Adaptive Management of Virtual Network Resources

By
Bassem Wanis

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in Partial Fulfillment of The Requirements
for Doctor of Philosophy degree in
Computer and Electrical Engineering

School of Electrical Engineering and Computer Science (EECS)
Faculty of Engineering
University of Ottawa

©Bassem Wanis, Ottawa, Canada, 2015

Abstract

The past few years have witnessed a rapid emergence of large-scale, geographically dispersed, clouds offering in the form of an Infrastructure-as-a-Service (IaaS). The adoption of these services requires the deployment of new networking technologies. This in turn, ensures the performance of the offered cloud services. Network virtualization has been proposed as a key attribute of the future inter-networking paradigm, providing efficient resource management solutions. Among the challenges that need yet to be addressed is the necessity to provide dynamic quality differentiated network services. In addition, it is required to guarantee the availability of network resources in response to workload fluctuations. Finally, it is necessary to periodically re-optimize the resource provisioning to be able to provide efficient resource utilization. These challenges are the motivation behind this work which aimed at developing a novel adaptive resource management model based on network virtualization. First, the proposed work describes a novel Virtual-Network-as-a-Service (VNaaS) model offering differentiated network-aware cloud services, resulting in a guarantee for the quality of the offered applications. This is achieved by enabling the cloud application providers to accurately express their dynamic needs, demand constraints and their network latency tolerance. The proposed work also enables the infrastructure provider to offer Elasticity-as-a-Service (EaaS) for the communication links by estimating and reserving the adequate pool of resources needed to fulfill the network workload fluctuations. This EaaS is offered at differentiated levels according to the hosted applications bandwidth-sensitivity. Finally, the proposed work employs a novel network resource re-optimization technique. The latter efficiently performs rearrangement for the VN portions contributing to the fragmentation of the underlying network. Simulation results demonstrate the effectiveness of the presented work and the significant gains achieved in terms of better adaptive network resource management.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Ahmed Karmouch, for his effective supervision, patient guidance and helpful discussions throughout my graduate studies. It would not have been possible to complete this work without his ongoing assistance.

I would also like to extend appreciation to my co-supervisor, Professor Nancy Samaan, who has enriched and refined my knowledge with her valuable ideas and comments. Her technical help, encouragement and constructive suggestions were a source of inspiration and motivation all along my way.

I am delighted to have had the opportunity to work with exceptional colleagues and friends who provided me with encouragement throughout my research. I would like to thank all my lab mates for their cooperation and advice.

I am also grateful to my parents, Victor and Amal, who have given me the chance for a good education, and so much love and support over the years. I probably owe them much more than I realize at this time. Their positive influence provided me with the determination to persevere with my studies. They have always been a source of inspiration throughout my life and their selfless sacrifices shall never go unnoticed. My thanks also extend to my brother, Sameh, for his constant encouragement and support.

Last but not least, my deepest thanks go to my fiancée, Sally, for whom words are not enough to thank her for her patience, love and emotional support which contributed to making the successful completion of my studies.

To my family and my fiancée

Contents

List of Figures

List of Tables

Acronyms

AWS Amazon Web Services. 38

CC Cloud Customer. 46–60, 62–66, 71–74, 77–82, 84–86, 88–92, 100, 119, 128

CLT Central Limit Theorem. 101, 102, 104, 129

CSP Cloud Service Provider. 3, 4, 8, 10, 34, 37, 40–42, 47, 48, 50, 51, 53, 54, 56, 57, 62, 64, 71–75, 77–82, 84–86, 89–97, 99–104, 113, 116–121, 127, 130–132, 152, 153

EaaS Elasticity-as-a-Service. 13, 15, 16, 90, 91, 93–102, 113, 116–119, 121, 123–125, 127, 128, 130, 131, 152–154

EC2 Elastic Compute Cloud. 43

FCT Flow Completion Time. 43

HCS House Cleaning Scheme. 135, 137–140, 142, 146–148

IaaS Infrastructure-as-a-Service. 1, 37, 49

IP Internet Protocol. 18

IPTV Internet Protocol TV. 24, 65

ISP Internet Service Provider. 21, 22

LAN Local Area Network. 18

lub Least Upper Bound. 119

MCF Multi Commodity Flow. 28–30, 34

MILP Mixed Integer Linear Programming. 28

MIQP Mixed Integer Quadratic Programming. 30

MSE Mean Square Error. 125, 126

NIC Network Interface Cards. 43, 44

QoE Quality of Experience. 39, 42, 44

QoS Quality of Service. 5, 7, 11, 12, 36, 37, 151

SLA Service Level Agreement. 6, 8, 10, 31, 54, 56, 79, 94, 113, 116, 117

SLO Service Level Objectives. 38, 95, 96

SN Substrate Network. 4, 5, 8–10, 14, 16–18, 21, 23, 45, 132–136, 138–140, 142, 144–151, 153, 154

SP Service Provider. 2, 4, 6, 11, 12, 14, 20–23, 31, 34, 41, 49, 91, 120, 152

SVC Stochastic Virtual Cluster. 41

VLAN Virtual Local Area Network. 17, 18

VM Virtual Machine. 3, 7, 36, 37, 43, 44, 46, 48–52, 54–56, 65, 80, 88, 91–93, 95, 96

VN Virtual Network. 2–17, 20, 21, 23–35, 37, 40, 45, 48–56, 59, 65, 88, 91, 92, 100, 132–136, 138–147, 149–155

VNaaS Virtual-Network-as-a-Service. 14–16, 45–47, 50, 51, 53, 54, 56–58, 60, 61, 63–66, 72–74, 77–82, 84–86, 88, 95, 121

VNO Virtual Network Operator. 23

VNP Virtual Network Provider. 21, 23–26, 28, 31

VNRMS Virtual Network Resource Management System. 28

VoD Video on Demand. 12, 40, 49

VPLS Virtual Private LAN Service. 18

VPN Virtual Private Network. 17–19

Chapter 1

Introduction

1.1 Overview

The past few years have witnessed a rapid emergence of large-scale, geographically dispersed, public clouds offering various computational and networking resources in the form of an Infrastructure-as-a-Service (IaaS) on a per-needed basis [7–9]. These public clouds brought along opportunities for a wide range of novel distributed applications such as algorithmic trading, Netflix's video streaming, virtual enterprises, MapReduce-based applications, Facebook's social networking, and compute-intensive scientific experimentations [10–13]. The adoption of these offerings is motivated by the fact that providers of these applications need not to invest heavily in building their own infrastructures; rather, they need only to pay for their resource needs according to their hosted applications' demands. Furthermore, the ability to rent resources that are distributed over multiple data centers, dispersed in various geographic locations, effectively increased their applications' reliability and provided them with the means to further decrease their end-users' experienced latency by placing the used resources closer to their end-users [7, 14, 15]. Therefore, networking plays a crucial role in providing data communications within cloud data centers, as well as among data centers distributed at different locations [7, 11, 16, 17].

To cope with this emergence, the deployment of new networking technologies becomes indispensable since such highly distributed data environments have network requirements that clearly differ from those of general-purpose networks. For instance, bandwidth/latency sensitive cloud-based applications (e.g., video streaming, algorithmic trad-

ing, online gaming, etc.) require more than just a best-effort network in order to be able to meet these applications requirements (e.g., experienced latency, throughput, etc.) [18]. However, due to the competitive objectives and the dissimilarity between the existing multiple infrastructure providers, the employment of new networking technologies turns into an impractical process [19–22]. As a result, network virtualization has been proposed as a promising way to overcome the Internet ossification problem by facilitating the introduction of architectural modifications to the current Internet [20]. In addition, network virtualization is the key to the current and future success of cloud computing due to its great contribution to the improvement of network utilization, the reduction of deployment costs, and the maintenance of high standards of manageability, scalability, and isolation [23].

The performance of cloud-based applications is dependent upon the efficient utilization of the underlying networks resources. In addition, in many cases data communications may become an obstacle that limits the performance of bandwidth/latency sensitive cloud applications [16, 24, 25]. However, most cloud providers simply offer virtual computing/storage resources with little communication guarantees, which may hurt the performance of the deployed services.

Network virtualization allows the coexistence of multiple Virtual Networks (VNs) over the same physical substrate. It enables multiple VN users (i.e., the Service Provider (SP) who is renting computing, storage and network resources to offer its cloud-based application) to rent the needed bandwidth by provisioning the network resources which guarantee the quality of the offered services while minimizing the cost. The two main participating entities in network virtualization (i.e., the infrastructure providers and the VN users) are driven by different objectives. The infrastructure providers are trying to

maximizing their profits, while the VN users aim at assuring the quality of their hosted services while minimizing the cost subject to a pre-existing budget [26]. Unfortunately, due to the various quality requirements of current cloud-based applications, the workloads fluctuations, and the fragmentation of the available and utilized network resources, the problem of effectively provisioning the underlying network resources has became one of the most challenging issues with respect to network virtualization resource management, and in particular, in the field of cloud computing networking. In addition, network resource pricing is another challenging issue that aims at maximizing the infrastructure providers revenue while regulating the VN users' demands.

The focus of this dissertation is to explore the benefits of network virtualization with the goal of providing quality differentiated VN services which enable the dynamic provisioning of network resources, while at the same time guaranteeing the service quality. Currently, most cloud-based services are hosted among dedicated Virtual Machines (VMs) residing on physical servers that are dispersed throughout multiple data centers [11, 17, 27]. For this reason, consideration is given to the case of large-scale public clouds that are geographically distributed. This is done in order to demonstrate how the proposed techniques provide efficient data communication among dispersed data centers (i.e., inter-data centers communications) [12, 16, 27, 28]. However, most Cloud Service Providers (CSPs) simply offer VMs without communication guarantees, and in particular, the inter-data centers communications resulting in negatively affecting the performance of the deployed services.

This chapter briefly discusses the network virtualization concept and some challenging issues that arise in network virtualization resource management. Furthermore, an outline is provided of the contribution of this work in comparison to current research literature.

Finally, the organization of the remainder of the thesis is given.

1.2 Network Virtualization

Network Virtualization allows multiple VNs to coexist over the same physical infrastructure in a seamless manner. This results in the ossification problem being overcome and solutions being provided for a wide range of networking challenges [19, 29]. A network virtualization environment can be defined as the network environment which enables the deployment of multiple heterogeneous network architectures by sharing the same infrastructure resources. These physical infrastructure resources can be administered by one or multiple infrastructure providers where the CSPs are decoupled from the infrastructure providers[1] [29]. Each SP should be able to use an arbitrary network topology, routing or forwarding functions as well as customized control protocols independent of the underlying physical network and other coexisting VNs [29].

An individual VN consists of a subset of the Substrate Network (SN) resources composed from the virtual nodes and the virtual links. A virtual node can either be a virtual host or a virtual router. A virtual host acts as a packet source or a sink and is never located inside the network [29]. A virtual router acts as a router in the physical network. Its main functionality is to forward packets according to the protocols of that virtual network. Similarly, a virtual link is constructed by partitioning the interconnecting link resources which may span over one or more connected physical links (i.e., a path in the underlying physical topology).

The main objective of enabling multiple VNs to coexist together on a shared physical SN is the flexibility with every aspect of networking. The resources needed to guarantee

[1]For convenience, in this thesis, the 4WARD Network Virtualization architecture [1, 2] is adopted as will be shown in chapter 2.

the performance of the cloud-based application can be abstracted as a VN request [30]. The infrastructure providers try to efficiently allocate resources to each VN request to ensure a high utilization of their SN resources. The VN embedding process involves mapping virtual nodes/links to substrate nodes/paths [31], [32] in an efficient manner, in order to ensure high and balanced utilization of the substrate network. Traditional VN embedding approaches statically allocate a fixed share of bandwidth for each virtual link as the basic mean for service quality guarantees [29]. While static allocation provides an optimal performance for short-lived small VNs, it is inefficient and in many cases may cause worse utilization and performance than existing network sharing techniques, in particular, in the case of long-term cloud hosted applications [33].

A great challenge for VN embedding approaches is to dynamically provision the network resources among hosted VNs to be able to adapt to the dynamic variations of the hosted applications' needs (e.g., [34] [35]). Moreover, with today's advances in cloud applications, network virtualization should allow running multiple heterogeneous VNs which are customized for different performance objectives. In the following sections, further elaborations will be given to the motivation of using network virtualization and some challenging issues that arise from the nature of current large-scale long-term distributed cloud applications.

1.3 The Motivation for Using Network Virtualization

While the cloud-based applications have recently been receiving widespread attention, a best-effort network delivery service is not the best for all purposes [19]. Typically, applications such as algorithmic trading, Google Docs sharing, Netflix's video streaming, virtual enterprises, MapReduce applications, Facebook's social networking and compute-intensive scientific experimentations, have a strict Quality of Service (QoS) requirements

and strongly depend on the performance of the underlying physical network [10–12]. The unpredictability of the communication links leads to quality uncertainty for the hosted services, which is unacceptable by most applications, especially for mission-critical applications. Nonetheless, due to the competition among the SPs, any slight reduction in the quality of the hosted services results in a significant loss in revenue [28]. In addition, the sharing of network resources among several collocated applications, without providing the appropriate isolation, adversely affects the quality of the provided services. For instance, the unexpected workload fluctuation of one application may adversely affect other applications sharing the same communication link. Thus, without performance isolation, there is no protection against selfish, compromised and malicious users [36]. As a result, network virtualization techniques are employed to provide accurate network resource allocation and performance isolation based on the contractual Service Level Agreement (SLA). Any SLA violation incurs monetary penalties in order to guarantee the required applications performance.

Several challenges arise concerning accurately provisioning the network resources based on the applications time-varying needs while still maintaining differentiated service qualities according to each applications sensitivity to network performance. First, current solutions either randomly or statically provision the network resources , and both can lead to inefficient resource utilization [37]. Even with some effort to dynamically provision the network resources among hosted VNs (e.g., [34] [35]), including those that target cloud environments (e.g., Seawall [36] and NetShare [36, 38, 38]), there is still difficulty in estimating the expected peak-traffic loads of each VN throughout its life-time. As a result, VN users cannot control the quality, and in particular, the experienced latency, of their hosted applications due to the unpredictability of the communication links. The VN users need to be able to optimally decide and demand the appropriate VN resources

and their quality levels according to the hosted application sensitivity. In addition, little work has been proposed (e.g., [26]) to develop appropriate mechanisms that can help VN users efficiently allocate their operating budgets with the goal of renting the right amount of network resources according to their expected applications demands.

Another important observation is that the majority of existing virtualization techniques do not take into consideration the nature of the hosted services when making the allocation decisions. Clearly, ultra latency-sensitive applications such as online gaming require rigid resource allocation. However, other time-insensitive applications (e.g., VM image distribution, movement of big data in the cloud, and offline laboratory experiments [39]) can tolerate some experienced network latency where the quality of the delivery network is not critical to their performance, especially if this brings along a price reduction. Additionally, in order to be able to have a strict QoS management, the employment of a service monitoring mechanism is necessary to regulate the offerings and demands of the hosted services. Furthermore, differentiated QoS requires an appropriate pricing mechanism that reflects the actual market needs and efficiently regulates the network resources demands on the differentiated QoS.

1.3.1 Virtual Network Demand Fluctuation

Long-term distributed cloud-based applications are characterized by a high degree of workload fluctuation resulting in dynamic traffic needs [40–42]. This fluctuation is mainly attributed to the nature of the offered services and/or other external events that may result in incremental growth or sudden variation in popularity of who is using the hosted service (e.g., releasing of a new movie, publishing of an article about a particular company on a highly visited news site, or popularity participating in a conference). A great

challenge for VN provisioning techniques is to effectively provision the network resources needed to ensure that the hosted applications meet their performance guarantees. Dynamic provisioning techniques are required in order to provide a continuous down- or up-scaling of the amount of allocated resources with the goal of accurately reflecting the time-varying dynamic needs of long-term cloud hosted applications [40–42]. Only provisioning a fixed amount of bandwidth, for each time interval, is not sufficient since it assumes that there is no workload fluctuation within this interval.

One of the significant characteristics that elevates the popularity of cloud-based applications is its *Elasticity*. Cloud elasticity provides the ability to dynamically allocate the resources according to the dynamic application needs [43–46]. Nonetheless, little attention is paid to the elastic provisioning of network connectivity which highly affects the performance of the VN and their hosted applications [47]. Therefore, the VN traffic load varies significantly, stressing the need for a network elasticity service. This value-added service allows the consumed network resources to be down- or up-scaled in a continuous manner, according to the VN current load. Existing approaches are limited to either provisioning unbounded network resources, or statically provisioning the expected peak-traffic value. Both solutions lead to inefficient resource utilization [37]. Likewise, under-provisioning is not acceptable since it results in performance degradation which implies violating the SLA and incurring monetary penalties (e.g., [34] [35]). Nonetheless, these techniques do not suggest a way that guarantees the availability of the network resources in response to workload changes.

To overcome the aforementioned problems, a network resource management approach is needed to be able to dynamically estimate and re-size the SN pool in order to guarantee the availability of network resources. This results in enabling the CSPs to offer the promised network elasticity. In addition, this approach needs to offer differentiated

elasticity levels according to the degree of bandwidth-sensitivity for the cloud-hosted applications.

1.3.2 Substrate Network Fragmentation

Network virtualization techniques aim at efficiently allocating the underlying SN resources to the hosted VNs. Unfortunately, over time, and due to the dynamic provisioning of network resources (i.e., up- or down-scaling of the amount of allocated resources) and the initiation and termination of VNs, the available and utilized SN resources become *fragmented*. SN fragmentation is conceptualized as the state in which the SN resources are used inefficiently, resulting in adversely affecting the SN utilization, the VNs performance and the effectiveness of the dynamic provisioning for future demands [48]. This in turn, gradually degrades the performance of these network virtualization techniques and prevents efficient dynamic provisioning for resources. On the one hand, inefficient utilization of the SN increases the unavailability of network resources (i.e., increases the rejection ratio of future VN requests), thus reducing the infrastructure providers revenue. On the other hand, the unavailability of SN resources may result in selecting longer paths during the dynamic provisioning or the embedding process of new VN requests which in turn reducing the performance of the embedded VNs. In addition, the unavailability of SN resources may prevent the embedding of VN requests. Thus, overcoming the SN fragmentation problem would benefit both the infrastructure providers and the VN users. To overcome the aforementioned problem, a number of SN re-optimization techniques have been developed which mainly try to reconfigure the resource allocation, in either a periodic or event-triggered manner (e.g., [32], [49], [50], [51], and [52]). Such network virtualization re-optimization techniques vary from either the reallocation of some vir-

tual links while keeping the virtual nodes unchanged [50], or the reallocation of a single virtual node at a time with its attached hanging virtual links [32]. Nonetheless, these restrictions limit the enhancement in the utilization of the fragmented SN resources. Thus, effectively detecting the congested VN portions and their topologies which need to be re-allocated is not a trivial task. Moreover, the experienced service interruption time during migration is another major challenge that most existing solutions do not take it into consideration. However, this interruption is unacceptable for real-time and critical applications and might result in the CSP paying penalties due to the SLA violation [53]. Thus, an efficient re-optimization technique must be equipped with a migration technique that reduces the migration cost. The majority of the existing techniques in the literature focus on computing, storage and network elasticity within special topologies (i.e., the network topology of the data centers) [36,38,54–56], with little attention to the general networks that usually connect several data centers. Dynamically re-optimization of the SN is needed to guarantee the availability of network resources whenever needed.

To sum up, existing techniques have certain limitations in the degree of flexibility for the allowance of VN users to accurately express their dynamic differentiated service quality needs, constraints, and tolerance to the network latency. Also, a number of the dynamic approaches proposed in the literature have assumed that there is advanced knowledge of a deterministic traffic load of the hosted VNs. A crucial issue that also contributes to an extension of this problem is determination of a methodology providing a pricing mechanism that efficiently regulates the VN users demands. This must occur while at the same time stimulating the infrastructure provider to offer service differentiation.

1.4 Thesis Objectives

Recently, adaptive network resource management has received a great deal of attention, and much progress has been made since the wide spread use of cloud-based applications which have strict QoS requirements. This is particularly applicable to latency-sensitive applications (e.g., online gaming, video streaming, algorithmic trading, etc). This section presents the objectives of providing an adaptive network resource management model that provide solutions to the aforementioned challenging issues. These include being able to ensure the required performance of the cloud-based applications, despite the dynamic demands of heterogeneous VNs (i.e., each VN may have a different degree of latency/bandwidth-sensitivity).

To provide better large-scale cloud-based applications, SPs have to decide the amount of network resources which maximize their utilities. Existing resource provisioning approaches require the SPs to deterministically estimate and reserve the exact amount of bandwidth demand in the abstractions, which could be difficult and may result in inefficient bandwidth provisioning (i.e., causing over- or under- provisioning problems) due to the demand uncertainty [57–62]. Little work has been proposed (e.g., [44, 59]) to find an efficient provisioning approach which deals with the network demand uncertainty.
In this thesis, a model is proposed to break down the network resources provisioning process into two aspects, namely the contracted bandwidth and the required elasticity level. Thus, the SP has to decide, for each time interval, the amount of bandwidth resources which maximize its utility subject to pre-allocated budget constraints. Then, in order to be able to cope with the time-varying workload fluctuation (i.e., the requested excess of bandwidth over the originally contracted bandwidth) within each time interval, the SP requires an elasticity level that probabilistically models the bandwidth demand un-

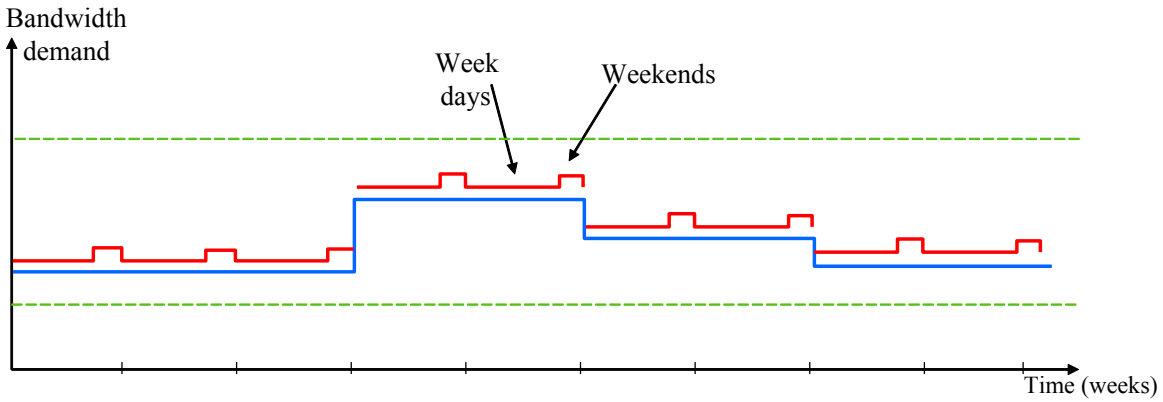

Figure 1.1: An example of the two provisioning aspects

certainty. For example, considering the case of a Video on Demand (VoD) cloud-based application, the SP has to decide for each time interval (one week in this example) the amount of network resources that maximizes its utility subject to the pre-allocated budget constraints. The SP may change its utility function at each time interval to accurately reflect the nature of its currently active cloud services. Accordingly, the requested bandwidth amount may vary due to the release of new movies as shown by the blue line in Fig. 1.1. In addition, the VoD SP requires an elasticity level which guarantee the availability of network resources to fulfill the dynamic workload changes during the week days and the weekends. This is shown by the red line in Fig. 1.1. The degree of bandwidth availability differs according to the selected elasticity level.

The aforementioned limitations of current VN resource management approaches exemplify a set of strong motivations for developing a novel adaptive VN resource management approaches which provide solutions to these challenging issues. This must be carried out in order to ensure the required QoS of cloud-based applications despite their differentiated dynamic needs. The proposed adaptive approaches must offer the VNs with differentiated service quality guarantees, while providing the VN users the flexibility to

accurately express their dynamic needs. Additionally, these approaches must guarantee the availability of network resources to be able to deal with traffic workload fluctuations. Finally, the proposed model should provide an efficient network re-optimization technique equipped with an effective migration technique, guaranteeing the availability of network resources whenever needed. It is also worth noting that the security issues related to VN resource management approaches are not addressed in this thesis [63–66]

1.5 Dissertation Overview

This dissertation focuses on the issue of providing adaptive management of VNs. The proposed model enables the VN users to accurately define their network needs by providing an easy model to present the differentiated service qualities, and in particular, the service latency, for their hosted applications. The proposed model considers several significant constraints that strongly affect the performance of the model. Also, the complexity of the model is reduced by modeling the VN demands as a two-stage budget allocation problem. To regulate the VN users demands, a dynamic pricing mechanism is proposed based on Ramsey pricing. Moreover, the model ensures the availability of network resources to overcome the bandwidth demand uncertainty. In order to ensure resource availability, an Elasticity-as-a-Service (EaaS) approach is proposed which estimates and reserves the optimal pool of network resources needed to fulfill the expected fluctuation of the virtual links. In order to benefit from the above approaches, the underlying physical network should be periodically re-optimized to ensure its efficient utilization. The objectives of the presented research work can be summarized as follows:

- ***Provide quality differentiated VN service model***: to be able to accurately reflect the actual VN differentiated dynamic networking needs and to map the VN

requirements to multiple differentiated service qualities.

- ***Maximize the welfare of both the infrastructure provider and VN user***: to prevent the infrastructure provider monopoly of prices by maximizing the welfare of both participating entities.

- ***Increase SN utilization***: to be able to fulfill the VN dynamic fluctuations and to increase the acceptance ratio of future VN requests, by overcoming the SN fragmentation problem.

- ***VN Elasticity***: to guarantee the availability of network resources in response to workload changes.

1.6 Summary of Contributions

The goal of this thesis is to design a new model for adaptive VN resource management. The major contributions of this proposal can be summarized as follows:

- ***Differentiated Virtual-Network as-a Service [67]***

  In this research, a new Virtual-Network-as-a-Service (VNaaS) model is proposed that allows the infrastructure provider to offer differentiated network-aware VN services to its users (i.e., cloud based SPs). In this model, each virtual link in the VN is split into a number of virtual pipes, each with a specific expected latency and a corresponding differentiated price. Also, the VN dynamic demand for a VNaaS is formulated as an optimization problem in which the VN user seeks to request the optimal bandwidth resources for each virtual pipe. This is carried out in order to maximize the VN user's utility obtained from providing the hosted service, subject to the VNaaS pre-allocated budget constraints. Then an analysis is given to demonstrate how the complexity of this problem can be reduced, by posing it as

a two-stage budget allocation problem. For the first-stage budgeting, a closed formulation is derived to calculate an optimal price index and an effective bandwidth for each virtual link in the VN. In the second stage, the obtained effective bandwidth vector and price indices are employed to derive a formulation to extend the solution of the original problem to include additional internal routing constraints resulting from servicing their own end-users. Finally, a novel Ramsey-based [68] latency-aware network resource pricing mechanism is proposed to efficiently regulate the VN users demands while giving the infrastructure provider an incentive to offer service differentiation. The proposed mechanism employs dynamic network resource pricing to regulate the demand of incoming VNaaS requests.

- ***Modeling and Pricing VN Elasticity-as-a-Service [69]***

  First, an efficient approach is developed that enables the infrastructure provider to estimate and reserve the optimal pool of network resources needed to fulfill the demands imposed by the network workload fluctuations of applications subscribing to the VN EaaS. This approach allows the infrastructure provider to offer communication EaaS at differentiated levels based on the degree of bandwidth-sensitivity of the hosted applications. In order to capture the inter-data centers network activity of hosted applications, their workloads are modeled in this work using Markovian modeling. Also, a novel dynamic pricing mechanism for network EaaS offerings is implemented that can be employed to maximize the expected long-term revenue, and to regulate network elastic demands.

- ***Substrate Network re-optimization via Live VN Migration [70]***

  An efficient technique is developed for the selection and re-allocation of VN portions

that are contributing to the fragmentation problem of the underlying substrate network. The technique takes into consideration the trade-off between the benefit from increasing the SN utilization and the cost incurred by the VN migration. Also, a novel VN live migration technique is presented that significantly reduces the service interruption time during migration.

1.7 Organization of the Thesis

The remainder of the this thesis is organized into the following chapters. Chapter 2 presents some of the related work that forms the background history of this research and presents state-of-the-art research that is currently adopted. Some of the issues addressed by various research groups and the limitations of their work are identified. Chapter 3 outlines the design of the proposed differentiated service qualities VNaaS model. Chapter 4 presents a novel VN resource management approach that enables the offering of network EaaS by estimating the required bandwidth pool in order to guarantee the availability of network resources whenever needed. Chapter 5 presents an efficient technique to overcome the SN fragmentation problem. Chapter 6 summarizes the research work and presented contributions, and discusses direction for future research.

Chapter 2

Background

In recent years, there has been an increasing interest in developing new technologies for the sake of improving the Internet services, and in particular, Cloud computing services. A great challenge for guaranteeing the performance of the hosted cloud applications is to employ efficient networking that plays a crucial role in providing data communications within cloud data centers, as well as among data centers distributed at different locations. Network virtualization has been proposed as a promising way to provide efficient network resource management. However, the underlying concept of virtualization is not absolutely new as verious related concepts were introduced few decades ago. During this period, some concepts were successfully adopted by industrial applications such as Virtual Local Area Networks (VLANs), Virtual Private Networks (VPNs) and Overlay Networks [22].

This chapter presents a background overview of the main concepts of network virtualization and other related concepts, aiming at providing the fundamental conceptual differences between them. This thesis builds and extends upon network virtualization technology to provide an effective adaptive VN resource management solution. In addition, the network virtualization business model, players and system architecture are introduced. Finally, this chapter provides the readers with an overview of the state of the art active research initiatives in the areas of dynamic VN resource management and SN re-optimization techniques.

2.1 Evolution of Virtualization Concepts

The past thirty years have seen increasingly rapid advances in the field of computer networking. The term virtual networks was already introduced in late 1980's as a solution to reduce the complexity and to minimize the cost of large-scale communication networks [71–73]. These objectives are met through the logically partitioning of large communication networks into smaller networks. Afterward, with the enormous growth of the Internet, several attempts have been made to develop the concept of the co-existence of several logically separated networks on a shared SN. From these technologies, VLANs [74], VPNs [75, 76] and Virtual Private LAN Services (VPLSs) [77] were widely accepted by the industrial community as successful virtual network technologies to achieve logical separation among communication links that co-exist over the same physical SN. Extending the functionalities of the substrate internet is a distinguishable characteristic of all those technologies, even though they were introduced aiming diverse objectives [21, 22].

2.1.1 Virtual Local Area Network (VLAN)

A VLAN is a group of end stations on one or more Local Area Networks (LANs) that are configured to communicate with each other, regardless of their physical location. These end stations are considered logically connected to the same network while, in fact, they are located on a number of different LAN. VLANs enable the creation of several logical networks hosted among a single physical LAN or even more than one LAN by logically combining several physical LANs to appear as a single LAN. VLANs are usually associated with Internet Protocol (IP) subnetworks (i.e., all the end stations in a particular IP subnet belong to the same VLAN) [78].

2.1.2 Virtual Private Network (VPN)

A VPN is a logical private network that is built by using shared or public networks, such as the Internet [76]. VPN enables private communication among remote sites or users as if they are directly connected [75]. VPN uses virtual connections hosted over the internet from a private network to a remote site. These virtual connections are secured through the use of dedicated connections [79], virtual tunneling protocols [80], or traffic encryption. VPN can be implemented through virtualization into layer 1 VPN, Layer 2 VPN, Layer 3 VPN and/or higher layer VPNs according to the protocols used in the data plane. To be able to provide VPN at higher layers, encryption protocols should be applied [81]. However, VPNs have some limitations that make them unsuitable for every situation. Employing the tunneling technique in VPNs imposes constraints when deploying network services, since it only offers a short-term ad-hoc solution instead of providing long-term global solutions. Lack of control over the availability and performance is another significant disadvantage in VPNs. In addition, link speed usually becomes slower due to the high processing required to perform the encryption needed to maintain the valuable VPN security [82].

2.1.3 Network Overlaying

One of the widely discussed network concepts that addresses the Internet ossification problem is network overlaying. There is no need for any modification at the underlying network core since network overlaying can be deployed at the application layer. Moreover, network overlaying enables the implementation of routing and traffic management techniques, tailored to execute application services on top of the underlying wide area networks [83, 84]. Modern overlays run on top of the internet infrastructure, building application layer networks and managed packets by routing algorithms specified by the

overlay network designers. Thus, new functionalities and new services can be deployed instantly using overlay networks without the necessity of reconfiguring routers and hardware upgrades which usually use a massive amount of resources. In other words, network overlaying deploys new software on top of the existing ones without the need of deploying either new equipment nor modifying the existing routing protocols or networking software. Another advantage of network overlaying is its flexibility to choose where to place the overlay nodes, also, network overlaying combines both; packet switching and circuit switching advantages. In addition, overlay networks can dynamically adjust to overcome congestion by selecting the uncrowded network routes [83]. However, despite those advantages, there is a notable overhead costs associated with overlay path selection, path splitting, additional redundancy requirements and additional security requirements which demand a significant level of processing overhead. For these reasons, network overlaying is not widely recognized as a valuable solution for Internet ossification.

2.2 Network Virtualization

Network Virtualization allows multiple VNs to coexist over the same physical infrastructure in a seamless manner, resulting in overcoming the ossification of the Internet and providing solutions to a wide range of networking challenges [23, 29]. According to [29], network virtualization can be defined as the network environment which enables the deployment of multiple heterogeneous network architectures provided from different SPs. This can be done by sharing the same infrastructure resources through the abstraction of the logical layer above the physical resource. In other words, network virtualization allows the sharing of physical resources, although each VN operates in isolation from other collocated VNs. Network Virtualization separates services from the substrate infrastructure by introducing two new business entities: SP and Infrastructure Provider.

Infrastructure providers try to efficiently allocate a portion from their physical resources to each VN to ensure a high utilization of their SN resources. SP uses the VN to offer its end-to-end services to end users.

Virtual node is a logical entity, constructed by partitioning the resources of substrate network equipment (e.g. router), such as processing power and memory . The virtual node is capable of controlling the allocated subset of resources independently, which guarantees the isolation from other coexisting virtual nodes in the same substrate node. Similarly, a virtual link is constructed by partitioning interconnecting link resources such as bandwidth and guaranteeing the isolation from coexisting logical links, to function independently [19–22, 85]. A VN is a set of virtual nodes interconnected by virtual links. Any substrate node can host multiple virtual nodes belonging to one or more virtual networks. In other words, a particular substrate node may be hosting no or any number of virtual nodes based on location processing and node constraints, as defined by the Virtual Network Provider (VNP). Similarly, a virtual link can span over multiple substrate links when it requires interconnecting virtual nodes hosted by non-neighboring substrate nodes.

2.2.1 Virtual Network Business Model

In network virtualization, more than one separated role exists to achieve high flexibility and enhance the management and assist defining roles of each player and their interactions [86, 87]. The principal actors in current Internet are the SPs and the Internet Service Providers (ISPs). An ISP owns and manages an underlying physical infrastructure. In addition, the ISP offers customers access to the Internet by relying on its own infrastructure, by renting infrastructure from someone, or by any combination of the two (e.g., provides bit-pipes and end-to-end connectivity to end-users [87]). In other words,

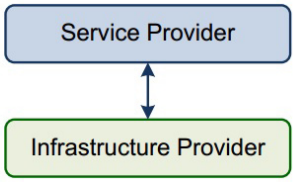

Figure 2.1: Main roles in traditional business models [1, 2]

ISPs provide a connectivity service, very often on their own infrastructure, even if they also lease part of that infrastructure to other ISPs, as shown by Fig. 2.1. The SPs offer application, data and content services to end-users. However, the distinction between these roles has often been hidden inside a single company. The ISP that is responsible for day-to-day operation of the network is rarely the one that is planning and specifying the evolution of the network.

A new level of abstraction is introduced to consider network virtualization that enables the concurrent existence of several, potentially service-tailored, networks. This leads to the redefinition of existing roles as well as the addition of new business roles. However, these business roles do not necessarily need to be handled by distinct business entities. Any business entity can perform one or more of these roles. The purpose behind the introduction of these roles is to minimize the associated costs of maintaining physical resources. The 4WARD [1, 2] network virtualization business model defines four major business roles as shown by Fig. 2.2:

- Physical Infrastructure Provider,
- Virtual Network Provider (VNP),
- Virtual Network Operator (VNO),
- Service Provider (SP).

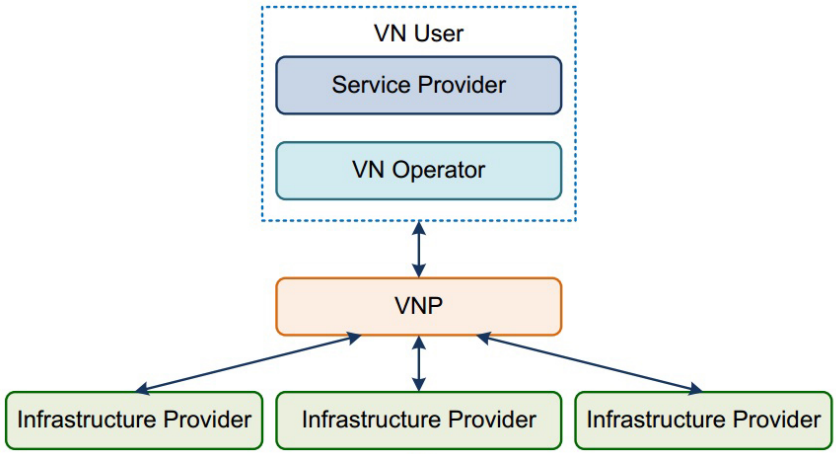

Figure 2.2: Main roles in Network virtualization business models [1, 2]

According to the 4WARD model, the Physical Infrastructure Provider owns and manages the physical infrastructure (i.e., the SN), and provides wholesale of raw bit and processing services (i.e., slices), which support network virtualization. The VNP is responsible for assembling virtual resources from one or multiple physical infrastructure providers into a virtual topology. The Virtual Network Operator (VNO) is responsible for the installation and operation of a VN over the virtual topology provided by the VNP according to the needs of the SP, and thus realizes a tailored connectivity service. The SP uses the VN to offer its service. This can be a value-added service and then the SP acts as an application service provider or a transport service with the SP acting as a network service provider.

The composition of the underlying network is transparent to VN users who deploy services, using the VNs deployed on top of the virtualization layer. Here, the 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 a VN provider who provides single VN that can be configured according to their service requirements. Hence, a VN user such as a video streaming service provider (e.g., Internet Protocol TV (IPTV)) who wants to extend its services to a remote area, only needs to negotiate with the VNP for specific hardware resources required for those service extension. The VNP will then communicate with the physical infrastructure providers to obtain the required resources. The physical infrastructure providers need to enable the creation of virtual nodes and links over their available resources and provide them to the VNP. The VNP will create the VN slice which is a lifeless set of virtual resources as requested by the VN User. The VN Users will deploy customized protocols by programming the allocated network resources with the aim of obtaining an end-to-end connectivity to provide the desired customers' services (e.g., end-users of the IPTV service). In certain circumstances such as when the system consists of multiple users, multiple VNPs and multiple physical infrastructure providers, a broker can play a vital role in the system by matching the VN users requirements with the VNP and the infrastructure providers services [88].

The previously mentioned business roles can be performed by the main actors in cloud computing, with the goal of employing network virtualization techniques in order to provide efficient network resource management. Fig. 2.3 shows the mapping between the main business roles in the Network virtualization model and Cloud environment. Throughout this thesis, each corresponding actors may be used interchangeably.

