Resource Provisioning for Inter-Domain IoT Services in Edge-Cloud SDN-based Networks

by

Duong Tuan NGUYEN

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, FEBRUARY 3, 2022

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Mohamed Cheriet, Thesis supervisor
Department of Systems Engineering, École de Technologie Superieure

Mr. Kim-Khoa Nguyen, Thesis Co-Supervisor
Department of Electrical Engineering, École de Technologie Superieure

Mr. Lameiras Koerich Alessandro, Chair, Board of Examiners
Department of Software Engineering and IT, École de Technologie Superieure

Mr. Mohamed Faten Zhani, Member of the Jury
Department of Software Engineering and IT, École de Technologie Superieure

Mr. Roch Glitho, External Examiner
Institute of Information Systems Engineering, Concordia University

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

FOREWORD

This dissertation is submitted for the degree of Doctor of Philosophy at the University of Quebec, Ecole de Technologie Superieure (ETS). The research described herein was conducted under the supervision of Professor Mohamed Cheriet in the Department of Systems Engineering and Professor Kim-Khoa Nguyen in the Department of Electical Engineering, during the period from January 2016 to December 2021.

This work is to the best of my knowledge original, except where acknowledgements and references are made to previous work. Neither this, nor any substantially similar dissertation has been or is being submitted for any other degree, diploma or other qualification at any other university

Part of this work has been presented in the following publications:

1. D. T. Nguyen, K. K. Nguyen, S. Khazri and M. Cheriet, "Real-time optimized NFV architecture for internetworking WebRTC and IMS," 2016 17th International Telecommunications Network Strategy and Planning Symposium (Networks), Montreal, QC, Canada, 2016, pp. 81-88, doi: 10.1109/NETWKS.2016.7751157.
2. D. T. Nguyen, K. K. Nguyen and M. Cheriet, "Optimized IoT service orchestration," 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, Canada, 2017, pp. 1-6, doi: 10.1109/PIMRC.2017.8292756.
3. D. T. Nguyen, K. K. Nguyen and M. Cheriet, "NFV-Based Architecture for the Interworking Between WebRTC and IMS," in IEEE Transactions on Network and Service Management, vol. 15, no. 4, pp. 1363-1377, Dec. 2018, doi: 10.1109/TNSM.2018.2876697.
4. D. T. Nguyen, C. Pham, K. K. Nguyen and M. Cheriet, "Virtual Network Function Placement in IoT Network," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 2019, pp. 1166-1171, doi: 10.1109/IWCMC.2019.8766491.
5. D. T. Nguyen, C. Pham, K. K. Nguyen and M. Cheriet, "Placement and Chaining for Run-Time IoT Service Deployment in Edge-Cloud," in IEEE Transactions on Network and Service Management, vol. 17, no. 1, pp. 459-472, March 2020, doi: 10.1109/TNSM.2019.2948137.

6. D. T. Nguyen, C. Pham, K. K. Nguyen and M. Cheriet, "SACO: A Service Chain Aware SDN Controller-Switch Mapping Framework," 2019 15th International Conference on Network and Service Management (CNSM), Halifax, NS, Canada, 2019, pp. 1-8, doi: 10.23919/CNSM46954.2019.9012747.
7. D. T. Nguyen, C. Pham, K. K. Nguyen and M. Cheriet, "Jointly Optimized Resource Allocation for SDN Control and Forwarding Planes in Edge-Cloud SDN-based Networks," submitted to IEEE Internet of Things Journal (December, 2021).

ACKNOWLEDGEMENTS

I would like to thank all the people who are continuously supporting me with their valuable encouragements. I would never have been able to continue my work with same energy and motivation without the guidance of my professors and committee members, help from friends, and support from my parents and my family.

I would like to express my deepest gratitude to my supervisor Professor Mohamed CHERIET, for his excellent guidance, endless caring, patience, and providing me with an excellent atmosphere for doing research.

I am extremely grateful to Professor Kim Khoa NGUYEN, who let me experience the research exercise in the field with nice and practical discussions beyond the textbooks, patiently corrected my writing and drove my work path.

I would like to thank "Synchromedia lab" team members, who as good friends, were always willing to help and give their best suggestions. My work would not have been possible without their helps.

Special thanks to my parents Son Nguyen and Thuc Duong, my brother Nguyen Nguyen, and my entire family. They were always supporting me and encouraging me with their best wishes.

Last but not least, I would like to extend extra special thanks to my wife Kha Tran and my son An Justin for their patience and their sacrifices. They were always there cheering me up and stood by me through the good times and bad.

Duong Tuan NGUYEN

Provisionnement des ressources pour les services IdO inter-domaines dans les réseaux infonuagique-périphérique programmable

Duong Tuan NGUYEN

RÉSUMÉ

L'Internet des objets (IdO) se caractérisant par un grand nombre des objets connectés et une multitude d'applications intelligentes a été introduit pour améliorer la qualité de la vie humaine. L'hétérogénéité des technologies de contrôle et de communication des objets, la charge de travail fluctuante et l'exigence de faible latence des applications IdO imposent une grande pression sur les ressources de réseau que les solutions traditionnelles ne sont pas capable de gérer alors qu'il reste encore des obstacles pour les nouveaux paradigmes de gestion tels que SDN/NFV. En particulier, les contrôleurs SDN font face à des défis d'insuffisance des ressources pour prendre de décision de transfert pour plusieurs requêtes provenant simultanément des équipements de communication SDN servant des services IdO. De l'autre côté, dans l'architecture infonuagique-périphérique, il est important de définir une stratégie pour optimiser l'allocation des ressources virtuelles requises soit sur l'infonuagique à distance pour les tâches de calcul intensif, soit à proximité des utilisateurs finaux sur des équipements locaux pour réduire la latence de bout-en-bout. Dans la littérature, aucun des travaux antérieurs n'a pris en compte le problème con-joint du placement et du chaînage des ressources virtuelles pour les réseaux SDN d'infonuagique-périphérique. Ainsi, l'objectif de cette thèse est de concevoir une stratégie efficace pour le placement et le chaînage des ressources de calcul de réseau en même temps sur les plans de contrôle et de transfert SDN. Nous présentons un modèle de latence de bout-en-bout pour les services d'interfonctionnement déployés sur plusieurs domaines différents, tels que IMS et WebRTC, qui minimise le coût total de l'utilisation des ressources et des opérations de service tout en répondant aux exigences de QdS et QdE ainsi qu'en maintenant la stabilité du système en fonction des demandes dynamiques.

Afin d'atteindre cet objectif, trois problèmes clés doivent être étudiés dans notre cadre de travail comme les suivants: i) comment optimiser le placement et le chaînage des ressources VNF dans le réseau infonuagique-périphérique? ii) comment modéliser et mettre en œuvre une solution optimisée de placement et de chaînage des ressources dans un réseau hétérogène avec un grand nombre de nœuds opéré par SDN? iii) comment améliorer la QoS en termes de latence du service de bout-en-bout pour les services d'interfonctionnement multi-domaines selon la disponibilité des ressources physiques?

Pour le premier problème, nous modélisons le placement et le chaînage des VNF en tenant en compte le trafic agrégé des appareils IdO, la latence de bout-en-bout des chaînes de services, et les liens entre l'infonuagique, les passerelles IdO, et les nuages péréphériques. Pour résoudre le problème d'optimisation non-convexe formulé, nous concevons une solution basée sur l'approximation de Markov qui adopte des techniques multistart et batching (MBMAP) pour résoudre le problème d'optimisation combinatoire. Notre solution est exécutée de manière distribuée et par conséquent accélère le taux de convergence.

Pour résoudre le deuxième problème, nous présentons un modèle d'allocation de ressources pour les plans de contrôle et de transfert SDN et le formulons comme un problème d'optimisation conjointe. Nous adoptons le cadre d'optimisation des files d'attente de Lyapunov pour transformer ce problème d'optimisation à long terme en une série de problèmes en temps réel et utilisons la méthode de remontée de gradient exponentielle sur les problèmes transformés pour trouver une solution quasi-optimale. De plus, une architecture d'implémentation pour l'orchestration de contrôleurs de ressources hétérogènes est également conçue.

Enfin, pour résoudre le troisième problème, nous présentons une architecture d'interfonctionnement basée sur NFV permettant l'orchestration multi-domaines, e.g., les domaines IMS et WebRTC, et définissons les échanges de messages requis pour le chaînage de services. Notre modèle de latence du service de bout-en-bout représente l'objectif d'allocation des ressources en temps réel. Un algorithme en temps réel basé sur le cadre d'approximation de Markov est conçu pour allouer des ressources VNF avec un coût optimal et minimiser l'impact de la violation de QoS pendant la période de mise à l'échelle. Les résultats expérimentaux montrent que notre algorithme répond efficacement aux demandes de service fluctuantes avec un coût de service réduit de 19% tout en respectant la QdS.

Mots-clés: service IdO, SDN/NFV, calcul périphérique, placement et chaînage VNF, placement du contrôleur SDN, approximation de Markov, optimisation de la file d'attente Lyapunov, IMS, WebRTC

Resource Provisioning for Inter-Domain IoT Services in Edge-Cloud SDN-based Networks

Duong Tuan NGUYEN

ABSTRACT

The introduction of IoT networks with a large number of devices and diversity of smart IoT applications aims to improve the quality of human life. The heterogeneity of control and communication technologies from device to device, the fluctuating workload and low latency requirements of IoT applications pose a great pressure onto core network resource that traditional networks' solutions are insufficient to handle while there are still obstacles that must be overcome before novel paradigms like SDN/NFV, edge-cloud computing paradigm can advance to be widely deployed. In particular, SDN controllers will face resource scalable challenges to deal with numerous queries from SDN forwarding elements for appropriate forwarding actions for traffic flows of IoT services that they encounter. On the other hand, in edge-cloud architecture, it is important to define a strategy to optimally allocate the virtual resources either at remote clouds for computation-intensive tasks or close to end-users at edge cloud on equipment within their premises to reduce E2E service latency. No prior work has taken into account the problem of virtual resource placement and chaining for edge-cloud SDN-based networks. Thus, the goal of this dissertation is to design efficient computing and networking resource placement and chaining strategy over both SDN control and forwarding planes. We present an E2E latency model for interworking services deployed over multiple domains that minimizes the total cost of resource usage and service operations while meeting QoS and QoE requirements as well as maintaining the system's queue stability given dynamic service demand.

In order to meet this goal, three key problems are to be addressed in our framework and are summarized as follows: i) how to optimize VNF resource placement and chaining in edge-cloud networks? ii) how to model and optimize resource placement and chaining solution in a heterogeneous network with a large number of nodes operated by SDN? iii) how to improve the QoS in terms of E2E service latency for multi-domain interworking services according to the availability of physical resources?

Our first contribution is to model the VNF placement and chaining problem regarding aggregated input traffic from IoT devices. The E2E latency of service chains, the links connecting clouds, IoT gateways, and edges. To solve the formulated non-convex optimization problem, we design a Markov approximation-based solution that adopts multistart and batching techniques (MBMAP) to solve the combinatorial optimization problem. Our solution runs distributedly and consequently accelerates convergent rate.

To address the second problem, we present a comprehensive model of resource allocation for both SDN controlling and forwarding planes and formulate it as a joint optimization problem. We adopt the Lyapunov queueing optimization framework to transform this long-term optimization problem into a series of real-time problems and employ the exponentiated gradient ascent method

on the transformed problems to find a near-optimal solution. In addition, an implementation architecture for the orchestration of heterogeneous resource controllers is also designed.

Finally, to address the third problem, we present an NFV-based interworking architecture enabling multi-domain orchestration, i.e. IMS and WebRTC domains, and provide the message exchanges required for service chains. Our E2E service latency model represents the real-time resource allocation objective. A real-time algorithm based on the Markov approximation framework is designed to allocate VNF resources with optimal cost and minimize impact from QoS violation during scaling periods. Experimental results reveal that our algorithm effectively responds to fluctuating service demands with a service cost reduced by 19% with respect to QoS requirements.

Keywords: IoT service, SDN/NFV, edge computing, VNF placement and chaining, SDN controller placement, Markov approximation, Lyapunov queueing optimization, IMS, WebRTC

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ALGORITHMS

Page

LIST OF ABBREVIATIONS

IMS	IP Multimedia Subsystem
QoS	Quality of Service
RMARA	Markov Approximation-based Resource Allocation
WebRTC	Web Real-Time Communication
NFV	Network Function Virtualization
SDN	Software Defined Network
SLA	Service Level Agreement
WSF	Web Server Function
WRIG	WebRTC ↔ IMS Interworking Gateway
NFVI	NFV Infrastructure
NFVIaaS	NFV Infrastructure (NFVI)-as-a-Service
VNF	Virtual Network Function
ICE	Iteractive Connectivity Establishment
VM	Virtual Machine
E2E	End-to-End
E2ESO	End-to-End Service Orchestrator
SP	service provider
SP	service providers
mSF	micro-Service Function

IoT	Internet of Things
M2M	Machine-to-Machine
VNFP	VNF optimal placement and routing problem
VNE	virtual network embedding
IP	Integer Programming
ILP	Integer Linear Programming
MILP	Mixed integer linear programming
QoE	Quality of Experience
CSCF	Call Session Control Function
HSS	Home Subscriber Server
DPI	Deep Packet Inspection
STUN	Session Traversal Utilities for NAT
TURN	Traversal Using Relays around NAT
EGA	Exponentiated Gradient Algorithm
MGA	Mirror Gradient Algorithm
DC	Data center
EDC	Edge Data Center
CDC	Core Data Center

INTRODUCTION

Recent advent of next-generation wireless communication technologies (i.e., 5G and beyond) has witnessed the emergence of the Internet of Things (IoT) in terms of not only the number of deployed IoT devices but also the type of devices embedded with high-data-rate sensors (e.g., wearable devices, smartphones, vehicles). This new class of endpoints promises the potential of innovative services that are to promote quality of the human life and thus have high social and business impact like smart healthcare systems, virtual realities applications, connected and autonomous vehicles. These applications are common in experiencing workload fluctuations over time, and involving computing-intensive processing tasks across multiple domains for the potentially enormous amount of data generated by IoT objects. They are also time-sensitive in the sense that some require establishing a connection in a certain time (e.g., mission-critical services), some need guaranteed latency in milliseconds, some depend on bounded packet delay variations, and some must process and transport different types of data in milliseconds, seconds, or minutes. As a matter of the fact, the IoT paradigm has fostered the development of resource provisioning solutions to deliver a variety of IoT-based services.

For network infrastructure operators, a very large number of heterogeneous devices with a plethora of data traffic over multiple domains and real-time requirements of smart IoT services have placed a great pressures onto core network's resource. Due to the complexity of network control protocols and the lack of elastic resource scaling, traditional network technologies often struggle to handle this new demand. To this end, SDN/NFV comes as an interesting solution with several inherent capabilities such as network control programmability, scalability, on-demand resource allocation as well as facilitating service provisioning. SDN shifts the complex control logic to the centralized controller and provides an ability to control the heterogeneous IoT devices in a uniform manner. The separation of the control plane from the physical devices helps to quickly bring into effect instant changes of network conditions, which are indeed the key characteristics of IoT deployments. On the other hand, NFV leverages virtualization technology

to transform legacy hardware-based, dedicated middleboxes into single modules of software programmed to perform a particular VNF. To this end, NFV allows network functions to be modularized, managed independently and therefore facilitates the deployment of VNFs on general purpose servers as well as dynamically migrating VNFs from one place to any place of the network. In the context of IoT services, virtualized resource for service functions or SDN controllers/switches can be placed on demand and close to IoT terminals so as to reduce data communication delay, bandwidth demands on network links to remote data centers, especially support the mobility and multi-domain distributed applications. Having said that, SDN/NFV is the envisioned framework to meet the requirements of IoT services in an efficient, scalable, seamless and cost-effective manner.

0.1 Research Motivation and Challenges

Despite the benefits an IoT system has gained from SDN/NFV principles, there are still obstacles that must be overcome before SDN/NFV can advance to be widely deployed. In this section, we discuss the challenges faced by the adoption of SDN/NFV into IoT networks from resource provisioning perspective as well as the motivation for this thesis.

In this thesis, we consider an interworking system between IMS and WebRTC service domain with the adoption of SDN paradigm and NFV technology as shown in Fig. 0.1. IoT service requests from end-users are realized via traffic flows from IoT devices that are aggregated by IoT gateways, i.e. as IMS User Equipments (UEs) and dispatched to corresponding endpoints, i.e. as WebRTC IMS Clients (WICs) depending on IoT applications. In the SDN networks, IoT traffic flows are routed through multiple domains thanks to forwarding rules managed by SDN controllers and are processed by corresponding service functions. Such the functions are performed by VNF instances whose resources are placed on physical nodes according to VNF controllers. Note that it can happen for more than one VNF instance to reside at the same node as well as multiple nodes hosting instances of the same VNF.

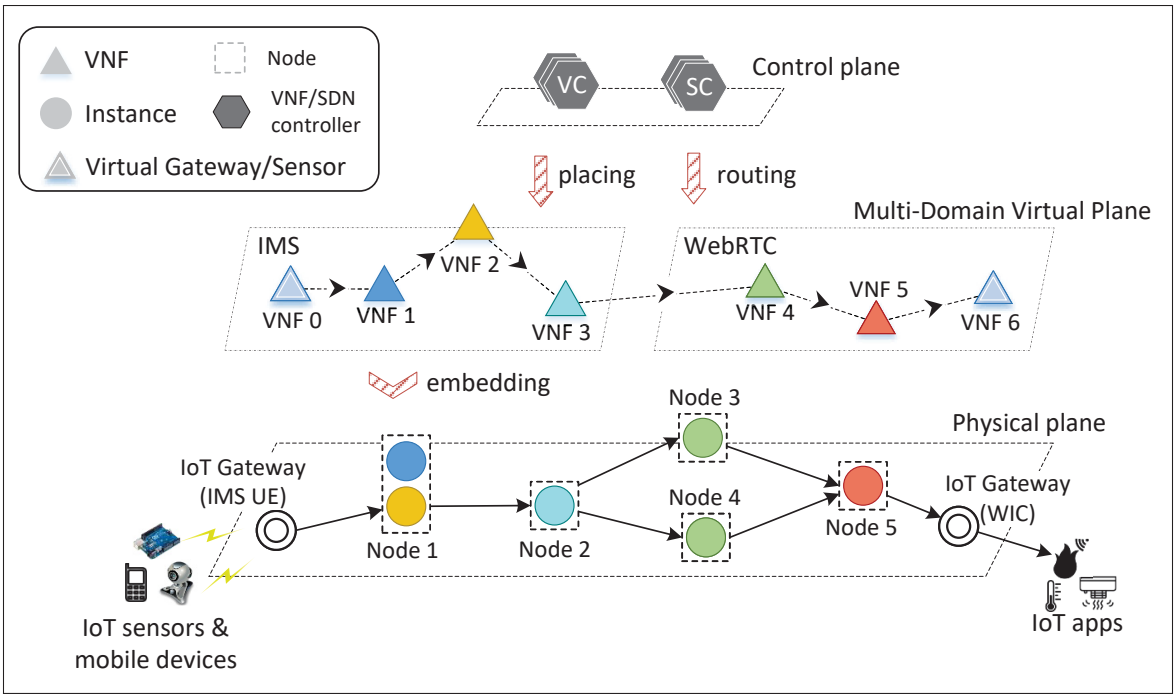


Figure 0.1 Resource placement and chaining for Multi-domain IoT Service Chain

0.1.1 Challenges for VNF Placement in Edge-Cloud Networks

Edge computing paradigm and NFV technologies enhance the QoS of users by moving computing resources into mobile access networks within the proximity of IoT nodes. The adoption of NFV can reduce the usage cost of network functions by moving the implementations of network functions from hardware to software that run in Virtual Machines (VMs). Furthermore, for SDN-based IoT networks, device virtualization plays an important role to provide a unified control mechanism for all heterogeneous IoT endpoints.

Along with the advantages, edge computing and virtualization technology introduce new challenges for a manageable and flexible resource allocation regarding the fluctuating service demand that may lead to over- or under-provisioning problems. In the over-provisioning problem, the allocated fog resources for certain IoT services are more than the actual demands of the IoT user to use computing resources. This issue results in high cost and low resource utilization if

the fog resources are idle or have light workloads. On the other hand, in the under-provisioning case, the allocated fog resources for particular IoT services are unable to service the actual IoT user demand for computing resources may cause delay violations, which result in the loss of IoT users. Therefore, it is crucial to dynamically place the appropriate number of VNFs needed to serve the submitted IoT workload while satisfying the QoS requirements.

Given the presence of service function chains and their complicated interaction, it is challenging to determine the right amount of VNFs' resources for processing the IoT services during its serving time. Scaling any service needs to take into account other chained services. Without a coordinated allocation strategy, one service could be overloaded by traffic of other services in the same chain and as a result, adversely affect entire QoS of the service chain. Note that the large number of available service functions involved in constructing IoT applications is also another factor that increases the optimization problem's size.

0.1.2 Challenges in Resource Allocation for Controlling and Forwarding Planes in SDN Networks

SDN-based technologies have major impact on IoT regarding the heterogeneity of IoT devices, the workload fluctuation over time and multi-domain distributed service components. At edge networks, SDN provides its operator with a global view of IoT network. As a result, it helps to not only obtain IoT service information thanks to the ability to control the heterogeneous devices in a uniform manner but also enforce appropriate forwarding rules to manage the network more efficiently. The network programmability plays an important role in supporting IoT service requirements since it allows resource at access networks to be dynamically allocated to ensure load balancing of the traffic as well as efficiently handles application-specific routing requests at core networks.

Given the importance of the controllers in obtaining these SDN advantages, the scalability at the control plane is apparently one of the initial concerns and has been actually considered by extensive literature. In edge IoT architectures, placing many controller instances in proximity to the edge nodes rather than solely at the remote cloud can solve the performance bottleneck of the remote controller and guarantee a timely manner for a diverse set of IoT services' requests with different requirements. This is not a trivial task given many metrics such as the presence of resource-constrained edge nodes, the workload at control plane, the overhead of inter-controllers control messages, all of which would turn SDN into a nightmare if not properly considered in controller placement.

In fact, a number of related efforts have been devoted to tackling the SDN scaling concerns from different angles of view: i) at data plane by delegating some decision making work to the forwarding devices Ros & Ruiz (2014), or improving more powerful processors to perform forwarding tasks Mogul & Congdon (2012), ii) at controller side with high performance controllers Voellmy, Wang, Yang, Ford & Hudak (2013b) with various well-known techniques, e.g. buffering, pipelining and parallelism. In general, most of the strategies try to address scalability issues of SDN by obtaining an optimal mapping between controllers and forwarding switches given service demand and available resource allocated at each plane.

0.1.3 Challenges with Multi-domain Interworking IoT Service

From the perspective of telecom service providers (SPs), a growing application of virtualization technology and SDN paradigm has been witnessed in many prior proposals, i.e. virtual IMS Carella *et al.* (2014), provided its important role in telecom core networks. However, it is a non-trivial task for the SPs to operate IoT services on top of telecom systems like IMS with regards the interacting complexity of its own entities, i.e., call session control function (vCSCF), home subscriber servers (vHSS). Note that integrating third-party services, i.e. Web Real-Time Communication (WebRTC) Holmberg, Hakansson & Eriksson (2015) from other domains into

an existing platform is a typical business practice for telcos to gain revenue. This increases the inherent system complexity with additional VNFs and creates new constraints to the optimization problem of provisioning resources, in terms of a variety of VNFs resource configurations (or templates) and inter- or intra-domain network latency. It also requires a well-defined exchanging mechanism of message flows among VNFs so that the workload at each VNF can be obtained for optimal resource allocation.

Having said that, a poorly designed resource allocation scheme is unable to efficiently handle services across multiple interworking NFV-based systems, i.e. IP Multimedia Subsystem (IMS), WebRTC. These services are characterized by not only a very large number of subscribers from each domain, but also have low E2E service latency requirements and likely experience fluctuating demand over time windows. Consequently, failing in providing services means not leveraging the benefits provided by novel technologies and network paradigms and breaks the original commitment to employ services like WebRTC, which was intended to help telco operators expand the number of potential endpoints for multimedia sessions Nguyen, Nguyen, Khazri & Cheriet (2016).

0.1.4 Research Motivation

In order to meet the IoT services' requirements, all the aforementioned challenges need to be jointly tackled together. In other words, it is important to have a solution of resource allocation with minimum costs given the requirement of low service delay, fluctuating service demand and the deployment of edge clouds from different operators. Having said that, the entire system cost in terms of the amount of allocated network/computing resource, and any cost incurred during the period of operating service, i.e. QoS violation fine, migration, need to be modeled as an objective function. E2E service latency needs to be determined to ensure that the resource allocation does not cause significant QoE degradation - a key success factor for enabling smart environment vision. In addition, the connectivity between SDN controllers and switches as

well as the placement of virtual network functions on edge clouds should be considered in the cost optimization problem. By doing so, not only the system scalability requiremnt can be fulfilled but also the system's queue stability - a critical metric for SDN controllers due to their vital role, could be ensured. Although many studies have been conducted toward a solution for each concern, namely E2E latency (cite), SDN controllers - switches mapping (cite), and VNF placement (cite), a few works investigate how to address all of them together in a holistic solution.

From the implementation perspective, the communication interfaces between global service orchestrator and resource controllers, i.e. network controllers, cloud platform controllers and device controllers need to be determined. A functional architecture should be designed to provide the implementation details of how the global orchestrator collects data from the controllers as well as performs necessary requests to allocate resource.

As a summary, the optimal resource allocation for multi-domain IoT services in Edge-Cloud SDN-based networks can be obtained by:

- leveraging the global knowledge of network topology to define the most efficient placement for virtual functions on either edge clouds close to end-users or remote clouds with powerful computing resource.
- efficiently mapping SDN controlling and forwarding planes, or controllers and switches, in other words, to maximize resource utilization at control plane while ensuring the stability of the entire system using queueing theory.
- modeling E2E service latency for different forms of service function chains that are deployed over multiple domains, to avoid significant QoE degradation.
- designing communication interfaces between service orchestrator and resource controllers, via which the optimization problem can be implemented and deployed.

0.2 Problem Statement and Research Questions

0.2.1 Problem Statement

The research problem addressed in this thesis is stated as follows: how to achieve an optimal resource allocation strategy for multi-domain interworking IoT services in edge-cloud SDN-based networks while ensuring the QoE and system's queue stability given the dynamic service demand and network topology in terms of link capacity and the availability of physical resource?

0.2.2 Research Questions

In order to address the above problem statement and drive our work methodology that will be discussed in Section 2, we further detail the problem statement into four research questions (RQ) as follows:

0.2.2.1 Research Question RQ-1

RQ-1: How to optimize VNF resource placement and chaining for run-time IoT service deployment provided the topology of edge-cloud networks?

It is challenging to provide cloud based IoT services with minimal resource usage cost. To obtain this goal, it requires an optimal strategy to place and chain VNFs in an edge-cloud network regarding delay sensitive and fluctuating service demand of IoT applications as well as the constantly changing network topology. For the former aspect, previous VNE approaches have mainly focused on the communication between VNFs at data center Li, Hong, Xue & j. Pei (2018a) in modeling service latency. However, modeling End-to-End service delay in VNF placement problem requires considering not only the connection between clouds where VNFs are allocated but also between clouds and underlying IoT networks where sensors and IoT gateways are deployed. Although many studies discuss the problem specific to IoT service chains, a few

works investigate the impact of VNFs' input data from IoT devices. In detail, the common solution is to assume continuous bandwidth demand which might be over-provisioned for sparse traffic aggregated at IoT gateways and can be improved by precisely estimating the service demand at each VNF instance. This is challenging given the complicated interaction of relevant IoT terminals and IoT gateways with VNFs at clouds, i.e. from VNFs to IoT gateway to collect sensing data, or in the reverse direction to activate certain device's functions. Mathematically, the presence of such IoT devices introduces additional elements which increase the inherent system complexity and thus create new constraints to VNE problem.

0.2.2.2 Research Question RQ-2

RQ-2: How to jointly optimize VNF resource placement for both SDN control and forwarding planes?

It is not trivial to address RQ-1 in the context of SDN network given the fact that the optimization problem of resource placement for both SDN control and forwarding planes is NP-hard. In addition, jointly including nodes from two planes might result in a complicated model due to the heterogeneity of network traffic and communication protocols employed at each plane. It is therefore important to have a careful design of the entire system model so that even a sub-optimal solution can be obtained in an acceptable time with a greedy heuristic method. Note that in order to adopt the proposed solution, it is necessary to orchestrate the operations of SDN controllers and resource controllers, i.e. to manage the exchanged messages, to synchronize the controller which is another challenge to answer RQ-1.

0.2.2.3 Research Question RQ-3

RQ-3: How to improve the QoS in terms of E2E service latency for multi-domain interworking services given the availability of physical resource?

In order to address RQ3, it requires to extend the model of E2E service latency, used in RQ1 and RQ2 if any, to large-scale services with components provisioned by multiple service providers. This is a challenging task due to i) the interacting complexity of service function components that are implemented as VNFs deployed across multiple domains ii) heterogeneous VNFs that require a well-defined messaging mechanism to obtain precise workload for an optimal resource allocation iii) the increasing complexity to the models in RQ1 and RQ2, that make it difficult to obtain the ultimate objective, i.e. QoS improvement.

0.3 Outline of the thesis

This introductory chapter defines some key concepts of the thesis, explains the general context and presents the problem statement. Chapter 1 reviews the prior work related to the scope of the research problems. In Chapter 2 the general methodology to address the various research questions of the problem and the objectives of the thesis are mentioned. The resulting thesis diagram for optimizing resource placement and chaining with respect to the QoS/QoE requirements is then described. Then, the three next chapters present the three articles published in response to the specific research questions. The three articles are outlined as follows:

- Chapter 3: NFV-based Architecture for the Interworking between WebRTC and IMS
- Chapter 4: Placement and Chaining for Run-time IoT Service Deployment in Edge-Cloud
- Chapter 5: Multi-Tiered Resource Placement in Edge-Cloud SDN-based Networks for IoT Services

Chapter 6 provides a critical discussion of some concepts of the thesis that highlights the strengths and weaknesses of the proposed methods. Finally, the general conclusion summarizes the work presented in this thesis and provides future horizons.

CHAPTER 1

LITERATURE REVIEW

This chapter presents a review of the state-of-the-art methods related to the resource allocation optimization problem for multi-domain IoT services in Edge-Cloud SDN-based networks. This chapter is divided into three sections that are in line with the challenges discussed in the introduction and faced by network infrastructure operators to obtain an efficient strategy to provision resource.

1.1 IoT Service and Resource Management

Integrating WebRTC into IMS domain enables the expansion of communication service to any endpoint with a WebRTC compliant Web browser GSM (2016). Such the multi-domain integration facilitates the deployment IoT solutions for both service providers and network infrastructure operators. The discussion is addressing various parts of IoT service and network resource management. In detail, E2E service latency needs to be modeled for the purpose of fulfilling real-time requirements of IoT applications. Regarding the fact that an IoT-related scenario typically involves service functions from different network domains, i.e. edge clouds, it is necessary to explore the state-of-the-art of adopting virtualization technology in a multi-domain system where IoT service chains are implemented. IMS $\rightleftarrows$ WebRTC is investigated as a specific use case of multi-domain multi-provider IoT applicatiosn.

1.1.1 Resource Orchestration for Multi-Domain NFV-based Network

In the multi-domain network environment with multiple infrastructure operators, resources are typically allocated in a heterogeneous manner in terms of the virtual computing, storage, and network resources, and the non-virtualized resources. Pham & Chu (2019) takes the competition among the operators into account while tackling the optimal resource orchestration problem from game theory perspective. They analyze a game on an NFV platform where an NSP can reserve resources for optimizing the utility in an individual manner. Also focusing on the presence of

multiple domains with radio resource virtualization, Liu & Han (2019) present a VirtualEdge system that enables the dynamic creation of virtual nodes (vNodes) on top of a physical cellular edge computing nodes to serve the traffic and workloads of a network slice. VirtualEdge introduces a realizable multidomain resource orchestration and virtualization that provides isolation among network slices. The authors design a new learning-assisted algorithm that allows the resource orchestrator to optimize the utilization of physical resources without knowing the utility functions of individual vNodes. Uniyal *et al.* (2020) present a hierarchical architecture (5GUKEx) to enable E2E orchestration with minimum overhead in complexity and performance while also allowing operators to maintain full control of their infrastructure. 5GUKEx allows operators to use their existing MANO systems for the single domain orchestration and build a multi-domain API based on standardized models exposed by service catalogs to coordinate the end-to-end service orchestration and interconnection. Aiming to accurately discover resource to perform data-intensive analytics, Xiang *et al.* (2019) propose a unified resource orchestration framework Unicorn for multi-domain and geo-distributed applications. However, its capability of discovering resources in IoT network to support the scenarios in which IoT devices are involved is not discussed.

One of the most discussed concepts for designing NFV-based 5G network is network slicing which enables a single physical network to be sliced into multiple virtual networks according to specific services and business goals Chatras, Tsang Kwong & Bihannic (2017). Given the benefits of network slicing to the operators, i.e. increasing service flexibility, enhancing network resource efficiency, it raises new challenges of resource management and orchestration, especially for multi-domain networks. Boubendir *et al.* (2018) introduce a Proof-of-Concept demonstrating the federation and orchestration of access and edge resources by a network operator to create and deploy customized network slices dynamically over a cross-domain network. Rosa, Santos & Rothenberg (2015) discussed three cases of multi-domain, distributed NFV, i.e. management and orchestration, bandwidth negotiation, and reliability. Regarding the problems of NFV orchestration and network slicing, a NFV-based architecture and its realization on the LTE network are proposed in Katsalis, Nikaein & Edmonds (2016). The authors in Li,

Zhou, Feng, Li & Xu (2018b) describe a horizontal-based multi-domain orchestration framework in SDN/NFV-enabled satellite and terrestrial networks. They model the service chain placement regarding the resource in terms of CPU and memory, which might be inconsistent with situations in which the SPs deploy their service on NFV infrastructure. Another effort of 5G Operating System (OS) towards 5G communications aiming to abstract away the complexities of underlying 5G infrastructure for an efficient and flexible service orchestration is discussed in Dräxler *et al.* (2018).

1.1.2 E2E Service Chain Latency Model

One key factor to the success of orchestrating service besides an efficient strategy of resource allocation is the low latency service function chaining. Regarding service latency modeling and optimization, an increasing number of models are being proposed to optimize service chain function orchestration. Sun *et al.* (2020) build a mathematical model of the service chain deployment problem and propose a BFS-based algorithm called SFCDO, that optimizes resource consumption and end-to-end delay of the deployment path. SFCDO takes care of the health of network nodes with an optimal selection strategy for the node with the lowest current load rate, distributes the load evenly on all nodes, and achieves load balancing. Close to our approach from system view, that is to consider the bi-directional communication between edges and core cloud, Santos, Wauters, Volckaert & De Turck (2020) introduce an SFC controller to optimize the placement of service chains in Fog environments, specifically tailored for smart city use cases. The controller is implemented as an extension to the scheduling features available in container orchestration platform, enabling the allocation of container-based SFCs while optimizing resource provisioning and reducing the E2E latency. Regarding various execution sequences of service functions, Mouradian *et al.* (2019) also aim to seek an efficient placement strategy to map the service components onto the infrastructure nodes including the mobility of fog nodes, a phenomenon that may happen in real systems. The authors use the random waypoint mobility model for fog nodes to calculate the expected makespan and application execution cost.

To minimize the weighted function of the makespan and cost, an ILP formulation is presented and can be solved to find sub-optimal placements using Tabu Search-based algorithm.

Using Markov approximation method, Wang, Lan, Zhang, Hu & Chen (2015) demonstrate how the combinatorial problem of optimizing the dynamic function composition for network service chains can be addressed in a distributed manner. Bhamare *et al.* (2017) discuss optimization model to reduce the overall End-to-End (E2E) latency by reducing the inter-cloud traffic *w.r.t* multiple VNF instances across multiple clouds. In the study by Gupta *et al.* (2018), different deployment strategies for service chaining are explored in a so-called Network-enabled Cloud and develop a model for chaining VNFs with minimal resource consumption while fulfilling service requirements. Another work of Sun, Li, Liao & Chang (2018) also tackles the resource allocation problem of orchestrating service chains by adopting full mesh aggregation method. Similarly, Riera *et al.* (2016) and Zhang, Wang, Kim, Palacharla & Ikeuchi (2016) respectively employ a dynamic approach and a vertex-centric distributed algorithm to address the issue of optimally allocating networking and IT resources for VNF hosting.

1.1.3 IMS ↔ WebRTC Use Case: Resource Allocation of NFV-based System

Conventional interworking architecture between WebRTC and IMS has motivated existing works. In Amirante, Castaldi, Miniero & Romano (2014), a modular-based gateway architecture Janus is designed as a bridge between legacy IMS protocols and WebRTC. Similarly, an IMS integrated WebRTC prototype is used in Cruz & Barraca (2015) to evaluate the performance in terms of call throughput and mouth-to-ear delay. Reference 3GPP (2018) is another effort of 3GPP to re-architecture the IMS platform to enable access by WebRTC-based clients. From the architectural perspective, what makes our work distinguish from 3GPP (2018) is that we consider the use case in which both WebRTC and the interworking SP are third-party services on top of the IMS system. The realization of the proposed architecture as well as the message flows are therefore done without any change to the functionalities of conventional IMS entities. While a lot of progress has been made to address the heterogeneous characteristic between two domains, these works do not take into account the adoption of NFV paradigm which allows

VNFs to be deployed on different shared physical and virtual resources in order to guarantee scalability and performance requirements. Such the NFV adoption sets our work apart from other related works.

With the widely used IMS platform, adapting the NFV architecture ETSI ISG on Network Functions Virtualization (2013) to an IMS-based system has been investigated in many studies. One of the methods presented in Carella *et al.* (2014) is to deploy service components as corresponding virtual units. The approach is preferable due to its compatibility with IMS or WebRTC specification and the simple implementation to leverage the advantage of NFV. In Lu, Pan, Lei, Liao & Jin (2013), the authors propose an algorithm to address the problem of dynamic resource allocation. However, they only consider resource aspects in terms of CPU and memory and do not clarify how well their algorithm can handle changes in service demand. Duan et al. Duan, Wu, Le, Liu & Peng (2017) propose an NFV management system to deploy IMS-related service chains based on predicted workload and per-instance processing capacity. Another auto-scaling scheme (VLB-CAC) at the VNF level for optimal allocation of IMS server' resources to admitted calls is introduced in Montazerolghaem, Yaghmaee, Leon-Garcia, Naghibzadeh & Tashtarian (2016). VLB-CAC is responsible for finding the optimal call acceptance rate for each SIP server by solving an optimization problem that prevents overload. Despite being designed based on the NFV architecture proposed by ETSI ETSI ISG on Network Functions Virtualization (2013), it is challenging to efficiently employ these works to the WebRTC $\leftrightarrow$ IMS interworking service without respecting the requirement of deploying multi-domain and multi-provider services. Inspired by the use cases presented in ETSI ETSI ISG on Network Functions Virtualization (2018), a distributed NFV-based architecture facilitating the deployment of such the services is proposed.

1.1.4 Discussion

Significant efforts have been actually devoted to the investigation of an NFV-based architecture for multi-domain service orchestration Dräxler *et al.* (2018); Katsalis *et al.* (2016); Rosa *et al.* (2015); Li *et al.* (2018b); ETSI ISG on Network Functions Virtualization (2013). While

the proposed solutions, in general, enable the deployment of service chains in cross-domain infrastructure, they are not detailed enough to completely model the interworking scenarios between services from different domains. Specifically, there is no explicit explanation of how an interworking service provider communicates with others to set up cross-domain service chains. The abstraction of domain resource is also described in a generic way and consequently makes it hard to apply into a specific situation, i.e. model the performance of WebRTC $\leftrightarrow$ IMS interworking system.

In general, our work features requirements that are not taken into account in previous studies. We focus on the elasticity of resource allocation in response to service demand changes and propose a real-time Markov approximation-based algorithm that minimizes total service cost concerning delay and resource constraints. Moreover, for service across multiple NFV-based systems like IMS and WebRTC and their components with different requirements, both the heterogeneous VNF configurations and a distributed algorithm should be considered.

1.2 VNF Resource Allocation for IoT Services

In this section, we review the different VNF resource allocation approaches proposed in the most recent and relevant works, and how the VNF placement and routing problem can be applied where fog and cloud resources for IoT services are available.

1.2.1 VNF Placement

One of the important decisions that resource orchestrators have to make when adopting virtualization technology is to find the optimal placement of VNFs on underlying physical resources. With a great attention of research community, this problem has been tackled from different aspects. For instance, it is formulated in different models, i.e. ILP Xu *et al.* (2018); Qi, Shen & Wang (2019), Binary ILP Liu, Li, Zhang, Su & Jin (2017); Liu, Pei, Hong & Li (2019), mixed ILP Hawilo, Jammal & Shami (2019); Tang, Zhou & Chen (2019), or mixed integer quadratically constraints programming Mehraghdam, Keller & Karl (2014), etc., which

are NP-hard in general. Hence it is challenging to obtain the optimal solution in polynomial time, especially in large-scale network settings. From the view of optimization target, the objective can be to minimize the physical machines on which VNFs are allocated Zhou *et al.* (2017); Li, Hong, Xue & Pei (2019), the total resource consumption Tang *et al.* (2019) or the total service delay Agarwal, Malandrino, Chiasserini & De (2019) etc. In this section, we review different VNF placement approaches proposed in the literature and how the problem is applied given the availability of heterogeneous resources in multi-domain environment.

Regarding cloud and edge computing enabled networks, Li *et al.* (2019) studied the VNF placement problem with the objective of minimizing used physical machines and the resource consumptions in nodes and links. The authors make a complete formulation of the problem mathematically, which is modeled as an ILP, and propose an efficient polynomial time heuristic to solve for an optimal placement of VNFs as the problem scale is large. Taleb, Bagaa & Ksentini (2015) propose the solutions for the problem of VNF embedding for virtual 5G network infrastructure while dealing with the mobility features and service usage behavioral patterns of mobile users. The solutions address two conflicting objectives, which are the insurance of Quality of Experience (QoE) via the placement of VNFs of data anchor gateways closer to end-users and the avoidance of the relocation of mobility anchor gateways via placing their corresponding VNFs far enough from users.

Apart from user mobility and E2E latency, the constantly changing network dynamics as a well-known characteristic of IoT network is addressed by Cziva, Anagnostopoulos & Pezaros (2018). A mechanism to dynamically re-schedule the optimal placement based on temporal network-wide latency fluctuations using optimal stopping theory is explained to achieve latency-optimal allocation of VNFs in next-generation Edge networks. Another IoT service specific is the presence of micro-data centers, known as edge cloud, whose locations significantly affect the requirement of ultra-short latency and have been investigated in Laghrissi, Taleb, Bagaa & Flinck (2017). The work also introduced an advanced predictive VNF placement strategy, which is an enhancement of the classic predictive VNF placement algorithm. A more sophisticated

VNF placement strategy is proposed to improve the performance of network slice planner tool considering other factors such as incidents, signal disturbance and dysfunction of equipments.

The study in Gong, Jiang, Wang & Zhu (2016) addresses the virtual network embedding (VNE) problem regarding the constraints related to the location of substrate nodes. In order to achieve the integrated node and link mapping, the authors discuss compatibility graph-based algorithms in which each node represents a candidate substrate path for a virtual link, and each link indicates the compatible relation between its two endpoints. Cao, Yang & Zhu (2018) adopts network topology information including node location to solve the problem. To deal with the issue of inefficient resource utilization of substrate network in the long run, the proposed algorithm adopts a novel node-ranking approach to rank all substrate and virtual nodes before embedding each given VN. In general, all of these prior works mainly focus on VNF location optimization for services between end-user and corresponding VNFs, not service function chain.

Another work for placing service chains across multiple clouds in Gupta *et al.* (2017) adopts machine learning technique for a predictive model combined with random cloud selections. Tackling the issue of deploying network services across multiple Points of Presence (PoPs), the framework in Riera *et al.* (2016) optimize various metrics, i.e. cost of placing VNFs to PoPs, overall delay, and total resource link usage. Bhamare *et al.* (2017) propose an analytical model for the placement of service function chains in multi-cloud environments. They consider inter-cloud traffic *w.r.t* the fact that inter-cloud links are more likely to be congested and more expensive compared to the links within a single datacenter. Further, comprehensive surveys on VNF chain placement are available for interested readers Bhamare, Jain, Samaka & Erbad (2016); Herrera & Botero (2016); Laghrissi & Taleb (2019); Chen, Yin, Wang, Shi & Yao (2020).

1.2.2 VNF Routing

Given that the VNFs are already placed in the network, some studies have concentrated on finding a path going through the required VNFs so as to satisfy constraints on delay, capacity, etc.

For example, Yang, Li, Trajanovski & Fu (2020) formally defines the traffic routing problem in stochastic NFV networks in which VNFs are assumed to be placed in a pre-defined order as SFC. Unlike most existing work in which network resources are considered to be deterministic, the authors deal with stochastic network factors caused by, e.g., inaccurate data, expired exchanged information, insufficient estimation of the network. A tunable heuristic algorithm is proposed to find the multi-constraint path for each adjacent VNF pair. The work by Gao & Rouskas (2020) is another effort addressing the routing problem of service chain in an online scenario. Particularly, in this work, service requests are unknown in advance in the sense that they might arrive at arbitrary moments and must be placed onto the network without prior knowledge of future requests. The major objective is to route the service chain and jointly minimize the maximum network congestion as well as the number of hops from the source to the destination.

In Ye, Zhuang, Li & Rao (2019), an analytical model to evaluate E2E packet delay for multiple traffic flows traversing a common embedded VNF chain is presented. The authors establish a tandem queueing model to describe packets of each flow passing through an NFV node and its outgoing link. In this work, the tradeoff between high performance and fair resource allocation for each traffic flow is maximized by employing an allocation strategy among traffic flows sharing resources at each NFV node. Note that as resources can only be placed onto servers located in networked data centers, the traffic of service function chains has significant impact on network load balancing. To mitigate this problem, resources should be placed within a smaller domain of the network to avoid the so-called distance-to-data center problem Carpio, Dhahri & Jukan (2017). This motivated the authors to study the problem of replicating various service components, and especially how they can be chained to load balance the network.

Recently, the graph layering method has proven the efficiency in finding a path through service function chains implemented as VNFs and has been adopted in many studies Van Bemten, Guck, Vizarreta, Machuca & Kellerer (2018); Sallam, Gupta, Li & Ji (2018); Ghaznavi, Shahriar, Kamali, Ahmed & Boutaba (2017); Dwaraki & Wolf (2016); Sen, Choudhuri & Basu (2020). The idea is to consider the network as a graph and replicate it into different layers in which only corresponding VNFs for service function chains are inter-connected. The service requests can be

optimally routed through every layer from the top to the bottom. For instance, Sen *et al.* (2020) propose a dynamic programming based algorithm for optimally mapping service chains to physical network nodes based on the layered graph regarding the presence of network structural dependency, e.g. the dependency of successful communication on a node. Van Bemten *et al.* (2018) presents a fast and close to an optimal algorithm for finding the constrained shortest path vising an ordered set of specified network nodes via the layered graph. By constructing a layered graph with a small size, the pruning algorithm proposed by Sallam *et al.* (2018) enables to minimize the computational complexity to solve the SFC-constrained shortest path problem. Other works also relying on layered graph technique include Ghaznavi *et al.* (2017) with a local search heuristic to find the path between two layers with the objective of minimizing the network resource cost, Dwaraki & Wolf (2016) that uses conventional shortest-path algorithms e.g., Dijkstra to calculate the path between the source and destination nodes.

