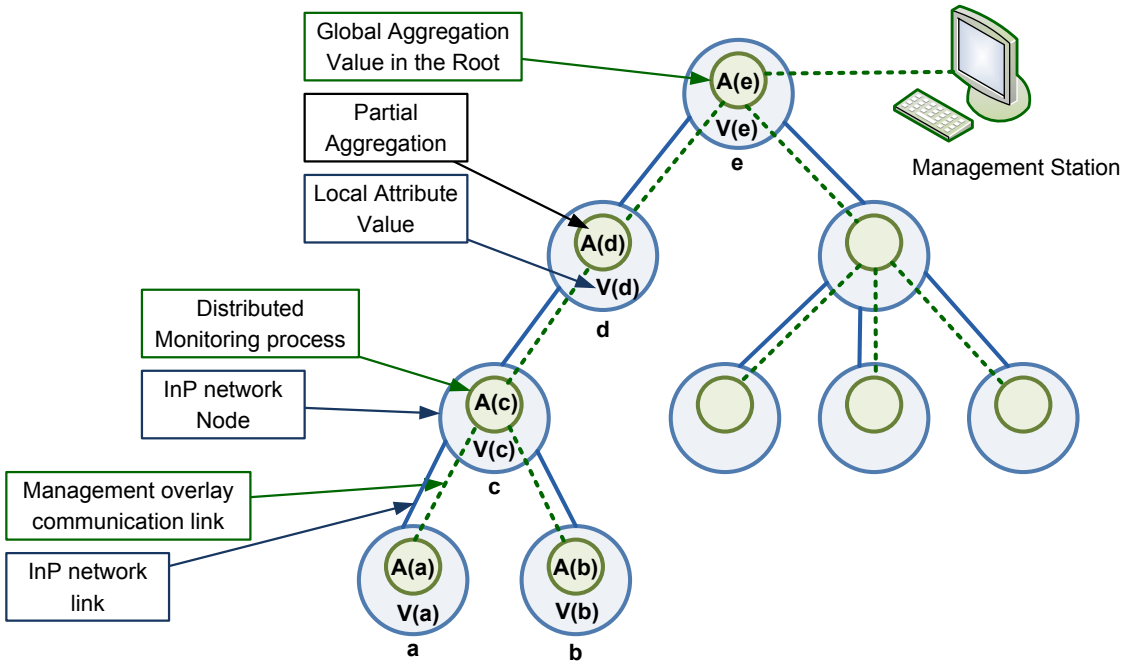

Figure 3.6 Incremental Aggregation over Network Spanning Tree [60]

We make use of the network monitoring techniques introduced by the GAP to find total, average, maximum and minimum values of bandwidth and CPU capacity of substrate

network resources by calculating aggregates instead of monitoring the values of each attribute individually. This will reduce the overhead when discovering network resources of substrate networks. Aggregates assure significant reduction of the number of messages passed over the network and avoidance of monitoring packet congestion around the centralized monitoring node. This will make it possible to minimize the monitoring overhead of dynamic attributes of the network, and hence, consider dynamic attributes in performing resource discovery.

Monitoring agents in each InP construct optimum spanning trees for their network graphs. We have incorporated the functions of all three monitoring entities namely; distributed monitoring process instances that installed in each InP network node, overlay used to communicate between monitoring process instances and management station, in to a single monitoring agent installed in an InP network.

After constructing the tree, a distributed pull-based approach is used to calculate aggregation values at each node of the tree by dividing the tree into depth levels and starting from the nodes at highest depth. For dynamic node attributes such as available processing power, nodes are capable of calculating their local aggregation values locally. But for link attributes such as available bandwidth, corresponding aggregation value is calculated in one of the connecting nodes. Initially, aggregation values are calculated in every substrate node and the values will be forwarded to the resource discovery module by the monitoring agent. Thereafter, aggregation values are calculated for individual substrates only after a successful VN embedding or successful VN departure. Since aggregates are calculated each time after a successful embedding or departure of a VN,

resources used to calculate aggregates do not impose a significant overhead on InPs. In Section IV, we present and analyze the required time to construct optimum spanning trees and calculate aggregates.

3.8 Initial Request Filtering

Clustering approaches in [6], [7] and [9] perform filtering of VN requests by rejecting the requests which do not comply with the description of the root of the dendrogram. Requests satisfying this initial requirement will go along the branches of dendrogram and return a cluster that gives best match for the required resources. However, these clustering approaches only capable of guaranteeing that VN requests satisfy general static attributes such as VN type, virtualization environment, operating system, network stack and link type. It is obvious that although all these requirements are satisfied, possibility of rejection still higher at candidate selection and binding due to mismatching of dynamic requirements such as available CPU capacity and link bandwidth.

Authors in [6], [7] and [9] have encompassed this point by arguing that maintaining up-to-date values of real-time dynamic (non-Functional or non-Static) attributes require heavy monitoring process between discovery framework and substrate nodes. Disregarding the fact that dynamic attributes lead to higher selection and binding delays and inefficient utilization of resources, since a VN request fall into a particular cluster need to be forwarded to every InP, who meets the description of that cluster, requesting their responses.

We use aggregation values to reinforce the initial matching mechanism by filtering VN requests over the set of conditions that assure each VN request satisfies certain set of

basic requirements. For each VN request, Maximum CPU requirement, Total CPU requirement, Maximum BW requirement and Minimum total BW requirement will be calculated. After that, each of these requirements will be compared respectively with Maximum CPU aggregation, Total CPU aggregation, Maximum BW aggregation and Total BW aggregation, obtained from substrates. Calculated attribute values of a particular VN request will be compared with aggregation values of all substrates. If a particular substrate fails to satisfy either of these conditions, VNP will simply ignore that substrate and will consider the next substrate in the list. For a particular VN request, if all the selected substrates in the pre-matching phase fail to satisfy the conditions, that particular VN request will be rejected.

3.9 Sorting Selected Infrastructure Providers

Sorting InPs according to a criterion which is proportional to their potential of embedding a new VN request will assist the VNP to significantly reduce the InP selection process latency. We have chosen average path bandwidth when sorting substrates because the capacity of a substrate network to embed VNs is mainly determined by the ability of its paths to embed virtual links.

In a substrate network $s \in S$ where S is the set of all substrate networks, let there be nodes u and v such that $u \neq v$ and $u, v \in U$, where U is the set of all the nodes in the substrate s. The shortest path between two nodes u and v in s is noted p_{uv}^{s}. The bandwidth of path p_{uv}^{s} is defined as:

$$Bw(p_{uv}^{s}) = \min_{l \in p_{uv}^{s}} Bw(l) \tag{3.1}$$

where $Bw(l)$ is the bandwidth of link l in the shortest path p_{uv}^{s} .

The average path bandwidth of node u is

$$AvgBw(u) = \frac{\sum_{p_{uv}^{s}, \forall v \in U} Bw(p_{uv}^{s})}{n-1} \tag{3.2}$$

And Aggregation value of average path bandwidth in substrate s is

$$AggBw(s) = \sum_{u \in U} AvgBw(u) \tag{3.3}$$

The aggregation process is carried out for average path bandwidth over optimum spanning tree constructed for a substrate and final aggregation value is obtained from the root of the spanning tree. Substrate networks will be sorted in a queue based on their aggregation value shown in Equation (3.3).

The first substrate in the queue that has maximum of average path bandwidth value will be given priority when forwarding for embedding. In case of first substrate fail to embed the request, request will be forwarded to next substrate in the queue.

3.10 Summary

The resource discovery is an important phase in the VN embedding process, since efficient resource discovery scheme can assist the VNP to reduce the resource wastage by finding most suitable InPs in advance, to forward VN resource request specifications. The

InPs also can benefit from an efficient resource discovery scheme that can allow them to identify VN requests that do not overstress their networks. The VN users also can get an important feedback on their VN request in the initial stage of the embedding process. Hence, we recognize an efficient resource discovery scheme that can improve the overall efficiency of the VN embedding process.

In this chapter we discussed the pros and cons of existing resource discovery schemes available in the literature. Then we have presented our proposed solutions to those challenges and our resource discovery framework. We also illustrate the components and operation of the proposed network virtualization resource discovery framework.

4 Vector-based Resource Discovery for Network Virtualization

4.1 Objective

The aggregation-based resource discovery technique discussed in Chapter 3 improves the overall efficiency of the VN embedding process but, the basic aggregation technique used in the framework contains several limitations inherent to the basic aggregation which requires further development to make it a fully performing devise.

In this chapter, we investigate the limitations of basic aggregation approach developed in Chapter 3 to make resource discovery decisions. Then we introduce a vector-based aggregation technique that can be used to further enhance the performance of resource discovery framework. We also highlight the architectural improvements introduced on the system architecture of the resource discovery framework illustrated in the Figure 3.5. Then we formulate the vector based aggregation technique and attempt to prove mathematically how the proposed vector-based aggregation technique amplify the controllability over the tradeoff between costs of monitoring vs. accuracy of the monitored output. Finally we explain the improvements we introduced in to the initial filtering and substrate sorting functions, utilizing the monitored results obtained from vector based aggregation technique.

4.2 Weaknesses of the basic aggregation technique

When the common aggregation technique is employed to monitor a particular dynamic network resource attribute, the final aggregation result at the root of the network spanning tree will only contain the total aggregation value of that attribute in the whole network. With minor modification to the aggregation algorithm, maximum and minimum values of that variable can be acquired without imposing significant additional overhead. However the effectiveness of having limited information of network resources such as total, average, maximum and minimum values of available processing power and bandwidth can lead to a wide criticism due to several reasons.

Aggregation results do not disclose sufficient details to find the distribution pattern or deviation of network resources. In situations particularly only with few nodes with higher resource availability and relatively large number of nodes almost reached at their maximum capacity, aggregation will provide total and average resource availability completely ignoring uneven distribution of resources. Under such scenarios, dependence on network virtualization resource discovery decisions based on aggregation values proved to be highly inefficient.

Controllability of the tradeoff between accuracy over cost is arguably low in basic aggregation. However, the introduction of filtering scheme has allowed *GAP* [59] to address this issue up to a certain degree because it increases the accuracy of the monitored attribute values by reducing the filter width, which will results in higher sensitivity of the resource monitoring system at a higher processing cost and messaging overhead. Though the frequency of supplying monitoring results get increased, the degree

of controllability is still insufficient, as system is capable of providing only total and average values of dynamic network attributes.

4.3 Vector-based Resource Discovery Framework

The controllability of accuracy can be significantly increased by following a vector based approach when dynamic network attributes are monitored. We have improved the previously introduced aggregation based virtualization resource monitoring scheme by replacing the basic aggregation technique using a vector-based aggregation technique. Enhancements to our work discussed in this chapter have been inspired by the histogram based network monitoring approach introduced by [61].

The enhanced system architecture is illustrated in the Figure 4.1 with noticeable improvements to resource discovery module. The main objective of the modifications is to allow resource discovery module to perform pre-processing which construct request vector for each VN request and take resource discovery decisions by comparing request and aggregation vectors.

Similar to *ADVNE* architecture, monitoring agent installed in each InP network construct BFS spanning tree for the InP network graph. In an event of a new request arrival to the resource discovery module through the VNP's interface, static attribute matching will be performed with the available static attribute information in the repository. If the static attributes matching is successful, the pre-processing component in the dynamic attribute matching module will compute vectors corresponding to the VN request. Meanwhile, the dynamic attribute matching module will send a request to the monitoring agent of the InP which has the best matching static attributes.

