Reliable embedding of virtual networks in Cloud's backbone network

Oussama Soualah

UNIVERSITÉ PARIS-EST

ÉCOLE DOCTORALE MSTIC

Thèse de doctorat

Spécialité : Informatique et Réseaux

M. OUSSAMA SOUALAH

Titre :

Instanciation avec fiabilité des réseaux virtuels dans le réseau cœur du Cloud

Rapporteurs :

M. Bernard COUSIN, Professeur à l'Université de Rennes 1

M. Marcelo DIAS DE AMORIM, Directeur de recherche au CNRS - LIP6/UPMC

Examinateurs :

M. Djamal ZEGHLACHE, Professeur à Telecom Sud Paris, France

M. Cédric ADJIH, Chargé de recherche à l'Inria Saclay, France

M. Dominique VERCHERE, Ingénieur de recherche à Alcatel-Lucent Bell Labs, France

M. Jeremie LEGUAY, Ingénieur de recherche à Huawei, France

Thèse dirigée par :

M. Nadjib AITSAADI, Encadrant, Maître de conférences à l'Université de Paris-Est Créteil

M. Abdelhamid MELLOUK, Directeur de thèse, Professeur à l'Université de Paris-Est Créteil

Thèse soutenue le 15-06-2015

UNIVERSITÉ PARIS-EST

ÉCOLE DOCTORALE MSTIC

DOCTOR OF SCIENCE

Specialization : COMPUTER SCIENCE

Mr. OUSSAMA SOUALAH

Topic :

Reliable Embedding of Virtual Networks in Cloud's Backbone Network

Reviewers :

Mr. Bernard COUSIN, Professor at University of Rennes 1

Mr. Marcelo DIAS DE AMORIM, Research Director at CNRS - LIP6/UPMC

Examiners :

Mr. Djamal ZEGHLACHE, Professor at Telecom Sud Paris, France

Mr. Cédric ADJIH, Scientific Researcher at Inria Saclay, France

Mr. Dominique VERCHERE, Research Engineer at Alcatel-Lucent Bell Labs, France

Mr. Jeremie LEGUAY, Research Engineer at Huawei, France

Thesis supervised by :

Mr. Nadjib AITSAADI, Advisor, Associate Professor at University of Paris-Est Creteil

Mr. Abdelhamid MELLOUK, Supervisor, Professor at University of Paris-Est Creteil

Defense: 15-06-2015

Dedicaces

To my lovely parents
To my wife
To my brother, my sister and my nephew
To all my family
I dedicate this work.

Acknowledgements

Above all, I owe it all to Almighty GOD who made all things possible and give me the light to complete this task.
At the end of my PhD thesis, I would like to thank all those who made this thesis possible.
I would like to express my sincere gratitude to Dr. Nadjib Ait Saadi and Pr. Abdelhamid Mellouk, my research supervisors, for their supports, patient guidances and encouragements. I will always be grateful.
My cordial thanks to the reviewers of my thesis, Pr. Bernard Cousin and Dr. Marcelo Dias De Amorim. Thank you for their important feedbacks to improve my research.
I am grateful for Pr. Djamal Zeghlache, Dr. Cedric Adjih, Dr. Dominique Verchere and Dr. Jeremie Leguay for accepting to examine my thesis and attend my dissertation defence.
I would like to thank all my friends in Tunisia and France, and all my colleagues in the LiSSi laboratory.

Résumé

Le Cloud computing a connu un succès impressionnant au cours des dernières années. Cette expansion en permanence se manifeste clairement, non seulement par l'omniprésence du Cloud dans les environnements académiques et industriels, mais aussi par la hausse impréssionnante du chiffre d'affaire des services Cloud. Ce succès important réalisé, en peu de temps, est obtenu grâce à l'élasticité du Cloud. Afin de bien tirer profit de cette nouvelle architecture logicielle et matérielle, les fournisseurs de service Cloud requièrent des stratégies adéquates pour une gestion efficace des équipements physiques continuellement alloués à plusieurs clients simultanément. En effet, le fournisseur Cloud doit respecter l'accord de niveau de service (i.e., Service Level Agreement) et assurer la continuité de service afin de minimiser le coût des pénalités.
Dans cette thèse, nous abordons la problèmatique de l'instanciation (i.e., *mapping*) des réseaux virtuels au sein du réseau coeur du Cloud tout en considérant la fiabilité des équipements physiques (i.e., routeurs et liens). Notre objectif principal est de maximiser le chiffre d'affaires du fournisseur Cloud par le biais de i) la maximisation du taux des réseaux virtuels acceptés dans le réseau coeur du Cloud et ii) la minimisation des pénalités induites par les interruptions de service en raison de pannes des équipements du réseau. Il a été démontré que ce type de problème est NP-dur, aucune solution optimale ne peut donc être calculée à large échelle. Dans ce cadre, nous avons proposé un algorithme, nommé PR-VNE, favorisant l'utilisation des équipements les plus fiables, lors de l'instanciation des réseaux virtuels, afin de minimiser l'impact néfaste des pannes. Cette proposition exploite les avantages de la métaheuristique de colonie d'abeilles pour assurer une qualité optimisée de mapping de la solution en termes d'optimalité. Il est à souligner que PR-VNE ne réserve aucune ressource de secours dans le cas où une panne se présente. Afin de remédier aux pannes imprévisibles, nous avons défini un algorithme qui adopte non seulement la stratégie préventive mais aussi un mécanisme curatif. Cet algorithme, nommé CG-VNE, est conçu sur la base d'une modélisation basée sur la théorie des jeux. CG-VNE reconfigure (re-instancie) les ressources virtuelles impactées par les pannes dans d'autres équipements réseau sans pour autant réserver des ressources de secours. Finalement, nous avons considéré la vision macroscopique en définissant un algorithme qui traite le problème d'allocation par lot de requêtes tout en considérant la fiabilité. Cet algorithme, nommé BR-VNE, se base principalement sur la stratégie *Monte-Carlo Tree Search* pour trouver la meilleure séquence de mapping. L'évaluation des performances de nos propositions prouve leur efficacité par rapport aux méthodes de l'état de l'art en termes de : i) taux de réseaux virtuels acceptés, ii) taux de requêtes impactées par les pannes et iii) revenu du fournisseur de service Cloud.

Mots-clés :

Informatique en nuage, Virtualisation des réseaux, Fiabilité, Instanciation de réseaux virtuels, Optimisation multi-objective, Algorithme d'optimisation par colonie d'abeille, Théorie des jeux, exploration d'arbre Monte-Carlo.

Abstract

Cloud computing has known an impressive success over the last few years. In fact, it is continuously emerging personal and profesional life thanks to its elasticity, pricing model, etc. This innovative paradigm has attracted both industrial and research communities and becomes omnipresent. In order to take benefit from the Cloud expansion, providers require effecient management strategies to properly supply their physical capabilities such as network resources. Besides, Cloud providers have to respect the Service Level Agreement and avoid any outage impact in order to guarantee the Cloud-based service continuity.
In this thesis we tackle the problem of reliable virtual network embedding within the Cloud backbone by considering the impact of physical equipments' outages. Our main focus is to improve the provider's turnover by i) maximizing the acceptance rate of the incoming virtual networks issued from clients' request and ii) minimizing the penalties induced by service disruption due to the physical failures. This optimization is multi-objective and non-linear, which has been proved that is NP-hard problem. To cope with this complexity and since reaching the optimal solution is computationally intractable in large scale networks, we propose different strategies that aim to respect the aformentionned objectives. First, we propose a preventive approach named `PR-VNE` that urges the use of reliable network resources in order to avoid the physical failures impact. `PR-VNE` strongly relies on the Artificial Bee Colony metaheuristic and classification algorithm. It should be highlighted that `PR-VNE` does not adopt any recovering mechanism to deal with the network outages. Second, we propound a new reactive approach named `CG-VNE` that does not consider any backup resources but re-embeds the impacted virtual resources once an outage occurs in the underlying network. As well as this reactive mechanism, `CG-VNE` adopts the same preventive strategy like `PR-VNE` by avoiding the unreliable resources during the mapping process. It should be noted that `CG-VNE` is based on Game Theory framework by defining a collaborative mapping game. Finally, we deal with the survivable batch mapping problem that considers the embedding of a virtual networks set instead of one client request. We introduce a new reliable batch embedding strategy named `BR-VNE` that relies on Monte-Carlo Tree Search algorithm. `BR-VNE` delegates the embedding of one virtual network request to any online algorithm and focuses to find the best mapping sequence order. The performance evaluation of our three proposals leads to efficient results.

Key Words:

Cloud computing, Network virtualization, Reliability, Virtual network embedding, Multi-objective optimization, Artificial bee colony metaheuristic, Game Theory, Batch mapping, Monte-Carlo Tree Search.

Table of contents

1 Introduction **17**
- 1.1 Cloud computing definition and architecture . . . 18
  - 1.1.1 Definition of Cloud computing . . . 19
  - 1.1.2 Cloud architecture . . . 19
  - 1.1.3 Software stack . . . 22
  - 1.1.4 Cloud computing and virtualization . . . 23
  - 1.1.5 Deployement models of a Cloud . . . 25
- 1.2 Cloud service and pricing models . . . 26
  - 1.2.1 Cloud service models . . . 26
  - 1.2.2 Pricing model in Cloud computing . . . 28
- 1.3 Problem statement: Reliable virtual network mapping . . . 28
- 1.4 Thesis contribution . . . 31
- 1.5 Thesis outline . . . 33

2 Virtual network resource provisioning overview **35**
- 2.1 Overview of basic virtual network Eembedding without reliability constraint . . . 36
  - 2.1.1 Online approaches . . . 36
  - 2.1.2 Batch approaches . . . 37
- 2.2 Overview of Virtual Network Embedding with Reliability Constraint . . . 38
  - 2.2.1 Distributed approaches . . . 38
  - 2.2.2 Centralized approaches . . . 40
    - 2.2.2.1 Substrate router failures . . . 40
    - 2.2.2.2 Substrate link failures . . . 42
    - 2.2.2.3 Substrate router and link failures . . . 43
    - 2.2.2.4 Regional failures . . . 44
  - 2.2.3 Summary . . . 48
- 2.3 Conclusion . . . 48

3 Preventive reliable virtual network embedding **51**
- 3.1 Problem statement . . . 52
  - 3.1.1 Virtual and substrate network models . . . 52

3.1.2 Formalization of $\mathcal{VN}$ preventive reliable embedding problem 53
3.2 Proposal: Advanced-PR-VNE 56
3.2.1 Embedding of virtual access routers 56
3.2.2 Erecting of solution components 57
3.2.3 Reliable embedding of solution component $\mathcal{SC}_i$ 57
3.2.3.1 Job of scout bees 57
3.2.3.2 Job of worker bees 59
3.2.3.3 Job of onlooker bee 62
3.3 Performance evaluation 62
3.3.1 Simulation environment 62
3.3.2 Performance metrics 63
3.3.3 Simulation results 64
3.3.3.1 Unsplittable virtual link embedding approach 65
3.3.3.2 Splittable virtual link embedding approach 68
3.3.3.3 Summary 71
3.4 Conclusion 73

4 Reactive survivable virtual network mapping 75
4.1 Problem statement 76
4.1.1 Virtual and substrate network models 76
4.1.2 Formalization of the reliable $\mathcal{VN}$ embedding problem 77
4.1.2.1 Modelization of virtual router mapping 77
4.1.2.2 Virtual router embedding basing on Game Theory 78
4.1.2.3 Modelization of virtual link embedding 80
4.1.2.4 Virtual network mapping constraints 81
4.2 Proposal: Advanced-CG-VNE 82
4.2.1 Embedding of virtual routers 82
4.2.2 Embedding of virtual links 84
4.2.2.1 Advanced-CG-VNE-Unsplit 84
4.2.2.2 Advanced-CG-VNE-Split 86
4.2.3 Reliability support 87
4.3 Performance evaluation 87
4.3.1 Simulation environment 87
4.3.2 Performance metrics 89
4.3.3 Simulation results 90
4.3.3.1 Unsplittable virtual link embedding approach 90
4.3.3.2 Splittable virtual link embedding approach 93
4.3.3.3 Summary 96
4.4 Conclusion 97

5 A Batch approach for a survivable virtual network embedding 101
5.1 Problem formalization 103
5.1.1 Substrate and virtual networks models 103
5.1.2 Reliable batch-embedding $\mathcal{VN}$ problem 104

5.1.3 Reliable embedding of one virtual network problem . . . 107
5.2 Proposal: BR-VNE strategy . . . 107
5.2.1 Tree search optimization algorithms . . . 108
5.2.2 BR-VNE description . . . 109
5.2.2.1 Tree initialization . . . 109
5.2.2.2 Selection of node to explore . . . 109
5.2.2.3 Generation of sub-branch . . . 110
5.2.2.4 Update of nodes' relevance . . . 110
5.2.2.5 Selection of final solution . . . 111
5.3 Performance evaluation . . . 111
5.3.1 Simulation environment . . . 111
5.3.2 Performance metrics . . . 112
5.3.3 Simulation results . . . 113
5.3.3.1 BR-VNE setting parameters . . . 113
5.3.3.2 Comparison with batch-embedding approach . . . 115
5.3.3.3 Comparison with online-reliable unsplittable strategy . . . 118
5.3.3.4 Comparison with online-splittable strategy . . . 120
5.4 Conclusion . . . 124

6 Conclusion **127**
6.1 Summary of contributions . . . 127
6.2 Future work . . . 129
6.2.1 Short-term perspectives . . . 129
6.2.2 Mid-term perspectives . . . 130
6.3 Publications . . . 131

References **133**

Chapter 1

Introduction

Contents

1.1 Cloud computing definition and architecture 18
1.1.1 Definition of Cloud computing 19
1.1.2 Cloud architecture 19
1.1.3 Software stack 22
1.1.4 Cloud computing and virtualization 23
1.1.5 Deployement models of a Cloud 25
1.2 Cloud service and pricing models 26
1.2.1 Cloud service models 26
1.2.2 Pricing model in Cloud computing 28
1.3 Problem statement: Reliable virtual network mapping 28
1.4 Thesis contribution 31
1.5 Thesis outline 33

The Internet has successfully ensured the world wild communication during last decades. Nowadays, it is clear that the remarkable emergence of new network services (i.e., gaming, video streaming, visio conference, etc.) requires a further growth of the Internet in order to respect the negociated Service Level Agreement (i.e., SLA) for each client. Indeed, while applications on the top level of the Internet are continuously evolving, its core architecture remaines unchanged except few small patches [1]. In other terms, enhancing the current design of the Internet is restricted to some incremental updates. Besides, deploying new technologies to improve the Internet performances is becoming more difficult [2] [3]. Consequently, most network researchers agree that redesigning the Internet architecture is an important need and not a luxury [4]. To overcome this deadlock and to fend off the Internet ossification, current industrial businesses and end-user activities strongly rely

on services provided by Cloud computing. It is noteworthy that 180 billion USD is the estimated global Cloud services market by the end of 2015 basing on a recent Gartner study [5].