1.2.3 VNF Jointly Placement and Routing

Regarding the limitation of solving VNF placement and routing separately, some studies have extended the problem to take the routing into account with the VNF placement decision Kuo, Liou, Lin & Tsai (2016); Bari *et al.* (2016); Tajiki, Salsano, Chiaraviglio, Shojafar & Akbari (2019); Farkiani, Bakhshi & MirHassani (2019); Sun *et al.* (2019); Jang, Suh, Pack & Dán (2017). In particular, three primary phases are i) to place the VNFs on physical nodes, ii) provision them to service requests from users, and then iii) route the traffic through these VNFs in the order defined via service function chains. Any decision can be made with regard to the constraints of capacity together with latency requirements, optimizing physical machines as well as energy consumption. For instance, in Kuo *et al.* (2016) study the joint VNF placement and path selection problem considering the relation between the link and server usage. The authors explain how the proper link and resource allocated for each VNF in the service chain, i.e. either via additional server or available resources of existing servers, are adapted according to network conditions and service requirements. They then propose a chain deployment algorithm to follow the guidance of the link and server usage. Another work in Allybokus, Perrot, Leguay,

Maggi & Gourdin formulate the problem as an ILP that allows one to find the best feasible paths and virtual function placement for a set of services with respect to a total cost including the routing and VNF operation cost. The authors take into account the (total or partial) order constraints for Service Function Chains of each service and other constraints such as end-to-end latency, anti-affinity rules between network functions on the same physical node and resource limitations in terms of network and processing capacities. In addition to optimizing the VNF placement and routing, the authors in Tajiki *et al.* (2019) present a novel resource allocation architecture that enables energy-aware SFC for SDN-based networks regarding also constraints on delay, link utilization, server utilization. A similar study presented in Varasteh, Madiwalar, Van Bemten, Kellerer & Mas-Machuca (2021) formulates the power-aware and delay constrained joint VNF placement and routing (PD-VPR) problem as ILP. The authors propose an online heuristic framework in which the centrality-based ranking method is adopted for the placement sub-problem and the Lagrange Relaxation based Aggregated Cost algorithm is used for the routing sub-problem.

1.2.4 Discussion

In general, the VNF placement and routing problem has been investigated by a large body of work with different objectives, such as minimizing total deploying cost, total E2E delay, network resource usage as well as routing costs or maximizing reliability. Even though all the aforementioned studies are applicable to any applications, including IoT services, it requires adapting the model to align with the underlying IoT network as well as the service real-time and user experience requirements. Some efforts Taleb *et al.* (2015); Laghrissi *et al.* (2017); Gong *et al.* (2016); Cao *et al.* (2018); Li *et al.* (2019) have been actually conducted to address the problem under IoT context. Closest to our work is MaxZ Agarwal, Malandrino, Chiasserini & De (2018) that proposes a model accounting for services involved in 5G networks such as IoT, Machine-to-Machine (M2M) applications. Adopting a queueing model for VNFs, the authors deal with traffic not only between VNFs but also from outside the system, which might be applied for the case of IoT devices. However, MaxZ neglects IoT nodes, in terms of their

resource capacity and connection delay, and this, as confirmed by our numerical results, can yield sub-optimal performance. In this work, we consider three system features, i.e. distributed clouds, multiple VNF instances, connections between clouds and underlying IoT networks, which are not taken into account in prior works.

1.3 Network Resource Allocation for SDN-based IoT Network

In this section, we first provide an overview of the state-of-the-art solutions relevant to context-aware IoT service orchestration architecture, VNF and SDN controller placement. Then we specify the need for employing contextual information in service orchestration as well as in allocating network and computing resource.

1.3.1 IoT Service Orchestration

According to Sousa, Perez, Rosa, Santos & Rothenberg (2019), service orchestration is the ability to coordinate resources and services in multiple administrative domains that cover various technology domains. In IoT context, a high-level service is typically decoupled into more granular service components or functions from different layers, i.e. operational layer, resource layer. IoT service orchestration is a process that enables the automation of coordinating such service components during the execution phase. Many aspects of service orchestration have been addressed in prior studies. In de Brito *et al.* (2017), the architecture that enables service orchestration for Fog infrastructure has been proposed. Cerroni et al. in Cerroni *et al.* (2017) proposed a reference architecture that allows various domains (i.e. IoT, SDN, NFV) to be managed and facilitate service orchestration with the help of a virtual network function manager and NFV orchestrator. Reference Gochhayat *et al.* (2019) presents a lightweight context-aware IoT service architecture that supports IoT push-based service in an efficient manner by modeling end-users in terms of their contextual and profile information. Also focusing on user tasks to counter the heterogeneity of IoT domains, an adaptive service composition framework is introduced in Urbieta, González-Beltrán, Mokhtar, Hossain & Capra (2017).

Close to our approach from the architectural view is the work in Fichera, Gharbaoui, Castoldi, Martini & Manzalini (2017). With respect to the effective and reliable provisioning of 5G services, the authors presented a 5G architecture that handles such a flexible environment of virtual resources like 5G to orchestrate in a generalized way. Nevertheless, similar to other works that mainly focus on a context-aware IoT orchestration architecture, Fichera *et al.* (2017) does not explain how contextual aspects can be orchestrated as a service with those offered from network controllers and cloud platform. In addition, the dynamic nature of IoT applications is not elaborated in Fichera *et al.* (2017).

1.3.2 VNF & SDN Controller Placement

VNF placement problem along with contextual information has been studied in a large number of published papers. In Taleb *et al.* (2015), the authors deal with service usage behavioral patterns of mobile users in obtaining an optimal scheme of allocating VNFs resource. Cziva *et al.* (2018) takes into account the dynamic nature of IoT network apart from user mobility and presents a dynamic scheduler of placing VNFs on a distributed edge infrastructure. It is typically assumed that service components are implemented as VNFs. The joint VNF placement and path selection problem is thus investigated in Kuo *et al.* (2016) from the service chain perspective regarding the relation between link and server usage. In Bhamare *et al.* (2017), analytical models for service function placement are adopted in multi-cloud environments.

Similar efforts have been explored in finding an optimal placement scheme for SDN controllers. The work Dixit, Hao, Mukherjee, Lakshman & Kompella (2013) presents an architecture that allows optimally placing switches under the management of elastic distributed SDN controllers. The authors in Wang, Liu & Xu (2017) present a dynamic scheme to assign switches to controllers with a goal to balance the controller load while keeping the control traffic overhead low. Switch migration schemes are proposed to minimize the migration cost Xu *et al.* (2019) or to reduce flow setup time Ye, Cheng & Luo (2017).

However, all the aforementioned studies are limited to the sequential execution of service functions. In this work, we leverage contextual knowledge to solve joint placement optimization. We profile various aspects of contextual information and demonstrate how the contextual profiles can be adopted with other information provided by resource controllers in the system model to maximize user experience while achieving an optimal placement scheme for VNFs and SDN controllers. Another factor that distinguishes our work from the rest is that it accounts for different ways in which service functions are linked. Given the inherently complicated relation between IoT service components, we believe that our solution can pave the way toward accelerating the progress of deploying IoT services in the industry.

1.3.3 Discuss

Extensive research work has been devoted proposing an IoT service orchestration architecture considering different contextual aspects. For IoT devices, it is about network access, user mobility or locations whereas for service, it is about Quality of Service (QoS) requirements, cost, and for network, topology, bandwidth, communication latency Gochhayat *et al.* (2019); Urbieta *et al.* (2017); Fichera *et al.* (2017) matter. Especially, user mobility is involved as a strategy for orchestrating service functions, placing VNF and/or SDN controllers to maximize user experience while optimizing resource availability Song, Lee, Cho, Lim & Chung (2019); Harutyunyan, Shahriar, Boutaba & Riggio (2019); Gharbaoui *et al.* (2018). However, in order to orchestrate heterogeneous service functions to deliver an E2E IoT service in an arbitrary smart environment and during the operational phase with confidence, it needs to consider context awareness, VNF resource provisioning, SDN controller altogether. These concerns have not been fully done in any of the prior works to the best of our knowledge. Obtaining a jointly optimal result from such consideration is challenging for two reasons, i.e., the complicated interaction between relevant systems, i.e., cloud platform, network controllers, and the introduction of additional constraints as well as variables to the optimization problem. While the former calls for a careful mechanism to manage the message exchange and to synchronize systems' states,

the latter exponentially increases the complexity of the resource allocation problem and thus makes it even more difficult to solve efficiently.

1.4 General Discussion

As part of any service orchestration platform, improving E2E service latency has always been a key feature to enhance the QoE from the perspective of end-users. However, prior works are no longer appropriate for future smart community applications regarding the real-time service requirement and the heterogeneous IoT network. With a target to reduce E2E latency for interworking services deployed across multiple domains, traditional approaches focused on provisioning resource to enable the deployment of service chains in cross-domain infrastructure without providing the details of modeling the interworking scenarios between service components in different domains. The proposed solutions have not leveraged the distributed system to enable parallel computing and thus improving the performance of solving optimization problems of resource allocation.

On the other hand, many studies have been proposed to obtain an optimal strategy of virtual resource placement in SDN networks. However, with the consideration only on a "single-layer" optimization goal, i.e. either VNF placement and routing problem or SDN controller placement, these solutions reveal the inefficiency when being adopted for IoT-enabled networks where both cloud and edge resource are available. The growing adoption of virtualization enables finer-grained resource allocation capabilities and as a result challenges the attainment of an optimal solution of allocating resource with the respect to low service latency requirements. It is truly important to take into account the problem of resource placement jointly from service components implemented as VNF and SDN controllers whereas ensuring the entire system's queue stability as a counter for dynamic service demand from IoT services.

Table 5.1 summarizes some works on the service chain placement problem.

Table 1.1 Related works on interworking IoT service chain optimization problem in edge-cloud SDN-based networks

Reference	IoT Service and Resource Management			
	Multi-domain	E2E latency	Dynamic Optimal	Distributed Algorithm
Pham & Chu (2019); Liu & Han (2019); Uniyal *et al.* (2020); Xiang *et al.* (2019)	Yes	No	Yes	No
Sun *et al.* (2020); Santos *et al.* (2020); Mouradian *et al.* (2019); Montazerolghaem *et al.* (2016)	No	Yes	Yes	No
Bhamare *et al.* (2017); Gupta *et al.* (2018); Sun *et al.* (2018)	Yes	No	No	Yes
Our work	Yes	Yes	Yes	Yes

Reference	VNF Placement for IoT Services			
	Topology Awareness	Distributed Clouds	VNF Placement	VNF Chaining
Li *et al.* (2019); Xu *et al.* (2018); Qi *et al.* (2019); Cziva *et al.* (2018); Laghrissi *et al.* (2017)	No	Yes	Yes	No
Gong *et al.* (2016); Cao *et al.* (2018); Gupta *et al.* (2017); Riera *et al.* (2016)	Yes	Yes	Yes	No
Yang *et al.* (2020); Gao & Rouskas (2020); Sen *et al.* (2020); Van Bemten *et al.* (2018); Sallam *et al.* (2018)	No	Yes	No	Yes
Ye *et al.* (2019); Varasteh *et al.* (2021); Farkiani *et al.* (2019); Tajiki *et al.* (2019)	No	No	Yes	Yes
Our work	Yes	Yes	Yes	Yes

Reference	SDN-based IoT Network			
	Controller ↔Switch	VNF Placement	System Stability	-
Dixit *et al.* (2013); Wang *et al.* (2017); Xu *et al.* (2019); Ye *et al.* (2017)	Yes	No	No	-
Taleb *et al.* (2015); Cziva *et al.* (2018); Kuo *et al.* (2016); Bhamare *et al.* (2017)	No	Yes	No	-
Our work	Yes	Yes	Yes	-

CHAPTER 2

OBJECTIVES AND GENERAL METHODOLOGY

This chapter explains in details the objectives of the thesis with respect to the aforementioned research problem and research questions as well as the limitation of prior works. Then the general methodology in three phases to obtain the objectives will be described. The relationship between the phases is visualized to facilitate the reading of this thesis.

2.1 Research Hypothesis

The research hypothesis (RH) of this framework is defined as follows:

RH: By jointly optimizing the VNF resource placement and chaining over SDN controlling and forwarding planes, and by taking into account interworking services deployed across multiple domains with the dynamic service demand from IoT devices, we improve entire system performance with stability, enhance QoS, QoE and minimize the total cost of allocating cloud computing and network resource.

2.2 Main Objective

The main objective (MO) of this work is defined as follows:

MO: Achieve an efficient computing and networking resource placement and chaining strategy over both SDN controlling and forwarding planes, and a model of E2E service latency for interworking services deployed over multiple domains that minimize the total cost of resource usage and service operations while meeting QoS and QoE requirements as well as maintaining system's queue stability given dynamic service demand.

2.2.1 Specific Objectives

The main objective (MO) can be further divided into three specific objectives SO-1, SO-2 and SO-3 that respectively address the above research questions RQ-1, RQ-2 and RQ-3:

2.2.2 Specific Objective SO-1

SO-1: Enabling the deployment of IoT services across network topologies including multiple edges and clouds with minimal E2E service latency.

In order to deploy an optimal VNF resource placement strategy to IoT edge-cloud networks for large-scale IoT applications, it is necessary to investigate the impact of VNFs' input data from IoT devices. Thus we adopt an IoT network topology-aware approach aiming to achieve an efficient VNF placement and chaining strategy in IoT networks.

2.2.3 Specific Objective SO-2

SO-2: Optimizing resource allocation for an elastic SDN control plane and a scalable SDN forwarding plane given dynamic service demand from IoT networks.

In order to reduce total resource cost for SDN/NFV-based networks to provide IoT delay-sensitive services with fluctuate service demand, we need to efficiently not only allocate resource for SDN controllers mapping to forwarding network functions but also place computing functions onto physical hosts and chain them. As a matter of fact, the fast growth in connected IoT devices and the ever-increasing number of services have led to an incredibly huge amount of traffic to SDN networks. Under dynamic traffic demand, not only do nodes in forwarding plane face huge scalability challenges but also SDN controllers at control plane have to process a large number of flow setup messages. In other words, we need a strategy to assign switches to controllers. And due to the vital role of SDN controllers in terms of the impact on traffic of both forwarding and computing nodes, such the strategy should take into account the stability of the entire system from queueing perspective.

2.2.4 Specific Objective SO-3

SO-3: Accommodating QoS requirement in terms of E2E service latency for multi-domain interworking service chains.

Today, many large-scale IoT applications are composed of different components which need to interwork across multiple domains. In order to effectively employ the resource allocation model obtained in SO-1 and SO-2, it is important to model E2E service latency of such IoT applications. Having said that, our objective is to ensure the QoS requirements of multi-domain interworking IoT services.

2.3 General Methodology

We propose three consecutive methodologies M1, M2 and M3 to respectively address the requirements of the research questions RQ1, RQ2 and RQ3 (discussed in Section 0.2.2) as well as the specific objective SO1, SO2 and SO3 (discussed in Section 2.2.1). The three methodologies are defined as follows:

2.3.1 Methodology M1: IoT Network Topology Aware VNF Placement

The methodology M1 addresses the research question RQ1 and the specific objective SO1. In this methodology, the VNF placement and chaining problem model is extended to IoT applications that involve multiple IoT terminals. We introduce an orchestration system that enables to collect resource profile relevant to VNF placement and network traffic sent by IoT devices via IoT gateways as input data for IoT services. We leverage contextual IoT network topology to improve the E2E service latency model and the performance of resource allocation algorithm. The methodology M2 is summarized as follows:

- Design an implementation system that enables to deploy the optimization solution for service chains across multiple edges and cloud.
- Model the VNF placement and chaining problem regarding service chains and aggregated input traffic from IoT devices

- Design an IoT network topology-aware VNF placement algorithm in a distributed manner.

2.3.2 Methodology M2: Optimizing Resource Placement for SDN Controllers and Forwarding Nodes

The methodology M2 addresses the research question RQ2 and the specific objective SO2. In this methodology, we provide detail of the functional components of a resource orchestrator. Compared to existing resource allocation solutions, we take a step further on the consideration of both control and forwarding planes in SDN networks. Based on that, a joint optimization problem of SDN controller placement and VNF service chaining is considered, along with the guarantee of system's stability by adopting Lyapunov optimization framework. The methodology M2 is summarized as follows:

- Design a functional diagram for the orchestration of heterogeneous resource controllers
- Define a comprehensive model of resource placement and chaining problem for both SDN control and forwarding planes
- Adopting Lyapunov queueing framework to real-time optimization problem while ensuring entire system's queue stability
- Solving the transformed optimization problem using Exponential Gradient Ascent in countering the exponential growth of network size.

2.3.3 Methodology M3: E2E service latency modeling over the time for multi-domain interworking IoT services

The methodology M3 addresses the research question RQ3 and the specific objective SO3. In this methodology, we design a cloud-native interworking system between IMS and WebRTC domain. We then determine service chains with detailed message flows for common use cases, i.e. session setup and calling. The whole system with the 2 use cases is deployed as a testbed to assess the proposed real-time algorithm of optimizing resource allocation regarding E2E service latency over the time and potential QoS violation penalties. The methodology M3 is summarized as follows:

- Design a NFV-based interworking architecture enabling multi-domain orchestration of service providers, i.e. IMS and WebRTC domains.
- Detail message flows between VNFs implementing interworking service components and model E2E service latency over the time with various resource provisioning strategies.
- Experimental analyze the proposed solution with an interworking IMS $\rightleftarrows$ WebRTC testbed.

A summary diagram of the thesis is presented in Fig. 2.1.

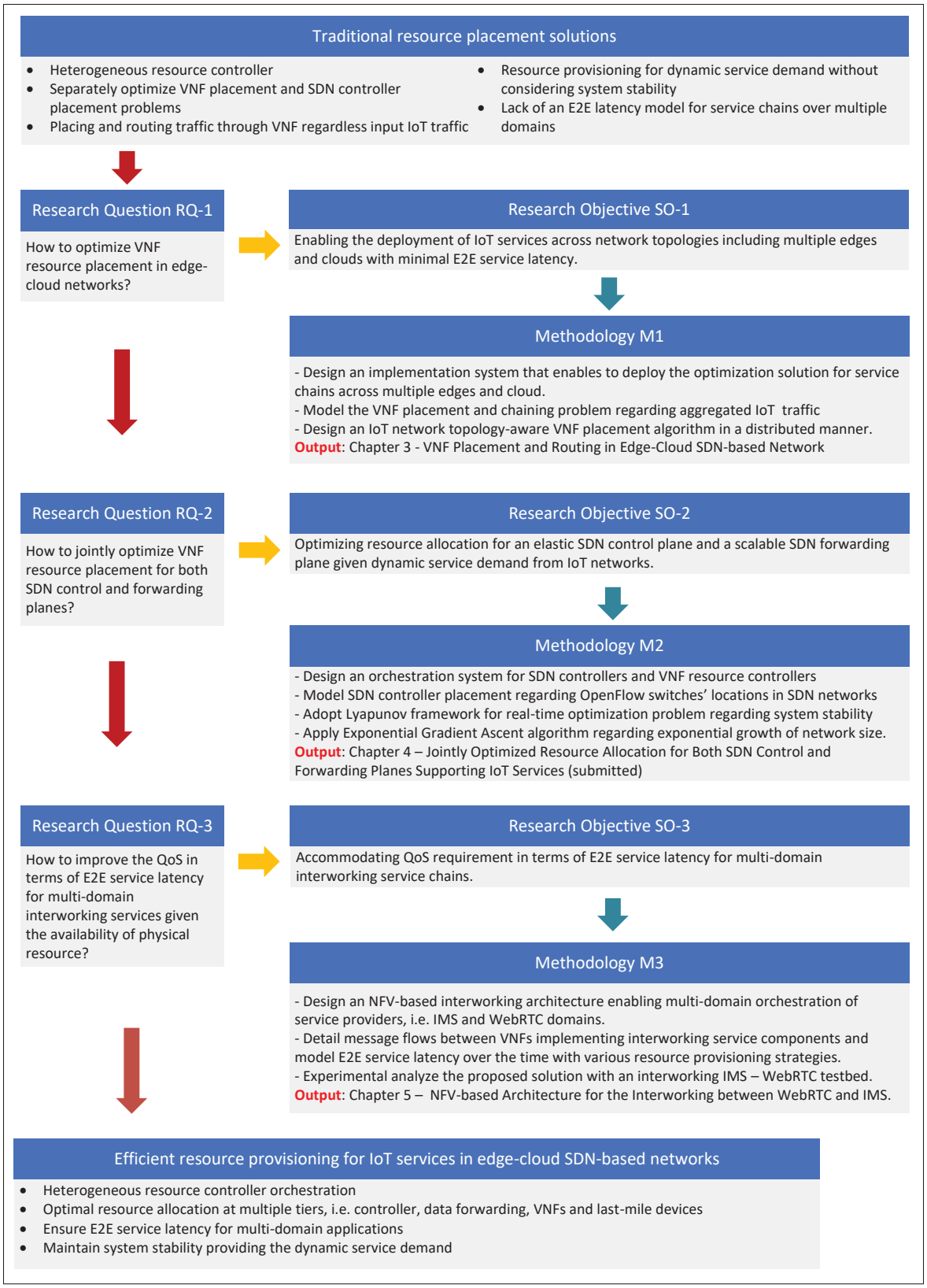


Figure 2.1 Outline diagram of the thesis

CHAPTER 3

PLACEMENT AND CHAINING FOR RUN-TIME IOT SERVICE DEPLOYMENT IN EDGE-CLOUD

Duong Tuan Nguyen[1] , Kim Khoa Nguyen[1] , Mohamed Cheriet[1]

[1] Department of Automation Production, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Article published in *IEEE Transactions on Network and Service Management*, March 2020

Abstract

This paper investigates an efficient placement and chaining of Virtual Network Functions (VNFs) to provide cloud based IoT services with minimal resource usage cost. We take into account bandwidth capacity and link delay of network connection between clouds where VNFs are allocated and underlying IoT networks where sensors and IoT gateways are deployed. Regarding the constantly changing network dynamics, input traffic of service components is considered at the lower granularity level of messages based on the communication between each VNF and corresponding sensors via IoT gateways. From the algorithm perspective, the specific topology of multiple edge clouds is leveraged to improve the solution. In this paper, we present an NFV-based high-level architecture for a system that enables the deployment of IoT services across multiple edges and clouds. We formulate the VNF placement problem using a non-convex Integer Programming model. Taking into account different IoT topologies, we devise two algorithms for small- and large-scale networks to find the near optimal solution: i) a customized Markov approximation with two techniques, i.e. multi-start and batching, and a node ranking-based heuristic. Simulation and experimental results show that the proposed solution improves the cost up to 21% compared to state-of-the-art schemes.

Keywords: VNF Placement, Service Function Chain, IoT Services, Edge/Cloud Computing, QoS.

3.1 Introduction

With a dramatic growth in the volume of network traffic over the last decade, NFV ETSI ISG on Network Functions Virtualization (2013) has been considered as a promising solution whereby network services are provisioned in software-based network functions or elements, i.e. bridges, routers. Thanks to virtualization technology, heterogeneous virtual networks can coexist in the same physical (or substrate) network and share the resources efficiently. This paper focuses on a class of IoT services which are typically composed and deployed at run-time to respond to user's need in a specific context Miorandi, Sicari, Pellegrini & Chlamtac (2012). Adopting NFV paradigm allows high flexibility to adapt to the change of service demand, which is critical in the success of IoT application delivery with regard to service performance and reliability.

Along with NFV advantages is a key challenge related to the optimal allocation of resources of a substrate network to virtual network requests, or VNE. Despite being intensively investigated in literature Cziva *et al.* (2018); Li *et al.* (2018a), deploying such VNE solutions in an arbitrary IoT environment with confidence is still challenging regarding delay sensitivity of IoT services and the constantly changing network dynamics. For the former aspect, previous VNE approaches has mainly focused on the communication between VNFs at data center Li *et al.* (2018a) in modeling service latency. In IoT context, modeling E2E service delay for VNF placement problem requires to consider not only VNF-VNF connection but also between VNF and IoT sensors via IoT gateways which has not been considered in prior work. This is challenging given the complicate interaction of relevant IoT sensors and IoT gateways with VNFs at clouds, i.e. from IoT devices to the VNFs that need to collect sensing data, or in reverse direction to activate certain device's functions. Mathematically, the presence of such the IoT devices introduces additional elements which increase the inherent system complexity and thus creates new constraints to VNE problem.

Regarding the dynamic nature of network traffic, the bandwidth resource Agarwal *et al.* (2018) should be considered in the VNF placement and chaining problem. Unlike previous studies that addressed this issue by assuming continuous bandwidth demand which is not completely

appropriate for IoT devices Zheng, Tsiopoulos & Friderikos (2018); Dieye *et al.* (2018); Mechtri, Ghribi & Zeghlache (2016b); Ghaznavi *et al.* (2017), we go a step further in this paper by investigating the impact of VNFs' input traffic at the lower granularity level of discrete messages via connections between IoT networks and clouds. In addition, we argue that placing VNFs based on resources allocated statically in advance might be not optimal in reality. For practical techniques such as statistical multiplexing of service requests to benefit system resource usage Habibi, Fazli & Movaghar (2019) which are appropriate for IoT applications, network resource should be taken into account at a higher dynamic level, i.e discrete messages, rather than in a static manner as in existing approaches Cziva *et al.* (2018).

The paper contribution is three-fold. First, we design a system that enables the deployment of IoT applications in form of service chains across multiple edges and clouds. Second, we propose a model for the optimization problem of VNF placement and chaining with aggregated traffic from IoT gateways and formulate it as a non-convex Integer Programming (IP) problem. The novelty of our model lies in the consideration of input traffic of VNF and the presence of IoT devices, i.e sensors, gateways in IoT services. Particularly, we model the latency for service chains while taking into account the specifications of connection between clouds where VNFs are deployed and IoT gateways, such as the distance to IoT devices and the connectivity to multiple edges and clouds. Third, regarding the NP-hardness of proposed problem, we introduce a Markov approximation based framework that adopts multistart and batching techniques (MBMAP) to solve the combinatorial network problem. The framework exploits underlying IoT infrastructure to perform algorithms in a distributed manner and consequently accelerate convergent rate which has been known as a limitation of Markov-based algorithms due to the large space of states. We also present another heuristic (NRP) that employs the concept of node rank in placing VNFs. The heuristic aims for large-scale networks and is considered as a baseline to demonstrate the advantage of MBMAP given a large number of possible states. Our source code is available online[1] for other researchers to use and modify. Simulations and experiments' results show the effectiveness of the proposed MBMAP over prior works that do not consider the IoT network.

[1] https://github.com/hoangtuansu/smal

The rest of this paper is organized as follows. Section 3.2 reviews prior works. System modeling and the formulation of the optimization problem are explained in section III. Then Markov-based approximation algorithm MBMAP and node-ranking heuristic NRP are described in section IV. Section V presents performance evaluation of the proposed methods with simulation and testbed settings. Finally, conclusions are drawn.

3.2 Related Work

With the rapid growth of virtualization technology, a large number of recent publications have studied VNF optimal resource provisioning and service chain routing. The VNE has been investigated in the literature from various aspects, such as system models Zheng *et al.* (2018); Dieye *et al.* (2018), objectives Cziva *et al.* (2018); Li *et al.* (2018a) and solutions Dieye *et al.* (2018); Mechtri *et al.* (2016b). In this section, we summarize the main results on VNE for IoT services and explain how our work is distinguished from the others.

In Taleb *et al.* (2015), the authors propose the solutions for the problem of VNF embedding for virtual 5G network infrastructure while dealing with the mobility features and service usage behavioral patterns of mobile users. The solutions address two conflicting objectives, which are the insurance of QoE via the placement of VNFs of data anchor gateways closer to end users and the avoidance of the relocation of mobility anchor gateways via placing their corresponding VNFs far enough from users. Apart from user mobility, the constantly changing network dynamics as a well-known characteristic of IoT network is addressed in Cziva *et al.* (2018). Another IoT service specific is the presence of micro-data centers, known as edge cloud, whose locations significantly affect the requirement of ultra-short latency and has been investigated in Laghrissi *et al.* (2017). The study in Gong *et al.* (2016) address the VNE problem regarding the constraints related to the location of substrate nodes. Cao *et al.* (2018) adopts network topology information including node location to conduct a node-ranking approach to solve the problem. In general, all of these prior works mainly focus on VNF location optimization for services between end-user and corresponding VNFs, not service function chain.

Focusing on the relation between link and server usage, the authors in Kuo *et al.* (2016) investigate the joint VNF placement and path selection problem. While the approach can be generalized to include the underlying IoT network, it requires an effort to adapt the model for distributed clouds as well as the formulation of constraints on service chain latency in the IoT context. A similar approach in Allybokus *et al.* considered partial orders and anti-affinity rules which states that two VNFs cannot handle the same service chain on the same node

In Bhamare *et al.* (2017), the authors propose an analytical model for the placement of service function chains in multi-cloud environments. However, they only consider inter-cloud traffic *w.r.t* the fact that inter-cloud links are more likely to be congested and more expensive compared against the links within a single datacenter. Another work for placing service chains across multiple clouds in Gupta *et al.* (2017) adopts machine learning technique for a predictive model combining with random cloud selections. Tackling the issue of deploying network services across multiple Points of Presence (PoPs), the framework in Riera *et al.* (2016) provides an optimization model on various metrics, i.e. cost of assigning VNFs to PoPs, overall delay, and overall resource link usage.

Closer to our work is MaxZ Agarwal *et al.* (2018) that proposes a model accounting for services involved in 5G networks such as IoT, M2M applications. Adopting a queueing model for VNFs, the authors deal with traffic not only between VNFs but also from outside the system, which might be applied for the case of IoT devices. However, MaxZ neglects IoT nodes, in terms of their resource capacity and connection delay, and this, as confirmed by our numerical results, can yield sub-optimal performance.

In this work, we consider three system features, i.e. distributed clouds, multiple VNF instances, connections between clouds and underlying IoT networks, which are not taken into account in prior works.

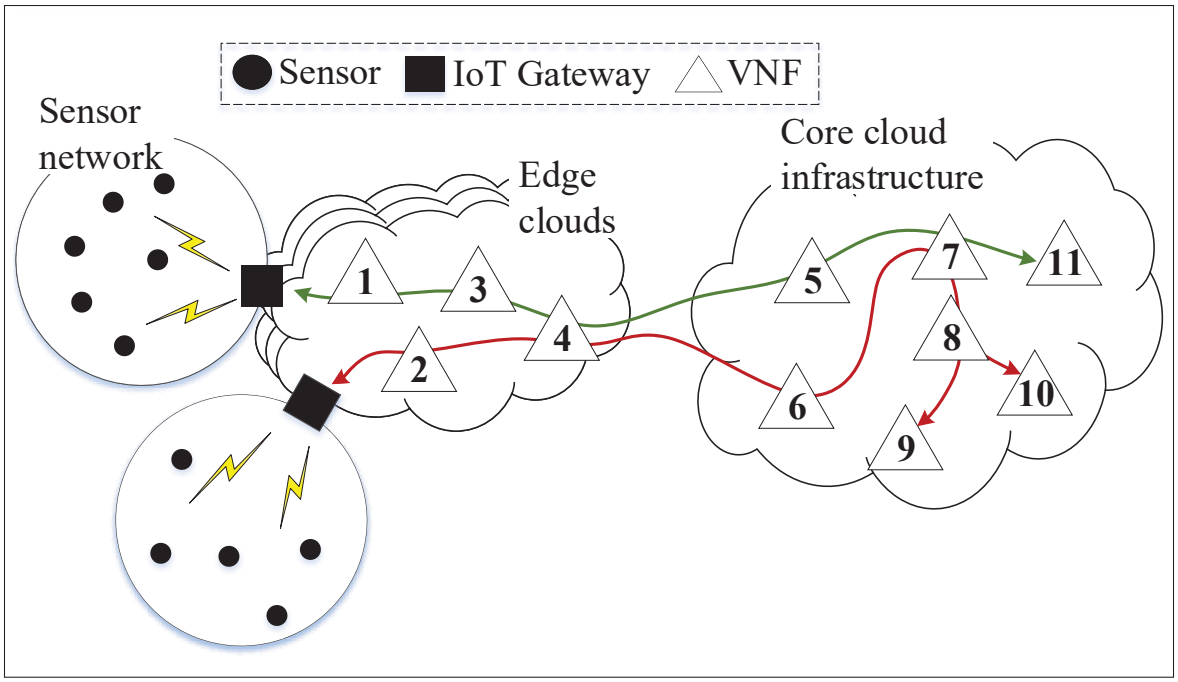


Figure 3.1 Multi-cloud service function chain for IoT applications

3.3 System Architecture

In this section, we describe the system that performs service function chaining on multiple clouds for IoT applications. The overall architecture as a reference for implementing and deploying proposed solution with an illustrative IoT-based use case is explained.

3.3.1 System Description

Fig. 3.1 depicts a system composed of multiple clouds where VNFs are deployed to implement service functions. Each VNF can be replicated on different places depending on the number of licenses that the provider has purchased Luizelli, Bays, Buriol, Barcellos & Gaspary (2015). A VNF can process network traffic from other VNFs or sensor devices (or nodes) scattered in a sensor field via IoT gateways. While a sensor may have multiple interfaces, i.e. Bluetooth, WiFi, LTE, due to its constrained resource, only one interface is activated at a given time and connects to one gateway within its coverage. Each node either collects data (i.e. temperature, noise) or

performs a certain function (i.e. sprinkle, smart light). Toward sensor side, the VNF either receives and processes data from that sensor or sends a control message to activate its function.

The IoT gateways aggregate data from connected sensors and communicate with VNFs through the network link between the gateways and the clouds. A gateway might have various interfaces (i.e., wireline, cellular, LoRA) and thus connects to several clouds at the same time through the Internet. A user request for a service will be served by a chain of service functions performed by VNFs which interact with the IoT gateways to retrieve the input data or trigger the commands from or towards the sensor. In this work, we consider a common IoT case in which service functions are executed in sequential or branching manners Halpern & Pignataro (2015).

3.3.2 Overall Architecture

From the aforementioned system, we design an implementation architecture that takes into account not only the presence of multiple clouds but also the service management for micro-services as service functions, and underlying IoT network as shown in Fig. 3.2.

At each edge, Optimization Agent (OpAg) and VNF Allocator (VAl) are two main components of Edge Orchestrator (EdOr). Specific components and functionalities of EdOr are similar to MANO reference architecture that can be found in ETSI ISG on Network Functions Virtualization (2013). The role of OpAg is to expose both resource and service function's information of the edge to the Global Optimizer (GlOp) at Core Orchestrator (CoOr) which is in turn deployed at Core Cloud. Placement scheme returned by OpAg is used by VAl component to perform cloud's resource allocation.

The Network Controller is responsible for controlling network resources and establishing connectivity between VNFs that implement service functions. It maintains a list of IoT network topologies including gateways and sensors, from which providing necessary information, i.e. data rate, latency, as input of OpAg.

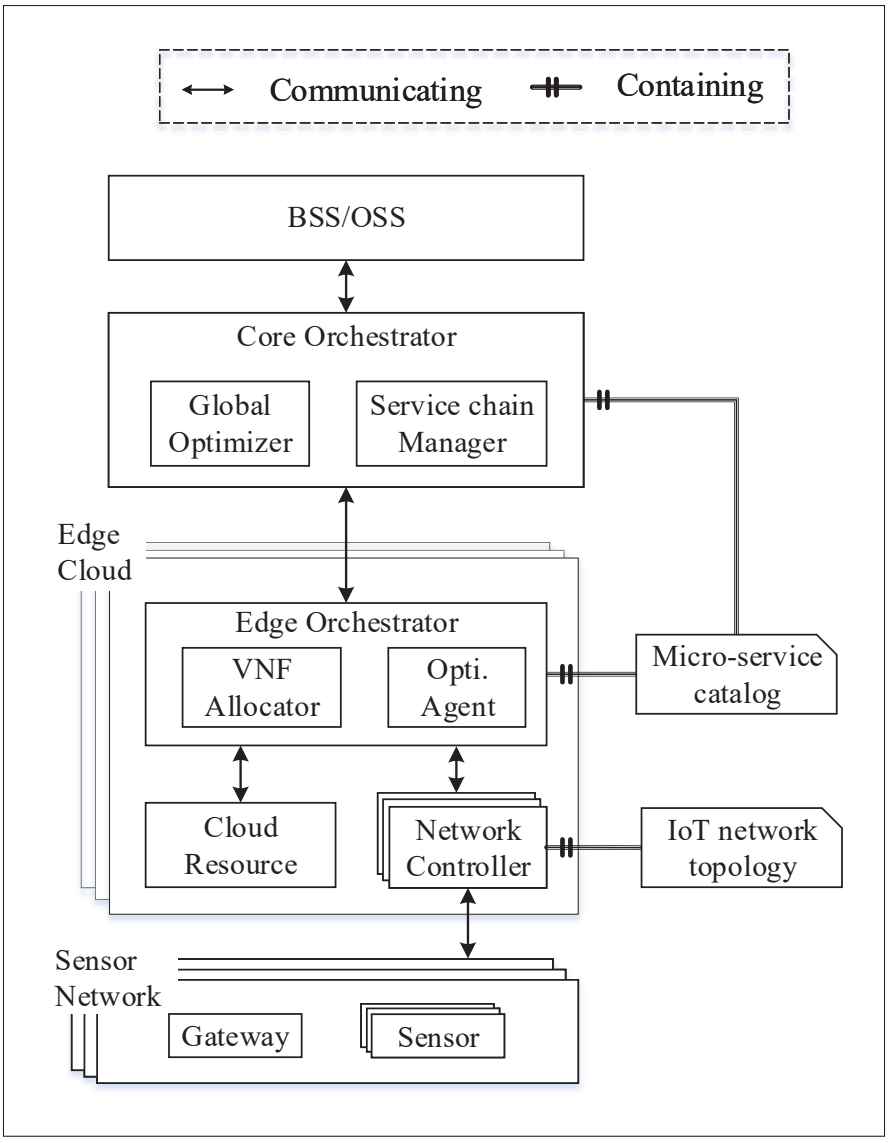


Figure 3.2 Implementation architecture

At Core Cloud, Service Chain Manager component of the CoOr retrieves information from BSS/OSS system to construct a catalog of IoT applications. Similarly to the EdOr, the CoOr's catalog is served as the input of GlOp and might be related to multiple edge clouds.

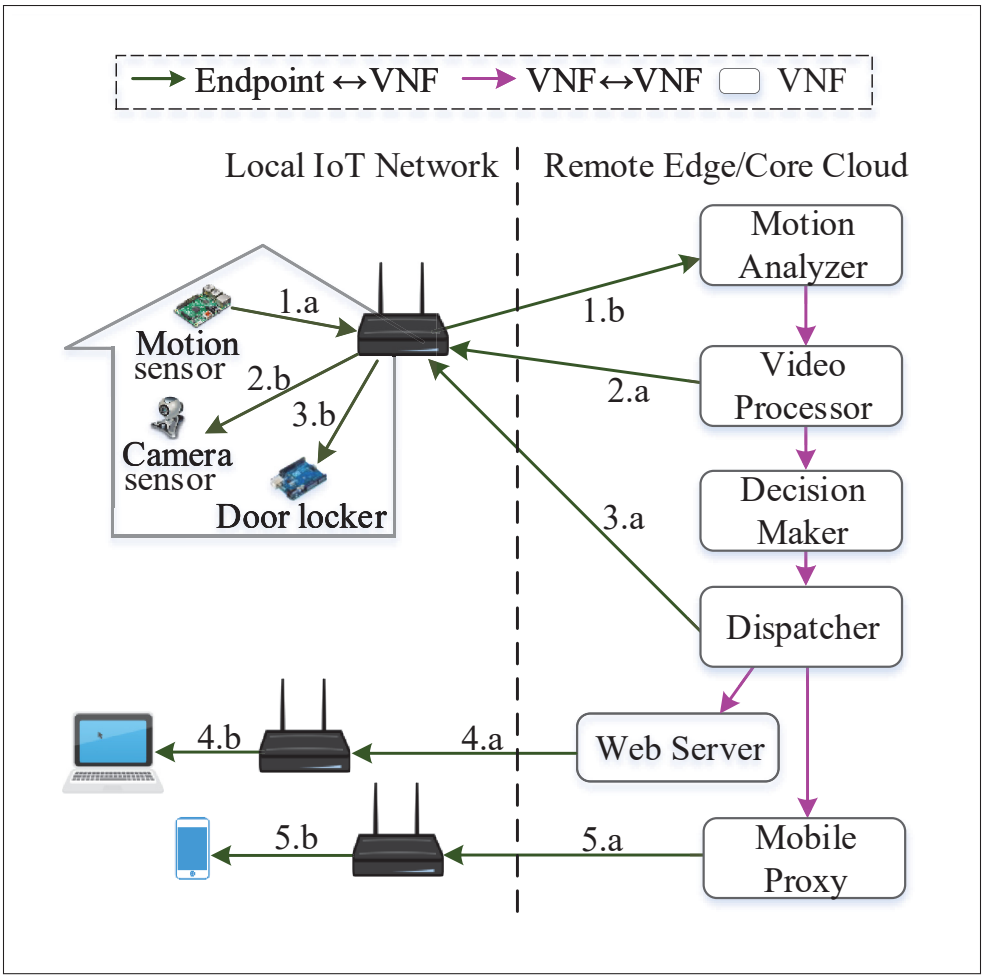


Figure 3.3 An illustrative IoT service chain with multiple end-points

3.3.3 Illustrative Use Case

An illustrative example of the service chain that is formed in accordance with a security surveillance scenario as shown in Fig. 3.3. In this use case, a motion sensor (MS) is the initial source, a camera sensor (CS) provides supporting data to improve the decision making process, and destination nodes include Door Locker, a laptop representing surveillance service provider, and a mobile phone as a user. Service functions, e.g. Motion Analyzer, Video Processor (VP), Decision Maker (DM), Dispatcher (DP), Web Server (WS) and Mobile Proxy (MP), are implemented as VNFs running on edge-cloud. Upon detecting a motion (step 1a & 1b), Motion Analyzer (MA) checks whether or not it is a suspicious move, i.e., not via the main door. If it is

the case, the MA will inform VP to trigger the camera sensor to perform at a higher resolution (step 2). Using face recognition, DM decides whether or not it is an intrusion if the person is not identified as a home user. DP receives decision result from DM and send an activation message to Door Locker (step 3) as well as notifies other two endpoints (step 4 & 5) which are also behind IoT gateways. Note that the DP can be configured to forward the message to more than one endpoints at step 4 & 5.

In this use case, placing service functions or VNFs with only consideration of data center's resource does not guarantee the performance of IoT services. IoT traffic in terms of sets of discrete messages, if ignored, may result in a sub-optimal placing solution as confirmed by our simulation and experimental results. Moreover, the communication delay between local IoT networks and remote edge/core clouds also plays an important role in E2E service latency, which is critical in many scenarios, e.g., security surveillance.

3.4 System Modeling & Problem Formulation

We model the system described in Fig. 3.1 as a directed graph $\mathcal{G} = \{N \cup G, E\}$ where $N \cup G$ and E are the sets of nodes and links respectively. To facilitate the model, both edge and core cloud are referred to the set of N clouds and G is the set of IoT gateways. A link $(q, q') \in E$ connecting two entities either clouds or gateways or both represents a logical communication link between them. $\Phi^B_{q,q'}$ and $l_{q,q'}$ denote the capacity and delay of edge (q, q'), respectively. While $\Phi^B_{g,n}$ is estimated based on the communication technology of gateway's network attachment point, $\Phi^B_{n,n'}$ is usually determined by the contract between network infrastructure providers. We use m^{com}_n and $m^{net}_{q,q'}$ to define the cost of one computing resource unit at $n \in N$ and one network bandwidth unit over the link (q, q'), respectively. The mathematics notations are summarized in Table 5.2.

We collect the set V of VNFs hosted at the clouds. For any VNF $v \in V$, let κ_v denote the number of v's replications (or instances), v_i where $i = 1, \ldots \kappa_v$ is the i-th replication of v, b^{out}_v the required bandwidth for an IoT gateway to send v's aggregated messages to other VNFs,

Table 3.1 Notation List

General Inputs	
N, G, V, C	Set of clouds, IoT gateways, VNFs and chains
V_g	Set of VNFs associated with gateway g
$\alpha_{g,n}$	Indicator of the association between g and n
κ_v	Number of VNF v's replications
$\lambda_c, \lambda_v^{sen}, \lambda_{v_i}$	Arrival rate of service chain c's request, data from sensor to VNF v and network traffic at VNF instance v_i
τ_v	1 if v is an IoT-based VNF, 0 otherwise
$\beta_{u,v}^c$	1 if u links to v in service chain c, 0 otherwise
Service Latency	
Φ_c^L	Maximum tolerated delay of service chain c
L_{c,v_i}^{ctl}	Transmission delay between instance v_i and sensor
$L_{u,v}^{com}$	Transmission delay between two VNFs u and v
L_c	Total delay of service chain c
$\Gamma_{v_i}(\Gamma_g)$	Processing (aggregation) delay of instance v_i (gateway g)
System Resource	
b_v^{out}	Output network bandwidth of VNF v
b_v^{sen}	Network bandwidth between VNF v and sensor
Φ^B	Link capacity between any two nodes
$B_{q,q'}$	Network bandwidth between two nodes q, q'
r_n	Compute resource for n to process a bandwidth unit
R_n	Compute resource allocated at cloud n
System Cost	
m_n^{com}	Cost per compute resource unit at cloud n
$m_{k,q}^{net}$	Cost per bandwidth unit of the link between nodes k, q
$M_{net}(M_{com})$	Network (compute) resource cost of the whole system
M_{sys}	Total system cost
Decision Variables	
$x_{v_i}^n$	1 if VNF instance v_i is at cloud n, 0 otherwise
$y_{v_i}^c$	1 if service chain c uses VNF instance v_i, 0 otherwise

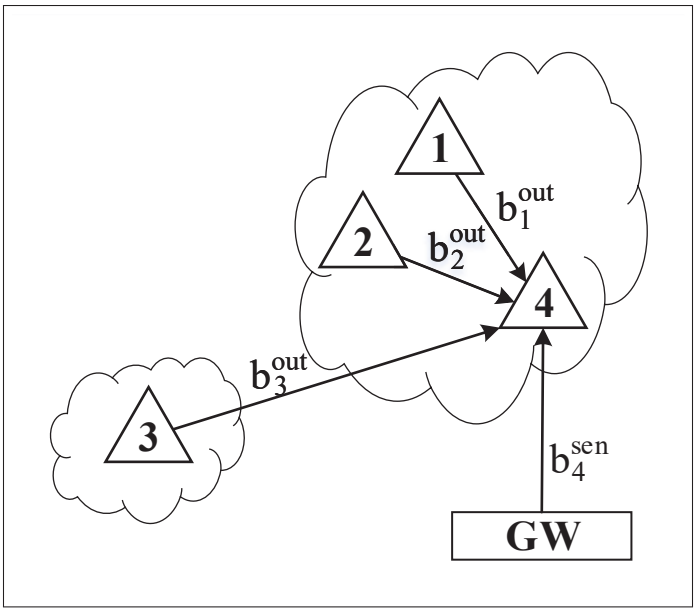


Figure 3.4 Details of bandwidth required by a VNF

and $\tau_v \in \{0, 1\}$ indicates whether v is an IoT-based VNF ($\tau_v = 1$) or not ($\tau_v = 0$). The implementation of VNFs is realized via virtual machines which are typically shifted in different templates (or configurations) in terms of CPU, memory, storage, and so on, depending on the cloud they are provisioned. Having said that, we use r_n to denote units of resource allocated for a VNF instance at the cloud n to process a bandwidth unit.