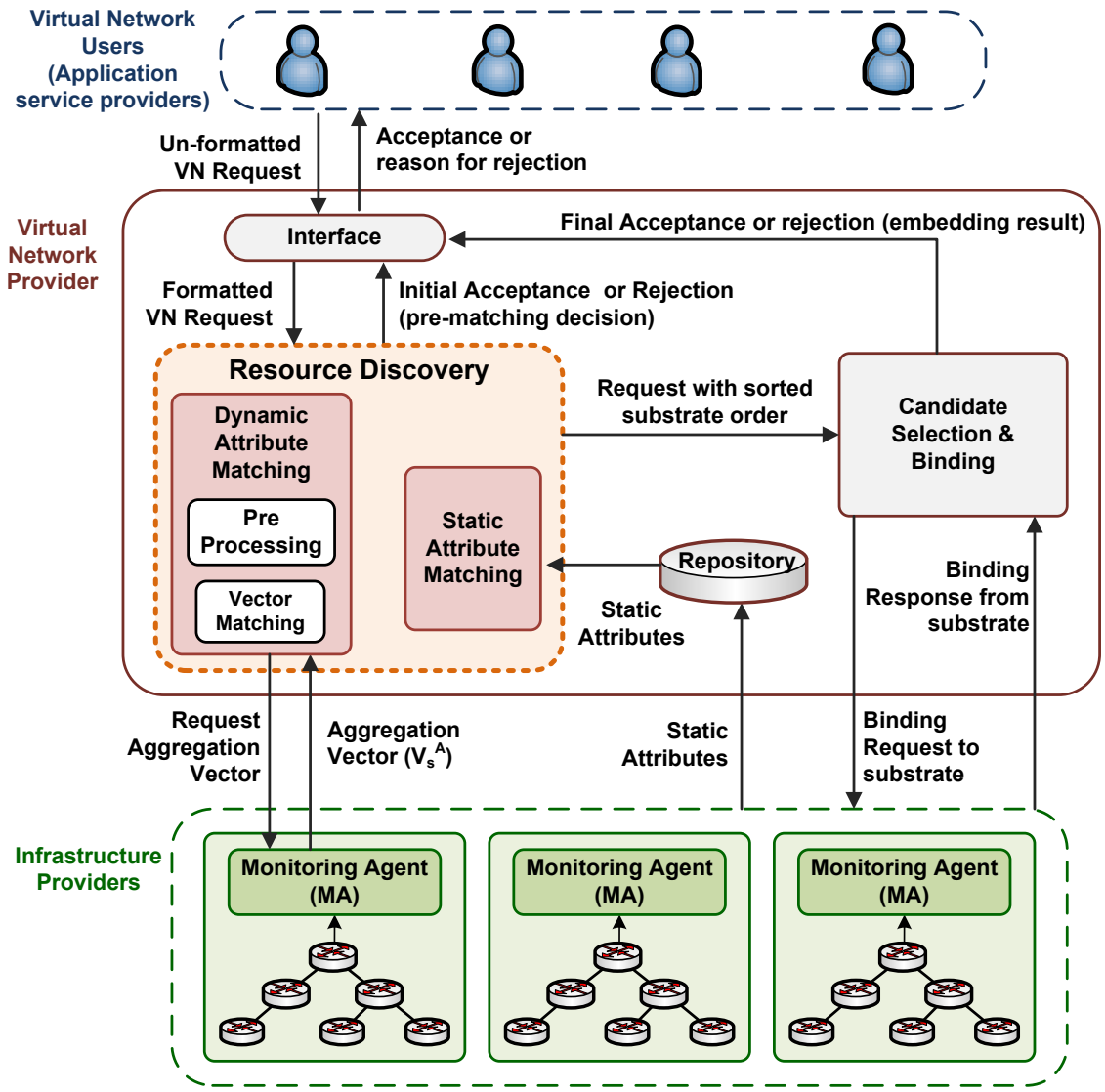

Figure 4.1 Enhanced Resource Discovery Framework with Vector based Aggregation

Monitoring agents can perform aggregation and construct aggregation vector either at departure of a request or when it receives an aggregation vector request from VNP (online). Both these options have their pros and cons in terms of delay, overhead and accuracy. To further increase the controllability over the tradeoff between accuracy over cost, we modified the mobile agent to perform aggregation-based on the decision of

VNP. Hence, VNP can determine whether to follow online or offline approach during run-time, to acquire aggregation vectors.

In both online and offline approaches, monitoring agents construct aggregation vectors for each local dynamic attribute by normalizing the attribute value with respect to a normalizing constant which can also be defined by the VNP. After normalizing, the attribute value will be round-off to the floor (lower bound) value. The number of elements (positions) in the normalized vector will be equal to normalizing constant. The procedure of computing normalized vector for local attribute value of a node is formulated in the next section.

After obtaining aggregation vector from the InP, the vector comparing component in the dynamic attribute matching module compare the aggregation vector received from the InP with request vector calculated by pre-processing component. If the aggregation vector is weaker than the request vector, corresponding InP network will be ignored and next InP will be considered. If the aggregation vector is stronger than the request vector, corresponding substrate will be placed in a priority queue, which sort the selected substrates based on their embedding potential. If none of the aggregation vectors (constructed for each substrate) are stronger than the request vector, request will be returned to the user with the pre-matching failure message.

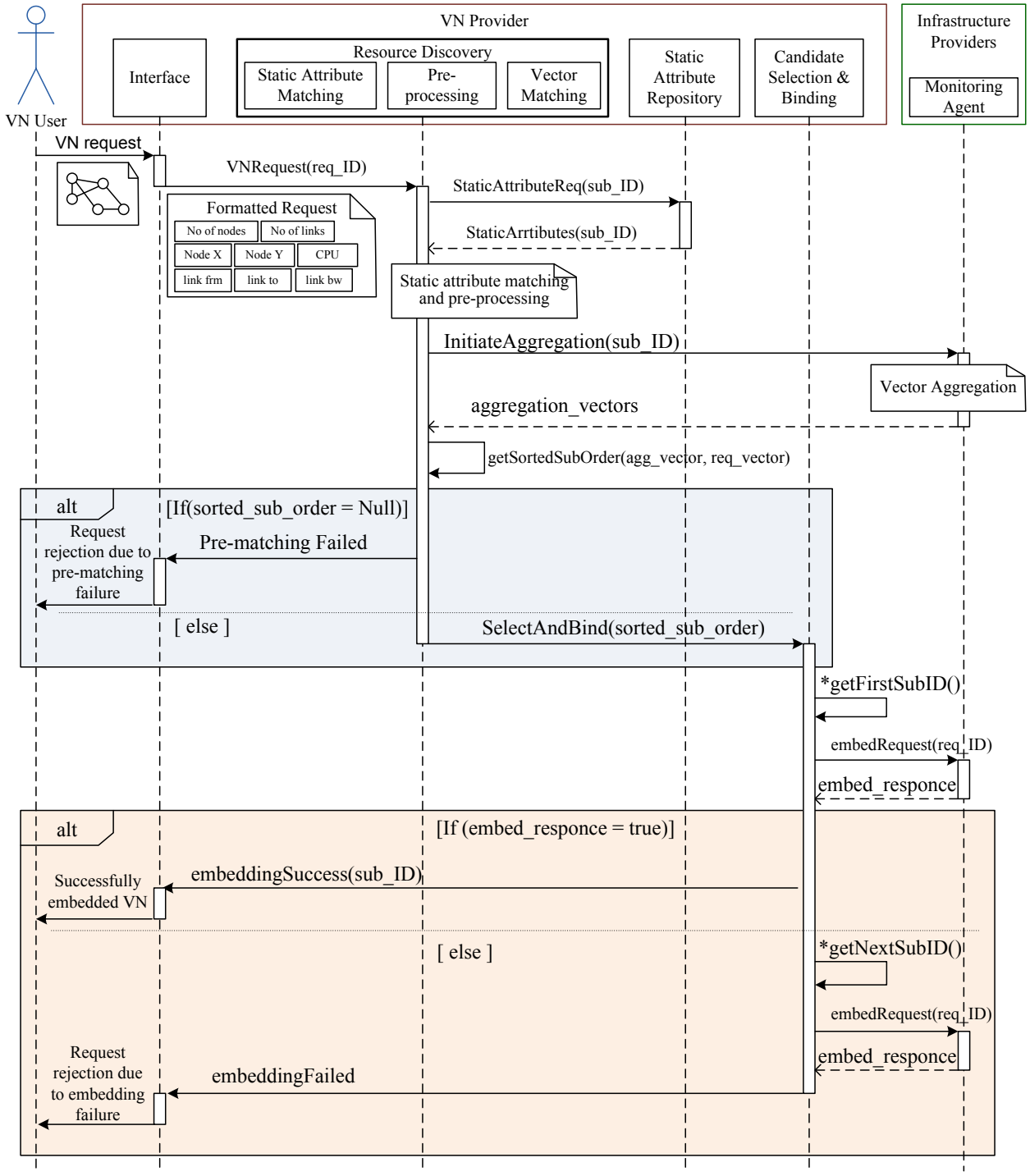

Figure 4.2 Operational Sequence of Vector-based Resource Discovery Framework

After comparing all the aggregation vectors, the sorted order of substrates will be directed to the selection and binding module along with the VN request information to forward it to the first substrate of the sorted order. If the response for the binding request comes as a binding failure, the request will be forwarded to the next substrate in the sorted order. If none of the substrates in the sorted order is capable of binding the request, it will be returned to the VN user along with the binding failure message.

Sequence diagram in Figure 4.2 illustrates this whole embedding process for online aggregation, highlighting the interactions between each component of the framework. If the offline aggregation technique is used instead of online aggregation, "*initiate Aggregation( )"* message will be sent to the monitoring agent after successful embedding or departure of a new request.

4.4 Vector-based Aggregation Technique

The objective of the introduced vector-based aggregation mechanism is to provide the VNP with some powerful tools to obtain substrate network information that are critical when taking resource discovery decisions, while managing the monitoring costs. Computation of aggregation vectors for each substrate network and construction of request vectors are mathematically formulated in this section to analyze the ability of handling the round-off error injected to aggregation vectors at each aggregation step.

Let S be the set of substrate networks and N_s be the set of nodes in substrate s where $s \epsilon S$. For node $n \in N_s$, the local value of the monitored dynamic attribute at time t is denoted by $l_s^n(t)$. The maximum integer value that can be taken by $l_s^n(t)$ is defined as:

$$L = \left\lceil \max_{n \in N_s, s \in S, \forall t} \left(l_s^n(t) \right) \right\rceil$$

For the local dynamic attribute value $l_s^n(t)$, a vector $V_n^{l_s}(t)$ is constructed by normalizing the $l_s^n(t)$, with respect to a constant u. The cardinality of the vector $V_n^{l_s}(t)$ will be given by $k = \frac{L}{u}, \forall n_s \in N_s, \forall s \in S$ and the value for the normalization constant u is defined by VNP ensuring the satisfaction of the condition $k \in \mathbb{Z}^+$. The i^{th} element of $V_n^{l_s}(t)$ is given by $v_i^{l_s^n}(t) \in V_n^{l_s}(t), \{i \in \mathbb{Z}^+ | 1 \leq i \leq k\}$. The mean value w_i for the interval between upper bound ui and lower bound $u(i-1)$ is given by $w_i = \left(\frac{u(i-1) + ui}{2}\right)$.