Cloud computing success is mainly achieved thanks to its economic billing system that is based on pay-as-you-go (i.e., pay-per-use) model. Moreover, Cloud computing economic capitalizes on its elasticity to overcome the risks of overprovisioning (i.e., underutilization) and underprovisioning (i.e., saturation) that some companies may suffer from [6]. As a result, this new paradigm (i.e., Cloud computing), described as the long-held dream of computing as a utility, makes the illusion of infinite computing resources (i.e., hardware and/or software) available on demand. It should be noted that the key feature behind Cloud computing expansion and the illusion of infinite computing resources is virtualization. The latter enables the coexistence of different processes issued from several clients by guaranteeing the isolation between them. Furthermore, virtualization allows the continuous extensibility and shrinking of clients demands.

Despite the fast transition towards Cloud computing use, some critical issues are raised and remain not fixed. The main challenges that IT community and Cloud providers should fix sooner are mainly related to the following fields: i) safely data saving, ii) high availibility iii) security, iv) interoperability, v) privacy and data confidentiality [7], etc. Besides, one of the critical issues that continuously damages the Cloud-based business is the physical outages occured on the underlying hardware. For instance, in the north of America unplanned Information Technology (IT) failures cost more that 5000 USD per minute [8].

In this thesis, we tackle the problem of network resource provisioning within Cloud's backbone while taking into consideration physical outages. For this aim, we will propose new efficient resource allocation strategies while considering the reliability of the underlying physical infrastructure and maximizing the revenue of the Cloud provider.

This chapter is organized as follows. First, we introduce the Cloud computing paradigm with its definition. Second, we highlight the importance of Cloud-based services and its pricing models. Third, we introduce Cloud computing backbone connecting different data centers, as well as we describe the virtual network mapping problem with the consideration of the reliability issue. In other terms, we describe the problem that will be addressed by our research. Finally, we underline our contributions.

1.1 Cloud computing definition and architecture

Nowadays, Cloud computing market continues to soar. Based on the Cisco study [9], spending on Cloud computing infrastructure and platforms will grow at 30% CAGR (i.e., Compound Annual Growth Rate) from 2013 through 2018 compared with 5% growth for the overall enterprise IT.

1.1.1 Definition of Cloud computing

Inspite of its great expansion, the Cloud computing concept suffers from the absence of a standard definition. For instance, through the literature, the number of Cloud computing definitions exceeds twenty [6] [10] [11]. Fortunatly, the National Institute of Standards and Technology (NIST) definition [12] is well adopted by industrials and academic researchers: "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

Basing on the above definition, Cloud computing paradigm mainly includes five essential characteristics:

1. On-demand self-service: A client is able to unilaterally provision computing capabilities (i.e., hardware and/or software) as needed automatically without a human interaction with the service provider.

2. Broad network access: Cloud resources are accessible via the network using only an Internet browser. Thus, its capabilities are easily requested from heterogeneous thin or thick client end-devices (e.g., mobile phones, tablets, laptops, and workstations, etc.).

3. Resource pooling: Basing on a multi-tenant model, the Cloud provider's resources are pooled in such a way to dynamically serve multiple consumers with different physical and virtual capabilities.

4. Rapid elasticity: The Cloud user has the illusion of an infinite available resources thanks to the flexibility of the resources provisioning.

5. Measured service: Cloud system is a charge-per-use (i.e., pay-per-use) basis. Indeed, a Cloud provider optimizes resource use by leveraging a metering capability. This model is benefit for both; consumer and provider.

1.1.2 Cloud architecture

Cloud computing extensively relies on data centers in order to perform High Performance Computing (HPC), abundant storage, etc. Consequently, interconnecting data centers to ensure the communication is becoming a high requirement to ensure Cloud-based applications continuity. In [13], authors proposed an improvement of VICTOR architecture [14]. This new network architecture represents the Cloud's backbone illustrated in Figure 1.1. In fact, basing on a fully virtualized routers, the Cloud-based applications can be processed in dedicated slices simultaneously in the same machine. As depicted in this figure, there are two applications (i.e., green and red colors) that are served basing on this architecture. These applications are consuming physical resources in

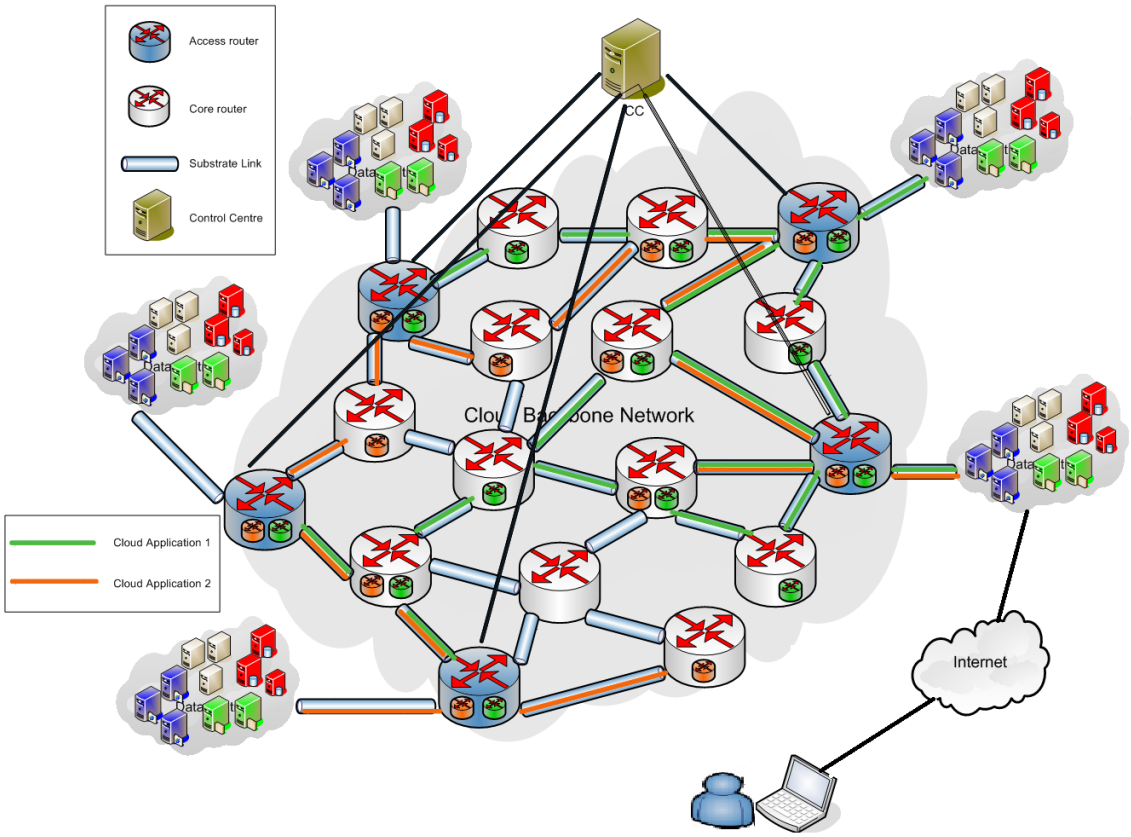


Figure 1.1: Cloud Backbone [13]

the underline routers and links. The Central Controller (CC), which is a high performance server, implements the policy adopted by the infrastructure provider to lease the backbone capabilities. Besides, the CC manages and controls the substrate network. It should be highlighted that we distinguish two kinds of routers: i) core and ii) access (i.e., edge). The access routers connect one or more data center(s) to the Cloud backbone, while a core router serves to join geographically distant data centers. It is worth pointing out that both types of routers are able to host different kinds of customer applications. Basing on a Cisco study [9], the Cloud-based traffic inter-data centers is compared to the traditional traffic between 2013 and 2018. As depicted by Figure 1.2, the Cloud-based traffic is continuously taking extent. In fact, as it is shown in Table 1.1 the Cloud-based traffic circulating between data centers is estimated to reach 6496 ZetaBytes (ZB) versus only 2079 ZB for the traditional traffic. This comparison emphasizes the importance of the Cloud-based traffic and so the Cloud backbone.

Table 1.2 describes a deep analysis of Cloud-based traffic. In other terms, it compares the different network traffic with respect to other criteria. It is straightforward to see that the maximum

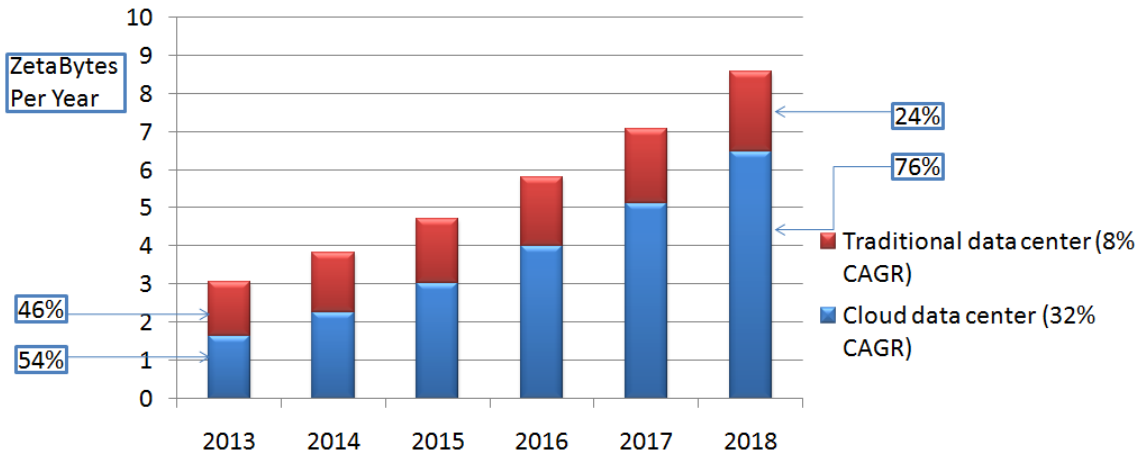


Figure 1.2: Cloud-based traffic versus traditional traffic [9]

workload is achieved within the data center (i.e., intra data center traffic). On the other hand, the traffic between data centers (i.e., inter data centers or backbone communication) also remains important even if it is the lowest traffic load. In fact, it is estimated to be about 729 ExaBytes (EB) by the end of 2018. This huge volume of data transiting between data centers over the Cloud backbone must be handled by a fair management policy in order to ensure the communication. Hence, the Cloud backbone is highly important and Cloud providers have to take care about it in order to guarantee a reliable Cloud-based application. In this context, it is noteworthy that the network ca-

Table 1.1: Cloud-based traffic versus traditional traffic, 2013 - 2018 [9]

By Type (EB per Year)	2013	2014	2015	2016	2017	2018	CAGR 2013 - 2018
Cloud data center	1,647	2,277	3,050	3,994	5,131	6,496	32%
Traditional data center	1,417	1,551	1,680	1,805	1,941	2,079	8%

Table 1.2: Global Data Center Traffic, 2013 - 2018 [9]

By Type (EB per Year)	2013	2014	2015	2016	2017	2018	CAGR 2013 - 2018
Data center to user	513	643	797	979	1,196	1,457	23%
Data center to data center	202	266	346	447	572	729	29%
Within data center	2,350	2,920	3,587	4,373	5,303	6,389	22%

pabilities can be leased to customers and this is defined by Network as a Service (NaaS). In NaaS, the end-user specifies the networking requirements to the service provider who translates them into a Virtual Network ($\mathcal{VN}$) topology. A virtual network is a logical topology typified with resources (i.e., bandwidth, cpu, memory) requirements describing the requested SLA by the client.

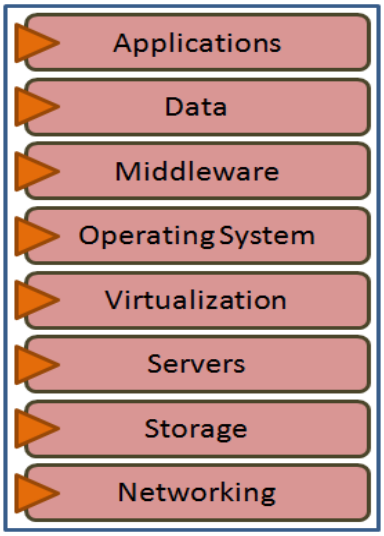


Figure 1.3: Cloud's machine software stack [9]

1.1.3 Software stack

After the macrospic presentation of the Cloud computing, let's introduce the architecture of a generic machine used in the Cloud environment. Figure 1.3 describes a generic Cloud-based machine software stack. In fact, a generic Cloud-based machine can be whether a server in a data center or a router in the Cloud backbone. Hereby, we define these layers in a bottom-up description.

- Networking: It includes all network capabilities that enable the communication between the current machine and its neighbours. This layer is composed by the physical network interfaces and the correspondant drivers. It should be highlighted that these interfaces can be wired or wireless based.
- Storage: This layer is composed by the hardware disk(s) and the underlying drivers.
- Servers: The current layer includes the computing capabilities that the machine can provide.
- Virtualization: This layer allows the use of the machine by different customers basing on their applications preferences. In other terms, it enables the provider to minimize the under-utilization of the machine by exposing its capabilities to different clients. This virtualization layer is crucial for Cloud deployement and expansion, that's why it will be deeply presented in Section1.1.4.
- Operating System (OS): Basing on virtualization layer, the provider can deploy different operating systems in order to diversify the offered services. Hence, the revenue will be enhanced by serving and statisfying the maximum of clients.

- Middleware: This layer can be introduced as a complementary software bloc that provides services to some applications beyond those provided by the OS. Besides, it is an intermediate layer that facilitates the interaction between different software.

- Data: This layer organizes the storing of data and provides useful Application Program Interfaces (APIs) to properly manage the access and the update of the saved data.

- Applications: This is the top layer that defines the interaction between the end-user and the Cloud service. Usually, the Cloud-based service is accessible basing on Internet connection via a web browser.

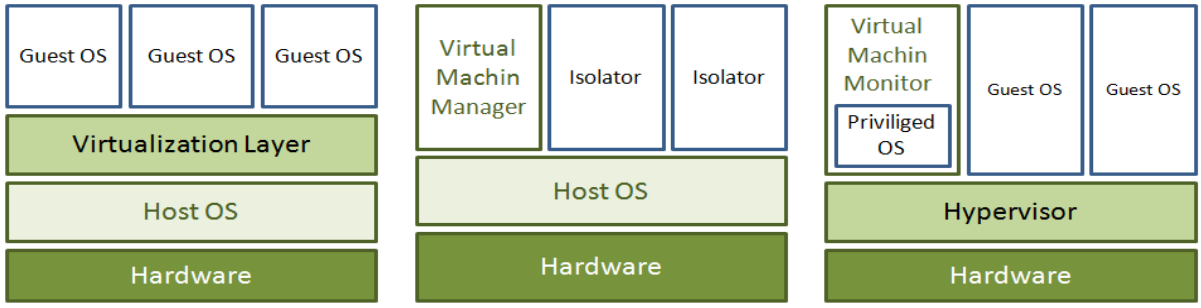
1.1.4 Cloud computing and virtualization

It is straightforward to see that virtualization is the key feature behind the Cloud computing expansion. It was claimed in [15] that "*what makes Cloud computing new and differentiates it from Grid computing is virtualization*". The basic role of virtualization is to set up independent Virtual Machines (VMs) that are isolated from each other as well as the underlying infrastructure. Technically, we distinguish three main kinds of virtualization [16] [17]: i) Full virtualization, ii) Container virtualization [17] (i.e., Isolation [16]) and iii) Para-virtualization.