Given a gateway $g \in G$, a VNF v is associated with the gateway g if it is an IoT-based and g has a connection to its corresponding sensor. A set of such the VNFs is presented after g, i.e. V_g. Additionally, it is assumed that the sensor of the IoT-based VNF v's sensor generates sensing data at the rate λ_v^{sen} which requires b_v^{sen} units of bandwidth. For the sensors controlled by VNFs rather than generating sensing data, λ_v^{sen} and b_v^{sen} are set to 0.

Given C as the set of independently and identically distributed (i.i.d) service chains, each $c \in C$ is characterized by λ_c the initial service rate, $o(c)$ the source VNF, $d(c)$ the destination VNFs, Φ_c^L the maximum tolerated delay and $\vec{V}_c$ the directed tree composed of related VNFs. Any VNF can be shared by different service chains. One use case for such the shared VNF's instance is that the firewall function can be employed to filter traffic of multiple chains. For

the sake of simplifying the latency model of a service chain, the notation $\vec{V}_{c\dashv v}$ is used to present the sub-sequence (or a path) from the first VNF in c to v. From Fig. 3.3, $o(c)$ is the VNF MA while $d(c)$ is the set $\{DP, WS, MP\}$. An example of $\vec{V}_{c\dashv v}$ with v as VNF-WS is $\{MA \rightarrow VP \rightarrow DM \rightarrow DP \rightarrow WS\}$ or $\{MA \rightarrow VP \rightarrow DM \rightarrow DP \rightarrow MP\}$ with v as VNF-MP. Considering any two VNFs v and v', the notation $\beta^c_{v,v'} \in \{0,1\}$ with $\beta^c_{v,v'} = 0$ if $v \equiv v'$ indicates whether or not they are linked together regardless their instances and and $\sum_{v' \in V_c} \beta^c_{v,v'} = 1, \forall v \in V_c$.

The output of our model is the optimal solution of the VNF placement problem for the given set of inputs and is represented by decision binary variables $\mathbf{x} = \{x^n_{v_i}\}^{n \in N}_{v \in V, 1 \leq i \leq \kappa_v}$ and $\mathbf{y} = \{y^c_{v_i}\}^{c \in C}_{v \in V, 1 \leq i \leq \kappa_v}$. Precisely, $x^n_{v_i} = 1$ if v_i is allocated at n and 0 otherwise whereas $y^c_{v_i} \in \{0,1\}$ indicates the assignment of the replica v_i to requested service chain c.

3.4.1 Resource Constraint

The bandwidth required for the communication channel between the VNFs at the same cloud and associated with the same gateway g should not exceed the link capacity between g and n. Hence, with $\mathbf{x}^n_v = \sum_{1 \leq i \leq \kappa_v} x^n_{v_i}$, we get

$$B_{g,n}(\mathbf{x}) = \sum_{v \in V_g} b^{sen}_v \mathbf{x}^n_v \leq \Phi^B_{g,n} \tag{3.1}$$

Similarly, the total amount of bandwidth that any two consecutive VNFs in any service chain, that connects n' to n must be lower than $\Phi^B_{n',n}$. This value $B_{n',n}(\mathbf{x})$ is computed based on the data that a VNF instance generates towards its connected VNF of the same chain. Since each chain only has one pair of any two VNFs v', v, the value of $B_{n',n}(\mathbf{x})$ is obtained in terms of $\mathbf{x}^{n'}_{v'}$ and $\mathbf{x}^n_v$, that is

$$B_{n',n}(\mathbf{x}) = \sum_{c \in C} \sum_{v',v \in V} \beta^c_{v,v'} b^{out}_v \mathbf{x}^{n'}_{v'} \mathbf{x}^n_v \leq \Phi^B_{n',n} \tag{3.2}$$

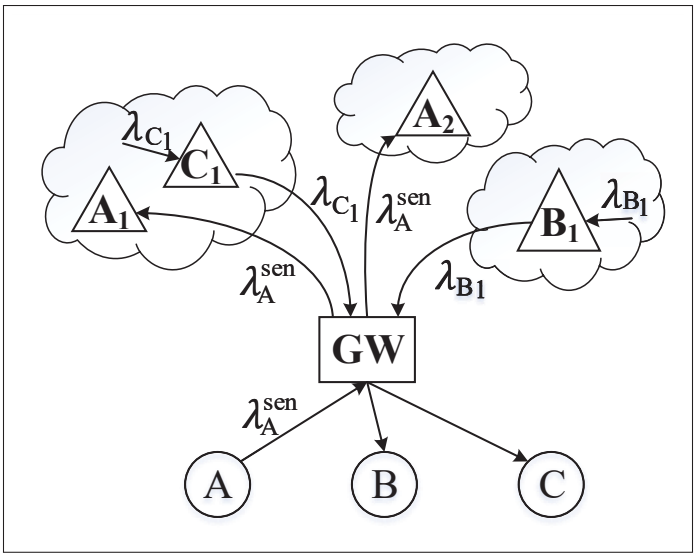


Figure 3.5 Arrival rate at VNF in details

For a VNF instance, there are two input data sources from its precedent connected VNFs of service chains, and the sensors in case of an IoT-based VNF as shown in Fig. 3.4. The bandwidth for an instance of v, i.e. B_v, is

$$B_v = \sum_{c \in C} \sum_{v' \in V} \beta^c_{v',v} b^{out}_v + \tau_v b^{sen}_v \tag{3.3}$$

The total amount of resource $R_n(\mathbf{x})$, $\forall n \in N$ needed to deploy a VNF for the cloud n considering resource availability Φ^R_n is computed as

$$R_n(\mathbf{x}) = \sum_{v \in V} \mathbf{x}^n_v r_n B_v \leq \Phi^R_n \tag{3.4}$$

3.4.2 System stability

We model a VNF replica as a $M/M/1$ queueing system with μ^n_v the service processing capacity and λ_{v_i} the arrival rate. Similar to the bandwidth, λ_{v_i} is also attributed to the traffic from two sources: precedent VNFs of v_i in all the chains of C, and its sensor through the gateway with

$\tau_v = 1$ and hence

$$\lambda_{v_i}(\mathbf{y}) = \sum_{c \in C} \sum_{v' \in V} \sum_{1 \le j \le \kappa_{v'}} \beta^c_{v',v} \lambda_{v'_j} y^c_{v'_j} y^c_{v_i} + \tau_v \lambda^{sen}_v \tag{3.5}$$

As there is only one precedent VNF of v in c, the Equ. (3.5) can be written in terms of λ_c as follows

$$\lambda_{v_i}(\mathbf{y}) = \sum_{c \in C} \left(\lambda_c y^c_{v_i} + \sum_{v' \in \vec{V}_{c \dashv v'}} \tau_{v'} \lambda^{sen}_{v'} \right) \tag{3.6}$$

Note that although the sensors are typically configured to periodically sense ambient conditions, the sensing periods are different from a sensor to others. Thus, it can be assumed that the data generated by sensors follows the Poisson process. In other words, the arrival of traffic to the IoT gateway can be considered as a Poisson process. It is reasonable to model the gateway as a $M/M/1$ queueing system, with μ_g the service processing rate together with λ_g. From Fig. 3.5,the gateway receives data from v's sensor if $\lambda^{sen}_v > 0$ and from its associated IoT-based VNFs if $\lambda^{sen}_v = 0$. Note that the IoT gateway's presence does not change the value of λ^{sen}_v as arrival rate of a $M/M/1$ system is equal to departure rate. This yields

$$\lambda_g(\mathbf{x}, \mathbf{y}) = \sum_{\substack{(n,g) \in E \\ v \in V_g}} \sum_{1 \le i \le \kappa_v} x^n_{v_i} (\lambda^{sen}_v + \lambda_{v_i}(\mathbf{y})) \tag{3.7}$$

To guarantee a VNF instance is not overloaded, the average time between two successive messages must be greater than the mean processing time by any server of v to a message. In other words, we require the stability condition for the system to be stable, that is

$$\lambda_{v_i}(\mathbf{y}) < \sum_{n \in N} x^n_{v_i} \mu^n_{v_i} \tag{3.8}$$

$$\lambda_g(\mathbf{x}, \mathbf{y}) < \mu_g \tag{3.9}$$

3.4.3 Service Latency Constraint

In order to formulate the latency of a service function chain, it needs to retrieve the formulation for the processing time at each VNF instance v_i, i.e. Γ_{v_i} and the aggregation time at g, i.e. Γ_g. From (3.5) and (3.7), we have

$$\Gamma_g(\mathbf{x}, \mathbf{y}) = \Big(\mu_g - \lambda_g\Big)^{-1}, \forall g \in G \tag{3.10}$$

$$\Gamma_{v_i}(\mathbf{x}) = \Big(\sum_{n \in N} x^n_{v_i} \mu^n_v - \lambda_{v_i}(\mathbf{y})\Big)^{-1}, \forall v \in V \tag{3.11}$$

Assuming that all the VNFs in the same cloud are incurred the same delay of communicating with external entities and the delay between a gateway and a sensor is negligible to be ignored. The delay of a c's control message from a IoT-based VNF instance v_i, if exists, to its sensor through g is

$$L^{ctl}_{c,v_i}(\mathbf{x}, \mathbf{y}) = \sum_{\substack{(g,n) \in E \\ v \in V_g}} \tau_v x^n_{v_i} (\Gamma_g + l_{g,n}) \tag{3.12}$$

Next, given two VNFs v and v', the following is the formulation of the inter-network delay between their hosting clouds

$$L^{com}_{v,v'}(\mathbf{x}) = \sum_{(n,n') \in E} \mathbf{x}^n_v \mathbf{x}^{n'}_{v'} l_{n,n'}, \forall v, v' \in V \tag{3.13}$$

Given a source and multiple destinations, the total delay for a service chain is the maximum delay for transmitting a message to all the destination nodes which must not be greater than the maximum tolerated latency Φ^L_c. As a result

$$L_c = \max_{v \in d(c)} \sum_{u \in \vec{V}_{c \to v}} \sum_{1 \le i \le \kappa_u} \Big(\sum_{w \in \vec{V}_{c \to v}} y^c_{u_i} \beta^c_{u,w} L^{com}_{u,w} + y^c_{u_i} \Gamma_{u_i} \Big) + \sum_{1 \le j \le \kappa_v} y^c_{v_j} L^{ctl}_{c,v_j} \le \Phi^L_c, \forall c \in C$$

In Equ. (4.6), L_c is composed of the transmission latency between every pair of VNFs, i.e. the first term inside the brackets, the time for each VNF to process the message, i.e. the second term at the next line, as well as the time for the last node to activate its corresponding sensor, i.e. the last term.

3.4.4 System Cost

In this paper, we also consider total system cost which is the weighted sum of the cost of allocated network bandwidth (M^{net}) and that of computing resource (M^{com}), that is

$$M^{sys} = \omega M^{net} + (1-\omega)M^{com} = \omega \sum_{(q,q')\in E} B_{q,q'} m^{net}_{q,q'} + (1-\omega) \sum_{n\in N} R_n m^{com}_n \tag{3.14}$$

3.4.5 Problem Formulation

Let $\alpha_{g,n} \in \{0,1\}$ represent the connection between g and n. Based on above analysis, IoT VNF placement problem is formulated as the following constrained optimization, i.e. by i, j indicate the instances' indices of VNFs v and u, respectively:

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \quad M^{sys} = \omega M^{net} + (1-\omega)M^{com} \tag{3.15}$$

subject to (4.2), (4.4), (4.3), (4.14), (3.9), (4.6)

$$(\forall v \in V_g, 1 \le i \le \kappa_v) : x^n_{v_i} \le \alpha_{g,n} \tag{3.16}$$

$$(\forall v \in V_g, 1 \le i \le \kappa_v) : \sum_{n\in N} x^n_{v_i} \le min(\kappa_v, \sum_{m\in N} \alpha_{g,m}) \tag{3.17}$$

$$(\forall c \in C, u \in V_c) : \sum_{1\le j\le \kappa_u} y^c_{u_j} = 1 \tag{3.18}$$

$$(\forall (u,v) \in V, 1 \le i \le \kappa_v, 1 \le j \le \kappa_u) : x^n_{v_i} \in \{0,1\}, y^c_{u_j} \in \{0,1\} \tag{3.19}$$

Our objective is to find a placement scheme to minimize the total cost incurred in the system. Equ. (4.13) implies that a cloud does not provision the instance of a VNF if the gateway connecting to that instance is not associated with that VNF. In this case, both $x_{v_i}^n$ and $\alpha_{g,n}$ are set to zero. If the gateway is associated with n, $\alpha_{g,n}$ is set to 1 and $x_{v_i}^n$ can be a free variable. Moreover, the number of deployed VNF instances must not exceed the number of connections between its associated gateways and the clouds as specified by constraint (3.17). Equ. (4.10) stipulates a VNF cannot be involved more than one time by a service chain and so do its instances.

3.5 IoT Topology-Aware VNF Placemenet

The problem (5.19) is NP-hard and it is difficult to obtain an exact solution in the polynomial time. Hence, a Markov-based approximation (MA) framework Chen, Liew, Shao & Kai (2013) is adopted to find a near-optimal solution within an acceptable period of time. In this section, we present multistart and batching techniques that are implemented regarding IoT topology. We explain how these techniques are incorporated with MA framework, *a.k.a* MBMAP, to address slow convergence drawback.

3.5.1 Batching Markov Approximation Framework

3.5.1.1 Log-sum-exp Approximation

Let $f = \{\mathbf{x}, \mathbf{y}\}$ indicate a specific VNFs placing scheme and $\mathcal{F}$ be the set of feasible configuration defined by constraints of problem (5.19). A change of any VNF instance either allocated at a cloud or a service chain will lead to another configuration or new state in the context of Markov chain. Let M_f^{sys} denote system cost under a configuration f. The problem (5.19) is re-written as follows:

$$\underset{\mathbf{p}\geq 0}{\text{minimize}} \sum_{f\in\mathcal{F}} p_f M_f^{sys} \tag{3.20}$$

$$\text{s.t.} \sum_{f\in\mathcal{F}} p_f = 1 \tag{3.21}$$

where p_f is the probability of choosing configuration f. Adopting log-sum-exponential approximation approach in Chen *et al.* (2013), the problem (3.20) is approximated as

$$\underset{\mathbf{p}\geq 0}{\text{minimize}} \sum_{f\in\mathcal{F}} p_f M_f^{sys} + \frac{1}{\delta}\sum_{f\in\mathcal{F}} p_f log(p_f) \tag{3.22}$$

$$\text{subject to} \sum_{f\in\mathcal{F}} p_f = 1 \tag{3.23}$$

where δ is a positive constant and a gap upper-bound by $\frac{1}{\delta}log|\mathcal{F}|$.

By solving the Karush-Kuhn-Tucker (KKT) conditions of the problem (3.22), we obtain the optimal and close-form probability solution, that is

$$p^*(\mathbf{M}_f^{sys}) = \frac{exp(-\delta M_f^{sys})}{\sum_{f'\in\mathcal{F}} exp(-\delta M_{f'}^{sys})}, \forall f \in \mathcal{F} \tag{3.24}$$

Obviously, the more optimal a configuration is chosen for the whole system, the closer the system cost is to the optimal value with the aforementioned gap. However, in order to compute p_f^* for each configuration, it requires to take into account the whole feasible configuration space to compute (3.24), i.e. the sum at the denominator, which is inefficient due to the large solution space $\mathcal{F}$. Instead, a Markov chain is constructed in a way that the stationary distribution of each state is p_f^*. While the existence of such the chain has been already proven in Chen *et al.* (2013), the states and the transition mechanism respecting to a transition probability need to be defined.

3.5.1.2 Markov Chain Construction Procedure

Let two configurations f, f' in $\mathcal{F}$ represent two states of the time-reversible ergodic Markov chain with the stationary probability $p^*(\mathbf{M}_f^{sys})$. The transition probability between f and f', which are $t_{(f\to f')}$ and symmetrically defined $t_{(f'\to f)}$, must satisfy following balanced equation:

$$p^*(\mathbf{M}_f^{sys})t_{(f\to f')} = p^*(\mathbf{M}_{f'}^{sys})t_{(f'\to f)} \tag{3.25}$$

There are many values of $t_{(f \to f')}$ and $t_{(f' \to f)}$ in Equ. (3.25). We choose the following option with $t_{(f \to f')}$ defined symmetrically, which is:

$$t_{(f \to f')} = \rho \; exp\Big(\frac{1}{2}\delta(M_f^{sys} - M_{f'}^{sys})\Big) \tag{3.26}$$

where ρ is a conditional non-negative constant. Intuitively, this can be understood that if a transition results in a lower system cost, i.e. $M_f^{sys} > M_{f'}^{sys}$, the value of $t_{(f \to f')}$ increases and makes the occurrence of f' more likely. A basic procedure to construct a Markov chain toward the stationary distribution is thus given as :

- **Step 1**: Initialize a feasible configure f_0, in terms of placing VNFs instances onto clouds, i.e. $\mathbf{x}_0$ and assigning them to service chains, i.e. $\mathbf{y}_0$. Compute the system cost $M_{f_0}^{sys}$.
- **Step 2**: From f, generate a new VNF placement scheme, in terms of $\mathbf{x}'$ and $\mathbf{y}'$ for a new configuration f' with a corresponding cost $M_{f'}^{sys}$.
- **Step 3**: Compute the transition probability based on Equ. (3.26) and set the best configuration to either the current one f or the newly generated one f'.
- **Step 4**: Go back Step 2 until stopping criteria is met.

3.5.1.3 Multistart and Batching Based Markov Approximation Placement Framework

Our MBMAP framework is designed following several observations. First, an inherent limitation of the Markov method is the slow convergence rate due to the large space of states. In the worst case, an algorithm might go through $O(2^{\sum_{v \in V} \kappa_v (|C|+|N|)})$ states to retrieve the optimal placing scheme of the problem (5.19). In practice, there is typically a stopping criteria to achieve a near-optimal solution within an acceptable time. Therefore, we argue that the more space's size and the number of computation steps are reduced, the "nearer" optimal a solution could be found. For space's size, it can be done by eliminating or fixing variables that do not satisfy constraints. Procedure SPACEREDUCE in Algorithm 3.1 is an example of assigning constant values to a subset of variables. It can be intuitively understood that a VNF instance should not be placed on a cloud that does not connect to the gateway associated with that VNF. By doing so, we ignore states with invalid placements and thus enhance algorithm's performance.

Algorithm 3.1 Solution state reduction procedure - SpaceReduce

```
1 Set instances' number less than that of clouds
2 for v ∈ V, n ∈ N do
3 |   Set g as the gateway associated with v
4 |   if g connects to n then
5 |   |   x^n_{v_i} ← 0, ∀1 ≤ i ≤ κ_v
6 |   end if
7 end for
```

Algorithm 3.2 Efficient computation support procedures - CostDiff

```
1 Input: Current configuration f, previous configuration f′, newly replaced VNFs V′,
    cost difference Δ
2 Output: Cost difference under 2 configurations
3 Set Δ ← 0
4 for v's instance v_i ∈ V′ do
5 |   Set g as the gateway associated with v
6 |   Let n, n′ be clouds connecting to v_i under f, f′
7 |   Δ ← Δ + (m^net_{n′,g} − m^net_{n,g})(b^sen_v + b^out_v)
8 |   Δ ← Δ + B_v(m^com_{n′} r_{n′} − m^com_n r_n)
9 end for
```

Similarly, procedure COSTDIFF illustrates an efficient method to compute cost difference term of transition probability in Equ. (3.26). The idea is to compute the cost associated with each transition and to have it added to the original cost of the current state to obtain the value associated with the newly formed state rather than manually calculating the cost of each state. Note that the loop at line 4 of COSTDIFF can be avoided by performing lines 6-8 upon changing the placement to any VNF instance.

Second, it may take time for a Markov approximation basic procedure to retrieve the feasible configuration at the beginning (Step 1) as well as from another (Step 2). To tackle this issue, we design a batching transition placement heuristic based on the observation that a VNF instance should be placed in a cloud which not only has the most amount of available resources but also is close to that VNF's associated gateway. In other words, the preference on a cloud $n \in N$ varies

for different VNFs considering that cloud's residual resource and the delay with a corresponding IoT gateway. Given v-associated gateway g, we define $\mathcal{P}(v, n)$ as the preferential function on n of v as

$$\mathcal{P}(v, n) = l_{g,n}\Phi_n^R\Phi_{g,n}^B \sum_{n' \in H} \Phi_{n,n'}^B \tag{3.27}$$

Our strategy is illustrated in Algorithm 3.3 with $f = NULL$ to indicate the case of creating initial state and $f \neq NULL$ for the generation of new states from the current one. If it is the first case, all the VNFs in $V' \equiv V$ will be placed in its most preferential network with $nI = 1$. The randomness of the transition is guaranteed by line 5 where only one VNF v is randomly selected and a random number of most preferential networks (line 9-11) are used to place v whenever the procedure BTRANS is invoked. Each placement of the selected VNF on a chosen cloud, which is not done in the current configuration, is considered as a new state (line 13). Note that in the Markov framework, the procedure BTRANS should be repeatedly performed until all the constraints of the problem (5.19) are satisfied.

Third, the Markov approximation method can be accelerated by leveraging the presence of multiple edge clouds in IoT network to deploy a distributed implementation which can be done via several approaches. The most common one is based on partitioning the problem such that the partitions could be run in parallel and then merged. However, this approach is not generalized for the placement problem which may involve different parameters or constraints depending on the applications. Instead, we have controllers at clouds explore the entire solution space in parallel and periodically compare the results. Our basic idea is to extend the basic Markov search strategy using a multi-start and batching approach (MBMAP), instead of performing with only one initial state. The details are provided in Algorithm 3.4 with two procedures, MASTERCTRL for a master controller (MC) and SLAVECTRL for slave ones (SC). At the beginning of MASTERCTRL, the MC generates a list of feasible states (line 4) and assign them as initially starting states to each idle slave controller (line 5-6). After that, the MC moves to a listening state and waits for data from the SCs at line 8 until receiving a certain number of

Algorithm 3.3 Batching transition placement algorithm - BTrans

1 **Input:** *Network topology $\mathcal{G}$, service requests C, set of VNFs V, current configuration f*
2 **Output:** *Set of new states $\mathcal{F}'$*
3 Set $\mathcal{F}' \leftarrow \emptyset, V' \leftarrow V, nI \leftarrow 1$
4 **if** $f \neq NULL$ **then**
5 | Select a random VNF $v \in V$ and set $V' \leftarrow \{v\}$
6 **end if**
7 **for** $v \in V'$ **do**
8 | **if** *V' has more than one VNF* **then**
9 | | Set $nI \leftarrow rand\Big(0, min(\kappa_v, \sum_{n \in N} \alpha_{g,n})\Big)$
10 | **end if**
11 | Let N' be the nI most preferential clouds of v using Equ. (3.27)
12 | **for** $n \in N'$ ***and*** *v not placed on n* **do**
13 | | Create a new state from f with the placement of v on n and add it to $\mathcal{F}'$
14 | **end for**
15 **end for**
16 **return** $\mathcal{F}'$

states. The loop exists if all the SCs complete their tasks (line 9). A set of states which have cost difference less than a threshold Δ are then randomly re-assigned to idle controllers (line 14-15). The lowest cost state f_{min} is also used to generate a batch of new states to assign in case there are still idle SCs. Note that all the potentially "good" states, i.e. f_{min} are tracked by the MC (line 13) and only the one with the lowest cost will be returned at the end of the procedure (line 21). This makes sure that the output is always the best one among those generated by the SCs.

For the SCs in the procedure SLAVECTRL, upon receiving a state f from the MC, a batch of states will be created and sorted in cost descending order (line 3). By doing this, we ensure that the SC preferably takes the state with lower cost into account first to perform the transition. There are two cases occurred at the SC's side. If the transition from f to f' does not happen, then f' will be reported to the MC (line 6) for the tracking purpose. If the transition does not lead to any significant cost improvement after several times, then the SC will restart its operation with a new state by going back to the listening state (line 1).

Algorithm 3.4 Placement Procedure at Master Controller - MasterCtrl

Input: *State distance threshold* Δ
Output: *State with minimum cost*
$S \leftarrow \emptyset$
Generate a batch of $|N|$ feasible states using procedure BTRANS with $f = NULL$
for *newly generated state* f **do**
    Assign f to an idle controller
end for
while *listening slave controllers* **do**
    if *all controllers complete* **then**
        break
    end if
    Let S', f_{min} be the set of received states, the state with the lowest cost, respectively
    $S \leftarrow S \cup \{f_{min}\}$
    for $f \in S'$ ***and*** $dist(f_{min}, f) \leq \Delta$ **do**
        Assign f to a randomly idle controller
    end for
    if *there are still idle controllers* **then**
        Invoke BTRANS to generate new states from f_{min} and assign to idle controllers
    end if
end while
return the minimum cost state in S

3.5.2 Node Ranking-based Placement Heuristic

In order to evaluate MBMAP performance, a node ranking-based placement heuristic (NRP) is proposed. The NRP is developed as a deterministic algorithm based on the BTRAN procedure. In particular, we define the VNF ranking function $\mathcal{R}(v)$ based on the number of VNFs that have connections to v regardless the service chain as follow:

$$\mathcal{R}(v) = \frac{1}{\kappa_v} \sum_{c \in C} \sum_{u \in V} (\beta^c_{u,v} + \beta^c_{v,u}) \tag{3.28}$$

The usage of $\mathcal{R}(v)$ allows the placement process to prioritize VNFs which are more important in terms of the popularity among service chains and the number of instances. NRP procedure is described in Algorithm 3.6 which starts by constructing an ordered VNF list by $\mathcal{R}$ using Equ.

Algorithm 3.5 Placement Procedure at Master Controller - Slave Controller

```
1  while listening master controller do
2  |  Set received state as current state f
3  |  Generate a batch of states from f and sort them in cost descending order
4  |  for f′ of the batch do
5  |  |  if f not transit to f′ then
6  |  |  |  Send f′ to the master controller
7  |  |  end if
8  |  |  else if small cost improvement then
9  |  |  |  Send f′ to the master controller and go back line (1)
10 |  |  end if
11 |  |  else
12 |  |  |  Go back line (3)
13 |  |  end if
14 |  end for
15 end while
```

(3.28). Each VNF v is placed one by one (line 3) onto preferable clouds as long as that cloud has enough bandwidth, i.e. $\Phi_n^R > r_n B_v$, $\Phi_{g,n}^B > b_v^{sen}$, $\Phi_{n,n'}^B > b_v^{out}$ as realized by the condition at line 5. The preference $\mathcal{P}(v,n)$ is updated (line 6) after placing a certain VNF. If none of the preferable clouds has enough resource to host the VNF, the algorithm continues with other clouds (line 9) as an effort to deploy VNFs. After iterating through all the VNFs, the ranking values of unplaced VNFs, if any, are increased by a pre-defined amount (line 3.5.3). As a result, such the unplaced VNFs will be more likely placed on suitable clouds. The feasible solution of the problem (5.19) with its cost is stored at line 15, and the one with the lowest cost will be returned upon meeting the stopping criteria (line 23).

3.5.3 Discussion

In general, NRP is simpler to implement than MBMAP as it mainly relies on ranking functions and sorting procedure. The complexity of each iteration in NRP (line 2-11) is contributed by sorting VNFs at line 2, i.e. $O(|C||V|log(|V|))$, the loop at line 3, i.e. $O(|N|\sum_{v\in V}\kappa_v)$. In the worst case, the condition at line 9 is always reached and the complexity of the loop at line 5 is

Algorithm 3.6 Node ranking-based placement algorithm - NRPlacement

```
1  V' ← ∅
2  Sort VNFs of V in R(V)-descending order
3  for v ∈ sorted V do
4  |   Set nI ← min(κ_v, Σ_{n∈N} α_{g,n}) and let N' be the nI most preferential clouds of v
   |    using Equ. (3.27)
5  |   for n ∈ N', n has enough resource do
6  |   |   Place v on n and update P(v, N) with n's residual resource
7  |   end for
8  |   if v is not placed yet then
9  |   |   N' ← {N\N'}
10 |   |   Go back line 9 if N' is empty
11 |   |   V' ← V' ∪ {v}
12 |   end if
13 end for
14 while stopping criteria is not met do
15 |   if constraints are satisfied then
16 |   |   Store current scheme with its cost
17 |   end if
18 |   for v ∈ V' do
19 |   |   Increase R(v) by a pre-defined parameter
20 |   end for
21 |   Go back line 2
22 end while
23 return scheme with lowest cost stored at line 16
```

$O(|N|)$. Note that the advantage of NRP lies in its fast convergence speed with the much lower number of feasible states. Its limitation is to easily get stuck in local optimum due to greedily place VNFs until all the constraints are satisfied. A trigger at line in Alg. 3.6 is not enough to make a significant "jump" regarding the ranking difference between nodes.

From the implementation perspective, several options can be considered for MBMAP, i.e. MC/SC selection, batch's size. In Alg. 3.4, the MC's operations include, i) to keep track of states generated from the SCs and assign them to other idle SCs and ii) to generate new states only if there are not enough states to assign. The more states a SC generates, i.e. the more processing resource the SC requires, the less possibility the MC invokes BTRANS procedure.

In other words, the larger batch of states the BTRANS procedure generates, the less resource the MC requires to manipulate states, the more powerful the SCs are and consequently more resource in total is allocated for controllers since the number of SCs is typically higher than that of MCs. However, regarding the convergence speed, a large batch's size enables MBMAP to explore more candidate solutions and thus faster at discovering the optimal solution. Similarly, there is also a trade-off in setting the number of controllers between the allocated resource and the purpose of driving the algorithm into new regions of the solution space. One way to deal with the parameter is to start with several controllers to encourage the exploration of solutions near a local optimum and add more to push the search out of that local region based on some stopping criteria.

In the worst case, MBMAP might go through the entire space of up to $|\mathcal{E}| = O(2^{\sum_{v \in V} \kappa_v (|C|+|N|)})$ states. Every BTRANS invocation requires $O(1)$ step to transit between two states and $O(|C||N|^2|V|^2)$ steps to validate the new state. As a Markov-based approach, MBMAP is approximated by an entropy term $\frac{1}{\delta} \sum_{e \in \mathcal{E}} p_e log(p_e)$. The gap is therefore computed as $\frac{1}{\delta} log|\mathcal{E}|$, or $O(|V|logM)/\delta$. As pointed out in Chen *et al.* (2013), besides the batch's size and the number of controllers, δ is another parameter that can be adjusted as a trade-off between the requirement of fast convergence as well as small optimality gap and the system performance.

3.6 Performance evaluation

This section presents the performance analysis of proposed model. We assess the applicability of our VNF placement solution by comparing it with other solutions that do not consider IoT network characteristics, i.e. the term $l_{g,n}$ is ignored in the Equ. 3.27, or the multistart and batching techniques, are not adopted in MBMAP.

Table 3.2 Simulation parameters

Parameter	**Value**
Number of VNFs	70
Number of clouds	8
Number of IoT gateways	15
VNF instances (κ_v)	(4, 8)
Service arrival rate λ_c (ms^{-1})	(0.1, 0.9)
Service rate $\mu^n_{v_i}$, μ_g (ms^{-1})	(0.1, 0.3)
Sensor data rate λ^{sen}_v (ms^{-1})	(0.5, 1.0)
Bandwidth b^{out}_v & b^{sen}_v (Mbps)	(1.5, 2.5) & (0.3, 0.7)
Link latency l^n_g & $l^{n'}_n$ (ms)	(1.4, 0.02) & (1, 0.02)

3.6.1 Simulation Analysis

3.6.1.1 Simulation Settings

We build the simulation with 100 VNFs that are placed onto a fully meshed network topology of 8 clouds and 15 IoT gateways. A VNF can be replicated from 4 to 8 instances. Each gateway is configured to connect to a cloud with a probability 0.8 and is uniformly assigned to handle several sensors of 40 IoT-enabled VNFs. The simulation is performed on service chains with 6 VNFs as illustrated in section 3.3.3. Each chain consists of a single source node and from 1 to 6 destination nodes. Maximum tolerable service latency is set to 45ms.

From the deployment perspective, input data in terms of traffic rates from sensors is periodically generated at the rate λ^{sen}_v while service requests arrives according to a Poisson distribution with mean λ_c. The NRP algorithm is deployed at only one cloud and its output is applied to other clouds. For MBMAP algorithm, the connections between MCs and SCs are pre-established and maintained during the performance of the algorithm. From such input data, a MC script implements Alg. 3.1 to initialize a state composing of adjacency matrices that represent the placement of VNFs onto clouds and the assignment of VNFs instances to service chains

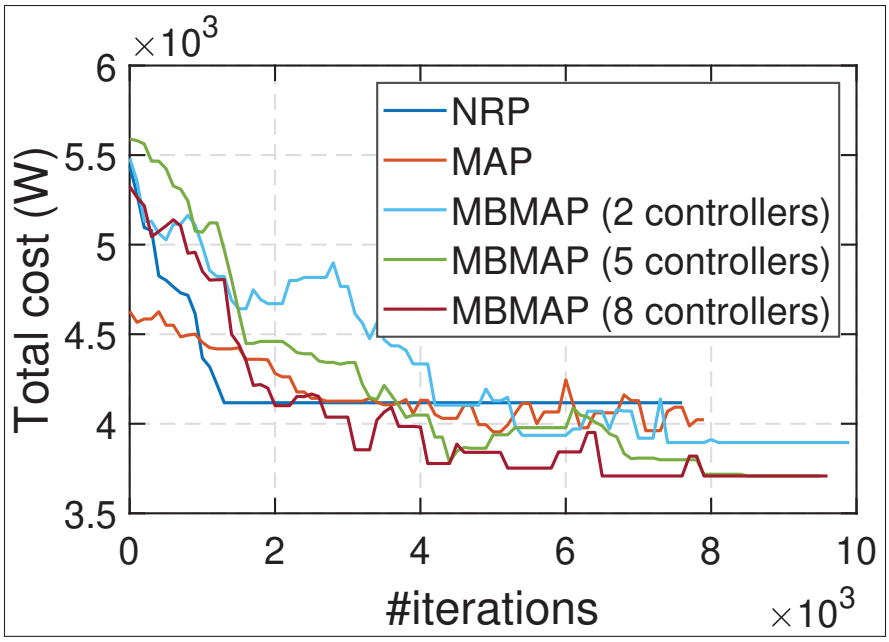


Figure 3.6 Evaluation of convergence of proposed algorithms

according to Alg. 3.4. It also calculates a batch of states by using BTRANS procedure and send them to SCs whenever there are idle SCs. A script at the SC performs a basic procedure combining with a batching technique and sends back to the MC the state with the lowest cost using Alg. 3.5.

To evaluate the effect of the IoT network on VNF placement decision making, we define an IoT density as the ratio between the number of IoT-based VNFs and the total number of VNFs. We consider two density levels, i.e. low, and high with the ratios 0.1, 0.7 respectively. Simulation parameters are summarized in Table 5.4. The value of r_n ranges between 2 and 4. System cost is considered from the aspect of power consumption (Watt). According to Nonde, Elgorashi & Elmirgahni (2016), the power consumption by a router port supporting 1Gbps connection speed is about 21.25W and 11.25W to run a CPU per hour. For the purpose of comparison, normalized unit costs of computing resource and bandwidth are set to 1W and 2W respectively.

3.6.2 Simulation Results

We next present our simulation results on our proposed MBMAP framework and NRP from three aspects, namely convergence time, system cost and resource utilization. The simulation is performed through time slots during which the controllers receive different service demands and makes a decision of placing VNFs. It is assumed that during each slot, system configuration parameters, e.g. network topology, physical/virtual node settings, etc., remains unchanged. The algorithm is assumed to converge during this slot and the deployment of VNFs is performed in the remaining time of the slot.

3.6.2.1 Convergence

We investigate the convergence of the proposed algorithms including MBMAP with different numbers of controllers, NRP and basic MAP. Fig. 3.6 shows that NRP converges very fast and returns the solution after several iterations. This is due to NRP mainly depends on ranking functions to retrieve an optimal placement. In contrast, it takes more time for Markov-based approaches, i.e. MBMAP and MAP, to converge toward an optimal result, especially with a large space of states. Unlike MAP, MBMAP leverages the presence of multiple controllers at each IoT edge clouds to implement the multistart and batching technique. It not only allows MBMAP to explore more potential states but also prevents MBMAP to get trapped forever at a locally optimal solution. As a result, our proposed mechanism can converge faster than MAP within 500 iterations and approximate the optimal solution as the number of controllers increases.

3.6.2.2 System Cost

In Fig. 3.7, we run all the algorithms on 100 service chains by varying ω from 0.1 to 0.9 to see how cost components, i.e. M^{net}, M^{com} are affected. The results show that NRP and baseline have more impact on the computation cost and as a result, the improvement of the network cost is very limited even when emphasizing the importance of the network traffic cost (i.e. $\omega = 0.1$). This is due to NRP and the baseline relies on the function $\mathcal{P}$ which is attributed

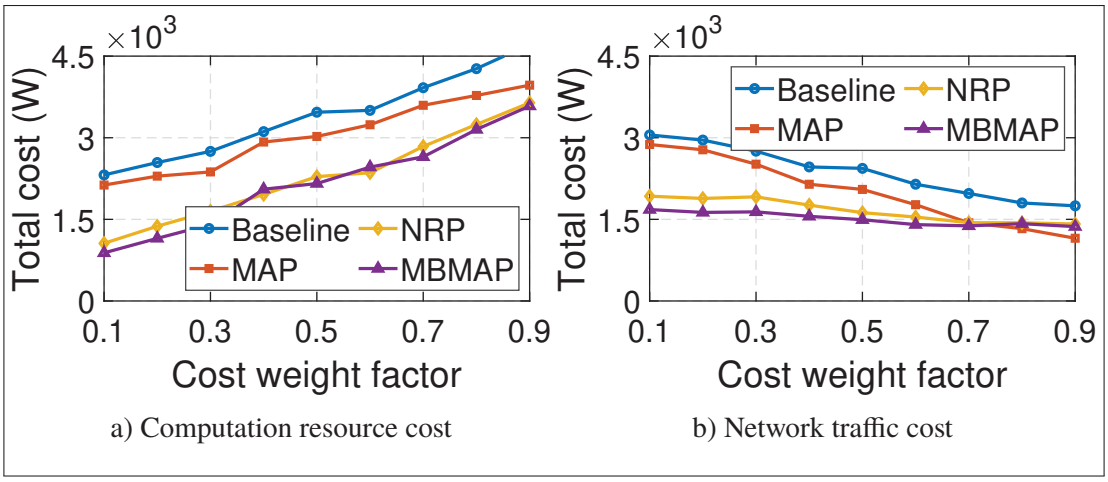


Figure 3.7 Cost component comparison with different cost weight factors

more by the computation resource than the network resource. MBMAP and MAP jointly control the computation cost and the network traffic cost in a more dynamic way and therefore obtain the lower total cost in all considered cost importance. From Fig. 3.7, the approaches obtains the balance between computing and bandwidth cost at various ω, i.e. 0.3, 0.32, 0.33 and 0.35 for the baseline, MAP, NRP and MBMAP respectively. Regarding the difference between cost components, these ω's values can be seen as Pareto optimal solutions. However, for the purpose of simplicity, we set ω to the average value 0.33 so that the cost difference incurred by different approaches is not significant to avoid extreme cases.

We next evaluate the total cost incurred by using placement approaches given different parameters, i.e. the number of service chains or service arrival rate λ_c, IoT density levels and cost's weight factor ω. In Fig. 3.8, the MAP approach adopts the standard Markov-approximation framework as described in section 3.5.1.2 and the baseline is a ranking-based heuristic like NRP but excludes the delay parameter $l_{g,n}$ from Equ. (3.27). We can observe that such the exclusion induces a significant gap in system cost between the baseline and the other strategies, especially when more VNFs related to IoT devices present in the system. Three remaining algorithms are comparable to each other, i.e. 10 - 30 service chains with a negligible cost difference. However, at a high load of more than 40 service chains, MBMAP steps out of the others with a reduction of 13.8% on the total cost. To analyze this difference between the algorithms, we investigate the CDF

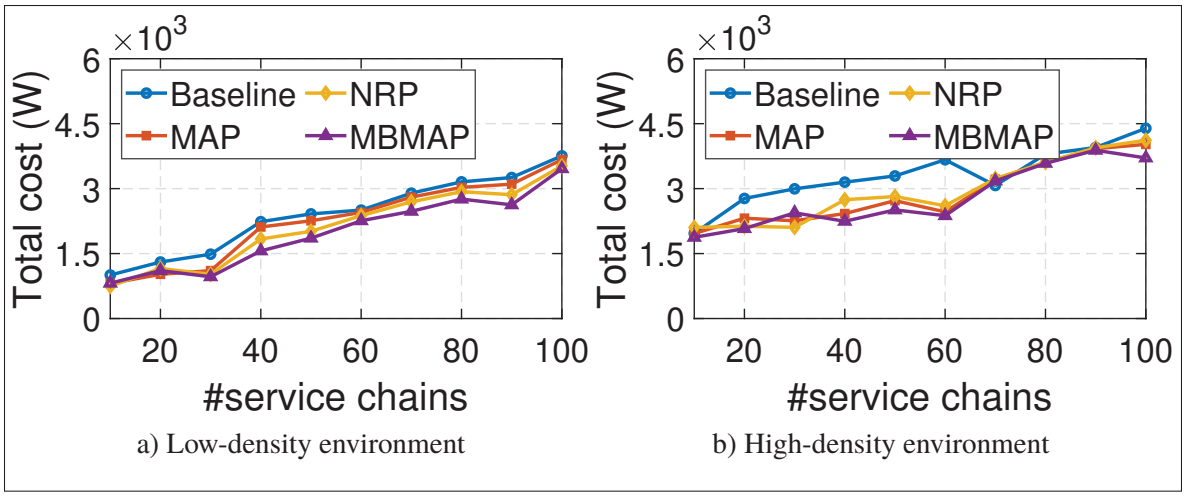


Figure 3.8 Cost comparison with different level of IoT density

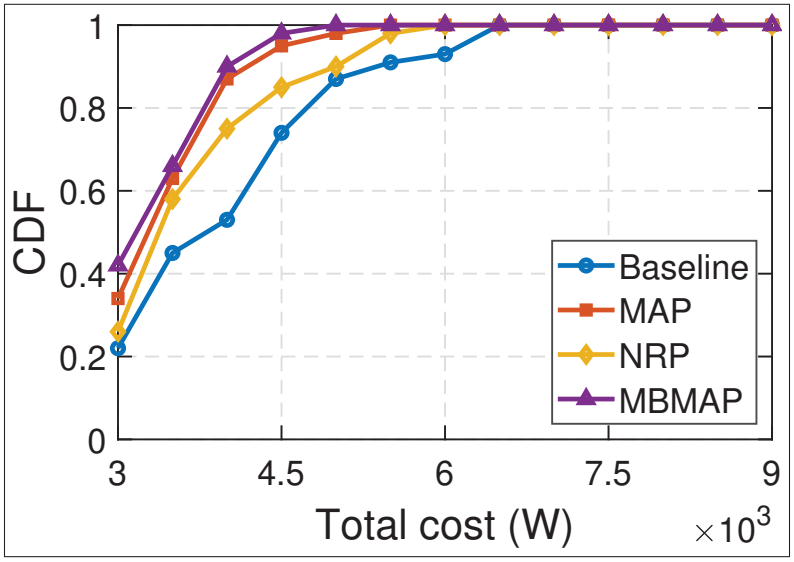


Figure 3.9 Distribution of system cost by service rates

of system cost across different service demands. Fig. 3.9 shows that MBMAP overlaps with MAP, which indicates how close these approaches are. From the perspective of Markov chain, it guarantees that the combination of multistart and batching techniques into the original Markov approximation framework does not break Markov property when constructing Markov chain. On the other hand, NRP results in a better cost than the baseline and this matches with the results of component costs in Fig. 3.7.

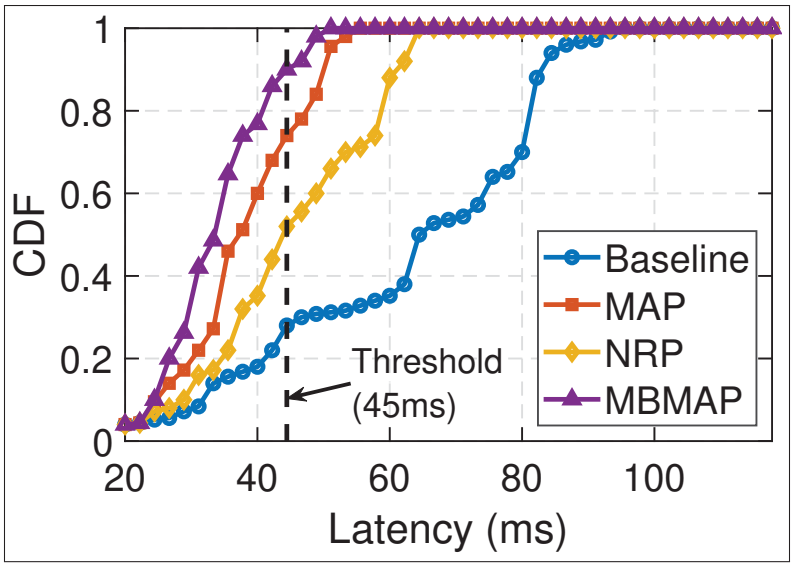


Figure 3.10 Distribution of service latency by service rates

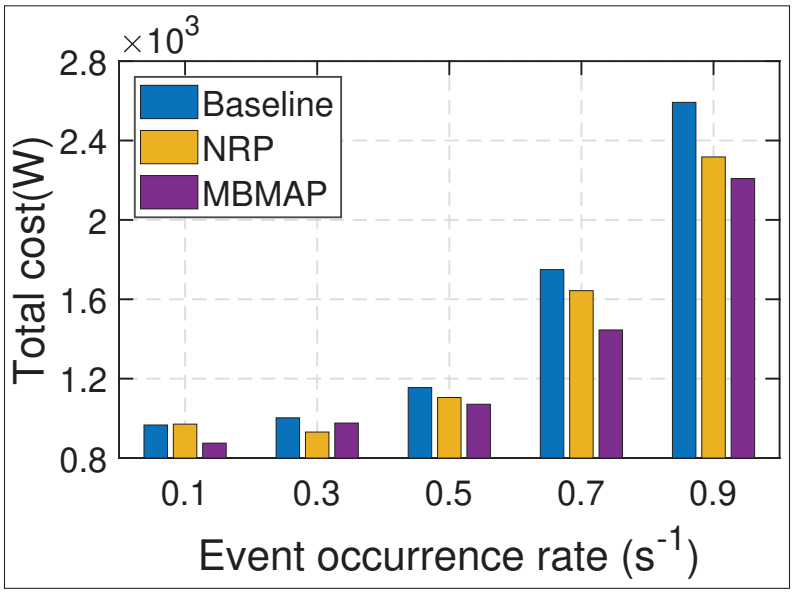


Figure 3.11 Evaluation of total system cost

3.6.2.3 Service Latency

We perform the analysis under the high-density condition because it is close to the practical environment in which some VNFs are IoT-based entities and some are not. This setting is

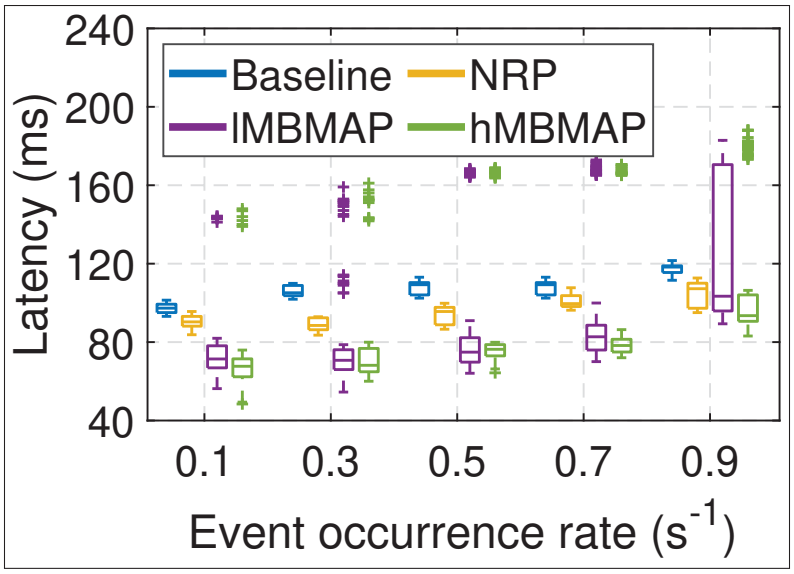


Figure 3.12 Surveillance session setup latency

used in the rest of the paper, except where the differentiation is required. To understand the performance of proposed algorithms on Quality of Service (QoS), we plot the CDF of service latency across different service arrival rate and the number of service chains. As can be seen in Fig. 3.10, MBMAP and MAP result in better latency than NRP and the baseline. More than 90% service requests are served by MBMAP with latency less than the threshold. For other algorithms, this value is 78% with MAP, 53% with NRP, and 29% with the baseline. Notice that the number of IoT end-points does not have a significant impact on service latency as the difference of service latency between a 1-target chain and a 6-targets chain is small. This is because the latency is computed as the maximum value among those between source nodes and all the destination nodes. In contrast, the length of service chains affects not only service latency but also demonstrates the improvement of the proposed algorithms. With "longer" service chains, there are likely more instances of each VNF that need to be allocated and therefore resulting to more feasible options of placement (for variables $x^n_{v_i}$) and assignment (for variables $x^c_{v_i}$) instances to a chain even with the same length. Simple heuristics like NRP or the baseline do not leverage this fact to improve their result whereas methods like MBMAP and MAP exploit the introduction of new feasible solutions to obtain a more optimal placement scheme.