For any node, the local vector for monitored attribute $V_n^{l_s}(t)$ can be given by:

$$V_n^{l_s}(t) = \left\{ v_1^{l_s^n}, v_2^{l_s^n}, \dots, v_k^{l_s^n} \right\}$$

where $v_i^{l_s^n}(t)$ is defined as:

$$v_i^{l_s^n}(t) = \begin{cases} 1 & if\ i = \left\lceil \frac{l_s^n(t)}{u} \right\rceil \\ 0 & otherwise \end{cases}. \quad (4.1)$$

There will be a single normalizing constant u for all the InP networks and the same value will be used with the VN requests when calculating corresponding vector for the value of the attribute. When the VNP sends a request to InP asking for final aggregation vector for a specific dynamic attribute, the monitoring agent trigger aggregation process by sending messages to leaf nodes of the spanning tree over the management overlay to start calculating partial aggregation vectors. Partial aggregation vectors of the leaf nodes are

identical to their local attribute vectors. After constructing the partial aggregate vectors, each leaf node of the spanning tree forward their locally calculated partial aggregation vector to its parent. The parent node calculates partial aggregation vector by element wise addition of local attribute vectors with partial aggregation vectors received from its children.

Let T_n be the sub spanning tree rooted node n and C_n be the set of immediate children of the node n. For the node n we denote i^{th} element of the partial aggregation vector $V_n^{c_j}(t)$ received from its j^{th} immediate child $c_j^n \in C_n$ as $v_i^{c_j^n}(t)$. Thus the $V_n^{c_j}(t)$ can be written as $V_n^{c_j}(t) = \left\{ v_1^{c_j^n}, v_2^{c_j^n}, \dots, v_k^{c_j^n} \right\}$. Similarly partial aggregation vector computed in node n can be written as $V_n^{A_s}(t) = \left\{ v_1^{A_s^n}, v_2^{A_s^n}, \dots, v_k^{A_s^n} \right\}$ where $v_i^{A_s^n}(t)$ represent the i^{th} element of the partial aggregation vector in node n.

Vector operation to calculate partial aggregation value of node n in substrate s can be given by:

$$V_n^{A_s}(t) = V_n^{l_s}(t) + \sum_{c_j \in C_n} V_n^{c_j}(t). \tag{4.2}$$

Also element wise computation of $V_n^{A_s}(t)$ is given by:

$$v_i^{A_s^n}(t) = v_i^{l_s^n}(t) + \sum_{c_j \in C_n} v_i^{c_j^n}(t).$$

The partial aggregation vector computed at n is in the form of

$$V_n^{A_s}(t) = \left\{ v_1^{l_s^n}, v_2^{l_s^n}, \dots, v_k^{l_s^n} \right\}. \tag{4.3}$$

The total number of nodes in sub-tree T_n is given by $\sum_{i=1}^{k} p_{v_i^{l_s^n}}$.

At the end of the aggregation process, monitoring agent acquire the final aggregation vector $V_r^A(t)$ from node $n = r$ which is the root node of the spanning tree constructed for the InP network graph. Final aggregation vector calculated for the substrate s at the time t which is in the form of $V_r^{A_s}(t) = \left\{p_{v_1^{l_s^r}}, p_{v_2^{l_s^r}}, \ldots, p_{v_k^{l_s^r}}\right\}$ will be forwarded to the VNP by monitoring agent. Number of nodes $p_{v_i^{l^r}}$ corresponding to each local attribute value interval $v_i^{l_s^n}$ for a set of k intervals for each substrate $s \in S$ will be available for the VNP to perform resource discovery and InP selection decisions.

Figure 4.3 shows an example that can be used to illustrate the vector based aggregation mechanism that we use to monitor dynamic attribute value of InP substrate network s, with $u = 10$ and $L = 100$. Let's assume the proposed mechanism is employed to monitor residual CPU values as the dynamic attribute. Hence for this example, we have used arbitrary possible residual CPU values within the range, $0 \geq l_s^n(t) \geq L$ as shown in the Figure 4.3. Initially the leaf nodes such as ***a*** and ***b***, start computing their local residual CPU vectors $V_a^{l_s}$ and $V_b^{l_s}$. Then they compute partial aggregation vectors $V_a^{A_s}$ and $V_b^{A_s}$ which are equivalent to local attribute vectors (since leaf nodes don't have any children) and forward them to their parent ***c***.

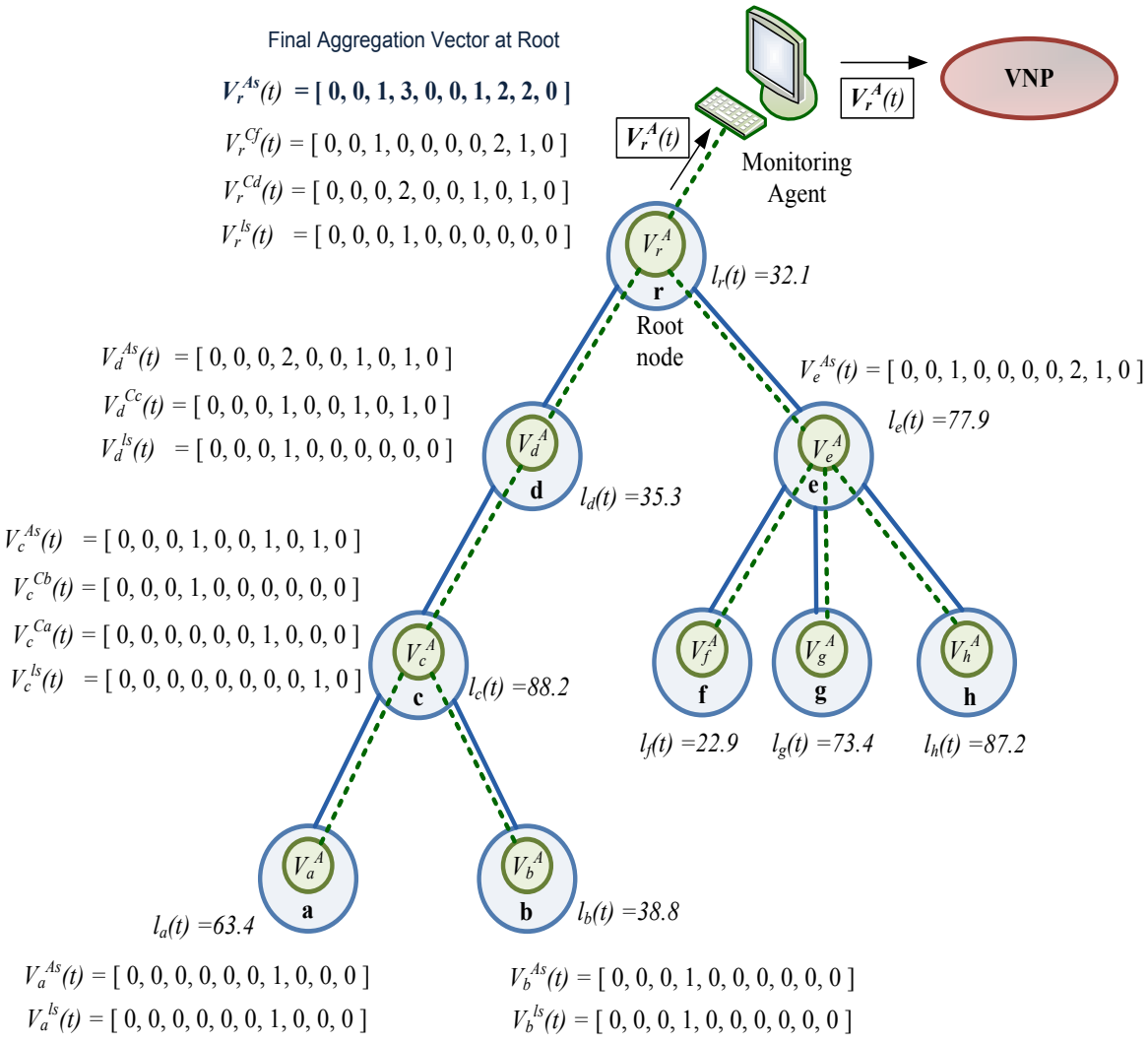

Figure 4.3 Vector-based Incremental Aggregation over Network Spanning Tree

The residual CPU values of nodes ***a*** and ***b*** at time t is 63.4 units and 38.8 units respectively. Hence the corresponding residual CPU vectors for nodes ***a*** and ***b*** will be;

$$V_a^{l_s}(t) = \{0,0,0,0,0,0,1,0,0,0\} = V_a^{A_s}(t)$$

$$V_b^{l_s}(t) = \{0,0,0,1,0,0,0,0,0,0\} = V_b^{A_s}(t)$$

Parent node **c** calculate its partial aggregation vector $V_c^{A_s}$, according to the Equation 4.2 using attribute vector $V_c^{l_s}$ and partial aggregation vectors $V_a^{A_s} = V_c^{C_a}$ and $V_b^{A_s} = V_c^{C_b}$ received from children ***a*** and ***b,*** respectively.

$$V_c^{C_a}(t) = \{0, 0, 0, 0, 0, 0, 1, 0, 0, 0\}$$

$$V_c^{C_b}(t) = \{0, 0, 0, 1, 0, 0, 0, 0, 0, 0\}$$

$$V_c^{l_s}(t) = \{0, 0, 0, 0, 0, 0, 0, 0, 1, 0\}$$

$$V_c^{A_s}(t) = \{0, 0, 0, 1, 0, 0, 1, 0, 1, 0\}$$

Partial aggregation vector $V_c^{A_s}$, is forwarded to its parent node, ***d*** and subsequently ***d*** calculates its partial aggregation vector $V_d^{A_s}$ following the same procedure. Similarly, all the nodes in the substrate spanning tree calculate partial aggregation vectors and ultimately, final aggregation vector $V_r^{A_s}$ is available in the root node ***r*** of the spanning tree.

$$V_r^{A_s}(t) = \{0, 0, 1, 3, 0, 0, 1, 2, 2, 0\}$$

VNP can extract more descriptive information about the availability and distribution of the monitored dynamic attribute, which is the residual CPU in this scenario. According to the relationship illustrated in Equation (4.3), $V_r^{A_s}$ gives number of nodes in the network that contains residual CPU values in the value interval corresponding to each index of it. The sum of products of the mean value of each interval with the magnitude of the corresponding element will provide an approximately equal value to the total available CPU resources in the network. Furthermore, total number of nodes in the network can be found by adding magnitudes of all the elements in $V_r^{A_s}$.

Table 4.1 illustrates the extraction process of this information from final aggregation vector $V_r^{A_s}$ in the given scenario.

Table 4.1 Vector-based Aggregation Results Analysis

Element index (i)	Value interval of v_i^{lr}		Interval mean w_i	Monitored data $p_{v_i^{l_s^r}}$	$p_{v_i^{l_s^r}} \times w_i$	Actual sum of residual CPU values	Error ϵ
	$u(i-1)$	ui					
1	0	10	5	0	0	0	0
2	10	20	15	0	0	0	0
3	20	30	25	1	25	22.9	+1.1
4	30	40	35	3	105	106.2	-1.2
5	40	50	45	0	0	0	0
6	50	60	55	0	0	0	0
7	60	70	65	1	65	63.4	+1.6
8	70	80	75	2	150	151.3	-1.3
9	80	90	85	2	170	175.4	-5.4
10	90	100	95	0	0	0	0
$\sum$				**9**	**515**	**519.2**	**-5.2**

Monitoring error is inevitable in the aggregation vector based technique. Indirectly, local attribute value $l_s^n(t)$ at each node is rounded to the mean value w_i of the interval. Hence, at each node, monitoring error is added to the final output. However, it can be noted that combined effect of some of these errors in most practical scenarios tend to nullify since both negative and positive errors are created in the same vector resulting relatively less average error.