An Illustrative Example

Two possibly heterogeneous virtual networks, *VN1* and *VN2*, created by two service providers *SP1* and *SP2*, respectively, are presented in Fig. 2.4. *SP1* composed *VN1* on

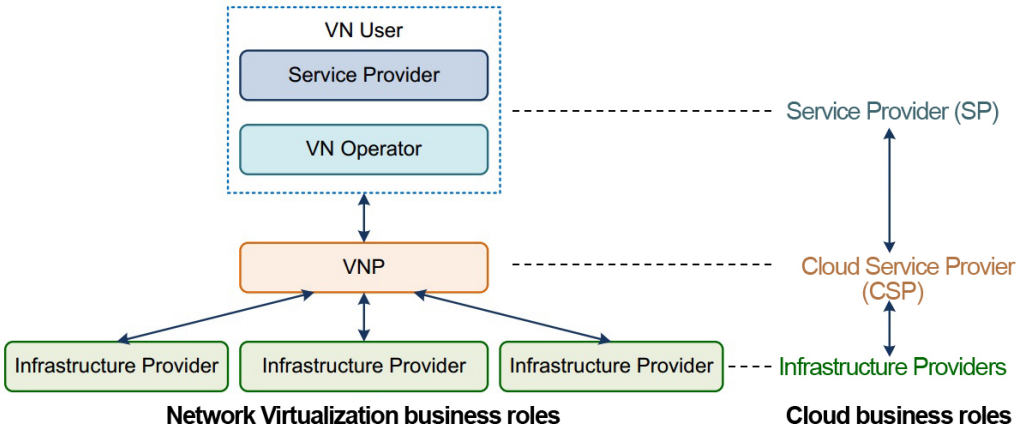

Figure 2.3: The mapping between the main business roles in the Network virtualization model and Cloud environment.

top of the substrate resources managed by two different infrastructure providers *InP1* and *InP2*, and provide end-to-end services to the end users *U2* and *U3*. *SP2*, on the other hand, deploying its virtual network *VN2* by combining resources from infrastructure provider *InP1*, with a child VN from service provider *SP1*. End users *U1* and *U3* are connected through *VN2*. The owner of a VN is free to implement any end-to-end services by selecting custom packet formats, routing protocols, forwarding mechanisms, as well as control and management planes.

2.2.2 Virtual Network Architecture

Network virtualization introduces the new virtualization layer which decouples the role of the traditional network service provider to VNP and infrastructure provider. The virtualization layer is managed by the VNP which lease resources from one or multiple infrastructure providers, to construct VN slices based on the VN user's requirements. This happens in order to deploy VNs to end-to-end connectivity and custom network solutions to the VN users. The major architectural characteristic of these VNs is the

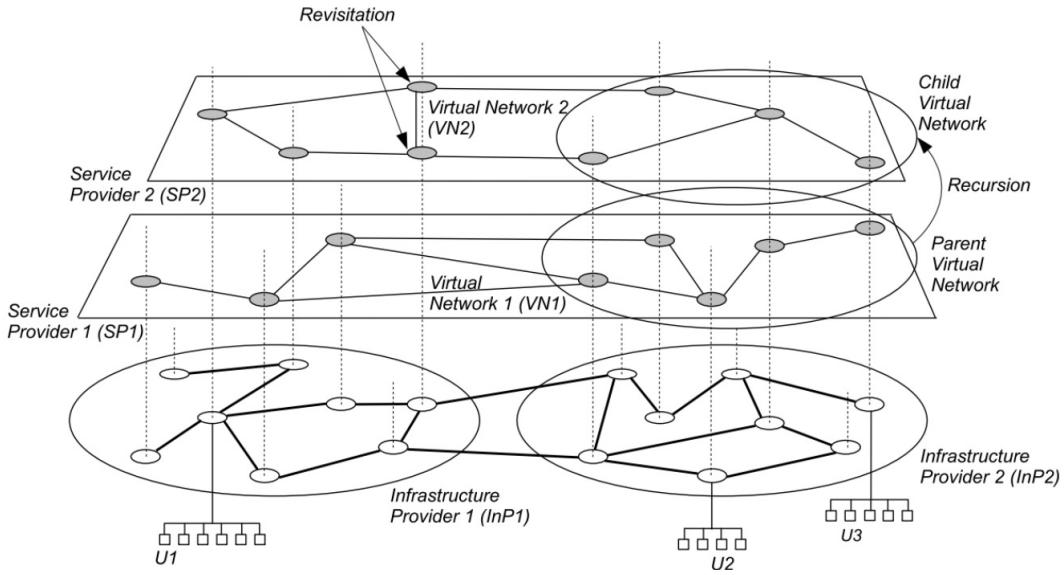

Figure 2.4: Network Virtualization roles example [1, 2]

coexistence, which allows multiple heterogeneous VNs deployed by the same or different VNPs to be executed simultaneously, yet isolated. In addition, the deployed VNs perform their networking functions on a shared substrate composed of multiple infrastructure providers. As an example, Fig. 2.5 shows coexistence of *VN1* and *VN2* that are deployed by the same VNP, on top of the physical resources of *InP1* and *InP2*. The explaination for this is because each 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 that 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. As an example, in Fig. 2.5, two virtual nodes p_1 and p_2 which respectively belong to *VN1* and *VN2* 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 unstable

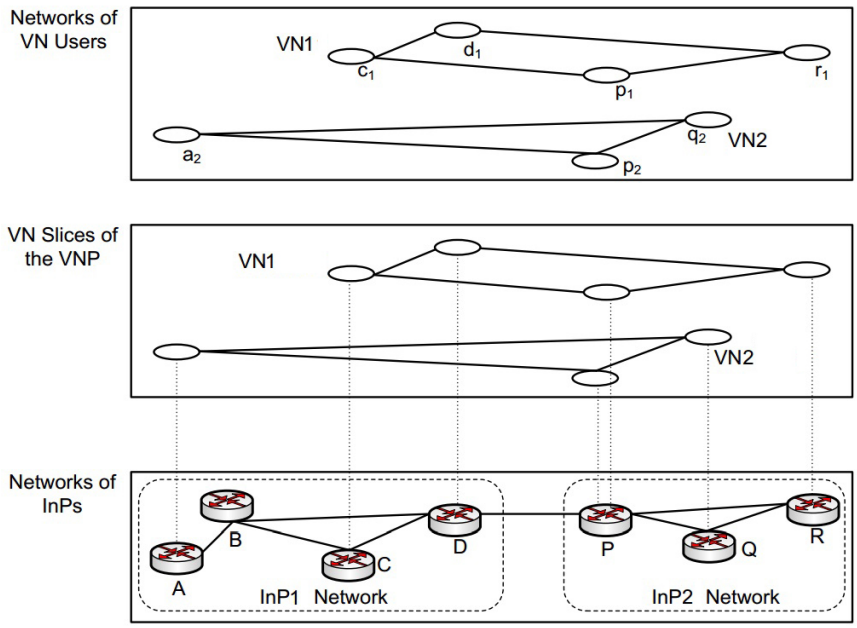

Figure 2.5: High-level Architecture of Network Virtualization

state (i.e., its queue becomes full), it should not affect the other virtual nodes residing on the same physical node. 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 coexisting 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. Let (x-y) represents the link between nodes x and y. According to Fig. 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 *VN2*. Also multiple virtual links either from the same VN or from different VNs, can use bandwidth and other link resources of the same physical links such as virtual links (d_1-r_1) and (c_1-p_1) of *VN1* and (a_2-p_2) and (a_2-q_2) of *VN2*. These links operate 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. These providers can keep core network equipment and critical topology information hidden from the VNPs in order to achieve traffic engineering and network security requirements. As an example, router B of *InP1* in Fig. 2.5 is not revealed to the VNP and hence, the VNP does not have the actual topology information of *InP1*.

2.3 Network Virtualization Techniques

In this section, a brief overview about network virtualization embedding and re-optimization techniques is presented. Also, an overview is provided of the state of the art active research that employed network virtualization techniques in the context of cloud computing, and in particular, data centers communications.

2.3.1 Embedding Approaches

The authors in [89] proposed a resource allocation scheme for the Virtual Network Resource Management System (VNRMS). Physical resources are projected onto the virtual domain to allow logical operations. Once a root VN is established, a process called spawning allows a number of child VNs to be generated. The proposal is based on the solution to the Multi Commodity Flow (MCF) problem [90]. The authors in [91] proposed a VN embedding algorithm by relaxing the Mixed Integer Linear Programming (MILP) model by introducing a stronger correlation between the node mapping and the link mapping phases. In other words, the mapping of the virtual nodes to the substrate nodes is performed in a way that facilitates the mapping of virtual links to the physical

paths in the following phase by introducing the meta-nodes to the substrate graph. The primary objective is to enhance the mapping process in order to increase the acceptance ratio and hence the revenue. The authors in [50] proposed a VN embedding technique which enables path splitting and path migration algorithms. First, the proposed algorithm uses a greedy approach to assign the nodes by sorting the VN requests based on their potential revenues, and then processes them in order of priority. For each virtual node, candidate substrate nodes are determined based on the available resources and its requirements. At the final step, for each virtual node, the substrate node with the maximum available resources, is allocated. For link mapping, node-mapped VN requests are sorted, prioritized and processed in the same way. For each virtual link, the k-shortest paths are computed [92], and the bandwidth requirements are considered. If none of the paths are able to satisfy the bandwidth requirement, the request will be put in a queue for future processing. Mapping of each virtual path exclusively to a single substrate path results in inefficient bandwidth allocation. Making physical paths shared by different virtual links improves resource use and provides more exibility and granularity in processing. Under flexible virtual link splitting over multiple physical paths [33, 93], link mapping can be reduced to a MCF problem [90]. The traffic ratio among the physical paths is known as the splitting ratio. Node remapping facilitates the path splitting process.

In [94], the authors proposed a topology-based technique (i.e., a family of backbone-star substrate network topologies with traffic constraints). The proposed technique first selects an arbitrary mapping of backbone nodes, then connects each access node to the nearest backbone node and computes the shortest paths [92] between backbone nodes and access nodes. Then the proposed technique determines each link's capacities for

traffic constraints using maximum flow computations, and finally, searches an alternative backbone node mapping with the same VN topology and link capacities. The main objective is to minimize the VN configuration cost which is modeled as a Mixed Integer Quadratic Programming (MIQP) problem. The authors in [95] proposed an approach that overcomes the VN bandwidth allocation bottleneck problem. A bottleneck occurs when the bandwidth required from a physical link exceeds the bandwidth actually available. A virtual link becomes a bottleneck when it does not have its required bandwidth, and a VN is bottlenecked when one or more of its virtual links is bottlenecked. Two types of connections are possible: restricted and unrestricted. A connection is restricted when the maximum bandwidth requirement of the current link is limited by that of a previous link. A connection is unrestricted when the maximum bandwidth requirement is limited by the current link. The fair allocation mechanism first proposes an equal bandwidth distribution among links. However, equal distribution can cause bottleneck situations when too much bandwidth is allocated to restricted connections. To prevent this, the allocation mechanism is modified to distribute the bandwidth based on the connection type. First, restricted connections are allocated bandwidth based on their requirements and then the remaining bandwidth is equally distributed among unrestricted connections. In [96], the embedding process is modeled as a MCF problem to the VN links (the embedding of virtual nodes is not considered). To avoid bottlenecks, a traffic predictor is integrated into the MCF solver in order to predict traffic forecast on the most congested physical link (the bottleneck) and allocate bandwidth based on that prediction. Traffic forecast is made with the assumption of the Poisson process traffic pattern. This assumption is made for simplicity and it does not reflect necessarily the real traffic pattern in the network virtualization environment.

The authors in [97] proposed an overbooking scheme according to SLA that determines the desired level of availability. Two levels of availability are defined: full availability of requested resources and limited availability (i.e., at least a certain share of required resources is guaranteed). The second level determines the reduction of allocated resources. In this context, authors focus on the gain borderline which represents the possibility of accommodating one extra user beyond the pre-determined allocation without violating the SLA. In [98], a resource embedding scheme is proposed. The idea is to place virtual nodes as close as possible in the physical network. A k-shortest paths algorithm [92] is used to find paths between each pair of nodes. The impact of varying k for k-shortest paths on different substrate networks is analyzed. To embed a new VN, virtual nodes with bigger resource requirements are mapped first. This is to ensure that, if the substrate network cannot satisfy the demand of the incoming VN request, the process stops without wasting more time on demands from other virtual nodes. The authors in [99] proposed a bandwidth allocation scheme based on game theory. They assume one infrastructure provider and several SPs compete on its bandwidth resources. The problem of resource allocation is expressed as a non-cooperative game model based on Nash Equilibrium. To find Nash Equilibrium, an iterative algorithm is proposed. Using a very simple scenario (a physical topology containing two nodes and two links), the authors show the convergence property of their proposal.

The authors in [100] proposed a VN embedding technique based on the concept of Subgraph isomorphism detection. Subgraph isomorphism detection between two graphs G and H consists of determining whether G contains a subgraph isomorphic to H which is an NP-complete problem [101]. In [102], the authors addressed the case when one VNP traded with multiple infrastructure providers. The resource allocation process is

divided into resource matching, embedding and binding. Several heuristic approaches are proposed [103, 104] for the VN embedding problem. The Ford-Fulkerson algorithm [103] is used to resolve the problem in case the cost of allocating a virtual link, from a peering link, is always bigger than the cost of allocating a virtual link from the same infrastructure provider.

The authors in [104] proposed a heuristic VN embedding technique by introducing the concept of hidden hops. This concept refers to the fact that intermediate nodes used to embed a virtual link (on a physical path) will require a non-negligible amount of CPU resources to configure virtual interfaces and to forward corresponding data. Hence, this CPU demand should be taken into account when performing virtual network embedding.

2.3.2 Virtual Network Re-optimization Techniques

The main focus of the network virtualization literature has been directed towards the *VN* embedding and resource discovery (e.g., [50, 51, 89, 91, 94–100, 105–107]). Also, efficient re-optimization techniques for the optimal use of the physical network resources are proposed. For example, Papagianni et al. [108] proposed a method for efficient mapping of VNs onto a shared substrate, interconnecting previously isolated islands of computing resources. The proposed mapping technique aims at minimizing the overall number of hops serving a virtual link. Periodic re-optimization of the bandwidth allocation of congested physical links is carried-out in [50] to redistribute the network resources among virtual links. In this work, virtual link migration and path-splitting percentages are used to reconfigure the VN's use of the physical resources. An alternative substrate path is selected by performing updated path-splitting percentages for the virtual links to increase the utilization of the substrate network. This technique focuses on re-optimizing

substrate network resources by migrating only the virtual links.

Y. Zhu et al. [51] proposed a periodic VN reconfiguration technique which selects a set of VNs, and re-allocates these whole VN topologies. The candidate VNs are marked for re-allocation if they are already embedded over an overloaded physical node or link.

In [49], the authors proposed a self-organizing virtual node reallocation scheme to manage the physical resources, with the goal of reducing the length of the paths serving the virtual links. Using heuristics and considering local information, distributed virtual managers detect congested virtual links and decide which virtual nodes have to migrate to new particular substrate nodes. The main objective of this technique is to reduce the length of the substrate path by migrating and hosting virtual nodes over virtual pipes. Rather than migrating virtual links, Fajjari et al. [48] proposed a greedy approach for reallocating star components within individual VNs with the objective of freeing physical resources whenever a new VN request is rejected. As will be shown in chapter 5, a technique is proposed that identifies congested VNs portions that are contributing to the fragmentation of the substrate resources, then re-allocates these portions into other underutilized resources. N. Butt et al. [52] presented a technique to differentiate among substrate resources based on their importance. Also, the authors proposed a reactive approach to detect and fix the bottleneck resources by freeing up some capacity on the nodes/links which cause VN request rejection (re-optimization is invoked whenever a VN request gets rejected). I. Fajjari at al. [32] proposed a greedy approach for virtual network reconfigurations by migrating a set of star moving candidates in order to minimize the number of congested substrate links, each time a VN request is rejected.

Reallocating congested VN portions aims to minimize the data centers' network fragmentation problem and leaves room for future VN requests. In general, while com-

plementary to the proposed work, the aforementioned approaches focus mostly on the efficient utilization of the physical network rather than on enhancing the performance of the hosted VNs.

2.3.3 Dynamic Virtual Network Resource Management

Another significant research effort focuses on the dynamic provisioning of network resources to accurately reflect the time-varying dynamic needs of the hosted applications and to provide efficiency resource allocation. He et al. [33] proposed DaVinci, an adaptive VN resource allocation framework where the virtualized resources are dynamically adapted based on the CSPs' knowledge of the VNs performance objectives. Fajjari et al. [109] proposed a dynamic adaptive VN resource allocation approach named Adaptive-VNE. This approach utilizes the unused bandwidth in order to be reassigned to incoming virtual network requests according to the occupancy rate of the embedded virtual links. Zhou et al. [99] proposed a dynamic VN allocation scheme that allocates the bandwidth among multiple VNs based on their payoff function. Each payoff function is composed of three parts; a utility function, a pricing function and a congestion function. However, this scheme focuses mainly on the interaction between the infrastructure providers and the SPs. In [96], the authors proposed using MCF solver to embed virtual links based on an periodical traffic prediction to adjust the links' bandwidth allocation with the largest occupation. In [95], the authors proposed an adaptive bandwidth allocation to avoid bottleneck substrate links. First, the proposed algorithm allocates bandwidth to restricted connections based on their requirements. Then the remaining bandwidth is distributed equally among unrestricted connections.

In [110] the authors proposed a dynamic resource reconfiguration approach to achieve high resource utilization and to increase the infrastructure providers' income. The proposed VNR-GA approach handles dynamic VN users' needs and adapts the VN resources according to the applications' requirements. The approach is based on Genetic meta-heuristic and applies resources migration techniques to reallocate the resource of already instantiated VNs. In [111], a new dynamic VN model is proposed that determines the users' current demands based on their utility functions. Accordingly, the VN users determine their requested resources based on a *time-of-use* pricing. However, these two approaches do not explicitly provide any service guarantees.

2.4 Resource Allocation in Data Center Networks

In the previous sections, a review has been conduced of network virtualization techniques that are used for general networks. In this particular section, a review is made of some of the presented network virtualization approaches that are employed in cloud computing environments.

2.4.1 Data Center Networks Virtualization

Current research efforts focus on intra-data center networks and employ virtualization to provide bandwidth sharing between single data center tenants (cloud services). For example, Guo et al. [54] proposed SecondNet, a low time-complexity heuristics-based algorithm that allocates a data center VN using clustered neighboring servers. Current demands are met by increasing or decreasing the bandwidth reservation along existing paths, or through migrating virtual links to new paths. The bandwidth guarantees are

provided under the assumption that each data center's structure is known in advance and is managed by a single entity. As a result, its performance strongly depends on the topology of the physical network [61]. Similarly, Rodrigues et al. [55] proposed GateKeeper, which provides QoS guarantees and high bandwidth utilization among virtual links while achieving high bandwidth utilization of the network. To ensure high bandwidth utilization, GateKeeper sets two bandwidth rates for each virtual link (i.e., minimum guaranteed rate and maximum allowed rate). According to the level of congestion of the network, the sender may control its traffic rate. Unfortunately, Gatekeeper does not consider other QoS properties apart from bandwidth and it lacks maturity. Benson et al. [56] proposed CloudNaaS, a framework that supports the deployment and management of cloud enterprise applications. The architecture supports both best-effort and bandwidth reservation based services among VMs. Nonetheless, the increase in the number of paths with bandwidth reservation may lead to congestion/starvation of other services, in addition to an inefficient network utilization. Another bandwidth allocation technique, termed Seawall [36], performs end-to-end proportional sharing of the network resources according to an associated weight with each traffic source. This technique provides bandwidth isolation using congestion-controlled tunnels implemented in hosts.

Similarly, Lam et al. [38] proposed Netshare, a statistical multiplexing mechanism that proportionally allocates bandwidth for the tenants. Unlike Seawall, Netshare shares link bandwidth among tenants rather than sender virtual machines to provide a fair allocation among different services. Unfortunately, this approach can not be deployed using any switches, due to the requirement of special features enabled by Fulcrum switches. Moreover, using the fairness mechanism excludes providing bandwidth guarantees. The PortLand approach [112] addresses the issues of VMs' population scalability, migration

and management. Its main limitations are a requirement for fat-tree multi-rooted physical topology and lack of bandwidth guarantees. The Oktopus approach [60] provides performance guarantees and separate VN abstractions for tenants in tree-like physical topologies. An extension to Oktopus is proposed by [113] where the tenant specifies the corresponding bandwidth demand for each VM assuming that the bandwidth requests from the VMs are heterogeneous. In [114], the authors proposed an architecture of dynamic bandwidth allocation and guarantee on resource fairness, to provide network performance isolation and protocol independent bandwidth guarantees. This approach dynamically limits the rate of each flow at the network edge. The NetLord architecture [115] relies on L2-in-L3 encapsulation and provides good scalability, logical isolation between tenants, low performance overheads, and centralized management. Unfortunately, NetLord does not provide any bandwidth or QoS guarantees.

In general, the aforementioned approaches are not suitable for inter-data centers communications. They also do not provide the VN users (i.e., SP) with the means to accurately define their network resource demands. In addition, they do not focus on mechanisms that relate the pricing of the VN service with the profit gained from these active distributed cloud applications. Finally, a number of these approaches assume that the CSP has full knowledge of the traffic patterns of the hosted cloud services in advance.

2.4.2 Data Center Network Elasticity

To date, the majority of the existing techniques in the literature that target elasticity for the IaaS service model, focus on computing, storage, and network resources within the same data center, with little attention to inter-data center network resources elasticity. Several commercial and academic elasticity approaches offer elasticity features that

can be categorized as either rule-based reactive approaches, or predictive approaches. The former defines the set of actions that should be taken when one or multiple observed parameters (e.g., CPU usage, network traffic, disk access) exceed/fall-behind a pre-defined threshold. The latter adjusts the allocated resources based on the prediction of future violations and workload forecasts. *Auto-Scaling* offered by Amazon Web Services (AWS) [116] is based on a rule-based reactive approach that adds/releases cloud instances based on monitored parameters. Lim et al. [117] proposed a rule-based reactive approach where elasticity actions take place according to a target resource demand range, rather than a single target value, to avoid resource thrashing (i.e., constantly adding/releasing cloud instances).

Dawoud et al. [118] proposed a dynamic resource provisioning approach that aims at minimizing the resources required to handle the future workload by following a linear prediction approach that decides the next allocation based on the the last allocation and consumption. Roy et al. [119] proposed a predictive approach for workload forecasting that aims at minimizing the resource provisioning costs while guaranteeing the application QoS. Gong et al. [42] proposed PRESS, a predictive elasticity scheme that is based on resource demand patterns prediction through Fast Fourier Transform FFT, to minimize the unused resources while preventing Service Level Objectives (SLO) violations. Sharma et al. [43] proposed *Kingfisher*, an elasticity provisioning approach that takes into consideration workload fluctuation while trying to minimize the virtual resources costs. *Kingfisher* computes the optimal resource provisioning based on the cost of migrating and/or replicating VM instances, also considering the transition time to allocate more resources.

Rao et al. [40] proposed a self-tuning fuzzy control rule-based multi-objective resource allocation approach that guarantees cloud applications QoS. Netflix is using *Scryer* [120], a predictive auto-scaling engine to deliver the best Quality of Experience (QoE) to Netflix customers. *Scryer* predicts upcoming workload based on two prediction algorithms, namely, an augmented linear regression-based and an FFT based algorithms. Li et al. [41] proposed a resource management approach that consider future demand fluctuations while provisioning the physical machines and links capacities.

Research efforts that are related to data center network provisioning mainly focus on intra-data center network by applying data center network virtualization. Guo et al. [54] proposed SecondNet, a low time-complexity heuristics-based algorithm that allocates a virtual data center using clustered neighboring servers. Current demands are met by increasing or decreasing the bandwidth reservation along existing paths or through migrating virtual links to new paths. Similarly, Rodrigues et al. [55] proposed GateKeeper which provides guaranteed bandwidth among virtual links while achieving high bandwidth utilization of the network. To ensure high bandwidth utilization, GateKeeper sets two bandwidth rates for each virtual link (i.e., minimum guaranteed rate and maximum allowed rate). According to the congestion level of the network, the sender may control its traffic rate.

Benson et al. [56] proposed CloudNaaS, a framework that supports the deployment and management of cloud enterprise applications. The architecture supports both best-effort and bandwidth reservation based services among VMs. Nonetheless, the increase in the number of paths with bandwidth reservation may lead to congestion/starvation of other services , and inefficient network utilization. Another bandwidth allocation technique, termed Seawall [36], performs end-to-end proportional sharing of the network

resources according to an associated weight with each traffic source using congestion-controlled tunnels. Similarly, Lam et al. [38] proposed Netshare, a statistical multiplexing mechanism that proportionally allocates bandwidth for the tenants.

To date, a minimal research effort has addressed the problems of dynamic resource provisioning of inter-data centers connectivity. One example is the work of Ajay et al. [17] which proposed an architecture to enable the dynamic configuration of the inter-data centers network by offering a bandwidth-on-demand service. Ghosh et al. [121] suggested an inter-data centers traffic management design by optimizing the sending rates and controlling the network routing, across multiple application traffic classes. Carella el al. [122] proposed an elasticity engine that is based on brokerage of cloud resources among multiple CSPs, in order to optimally allocate the cloud resources in federated cloud environments. In general, the aforementioned approaches do not consider the problem related to the availability of inter-data centers network resources and the issue of accurately estimating the reserved pool size needed in order to provide network elasticity.

2.4.3 Bandwidth Uncertainty

Few works consider the demand uncertainty of VN users and how to estimate the reserved pool size needed to provide network Elasticity. In [123], the authors proposed an approach that estimates how much bandwidth is required to be reserved in the case of VoD provider. This approach predicts the demand statistics of VoD providers in the near future based on demand history available from cloud monitoring services. The demands of different VoD providers are mixed, based on anti-correlation and probabilistic bandwidth guarantees which are sold to the VoD providers. In [58], the authors proposed that

the SP simply needs to specify a percentage of its bandwidth demand to be served with guaranteed performance. From the historical workload data, CSP can perform statistical learning to predict SP demands and make bandwidth reservations. Divakaran et al. [59] proposed a probabilistic model where bandwidth requirements are specified along with some probabilities. The SPs have to perform a rough estimate of the bandwidth needs, as well as, the duration of the request.

In [124], the authors proposed a bandwidth allocation approach that considers both bandwidth-guarantee (BG) and time-guarantee (TG) requests. In addition, they proposed the bandwidth demand as a function of time to reflect the time-varying bandwidth demands. Yu et al. [57] proposed a Stochastic Virtual Cluster (SVC) which probabilistically models the bandwidth demand uncertainty. In [125], the authors proposed an approach to automatically derive the time-varying nature of the networking requirement of cloud applications using bandwidth capping. In general, the aforementioned approaches do not consider the case of differentiated applications sensitivity to bandwidth availability. Also, the demand repetitive patterns are not considered during the estimation process which introduces a great computation overhead and requires the re-estimation of the required bandwidth each evaluation period.

2.5 Sources of Latency on Inter-data Centers Communications

Cloud-based applications typically fall into two categories: throughput-oriented applications (e.g., file transfer or email applications) which are not sensitive to the delivery times of individual packets and latency-sensitive applications (e.g., online gaming and algorithmic trading [17,121]) which are sensitive to network latency [126]. Recently, with the increase in the number of cloud-based latency-sensitive applications, CSPs strive to

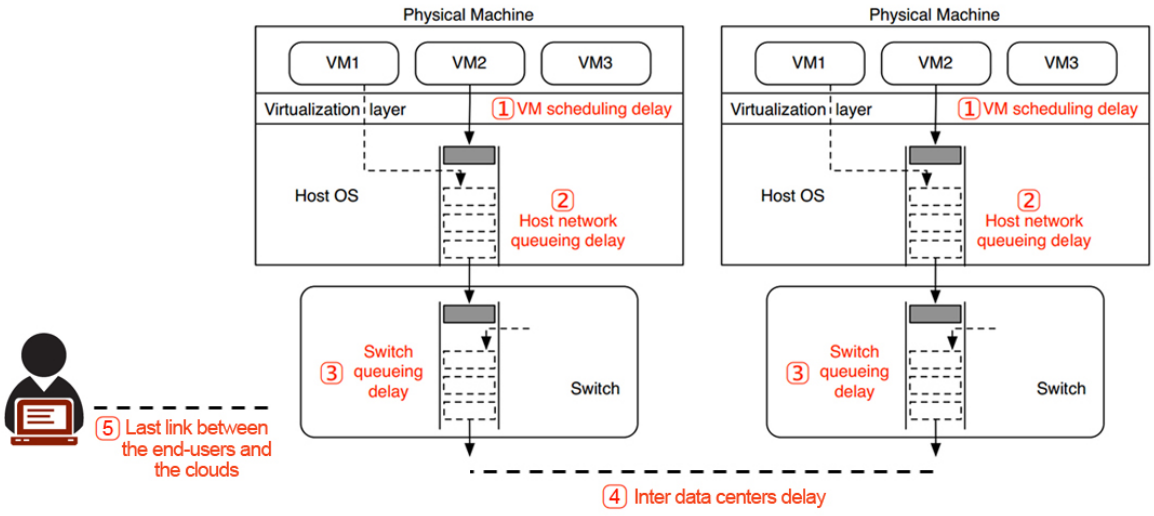

Figure 2.6: Sources of Latency [3]

reduce the latency of the data center's internal network, as well as the inter-data centers communications in order to provide a predictable performance [3]. Also, network latency impacts directly the QoE of the end-users and strongly affects the CSPs' revenue [127]. There are several sources of latency in large-scale, geographically dispersed, public cloud environment as shown by Fig.2.6.

To see how current cloud-based applications have restricted latency requirements, here are some latency values according to [128]. The access latency required to offer an efficient remote desktop below 50 ms to get accepted quality of experience. This constraint is almost impossible to satisfy if the end-user lives in the USA and is requesting information from a data center located in Europe. Ultra latency-sensitive applications such as online gaming pushes cloud access latency down to at least 30 ms. Applications which require real-time image and/or voice recognition will almost certainly require cloud access latency to be down to no more than a few ms. In addition, the inter-data centers communication between virtualized nodes/links requires the minimum possible latency between access equipment. To be able to identify the different sources of delays that

current cloud-based applications may experience, consider a scenario where a client VM sends queries to a server VM. The latter is located in a different data center and needs to perform the request of an end-user. The time from when the client sends the query and receives the complete response is defined as Flow Completion Time (FCT) [3]. Xu et al. [3] evaluated the VM scheduling delay and the switch queueing delay using live AmazonElastic Compute Cloud (EC2) experiments. Here are a list of possible sources of delay:

- VM scheduling delay
- Host network queueing delay Network Interface Cards (NIC)
- Switch queueing delay
- Last link between the end-users and the clouds
- Inter-data centers latency

VM Scheduling Delay

When the physical machine which hosts the VM receives a packet, that VM get notified. However, the actual processing can not start until this packet is scheduled by Hypervisor [129]. VM scheduling delay is usually less than one millisecond. However, in the case of many VMs sharing the same physical machine, it may become as large as tens of milliseconds [130]. Due to VM scheduling delay, there is a large tail latency between EC2 instances [131].

Host Network Queueing Delay

The I/O requests of all the hosted VMs should first go through the host network stack to be processed. The host network stack is another source of excessive latency due to

processing several requests which may result in filling up its queues. Techniques like kernel bypass and zero copy [132, 133] are able to reduce the end-host and the NICs latencies (e.g., Current 10Gbps NICs can achieve less than 1.5μs per packet latency at the end-host [126]). In addition, dedicated virtual queues for each VM with an appropriate packet pacing that corresponds to the allocated service rate can enhance the link latency. Also allowing a small unused fraction of the allocated bandwidth (typically $0 - 10\%$) to act as a headroom for traffic burstiness can enhance the link latency [17, 134]. These mechanisms ensure that the specified latency is achieved as long as VMs abid by the specified traffic rates [135].

Switch Queueing Delay

Packets on the wire may experience switch queueing delay on congested links. If a VM is sharing the same physical intra-data center link with another VM that receives/sends large bursts of traffic, this former VM may experience queueing delays on its access link, even if it is only using a small fraction of the bandwidth assigned to it.

Last Link Between the End-users and the Clouds

Packets also may experience another source of delays which is attributed to the last link between the end-users and the clouds. To solve this kind of delay, it involves building geographically distributed data centers placed as close as possible to the end-users and developing VM placement mechanisms that focus on meeting the end-users' latency constraints [14, 136]. The problem of intra-data center communication has been studied extensively [36, 38]. For example, Facebook runs at least 3 datacenters [137] and holds more than 6 billion photographs [138]. To maintain a high QoE for its customers, a new collection needs to be quickly replicated at sites closer to the users [127].

Inter-data Centers Latency

Packets may also experience another source of delays which is attributed to the congestion of inter-data centers links. The lack of inter-data center bandwidth management solutions still remains a pressing problem particularly for ultra latency-sensitive distributed cloud applications. These sources of delays can dramatically increase the overall experienced latency.

2.6 Summary

Chapter 2 has addressed a global view of the main concepts related to network virtualization and compared these concepts with related concepts. An overview of network virtualization's features and the necessary background information required to clearly introduce this thesis work, was presented. In addition, an overview of the state of the art active research initiatives in the areas of dynamic VN resource management and SN re-optimization techniques was presented. In the following chapter, this thesis' proposed VNaaS model which provides differentiated VN service qualities, will be presented.

Chapter 3

Differentiated Virtual-Network as-a Service

The previous chapter introduced some of the related work that exists in regard to the resource management of network virtualization in general. Currently, Cloud Customers (CCs) who offer the cloud-based services, cannot control the quality, and in particular, the experienced latency of their distributed cloud applications due to the unpredictability of the communication links among their hosting distributed data centers. To overcome this problem, chapter 3 describes a novel VNaaS model to host these applications. In contrast to existing randomly or statically provisioned inter-data center bandwidth sharing models, using the proposed model, the CCs can accurately express their varying network resource needs, demand constraints, and tolerance to the cloud latency. More precisely, the proposed differentiated VNaaS allows the flexibility to the CC to specify the desired time-varying bandwidth needs in order to create virtual links, connecting their distributed VMs, comprised of multiple virtual pipes with differentiated service qualities.

Section 3.1 gives a general overview, presenting some of the challenges and limitations of existing inter-data center bandwidth provisioning models. Section 3.2 highlights the proposed differentiated VNaaS model. This is followed by the development of a novel two stage-budgeting mechanism to aid the CC in optimally demanding the appropriate VNaaS resources for each virtual pipe. This is carried out according to the hosted application utility and the CC's budget constraints. In the first budgeting stage, the CC

calculates an optimal *effective service rate* for each virtual edge used for communication between the distributed virtual machines, in addition to a corresponding edge budget and a price index. In the second stage, the virtual link budget is distributed to purchase bandwidth for the link's virtual pipes, each with different service quality and pricing. The proposed two stage-budgeting and demand allocation for the differentiated VNaaS is described in Section 3.3. This section also extends the proposed model to allow the CC to enforce virtual link capacity constraints on the effective service rates resulting from the traffic matrix on the VNaaS . The proposed resource pricing and monitoring techniques are presented in Section 3.4. Finally, performance evaluation results of the proposed model, along with the novel demand allocation and pricing methods are discussed in Section 4.7 while Section 3.8 concludes the chapter.

3.1 Overview

The past few years have seen a rapid growth of large-scale clouds that are geographically dispersed. These infrastructures have resulted in a wide range of novel distributed cloud-based applications such as algorithmic trading, Google Docs sharing, Netflix's video streaming, virtual enterprises, MapReduce applications, Facebook's social networking and compute-intensive scientific experimentations [10–12]. Unfortunately, CSPs, thus far, have failed to provide any sort of guarantees on the quality, and in particular, the latency, of these distributed cloud applications. This is largely attributed to the unpredictability of the communication links among their hosting distributed data centers [12,27,28]. Nonetheless, due to the fierce competition in the cloud computing market, any slight reduction in the quality of the delivered services results in a significant loss in revenue [28].

The first step taken by the CSPs to solve this problem addresses the latency attributed to the last link between the end-users and the clouds. This step involves building geographically distributed data centers placed as close as possible to the end-users and developing VM placement mechanisms that focus on meeting the end-users' latency constraints [7, 14, 15, 136]. The more difficult second step in the solution involves reducing the inter-data centers latency occurring over backbone networks that are owned or leased by the CSPs [135]. While the problem of intra-data center communication has been studied extensively [36, 38], the lack of inter-data center bandwidth management solutions still remains a pressing problem, particularly for ultra latency-sensitive distributed cloud applications (e.g., online gaming and algorithmic trading) [17, 121].

In this chapter, a novel inter-data centers resource management approach is proposed based on network virtualization [29, 139, 140]. Ideally, virtualization allows each CC to rent the bandwidth needed for its inter-data centers communication in order to dynamically create a VN that operates in isolation from other collocated cloud applications. Similar to other cloud resources, CCs (e.g., video on demand providers such as Netflix, virtual enterprise operators and individual users) should be able to specify and pay for the communication resources used by their cloud-hosted applications. This is carried out in terms of the used bandwidth in addition to the required and achieved latency. Unfortunately, traditional VN embedding approaches, thus far, statically allocate a fixed amount of bandwidth for each virtual link as the basic mean for service quality guarantees [29]. While these approaches are optimized for short-lived and small VNs, they might not be suitable for large-scale distributed cloud services. First, they require the CCs to firmly estimate, reserve and pay for bandwidth using expected peak-traffic values. This can lead to unnecessarily higher cloud-services prices. Secondly, CCs are not al-

lowed to dynamically down- or up-scale the amount of resources needed for the duration of their services, even with some efforts to dynamically provision the rented resources among hosted VNs (e.g., [33, 34]), including those that target cloud environments (e.g., Seawall [36] and NetShare [38]). Another important observation is that the majority of existing virtualization techniques do not take into consideration the nature of the cloud services when making the allocation decisions. Clearly, latency-sensitive applications such as online gaming require rigid resource allocation. However, other delay-tolerant applications (e.g., VM image distribution, movement of big data in the cloud and offline laboratory experiments [39]) can accommodate some experienced network latency where the quality of delivery network is not critical to their performance, especially if this brings along a price reduction. Also, it is worth noting that while the availability of IaaS services has freed SPs (e.g., Netflix) from the continuous tasks of network provisioning and traffic management, corresponding services are still not offered by IaaS providers. In addition, bandwidth over-provisioning remains the main solution for applications with stringent bandwidth requirements such as VoD applications [123], leading to additional unnecessary cloud service costs.

In general, while the problem of bandwidth allocation for intra-data center VM communication has been studied extensively [36, 38], the lack of inter-data center bandwidth management solutions still remains a pressing problem, particularly for ultra latency-sensitive distributed cloud applications (e.g., online gaming and algorithmic trading) [17, 121, 141]. Also, little work has been proposed (e.g., [26]) aimed at developing appropriate mechanisms that can help the CCs efficiently allocate their operating budgets to rent the right amount of the cloud resources according to their expected applications' demands.

3.1.1 Contributions

In this chapter, a novel quality differentiated cloud-based VN service model is proposed that can accurately reflect the actual CC dynamic cloud networking needs. This is carried out while incorporating two emerging demands, namely, the existence of geographical VM placement constraints [142] as well as resource demand determination subject to a pre-existing CC budget [26]. Meanwhile, the proposed model allows the CSP to maximize its profit through the added-value of the VNaaS and price differentiation. The necessary mechanisms to aid the CC in specifying its VN demand choices and the CSP in realizing this service is also implemented. To this end, the contributions of the proposed work can be summarized as follows:

- A new VNaaS model is proposed that allows the CSP to offer differentiated network-aware cloud services to its CCs . In this model, each virtual link in the CC's VN between any two data centers is split into a number of virtual pipes, each with a specific expected latency and a corresponding differentiated price.

- The CC's demand for a VNaaS is formulated as an optimization problem in which the CC seeks to request the optimal bandwidth resources for each virtual pipe. This is done in order to maximize its utility obtained from serving the cloud customers, subject to the VNaaS pre-allocated budget constraints. Then analytically, it is demonstrated how the complexity of this problem can be reduced, by posing it as a two-stage budget allocation problem. For the first-stage budgeting, a closed formulation is derived to calculate an optimal price index and an effective bandwidth for each virtual link in the VN . In the second stage, the obtained effective bandwidth vector and price indices are employed to derive a formulation to extend the solution of the original problem to include additional internal routing constraints

resulting from servicing their own end-users (e.g., employees in a virtual enterprise or end-users of an online-gaming service).

- Finally, a novel Ramsey-based [68] latency-aware cloud resource pricing mechanism is proposed to efficiently regulate the cloud demands while giving the CSP an incentive to offer service differentiation. The proposed mechanism employs dynamic network resource pricing to regulate the demand of incoming VNaaS requests.