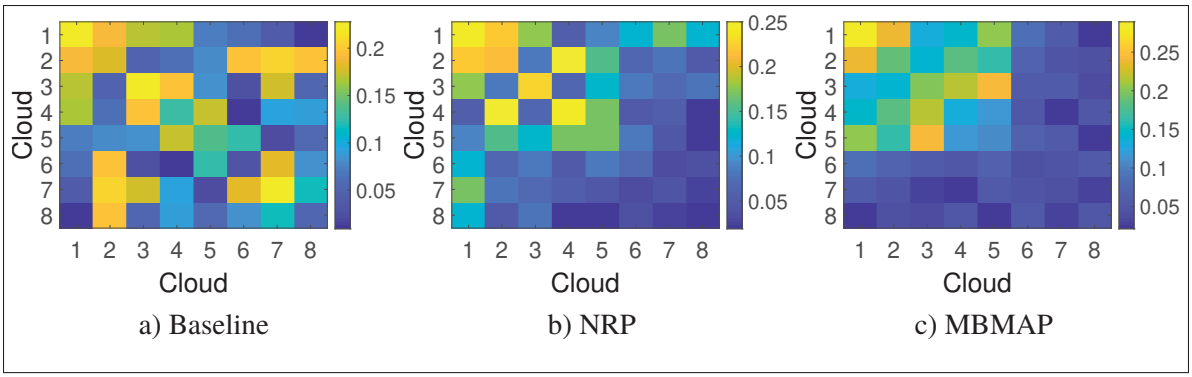


Figure 3.13 Evaluation of link utilization

3.6.3 Experimental Analysis

3.6.3.1 Testbed Settings

The experimental analysis is conducted on the basis of communication sessions between IoT endpoints as shown in Fig. 3.3. The set N is composed of 8 clouds that are realized by 8 blade servers each of which has 24 physical CPUs and 96GB of memory. VNFs are implemented via Virtual Machines (VMs) configured with different settings depending on the requirements of corresponding service functions, i.e. between $2 \sim 4$ virtual CPU (vCPU) and $4 \sim 16$ GB virtual memory (vMem) as detailed in Table 4.3. This configuration for heavy tasks is reasonable and has been used in several related works, i.e. Wang, Pan & Esposito (2017a).

Table 3.3 VNF Resource Configuration

Service functions	**(vCPU, vMem, #instances)**
Motion Analyzer	(2, 8, 4)
Video Processor	(4, 16, 3)
Decision Maker	(2, 4, 4)
Dispatcher	(2, 4, 1)
Web Server	(4, 8, 2)
Mobile Proxy	(2, 8, 5)

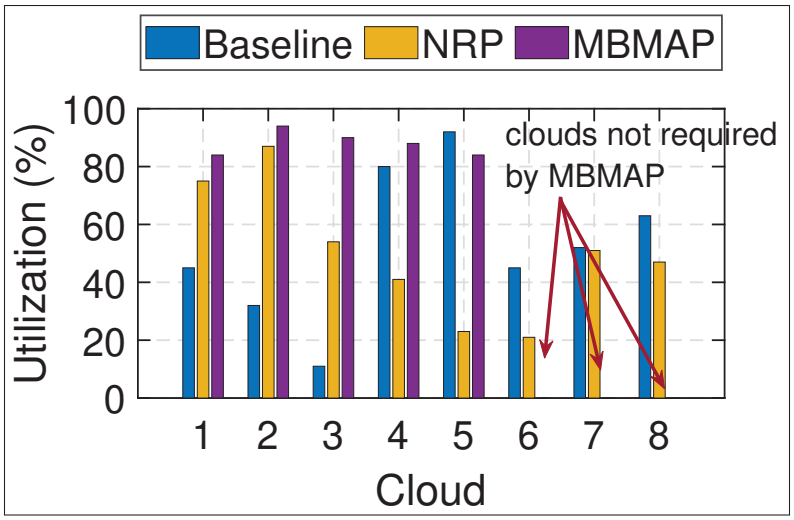


Figure 3.14 Evaluation of resource utilization over clouds

The number of VMs is limited to not cause over 85% CPU usage in order to guarantee the system performance. According to the testbed scenario, the set V is composed of one standalone VNF, i.e. Decision Maker, and five IoT-enabled VNFs. Each VNF has from 1 to 6 instances. Eight controllers in MBMAP are deployed along with VMs for VNFs on all the blade servers.

Toward IoT side, 6 OpenWRT-based Access Points (APs) are set up as IoT gateways that handle traffic from 3 sensors, 2 laptops and 3 mobile phones. A script is deployed at the AP to control the transmission of sensor data towards corresponding VNFs. Without affecting the final result, a script is programmed to send data at specific time to represent the occurrence of an intrusion which causes a significant difference of recorded data between two consecutive moments. Network bandwidth between gateways and clouds are pre-configured by APs while the delay is managed by scripts at blade servers.

3.6.3.2 Experimental Results

To show the advantage of our proposed method, we measure the total latency of surveillance service. The event occurrence rate varies from 0.1 to 0.9 according to Poisson distribution during 10 time slots to represent service demand on the network. To show how the convergence

of the algorithms affect the overall QoS, we perform the experiments under two conditions, i.e. high and low rate change of occurrence rate, with the duration of time slots 1 and 10 minutes respectively. Accordingly, a proxy is deployed to hold packets from the source node until the placement scheme is obtained by the algorithm. Fig. 3.11 shows that while MBMAP and NRP obtain scheme at a lower cost, i.e. $11 \sim 21\%$ the baseline. However, in Fig. 3.12, MBMAP causes a long delay for sessions occurred at the beginning of each slot. In the case of 1-minute slots (hMBMAP), the extremely long sessions represented as outliers significantly affect the latency median and make this value higher a bit than that of 10-minute slots (lMBMAP). Especially, at the event rate $0.9s^{-1}$, lMBMAP results in a higher variation of sessions' delays and more skewed data than other approaches as well as hMBMAP. In addition, at both of time slot's durations, with the execution time approximating 0 due to the small size of input data, NRP and the baseline barely induce any overhead to the hypervisors and therefore the performance of deployed VMs as the MBMAP's controllers do. Note that the MAP algorithm does not appear in Fig. 3.12 because its performance is comparable to MBMAP regarding the small-scale testbed.

Regarding resource utilization, Fig. 3.13 represents the link utilization between all pairs of clouds and Fig. 3.14 illustrate how computing resource is distributed across the clouds (or blade servers). By using the baseline Fig. 3.13a, VNFs are placed onto all the 8 clouds and thus entails the utilization of network bandwidth at every link connecting them. In Fig. 3.13b, the communication traffic barely go through the links between clouds 7, 8 with clouds 5,6. In contrast, as shown in Fig. 3.13c, most of the network usage is concentrated at some clouds for MBMAP. Unlike prior approaches that try to place VNFs on the same place as much as possible, our algorithm places them in accordance with the impact of the IoT gateways. Correspondingly, computation resource is over-provisioned by the baseline since all the clouds are active but operating with less than 80% allocated resource. For NRP, even though the resource is used more efficiently with more than 85% of resource utilized at clouds 1, 2, and 4, there is a large bias in the amount of virtual resource between them and the other clouds. For example, the 6^{th} cloud needs only 31% CPU usage whereas the 7^{th}, 8^{th} asks for 12% and 9%. On the other hand,

MBMAP requires only 5 clouds with a more efficient mechanism of provisioning in such a way that blade servers are fully used with the CPU utilization close to 93%.

3.7 Conclusion

This paper studies the VNF optimal placement problem in NFV-based edge cloud systems taking IoT network topology into consideration. We consider IoT service chains composed of multiple VNFs that are geographically deployed onto edge clouds close to IoT endpoints. The VNFs communicate not only with each other but also with IoT gateways that typically aggregate data from IoT sensor network as contextual information into discrete messages and forward them toward VNFs at the server side. We define an analytical model of system cost in terms of computation resource and network bandwidth with regard to service latency and the availability of each resource at edge clouds. We then formulate the problem of minimizing the total system cost with respect to constraints on available resource and QoS requirements. To obtain an optimal placement solution, two algorithms for small and large-scale network settings are proposed respectively, namely a Markov-based approximation approach that leverages the presence of multiple edge cloud to adopt multistart and batching techniques, and a node ranking heuristic.

We implement these two algorithms and validate their performance via simulation and testbed. The testbed is configured according to an IoT-base surveillance use case. The results show that with the consideration of IoT network topology in making VNF placement decision can save on system cost up to 21% depending on the size of the network.

In future, we will take into account the mobility of IoT devices that requires to update the proposed model to reflect the dynamic connection between VNF and IoT gateways. The online placement algorithm in this situation is needed to handle highly dynamic IoT network change.

CHAPTER 4

JOINTLY OPTIMIZED RESOURCE ALLOCATION FOR SDN CONTROL AND FORWARDING PLANES IN EDGE-CLOUD SDN-BASED NETWORKS

Duong Tuan Nguyen[1] , Chuan Pham[1] , Kim Khoa Nguyen[1] , Mohamed Cheriet[1]

[1] Department of Automation Production, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Article submitted for publication, December 2021.

Abstract

The emergence of Software-Defined Networking (SDN) paradigm and Network Function Virtualization (NFV) technology enables the implementation of novel edge-cloud SDN-based network infrastructure and accelerates the deployment of services with real-time requirements. However, it is challenging to achieve a jointly optimal resource placement on both SDN controller and forwarding planes in edge-cloud SDN-based networks. In this paper, we jointly address two issues, that are SDN controller-switch assignment and Virtual Network Function (VNF) placement. We propose a model and formulate the joint optimization problem of dynamically provisioning resource for SDN controllers and VNFs. To cope with fluctuate service demand, it is vital to maintain the stability of the entire system composed of controllers, forwarding elements, i.e. switches, and physical machines where VNFs are hosted. We thus apply the Lyapunov optimization framework to transform a long-term optimization problem into a series of real-time problems and employ the exponentiated gradient ascent method to find a near-optimal solution. Based on simulation and experiments, the results obtained show that our approach improves the system cost up to 11 $\sim$ 40% depending on service demand and network size.

Keywords: Resource allocation, VNF placement, SDN controller-switch assignment, IoT service, Lyapunov optimization, Exponentiated gradient ascent.

4.1 Introduction

Emerging distributed cloud architecture, such as fog/edge computing enables to shift a significant amount of computing functionality toward the edge of the network, close to end-users. Being made up for the shortcomings of centralized cloud computing model, this paradigm is a key enabler for the deployment of both computation-intensive and delay-sensitive IoT applications, such as real-time communication services or video surveillance Chen, Li, Deng, Li & Yu (2019). Edge computing allows requests from end-users' equipment to be processed by placing responsible application modules or service functions in proximity and consequently reducing the pressure on network bandwidth utilization with the lowest latency. Specifically, with the collaboration of edge and cloud networks, service requests can be elastically executed at either core cloud or edge server depending on their characteristics, i.e. complex/simple computing, high/low latency requirement Pan & McElhannon (2018).

In fact, resource management remains challenging in edge-cloud network regarding to a diverse set of IoT services with dynamic service demand and the increasing real-time requirements Fang & Ma (2021). Thanks to novel softwarization technologies such as SDN, NFV, this task is facilitated in the sense that network functions are programmable and centralized at control entities. Hence, resources required by IoT services can be easily and quickly configured and deployed. However, determining how the resource is allocated optimally in real-time is a non-trivial task. Not only virtual network functions (VNF) serving service requests need to be dynamically placed on physical servers at either edge or core cloud, *a.k.a* VNF placement problem (VPP) Liu, Guo, Liu & Yang (2021), but also it requires to efficiently map SDN controllers to switches that forward traffics between VNF in order to reduce flow setup latency, *a.k.a* SDN controller-switch assignment problem (SCSA) Guo, Zhang, Feng, Wu & Lan (2020).

To the best of our knowledge, while both VPP and SCSA have been well researched, jointly solving them to obtain a holistic resource allocation strategy at control and forwarding planes has been not addressed in any of prior works. It turns out to be a very difficult problem regarding the complicated interaction between relevant systems, i.e., network controllers, cloud platform,

container orchestrator to handle multiple service function chains and the increasing size of the optimization problem with additional variables as well as constraints. As a result, it is harder to solve in a real-time manner with the dynamic service demand while ensuring the stability at control plane.

The paper's contribution is two-fold. First, we propose a model for the joint optimization problem of allocating resource for both SDN controller and VNFs placement (SCVP) and formulate our objective as an Integer Programming (IP) problem. Apart from jointly solving SCSA and VPP, we go a step further than prior works by taking real-time requirement into account which is a common characteristic of IoT smart services. The model considers not only dynamic service demand and its impact on E2E service latency but also the delay induced by the communication from the switch that handling VNFs' traffic to its SDN controller, i.e. setup flow delay. Second, to address dynamic service demand while maintaining the entire system stability, we apply Lyapunov optimization framework Neely (2010) to transform this original problem into a series of real-time problems without a *priori* knowledge. We then employ an Exponential Gradient Ascent method on transformed problems, whose performance in countering the exponential growth of network size has been assessed by Vigneri, Paschos & Mertikopoulos (2019). Via simulation and experiments, the results show that our proposed system outperforms previous approaches regarding the presence of heterogeneous applications composed as service function chains with dynamic service demands.

The remainder of the paper is organized as follows. After reviewing related work in Section II, we introduce the model and the formulation of SCVP problem in Section IV. Section V explains the proposed algorithm and section VI provides a performance evaluation of our methods with simulation settings. Finally, conclusions are drawn.

4.2 Related Work

In this section, we provide an overview of the state-of-the-art solutions relevant to VNF and SDN controller placement in edge-cloud SDN-based network with respect to real-time service requirements.

4.2.1 VNF Resource Placement

VNF placement problem (VPP) has been studied in many published papers. Closest to our network system model, edge-cloud SDN-enabled network system, is the work on dynamically placing VNFs and routing service chain's traffic in Liu *et al.* (2021). The authors target the problem with the aim of achieving network load balance, reducing delay, packet loss and jitter in the network. A multi-stage graph-based heuristic algorithm is proposed without however considering the impact of the mapping between SDN controller and switch on flow setup latency.

Regarding the deployment of extreme low-latency services for Internet-of-Things (IoT) applications, Xu *et al.* Xu *et al.* (2020) present a QoS-aware solution for VPP in multi-tier mobile edge networks. Varasteh *et al.* Varasteh *et al.* (2021) take into account the power consumption and resource constraints of physical machines and forwarding switches in the power-aware and propose a fast heuristic framework to solve VPP in an online manner. constrained joint VPP. In Poularakis, Llorca, Tulino, Taylor & Tassiulas (2020), service placement and request routing and mobile edge-cloud network is studied with consideration to multidimensional resource requirements, i.e. storage, computation, communication constraints. In Taleb *et al.* (2015), the authors deal with service usage behavioral patterns of mobile users in obtaining an optimal scheme of allocating VNFs resource. Cziva *et al.* (2018) takes into account the dynamic nature of IoT networks apart from user mobility and presents a dynamic scheduler of placing VNFs on a distributed edge infrastructure. It is typically assumed that service components are implemented as VNFs. The joint VNF placement and path selection problem is thus investigated in Kuo *et al.* (2016) from the service chain perspective regarding the relation between link and server

usage. In Bhamare *et al.* (2017), analytical models for service function placement are adopted in multi-cloud environments.

4.2.2 SDN Controller Placement

Similar efforts have been explored in finding an optimal placement scheme for SDN controllers. In Guo *et al.* (2020), Guo *et al.* introduce a heuristic solution for SCSA that optimizes the controller resource demand. Unlike existing works that calculate the assignment mainly based on the number of flow path setup requests generated from the switches, the authors analyze the diversity of the resource demand for flow path setup requests. The decision is made based on the fact that resource demand at control plane can be reduced if the switches on the path can be assigned to fewer controllers.

The work Dixit *et al.* (2013) presents an architecture that allows optimally placing switches under the management of elastic distributed SDN controllers. The authors in Wang *et al.* (2017) present a dynamic scheme to assign switches to controllers to balance the controller load while keeping the control traffic overhead low. Switch migration schemes are proposed to minimize the migration cost Xu *et al.* (2019) or to reduce flow setup time Ye *et al.* (2017).

However, all the studies are specificially to target either VPP or SCSA. In this work, we jointly solve both in a real-time manner. We take into account the fluctuation of service demand from a set of service function chains while maintaining the stability at control plane which is the vital part in SDN networks.

4.3 System Model and Problem Formulation

4.3.1 Network System Model

In this work, we study an SDN-based edge-cloud network that consists of one core cloud and a set of edge nodes as shown in Fig. 4.1. Without loss of generality, we suppose that both edge and core clouds operate as a single server with co-located physical resources. VNFs are

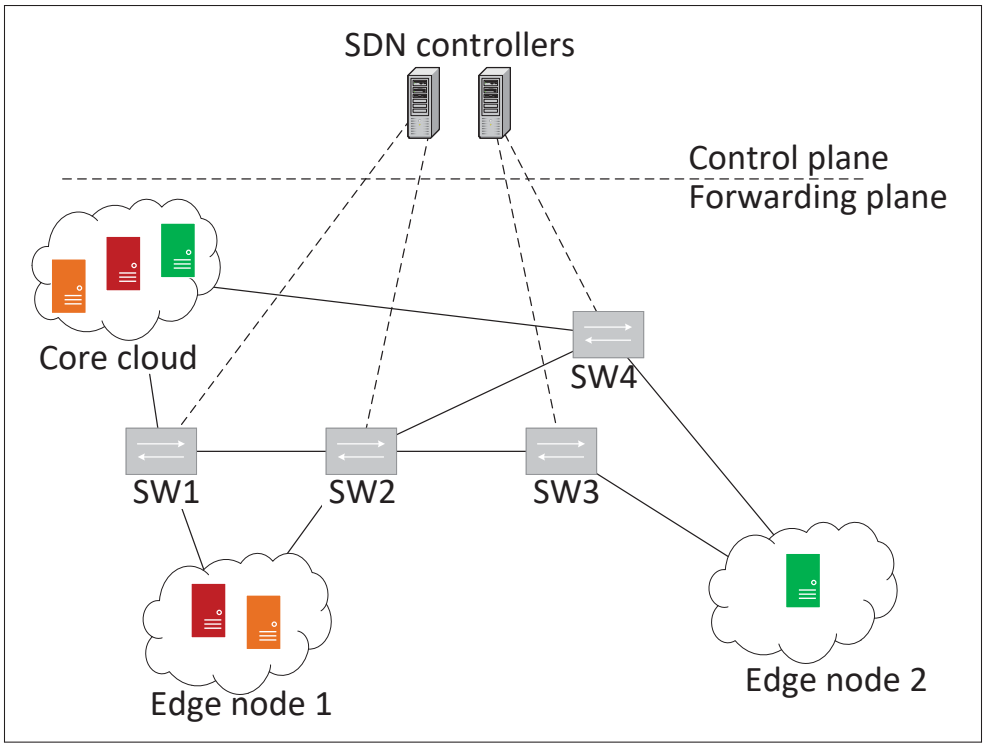


Figure 4.1 Edge-cloud SDN-enabled network system with a set of switches, edge/cloud with physical resource, and colored VNFs

differentiated by their colors and deployed onto the physical servers. The rest are switch nodes under the control of SDN controllers.

Let N_h, N_s, V denote the set of physical server hosts, the set of forwarding nodes (or switches) and the set of VNFs respectively. A set P of SDN controllers are placed across multiple sites to control VNFs' traffic forwarding by connected switches. The notation l is used to define the transmission delay between any two nodes, that is, $l_{h,h'}$ for physical hosts $h, h' \in N_h$, $l_{s,s'}$ for switches $s, s' \in N_s$, and $l_{s,p}$ for switch s and controller p, if mapped. Similarly, $B_{h,h'}$, $B_{s,s'}$ represent the bandwidth capacity allocated for the path between h, h' and between s, s' respectively.

Decision variables: For our ultimate goal, we define three main decision variables, namely i) $x_{v,h} \in \{0, 1\}$ indicates if VNF v is placed on the host h, ii) $y_{s,p} \in \{0, 1\}$ indicates if the switch s is mapped to the controller p. Assuming that a switch is controlled by only one controller at a

Table 4.1 Notation List

General Inputs	
N_h, N_s	Set of physical server hosts and switches
V, P	Set of VNFs and SDN controllers
F	Set of service request flows
k_v	Number of VNF v's instances
λ_f	Arrival rate of service flow f's request
$a_v^f, a_{v,v'}^f$	Indicator of the presence of VNF v and two consecutive VNFs v, v' in flow f
λ_p, μ_p	Arrival rate and processing rate of packets at p
Service Latency	
L_f, L_f^{set}	Threshold of latency and setup delay for flow f
$B_{h,h'}$	Configured capacity for the path between h, h'
l	Transmission delay between any two nodes
$b_{h,h'}$	Bandwidth used by two hosts h and h'
System Resource & Queue	
r_v	Physical resource required to operate VNF v
R_h	Physical resource pre-configured for host h
$Q_p(t)$	Number of packets in p's queue at time t
System Cost	
C_h	Cost per compute resource unit at host h
$C_{h,h'}$	Cost per bandwidth unit of the link between h, h'
C_p	Cost to deploy a controller p
C	Total system cost
Decision Variables	
$x_{v,h}$	1 if VNF v is deployed at host h, 0 otherwise
$y_{s,p}$	1 if switch s is assigned to controller p, 0 otherwise

time.

$$\sum_{p \in P} y_{s,p} = 1, \forall s \in N_s \tag{4.1}$$

Resource constraints: For each VNF $v \in V$, there might be many v's instances deployed over multiple physical hosts according to the license purchased. Let k_v denote the maximum number of the v's instances.

$$\sum_{h \in N_h} x_{v,h} \leq k_v, \forall v \in V \tag{4.2}$$

The system is assumed to serve a set of service request flows F where each flow $f \in F$ is represented as a chain of VNFs with service request arrival rate λ_f. We use $a_v^f \in \{0, 1\}$ and $a_{v,v'}^f \in \{0, 1\}$ to indicate that f's traffic goes through v and v, v' are two consecutive VNFs of the flow f, respectively. For any $h, h' \in N_h$,

$$b_{h,h'} = \sum_{f \in F} \sum_{v,v' \in V} x_{v,h} x_{v',h'} a_{v,v'}^f b_{v,v'} \leq B_{h,h'} \tag{4.3}$$

In order for v's to process one request, it requires r_v units of resource (e.g. total available CPU frequency in GHz) whereas in order to transmit data, v needs $b_{v,v'}$ units of network bandwidth (e.g. total available bandwidth in Gbps) to exchange data with another VNF v'. A VNF v can be only placed on a host $h \in N_h$ if it requires resource less than the amount of resource configured for h, i.e. R_h, and the bandwidth less than bandwidth capacity from h to any other hosts. For any $h \in H$, we have

$$r_h = \sum_{f \in F} \sum_{v \in V} x_{v,h} a_v^f \lambda_f r_v \leq R_h. \tag{4.4}$$

Similarly, a controller can control only a subset of switches according to its resource capacity defined by R_p and the units of resource required to handle each switch r_s^{ctrl}. This restriction is formulated as follows:

$$r_p = \sum_{s \in N_s} y_{s,p} r_s^{ctrl} \leq R_p. \tag{4.5}$$

Service latency: Each flow f has its own requirement of latency L_f. Therefore, the total delay for a flow to traverse from source node to destination node of a service chain should not exceed l_f. In other words, $\forall f \in F$:

$$l_f = \sum_{v,v' \in V} \sum_{h,h' \in N_h} x_{v,h} x_{v',h'} a^f_{v,v'} l_{h,h'} \leq L_f. \tag{4.6}$$

We also consider flow setup delay L^{set}_f due to its significant impact on IoT networks, especially safety-critical systems Kumar *et al.* (2017). In fact, this value is obtained by summing up the flow latency as the left term in constraint 4.6 and the latency for controllers to process PACKET-IN message. For any $f \in F$, we have:

$$\sum_{v \in V} \sum_{h \in N_h} \sum_{p \in P} \sum_{s \in N_s} x_{v,h} y_{s,p} A^{f,v}_{h,s} l_{s,p} + l_f \leq L^{set}_f. \tag{4.7}$$

where $A^{f,v}_{h,s} = a^f_v a_{h,s}$.

4.3.2 Queueing and System Stability Model

Due to the key value of controllers in an SDN network, we take system stability as the important consideration when solving SDN controller-switch mapping problem. Having said that, it assumes that each SDN controller maintains a queueing system to receive and process flow traffic from mapped switches over time slots $t = 0, 1, \ldots, T$. Let $\mathbf{\Theta}(t) = (Q_p(t), p \in P)$ be the queue backlog vector of the whole control plane, where $Q_p(t)$ is the number of blocked flow traffic packets in p's queue at the beginning of time slot t. Given the queue lengths of all controllers are all 0 at time slot 0, i.e., $Q_p(t) = 0, \forall p \in P$, the value of $Q_p(t)$ evolves over time as follows[2]:

$$Q^{(t+1)}_p = \left[Q^{(t)}_h + \lambda^{(t)}_p - \mu^{(t)}_p\right]^+ \tag{4.8}$$

[2] Throughout this paper, we denote $[z]^+$ as $\max\{z, 0\}$.

where $\lambda_p^{(t)}$ and $\mu_p^{(t)}$ are the arrival rate and the processing rate of packets at controller p. In this paper, the processing rates μ_p are known at every time slot whereas $\lambda_p^{(t)}$ at every SDN controller is determined based on the rate of the flow traffic going through VNFs as follows:

$$\lambda_p^{(t)} = \sum_{f \in F} \frac{\mathbf{1}_{z_p^f \geq 1}}{\Delta_f} \lambda_f^{(t)}, \forall p \in P \tag{4.9}$$

where $z_p^f = \sum A_{h,s}^{f,v} x_{v,h} y_{s,p}$ is the sum for all $v \in V$, $h \in N_h$, $s \in N_s$, $\mathbf{1}_{z_p^f \geq 1}$ is the indicator function, and Δ_f is the expiration time configured for forwarding rules installed at switches.

In a system where network nodes are highly related in that they might forward traffic of the same service chain, queue blocking delay adversely impacts the entire service chain's delay. A constraint is therefore imposed on the queue backlog to make sure an acceptable delay. According to Neely (2010), it can be achieved as long as the queueing system is stable, or in other words, its long-run averages hold the following condition:

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{p \in P} E\{Q_p^{(t)}\} < \infty \tag{4.10}$$

4.3.3 Problem Objective and Formulation

The objective of SDN Controllers and VNFs Placement (SCVP) problem is to minimize the total cost incurred by i) the allocation of computing resource at physical hosts for VNFs, ii) the deployment of SDN controllers at different sites, and ii) the reservation of network bandwidth of any links to carry traffic flow. In practice, the cost to perform these tasks can be modeled according to the energy consumption of each allocated unit of resource as follows:

- C_h: cost to allocate one unit of resource at host h
- $C_{h,h'}$: cost to allocate one unit of network bandwidth resource for the path between h and h'.
- C_p: cost to deploy an SDN controller p.

Our goal is to design an optimization engine that takes network topology and service chains' requirements as input and decides which controller handles which switches, and which switch forward traffic from VNFs deployed on the server connects to the switch to minimize the total cost as the weighted sum of three components, with $\alpha + \beta + \gamma = 1, \alpha > 0, \beta > 0, \gamma > 0$:

$$C = \sum_{h,h' \in N_h} (\alpha C_h r_h + \beta C_{h,h'} b_{h,h'}) + \gamma \sum_{p \in P} \sum_{s \in N_s} C_p y_{s,p} \tag{4.11}$$

The SCVP problem is formulated as follows:

$$\begin{aligned}
&\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} && C && (4.12)\\
&\text{subject to} && (4.1) - (4.7), (4.10) &&\\
& && x_{v,h} \in \{0,1\}, \forall v \in V, h \in N_h && (4.13)\\
& && y_{s,p} \in \{0,1\}, \forall s \in N_s, p \in P && (4.14)
\end{aligned}$$

4.4 Algorithm Design and Analysis

In this section, we develop an algorithm to solve the problem (5.19). First, we notice that the constraint (4.10) requires a priori knowledge of system status. Unfortunately, it might be not only impractical to keep tracking network system or services demand in a realistic environment during every time slot from the beginning but also difficult to rapidly retrieve historical information to address the optimization problem. In response, we adopt Lyapunov optimization framework to transform the original problem into the one that can be solved in each time slot. Second, it is required to orchestrate resource from an exponentially large number of nodes. As such, any algorithm should be dimension-free, i.e., its execution time must be independent or at least nearly independent on the size of the network. Inspired by the result from Vigneri *et al.* (2019), we employ a scalable and efficient approach based on gradient exponentiation called Exponentiated Gradient Algorithm (EGA).

4.4.1 Lyapunov Optimization Framework

To satisfy the constraint (4.10), the queue $\mathbf{\Theta}(t)$ must be mean rate stable as stated by Neely (2010). We therefore define the Lyapunov function and a so-called one-step conditional Lyapunov drift at time slot t as follows:

$$L(\mathbf{\Theta}(t)) = \frac{1}{2}\sum_{p\in P} Q_p(t)^2 \tag{4.15}$$

$$\Delta(\mathbf{\Theta}(t)) = E\big[L\big(\mathbf{\Theta}(t+1)\big) - L\big(\mathbf{\Theta}(t)\big)|\mathbf{\Theta}(t)\big] \tag{4.16}$$

Intuitively, $\Delta(\mathbf{\Theta}(t))$ is the difference of queue length between two consecutive slots. Therefore, the closer an algorithm pushes this value towards its bound, the more stable the system queue is. We have the following theorem to determine such the bound.

Theorem 1. *For any queue backlog $\mathbf{\Theta}(t)$ under any placement scheme, $\Delta(\mathbf{\Theta}(t))$ is upper bounded by*

$$\Delta(\mathbf{\Theta}(t)) \leq \frac{1}{2}E\big[\sum_{p\in P}\big(\lambda_p(t)^2 + \mu_p(t)^2\big)|\mathbf{\Theta}(t)\big] + \sum_{p\in P} Q_p(t)\lambda_p(t) - E\big[\sum_{p\in P} Q_p(t)\mu_p(t)|\mathbf{\Theta}(t)\big] \tag{4.17}$$

Proof. Using the definition of $L(\mathbf{\Theta}(t))$ from section 4.3.2 and substituting (4.15) into (4.16) yield:

$$\begin{aligned} L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t)) &= \frac{1}{2}\sum_{p\in P}\big(Q_p(t+1)^2 - Q_p(t)^2\big) \\ &= \frac{1}{2}\sum_{p\in P}\Big[(max[Q_p(t) - \mu_p(t), 0] + \lambda_p(t))^2 - Q_p(t)^2\Big] \\ &\leq \sum_{p\in P}\Big[\frac{\lambda_p(t)^2 + \mu_p(t)^2}{2} + Q_p(t)\big(\lambda_p(t) - \mu_p(t)\big)\Big] \end{aligned} \tag{4.18}$$

Taking the expectation of two sides in (4.18), we have

$$\Delta(\mathbf{\Theta}(t)) \leq B + E\Big[\sum_{p\in P} Q_p(t)\big(\lambda_p(t) - \mu_p(t)\big)|\mathbf{\Theta}(t)\Big]$$
$$= B + E\Big[\sum_{p\in P} Q_p(t)\lambda_p(t)|\mathbf{\Theta}(t)\Big] - \sum_{p\in P} Q_p(t)\mu_p(t) \quad (4.19)$$

where $B = E\Big[\sum_{p\in P}\Big[\frac{\lambda_p(t)^2+\mu_p(t)^2}{2}|\mathbf{\Theta}(t)\Big]$ and the fact that service requests' arrivals are independent and identically distributed over time slots and hence independent of current queue backlogs, or $E\big[\mu_p(t)|\Theta(t)\big] = E\big[\mu_p(t)\big] = \mu_p$. □

However, while minimizing a bound on $\Delta(\mathbf{\Theta}(t))$ would stabilize the system and satisfy constraint (4.2), it may result in a high cost $C^{(t)}$. Instead, by incorporating queue stability into system cost, we introduce a Lyapunov drift-plus-penalty function as follows

$$\Delta(\mathbf{\Theta}(t)) + VE[C^{(t)}|\mathbf{\Theta}(t)] \quad (4.20)$$

where $V > 0$ is a control parameter to determine how much cost minimization is emphasized. Our goal is to not only obtain a small $E[C^{(t)}|\mathbf{\Theta}(t)]$ so as to not incur a large system cost, but also achieve a small $\Delta(\mathbf{\Theta}(t))$ to lower the congestion of service requests given highly fluctuated IoT service demand. Then from Theorem 1, the objective function of problem (5.19) can be re-formulated in combining with constraint (4.10). The problem is re-stated as: every slot t, given the current queue states $\mathbf{\Theta}(t)$, $\mathbf{x}^{(t)}, \mathbf{y}^{(t)}$ must satisfy constraints (4.1)-(4.7), (4.13), (4.14) and

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \;\; C(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) = VC^{(t)} + \sum_{p\in P} Q_p(t)\lambda_p(t) \quad (4.21)$$

Based on Lyapunov optimization framework, we design algorithm 4.1 in terms of steps to solve the approximation of problem (5.19). At each slot, upon obtaining the solution of the optimization problem (4.21), the queue backlog of each relevant system nodes is updated. The

Algorithm 4.1 System Stability Relax Algorithm

1 $Q_n(0) \leftarrow 0, \forall n \in N$
2 At each slot t, collect nodes' queue backlog $Q_n(t)$, processing rate $\mu_n(t)$ and service request $\lambda_n(t)$
3 Solve the problem (4.21) to obtain $\mathbf{x}^{(t)}, \mathbf{y}^{(t)}$
4 Update the queue backlog according to (4.8)
5 $t \leftarrow t + 1$

average total cost returned by Alg. 4.1 is near-optimal in the sense that the closeness between it and C^{opt} the optimal value of the original problem (5.19) is guaranteed by following theorem:

Theorem 2. *For any $V > 0$ and $\varepsilon \leq 0$, we have*

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} E[C^{(t)}] \leq C^{opt} + \frac{B}{V} \tag{4.22}$$

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{p\in P} E[Q_p(t)] \leq \frac{B + VC^{opt}}{\varepsilon} \tag{4.23}$$

Proof. To prove this theorem, we use a result that has been proven by the theorem 4.5 in Neely (2010), which states that there exists a stationary optimal control policy that achieves C^{opt} and such that $E\left[\lambda_p(t)\right] \leq E\left[\mu_p(t)\right] - \varepsilon, \forall n \in P$.

Thus, from the fact that Alg. 4.1 minimizes $C^{(t)}$ among all policies that satisfy (4.1)-(4.7), (4.13), (4.14), that is:

$$\begin{aligned}\Delta(\mathbf{\Theta}(t)) + VE[C^{(t)}|\mathbf{\Theta}(t)] \leq B + VE[C^{(t)}|\mathbf{\Theta}(t)] + \\ E\Big[\sum_{p\in P} Q_p(t)\big(\lambda_p(t) - \mu_p(t)\big)|\mathbf{\Theta}(t)\Big] \leq \\ B + VC^{opt} - \varepsilon \sum_{p\in P} E[Q_p(t)]\end{aligned} \tag{4.24}$$

From the definition of $\Delta(\mathbf{\Theta}(t))$ in (4.16), summing both sides of the above inequality (4.24) over all $t = 0, \ldots, T-1$ and eliminating identical terms yields:

$$\sum_{t=0}^{T-1} \left(\Delta(\mathbf{\Theta}(t)) + V \sum_{t=0}^{T-1} E[C^{(t)}]\right) = \sum_{t=0}^{T-1} \left(E\left[L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))\right]\right) + V \sum_{t=0}^{T-1} E[C^{(t)}] = E\left[L(\mathbf{\Theta}(T))\right] - L(\Theta(0)) + V \sum_{t=0}^{T-1} E[C^{(t)}] \leq T(B + VC^{opt}) - \varepsilon \sum_{t=0}^{T-1} \sum_{p \in P} E[Q_p(t)] \quad (4.25)$$

Then by dividing T and V on both sides of the last inequality as well as neglecting non-negative terms leads to:

$$\sum_{t=0}^{T-1} E[C^{(t)}] \leq \frac{B}{V} + C^{opt} + \frac{L(\Theta(0))}{T} \quad (4.26)$$

$$\frac{1}{T} \sum_{t=0}^{T-1} \sum_{p \in P} E[Q_p(t)] \leq \frac{B + VC^{opt}}{\varepsilon} + \frac{L(\Theta(0))}{\varepsilon T} \quad (4.27)$$

Taking the limit $T \to \infty$ over both sides of (4.26) & (4.27) proves Theorem 2. □

Remarks: 1) The bounds in (4.22) & (4.23) give some insights about the trade-off between time-average placement cost and system queue backlog regarding the value of control parameter V. Particularly, a large V makes B/V in (4.22) small and so is the gap between the cost obtained from Alg. 4.1 and the optimum. It also enlarges the upper bound of node time-averaged queue backlog and as a result, offering high service latency due to more service requests are held by the network. Having said that, it is necessary to choose a compromise between system cost and network stability given constraint (4.14) on service chain completion time. 2) Unfortunately, the

problem (4.21) is still NP-hard and difficult to find the optimal solution in the polynomial time. Regarding this issue, we use the approach in Vigneri *et al.* (2019).

4.4.2 Exponential Gradient Ascent Method

4.4.2.1 Problem Relaxation

In order to adopt gradient method, we need to reformulate the discrete optimization problem (5.19) into the continuous optimization problem. There are two steps to perform. We first replace $x_{v,h} \in \{0,1\}$ and $y_{s,p} \in \{0,1\}$ with $\dot{x}_{v,h} \in [0,1]$ and $\dot{y}_{s,p} \in [0,1]$, respectively. Then product terms in the objective function and constraints are linearized by newly defined variables. In details, we substitute $x_{v,h}x_{v',h'}$ (or actually $\dot{x}_{v,h}\dot{x}_{v',h'}$) in (4.6), (4.7) with $\ddot{x}^{v,v'}_{h,h'}$ and following constraints $\forall v, v' \in V, \forall h, h' \in N_h$:

$$\ddot{x}^{v,v'}_{h,h'} \leq \min\{\dot{x}_{v,h}, \dot{x}_{v',h'}\} \tag{4.28}$$

$$\ddot{x}^{v,v'}_{h,h'} \geq \dot{x}_{v,h} + \dot{x}_{v',h'} - 1 \tag{4.29}$$

The newly added constraint (4.28) guarantees that if any of $\dot{x}_{v,h}$ and $\dot{x}_{v',h'}$ is close to 0, their product or $\ddot{x}^{v,v'}_{h,h'}$ should be also close to 0. And if both of $\dot{x}_{v,h}$ and $\dot{x}_{v',h'}$ get closer to 1, $\ddot{x}^{v,v'}_{h,h'}$ will be close to 1 as imposed by constraint (4.29). Similarly, we define $\ddot{y}^{s,p}_{v,h}$ as the substitution for the product $\dot{x}_{v,h}\dot{y}_{s,p}$ and constraints as follows:

$$\ddot{y}^{s,p}_{v,h} \leq \min\{\dot{x}_{v,h}, \dot{y}_{s,p}\} \tag{4.30}$$

$$\ddot{y}^{s,p}_{v,h} \geq \dot{x}_{v,h} + \dot{y}_{s,p} - 1 \tag{4.31}$$

Upon obtaining the optimal values of the continuous variables, it is necessary to decide whether or not the placement of VNFs onto physical servers and the mapping between switches to controllers are performed. We adopt a greedily random rounding technique in the sense that i)

Algorithm 4.2 Exponentiated Gradient-based Algorithm for SCVP Problem

Randomize a feasible $\mathbf{z}^0$ as an interior point of Ω
while *gap between two consecutive iterations is still considerable* **do**
    for *hyperplane A defining* Ω **do**
        Generate two distint points $\mathbf{z}_A^+ = \mathcal{P}_A^+(\mathbf{z})$ and $\mathbf{z}_A^- = \mathcal{P}_A^-(\mathbf{z})$ as defined in Moretti (2003)
        Set $\mathbf{z}_A = (\mathbf{z}_A^+ + \mathbf{z}_A^-)/2$ as the A-based midpoint
    end for
    Averaging all generated midpoints for next iteration
end while
while *stopping criteria does not meets* **do**
    With P-center point from above loop, generate new value of $\mathbf{z}$ according to (4.34)
end while
$\bar{\mathbf{z}}$ as the average of generated $\mathbf{z}$'s values
Return $\arg\min_{\mathbf{z}\in\{\bar{\mathbf{z}},\mathbf{z}_{\forall A}^+,\mathbf{z}_{\forall A}^-\}} C(\mathbf{z})$

$x_{v,h}$ and $y_{s,p}$ are set to 1 according to the probability $\dot{x}_{v,h}$ and $\dot{y}_{s,p}$ respectively, and ii) this step is repeatedly performed until we achieve a feasible solution.

4.4.2.2 Exponentiated Gradient Algorithm

The adoption of the Exponentiated Gradient Algorithm (EGA) to solve the relaxed continuous problem of (4.21) is motivated by the analysis in Vigneri *et al.* (2019) which elaborates the dependence of Mirror Gradient Algorithm (MGA)'s convergence rate on the dimensionality of the problem, which is $\mathcal{D} = (|V||N_h| + |N_s||P|)$ in this paper. Before presenting how the variables are updated by EGA, we introduce several following notations:

1. Ω: feasible region of the problem (4.21)
2. $\mathbf{z} = (z_1, z_2, \ldots, z_{\mathcal{D}}) \in \Omega$: a combined vector composed of $\mathbf{x}$' and $\mathbf{y}$'s elements. Note that $\sum_{i=1}^{\mathcal{D}} z_j \leq |V| + |N_s|$.
3. $\mathbf{z}^i$: the state of the algorithm at iteration i
4. $\nabla C(z_j^i)$: the subgradient of the objective function C of (5.19) at the j-th element of $\mathbf{z}^i$.

As a gradient ascent-based method, the idea of EGA is to move the variable $\mathbf{z}$ towards the optimal solution according to a certain updating function. Let $D_\phi(\mathbf{z}, \mathbf{z}^i)$ represent the Bregman divergence between $\mathbf{z}$ and $\mathbf{z}^i$ under the function ϕ, that is

$$D_\phi(\mathbf{z}, \mathbf{z}^i) = \phi(\mathbf{z}) - \phi(\mathbf{z}^i) - \langle \nabla\phi(\mathbf{z}^i), \mathbf{z} - \mathbf{z}^i \rangle \tag{4.32}$$

From the perspective of MGA, the updating function for $\mathbf{z}$ solves the problem:

$$\mathbf{z}^{i+1} = \arg\min_{\mathbf{z}\in\Omega} \{\langle \gamma \nabla C(z^i), \mathbf{z}\rangle + D_\phi(\mathbf{z}, \mathbf{z}^i)\} \tag{4.33}$$

Given $\phi(\mathbf{z}) = \sum z_i log(z_i)$, the following result obtaind from (4.33) has been proven already in Vigneri *et al.* (2019):

$$z_j^{i+1} = \frac{z_j^i(|\mathbb{V}| + |\mathbb{S}|)\exp(-\gamma \nabla C(z_j^i))}{|\mathbb{V}| + |\mathbb{S}| + \sum_{j=1}^{\mathcal{D}} z_j^i \exp(-\gamma \nabla C(z_j^i))} \tag{4.34}$$

Choosing initial candidate: Selecting a "center" point as the initial candidate as in Vigneri *et al.* (2019) might not work in this paper since it could violate other constraints as linear inequalities, i.e., (4.2)-(4.7), (4.13), (4.14). We observe that Ω is a bounded convex polytope since it is the intersection of half-spaces and is bounded by $\mathbf{0}$ and $\mathbf{1}$. The techniques for finding a center point of a polytope have been studied in many prior works. We decide to use the method to approximate the center of a linear programming polytope Moretti (2003) for three reasons: i) it provides a so-called P-Center point near the polytope's center in few simple and easy steps partially depending on the number of constraints, ii) it is robust to the addition of redundant inequalities, which typically occurs in our system models, and iii) the method allows itself to be reusable if some constraints are changed between time slots.

EGA procedure is summarized in Alg. 4.2. The first loop (step 2 - 8) is to find the center point of Ω which is used as the initial candidate for the second one (step 9 - 11). At the first loop, two points are generated by projecting a feasible point $\mathbf{z}$ onto every hyperplane. Then the average of

all the midpoints is chosen for the next iteration. With an initial candidate, (4.34) is applied until the stopping criteria met. Note that, we also take into account all the generated feasible points from the first loop. Specifically, step 13 returns $\mathbf{z}$ that incurs the minimum cost. In general, the basic steps are identical to that in Vigneri *et al.* (2019). What distinguishes between our work and Vigneri *et al.* (2019) is the procedure of initializing the candidate and leveraging the generated points to get the optimal total cost.

According to Vigneri *et al.* (2019), EGA converged at a speed depending on the dimensionality $\mathcal{D}$ of $\mathbf{z}$ and bounded by:

$$C^{opt} - C(\bar{\mathbf{z}}) \leq \frac{log(\mathcal{D}+1)}{q\gamma} + \frac{\gamma L^2}{2} \tag{4.35}$$

where $\bar{\mathbf{z}}$ is the output of Alg. 4.2, q is the number of iterations, $\hat{C}$ denotes the upper bound of $\nabla C(z_j), \forall 1 \leq j \leq \mathcal{D}$ and γ is gradient step size.

4.4.3 Prototype of SCVP Solution

With the optimization model, we next present the overview of the SCVP solution to discuss how to integrate the optimization module to optimize resource allocation in SDN-based networks. As the proof-of-concept, the prototype is based on interworking scenarios, i.e. session setup, calling flows, between IMS and WebRTC from our previous work Nguyen, Nguyen & Cheriet (2018) with the presence of various controllers and an E2E service orchestrator as shown in Fig. 4.2. Service functions together with forwarding functions are deployed as VNFs running inside containers or standalone VMs. All the operations of placing VNF onto containers or VMs as well as setting up the connectivity between virtual switches and SDN controllers are executed by an End-to-End service orchestrator (E2ESO).