1. **Full-virtualization**: As depicted by Figure 1.4(a), full-virtualization represents an exact emulation of the underlying hardware. It offers isolation and security for virtual machines and simplifies migration and portability. One interesting sub-kind of the full-virtualization is the Hardware-Assisted Virtualization [18] which was first introduced by IBM in 1972. The latter has the advantage to minimize the maintenance overhead comparing to other virtualization technique as well as to achieve good performance. However, it relies on a specific CPU architecture that is not available in all chipsets (i.e., CPUs). The most popular software using the full-virtualization technique are VMware [19] and VirtualBox [20].

2. **Container virtualization (i.e., Isolation)**: It allows the creation of completely isolated process contexts within the Operating System, instead of running different machines. Accordingly, it is seen as the best solution basing on the performance metric. However, isolation exists only for Linux systems. Figure 1.4(b) describes this technique. For instance, OpenVZ [21] is a solution adopting the isolation technique.

3. **Para-virtualization**: It is implemented basing on a modified version of the OS kernel. As shown in Figure 1.4(c), the para-virtualization provides a virtual machine abstraction that is similar but not identical to underlying hardware. Xen [22] and HyperV are examples of para-virtualization products.



(a) Full virtualization (b) Isolation (c) Para-virtualization

Figure 1.4: Kinds of virtualization [16]

Thanks to virtualization, Cloud provider can optimize the use of the physical resources by making them available to different customers while ensuring isolation between all processes. Moreover, Infrastructure as a Service (IaaS) providers extensively make use of virtualization in order to decompose and allocate the physical capabilities in an ad-hoc manner. Consequently, Cloud providers meet the shrinking and/or growing resource demand from clients. It should be highlighted that virtualization can be applied on: i) cluster, ii) data storage server, iii) network equipment and so forth. In this context, network virtualization aims to define a high-level of abstraction that decouples the physical layer from the virtual layer. In the literature, network researchers introduced various logical and physical network virtualization strategies. Even if these scientific investigations did not make use of the current virtualization technique, but they can be described as the predecessors of the current network virtualization architecture within the Cloud-based environment. Hereafter, we summarize them:

- Virtual Local Area Network (VLAN) [23]: It introduces a group of logical networks operating within the same broadcast domain regardless the underlying hardware infrastructure.

- Virtual Private Network (VPN) [24] [25] [26]: It is a use case of network virtualization environment where a dedicated connection between different sites is implemented via secured and private tunnels. It should be noted that this logical tunnels make use of a public or shared communication networks.

- Overlay Networks [27]: An overlay network is a logical network that is operating over one or more existing physical networks. The starting of the Internet can be considered as an overlay on top of the telecommunication network. Unlike VPNs, overlay networks have no geographic limitations. However, overlays require high maintenance which is a high costing task for companies.

Whereas, most of these investigations, made in the field of virtual networking, may be considered as limited attempts to fix some existing issues related to the current Internet architecture. Fortunatly, network virtualization within Cloud networks is considered as a potential solution that is expected to solve the Internet ossification problem [28].

Network virtualization allows to optimize the use of the underlying infrastructure by splitting it to different clients in secured and independent way. There are two main components in Cloud-based network virtualization; i) link virtualization and ii) node virtualization [28]. In fact, the substrate link will be able to simultaneously host different logical tunnels (i.e., virtual links). While, substrate routers are virtualized in order to have multi stack protocols implementation. Consequently, Cloud provider can serve a multitude of customers and satisfy their demands without any restrictions.

In network virtualization, the role of the Internet Service Provider (ISPs) is decoupled into two parts: i) the Infrastructure Provider (InP) and ii) the Service Providers (SPs) [27]. The InP owns and manages the physical network equipment. Parallely, the infrastructure is leased to different clients while ensuring isolation basing on the allocation of dedicated slices. On the other hand, the service provider satisfies the demands of end-users by requesting virtual network resources from one or more InP(s). In fact, the latter (i.e., SP) is considered as the broker between the end-user and the InPs. The SP translates the client's requirements to a virtual network topology which will be hosted in the InP physical network.

1.1.5 Deployement models of a Cloud

More recently, Cloud community [12] has introduced four deployment models in order to define a Cloud service:

1. **Private Cloud**: The related infrastructure is exclusively operated for a private organization. This Cloud infrastructure can be directly owned and managed by the corresponding organization or a third party. The motivation behind this kind of deployement is mainly to ensure further security, data privacy and trust.

2. **Community Cloud**: Unlike the Private Cloud, many organizations jointly build and share the same: i) Cloud infrastructure, ii) requirements, iii) policies, etc. There is no restriction regarding the management of the Cloud infrastructure. In fact, it can be ensured by one or many organizations in the community or a third party.

3. **Public Cloud**: Currently, this is the most dominant Cloud deployement [29]. In this kind of deployement, the Cloud infrastructure is provisioned for open use by the general public without any restriction. Public Cloud may be owned, managed, and operated by a business, academic organization, or a combination of them. Many popular Cloud services are public Clouds such as Google AppEngine, Amazon EC2, Force.com, etc.

4. **Hybrid Cloud**: This Cloud deployement model is a composition of two or more different Cloud infrastructures (i.e., private, community, or public) that remain unique entities, but are bound together by a dedicated technology ensuring data and application portability (e.g., Cloud bursting for load balancing between Clouds). Organizations opt for the hybrid Cloud model in order to optimize their resources. Besides, it allows organizations to esure some level of privacy and security. Hybrid Cloud has raised the issues related to standardization and Cloud interoperability that remain open problems.

1.2 Cloud service and pricing models

Hereafter, we will present the general taxonomy of the Cloud-based services. Then, we discuss the potential pricing models that can be used to organize the remuneration process between the customer and Cloud provider.

1.2.1 Cloud service models

The IT community has categorized the Cloud service models to three main services [12]:

1. **Software as a Service (SaaS)**: The capability provided to the customers is to use software and applications hosted in the provider's infrastructure. These applications are runnable via heterogeneous end-devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. This model alleviates the clients from license issues, updating and upgrading software. This service model is sometimes called "software on demand". Examples of SaaS include Google mail, Microsoft 365, Salesforce, Citrix GoToMeeting, Cisco WebEx, and so forth.

2. **Platform as a Service (PaaS)**: Also referenced as Cloud platform services. It is a development platform which allows Cloud consumers to develop their own services and applications basing on the provided Cloud APIs (Application Programming Interface). In other terms, PaaS allows to developers to create and customize their own applications using the available software components (i.e., APIs). This kind of Cloud service is similar to the manner in which a programmer may create macros in Excel for further use. PaaS makes programming, testing, and deployment of applications easy, simple, quick and cost-effective. For instance, Google's App Engine provides APIs to interact with Google's Cloud.

3. **Infrastructure as a Service (IaaS)**: The capability supplied in this service are hardware resources. Instead of purchasing clusters, data centers or network equipment, end-users can fully access to the provider's infrastructure and use it on the fly. IaaS allows industrials, academics and companies to overcome the problem of overprovisioning (underutilization)

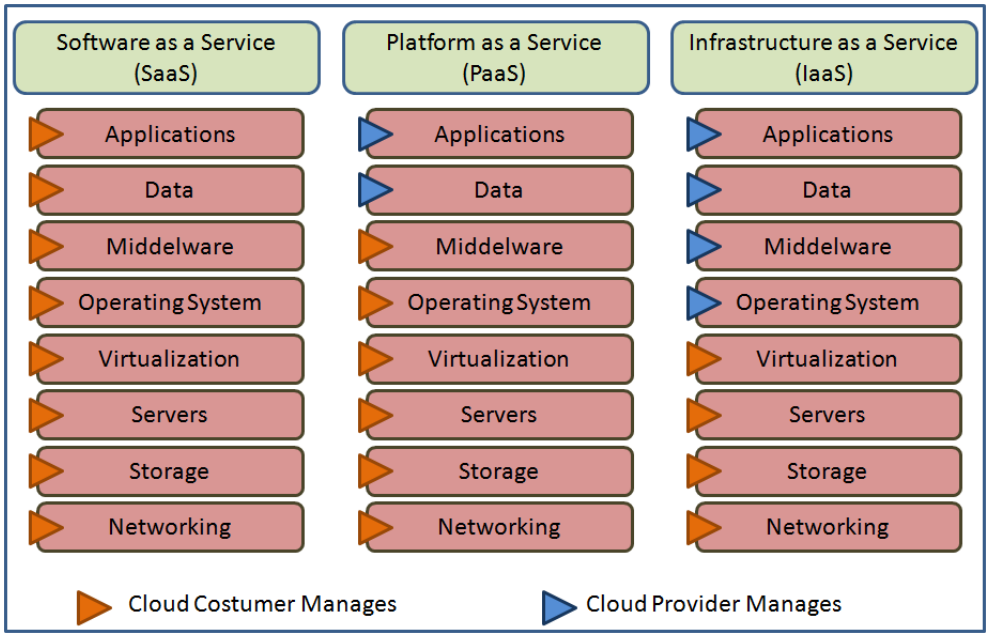


Figure 1.5: Cloud Service Models: SaaS, PaaS, and IaaS [9]

and underprovisioning (saturation) by allocating on-demand physical resources. Examples of IaaS include Amazon Web Service (AWS), Google Compute Engine (GCE) and Microsoft Azure.

Figure 1.5 summarizes the aformentionned service models. In fact, it clearly describes what are the authorized management tasks to both: i) the Cloud customer and ii) the Cloud provider. A deep classification includes Data storage as a Service (DaaS) and Network as a Service (NaaS), but they are considered also as a sub kind of IaaS. Figure 1.6 describes the general growth of Cloud-based business revenue from 2013 and a forecast till 2018. As depicted by this figure, Cloud's business reaches 24% CAGR of general growth from 2013 to 2018. Furthermore, a detailed view about the growing of each one of the Cloud services (i.e., SaaS, PaaS and IaaS) is presented in Figure 1.6. It is clear that PaaS has the lowest business revenue, nevertheless Cloud platform service (i.e., PaaS) growing is estimated to reach 16% by the end of 2018. It is noteworthy that IaaS was leading the Cloud-based business in 2013 by ensuring 49% of the whole revenue but it is estimated that SaaS will take the leadership by reaching 64% of the global turnover by the end of 2018. However, it should be highlighted that both of IaaS and SaaS are growing up by 3% and 37%, respectively.

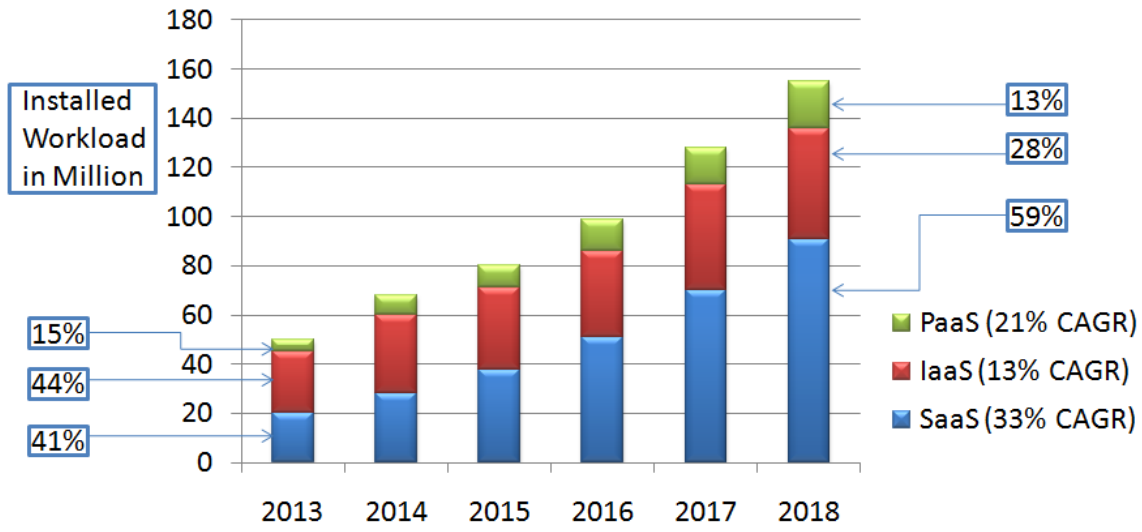


Figure 1.6: Analysis of public Cloud-based business's growing [9]

1.2.2 Pricing model in Cloud computing

From the Cloud provider's perspective, improving the turnover is the main objective. Indeed, it was claimed in [30] that Cloud computing expansion and success are directly dependent on the developping an adequate pricing model. The pricing process can be categorized to two main groups: i) fixed and ii) dynamic (i.e., market-dependent). In the latter pricing model, the customer is charged based on the real time market conditions [31]. On the other hand, fixed pricing essentially includes two sub-models:

1. **Pay-per-use** (i.e., pay-as-you-go) model: Customers pay for the amount they consume of or for the time period they access to the service.

2. **Subscription**: The client pays a fixed amount to benefit from a service for a long period at any convenient time.

It is noteworthy that the leading companies in the Cloud-based market like Amazon, Google and Microsoft make use of the pay-per-use model which capitalizes on the Cloud elasticity.

1.3 Problem statement: Reliable virtual network mapping

In NaaS context, the main task of the infrastructure provider (i.e., Cloud provider) is to allocate the physical network capabilities to the incoming clients. As depicted by Figure 1.7, three actors are intervening.