3.2 A Differentiated Virtual-Network-as-a-Service (VNaaS) Model

The proposed work focuses on CCs with large-scale distributed cloud hosted applications [11, 15] (e.g., Netflix video streaming, algorithmic trading, online gaming, scientific experimentation, and social networking) that rely on the cloud infrastructure to run their services to the end-users. Hence, a CSP that owns a number of geographically distributed data centers connected through the CSP's owned or rented high-capacity network, is considered [17, 121, 141] . In turn, the CCs' applications are executed on top of VMs distributed on those data centers. These VMs are placed as close as possible to the end-users in order to reduce the last link latency using efficient VM placement techniques [27]. More precisely, the CCs rent a number of VMs communicating with each other and with virtual switches and routers, that are needed to perform some additional network functionalities (e.g., packet re-routing and provisioning), using the CSP's distributed cloud. The latter can be modeled as a weighted undirected, possibly complete, graph $G(N, L)$, where N and L represent the sets of data centers and inter-data centers links, respectively. c_i denotes the typically large capacity of $e_i \in L$. The CSP sells the cloud resources in the form of VNs to the CCs . A rented VN, v, is modeled as a weighted undirected, graph $G^v(N^v, E^v)$, with the sets of virtual nodes, N^v, and virtual links, E^v such that $m^v{=}|E^v|$ [26]. Since the main focus of this chapter is to manage the

CC's inter-data centers traffic, a single virtual node in the set N^v is used to collectively refer to all the resources (e.g., a number of VMs or storage units) that are rented by the CC in a given data center in the set N. The allocation of the these VMs is assumed to be performed by employing appropriate VM placement mechanisms that satisfy the CC's geographical constraints and minimize the end-users experienced latency [7, 14, 15]. Similarly, a virtual link in E^v is used to host all the CC's traffic between two data centers. Finally, as previously indicated in the model, the proposed work focuses on CCs that rent VNs for a long-term to offer various services to end-users (e.g., video on demand and online gaming).

3.2.1 Static Virtual Network (VN) Models and Their Limitations

Existing VM placement [27] and VN embedding approaches (e.g., [31], [48]), map the VN resources (i.e., VMs and virtual links) into corresponding physical resources (i.e., physical nodes and paths) by reserving for every virtual link $e^v \in E^v$ a static bandwidth demand b_e^v that was estimated by the VN user throughout the VN life-time. This assumption becomes unrealistic in cloud environments; on the one hand, CCs may not be able to accurately estimate their demands before service execution. However, such a static demand, may result in overcharging the CCs for resources that they are not continuously used low inter-VM traffic while denying their increased resource requests during high-peaks of VM communication [28]. In addition, these CCs usually dedicate a predefined budget that depends on the profit obtained from selling its cloud-hosted applications, and hence, the VN resources must be allocated while satisfying this budget constraint. This budget constraint is usually relaxed in existing approaches.

Finally, static VN many not provide the optimal profit for CSPs due to resource over-provisioning and fragmentation [108, 111, 143]. In the following section, a novel VNaaS model is proposed that overcomes these limitations.

3.2.2 Proposed VNaaS Model

In the proposed model the CSP allows the CCs to define each virtual link in terms of a set of smaller virtual pipes, each receiving a different latency guarantee and price, rather than offering a single virtual link service with a static resource reservation. This cost model is more suitable for cloud services where CCs only pay for the service quality they receive. It is also suitable for large-scale clients (e.g., virtual enterprises and social network providers) with a wide range of traffic types.

Formally, the CSP offers J (≥ 2) classes of service to serve the inter-data centers traffic of the CCs ; each class $j \in \{1, \cdots, J\}$ is characterized with a measure of service latency ϵ_j, as will be explained in the following section. Class J provides the least latency, resulting mainly from the geographical distance between the two communicating data centers.

It is worth noting here that, while the experienced latency between any two data centers is bounded by their geographical distance, factors such as buffering delays, link congestion and switching at the source/destination can dramatically increase the overall experienced latency. Based on this, recent research approaches are followed (e.g., [144]) that rely on several mechanisms implemented in current data centers, namely, the use of virtual network interface cards with kernel bypass to reduce the switching latency [144]. In addition, dedicated virtual queues for each virtual pipe with an appropriate packet pacing that corresponds to the allocated service rate while allowing a small unused

fraction for each service class (typically $0 - 10\%$) of the allocated bandwidth to act as a headroom for traffic burstiness [17, 134]. These mechanisms ensure that the specified latency is achieved as long as VMs abide by the specified traffic rates [141].

Using the above latency classes, the CSP negotiates with each CC a VNaaS v as defined in terms of a SLA, and that can be defined as follows,

$$SLA^v(G^v, \underline{\boldsymbol{b}}^v, \overline{\boldsymbol{b}}^v, T^v, \Upsilon^v, \boldsymbol{\epsilon}) \tag{3.1}$$

In the above SLA for a VNaaS v, the CC describes the desired topology, $G^v(N^v, E^v)$, with m^v virtual links and the expected lower and upper bounds (i.e., out-of- and in-peak demands, respectively) for the requested bandwidth for each virtual link rather than a fixed demand, as defined by the vectors $\underline{\boldsymbol{b}}^v$, $\overline{\boldsymbol{b}}^v$, respectively. The CC also specifies the maximum value of the monetary budget, Υ^v, that it is willing to allocate per each time interval during the VN service lifetime T^v. Given the VNaaS requirements, the CSP, provides the CC with a negotiated service level (in terms of the expected latency) for each class of service as defined by the vector $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_J)$. It is worth noting here that, any SLA negotiation protocol can be used during the SLA negotiation process of the proposed VNaaS model [145–148].

Once the SLA becomes active, each CC uses these classes to describe its actual resource demands at each time interval [1] through a demand matrix $D^v \equiv [d^v_{ij}]$. The matrix element d^v_{ij}, is the demanded bandwidth on a virtual pipe hosted on the CSP's link $e_i \in E$ between two data centers in two different geographical areas requested by the CC for v and receiving a priority j. More precisely, at each time interval the CSP posts its current prices, as defined by the matrix $P \equiv [p_{ij}]$, where p_{ij} is the price of a unit of bandwidth

[1] The duration of these time intervals is specified by the CSP and depends mostly on the nature of the hosted VNs [111].

for a virtual pipe in class j for link e_i. Then, the CC uses P to calculate D^v. Whenever it is more convenient, the vectors $\boldsymbol{d}_i^v$ and $\boldsymbol{p}_i$ will be used, to express the demand and prices for virtual link e_i^v in all classes, respectively. The CC calculates these demands such that they maximize its net benefit by using these resources to serve its end-users.

Similar to existing approaches in literature (e.g., [111, 149, 150]), a utility function $U^v(\boldsymbol{x}^v)$ is used, where $\boldsymbol{x}^v = (x_1^v, \cdots, x_{m^v}^v)$ in order to quantitatively measure the CC's benefit from obtaining x_i^v, the *effective bandwidth* of a virtual link $e_i^v \in E^v$. Clearly, x_i^v depends on the demanded bandwidth in each virtual pipe and the achieved latency for traffic in that pipe, i.e., $x_i^v = f_i^v(\boldsymbol{d}_i^v, \boldsymbol{\epsilon})$. Hence, $f_i^v(.)$ is used by the CC to describe the criticality of traffic (e.g., background service analytics versus client-related traffic) in each pipe. A CC may change this utility function at each time interval to accurately reflect the nature of its currently active cloud services. The estimation of the expected traffic and the utility functions $U^v(.)$ and f_i^v is carried out by the CC by monitoring the performance of its services that are hosted on G^v using techniques such as that described in [123].

The CC also internally maintains a routing constraints matrix R^v, with rank $0 < n^v < m^v$, that describes a mandatory relationship among the effective bandwidth allocated to virtual links, i.e., $R^v\boldsymbol{x^v}{=}\boldsymbol{k}^v$. This matrix is automatically generated by the employed routing mechanism over the VN and ensures the consistency between the management operations on the VN and the allocated physical resources [26, 151, 152]. Clearly, the relations $R^v\underline{\boldsymbol{b}}^v{=}\boldsymbol{k}^v$ and $R^v\bar{\boldsymbol{b}}^v{=}\boldsymbol{k}^v$ must hold. The incorporation of the constraints of this matrix stems from the fact that most applications run as a collection of VMs that communicate to jointly provide a service or to compute a common task (e.g., a multi-

tier web application with VMs implementing a front end, running application server and databases [153]). Hence the variations in the traffic and/or the available resources on each virtual link connecting these VMs must be dependent. This matrix can also be used to define resource demand dependencies similar to the work in [154].

An example of a VNaaS topology with these constraints is depicted in Fig. 3.1. A discussion on the role of these constraints is given in the following section.

The utility function $U^v(.)$ is assumed to be quasi-concave, increasing and twice differentiable in the domain defined by $x_i^v \in [\underline{b}_i^v, \overline{b}_i^v]$, for all $e_i^v \in E^v$, following existing approaches in the literature [155]. In addition, $f_i^v(.)$ is assumed to also be non-decreasing, continuous and differentiable in all d_{ij}^v.

Table 3.1 provides a description of the adopted notations.

This chapter focuses on VNaaS serving CCs that are characterized with relatively longer life-times and time varying demands. Hence, in contrast to existing approaches that address only the initial phase of embedding the VNs, this work focuses on the ongoing process of continuously re-optimizing the physical network resources allocated to the VNaaS. This is carried out with the objective of minimizing the CC's cost of purchasing the VNaaS and meeting its budget constraints while maximizing both the CC's net utility and the CSP's profit. How the CC determines its demand matrix D^v throughout the lifetime of the VNaaS is then illustrated in this chapter.

Table 3.1: A summary of adopted notations (VNaaS).

Variable	Description
$G(N, L)$	the graph representing the CSP's geo-distributed Data centers network.
N	the sets of data centers.
L	the sets of inter-data centers links.
c_i	the capacity of inter-data centers link $e_i \in L$.
$\boldsymbol{\epsilon}$	latency specification vector for the J offered service classes.
$G^v(N^v, E^v)$	the graph representing a CC's VNaaS request v, with m^v virtual links.
N^v	the sets of virtual nodes.
E^v	the sets of virtual links.
m^v	the number of virtual links, $m^v = \|E^v\|$.
T^v	the lifetime of CC's VNaaS request v.
D^v	demand matrix of v, with elements d^v_{ij} representing the demand of a virtual pipe in class j on virtual link e^v_i.
$\boldsymbol{d}^v_i$	the demand vector of all classes of e^v_i.
$\boldsymbol{x}^v$	vector of the effective bandwidth for e^v_i, with $x^v_i = f^v(\boldsymbol{d}^v_i, \boldsymbol{\epsilon})$.
P	the prices for the CSP network with p_{ij} representing the unit price of bandwidth for a link e_i using service class j.
$\underline{\boldsymbol{b}}^v$	vector of minimum acceptable effective bandwidth for virtual links in VNaaS v.
$\overline{\boldsymbol{b}}^v$	vector of the maximum demand for the effective bandwidth for virtual links in VNaaS v.
$U^v(\boldsymbol{x}^v)$	Utility function of the CC when using a VNaaS v.
Υ^v	the VNaaS budget per time interval.
Υ^v_i	the calculated virtual link e^v_i budget, $i \in \{1, \cdots, m_v\}$.
$\boldsymbol{\pi}^v$	vector of calculated price indices for the VNaaS links.
R^v	routing constraints matrix of VNaaS v.
$\hat{\boldsymbol{x}}^v$	new vector of the effective bandwidth for virtual links in VNaaS v satisfying the capacity constraints.
$\hat{\Upsilon}^v_i$	the calculated virtual link e^v_i budget satisfying the capacity constraints, $i \in \{1, \cdots, m_v\}$.
$\hat{\boldsymbol{\pi}}^v$	vector of calculated price indices for the VNaaS links with the capacity constraints.

3.3 VNaaS Demand Calculation

3.3.1 Two Stage Budgeting

As indicated previously, after negotiating the SLA at each time interval, each CC is given new prices P and has to decide on the optimal demand matrix D^v which maximizes the CC's demand decision problem (3.2). The demand matrix D^v defines the amount of demanded bandwidth for each virtual pipe with a given latency threshold on the virtual links. These demands are determined based on the nature of the CC service traffic (e.g., real-time application and video conference versus control traffic), as defined by the VNaaS utility, subject to its pre-allocated budget and routing constraints. Initially, in this chapter, focus is only on the demand allocation subject to the CC's budget and then consideration is given to the effects of the routing demands in the next section.

Formally, the CC's demand decision problem can be expressed through the following optimization problem:

$$\mathbf{P1}: \qquad D^{v*} = \arg \max_{D^v=[d_{ij}^v]} U^v(D^v) \tag{3.2}$$

$$\text{s.t.} \sum_{i=1}^{m^v} \sum_{j=1}^{J} d_{ij}^v \times p_{ij} \leq \Upsilon^v \tag{3.3}$$

Rather than directly solving this problem to find the elements of D^v, we are interested in posing **P1** as a two-stages budget allocation problem, where in the first stage the CC focuses only on allocating an appropriate portion of its original budget Υ^v to each virtual link. Clearly, the first stage of this budgeting procedure can take place only if one can calculate a *price index* for each virtual link. Simply put, the inverse of this price index reflects the utility of a monetary unit spent by the CC to rent resources for that link. In

the second stage, the CC can then distribute the calculated link budget to rent resources in each latency class on that link.

The proposed approach has several advantages in addition to that of reducing the complexity of the original problem formulation. First, it allows the CC to determine the impact of the demand of each virtual link on its total budget and hence project the effects of an increase in a single virtual link demand on the overall VNaaS cost. Secondly, while in current approaches, the CC can only rely on demand measurements for individual services, the proposed budgeting model can aid the CC to accurately identify sources of revenue gains and losses in a straightforward manner. Finally, it allows for a more efficient construction of the internal routing matrix on the rented VN to reduce its overall service cost.

To elucidate the main idea behind the proposed two-stage budgeting mechanisms, assume first that each CC knows a sub-budget Υ_i^v, and that it must use it to only rent resources on a virtual link $e_i^v \in E^v$, then the problem of determining the demand for the virtual pipes hosted on e_i^v can be represented as:

$$\begin{aligned} \textbf{P2:} \quad \boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, \Upsilon_i^v) &= \arg\max f_i^v(\boldsymbol{d}_i^v, \epsilon), \\ &\text{s.t.} \sum_{j=1}^{J} d_{ij}^v \times p_{ij} \leq \Upsilon_i^v \end{aligned} \tag{3.4}$$

This problem can be solved once for each virtual link by first obtaining its Lagrangian:

$$\mathcal{L}_i^v = f_i^v(\boldsymbol{d}_i^v, \boldsymbol{\epsilon}) + \lambda_i^v(\Upsilon_i^v - \sum_{j=1}^{J} d_{ij}^v \times p_{ij}) \tag{3.5}$$

The optimal values $\boldsymbol{d}_i^{v^*}$ then follow from $J-1$ equations, obtained form the Karush-

Kuhn-Tucker (KKT) conditions [156]:

$$\frac{\frac{\partial f_i^v(.)}{\partial d_{ij_1}^{v^*}}}{\frac{\partial f_i^v(.)}{\partial d_{ij_2}^{v^*}}} = \frac{p_{ij_1}}{p_{ij_2}}, \forall j_1, j_2 \in \{1, \cdots, J\} \tag{3.6}$$

which along with the budget constraint (3.4), at equality, can be used to find $\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, \Upsilon_i^v)$.

Now the problem that remains is to determine under what conditions can a budget Υ_i^v be calculated that must be allocated to each $e_i^v \in E^v$ when the CC has an overall budget Υ^v and is given the price matrix P. This question is answered with the following proposition.

Proposition 1. *If the CC utility for a VNaaS v, $U^v(\boldsymbol{x}^v)$, is weakly separable w.r.t. the effective rates for the virtual links, i.e., if it can be expressed as:*

$$U^v(\boldsymbol{x}^v) = \hat{U}^v(f_1^v(.), \cdots, f_m^v(.)) \tag{3.7}$$

then, given virtual links price matrix P, there exists for each virtual link $e_i^v \in E^v$ a price index π_i^v, and a budget Υ_i^v, such that $\pi_i^v \equiv \pi_i^v(\boldsymbol{p}_i)$, is dependent only on the latency class prices for e_i^v and

$$x_i^{v^*} \times \pi_i^v = \Upsilon_i^v, \tag{3.8}$$

where $\sum_{i=1}^{m^v} x_i^{v^*} \times \pi_i^v = \Upsilon^v$, $x_i^{v^*} = f_i^v(\boldsymbol{d}_i^{v*}, \boldsymbol{\epsilon})$, if and only if, for every $e_i^v \in E^v$, $f_i^v(\boldsymbol{d}_i^v, \boldsymbol{\epsilon})$ is a either a homothetic [2] or homogenous function of degree one in $\boldsymbol{d}_i^v$.

Proof:

First, it is shown that $\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, \Upsilon_i^v)$ is homogeneous of degree one in Υ_i^v, i.e., that $\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, \Upsilon_i^v) = \Upsilon_i^v.\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, 1)$. This observation may be obtained directly from examining the first order

[2] A function is homothetic if it is a monotone transformation of a homogeneous function. In turn, a function $f(x)$ is homogenous of degree one in a variable x if $f(tx) = tf(x)$, where $t > 0$ is any scalar.

conditions for the maximization problem **P2**, stated in (3.6) and (3.4) at equality. Clearly, letting $\Upsilon_i^v = 1$ and solving **P2**, the expression in (3.6) is obtained and the budget constraint (3.4) is scaled by Υ_i^v. This implies that $f_i^v(\boldsymbol{d}_i^{v*}(\boldsymbol{p}_i, \Upsilon_i^v), \boldsymbol{\epsilon}) = f_i^v(\Upsilon_i^v, \boldsymbol{d}_i^{v*}(\boldsymbol{p}_i, 1), \boldsymbol{\epsilon})$. To prove the sufficiency of the proposition, it is noted that since the CC utility is weakly separable, it can be expressed as:

$$U^v(\boldsymbol{x}^v) = \hat{U}^v(f_1^v(.), \ldots, f_m^v(.)) \tag{3.9}$$

where $\hat{U}^v$ is the utility expressed as a function of separate functions of each virtual link demands $f_i^{v^*}(\mathbf{d}_i^{v^*}, \boldsymbol{\epsilon})$. Since $f_i^v(\mathbf{d}_i^{v^*}, \boldsymbol{\epsilon})$ is either a homothetic or homogenous function of degree one in $\mathbf{d}_i^{v^*}$, so $x_i^{v^*}$ may be written as:

$$x_i^{v^*} = f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i, \Upsilon_i^v), \boldsymbol{\epsilon}) = \Upsilon_i^v \cdot f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i, 1), \boldsymbol{\epsilon}) \tag{3.10}$$

or conversely, $x_i^{v^*} \frac{1}{f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i,1),\boldsymbol{\epsilon})} = \Upsilon_i^v$. Letting $\pi_i^v = \frac{1}{f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i,1),\boldsymbol{\epsilon})}$, we have $\pi_i^v = \pi_i^v(\mathbf{p}_i)$ and $x_i^{v^*} \times \pi_i^v = \Upsilon_i^v$, and sufficiency is established.

To prove necessity, let

$$x_i^{v^*} \times \pi_i^v = \Upsilon_i^v \tag{3.11}$$

such that π_i^v is dependent only on the price vector $\mathbf{p}_i$. Since, as has been shown earlier in the proof, $\mathbf{d}_i^{v^*}(\mathbf{p}_i, \Upsilon_i^v)$ is homogeneous in Υ_i^v, then it allows $\mathbf{d}_i^{v^*}(\mathbf{p}_i, k\Upsilon_i^v) = k\mathbf{d}_i^{v^*}(\mathbf{p}_i, \Upsilon_i^v)$ for some constant k. In other words, if the CC requires to increase its demand from $\mathbf{d}_i^{v^*}$ to $k\mathbf{d}_i^{v^*}$ while the price $\mathbf{p}_i$ is fixed, it must increase the allocated budget to $k\Upsilon_i^v$. However, from (3.11) since by assumption π_i^v is dependent only on the price vector $\mathbf{p}_i$, then an increase in Υ_i^v by a factor of k implies a direct increase in x_i^v to kx_i^v. Hence, x_i^v must be either a homogenous function of degree one in $\mathbf{d}_i^v$, i.e., $kx_i^v = kf(\mathbf{d}_i^v, \boldsymbol{\epsilon}) = f(k\mathbf{d}_i^v, \boldsymbol{\epsilon})$ or at least, there exists a monotonic transformation of f, such that $f'(kx_i^v) =$

$kf(\mathbf{d}_i^v, \boldsymbol{\epsilon})$, hence, necessity is established.

The above proposition states that if utility $U^v(\boldsymbol{x}^v)$ of the VNaaS v can be further expressed as a function of individual functions that each depends only on demand for a single virtual link, $f_i^v(.)$, then one can calculate a separate price index π_i^v and an allocated budget Υ_i^v for each virtual link iff $f_i^v(\boldsymbol{d}_i^v, \boldsymbol{\epsilon})$ is a homogenous function or can be transformed to a homogenous function.

A direct implication of the homothetic property of $f_i^v(.)$ of degree one in $\boldsymbol{d}_i^v$, is that $x_i^{v^*} = f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i, \Upsilon_i^v), \boldsymbol{\epsilon}) = \Upsilon_i^v \cdot f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i, 1), \boldsymbol{\epsilon})$, and hence, a change in the budget, Υ_i^v, allocated for link,e_i^v, introduces only a scaling of the demand, given a single monetary unit as a budget for that link. Clearly, the necessity and sufficiency of the proposition indicates that this is the only scenario where the CC can calculate a budget and a price index that is independent from the prices posed on other links. However, it is worth noting, however, that defining a separate utility for each virtual link is an efficient means to reflect the varying types of traffic flow that can share the link (e.g., streaming video, exchanging service updates or routing management packets). Before calculating a budget for each virtual link, the price index for each link must be calculated. The inverse of this index reflects the utility of a monetary unit spent by the CC to rent resources for that link. The following lemma provides further information on these price indices.

Lemma 1. *The price index $\pi_i^v(\boldsymbol{p}_i)$, for a virtual link $e_i^v \in E^v$, given the CSP vector of prices for different classes, $\boldsymbol{p}_i$, on that link, satisfies the following relation:*

$$\pi_i^v(\boldsymbol{p}_i) = \frac{1}{f_i^v(\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, 1), \boldsymbol{\epsilon})} \tag{3.12}$$

Proof:

The results of the lemma can be obtained directly from proposition 1, by rewritting (3.8)

as:

$$\pi_i^v = \frac{\Upsilon_i^v}{x_i^{v^*}} = \frac{\Upsilon_i^v}{f_i^v(\mathbf{d}_i^{v^*}(\mathbf{p}_i, \Upsilon_i^v), \boldsymbol{\epsilon})} \tag{3.13}$$

and letting $\Upsilon_i^v = 1$.

The above lemma states that the price index of a unit of bandwidth for a virtual link e_i^v is equal to the inverse of the CC's effective bandwidth when it has a single monetary unit (e.g, a dollar) to spend on its virtual link. One interesting observation from the above lemma is that while the price vector $\boldsymbol{p}_i$ for the J service classes is common for all CCs, the price index, $\pi_i^v(\boldsymbol{p}_i)$ for the same inter-data centers links, varies between CCs. Clearly, this price index reflects the relative worth of the various services offered by the CCs and the amount of profit obtained from a unit of bandwidth from each class. In other words, the price index for each CC is mainly dependent on the shape of the CC's utility gained from servicing its end-users as reflected by $f_i^v(.)$.

The results of the above lemma paves the way for describing the proposed two stage budgeting mechanism for each CC in order to determine its optimal resource demands bundle on its VNaaS :

First stage:

1. The CC first solves m^v problems of the shape **P2**, one for each virtual link $e_i^v \in E^v$ using a unit value for the budget, $\Upsilon_i^v = 1$, to obtain $\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, 1)$.

2. Next, the CC computes the price index π_i^v by substituting $\boldsymbol{d}_i^{v^*}(\boldsymbol{p}_i, 1)$ in Lemma 1 for each virtual link e_i^v.

Second Stage:

1. Using the obtained price indices in the first step, the CC transforms the original

problem **P1** by replacing the utility $U^v(.)$, with the new separable utility $\hat{U}^v$ as defined in Proposition 1. The new problem can now be recast as:

$$\mathbf{P3}: \boldsymbol{x}^{v*}(\Upsilon^v, \boldsymbol{\pi}^v) = \arg\max_{\boldsymbol{x^v}} \hat{U}^v(x_1^v, \cdots, x_{m^v}^v),$$
$$\text{s.t.: } \sum_{i=1}^{m^v} x_i^v \times \pi_i^v \leq \Upsilon^v \tag{3.14}$$

With $m^v - 1$ first order conditions:

$$\frac{\frac{\partial \bar{U}^v}{\partial x_{i_1}^v}}{\frac{\partial \bar{U}^v}{\partial x_{i_2}^v}} = \frac{\pi_{i_1}}{\pi_{i_2}^v}, \forall e_{i_1}^v, e_{i_2}^v \in E^v \tag{3.15}$$

and the overall budget constraint (3.14), at equality to obtain $\boldsymbol{x}^v$.

2. Knowing that $x_i^v \times \pi_i^v = \Upsilon_i^v$, the CC can now obtain the budget Υ_i^v for each individual link e_i^v.

3. Finally, for each of the virtual links e_i^v, Υ_i^v can be substituted in **P2** to obtain the desired demand vector $\boldsymbol{d}_i^{v^*}$ and in turn the demand matrix D^v.

3.3.2 An Illustrative Example:

Consider the VNaaS serving an IPTV provider as depicted in Fig.3.1 and assume that the CSP offers three service classes, namely gold, silver and bronze, with $\boldsymbol{\epsilon} = (1, 0.95, 0.9)$ of the RRT latency and hourly prices $\boldsymbol{p}_i = (8¢, 6¢, 4¢)$ for virtual pipes in the three classes over all links. Let the IPTV provider's net utility for the five virtual links be $U^v(\boldsymbol{x}^v) = (\Pi_{i=1}^5 (x_i^v - \underline{b}_i^v)^{\frac{1}{5}})$, and $\underline{b}_i^v = 0$ for simplicity. All virtual links have an effective rate with a constant elasticity of substitution in classes [156], i.e., $x_i^v = f_i^v(\boldsymbol{d}_i^v, \boldsymbol{\epsilon}) = (\sum_{j=1}^3 \bar{\epsilon}_j (d_{ij}^v)^{\frac{1}{2}})^2$, with $\bar{\epsilon}$ obtained from normalizing $\boldsymbol{\epsilon}$, and the IPTV provider's hourly budget, say (for

simplicity) $\Upsilon^v = 10$ dollars. Following the proposed two stage budgeting, the IPTV provider first solves **P2** for each virtual link. It is easy to verify that the solution is $d^{v*}_{ij}(\boldsymbol{p}_i, 1) = \frac{\bar{\epsilon}^2_j/(p_{ij})^2}{\sum_{j=1}^{3} \bar{\epsilon}^2_j/p_{ij}}$. Then the price indices are obtained for all the virtual links by substituting for $d^{v*}_{ij}(\boldsymbol{p}_i, 1)$ in Lemma 1: $\pi^v_i = (\sum_{j=1}^{3} \bar{\epsilon}_j \frac{\bar{\epsilon}_j/p_{ij}}{(\sum_{j=1}^{3} \bar{\epsilon}^2_j/p_{ij})^{\frac{1}{2}}})^{-2}$. In the second stage, $\boldsymbol{\pi}^v$ is used in **P3** to obtain $x^v_i = \frac{1}{5\pi^v_i} \times \Upsilon^v = \frac{2}{\pi^v_i}$ and to obtain the budget for each link: $\Upsilon^v_i = \pi^v_i \times x^v_i = 2$. Finally, Υ^v_i is substituted in **P2** to obtain d^v_{ij}.

3.3.3 VNaaS Demand with Routing Constraints

As indicated previously, the proposed VNaaS model also provides the CC with the means to impose its internal VN routing policy on the allocated resources to its virtual links while calculating its VNaaS demands. This objective is described by the routing matrix from the flows hosted on the VN. More precisely, the relation $R^v \boldsymbol{x^v} = \boldsymbol{k^v}$ is used to reflect the entire expected distribution of the traffic load on G^v, where $R^v \equiv [r_{ci}]$. For simplicity, it is assumed that R^v is in its reduced form such that $r_{ci} = 0$, $\forall c > i$, $c \in \{1, \cdots, n^v\}$, noting that any matrix resulting from a set of routing constraints can be modified to that form. If the CC's objective is limited to maximizing the net utility functions of all the virtual links in the VNaaS, without considering the constraints imposed by the hosted traffic, this calculated demand may lead to unbalanced allocation and consequently a waste of resources. In other words, the re-allocation technique may increase the allocated bandwidth to some virtual links due to availability of under-utilized resources and hence reduced prices while decreasing the allocated bandwidth to other virtual links due to the unavailability of free resources. This allocation which does not respect the internal routing behavior for the VN may lead to overwhelming some portions of the VNaaS (VMs/links) leading to bottlenecks and a decrease in the hosts cloud applications performance despite the extra bandwidth allocated to some virtual

links.

Fig. 3.1 depicts an example of a VNaaS which is used to by an IPTV service provider. The VN consists of five virtual links and their associated constraints matrix. The first constraints, for example, states that $x_2^v + x_4^v - x_5^v = 0$. It is worth noting here that, these constraints may change over time to reflect the actual traffic on the VNaaS . For the cloud IPTV service example depicted in Fig. 3.1, the constraints between the IPTV service distribution links may change based on the actual traffic of serviced areas which varies according to the end-users time-zones.

To avoid recalculating the VNaaS demand matrix D^v due to a variation in R^v, it might be easier for the CC to calculate its demand without incorporating these constraints, and then update these demands according to the constraints that reflect the current traffic matrix. The following proposition achieves this goal.

Proposition 2. *Let $\boldsymbol{x}^{v^*}(\Upsilon^v, \boldsymbol{\pi}^v)$ be the optimal demand for VNaaS, defined by problem **P3**, given virtual link price indices $\boldsymbol{\pi}^v$ and a budget Υ^v. Furthermore, let $\hat{\boldsymbol{x}}^{v^*}(\Upsilon^v, P, R^v, \boldsymbol{k}^v)$, be the optimal demand for the P3 after adding the capacity constraints $R^v\hat{\boldsymbol{x}}^{v^*} = \boldsymbol{k}^v$, then the two demand vectors are related through the following relation:*

$$\hat{\boldsymbol{x}}^{v^*}(\Upsilon^v, \boldsymbol{\pi}^v, R^v, \boldsymbol{k}^v) = \boldsymbol{x}^{v^*}(\hat{\Upsilon}^v, \hat{\boldsymbol{\pi}}^v) \tag{3.16}$$

where a new budget,$\hat{\Upsilon}^v$, and price indices, $\hat{\boldsymbol{\pi}}^v$ is defined as:

$$\hat{\Upsilon}^v \equiv \hat{\Upsilon}^v(\boldsymbol{\theta}^v) = \Upsilon^v + \boldsymbol{\theta}^{v'}\boldsymbol{k}^v \tag{3.17}$$

$$\hat{\boldsymbol{\pi}}^v \equiv \hat{\pi}^v(\boldsymbol{\theta}^v) = \boldsymbol{\pi}^v + \boldsymbol{\theta}^{v'}R^v \tag{3.18}$$

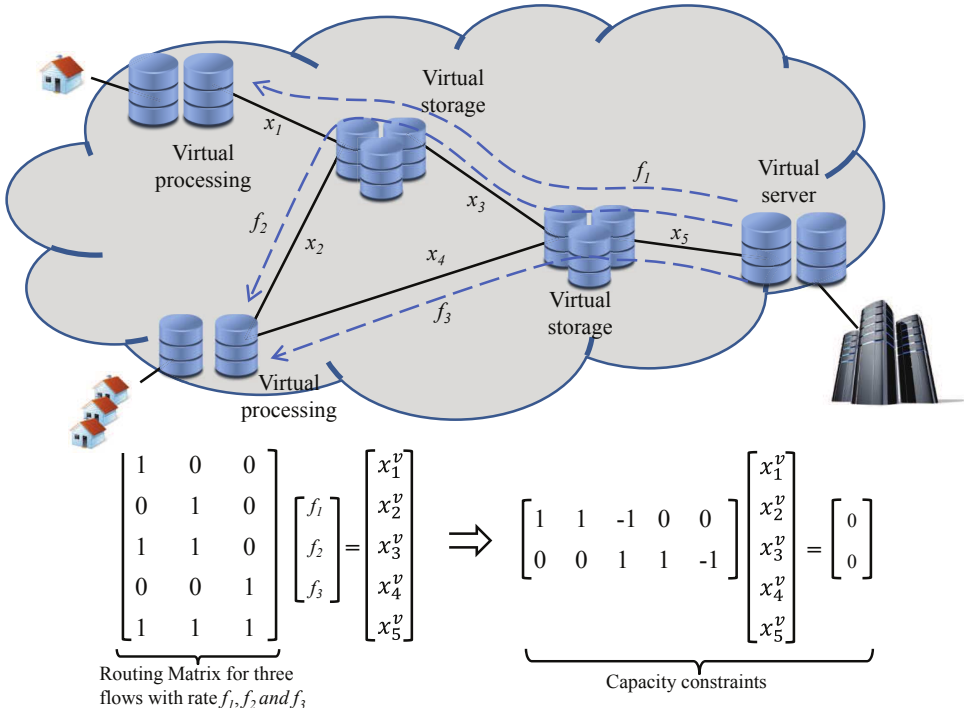

Figure 3.1: An example of a VNaaS hosting an IPTV cloud service

where $'$ *is the transpose operation and the vector* $\boldsymbol{\theta}^v$ *solves:*

$$\boldsymbol{k}^v = R^k.\boldsymbol{x}^{v^*}(\hat{\Upsilon}^v(\boldsymbol{\theta}^v), \hat{\boldsymbol{\pi}}^v(\boldsymbol{\theta}^v)) \tag{3.19}$$

Proof:

First, the proof is started by assuming that two additional routing constraints (3.20) and (3.21) are added to the problem **P2**, then, the solution is generated for the case of more than two additional constraints.

$$\sum_{i=1}^{m^v} x_i^v \times r_{c_1 i} = k_{c_1} \tag{3.20}$$

$$\sum_{i=1}^{m^v} x_i^v \times r_{c_2 i} = k_{c_2} \tag{3.21}$$

To simplify **P2** after adding the additional constraints c_1 and c_2, x_{m^v} and x_{m^v-1} are replaced by solving them as a function of other x_i where $i = 1, \ldots, m-2$, as follows:

$$x_{m^v-1} = \frac{\alpha_{k_{c_1}k_{c_2}}}{\alpha_{m^v-1}} - \sum_{i=1}^{m^v-2} \frac{\alpha_i}{\alpha_{m^v-1}} x_i, \tag{3.22}$$

$$x_{m^v} = \frac{\beta_{k_{c_1}k_{c_2}}}{\beta_{m^v}} - \sum_{i=1}^{m^v-2} \frac{\beta_i}{\beta_{m^v}} x_i \tag{3.23}$$

where,

$$\alpha_i \equiv r_{c_1 i} r_{c_2 m^v} - r_{c_1 m^v} r_{c_2 i} \tag{3.24}$$

$$\beta_i \equiv r_{c_1 m^v-1} r_{c_2 i} - r_{c_1 i} r_{c_2 m^v-1} \tag{3.25}$$

$$\alpha_{k_{c_1}k_{c_2}} \equiv k_{c_1} r_{c_2 m^v} - r_{c_1 m^v} k_{c_2} \tag{3.26}$$

$$\beta_{k_{c_1}k_{c_2}} \equiv r_{c_1 m^v-1} k_{c_2} - k_{c_1} r_{c_2 m^v-1} \tag{3.27}$$

Substituting the expressions (3.22) and (3.23) for x_{m^v-1} and x_m^v, respectively, into the utility function of **P2** and the first budget constraint, will give the following maximization problem without the two variables x_{m^v} and x_{m^v-1}, as follows:

$$\max_{x_1,\ldots,x_{m^v-2}} U^v(x_1, \ldots, x_{m^v-2}, \frac{\alpha_{k_{c_1}k_{c_2}}}{\alpha_{m^v-1}} - \sum_{i=1}^{m^v-2} \frac{\alpha_i}{\alpha_{m^v-1}} x_i, \frac{\beta_{k_{c_1}k_{c_2}}}{\beta_{m^v}} - \sum_{i=1}^{m^v-2} \frac{\beta_i}{\beta_{m^v}} x_i) \tag{3.28}$$

$$\sum_{i=1}^{m^v-2} \gamma_i x_i = y - \Gamma \tag{3.29}$$

where,

$$\gamma_i \equiv \pi_i - \frac{\pi_{m^v-1}\alpha_i}{\alpha_{m^v-1}} - \frac{\pi_m^v \beta_i}{\beta_m^v} \tag{3.30}$$

$$\Gamma \equiv \frac{\pi_{m^v-1}\alpha_{k_{c_1}k_{c_2}}}{\alpha_{m^v-1}} - \frac{\pi_m^v \beta_{k_{c_1}k_{c_2}}}{\beta_m^v} \tag{3.31}$$

Now, consider another maximization function similar to (3.29), but after adding two extra variables (z_1 and z_2) and two extra prices (q_1 and q_2)

$$\begin{aligned} \max_{x_1,\ldots,x_{m^v-2},z_1,z_2} & U^v(x_1,\ldots,x_{m^v-2},z_1 \\ & - \sum_{i=1}^{m^v-2} \frac{\alpha_i}{\alpha_{m^v-1}} x_i, z_2 - \sum_{i=1}^{m-2} \frac{\beta_i}{\beta_{m^v}} x_i) \end{aligned} \tag{3.32}$$

s.t.

$$\sum_{i=1}^{m^v-2} \gamma_i x_i + q_1 z_1 + q_2 z_2 = y - \Gamma \tag{3.33}$$

According to [157], (3.33) can be rewritten in terms of the indirect utility function $h^i(.)$, as follows:

$$x_i = h^i(\tau_1,\ldots,\tau_{m^v-2}, q_1, q_2, y - \Gamma) \tag{3.34}$$

where, $i = 1,\ldots,m^v - 2$

$$z_1 = \sum_{i=1}^{m^v} \frac{\alpha_i}{\alpha_{m^v-1}} h^i(\tau_1,\ldots,\tau_{m^v-2}, q_1, q_2, y - \Gamma) \tag{3.35}$$

$$z_2 = \sum_{i=1}^{m^v} \frac{\beta_i}{\beta_{m^v}} h^i(\tau_1, \ldots, \tau_{m^v-2}, q_1, q_2, y - \Gamma) \tag{3.36}$$

where,

$$\tau_i \equiv \pi_i + \frac{\alpha_i}{\alpha_{m^v-1}}(q_1 - \pi_{m^v-1}) + \frac{\beta_i}{\beta_m^v}(q_2 - \pi_m^v) \tag{3.37}$$

and,

$$\Gamma \equiv \frac{\pi_{m^v-1}\alpha_{k_{c_1}k_{c_2}}}{\alpha_{m^v-1}} + \frac{\pi_{m^v}\beta_{k_{c_1}k_{c_2}}}{\beta_{m^v}} \tag{3.38}$$

Also according to [157], (3.34) can be written as follows:

$$x_i = h^i(\tau_1, \ldots, \tau_{m^v-2}, q_1, q_2, y - \Gamma + q_1\frac{\alpha_{k_{c_1}k_{c_2}}}{\alpha_{m^v-1}} + q_2\frac{\beta_{k_{c_1}k_{c_2}}}{\beta_{m^v}}) \tag{3.39}$$

where $i = 1, \ldots, m^v$. For simplification, θ_1 and θ_2 are defined as follows:

$$\theta_1 \equiv \frac{q_1 - \pi_{m^v-1}}{\alpha_{m^v-1}} \tag{3.40}$$

$$\theta_2 \equiv \frac{q_2 - \pi_{m^v}}{\beta_{m^v}} \tag{3.41}$$

Therefore, $\alpha_{k_{c_1}k_{c_2}}$ and $\beta_{k_{c_1}k_{c_2}}$ can be solved as:

$$\alpha_{k_{c_1}k_{c_2}} = \sum_{i=1}^{m^v} \alpha_i h^i(\pi_1 + \theta_1\alpha_1 + \theta_2\beta_1, \ldots, \tag{3.42}$$
$$\pi_m^v + \theta_1\alpha_m^v + \theta_2\beta_m^v, \Upsilon^v + \theta_1\alpha_{k_{c_1}k_{c_2}} + \theta_2\beta_{k_{c_1}k_{c_2}})$$

$$\beta_{k_{c_1}k_{c_2}} = \sum_{i=1}^{m^v} \beta_i h^i(\pi_1 + \theta_1\alpha_1 + \theta_2\beta_1, \ldots, \pi_m^v + \theta_1\alpha_m^v + \theta_2\beta_m^v, \Upsilon^v + \theta_1\alpha_{k_{c_1}k_{c_2}} + \theta_2\beta_{k_{c_1}k_{c_2}}) \tag{3.43}$$

And the solution to **P2** after adding the two additional constraints can be expressed as follows:

$$x_i = h^i(\pi_1 + \theta_1\alpha_1 + \theta_2\beta_1, \ldots, \pi_m^v + \theta_1\alpha_m^v + \theta_2\beta_m^v, \Upsilon^v + \theta_1\alpha_{k_{c_1}k_{c_2}} + \theta_2\beta_{k_{c_1}k_{c_2}}) \tag{3.44}$$

The optimal demand $\bar{\boldsymbol{x}}^{v^*}$ for the optimization problem **P2** after adding the two constraints is related to the optimal demand $\boldsymbol{x}^{v^*}$ before adding them as follows:

$$\bar{\boldsymbol{x}}^{v^*}(\Upsilon^v, \boldsymbol{\pi}^v, R^v, \boldsymbol{k}^v) = \boldsymbol{x}^{v^*}(\hat{\Upsilon}^v, \hat{\boldsymbol{\pi}}^v) \tag{3.45}$$

where,

$$\hat{\Upsilon}^v = \Upsilon^v + \theta_1\alpha_{k_{c_1}k_{c_2}} + \theta_2\beta_{k_{c_1}k_{c_2}} \tag{3.46}$$

$$\begin{aligned} \hat{\pi}_1^v &= \pi_1 + \theta_1\alpha_1 + \theta_2\beta_1 \\ \hat{\pi}_1^v &= \pi_1 + \theta_1\alpha_1 + \theta_2\beta_1 \end{aligned} \tag{3.47}$$

Generalizing the solution in case of more than two additional constraints can be extended from (3.45).