In detail, E2ESO interacts with underneath resource controllers via Resource Adapters (RAs). Several options that the RA supports to exchange with controllers include i) YAML files with Container Manager to create containers and necessary configurations, i.e. specified via Dockerfile, ii) Heat template files with VM managers to spawn VMs and iii) JSON message

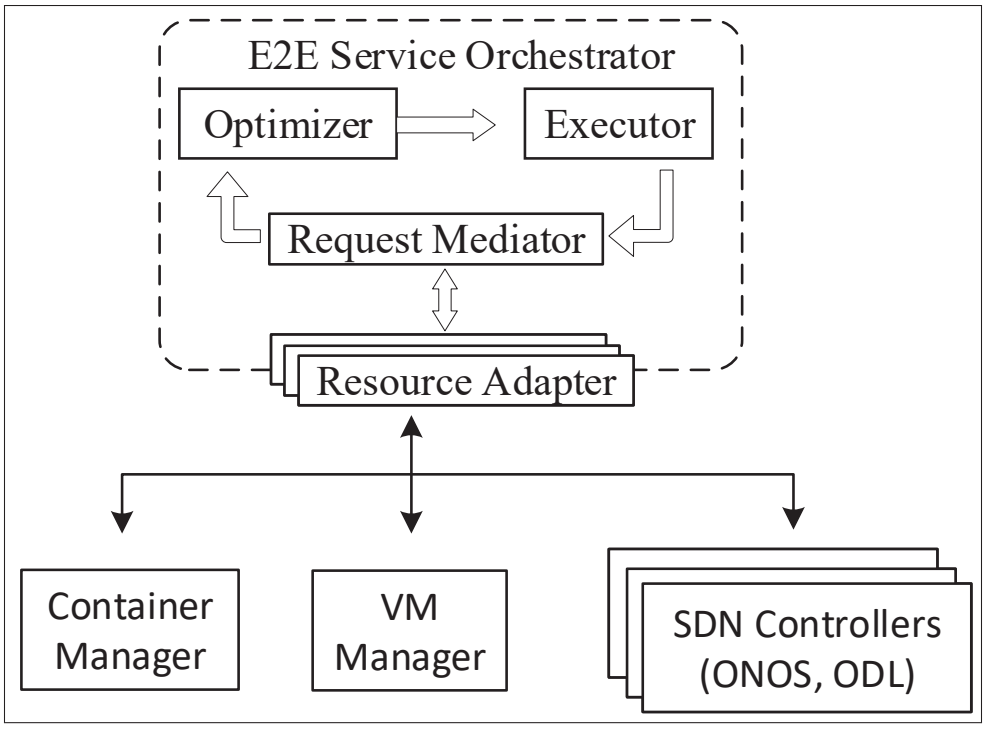


Figure 4.2 Overview of SCVP solution

to SDN controllers. Request Mediator can transform all the messages from RAs in any form into SCVP problem's input data used by Optimizer. The output of Optimizer in terms of VNF placement and SDN controller-switch mapping is consumed by Executor that decides where to dispatch and how to deploy that placement. To achieve this purpose, the Executor also collects status data from the controllers through the RAs (not shown in Fig. 4.2).

4.5 Performance Evaluation

In this section, we present the numerical results with various evaluation schemes to evaluate our proposed algorithm. Specifically, we focus on quantifying the convergence result, the objective value, the latency awareness and the acceptance rate of the system in comparison to the state-of-the-art methods. We also assess the applicability of our solution in a large-scale setting and experimental testbed.

4.5.1 Simulation Settings

Topology settings: We consider an implementation with a network system in which the ratio of the numbers of nodes, i.e. controllers, switches, and physical servers, is 3:3:6 respectively. This setting is adopted in several previous works, i.e. Liu *et al.* (2021). The link bandwidth capacity is randomized between 1Gbps and 10Gbps and the delay ranges from 2ms to 5ms Knight, Nguyen, Falkner, Bowden & Roughan (2011). The cost C_h for each unit of computing resource is set to the electricity price of the site where the blade servers locate, which ranges from \$0.7 to \$1.2 EnergyHub. The bandwidth cost $C_{h,h'}$ is simply determined based on the energy that endpoints consume to transmit or receive every 1Gb data, which is taken randomly between \$1.4 and \$2.4.

Table 4.2 Simulation parameters

Parameter	**Value**
Service arrival rate λ_f (ms^{-1})	(0.1, 1.0)
Service rate μ_p (ms^{-1})	(0.1, 0.3)
Bandwidth demand $b_{v,v'}$ (Mbps)	(10, 120)
Link bandwidth B (Gbps)	(1, 10)
Link latency l (ms)	(2, 5)
Computing resource cost C_h	(0.7,1.2)
Network bandwidth cost $C_{h,h'}$	(1.4,2.4)

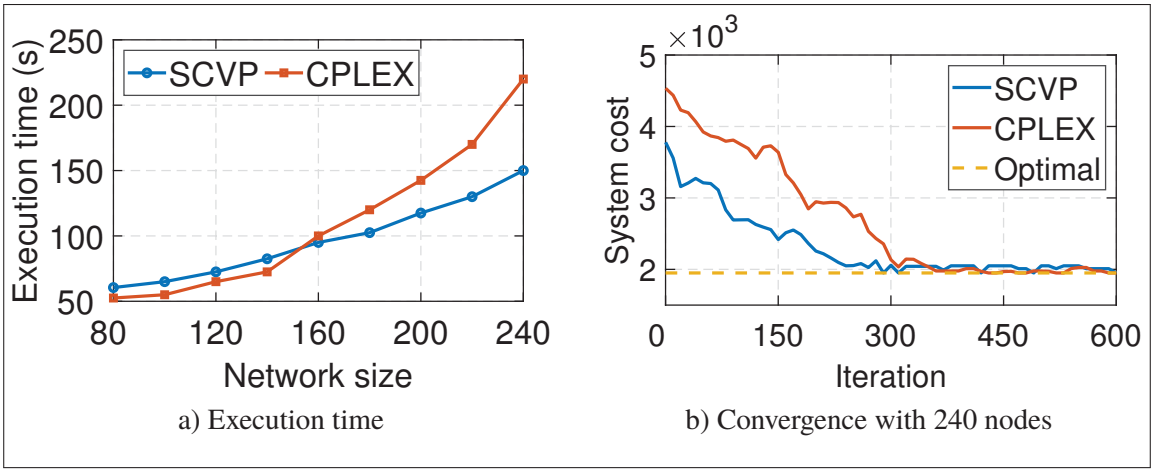
Service chain settings: We consider service chains composed of from 6 to 10 functions Nguyen *et al.* (2018) each of which is implemented as a corresponding VNF with 4 instances and bandwidth demand varies from 10 to 120Mbps. The system can serve up to 20 service flows with request's arrival rate varying between $0.1ms^{-1}$ and $1ms^{-1}$. Table 5.4 summaries other simulation parameters.

Comparison Algorithm: We compare SCVP with the optimal solution obtained by CPLEX solver with following methods:

- **Host-fit**: We prioritizes physical hosts by their residual resource capacity and place as many of VNFs as possible to the most available server. The switches are randomly mapped to the controllers as long as none of the constraints is violated.
- **Controller-fit**: The controllers are sorted by the capacity and the switches are greedily mapped to the most available one. VNFs are then randomly deployed onto physical servers with available resource.
- **Best-fit**: This method is performing the sort for both SDN controllers and physical nodes in parallel.
- **DVPRP**: The method *Dynamical VNF Placement and Routing Problem* for SFC Request flows proposed by Liu *et al.* (2021) jointly consider multiple resource constraints on nodes, links and QoS requirements in solving VNF resource allocation. However, the dynamic placement of SDN controllers and flow setup delay are not involved. As we have a different system model and parameters from Liu *et al.* (2021), to evaluate the outcome of our work, we only adopt their idea with the assignment of switches to SDN controllers determined via Controller-fit approach.
- **FASA**: A heuristic algorithm *Flow-path Aware Switch Assignment* introduced in Guo *et al.* (2020) is used to minimize the demand of controller resources regarding flow fluctuation and setup time. However, other metrics like end-to-end flow latency, the stability are not considered. The placement of VNFs is determined via Host-fit.

4.5.2 Simulation Results

To evaluate the performance of SCVP, we compare it with aforementioned methods in terms of algorithm convergence, system cost, service latency, resource utilization and system stability in terms of the average number of messages waiting in SDN controllers' queues. We use the average of the results out of 30 simulation runs as the value. We perform the simulation through time slots and use the average of the results out of 30 runs as the value.



a) Execution time

b) Convergence with 240 nodes

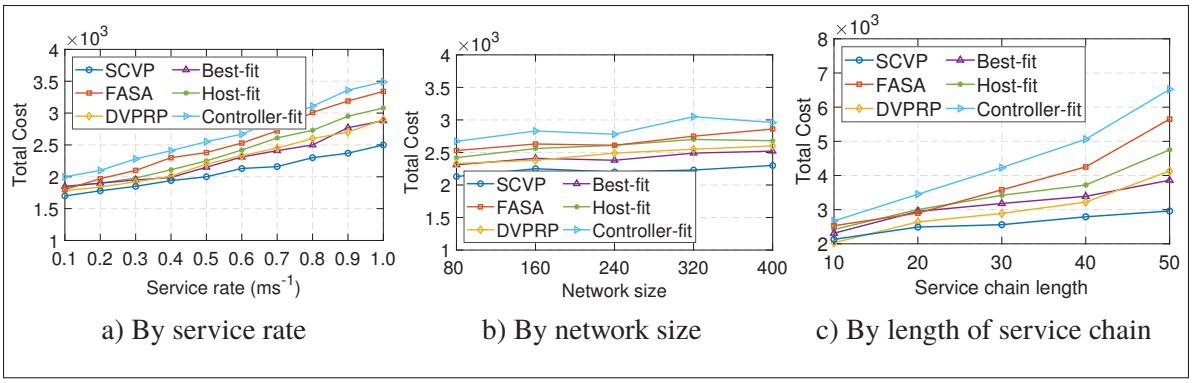
Figure 4.3 Convergence of algorithm with different network settings

4.5.2.1 Convergence

We run the simulations to understand the convergence of SCVP regarding the adoption of CPLEX solver to obtain the global optimal value. We first consider the execution time to achieve the optimal solution with the number of network nodes ranging from 80 to 240 nodes. As can be seen in Fig. 4.3a, with small and medium networks, i.e. less than 150 nodes, SCVP takes more time, i.e. 12s-25s to obtain the optimal results wherein it spends about 20s to find the initial candidate. SCVP exhibits its strength in dealing with the growth of network size, i.e. greater than 150 nodes. In details, as shown in Fig. 4.3b, SCVP shows a good performance when it approaches the solution sooner than the solver around 280 iterations, fewer than 345 iterations of CPLEX.

4.5.2.2 System Cost

Fig. 4.4a shows the comparison on total system cost among the algorithms of Controller-fit, Host-fit, Best-fit, DVPRP, FASA and ours. When the service requests come to the system at rate $1ms^{-1}$, the performance of SCVP is the best, which helps reduce about 16% - 23% of the cost comparing to DVPRP, Best-fit, Host-fit and 33%-40% to FASA or Controller-fit. In SCVP, we obtain a dynamical placement of both VNFs and SDN controller-switch connectivity,



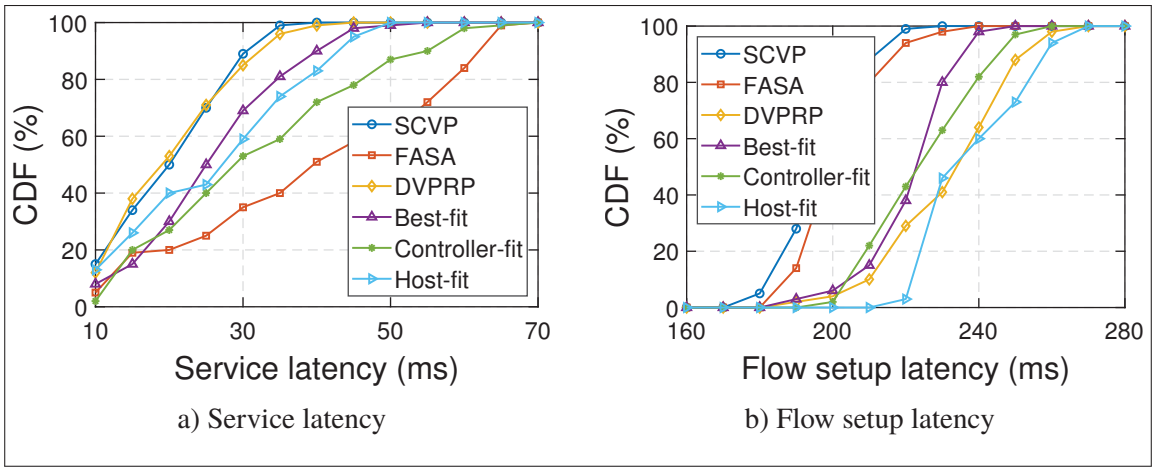
a) By service rate b) By network size c) By length of service chain

Figure 4.4 Evaluation of total system cost

consequently less resource is required to handle service requests. Since FASA and Controller-fit cannot ensure the optimal usage of physical resources for VNFs, which outnumbers the SDN controllers by about 9:1, their performance is worse than that of Host-fit, DVPRP. Note that there is no clear difference between Best-fit and DVPRP. This can be explained that while Best-fit does not obtain a VNF placement as optimal as DVPRP does, the resource required by Best-fit for SDN controllers is much lower than that by DVPRP, as a result, makes its cost comparable to DVPRP.

In Fig. 4.4b, we can see that the cost is not considerably fluctuated as there are more network nodes. SCVP remains to be the most optimal with lowest cost, i.e. 11% - 36%, while other algorithms even benefits from the increase of network size. For example, Host-fit and Best-fit can obtain better placement solutions with a network of 500 nodes than it does with 400 nodes. This is because there are more likely nodes to achieve a better VNF placement or controller-switch assignment. Practically, this gives an idea of properly adopting approaches under various network settings regarding the cost and as explained later, the convergence speed of the algorithms.

As the length of service chains increases, the system requires more resource at VNFs to process the requests and at controllers to handle the traffic forwarding tasks. The results shown in Fig. 4.4c justify the advantage of jointly considering the resource at both control and forwarding planes in minimizing the system cost. While there is no significant among the algorithms for service chains composed of less than 10 VNFs, SCVP appears to be the best solution for long



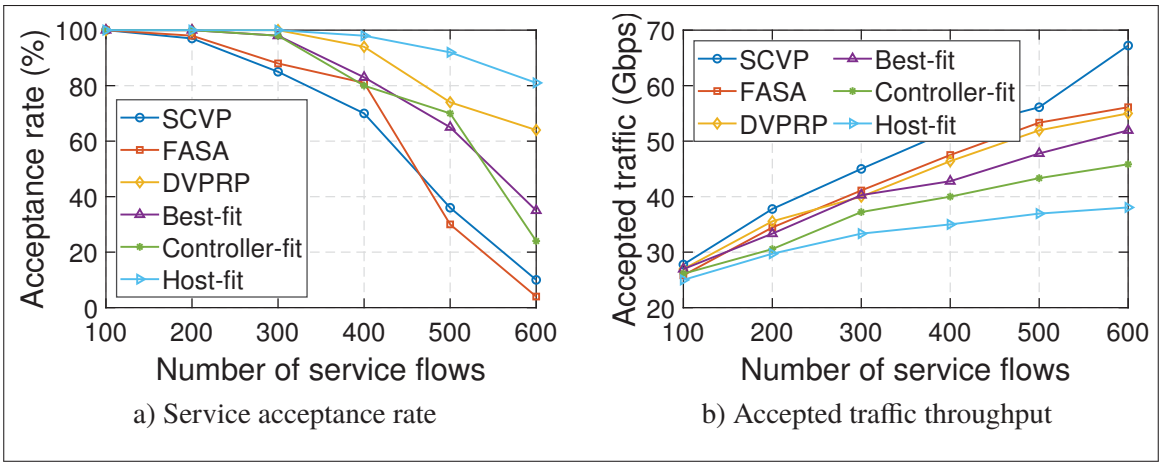
a) Service latency b) Flow setup latency

Figure 4.5 Service chain latency and flow setup latency distribution

service chains, i.e. more than 10 VNFs. In details, it not only optimizes the resource for VNFs but also keeps the resource for active SDN controllers low while guaranteeing service latency and flow setup delay lower than pre-defined threshold. With more than 40 VNFs per service chains, the total cost goes up drastically for other algorithms but best-fit and SCVP schemes thanks to a holistic strategy of provisioning system resource. Additionally, the impact service chain's length on VNF resource cost is much higher than that on SDN controller resource. This explains why the cost incurred by host-oriented methods like DVPRP and host-fit method is in general lower than that incurred by controller-oriented methods, i.e. FASA, controller-fit, respectively.

4.5.2.3 Service Quality

To give an insight of SCVP's benefits to QoS, we investigate the distribution of service chain's latency. As shown in Fig. 5.4.2, SCVP and DVPRP optimize the resource allocated for VNFs at physical servers, their performance is better than other algorithms with service chains delay less than $30ms^{-1}$. Best-fit and Host-fit approaches enable service chain with latency up to $40ms^{-1}$ and $55ms^{-1}$, which are much better than FASA and controller-fit with the maximum delay value at $78ms^{-1}$ and $85ms^{-1}$.



a) Service acceptance rate

b) Accepted traffic throughput

Figure 4.6 Evaluation of link utilization

In Fig. 4.5b, SCVP strategy achieves the lowest flow setup delay, i.e. 230ms, which is between 10ms and 40ms lower than that of FASA, DVPRP or Greedy ones. Unlike Fig. 5.4.2 in which SCVP and DVPRP as a host-oriented resource allocation method, are comparable in service latency, SCVP is actually better than FASA in setting up a traffic flow. We note that both SCVP and FASA improves the communication delay between switches and controllers, SCVP does a step further by optimizing the route for traffic between VNFs and as a result, obtains a lower latency than FASA in total.

Fig. 4.6a shows the average acceptance rate of various algorithms. Average acceptance rate is understood as the service flows served by the whole system regarding the total request arrivals. In the simulation, SCVP performs the best, which gets about 20% higher in accepting service requests than that of Best-fit, which is in turn about 40%-65% higher than that of the others. The outperformance of Best-fit especially over FASA or DVPRP justifies the advantage of joinly taking into account resource at both control and VNF planes in provisioning IoT applications as service function chains. With SCVP, apart from the presence of flow setup constraint, we also ensure the stability of SDN controllers to reduce the chance a flow is dropped by switches. As for host-oriented methods, i.e. DVPRP, Host-fit, or controller-oriented methods, i.e. FASA, Controller-fit, the allocation of either physical resource for VNFs or the assignment of switches

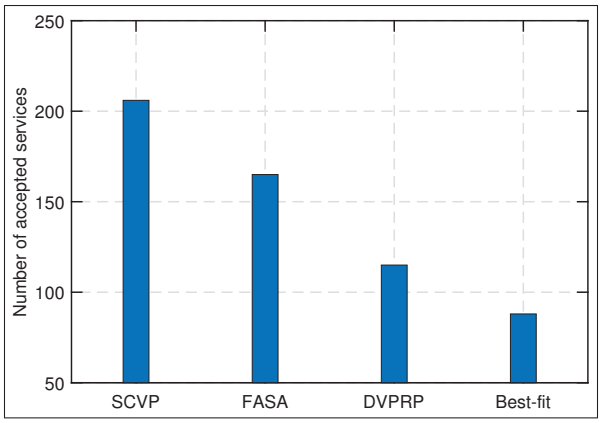


Figure 4.7 Comparing total at different service request rates

to the number of available SDN controllers is insufficient to handle as many service flows as SCVP or Best-fit does.

In order to justify the result in Fig. 4.6a, we conduct an evaluation of algorithms based on an additional parameter, i.e. average accepted throughput. The metric is measured as the total bandwidth of service traffic flows that are successfully received in the network. As shown in Fig. 4.6b, SCVP enables an average amount of throughput about 8Gbps higher than that of DVPRP, over 20Gbps higher than that of other approaches. This is aligned to what we observe in Fig. 4.6a about SCVP with the highest acceptance rate of service requests.

4.5.2.4 System Capacity

We run simulations to understand the scalability of SCVP using a stress test approach. In details, we scale up the number of latency-sensitive services until the system meets the limitation of resource capacity, i.e., the system drops new services because there are no available resources that can meet all requirement constraints. The maximum number of accepted service requests is then obtained and compared between various schemes. We simplify the output plot by excluding Host-fit and Controller-fit methods due to their poor performance.

Fig. 4.7 shows that the scalability of SCVP is considerably higher than FASA, DVPRP and Best-fit as it can admit 206 requests in total while those of other approaches are 165, 115 and 88

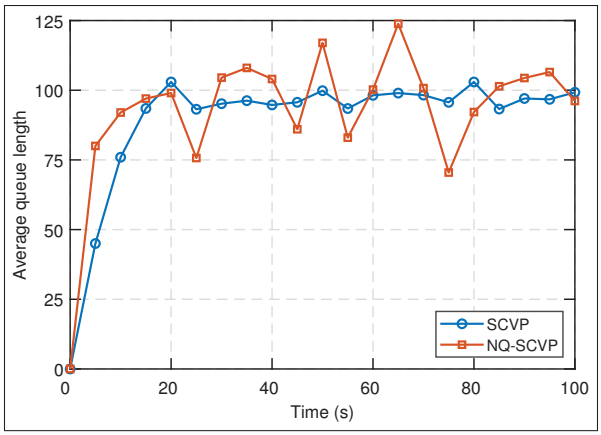


Figure 4.8 Average queue length over time

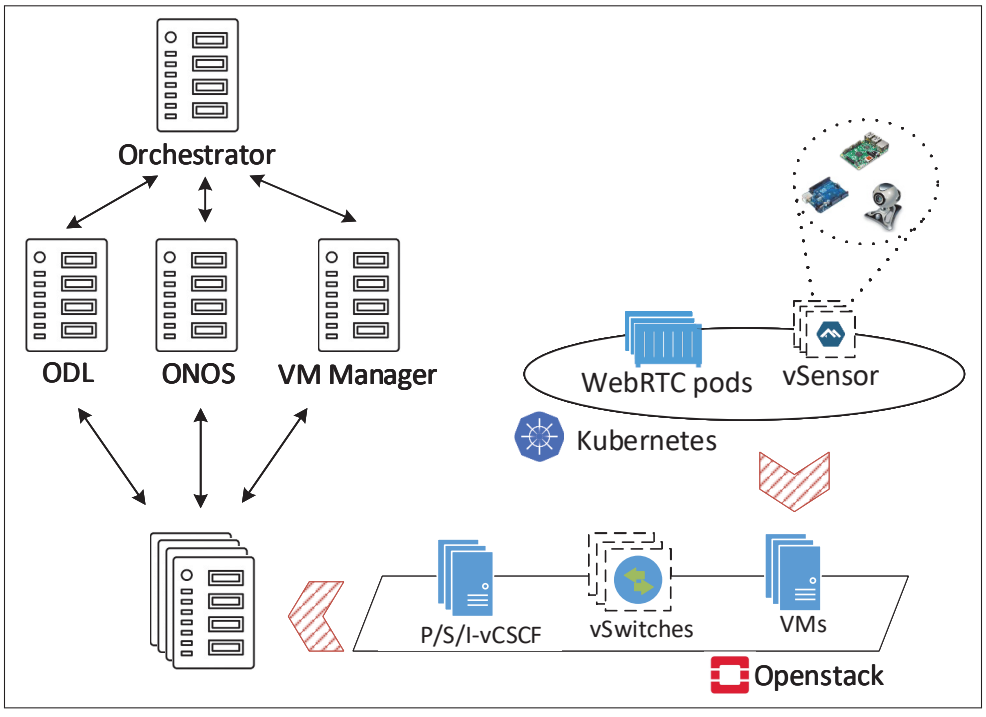


Figure 4.9 Experimental system Setup

requests. The results indicate that for the same amount of available resource, the optimal VNF placement and switch-controller assignment of SCVP is able to increase the number of deployed applications. From network operators' perspective, this can help to either increase the revenue or reduce the service price for the sake of competitiveness.

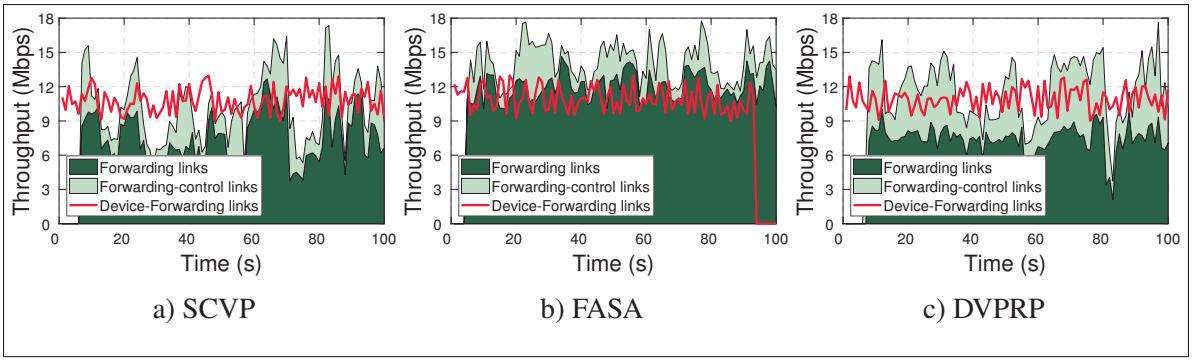


Figure 4.10 Measured throughput across network with different algorithms

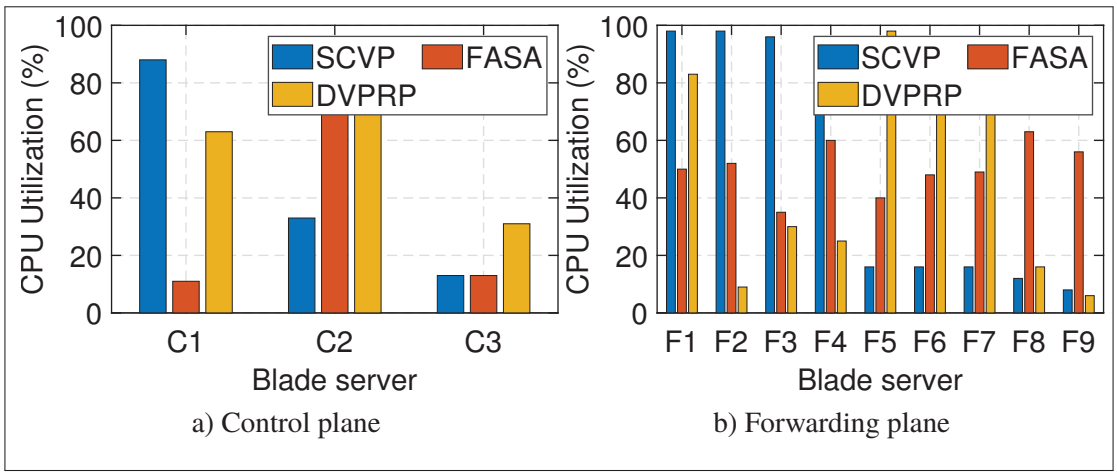


Figure 4.11 CPU utilization of blade servers

We are also interested in measuring the average number of messages in the queue at SDN controllers over time slots. We conduct simulations with the focus on only SCVP and non-queue SCVP (NQ-SCVP) regarding the fact that FASA, DVPRP do not take into account controllers' stability. Fig. 4.8 presents the time evolution of average controllers' backlog queue length of SCVP and NQ-SCVP. Under SCVP's resource allocation scheme, a quite stable pattern of backlog queue is shown with the average queue backlog slightly fluctuates within a window of 7-10 messages. NQ-SCVP shows a large fluctuation in queue length, which indicates a greater chance for a service request to be rejected when the controllers are overloaded.

Table 4.3 VNF Resource Configuration

Service functions	vCPU	vRAM	instance
P-CSCF	2	4	4
S-CSCF	2	4	3
I-SCCF	4	16	4
HSS	4	16	2
WRIG	4	8	3
WSF	2	8	3

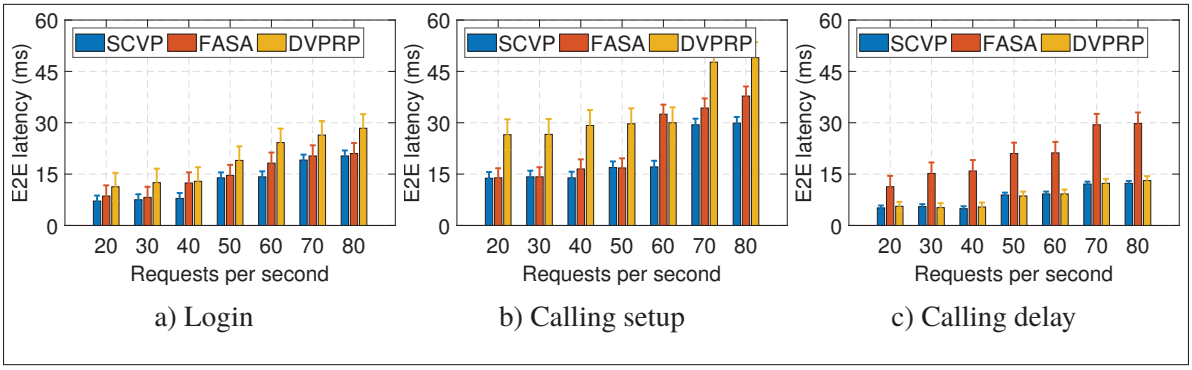


Figure 4.12 E2E delay for different service chain classes

4.5.3 Experimental Evaluation

4.5.3.1 Experimental Setup

We conduct the experiment with interworking scenarios, i.e. session setup, calling flows, between IMS and WebRTC involved as depicted in our previous work Nguyen *et al.* (2018). The system is composed of 13 blade servers where 1, 3, 9 blades are considered as upper bound of resources to host controllers, switches, and servers with respect to the ratio in simulations. Each blade has 24 CPUs, 96Gb RAM and is equipped with programmable network interface cards with bandwidth up to 100Gbps.

For controllers, we install OpenDayLight for IMS domains and ONOS for WebRTC domain, both of which run as VMs with 2 virtual CPUs (vCPUs) and 4GB virtual RAM (vRAM) as the upper bound R_p. The set P therefore has no more than 6 controllers. Each switch is containerized and allocated with 1 vCPU, 512Mb vRAM along with the installation of Openswitch. Within the switch container, a script is installed in advance to help to switch the connectivity from the switch to various controllers. Having said that, the set of switches is limited to 30 switch-featured VMs.

In order for the experiment to work, we also deploy IMS and WebRTC functions onto VMs with the number of vCPUs between 2 and 8, and from 4Gb to 16Gb vRAM depending on the requirement of functions. The configuration details are given in Table 4.3.

The entire system runs on Openstack VMs and Kubernetes containers/pods as shown in Fig. 4.9. We deploy a monitoring tool to collect the status of resource via pre-installed Resource Adapter agents at each VM/pod/containers and a logging server processing log files, of IMS as well as WebRTC service components. Based on the monitoring and logging data, E2ESO will make a decision of scaling up/down resource.

4.5.3.2 Experimental Results

We do not consider the greedy approach due to its poor performance and focus on the improvement of SCVP over FASA and DVPRP which do not provide a jointly optimal resource allocation at control and forwarding planes. Fig. 4.10 shows the measurement of the traffic throughput on the links between physical servers over 100 time slots. The total throughput between end devices, i.e. IMS terminals, WebRTC clients, and connected servers is identical for both SCVP, FASA and DVPRP. In total, SCVP requires less bandwidth resource than FASA or DVPRP do. In Fig. 4.10a and 4.10b, we see that the throughput on forwarding links mainly between VNFs on blade servers required by SCVP is less than that by FASA while the throughput on forwarding-controlling links is comparable between them. This is understandable since both SCVP and FASA ensure the controllers to be efficiently mapped to switches. Comparing to DVPRP in Fig. 4.10c, it is obvious that SCVP consumes less resource of forwarding-controlling

links than DVPRP does because DVPRP adopts the greedy approach to assign switches to controllers, which is not as optimal as EGA. Note that unlike the measurement of accepted throughput in Fig. 4.6b, in Fig. 4.10, we solely focus on the physical links between blade servers. The traffic exchanged via virtual links, i.e. between two VMs within the same blade server is not counted.

We then compare the CPU utilization of blade servers between three approaches. The result is given in Fig. 4.11. For control plane, SCVP and FASA ensures to maximize the usage of each blade server's CPUs, i.e. more than 85% and thus spare the controller 3. DVPRP deploy the controllers over all the controllers even though none of them runs with full capacity. Similarly, for forwarding plane in Fig. 4.11b, only 5 blade servers are needed by SCVP. FASA mainly optimizes the resource at control plane and thus inefficiently allocate VNFs on blade servers. Note that in this experiment, as we equally consider weight parameters in Eq. (4.11), i.e. α, β, and γ, the output is expected to intuitively obtain the deployment on as few blade servers as possible. For the concern of resource load balancing, it can easily adapt the algorithm by lowering the upper bound of available resource at each server.

In Fig. 4.12a and 4.12b, the delay of login and calling setup flows are considered as the delay of setting up a session. DVPRP results in the highest latency for these two scenarios due to the sub-optimal mapping between SDN and switches. The difference of setup flow between DVPRP and SCVP or FASA is quite significant given the fact that it takes much longer for a message to be processed by controllers, i.e. about $19.02ms$ comparing to the latency for a message to be exchanged and processed by VNFs, i.e. $2.3ms$. Regarding E2E latency for calling session as shown in Fig. 4.12c, SCVP achieves lowest delay and therefore provides the best quality of real-time communication service. We notice that the variance of FASA is also higher than those of SCVP and DVPRP. With FASA, we see that the media traffic flow goes through VNFs scattered over various blade servers whereas most of traffic flows traverse within the same blade servers under the placement schema of SCVP and DVPRP and thus less fluctuate than those by FASA.

4.6 Conclusion

This paper presents a solution of optimally provisioning resource at both control and forwarding planes in edge-cloud SDN-based networks. The approach is shown to be robust to the dynamic nature and real-time requirements of IoT services.

We model the optimal placement problem of VNF resource and network connectivity between SDN and forwarding switches. The problem is considered with a critical number of nodes from a large-scale system where heterogeneous IoT services are hosted. To obtain an optimal placement solution while ensuring system stability, we adopt the Lyapunov optimization framework and implement the exponential gradient ascent method. The performance is validated via simulation and testbed in which dynamic service demand is involved. The results show that regarding the impact of dynamic service demand on how IoT applications are constructed, total system cost can be saved up to $11 \sim 40\%$ depending on service demand and fluctuation rate.

To bring the proposed approach to industry, we plan to investigate micro-service design patterns in future work, so that the entire system can be deployed via containers on Container as a Service infrastructure.

CHAPTER 5

NFV-BASED ARCHITECTURE FOR THE INTERWORKING BETWEEN WEBRTC AND IMS

Duong Tuan Nguyen[1] , Kim Khoa Nguyen[1] , Mohamed Cheriet[1]

[1] Department of Automation Production, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Article published in *IEEE Transactions on Network and Service Management*, December 2020

Abstract

The emerging paradigm of Network Function Virtualization (NFV) technology promises an efficient solution for optimized service deployment in the cloud computing environment thanks to its ability to dynamically add or remove virtual resources when there is a change in workload. Nevertheless, telecom providers are still facing a challenging issue in efficiently adopting NFV to deploy Web Real-Time Communication (WebRTC) service on top of IP Multimedia Subsystem (IMS). Providing WebRTC service increases the inherent complexity of the IMS system in terms of the number of service nodes as VNF and the way they interact, both of which play significant roles in the problem of optimally allocating resources. This paper proposes a virtualized interworking system between IMS and WebRTC called NFV-based interworking architecture (NWII), and describes the mechanism for VNF to exchange messages with each other. We present an analytic system model considering the constraints of resources, QoS, and service costs. A real-time Markov approximation-based resource allocation algorithm (RIDRA) is then designed allowing a provisioned resource at service nodes to be reconfigured in time to meet performance requirements. The proposed solution is evaluated on the large scale by simulation and on the small scale by our developed testbed. Experimental results reveal that our algorithm effectively responds to fluctuating service demands with a service cost reduced by 19% via efficiently allocating virtual resources while maintaining QoS requirement.

Keywords: Resource allocation, NFV, WebRTC, IMS, Markov approximation-based algorithm, multi-domain service orchestration

5.1 Introduction

Nowadays to accommodate significant traffic growth, network operators are facing many issues, from space to locate new physical servers or network equipment to increasing costs of energy. They also need the skills to design, operate and manage such complex hardware-based infrastructure. To deal with these issues, a novel paradigm, called NFV ETSI ISG on Network Functions Virtualization (2013) is introduced. NFV enables network services to be provisioned via software-based network functions and network elements, i.e., bridges, routers, sitting on top of general-purpose servers instead of using specialized hardware. NFV also enables a system adapting to the change of service demand. In other words, both physical infrastructure providers and SPs benefit from NFV through reduced costs and energy consumption, improved manageability and shortened time-to-market, which are all critical factors in the success of service delivery as regards performance and reliability.

For telecom SPs, a growing NFV adoption has been realized via many prior proposals related to IMS Carella *et al.* (2014) due to its important role in telecom core networks. IMS is designed by Third Generation Partnership Project (3GPP) as a way to integrate with a high-quality network of telecommunication carriers for pervasive data access to Internet services. However, optimally scaling resources for the NFV-based IMS system is a challenging task due to the interacting complexity of virtual IMS entities, i.e., call session control function (vCSCF), home subscriber servers (vHSS). Moreover, typically integrating third-party services from other domains into the IMS platform introduces additional VNFs which increases the inherent system complexity and therefore creates new constraints to the optimization problem of provisioning resources. In detail, a variety of VNFs resource configurations (or templates) and inter- or intra-domain network latency all need to be taken into account. It also asks for a well-defined messaging mechanism among VNFs so that the workload at each VNF can be obtained for optimal resource allocation. Regarding the interoperability with WebRTC Holmberg *et al.* (2015) designed

for real-time communications in the Web environment, a poorly designed resource allocation scheme is unable to handle two NFV-based systems of IMS and WebRTC with a very large number of subscribers and therefore does not leverage the benefits provided by NFV technology. Eventually, it breaks the original commitment to employ WebRTC, which was intended to help IMS operators expand the number of potential endpoints for multimedia sessions Nguyen *et al.* (2016).

In this paper, we assume that IMS system and WebRTC service are virtualized and operated by SPs who are typically not willing to disclose detailed information about their network topology. The interworking function is implemented as a stand-alone entity which can be deployed as a part of IMS system or by WebRTC SP in reality. In addition, service components are assumed to be deployed as VNFs running inside an NFV Infrastructure (NFVI)-as-a-Service (NFVIaaS) provided by one or multiple providers. Our main focus is the problem of service chain resource allocation regarding common sessions going through the VNFs of the entirely NFV-based WebRTC $\leftrightarrow$ IMS interworking system.

Recently, significant efforts have been devoted to the investigation of an NFV-based architecture for multi-domain service orchestration ETSI ISG on Network Functions Virtualization (2013); Rosa *et al.* (2015); Katsalis *et al.* (2016); Dräxler *et al.* (2018); Li *et al.* (2018b). While their proposed solutions, in general, enable the deployment of service chains in cross-domain infrastructure, they are not detailed enough to completely model the interworking scenarios between IMS and WebRTC. Specifically, there is no explicit explanation of how an interworking SP communicates with other SPs to setup cross-domain service chains. The abstraction of domain resource is also described in a generic way and consequently makes it hard to model the performance of WebRTC $\leftrightarrow$ IMS interworking system. In this work, our main contributions are twofold. First, we model IMS $\leftrightarrow$ WebRTC NFV-based service chaining considering system resource, E2E service latency, and system cost. Second, we develop a real-time inter-domain resource allocation algorithm (RIDRA) based on the Markov approximation framework Chen *et al.* (2013) which is robust in solving the combinatorial optimization problem and efficient with the distributed implementation. In addition, we specify 5G Operating System (OS) reference

architecture Dräxler *et al.* (2018) for the specific interworking scenario between IMS and WebRTC. The details of message exchanged between IMS and WebRTC's VNFs for two common service chains i.e., login and calling are also described.

The rest of this paper is organized as follows. Section II reviews prior works about VNF resource allocation and virtualized IMS with WebRTC. Section III proposes an NFV-based WebRTC ↔ IMS interworking architecture, its realization in the interworking service and the interactions between relevant VNFs in login, calling sessions. System modeling and the formulation of the problem are explained in section IV. Then, the resource management system and a real-time inter-domain resource allocation algorithm (RIDRA) are discussed in section V. Section VI provides a performance evaluation of the proposed algorithm with both simulation settings and experimental testbed. Finally, conclusions are drawn.

5.2 Related Work

5.2.1 Multi-Domain Service Chain: Architecture & Modeling

The topics related to service chain in multiple domains have been extensively studied in prior works. In Boubendir *et al.* (2018), a Proof-of-Concept demonstrates the federation and orchestration of access and edge resources by a network operator to create and deploy customized network slices dynamically over a cross-domain network. In terms of architecture, Rosa et al. Rosa *et al.* (2015) discussed three cases of multi-domain, distributed NFV, i.e. management and orchestration, bandwidth negotiation, and reliability. Regarding the problems of NFV orchestration and the network slicing, a NFV-based architecture and its realization on the LTE network are proposed in Katsalis *et al.* (2016). However, none of these two studies proposed models or algorithms for the deployment of service chain. The authors in Li *et al.* (2018b) introduce a horizontal-based multi-domain orchestration framework in SDN/NFV-enabled satellite and terrestrial networks. They model the service chain placement regarding to the resource in terms of CPU and memory, which might be inconsistent with situations in which the SPs deploy their service on NFV infrastructure. Another effort of 5G Operating System

(OS) towards 5G communications aiming to abstract away the complexities of underlying 5G infrastructure for an efficient and flexible service orchestration is discussed in Dräxler *et al.* (2018). In this work, we exploit the proposed 5G OS reference architecture and specify its components to enable the NFV-based interworking system between IMS and WebRTC.

In terms of system modeling and optimization, an increasing number of models are being proposed to optimally allocate resource when composing service chain. In Wang *et al.* (2015), a model of dynamic function composition is presented together with a distributed algorithm using Markov approximation method. Authors in Bhamare *et al.* (2017) discuss optimization model to reduce the overall E2E latency by reducing the inter-cloud traffic *w.r.t* multiple VNF instances across multiple clouds. In Gupta *et al.* (2018), Gupta et al. explore different deployment strategies for service chaining in a so-called Network-enabled Cloud and develop a model for chaining VNFs with minimal resource consumption while fulfilling service requirements. Another work Sun *et al.* (2018) also tackles the resource allocation problem of orchestrating service chains by adopting full mesh aggregation method. Similarly, the authors in Riera *et al.* (2016) and Zhang *et al.* (2016) respectively employ a dynamic approach and a vertex-centric distributed algorithm to address the issue of optimally allocating networking and IT resources for VNF hosting.

Table 5.1 Related works on service chain optimization problem

Reference	**Static optimal**	**Dynamic optimal**	**Distributed algorithm**	**Multi-domain**
Wang *et al.* (2015)	Yes	Yes	Yes	No
Bhamare *et al.* (2017); Gupta *et al.* (2018); Sun *et al.* (2018)	Yes	No	No	Yes
Riera *et al.* (2016)	Yes	Yes	No	Yes
Zhang *et al.* (2016)	No	No	Yes	Yes
This paper	Yes	Yes	Yes	Yes

Table 5.1 summarizes some works on the service chain placement problem. Our work features all four requirements which are not taken into account in previous studies. In particular, we

focus on the elasticity of resource allocation in response to service demand changes and propose a real-time Markov approximation-based algorithm that minimizes total service cost concerning delay and resource constraints. Moreover, for service across multiple NFV-based systems like IMS and WebRTC and their components with different requirements, both the heterogeneous VNF configurations and a distributed algorithm should be considered.

5.2.2 IMS: WebRTC Interworking and Virtualization Architecture

Conventional interworking architecture between WebRTC and IMS has motivated existing works. In Amirante *et al.* (2014), a modular-based gateway architecture Janus is designed as a bridge between legacy IMS protocols and WebRTC. Similarly, an IMS integrated WebRTC prototype is used in Cruz & Barraca (2015) to evaluate the performance in terms of call throughput and mouth-to-ear delay. Reference 3GPP (2018) is another effort of 3GPP to re-architecture the IMS platform to enable access by WebRTC-based clients. From the architectural perspective, what makes our work distinguish from 3GPP (2018) is that we consider the use case in which both WebRTC and the interworking SP are third-party services on top of the IMS system. The realization of the proposed architecture as well as the message flows are therefore done without any change to the functionalities of conventional IMS entites. While a lot of progress has been made to address the heterogeneous characteristic between two domains, these works do not take into account the adoption of NFV paradigm which allows VNFs to be deployed on different shared physical and virtual resources in order to guarantee scalability and performance requirements. Such the NFV adoption sets our work apart from other related works.

With the widely used IMS platform, adapting the NFV architecture ETSI ISG on Network Functions Virtualization (2013) to an IMS-based system has been investigated in many studies. One of the methods presented in Carella *et al.* (2014) is to deploy service components as corresponding virtual units. The approach is preferable due to its compatibility with IMS or WebRTC specification and the simple implementation to leverage the advantage of NFV. In Lu *et al.* (2013), the authors propose an algorithm to address the problem of dynamic resource allocation. However, they only consider resource aspects in terms of CPU and memory and

do not clarify how well their algorithm can handle changes in service demand. Duan et al. Duan *et al.* (2017) propose an NFV management system to deploy IMS-related service chains based on predicted workload and per-instance processing capacity. Another auto-scaling scheme (VLB-CAC) at the VNF level for optimal allocation of IMS server' resources to admitted calls is introduced in Montazerolghaem *et al.* (2016). VLB-CAC is responsible for finding the optimal call acceptance rate for each SIP server by solving an optimization problem that prevents overload. Despite being designed based on the NFV architecture proposed by ETSI ETSI ISG on Network Functions Virtualization (2013), it is challenging to efficiently employing these works to the WebRTC ↔ IMS interworking service without respecting the requirement of deploying multi-domain and multi-provider services. Inspired by the use cases presented in ETSI ETSI ISG on Network Functions Virtualization (2018), a distributed NFV-based architecture facilitating the deployment of such the services is proposed.

5.3 NFV-Based Interworking Architecture

In this section, the proposed NFV-based WebRTC ↔ IMS architecture (NWII) is presented with the details of functional core components along with their interact interface. The NWII's realization for the WebRTC ↔ IMS interworking service, which provides an insight of how WebRTC and IMS's VNFs communicate with each other is then presented. The WebRTC ↔ IMS Interworking Gateway (WRIG)'s structure and the message flows between VNFs are also explained.

5.3.1 Overview of Core Components

The NFV-based interworking architecture (NWII) and its corresponding components of the 5G OS reference architecture Dräxler *et al.* (2018) are shown in Figure5.1. Similar to the 5G OS's Service Management (SM) component, Inter-Service Manager (ISM) converts high-level requests of composing an interworking service from tenants into concrete actions. For instance, it shall ask Sub-Service Managers (SSMs) for the details of IMS and WebRTC service placement

via the interface with Multi-Domain Orchestrator (MdO) or the topology of specific domains via Sm-Ro.