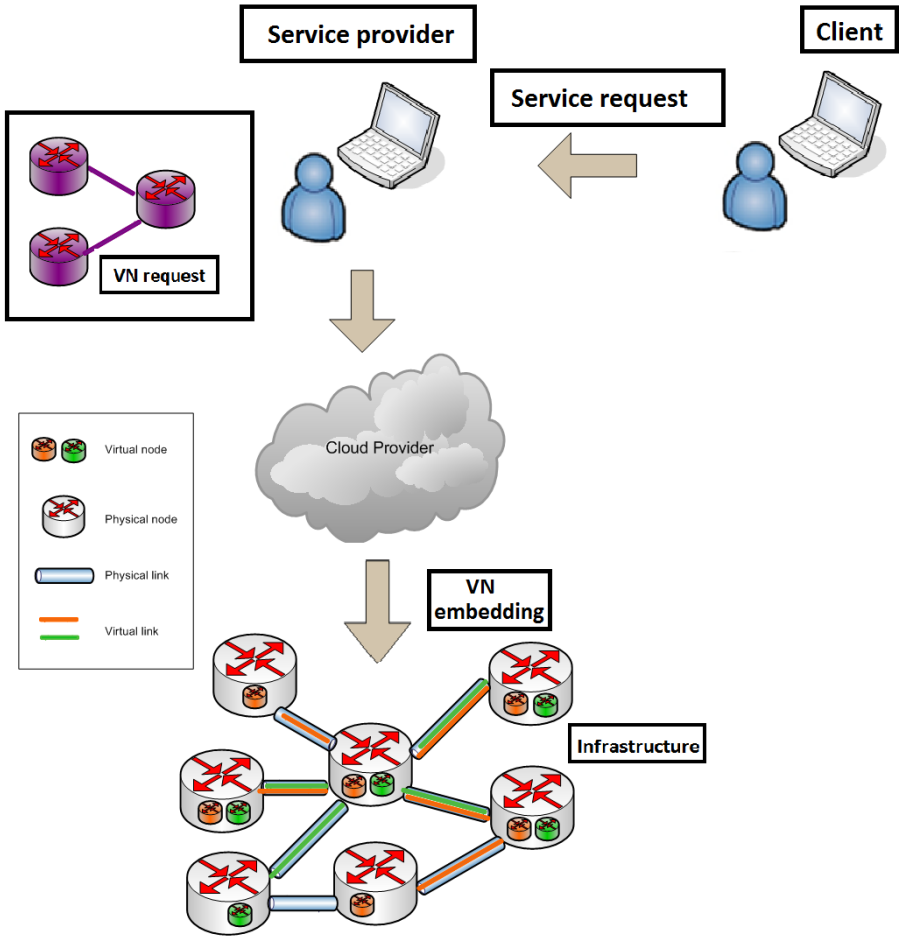


Figure 1.7: NaaS Scenario

1. **Client**: Also called the customer or the end-user. The latter actor can be an academic institute, a company, a government, etc. The end-user specifies the required Quality of Service (QoS) to be obtained during the Cloud service. Besides, the customer directly communicates and negociates with the service provider.

2. **Service Provider (SP)**: It is working as a broker between the two remaining actors. In fact, the different clients' network requests with the required QoS are processed in order to create a Virtual Network ($\mathcal{VN}$) which represents a logical topology with resource requirements. This $\mathcal{VN}$ will be instanciated (mapped or embedded) into the Substrate Network ($\mathcal{SN}$). Afterwards, the service provider looks for the suitable Infrastructure Provider (InP) or a combination of InPs [27]. The selection of the convenient InP is concluded by the signature of the Service Level Agreement (i.e., SLA) between the SP and InP(s).

Figure 1.8: Physical failure impacts

3. **Infrastructure Provider (InP)**: Also called the Cloud Provider ($\mathcal{CP}$). The $\mathcal{CP}$ maintains and manages the physical network equipment. Besides, the policy to handle the requests issued from different clients is defined. The InP's policy includes i) pricing parameters, ii) maintaining the network equipment iii) clients' arriving process (i.e., online or batch) , iv) embedding strategy, v) the reliability strategy and so forth. The main objective of the $\mathcal{CP}$ consists in enhancing the revenue by: i) accepting the maximum of clients and ii) respecting the signed SLA in order to avoid penalties induced by its violation.

During the mapping process of a $\mathcal{VN}$, the physical resources of the $\mathcal{CP}$'s backbone will be consumed. In fact, each virtual router requires both computing (i.e., cpu) and memory capabilities. Moreover, each virtual link has a bandwidth demand that should be satisfied by consuming the available bandwidth in some substrate link(s). When the leasing period of a $\mathcal{VN}$ is expired, all the allocated physical resources for this $\mathcal{VN}$ will be released. This dynamicity in the Cloud backbone urges the $\mathcal{CP}$ to adopt a convenient embedding algorithm in order to maximize the revenue by rejecting the minimum of clients. This requirement of a judicious mapping algorithm is also motivated by the finite hardware resources available in the $\mathcal{SN}$.

Due to the sharing of same network equipment by multiple clients, one physical failure (i.e., or outage) will impact many customers. In fact, virtualization is considered in this context as a double-edged weapon. Indeed, it allows an optimized usage of the $\mathcal{CP}$'s backbone but, at the same time it leads to more damage caused by the service interruption due to physical outages. As depicted by Figure 1.8, one failure in the physical router simultaneously impacts three clients. Moreover, all the attached physical links to the failed substrate router will be down. Hence, more clients will be impacted. So, the $\mathcal{CP}$ will pay more penalties due to the SLA violation. Consequently, the $\mathcal{CP}$ should adopt a fault tolerant mechanism in order to avoid penalties. In order to highlight the revenue losses caused by unplanned IT outages, we mention the below statistics [32]:

- IT businesses in North of America are collectively losing 26.5 billion USD in revenue each

year through IT downtime and data recovery. On average, each company loses $159,331$ USD yearly.

- North American businesses collectively suffer from $1,661,321$ hours of IT downtime each year. That is an average of 10 hours per company, per year.
- During these periods, when business critical systems are interrupted, companies estimate that their ability to generate revenue is reduced by 29%.
- Post IT downtime, (i.e., when IT systems are up and running), there is an additional delay of 7.5 hours per year at each firm during which time data is still being recovered. Across North America, that's another $1,255,220$ hours when business operations are not fully operational.
- In this post-outage period when data recovery is taking place, company revenue generation is still severely hampered, down by an average of 17%.

In this thesis, we will address the problem of the reliable virtual networks embedding within Cloud's backbone network. Our focus is to design new survivable mapping algorithms in the $\mathcal{CP}$ side to reach the objective of maximizing the turnover. In other terms, we will tackle the problem of service provisioning within the Cloud's backbone network while considering also the reliability. Indeed, we deal with the Network as a Service (NaaS) problematic within the Cloud Provider ($\mathcal{CP}$) backbone as well as taking into consideration the reliability issue.

The basic virtual network embedding problem without considering the reliability is NP-hard [33] [34] [35]. In fact, the following problem with constrained physical capabilities on substrate routers and links, can be reduced to the NP-hard multi-way separator problem [27], even if all the requests are known in advance. Besides, even the simplified version of this problem assuming all the virtual routers are already mapped, the embedding of the virtual links with bandwidth constraints into substrate paths is still NP-hard in the unsplittable flow scenario [27]. Accordingly, the survivable virtual network embedding is NP-hard problem.

1.4 Thesis contribution

In this section, we will summarize the main contributions of this thesis. In fact, the significant contributions are described as follows.

- **A survey of the survivable virtual network mapping strategies**

  First, we provide a brief overview of the virtual network embedding that do not tackle failures constraint. The proposed approaches in the literature can be categorized according to [36]: i) centralized or distributed, ii) static or dynamic. Second, we will provide a deep survey of the

survivable virtual network mapping approaches. A taxonomy of this kind of strategies can be based on the main following criteria: i) centralized or distributed, ii) proactive or reactive, iii) the type of the protected resource (i.e., router, link, geographic region) iv) backup use, v) curative or preventive. It is noteworthy that all the reliable virtual network embedding strategies found in the literature are online strategies. In other words, they handle the client requests as soon as they arrive to the Cloud provider. It should be noted that online approaches are not well adopted by Cloud providers because they aim to: i) reduce the access to the network capabilities and ii) maximize the accepting rate by having a macrospic view on the postponed requests.

- **Proposed algorithms for the reliable virtual network embedding problem**

  To tackle the aformentionned complexity of the survivable virtual network mapping, we propose a selection of reliable resource allocation algorithms. Our proposed strategies are threefold. In fact, we follow a progressive enhancement in the way we define our proposals. First, we design a preventive approach that does not allocate any backup resources. Second, we propound a reactive approach that enhances the first proposal. Indeed, it adopts a recovery aproach without relying on a dedicated backup. Finally, we propose a batch reliable virtual network mapping algorithm. The latter algorithm makes use of the previous algorithms (i.e., first and second proposed strategies) to define a time-window approach that embeds client requests in a batch mode. To the best to our knowledge, we are the first to define: i) preventive approach, ii) reactive strategy and iii) survivable batch $\mathcal{VN}$ embedding algorithm. Hereafter, we sum-up each one of our proposal.

  - In order to cope with the complexity of the survivable virtual network embedding problem, we opt for the use of metaheuristic since they generally ensure reaching an optimized (i.e., near-optimal) solution. Indeed, we make use of Artificial Bee Colony metaheuristic [37] [38] to define our Preventive Virtual Network Embedding strategy denoted `PR-VNE` [39]. In other terms, we divide our virtual network topology into star topologies called Solution Components ($\mathcal{SC}$s) basing on "divide and conquer" rule then we mimic the behaviour of bees to embed them. To do so, artificial worker bees are defined to simultaneously simulate the embedding of the generated $\mathcal{SC}$s. Later on, basing on the evaluated nectar by each worker bee that matches with the relevance of one $\mathcal{SC}$ mapping, the artificial onlooker bee selects the suitable mapping of the current $\mathcal{SC}$. It is worth pointing out that `PR-VNE` deals with reliability by only preventing the non-reliable resources. Despite its originality, once a physical failure occurs, `PR-VNE` does not recover the impacted clients.
  - Aiming to to overcome the limitation of `PR-VNE`, we define a reactive approach that

re-embeds the impacted virtual resources (i.e., router and/or link) by a hardware outage. This new strategy called Coordination Game for Virtual Network Embedding (CG−VNE) [40] strongly relies on a Game theory [41] framework. In fact, we define fictitious players that collaborate together and play on behalf of the Cloud provider in order to improve the revenue. CG−VNE deals with reliability in two stages: i) preventing the non-reliable hardware equipment and ii) re-mapping the impacted virtual resources by a physical failure in other substrate resources. Our solution does not rely on dedicated backup resources in order to optimize the use of network capabilities and espacially bandwidth. We define a central controller that periodically receives snpashots from the substrate routers. In such way, the recovery mechanism is ensured without any service interruption thanks to the migration of the impacted resources to new physical locations. Accordingly, the Cloud provider turnover is enhanced by i) maximizing the acceptance rate thanks to cooperative mapping game and ii) minimizing the hardware outages impacts. However, infrastructure provider usually prefers the batch mapping mode that minimizes the access to the backbone network in order to reduce some failures due to management issues.

– Despite its importance in the industries, batch mapping took less focus than the online one. Basing on the aformentionned algorithms, we define a Batch Reliable Virtual Network Embedding (BR−VNE) [42] approach that strongly relies on artificial intelligence. In fact, the most important criterium that influences the batch embedding process is defining the mapping order. In this context, we notice that seeking for the best order is equivalent to finding the best path from the root to a leave in the mapping decision tree. Consequently, the batch embedding process can be seen as a tree search problem. BR−VNE is based on Monte-Carlo Tree Search (MCTS) [43] algorithm that is extensively used in the computer gaming problem. BR−VNE deals with physical failures with the same reactive strategy adopted by CG−VNE without any dedicated backup.

1.5 Thesis outline

This thesis is organized as follows. In Chapter 2, we will sum-up the related work dealing with virtual network embedding without considering the reliability. Then, we will discuss the different reliable virtual network resource provisioning strategies found in existing literature and we will classify them basing on different criteria. In Chapter 3, we will describe our Preventive Virtual Network Embedding proposal based on Artificial Bee Colony metaheuristic (PR−VNE). Afterwards, we will present our reactive approach CG−VNE in Chapter 4. The latter is defined basing on a Game Theory framework. In Chapter 5, we will describe our Batch Reliable Virtual Network Embedding (BR−VNE) algorithm. Finally, Chapter 6 will conclude the thesis and presents our future

work in this research area.

Chapter 2

Virtual network resource provisioning overview

Contents

2.1 Overview of basic virtual network Eembedding without reliability constraint 36

2.1.1 Online approaches 36

2.1.2 Batch approaches 37

2.2 Overview of Virtual Network Embedding with Reliability Constraint 38

2.2.1 Distributed approaches 38

2.2.2 Centralized approaches 40

2.2.3 Summary 48

2.3 Conclusion 48

Few survivable virtual network embedding algorithms have been proposed in the literature. Indeed, we can classify the related methods into two main groups: i) *centralized* and ii) *distributed* approaches. Besides, in each group, the reliable virtual network embedding strategies assume that failures can occur in i) routers and/or links failures in the whole $\mathcal{SN}$ or ii) in a defined geographic region. Hereafter, we will present a summary of the prominent survivable strategies with respect to more criteria. To do so, first of all, in Section 2.1 we start by presenting a short overview of the baseline strategies which do not consider the reliability of physical resources. Afterwards, Section 2.2 will provide a deep overview of the survivable virtual network mapping strategies.

2.1 Overview of basic virtual network Eembedding without reliability constraint

The virtual network embedding problem is NP-hard [33] [34] [35]. In order to skirt its complexity, IT researchers consider sometimes an infinite substrate resources or relax some other conditions. Besides, they opt to heuristic approaches to solve the virtual network mapping problem since the optimal solution is computationally intractable in large-scale networks. Hereinafter, we will summarize the most prominent online and batch algorithms.

2.1.1 Online approaches

In [44], the authors propounded original strategies (i.e., a set of algorithm variants) that coordinate the virtual router and link embedding stages. The simultaneous combination of router and link mapping is a major pros of this work. The authors enhance the substrate network by fictitious resources called meta-nodes and meta-links. A meta node is created to match with the potential substrate routers that can host the current virtual router. Then, the connection is ensured by establishing meta-links between the meta-node and its potential routers. It is noteworthy that the meta-link capability is infinite. Later on, the authors modelize the embedding problem as Mixed Integer Linear Programming (MILP). Basing on the above formalization an optimized solution is reached after a constraint relaxation and two variants are proposed: i) Deterministic-ViNE (`D-VINE`) and ii) Randomized-ViNE (`R-VINE`). We notice that the choice of the potential substrate routers is based on geographic criterium, however this can be a limitation to reach a pertinent solution. Besides, proposing a splittable approaches can be supported by the network provider only in the future because current routers do not implement such kind of splittable behaviour.

In [45] [46], the authors formulated the virtual network embedding problem as a non-cooperative game. In the proposed mapping game each one of the $\mathcal{VN}$ is represented by a decision maker (i.e., player). These players behave selfishly as competitors to satisfy their required bandwidth from the $\mathcal{SN}$ resources. In these papers, the authors did not justify why they devise a non-cooperate game. Furthermore, since the mapping algorithm will be executed by the $\mathcal{CP}$, the proposed solutions should enhance the revenue. In the proposed algorithms, the authors did not explain how may this non-cooperative game improve $\mathcal{CP}$ turnover. Besides, we notice that the authors did not deal with i) possible failures within the $\mathcal{SN}$ and ii) the substrate and virtual router capacities (i.e., memory and processing power). In fact, they only focused on the link resource (i.e., bandwidth) allocation. Moreover, the performance evaluation of the new proposal has been performed in a small-size $\mathcal{SN}$ which is not realistic. Accordingly, the authors did not study the proposal scalability. In fact, in this kind of small-size networks, the optimal solution can be computed in polynomial time without any optimized solution. Thus, the added value of the paper is not well defended. We notice, also, that the rejection rate of virtual network requests was not estimated in simulation scenarii. Indeed,

this latter metric is very important to describe the provider's turnover. Finally, the authors did not compare the propounded strategy with regard to the prominent related algorithms.