The above proposition shows how the CC can incorporate its service demand con-

straints as they change to its resource demands from the CSP as follows,

1. The first step to achieve this goal is to use the results in the previous section to obtain a closed form solution $\boldsymbol{x}^v(\Upsilon^v, \boldsymbol{\pi}^v)$ for **P1** where $\boldsymbol{x}^v$ can be represented as a function of the budget and price vectors (e.g., in the previous example we had $x_i^v = \frac{1}{5\pi_i^v} \times \Upsilon^v$).

2. Then, we obtain $\boldsymbol{x}^{v^*}$ as a function of $\boldsymbol{\theta}^v$ by substituting the original budget and price index vectors with (3.17) and (3.18), respectively.

3. Now, the value of the auxiliary vector $\boldsymbol{\theta}^v$ can be calculated by solving (3.19) for $\boldsymbol{\theta}^v$. It is worth noting that the size of the vector $\boldsymbol{\theta}^v$ is equal to the number of routing constraints.

4. Next, the resulting vector is used to obtain the numerical values for the constrained problem's resulting new CC's budget and price indices using (3.17) and (3.18), respectively.

5. These two vectors are then used to obtain the modified demand vector using (3.16).

Applying the above proposition to the example in Fig. 3.1, given the solution $\boldsymbol{x}^v$obtained from the unconstrained problem, the new effective rate $\hat{x}_i^v$ can be calculated after adding the new constraint: $\hat{x}_3^v + \hat{x}_4^v - \hat{x}_5^v = 0$. First, θ that solves the relation is obtained:

$$0 = \Upsilon^v \times \left(\frac{1}{5\pi_1^v} + \frac{1}{5\pi_2^v} + \frac{1}{5(\pi_3^v + \theta)} + \frac{1}{5(\pi_4^v + \theta)} + \frac{1}{5(\pi_4^v - \theta)}\right) \tag{3.48}$$

It is then easy to show that, given that the CC's demand without the routing constraints is $x_i^v = \frac{1}{5\pi_i^v}\Upsilon^v$, the new demand $\hat{x}_i^v = x_i^v$, for $i = 1, 2$, and $\hat{x}_3^v = \frac{1}{5}\frac{\Upsilon^v}{\pi_3^v + \theta}$, $\hat{x}_4^v = \frac{1}{5}\frac{\Upsilon^v}{\pi_4^v + \theta}$ and $\hat{x}_5^v = \frac{1}{5}\frac{\Upsilon^v}{\pi_3^v - \theta}$.

3.4 Cloud Service Provider (CSP) Resource Pricing

This section illustrates how to calculate the optimal values for the CSP VNaaS prices to regulate the demand of the CCs at a given time interval. The CSP service pricing is approached as a regulated monopoly problem where it chooses the prices that can maximize the social welfare of the system given a profitability constraint set by a third party (e.g., laws or regulations). This model differs from traditional approaches (e.g., see previous work by the authors in [111]) which aim at setting the prices at the marginal cost of the services. Also, in contrast to the existing pricing models which mostly aim at regulating the CCs demands and congestion, the main goal of this model is to provide an incentive for the CSP to ration the offering of a variety of service classes. It also implies that if the demand for one service class is affected by the prices for a smaller range, then the price of these services should include a larger share of the overall service costs (i.e., prices are inversely proportional to the demand sensitivity in each service class). In other words, prices for each class should be adjusted according to the willingness of the clients to pay for bandwidth in that service class. In this manner, as congestion increases, the shares of the bandwidth cost allocation, and in turn the prices, increase at a higher rate for those classes where clients have inelastic demands and/or are willing to pay more for these low latency classes.

To obtain the price for each inter-data centers link $e_i \in E$, let $y_{ij} = \sum_{v=1}^{V} d_{ij}^{v}$, be the demand of class j of all CCs services on the CSP's link e_i, and the vector $\boldsymbol{y_i} = (y_{i1}, \cdots, y_{iJ})$ be the demand vector for all service classes on e_i. With a slight abuse of notation, in this section, the ratio of the link bandwidth headroom allocated to each latency class j to guarantee differentiated latency is denoted by ϵ_j. This is justified by the fact that the latency of each class is directly proportional to the amount of ad-

ditional bandwidth allocated to that class using a virtual queue [17, 134]. In this case, the CSP will only accept new VNaaS requests as long as servicing a new request will not violate the conditions $\sum_{v=1}^{V}\sum_{j=1}^{J}\underline{b}_{ij}\epsilon_j \leq c_i$, and that the CC budget is sufficient to purchase $\underline{b}_{ij}$, where V is the number of VNaaS currently hosted by the CSP. The optimal prices for the VNaaS services are the ones that maximize the welfare of both the CSP and the CCs; clearly, the main objective of the CSP is to maximize the revenue obtained while minimizing its service costs. On the other hand, the CCs welfare is proportional to the area under the demand curve, i.e., $\int_{\boldsymbol{p}_i}^{\infty}\boldsymbol{y}_i d\boldsymbol{p}_i$.

The adopted Ramsey prices [68] aim at preventing the CSP monopoly of prices by maximizing the welfare of both the CCs and the CSP subject to a preset profit threshold, Γ, on the net profit. This threshold can be determined according to market competition or regulations. Adding the link capacity constraint, the CSP's problem can be formulated as follows,

$$\textbf{P4:} \quad \max_{\boldsymbol{p}_i} \quad \int_{\boldsymbol{p}_i}^{\infty}\boldsymbol{y}_i(\boldsymbol{p}_i)d\boldsymbol{p}_i + \boldsymbol{y}_i\boldsymbol{p}_i - C(\boldsymbol{y}_i), \tag{3.49}$$

$$\text{s.t. } \boldsymbol{y}_i\boldsymbol{p}_i - C(\boldsymbol{y}_i) \leq \Gamma \tag{3.50}$$

$$\sum_{j=1}^{J} y_{ij}\epsilon_j \leq c_i \tag{3.51}$$

where $C(\boldsymbol{y}_i)$ is the cost incurred by the CSP to own or rent the network links connecting the data centers. This cost is comprised of a fixed cost and variable costs. The former may for example include the cost of renting the bandwidth from an infrastructure provider or of purchasing and maintaining its own network. The latter variable costs depend on the service demands $\boldsymbol{y}_i$ and differ per class. They include, for example, the extra cost of service in each class due to reserving additional bandwidth in order to reduce the

classes latency. Before solving the above problem, the parameters $\varepsilon_{jk} = \frac{\partial y_{ij}}{y_{ij}} / \frac{\partial p_{ik}}{p_{ik}}$ must be defined to represent the cross elasticity of demand between classes. In other words, ε_{jk} measures the change of demand for bandwidth in class j with latency ϵ_j to the change in the price of bandwidth in class k. Clearly, $\varepsilon_{jj} < 0$, i.e., an increase in the price of a service class leads to a decrease in the demand. Similarly, $\varepsilon_{jk} > 0$, since an increase in the price of any class moves a portion of its demand to a different class. These parameters can easily be estimated by the CSP through historical data analysis [158]. In defining the demand elasticity matrix $\mathcal{E} = [\varepsilon_{jk}]$, the following proposition provides a closed form solution to the above problem **P4** of the CSP.

Proposition 3. *The optimal CSP data center network prices* $\boldsymbol{p}_i = (p_{i1}, \cdots, p_{iJ})$ *with* J *guaranteed latency classes for an inter-data centers link* e_i *satisfy:*

$$\hat{p}_{ij} = \frac{p_{ij} - \frac{\partial C_i(\boldsymbol{y}_i)}{\partial y_{ij}} + \lambda_i \epsilon_j}{p_{ij}} \tag{3.52}$$

where $\hat{\boldsymbol{p}}_i = (\hat{p}_{i1}, \cdots, \hat{p}_{iJ})$ *is obtained from:*

$$\hat{\boldsymbol{p}}_i = \lambda \mathcal{E}^{-1}.\mathbf{1} \tag{3.53}$$

such that λ *is a scalar that is proportional to the CSP profit threshold* Γ*, and* λ_i *is the shadow price for the increased demand on the link* e_i*, such that* $\lambda_i = 0$*, when the net demand for* e_i*'s bandwidth can be satisfied with* $\sum_{j=1}^{J} y_{ij}\epsilon_j < c_i$*.*

Proof:

First, obtaining the Lagrangian of **P4**:

$$\mathcal{L} = \sum_{k=1}^{J} \int_{0}^{\infty} y_{ij} dp_{ij} - \int_{0}^{p_{ij}} y_{ij} dp_{ij} + y_{ik} p_{ik} - C(\boldsymbol{y}_i) + \lambda_\Gamma (\sum_{k=1}^{J} y_{ik} p_{ik} - C(\boldsymbol{y}_i) - \Gamma) + \lambda_i' (\sum_{k=1}^{J} y_{ik} \epsilon_k - c_i) \tag{3.54}$$

where λ_Γ and λ_i' are the Lagrangian multipliers. λ_Γ represents the marginal increase in welfare when relaxing the profit threshold constraint. The optimal values for $\hat{\boldsymbol{p}}_i$, λ_Γ and λ_i' that maximize (3.49) follow from the Karush-Kuhn-Tucker (KKT) conditions [156]:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_{ij}} &= -y_{ij} + (1+\lambda_\Gamma) \left[y_{ij} + \sum_{k=1}^{J} (p_{ik} \frac{\partial y_{ik}}{\partial p_{ij}} - \frac{\partial C(\boldsymbol{y}_i)}{\partial y_{ik}} . \frac{\partial y_{ik}}{\partial p_{ij}}) \right] \\ &\quad + \lambda_i' \sum_{k=1}^{J} \epsilon_k \frac{\partial y_{ik}}{\partial p_{ij}} = 0 \\ &= \lambda_\Gamma y_{ij} + (1+\lambda_\Gamma) \sum_{k=1}^{J} (p_{ik} - \frac{\partial C(\boldsymbol{y}_i)}{\partial y_{ik}} + \frac{\lambda_i' \epsilon_k}{(1+\lambda_\Gamma)}) \frac{\partial y_{ik}}{\partial p_{ij}} \end{aligned} \tag{3.55}$$

According to Roys identity [159], $\frac{\partial y_{ik}}{\partial p_{ij}}$ can be replaced by $\frac{\partial y_{ij}}{\partial p_{ik}}$, we get:

$$\frac{-\lambda_\Gamma y_{ij}}{(1+\lambda_\Gamma)} = \sum_{k=1}^{J} (p_{kj} - \frac{\partial C(\boldsymbol{y}_i)}{\partial y_{ik}} + \frac{\lambda_i' \epsilon_k}{(1+\lambda_\Gamma)}) \frac{\partial y_{ij}}{\partial p_{ik}} \tag{3.56}$$

By substituting the cross elasticity of demand between classes $\varepsilon_{jk} = \frac{\partial y_{ij}}{y_{ij}} / \frac{\partial p_{ik}}{p_{ik}}$ into (3.55), it can be written as:

$$\frac{-\lambda_\Gamma}{1+\lambda_\Gamma} = \sum_{k=1}^{J} \varepsilon_{jk} \frac{(p_{ik} - \frac{\partial C(\boldsymbol{y}_i)}{\partial y_{ik}} + \frac{\lambda_i' \epsilon_k}{(1+\lambda_\Gamma)})}{p_{ik}} \tag{3.57}$$

Define $\hat{p}_{ik}$ as:

$$\hat{p}_{ik} = \frac{p_{ik} - \frac{\partial C(\boldsymbol{y}_i)}{\partial y_{ik}} + \lambda_i \epsilon_k}{p_{ik}} \tag{3.58}$$

Where, $\lambda_i = \frac{\lambda'_i}{(1+\lambda_\Gamma)}$, $\forall i$. Putting (3.57) into the vector form, we obtain:

$$\frac{-\lambda_\Gamma}{1+\lambda_\Gamma}\mathbf{1} = \mathcal{E}.\hat{\boldsymbol{p}}_i \tag{3.59}$$

$$\hat{\boldsymbol{p}}_i = \lambda\mathcal{E}^{-1}.\mathbf{1} \tag{3.60}$$

where, $\lambda = \frac{-\lambda_\Gamma}{1+\lambda_\Gamma}$.

In the above proposition, the scalar λ, is the Lagrangian multiplier of the CSP's profit constraint and depends only on the preset profit Γ. It measures the net gain of the CCs as Γ decreases by a monetary unit. On the other hand, λ_i is the shadow price for each link e_i that becomes positive, hence, adding to the price of each class, as the demand for the resources of the link exceeds its capacity.

It is also worth noting that the results of the above proposition indicate that in order the calculate the inter-data center link prices for each class, the CSP needs only to know its expected demand vector y_i, which can be obtained by monitoring the cloud resources [123], and the associated cost $C(\boldsymbol{y}_i)$.

3.5 Re-optimization Monitoring Scheme

In this section, a description is given of the proposed CSP's service monitoring scheme that facilitates the implementation of the proposed service demand and pricing mechanisms. Since the CSP's serviced VNaaS request are long-lived services with varying demands, a semi-centralized monitoring mechanism was implemented that can reallocate the reserved rates via adjusting the corresponding virtual queues and rate pacers [144]

as described before. This semi-centralized monitoring mechanism is appropriate for the inter-data centers network monitoring, since in practice, the number of geographically distributed data centers is relatively small (e.g., Google currently has twelve geographically distributed data centers around the world). Also, semi-centralized monitoring approach attempts to combine the strengths of both; the optimal allocation and pricing obtained via centralized monitoring and the scalability and the low overhead of distributed monitoring. Accordingly, the allocated bandwidth at time interval $t-1$, $x_e^v(t-1)$ is monitored and compared against the current need $x_e^v(t)$ and the necessary rate adjustments are performed to ensure a high utilization of the CSP's resources while meeting the VNaaS demands.

Although centralized monitoring approaches work well for small sized inter-data centers networks, they may scale poorly for large sized inter-data centers networks. To address this problem, a two-level monitoring scheme is proposed where the first level is performed by the CSP or the infrastructure provider by periodically monitoring the physical infrastructure. The output from level-one monitoring is the current net CCs demand y_{ij} of each class j on link e_i. The main objective of having this monitoring module is to periodically reallocate the CSP resources among the VNaaS, locally at each data center in order to achieve the desired performance [54,55]. Moreover, a monitoring agent (i.e., a software module) runs on every data center to collect data about the demand y_{ij} on each inter-data center link e_i. Then each agent periodically reports the collected data to a centralized CSP management entity which manages the entire CSP's network and calculates the overall pricing as previously described, and as depicted by Fig. 3.2.

The second monitoring level provides the added service of calculating the CC's current demand by monitoring their applications' varying traffic on their behalf. In this

case, the CC provides only high-level cloud service-specific performance requirements to the CSP . In turn, the latter translates them to low-level, resource-specific demands by specifying the corresponding service quality parameters (e.g., the CC may require a specific threshold for the service response time and the CSP translates the required response time to a set of virtual pipes with bandwidth and latency requirements). This process of translating from high-level performance requirements to specific service parameters may result in inaccurate resource provisioning due the dynamic nature of the cloud services and/or the difficulty to rigidly define the relation between high and low-level requirements. Consequently, the CSP monitors the actual performance and compares it with the profile performance through the monitor interface and according to the degree of deviation from the profile performance. The CSP may request an increase or decrease in the amount of allocated resources to achieve the required high-level, service-specific performance. In addition, this monitoring module helps the CSP to avoid unnecessary SLA violations triggers during a period where the CC's demand is reduced. Fig. 3.2 provides a schematic description of the overall monitoring process.

The demand request adjustment process is triggered at each time interval. The length of this time interval depends on the nature of the cloud services offered by the VNaaS (e.g., IPTV service which depends on the amount of traffic coming from different regions according to the time of the day).

3.6 Performance Evaluation

3.6.1 Simulation Environment

In order to evaluate the performance of the proposed dynamic VNaaS demand and pricing techniques, the simulator in [27] is extended to simulate a portion of Yahoo! network which connects five of its data centers (i.e., data centers located at Dallas,

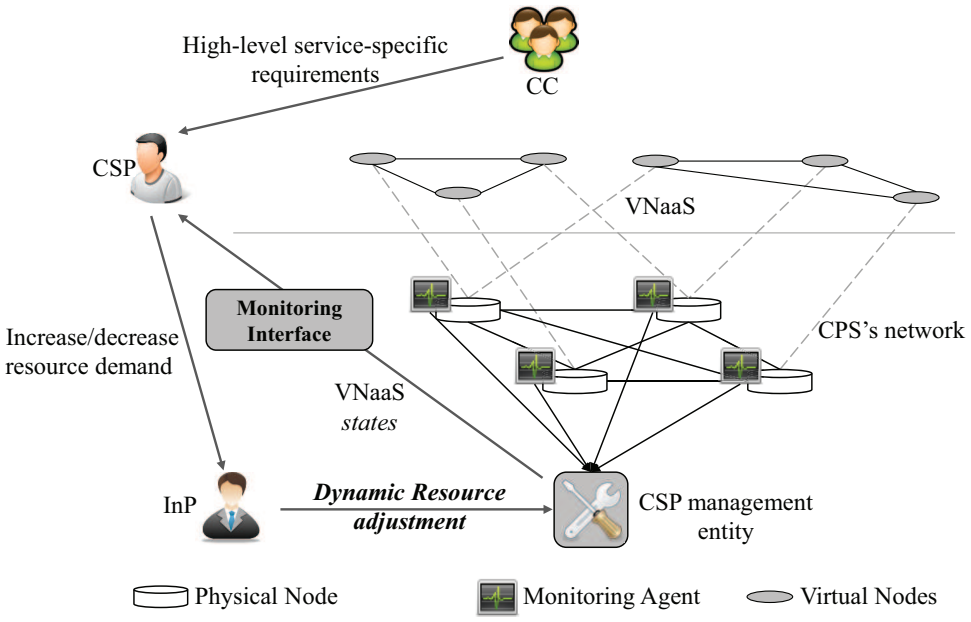

Figure 3.2: The proposed two-level monitoring mechanism

Palo Alto, Washington DC, United Kingdom, and Hong Kong) as shown by Fig. 3.3. Yahoo! organizes their data centers in a hierarchical way. At each of the data centers, Yahoo's border routers connect to several other ISPs to reach its clients and other data centers. These data centers are also directly connected to each other through a private network service (e.g. leased lines), and hence may carry traffic for each other through this private network. The inter-data centers traffic is classified into two categories, namely, client-triggered traffic and background traffic. The former is triggered by the front-end customer-facing services such as web search, email, online chat, online-gaming, and video streaming. The latter is due to internal tasks such as routine background computation (e.g., search indexing), periodic data back-up, and data synchronization [4].

The inter-data centers traffic of the CSP's network links is generated according to the traffic traces available through the anonymized Yahoo! datasets [160]. Furthermore, the port numbers inferred in [4] are used to distinguish the flows marking the inter-data centers traffic. The inter-data centers traffic traces includes both the inbound and outbound

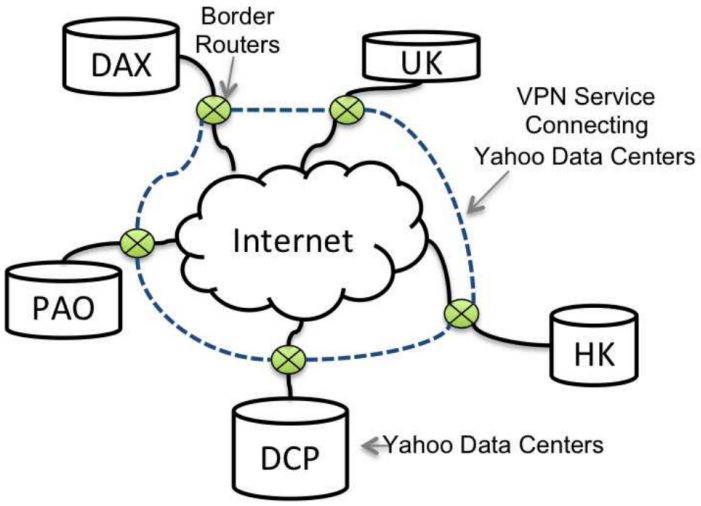

Figure 3.3: The inter-data centers network of five Yahoo! data centers [4].

traffic. Each record in the traces contains a sampled flow information, which includes the following fields: a)timestamp, b)source and destination IP addresses and transport layer port numbers, c)source and destination interface on the router, d)IP protocol, e)number of bytes and packets exchanged [4]. The CSP offers three service classes (Gold, silver, bronze) in addition to a best effort service with a flat price rate. The proposed dynamic VNaaS model focuses on enhancing the overall data centers network utilization and the degree of satisfaction of the CCs that own the VNaaS requests, in terms of bandwidth resources. Hence, the satisfaction of the VMs' demands is not considered in the evaluation and assumed that they have been placed according to the CCs location constraints. All CCs' utilities are set to that function described in the example of Section 3.3.2.

3.6.2 Evaluation Results

CSP profit and resources utilization

In order to evaluate the proposed latency differentiation technique and to show its sig-

nificant impact on increasing the CSP profit, a comparison is made between the CSP net profit after serving 100 VNaaS requests, in the case of applying the proposed technique (i.e., Ramsey-based pricing) versus treating all the VNaaS requests equally (i.e., static pricing without latency differentiation). Additionally, a comparison is made between the results when service differentiation is employed by assigning different weights for each service class [33], and the case where the price is dynamically changing according to the supply and demand model without service differentiation [161]. The CSP net profit is calculated as the difference between the revenue obtained from hosting the VNaaS at the required latency class and the cost of the resources incurred by hosting these VNaaS.

Fig. 3.4 plots the average normalized CSP net profit at each latency class. In other words, Fig. 3.4 shows the scaled average CSP net profit, so that the minimum and maximum values of the net profit are 0 and 1, respectively. Since the cost of allocating a unit bandwidth at latency class j' is higher than that of allocating the same amount at latency class j according to the Ramsey pricing model where the latency class j' has a higher priority than j, the average profit from the highest latency class is much more than the Best Effort one. As indicated by Fig. 3.4, the CSP net profits are almost equivalent for all latency classes when applying the static pricing with no latency differentiation. In the case of the supply and demand pricing model, the CSP net profit depends only on the demands at each latency class. In the case of applying the weighted pricing, the highest latency class contributes a higher CSP profit. However, the scheme's insensitivity to the CCs willingness to pay reduces the CSP net profit. It is clear that the total CSP profit (i.e., the sum of the normalized profit from all latency classes for all 100 VNaaS requests in terms of monetary units) in the case of having latency differentiation (≈ 2.26) is higher than the total CSP profit in the other cases (weighted pricing ≈ 1.98, supply and demand

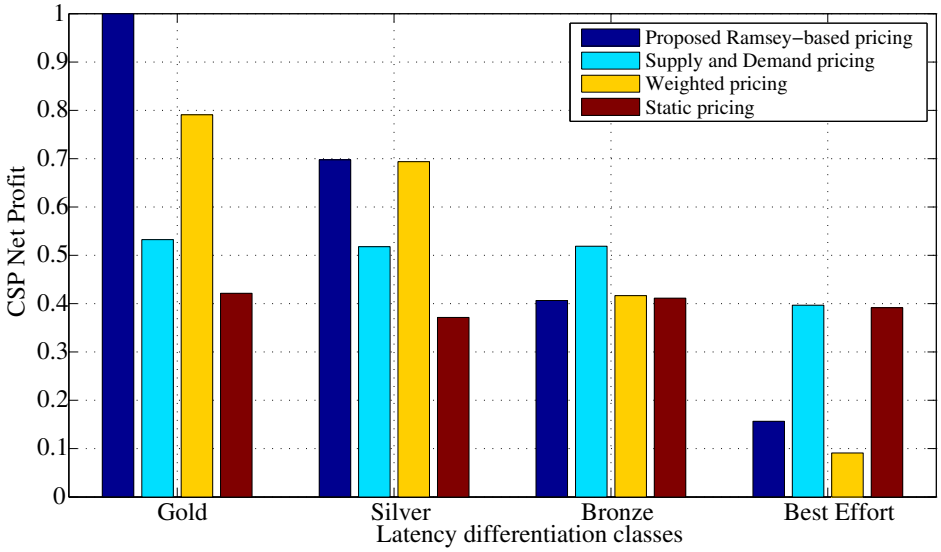

Figure 3.4: CSP revenue for each of the latency classes.

pricing ≈ 1.94, and static pricing ≈ 1.6). Additionally, Fig. 3.5 demonstrates the CSP accumulated profit with time. It is clear that the proposed pricing model outperforms the three other pricing techniques.

To demonstrate how the proposed approach enhances the resources utilization, the average CSP resource utilization is measured when the number of simultaneously hosted VNaaS varies between 1-25 VNaaS . Fig. 3.6 compares the average CSP inter-data centers network utilization, as the number of VNaaS requests increases, when using the proposed mechanism versus that obtained with the use of the allocation mechanisms using the three other pricing models. It is clear that the proposed technique can accurately allocate the actual CCs dynamic cloud networking needs while preventing resource over-provisioning and minimizing the distributed data centers network fragmentation problem. This in turn increases the acceptance ratio of future VNaaS requests. Additionally, using the proposed pricing mechanism while regulating the demand of the CCs provides higher CSP resource utilization, since it provides the optimal prices for the VNaaS services

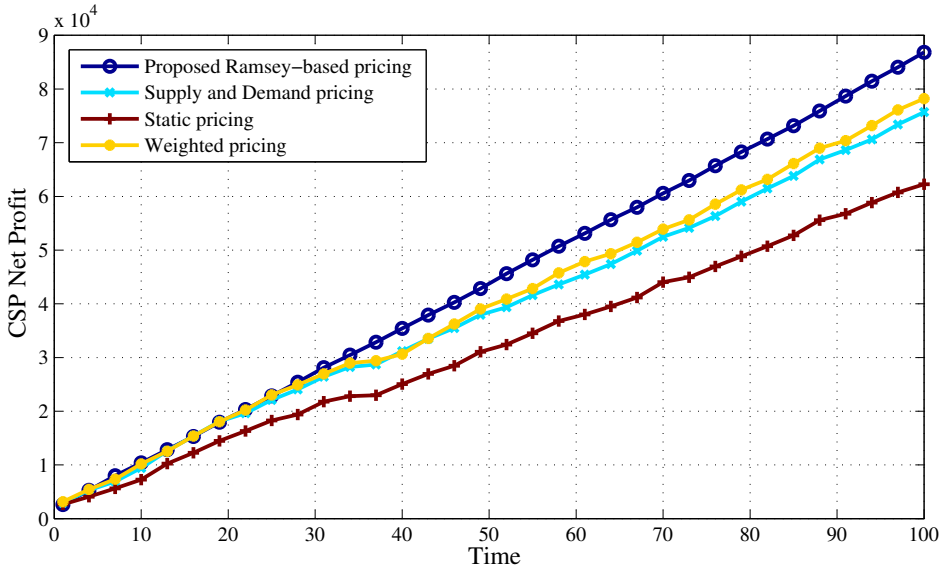

Figure 3.5: CSP's accumulated revenue.

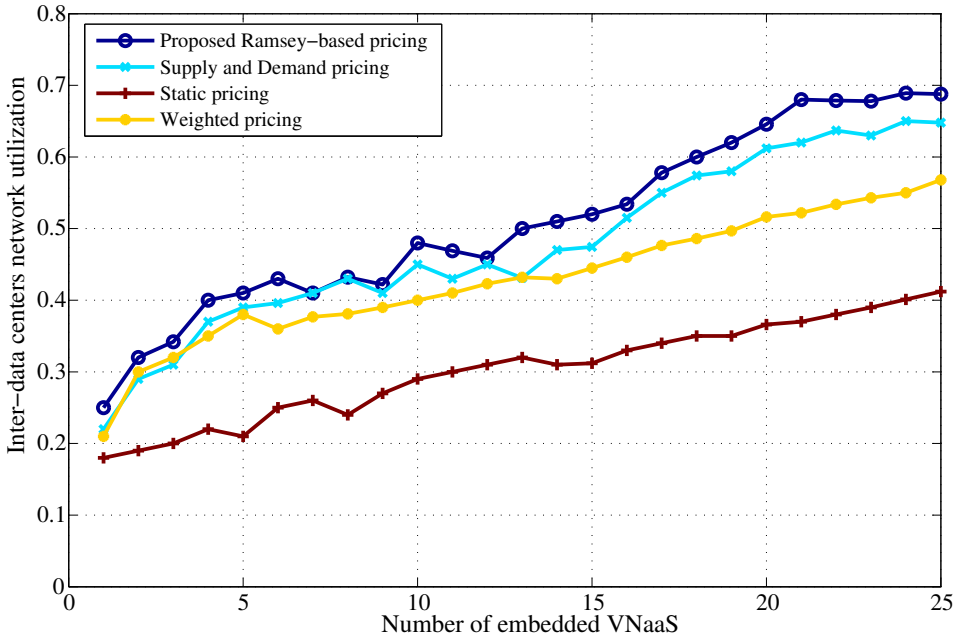

Figure 3.6: CSP's inter-data centers network utilization.

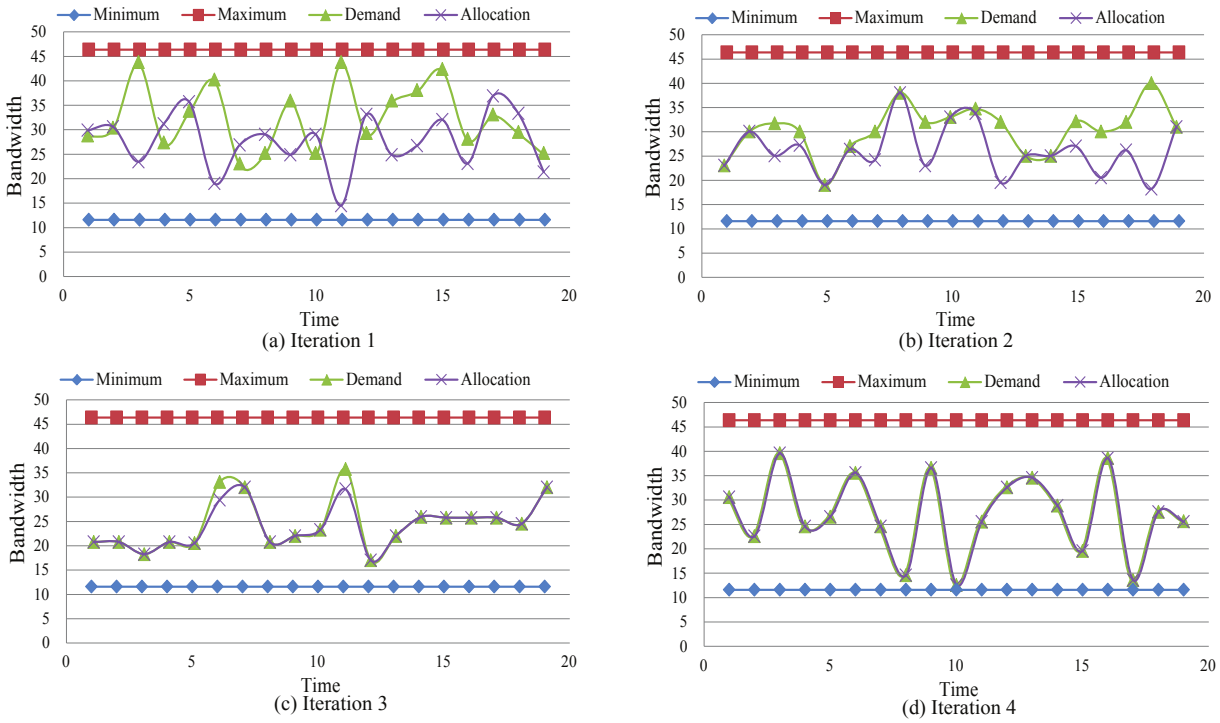

Figure 3.7: The deviation between the allocated and demanded bandwidth during the transient phase of price adjustment process among four iterations.

while encouraging the CCs to request their networking needs. This maximizes both the welfare of both the CSP and the CCs . Also, it can be seen that the allocation mechanisms using the static and the weighted pricing models, without benefiting from demand regulation, may result in over-provisioning the actual CCs dynamic demands. This can lead to inefficient inter-data centers resources utilization. Also, the performance of the proposed approach outperforms that of the supply and demand based pricing since the latter regulates the CC's demands without a resource elasticity offerings.

The CCs calculate their demand matrix D^v based on the prices posted by the CSP and their budgets, However, CSP adjusts the prices in a way that maximizes its welfare and that of the CCs based on CCs demands. Therefore, there exists a transient phase before the prices converge to the optimal values that maximize the welfare. Fig.

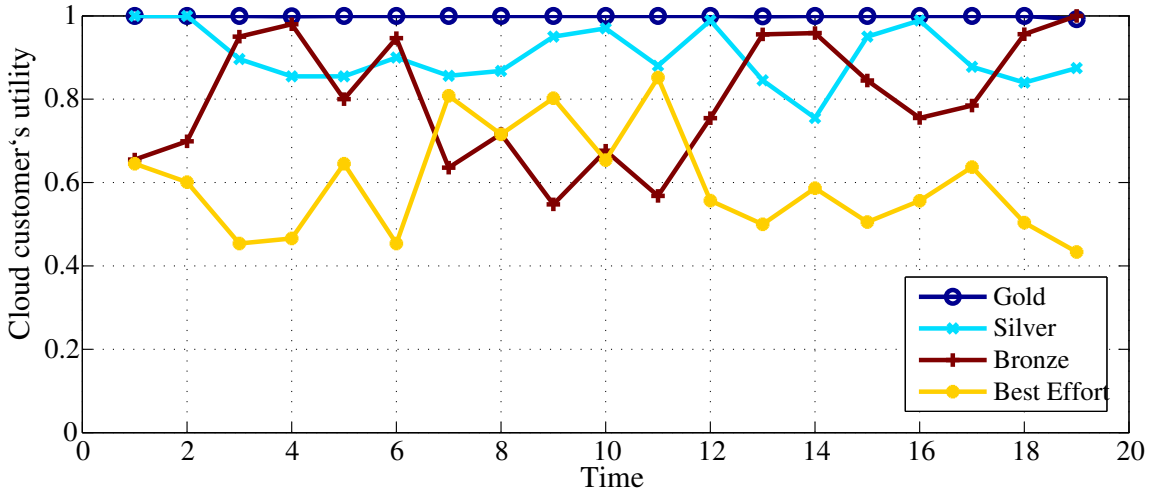

Figure 3.8: The CC utility versus time for each latency class.

3.7 shows how the allocated bandwidth deviates from the demanded amount due to the price adjustment process for a specific VNaaS request. Initially, the published prices listed by the CSP do not reflect the actual market needs and fail to satisfy the demanded resources as shown by Fig. 3.7(a). This graph represents the resource allocation during the first iteration of the price adjustment process done by the CSP . Fig. 3.7(d) shows that there is almost no difference between the demanded and allocated bandwidth when the prices are adjusted to the optimal values that maximize the welfare of both the CSP and the CCs.

Cloud Clients' Utility

The effect of latency classes differentiation is clearly reflected on the CC's utility function output according to the allocated bandwidth for each latency class. Fig. 3.8 plots one CC's utility value over time as derived from the bandwidth allocated from each latency class. This in turn may for example, reflect the utility of different CC appli-

cations that are consuming resources in these classes. It is apparent that high latency classes are prioritized over low latency classes, resulting in high utility values for them (i.e., utility function output of the Gold latency class is almost equal to 1 over time, while utility function output of the best effort latency class varies according to the availability of the resources). Moreover, Fig. 3.9 plots the average utility function output at four latency classes. It is clear that the CC's utility values when using this research's proposed latency classes differentiation mechanism, outperforms the other techniques, since this allocation model prioritizes the higher latency classes while taking into consideration their regulated demands according to the published prices. In the case of the weighted pricing allocation, the higher classes are prioritized without regulating the CCs demands. This results in a less efficient allocation and hence a lower CCs utility. The supply and demand utility is based on the current available and requested resources which does not accurately reflect the current CC needs. Also using the the static allocation, the average utility function output is close to being equal since no differentiation exists between latency classes.

Fig. 3.10 plots the optimal demand for each virtual pipe given the CSP prices and the pre-allocated CC's budget. The utilities are set to that described in the example of Section 3.3.2. When the CC is constrained by a small budget, it is better to request more bandwidth at lower latency classes which are characterized by lower prices.

3.7 VNaaS Limitations

Assumptions and limitations are categorized into two categories; namely, assumptions for simplicity which can be easily relaxed, and Scheme limitations.

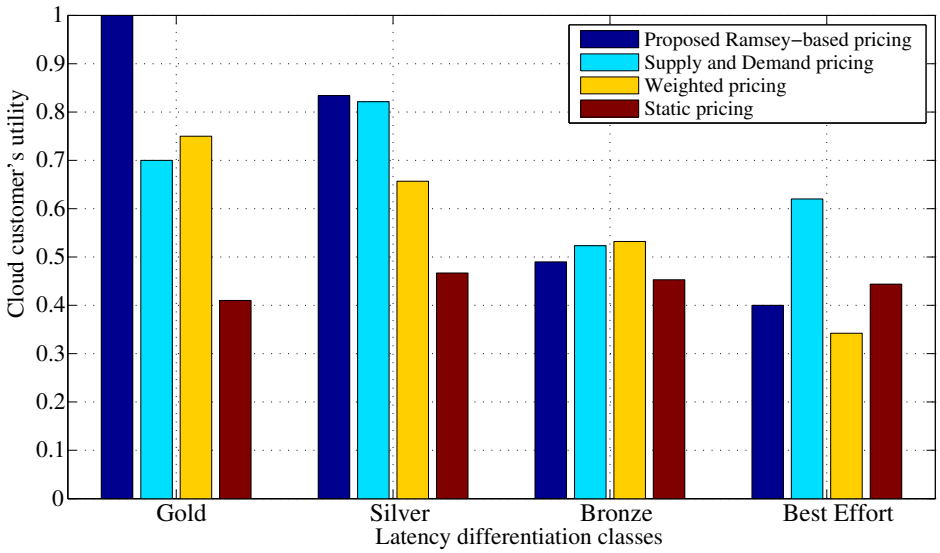

Figure 3.9: VNaaS users utility function for each of the latency classes.