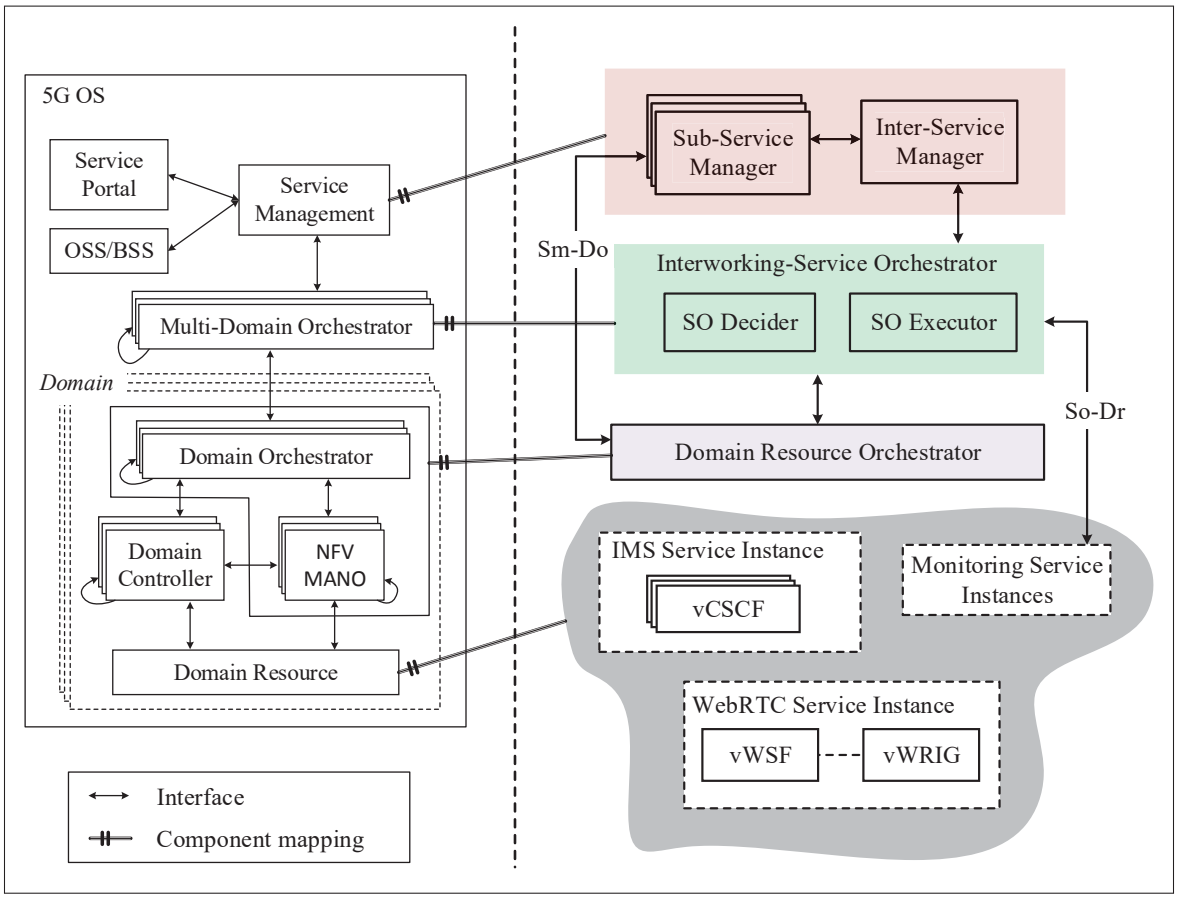


Figure 5.1 NFV-based interworking architecture between IMS and WebRTC with reference architecture from Dräxler *et al.* (2018)

As part of the 5G OS's MdO, the task of policy-based entity SO Decider (SOD) of Interworking-Service Orchestrator (ISO) is to make a decision on selecting the optimal component SPs to compose the interworking service chain. To solve such a multi-objective optimization selection problem, the SOD can collect analytical data about IMS or WebRTC service components from Monitoring Service instances via the interface So-Dr. SO Executor (SOE) orchestrates selected SPs via the interface with the Domain Resource orchestrator (DRO) which plays a role of both Domain Orchestrators and NFV MANO of the 5G OS architecture. The interested reader is referred to Dräxler *et al.* (2018) for more details of other blocks and interfaces.

5.3.2 WebRTC ↔ IMS Interworking NFV-based Service

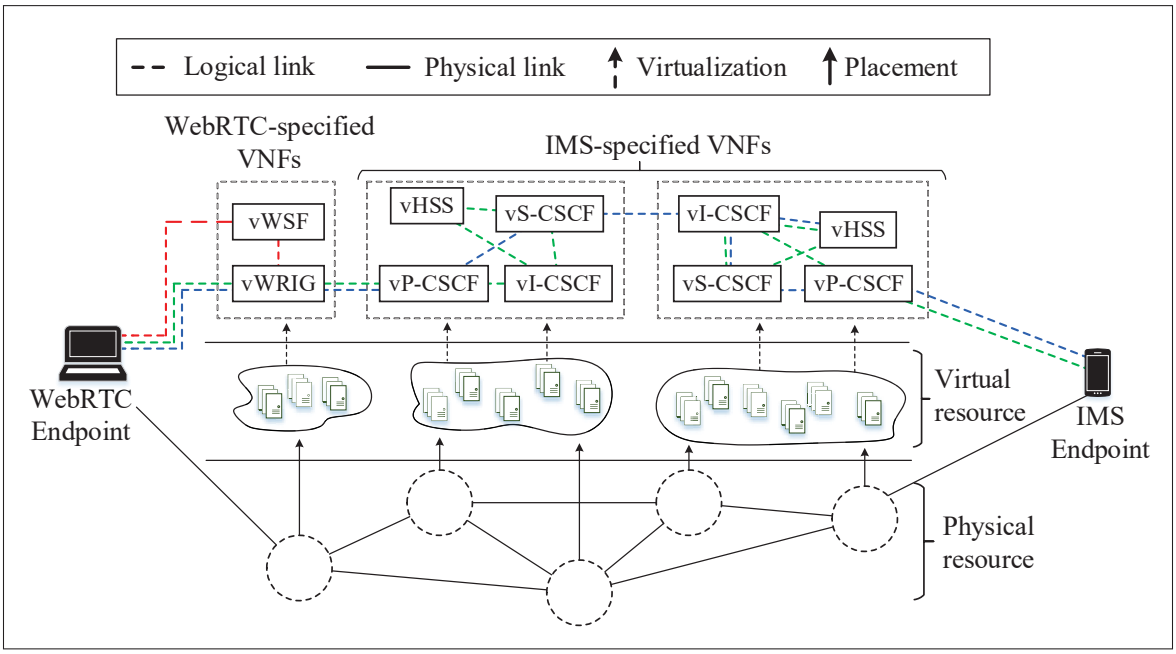


Figure 5.2 NWII Realization for WebRTC ↔ IMS Interworking Service

The NWII is realized via the WebRTC ↔ IMS interworking service as shown in Figure5.2. The P/I/S-CSCFs and HSS are IMS VNFs which functionalities are already defined in relevant 3GPP specifications Camarillo & Garcia-Martin (2007). Inside the WebRTC domain, the WSF handles incoming HTTP requests to provide the Web application for downloading to the browser of WebRTC users. Note that unlike the proposed architecture in 3GPP (2018) that requires to modify to the P-CSCF's functionalities to integrate WebRTC-enabled features, our approach of deploying the WRIG as a separated entity not only benefits from the elasticity of the NFV technology but also open more chances for conventional IMS system to inter-operate with different WebRTC SPs. Moreover, as the first entry point for the IMS endpoint, such the enhanced version of P-CSCF or eP-CSCF may result in a waste of resource since not every connection to the eP-CSCF is from WebRTC clients.

Service chains demonstrate traffic flows of different service sessions. For example, the login requests from a WebRTC endpoint going through Web Server Function (WSF) which in turn performs queries for WRIG's status is shown by the red dash line. It happens the same with the

registration phase of the WebRTC endpoint or the IMS endpoint with IMS domain (the green line) and the calling session between the WebRTC endpoint and an IMS endpoint (the blue line). VNFs are assumed to be virtualized in heterogeneous virtual instances which are placed inside an NFVI.

The WRIG plays two roles, i.e., a WebRTC endpoint and an IMS one, and is composed of functional components as shown in Figure5.3. Upon receiving WebRTC messages over the WebSocket protocol, the WebSocket component collects WebRTC information and sends it to the WebRTC stack (WS) which is responsible for carrying out WebRTC sessions with a WebRTC client. To setup multimedia connections between WRIG and WebRTC clients, the WS informs the Interactive Connectivity Establishment (ICE) agent during the negotiation process for Network Address Translation (NAT) traversal and connectivity checks. The WRIG is also IMS compliant thanks to IMS Stack (IS), which is in charge of handling IMS SIP messages exchanged between WRIG and IMS subscribers. WebRTC sessions and IMS sessions are orchestrated by a so-called Session Management (SM) component. The duties of SM include synchronizing a session from both sides to make sure that the communication is successfully established or tears down and bilaterally interprets between WebRTC messages sent by WebRTC users fed by the WS, and IMS/SIP messages sent by IMS users fed by the IS.

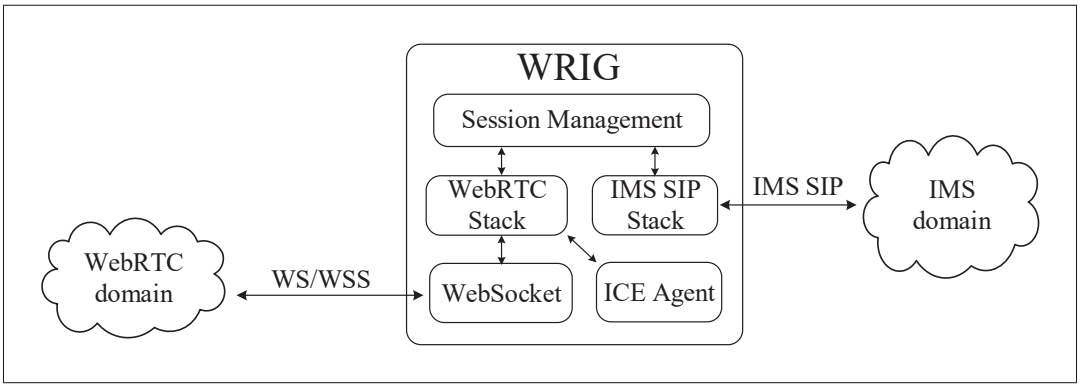


Figure 5.3 Functional components of the WRIG

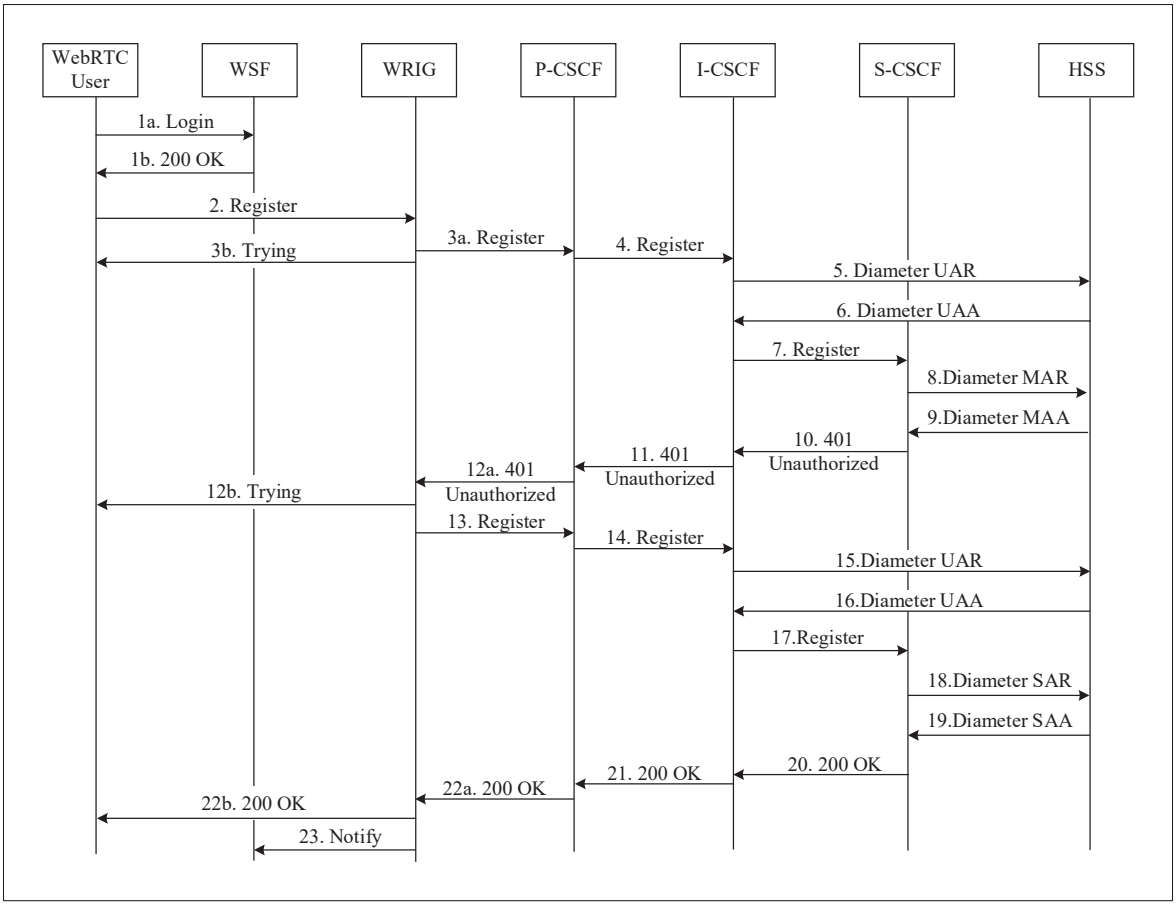


Figure 5.4 Message flows of WebRTC ↔ IMS Registration procedure

5.3.3 Session Setup Procedures

We briefly describe the session setup procedure in terms of message exchanges between VNFs. In this section, we focus on the procedure when a WebRTC user registers himself with an IMS domain and the procedure of establishing a WebRTC → IMS session. The reverse direction and other scenarios (e.g., ICE flows) can be developed in a similar way.

5.3.3.1 Registration Procedure

For a terminal to get access to any IMS service, it must register in advance. In the interworking case, the IMS registration procedure is performed by the WRIG as an IMS user requested by the WSF when receiving the LOGIN message from a WebRTC user (WU). The whole procedure

is depicted in Figure5.4. At step 1a, the LOGIN message is processed by the WSF to check whether or not the WU is authorized to communicate with IMS subscribers. If yes, the WSF will reply with a 200 OK message (step 1b) with the necessary information for the WU to contact the WRIG. Such information might include user profiles (e.g., user identity, secret value) as well as the WRIG's address in case the WRIG function is servicing by many servers. The WU directly sends the WRIG a WebRTC REGISTER message (step 2) that is then converted into an IMS REGISTER message routed via the P-CSCF (step 3a). The WU is notified at step 3b. The REGISTER message is then traversed from the P-CSCF to the I-CSCF (step 4). The details of steps 5 - 22a can be found in Camarillo & Garcia-Martin (2007). At step 12a, the WRIG is challenged to authenticate the WU with a 401 UNAUTHORIZED message and notifies the WU about this status at step 12b. At step 22b, the registration phase is complete when the WU gets the 200 OK message. At the same time, the WSF is informed about the WU's status by the WRIG (step 23).

5.3.3.2 WebRTC ↔ IMS Calling Setup Procedure

Figure5.5 illustrates message flows between involved VNFs of a calling session. We consider the case in which the WRIG is registered as a subscriber of IMS network 1 (IN1) and initiates a call session to another IMS user in IMS network 2 (IN2). In this scenario, offer/answer exchanges between two WUs or two IMS users are properly aligned with each other. In particular, the emission of an OFFER by the WU to the WRIG (step 1) results in an OFFER in the form of an INVITE message by the WRIG as an IMS user to the P-CSCF (step 2) of the IN1. The information about the calling session can be conveyed to the WSF for certain purposes (e.g., recording history, charging). The receipt of the 183 Session Progress from the P-CSCF triggers a Provision Answer (PRANSWER) to be sent back as a notification for the WebRTC caller (step 13b) to release unnecessarily allocated resources (i.e., Iteractive Connectivity Establishment (ICE) components, video decoders). The WU also receives a corresponding RINGING message (step 37b) when the WRIG gets the 180 RINGING message from the P-CSCF1. The final 200 OK message from the P-CSCF in IN1 (step 52a) leads to a corresponding ACCEPT message with eventual ANSWER

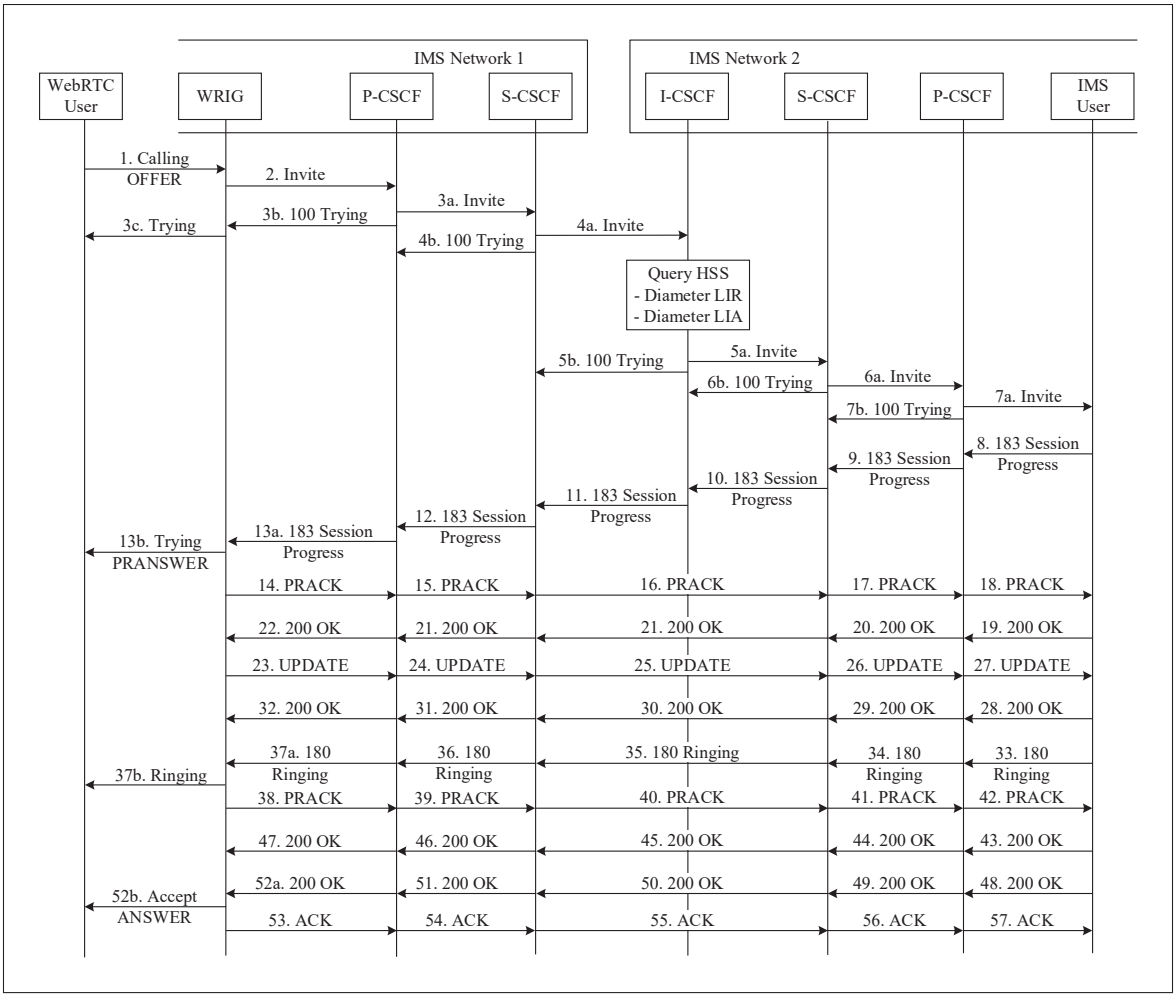


Figure 5.5 Message flows of WebRTC → IMS Calling procedure

information sent back to the WebRTC caller by the WRIG (step 52b). Similar to what occurs in the registration procedure, the WSF might be informed by the WRIG after step 52b (not explicitly shown) about the status of the calling session. All of the other messages exchanged between the entities in IN1 and IN2 are described in Camarillo & Garcia-Martin (2007).

Note that despite the compulsory use of ICE servers in WebRTC services Holmberg *et al.* (2015), these relevant VNF are excluded from the calling setup procedure. This is because service chains that send ICE transportation information are separated from those that carry session description, i.e., login, calling according to Uberti, Jennings & Rescorla (2017). This separation allows for a faster ICE startup since it can start as soon as any transport information is available rather than

waiting for all of it. Having said that, we assume that the sessions other than login or calling is performed out-of-band and therefore the details of flows are not taken into account in this paper.

5.3.4 Discussion

As the number of end users in either the WebRTC or the IMS domain increases, more flows going through VNFs are created, and thus they need to be effectively managed, especially at the WRIG that is required to synchronize sessions with IMS system as an IMS subscriber as well as with WebRTC system as a WebRTC peer. In reality, the number of session types may be much higher due to the deployment of other VNFs (e.g., media gateways, ICE-enabled VNF). Each session typically involves a different number of distinguishing VNFs with various interaction ways between them, namely, sequential, in parallel or both.

For SPs who only have general information about the resource, i.e., virtual CPU/memory/storage, network bandwidth, it is a challenge to make a decision on optimal resource allocation to achieve the objectives of end-to-end QoS. To exploit the internal communication between VNFs, such a decision must bear in mind many variables, for example, message or session types, classes of virtual instances, processing/bootstrapping time of each instance, as well as resource availability. An allocation scheme should orchestrate VNFs in a way that none of them handles a heavy load while the others are idle, which would adversely affect the whole system performance.

5.4 System Model and Problem Description

In this section, we present a system model regarding resources, E2E service latency and system cost. An optimization problem is formulated to find configurations for VNFs to achieve minimal total cost while satisfying the constraints of delay and resources. Let N, V denote the set of domains and the set of VNFs involved in WebRTC $\leftrightarrow$ IMS interworking system. We consider various sessions as service chains in the formed of VNFs of V which locate in one or many domains of N. Given a domain $n \in N$, there are available configurations for its VNFs, which are different from each other by the number of allocated resources, i.e. CPU, memory, storage.

The configurations are also different by the domain. For example, a configuration-*small* (or -*large*) instance in domain n can serve 1000 users whereas a configuration-*small* (or -*large*) one in domain n' can afford the workload of 100 users. To overcome this issue, we denote M as all the VNF configurations and $\alpha_{m,n} \in \{0, 1\}$ as the indicator if the configuration m is used in n. Along with the computing capacity of VNF, the transmission capacity between them (or network bandwidth) is typically provisioned by a network entity, i.e. software-defined controller Kreutz *et al.* (2015). Let $B_{m,m'}$ and $\omega_{m,m'}$ respectively denote the bandwidth and network latency between two VNF configurations m, m'. E2E service delay is computed on the topology of the chain. The mathematics notations defined in this section are summarized in Table 5.2 for easy reference.

5.4.1 System Resource Model

We denote R as the set of resources. Given $m \in M$, $r \in R$, let $a_{m,r}$, $t(m)$, and μ_m respectively denote the number of resources r configured for configuration-m VNF, the creation time, and the processing capacity of VNF. A VNF v is configured according to one of available configurations in M. This configuration is presented as $x_{v,m}$, which value is 1 if v is assigned with configuration m and 0 otherwise. We also use $y_{v,n} \in \{0, 1\}$ to indicate whether or not v locates in n.

Let S denote the set of service chains. In this work, we consider S with only login and calling chains. For each chain $s \in S$, there are $V_s \subset V$ VNFs that are involved to construct a corresponding service path L_s whose length is l_s. Any message belonging to s first arrives at the first node of L_s at the mean rate λ_s. The message is then forwarded to next node until it reaches L_s's last node. For the connection between a pair of VNFs u and v of V_s, after being served by VNF u, messages are then transmitted to VNF v successfully with the probability $p_{uv} \leq 1$ or not for network connectivity reasons. If the message transmission fails, the VNF will restart the transmission process.

We define $\lambda_{uv,s}$ as the message rate of s transmitted from u to v; $\lambda_{v,s}$ as the total rate at v. The occurrence of service chains is assumed to be independent and any VNF can be shared between

Table 5.2 Notation List

General Inputs	
N, V	Set of domains and set of VNFs in all the domains
S, M, R	Set of service chains, VNF configurations and resource
V_s	Involved VNFs of service chain s
L_s	Ordered list of VNFs in V_s with the length l_s
λ_v	Overall message incoming rate at VNF v
β	Upper bound of traffic intensity for any VNF
$\alpha_{m,n}$	1 if the configuration m is used in n, 0 otherwise
Service Latency	
μ_m	Processing capacity of configuration-m VNF
d_v	Delay time of a message at VNF v
$\omega_{m,m'}$	Transmission time between two configurations m, m'
d_{uv}	Time to transmit a message from VNF u to VNF v
D_s	Latency of session s
σ_s	Maximum delay of service chain s
System Resource	
$a_{m,r}$	Number of resource r for configuration-m VNF
$g_{n,r}$	Upper bound of resource r in domain n
$B_{m,m'}$	Network bandwidth between two configurations m, m'
$A_{n,r}$	Number of resource r allocated for domain n
System Cost	
c_r^n	Cost for allocating resource r in domain n
c_b	Cost of using network bandwidth
U_V	Cost for operating the whole system every time unit
C_V	Total system cost
$f(s)$	Fine function of violating chain s's SLA
Variables	
$x_{v,m}$	1 if configuration m VNF v is used, 0 otherwise
$y_{v,n}$	1 if VNF v locates in domain n, 0 otherwise

the chains. The formulation of the overall incoming rate is retrieved λ_v as follows:

$$\lambda_v = \sum_{s \in S} \lambda_{v,s} = \sum_{s \in S} \sum_{u \in V_s} \lambda_{uv,s} \tag{5.1}$$

The total number of resources r allocated for the system in domain n denoted as $A^e_{n,r}$ must not exceed the available resource r in n and is given by:

$$A_{n,r} = \sum_{s \in S} \sum_{v \in V_s} \sum_{m \in M} a_{m,r} x_{v,m} y_{v,n} \leq g_{n,r} \tag{5.2}$$

5.4.2 Service Latency Model

In order to model latency D_s of a service chain s, we consider transmission delay between endpoints with the corresponding VNFs or between VNFs d_{uv} and processing delay at each VNF d_v. Regarding the connectivity between endpoints and servers, we only take into account the transmission delays of the wireless link because that of the wired link is negligible due to high available bandwidth and low bit error rates. The average delay d^n between an endpoint and its server denoted is calculated according to Munir & Gordon-Ross (2010) as:

$$\begin{aligned} d^n = (K-1)\tau + \frac{d^{frm}}{(1-\bar{q}_n^{N_{TCP}})(1-2\bar{q}_n)} \\ + \frac{1-\bar{q}_n}{1-\bar{q}_n^{N_{TCP}}} d^{frm} \left[\frac{\bar{q}_n^{N_{TCP}}}{1-\bar{q}_n} - \frac{2^{N_{TCP}+1}\bar{q}_n^{N_{TCP}}}{1-2\bar{q}_n} \right] \end{aligned} \tag{5.3}$$

where K is the number of frames per packet, τ is the inter-frame time, d^{frm} is the end-to-end frame propagation, $\bar{q}_n$ indicates the packet loss rate of domain n and N_{TCP} denotes the maximum TCP transmissions in case of packet loss.

Each VNF is assumed as a M/M/1 queue model, and hence the time to process a message d_v is obtained according to Gautam (2012), which is:

$$d_v = \frac{1}{\sum_{m \in M} x_{v,m} \mu_m - \lambda_v} \tag{5.4}$$

The transmission time d_{uv} is given by:

$$d_{uv} = \sum_{m \in M} \sum_{m' \in M} x_{u,m} x_{v,m'} \omega_{m,m'} \tag{5.5}$$

To guarantee VNF v is not overloaded, the average time between two successive messages must be greater than the mean processing time by any server of v to a message. In other words, we require the traffic intensity ρ_v at v as the rate between the arrival rate at each server and its corresponding service (or processing) time, and this must satisfy the stability condition Gautam (2012) as follows:

$$\rho_v = \sum_{s \in S} \frac{\lambda_{v,s}}{d_v} = \frac{\lambda_v}{d_v} < \beta \tag{5.6}$$

where $\beta \in (0, 1]$ is the stability bound of VNFs.

Next, we adopt $D_{s,k}$ with $k \leq l_s$ to indicate the time that a message traverses from the first VNF, i.e., L_s^1, to the k^{th} node, i.e., L_s^k. For the sake of obtaining D_s in terms of d_v, d_{uv}, and the probability p_{uv}, we use a recursive computation based on the pre-computed value of D_{s,l_s-1}. In detail, with u, v as the last two VNFs of L_s, the expected value of D_s or $E[D_{s,l_s}]$ is calculated as the sum of the processing time for a message by the first $(l_s - 1)$ nodes, the transmission time d_{uv} and the processing time d_v, or:

$$\begin{aligned} E[D_{s,l_s}] = E[D_{s,l_s-1}] + E[d_{uv}] + E[d_v] &= E[D_{s,l_s-1}] + \Big(p_{uv} d_{uv} + 2(1 - p_{uv}) d_{uv}\Big) + d_v \\ &= E[D_{s,l_s-1}] + (2 - p_{uv}) d_{uv} + d_v \end{aligned} \tag{5.7}$$

Let δ_s denote the maximum delay of s. Solving (5.7) in terms of d_v, d_{uv}, and p_{uv} yields the session latency with the constraint as:

$$D_s = \sum_{q \in V_s} d_q + \sum_{u,u^* \in L_s} \Big((2 - p_{uu^*}) d_{uu^*} \Big) + d^*_{sys} \leq \delta_s \tag{5.8}$$

where d^*_{sys} represents the transmission delay between the WebRTC or IMS endpoint and the system which might be the first VNF, i.e., L^1_s or the last one, i.e., $L^{l_s-1}_s$ depending on how the session is constructed. The notation u^* is defined as the next VNFs of u in L_s. Note that d^*_{sys} is computed as the sum of d^n from (5.3) if endpoints involve during a session more than one time.

Regarding the change of workload, the number of resources allocated for the system may vary at a period of time during which the workload is constant. We use the superscript notation (i) for relevant parameters to indicate the index of the scheme defined as that period. $(i-1)$ and $(i+1)$ are understood as the previous and next schemes of the i^{th} scheme. We assume that system performance is not affected during the period of reallocating resources and that the message will benefit immediately from the newly allocated resource when the process is completed. Due to bootstrapping duration $t(m)$, a session can be conducted in three ways under different schemes as illustrated in Figure5.6: i) the session has already started before the allocation decision is made, and concludes when the system resource is under the next scheme; ii) the session starts after the allocation moment and lasts to the next scheme, and iii) the session is conducted with the next scheme.

Figure5.6 illustrates different cases in which a session can be performed regarding two consecutive schemes. The decision of adapting resources allocated by the current scheme to the new scheme is made at the time t_{alloc}, and the process concludes at $t_0 = t_{alloc} + t^{(i)}_V$. The session starts at t_{ses}, where t_{ses} resides in any of three consecutive intervals $(-\infty, t_{alloc}]$, (t_{alloc}, t_0) and $[t_0, +\infty)$. The value of $t^{(i)}_V$ is the amount of time for the whole system to move completely to the new resource and is retrieved as:

$$t^{(i)}_V = \max_{v \in V} \Big(\sum_{m \in M} x^{(i)}_{v,m} t(m) \Big) \tag{5.9}$$

Essentially, the last two cases (ii) and (iii), are considered as one case in the sense that the chain s starts performing within the first $t_V^{(i)} - t_{ses} + t_{alloc}$ units of time and lasts until the system moves to the new scheme.

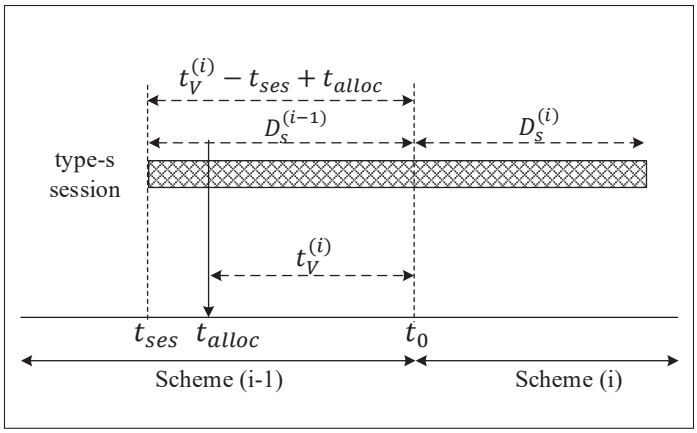
There is a possibility that a session can last for multiple schemes. For simplicity of analysis, we assume that the lifespan of a session is limited within two schemes. Regarding the delay, we use $D_{s,h\to k}^{(i)}$ apart from $D_{s,k}^{(i)}$ to define the delay from node L_s^h to node L_s^k of the service path. With $h = 1$, $D_{s,1\to k}^{(i)}$ can be implicitly understood as $D_{s,k}^{(i)}$. Given t_{alloc}, t_{ses}, $t_V^{(i)}$, $n_{v,\forall v\in V}^{(i-1)}$ and $n_{v,\forall v\in V}^{(i)}$, we can acquire the value of h and $k = h + 1$ such that $k \leq l_s$ as well as:

$$D_{s,h}^{(i-1)} \leq t_V^{(i)} - t_{ses} + t_{alloc} \leq D_{s,k}^{(i)} \tag{5.10}$$

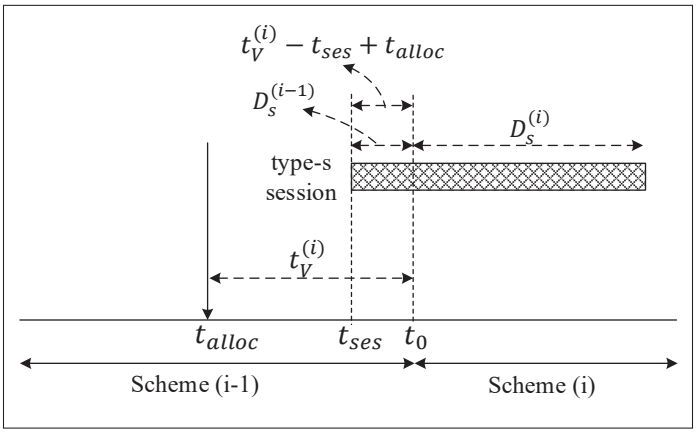
Intuitively, k is the index of the VNF of path L_s, from which its subsequent messages begin to be processed by the resource allocated. Using the same notations h and k defined above, the calculation of $D_s^{(i)}$ in (5.8) is generalized for the session across multi-schemes as follows:

$$D_s^e = E[D_{s,h}^{(i-1)}] + E[D_{s,k\to l_s}^{(i)}] + E[d_{L_s^h L_s^k}] = \sum_{p\in\mathcal{F}_1} d_p^{(i-1)} + \sum_{q\in\mathcal{F}_2} d_q^{(i)} + \sum_{u,u^*\in\mathcal{F}_1} (2 - p_{uu^*}) d_{uu^*}^{(i-1)} + \sum_{v,v^*\in\mathcal{F}_2} (2 - p_{vv^*}) d_{vv^*}^{(i)} \tag{5.11}$$

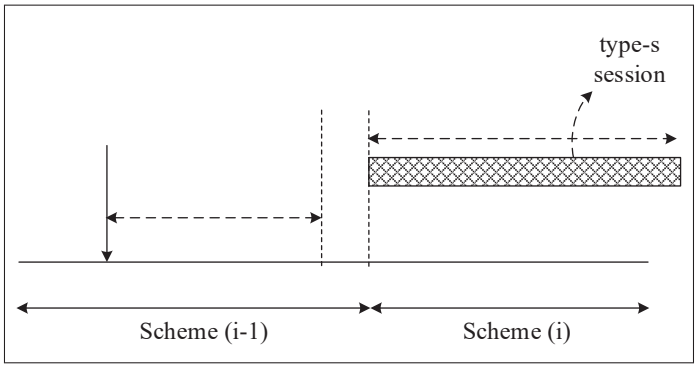
where $\mathcal{F}_1$, $\mathcal{F}_2$ are the set of the first h and the remaining $(l_s - h)$ elements of L_s. Note that equation (5.11) still holds in special cases when a session lasts for a single scheme, either starting by t_{alloc} and finishing by t_0 or starting after t_0, as shown in Figure5.6c. In the former case where $t_{ses} + D_s^{(i+1)} \leq t_0$, from (5.10), we have $h = l_s$ and k does not exist; consequently $E[D_{s,k\to l_s}^{(i)}] = E[d_{L_s^h L_s^k}] = 0$. In the latter, where $h = 0$, $k = 1$, we have $E[D_{s,h}^{(i+1)}] = E[d_{L_s^h L_s^k}] = 0$.



a) Service occurs before reallocation period



b) Service occurs during re-allocation period



c) Service occurs after reallocation period

Figure 5.6 Sessions over allocation periods

5.4.3 Cost Model

The total system cost is essentially contributed by three components: the computing cost of VNF configurations C_{com}, the cost of network bandwidth C_{net} and the penalty caused by QoS violation C_{sla}. Let c_r^n and c_b be the cost of allocating resource r in n and network bandwidth respectively at a unit of time. Then C_{com} and C_{net} are calculated based on the resource configured for VNFs and the allocated bandwidth between any two consecutive VNFs and the last one is computed by considering the overloaded VNFs which have insufficient resource and therefore overloaded during the period the system is moving to next scheme. C_{com} and C_{net} are computed as:

$$C_{com} = \max_{s \in S} D_s \sum_{n \in N} \sum_{r \in R} A_{n,r} c_r^n \tag{5.12}$$

$$C_{net} = \max_{s \in S} D_s \sum_{u,v \in V} \sum_{m,m' \in M} B_{m,m'} x_{u,m} x_{v,m'} c_b \tag{5.13}$$

To model the penalty cost, we first denote $f(s)$ as the fine function for every unit of time of SLA violation in terms of type-s session delay. During the scale-up event, any session taking place within t_{alloc} and t_0 will tolerate a longer latency than expected with a new allocation scheme. We introduce the notation $\Delta_s^{(i)}$ to represent the difference in delay for a message of a type-s session to traverse from node L_s^1 to node L_s^h under two consecutive schemes where h is computed by equation (5.10), that is, $\Delta_s^{(i)} = E[D_{s,h}^{(i)}] - E[D_{s,h}^{(i-1)}]$.

Since the value of h varies depending upon when a session starts, i.e., t_{ses}, we take into account the expected value $\Delta_s^{(i)}$. To facilitate the calculation of $E[\Delta_s^{(i)}]$, we assume that $t_{ses} \geq t_{alloc}$. In practice, this can be understood to mean that any session that begins by the time t_{alloc} will be processed by its current scheme until it is completed. $E[\Delta_s^{(i)}]$ can be retrieved from the expected value of h in equation (5.10), such that:

$$\begin{aligned} E[\Delta_s^{(i)}] &= E\left[E[D_{s,h}^{(i)}] - E[D_{s,h}^{(i-1)}]\right] \\ &= E[D_{s,E[h]}^{(i)}] - E[D_{s,E[h]}^{(i-1)}] \end{aligned} \tag{5.14}$$

where $E[h]$ satisfies the following condition:

$$D_{s,E[h]}^{(i-1)} \leq E[t_V^{(i)} - t_{ses} + t_{alloc}]$$
$$= t_V^{(i)} - E[t_{ses}] + t_{alloc} = \frac{t_V^{(i)}}{2} \leq D_{s,E[k]}^{(i)} \tag{5.15}$$

In order to obtain the total fine for a session, we retrieve the expected number of sessions of the same type occurring during the scaling-up period $\hat{n}_s^{(i-1)}$ as $\hat{n}_s^{(i-1)} = t_V^{(i)}\lambda_s$. Using the equations (5.14), we can derive the following average cost caused by SLA violation $U_{vlt}^{(i)}$ in terms of $f(s)$, $\Delta_s^{(i)}$ and $\hat{n}_s^{(i)}$ as:

$$E[U_{vlt}^{(i)}(s)] = E[\Delta_s^{(i)}\hat{n}_s^{(i-1)}f(s)] = E[\Delta_s^{(i)}]\hat{n}_s^{(i-1)}f(s) \tag{5.16}$$

Note that the delay is not affected when scaling down; this results in $\Delta_s^{(i)} \approx 0$ and consequently $U_{vlt}^{(i)}(s) \approx 0$. Thus, equation (5.16) can be used for both scale-up or scale-down cases.

Our goal is to find a scheme for allocating resources so as to minimize the total cost for deploying the whole system C_V while satisfying the constraints of service delay and the availability of resources. The value of C_V can be calculated in regard to the number of type-s sessions $\bar{n}_s^{(i)}$ for all $s \in S$ as:

$$C_V = C_{com} + C_{net} + \sum_{s \in S} \bar{n}_s E[U_{vlt}(s)] \tag{5.17}$$

However, instead of directly discovering the value of $\bar{n}_s^{(i)}$ by employing the same method for $\hat{n}_s^{(i-1)}$, which requires knowledge about the period of the unknown i^{th} scheme, we normalize C_V with $w_s = \bar{n}_s / \sum_{s \in S} \bar{n}_s$, which could be interpreted as the frequency of s and estimated with a sampling method based on historical statistical data, as follows:

$$C_V = C_{com} + C_{net} + \sum_{s \in S} w_s C_{V_s}(s) \tag{5.18}$$

We formulate the resource allocation cost minimization problem as follows:

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \quad C_V \tag{5.19}$$

$$\text{subject to} \quad (5.2), (5.6), (5.8)$$

$$\sum_{n \in N} y_{v,n} = 1, \forall v \in V \tag{5.20}$$

$$\sum_{n \in N} y_{v,n} \alpha_{m,n} = x_{v,m}, \forall v \in V, m \in M \tag{5.21}$$

$$\sum_{m \in M} x_{v,m} = 1, \forall v \in V \tag{5.22}$$

$$x_{v,m} \in \{0, 1\}, \forall v \in V, m \in M \tag{5.23}$$

$$y_{v,n} \in \{0, 1\}, \forall v \in V, n \in N \tag{5.24}$$

Constraint (19) and (5.20) ensure that a VNF is assigned with one configuration at a domain. Constraint (5.21) indicates that the configuration of a VNF must be available at the domain where that VNF locates.

5.5 Real-time Inter-Domain Resource Allocation

In this section, we describe the real-time resource allocation mechanism RIDRA regarding the presence of multiple domains. The optimal scheme is retrieved via either greedy approach or Markov approximation method. Having said that, our algorithm can be divided into two parts: i) adopting a batch learning approach on received messages to predict the moment when the workload changes and the allocation decision is made and ii) determining the new scheme by solving (5.19). By learning over groups rather than with each message, the overhead caused by continuously solving the optimization problem can be avoided. We use vectors $\mathbf{\Lambda}$ to represent the system workload in terms of the arrival rate at each VNF as $\mathbf{\Lambda} = \langle \lambda_1, \lambda_2, \ldots, \lambda_{|V|} \rangle$.

Algorithm 5.1 Real-time inter-domain resource allocation

```
1  Input: Knowledge base KB, a batch of messages B and its size sz
2  Output: Resource allocation schema e
3  B ← {};
4  while receiving a message ω do
5  |   B ← B ∪ {ω}
6  |   if size(B) < z then
7  |   |   continue
8  |   end if
9  |   Λ ← learning(KB, B)
10 |   B ← {}
11 |   update(KB)
12 |   if RIDRA's constraints are violated then
13 |   |   e ← ResAllocator()
14 |   end if
15 end while
```

In Algorithm 5.1, messages are received in a batch $\boldsymbol{B}$ (step 5) until the batch is full (steps 6-7). At step 9, the knowledge base built from historical data is leveraged together with the collected data in the batch to anticipate incoming arrival rates at VNFs. The KB is updated with new data at step 11. If the change of $\boldsymbol{\Lambda}$ causes a violation of constraints (step 12), the algorithm for solving (5.19) will be re-executed (step 13). The procedure $ResAllocator$ can be either $GreedyAllocator$ or $MARAllocator$.

5.5.1 Greedy Resource Allocation Mechanism

In Algorithm 5.2, the resource allocated for all the VNFs are performed in a step by step manner. The *next* procedure (step 3, 5, 11) returns the configuration that requires a minimum number of resource to the current one. The VNFs are preferably configured by the next option available in the same domain as illustrated at step 4 & step 8. If the resource for that VNF exceeds its limit, the VNF will be moved to another domain of the next configuration (step 11-11).

Algorithm 5.2 Greedy resource allocator

```
1  for v in V do
2  |  Set m as the current configuration of v
3  |  m' ← next(m)
4  |  while dom(m')! = dom(m) and m'! = NUL do
5  |  |  m' ← next(m')
6  |  end while
7  |  if m'! = NUL then
8  |  |  ⟨x_{v,m}, x_{v,m'}⟩ ← ⟨0, 1⟩
9  |  end if
10 |  else
11 |  |  m' ← next(m) ⟨x_{v,m}, y_{v,dom(m)}⟩ ← ⟨0, 0⟩ ⟨x_{v,m'}, y_{v,dom(m')}⟩ ← ⟨1, 1⟩
12 |  end if
13 end for
```

5.5.2 Markov-Approximation Resource Allocation Mechanism

Let $\mathcal{E}$ denote the set of all feasible schemes that satisfy all the constraints in RIDRA. Each scheme is represented as a 3-dimensional matrix $\mathcal{M}^{VxMxN}$. Adapting the idea in Chen *et al.* (2013), we consider that each scheme is a state in the Markov chain and is selected with a probability p_e. At any time, the best scheme will have the highest probability and the transition rate q between two candidate schemes (i), $(i+1)$ is defined as

$$\begin{aligned} \mathcal{T}((i) \rightarrow (i+1)) &= \frac{1}{1 + exp\big[-\theta(C_V^{(i)} - C_V^{(i+1)})\big]} \\ \mathcal{T}((i+1) \rightarrow (i)) &= \frac{1}{1 + exp\big[-\theta(C_V^{(i+1)} - C_V^{(i)})\big]} \end{aligned} \tag{5.25}$$

where θ is constant according to Chen *et al.* (2013). The transition rate is calculated according to $C_V^{(i)}$ and $C_V^{(i+1)}$. If $C_V^{(i)} < C_V^{(i+1)}$, then the system stays with scheme (i) due to $\mathcal{T}((i+1) \rightarrow (i)) \approx 1$ and vice versa. If $C_V^{(i)} = C_V^{(i+1)}$, then the system equally selects (i) or $(i+1)$.