In [47], the authors propounded a new strategy, basing on the Ant Colony metaheuristic [48] [49], named VNE−AC to tackle the high complexity of the virtual network embedding problem by considering the mapping of virtual links in unsplittable manner. The authors decomposed the virtual network in elementary topologies in order to apply the divide and conquer principle. The main objective of the authors is to maximize the Cloud provider's revenue by accepting the maximum of clients. However, this proposal suffers from the lack of an optimal network load balancing. In order to overcome the problem of the overloaded substrate links, the authors proposed a new strategy named VNR [50] that reconfigures the network. It is noteworthy that this approach is a hybrid (i.e., reactive and proactive) strategy. VNR relies on the migration of some mapped virtual routers and their attached virtual links to other physical equipment in order to load balance the substratenetwork. By doing so, the Cloud's backbone would be able to host more clients. Accordingly, the provider turnover will be improved. Later on, in order to avoid the drawbacks of a static approach, the authors propounded an adaptive resource allocation scheme denoted Adaptive−VNE [51]. This new strategy is based on the K-supplier method to embed the elementary topologies of a virtual network request. It is worth pointing out that the authors did not consider the mapping of virtual links in splittable manner in order to gauge their strategies with this class of algorithms.

2.1.2 Batch approaches

Few methods tackled the problem of $\mathcal{VN}$ mapping in batch mode. Hereafter, we will describe some of the most prominent ones.

In [52], the authors propounded a 2-stages algorithm denoted by Batch-Baseline (B−Base) dealing with $\mathcal{VN}$s embedding in batch mode. B−Base sorts all queued $\mathcal{VN}$ requests with respect to their revenues and handles their mapping on this order. In the first stage, B−Base maps only virtual routers for the incoming $\mathcal{VN}$ requests. Then, the virtual links are embedded based on an unsplittable approach. We notice that B−Base adopts a greedy strategy for virtual routers and link mapping. In fact, this kind of approach may converge to a local optimum which will impact the $\mathcal{CP}$'s revenue. On the contrary, $\mathcal{CP}$ can earn more by accepting other $\mathcal{VN}$ requests with less revenue but the cumulative revenue would be higher than the one generated by accepting the requests having the maximum revenue. Moreover, the survivability of $\mathcal{VN}$ embedding is not considered.

In [53], the authors proposed a periodical auction-based planning of embedding process in order to handle the incoming set of $\mathcal{VN}$ requests. The propounded strategy modeled $\mathcal{VN}$s as bidders competing to win the access to the $\mathcal{SN}$ resources. In other words, each request is a selfish bidder that tries to be embedded within Cloud's backbone network. We notice that the proposed approach focuses on the competition between $\mathcal{VN}$s. However, the authors did not detail how can this approach (i.e, selfish) improves the cumulative turnover of $\mathcal{CP}$. In fact, the $\mathcal{VN}$s requests offering the

maximum of bids would have more chance to be accepted regardless of the total revenue generated within the $\mathcal{SN}$.

In [54], the authors proposed a new algorithm dealing with a Multiple Virtual Network requests Embedding (MVNE). The main objectives are: i) maximizing the $\mathcal{CP}$'s revenue and ii) balancing the usage rate of resources in the $\mathcal{SN}$ in order to host more $\mathcal{VN}$ requests. MVNE algorithm is based on a Mixed Integer Program. Unfortunately, this approach is not scalable. Accordingly, this proposal cannot be exploited in a large scale $\mathcal{SN}$.

In [55], the authors proposed a new algorithm, named Window-based virtual Network Embedding (WiNE), in order to gauge the lookahead impact. It discretizes the time into sequential time-windows and stores the incoming clients' requests to be processed in a batch manner at the end of a time-window. It is worth pointing out that each request is enhanced by a new information which is the maximum waiting period. This parameter is important to specify the maximum duration that a request can be postponed in order to be processed. At the end of this period, if the request is not mapped it would be rejected. The main advantage of WiNE is that it does not depend on the underlying online strategy. WiNE processes as follow. It sorts the stored virtual networks basing on their revenue. Then it embeds the top of the queue basing on the underlying online strategy. If the mapping of the current request fails due to physical resources lack, so it will be postponed to the next time-window processing. The authors made use of a 50-routers substrate network and varied the arrival rate from 4 to 8 time units per 100 time units. We notice that these parameters are not defined for a scalable scenarii and should be stressed more in order to deeply evaluate the performance of WiNE. Besides, sorting the requests basing on revenue could lead to a non optimized batch mapping.

2.2 Overview of Virtual Network Embedding with Reliability Constraint

Hereafter, we provide a deep survey of the survivable virtual network mapping approaches. A taxonomy of this kind of strategies can be defined based on the main following criteria: i) centralized or distributed, ii) proactive or reactive, iii) the type of the protected resource (i.e., router and/or link, geographic region) iv) backup use, v) curative or preventive. It is noteworthy that all reliable virtual network embedding strategies found in the literature are online approaches.

2.2.1 Distributed approaches

The authors introduced in [56] a new Distributed Fault-Tolerant Embedding (DFTE) algorithm. The new propounded strategy deals with router and link failures (or service degradation). Indeed, the proposed strategy relies on a multi-agent system to tackle the survivable virtual network mapping. In other terms, this new framework is composed of autonomous agents integrated into the substrate

routers that carry out a decentralized fault-tolerant virtual network embedding algorithm to cope with router and link failures. The new approach is basically a two-step strategy: i) Mapping the virtual infrastructure (VInf) request in a distributed manner and ii) continuously, checks the router and link state (i.e., by sending alive-message) and then reacts when failure is detected. The decentralized embedding algorithm relies on the approach detailed in [57]. Hereafter, we provide a summary:

- Resource description and advertisement: Infrastructure Providers (InPs) describe and customize their offered substrate resources.
- Resource discovery and matching: It consists of searching and finding resource candidates that comply with the requirements specified by the $\mathcal{VN}$ request.
- $\mathcal{VN}$ embedding: InP finds the optimal $\mathcal{VN}$ embedding in distributed way basing on the earlier steps.
- $\mathcal{VN}$ binding: The selected substrate resources are allocated by the InPs in order to instantiate the requested $\mathcal{VN}$.

Once the $\mathcal{VN}$ request is mapped, the proposed framework ensures: i) Detecting and identifying local changes through monitoring (e.g., node/link failure, performance degradation), ii) selecting new substrate resources to maintain $\mathcal{VN}$ topologies operational, iii) instantiating a virtual router in the new selected substrate router and iv) binding the virtual router along with the virtual links that are affected. The framework relies on the multi-agent based approach to ensure distributed negotiation and synchronization between the substrate routers. The negotiation and synchronization are optimized thanks to a clustering based technique. Hereby, we summarize the Distributed Fault-Tolerant Embedding Algorithm:

- Upon detecting a virtual node failure, send a notification message to all agents in the same cluster.
- Compute dissimilarity metric between the affected router and neighbour routers. It is noteworthy that dissimilarity metric describes the pertinence of the evaluated router. The router having the maximum metric is the worst choice that can be made.
- Exchange, via messages, the computed dissimilarity metrics within the same cluster. The agent compares its dissimilarity metric with all substrate nodes. The physical router having the minimum dissimilarity with the affected one will be selected.
- Map the associated virtual links to the substrate paths using a distributed shortest-path algorithm [58].

- Once link failure is detected, the algorithm proceeds directly to the previous step.

In order to evaluate the performance of the proposed framework, the authors resort to simulations. They focus specially on performance and scalability evaluation. The performance metric is basically the delay incurred by the proposed adaptive $\mathcal{VN}$ algorithm. Substrate topology is composed of 10 nodes and $\mathcal{VN}$ request is 5-routers. The time required to adapt the $\mathcal{VN}$ with the distributed embedding algorithm is always less than 2 seconds and is decreased further in the presence of multiple clusters. The authors deal with scalability evaluation through GRID 5000 [59] platform. GRID 5000 has been used to generate full mesh substrate topologies with different sizes (from 0 up to 100 nodes). Results show that the number of messages exchanged between substrate nodes decreases with clustering. Besides, the proposed strategy keeps a good recovery time from router failure.

It should be highlighted that the authors in this article define their proposal basing on multi-agent system, so all the underlying physical network resources should be updated to include an agent-based mechanism. This is not an easy task to be supported by the infrastructure provider in the near future. Furthermore, the authors did not explain whether each router in one cluster should have a snapshot of all the routers within its cluster or not. In fact, it is a mandotory task to ensure the migration upon detecting a router failure. However, this synchronisation may overload the network as well as occupy router memory. We notice, also, that the authors do not evaluate the resource consumption of the network capacities. Moreover, they do not deal with rejection rate of virtual networks which is directly impacted by recovery mechanism since more physical resources will be used. In fact, this metric expresses the earned benefit.

2.2.2 Centralized approaches

The most research investigation tackles the survivable $\mathcal{VN}$ embedding in centralized way. In fact, they assume the existence of a centralized controller that not only manages the mapping process but also ensures some level of fault-tolerant mechanism. The centralized strategies can be classified also basing on the protected physical resource(s): i) router, ii) link, iii) router and link, iv) geographic zone. Hereby, we summarize them respecting the aforementioned taxonomy.

2.2.2.1 Substrate router failures

In [60], the authors addressed the $\mathcal{VN}$ reliable embedding problem and proposed a new Opportunistic Redundancy Pooling (`ORP`) algorithm. The main idea behind the proposal is to share redundant (i.e., backup) resources between many $\mathcal{VN}$s. In other words, backups of some $\mathcal{VN}$s are pooled together. In such a way, the number of redundant resources is minimized. The number of backup routers is calculated analytically so that each $\mathcal{VN}$ can guarantee the required level of reliability. `ORP` operates as following. First, the $\mathcal{VN}$ request is increased with backup links and routers. Then, the problem is formulated as a Mixed Integer Linear Programming (MILP) solved by the open-

source CBC solver. The aforementioned steps are summarized in the pseudo-code Algorithm 1. To evaluate the performance of ORP, it is compared with related $\mathcal{VN}$ reliable embedding strategies assuming no-share of redundancy resources between $\mathcal{VN}$s. Besides, ORP is compared with the baseline approach (i.e., does not consider survivability) in order to gauge the additional amount of resources consumed for reliability. The simulation results show that ORP is better than no-share approaches in terms of rejection rate of $\mathcal{VN}$s and number of required backup resources. In [61], more technical details of ORP are exposed. In fact, the authors extended the Virtual eXecution Description Language (VXDL) [62] to enable the specification of reliable virtual infrastructures. Unfortunately, ORP does not deal with substrate link failures. Moreover, the match between a critical router and its associated backups is not detailed.

Algorithm 1: ORP [60] [61]

1 Inputs: Virtual and substrate networks described by graphs
2 Output: Working and backup mappings
3 Analytically computing the backup routers
4 $\mathcal{VN}$ graph is increased with backup routers and links
5 Mapping problem formulation basing on MILP
6 Solve the relaxed MILP mapping problem using CBC solver

In [63], the authors tackled the resilience $\mathcal{VN}$ embedding problem by considering substrate routers failures. A new heuristic approach is proposed named Survivable Virtual Infrastructure Mapping (SVIM). The latter is based on the K-Redundant scheme for surviving facility router failure ($K \geq 1$), thus it is denoted K-Redundant algorithm. Note a K-Redundant scheme means that K substrate routers are dedicated to serve as a backup for all critical routers. The objectives are i) to minimize the usage rate of physical resource and ii) to maximize the reliability of $\mathcal{VN}$s. This optimization problem is formulated as a Mixed Integer Linear Programming (MILP). K-Redundant solves the above problem based on two stages. First, the $\mathcal{VN}$ request is increased with redundant virtual routers and links attached to the selected routers. Note that this stage deals with only the critical virtual routers. In the second stage, the increased $\mathcal{VN}$ request is mapped in the $\mathcal{SN}$ with respect to the usage rate of resource. In order to minimize the total cost of the mapping, K-Redundant makes use of the backup share approach. Indeed, since the authors assumed that only one substrate router can fail at a time, backup paths belonging to different critical virtual routers can share the same bandwidth. Besides, the cross-share approach is adopted. It consists in sharing the original primary paths with the corresponding backup paths. The $\mathcal{VN}$ embedding process is based on D-ViNE [55] algorithm to find the working mapping for the original $\mathcal{VN}$ request. Then, K-Redundant solves the simplified MILP using CPLEX to embed the backup resources. These steps are presented in the pseudo-code Algorithm 2. The results show that if $K > 1$, the reliability increases whereas the router cost's ratio is higher than 1. Unfortunately, physical links

are assumed to remain operational at all times, which is not realistic. Besides, the rejection rate of $\mathcal{VN}$ requests is not evaluated.

Algorithm 2: `K-Redundant` [63]

1 Inputs: Virtual and substrate networks described by graphs, k value
2 Output: Working and backup mappings
3 Get the working mapping using `D-VINE` [55]
4 Establish an augmented reliable graph with backup node(s) basing on k value
5 Solve the reduced MILP mapping problem (backup) using CPLEX

2.2.2.2 Substrate link failures

In [64], the authors proposed a new Survivable Virtual Network Embedding algorithm denoted by `SVNE`. In fact, when a substrate link failure occurs, a fast re-routing strategy is executed by exploiting only the backup bandwidth allocated in all physical links. In other words, in order to protect against single substrate link failure, `SVNE` dedicates a certain rate of its bandwidth for a backup usage. `SVNE` heuristic is composed of three stages. First, `SVNE` proactively computes a set of possible backup detours for each substrate link. Then, `SVNE` embeds each new $\mathcal{VN}$ request by calling the $\mathcal{VN}$ embedding strategy `D-ViNE` [55]. Finally, when a link failure occurs, a reactive backup detour optimization solution is invoked. It reroutes the affected bandwidth along candidate backup detours selected in the first stage. To do so, the authors formulated the problem of the $\mathcal{VN}$ embedding problem and re-mapping of virtual links as linear programs. The main objective is to minimize the bandwidth consumption and the penalties due to link failures. The `SVNE` algorithm is summarized in the pseudo-code Algorithm 3. Based on simulations, `SVNE` outperforms the baseline strategies (i.e., reliability is not considered). Unfortunately, the succesful recovery rate of virtual links impacted by failures is not evaluated. Besides, the substrate router failures are not considered.

Algorithm 3: `SVNE` [64]

1 Compute detour paths for each substrate link
2 Modelize the mapping problem on MILP
3 Embed the $\mathcal{VN}$ basing on MILP resolution
4 Re-embed the impacted link using the pre-computed detours and basing on MILP resolution

In [65], the authors propounded `RMap` algorithm dealing with the mapping of $\mathcal{VN}$s by considering failures of substrate links. To maximize the resilience, `RMap` allocates backup links. To do so, first `RMap` embeds the $\mathcal{VN}$ request by embedding virtual routers then deals with virtual links. Next, the $\mathcal{SN}$ is formulated as a weighted graph by defining for each link its stress value. Note that

the latter quantifies the number of virtual links transiting through the substrate link. Afterwards, if a link stress is higher than a predefined threshold, RMap will compute its backup detour based on Loop Free Alternate resilience approach [66]. When a link failure occurs, virtual links transiting over this substrate link will migrate to the backup links. Hence, the offered service will not be interrupted. Based on simulations, RMap achieves 70% of virtual link protection. However, RMap ensures protection only for stressed links. Thus, if a failure occurs within a non-stressed link, all $\mathcal{VN}$s will be impacted. Furthermore, the rejection rate of $\mathcal{VN}$ requests is not evaluated. Finally, we notice that the calibration of the links' stress threshold has not been discussed.