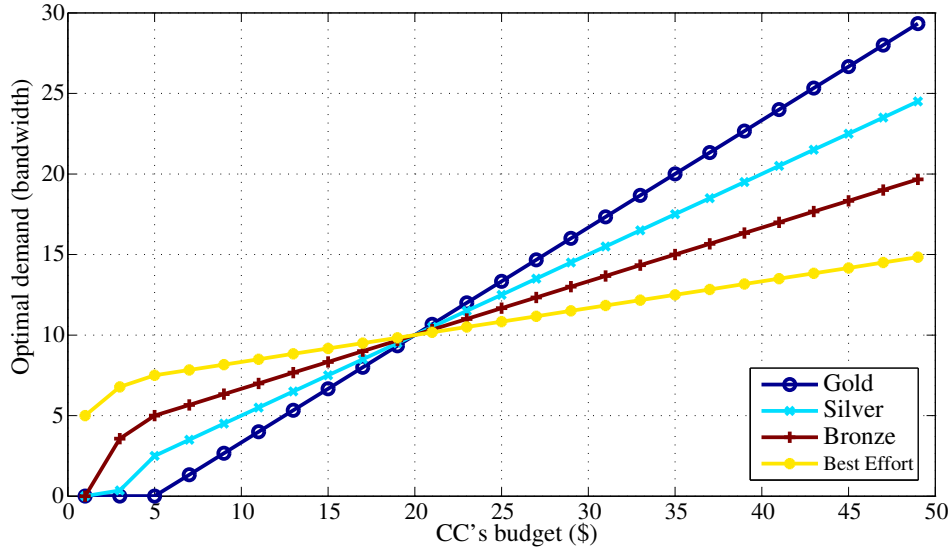

Figure 3.10: Optimal bandwidth demand for each virtual pipe as the CC's budget increases.

Assumptions for simplicity:

- The VNaaS model assumes the existence of appropriate VM placement mechanisms for the initial phase of embedding the VNs, hence, the proposed VNaaS model focuses on the ongoing process of continuously re-optimizing the physical network resources allocated to the VNaaS.
- There is an assumption that the SLA negotiations process are performed according to any SLA negotiation Protocol.
- The VNaaS pricing model assumes the existence of a preset profit threshold, Γ, on the net profit. This threshold can be determined according to market competition or regulations.

Scheme limitations:

- The VNaaS model targets the inter-data centers networks (i.e., may provide partially optimal allocation in case of intra-data center networks due to the special topologies of the latter).
- The VNaaS model focuses on VNs serving CCs that are characterized by relatively longer life-times and time varying demands (i.e., may provide partially optimal allocation in the case of short-lived and small VNs).
- The utility function $U^v(.)$ is assumed to be quasi-concave, increasing and twice differentiable in the domain defined by $x_i^v \in [\underline{b}_i^v, \overline{b}_i^v]$, for all $e_i^v \in E^v$, following existing approaches in the literature [155].
- $f_i^v(.)$ is assumed to be non-decreasing, continuous and differentiable in all d_{ij}^v.

3.8 Summary

In this chapter, a novel model for the virtualization of geographically-distributed data center networks hosting cloud services has been presented. This model facilitates the description of various service qualities, and in particular, the service latency, for the hosted cloud applications. The CC's demand is formulated as an optimization problem in which the CC seeks to request the optimal bandwidth resources for a number of individual virtual pipes connecting its VMs. This is carried out in order to maximize its utility which is obtained from serving its running applications subject to the CCs budget constraints. Then, a demonstration on how the complexity of this problem can be reduced has been presented. This was carried out by posing it as a two-stage budget allocation problem and derived closed formulation in order to calculate an optimal price index and an effective bandwidth for each virtual link. Also the solution of the original problem has been extended to include additional internal routing constraints. Finally, a corresponding Ramsey-based pricing mechanism has been proposed that can be used by the CSP to realize the CCs demands. Experimental results have shown an improvement in cloud services quality for the CCs and an increase in the CSP's profit.

Chapter 4

Modeling and Pricing VN Elasticity-as-a-Service

The previous chapter presented the proposed Virtual Network-as-a-Service (VNaaS) model and its features. This model allows the CCs to decide the contracted bandwidth amount that maximize their utilities. Chapter 4 describes a novel Elasticity-as-a-Service (EaaS) model which enables the large-scale clouds to cope with the time-varying workload fluctuation. Consideration is given to the scenario of large-scale public clouds that are geographically distributed to demonstrate how the proposed techniques ensure the performance of distributed cloud applications. This chapter can be outlined as follows. Section 4.3 presents the proposed elastic service offerings model. Sections 4.4 and 4.5 discuss the proposed elasticity model, considering the case of a single elasticity class and multiple classes, respectively. In sections 4.6, the EaaS pricing mechanism is presented. Simulation and performance evaluation results are discussed in Section 4.7. Finally, Section 4.9 concludes this chapter.

4.1 Overview

CSPs strive to effectively provision their cloud resources to ensure that their hosted long-term distributed applications meet their performance guarantees. However, accurately provisioning the inter-data centers network resources remains a challenging problem due to the cloud hosted applications' workload fluctuation. In this chapter, a novel approach is proposed that enables a CSP to offer EaaS for inter-data centers communication in

order to ensure the performance of large-scale distributed cloud services which are long-term CCs.

The contributions of the proposed work are three fold; first, an efficient approach is developed that enables the CSP to estimate and reserve the required pool of network resources. This is carried out to fulfill the demands imposed by the network workload fluctuations for the applications subscribing to this service. This approach allows the CSP to offer communication EaaS at a specific elasticity level based on the distributed cloud applications' degree of bandwidth-sensitivity.
The second contribution is to consider multiple elasticity levels and allow the CCs (i.e., the SPs) who require the same elasticity level to have different time-varying behavior. The third contribution is a novel dynamic pricing mechanism for network EaaS offerings that can be employed by the CSP to maximize the expected long-term revenue, and to regulate the EaaS demands. In order to capture the inter-data centers network activity of the hosted applications, their workloads are modeled using Markovian modeling. Performance evaluation results demonstrate the efficiency of the proposed approach, the higher accuracy of the prediction method, and the increase in the CSPs net profit.

4.2 Inter-data Centers Workload Fluctuation

The ability of cloud computing environments to offer scalable and cost-efficient computing and networking resources has contributed to the growth of large-scale geographically distributed applications. These applications are hosted on clouds and use dedicated VMs residing on physical servers that are dispersed throughout multiple data centers [7, 17]. Usually, CCs rent VNs for a long-term to offer various services to end-users (e.g., video on demand and online gaming). The distributed deployment of the cloud applications

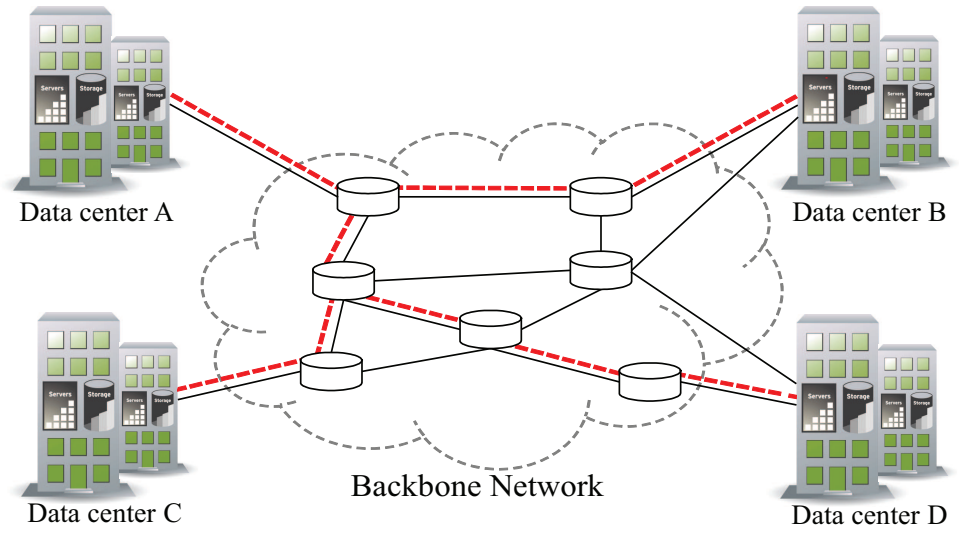

Figure 4.1: An example of an inter-data centers backbone network

is carried out with various objectives such as the availability against hardware failures, energy saving factors, environmental constraints, and cost reduction [17, 162]. For instance, large-scale social media streaming applications need to be hosted among several data centers in order to migrate their contents according to different group of users' dynamic demands with the goal of increasing service availability [163]. In addition, efficient replication mechanisms are employed throughout geographically distributed data centers, to achieve higher service availability against hardware failures [17, 162].

To facilitate the communication among these distributed VMs, CSPs own or lease a high-capacity backbone network to carry the CCs traffic between data centers [11,17,121]. Network virtualization techniques [29] are then employed for the network resource management [164] as shown by Fig. 4.1. Ideally, virtualization allows each CC to rent the bandwidth needed for its inter-data centers communication in order to dynamically create a VN that operates in isolation from other collocated cloud applications. Unfortunately, the majority of current distributed cloud-based applications are long-lived services and

characterized by a high degree of workload fluctuation. In turn, these applications require a continuous down- or up-scaling for the amount of allocated resources to accurately reflect the time-varying application's needs [40–42]. This fluctuation is largely attributed to the nature of the offered services and/or other external events that may result in incremental growth or sudden variation in popularity (e.g., releasing of a new movie, publishing of an article about a particular company on a highly visited news site, or popularity participating in a conference).

Cloud service elasticity refers to the ability of the CSP to dynamically allocate the resources according to the cloud application needs. Nonetheless, one of the limitations of the current cloud resource provisioning techniques is that they mainly focus on efficient and dynamic allocation of computational and network resources within data centres [36], [38], without paying much attention to inter-data centers network provisioning, this highly affects the performance of these distributed applications [47]. Moreover, VMs replication/migration and internal background processes (e.g., periodical data backup, data indexing, database distribution and synchronization, video encoding and caching, and MapReduce jobs) result in a huge amount of data being transferred among different data centers [41, 127]. Therefore, the inter-data centers traffic load varies significantly, which stresses the need for inter-data centers network EaaS, a value-added service where the CSP allows the hosted applications to down- or up-scale its consumed resources in a continuous manner.

Existing approaches are limited to either provisioning unbounded network resources, or statically provisioning the expected peak-traffic value. Both solutions lead to inefficient resource utilization [37]. Likewise, under-provisioning is not acceptable since it results

in performance degradation which implies violating the SLA and incurring monetary penalties. So far, however, very few inter-data centers dynamic network provisioning techniques have been developed (e.g., [47]). Nonetheless, these techniques do not suggest a way that ensures the availability of network resources in response to workload changes. Moreover, these techniques either assume that the time-varying probability distribution of cloud applications is well known, or ignore the repetitive behavior of long-lived cloud applications, resulting in inefficient prediction approaches [57, 58, 62, 123].

To overcome the aforementioned problems, the thesis proposes a novel inter-data centers network resource management approach that dynamically re-sizes the inter-data centers bandwidth pool in order to ensure the availability of network resources. This enables the CSP to offer the promised network EaaS. The CSP decides for each type of distributed cloud-based application, a pre-defined elasticity level that reflects the degree of bandwidth-sensitivity that it can tolerate. Based on the decided elasticity level and the expected traffic workload fluctuation, the CSP estimates the amount of network resources needed to be reserved for the next time interval[1]. After estimating the required EaaS pool size, the CSP either sets apart a portion of the network resources, if available, or it can outsource cloud resources across multiple CSPs [122].

In order to have an accurate prediction, the inter-data centers traffic workload is modeled using a Markov chain model, taking advantage of the temporal variability of workload fluctuations. In addition, Markovian modeling offers significant advantages over other modeling techniques in terms of the complexity of analysis, scalability and time. This model can easily and accurately capture the repetitive workload behavior of long-term stationary applications (e.g., Netflix) in a straight forward manner and within

[1]The duration of the time intervals is specified by the CSP and depends on the nature of the hosted distributed cloud applications [111]

a reasonable time [42, 165, 166].

4.3 A Novel Elastic Service Offerings

One of the CSP's objectives is to effectively manage the underlying physical network by determining what portion of the network resources should be reserved to offer the EaaS, and the portion that may be rented to serve new cloud VNaaS requests. To answer this question, the CSP needs first to accurately estimate the expected network elastic demands, then compares the trade-off between the long-term revenue earned from the EaaS offerings, and the revenue from accepting new cloud VNaaS requests. The CSP must take into consideration the penalties that may be imposed due to the failure of meeting the elasticity service level due to the unavailability of sufficient network resources. Before describing this thesis' proposed approach, an inter-data centers communication EaaS model is proposed. The CSP considers each inter-data centers link as a separate network resource and estimates the bandwidth amount needed to be reserved according to the expected fluctuation of the virtual links mapped over it. To this end, the CSP's distributed cloud can be modeled as a weighted undirected graph $G(N, L)$, where N and L represent the sets of data centers and inter-data centers links, respectively, as shown by Fig. 4.1. c_e denotes the capacity of $e \in L$, and x_e denotes the EaaS reserved bandwidth pool size of $e \in L$. Each distributed application v is hosted on a set of distributed VMs connected through the inter-data centers network and can be modeled as a weighted undirected graph $G^v(N^v, E^v)$, with the sets of virtual nodes N^v (i.e., VMs, virtual switches, and virtual routers), and virtual edges E^v hosted on an inter-data centers link e. The elastic bandwidth demand of each link $e^v \in E^v$ is denoted by d^v (i.e., d^v is the requested excess of bandwidth over the originally contracted values). Each distributed application specifies its SLO which requires a specific level of network elasticity, as will

now be illustrated.

According to the SLO, the CSP allocates the required resources needed to ensure the application performance objectives. Beside the elasticity of the allocated distributed VMs resources (i.e., CPU, storage, and memory), the CSP considers the elasticity of the inter-data centers network in order to be able to deal with traffic workload fluctuations of each virtual link hosted on the network.

4.3.1 A Novel Elasticity-as-a-Service (EaaS) model

The proposed EaaS model allows the CSP to offer differentiated elasticity services to its distributed cloud applications. The offered elasticity levels vary from perfect elasticity (i.e., ∼100% elasticity level means that the virtual link bandwidth appears to be unlimited) to partial elasticity (i.e., resources are guaranteed up to a certain level). The elasticity level is determined based on the nature of the distributed cloud application and the current stage of the application's life-cycle. In general, applications that belong to the same application type and share the same time-varying behavior have the same network workload demands [167] and can be assigned the same EaaS class (e.g., distributed video streaming cloud applications that serve the same area have the same traffic fluctuation and distribution parameters). According to that, all the distributed applications that have the same traffic workload patterns and similar degree of bandwidth-sensitivity are assumed to be categorized into the same class and are assigned the same elasticity level. Fortunately, this assumption will be released in section 4.5. Then the CSP calculates the required network resource pool size for this class according to its elasticity level.

In section 4.4, only one class of EaaS is considered and the case of multiple classes

will be demonstrated in section 4.5. The CSP estimates the bandwidth pool size $x_e(t)$ on each $e \in L$ for the n cloud applications that belong to the same class at a time. Each EaaS class is characterized by a probability $1 - \alpha$ which represents the probability of elasticity service (i.e., α is the probability that one or more of the applications in that class will request additional resources and that the CSP will fail to satisfy the request). In other words, the value α represents the probability that the total bandwidth elasticity demands $\sum_{j=1}^{n} d^{v_j}(t)$ for the n cloud applications that belong to the same EaaS class, during a specific time interval t exceed the EaaS reserved pool $x_e(t)$ for that class at $e \in L$, i.e.,

$$Pr(\sum_{j=1}^{n} d^{v_j}(t) \geq x_e(t)) = \alpha \quad \forall e \in L \tag{4.1}$$

Clearly, α is inversely proportional to the contracted EaaS level. For instance, bandwidth-sensitive distributed cloud applications (e.g., video streaming) need a high EaaS level (i.e., low probability of demand violation, $\alpha \sim 0$), while bandwidth-insensitive distributed cloud applications (e.g., offline laboratory experiments), which are able to tolerate bandwidth under-provisioning, may require a low elasticity level (i.e., can tolerate large α).

4.3.2 Application Demand Modeling

Predicting the varying inter-data centers communication demands of distributed cloud hosted applications is a necessary step in order to enable the CSP to accurately reserve the resources required to provide the required EaaS. The CSP can determine the expected bandwidth demands by monitoring their workload patterns over time to model the dynamics of the distributed applications [42, 166]. Accordingly, a discrete-time Markov chain model is built based on historical network workload patterns to be able to estimate the short-term bandwidth demands. The state model is first derived by observing

the application operation and stages, and then converted into a Markov chain representation in which each state of the model represents the time-varying probability distribution of the inter-data centers traffic for a given EaaS class. Long-term distributed cloud applications are stationary and have repeated patterns[2] which emphasize the suitability of using Markov modeling [168]. Consideration is given to a Markov chain model consisting of a finite set of states, $\mathcal{S} = \{s_1, s_2, \ldots, s_k\}$, each state $s_i \in \mathcal{S}$ is characterized by a mean $\mu(s_i)$ and a variance $\sigma^2(s_i)$ which are thought to be the main parameters that describe the probability distribution of the traffic workload for a given EaaS class at this state. At each period $t = 1, 2, \ldots$, the probability distribution parameters can be inferred from the current state $s_i \in \mathcal{S}$ that follows a Markov process characterized by a state transition matrix $\Pi = [\pi(s_i|s_j)]$, where $s_i, s_j \in \mathcal{S}$. The process starts at a specific state and moves gradually from one state $s(t-1)$ at time $t-1$ to another state $s(t)$ at time t, with a probability denoted by $\pi(s_j|s_i)$ as follows,

$$\pi(s_j|s_i) = Pr(s(t) = s_j|s(t-1) = s_i) \tag{4.2}$$

The conditional probability $\pi(s_j|s_i)$ follows a Markov process since the next state s_j is dependent only on the current observed state s_i, and is independent of the sequence of states that preceded it. Formally, it can be written as,

$$\begin{aligned} &Pr(s(t) = s_j|s(t-1) = s_i, \ldots, s(1) = s_1) \\ &\quad = Pr(s(t) = s_j|s(t-1) = s_i) \end{aligned} \tag{4.3}$$

The state transition coefficients have the properties that $\pi(s_i|s_i) \geq 0$ and $\sum_{j=1}^{k} \pi(s_i|s_j) = 1$. Fig. 4.2 illustrates an example of a Markov chain model consisting of k states and its

[2]The CAIDA Equinix inter-data centers traces [6] show the repeated behavior of inter-data centers traffic, as will be seen in section 4.7.

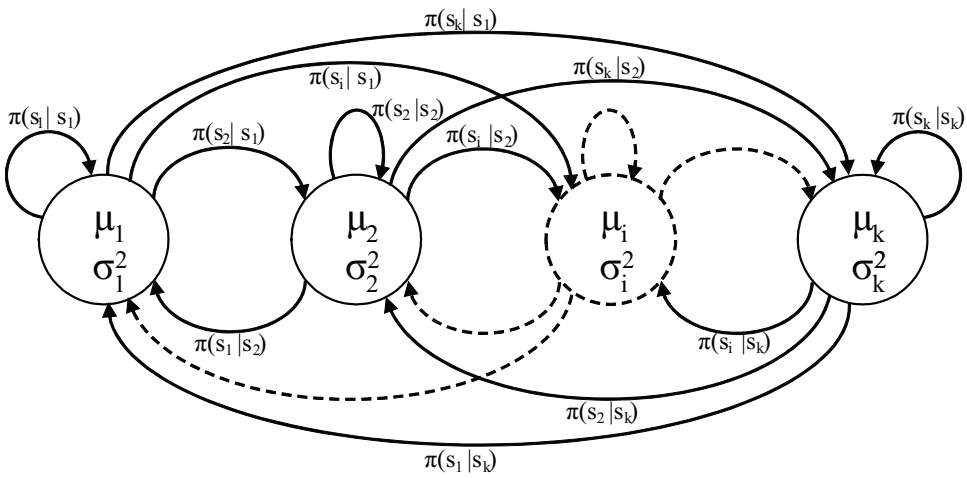

Figure 4.2: Markov-based inter-data centers network model

state transitions.

The CSP estimates the mean and the variance of the probability distribution of the network traffic in discrete time intervals according to the state transition matrix Π given by the first-order Markov chain model which is inferred from historical monitoring. Based on these parameters, the CSP calculates the reserved bandwidth pool size that is needed to provide EaaS for the hosted applications at run-time.

Table 4.1 provides a description of the adopted notations.

4.4 EaaS Reserved Resource Pool Calculation: Single EaaS Class

At each time interval t and for a given EaaS class, the CSP derives the estimated state $s(t) = s_t$ of the Markov chain process for each link e, and determines the mean $\mu(s_t)$ and the variance $\sigma^2(s_t)$ of the bandwidth demands distribution. The bandwidth demand $d^{v_j}(t)$ for an application j, $j = 1, \ldots, n$, is considered as an independent and identically distributed $(i.i.d)$ random variable, with mean $\mu = E(d^{v_j})$ and variance

Table 4.1: A summary of adopted notations (EaaS).

Variable	Description
$G(N, L)$	the graph representing the CSP's geo-distributed Data centers network.
N	the sets of data centers.
L	the sets of inter-data centers links.
c_e	the capacity of inter-data centers link $e \in L$.
x_e	the EaaS reserved bandwidth pool size of $e \in L$.
$G^v(N^v, E^v)$	the graph representing a CC's VN v.
N^v	the sets of virtual nodes.
E^v	the sets of virtual links.
d^v	the elastic bandwidth demand of each link $e^v \in E^v$.
n	the number of cloud applications that belong to a certain EaaS class.
α	the probability that the total bandwidth elasticity demands for the n cloud applications that belong to the same EaaS class, exceeds the EaaS reserved pool.
$\mathcal{S}$	Markov chain states.
$\mu(s_i)$	the mean of the EaaS demands at state s_i.
$\sigma^2(s_i)$	the variance of the EaaS demands at state s_i.
Π	the state transition matrix.

$\sigma^2 = Var(d^{v_j})$. According to the Central Limit Theorem (CLT) [169], since the number of hosted cloud applications n is sufficiently large, the distribution of the arithmetic mean $\bar{d_n}(t) = (d^{v_1}(t) + d^{v_2}(t) + \cdots + d^{v_n}(t))/n$ of the bandwidth demand at state s_t can be approximated as a normal distribution, with mean $\mu(s_t)$ and variance $\sigma^2(s_t)/n$, regardless of the actual unknown distribution of the bandwidth demand (e.g., Exponential, Poisson, Uniform, ...) [169]. Hence, the distribution of the sum of the n random variables $\sum_{j=1}^{n} d^{v_j}(t)$ can also be approximated by a normal distribution with mean $n\mu(s_t)$ and variance $n\sigma^2(s_t)$. Formally, let $D(t) = \sum_{j=1}^{n} d^{v_j}(t)$, $\mu_D(t) = E[D(t)] = n\mu(s_t)$, and $\sigma^2_{D(t)} = Var(D(t)) = n\sigma^2(s_t)$. The CSP calculates the resource pool size $x_e(t)$ at each link e to be reserved at the required elasticity level α, so that the probability that the total bandwidth demands in that EaaS class may exceed the total reserved pool is less than or equal the predefined value α, i.e., $Pr(D(t) \geq x_e(t)) \leq \alpha$. According to the CLT,

$$\frac{D(t) - \mu_{D(t)}}{\sigma_{D(t)}} \sim N(0, 1). \tag{4.4}$$

The estimated pool size at state s_t for each link e can be calculated to satisfy the constraint:

$$Pr(\underbrace{\frac{D(t) - \mu_{D(t)}}{\sigma_{D(t)}}}_{N(0,1)} \geq \underbrace{\frac{x_e(t) - \mu_{D(t)}}{\sigma_{D(t)}}}_{z_{1-\alpha}}) \leq \alpha \quad \forall e \in L \tag{4.5}$$

where $z_{1-\alpha}$ is the $1 - \alpha$ percentile of the Standard Normal distribution. Thus, $x_e(t)$ can now be calculated as follows,

$$x_e(t) = n\mu(s_t) + z_{1-\alpha}\sqrt{n}\sigma(s_t) \tag{4.6}$$

It is worth noting, that this calculation is carried out per each state s_t for each EaaS class.

An Example: Let the workload of a given EaaS class in the current state s_t be characterized with a mean $\mu(s_t) = 10$ Mbps, and a variance $\sigma^2(s_t) = 16$ Mbps, with $n = 30$ virtual links sharing the same link in that EaaS class. At time interval t, the estimated pool size to provide EaaS with an elasticity service probability $1 - \alpha = 0.95$, can be calculated by substituting these values in (4.6). The results are: $x_e(t) = 30 \times 10 + 1.645 \times \sqrt{30} \times 4 = 336.04$ Mbps.

4.5 EaaS Reserved Resource Pool Calculation: Multiple EaaS Classes

In this section, the general EaaS model is proposed, where the assumption that all the applications that belong to the same EaaS class should share the same time-varying behavior and have the same probability distribution parameters is relaxed. Also, multiple elasticity classes are considered during the calculation of the required pool size for each class. The CSP offers $\mathcal{C}$ (≥ 2) elasticity classes with differentiated elasticity levels α_c. All the n_c applications that belong to each class $c \in \{1, \ldots, \mathcal{C}\}$ are categorized into $|\mathcal{K}_c|$ clusters, where $\mathcal{K}_c = \{k_{c,1}, \ldots, k_{c,i}, \ldots, k_{c,|\mathcal{K}_c|}\}$, and all the applications within a cluster share the same probability distribution parameters following these assumptions:

- The bandwidth demands for each cluster $k_{c,i}$ are independent, where $c \in \{1, \ldots, \mathcal{C}\}$, and $i = \{1, ..., |\mathcal{K}_c|\}$.
- Cluster $k_{c,i}$ contains $n_{c,i}$ i.i.d bandwidth demands, $d^{v_1}(t), ..., d^{v_{n_{c,i}}}(t)$ distributed as $d_{c,i}(t)$, with mean $\mu_{c,i}$ and variance $\sigma^2_{c,i}$, where $c \in \{1, \ldots, \mathcal{C}\}$, and $i = \{1, ..., |\mathcal{K}_c|\}$.
- Assume $n_{c,i}$ are large enough to implement the CLT.

For any random variable X, let $X^* = \frac{X - E[X]}{\sqrt{var[X]}}$.

Let the total demand of cluster $k_{c,i}$, $D_{c,i}(t) = \sum_{j=1}^{n_{c,i}} d^{v_j}(t)$. Clearly, $E[D_{c,i}(t)] = n_{c,i}\mu_{c,i}(t)$

and $Var[D_{c,i}(t)] = n_{c,i}\sigma^2_{c,i}(t)$. By applying CLT $D^*_{c,i}(t) \sim N(0,1)$.
Let $D_c(t) = \sum_{i=1}^{|\mathcal{K}_c|} D_{c,i}(t)$, with mean $\mu_c(t)$ and variance $\sigma^2_c(t)$, where $\mu_c(t) = E[D_c(t)] = \sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\mu_{c,i}(t)$ and $\sigma^2_c(t) = var[D_c(t)] = \sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\sigma^2_{c,i}(t)$.
Furthermore, by applying CLT, $D^*_c(t) \sim N(0,1)$ as follows,

$$D^*_c(t) = \frac{D_c(t) - \mu_c(t)}{\sigma_c(t)} \sim N(0,1) \tag{4.7}$$

Next, different approaches are described to help determining the required reserved bandwidth pool size for each cluster $k_{c,i}$ within class c, namely, the individual approach and the global approach.

4.5.1 The Individual Approach

The CSP specifies for each cluster $k_{c,i}$ the required elasticity level which is the same as that of the class α_c. For each cluster the required reserved bandwidth pool size is calculated as the case of single elasticity class using (4.6), i.e.,

$$x^{c,i}_e(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\sqrt{n_{c,i}}\sigma_{c,i}(t) \tag{4.8}$$

This approach considers only the size of each cluster $k_{c,i}$ and does not consider the effect of the total number of applications $n_c = \sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}$ sharing the same class $c \in \{1,\ldots,\mathcal{C}\}$.

4.5.2 The Global Approach

The CSP determines the required bandwidth pool size for each cluster such that the elasticity level is α_c. According to the results from the previous section, since the number of hosted cloud applications per each cluster $n_{c,i}$ is sufficiently large, the distribution

of the arithmetic mean $\bar{d}_{n_{c,i}}(t) = (d^{v_1}(t) + d^{v_2}(t) + \cdots + d^{v_{n_{c,i}}}(t))/n_{c,i}$ of the bandwidth demand at state s_t can be approximated as a normal distribution with mean $\mu_{c,i}(s_t)$ and variance $\sigma^2_{c,i}(s_t)/n_{c,i}$, regardless of the actual unknown distribution of the bandwidth demand [169]. Since all $D_{c,i}(t)$, where $c \in \{1, \ldots, \mathcal{C}\}$, and $i = \{1, ..., |\mathcal{K}_c|\}$, are independent random variables that are normally distributed, then their sum is also normally distributed. Based on this fact, the CSP is able to calculate the required total pool size ensuring that each individual cluster $k_{c,i}$ receives its required elasticity level α_c as follow,

$$Pr(\sum_{i=1}^{|\mathcal{K}_c|} D_{c,i}(t) \geq \sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t)) \leq \alpha_c. \tag{4.9}$$

According to the assumption that all $n_{c,i}$ are large enough to implement the CLT, the results are:

$$\sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t) = \mu_c(t) + z_{1-\alpha_c}\sigma_c(t) = \sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\sqrt{\sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\sigma^2_{c,i}(t)} \tag{4.10}$$

Equation (4.10) presents the total bandwidth pool size needed to be reserved in order to achieve the elasticity level α_c required by class c, where $c \in \{1, \ldots, \mathcal{C}\}$.

By comparing the total reserved pool size according to the individual and the global approach, the following proposition can be concluded:

Proposition 4. *The global approach requires less resources to be reserved, hence cheaper*

elasticity classes than the individual approach. This inequality can be expressed as

$$\underbrace{\sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\sqrt{\sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}\sigma^2_{c,i}(t)}}_{Global\ approach} < \sum_{i=1}^{|\mathcal{K}_c|} \underbrace{\left(n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\sqrt{n_{c,i}}\sigma_{c,i}(t)\right)}_{Individual} \tag{4.11}$$

Proof:

The inequality, $\sum a_i^2 \leq (\sum a_i)^2$ where $a_i > 0$, implies that the left hand side of (4.11) is the sum of all the required pools needed for each cluster calculated according to the global approach for the elasticity level α_c. However, the right hand side of (4.11) is the sum of all the required bandwidth pool calculated according to the individual approach by taking the elasticity level α_c for each cluster. Thus the global approach requires less resources to be reserved. Hence cheaper elasticity classes will be offered.

The problem that remains is how to calculate the bandwidth pool size needed for each cluster in order to provide an accurate pricing scheme as can be seen in section 4.6. Several approaches can be used to calculate the required pool size for each cluster $k_{c,i}$. First, let the required pool size $x_e^{c,i}(t)$ for each cluster

$$k_{c,i}$$

be defined as a function of $T_{c,i}$ as follows,

$$x_e^{c,i}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c} n_{c,i} T_{c,i}, \tag{4.12}$$

where $T_{c,1}, \ldots, T_{c,|\mathcal{K}_c|}$ are non-negative numbers such that,

$$\sum_{i=1}^{|\mathcal{K}_c|} n_{c,i} T_{c,i} = \sqrt{\sum_{i=1}^{|\mathcal{K}_c|} n_{c,i} \sigma_{c,i}^2(t)} = \sigma_c(t) \tag{4.13}$$

Clearly, there is an infinite number of options to determine the value of $T_{c,i}$, where $i = 1, ..., |\mathcal{K}_c|$. These approaches include the following:

(1) Uniform Allocation

The class standard deviation $\sigma_c(t)$ defined in (4.13), is split equally among all the applications within this class, that is $T_{c,1} = ... = T_{c,|\mathcal{K}_c|} = \frac{\sigma_c(t)}{n_c}$, where $n_c = \sum_{i=1}^{|\mathcal{K}_c|} n_{c,i}$. The reserved bandwidth pool size for cluster $k_{c,i}$ is:

$$x_e^{c,i}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c} n_{c,i} \frac{\sigma_c(t)}{n_c}, \tag{4.14}$$

(2) Semi-uniform Allocation

The class standard deviation $\sigma_c(t)$ is split equally among the clusters. Here $T_{c,i} = \frac{\sigma_c(t)}{n_{c,i}|\mathcal{K}_c|}$. The reserved bandwidth pool size for cluster $k_{c,i}$ is:

$$x_e^{c,i}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c} \frac{\sigma_c(t)}{|\mathcal{K}_c|}, \tag{4.15}$$

(3) Proportional Allocation

The class variance $\sigma_c^2(t)$ is split among the clusters proportionally to their variances. Thus, $T_{c,i}$ is determined as $\frac{r_{c,i}\sigma_c(t)}{n_{c,i}}$, where $r_{c,i} = \frac{n_{c,i}\sigma_{c,i}^2(t)}{\sigma_c^2(t)}$. The reserved bandwidth pool size for cluster c, i is:

$$x_e^{c,i}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c} \frac{r_{c,i}\sigma_c(t)}{n_{c,i}}, \tag{4.16}$$

At this point, the problem of accurately determining the required pool size for each

cluster $k_{c,i}$ will be modeled as a non-linear optimization problem.

4.5.3 EaaS Pool Calculation given the EaaS level: Optimization Problem

The main objective here is to ensure that the probability of the total bandwidth fluctuations exceeding the total reserved pool is a pre-determined small number α_c, $0 < \alpha_c < 1$. In addition, the reserved pool size should be proportional to the expected bandwidth fluctuation for each cluster. In other words, the reserved pool size should increase as the expected fluctuation increases. This problem is modeled as a non-linear optimization problem such that the sum of the expected weighted squared distances between the actual fluctuation and the reserved pool of each cluster is minimized.

Let $\boldsymbol{x}_e^c = (x_e^{c,1}, \ldots, x_e^{c,i}, \ldots, x_e^{c,|\mathcal{K}_c|})$, and $r_1, \ldots, r_i, \ldots, r_{|\mathcal{K}_c|}$ be positive numbers. The optimization problem can be modeled as a minimization problem as follows,

$$\begin{aligned} \boldsymbol{x}_e^{c*}(t) = \quad & \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} \left[\frac{1}{r_i} E(D_{c,i}(t) - x_e^{c,i}(t))^2 \right] \right\} \\ \text{s.t.} \quad & \sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t) = \mu_c(t) + z_{1-\alpha_c}\sigma_c(t) \end{aligned} \tag{4.17}$$

Proposition 5. *The minimization problem* (4.17) *has a unique solution which is:*

$$x_e^{c,i*}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\frac{r_i\sigma_c(t)}{r} \qquad i = 1, \ldots, |\mathcal{K}_c| \tag{4.18}$$

where $r = \sum_{i=1}^{|\mathcal{K}_c|} r_i$.

Proof:

Note that for $i = 1, ..., |\mathcal{K}_c|$, and $c \in \{1, \ldots, \mathcal{C}\}$, and using elementary properties of variances of random varaibles:

$$\begin{aligned} Var(D_{c,i}(t) - x_e^{c,i}(t)) &= E(D_{c,i}(t) - x_e^{c,i}(t))^2 - E^2(D_{c,i}(t) - x_e^{c,i}(t)) \\ E(D_{c,i}(t) - x_e^{c,i}(t))^2 &= Var(D_{c,i}(t) - x_e^{c,i}(t)) + E^2(D_{c,i}(t) - x_e^{c,i}(t)) \\ &= Var(D_{c,i}(t)) + E^2(D_{c,i}(t) - x_e^{c,i}(t)) \\ &= n_{c,i}\sigma_{c,i}^2(s_t) + (E(D_{c,i}(t)) - x_e^{c,i}(t))^2 \end{aligned} \tag{4.19}$$

Hence,

$$E(D_{c,i}(t) - x_e^{c,i}(t))^2 = n_{c,i}\sigma_{c,i}^2(t) + (n_{c,i}\mu_{c,i}(t) - x_e^{c,i}(t))^2 \tag{4.20}$$

Substituting (4.20) in the objective function of the minimization problem, yields the following:

$$\begin{aligned} \boldsymbol{x}_e^{c*}(t) &= \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} \left[\frac{1}{r_i} E(D_{c,i}(t) - x_e^{c,i}(t))^2 \right] \right\} \\ &= \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} \left[\frac{1}{r_i} (n_{c,i}\sigma_{c,i}^2(t) + (n_{c,i}\mu_{c,i}(t) - x_e^{c,i}(t))^2) \right] \right\} \\ &= \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} \frac{n_{c,i}\sigma_{c,i}^2(t)}{r_i} + \sum_{i=1}^{|\mathcal{K}_c|} \frac{1}{r_i} (n_{c,i}\mu_{c,i}(t) - x_e^{c,i}(t))^2 \right\} \end{aligned} \tag{4.21}$$

Clearly, eliminating the expression $\sum_{i=1}^{|\mathcal{K}_c|} \frac{n_{c,i}\sigma_{c,i}^2(t)}{r_i}$ will not change the optimal solution.