If the actual local attribute value at node n is $l_s^n(t)$ and the mean value of the corresponding value interval is w_i, we can denote the absolute error added to the aggregation vector from the node n as ϵ_n, where

$$\epsilon_n = |l_s^n(t) - w_i| \tag{4.4}$$

and the percent error δ is given by $\delta = \left|\frac{l_s^n(t) - w_i}{l_s^n(t)}\right| \times 100\%$. However, the maximum error occurs when $l_s^n \to u(i)$ or $l_s^n \to u(i-1)$. From (4.4), the maximum error is

$$\max_{n \in N} \epsilon_n = \lim_{l_s^n \to u(i) \vee l_s^n \to u(i-1)} (l_s^n(t) - w_i)$$

After substitution for w_i, we get

$$\max_{n \in N}(\epsilon_n) = \frac{u}{2}$$

and thus $\lim_{u \to 0} \epsilon_n = 0$

Therefore by decreasing the value of u, we can reduce the error introduced to the aggregation vector, ensuing high accuracy in the monitoring output. However we know that the number of elements in aggregation vector is given by k, where $k = \frac{L}{u}$. Substitution of nominal values for u will result in variation of k such that;

$$\lim_{u \to 0} k \to \infty$$

Hence it's evident that reducing the value of u will increase the size of the aggregation vector resulting higher number of monitoring packets in the network. Therefore, this

technique only provides more controllability for VNP over the recognized tradeoff between accuracy and cost, applicable for resource monitoring.

4.5 Enhanced selection and Initial Filtering

With the aid of more refined information provided by the vector based aggregation technique, we can improve the InP selection and filtering process introduced in the *ADVNE* to perform more effective and accurate selection and filtering.

In order to make use of this new aggregation vector obtained from InP networks, VNP is required to pre-process the virtual network requests and construct the vector, equal in number of elements to final aggregation vector for each required dynamic attribute. Since all the node requirements and link requirements are available, VNP can easily compute request vector for each dynamic attribute. Normalizing constant u and maximum integer value for the local attribute L will be the same for both requests and substrates. Hence, number of elements k in aggregation vectors from substrates will be equal to the number of elements in the VN request vector V_q^R , where q denotes the request number (or request ID).

For the VN request q which has M set of virtual nodes, assume virtual node $m \in M$ require l^m amount of resources for a particular dynamic network attribute. VN request vector V_q^R is calculated by normalizing l^m , $\forall m \in M$ using the normalizing constant u . The i^{th} element of the V_q^R , is denoted by ${v_i}^{l^m}$ and its magnitude gives the number of virtual nodes that demand dynamic attribute value, which falls in the range corresponding to the element index.

VN request vector V^{R_q} will be in the form of

$$V^{R_q} = \{ {v_1}^{l^m}, {v_2}^{l^m}, \ldots, {v_k}^{l^m} \}$$

where $| {v_i}^{l^m} |$ = number of virtual nodes in the VN request that satisfy the condition $u(i-1) \leq l^m < ui$. Since all the requirements and information needed to compute request vectors are provided by the VN users, at the initial stage of the VN life cycle, by performing simple arithmetic normalization, so that VNP can perform pre-processing VN requests conveniently without causing significant processing overhead.

If the online aggregation technique is employed, at the arrival of a new VN request, VNP notify monitoring agents to compute aggregation vectors and at the same time start pre-processing the new request to extract request vector. After obtaining both the request and aggregation vectors, vector comparing module perform element wise comparison operation which can be demonstrated using the following figure.

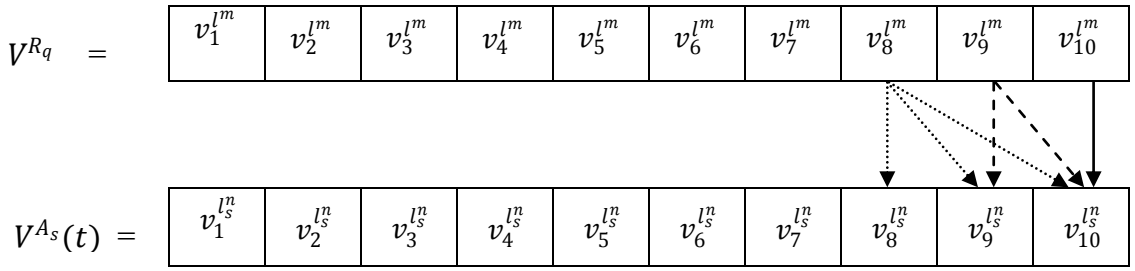

Figure 4.4 Element-wise Matching of Request and Aggregation Vectors

Figure 4.4 shows request vector V^{R_q} generated for request-ID q and aggregation vector $V^{A_s}(t)$ is constructed for substrate network s at time t using the normalization constant $u = 10$. For any given i value, let j be a positive integer that can be defined

as $(i + j) = k$, where k is the cardinality of both aggregation and request vectors. We can mathematically represent the substrate selection criteria from

$$\left| v_i^{l_s^n} \right| + \left| v_{i+1}^{l_s^n} \right| + \left| v_{i+2}^{l_s^n} \right| + \ \ldots \ + \left| v_{(i+j)}^{l_s^n} \right| \ \geq \ \left| v_i^{l_\alpha^m} \right| + \left| v_{i+1}^{l_\alpha^m} \right| + \left| v_{i+2}^{l_\alpha^m} \right| + \ \ldots \ + \left| v_{(i+j)}^{l_\alpha^m} \right|$$

Hence for $\forall i \mid \ 0 < i \leq k,$ and $\forall j \mid 0 \leq j < k, \ (i + j) = k$,

$$\sum_{j=0}^{k-i} \left| v_{(i+j)}^{l_s^n} \right| \geq \sum_{j=0}^{k-i} \left| v_{(i+j)}^{l_\alpha^m} \right| \tag{4.5}$$

All the substrates that satisfy the condition given in equation (4.5) will be sorted based on their embedding potential, before forwarding to the candidate selection and binding module. For a particular VN request, if none of the substrates could satisfy the condition given in Equation (4.5), the request will be rejected due to pre-matching failure.

4.6 Substrate Sorting

In the ADVNE framework, we generated the substrate order based on average minimum shortest path bandwidth values, assuming the fact that bandwidth is more critical than other dynamic network resources such as CPU and memory. In the vector based approach, we cannot find the total aggregation value of minimum path bandwidth values, since vectors can only provide estimations for total and average dynamic resource availability. Moreover, finding minimum path bandwidth values impose significant processing overhead to substrate networks, especially when they have higher number of nodes. Due to these reasons, we have decided to sort the selected substrate networks by assigning weights to each element of bandwidth vectors.

Bandwidth aggregation vectors of each selected substrate network will be obtained by the resource discovery module and it will assign weights proportionally to each element based on their element index. Let X be the selected set of substrates for the VN request q and bandwidth aggregation vector of the substrate x is denoted by $V_{Bw}^{A_x}(t)$. The i^{th} element $v_i^{l_x^n}$ of the bandwidth vector $V_{Bw}^{A_x}(t)$ is weighted by weighting factor $1/(k-i+1)$.

We find weighted average value corresponding to $V_{Bw}^{A_x}(t)$ by

$$E\left(V_{Bw}^{A_x}(t)\right) = \frac{\sum_{i=1}^{k} \frac{\left|v_i^{l_x^n}\right|}{(k-i+1)}}{\sum\left|v_i^{l_x^n}\right|}$$

If the weighted average bandwidth corresponding to substrate network $y \in X$ is equal to $E\left(V_{Bw}^{A_x}(t)\right)$, and if weighted average bandwidth values of substrates x and y satisfy the condition

$$E\left(V_{Bw}^{A_x}(t)\right) > E\left(V_{Bw}^{A_y}(t)\right)$$

substrate x will be moved forward in the sorted order and chosen before y for selection and binding. After constructing the complete sorted substrate order from the selected X set of substrates, resource discovery module will forward both request and sorted order to the candidate selection and binding module. Candidate selection and binding module will select the best substrate to host the VN request by forwarding the binding request to each substrate, starting from the first in the sorted order, until it receives a successful binding response.

4.7 Summary

We have recognized some noteworthy limitations of the basic aggregation technique which is only capable of providing total, minimum, maximum and average values of dynamic network attribute values. This limited information only offers general view on the resource utilization patterns of InP networks and is incapable of providing sufficient information to take firm resource discovery decisions.

In this chapter, we attempt to improve the aggregation process by introducing vector-based aggregation technique that provides VNP with a more detailed picture of InP resources. However, it does not reveal critical security information of the InP infrastructure to the VNP, and most importantly by providing the ability to control the accuracy of dynamic attribute monitoring, and the vector based aggregation technique grants more controllability over the unavoidable monitoring overhead.

We have improved the monitoring framework system architecture introduced in Chapter 3 to support additional functions required to carry out vector based aggregation. We also mathematically proved the ability of the introduced vector based technique to decrease the monitoring overhead at the cost of accuracy degradation of monitored results.

5 Performance Evaluation

5.1 Objective

In this chapter, we evaluate the performance of our proposed basic aggregation and vector based aggregation approaches by conducting simulations. From the presented simulation results, we show that the proposed resource discovery scheme improve the overall performance of the VN embedding process.

5.2 Evaluation Strategy

At first, we analyze scalability and performance of algorithms [59], [65] employed to construct aggregation tree and calculate final aggregation value for a particular dynamic attribute at a given time instant. Next, we compare the aggregation based monitoring with centralized monitoring in terms of number of exchanged monitoring messages. After that, we evaluate the efficiency of the basic aggregation based resource discovery approach by performing a time analysis. We compare average time taken by VNs embedding for each possible final outcome: successful embedding, basic pre-matching rejection, node rejection and edge rejection.

Finally we analyze the ability of the vector based resource discovery approach to effectively manage the accuracy and cost of pre-matching to obtain optimum output in the resource discovery stage of the VN embedding process. We conduct four simulation runs for four different vector sizes; 5, 10, 20 and 50. We have measured the accuracy and cost of the resource discovery phase by finding the number of VN rejections and average

rejection time respectively for each vector size. We have also compared the total processing time taken corresponding to vector based pre-matching, basic pre-matching and without pre-matching approaches.

5.3 Simulation Setup

For the numerical analysis, we have used an open source C++ based discrete event simulator known as ViNE [62]. We have modified the original ViNE by adding new functions to support multiple InPs, construct optimal spanning tree, calculate (basic and vector based) aggregates, filter VN requests and sort substrate networks before selection and binding phases. We have used a Linux operating system in a virtual machine with quad core 3.4GHz processor and 4 GB memory for simulations.

5.4 Substrate and VN Request Generation

Substrate networks and VN requests were generated using open source, GT-ITM topology generator [63]. Substrates and virtual topologies were created using Waxman [64] random graph model with $\alpha = 0.5$ and $\beta = 0.2$ and 10% of the requests have star connected topologies while remaining 90% of the requests and all the substrates have general random topologies.