The procedure MARAllocator starts with a very basic scheme by deploying VNFs at the domain with the configuration asking for the lowest resource (step 3-9). The next candidate $(i+1)$ is generated from (i) by randomly selecting a VNF i and assigning a random configuration at a

Algorithm 5.3 Markov Approximation-based resource allocator

1 **Input:** *maximum count (stopping criteria) MC*
2 **Output:** *Resource allocation schema*
3 Update constraint (5.6) with predicted $\lambda_{\forall v \in V}$
4 **if** $x_{v,m}$ & $y_{v,n}$ *is not set* **then**
5 | Set v at domain with smallest configuration
6 **end if**
7 **else**
8 | $\langle x_{v,m}^{(i)}, y_{v,n}^{(i)} \rangle \leftarrow \langle x_{v,m}, y_{v,n} \rangle$
9 **end if**
10 $count \leftarrow 0$
11 **while** $count < MC$ **do**
12 | $\mathcal{M}^{(i+1)} \leftarrow copy(\mathcal{M}^{(i)})$
13 | $\{i, j\} \leftarrow \{rand(1, |V|), rand(1, |M|\}$
14 | $\mathcal{M}^{(i+1)}[i, :, :] \leftarrow 0$
15 | $\mathcal{M}^{(i+1)}[i, j, dom(j)] \leftarrow 1$
16 | **if** $rand(0, 1) > \mathcal{T}((i) \rightarrow (i+1))$ **then**
17 | | **continue**;
18 | **end if**
19 | **if** $C_V^{(i)} > C_V^{(i+1)}$ **then**
20 | | $\mathcal{M}^{(i)} \leftarrow copy(\mathcal{M}^{(i+1)})$
21 | **end if**
22 | **else if** $C_V^{(i)} \leq C_V^{(i+1)}$ **then**
23 | | $count \leftarrow count + 1$
24 | **end if**
25 **end while**

corresponding domain $dom(j)$ (step 13-15). This new scheme is chosen according to transition probability (step 16) and the cost difference between the current scheme and the newly generated scheme (19). The procedure repeats until the stopping criteria are met when there is no longer any significant cost improvement (step 22).

5.5.3 Discussion

5.5.3.1 Deployment of Algorithm

Regarding the architecture in Figure5.1, the ISO receives the workload information in terms of the arrival rate at each VNF which is reported by the DRO at each domain. The resource allocator is deployed at either the ISO or the DRO depending on which approach is used. The GreedyAllocator is executed at the ISO to make the decision of allocating resource and have the descriptors of service chains at each domain updated or re-configured accordingly. Unlike the GreedyAllocator, the MARAllocator is mainly performed by the DRO components at involved domains. The ISO only keeps the list of sub-optimal schemes. The DROs consults the ISO whenever it generates a candidate state and keeps doing that until a new state is obtained. A scheme that is verified to be suboptimal is then informed to the ISO to update the list. By doing so, the DRO can benefit from the results from the others and as a result, reduce the execution time to find the optimal solution.

5.5.3.2 Time Window to Update Parameters

In order to achieve an efficient system utilization, the time interval between two consecutive $\mathbf{\Lambda}$ checks must be longer than the system transition time $t_V^{(i)}$. In Algorithm 5.1, this interval is controlled by the pre-defined and static batch size z. While this approach is easy to implement, it is challenging to determine an optimal value of z. If z is small, $\mathbf{\Lambda}$ may change before the system completes moving to the next scheme. In contrast, a large batch size may cause the missing of scaling the resource in the presence of highly fluctuated service requests. A possible way to mitigate this issue is to utilize the dynamic batch size with a traffic model or context awareness. For instance, from Cao, Chi, Hao & Xiao (2008), z can set high when the request rate is quite stable between 01:00 and 08:00 and low between 08:00 and 24:00 when the demand significantly changes every hour. Determining the optimal number of messages or the length of the lookahead interval for each batch is an interesting research challenge that is beyond the scope of this work, and we plan to pursue it in the future.

Regarding the running time, Algorithm 5.1 performs continuously $O(z)$ computations for each message it receives. The constraints' violations are examined through $O(|V||S|)$ sessions for the delay constraints (5.8), $O(|V||M|)$ VNFs for the traffic intensity (5.6), and $O(|V||R||S||M||N|)$ times for resource constraints (5.2). We also notice that the computation complexity of the learning procedure depends on the selected method. With the $ARIMA(z, d, q)$ model used in accordance with Calheiros, Masoumi, Ranjan & Buyya (2015), once values from the knowledge base are available, it has complexity $O(z)$ where z is the batch size and also the order of the autoregressive component. The total complexity for checking the constraints affected by the arrival rate is $O(max(z^2, z|V||R||S||M||N|))$.

5.5.3.3 RIDRA Convergence Speed and Optimality Gap

The performance of the RIDRA is mainly dependent on how fast the MARAllocator converges. Although MARAllocator can find a close-optimal solution, one challenge that Markov-based approaches typically face is low convergence speed due to the exploration of candidate solutions on a huge feasible set. In the worst case when all the sessions require the involvement of $|V|$ VNFs, the size of the state space in the Markov chain is $|\mathcal{E}| = |M|^{|V|}$. One method to lessen this obstacle is to perform the MARAllocator in parallel and distributed manner regarding the generation of the next state. Also, as an approximation approach, the optimality gap is also considered. The MARAllocator is approximated by an entropy term $\frac{1}{\theta}\sum_{e\in\mathcal{E}} p_e log(p_e)$. The gap is therefore computed as $\frac{1}{\theta} log|\mathcal{E}|$, or $O(|V|logM)/\theta$. As mentioned in Chen *et al.* (2013), the parameter θ can be adjusted as a trade-off between the requirement of fast convergence as well as small optimality gap and the system performance.

5.6 Performance Evaluation

This section presents simulation and experimental results to show the total cost improvement of the proposed RIDRA during the scaling phase while satisfying the resource and delay requirements for service chains. We deploy VNFs with different templates onto VMs with

corresponding templates. The system is simulated for large-scale evaluation and deployed as a testbed in the data center for small-scale experiments to validate the algorithm.

5.6.1 Simulation Analysis

5.6.1.1 Simulation Setting

The simulation is performed with seven templates characterized by the number of virtual CPU, the amount of virtual memory and network bandwidth. The resource settings for VM templates are set according to those offered by Microsoft Azure IaaS provider, as shown in Table **??**. To avoid unexpected delay caused by the connection to the remote system when measuring startup time Mao & Humphrey (2012), a local OpenStack-based cloud platform is used. Moreover, we conduct an experiment on these local VMs for the processing delay metric due to the lack of relevant studies. Additionally, CPU utilization of VMs is limited at 85%, (i.e., $\beta = 0.85$) and a SIP traffic generator is used to simulate the requests from users to the IMS system. We compute the processing delay as the interval time of two consecutive messages in a SIP transaction.

Table 5.3 Parameters of VM templates used in the simulation process

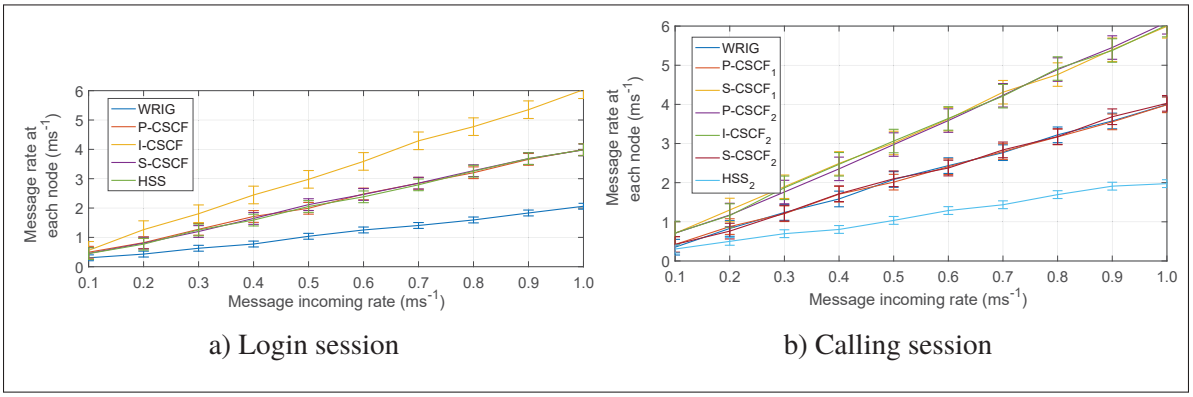
Template	vCPU	vMem (GB)	Startup time (s)	Processing time (ms)
vi.tiny	1	1	4.3	1.2
vi.small	1	2	4.0	0.8
vi.medium	2	4	4.0	0.7
vi.large	2	8	4.0	0.7
vi.xlarge	4	8	7.0	0.4
vi.2xlarge	4	16	7.0	0.3
vi.4xlarge	8	16	12.0	0.2

The set V is composed of nodes in three networks: WebRTC, IMS_1 and IMS_2. We designed our simulation with relevant VNFs for login and calling sessions. The connection probability and transmission delay are different between intra- and inter-domain VNFs. All parameters used for the proposed optimization approach are listed in Table 5.4. The resource cost is normalized as the weights in Ye *et al.* (2017). The return value of $f(S)$ is calculated on the basis of how

long a user is willing to wait in proportion to the session delay when SLA is violated Wu, Garg & Buyya (2012). About the delay between the endpoints and the server, the parameters in (5.3) are configured according to Munir & Gordon-Ross (2010), in particular, $K = 1$, $\tau = 2.5ms$, $d^{frm} = 0.049ms$, $N^{TCP} = 3$ and $\bar{q} = 0.02$.

Table 5.4 Parameters used for the proposed optimization approach

Parameter	**Value**
Resource set R	{CPU, Memory}
Session set S	{login, calling}
Resource cost $c^n(R)$	$\{\mathcal{N}(5, 0.2), \mathcal{N}(2, 0.2)\}$
Network cost c_b	$\{\mathcal{N}(3, 0.2)\}$
Session fine $f(S)$	{2.5, 5}
Connection probability p_{uv}	0.80 (inter), 0.95 (intra)
Transmission latency ω_{uv}	$\mathcal{N}(0.7, 0.02)$ (intra), $\mathcal{N}(1, 0.02)$ (inter)



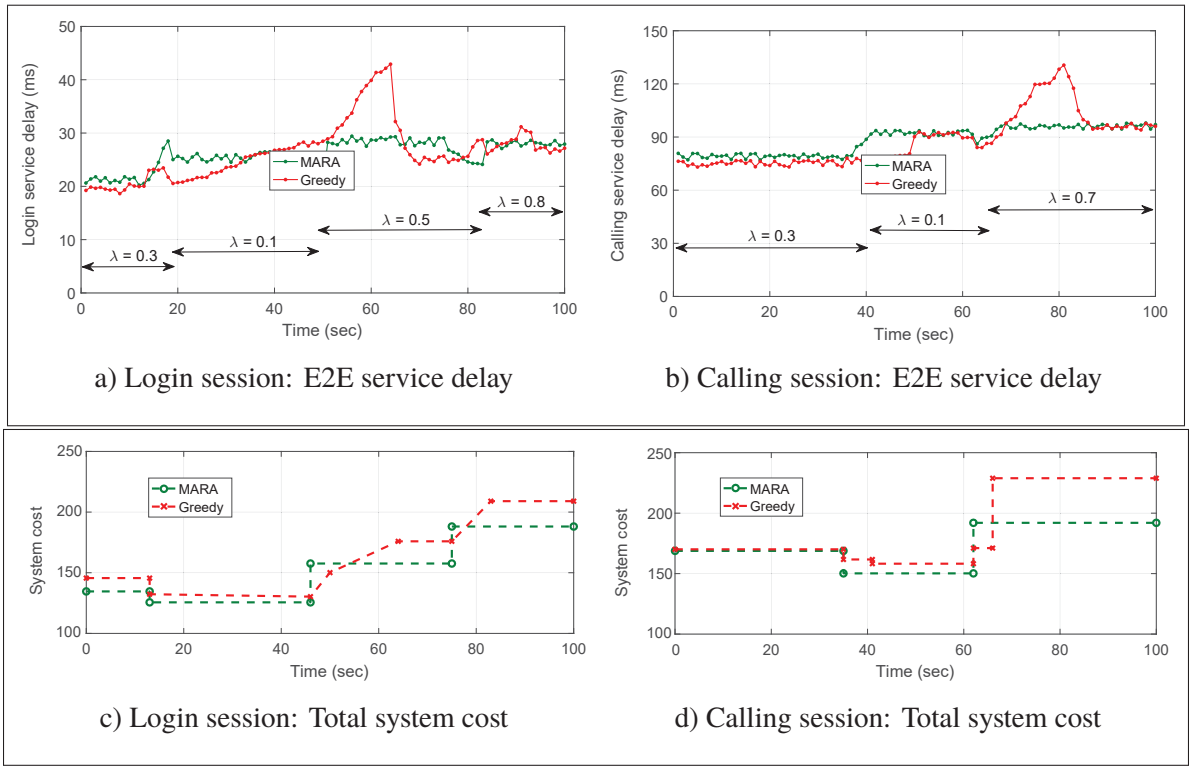
a) Login session b) Calling session

Figure 5.7 Message incoming rates at VNFs given initial request rate

We simulate the interworking system as a network using the Simulink library in a MATLAB environment. A node is characterized as a processing module in the form of multiple servers that are homogeneous and therefore process messages at the same rate. The arrival rate of a service chain is controlled by modifying the intergeneration time at the request generator, which plays a role as the WebRTC client in Figs. 5.4 and 5.5. While the processing time is constant at every node, the change of λ_s represents the change of request intensity or service demand. We also adopt a simple Exponential Moving Average (EMA) model to predict λ_s.

5.6.1.2 Simulation Results

Given the message arrival rate from the endpoint, it is necessary to retrieve the arrival rate at each VNF. To obtain these values, we conduct experiments on local VMs where the VNFs are deployed, by having the WebRTC client generate requests at different intervals and measure the arrival rate at each VNF. To reduce the adverse impact of out-of-band traffic, these VMs are configured with minimum resources and can be considered negligible to those of the hypervisors. Figure5.7 demonstrates the incoming message rate at each node given λ_s. In the long run, a VNF exchange messages with the others as defined by its corresponding message flow. In Figure5.7a, the WRIG node receives a message from both WSF and P-CSCF, whereas during the calling phase (Figure5.7b), there are also other messages arriving at the WRIG from ICE-enabled servers.



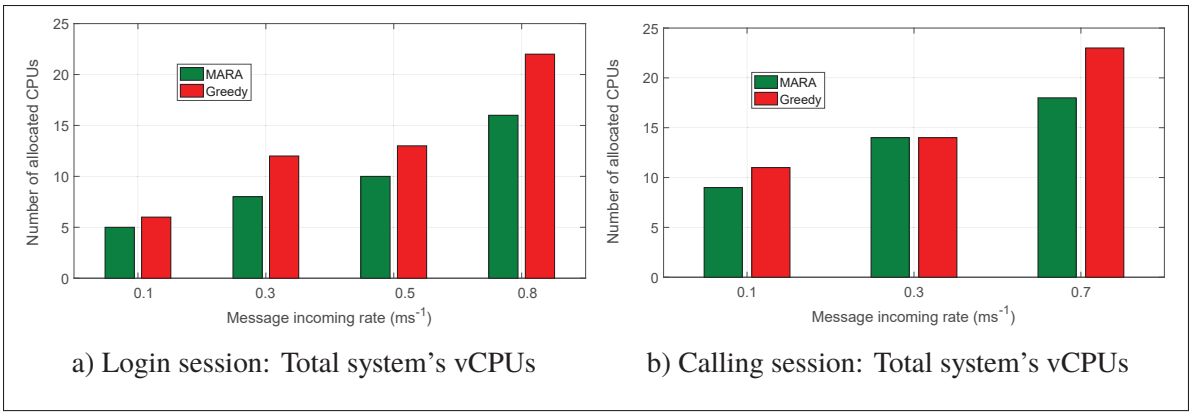
a) Login session: E2E service delay

b) Calling session: E2E service delay

c) Login session: Total system cost

d) Calling session: Total system cost

Figure 5.8 Comparison of system cost and latency between two approaches



a) Login session: Total system's vCPUs b) Calling session: Total system's vCPUs

Figure 5.9 Comparison of system resource between two approaches

The I-CSCF node is the most stressed node during the login phase since it communicates with all the IMS nodes, whereas the other nodes, i.e., P-CSCF, S-CSCF, and HSS, only interact with other two entities. Similarly, the major load of calling requests is handled by S-CSCF in IMS network 1, and by P-CSCF, I-CSCF in IMS network 2.

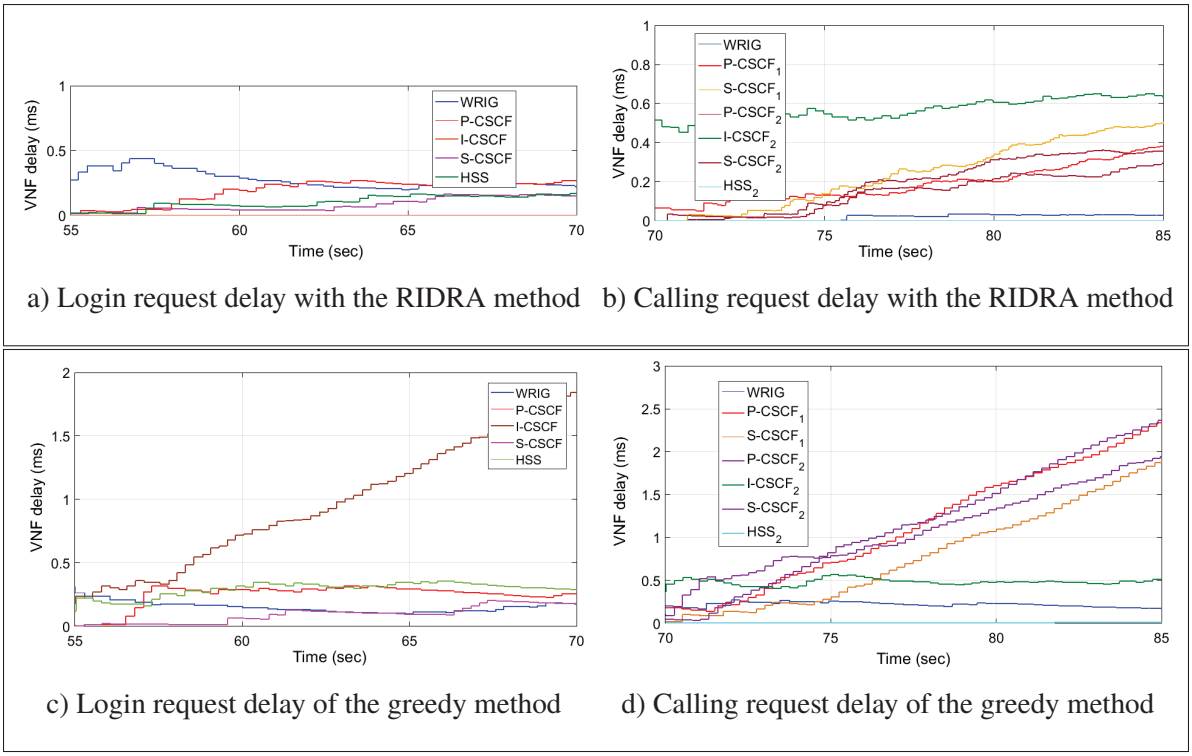
On the basis of the relation between λ_s with $\lambda_{v,s} \forall v \in V$, we performed the simulation continuously during 100 seconds to see how the resource is adapted according to the different values of λ_s. The workload predictor detects the change of λ_s several seconds before it occurs. We concentrate on the comparison of total system cost taking into account the service latency between the two allocation approaches: i) the proposed optimal approach with RIDRA, and ii) the greedy approach as the baseline by which the resources are greedily allocated as long as the constraints (5.2), (5.6), (5.8) are satisfied.

Figure5.8a-5.8d illustrate how RIDRA helps reduce the system cost while respecting the session delay bound. Comparing to the greedy scheme, there are some situations in which RIDRA uses fewer resources and lead to a longer delay. This may be clearly seen during the period $\lambda_s = 0.1$ in Figure5.8a & 5.8b. We also observe that RIDRA works better in the presence of a high workload change. For instance, when λ_s changes from 0.1 to 0.5 in Figure5.8a, the greedy approach is unable to allocate immediately enough resource, and thus the latency keeps increasing over the upper bound. This issue does not happen with RIDRA because the optimal

number of the resource has been already allocated in advance. That is why there is a small period during which the system is over-provisioned around the 50^{th}. Similarly, the advantage of RIDRA is always illustrated in Figure5.8b when λ_s changes from 0.1 to 0.7.

Regarding the total system cost, RIDRA can generally save on the cost up to 19% compared to the greedy algorithm. Under normal circumstances when the system is stable, the difference between the two approaches is not significant. However, when the demand exceeds the capacity of the system to some extent, the greedy approach not only degrades the service performance with more severe SLA violations but also is more wasteful of resources and consequently results in a much higher cost than that of RIDRA.

In order to get more insights into a system's behavior, we investigated the total number of vCPUs allocated for the whole interworking system between two methods. The results shown in Figure5.9 are aligned with what is observed about the system cost in Figure5.8c & 5.8d. We also notice the waiting time of requests at server nodes during the peak time of service demand, i.e., between the 55^{th} and 70^{th} seconds in Figure5.8a, and between the 70^{th} and 85^{th} seconds in Figure5.8b. As seen in Figure5.10, the service demand is equally distributed at VNFs with optimally allocated resources, whereas the load distribution is highly biased under the greedy scheme. Up to some extents, the greedy allocation method helps mitigate the issue of violating delay constraint regarding the resource limit. However, it is not efficient when facing with the burst workload because the newly added resource at some VNFs leads to the high demand at the others which may not have sufficient resource to handle and eventually degrade the entire service performance. In Figure5.10b, it takes a very short time for the calling requests to be processed by the HSS of IMS network 2. This is because HSS_2 is involved only one time, compared to P/S-$CSCF_1$, P/S-$CSCF_2$ with 11 times, and I-$CSCF_2$ with three times, to process calling requests and even the VM with minimum settings is enough to handle them. Note that in Figure5.10c, the delay is mainly caused by the bottle-neck at I-CSCF; consequently, the resources at other nodes are over-provisioned with very low processing latency. This issue does not happen in the calling session since there are more VNFs involved and more sharing of the load.



a) Login request delay with the RIDRA method b) Calling request delay with the RIDRA method

c) Login request delay of the greedy method d) Calling request delay of the greedy method

Figure 5.10 Request processing delay for login and calling sessions at each VNFs

5.6.2 Experimental Analysis

5.6.2.1 Testbed Configuration

Our testbed is configured with eight server blades each of which has 12 physical CPUs and 96GB of memory. The delay and transmission probability of the connections between VMs are controlled via scripts. All the VMs' CPU usages are limited to 85%. On the server side, we installed OpenIMSCore as an IMS system. In current WebRTC-enabled browser (i.e., Chrome, Firefox), the ICE sessions are implemented as a built-in function and automatically revoked whenever a calling session is about to establish. The unexpected involvement of ICE entities (e.g., STUN, TURN) may adversely impact on the efficiency and reliability of the proposed RIDRA algorithm. To overcome this issue, we manage to include pre-configured ICE information to the Session Description in calling requests. We use Janus Amirante *et al.* (2014) as the WRIG with

a plugin intercepting requests from the WebRTC client and convert them to corresponding SIP requests according to the sender information.

On the WebRTC client's side, we deployed five laptops which operate as either IMS clients or WebRTC clients. Since there is no existing WebRTC traffic generator, we developed an automation testing tool for the Web applications that can be applied to most Web browsers. The script helps open new tabs automatically, filling in the required information (e.g., URL of the Web server, user login information, callee name, etc.) and clicking buttons (e.g., login, call). The script could also control the request rate by changing the interval time between two consecutive clicks on *login* or *call* buttons.

To conduct the experiment, we first launched 40 WebRTC browser instances on each laptop. Then the auto script was performed at different intervals to represent the login request rate. The end-to-end delay is measured at browser clients while the VNF delay is computed at VMs. For calling testing, due to the limited number of Webcams for each WebRTC client, we could not measure the delay of the whole session. Instead, we focused on successful Session Request Delay metrics as explained in Malas & Morton (2011).

5.6.2.2 Experimental Results

We started the experiment with basic configuration whereby each VNF is assigned to a VM of *vi.tiny* VM. All the signed-in WebRTC sessions are automatically logged out before the service request rate increases, and similarly with the calling sessions. For the calling experiment, we divided the laptops into two groups, playing roles of caller and callee, respectively. We conducted 20 experiments for different cases to explore how QoS is guaranteed and how efficiently allocated VNF operate from the perspective of CPU usage during the peak time of service demand. The results are presented in Figs. 5.11 and 5.12.

From Figure5.11, the cost difference between RIDRA and the greedy algorithm is evaluated as the workload of each service chain corresponding to login and calling sessions increases. It can be seen that the system cost incurred by RIDRA is always less than that when using the greedy

algorithm. We also see that as the rate increases, the greedy algorithm is much slower than RIDRA in allocating enough resource and as a result causing a higher possibility of violating SLA as well as the increase in cost.

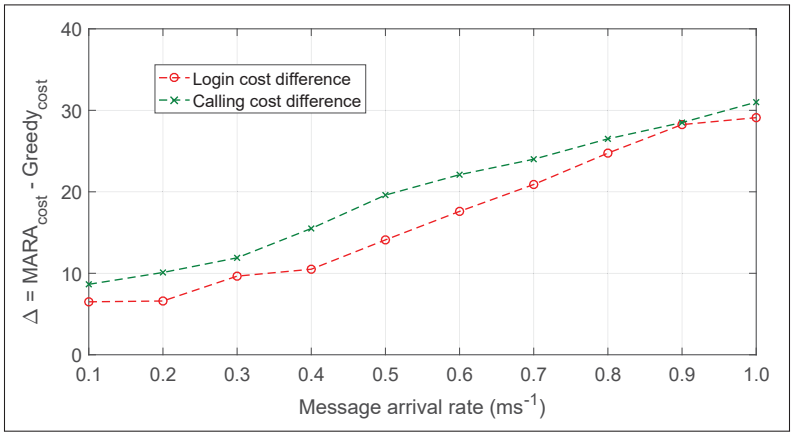


Figure 5.11 Service cost optimized by RIDRA

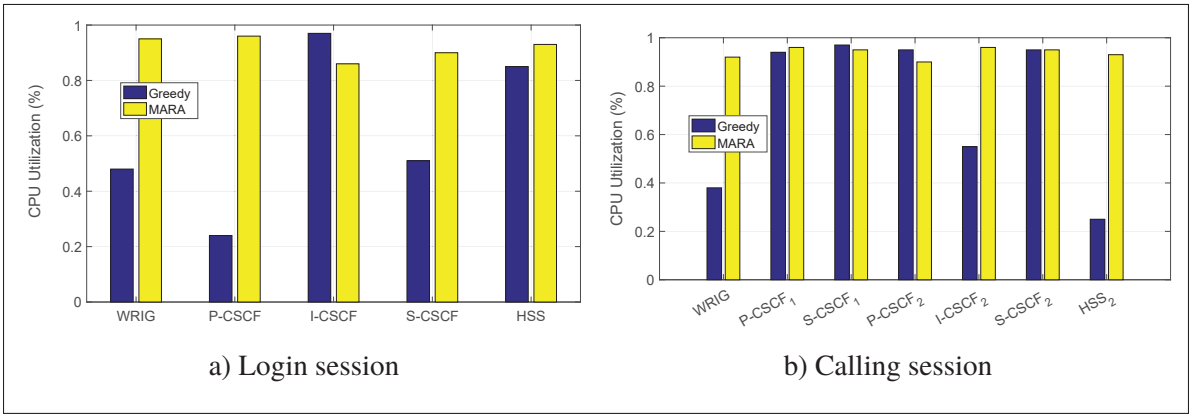


Figure 5.12 CPU usage comparison with two approaches during the peak time

In Figure5.12, during the peak period of service demand, the virtual resources are efficiently provisioned in such a way that there is no significant difference between the VMs' CPU usage. Under the greedy scheme, the overhead at several VNFs (i.e., I-CSCF in a login session, or P-CSCF and S-CSCF in a calling session) results in a decrease of service demand rate at other VNFs; consequently, the resource allocated for such the VNFs is not efficiently used.

5.7 Discussion and Future Work

The virtualization paradigm is being increasingly adopted by organizations in order to deploy their services more effectively and efficiently thanks to its elasticity of resource provisioning. However, such an advantage comes with the challenge of obtaining an optimal resource allocation method at VNF level in the WebRTC ↔ IMS interworking systems across multiple domains and handled by multiple providers.

In this paper, we highlight the key components of the NFV-based interworking architecture between WebRTC and IMS domains in terms of the architecture NWII, its realization for WebRTC ↔ IMS scenario, and service chains between VNFs for login and calling use cases. The architecture is designed based on 5G Operating System reference architecture with respect to the heterogeneous requirements of different providers, domains and the adaptation to any change of IMS or WebRTC specification.

Given the service chains through VNFs, we define an analytical model of service cost in regard to system resources and service latency. We formulate our objective that aims to minimize service cost given VNFs in configurations available in various domains. We present a real-time inter-domain algorithm (RIDRA) to optimally allocate resources in order to reduce service cost regarding QoS requirements. Furthermore, RIDRA is deployed in a distributed manner to leverage the computation in parallel.

The proposed RIDRA has been fully implemented and tested via simulation and the testbed for different settings. The workload relation between VNFs in terms of message rate under ideal conditions is determined via simulation. The experimental results show that our optimal allocation resource scheme saves on service cost up to 19% depending on the service chain and service demand.

Despite its advantages, the limitation of RIDRA caused by MARAllocator regarding the slow convergence and the optimality gap may result in SLA violations. While this issue is partially mitigated by using a control parameter, it requires powerful computing capacity. In practice, the

RIDRA adoption is impractical when the communication between users involves more VNFs, and the Markov state space becomes larger. This opens up a number of direction for future investigation. First, the convergence of MARAllocator can be enhanced either via the control parameter or combining with other approaches, e.g., game theory. Another possibility is to implement MARAllocator in a distributed manner in systems that are typically more robust and dynamic than those running centralized algorithms. Second, we intend to consider other scenarios beyond login and calling such as ICE-related sessions which play a significant role in the practical systems regarding their massive deployment due to security reasons in communication services. Especially, considering the sequential or concurrent execution of ICE service chains with the calling ones within a distributed NFV infrastructure, either MARAllocator needs to be modified or effective mechanisms must be devised to meet the desideratum. Third, the MARAllocator implementation should take into account the case in which the shared information between domains is affected due to the involvement of policy and business concerns.

Acknowledgment

The authors thank NSERC and Ericsson for funding the project CRDPJ 469977. This research also receives support from the Canada Research Chair, Tier 1, hold by Mohamed Cheriet.

CHAPTER 6

SUMMARY AND DISCUSSION

The general objective of this thesis has been to design a holistic resource provisioning solution for edge-cloud SDN-based network with respect to real-time requirements of multi-domain interworking IoT services. Our work is based on a hypothesis of improving entire system performance with stability, enhanced QoS, QoE as well as minimizing the total system cost given optimized allocation of VNF resource at SDN controlling and forwarding planes for dynamic demand of interworking multi-domain services. The proposed methodology consists of three themes, which we covered in this work: Chapter 3 introduced an orchestration system that enables to collect resource profiles relevant to VNF placement and network traffic sent by IoT devices via IoT gateways as input data for IoT services. We leverage the knowledge of IoT network topology to improve the E2E service latency model and the performance of resource allocation algorithm. Chapter 4 presented a solution for jointly optimizing resource allocation for both SDN controlling and forwarding planes given dynamic service demand while maintaining the entire system's queue stability with minimal resourse usage's cost. Finally, a real-time algorithm of optimizing resource allocation for an edge-cloud enabled interworking network system regarding E2E service latency over the time and potential QoS violation penalties is proposed in Chapter 5. The scalability of our proposed algorithm is justified via experimental results in that system resource is optimally scaled up or down according to fluctuating service demand or network size. Each theme is the subject of a separate published journal article to disseminate as widely as possible. Below, we highlight the strengths and weaknesses of the proposed methods as reflected in each theme.

6.1 IoT Network Topology Aware VNF Placement

The first theme covers the issue of placing VNF in NFV-based edge cloud systems with the aim of offering better IoT traffic processing by taking IoT network topology into consideration. In chapter 3, we defined an analytical model of system cost in terms of computation resource

and network bandwidth with regard to service latency and the availability of each resource at edge clouds. The problem of minimizing the total system cost is formulated with respect to constraints on available resource and QoS requirements as well as two algorithms for small and large-scale network settings are proposed based on the Markov approximation framework and a node ranking heuristic. This method has been defined in an article published by IEEE Transactions of Network and Service Management. We implement these two algorithms and validate their performance via simulation and testbed. The testbed is configured according to an IoT-based surveillance use case. The results show that the consideration of IoT network topology in making VNF placement decision can save on system cost up to 21% depending on the size of the network.

6.2 Optimizing SDN Controller Placement for Efficient VNF Traffic Routing

In Chapter 4, we modeled the joint optimization problem of VNF resource allocation for both SDN controlling and forwarding planes in terms of the placement of VNFs onto physical nodes and the mapping between SDN controllers and forwarding nodes. The problem is considered with a large number of nodes as inputs from a large-scale system where heterogeneous IoT services are hosted. The probabilistic change of service demands and transmission latency from end-users to network gateways caused by user moving is taken into account. To compute an optimal placement solution while ensuring the system's queue stability, we adopt the Lyapunov optimization framework and implement the exponential gradient ascent method. This method has been submitted to IEEE Transactions of Network and Service Management. The performance is validated by extensive experiments simulating context-aware service demand. The results show that our proposed solution can save up to 15.5 $\sim$ 18.2% of the total system costs depending on the fluctuation rate of service demand.

6.3 E2E service latency modeling over the time for multi-domain interworking IoT services

In order to provide end-users in smart environments with optimal QoE, three characteristics of IoT services need to be considered, which are low E2E service latency requirements, a service can be composed of various service components deployed on multiple domains, and service can be exposed to fluctuate service demand over time windows. In Chapter 5, we presented a model to minimize the total system cost with respect to these characteristics. We discuss in detail how service components are linked and exchange messages to construct interworking IoT-enabled scenarios between IMS and WebRTC domains. Regarding highly fluctuating service demand, it is necessary to have a real-time algorithm for the strategy of optimally allocating resource every time slot. Our approach is to employ a batch learning technique on received messages to determine when service demand changes and then make an allocation decision by solving an optimization problem using Markov approximation method. To demonstrate the cost improvement of the proposed solution during the scaling phase, we deploy a testbed in the data center and compare our solution with greedy mechanism. The result reveals that our algorithm effectively responds to fluctuating service demands with a service cost reduced by 19% via efficiently allocating virtual resources while maintaining QoS requirements. This study has been published in IEEE Transactions of Network and Service Management.

CONCLUSION AND RECOMMENDATIONS

7.1 General Conclusion

To improve the quality of human life, IoT paradigm is bringing several new challenges that traditional networks' solutions are insufficient to handle. The introduction of interworking IoT services over multiple domains and the explosion in the number of heterogeneous IoT devices require an efficient communication control and novel service deployments considering the adoption of SDN paradigm and NFV technology. In particular, such IoT services with low latency requirements are usually facing an ever-increasing demand with a high traffic fluctuation. For NFV-based edge cloud systems, it is therefore necessary to obtain a strategy of placing resource taking into account IoT network topology to meet QoS criteria. From SDN point of view, various SDN controllers need to be orchestrated and placed toward forwarding planes so as to efficiently steer VNF traffic. A combination of optimizing resource allocation for both SDN controllers and VNFs at forwarding planes with a model of multi-domain service latency is important to enhance entire network scalability and maintain low system cost.

In this work, our first attempt was to tackle the IoT Network Topology-Aware Placement problem for optimizing dynamic resource allocation in NFV-based edge cloud systems. We consider IoT service chains composed of multiple VNFs that are geographically deployed onto edge clouds close to IoT endpoints. The VNFs communicate not only with each other but also with IoT gateways that typically aggregate data from IoT sensor network as contextual information into discrete messages and forward them toward VNFs at the server side. Then an analytical model of system cost in terms of computation resource and network bandwidth with regard to service latency and the availability of each resource at edge clouds is defined. The problem of minimizing the total system cost is formulated with respect to constraints on available resource and QoS requirements. To obtain an optimal placement solution, two algorithms for small and large-scale network settings are proposed respectively, namely a Markov-based approximation

approach that leverages the presence of multiple edge cloud to adopt multistart and batching techniques, and a node ranking heuristic. We implement these two algorithms and validate their performance via simulation and testbed. The testbed is configured according to an IoT-based surveillance use case. The results show that the consideration of IoT network topology in making VNF placement decisions can save on system cost up to 21% depending on the size of the network.

We go a step further by extending the scope toward control plane in the context of SDN network. Various resource controllers are orchestrated with the ultimate goal to deliver E2E IoT service, which appears to be robust to the fluctuating demand nature of IoT services and different execution manners of their functions. We model the optimal placement problem of VNF resource and network connectivity between SDN and forwarding switches. The problem is considered with a critical number of nodes from a large-scale system where heterogeneous IoT services are hosted. The probabilistic change of service demands and transmission latency from end-users to the core network where VNFs are deployed are also taken into account. To obtain an optimal placement solution while ensuring the system's queue stability, we adopt the Lyapunov optimization framework and implement the exponential gradient ascent method. The performance is validated via simulation and testbed in which dynamic service demand is involved and shows the advantage of our proposal in saving total system cost up to 15.5 18.2% depending on service demand and fluctuation rate.

Finally, we presented an NFV-based architecture for multi-domain interworking IoT services and model E2E service latency provided with the details of message flows between service functions of WebRTC $\leftrightarrow$ IMS applications. In the model, we take into account the fluctuating service demand over time slots and the cost incurred during the resource migration phase. We formulate our objective that aims to minimize service cost given VNFs in configurations available in various domains and design a real-time inter-domain algorithm (RIDRA) to optimally allocate

resources in order to reduce service cost regarding QoS requirements. The proposed RIDRA has been fully implemented and tested via simulation and the testbed for different settings. The workload relation between VNFs in terms of message rate under ideal conditions is determined via simulation. The experimental results show that our optimal allocation resource scheme saves on service cost up to 19% depending on the service chain and service demand.

In future work, we will focus on: i) improving the performance of Markov-based approximation approach by combining with other techniques, i.e. game theory in selecting next matching candidates so as to increase convergence time, ii) including the mobility of IoT devices, that requires to update the proposed model to reflect dynamic connectivities between VNFs and IoT gateways, iii) extending E2E service latency model to other interworking scenarios apart from logging and calling scenarios as mentioned in Chapter 5, iv) deploying the proposed solution onto container orchestration platform like Kubernetes to evaluate its applicability in practice. In addition, we also investigate In-Network computing paradigm and explore its capabilities in improving network system performance for IoT-enabled last-mile delivery.

7.1.1 Major contributions

The major contributions of this thesis are:

- Jointly Optimizing SDN Controller and VNF Resource Placement (SCVP): VNF resource allocation model that considers both SDN control and forwarding planes in terms of switches and physical machines given fluctuating service demand and the requirement of the entire control system's stability.
- IoT Network Topology-Aware VNF Placement Model: VNF resource allocation model that takes into account the impact of VNFs' input traffic at the lower granularity level of discrete messages via connections between IoT networks and clouds.

- Real-time Inter-Domain Resource Allocation: A model of E2E service latency for multi-domain interworking services and an optimal real-time method of resource allocation that is robust in solving the combinatorial optimization problem and efficient with the distributed implementation.
- Deploy an NFV-based interworking testbed for WebRTC ↔ IMS which is used to justify the improvement of all the proposed algorithms.

BIBLIOGRAPHY

3GPP. (2018). *IP Multimedia Subsystem* (Report n°23.228). Retrieved from: http://www.3gpp.org/ftp/Specs/2018-09/Rel-15/23_series/23228-f30.zip.

Aazam, M. & Huh, E. (2014, Aug). Fog Computing and Smart Gateway Based Communication for Cloud of Things. *2014 International Conference on Future Internet of Things and Cloud*, pp. 464-470. doi: 10.1109/FiCloud.2014.83.

Aazam, M., Harras, K. A. & Zeadally, S. (2019). Fog Computing for 5G Tactile Industrial Internet of Things: QoE-Aware Resource Allocation Model. *IEEE Transactions on Industrial Informatics*, 15(5), 3085-3092. doi: 10.1109/TII.2019.2902574.

Abhayawardhana, V. S. & Babbage, R. (2007, April). A Traffic Model for the IP Multimedia Subsystem (IMS). *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, pp. 783-787. doi: 10.1109/VETECS.2007.171.

Adam, O., Lee, Y. C. & Zomaya, A. Y. (2017). Stochastic Resource Provisioning for Containerized Multi-Tier Web Services in Clouds. *IEEE Transactions on Parallel and Distributed Systems*, 28(7), 2060-2073. doi: 10.1109/TPDS.2016.2639009.

Agarwal, S., Malandrino, F., Chiasserini, C. & De, S. (2018, April). Joint VNF Placement and CPU Allocation in 5G. *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pp. 1943-1951. doi: 10.1109/INFOCOM.2018.8485943.

Agarwal, S., Malandrino, F., Chiasserini, C. F. & De, S. (2019). VNF Placement and Resource Allocation for the Support of Vertical Services in 5G Networks. *IEEE/ACM Transactions on Networking*, 27(1), 433-446. doi: 10.1109/TNET.2018.2890631.

Alasaad, A., Shafiee, K., Behairy, H. M. & Leung, V. C. M. (2015). Innovative Schemes for Resource Allocation in the Cloud for Media Streaming Applications. *IEEE Transactions on Parallel and Distributed Systems*, 26(4), 1021-1033. doi: 10.1109/TPDS.2014.2316827.

Alleg, A., Ahmed, T., Mosbah, M., Riggio, R. & Boutaba, R. (2017, Nov). Delay-aware VNF placement and chaining based on a flexible resource allocation approach. *2017 13th International Conference on Network and Service Management (CNSM)*, pp. 1-7. doi: 10.23919/CNSM.2017.8255993.

Allybokus, Z., Perrot, N., Leguay, J., Maggi, L. & Gourdin, E. Virtual function placement for service chaining with partial orders and anti-affinity rules. *Networks*, 71(2), 97-106.

Alsarhan, A., Itradat, A., Al-Dubai, A. Y., Zomaya, A. Y. & Min, G. (2018). Adaptive Resource Allocation and Provisioning in Multi-Service Cloud Environments. *IEEE Transactions on Parallel and Distributed Systems*, 29(1), 31-42. doi: 10.1109/TPDS.2017.2748578.

Amirante, A., Castaldi, T., Miniero, L. & Romano, S. P. (2014). Janus: A General Purpose WebRTC Gateway. *Proceedings of the Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm)*, pp. 1–8.

Azure, M. Microsoft Azure Cloud Computing Platform & Services. Retrieved from: https://azure.microsoft.com.

Bach, T. e. a. (2014). Combination of IMS-based IPTV Services with WebRTC. *Proc. Int. Multi-Conf. Comput. Global Inf. Technol., Seville*.

Baktir, A. C., Ozgovde, A. & Ersoy, C. (2017, May). Enabling service-centric networks for cloudlets using SDN. *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 344-352. doi: 10.23919/INM.2017.7987297.

Bari, F. et al. (2016). Orchestrating Virtualized Network Functions. *IEEE Trans. Netw. Service Manage.*, 13(4), 725-739. doi: 10.1109/TNSM.2016.2569020.

Bari, M. F., Chowdhury, S. R., Ahmed, R. & Boutaba, R. (2015, Nov). On orchestrating virtual network functions. *2015 11th International Conference on Network and Service Management (CNSM)*, pp. 50-56. doi: 10.1109/CNSM.2015.7367338.

Beck, M. T. & Botero, J. F. (2015, Dec). Coordinated Allocation of Service Function Chains. *2015 IEEE Global Commun. Conf. (GLOBECOM)*, pp. 1-6. doi: 10.1109/GLOCOM.2015.7417401.

Beloglazov, A. & Buyya, R. (2013). Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers under Quality of Service Constraints. *IEEE Transactions on Parallel and Distributed Systems*, 24(7), 1366-1379. doi: 10.1109/TPDS.2012.240.

Bhamare, D., Jain, R., Samaka, M. & Erbad, A. (2016). A survey on service function chaining. *Journal of Network and Computer Applications*, 75, 138-155. doi: https://doi.org/10.1016/j.jnca.2016.09.001.

Bhamare, D. et al. (2017). Optimal virtual network function placement in multi-cloud service function chaining architecture. *Computer Communications*, 102, 1 - 16. doi: https://doi.org/10.1016/j.comcom.2017.02.011.

Bobroff, N., Kochut, A. & Beaty, K. (2007, May). Dynamic Placement of Virtual Machines for Managing SLA Violations. *2007 10th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 119-128. doi: 10.1109/INM.2007.374776.

Boubendir, A. et al. (2018, June). 5G Edge Resource Federation: Dynamic and Cross-domain Network Slice Deployment. *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pp. 338-340. doi: 10.1109/NETSOFT.2018.8460118.

Calheiros, R. N., Masoumi, E., Ranjan, R. & Buyya, R. (2015). Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications' QoS. *IEEE Transactions on Cloud Computing*, 3(4), 449-458. doi: 10.1109/TCC.2014.2350475.

Camarillo, G. & Garcia-Martin, M.-A. (2007). *The 3G IP multimedia subsystem (IMS): merging the Internet and the cellular worlds*. John Wiley & Sons.

Cao, H., Yang, L. & Zhu, H. (2018). Novel Node-Ranking Approach and Multiple Topology Attributes-Based Embedding Algorithm for Single-Domain Virtual Network Embedding. *IEEE Internet of Things Journal*, 5(1), 108-120. doi: 10.1109/JIOT.2017.2773489.

Cao, Z., Chi, C., Hao, R. & Xiao, Y. (2008, Nov). User Behavior Modeling and Traffic Analysis of IMS Presence Servers. *2008 IEEE Global Telecommunications Conference*, pp. 1-5. doi: 10.1109/GLOCOM.2008.ECP.474.

Carella, G., Corici, M., Crosta, P., Comi, P., Bohnert, T. M., Corici, A. A., Vingarzan, D. & Magedanz, T. (2014, June). Cloudified IP Multimedia Subsystem (IMS) for Network Function Virtualization (NFV)-based architectures. *2014 IEEE Symposium on Computers and Communications (ISCC)*, Workshops, 1-6. doi: 10.1109/ISCC.2014.6912647.

Carpio, F., Dhahri, S. & Jukan, A. (2017). VNF placement with replication for Loac balancing in NFV networks. *2017 IEEE International Conference on Communications (ICC)*, pp. 1-6. doi: 10.1109/ICC.2017.7996515.

Cerrato, I., Palesandro, A., Risso, F., Suñé, M., Vercellone, V. & Woesner, H. (2015). Toward dynamic virtualized network services in telecom operator networks. *Computer Networks*, 92, 380 - 395. Software Defined Networks and Virtualization.

Cerroni, W., Buratti, C., Cerboni, S., Davoli, G., Contoli, C., Foresta, F., Callegati, F. & Verdone, R. (2017, July). Intent-based management and orchestration of heterogeneous open-flow/IoT SDN domains. *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1-9. doi: 10.1109/NETSOFT.2017.8004109.

Chase, J. & Niyato, D. (2017). Joint Optimization of Resource Provisioning in Cloud Computing. *IEEE Transactions on Services Computing*, 10(3), 396-409. doi: 10.1109/TSC.2015.2476812.

Chatras, B., Tsang Kwong, U. S. & Bihannic, N. (2017). NFV enabling network slicing for 5G. *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, pp. 219-225. doi: 10.1109/ICIN.2017.7899415.

Chayapathi, R., Hassan, S. F. & Shah, P. (2016). Network Functions Virtualization (NFV) with a Touch of SDN.

Chen, J., Li, K., Deng, Q., Li, K. & Yu, P. S. (2019). Distributed Deep Learning Model for Intelligent Video Surveillance Systems with Edge Computing. *IEEE Transactions on Industrial Informatics*, 1-1. doi: 10.1109/TII.2019.2909473.

Chen, M., Liew, S. C., Shao, Z. & Kai, C. (2013). Markov Approximation for Combinatorial Network Optimization. *IEEE Trans. Inf. Theory*, 59(10), 6301-6327. doi: 10.1109/TIT.2013.2268923.