For simplicity, let $w_{c,i}(x_e^{c,i}(t)) = n_{c,i}\mu_{c,i}(t) - x_e^{c,i}(t)$, where $i = 1, ..., |\mathcal{K}_c|$, and $c \in \{1, \ldots, \mathcal{C}\}$. Problem (4.21) is equivalent to the following problem:

$$\begin{aligned} \boldsymbol{x}_e^{c*}(t) = \quad & \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} \left[\frac{1}{r_i} w_{c,i}^2(x_e^{c,i}(t)) \right] \right\} \\ & \text{s.t.} \sum_{i=1}^{|\mathcal{K}_c|} w_{c,i}(x_e^{c,i}(t)) = -z_{1-\alpha_c}\sigma_c(t) \end{aligned} \tag{4.22}$$

The constraint in problem (4.22) can be rewritten as follows,

$$-w_{c,1}(x_e^{c,1}(t)) = \sum_{i=2}^{|\mathcal{K}_c|} w_{c,i}(x_e^{c,i}(t)) + z_{1-\alpha_c}\sigma_c(t) \tag{4.23}$$

let $G(x_e(t)) = \sum_{i=1}^{|\mathcal{K}_c|} \frac{1}{r_i} w_{c,i}^2(x_e^{c,i}(t))$. Substituting (4.23) in $G(x_e(t))$ yields:

$$\begin{aligned} G(x_e(t)) = \quad & \frac{1}{r_1} w_{c,1}^2(x_e^{c,1}(t)) + \sum_{i=2}^{|\mathcal{K}_c|} \frac{1}{r_i} w_{c,i}^2(x_e^{c,i}(t)) \\ = \quad & \frac{1}{r_1} \left(\sum_{i=2}^{|\mathcal{K}_c|} w_{c,i}^2(x_e^{c,i}(t)) + z_{1-\alpha_c}\sigma_c(t) \right)^2 + \sum_{k=2}^{|\mathcal{K}_c|} \frac{1}{r_i} w_{c,i}^2(x_e^{c,i}(t)) \end{aligned} \tag{4.24}$$

Thus, problem (4.22) is equivalent to:

$$\boldsymbol{x}_e^{c*}(t) = \arg\min_{\boldsymbol{x}_e^c(t)} \{G(x_e(t))\} \tag{4.25}$$

First, the extreme points of $G(x_e(t))$ are found, that is, the vectors $\boldsymbol{x}_e^c(t)$ for which

all the first partial derivatives vanish. The first partial derivative of $w_{c,i}(x_e^{c,i}(t))$ is -1 and the first partial derivative of (4.24) are:

$$\frac{\partial G(x_e(t))}{\partial x_{c,i}} = \frac{-2}{r_1}\left(\sum_{i=2}^{|\mathcal{K}_c|} w_{c,i}(x_e^{c,i}(t)) + z_{1-\alpha_c}\sigma_c(t)\right) - \frac{2}{r_i}w_{c,i}(x_e^{c,i}(t)) \qquad \forall i = 2, ..., |\mathcal{K}_c| \tag{4.26}$$

Equating the $|\mathcal{K}_c| - 1$ partial derivatives in (4.26) to 0 yields the following system of linear equations for $w_{c,1}((x_e^{c,1}(t))), ..., w_{c,|\mathcal{K}_c|}((x_e^{c,|\mathcal{K}_c|}(t)))$:

$$\begin{cases} (\frac{1}{r_1} + \frac{1}{r_2})w_{c,2} + \frac{1}{r_1}\sum_{\substack{i=2 \\ i\neq 2}}^{|\mathcal{K}_c|} w_{c,i} = -\frac{z_{1-\alpha_c}\sigma_c}{r_1} \\ (\frac{1}{r_1} + \frac{1}{r_3})w_{c,3} + \frac{1}{r_1}\sum_{\substack{i=2 \\ i\neq 3}}^{|\mathcal{K}_c|} w_{c,i} = -\frac{z_{1-\alpha_c}\sigma_c}{r_1} \\ \vdots \\ (\frac{1}{r_1} + \frac{1}{r_{|\mathcal{K}_c|}})w_{c,|\mathcal{K}_c|} + \frac{1}{r_1}\sum_{\substack{i=2 \\ i\neq |\mathcal{K}_c|}}^{\mathcal{K}} w_{c,i} = -\frac{z_{1-\alpha_c}\sigma_c}{r_1} \end{cases} \tag{4.27}$$

where $w_{c,i} = w_{c,i}(x_e^{c,i}(t))$. Subtracting the first equations from each of the other equations yields the following set of linear equations (written in matrix form)

$$\begin{bmatrix} \frac{1}{r_1} + \frac{1}{r_2} & \frac{1}{r_1} & \dots & \dots & \dots & \frac{1}{r_1} & -\frac{z_{1-\alpha_c}\sigma_c}{r_1} \\ \frac{1}{r_2} & -\frac{1}{r_3} & 0 & \dots & \dots & 0 & 0 \\ \vdots & 0 & 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & 0 & \dots & \dots & 0 & \vdots \\ \frac{1}{r_2} & 0 & \dots & \dots & 0 & -\frac{1}{r_{|\mathcal{K}_c|}} & 0 \end{bmatrix} \tag{4.28}$$

The last $|\mathcal{K}_c| - 2$ rows yield,

$$r_2 w_{c,i} = r_k w_{c,2} \qquad \forall i = 3, \dots, |\mathcal{K}_c| \tag{4.29}$$

The first row presents the equation

$$(\frac{1}{r_1}+\frac{1}{r_2})w_{c,2}+\frac{1}{r_1}\sum_{i=3}^{|\mathcal{K}_c|}w_{c,i}=-\frac{z_{1-\alpha_c}\sigma_c}{r_1} \tag{4.30}$$

By substituting (4.29) in (4.30) and then multiplying (4.30) by $r_1 r_2$, the following results are obtained:

$$\sum_{i=1}^{|\mathcal{K}_c|}r_i w_{c,2}=-r_2 z_{1-\alpha_c}\sigma_c(t) \tag{4.31}$$

Hence,

$$w_{c,2}=\frac{-r_2 z_{1-\alpha_c}\sigma_c(t)}{r} \tag{4.32}$$

where $r=\sum_{i=1}^{|\mathcal{K}_c|}r_i$. Substituting (4.32) in (4.29) yields,

$$w_{c,i}=\frac{-r_i z_{1-\alpha_c}\sigma_c(t)}{r}, \qquad i=3,\ldots,|\mathcal{K}_c| \tag{4.33}$$

Finally, substituting (4.32) and (4.33), in (4.23) yields

$$w_{c,1}=\frac{-r_1 z_{1-\alpha_c}\sigma_c(t)}{r} \tag{4.34}$$

Since $w_{c,i}(x_e^{c,i}(t))=n_{c,i}\mu_{c,i}(t)-x_e^{c,i}(t)$ the extreme point $\boldsymbol{x}_e^{c*}=\{x_e^{c,1*},\ldots,x_e^{c,i*},\ldots,x_e^{c,|\mathcal{K}_c|*}\}$ for problem (4.25) is

$$x_e^{c,i*}(t)=n_{c,i}\mu_{c,i}(t)+z_{1-\alpha_c}\frac{r_i\sigma_c(t)}{r} \qquad i=1,\ldots,|\mathcal{K}_c| \tag{4.35}$$

Next, it is required to verify that the Hessian matrix of the function given by (4.24) at $\boldsymbol{x}_e^{c*}(t)$ is a positive definite matrix to prove that (4.35) is the unique optimal solution [170].

The second partial derivatives of $G(x_e(t))$ is derived as follows,

$$\begin{aligned}\frac{\partial^2 G(x_e(t))}{\partial x_{c,i}^2} &= \frac{2}{r_1} + \frac{2}{r_{|\mathcal{K}_c|}} \\ &= \frac{2(r_1 + r_{|\mathcal{K}_c|})}{r_1 r_{|\mathcal{K}_c|}} \qquad i = 2, \ldots, |\mathcal{K}_c|\end{aligned} \tag{4.36}$$

$$\frac{\partial^2 G(x_e(t))}{\partial x_{c,i} \partial x_{k_{c,j}}} = \frac{2}{r_1} \qquad 2 \leq i < j \leq |\mathcal{K}_c| \tag{4.37}$$

Hence, the Hessian matrix is

$$H = \begin{bmatrix} \frac{2(r_1+r_2)}{r_1 r_2} & \frac{2}{r_1} & \frac{2}{r_1} & \ldots & \frac{2}{r_1} \\ \frac{2}{r_1} & \frac{2(r_1+r_3)}{r_1 r_3} & \frac{2}{r_1} & \ldots & \vdots \\ \vdots & \ldots & \ddots & \ldots & \vdots \\ \vdots & \ldots & \ldots & \ddots & \frac{2}{r_1} \\ \frac{2}{r_1} & \ldots & \ldots & \frac{2}{r_1} & \frac{2(r_1+r_{|\mathcal{K}_c|})}{r_1 r_{|\mathcal{K}_c|}} \end{bmatrix} \tag{4.38}$$

The property of a positive definite is not changed by multiplying the matrix by a positive scalar $\frac{r_1}{2}$ [171]. Thus, it is sufficient to show that the following matrix is a positive definite:

$$\tilde{H} = \begin{bmatrix} \frac{(r_1+r_2)}{r_2} & 1 & 1 & \ldots & 1 \\ 1 & \frac{(r_1+r_3)}{r_3} & 1 & \ldots & \vdots \\ \vdots & \ldots & \ddots & \ldots & \vdots \\ \vdots & \ldots & \ldots & \ddots & 1 \\ 1 & \ldots & \ldots & 1 & \frac{(r_1+r_{|\mathcal{K}_c|})}{r_{|\mathcal{K}_c|}} \end{bmatrix} \tag{4.39}$$

$\tilde{H}$ is a symmetric, hence, (4.35) is the unique solution to (4.17).

4.5.4 EaaS Pool Calculation Given the Expected Fluctuation: Optimization Problem

In the previous section, the required pool size for each cluster was found such that the expected squared deviation between the actual traffic needs and the reserved pool is minimized, subject to a given elasticity level. However, in some cases it is difficult for the CSP to rigidly define the required EaaS level for each application. Nevertheless, the CSP may define the expected deviation between the actual fluctuation and the reserved pool according to each cloud application behavior and the negotiated SLA. According to this difficulty, in this section, the problem will be addressed from another point of view; the minimization of the elasticity level under the constraint that the expected squared deviation between the actual fluctuation and the reserved pool is given.

The new optimization problem can be modeled as a minimization problem as follows,

$$\begin{aligned} \boldsymbol{x}_e^{c*}(t) = \quad & \arg\min_{\boldsymbol{x}_e^c(t)} \left\{ Pr\left(D_c(t) > \sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t) \right) \right\} \\ \text{s.t.} \quad & \sum_{i=1}^{|\mathcal{K}_c|} \left[\frac{1}{r_i} E(D_{c,i}(t) - x_{c,i}(t))^2 \right] = \psi \\ & \mu_{c,i}(t) \leq x_{c,i}(t) \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \end{aligned} \tag{4.40}$$

Proposition 6. *The minimization problem* (4.40) *has a unique solution which is:*

$$x_e^{c,i*}(t) = n_{c,i}\mu_{c,i}(t) + n_{c,i} r_i \sqrt{\frac{\Psi}{r}} \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \tag{4.41}$$

Where $r = \sum_{i=1}^{|\mathcal{K}_c|} r_i$ *and* $\Psi = \psi - \sum_{i=1}^{|\mathcal{K}_c|} \frac{n_{c,i}\sigma_{c,i}^2(t)}{r_i}$

Proof:

First, both sides of the objective function inequality are normalized as follows,

$$Pr\left(D_c(t) > \sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t)\right) = Pr\left(\frac{D_c(t) - \mu_c(t)}{\sigma_c(t)} > \frac{\sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t) - \mu_c(t)}{\sigma_c(t)}\right) = \tag{4.42}$$

Minimizing the left-hand side of the objective function of (4.42) with respect to $x_e^{c,i}$, where $i = 1, \ldots, |\mathcal{K}_c|$ is equivalent to the maximization of $\sum_{i=1}^{|\mathcal{K}_c|} x_e^{c,i}(t) - \mu_c(t)$. The problem can be rewritten after substituting (4.20) in the constraints and modifying the objective function as follows,

$$\begin{aligned} \boldsymbol{x}_e^{c*}(t) = \quad & \arg\max_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} (x_e^{c,i}(t) - \mu_{c,i}(t)) \right\} \\ \text{s.t.} \quad & \sum_{i=1}^{|\mathcal{K}_c|} \frac{n_{c,i}\sigma_{c,i}^2(t)}{r_i} + \sum_{i=1}^{|\mathcal{K}_c|} \frac{1}{r_i}(x_e^{c,i}(t) - n_{c,i}\mu_{c,i}(t))^2 = \psi \\ & \mu_{c,i}(t) \leq x_{c,i}(t) \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \end{aligned} \tag{4.43}$$

Let $\Psi = \psi - \sum_{i=1}^{|\mathcal{K}_c|} \frac{n_{c,i}\sigma_{c,i}^2(t)}{r_i}$, and $X_{c,i}(t) = x_e^{c,i}(t) - n_{c,i}\mu_{c,i}(t)$. Problem (4.43) can then be rewritten as follows,

$$\begin{aligned}
&\arg\max_{\boldsymbol{x}_e^c(t)} \left\{ \sum_{i=1}^{|\mathcal{K}_c|} X_{c,i}(t) \right\} \\
&\sum_{i=1}^{|\mathcal{K}_c|} \frac{1}{r_i} X_{c,i}(t)^2 - \Psi = 0 \\
&X_{c,i} \geq 0 \qquad \forall i = 1, \ldots, |\mathcal{K}_c|
\end{aligned} \tag{4.44}$$

To solve this problem, Lagrange multipliers are used as follows,

$$\mathcal{L}(X, \lambda) = \sum_{i=1}^{|\mathcal{K}_c|} X_{c,i}(t) + \lambda \left(\sum_{i=1}^{|\mathcal{K}_c|} \frac{1}{r_i} X_{c,i}(t) - \Psi \right) \tag{4.45}$$

Differentiating $\mathcal{L}(X, \lambda)$ with respect to $X_{c,1}, \ldots, X_{c,i}, \ldots, X_{c,|\mathcal{K}_c|}$ and λ yields a system of $|\mathcal{K}_c|$ equations:

$$\begin{cases} \frac{\partial \mathcal{L}(X,\lambda)}{\partial X_{c,i}} = 1 + 2\lambda \frac{X_{c,i}}{r_i} = 0 \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \\ \frac{\partial \mathcal{L}(X,\lambda)}{\partial \lambda} = \sum_{i=1}^{|\mathcal{K}_c|} -\Psi = 0 \end{cases} \tag{4.46}$$

From the first $|\mathcal{K}_c|$ equations of (4.46) the following expression is obtained:

$$X_{c,i} = -\frac{r_i}{2\lambda} \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \tag{4.47}$$

Since $X_{c,i} \geq 0$ then λ has to be negative. Substituting (4.47) in the last equation of (4.46) yields,

$$\sum_{i=1}^{|\mathcal{K}_c|} \frac{\left(-\frac{r_i}{2\lambda}\right)^2}{r_i} = \Psi \tag{4.48}$$

hence,

$$\frac{1}{4\Psi}\sum_{i=1}^{|\mathcal{K}_c|} r_i = \lambda^2 \tag{4.49}$$

Since $\lambda < 0$, substituting the negative root of λ from (4.49) in the first $|\mathcal{K}_c|$ equations of (4.46) yields a unique solution which is:

$$X^*_{c,i} = -\frac{r_i}{2\lambda} = r_i\sqrt{\frac{\Psi}{r}} \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \tag{4.50}$$

Hence, (4.41) is the unique solution to (4.40).

4.5.5 Multiple Heterogeneous Elasticity Classes

After estimating the required EaaS pool size $x_e^{c,i}(t)$ for each cluster i within each class c, the CSP can either set apart a portion of the network resources, if available, or it can outsource cloud resources across multiple CSPs [122]. However, in some scenarios, the total required pool size for all the classes is greater than the available resource. In this section, the problem of allocating the available resources among the required EaaS classes is modeled as an optimization problem according to the CSP revenue, and the penalties that may be imposed due to the failure of meeting the elasticity service level. The SLA provides a set of penalty functions $\mathcal{P}(.)$ which determine the amount to be payed by the CSP in case of violating the required elasticity level or the expected fluctuation.

Let $p_e^{c,i}(t)$ denotes the price of a unit EaaS at cluster i and each class c. The way of calculating the price vector that maximizes the CSP long-term revenue will be presented in the next section. $\boldsymbol{y}_e^c(t) = \{y_e^{c,1}(t), \ldots, y_e^{c,|\mathcal{K}_c|}(t)\}, \quad \forall c = 1, \ldots, \mathcal{C}$, denotes the output of the optimization problem and represents the set of the allocated bandwidth for each cluster i per each class c, taking into consideration the capacity constraint c_e of the

physical links $e \in E$. The optimization problem can be modeled as a maximization problem for the total CSP's revenue to prioritize higher EaaS classes as follows,

$$\begin{aligned} \boldsymbol{y}_e^{c*}(t) = \quad & \max_{\boldsymbol{y}_e^c} \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} \left\{ (y_e^{c,i}(t) * p_e^{c,i}(t)) - \mathcal{P}(y_e^{c,i}(t), x_e^{c,i}(t)) \right\} \\ \text{s.t.} \quad & \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} y_e^{c,i}(t) \leq c_e \\ & 0 \leq y_e^{c,i}(t) \leq x_e^{c,i}(t) \qquad \forall i = 1, \ldots, |\mathcal{K}_c| \end{aligned}$$

The value of the negotiated SLA penalty function is dependent on the difference between the required EaaS pool size $x_e^{c,i}(t)$ and the allocated resources $y_e^{c,i}(t)$ as shown in (4.51). Violating the SLA terms of a specific EaaS class imposes a penalty proportional with the denied EaaS. In addition, the function $\mathcal{F}(.)$ defines the shape of the penalty function to reflect the degree of dissatisfaction due to SLA violation. For instance, the law of diminishing marginal may be applied [172].

$$\mathcal{P}(y_e^{c,i}(t), x_e^{c,i}(t)) = \begin{cases} 0 & x_e^{c,i}(t) \leq y_e^{c,i}(t) \\ \mathcal{F}(x_e^{c,i}(t) - y_e^{c,i}(t)) & x_e^{c,i}(t) > y_e^{c,i}(t) \end{cases} \tag{4.51}$$

To solve this problem, Lagrange multipliers are used as follows,

$$\begin{aligned} \mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i}) = & \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} \left\{ (y_e^{c,i}(t) * p_e^{c,i}) - \mathcal{P}(y_e^{c,i}(t), x_e^{c,i}(t)) \right\} \\ -\lambda_1 (\sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} y_e^{c,i}(t) - c_e) & - \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} \lambda_2^{c,i} (y_e^{c,i}(t) - x_e^{c,i}(t)) - \lambda_3^{c,i} y_e^{c,i}(t) \end{aligned} \tag{4.52}$$

Differentiating $\mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i})$ with respect to $y_e^{c,i}$, λ_1, $\lambda_2^{c,i}$ and $\lambda_3^{c,i}$ $\quad \forall i = 1, \ldots, |\mathcal{K}_c|$ and $c = 1, \ldots, \mathcal{C}$ yields a system of equations as follows,

$$\frac{\partial \mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i})}{\partial y_e^{c,i}} = -\frac{\partial \mathcal{P}(y_e^{c,i}(t), x_e^{c,i}(t))}{\partial y_e^{c,i}} + p_e^{c,i} - \lambda_1 - \lambda_2^{c,i} - \lambda_3^{c,i} \leq 0$$
$$\forall i = 1, \ldots, |\mathcal{K}_c| \text{ and } c = 1, \ldots, \mathcal{C} \tag{4.53}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i})}{\partial \lambda_1} = \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{|\mathcal{K}_c|} y_e^{c,i}(t) - c_e \leq 0 \tag{4.54}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i})}{\partial \lambda_2^{c,i}} = y_e^{c,i}(t) - x_e^{c,i}(t) \leq 0 \tag{4.55}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{y}_e^c, \lambda_1, \lambda_2^{c,i}, \lambda_3^{c,i})}{\partial \lambda_3^{c,i}} = y_e^{c,i}(t) \geq 0 \tag{4.56}$$
$$\forall i = 1, \ldots, |\mathcal{K}_c| \text{ and } c = 1, \ldots, \mathcal{C}$$

These set of equations along with the constraints (4.54), (4.55) and (4.56) can then be solved to obtain the optimal vector $\boldsymbol{y}_e^c(t) = \{y_e^{c,1}(t), \ldots, y_e^{c,|\mathcal{K}_c|}(t)\}, \quad \forall c = 1, \ldots, \mathcal{C}$, according to the shape of the penalty function $\mathcal{P}(.)$.

4.6 Proposed EaaS Pricing Model

In this section, an elasticity dynamic pricing model is proposed that aims at maximizing the CSP expected long-term revenue. In contrast to the traditional static pricing models which are not tailored to the elastic behavior of cloud traffic workload, the main goal of this model is to provide the optimal price vector obtained from offering the EaaS that maximizes the expected long-term revenue. Let the price vector $\mathbf{P}_e^{c,i} = (p_e^{c,i}(s_1), \ldots, p_e^{c,i}(s_k))$ provides the prices of the EaaS at link e at cluster i and

class c, for each state $s_i \in \mathcal{S}$. The CSP calculates the price vector that maximizes its revenue during an operating time cycle $[0, T]$. This price vector is updated periodically to reflect any variations that have occurred to the EaaS demands patterns. Cloud applications are price-sensitive. In other words, when the prices increase, CCs seek to reduce the workload to the cloud which directly affects the workload traffic distribution. On the contrary, when the prices decrease, CCs will be encouraged to move more of their workload to the cloud. This can be done by varying the price of the cloud-based application. At each period $t \in [0, T]$, the bandwidth pool size $x_e^{c,i}(t)$ is reserved to fulfill the total bandwidth demands of EaaS cluster i at class c, which is calculated based on the current state $s(t) = s_t$ as shown in section 4.5. A scaling-up in the link total demand happens at time t, if $dx_e^{c,i}(t) > 0$, while $dx_e^{c,i}(t) < 0$ indicates a scaling-down ($dx_e^{c,i}(t) = 0$ means no fluctuation). The prices should be bound by a maximum and a minimum values, p^{max} and p^{min} respectively, $p_e^{c,i}(s_t) \in [p_e^{min}, p_e^{max}] \ \forall t \in [0, T]$.

Since, the traffic workload fluctuates over the period T, the CSP aims at finding the price vector that maximizes its revenue. The long-term excepted revenue from a given EaaS cluster i per a class c at link e can be calculated by,

$$\mathcal{R}_e^{c,i}(T) = \int_0^T p_e^{c,i}(s_t) x_e^{c,i}(t) dt \tag{4.57}$$

Now, the problem is to find the optimal price vector $\mathbf{P}_e^{c,i^*}$ that maximizes the CSP revenue denoted by $\mathcal{R}_e^{c,i^*}(T)$, and at the same time, reflects the link utilization level to prevent the under-utilization/congestion by decreasing/increasing the price vector of each link, respectively. The maximum CSP revenue $\mathcal{R}_e^{c,i^*}(T)$ can be considered as the Least Upper Bound (lub) of $\mathcal{R}_e^{c,i}(T)$ that satisfies the prices boundary constraint. To search for the optimal price vector, the dynamic programming algorithm is continuously applied backwards in time [173]. First, consideration is given to the revenue over the time

interval ∂t which denotes the duration of one state s_t. Then, recursively it is possible to derive the maximum CSP revenue and hence the optimal price vector. The probabilities to move to any state s_j, where $j = 1, 2, \ldots, k$, given that the current state is s_t at time t can be derived from the state transition matrix Π. Hence, the long-term excepted CSP revenue can be defined as follows,

$$\mathcal{R}_e^{c,i^*}(T) = \underset{\mathbf{P}_e^{c,i}}{lub}[\underbrace{p_e^{c,i}(s_t)x_e^{c,i}(t)\partial t}_{\text{actual revenue at } \partial t} + \underbrace{\sum_{j=1}^{k} \pi(s_j|s_t)\partial t.\mathcal{R}_e^{c,i^*}(T-\partial t)}_{\text{expected revenue at } T-\partial t}] \tag{4.58}$$

where, the optimal expected CSP revenue $\mathcal{R}_e^{c,i^*}(T)$ consists of the actual revenue during time ∂t, denoted by $p_e^{c,i}(s_t)x_e^{c,i}(t)\partial t$, where $x_e^{c,i}(t)$ is the actual pool size reserved to fulfill the bandwidth demands for cluster i at class c, and the second term represents the expected revenue from the remaining time $T - \partial t$ depending on the probabilities of the workload distribution according to the next state s_j. The expected pool size at state s_j can be calculated as $x_e^{c,i^*}(t) = n_{c,i}\mu_{c,i}(t) + z_{1-\alpha_c}\frac{r_i\sigma_c(t)}{r} \qquad i = 1, \ldots, |\mathcal{K}_c|$, where $r = \sum_{i=1}^{|\mathcal{K}_c|} r_i$ following the results from (4.18).

Thus, the optimal revenue $\mathcal{R}_e^{c,i^*}(T)$ and the optimal price vector $\mathbf{P}_e^{c,i^*}$ can be calculated by recursively substituting the equivalent expression of $\mathcal{R}_e^{c,i^*}(T - \partial t)$ into (4.58).

It is worth noting here that as Markov model is Learnt, a *tâtonnement*-like procedure is used to facilitate the corresponding rate of convergence of the prices. During this process, the optimal prices are calculated based on the state transition matrix until it converges to an equilibrium state when the prices reflect the actual supply and demand. This transient state may re-occur due to external events (e.g., the introduction of competitor SP which results in changing the stationary distribution).

After estimating the optimal long-term revenue $\mathcal{R}_e^{c,i^*}(T)$ through the operating time cy-

Figure 4.3: Equinix data centers [5]

cle $[0, T]$, the CSP is now able to compare the revenue of providing the EaaS, and the expected income from selling the extra network capacities to new VNaaS requests. In addition, the CSP can make the decision of either saving the network capacities for EaaS demands, or selling them.

4.7 Performance Evaluation

To evaluate the accuracy of our proposed inter-data centers EaaS approach, two paths are followed, namely through real inter-data centers' traces and simulations. First, since most data centers are located in North America, this chapter's evaluations are built around Equinix [5] which operates 62 data centers in 13 locations across the four time zones that cover North America, as shown by Fig. 4.3. The evaluation is performed based on the Equinix-Chicago traces. The traffic collection monitor is located at an Equinix data center in Chicago, IL, and is connected to a backbone link between Chicago, IL and Seattle, WA [6]. The traces in this dataset are anonymized using CryptoPan prefix-preserving anonymization [6] using the same key. The anonymized traces are stored in pcap format with timestamps truncated to microseconds [174]. The traces can be read with any software that reads the pcap (tcpdump) format. CoralReef Software Suite and libraries are used to read the traces. These traces are used to obtain packet sizes and time between packet arrivals for different application type. Fig. 4.4 and Fig. 4.5 depict

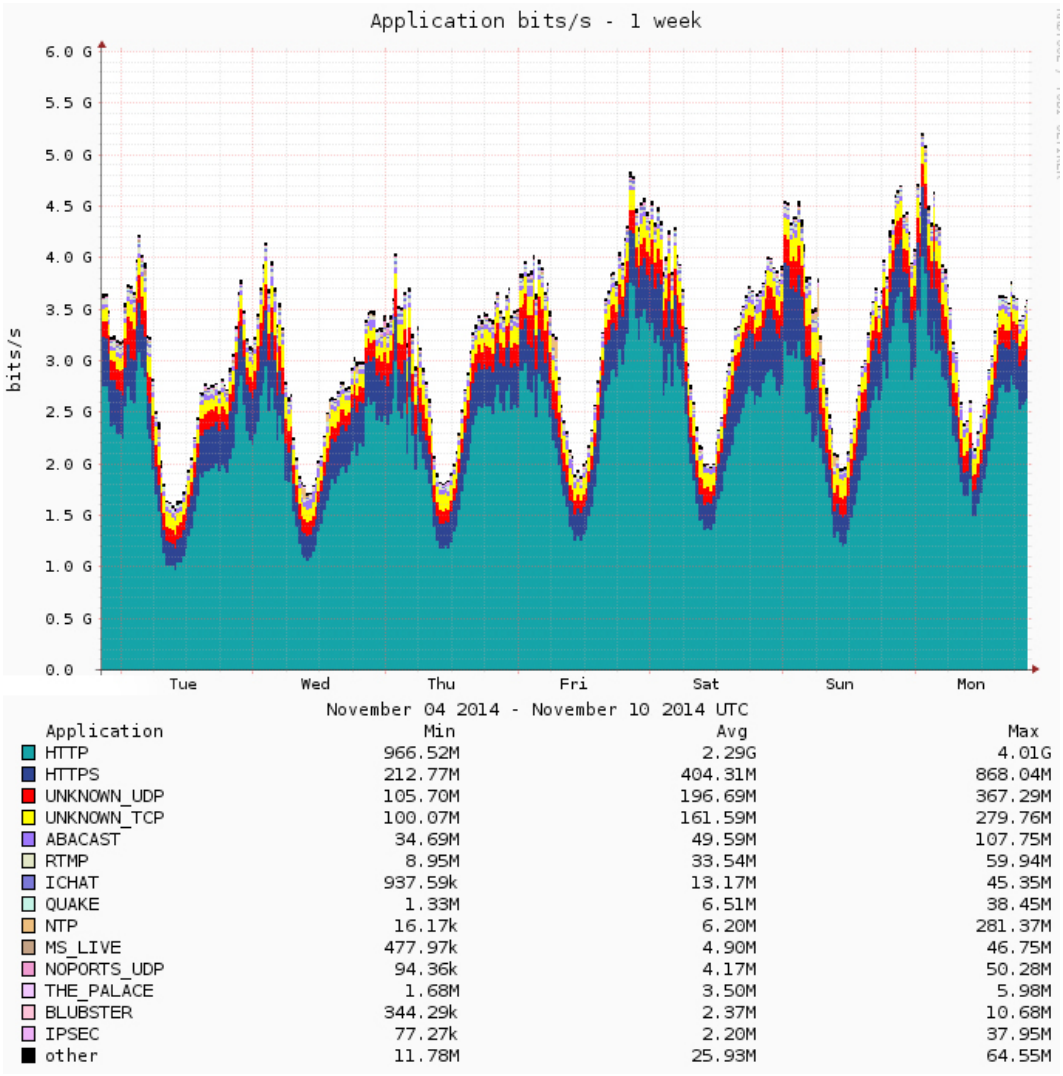

Figure 4.4: Equinix inter-data centers traffic (One week) [6]

the traffic of the Chicago inter-data centers link monitored over one week and four weeks, respectively. From Fig. 4.4 and Fig. 4.5, it is clear that the inter-data centers traffic for each application type is stationary and have repeated patterns, which emphasize the suitability of using Markov modeling.

The second path involves simulating a set of distributed data centers network, where the links' bandwidth values are uniformly distributed between 250 and 300 Mbps. 30

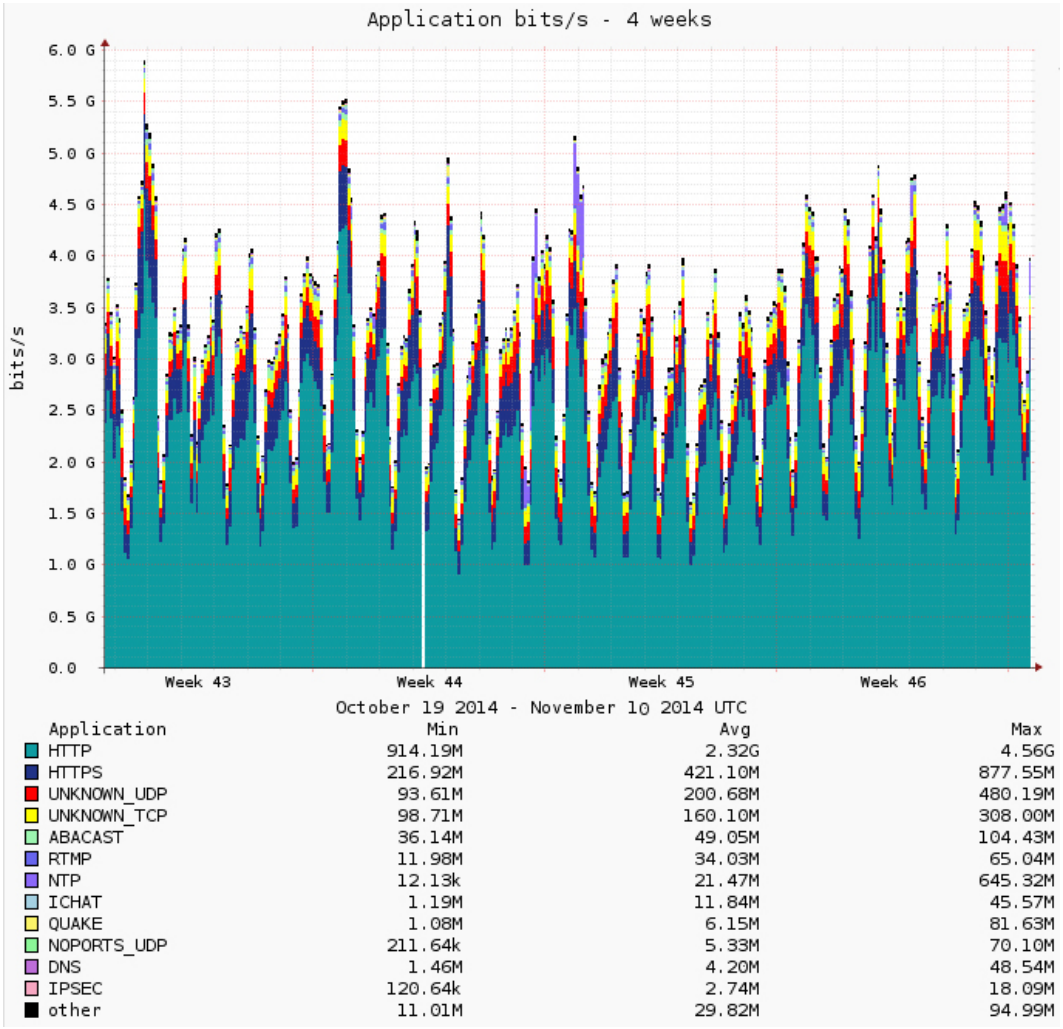

Figure 4.5: Equinix inter-data centers traffic (4 weeks) [6]

distributed cloud applications were simulated, where each inter-data center physical link is shared among them. The simulated cloud applications may experience three possible states of operation, which are characterized by three different traffic workload parameters; state(1) $\mu_1 = 10$ Mbps, $\sigma_1^2 = 10$ Mbps, state(2) $\mu_2 = 15$ Mbps, $\sigma_2^2 = 20$ Mbps, and state(3) $\mu_3 = 25$ Mbps, $\sigma_3^2 = 30$ Mbps. The Markov state transition matrix between states was generated randomly from a normal distribution and normalized to satisfy the transition matrix properties. The experiments simulated two weeks of operation.

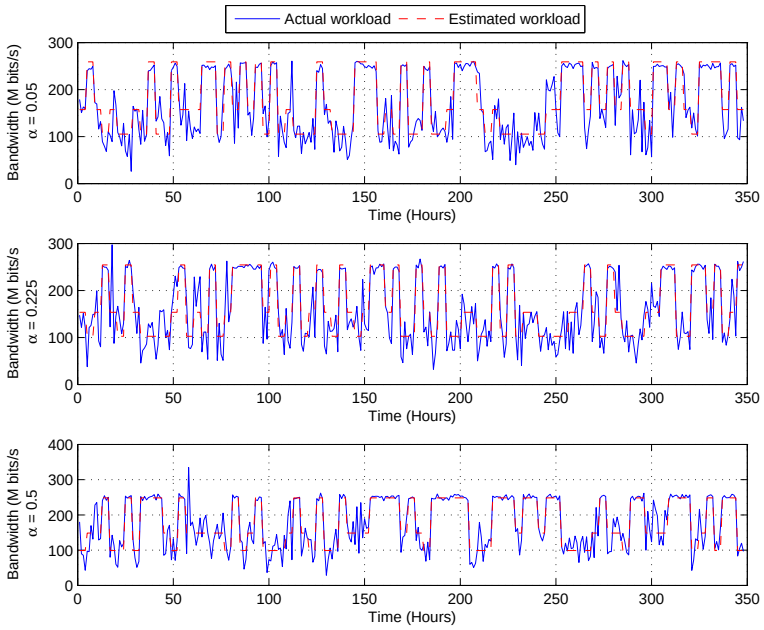

Figure 4.6: Inter-data centers workload fluctuation

Estimated Workload at Different Elasticity Levels

First, the impact of the elasticity level α is analyzed with respect to the amount of reserved network resources for the future EaaS demands. Fig. 4.6 depicts the actual workload fluctuation and the estimated fluctuation at three elasticity levels (i.e., $\alpha = 0.05, 0.225, 0.5$). It is obvious that at high elasticity levels with a low probability of unmet bandwidth demands (i.e., low α), the estimated workload and hence the reserved amount of network resources are always more than the actual workload, to prevent violating the EaaS level. However, at a lower elasticity level (i.e., $\alpha = 0.5$), the estimated workload can be close to the actual fluctuation but with a high probability that the actual demand exceeds the estimated one.

Furthermore, Fig. 4.7 presents the percentage of the met service elasticity demands at different elasticity levels. It can be seen that at high elasticity levels (i.e., $1 - \alpha \sim 1$),

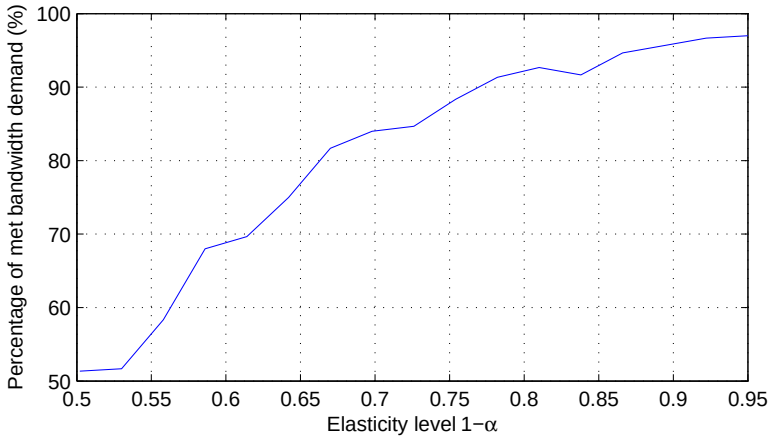

Figure 4.7: The percentage of the met bandwidth demands at different EaaS levels

the excess demand is always met. However, at medium elasticity levels (i.e., $1-\alpha \sim 0.5$), about 50 percent of the excess elastic demands are met with the reserved pool size.

Figures 4.8,4.9,4.10 and 4.11 present the actual traffic according to Equinix traces and the estimated fluctuation at a different number of Markov states (The value of α is set to 0.5). Four applications are selected from the Equinix traces; namely HTTP, HTTPs, UDP and RTMP. For simplicity, only the weekdays are considered and all the days are treated the same. In other words, the same set of states is visited each day. The period needed to learn Markov states was six weeks. It clear that when the number of states increases per day, the accuracy of the proposed EaaS model increases. In addition, if each day is treated separately by its own state, better estimation will be achieved.

Table 4.2 provides the values of the Mean Square Error (MSE) for each application type at different number of states per day.

Estimated Traffic Workload Accuracy

In order to evaluate the accuracy of our proposed approach in deciding the amount of

Table 4.2: The MSE values at different number of states per day.

Number of States	HTTP(Gbits/s)	HTTPS(Gbits/s)	UDP(Gbits/s)	RTMP(Gbits/s)
1 state/day	0.4444	0.0707	0.0379	0.0299
2 states/day	0.4323	0.0697	0.0297	0.0267
5 states/day	0.2651	0.0535	0.0248	0.0249
10 states/day	0.2172	0.0491	0.0231	0.0203

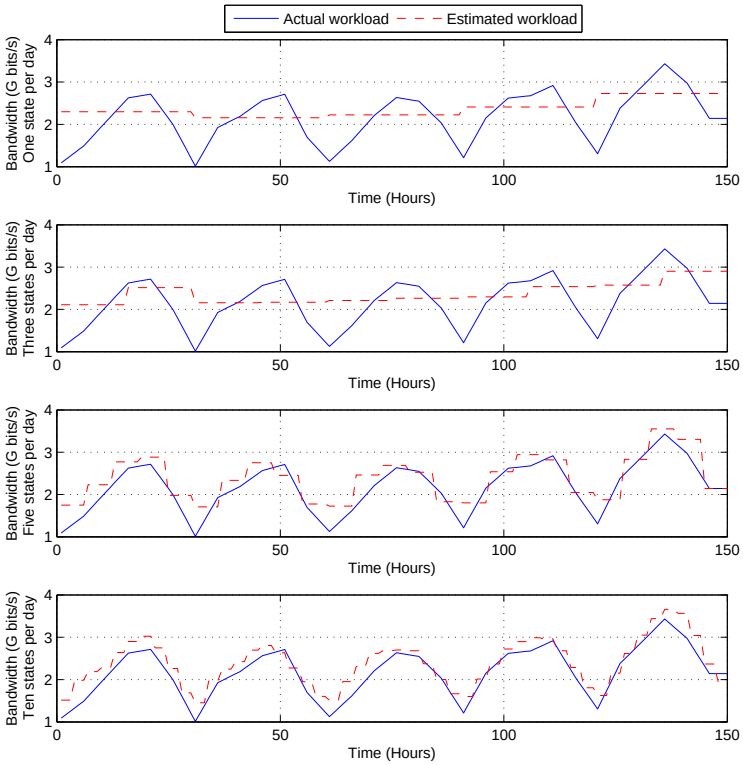

Figure 4.8: The actual workload vs the estimated (HTTP)

resources needed to be reserved for future demands elasticity, the MSE has been measured when a different number of applications, n, are being hosted. The MSE is defined as the expected sum of squared differences between the estimated and the corresponding actual fluctuations. As can be seen from Fig. 4.12, when the number of the applications sharing the same class per link increases, the MSE becomes closer to zero. This demonstrates

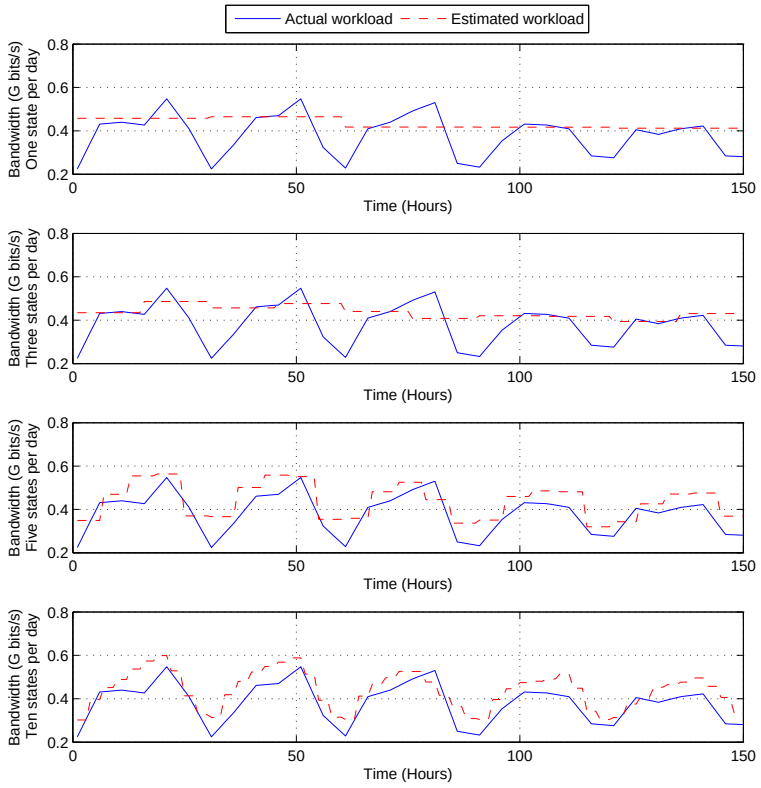

Figure 4.9: The actual workload vs the estimated (HTTPS)

the scalability of this model in cloud environments. It also can be seen that when the number of the cloud applications varies from 10 to 30, the MSE values increase from 0.14 to 0.04 which still are acceptable values.