We have created 20 substrate network sets, each contain 10 substrate networks with equal number of nodes within the set. Those substrate network sets are arranged by number of nodes, which vary from 10 to 200 with the interval of 10 nodes. In other words, the first set of substrates contain 10 substrate networks each contain 10 nodes, second set contain 10 substrate networks with 20 nodes in each and the last (20^{th}) set has 10 substrate

networks with 200 nodes in each substrate network. The CPU and bandwidth resources of substrate networks are real numbers uniformly distributed between 50 and 100. Each pair of substrate nodes are randomly connected with 0.5 probabilities.

We have used a set of 5000 VN requests which contain requests with number of nodes based on the probabilities shown in Table 5.1. The table also shows minimum, average, maximum number of nodes per request and maximum and minimum values of CPU and bandwidth values. VN requests are arrive following a Poisson process with an average rate of 4 requests per 100 time units. Each request has exponentially distributed lifetime with minimum of 250 and an average of 1000 time units. Each pair of VN nodes are connected by a virtual link with the probability of 0.5.

Table 5.1 Composition of VN requests

Number of Nodes			Node CPU		Link Bandwidth		Probability
Min	Ave	Max	Min	Max	Min	Max	
2	3	4	0	50	0	50	12%
4	5	6	0	50	0	50	12%
8	10	12	0	50	0	50	12%
12	15	18	0	50	0	50	12%
16	20	24	0	50	0	50	12%
20	25	30	0	50	0	50	8%
24	30	36	0	50	0	50	8%
28	35	42	0	50	0	50	8%
32	40	48	0	50	0	50	8%
36	45	54	0	50	0	50	8%

5.5 Performance Evaluation of Basic Aggregation Approach

Initially we have observed the scalability and behavior of the network tree construction algorithm and aggregation algorithm. We have used Edmonds's optimum branching algorithm as described in [65] to construct a spanning tree from the network graph. Both algorithms were evaluated by measuring the time taken over number of substrate nodes and the corresponding results were plotted as shown in Figure 5.1.

We can see that for substrates containing 500 nodes, average time taken to construct tree is less than 0.55 seconds and average time taken to calculate aggregates is less than 0.3 seconds. Therefore it is noticeable that both of these algorithms are scalable and do not impose significant time latency.

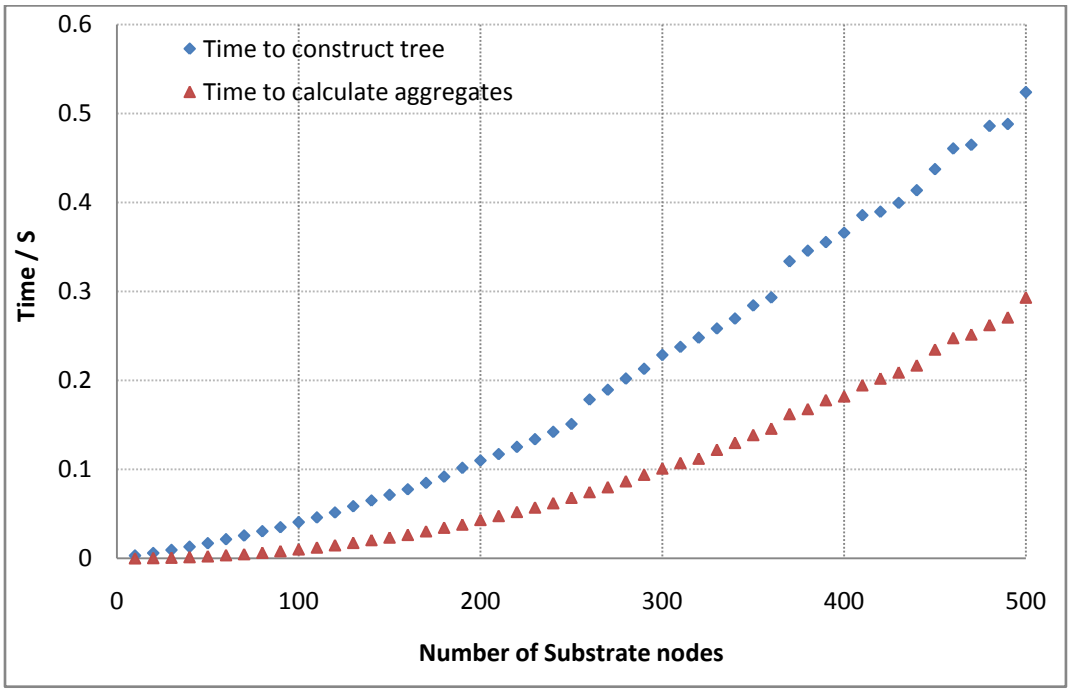

Figure 5.1 Performance and Behavior of Tree and Aggregation Algorithms

Numbers of monitoring messages exchanged in aggregation vs. centralized approaches are presented in Figure 5.2. In centralized messaging systems, all the nodes of the network sends update messages directly to a central node over shortest paths. Aggregation approach introduced by ADVNE starts sending update messages from child nodes to their parents along the spanning tree.

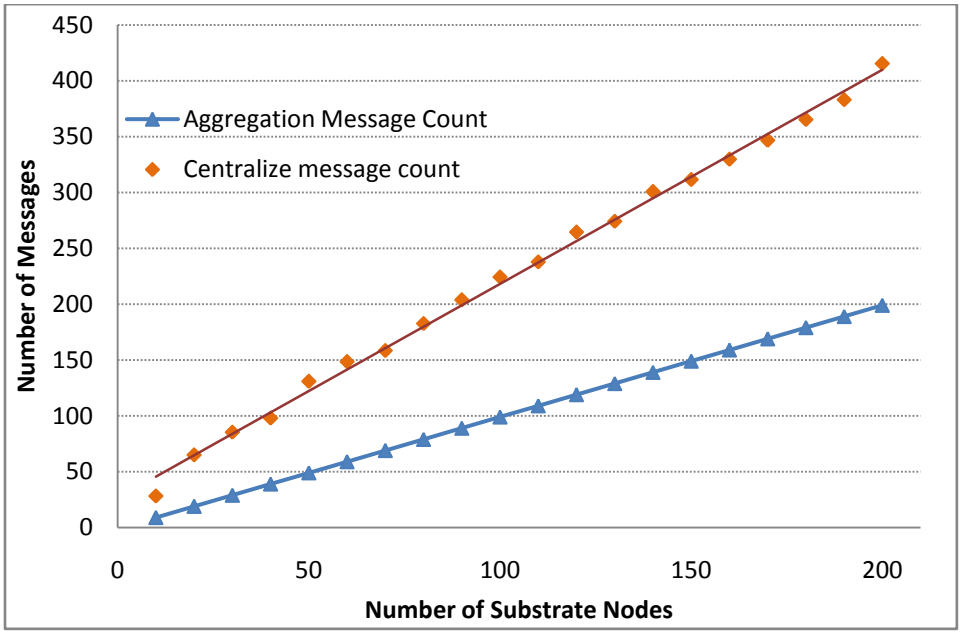

Figure 5.2 Number messages exchanged vs. Number of physical nodes in Aggregation and centralized approaches

Since child nodes and parents are always neighboring nodes, one update from any node n needs to exchange one message in the network. Hence for a substrate having N nodes, the total number of messages exchanged in the aggregation approach U_{agg} is

$$\boldsymbol{U_{agg} = N - 1}$$

For a particular VN request, final result can be either successful embedding, rejection due failure of node mapping, rejection due to failure of edge mapping or rejection from pre-matching. We have carried out time evaluation for each of these decisions. Averages of

the results were calculated for 10 runs for 5000 requests and 10 substrates containing number of substrate nodes between 10 and 200. Corresponding results are plotted in Figure 5.3 with logarithmic scale vertical axis. For small substrate networks, difference between average time of rejection due to pre-matching and rejection due to node mapping failure is between 101- 102.

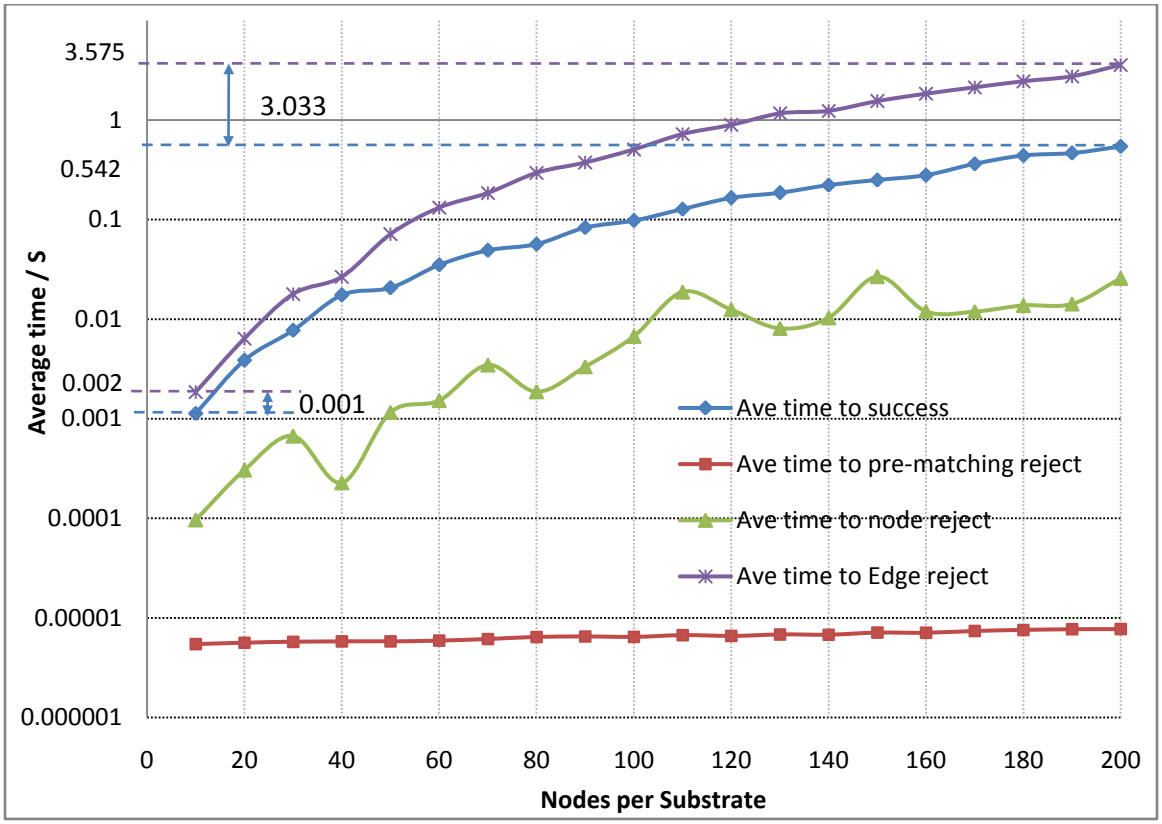

Figure 5.3 Average time, to take final decision of acceptance or rejection due to different factors

For larger substrates, this difference is between 10^3- 10^4. Thus we can assert that pre-matching can improve decision making time of VNP significantly with large scale substrates. If a VN request is rejected due to edge mapping failure, embedding algorithm has taken time to verify its node mapping possibility. Hence we can consider it as worst case scenario. However in Figure 5.3, we can see that with the increase of substrate

network size from 10 to 200, difference between average time to reject due to edge mapping failure and average time for successful mapping increases from 0.001 to 3.033 s. This noticeable difference is due to the fact that, all the substrates in the substrate set are explored before taking the decision of rejecting the VN request based on edge mapping failure.