2.2.2.3 Substrate router and link failures

In [67], the authors addressed the problem of $\mathcal{VN}$ embedding in the $\mathcal{SN}$. The authors assume that both substrate links and routers failures can occur in the $\mathcal{SN}$. Two strategies dealing with reliability of $\mathcal{VN}$s named i) Cluster and Path Protection (CPP) and ii) Virtual Network Protection (VNP) are proposed. With CPP, each logical connection (i.e., virtual link) is protected against substrate link failures by establishing two disjoint paths. Besides, two copies of the job (virtual router) are allocated to survive any single router (i.e., cluster) failure. With VNP, three disjoint paths and three copies of jobs are respectively allocated to survive against substrate links and routers failures. The task graph of a $\mathcal{VN}$ contains m connections and n tasks. Hence, to embed a virtual network, i) CPP needs a total of $2 \cdot n$ routers and $2 \cdot m$ pairs of disjoint paths, in contrast ii) $3 \cdot n$ virtual routers and $3 \cdot m$ connections are necessary with VNP.
CPP operates as following. First, for each unassigned virtual router, the compatible substrate router with minimum cost is selected. Then, virtual connections (primary and backup links) are mapped while maximizing the rate of sharing resources in order to minimize the mapping cost.
On the other hand, VNP computes for each $\mathcal{VN}$ three disjoint mappings. In fact, the embedding algorithm is similar to CPP but without considering backup links. VNP first instantiates the primary mapping. Then, all the physical resources (routers and links) used in the first step mapping are removed to deal with the second mapping (i.e., first backup). Afterwards, VNP computes the embedding of the second backup (i.e., third mapping), after removing all the resources used in the first and the second mapping for the same request. Consequently, VNP generates three disjoint $\mathcal{VN}$ mappings.
To evaluate the performance, two metrics are defined: i) $\mathcal{VN}$ blocking rate and ii) the average resource leasing cost. Simulation results show that VNP has a lower $\mathcal{VN}$ blocking rate than CPP when a sufficient computing resources and varying link capacities are assumed. Whereas, CPP has a lower $\mathcal{VN}$ blocking rate than VNP when a sufficient bandwidth and varying router capacities are assumed. Besides, the simulations illustrate that VNP achieves a lower average of job cost than CPP. Unfortunately, both CPP and VNP make use of a huge amount of network resources, which will deteriorate the rejection rate of $\mathcal{VN}$ requests and $\mathcal{CP}$'s benefit. Furthermore, we notice that

simulations consider only small-sized $\mathcal{VN}$ requests (3 routers), which is not realistic.

2.2.2.4 Regional failures

In [68] two survivable $\mathcal{VN}$ embedding algorithms have been proposed named Separate Optimisation with Unconstrained Mapping (SOUM) and Incremental Optimisation with Constrained Mapping (IOCM). The challenging point in [68] is the consideration of geographic region failures that can simultaneously affect a set of substrate resources (i.e., links and/or nodes). In general, a failure of a geographic region infers a simultaneous outage of nodes and links due to events such as natural disasters, etc. To do so, similar to [64], the problem is formulated as a mixed integer linear programming (MILP). The proposal embeds the $\mathcal{VN}$ request with respect to the survivability against any single regional failure as follows. First, the $\mathcal{VN}$ request is mapped depending on the adopted algorithm SOUM or IOCM. Then, the redundant nodes and links are allocated. Next, if the regional failure occurs then the virtual resources migrate to the backup resources. SOUM calculates the mapping of $\mathcal{VN}$ request regardless of any regional failures. Afterwards, for each regional failure it instantiates the backup resources in safe region(s). It is worth pointing out that the order of regional failures does not affect the mapping result. Unlike IOCM, the mapping depends on the order of dealing regional failures. The proposal has been evaluated and a comparison is performed between the SOUM and IOCM in terms of i) mapping cost, ii) average number of migrations and iii) recovery blocking probability. Unfortunately, the rejection rate of $\mathcal{VN}$s is not evaluated. Moreover, the authors did not compare the new approaches with baseline solutions (i.e., algorithms that do not consider reliability) basing on reject rate metric in order to evaluate the impact of allocating dedicated backup. Besides, the two proposed algorithms consume a huge amount of network capabilities to ensure survivability without an optimized strategy.

In [69], authors extend their work in [68] by first developing a Non-Survivable Virtual Infrastructure Mapping (NSVIM$\star$) heuristic. Based on NSVIM$\star$, they develop efficient SVIM heuristics namely Separate Optimization with Unconstrained Mapping and redundancy elimination (SOUM$\star$) and Incremental Optimization with Constrained Mapping and failure-avoidance (IOCM$\star$). In addition, they also develop a Mixed Integer Linear Programming (MILP) formulation to model the SVIM problem with the objective of minimizing the overall cost. In this paper, the authors assume that two (or more) regional failures can not simultaneously occur. Like [68], the authors ensure Virtual Infrastructure (VI), also called Virtual Network $\mathcal{VN}$, survivability against regional failures, so the inputs for SOUM$\star$ and IOCM$\star$ are: i) the substrate network, ii) $\mathcal{VN}$ request, iii) list of possible regional failures R and iv) the incoming requests. Similar to the NSVIM algorithm [68], the improved-NSVIM (NSVIM$\star$) approach, developed in this paper, satisfies the $\mathcal{VN}$ requests without any survivability requirement. Compared to NSVIM [68], NSVIM$\star$ uses an efficient $\mathcal{VN}$ node sorting strategy to map those $\mathcal{VN}$ nodes first that require large computing and bandwidth resources. Hereafter, we summarize the main stages of NSVIM$\star$:

- Sort the virtual routers by their degree.
- Choose a router with the highest degree.
- Find the set of substrate routers that can host the selected virtual router, if this set is empty the virtual request is rejected.
- Choose the substrate router with low cost.
- Ensure the connection between this router and its neighbour(s) which is (are) already mapped.
- Iterate this procedure (i.e., the aforementioned steps) until all virtual routers are mapped.

Similar to SOUM, SOUM$\star$ decomposes the SVIM problem into $|R| + 1$ separate NSVIM problems: i) one involving the initial pre-failure (i.e., working) mapping and ii) the others involving the after-failure (i.e., backup) mapping. The challenging idea with SOUM$\star$ comparing to SOUM is that SOUM$\star$ eliminates the redundancy mappings that cover the same regional failure by keeping the min-cost mapping. Let's take the example of two mappings $M1$ and $M2$. The first covers two regions $R1$ and $R2$. But $M2$ covers only $R1$. Besides, $M1$ cost is lower than $M2$ one. Accordingly, SOUM$\star$ keeps $M1$ and eliminates $M2$. At the end of SOUM$\star$ algorithm, the authors should have a min-cost set of mappings that covers whole regional failures. SOUM$\star$ can be summarized as below:

- Calculate the mapping for a specific regional failure. It also includes initial pre-failure (working) mapping.
- Calculate the cost of this mapping.
- Remove the redundant mapping(s) by keeping the low cost mapping.
- Iterate this procedure to all regional failures.

On the other hand, an optimized version of IOCM [68] denoted by IOCM$\star$ is proposed. The latter aims to minimize the additional computing and networking resources required to recover from physical failures. Similar to IOCM, the order in IOCM$\star$ impacts the whole mapping cost. IOCM$\star$ is based on new version of NSVIM$\star$ called Failed Avoidance NSVIM$\star$ (FA−NSVIM$\star$). It tries to avoid the mapping in a facility router within a regional failure. Hence, the backup number and the whole mapping cost will be reduced. Hereinafter, we summarize IOCM$\star$.

1. Consider k order of $|R|$ regional failures and pre-failure mapping (i.e., $k < (|R| + 1)!$).
2. Call FA−NSVIM$\star$ algorithm and get the initial mapping, which is affected the least by failures.

3. Basing on NSVIM$\star$, remap all substrate resources (i.e., links and routers) which are in the same specific regional failure, into other physical resources.

4. Iterate the previous step for all regional failures.

5. Calculate the cost of the current mapping (i.e., primary and backup resources).

6. Keep the current mapping if it is lower than its predecessor.

7. Consider another order (i.e., from the k order) and start from the second step.

To evaluate the performance of the two approaches, the authors use: i) small substrate network: 10 routers and 15 links, ii) large substrate network: 27 routers and 41 links. The computing capacity at facility routers and bandwidth capacity on the links follow a uniform distribution from 300 to 600 under unconstrained capacity and 50 to 150 when the capacity is constrained. They assume that the unit computing cost (e.g., using 1000 CPU hours) is uniformly distributed from 1 to 5 and unit bandwidth cost (e.g., for 1 Mbps) is from 5 to 10. The $\mathcal{VN}$ requests are generated randomly. The computing and bandwidth demands follow a uniform distribution from 10 to 30 and 10 to 50, respectively. For each regional failure they randomly choose various numbers of adjacent routers to fail (e.g., three routers). The authors used the following three performance metrics: i) the cost and ii) the mean number of migrations and iii) the recovery blocking probability, which is the ratio of the number of unrecoverable failure scenarios to the total number of failure scenarios. In order to evaluate the cost NSVIM$\star$, the authors compare it to NSVIM and MILP problem solved by CPLEX 8.0 but for a very small $\mathcal{VN}$ request. Results show that NSVIM$\star$ and NSVIM have a near-optimal (i.e., same cost as the MILP) performance for a small-size $\mathcal{VN}$ request. Furthermore NSVIM$\star$ achieves better performance than NSVIM when the size of the $\mathcal{VN}$ request increases. Besides, results show that SOUM$\star$ and IOCM$\star$ achieves a near-minimum cost and perform better than SOUM and IOCM for a small-size problem. For lager $\mathcal{VN}$ request, the proposed algorithms SOUM$\star$, IOCM$\star$ and IOCM have similar performance in terms of cost, and each one of them outperforms SOUM. Moreover, simulation proves the proposed algorithms IOCM$\star$ and IOCM lead to fewer average number of migrations than SOUM$\star$ and SOUM. Furthermore, results show that IOCM$\star$ leads to a fewer migrations than IOCM. This is explained by the use of FA-NSVIM$\star$ algorithm. Besides, results show that the SOUM$\star$ and SOUM lead to better recovery blocking probability than IOCM$\star$ and IOCM. We notice that the authors propounded their approaches basing on the assumption that only one regional failure fails at a time. But, this assumption is not realistic since many failures can simultaneously occur in different geographical regions. Furthermore, authors did not specify the path use between the primary and the backup node. Besides, they did not determine the period and bandwidth allocation for synchronization (i.e., sending snapshots). In fact, this period can cause communication degradation. On the other hand, authors did not evaluate the rejection rate. Indeed, this metric shows the approach efficiency to optimize the allocation of substrate resources.

In [70], the authors devise two kinds of algorithms for solving the Survivable Virtual Network Mapping (SVNM) problem efficiently: i) Lagrangian relaxation-based (LR−SVNM) algorithms including LR−SVNM−M and LR−SVNM−D and ii) Heuristic (H−SVNM) algorithms including H−SVNM−D and H−SVNM−M. Similar to [68], the authors propose survivable solutions to face regional failures: Shared Risk Group (SRG). The authors assume that only one region can fail at a time. These algorithms mainly aim to decompose the primal NP-hard problem into several sub-problems in order to reduce the computational complexity at the expense of increasing the total $\mathcal{VN}$ mapping cost. At the first step, the authors formulate the problem using mixed integer linear programming (MILP) approach. The MILP formulation is similar to [44]. The first propounded algorithm (i.e., H−SVNM) is an enhancement of SOUM⋆ [69]. In fact, the authors propose a modified version of D−ViNE algorithm [44], named D−ViNE⋆. Like SOUM⋆ [69], H−SVIM decompose the initial problem onto sub-problems basing on the number of regional failures, then it uses D−ViNE⋆ to deal with each mapping. This approach is called H−SVIM−D. Alternatively, the embedding of each sub-problem can be done by solving the MILP formulation if the problem is tractable. This algorithm is called H−SVIM−M. Like SOUM⋆, once getting the mapping of all sub-problems, H−SVIM−M or H−SVIM−D use the greedy min-cost set cover algorithm [69] to eliminate redundancies. On the other hand, the authors define new approach basing on Lagrangian relaxation method. This new proposal is named LR−SVNM. The main idea is to decompose the NP-hard problem into many sub-problems by relaxing certain constraints. Each one of these sub-problems may be solved using D−ViNE⋆ (i.e., LR−SVNM−D), or by MILP (i.e., LR−SVNM−M). To evaluate the performance of the propounded algorithms, the authors used 4 substrate network topologies in their simulations. All router and link bandwidth capacities are assumed to be 50 units. Virtual networks requests are generated randomly. The number of a $\mathcal{VN}$ routers is equal to a given number N and the average degree of connectivity of the $\mathcal{VN}$ request is about 2.5. They assume that each $\mathcal{VN}$ router requires 1 unit computing resource capacity and 1 unit of bandwidth resources by each of the communication demands between the $\mathcal{VN}$ nodes. MILP problem is solved by the CPLEX solver. The authors compared IOCM⋆, SOUM⋆, LR−SVNM−D, LR−SVNM−M, H−SVNM−D, H−SVNM−M and MILP. The performance metrics are: i) Total Mapping Cost and ii) Time Efficiency which is the time consumed by the algorithm to map a $\mathcal{VN}$. The simulation results show that the proposed algorithms have mapping cost lower than IOCM⋆ and SOUM⋆. In small topologies, curves show that LR−SVNM−M and H−SVNM−M outperform LR−SVNM−D and H−SVNM−D, respectively. Besides, H−SVNM−M is better than LR−SVNM−M in term of mapping cost. On the other hand, IOCM⋆ and SOUM⋆ are considerably better than the proposed algorithms in term of time efficiency. Moreover, H−SVNM is better than the Lagrangian relaxation-based algorithm LR−SVNM in terms of computational time. We notice that the authors propound their approaches basing on the assumption that only one regional failure fails at a time. But, this assumption is not realistic since many failures can occur in different geographical regions simultaneously (e.g., earthquake). We notice, also, that the authors did not detail

the synchronization procedure. Furthermore, the authors use a modified version of D-ViNE [44], but this algorithm has two parameters constraining the mapping process which are location and distance. Moreover, simulation results show that LR-SVNM-M and H-SVNM-M spend so much time to compute the $\mathcal{VN}$ embedding for networks having more than 10 routers. Accordingly, they are not scalable.

2.2.3 Summary

Table 2.1 summarizes the main survivable virtual network embedding algorithms found in the existing literature. Unfortunately, all of them are online algorithms and to the best of our knowledge no batch strategy has been proposed to deal with the reliable embedding of $\mathcal{VN}$ requests.