To study the impact of the traffic probability distribution on this thesis' proposed approach, Fig. 4.13 illustrates the estimated and the actual workload fluctuation generated by three different distribution functions: Exponential, Poisson, and Uniform. It can be shown that regardless of the actual distribution function, the workload fluctuation approximates to a normal distribution.

Fig 4.14 plots the CSP accumulated revenue, which is defined as the profit from

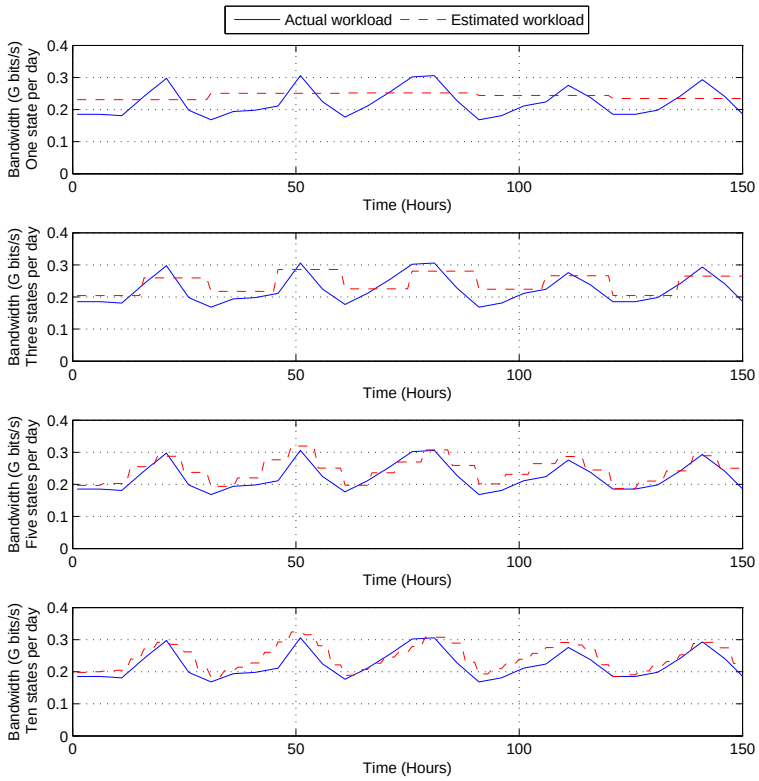

Figure 4.10: The actual workload vs the estimated (UDP)

selling the EaaS at four elasticity levels (i.e., $\alpha = 0.05, 0.1125, 0.225, 0.5$) minus the cost of reserving extra resources. The plot shows that the CSP accumulated revenue obtained from high elasticity class is higher because more bandwidth demands are met at a higher price.

4.8 EaaS Limitations

Assumptions and limitations are categorized into three categories; namely, assumptions for simplicity which can be easily relaxed, scheme limitations, and assumptions that need further work and can be considered as future work.

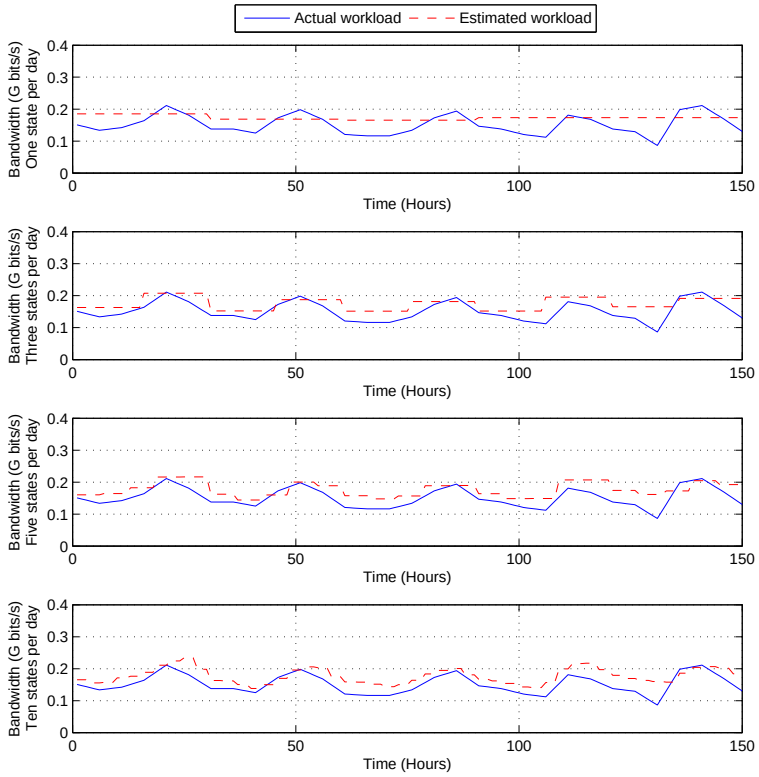

Figure 4.11: The actual workload vs the estimated (RTMP)

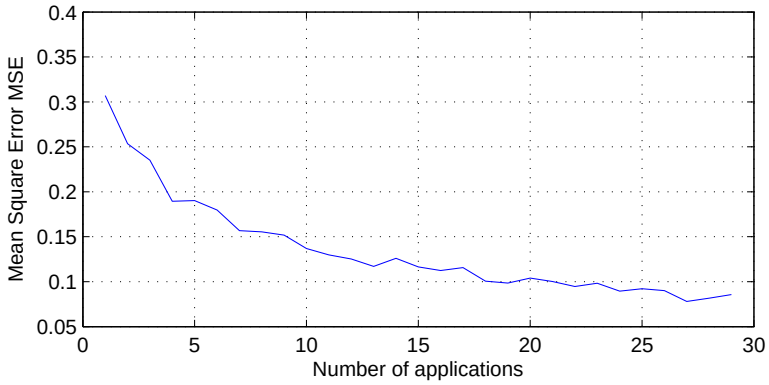

Figure 4.12: Elasticity pool reservation accuracy

Assumptions for simplicity:

- There is an assumption that the value of the EaaS level α can be determined

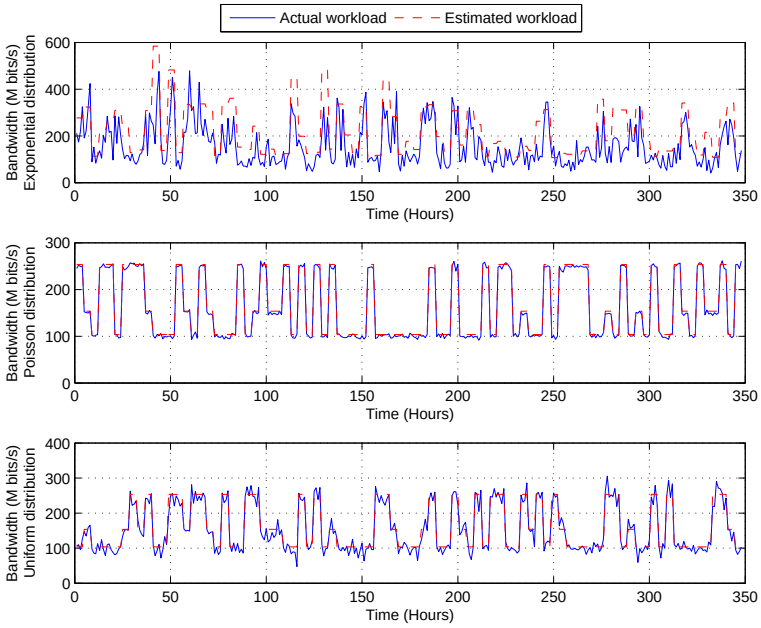

Figure 4.13: Inter-data centers workload fluctuation

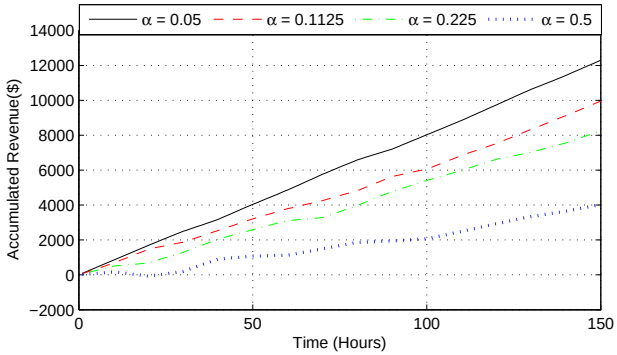

Figure 4.14: Accumulated CSP revenue

according to the bandwidth sensitivity of the offered application from historical observations and performance measurements.

Scheme limitations:

- The EaaS model targets the inter-data centers networks (i.e., may provide partially optimal allocation in case of intra-data center networks due to the special topologies of the latter).

- The EaaS model focuses on cloud applications owned by CCs that are characterized with relatively longer life-times and stationary behaviors (i.e., may provide partially optimal allocation in case of short-lived cloud applications).
- The bandwidth demands for each cluster are independent.
- Each cluster contains i.i.d bandwidth demands
- The number of applications per each class is large enough to implement the CLT (i.e., > 30).

Assumptions need further work:

- There is an assumption that the underlying physical network is own by a single infrastructure provider.
- There is an assumption that there is a knowledge of the entire underlying network (Centralized approach).

4.9 Summary

In this chapter, a novel approach has been proposed which enables the CSP to provide inter-data centers EaaS at differentiated levels for geographically distributed cloud applications. This EaaS takes place in order to guarantee the performance objectives of distributed cloud-based applications. The approach accurately estimates the amount of resources needed to be reserved to fulfill the future elastic inter-data centers workload fluctuation. Moreover, a corresponding EaaS dynamic pricing model was proposed that aims at maximizing the CSP expected long-term revenue. Experimental results have shown the accuracy of the model, and the increase in the CSPs profit.

Chapter 5

Substrate Network Re-optimization

In Chapter 4 the VN Elasticity-as-a-Service model and its features were presented. Chapter 5 is organized in the following way. Section 5.1 gives a general overview of some of the challenges and limitations of existing inter-data center bandwidth provisioning models. Sections 5.2 and 5.3 discuss the proposed SN resource re-optimization technique and the live migration schemes used to re-allocate the congested VN components. Simulation results and performance evaluation are discussed in Section 5.4. Finally, Section 5.6 concludes this chapter.

5.1 Overview

Network virtualization techniques aim at efficiently allocating the underlying SN resources to the hosted VNs. Unfortunately, over time, and due to the dynamic nature of current cloud applications and the frequent initiation and termination of VNs, the available and utilized SN resources become fragmented. This in turn, gradually degrades the performance of these techniques. As a result, CSPs aim at re-optimizing the inter-data center communication network in order to guarantee the performance of their hosted applications [17]. In this chapter, a novel proactive SN resource re-optimization technique is proposed that efficiently overcomes the fragmentation problem by performing appropriate re-arrangement, or house cleaning, for the available and utilized SN resources. To minimize the incurred computational overhead, the invocation of this technique is only triggered by certain events such as the departure of an expired VN. The contributions

of the proposed work are two fold; first, the development of an efficient technique for the selection and re-allocation of VN portions that are contributing to the fragmentation problem. The technique takes into consideration the trade-off between the benefit from increasing the SN utilization and the cost incurred by the VN migration. The second contribution involves a novel VN live migration techniques that significantly reduce the service interruption time during migration. Simulation experiments demonstrate the achieved gain in the SN resource utilization as well as in the VN acceptance ratio and the net revenue.

Network Virtualization allows the abstraction of a logical layer above the physical resource, enabling multiple virtual networks VNs to coexist over the same physical substrate network SN in a seamless manner. Infrastructure providers try to efficiently allocate resources to each VN request to ensure a high utilization of their SN resources. This in turn, increases the acceptance ratio of the VN requests and the providers revenue. The VN embedding process involves mapping the virtual nodes/links to the substrate nodes/paths [31], [32] in an efficient manner to ensure high and balanced SN utilization. However, the SN state changes over time due to the embedding of new VN requests, the expiration of some of the already hosted VNs, or due to changes in the SN resources [175]. Consequently, the substrate network becomes fragmented and inefficiently utilized and this results in an increase in the rejection ratio of future VN requests. This problem represents the main hurdle for various embedding techniques.

To overcome the aforementioned problem, a number of SN re-optimization techniques have been developed (e.g., [32], [49]). Nonetheless, these techniques have some limitations; a number of these techniques focuses on rerouting virtual links while keeping

virtual nodes unchanged [50]. Other techniques re-allocate a single virtual node one at a time and restrict the search for the new hosting substrate node to those that are within a one-hop distance of the old substrate node [32]. Restricting SN resources re-optimization to a single VN node migration greatly simplifies the migration process. Nonetheless, this restriction limits the enhancement in the utilization of the fragmented resources. This work supports the belief that simultaneous re-allocation of a number of virtual nodes may significantly improve substrate resource utilization much more than greedily re-allocating individual virtual nodes over several re-allocation iterations. It is also worth noting that, the majority of these existing solutions do not take into account the VN migration cost in terms of experienced service interruption time during migration. However, this interruption might result in the provider paying penalties due to service level agreement (SLA) violation [53].

To address these limitations, a new re-optimization technique is proposed that identifies congested VNs portions[1] which make some parts of the SN unavailable, then re-allocates these portions into other underutilized resources. The migration cost of re-allocating a VN component is reduced by using live node migration. A mechanism is proposed for the simultaneous migration of all virtual nodes within the same congested VN component. This results in a decrease in the service interruption time. Finally, the proposed work examines the trade-off between the gain achieved by dedicating a backbone network, or sharing the same substrate network for VN components migration.

[1]The terms VN portion and VN component are used interchangeably.

5.2 Proposed House Cleaning Scheme (HCS)

This chapter only focuses on the problem of re-optimizing the utilization of the SN resources. Hence, the assumption is made of the existence of an embedding technique that accepts the VN requests and performs the actual embedding (e.g., [51], [50], [32], [31], [106]). To minimize the incurred computational overhead and cost, the invocation of the SN House Cleaning Scheme (HCS) is only triggered by the expiration of a VN. The main idea is to detect bottlenecked substrate links which host portions of VNs, and try to find better embedding for these VN portions to release overutilized substrate resources by taking advantage of underutilized resources. In this way, fragmented resources will be reduced, hence, ensuring a high and balanced utilization of the SN resources.

To this end, a SN is modeled as a weighted undirected graph $G(N, E)$ with substrate nodes N and substrate edges E, $bw(e)$ denoting the bandwidth capacity of substrate edge $e \in E$. Similarly, the VN requests are modeled as a weighted undirected graph $G^V(N^V, E^V)$, with virtual nodes N^V and virtual edges E^V, $bw(e^v)$ denoting the bandwidth required by virtual edge $e^v \in E^V$. A virtual edge e^v can be mapped over a simple path of substrate links, denoted by $p(e^v)$.

The proposed optimization technique consists of four main steps as shown in Fig. 5.1. In the first step, SN links which are considered as barriers against future VN embedding are detected. In the second step, problematic VN components which contribute to SN links congestion are detected. The search for new candidate SN nodes to host the migrated VN components is performed in the third step. The last step makes the decision whether to accept the re-embedding or to reject it. Alg. 1 outlines the pseudo-code of the adopted steps.

Table 5.1: A summary of adopted notations (HCS).

Variable	Description
$G(N, E)$	the graph representing the SN.
N	the sets of substrate nodes.
E	the sets of substrate edges.
$bw(e)$	the bandwidth capacity of substrate edge $e \in E$.
$G^V(N^V, E^V)$	VN requests.
N^V	the sets of virtual nodes.
E^V	the sets of virtual links.
$\|E^V\|$	the number of virtual links per VN request.
$bw(e^v)$	the bandwidth required by virtual edge $e^v \in E^V$.
$p(e^v)$	the path where the virtual edge e^v is mapped.

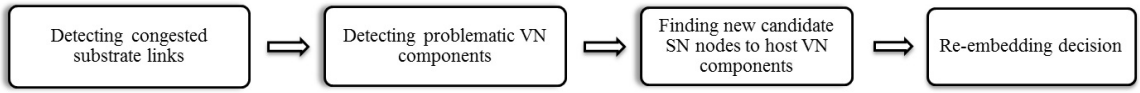

Figure 5.1: Steps of substrate network HCS

Table 5.1 provides a description of the adopted notations.

5.2.1 Detecting Congested Substrate Links

Congestion may reduce the efficiency in utilizing the SN resources over time, since the unavailability of some substrate links may result in selecting longer paths for embedding new VNs requests. Also, congested substrate links play an important role in rejecting VN requests. According to [32], about 99% of VN request rejections are attributed to the unavailability of sufficient substrate link bandwidth. In other words, the unavailability of one substrate link may cause a rejection of a complete VN request. A substrate link is considered to be congested if its stress, $S(e)$, is greater than a specific threshold β (Alg. 1: line 2), where

$$S(e) = \frac{\omega(e) \cdot \sum_{e^v \to e} bw(e^v)}{bw(e)} \tag{5.1}$$

Algorithm 1 Pseudo code of HCS re-optimization technique

Initialize PriorityQueue $PQ \leftarrow \phi$
$E_{congested} \leftarrow$ set of substrate links e: $S(e) > \beta$
$VN_{congested} \leftarrow$ set of congested VN
g $\leftarrow$ set of connected components from $VN_{congested}$
for each g_i in **g do**
    Calculate $V(g_i)$
    PQ.enqueue(g_i)
end for
$Iteration \leftarrow 0$
while (PQ is not empty & Iteration $< N_{opt}$) **do**
    $g_i \leftarrow PQ$.dequeue()
    $success \leftarrow$ re-allocation(g_i) into $G(N, E)$
    if success **then**
        updateStress(G)
        $migrationGain \leftarrow$ calculate $\mathcal{G}(g_i)$
        $migrationCost \leftarrow$ getMigrationCost(g_i)
        if migrationGain > migrationCost **then**
            commit re-allocation(g_i) into $G(N, E)$
        end if
    end if
    $Iteration \leftarrow Iteration + 1$
end while

Here $\omega(e)$ is the normalized link weight calculated offline for each substrate edge $e \in E$. $\omega(e)$ reflects the importance of e in the SN to prevent the congestion of critical SN portions. The value of ω is calculated based on substrate nodes *Centrality*, *Communicability*, and *Betweenness* [176]. The selection of the threshold value β is discussed in section V.

5.2.2 Detecting Problematic Virtual Networks Components

The next step in the proposed HCS is concerned with detecting problematic VNs portions by checking all already embedded VNs, and detecting the congested virtual links. This is carried out by identifying the intersection between the overloaded substrate links and VNs links (Alg. 1: line 3). The candidates for our re-optimization technique are the connected components of each VN which are mapped over congested substrate links. More precisely, for all VN, $G^V(N^V, E^V)$, the set $\mathbf{g} = \{g_1, g_2, \cdots, g_i, \cdots\}$ is constructed such that $g_i(N_i^V, E_i^V)$ is a subgraph of one of the VNs G^V where $N_i^V \subseteq N^V, E_i^V \subseteq E^V$. Nodes of each g_i are marked as either migrable or fixed. Migrable nodes are the nodes within g_i which are mapped over substrate nodes directly connected to the congested substrate links and need to be re-allocated. However, the fixed nodes are those which are connected to the migrable nodes within the VN by non-congested links and will not be migrated. The aim of marking some nodes in g_i as fixed, is to ensure that the same topology of the VN before and after the re-allocation is maintained. Either Breadth-first search, or Depth-first search can be used to extract the connected components g_i in linear time [177] (Alg. 1: line 4).

After detecting problematic VN components, candidates g_i are sorted according to a score $V(g_i)$ which reflects how much the congested g_i affects the overall utilization of the

substrate links. More precisely, $V(g_i)$ is the average ratio of the bw that g_i consumes on each SN link. $V(g_i)$ is scaled by the remaining life-time T_i of g_i to avoid re-embedding portions of VNs that are expiring soon (Alg. 1: lines 5-8). VN components with a high score $V(g_i)$ are potential candidates for the re-allocation. $V(g_i)$ can be calculated as follows:

$$V(g_i) = \frac{\sum_{e_i \in E_i^V} \sum_{e \in p(e_i)} \frac{bw(e_i)}{bw(e)}}{\sum_{e_i \in E_i^V} |p(e_i)|} \cdot T_i \qquad \forall g_i \in \mathbf{g} \tag{5.2}$$

where the operator $|.|$ returns the length of a path.

5.2.3 Finding New Candidate Hosting Substrate Network (SN) Nodes

According to the score $V(g_i)$, the first candidates g_i is selected (Alg. 1: line 11) and HCS tries to find new allocation which satisfies the re-optimization objective. The main objective during the re-embedding process is to maximize the migration gain $\mathcal{G}(g_i)$ which represents the gain obtained from reducing the average stress of the SN as well as that of reducing the amount of resources needed for the re-allocated components, after the re-allocation. Formally, this gain function can be calculated as follows:

$$\mathcal{G}(g_i) = \alpha_1 \frac{\sum_{e \in E'} S(e)}{\sum_{e \in E'} S'(e)} + \alpha_2 \frac{\sum_{e^v \in E_i^V} |p(e^v)| bw(e^v)}{\sum_{e^v \in E_i^V} |p'(e^v)| bw(e^v)} \tag{5.3}$$

The first term in (5.3) represents the gain obtained from reducing the average stress of the SN, which is calculated as the ratio between the stress of the links affected by the migration, before and after the re-allocation of g_i. The second term in (5.3) represents the gain obtained from reducing the amount of resources needed for the re-allocated components, which is calculated as the ratio between the total resources needed before and after the re-allocation of g_i. $S(e)$ and $S'(e)$ are the stress of e before and after the

re-allocation of g_i, respectively. Also, $p(e^v)$ and $p'(e^v)$ are the substrate paths hosting e^v, before and after the re-allocation respectively. E' is the subset of SN links affected by the migration, $E' \subseteq E$ where $E' = \bigcup_{e^v \in E_i^V} (p(e^v) \cup p'(e^v))$. The constants α_1 and α_2 are weighting factors dynamically tuned by the provider to reflect the priority of reducing the stress against reducing the amount of substrate resources needed. The re-optimization objective is to maximize $\mathcal{G}(g_i)$ for all VN connected components, i.e., it solves the problem,

$$\max \sum_{g_i \in \mathbf{g}} \mathcal{G}(g_i) \tag{5.4}$$

To find appropriate candidates to host g_i, the infrastructure provider can employ a modified version of the already employed embedding technique where g_i is mapped while keeping the same hosted nodes for g_i fixed virtual nodes.

5.2.4 Re-embedding decision

The decision to accept the suggested embedding by the previous step depends on the trade-off between the benefit from the re-embedding and the migration cost. This benefit can be calculated as the gain obtained from (5.3) multiplied by a constant that measures the monetary gain resulting from reducing the stress of a link, and reducing the resources needed for the re-allocated components (Alg. 1: lines 12-16). Section IV discusses this chapter's proposed live migration schemes and demonstrates how to calculate migration cost for the migration. The last step involves comparing the benefit against the migration cost (Alg. 1: line 17) and making the decision of whether to perform the actual re-allocation or not (Alg. 1: line 18). HCS iteratively repeats the re-embedding process for the successful g_i candidates for at most N_{opt} iterations (Alg. 1: line 10). To prevent overloading one section of the network during the re-allocation, the status of each substrate link is updated after VN embedding/departure, or the re-allocation of problematic

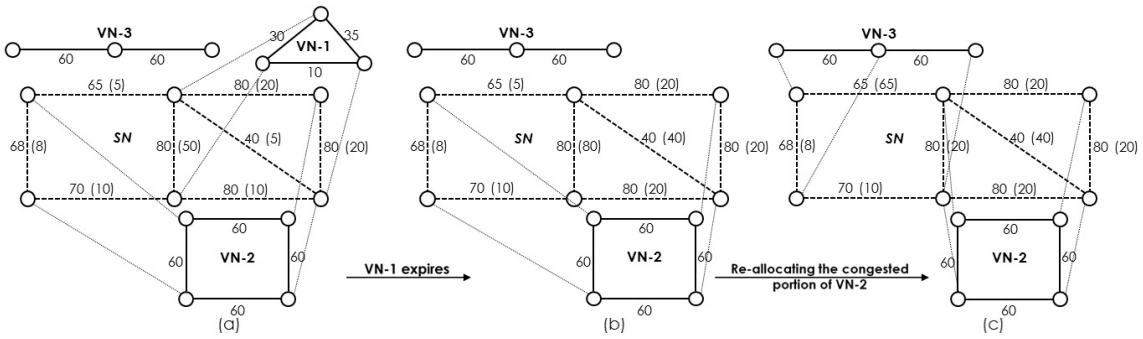

Figure 5.2: An example of substrate network re-optimization

VN portions. This avoids re-embedding VN portions over overutilized SN portions.

5.2.5 An Illustrative Example

Fig. 5.2, depicts an example scenario showing the gains obtained by the re-optimization technique used in this work. The numbers over the links represent each substrate link bandwidth capacity and the numbers between brackets represent their available bandwidths. Initially, VN-1 and VN-2 are embedded as shown in Fig. 5.2(a). VN request VN-3 cannot be embedded since there is no substrate link with 60 available bandwidth. Also, after VN-1 expires, VN-3 still cannot be embedded as shown in Fig. 5.2(b). But after re-allocating the congested portion of VN-2, the overall stress of the network decreases and the new embedding of VN-2 uses less substrate *bw* as shown in Fig. 5.2(c). This enables the acceptance of VN-3 request. The basic idea behind the detection of a connected VN component is shown by the example in Fig. 5.3, where a VN intersects with three congested substrate links and produces two connected VN components. The first component has two migrable nodes (A,B) and one fixed node (E) and the second component has three migrable nodes (D, E, H) and one fixed node (A).

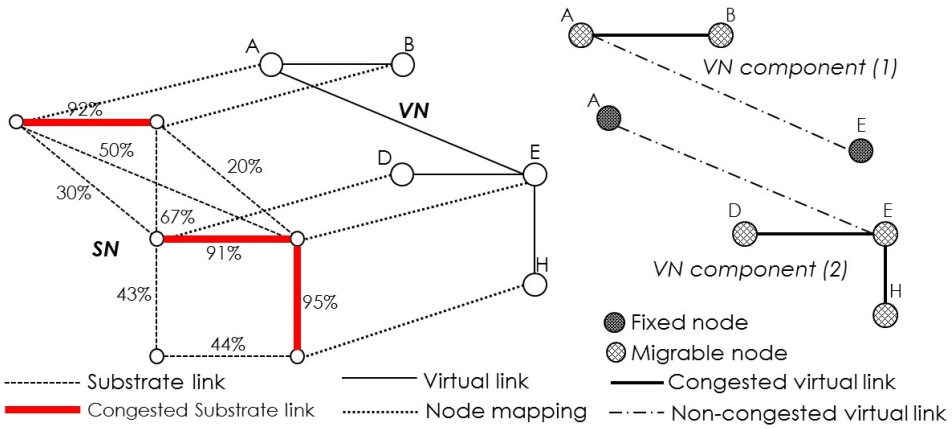

Figure 5.3: An example of Connected VN components

5.2.6 Complexity Analysis

Consider the situation where at time t, the SN is hosting n VNs. Updating the stress of the SN links is carried out during the embedding of a new VN request, after re-allocating a connected component, as well as after the expiration of a VN request. In general, it requires $\mathcal{O}(|E|)$ operations to detect congested substrate links. Also, the first step in the process of detecting problematic VN components can be accomplished in $\mathcal{O}(n|E^V|)$ by marking all virtual links mapped over congested substrate links during the updating of the substrate links status, where $|E^V|$ is the number of links per VN request, on average. The process of extracting a connected component g_i can be done in a linear time proportional to $\max(|N_i^V|, |E_i^V|)$ using either Breadth-first search, or Depth-first search [177]. The time complexity of sorting the set of candidates g_i according to $V(g_i)$ is $\mathcal{O}(m \cdot \log m)$, and where m is the number of connected VN components ($m \ll n^{\frac{|N^V|}{2}}$), and where $|N^V|$ is the number if nodes per VN request, on average. The last step of finding new candidates for hosting can be done in polynomial time (e.g., [31]). Hence, the dominating factor in the execution time of HCS is the time to re-embed the connected components. The scalability of the proposed HCS strongly depends on the last step of

finding new candidates for hosting the VN components. This can be done by employing a modified version of the already employed embedding technique. Several approaches in literature are proposed which study the scalability issue by proposing distributed approaches (e.g., [49]).

5.3 Live Migration of VN Portions

The proposed re-optimization process requires the migration of congested VN components from overloaded substrate resources to underutilized resources. The migration cost represents the penalty paid by the infrastructure provider due to violating the permissible SLA service interruption time. The traditional network migration process consists of freezing the running processes within the source node, then copying the virtual node data plane to the new physical node. The delay in completing the copying and restoring of node state and establishing new virtual links between the new virtual node and the old already connected virtual nodes would cause an unacceptable disruption time. However, current commercial routers which support virtualization have the control and data planes running in separate environments [178]. Therefore, their data planes run within the substrate node and their control planes are hosted within virtual environments. Reallocating congested VN components requires the migration of nodes marked as migrable for which the re-optimization technique finds better hosting substrate nodes. Employing a *Live migration* methodology for virtual nodes enables the migration of the control plane, without stopping the traffic through the old data plane. According to [178], at the new substrate node, the new control plane starts running and generating the new data plane while at the same time updates the old data plane at the initial substrate node. Then link migration begins immediately after the data plane is completely generated at

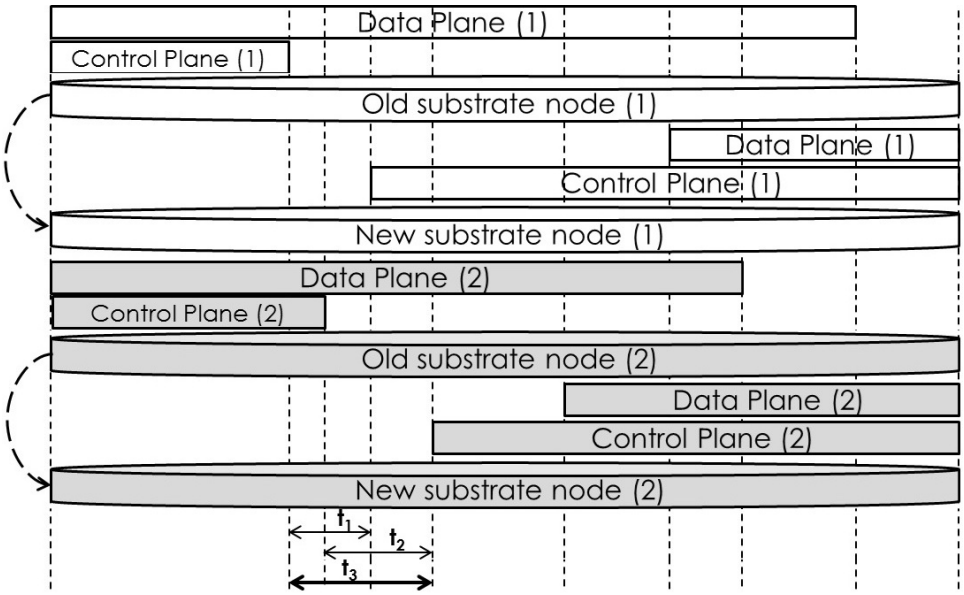

Figure 5.4: An example of two virtual nodes live migration

the new substrate node. During live migration of virtual nodes, the process of migrating the control plane is the only part which is subject to a short period of interruption time depending on the size of the virtual node. According to [178], the control plane downtime is between 0.924 and 1.008 seconds for core routers, and between 3.484 and 3.594 seconds for edge routers. Fig. 5.4 illustrates the process of migrating two virtual nodes, where the time periods t_1 and t_2 represent the interruption periods resulting from migration virtual node (1) and (2) respectively. The migration of more than one congested VN nodes can be performed simultaneously, since there are no dependencies between the migrated VN nodes. The time period t_3 represents the total service interruption time (time between the beginning of migrating the control plane of (1) and the time when the control plane of (2) is successfully migrated) resulting from the migration of two nodes simultaneously.

The interruption time due to control plane migration depends on the available bandwidth within the migration path, $path(v_i, v_i')$, between the old substrate node v_i and the

new substrate node v_i', when assuming a constant memory size for all virtual nodes. To perform control plane migration, two approaches are proposed: either using a separate backbone network dedicated to VN node migration or to simultaneously migrating g_i nodes using the same SN network. Migration is carried out by using a special backbone network, by separating the migration traffic from the network traffic of other VNs. In this case, the migration cost (the service interruption time) becomes dependent only on the length of the migration path $path(v_i, v_i')$. The migration cost can be calculated as follows:

$$MigCost_b(g_i) = p \cdot \delta \cdot \max_{v_i \in N_i^V} (shortest - path(v_i, v_i')) \tag{5.5}$$

where p is the expected monetary penalty per unit time of service interruption, and δ is the average delay per link in the backbone network link. The second proposed migration approach considers the simultaneous migration of g_i nodes over the same substrate network. Clearly, the service interruption time increases when the selected migration path contains bottlenecked links. Accordingly, if providing a backbone network for migration is not applicable or costly, This research's proposed optimization technique selects a migration path with the minimum delay (not necessarily the shortest path). The cost of simultaneously migrate g_i nodes can be calculated as:

$$MigCost_s(g_i) = p \cdot \max_{v_i \in N_i^V} (min - delay - path(v_i, v_i')). \tag{5.6}$$

5.4 Performance Evaluation

5.4.1 Simulation Environment

To study the efficiency of the proposed re-optimization technique, the discrete event simulator implemented in [31] was extended. The GT-ITM tool [179] is employed to

generate the SN and the VN requests. The substrate network size used during the experiments is set to 100 nodes, with the CPU capacities uniformly distributed between 50 and 100 computation cycles/unit time. Similarly, the bandwidth values of the links are uniformly distributed between 50 and 100 Mbps. The arrival events of the VN requests follow a *Poisson* distribution, with the arrival rate λ varied between 3-10 VNs per 100 time units, following similar setups in the existing work [31, 32, 50, 51]. The life time of the VN requests are exponentially distributed with an average of 600 time units. The number of virtual nodes are uniformly distributed (2-10 nodes per VN). VNs requests have bandwidth requirements following a uniform distribution between 0 and 70. Since, the proposed optimization technique focuses on enhancing the VN rejection ratio resulting from bandwidth shortage in the substrate links, VNs CPU requirements are relaxed so as to be uniformly distributed between 0 and 15. The experiments terminate after servicing 500 VN requests. The D-ViNE embedding algorithm [31] is used during the embedding process.

According to [32], the rejection rate is proportional to substrate links congestion, so with a small threshold values β, HCS avoids the overloading of substrate links. However, frequent triggering of HCS may degrade the system performance, so experimentally, β was chosen to be 0.7 which provides the minimum rejection rate without degrading the system performance.

5.4.2 Evaluation Results

Fig. 5.5 plots the accumulated net profit in three cases; (1) re-optimization using a backbone network, (2) re-optimization using the same SN, (3) without applying the re-optimization. As in [31], the net profit is calculated as the difference between the revenue

obtained from servicing the VNs and the cost of resources incurred by hosting these VNs. In this work, the migration costs calculated either by (5.5) or (5.6) are added to the cost. As indicated by Fig. 5.5, the net profit obtained when the SN is optimized using the proposed technique is always higher than that obtained without re-optimization. When using a backbone network for migration, the infrastructure provider dedicates a part of the SN resources to the backbone network (consisting of the shortest paths connecting all substrate nodes). Therefore, the bandwidth of some substrate links is reduced, which results in a reduction of the VN acceptance ratio and the total revenue. Experimentally, 10% of the bandwidth for each shortest path connecting all substrate nodes is allocated to the backbone network. On the other hand, executing simultaneous node migration introduces a small migration cost in both migration cases relative to the total revenue (using backbone network is better than shared SN). Fig. 5.6 shows the reduction in the rejection ratio achieved when the arrival rate of VN requests is varied between 3-10 VNs per 100 time units when applying the re-optimization technique HCS. This is in contrast to when no re-optimization is used and when the re-optimization process is restricted to the re-allocation of only one star moving candidate (virtual node with its attached hanging virtual links) per iteration, as proposed by [32]. It can be seen that applying our re-optimization technique significantly decreases the final rejection ratio compared to the case where no optimization is used. Moreover, it can be shown from the results that the proposed HCS technique decreases the rejection ratio by about 40% of that achieved by [32].

Fig. 5.7 depicts the time needed to perform the re-optimization technique. Note that the average execution time depends on the linear program solver (*glpk* used in D-ViNE simulator [31]) and the machine used for the experiments (Intel Xeon 3.47GHz processor,

Table 5.2: Average SN link stress.

Average SN links stress (%)		
Strategy	Average	Standard deviation
HCS Optimization	6.93%	3.07
Without Optimization	13.87%	8.20

4GB RAM). The two lines closest to the x-axis in Fig. 5.7 plot the average time needed to execute a single re-optimization operation for the proposed HCS technique and for the star moving technique [32]. Clearly, migrating a star moving candidate takes a much smaller amount of time than migrating a VN connected component. In order to compare the total execution time of the proposed scheme against the star moving candidate technique [32], a simple modification to [32] is performed. Instead of terminating the re-optimization process when the same maximum number of iterations is reached, the star moving algorithm runs until it achieves a rejection ratio as close as possible to that obtained by the proposed HCS technique. Fig. 5.7 shows a comparison between the total execution time taken to re-optimize the SN by both schemes. As shown, the overall execution time required by HCS is much smaller than that required by re-allocating star moving candidates since more iterations are required by the latter to reach similar results. Finally, table 5.2 compares the average SN link stress in the case of applying our re-optimization technique against that without re-optimization. It is clear that the average stress per SN link in the case of re-optimization is reduced to about half of that without re-optimization.

5.5 HCS Limitations

Assumptions and limitations are categorized into three categories; namely, assumptions for simplicity which can be easily relaxed, scheme limitations, and assumptions that need further work and can be considered as future work.