5.6 Performance Evaluation of Vector Based Aggregation Approach

The main objective of proposing a vector based pre-matching approach for VN resource discovery is to increase the controllability over the previously discussed tradeoff between accuracy and cost. We have further modified the ViNE [62] simulator to construct local vectors for CPU and bandwidth at each node and calculate aggregates along the network spanning tree towards the root node of each substrate network. Following the procedure which we have illustrated in Chapter 4, we have compared the aggregation vectors constructed for each substrate in the substrate set with the arriving VN request to shortlist substrates based on their embedding potential. If all the substrates in the considered substrate set failed to meet pre-matching requirements, the request will be designated as "rejected by pre-matching" and will not be considered for embedding. However some of the requests will still go through the pre-matching filter and rejected by node mapping and edge mapping, increasing the costs and decreasing the efficiency of the embedding process significantly as shown in Figure 5.3.

If we can increase the accuracy of the pre-matching process at the resource discovery phase and forward well refined requests to selection and binding phases, the efficiency of overall embedding process can be increased. Therefore, number of VN rejections at

resource discovery phase can be considered as a measurement of accuracy of the pre-matching process. Since we are interested in analyzing the behavior of vector based pre-matching approach, we obtained simulation results by changing the size of the aggregation and request vectors (VECT_SIZE) to four different sizes; 5, 10, 20 and 50. We also performed a simulation round employing basic pre-matching approach discussed in chapter 4 for comparison. Number of VN rejections from basic pre-matching and vector based pre-matching with different vector sizes over the substrate network sizes are plotted in Figure 5.4.

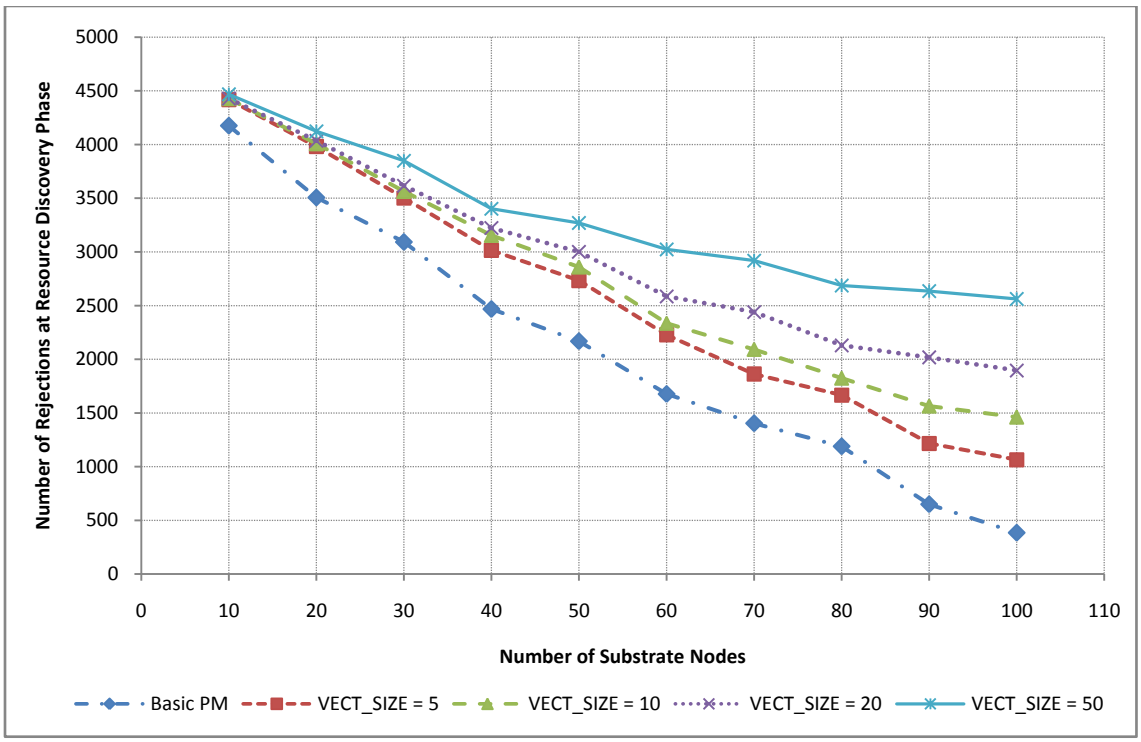

Figure 5.4 Number of rejections in the resource discovery phase vs. number of substrate nodes (Accuracy of pre-matching)

According to the Figure 5.4, we can observe that vector based approach has been able to reject more VN requests than the basic pre-matching approach. In other words, accuracy of the vector based approach is higher than the basic pre-matching technique. More

importantly, we can also notice that when we increase the size of the aggregation and request vectors, accuracy of the pre-matching has also increased in terms of the number of rejections. These results comply with Equation 4.5, which prove that by decreasing the normalizing constant (i.e. by increasing the vector size), we can reduce the error of resource monitoring in vector based resource discovery approach.

We have measured the cost of basic aggregation and vector based pre-matching schemes in terms of processing time taken by the simulator to reject a VN request in the resource discovery phase. We have plotted the average rejection time (i.e. cost of pre-matching) taken by basic aggregation pre-matching approach and vector based pre-matching approach over substrate network sizes in Figure 5.5.

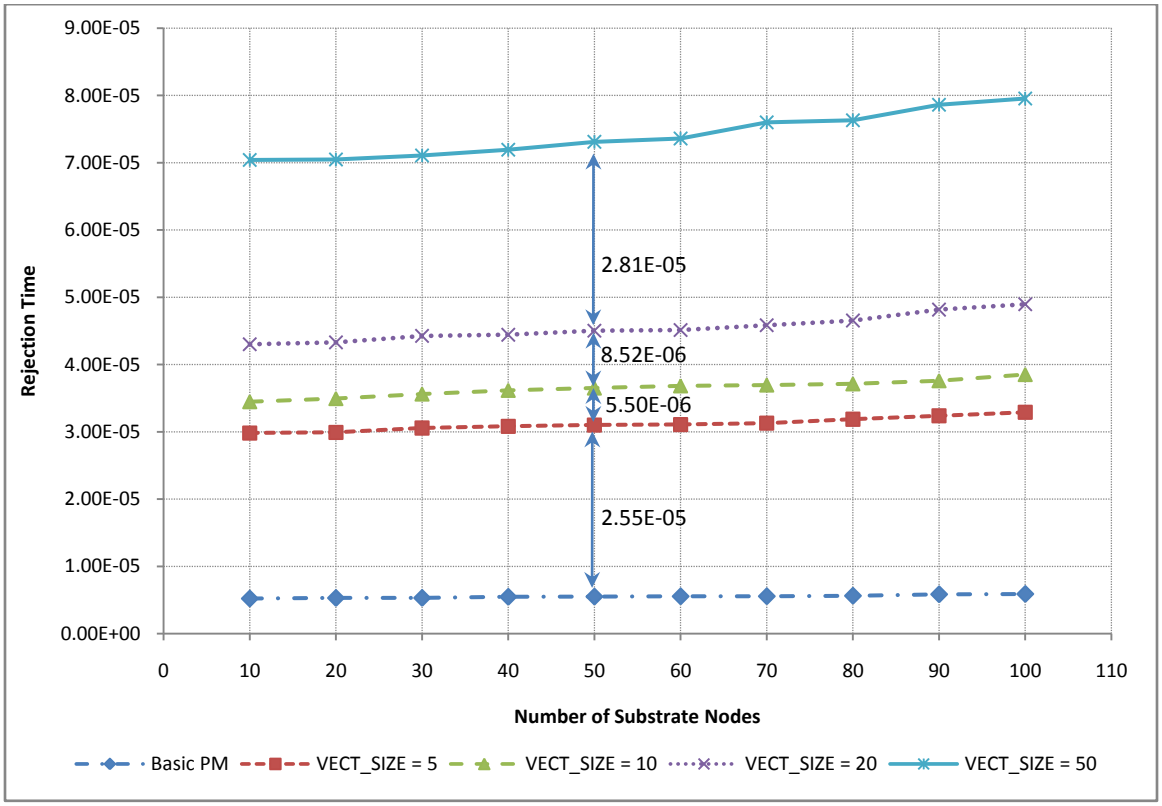

Figure 5.5 Average VN rejection time vs. number of substrate nodes (cost of pre-matching)

According to the Figure 5.5, we can observe that the vector based pre-matching approach has taken relatively more processing time than the basic aggregation based pre-matching approach. Also, we can notice a slight increase in the processing time with the vector size, when the vector size is changed between 5 and 20. However, the average processing time difference between the vector sizes 20 and 50 is relatively high. As an example, according to the Figure 5.5, for substrate size 50, the processing time difference between vector sizes 50 and 20 is 2.81×10^{-06}, which is more than the processing time difference between basic pre-matching and VECT_SIZE = 5, i.e. 2.55×10^{-06}. Therefore, it is clear that, by simply increasing the size of the vector, we cannot increase the overall efficiency of the VN embedding process. Existence of the tradeoff between the cost and the accuracy only allows us to find the optimal point where we can obtain most accurate results relatively less cost. This hypothesis can be further illustrated by Figure 5.6, where we plot total processing time taken by the simulator to process 5000 VN requests employing greedy approach for VN embedding, for six scenarios; no pre-matching at resource discovery, with basic aggregation based pre-matching and vector based pre-matching for VECT_SIZE = 5, 10, 20 and 50.