Chen, W., Yin, X., Wang, Z., Shi, X. & Yao, J. (2020). Placement and Routing Optimization Problem for Service Function Chain: State of Art and Future Opportunities. *Artificial Intelligence and Security*, pp. 176–188.

Cheng, X., Su, S., Zhang, Z., Wang, H., Yang, F., Luo, Y. & Wang, J. (2011). Virtual Network Embedding Through Topology-aware Node Ranking. *SIGCOMM Comput. Commun. Rev.*, 41(2), 38–47. doi: 10.1145/1971162.1971168.

Cheng, X., Su, S., Zhang, Z., Shuang, K., Yang, F., Luo, Y. & Wang, J. (2012). Virtual Network Embedding Through Topology Awareness and Optimization. *Comput. Netw.*, 56(6), 1797–1813. doi: 10.1016/j.comnet.2012.01.022.

Clearwater. (2014). Project Clearwater - IMS in the Cloud. Retrieved from: http://www.projectclearwater.org/.

Cotroneo, D., Natella, R. & Rosiello, S. (2017). NFV-Throttle: An Overload Control Framework for Network Function Virtualization. *IEEE Trans. Netw. Service Manage.*, 14(4), 949-963. doi: 10.1109/TNSM.2017.2752173.

Cruz, B. S. & Barraca, J. P. (2015, July). IMS centric communication supporting WebRTC endpoints. *2015 IEEE Symp. Comput. Commun. (ISCC)*, pp. 732-737. doi: 10.1109/ISCC.2015.7405601.

Curtis, A. R., Mogul, J. C., Tourrilhes, J., Yalagandula, P., Sharma, P. & Banerjee, S. (2011). DevoFlow: Scaling Flow Management for High-Performance Networks. *Proceedings of the ACM SIGCOMM 2011 Conference*, (SIGCOMM '11), 254–265. doi: 10.1145/2018436.2018466.

Cziva, R., Anagnostopoulos, C. & Pezaros, D. P. (2018, April). Dynamic, Latency-Optimal vNF Placement at the Network Edge. *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pp. 693-701. doi: 10.1109/INFOCOM.2018.8486021.

"DARPA". "Sample Optical Network Topology Files". Retrieved from: "http://www.monarchna.com/topology.html".

Darzentas, J. (1984). Problem Complexity and Method Efficiency in Optimization. *Journal of the Operational Research Society*, 35(5), 455-455. doi: 10.1057/jors.1984.92.

de Brito, M. S., Hoque, S., Magedanz, T., Steinke, R., Willner, A., Nehls, D., Keils, O. & Schreiner, F. (2017, May). A service orchestration architecture for Fog-enabled infrastructures. *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*, pp. 127-132. doi: 10.1109/FMEC.2017.7946419.

Dieye, M., Ahvar, S., Sahoo, J., Ahvar, E., Glitho, R., Elbiaze, H. & Crespi, N. (2018). CPVNF: Cost-Efficient Proactive VNF Placement and Chaining for Value-Added Services in Content Delivery Networks. *IEEE Transactions on Network and Service Management*, 15(2), 774-786. doi: 10.1109/TNSM.2018.2815986.

Dixit, A., Hao, F., Mukherjee, S., Lakshman, T. & Kompella, R. (2013). Towards an Elastic Distributed SDN Controller. *SIGCOMM Comput. Commun. Rev.*, 43(4), 7–12. doi: 10.1145/2534169.2491193.

Doe, J. (1999a). ACRONYM: Class name. Institution.

Doe, J. (1999b). ACRONYM: Class name [Class notes]. Retrieved from: http://www.google.ca.

Doe, J. (1999c). *The title of the thesis*. (Master's thesis, School, Address).

Doe, J. (1999d). *The title of the thesis*. (Master's thesis, School, Address). Retrieved from: Name of the database. (ID number).

Doe, J. (1999e). *The title of the thesis*. (Master's thesis, School, Address). Retrieved from: http://www.google.ca.

Doe, J. (1999f, October, 2). Web page title [Format]. Retrieved from: http://www.google.ca.

Doe, J. (1999g). Dataset title [Online dataset]. Retrieved from: http://www.google.ca.

Doe, J. [Nickname]. (1999h, October, 2). Web page title [Format]. Retrieved from: http://www.google.ca.

Doe, J. [Nickname]. (1999i, October, 2). Video title [Youtube Video]. Retrieved from: http://www.google.ca.

Doe, J. (1999j). *Patent type n°42*. Location: organization.

Doe, J. (1999k). *The title of the thesis*. (Ph.D. thesis, School, Address).

Doe, J. [Unpublished raw data]. (1999l) [Short description of the contents].

Doe, J. (1999m). Software name (Version 1.0) [Software]. Location: Publisher.

Doe, J. (1999n). *The title of the report* (Report n°42). Location: Publisher.

Doe, J. (1999o). *Document title*. Unpublished document, Institution, Location.

Doe, J. (1999p). *Document title*. Manuscript submitted for publication.

Doe, J. (1999q). *Document title*. Retrieved from: http://www.google.ca.

Doe, J., Doe, J. & Doe, J. (1999a). Title of the article. *Name of the journal*, 4(2), 220-242.

Doe, J., Doe, J. & Doe, J. (1999b). Title of the article. *Name of the journal*. In press.

Doe, J., Doe, J. & Doe, J. (1999c). *Title of the book* (ed. 3). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999d). *Title of the book* [Version]. doi: 123-456-789.

Doe, J., Doe, J. & Doe, J. (Eds.). (1999e). *Title of the book* (ed. 3). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999f). *Title of the book* [Version]. Retrieved from: http://www.google.com.

Doe, J., Doe, J. & Doe, J. (1999g). Title of the entry. In *Title of the encyclopedia* (ed. 3, vol. 4, pp. 220-242). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999h). In Doe, J., Doe, J. & Doe, J. (Eds.), *Title of the book* (ed. 3, vol. 4, ch. 7, pp. 220-242). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999i). Title of the chapter. In Doe, J., Doe, J. & Doe, J. (Eds.), *Title of the book* (ed. 3, vol. 4, pp. 220-242). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999j). Title of the article. In Doe, J., Doe, J. & Doe, J. (Eds.), *Title of the conference proceedings* (ed. 3, vol. 4, pp. 220-242). Location: Publisher.

Doe, J., Doe, J. & Doe, J. (1999k, October). *Title of the article*. Communication presented in Title of the conference proceedings, Location (pp. 220-242).

Doe, J., Doe, J. & Doe, J. (1999l). Title of the article. *Title of the conference proceedings*, 4(2), 220-242.

Doe, J., Doe, J. & Doe, J. (1999m, October). Title of the article. *Title of the magasine*, 4(2), 220-242.

Doe, J., Doe, J. & Doe, J. (1999n, October). Title of the article. *Title of the magasine*, pp. 220.

D'Oro, S., Galluccio, L., Palazzo, S. & Schembra, G. (2017). A Game Theoretic Approach for Distributed Resource Allocation and Orchestration of Softwarized Networks. *IEEE Journal on Selected Areas in Communications*, 35(3), 721-735. doi: 10.1109/JSAC.2017.2672278.

Dräxler, S., Karl, H., Kouchaksaraei, H. R., Machwe, A., Dent-Young, C., Katsalis, K. & Samdanis, K. (2018, June). 5G OS: Control and Orchestration of Services on Multi-Domain Heterogeneous 5G Infrastructures. *2018 European Conference on Networks and Communications (EuCNC)*, pp. 1-9. doi: 10.1109/EuCNC.2018.8443210.

Duan, J., Wu, C., Le, F., Liu, A. X. & Peng, Y. (2017). Dynamic Scaling of Virtualized, Distributed Service Chains: A Case Study of IMS. *IEEE Journal on Selected Areas in Communications*, 35(11), 2501-2511. doi: 10.1109/JSAC.2017.2760188.

Dwaraki, A. & Wolf, T. (2016). Adaptive Service-Chain Routing for Virtual Network Functions in Software-Defined Networks. *Proceedings of the 2016 Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, (HotMIddlebox '16), 32–37. doi: 10.1145/2940147.2940148.

El-Hoiydi, A. (2002, April). Aloha with preamble sampling for sporadic traffic in ad hoc wireless sensor networks. *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, 5, 3418-3423 vol.5. doi: 10.1109/ICC.2002.997465.

EnergyHub. Electricity Prices in Canada 2020. Retrieved on 2020-02-07 from: https://energyhub.org/electricity-prices/.

Eramo, V., Tosti, A. & Miucci, E. (2016). Server Resource Dimensioning and Routing of Service Function Chain in NFV Network Architectures. *JECE*, 2016. doi: 10.1155/2016/7139852.

Eramo, V., Miucci, E., Ammar, M. & Lavacca, F. G. (2017). An Approach for Service Function Chain Routing and Virtual Function Network Instance Migration in Network Function Virtualization Architectures. *IEEE/ACM Transactions on Networking*, 25(4), 2008-2025. doi: 10.1109/TNET.2017.2668470.

Erdos, P. & Rényi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1), 17–60.

et al., C. J. B. (2018). *Multi-domain Network Virtualization* (Report n°draft-bernardos-nfvrg-multidomain-04). Internet Engineering Task Force.

et al., M. G. (2015, Oct). Elastic virtual network function placement. *2015 IEEE 4th Int. Conf. Cloud Netw. (CloudNet)*, pp. 255-260. doi: 10.1109/CloudNet.2015.7335318.

et al., Z. Y. (2016). Joint topology design and mapping of service function chains for efficient, scalable, and reliable network functions virtualization. *IEEE Network*, 30(3), 81-87. doi: 10.1109/MNET.2016.7474348.

ETSI ISG on Network Functions Virtualization. (2012, Oct.). Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action.

ETSI ISG on Network Functions Virtualization. (2013, Oct.). Network Functions Virtualisation: Architectural Framework.

ETSI ISG on Network Functions Virtualization. (2016, Apr.). Network Functions Virtualisation (NFV); Reliability; Report on Models and Features for End-to-End Reliability.

ETSI ISG on Network Functions Virtualization. (2018, Jan.). Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Report on architecture options to support multiple administrative domains.

Fang, J. & Ma, A. (2021). IoT Application Modules Placement and Dynamic Task Processing in Edge-Cloud Computing. *IEEE Internet of Things Journal*, 8(16), 12771-12781. doi: 10.1109/JIOT.2020.3007751.

Farkiani, B., Bakhshi, B. & MirHassani, S. A. (2019). A Fast Near-Optimal Approach for Energy-Aware SFC Deployment. *IEEE Transactions on Network and Service Management*, 16(4), 1360-1373. doi: 10.1109/TNSM.2019.2944023.

Farooqi, N., Gutub, A. & Khozium, M. O. (2019). Smart Community Challenges : Enabling IoT / M 2 M Technology Case Study.

Fasolo, E., Rossi, M., Widmer, J. & Zorzi, M. (2007). In-network aggregation techniques for wireless sensor networks: a survey. *IEEE Wireless Communications*, 14(2), 70-87. doi: 10.1109/MWC.2007.358967.

Fichera, S., Gharbaoui, M., Castoldi, P., Martini, B. & Manzalini, A. (2017, July). On experimenting 5G: Testbed set-up for SDN orchestration across network cloud and IoT domains. *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1-6. doi: 10.1109/NETSOFT.2017.8004245.

Filali, A., Kobbane, A., Elmachkour, M. & Cherkaoui, S. (2018, May). SDN Controller Assignment and Load Balancing with Minimum Quota of Processing Capacity. *2018 IEEE International Conference on Communications (ICC)*, pp. 1-6. doi: 10.1109/ICC.2018.8422750.

Gao, L. & Rouskas, G. N. (2020). Congestion Minimization for Service Chain Routing Problems With Path Length Considerations. *IEEE/ACM Transactions on Networking*, 28(6), 2643-2656. doi: 10.1109/TNET.2020.3017792.

Garey, M. R., Graham, R. L. & Johnson, D. S. (1977). The Complexity of Computing Steiner Minimal Trees. *SIAM Journal on Applied Mathematics*, 32(4), 835–859. Retrieved from: http://www.jstor.org/stable/2100193.

Gautam, N. (2012). *Analysis of queues: methods and applications*. CRC Press.

Gharbaoui, M., Contoli, C., Davoli, G., Cuffaro, G., Martini, B., Paganelli, F., Cerroni, W., Cappanera, P. & Castoldi, P. (2018, Nov). Demonstration of Latency-Aware and Self-Adaptive Service Chaining in 5G/SDN/NFV infrastructures. *2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 1-2. doi: 10.1109/NFV-SDN.2018.8725645.

Ghaznavi, M., Shahriar, N., Kamali, S., Ahmed, R. & Boutaba, R. (2017). Distributed Service Function Chaining. *IEEE Journal on Selected Areas in Communications*, 35(11), 2479-2489. doi: 10.1109/JSAC.2017.2760178.

Giertzsch, F., Krüger, L. & Timm-Giel, A. (2018). Analytical Model for Performance Evaluation of Random Wireless Sensor Networks. *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, (MSWIM '18), 235–239. doi: 10.1145/3242102.3242144.

Gochhayat, S. P., Kaliyar, P., Conti, M., Tiwari, P., Prasath, V., Gupta, D. & Khanna, A. (2019). LISA: Lightweight context-aware IoT service architecture. *Journal of Cleaner Production*, 212, 1345 - 1356.

Gong, L., Jiang, H., Wang, Y. & Zhu, Z. (2016). Novel Location-Constrained Virtual Network Embedding LC-VNE Algorithms Towards Integrated Node and Link Mapping. *IEEE/ACM Trans. Netw.*, 24(6), 3648–3661. doi: 10.1109/TNET.2016.2533625.

Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. A., Patel, P. & Sengupta, S. (2009). VL2: A Scalable and Flexible Data Center Network. *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, (SIGCOMM '09), 51–62. doi: 10.1145/1592568.1592576.

GSM Association. (2016). *WebRTC to complement IP Communication Services*. Non-binding White Paper.

Guo, Z., Zhang, S., Feng, W., Wu, W. & Lan, J. (2020). Exploring the role of paths for dynamic switch assignment in software-defined networks. *Future Generation Computer Systems*, 107, 238-246.

Gupta, A. et al. (2018). On service-chaining strategies using Virtual Network Functions in operator networks. *Computer Networks*, 133, 1 - 16. doi: https://doi.org/10.1016/j.comnet.2018.01.028.

Gupta, L., Samaka, M., Jain, R., Erbad, A., Bhamare, D. & Metz, C. (2017, Jan). COLAP: A predictive framework for service function chain placement in a multi-cloud environment. *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1-9. doi: 10.1109/CCWC.2017.7868377.

Habibi, M., Fazli, M. & Movaghar, A. (2019). Efficient distribution of requests in federated cloud computing environments utilizing statistical multiplexing. *Future Generation Computer Systems*, 90, 451 - 460. doi: https://doi.org/10.1016/j.future.2018.08.032.

Halpern, J. & Pignataro, C. (2015). *Service Function Chaining (SFC) Architecture* (Report n°7665). RFC Editor. Retrieved from: InternetRequestsforComments.

Hardes, T., Dressler, F. & Sommer, C. (2017). Simulating a city-scale community network: From models to first improvements for Freifunk. *2017 International Conference on Networked Systems (NetSys)*, pp. 1-7.

Harutyunyan, D., Shahriar, N., Boutaba, R. & Riggio, R. (2019, June). Latency-Aware Service Function Chain Placement in 5G Mobile Networks. *2019 IEEE Conference on Network Softwarization (NetSoft)*, pp. 133-141. doi: 10.1109/NETSOFT.2019.8806646.

Hawilo, H., Jammal, M. & Shami, A. (2019). Network Function Virtualization-Aware Orchestrator for Service Function Chaining Placement in the Cloud. *IEEE Journal on Selected Areas in Communications*, 37(3), 643-655. doi: 10.1109/JSAC.2019.2895226.

Herrera, J. G. & Botero, J. F. (2016). Resource Allocation in NFV: A Comprehensive Survey. *IEEE Trans. Netw. Service Manage.*, 13(3), 518-532. doi: 10.1109/TNSM.2016.2598420.

Hoang, D. B. (2015). Software Defined Networking? Shaping up for the next disruptive step? *Australian Journal of Telecommunications and the Digital Economy*, 3(4).

Holmberg, C., Hakansson, S. & Eriksson, G. (2015). *Web Real-Time Communication Use Cases and Requirements* (Report n°7478). RFC Editor. Retrieved from: InternetRequestsforComments.

Hong, Y., Huang, C. & Yan, J. (2010, April). Analysis of SIP retransmission probability using a Markov-Modulated Poisson Process model. *2010 IEEE Network Operations and Management Symposium - NOMS 2010*, pp. 179-186. doi: 10.1109/NOMS.2010.5488458.

Huang, X., Bian, S., Shao, Z. & Xu, H. (2017, May). Dynamic switch-controller association and control devolution for SDN systems. *2017 IEEE International Conference on Communications (ICC)*, pp. 1-6. doi: 10.1109/ICC.2017.7997427.

Inayatullah, D. S., Aman, M., Rani, A., Zaheer, H. & Siddiqi, T. (2019). A New Technique for Determining Approximate Center of a Polytope. *Advances in Operations Research*, 2019, 1-7. doi: 10.1155/2019/8218329.

Ito, M., Nakauchi, K., Shoji, Y., Nishinaga, N. & Kitatsuji, Y. (2014). Service-Specific Network Virtualization to Reduce Signaling Processing Loads in EPC/IMS. *IEEE Access*, 2, 1076-1084. doi: 10.1109/ACCESS.2014.2359059.

Ivov, E., Rescorla, E., Uberti, J. & Saint-Andre, P. (2017). *Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol* (Report n°draft-ietf-ice-trickle-15). Retrieved from: http://www.ietf.org/internet-drafts/draft-ietf-ice-trickle-15.txt.

Jaeger, M. C., Rojec-Goldmann, G. & Muhl, G. (2004, Sept). QoS aggregation for Web service composition using workflow patterns. *Proceedings. Eighth IEEE International Enterprise Distributed Object Computing Conference, 2004. EDOC 2004.*, pp. 149-159. doi: 10.1109/EDOC.2004.1342512.

Jang, I., Suh, D., Pack, S. & Dán, G. (2017). Joint Optimization of Service Function Placement and Flow Distribution for Service Function Chaining. *IEEE Journal on Selected Areas in Communications*, 35(11), 2532-2541. doi: 10.1109/JSAC.2017.2760162.

Kapsalis, A., Raywad-Smith, V. J. & Smith, G. D. (1993). Solving the Graphical Steiner Tree Problem Using Genetic Algorithms. *Journal of the Operational Research Society*, 44(4), 397-406.

Kar, B., Wu, E. H. K. & Lin, Y. D. (2018). Energy Cost Optimization in Dynamic Placement of Virtualized Network Function Chains. *IEEE Trans. Netw. Service Manage.*, 15(1), 372-386. doi: 10.1109/TNSM.2017.2782370.

Katsalis, K., Nikaein, N. & Edmonds, A. (2016, Dec). Multi-Domain Orchestration for NFV: Challenges and Research Directions. *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*, pp. 189-195. doi: 10.1109/IUCC-CSS.2016.034.

Knight, S., Nguyen, H. X., Falkner, N., Bowden, R. & Roughan, M. (2011). The internet topology zoo. *IEEE Journal on Selected Areas in Communications*, 29(9), 1765–1775.

Kreutz, D., Ramos, F. M. V., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S. & Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, 103(1), 14-76. doi: 10.1109/JPROC.2014.2371999.

Kumar, R., Hasan, M., Padhy, S., Evchenko, K., Piramanayagam, L., Mohan, S. & Bobba, R. B. (2017). End-to-End Network Delay Guarantees for Real-Time Systems Using SDN. *2017 IEEE Real-Time Systems Symposium (RTSS)*, pp. 231-242. doi: 10.1109/RTSS.2017.00029.

Kuo, T., Liou, B., Lin, K. C. & Tsai, M. (2016, April). Deploying chains of virtual network functions: On the relation between link and server usage. *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1-9. doi: 10.1109/INFOCOM.2016.7524565.

Laghrissi, A. & Taleb, T. (2019). A Survey on the Placement of Virtual Resources and Virtual Network Functions. *IEEE Communications Surveys Tutorials*, 21(2), 1409-1434. doi: 10.1109/COMST.2018.2884835.

Laghrissi, A., Taleb, T., Bagaa, M. & Flinck, H. (2017, Dec). Towards Edge Slicing: VNF Placement Algorithms for a Dynamic amp;amp; Realistic Edge Cloud Environment. *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1-6. doi: 10.1109/GLOCOM.2017.8254653.

Li, D., Hong, P., Xue, K. & j. Pei. (2018a). Virtual Network Function Placement Considering Resource Optimization and SFC Requests in Cloud Datacenter. *IEEE Transactions on Parallel and Distributed Systems*, 29(7), 1664-1677. doi: 10.1109/TPDS.2018.2802518.

Li, D., Hong, P., Xue, K. & Pei, J. (2019). Virtual network function placement and resource optimization in NFV and edge computing enabled networks. *Computer Networks*, 152, 12-24. doi: https://doi.org/10.1016/j.comnet.2019.01.036.

Li, G., Zhou, H., Feng, B., Li, G. & Xu, Q. (2018b). Horizontal-based orchestration for multi-domain SFC in SDN/NFV-enabled satellite/terrestrial networks. *China Communications*, 15(5), 77-91. doi: 10.1109/CC.2018.8387988.

Li, X. & Qian, C. (2015, April). The virtual network function placement problem. *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 69-70. doi: 10.1109/INFCOMW.2015.7179347.

Li, X., Lu, R., Liang, X., Shen, X., Chen, J. & Lin, X. (2011). Smart community: an internet of things application. *IEEE Communications Magazine*, 49(11), 68-75.

Li, Y., Phan, L. T. X. & Loo, B. T. (2016, April). Network functions virtualization with soft real-time guarantees. *IEEE INFOCOM 2016 - The 35th Annual IEEE Int. Conf. Comput. Commun.*, pp. 1-9. doi: 10.1109/INFOCOM.2016.7524563.

Liao, L., Leung, V. C. M. & Chen, M. (2015). Virtualizing IMS Core and Its Performance Analysis. In Leung, V. C., Lai, R. X., Chen, M. & Wan, J. (Eds.), *Cloud Computing: 5th International Conference, CloudComp 2014, Guilin, China, October 19-21, 2014, Revised Selected Papers* (pp. 53–65). Cham: Springer International Publishing. doi: 10.1007/978-3-319-16050-4_5.

Lin, T., Zhou, Z., Tornatore, M. & Mukherjee, B. (2016). Demand-Aware Network Function Placement. *Journal of Lightwave Technology*, 34(11), 2590-2600. doi: 10.1109/JLT.2016.2535401.

Liu, J., Zhang, Y., Zhou, Y., Zhang, D. & Liu, H. (2015). Aggressive Resource Provisioning for Ensuring QoS in Virtualized Environments. *IEEE Transactions on Cloud Computing*, 3(2), 119-131. doi: 10.1109/TCC.2014.2353045.

Liu, J., Li, Y., Zhang, Y., Su, L. & Jin, D. (2017). Improve Service Chaining Performance with Optimized Middlebox Placement. *IEEE Transactions on Services Computing*, 10(4), 560-573. doi: 10.1109/TSC.2015.2502252.

Liu, L., Guo, S., Liu, G. & Yang, Y. (2021). Joint Dynamical VNF Placement and SFC Routing in NFV-Enabled SDNs. *IEEE Transactions on Network and Service Management*, 1-1. doi: 10.1109/TNSM.2021.3091424.

Liu, Q. & Han, T. (2019). VirtualEdge: Multi-Domain Resource Orchestration and Virtualization in Cellular Edge Computing. *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1051-1060. doi: 10.1109/ICDCS.2019.00108.

Liu, Y., Pei, J., Hong, P. & Li, D. (2019). Cost-Efficient Virtual Network Function Placement and Traffic Steering. *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1-6. doi: 10.1109/ICC.2019.8762060.

Lu, F., Pan, H., Lei, X., Liao, X. & Jin, H. (2013, Dec). A Virtualization-Based Cloud Infrastructure for IMS Core Network. *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, 1, 25-32. doi: 10.1109/CloudCom.2013.10.

Luizelli, M. C., Bays, L. R., Buriol, L. S., Barcellos, M. P. & Gaspary, L. P. (2015, May). Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions. *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 98-106. doi: 10.1109/INM.2015.7140281.

Mahmud, R., Kotagiri, R. & Buyya, R. (2018). Fog Computing: A Taxonomy, Survey and Future Directions. In Di Martino, B., Li, K.-C., Yang, L. T. & Esposito, A. (Eds.), *Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives* (pp. 103–130). Singapore: Springer Singapore.

Malas, D. & Morton, A. (2011). *Basic Telephony SIP End-to-End Performance Metrics* (Report n°6076). RFC Editor. Retrieved from: InternetRequestsforComments.

Manzalini, A., Lopezz, D. R., Lonsethagen, H., Suciu, L., Bifulcozz, R., Odinixi, M., Celozzixiii, G., Martinixv, B., Rissoy, F., Garayx, J., Foteinosk, V., Demestichasyy, P., Carullox, G., Tambascox, M. & Carrozzoxiv, G. (2017, July). A unifying operating platform for 5G end-to-end and multi-layer orchestration. *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1-5. doi: 10.1109/NETSOFT.2017.8004216.

Mao, M. & Humphrey, M. (2012, June). A Performance Study on the VM Startup Time in the Cloud. *2012 IEEE Fifth International Conference on Cloud Computing*, pp. 423-430. doi: 10.1109/CLOUD.2012.103.

Mauro, M. D. & Longo, M. (2015, July). Revealing encrypted WebRTC traffic via machine learning tools. *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)*, 04, 259-266.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. & Turner, J. (2008). OpenFlow: Enabling Innovation in Campus Networks. *SIGCOMM Comput. Commun. Rev.*, 38(2), 69–74. doi: 10.1145/1355734.1355746.

Mechtri, M., Ghribi, C. & Zeghlache, D. (2016a). A Scalable Algorithm for the Placement of Service Function Chains. *IEEE Trans. Netw. Service Manage.*, 13(3), 533-546. doi: 10.1109/TNSM.2016.2598068.

Mechtri, M., Ghribi, C. & Zeghlache, D. (2016b, June). VNF Placement and Chaining in Distributed Cloud. *IEEE 9th International Conference on Cloud Computing (CLOUD)*. doi: 10.1109/CLOUD.2016.0057.

Mehraghdam, S., Keller, M. & Karl, H. (2014). Specifying and placing chains of virtual network functions. *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pp. 7-13. doi: 10.1109/CloudNet.2014.6968961.

Miorandi, D., Sicari, S., Pellegrini, F. D. & Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7), 1497 - 1516. doi: https://doi.org/10.1016/j.adhoc.2012.02.016.

Mireslami, S., Rakai, L., Wang, M. & Far, B. H. (2015, Dec). Minimizing Deployment Cost of Cloud-Based Web Application with Guaranteed QoS. *2015 IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6. doi: 10.1109/GLOCOM.2015.7417230.

Mireslami, S., Rakai, L., Far, B. H. & Wang, M. (2017). Simultaneous Cost and QoS Optimization for Cloud Resource Allocation. *IEEE Trans. Netw. Service Manage.*, 14(3), 676-689. doi: 10.1109/TNSM.2017.2738026.

Moens, H. & Turck, F. D. (2014, Nov). VNF-P: A model for efficient placement of virtualized network functions. *10th Int. Conf. Netw. Service Manage. (CNSM) and Workshop*, pp. 418-423. doi: 10.1109/CNSM.2014.7014205.

Mogul, J. C. & Congdon, P. (2012). Hey, You Darned Counters! Get off My ASIC! *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, (HotSDN '12), 25–30. doi: 10.1145/2342441.2342447.

Montazerolghaem, A., Yaghmaee, M. H., Leon-Garcia, A., Naghibzadeh, M. & Tashtarian, F. (2016). A Load-Balanced Call Admission Controller for IMS Cloud Computing. *IEEE Transactions on Network and Service Management*, 13(4), 806-822. doi: 10.1109/TNSM.2016.2572161.

Morabito, R., Farris, I., Iera, A. & Taleb, T. (2017). Evaluating performance of containerized IoT services for clustered devices at the network edge. *IEEE Internet of Things Journal*, 4(4), 1019–1030.

Moretti, A. C. (2003). A weighted projection centering method. *Computational & Applied Mathematics*, 22, 19 - 36.

Mouradian, C., Kianpisheh, S., Abu-Lebdeh, M., Ebrahimnezhad, F., Jahromi, N. T. & Glitho, R. H. (2019). Application Component Placement in NFV-Based Hybrid Cloud/Fog Systems With Mobile Fog Nodes. *IEEE Journal on Selected Areas in Communications*, 37(5), 1130-1143. doi: 10.1109/JSAC.2019.2906790.

Munir, A. & Gordon-Ross, A. (2010). SIP-Based IMS Signaling Analysis for WiMax-3G Interworking Architectures. *IEEE Transactions on Mobile Computing*, 9(5), 733-750. doi: 10.1109/TMC.2010.16.

Neely, M. (2010). *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool.

Nguyen, D. T., Nguyen, K. K., Khazri, S. & Cheriet, M. (2016, Sept). Real-time optimized NFV architecture for internetworking WebRTC and IMS. *2016 17th Int. Telecommun. Netw. Strategy Planning Symp. (Networks)*, pp. 81-88. doi: 10.1109/NETWKS.2016.7751157.

Nguyen, D. T., Nguyen, K. K. & Cheriet, M. (2017, Oct). Optimized IoT service orchestration. *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1-6. doi: 10.1109/PIMRC.2017.8292756.

Nguyen, D. T., Nguyen, K. K. & Cheriet, M. (2018). NFV-based Architecture for the Interworking between WebRTC and IMS. *IEEE Transactions on Network and Service Management*, 1-1. doi: 10.1109/TNSM.2018.2876697.

Nguyen, D. T., Pham, C., Nguyen, K. K. & Cheriet, M. (2019a). Placement and Chaining for Run-time IoT Service Deployment in Edge-Cloud. *IEEE Transactions on Network and Service Management*, 1-1. doi: 10.1109/TNSM.2019.2948137.

Nguyen, D. T., Pham, C., Nguyen, K. K. & Cheriet, M. (2019b, Oct). SACO: A Service Chain Aware SDN Controller-Switch Mapping Framework. *2019 15th International Conference on Network and Service Management (CNSM)*, pp. 1-8.

Njah, Y., Pham, C. & Cheriet, M. (2020). Service and Resource Aware Flow Management Scheme for an SDN-Based Smart Digital Campus Environment. *IEEE Access*, 8, 119635-119653.

Nonde, L., Elgorashi, T. E. H. & Elmirgahni, J. M. H. (2016, Dec). Virtual Network Embedding Employing Renewable Energy Sources. *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6. doi: 10.1109/GLOCOM.2016.7842376.

Ojo, M., Adami, D. & Giordano, S. (2016, Dec). A SDN-IoT Architecture with NFV Implementation. *2016 IEEE Globecom Workshops (GC Wkshps)*, pp. 1-6. doi: 10.1109/GLOCOMW.2016.7848825.

Organization name. (1999). *Norm title*. Institution and norm number. Location: Organization name.

Paganelli, F., Ulema, M. & Martini, B. (2014). Context-aware service composition and delivery in NGSONs over SDN. *IEEE Communications Magazine*, 52(8), 97-105. doi: 10.1109/MCOM.2014.6871676.

Pan, J. & McElhannon, J. (2018). Future Edge Cloud and Edge Computing for Internet of Things Applications. *IEEE Internet of Things Journal*, 5(1), 439-449. doi: 10.1109/JIOT.2017.2767608.

Pham, T. & Chu, H. (2019). Multi-Provider and Multi-Domain Resource Orchestration in Network Functions Virtualization. *IEEE Access*, 7, 86920-86931. doi: 10.1109/ACCESS.2019.2926136.

Poularakis, K., Llorca, J., Tulino, A. M., Taylor, I. & Tassiulas, L. (2020). Service Placement and Request Routing in MEC Networks With Storage, Computation, and Communication Constraints. *IEEE/ACM Transactions on Networking*, 28(3), 1047-1060. doi: 10.1109/TNET.2020.2980175.

Qi, D., Shen, S. & Wang, G. (2019). Towards an efficient VNF placement in network function virtualization. *Computer Communications*, 138, 81-89. doi: https://doi.org/10.1016/j.comcom.2019.03.005.

Qin, Z., Denker, G., Giannelli, C., Bellavista, P. & Venkatasubramanian, N. (2014). A Software Defined Networking architecture for the Internet-of-Things. *2014 IEEE Network Operations and Management Symposium (NOMS)*, pp. 1-9.

Rai, V. & Mahapatra, R. N. (2005, March). Lifetime modeling of a sensor network. *Design, Automation and Test in Europe*, pp. 202-203 Vol. 1. doi: 10.1109/DATE.2005.196.

Ran, Y., Yang, J., Zhang, S. & Xi, H. (2017). Dynamic IaaS Computing Resource Provisioning Strategy with QoS Constraint. *IEEE Transactions on Services Computing*, 10(2), 190-202. doi: 10.1109/TSC.2015.2464212.

Rankothge, W., Le, F., Russo, A. & Lobo, J. (2017). Optimizing Resource Allocation for Virtualized Network Functions in a Cloud Center Using Genetic Algorithms. *IEEE Trans. Netw. Service Manage.*, 14(2), 343-356. doi: 10.1109/TNSM.2017.2686979.

Riera, J. F. et al. (2016, June). TeNOR: Steps towards an orchestration platform for multi-PoP NFV deployment. *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pp. 243-250. doi: 10.1109/NETSOFT.2016.7502419.

Ros, F. J. & Ruiz, P. M. (2014). Five Nines of Southbound Reliability in Software-Defined Networks. *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, (HotSDN '14), 31–36. doi: 10.1145/2620728.2620752.

Rosa, R. V., Santos, M. A. S. & Rothenberg, C. E. (2015, March). MD2-NFV: The case for multi-domain distributed network functions virtualization. *2015 International Conference and Workshops on Networked Systems (NetSys)*, pp. 1-5. doi: 10.1109/NetSys.2015.7089059.

Roy, P., Tahsin, A., Sarker, S., Adhikary, T., Razzaque, M. A. & Hassan, M. M. (2020). User mobility and Quality-of-Experience aware placement of Virtual Network Functions in 5G. *Computer Communications*, 150, 367 - 377.

Rutten, E., Marchand, N. & Simon, D. (2017). Feedback Control as MAPE-K Loop in Autonomic Computing. *Software Engineering for Self-Adaptive Systems III. Assurances*, pp. 349–373.

Sahin, S., Narayanaswamy, K. & Campos-Nanez, E. (2008, Oct). Call setup performance improvements over mobile satellite systems using IP multimedia system. *2008 IEEE International Workshop on Satellite and Space Communications*, pp. 321-325. doi: 10.1109/IWSSC.2008.4656823.

Sallam, G., Gupta, G. R., Li, B. & Ji, B. (2018). Shortest Path and Maximum Flow Problems Under Service Function Chaining Constraints. *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pp. 2132-2140. doi: 10.1109/INFOCOM.2018.8485996.

Salman, O., Elhajj, I., Kayssi, A. & Chehab, A. (2015, Dec). Edge computing enabling the Internet of Things. *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pp. 603-608. doi: 10.1109/WF-IoT.2015.7389122.

Santos, J., Wauters, T., Volckaert, B. & De Turck, F. (2020). Towards delay-aware container-based Service Function Chaining in Fog Computing. *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1-9. doi: 10.1109/NOMS47738.2020.9110376.

Sen, A., Choudhuri, S. & Basu, K. (2020). Structural Dependency Aware Service Chain Mapping for Network Function Virtualization. *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020*, pp. 1-6. doi: 10.1109/DRCN48652.2020.1570611214.

Sengupta, S., Garcia, J. & Masip-Bruin, X. (2019). Taxonomy and Resource Modeling in Combined Fog-to-Cloud Systems. *Proceedings of the Future Technologies Conference (FTC) 2018*, pp. 687–704.

Seth, S. & Singh, N. (2017). Dynamic Threshold-Based Dynamic Resource Allocation Using Multiple VM Migration for Cloud Computing Systems. In Kaushik, S., Gupta, D., Kharb, L. & Chahal, D. (Eds.), *Information, Communication and Computing Technology: Second International Conference, ICICCT 2017, New Delhi, India, May 13, 2017, Revised Selected Papers* (pp. 106–116). Singapore: Springer Singapore. doi: 10.1007/978-981-10-6544-6_11.

Sheoran, A., Sharma, P., Fahmy, S. & Saxena, V. (2017). Contain-ed: An NFV Micro-Service System for Containing e2e Latency. *Proceedings of the Workshop on Hot Topics in Container Networking and Networked Systems*, (HotConNet '17).

Shi, W., Zhang, L., Wu, C., Li, Z. & Lau, F. C. M. (2016). An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing. *IEEE/ACM Transactions on Networking*, 24(4), 2060-2073. doi: 10.1109/TNET.2015.2444657.

Simarro, J. L. L., Moreno-Vozmediano, R., Montero, R. S. & Llorente, I. M. (2011, July). Dynamic placement of virtual machines for cost optimization in multi-cloud environments. *2011 International Conference on High Performance Computing Simulation*, pp. 1-7. doi: 10.1109/HPCSim.2011.5999800.

Song, S., Lee, C., Cho, H., Lim, G. & Chung, J. (2019). Clustered Virtualized Network Functions Resource Allocation based on Context-Aware Grouping in 5G Edge Networks. *IEEE Transactions on Mobile Computing*, 1-1. doi: 10.1109/TMC.2019.2907593.

Sousa, N. F. S., Perez, D. A. L., Rosa, R. V., Santos, M. A. & Rothenberg, C. E. (2019). Network Service Orchestration: A survey. *Computer Communications*, 142-143, 69 - 94. doi: https://doi.org/10.1016/j.comcom.2019.04.008.

Sridharan, V., Gurusamy, M. & Truong-Huu, T. (2017). On Multiple Controller Mapping in Software Defined Networks With Resilience Constraints. *IEEE Communications Letters*, 21(8), 1763-1766. doi: 10.1109/LCOMM.2017.2696006.

Sun, G., Li, Y., Liao, D. & Chang, V. (2018). Service Function Chain Orchestration Across Multiple Domains: A Full Mesh Aggregation Approach. *IEEE Transactions on Network and Service Management*, 15(3), 1175-1191. doi: 10.1109/TNSM.2018.2861717.

Sun, G. et al. (2020). Low-Latency and Resource-Efficient Service Function Chaining Orchestration in Network Function Virtualization. *IEEE Internet of Things Journal*, 7(7), 5760-5772. doi: 10.1109/JIOT.2019.2937110.

Sun, G., Li, Y., Yu, H., Vasilakos, A. V., Du, X. & Guizani, M. (2019). Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. *Future Generation Computer Systems*, 91, 347-360. doi: https://doi.org/10.1016/j.future.2018.09.037.

Tai-hoon, K., Carlos, R. & Sabah, M. (2017). Smart City and IoT. *Future Generation Computer Systems*, 76, 159 - 162.

Tajiki, M. M., Salsano, S., Chiaraviglio, L., Shojafar, M. & Akbari, B. (2019). Joint Energy Efficient and QoS-Aware Path Allocation and VNF Placement for Service Function Chaining. *IEEE Transactions on Network and Service Management*, 16(1), 374-388. doi: 10.1109/TNSM.2018.2873225.

Taleb, T., Bagaa, M. & Ksentini, A. (2015, June). User mobility-aware Virtual Network Function placement for Virtual 5G Network Infrastructure. *2015 IEEE International Conference on Communications (ICC)*, pp. 3879-3884. doi: 10.1109/ICC.2015.7248929.

Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta, S. & Sabella, D. (2017). On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Communications Surveys Tutorials*, 19(3), 1657-1681. doi: 10.1109/COMST.2017.2705720.

Tang, H., Zhou, D. & Chen, D. (2019). Dynamic Network Function Instance Scaling Based on Traffic Forecasting and VNF Placement in Operator Data Centers. *IEEE Transactions on Parallel and Distributed Systems*, 30(3), 530-543. doi: 10.1109/TPDS.2018.2867587.

Uberti, J., Jennings, C. & Rescorla, E. (2017). *JavaScript Session Establishment Protocol* (Report n°draft-ietf-rtcweb-jsep-24). Internet Engineering Task Force.

Uniyal, N. et al. (2020). 5GUK Exchange: Towards sustainable end-to-end multi-domain orchestration of softwarized 5G networks. *Computer Networks*, 178, 107297. doi: https://doi.org/10.1016/j.comnet.2020.107297.

Urbieta, A., González-Beltrán, A., Mokhtar, S. B., Hossain, M. A. & Capra, L. (2017). Adaptive and context-aware service composition for IoT-based smart cities. *Future Generation Computer Systems*, 76, 262 - 274.

Van Bemten, A., Guck, J. W., Vizarreta, P., Machuca, C. M. & Kellerer, W. (2018). LARAC-SN and Mole in the Hole: Enabling Routing through Service Function Chains. *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pp. 298-302. doi: 10.1109/NETSOFT.2018.8460128.

Varasteh, A., Madiwalar, B., Van Bemten, A., Kellerer, W. & Mas-Machuca, C. (2021). Holu: Power-Aware and Delay-Constrained VNF Placement and Chaining. *IEEE Transactions on Network and Service Management*, 1-1. doi: 10.1109/TNSM.2021.3055693.

Vigneri, L., Paschos, G. & Mertikopoulos, P. (2019, April). Large-Scale Network Utility Maximization: Countering Exponential Growth with Exponentiated Gradients. *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 1630-1638. doi: 10.1109/INFOCOM.2019.8737600.

Voellmy, A., Wang, J., Yang, Y. R., Ford, B. & Hudak, P. (2013a). Maple: Simplifying SDN Programming Using Algorithmic Policies. *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, (SIGCOMM '13), 87–98. doi: 10.1145/2486001.2486030.

Voellmy, A., Wang, J., Yang, Y. R., Ford, B. & Hudak, P. (2013b). Maple: Simplifying SDN Programming Using Algorithmic Policies. *SIGCOMM Comput. Commun. Rev.*, 43(4), 87–98. doi: 10.1145/2534169.2486030.

Wamser, F., Lombardo, C., Vassilakis, C., Dinh-Xuan, L., Lago, P., Bruschi, R. & Tran-Gia, P. (2018, June). Orchestration and Monitoring in Fog Computing for Personal Edge Cloud Service Support. *2018 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pp. 91-96. doi: 10.1109/LANMAN.2018.8475113.

Wang, J., Pan, J. & Esposito, F. (2017a). Elastic Urban Video Surveillance System Using Edge Computing. *Proceedings of the Workshop on Smart Internet of Things*, (SmartIoT '17), 7:1–7:6. doi: 10.1145/3132479.3132490.

Wang, L., Lu, Z., Wen, X., Knopp, R. & Gupta, R. (2016). Joint Optimization of Service Function Chaining and Resource Allocation in Network Function Virtualization. *IEEE Access*, 4, 8084-8094. doi: 10.1109/ACCESS.2016.2629278.

Wang, P., Lan, J., Zhang, X., Hu, Y. & Chen, S. (2015). Dynamic function composition for network service chain: Model and optimization. *Computer Networks*, 92, 408 - 418. doi: https://doi.org/10.1016/j.comnet.2015.07.020.

Wang, S., Urgaonkar, R., He, T., Chan, K., Zafer, M. & Leung, K. K. (2017b). Dynamic Service Placement for Mobile Micro-Clouds with Predicted Future Costs. *IEEE Transactions on Parallel and Distributed Systems*, 28(4), 1002-1016. doi: 10.1109/TPDS.2016.2604814.

Wang, T., Liu, F. & Xu, H. (2017). An Efficient Online Algorithm for Dynamic SDN Controller Assignment in Data Center Networks. *IEEE/ACM Transactions on Networking*, 25(5), 2788-2801. doi: 10.1109/TNET.2017.2711641.

Wu, L., Garg, S. K. & Buyya, R. (2012). SLA-based admission control for a Software-as-a-Service provider in Cloud computing environments. *Journal of Computer and System Sciences*, 78(5), 1280 - 1299.

Xiang, Q. et al. (2019). Unicorn: Unified resource orchestration for multi-domain, geo-distributed data analytics. *Future Generation Computer Systems*, 93, 188-197. doi: https://doi.org/10.1016/j.future.2018.09.048.

Xiao, W., Bao, W., Zhu, X., Wang, C., Chen, L. & Yang, L. T. (2016). Dynamic Request Redirection and Resource Provisioning for Cloud-Based Video Services under Heterogeneous Environment. *IEEE Transactions on Parallel and Distributed Systems*, 27(7), 1954-1967. doi: 10.1109/TPDS.2015.2470676.

Xu, Y., Cello, M., Wang, I., Walid, A., Wilfong, G., Wen, C. H. ., Marchese, M. & Chao, H. J. (2019). Dynamic Switch Migration in Distributed Software-Defined Networks to Achieve Controller Load Balance. *IEEE Journal on Selected Areas in Communications*, 37(3), 515-529. doi: 10.1109/JSAC.2019.2894237.

Xu, Z., Liang, W., Galis, A., Ma, Y., Xia, Q. & Xu, W. (2018). Throughput optimization for admitting NFV-enabled requests in cloud networks. *Computer Networks*, 143, 15-29. doi: https://doi.org/10.1016/j.comnet.2018.06.015.

Xu, Z., Zhang, Z., Liang, W., Xia, Q., Rana, O. & Wu, G. (2020). QoS-Aware VNF Placement and Service Chaining for IoT Applications in Multi-Tier Mobile Edge Networks. *ACM Trans. Sen. Netw.*, 16(3). doi: 10.1145/3387705.

Yang, S., Li, F., Trajanovski, S. & Fu, X. (2020). Traffic routing in stochastic network function virtualization networks. *Journal of Network and Computer Applications*, 169, 102765. doi: https://doi.org/10.1016/j.jnca.2020.102765.

Ye, Q., Zhuang, W., Li, X. & Rao, J. (2019). End-to-End Delay Modeling for Embedded VNF Chains in 5G Core Networks. *IEEE Internet of Things Journal*, 6(1), 692-704. doi: 10.1109/JIOT.2018.2853708.

Ye, X., Cheng, G. & Luo, X. (2017). Maximizing SDN control resource utilization via switch migration. *Computer Networks*, 126, 69 - 80.

Yu, H., Qiao, C., Anand, V., Liu, X., Di, H. & Sun, G. (2010, Dec). Survivable Virtual Infrastructure Mapping in a Federated Computing and Networking System under Single Regional Failures. *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pp. 1-6. doi: 10.1109/GLOCOM.2010.5683951.

Zhang, Q., Wang, X., Kim, I., Palacharla, P. & Ikeuchi, T. (2016, June). Vertex-centric computation of service function chains in multi-domain networks. *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pp. 211-218. doi: 10.1109/NETSOFT.2016.7502415.

Zheng, D., Peng, C., Liao, X. & Cao, X. (2020). Toward Optimal Hybrid Service Function Chain Embedding in Multiaccess Edge Computing. *IEEE Internet of Things Journal*, 7(7), 6035-6045. doi: 10.1109/JIOT.2019.2957961.

Zheng, G., Tsiopoulos, A. & Friderikos, V. (2018). Optimal VNF Chains Management for Proactive Caching. *IEEE Transactions on Wireless Communications*, 17(10), 6735-6748. doi: 10.1109/TWC.2018.2863685.

Zhou, A., Wang, S., Cheng, B., Zheng, Z., Yang, F., Chang, R. N., Lyu, M. R. & Buyya, R. (2017). Cloud Service Reliability Enhancement via Virtual Machine Placement Optimization. *IEEE Transactions on Services Computing*, 10(6), 902-913. doi: 10.1109/TSC.2016.2519898.