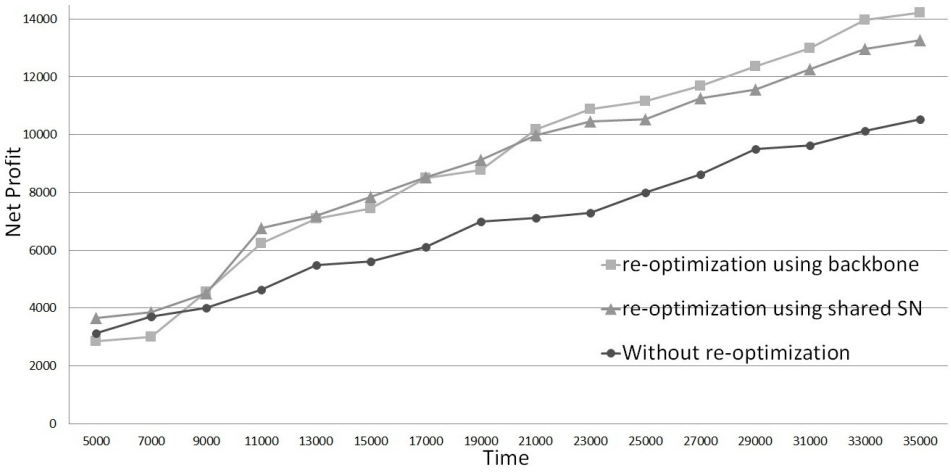

Figure 5.5: VN net profit vs. time

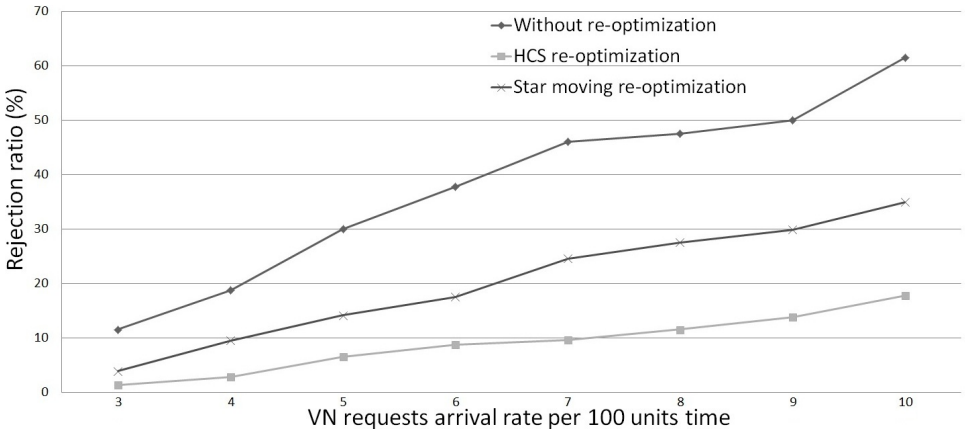

Figure 5.6: Rejection ratio vs. VN arrival rate λ

Assumptions for simplicity:

- This chapter only focuses on the problem of re-optimizing the utilization of the SN resources, assuming the existence of an embedding technique that accepts the VN requests and performs the actual embedding (e.g., [51], [50], [32], [31], [106]).
- To prevent frequent triggering of the proposed re-optimization technique, so experimentally, β was chosen to be 0.7 which provides the minimum rejection rate

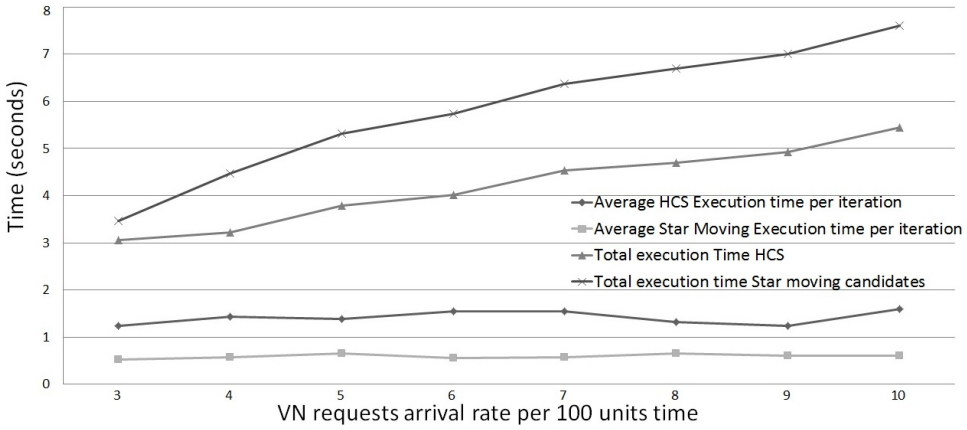

Figure 5.7: Excecution time

without degrading the system performance.

- There is an assumption that there is no location constraints which prevent the re-allocation of VN components. This assumption can be relaxed by marking these nodes as fixed nodes.

Scheme limitations:

- There is an assumption that the used routers support virtualization to perform the live migration. This is a valid assumption since current commercial routers support virtualization and have the control and data planes running in separate environments [178].

Assumptions need further work:

- There is an assumption that the underlying physical network is own by a single infrastructure provider.

- There is an assumption that there is a knowledge of the entire underlying network (Centralized approach).

5.6 Summary

In this chapter, a novel SN resource re-optimization technique is presented based on a house cleaning scheme executed only after the departure of VNs, by re-allocating congested VN portions, in order to maintain efficient resource utilization. Two live migration approaches were proposed to minimize the migration cost in terms of the service interruption time.

Chapter 6

Conclusion and Future Research Directions

Chapter 6 outlines the contributed research work and discusses plans and direction for possible future work. The chapter is organized as follows: Section 6.1 summarizes the conducted research contributions in the area of adaptive virtual network resource management. Section 6.2 sheds light on possible research for the future.

6.1 Conducted Research Work

The focus of this conducted research thus far, has been the development of an adaptive VN resource management model that guarantees the required cloud applications performance despite the dynamic traffic demands of heterogeneous VNs. The latter are characterized by different degrees of latency/bandwidth-sensitivity. The first step toward achieving that goal included a literature study of the VN resource management problem, and in particular, the dynamic provisioning of the SN resources and strategies needed to guarantee the required QoS. Several challenging issues were addressed: the necessity to provide a quality differentiated VN service which enables the dynamic provisioning of network resources while guaranteeing service quality. Also, consideration was given to the requirement of guaranteeing the availability of network resources in response to workload fluctuations. Finally, there is the need to overcome the SN fragmentation problem. Based on the identified limitations of current research work, a novel adaptive VN resource management model was proposed. In the following, a summary of the main

contributions of the current research work is given:

- *Differentiated Virtual-Network as-a Service* [67]

  A new Virtual Network as-a Service (VNaaS) model was proposed which enables the offering of differentiated service qualities in order to control the quality of the hosted VN and to overcome the unpredictability of the communication links. The model enables the VN users (i.e., the SP) to accurately express their dynamic network resources needs, constraints and their bandwidth/latency sensitivity. In order to enable the VN users to optimally decide the appropriate network resources, consisting of one or more service qualities according to the hosted application bandwidth/latency sensitivity and their budgets, a novel two stage-budgeting mechanism was developed and a closed formulation was derived to calculate the optimal bandwidth needs at each quality class. The proposed approach has several advantages in addition to that of reducing the complexity of the original problem formulation. This approach allows the VNusers to determine the impact of the demand of each virtual link on its total budget, and enables the VN users to accurately identify sources of revenue gains and losses in a straightforward manner. In addition, a novel Ramsey-based latency-aware network resource pricing mechanism was proposed to efficiently regulate the demands of incoming VNaaS requests.

- *Modeling and Pricing VN Elasticity-as-a-Service* [69]

  A novel VN resource management approach was proposed that dynamically re-sizes the reserved bandwidth pool in order to guarantee the availability of network resources, enabling the offering of the promised network EaaS offerings. Based on the elasticity level and the expected traffic workload fluctuation of each type of hosted application, the proposed approach enables the CSP to estimate the amount of

network resources needed to be reserved at the next time interval. This amount of network resources can be reserved either by setting apart a portion of the network resources, if available, or it can outsource network resources across multiple infrastructure providers [122]. In order to make an accurate prediction, in this thesis, the traffic workload was modeled using a Markov chain model, taking advantage of the temporal variability of workload fluctuations. In addition, a novel dynamic pricing mechanism for network EaaS offerings was proposed that can be employed by the CSP to maximize the expected long-term revenue, and to regulate the EaaS demands.

- *Substrate Network Re-optimization via Live Virtual Network Migration* [70]
  An efficient technique was proposed for the selection and re-allocation of VN portions that contribute to the fragmentation problem of the underlying physical network. This is done by simultaneously re-allocating a number of virtual nodes which significantly improve substrate resource utilization more than greedily re-allocating individual virtual nodes over several re-allocation iterations. In addition, a new live VN migration mechanism was proposed for the simultaneous migration of all virtual nodes within the same congested VN component resulting in a decrease in the service interruption time. Also, the proposed SN re-optimization technique examines the trade-off between the gain achieved by dedicating a backbone network, and the sharing of the same substrate network for VN components migration.

6.2 Planned and Future Research Work

The main focus for suggested future research work resulting from this thesis can be summarized as follows:

- In this thesis, a model is proposed to break down the network resources provisioning process into two aspects, namely the contracted bandwidth and the required elasticity level. Each aspect is investigated and studied well. However, there is a need to investigate how the VN users examine the trade-off between the amount of contracted bandwidth and the level of the EaaS, as the VN users aim at increasing their utility while minimizing the cost. Therefore, how to properly make resource provision plans is a challenging issue.

- Consider the relationship among the effective bandwidth allocated for each virtual link (i.e., internal routing matrix) while performing the EaaS reservation to ensure the consistency between the management operations on the VN and the allocated physical resources. This is because the variations in the traffic and/or the available resources on each virtual link must be dependent to ensures high resource utilization.

- In this thesis, there is an assumption that the underlying physical network is owned by a single infrastructure provider. However, in realty, this assumption needs to be relaxed. This requires considering multiple infrastructure providers while applying the proposed two-stage budgeting approach to distribute the pre-defined budget in an efficient way among the multiple infrastructure providers' resources. In addition, having multiple infrastructure providers may affect the pricing schemes and the calculation of the migration cost.

- The proposed substrate network re-optimization approach assumes the knowledge of the entire underlying network. This may be difficult to apply in a real system because knowing the information of the entire network can be challenging. To overcome this limitation, the SN re-optimization approach must be designed for dis-

tributed and parallel network re-optimization such that the network is partitioned into domains and the approach separately re-optimize each domain.

- Consider other objectives and constraints for the proposed adaptive VN resource management model, such as; operational costs reduction, energy efficiency, and green IT [27].

Overall, exploring the topic of this thesis was both stimulating and challenging. It is obvious this is a field of study with potential for further exploration in the future.

Bibliography

[1] R. Bless and C. Werle, "Control plane issues in the 4WARD network virtualization architecture," *Electronic Communications of the EASST*, vol. 17, 2009.

[2] J. Carapinha and J. Jiménez, "Network virtualization: A view from the bottom," in *Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures*, ser. VISA '09. New York, NY, USA: ACM, 2009, pp. 73–80.

[3] Y. Xu, M. Bailey, B. Noble, and F. Jahanian, "Small is better: Avoiding latency traps in virtualized data centers," in *Proceedings of the 4th Annual Symposium on Cloud Computing*, ser. SOCC '13. New York, NY, USA: ACM, 2013, pp. 7:1–7:16.

[4] Y. Chen, S. Jain, V. Adhikari, Z.-L. Zhang, and K. Xu, "A first look at inter-data center traffic characteristics via yahoo! datasets," in *INFOCOM, 2011 Proceedings IEEE*, April 2011, pp. 1620–1628.

[5] Equinix data centers. Accessed: 2015-01-05. [Online]. Available: www.equinix.com/

[6] Equinix inter-data centers traces. Accessed: 2015-01-05. [Online]. Available: www.caida.org/

[7] Q. Zhang, Q. Zhu, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Dynamic service placement in geographically distributed clouds," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 12, pp. 762–772, December 2013.

[8] P. Endo, A. de Almeida Palhares, N. Pereira, G. Goncalves, D. Sadok, J. Kelner, B. Melander, and J.-E. Mangs, "Resource allocation for distributed cloud: concepts and research challenges," *Network, IEEE*, vol. 25, no. 4, pp. 42–46, July 2011.

[9] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008. GCE '08*, Nov 2008, pp. 1–10.

[10] R. Carroll, S. Balasubramaniam, D. Botvich, and W. Donnelly, "Dynamic optimization solution for green service migration in data centres," in *Communications (ICC), 2011 IEEE International Conference on*, June 2011, pp. 1–6.

[11] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proceedings of the IEEE INFOCOM 2013, Turin, Italy*, 2013, pp. 854–862.

[12] K. Sripanidkulchai, S. Sahu, Y. Ruan, A. Shaikh, and C. Dorai, "Are clouds ready for large distributed applications?" *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 2, pp. 18–23, Apr. 2010.

[13] C. Chapman, W. Emmerich, F. Mrquez, S. Clayman, and A. Galis, "Software architecture definition for on-demand cloud provisioning," *Cluster Computing*, vol. 15, no. 2, pp. 79–100, 2012.

[14] M. Steiner, B. G. Gaglianello, V. Gurbani, V. Hilt, W. D. Roome, M. Scharf, and T. Voith, "Network-aware service placement in a distributed cloud environment," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 73–74, 2012.

[15] D. Espling, L. Larsson, W. Li, J. Tordsson, and E. Elmroth, "Modeling and placement of cloud services with internal structure," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[16] Q. Duan, Y. Yan, and A. Vasilakos, "A survey on service-oriented network virtualization toward convergence of networking and cloud computing," *Network and Service Management, IEEE Transactions on*, vol. 9, no. 4, pp. 373–392, December 2012.

[17] A. Mahimkar, A. Chiu, R. Doverspike, M. D. Feuer, P. Magill, E. Mavrogiorgis, J. Pastor, S. L. Woodward, and J. Yates, "Bandwidth on demand for inter-data center communication," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, ser. HotNets-X. New York, NY, USA: ACM, 2011, pp. 24:1–24:6.

[18] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined wan," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, Aug. 2013.

[19] J. Turner and D. Taylor, "Diversifying the internet," in *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, vol. 2, Dec 2005, pp. 6 pp.–760.

[20] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the internet impasse through virtualization," *Computer*, vol. 38, no. 4, pp. 34–41, April 2005.

[21] J. Brosemer and D. Enright, "Virtual networks: past, present and future," *Communications Magazine, IEEE*, vol. 30, no. 3, pp. 80–85, March 1992.

[22] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," *Comm. Mag.*, vol. 47, no. 7, pp. 20–26, Jul. 2009.

[23] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *Communications Magazine, IEEE*, vol. 51, no. 11, pp. 24–31, November 2013.

[24] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. Wright, "Performance analysis of high performance computing applications on the amazon web services cloud," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Nov 2010, pp. 159–168.

[25] G. Wang and T. Ng, "The impact of virtualization on network performance of amazon ec2 data center," in *INFOCOM, 2010 Proceedings IEEE*, March 2010, pp. 1–9.

[26] C. Wu, X. Lin, D. Yu, W. Xu, and L. Li, "End-to-end delay minimization for scientific workflows in clouds under budget constraint," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[27] A. Amokrane, M. Zhani, R. Langar, R. Boutaba, and G. Pujolle, "Greenhead: Virtual data center embedding across distributed infrastructures," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[28] A. G. Greenberg, J. R. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *Computer Communication Review*, vol. 39, no. 1, pp. 68–73, 2009.

[29] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, no. 5, pp. 862–876, Apr. 2010.

[30] G. Sun, V. Anand, H.-F. Yu, D. Liao, Y. Cai, and L. M. Li, "Adaptive provisioning for evolving virtual network request in cloud-based datacenters," in *Global Communications Conference (GLOBECOM), 2012 IEEE*, Dec 2012, pp. 1617–1622.

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

[32] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNR Algorithm: A Greedy Approach For Virtual Networks Reconfigurations," in *VNR Algorithm : A Greedy Approach For Virtual Networks Reconfigurations.* Houston, États-Unis: IEEE, 2011, pp. 6–11.

[33] J. He, R. Zhang-Shen, Y. Li, C.-Y. Lee, J. Rexford, and M. Chiang, "Davinci: Dynamically adaptive virtual networks for a customized internet," in *Proceedings of the 2008 ACM CoNEXT Conference*, ser. CoNEXT '08. New York, NY, USA: ACM, 2008, pp. 15:1–15:12.

[34] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 67–74, Jan. 2010.

[35] J. He, R. Zhang-shen, Y. Li, C. yen Lee, J. Rexford, and M. Chiang, "Davinci: Dynamically adaptive virtual networks for a customized internet," in *in Proc. CoNEXT*, 2008.

[36] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha, "Sharing the data center network," in *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, ser. NSDI'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 23–23.

[37] B. Nandi, A. Banerjee, S. Ghosh, and N. Banerjee, "Dynamic sla based elastic cloud service management: A saas perspective," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, May 2013, pp. 60–67.

[38] V. T. Lam, S. Radhakrishnan, R. Pan, A. Vahdat, and G. Varghese, "Netshare and stochastic netshare: predictable bandwidth allocation for data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 5–11, Jun. 2012.

[39] L. Zhang, C. Wu, Z. Li, C. Guo, M. Chen, and F. Lau, "Moving big data to the cloud: An online cost-minimizing approach," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 12, pp. 2710–2721, 2013.

[40] J. Rao, Y. Wei, J. Gong, and C.-Z. Xu, "Qos guarantees and service differentiation for dynamic cloud applications," *Network and Service Management, IEEE Transactions on*, vol. 10, no. 1, pp. 43–55, March 2013.

[41] K. Li, J. Wu, and A. Blaisse, "Elasticity-aware virtual machine placement for cloud datacenters," in *Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on*, Nov 2013, pp. 99–107.

[42] Z. Gong, X. Gu, and J. Wilkes, "Press: Predictive elastic resource scaling for cloud systems," in *Network and Service Management (CNSM), 2010 International Conference on*, Oct 2010, pp. 9–16.

[43] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh, "A cost-aware elasticity provisioning system for the cloud," in *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, June 2011, pp. 559–570.

[44] R.-H. Hwang, C.-N. Lee, Y.-R. Chen, and D.-J. Zhang-Jian, "Cost optimization of elasticity cloud resource subscription policy," *Services Computing, IEEE Transactions on*, vol. 7, no. 4, pp. 561–574, Oct 2014.

[45] N. R. Herbst, S. Kounev, and R. Reussner, "Elasticity in cloud computing: What it is, and what it is not," in *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13)*. San Jose, CA: USENIX, 2013, pp. 23–27. [Online]. Available: https://www.usenix.org/conference/icac13/technical-sessions/presentation/herbst

[46] G. Galante and L. de Bona, "A survey on cloud computing elasticity," in *Utility and Cloud Computing (UCC), 2012 IEEE Fifth International Conference on*, Nov 2012, pp. 263–270.

[47] T. Truong Huu, G. Koslovski, F. Anhalt, J. Montagnat, and P. Vicat-Blanc Primet, "Joint elastic cloud and virtual network framework for application performance-cost optimization," *J. Grid Comput.*, vol. 9, no. 1, pp. 27–47, Mar. 2011.

[48] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "Vnr algorithm: A greedy approach for virtual networks reconfigurations," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, 2011, pp. 1–6.

[49] C. Marquezan, L. Granville, G. Nunzi, and M. Brunner, "Distributed autonomic resource management for network virtualization," in *Network Operations and Management Symposium (NOMS), 2010 IEEE*, april 2010, pp. 463 –470.

[50] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 17–29, Mar. 2008.

[51] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, april 2006, pp. 1 –12.

[52] R. B. Nabeel Farooq Butt, N.M. Mosharaf Kabir Chowdhury, "Topology-awareness and re-optimization mechanism for virtual network embedding," *IFIP/TC6 NETWORKING*, May 2010.

[53] H. N. Van, F. Tran, and J.-M. Menaud, "SLA-aware virtual resource management for cloud infrastructures," in *Computer and Information Technology, 2009. CIT '09. Ninth IEEE International Conference on*, vol. 1, oct. 2009, pp. 357 –362.

[54] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: a data center network virtualization architecture with bandwidth guarantees," in *Proceedings of the 6th International COnference*, ser. Co-NEXT '10. New York, NY, USA: ACM, 2010, pp. 15:1–15:12.

[55] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gatekeeper: supporting bandwidth guarantees for multi-tenant datacenter networks," in *Proceedings of the 3rd conference on I/O virtualization*, ser. WIOV'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 6–6.

[56] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "Cloudnaas: a cloud networking platform for enterprise applications," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*, ser. SOCC '11. New York, NY, USA: ACM, 2011, pp. 8:1–8:13.

[57] L. Yu and H. Shen, "Bandwidth guarantee under demand uncertainty in multi-tenant clouds," in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*, June 2014, pp. 258–267.

[58] D. Niu, C. Feng, and B. Li, "Pricing cloud bandwidth reservations under demand uncertainty," *SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 1, pp. 151–162, Jun. 2012.

[59] D. Divakaran and M. Gurusamy, "Probabilistic-bandwidth guarantees with pricing in data-center networks," in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 3716–3720.

[60] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *Proceedings of the ACM SIGCOMM 2011 Conference*, ser. SIGCOMM '11. New York, NY, USA: ACM, 2011, pp. 242–253.

[61] J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhang, and J. Zhang, "Towards bandwidth guarantee in multi-tenancy cloud computing networks," in *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, Oct 2012, pp. 1–10.

[62] D. D. Mon and M. Gurusamy, "Towards flexible guarantees in clouds: Adaptive bandwidth allocation and pricing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, p. 1, 2014.

[63] S. Natarajan and T. Wolf, "Security issues in network virtualization for the future internet," in *Computing, Networking and Communications (ICNC), 2012 International Conference on*, Jan 2012, pp. 537–543.

[64] P. Chau and Y. Wang, "Security-awareness in network virtualization: A classified overview," in *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*, Oct 2014, pp. 545–550.

[65] Y.-L. Huang, B. Chen, M.-W. Shih, and C.-Y. Lai, "Security impacts of virtualization on a network testbed," in *Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference on*, June 2012, pp. 71–77.

[66] E. Abramov and O. Makarevich, "Deploying virtualization technology for verification of a network security," in *Application of Information and Communication Technologies (AICT), 2010 4th International Conference on*, Oct 2010, pp. 1–3.

[67] B. Wanis, N. Samaan, and A. Karmouch, "Efficient modeling and demand allocation for differentiated cloud virtual-network as-a service offerings," *Cloud Computing, IEEE Transactions on*, to appear.

[68] W. J. Baumol and D. F. Bradford, "Optimal departures from marginal cost pricing," *The American Economic Review*, vol. 60, no. 3, pp. 265–283, 1970.

[69] B. Wanis, N. Samaan, and A. Karmouch, "Modeling and pricing cloud service elasticity for geographically distributed applications," in *IM 2015 - Integrated Network Management, 2015 IFIP/IEEE International Symposium on*, Ottawa, Canada, May 2015.

[70] ——, "Substrate network house cleaning via live virtual network migration," in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 2256–2261.

[71] I. Toda, "DCNA higher level protocols," *Communications, IEEE Transactions on*, vol. 28, no. 4, pp. 575–584, Apr 1980.

[72] H. Shimizu, M. Oka, and R. Takaichi, "Evolution of an intelligent network," in *TENCON '92. "Technology Enabling Tomorrow : Computers, Communications and Automation towards the 21st Century.' 1992 IEEE Region 10 International Conference.*, Nov 1992, pp. 474–478 vol.1.

[73] J. Mizusawa, N. Shigematsu, and H. Itoh, "Virtual private network control system concept," in *Private Switching Systems and Networks, 1988., International Conference on*, Jun 1988, pp. 137–141.

[74] IEEE, "Standard for virtual bridged local area networks," *802.1Q*, 1988.

[75] A. Mason, *CCSP Self-Study: Cisco Secure Virtual Private Networks (CSVPN) (2Nd Edition)*. Pearson Higher Education, 2004.

[76] G. B. Paulsen and A. Walker, "Virtual private network system and method," 2000, uS Patent 6,055,575.

[77] P.-C. Wang, C.-T. Chan, and P.-Y. Lin, "Mac address translation for enabling scalable virtual private lan services," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 1, May 2007, pp. 870–875.

[78] "Approved draft ieee standard for local and metropolitan area networks– virtual bridged local area networks– amendment 4: Provider bridges (amendment to ieee 802.1q," *IEEE Std P802.1ad/D5.1*, 2005.

[79] E. H. Booth III, C. S. Lingafelt, P. T. Nguyen, L. Temoshenko, and X. Wang, "System and method to determine connectivity of a vpn secure tunnel," Nov. 9 2004, uS Patent 6,816,462.

[80] Z. Aqun, Y. Yuan, J. Yi, and G. Guanqun, "Research on tunneling techniques in virtual private networks," in *Communication Technology Proceedings, 2000. WCC-ICCT 2000. International Conference on*, vol. 1. IEEE, 2000, pp. 691–697.

[81] E. C. Rosen and Y. Rekhter, "Bgp/mpls ip virtual private networks (vpns)," 2006.

[82] R. Venkateswaran, "Virtual private networks," *Potentials, IEEE*, vol. 20, no. 1, pp. 11–15, Feb 2001.

[83] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys Tutorials, IEEE*, vol. 7, no. 2, pp. 72–93, Second 2005.

[84] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '02. New York, NY, USA: ACM, 2002, pp. 205–217.

[85] A. Belbekkouche, M. M. Hasan, and A. Karmouch, "Resource discovery and allocation in network virtualization," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 4, pp. 1114–1128, Fourth 2012.

[86] A. Alles *et al.*, "ATM internetworking," *Engineering InterOp, Las Vegas*, 1995.

[87] D. L. Tennenhouse and D. J. Wetherall, "Towards an active network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 5, pp. 81–94, Oct. 2007.

[88] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, "Network virtualization architecture: proposal and initial prototype," in *Proceedings of the 1st ACM workshop*

on Virtualized infrastructure systems and architectures, ser. VISA '09. New York, NY, USA: ACM, 2009, pp. 63–72.

[89] W. Szeto, Y. Iraqi, and R. Boutaba, "A multi-commodity flow based approach to virtual network resource allocation," in *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, vol. 6, Dec 2003, pp. 3004–3008 vol.6.

[90] B. Awerbuch and T. Leighton, "Improved approximation algorithms for the multi-commodity flow problem and local competitive routing in dynamic networks," in *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*. ACM, 1994, pp. 487–496.

[91] N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *INFOCOM 2009, IEEE*, April 2009, pp. 783–791.

[92] D. Eppstein, "Finding the k shortest paths," *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, Feb. 1999.

[93] J. He and J. Rexford, "Toward internet-wide multipath routing," *Network, IEEE*, vol. 22, no. 2, pp. 16–21, March 2008.

[94] J. Lu and J. Turner, "Efficient mapping of virtual networks onto a shared substrate," *Washington University in St. Louis, Tech. Rep*, 2006.

[95] J. Botero and X. Hesselbach, "The bottlenecked virtual network problem in bandwidth allocation for network virtualization," in *Communications, 2009. LATINCOM '09. IEEE Latin-American Conference on*, Sept 2009, pp. 1–5.

[96] Y. Wei, J. Wang, C. Wang, and X. Hu, "Bandwidth allocation in virtual network based on traffic prediction," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, Sept 2010, pp. 1–4.

[97] M. Fiedler, "On resource sharing and careful overbooking for network virtualisation," *Int. J. Commun. Netw. Distrib. Syst.*, vol. 6, no. 3, pp. 232–248, Apr. 2011.

[98] A. Razzaq and M. Rathore, "An approach towards resource efficient virtual network embedding," in *Evolving Internet (INTERNET), 2010 Second International Conference on*, Sept 2010, pp. 68–73.

[99] Y. Zhou, Y. Li, G. Sun, D. Jin, L. Su, and L. Zeng, "Game theory based bandwidth allocation scheme for network virtualization," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, Dec 2010, pp. 1–5.

[100] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures*, ser. VISA '09. New York, NY, USA: ACM, 2009, pp. 81–88.

[101] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, ser. STOC '71. New York, NY, USA: ACM, 1971, pp. 151–158.

[102] I. Houidi, W. Louati, W. Ben Ameur, and D. Zeghlache, "Virtual network provisioning across multiple substrate networks," *Computer Networks*, vol. 55, no. 4, pp. 1011–1023, 2011.

[103] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.

[104] J. F. Botero, X. Hesselbach, A. Fischer, and H. De Meer, "Optimal mapping of virtual networks with hidden hops," *Telecommunication Systems*, vol. 51, no. 4, pp. 273–282, 2012.

[105] H. Amarasinghe, A. Belbekkouche, and A. Karmouch, "Aggregation-based discovery for virtual network environments," in *IEEE International Conference on Communications, IEEE ICC'12, Ottawa, ON, Canada*, 2012.

[106] T. Ghazar and N. Samaan, "Hierarchical approach for efficient virtual network embedding based on exact subgraph matching," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, Dec. 2011, pp. 1 –6.

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

[108] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervello-Pastor, and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks," *IEEE Transactions on Computers*, vol. 99, no. PrePrints, p. 1, 2013.

[109] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "Adaptive-vne: A flexible resource allocation for virtual network embedding algorithm," in *Global Communications Conference (GLOBECOM), 2012 IEEE*, Dec 2012, pp. 2640–2646.

[110] B. Dab, I. Fajjari, N. Aitsaadi, and G. Pujolle, "Vnr-ga: Elastic virtual network reconfiguration algorithm based on genetic metaheuristic," in *Global Communications Conference (GLOBECOM), 2013 IEEE*, Dec 2013, pp. 2300–2306.

[111] T. Ghazar and N. Samaan, "Pricing utility-based virtual networks," *Network and Service Management, IEEE Transactions on*, vol. 10, no. 2, pp. 119–132, June 2013.

[112] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "Portland: A scalable fault-tolerant layer 2 data center network fabric," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 39–50, Aug. 2009.

[113] D. Li, J. Zhu, J. Wu, J. Guan, and Y. Zhang, "Guaranteeing heterogeneous bandwidth demand in multitenant data center networks," *Networking, IEEE/ACM Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[114] X. Luo, P. Zhang, T. Ye, Y. Jin, and J. Wang, "Dynamic bandwidth allocation amp; guarantee for virtualized networks in cloud," in *Information, Communications and Signal Processing (ICICS) 2013 9th International Conference on*, Dec 2013, pp. 1–5.

[115] J. Mudigonda, P. Yalagandula, J. Mogul, B. Stiekes, and Y. Pouffary, "Netlord: a scalable multi-tenant network architecture for virtualized datacenters," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 62–73, 2011.

[116] Amazon web services AWS. Accessed: 2014-06-12. [Online]. Available: http://aws.amazon.com/

[117] H. C. Lim, S. Babu, J. S. Chase, and S. S. Parekh, "Automated control in cloud computing: Challenges and opportunities," in *Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds*, ser. ACDC '09. New York, NY, USA: ACM, 2009, pp. 13–18.

[118] W. Dawoud, I. Takouna, and C. Meinel, "Elastic vm for cloud resources provisioning optimization," in *Advances in Computing and Communications*, ser. Communications in Computer and Information Science, A. Abraham, J. Lloret Mauri, J. Buford, J. Suzuki, and S. Thampi, Eds. Springer Berlin Heidelberg, 2011, vol. 190, pp. 431–445.

[119] N. Roy, A. Dubey, and A. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting," in *Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing*, ser. CLOUD '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 500–507.

[120] Scryer: Netflix. Accessed: 2013-10-06. [Online]. Available: http://techblog.netflix.com/2013/11/scryer-netflixs-predictive-auto-scaling.html

[121] A. Ghosh, S. Ha, E. Crabbe, and J. Rexford, "Scalable multi-class traffic management in data center backbone networks," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 12, pp. 2673–2684, 2013.

[122] G. Carella, T. Magedanz, K. Campowsky, and F. Schreiner, "Elasticity as a service for federated cloud testbeds," in *Communications Workshops (ICC), 2013 IEEE International Conference on*, June 2013, pp. 256–260.

[123] D. Niu, C. Feng, and B. Li, "A theory of cloud bandwidth pricing for video-on-demand providers," in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 711–719.

[124] D. Divakaran and M. Gurusamy, "Towards flexible guarantees in clouds: Adaptive bandwidth allocation and pricing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[125] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: Incorporating time-varying network reservations in data centers," in *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '12. New York, NY, USA: ACM, 2012, pp. 199–210.

[126] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, and M. Yasuda, "Less is more: Trading a little bandwidth for ultra-low latency in the data center," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 19–19.

[127] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," in *Proceedings of the ACM SIGCOMM 2011 Conference*, ser. SIGCOMM '11. New York, NY, USA: ACM, 2011, pp. 74–85.

[128] S. Secci and S. Murugesan, "Cloud networks: Enhancing performance and resiliency," *Computer*, vol. 47, no. 10, pp. 82–85, Oct 2014.

[129] Y. Hu, X. Long, J. Zhang, J. He, and L. Xia, "I/o scheduling model of virtual machine based on multi-core dynamic partitioning," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, ser. HPDC '10. New York, NY, USA: ACM, 2010, pp. 142–154.

[130] G. W. Dunlap, "Bulkscheduler development update," *Xen Summit Asia*, 2009.

[131] Y. Xu, Z. Musgrave, B. Noble, and M. Bailey, "Bobtail: Avoiding long tails in the cloud," in *Proceedings of the 10th USENIX Conference on Networked Systems De-*

sign and Implementation, ser. nsdi'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 329–342.

[132] P. Shivam, P. Wyckoff, and D. Panda, "Emp: Zero-copy os-bypass nic-driven gigabit ethernet message passing," in *Supercomputing, ACM/IEEE 2001 Conference*, Nov 2001, pp. 49–49.

[133] Y. Chen, R. Griffith, J. Liu, R. H. Katz, and A. D. Joseph, "Understanding tcp incast throughput collapse in datacenter networks," in *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, ser. WREN '09. New York, NY, USA: ACM, 2009, pp. 73–82.

[134] V. Jeyakumar, M. Alizadeh, D. Mazières, B. Prabhakar, C. Kim, and A. Greenberg, "EyeQ: Practical network performance isolation at the edge," in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, ser. nsdi'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 297–312.

[135] T. Szymanski, "Maximum flow minimum energy routing for exascale cloud computing systems," in *Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on*. IEEE, 2013, pp. 89–95.

[136] Q. Zhang, Q. Zhu, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Dynamic service placement in geographically distributed clouds," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 99, p. pp, 2013.

[137] Facebook statistics. Accessed: 2013-07-06. [Online]. Available: www.facebook.com/press/info.php?statistics

[138] D. Beaver, S. Kumar, H. C. Li, J. Sobel, and P. Vajgel, "Finding a needle in haystack: Facebook's photo storage," in *Proceedings of the 9th USENIX Conference*

on Operating Systems Design and Implementation, ser. OSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–8.

[139] S. Balasubramaniam, J. Mineraud, P. Perry, B. Jennings, L. Murphy, W. Donnelly, and D. Botvich, "Coordinating allocation of resources for multiple virtual IPTV providers to maximize revenue," *Broadcasting, IEEE Transactions on*, vol. 57, no. 4, pp. 826–839, Dec 2011.

[140] R. Clegg, S. Clayman, G. Pavlou, L. Mamatas, and A. Galis, "On the selection of management/monitoring nodes in highly dynamic networks," *Computers, IEEE Transactions on*, vol. 62, no. 6, pp. 1207–1220, June 2013.

[141] T. Szymanski, "Maximum flow minimum energy routing for exascale cloud computing systems," in *Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on*, Aug 2013, pp. 89–95.

[142] N. Grozev and R. Buyya, "Multi-cloud provisioning and load distribution for three-tier applications," *ACM Trans. Auton. Adapt. Syst.*, vol. 9, no. 3, pp. 13:1–13:21, Oct. 2014.

[143] C. Chapman, W. Emmerich, F. Marquez, S. Clayman, and A. Galis, "Elastic service definition in computational clouds," in *Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP*, April 2010, pp. 327–334.

[144] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, Aug. 2010.

[145] L. Wu, S. Garg, R. Buyya, C. Chen, and S. Versteeg, "Automated sla negotiation framework for cloud computing," in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, May 2013, pp. 235–244.

[146] F. Messina, G. Pappalardo, C. Santoro, D. Rosaci, and G. Sarne, "An agent based negotiation protocol for cloud service level agreements," in *WETICE Conference (WETICE), 2014 IEEE 23rd International*, June 2014, pp. 161–166.

[147] S. Krile, "Bandwidth optimization in sla negotiation process," in *Mobile Future, 2006 and the Symposium on Trends in Communications. SympoTIC '06. Joint IST Workshop on*, June 2006, pp. 36–39.

[148] S. Sharafeh, N. Moghadam, and M. Sanei, "An sla negotiation strategy for scheduling in grid," in *Electrical Engineering (ICEE), 2012 20th Iranian Conference on*, May 2012, pp. 794–799.

[149] H. Morshedlou and M. Meybodi, "Decreasing impact of sla violations:a proactive resource allocation approachfor cloud computing environments," *Cloud Computing, IEEE Transactions on*, vol. 2, no. 2, pp. 156–167, April 2014.

[150] H. Jin, X. Wang, S. Wu, S. Di, and X. Shi, "Towards optimized fine-grained pricing of iaas cloud platform," *IEEE Transactions on cloud Computing*, 2014.

[151] Y. Zhu and B. Li, "Overlay networks with linear capacity constraints," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 19, no. 2, pp. 159–173, 2008.

[152] Y. Zhu, B. Li, and K. Pu, "Dynamic multicast in overlay networks with linear capacity constraints," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, no. 7, pp. 925–939, 2009.

[153] L. Hu, K. Schwan, A. Gulati, J. Zhang, and C. Wang, "Net-cohort: Detecting and managing vm ensembles in virtualized data centers," in *Proceedings of the 9th international conference on Autonomic computing*. ACM, 2012, pp. 3–12.

[154] K. Konstanteli, T. Cucinotta, K. Psychas, and T. Varvarigou, "Elastic admission control for federated cloud services," *Cloud Computing, IEEE Transactions on*, vol. 2, no. 3, pp. 348–361, July 2014.

[155] F. Kelly, "Charging and rate control for elastic traffic," *European transactions on Telecommunications*, vol. 8, no. 1, pp. 33–37, 1997.

[156] M. Minoux, *Mathematical Programming: Theory and Algorithms*. U.K.: Wiley, 1986.

[157] W. M. Hanemann, *Consumer demand with several linear constraints: a global analysis*. Edward Elgar Publishing, 2006.

[158] H. Xu and B. Li, "Dynamic cloud pricing for revenue maximization," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[159] R. Roy, "La distribution du revenu entre les divers biens," *Econometrica, Journal of the Econometric Society*, pp. 205–225, 1947.

[160] Yahoo! research webscope program. Accessed: 2014-09-12. [Online]. Available: http://webscope.sandbox.yahoo.com/catalog.php?datatype=g

[161] M. Mihailescu and Y. M. Teo, "Dynamic resource pricing on federated clouds," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, May 2010, pp. 513–517.

[162] Y. Qu and N. Xiong, "Rfh: A resilient, fault-tolerant and high-efficient replication algorithm for distributed cloud storage," in *Parallel Processing (ICPP), 2012 41st International Conference on*, Sept 2012, pp. 520–529.

[163] Y. Wu, C. Wu, B. Li, L. Zhang, Z. Li, and F. C. M. Lau, "Scaling social media applications into geo-distributed clouds," in *INFOCOM, 2012 Proceedings IEEE*, March 2012, pp. 684–692.

[164] N. Bitar, S. Gringeri, and T. Xia, "Technologies and protocols for data center and cloud networking," *Communications Magazine, IEEE*, vol. 51, no. 9, pp. 24–31, September 2013.

[165] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 7, pp. 1366–1379, July 2013.

[166] S. Pacheco-Sanchez, G. Casale, B. Scotney, S. McClean, G. Parr, and S. Dawson, "Markovian workload characterization for qos prediction in the cloud," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, July 2011, pp. 147–154.

[167] W. D. Mulia, N. Sehgal, S. Sohoni, J. M. Acken, C. Lucas Stanberry, and D. J. Fritz, "Cloud workload characterization." *IETE Technical Review*, vol. 30, no. 5, 2013.

[168] D. Niu, Z. Liu, B. Li, and S. Zhao, "Demand forecast and performance prediction in peer-assisted on-demand streaming systems," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 421–425.

[169] A. Papoulis, "Probability, random variables, and stochastic processes," 1984.

[170] S. Nocedal, Jorge; Wright, *Numerical Optimization*. Springer Verlag, 2000.

[171] K. B. Petersen, M. S. Pedersen, J. Larsen, K. Strimmer, L. Christiansen, K. Hansen, L. He, L. Thibaut, M. Baro, S. Hattinger, V. Sima, and W. The, "The matrix cookbook," Tech. Rep., 2006.

[172] S. Shakkottai and R. Srikant, "Network optimization and control," *Found. Trends Netw.*, vol. 2, no. 3, pp. 271–379, Jan. 2007.

[173] D. P. Bertsekas, *Dynamic programming and optimal control*. Vol. 1. No. 2. Belmont, MA: Athena Scientific, 1995.

[174] TCPdump. Accessed: 2014-11-20. [Online]. Available: www.tcpdump.org

[175] Z. Cai, F. Liu, N. Xiao, Q. Liu, and Z. Wang, "Virtual network embedding for evolving networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, dec. 2010, pp. 1 –5.

[176] E. Estrada and D. J. Higham, "Network properties revealed through matrix functions," *SIAM Rev.*, vol. 52, no. 4, pp. 696–714, Nov. 2010.

[177] J. Hopcroft and R. Tarjan, "Algorithm 447: efficient algorithms for graph manipulation," *Commun. ACM*, vol. 16, no. 6, pp. 372–378, Jun. 1973.

[178] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe, and J. Rexford, "Virtual routers on the move: live router migration as a network-management primitive," in *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, ser. SIGCOMM '08. New York, NY, USA: ACM, 2008, pp. 231–242.

[179] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, vol. 2, mar 1996, pp. 594 –602 vol.2.