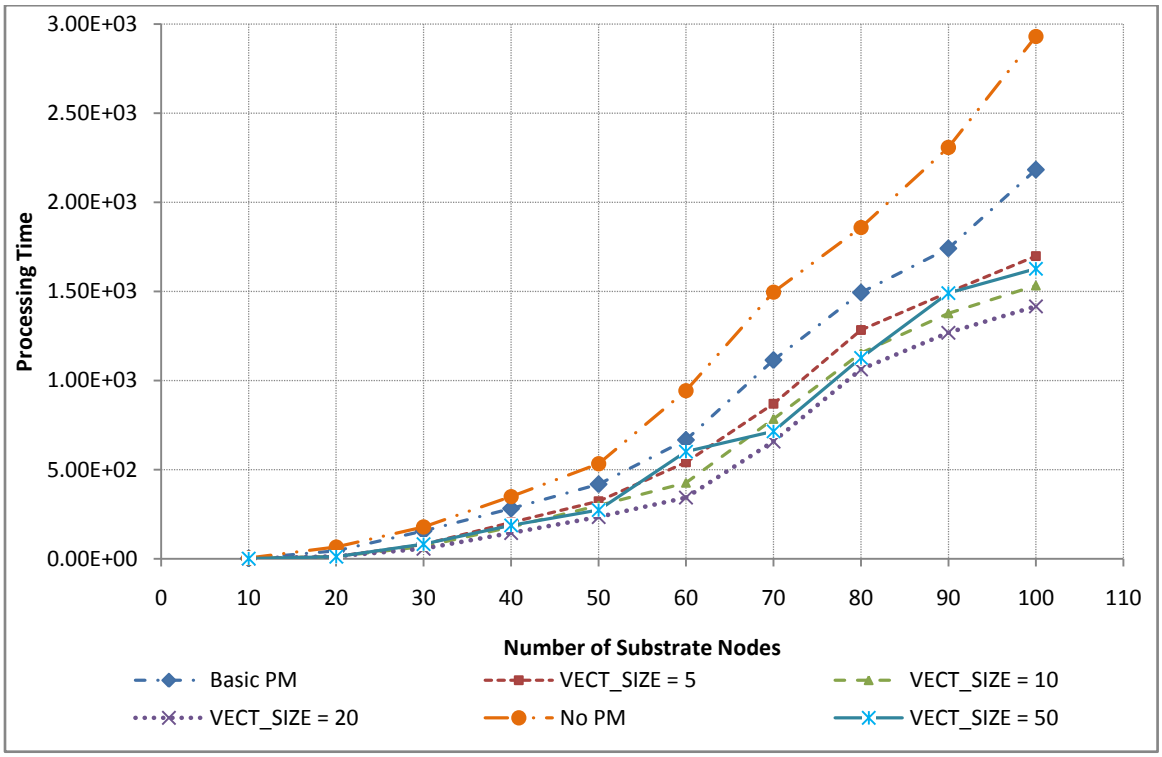

Figure 5.6 Total processing time taken for 5000 VN requests with different resource discovery approaches

From the graph in Figure 5.6, we can observe that the size of the vector is not proportional to the total processing time. In other words, the efficiency of the embedding process will not be increased only by increasing the size of the vector. Since cost of monitoring is also increased with the size of the vector, there has to be a certain vector size which maximizes the efficiency of the overall embedding process. In order to have a more clear insight over the simulation results, we have roughly calculated the efficiency (1/total processing time) of the embedding process for each vector size and plotted in the graph shown in Figure 5.7, for substrate sizes; 60, 70, 80, 90 and 100.

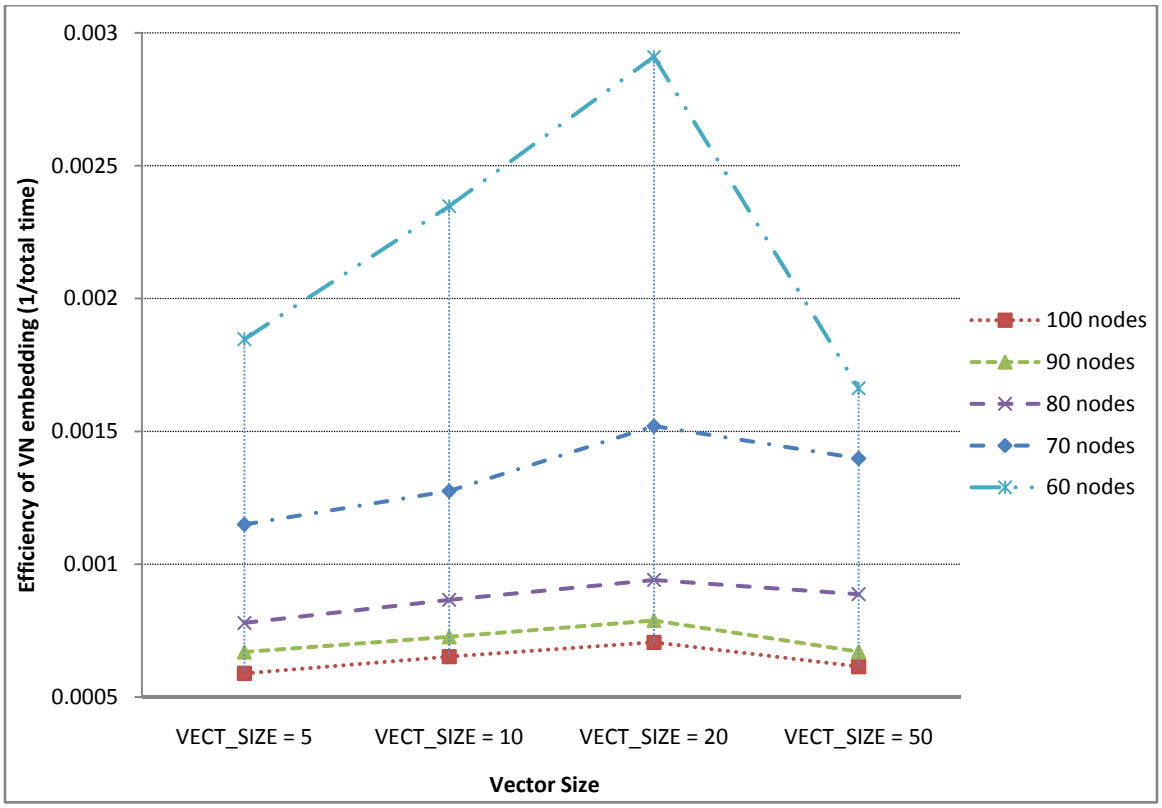

Figure 5.7 Efficiency of VN embedding over the size of vector

It is evident that the simulation results we have obtained are not sufficiently detailed enough to calculate an accurate value for the vector size which gives most efficient embedding time. However, according to the efficiency curves in Figure 5.7, we can declare that, for the simulated substrate networks, approximately the vector size 20 provides an efficient embedding time, for greedy based VN embedding scheme. Moreover, we assume that this value depends highly on the VN embedding approach and substrate and request topological characteristics.

5.7 Summary

In this chapter, we have evaluated the performance of our two proposed resource discovery approaches; basic aggregation based pre-matching and vector based pre-matching. We have studied scalability and performance of two key algorithms we have employed to implement our proposed resource discovery framework over a C++ based discrete event simulator known as ViNE [62]. We have carried out an overhead cost analysis to determine the performance improvement from the proposed basic aggregation based VN resource discovery approach. We also examined the ability of the proposed vector based resource discovery technique to maximize the accuracy of the dynamic attribute monitoring while minimizing the cost of monitoring. Based on the simulation results of the vector based resource discovery approach, we highlight that, overall efficiency of the embedding process has to be increased by recognizing the cost vs. accuracy tradeoff perceptible in the dynamic attribute monitoring.

6 Conclusion and Future Work

6.1 Conclusion

Network virtualization has gained a wide popularity among researches all over the world as a promising technology which is capable of eradicating the impasses of the current internet architecture. Although the network virtualization literature has been expanded within a short period of time, VN resource discovery still remains as an insufficiently explored domain.

Resource discovery plays an important role during the initial stages of the VN lifecycle, minimizing the resource wastage of the embedding process by identifying the potential substrates and substrate resources. Proficient network virtualization resource discovery scheme should be capable of utilizing static and dynamic attributes efficiently while recognizing the inevitable trade-off between accuracy and cost of monitoring. In this research work, we have mainly concentrated our attention towards finding an efficient dynamic network attribute utilizing technique at the resource discovery phase.

We have initially presented a comprehensive survey of network virtualization by examining interrelated technologies and research work. We have provided an analysis of the historical virtualization approaches and conceptual evolution of the network virtualization. We have also summarized previous and current projects related to network virtualization, by categorizing them highlighting the different characteristics of each of them.

Then we have given our attention to resource discovery for network virtualization, recognizing it as an inadequately explored branch of the network virtualization. We have introduced an aggregation-based virtual network embedding framework which increases the overall efficiency of VN embedding process. The introduced approach utilizes dynamic attributes of the network resources at the initial stage of embedding to filter InPs before forwarding the VN binding request. Most importantly, recourse discovery techniques should be fast and cost efficient enough to not to impose any significant load over the embedding process. Hence our proposed resource discovery scheme calculates aggregation values of dynamic attributes of the substrate networks which were filtered by static attribute matching process. By comparing aggregation values obtained from substrates with dynamic attribute requirements of VN requests, we have shortlisted a set of InPs which satisfy basic embedding requirements. Moreover, we have further enhanced the monitoring and aggregation process by calculating aggregation vectors for substrate networks. We have mathematically proved that, by changing the size of the vector, we can increase the accuracy of the monitored data while controlling the cost of monitoring. We also have reinforced this claim by carrying out simulations and analyzing obtained results.

6.2 Future Work

Our research work has opened several research directions from which we can extend our current research work.

6.2.1 Study the relationship between cost, accuracy and efficiency

First of all we have recognized that this research work calls for more sophisticated analysis with different network topology types and embedding approaches. From the simulation results, we have noticed that simply by increasing the cardinality of the vectors, we cannot increase the overall efficiency of the embedding process. Although the accuracy of the monitoring is increased with the size of the vector, increased cost of monitoring leads to decrease in overall efficiency of the embedding process. We have approximately obtained this optimum vector size with our current simulations. However we assume that this optimum vector size depends on number of parameters including substrate network topologies, available resources in substrate networks and employed candidate selection and binding approaches. Therefore we expect to commence more detailed analysis to have a better insight towards the behavior relationship between cost, accuracy and efficiency of VN resource discovery.

6.2.2 Dynamically changing the normalizing constant at runtime

We also identify the necessity of setting-up the vector size dynamically based on the optimum vector size corresponding to substrate resources and topology characteristics. For this, we must be capable of finding the optimum vector size for each substrate and assign those values the simulator during the run time. In our current simulation setup, we

assign normalizing value statically as a programming constant, which determines the size of the aggregation and request vectors. Hence we have to calculate optimum vector size during run-time and dynamically assign the corresponding normalizing constant in order to increase the overall performance of the embedding process.

6.2.3 Provisions for multiple VNPs, VN splitting and path splitting

As future research directions of our work we also look into a more challenging scenario, in which many VNPs are interacting with many InPs, this will further complicate the problem of VN resource discovery. Also we have to investigate on extending our resource discovery approaches in scenarios where VN splitting and path splitting operations are enabled at VN embedding. VN and virtual path splitting will relatively increase the acceptance ratio of the embedding process. Hence they have been recognized as techniques to achieve more enhanced embedding frameworks. Therefore we recognize the importance of extending our studies to accommodate VN and virtual path splitting functions as our future work.

7 References

[1] J. S. Turner and D. E. Taylor, "Diversifying the Internet," in *Proc. IEEE GLOBECOM'05*, St. Louis, MO, USA, 2005.

[2] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the internet impasse through virtualization," *IEEE Computer Mag*., vol. 38, no. 4, pp. 34–641, April, 2005.

[3] J. J. Brosemer and D. J. Enright, "Virtual Networks: past, present and future" *IEEE Commun. Mag*., vol. 30, no. 3, pp. 80 – 85, IEEE ComSoc, March, 1992.

[4] N. M. M. K. Chowdhury and R. Boutaba, "Network Virtualization: State of the Art and Research Challenges," *IEEE Commun. Mag*., vol. 47, no. 7, pp. 20 – 26, IEEE ComSoc, Jul. 2009.

[5] A. Belbekkouche, M. Hasan and A. Karmouch, "Resource Discovery and Allocation in Network Virtualization," *IEEE Communications S&T*, to be published.

[6] I. Houidi, W. Louati, D. Zeghlache, and S. Baucke, "Virtual Resource Description and Clustering for Virtual Network Discovery," in *Proc. IEEE ICC'09 Workshops*, Dresden, Germany, 2009, pp. 1 - 6.

[7] B. Lv, Z. Wang, T. Huang, J. Chen, and Y. Liu, "Virtual Resource Organization and Virtual Network Embedding Across Multiple Domains," in *Proc. of International Conference on Multimedia Information Networking and Security (MINES'10)*, Beijing, China, 2010, pp. 725 - 728.