Table 2.1: Overview of reliable $\mathcal{VN}$ embedding strategies

Algorithm	Strategy	Protection	Link Mapping	Backup	Recovery	Preventive	Proactive
ORP [60] [61]	Centralized	Router	Splittable	Yes	Yes	No	Yes
K-Redundant [63]	Centralized	Router	Splittable	Yes	Yes	No	Yes
RMap [65]	Centralized	Link	Unsplittable	Yes	Yes	No	Yes
SVNE [64]	Centralized	Link	Splittable	Yes	Yes	No	Yes
CPP, VNP [67]	Centralized	Router and link	Unsplittable	Yes	Yes	No	Yes
SOUM, IOCM [68]	Centralized	Regional	Unsplittable	Yes	Yes	No	Yes
SOUM⋆ [69]	Centralized	Regional	Unsplittable	Yes	Yes	No	Yes
IOCM⋆ [69]	Centralized	Regional	Unsplittable	Yes	Yes	Yes	Yes
LR-SVNM-M, LR-SVNM-D [70]	Centralized	Regional	Splittable	Yes	Yes	No	Yes
H-SVNM-D, H-SVNM-M [70]	Centralized	Regional	Splittable	Yes	Yes	No	Yes
DFTE [56]	Distributed	Router and link	Splittable	No	Yes	No	No

2.3 Conclusion

This chapter summarized the most prominent virtual network mapping strategies which do not consider any fault-tolerant mechanism. Later on, we presented an overview of the survivable virtual network embedding algorithms found in the literature. It is worth pointing out that all the proposed

approaches sequentially process the $\mathcal{VN}$ requests as soon as they arrive. Thus, we conclude that no survivable research work has been investigated to simultaneously deal with the mapping of a set (i.e., batch) of $\mathcal{VN}$s. In the next chapter, we will present our first contribution to tackle the reliable online $\mathcal{VN}$ embedding problem by proposing a preventive approach.

Chapter 3

Preventive reliable virtual network embedding

Contents

3.1 Problem statement 52
3.1.1 Virtual and substrate network models 52
3.1.2 Formalization of $\mathcal{VN}$ preventive reliable embedding problem 53
3.2 Proposal: `Advanced-PR-VNE` 56
3.2.1 Embedding of virtual access routers 56
3.2.2 Erecting of solution components 57
3.2.3 Reliable embedding of solution component $\mathcal{SC}_i$ 57
3.3 Performance evaluation 62
3.3.1 Simulation environment 62
3.3.2 Performance metrics 63
3.3.3 Simulation results 64
3.4 Conclusion 73

In this chapter, we address the resource allocation problem within Cloud's backbone network by considering the equipments (i.e., routers and links) failures in the substrate network. Our objective is to increase the revenue of the Cloud provider by i) maximizing the acceptance rate of virtual networks and ii) minimizing the penalties caused by the hardware crashes. This optimization problem is i) multi-objective, ii) non-linear and iii) NP-hard. To cope with this complexity, we propose a new Advanced Preventive Reliable Virtual Network Embedding strategy denoted by `Advanced-PR-VNE`. It is based on the artificial bee colony meta-heuristic and K-Means classification algorithm. Note that the proposal does not allocate any backup resources but promotes the use of reliable substrate resources. Two variants of `Advanced-PR-VNE` are defined according to

the mapping scheme of virtual links. The first one, denoted by `Advanced-PR-VNE-unsplittable`, consists in embedding a virtual link in one substrate path. The second variant, denoted by `Advanced-PR-VNE-splittable`, distributes the required bandwidth of a virtual link among a set of substrate paths. Based on extensive simulations, `Advanced-PR-VNE` outperforms the related reliable virtual network embedding strategies in terms of i) rejection rate of virtual networks, ii) rate of virtual networks impacted by physical failures and iii) Cloud provider's revenue. This chapter is organized as follows. Section 3.1 will formulate the virtual and substrate network models as well as the reliable $\mathcal{VN}$ mapping problem. Afterwards, Section 3.2 will describe the details of our proposal `Advanced-PR-VNE`. The performance evaluation will be exposed in Section 3.3. Finally, Section 3.4 will conclude the chapter.

3.1 Problem statement

In this section, we will formulate the $\mathcal{VN}$ reliable embedding problem. To do so, first we will describe the virtual and substrate network models. Then, we will formulate the $\mathcal{VN}$ survivable mapping optimization problem.

3.1.1 Virtual and substrate network models

A $\mathcal{VN}$ request is formulated as an undirected graph, denoted by $\mathcal{D} = (V(\mathcal{D}), E(\mathcal{D}))$ where $V(\mathcal{D})$ and $E(\mathcal{D})$ are respectively the sets of virtual routers and links. Each virtual router, $v \in V(\mathcal{D})$, is characterized by its i) requested processing power $B(v)$ and memory $M(v)$ and ii) its type: access or core $X(v)$. Note that if $X(v) = 1$, then v is an access virtual router. Otherwise, v is a core virtual router. Moreover, each virtual access router v is typified by a geographical zone denoted by $\mathcal{Z}_v$. It delimits the area in which v can be mapped. Furthermore, each virtual link $d \in E(\mathcal{D})$ is characterized by its required bandwidth $C(d)$.

The $\mathcal{SN}$ is formulated as an undirected graph, denoted by $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ where $V(\mathcal{G})$ and $E(\mathcal{G})$ are respectively the sets of physical routers and links. Each physical router, $w \in V(\mathcal{G})$, is characterized by its i) residual processing power $B(w)$, ii) remaining memory capacity $M(w)$, iii) type: access or core $X(w)$, and iv) reliability $\mathcal{R}^n(w,t)$ at time t. Note that if $X(w) = 1$, then w is an access substrate router. Otherwise, w is a core substrate router. Besides, the initial capacity in terms of processing power and memory are respectively denoted by $\hat{B}(w)$ and $\hat{M}(w)$. Each substrate link, $e \in E(\mathcal{G})$, is typified by its i) remaining bandwidth $C(e)$, ii) initial bandwidth capacity $\hat{C}(e)$ and iii) reliability $\mathcal{R}^l(e,t)$ at time t.

As in [71], we model the Mean Time Between Failures (MTBF) in the $\mathcal{SN}$ as a Weibull distribution. Consequently, we can estimate and predict the equipment's reliability at any time. The

Cumulative Distribution Function (CDF) of the MTBF is expressed as:

$$F(x) = 1 - \exp(-\left(\frac{x}{a}\right)^{b}),\ x \geq 0 \tag{3.1.1}$$

where a and b are the parameters of the Weibull distribution. It is worth noting that we assume heterogeneous ages in the $\mathcal{SN}$. Thus, each physical equipment in $\mathcal{G}$ has its own initial age denoted by $A(w)$ and $A(e)$ respectively for substrate router w and substrate link e. Accordingly, we classify the equipments within the substrate network into three groups: i) **young**, ii) **adult** and iii) **old**. So, at any instant t, we can evaluate the substrate resource's reliability. Indeed, the reliability of a i) substrate router w is equal to $\mathcal{R}^n(w,t) = F(A(w)+t)$ and ii) substrate link e is equal to $\mathcal{R}^l(e,t) = F(A(e)+t)$. It is noteworthy that the substrate resource's reliability is inversely proportional to its age. In other words, the probability of physical failure increases with time.

3.1.2 Formalization of $\mathcal{VN}$ preventive reliable embedding problem

As described above, the substrate resources in $\mathcal{G}$ are limited. So, the $\mathcal{SN}$ is unable to host an infinite number of $\mathcal{VN}$ requests. Accordingly, a smart and judicious $\mathcal{VN}$ mapping strategy in $\mathcal{G}$ is essential in order to i) maximize the acceptance rate of $\mathcal{VN}$ requests hence the $\mathcal{CP}$'s turnover grows up and ii) minimize the blackout rate (i.e., the rate of $\mathcal{VN}$ impacted by physical failures) hence $\mathcal{CP}$'s penalty grows down. Consequently, the $\mathcal{CP}$'s revenue is maximized by accepting more clients and paying less penalties.

The maximization of the acceptance rate of $\mathcal{VN}$ requests can be achieved by i) minimizing the amount of resources allocated for each $\mathcal{VN}$ request and ii) maximizing the load balancing of the resource usage rate within the $\mathcal{SN}$. To reach the above objectives, the residual and the standard deviation of residual resources should be respectively maximized and minimized. Formally,

$$\begin{gathered}\textbf{maximize}[\min\{C(e)\}, \min\{B(w)\}, \min\{M(w)\}] \\ \textbf{minimize}[std\{C(e)\}, std\{B(w)\}, std\{M(w)\}] \\ e \in E(\mathcal{G}),\ \ w \in V(\mathcal{G})\end{gathered} \tag{3.1.2}$$

Moreover, our objective consists in mapping the virtual **routers** and **links** within the substrate resources offering the highest reliability during the lifetime of each $\mathcal{VN}$.
For each virtual router $v \in V(\mathcal{D})$ in a $\mathcal{VN}$, the reliability objective is expressed as following:

$$\textbf{maximize}\{\mathcal{R}^n(w^v, T_{w^v})\}, w^v \in V(\mathcal{G}) \tag{3.1.3}$$

where w^v denotes the substrate router hosting v and T_{w^v} is the age of w^v when the virtual network $\mathcal{D}$ leaves the physical backbone network $\mathcal{G}$.

Concerning the virtual links' reliability, it will depend on the virtual link mapping approach. Formally, the following objective should be optimized:

- **Unsplittable virtual link embedding**: For each virtual link $d \in E(\mathcal{D})$,

$$\textbf{minimize}\Big\{ \prod_{e^d \in \mathcal{P}} \left(1 - \mathcal{R}^l\left(e^d, T_{e^d}\right)\right) \cdot \prod_{e^d \in \mathcal{P}} \left(1 - \frac{C\left(e^d\right)}{\hat{C}\left(e^d\right)}\right) \cdot \prod_{w^d \in \mathcal{P}} \left(1 - \mathcal{R}^n\left(w^d, T_{w^d}\right)\right) \Big\}, \mathcal{P} \in \textit{Paths}(d) \tag{3.1.4}$$

  where $\textit{Paths}(d)$ denotes the set of substrate paths which can host the virtual link d, $\{e^d\}$ and $\{w^d\}$ denote the set of substrate links and routers forming the substrate path $\mathcal{P} \in \textit{Paths}(d)$. As defined above, T_{w^d} and T_{e^d} are respectively the ages of the physical router w^d and link e^d when the virtual network $\mathcal{D}$ leaves the backbone network $\mathcal{G}$.
  Note that the above cost function quantifies the reliability provided by substrate path's equipments (i.e., routers and links) and residual bandwidth.

- **Splittable virtual link embedding**: For all virtual links (i.e., $E(\mathcal{D})$),

$$\textbf{minimize} \quad \Big\{ \sum_{d \in E(\mathcal{D})} \sum_{\mathcal{P} \in \textit{Paths-all}(d)} \left[b\left(\mathcal{P}, d\right) \cdot \textit{length}\left(\mathcal{P}\right) \cdot \left(1 - \mathcal{R}^p\left(\mathcal{P}\right)\right) \cdot \left(1 - \bar{C}\left(\mathcal{P}\right)\right)\right] \Big\}, \mathcal{P} \in \textit{Paths-all}(d) \tag{3.1.5}$$

  where i) $\textit{Paths-all}(d)$ is the set of all substrate paths connecting the substrate routers hosting the virtual link d's extremities, ii) $\textit{length}\left(\mathcal{P}\right)$ is the number of hops of $\mathcal{P}$, iii) $b\left(\mathcal{P}, d\right)$ is the allocated bandwidth for virtual link d in path $\mathcal{P}$, iv) $\mathcal{R}^p\left(\mathcal{P}\right)$ corresponds to the lowest reliability value of the equipments (router and link) forming the path $\mathcal{P}$, and v) $\bar{C}\left(\mathcal{P}\right)$ corresponds to the lowest usage rate value of the substrate links forming the path $\mathcal{P}$. Note that unlike the unsplittable-based approach, splittable virtual link embedding handles all the virtual links of a $\mathcal{VN}$ at once.
  Note that the above cost function quantifies i) the reliability provided by substrate path's equipments (i.e., routers and links), ii) residual bandwidth and iii) allocated bandwidth in the selected substrate paths.

It should be highlighted that the $\mathcal{VN}$ embedding process is a constrained problem. Hereafter, we will detail the main constraints.

Mapping of virtual routers is constrained so that for each $\mathcal{VN}$ request, $\mathcal{D}$, two virtual routers cannot be assigned to the same substrate router. Formally,

$$\sum_{v \in V(w)} x_{vw} \leq 1, \quad \forall w \in V\left(\mathcal{G}\right) \tag{3.1.6}$$

where x_{vw} is a binary variable indicating whether virtual router $v \in V\left(\mathcal{D}\right)$ is assigned to physical router $w \in V\left(\mathcal{G}\right)$, $V\left(w\right) \subseteq V\left(\mathcal{D}\right)$ denotes the set of virtual routers that can be embedded in physical router $w \in V\left(\mathcal{G}\right)$.

In addition, each virtual router must be assigned to only one physical router. Formally,

$$\sum_{w \in W(v)} x_{vw} = 1, \quad \forall v \in V(\mathcal{D}) \tag{3.1.7}$$

$W(v) \subseteq V(\mathcal{G})$ denotes the set of potential substrate routers for hosting virtual node $v \in V(\mathcal{D})$ taking into account its requirements such as: processing power $B(v)$, memory $M(v)$, geographical location $\mathcal{L}(v)$, etc. Virtual router, $v \in V(\mathcal{D})$, can be mapped in the substrate router, $w \in V(\mathcal{G})$, if i) the available residual resources (i.e., $B(w)$ and $M(w)$) are at least equal to those required (i.e., $B(v)$, $M(v)$) and ii) v has the same type (i.e., access or core) as w. Formally,

$$\forall v \in V(w) \begin{cases} (B(w) - B(v))\, x_{vw} & \geq 0 \\ (M(w) - M(v))\, x_{vw} & \geq 0 \\ (X(w) - X(v))\, x_{vw} & = 0 \end{cases} \tag{3.1.8}$$

Each physical link, $e \in E(\mathcal{G})$ cannot host more than its capacity. Formally,

$$C(e) \geq 0, \forall e \in E(\mathcal{G}) \tag{3.1.9}$$

We will now outline our $\mathcal{VN}$ reliable embedding optimization problem.

$$\begin{array}{ll}
\texttt{maximize} & \min\{C(e)\}, e \in E(\mathcal{G}) \\
\texttt{maximize} & \min\{B(w)\}, w \in V(\mathcal{G}) \\
\texttt{maximize} & \min\{M(w)\}, w \in V(\mathcal{G}) \\
\texttt{maximize} & \mathcal{R}^n(w^v, T_{w^v})\}, w^v \in V(\mathcal{G}), w^v \in V(\mathcal{G}) \\
\multicolumn{2}{l}{\texttt{Unsplittable-based virtual link mapping}} \\
\texttt{maximize} & \Big\{ \prod_{e^d \in \mathcal{P}} \mathcal{R}^l(e^d, T_{e^d}) \cdot \prod_{e^d \in \mathcal{P}} \frac{C(e^d)}{\hat{C}(e^d)} \cdot \\
 & \prod_{w^d \in \mathcal{P}} \mathcal{R}^n(w^d, T_{w^d}) \Big\}, \mathcal{P} \in \texttt{Paths}(d) \\
\multicolumn{2}{l}{\texttt{Splittable-based virtual link mapping}} \\
\texttt{minimize} & \Big\{ \sum_{d \in E(\mathcal{D})} \sum_{\mathcal{P} \in \texttt{Paths-all}(d)} [b(\mathcal{P}, d) \cdot \texttt{length}(\mathcal{P}) \cdot \\
 & (1 - \mathcal{R}^p(\mathcal{P})) \cdot (1 - \bar{C}(\mathcal{P}))] \Big\}, \mathcal{P} \in \texttt{Paths-all}(d)
\end{array}$$

$$\begin{array}{ll}
\texttt{subject to:} & \sum_{v \in V(w)} x_{vw} \leq 1, \quad \forall w \in V(\mathcal{G}) \\
 & \sum_{w \in W(v)} x_{vw} = 1, \quad \forall v \in V(\mathcal{D}) \\
 & (B(w) - B(v))\, x_{vw} \geq 0 \\
 & (M(w) - M(v))\, x_{vw} \geq 0 \\
 & (X(w) - X(v))\, x_{vw} = 0 \\
 & C(e) \geq 0, \forall e \in E(\mathcal{G}) \\
 & x_{vw} : binary
\end{array}$$

Problem 1: Preventive reliable $\mathcal{VN}$ embedding problem

The above problem is a non-linear and multi-objective combinatorial optimization. It has been proved in [72] [73] that it is NP-hard problem. Accordingly, the optimal solution cannot be computed within polynomial time for a large-scale $\mathcal{SN}$. To skirt this complexity, we propose a new preventive survivable $\mathcal{VN}$ embedding algorithm denoted by Advanced-PR-VNE. It is based on the artificial bee colony meta-heuristic and classification algorithm. In the next section, we will detail this proposal.