[8] J. Nogueira, M. Melo, J. Carapinha, S. Sargento, "A Distributed Approach for Virtual Network Discovery," *IEEE Globecom*, Miami, Florida, USA, 2010, pp. 277 - 282.

[9] I. Houidi, W. Louati, W. Ben Ameur, and D. Zeghlache, "Virtual Network Provisioning Across Multiple Substrate Networks," *Computer Networks (Elsevier)*, vol. 55 no. 4, pp. 1011-1023, 2011.

[10] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive Virtual Network Provisioning," in *Proc. ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA'10)*, New Delhi, India, pp. 41- 48, 2010.

[11] M. M. Hasan, H. Amarasinghe and A. Karmouch, " Network Virtualization: Dealing with Multiple Infrastructure Providers," *5th International Workshop on the Network of the Future (IEEE ICC'12 WS - FutureNet*), Ottawa, ON, Canada, to be published.

[12] GEANT2: Common Network Information Service Schema Specification. [Online] Available: http://www.geant2.net/

[13] NDL: Network Description Language. [Online] Available: http://www.science.uva.nl/research/sne/ndl/

[14] H. Amarasinghe, A. Belbekkouche and A. Karmouch, "Aggregation-based Discovery for Virtual Network Environments," *Communication QoS, Reliability and Modeling Symposium (CQRM) IEEE ICC'12*, Ottawa, ON, Canada, to be published.

[15] IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks, IEEE Std 802.1Q-2005, May 2006.

[16] P. Ferguson and G. Huston, "What is a VPN," *Cisco Systems*, Tec. Rep., April 1998.

[17] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)," *Network Working Group, Internet Engineering Task Force*, RFC 4364, Feb. 2006.

[18] L. Andersson and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology*," Network Working Group, Internet Engineering Task Force*, RFC 4026, March, 2005.

[19] R. Venkateswaran, "Virtual Private Networks," *IEEE Potentials*, vol. 20, pp. 11-15, Feb. 2001.

[20] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma and S. Lim, "A Survey and Comparison of Peer-to-peer Overlay Network Schemes," *IEEE Communications Surveys & Tutorials*, vol 7, no. 2, pp.72-93, 2005.

[21] S. Banerjee, B. Bhattacharjee and C. Kommareddy, "Scalable Application Layer Multicast," in *Proc. conference on Applications, technologies, architectures, and protocols for computer communications, ACM SIGCOMM*, vol. 32, no. 4, Oct. 2002, pp. 205 - 217.

[22] B. Zhang, S. Jamin and L. Zhang, "Host Multicast: A Framework for Delivering Multicast to EndUsers," in *Proc. of IEEE INFOCOM*, vol. 3, pp. 1366 – 1375, 2002.

[23] V. Hilt, A. Hari and M. Hofmann, "An Efficient and Robust Overlay Routing Scheme for VoIP," in *Proc. IEEE ICICS*, Bangkok, Thailand, Dec. 2005, pp. 508 - 512.

[24] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek and R. Morris, "RON - Resilient Overlay Networks," in *Proc. 18th ACM symposium on Operating systems principles (SOSP),* vol. 35, no. 5, pp. 131 – 145, Dec. 2001.

[25] J. Risson and T. Moors, "Survey of research towards robust peer-to-peer networks: Search Methods," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 50, no. 17, pp. 3485 – 3521, Dec. 2006.

[26] A. Alles, "ATM Internetworking," *Engineering InterOp*, Las Vegas, March 1995.

[27] D. L. Tennenhouse and D. J. Wetherall, “Towards an Active Network Architecture,” in *ACM SIGCOMM Computer Communication Review*, vol. 37, issue 5, pp. 81-94, Oct. 2007.

[28] T. R. Banniza et al., “4WARD – Architecture and Design for the Future Internet,” *First Project-wide Assessment on Non-technical Drivers*, Id:FP7-ICT-2007-1-216041-4WARD/D-1.1, Jan. 2009.

[29] R. Bless and C. Werle, “Control Plane Issues in the 4WARD Network Virtualization Architecture,” *Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen (WowKiVS 2009), Electronic Communications of the EASST*, Vol. 17, 2009.

[30] J. Carapinha and J. Jiménez, “Network virtualization: a view from the bottom,” in *Proc. 1st ACM workshop on Virtualized infrastructure systems and architectures (VISA ‘09),* 2009, pp. 73 – 80.

[31] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. W. A. Greenhalgh, M. Kind, O. Maennel, and L. Mathy, "Network Virtualization Architecture: Proposal and Initial Prototype," in *Proc. ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA'09),* Barcelona, Spain, 2009.

[32] N. M. K. Chowdhury and R. Boutaba, “A Survey of Network Virtualization,” David R. Cheriton School of Computer Science, *University of Waterloo, Rep. CS-2008-25, Oct*. 2008.

[33] R. Bless and C. Werle, "Network Virtualization from a Signaling Perspective," in *Proc. International Workshop on the Network of the Future (Future-Net'09)*, Dresden, Germany, 2009.

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

[35] J. He, R. Zhang-Shen, C.-Y. L. Y. Li, J. Rexford, and M. Chiang, “Davinci: Dynamically adaptive virtual networks for a customized internet,” in Proc. ACM CoNEXT’08, Madrid, Spain, 2008.

[36] Y. Zhu and M. Ammar, “Algorithms for assigning substrate network resources to virtual network components,” in Proc. IEEE INFOCOM’06, Barcelona, Spain, 2006.

[37] T. Guo, N. Wang, K. Moessner and R. Tafazolli, “Shared Backup Network Provision for Virtual Network Embedding,” in IEEE International Conference on Communications, *IEEE ICC’11*, 2011, pp. 1 - 5.

[38] M. R. Rahman, I. Aib, and R. Boutaba, “Survivable virtual network embedding,” in *Proc. 9th IFIP NETWORKING Conference*, Chennai, India, 2010.

[39] B. Nandy, D. Bennett, I. Ahmad, S. Majumdar and B. St.Arnaud, “User controlled lightpath management system based on a service oriented architecture,” 2006.

[40] User Controlled Lightpaths, [Online], Available: http://www.uclp.ca.

[41] A. Shoykhet, J. Lange and P. Dinda, "Virtuoso: A system for virtual machine marketplaces,"*Department of Computer Science, Northwestern University,* Rep. NWUCS-04-39, July 2004.

[42] A. Sundararaj and P. Dinda, "Towards virtual networks for virtual machine grid computing," in *Proc. USENIX Virtual Machine Research and Technology Symposium (VM'04),* 2004, pp. 14 - 16.

[43] M. Boucadair, B. Decraene, M.L. Garcia-Osma, A. J. Elizondo, J. R. Sanchez, B. Lemoine, E. Mykoniati, P. Georgatsos, D. Griffin, J. Spencer, J. Griem, N. Wang, M. Howarth, G. Pavlou, S. Georgoulas and B. Quoitin, "Parallel Internets framework," *AGAVE Deliverable,* Id: AGAVE/WP1/FTRD/D1.1/public, 2006.

[44] X. Jiang and D. Xu, "VIOLIN: Virtual internetworking on overlay infrastructure," *Purdue University*, Rep. TR-03-027, 2003.

[45] P. Ruth, X. Jiang, D. Xu and S. Goasguen, "Virtual distributed environments in a shared infrastructure," *Computer*, vol. 38, no. 5, pp. 63–69, May, 2005.

[46] G. Bhanage, I. Seskar, R. Mahindra and D. Raychaudhuri, "Virtual Basestation: Archytecture for an Open Shared WiMAX Framework," in *Proc. 2nd ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*, pp. 1-8, 2008.

[47] K. M. Park and C. K. Kim, "A framework for virtual network embedding in wireless networks," in *Proc. 4th International Conference on Future Internet Technologies,* Seoul, Korea *(CFI'09)*, 2009, pp. 5 - 7.

[48] G. Bhanage, D. Vete, I. Swskar and D. Raychaudhuri "SplitAP: Leveraging Wireless Network Virtualization for Flexible Sharing of WLANs," *in* Global Telecommunications Conference, *IEEE GLOCOM'10*, Jan 2010, pp. 1 - 6.

[49] J. Touch and S. Hotz, "The X-Bone," in *Proc. Third Global Internet Mini-Conference, GLOBECOM'98*, pp. 44-52, 1998.

[50] J. D. Touch et al., "A virtual internet architecture," *USC/Information Sciences Institute*, Rep. TR-570, 2003.

[51] J. E. van der Merwe et al., "The Tempest: A practical framework for network programmability," *IEEE Network Magazine*, vol. 12, no. 3, pp. 20-28, 1998.

[52] L. Peterson, T. Anderson, D. Culler and T. Roscoe, "A blueprint for introducing disruptive technology into the Internet," in *Proc. Computer Communication Review SIGCOMM'03*, vol. 33, no. 1, pp. 59–64, 2003.

[53] N. Spring, L. Peterson, A. Bavier and V. Pai, "Using PlanetLab for network research: myths, realities, and best practices," in *Proc. Operating Systems Review, SIGOPS*, vol. 40, no. 1, pp. 17–24, 2006.

[54] G.P. Group, "GENI design principles," *Computer*, vol. 39, no. 9, pp. 102–105, 2006.

[55] A. Bavier, N. Feamster, M. Huang, L. Peterson and J. Rexford, “In VINI veritas: realistic and controlled network experimentation,” in *Proc. ACM SIGCOMM’06*, New York, NY, USA, pp. 3–14, 2006,

[56] N. Feamster, L. Gao, and J. Rexford, “How to lease the Internet in your spare time,” *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 61–64, 2007.

[57] Y. Zhu, R. Zheng-Shen, S. Rangarajan, and J. Rexford, “CABERNET: Connectivity architecture for better network services,” in Proc. ACM CONEXT’08, Madrid, Spain, 2008.

[58] K. S. Lim and R. Stadler, “Weaver-Realizing a Scalable Management Paradigm on Commodity Routers,” in *Proc. 8th IFIP/IEEE International Symposium on Integrated Network Management, IM’03*, 2003, pp. 409 - 424.

[59] M. Dam, R. Stadler. "A Generic Protocol for Network State Aggregation," in *Proc. of Radiovetenskap och Kommunikation (RVK)*, Linköping 2005, pp. 14 - 16.

[60] A. G. Prieto and R. Stadler, "A-GAP: An Adaptive Protocol for Continuous Network Monitoring with Accuracy Objectives," *IEEE Trans. Network and Service Management*, vol. 4 no. 1, pp. 2 – 12, 2007.

[61] D. Jurca and R. Stadler, "H-GAP: Estimating Histograms of Local Variables with Accuracy Objectives for Distributed Real-Time Monitoring," *IEEE Trans. Network and Service Management*, vol. 7 no. 2, pp. 83 – 95, 2010.

[62] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *Proc. IEEE INFOCOM'09*, Rio de Janeiro, Brazil, 2009 pp. 783 - 791.

[63] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an Inter-network," in *Proc. of IEEE INFOCOM*, 1996, pp. 594-602.

[64] B. Waxman, “Routing of multipoint connections,” in IEEE journal on Selected Areas in Communications, vol. 6, no. 9, pp. 1617-1622, Dec. 1988.

[65] R. E. Tarjan, "Finding optimum branchings," Networks, Wiley & Sons Inc., vol. 7, no. 1, pages 25–35, 1977.