3.2 Proposal: Advanced-PR-VNE

In this section, we describe our proposal named **Advanced Preventive Reliable Virtual Network Embedding** (Advanced-PR-VNE). Hereafter, we will enumerate the main Advanced-PR-VNE's stages: i) *Embedding of virtual access routers*, ii) *Erecting of solution components* and iii) *Reliable embedding of solution component* based on artificial bee colony metaheuristic and clustering algorithm focused on the classification and the analysis of multivariable observations. Note that our proposal adopts the **"divide and conquer"** approach and does not allocate any backup resources in order to maximize reliability.

3.2.1 Embedding of virtual access routers

The rate of access routers in the $\mathcal{SN}$ is tiny compared to core routers. Besides, each access router is constrained by its geographic position. In fact, an access virtual router should be mapped in the nearest physical access router to end-users exploiting the virtual network in order to minimize latency and to increase bandwidth. For this reason, we propose to first embed virtual access routers and their connecting links (i.e., virtual links connecting two virtual access routers). To do so, for each $v \in V(\mathcal{D})$ and $X(v) = 1$ (i.e., access router), Advanced-PR-VNE selects within the geographical zone $\mathcal{Z}_v$ all the physical access routers $w \in V(\mathcal{G})$ satisfying the required resources denoted by $\mathcal{N}_v$. Formally,

$$\begin{aligned} \forall w \in \mathcal{N}_v, \quad & B(w) \geq B(v) \\ & M(w) \geq M(v) \\ & X(w) = X(v) = 1 \end{aligned} \tag{3.2.10}$$

Then, the virtual router v is mapped within $w^v \in \mathcal{N}_v$ offering the highest reliability. Formally,

$$\forall w \in \mathcal{N}_v, \quad \mathcal{R}^n(w^v, T_{w^v}) \geq \mathcal{R}^n(w, T_w) \tag{3.2.11}$$

where T_w is the age of router w when the $\mathcal{VN}$ of v leaves the $\mathcal{SN}$.

Afterwards, the virtual links $d \in E(\mathcal{D})$ connecting the virtual access routers are mapped. The details of this process will be exposed in Section 3.2.3.2. In fact, the virtual link mapping will be based on i) Dijkstra algorithm if an unsplittable approach is assumed or ii) Mixed Integer Linear Programming (MILP) resolution if a splittable apporach is considered.

Algorithm 4: $\mathcal{SC}_i$ mapping process

1 Send *scout* bees to locate candidate routers of $\mathcal{SC}_i$'s centre router
2 *Employed* bees explore and evaluate the nectar of candidate routers
3 The *Onlooker* bee selects the best *employed* bee's solution

3.2.2 Erecting of solution components

Once the access routers and their connecting virtual links have been mapped, the remaining virtual topology will be denoted by $\tilde{\mathcal{D}}$. We notice that some virtual links have one of their two extremities (i.e., virtual router) already mapped. We call this kind of virtual links **hanging** links.

`Advanced-PR-VNE` subdivides the virtual topology $\tilde{\mathcal{D}}$ into elementary star topologies named Solution Components $\{\mathcal{SC}_i\}$. Note that each $\mathcal{SC}_i$ is composed of one single virtual core router and all its attached hanging links. To generate the set of solutions components, `Advanced-PR-VNE` selects the virtual router having the highest number of hanging links to be extracted from $\tilde{\mathcal{D}}$ in order to build the current $\mathcal{SC}_i$. Then $\tilde{\mathcal{D}} \leftarrow \tilde{\mathcal{D}} \backslash \mathcal{SC}_i$ and the process is repeated until $\tilde{\mathcal{D}} = \emptyset$.

3.2.3 Reliable embedding of solution component $\mathcal{SC}_i$

As explained in the above section, each solution component $\mathcal{SC}_i$ contains one virtual core router denoted by v^i and a set of hanging links denoted by $\{d^i_j\}$. In order to map $\mathcal{SC}_i$ in the $\mathcal{SN}$, we propose to make use of the artificial bee colony metaheuristic [37] [38]. The latter is based on three types of artificial bees: i) *scout*, ii) *employed* and iii) *onlooker*. The metaheuristic operates as follows. First scout bees will search the potential candidate solutions (i.e., food source). Then, worker bees will evaluate for each candidate solution its fitness (i.e., nectar volume) and communicate the information to the *onlooker* bee. Finally, with respect to the quality of each candidate, the onlooker bee will orient the worker bees to the best food source. Note that onlooker bee centralizes all the information and makes a decision. In other words, onlooker bee has a global view. Algorithm 4 summarizes the pseudo algorithm of $\mathcal{SC}_i$ mapping process. Hereafter, we will present the process of each type of bee in order to solve the reliable embedding of $\mathcal{SC}_i$.

3.2.3.1 Job of scout bees

The main role of scout bees consists in selecting the eligible and relevant set of candidates (i.e., core routers) in the $\mathcal{SN}$ which can embed the $\mathcal{SC}_i$'s centre virtual core router denoted by v^i. In fact, scout bees are responsible for the exploration stage preceding the exploitation stage performed by employed bees. Scout bees' process is based on a clustering algorithm focused on the classification and the analysis of multivariable observations. Indeed, clustering algorithms are defined as an efficient tool considering the internal homogeneity and the external separation. As defined in [74]: "patterns in the same cluster should be similar to each other, while patterns in different clusters should not".

To build clusters of substrate routers, we make use of K-Means clustering algorithm [75] to group the physical routers typified by similar properties in the same cluster. The choice of K-Means clustering algorithm is motivated by the fact that it is one of the most suitable clustering approaches in terms of time convergence. Indeed, it has been proved in [74] [76] that the time complexity is polynomial. We recall that we study the online preventive reliable $\mathcal{VN}$ embedding problem in which the performance of the proposal strongly depends on the time convergence. K-means operates as following. First, K observations are randomly selected which will be considered as the centroid of K clusters. Then, for each observation, the euclidian distance is calculated to each centroid. Afterwards, each observation will join the cluster having the closest centroid in terms of the latter euclidian distance which is calculated with respect to the virtual coordinates of each observation. Next, the centroid of each cluster is updated by calculating the barycenter of the cluster. Note that the centroid can be a fictitious observation. The same process is repeated until the steady state of clusters. We recall that K and the observations are the inputs of the algorithm. On the other hand, the generated clusters are the outputs.

In our case, each core router in the $\mathcal{SN}$ models an observation. K-Means algorithm is based on the euclidian distance in order to partition the observations. It is worth noting that the euclidian distance between two substrate routers is virtual. To calculate it, we define virtual coordinates for each substrate router $w \in V(\mathcal{G})$. The defined virtual coordinates for physical router w should match the main parameters impacting the reliable embedding of $\mathcal{SC}_i$ such as: i) w's reliability, ii) w's residual resources (i.e., processing power and memory), and iii) the embedding cost of $\mathcal{SC}_i$'s hanging links by assuming v^i is hosted in w. Formally, each substrate core router $w \in V(\mathcal{G})$ is characterized by virtual coordinates in $\Re^5$: i) $B(w)$, ii) $M(w)$, iii) $\mathcal{R}^n(w,t)$, iv) W-HL-Cost(w), and v) Std-HL-Cost(w).
Note that i) W-HL-Cost(w) and Std-HL-Cost(w) are respectively the Worst and Standard deviation of $\mathcal{SC}_i$'s virtual Hanging Links embedding Cost by assuming v^i is hosted in w. It should be highlighted that the virtual hanging link embedding cost is equal to path's cost calculated by Dijkstra algorithm based on defined metric in equation (3.1.4). K-Means clustering algorithm classifies the substrate core routers with respect to the above virtual coordinates in $\Re^5$ taking into account their relevance in terms of available resources and reliability.

Let be $V^c(\mathcal{G})$, $V^K(\mathcal{G})$ and $\{Clus_i\}$, $i = 1 \cdots K$, respectively the set of core substrate routers, set of clusters' centroid and clusters. K-Means randomly selects K core routers in $V^c(\mathcal{G})$ which are considered as clusters' centroid. Then, the virtual euclidian distance $||w\psi||^5$ is calculated from each substrate router $w \in V^c(\mathcal{G}) \setminus V^k(\mathcal{G})$ with each centroid $\psi \in V^K(\mathcal{G})$. Afterwards, each substrate core router $w \in V^c(\mathcal{G}) \setminus V^k(\mathcal{G})$ will join the cluster having the closest centroid in terms of the calculated virtual euclidian distance. Formally,

$$Clus_i = \{w : ||w\psi_i||^5 = \min_{j \in 1,\cdots K} ||w\psi_j||^5\} \tag{3.2.12}$$

$w \in V^c(\mathcal{G})$ and ψ_i is the centroid of $Clus_i$. Next, the centroid of each cluster is updated and the same process is repeated until convergence.

Once the set of clusters $\{Clus_i\}$ is generated, the best one, denoted by $Clus_s$, is selected based on the performance of the final set of centroids $\{\psi_i\}$ calculated by `K-Means` algorithm. We recall that each centroid ψ_i is defined by the coordinates in $\Re^5$. We propose to consider each centroid ψ_i as a multi-objective observation. In order to select the most relevant, we calculate the set of pareto-optimal centroids. In a multi-objective optimization problem, the usual meaning of the "optimum solution" cannot be applied, given that a solution optimizing all the objectives does not generally exist. Resolving the problem involves seeking a feasible solution that yields the best compromise among the given objectives, chosen from a set of efficient, Pareto-optimal solutions. The identification of the best solution among the Pareto-optimal solutions must be based on the **preferences expressed by the user** (decision maker). To do that, first we remove the dominated centroid observations. Then, we select only one based on the ordered priority of coordinates defined hereafter:

1. W-HL-Cost$(\psi_i) \rightarrow$ minimization
2. $\mathcal{R}^n(\psi_i) \rightarrow$ maximization
3. $\min[B(\psi_i), M(\psi_i)] \rightarrow$ maximization
4. Std-HL-Cost$(\psi_i) \rightarrow$ minimization

It is worth noting that the above factious coordinates are calculated based on the barycenter of observations in each cluster.

Finally, all the substate core routers in $Clus_s$ which are not able to host $\mathcal{SC}_i$'s centre virtual router (v^i) will be removed.

The pseudo algorithm of scout bees is summarized in Algorithm 5. It is straightforward to see that the complexity of `K-Means` algorithm is polynomial [77] and equal to $O(5 \cdot K \cdot |V^c(\mathcal{G})|)$. It is worth pointing out that in the basic version of this chapter denoted by `Basic-PR-VNE`[1], $Clus_s$ is equal to the set of substrate routers located at $\mathcal{H}$-hops from the geographic barycenter of the substrate routers hosting the $\mathcal{SC}_i$ hanging links extremities. As we can see, it is a basic variant and it does not consider the reliability and resource performances. In fact, the shortest geographic distance does not reflect the relevance of candidates in the context of reliable virtual network embedding.

3.2.3.2 Job of worker bees

Once the candidate substrate routers are generated $Clus_s$, worker bees will simulate the mapping of $\mathcal{SC}_i$ by considering all substrate core routers $w \in Clus_s$. To do so, we propose two variants based

[1] Published in IEEE GLOBECOM 2013

Algorithm 5: Scout bee pseudo algorithm

Inputs: $V^c(\mathcal{G})$ and their virtual coordinates in $\Re^5$, K
Output: $\{Clus_i\}$
for $i \leftarrow 1$ *to* K **do**
    $\psi_i \leftarrow$ select randomly in $V^c(\mathcal{G})$
repeat
    for $i \leftarrow 1$ *to* K **do**
        $Clus_i \leftarrow \emptyset$
    $Clus_i = \{w : ||w\psi_i||^5 = \min_{j \in 1,\cdots K} ||w\psi_j||^5\}, w \in V^c(\mathcal{G}) \setminus V^k(\mathcal{G})$
    for $i \leftarrow 1$ *to* K **do**
        $\psi_i \leftarrow Barycentre(Clus_i)$
until *Steady state* ;
$Clus_s$ is selected
Remove non-eligible substate core routers in $Clus_s$

on the virtual link approach. The first one is denoted by `Advanced-PR-VNE-unsplittable`. It consists in embedding each virtual link in one unsplittable path in the $\mathcal{SN}$. The second variant is denoted by `Advanced-PR-VNE-splittable`. It maps each virtual link in a set of substrate paths. Hereafter, we will detail both variants.

1. `Advanced-PR-VNE-unsplittable`

   It makes use of Dijkstra algorithm to compute the shortest path with respect to the defined cost metric in equation (3.1.4). In fact, based on this latter, Dijkstra will favor the use of reliable resources in the $\mathcal{SN}$. Besides, the congestion of substrate links will be minimized since our proposal promotes the selection of substrate links having the highest amount of residual bandwidth[2]. It is noteworthy that considering resources reliability aims to avoid non-reliable equipments and so to prevent failures impact. Worker bees will evaluate the $\mathcal{SC}_i$'s mapping cost in each candidate router $w^j \in Clus_s$ denoted by η_i^j. Formally, it is equal to:

$$\eta_i^j = \left(1 - \mathcal{R}^n(w^j, T_{w^j})\right) \cdot \left(1 - \frac{B(w^j)}{\hat{B}(w^j)}\right) \cdot \left(1 - \frac{M(w^j)}{\hat{M}(w^j)}\right) + \text{Sum-HL-Cost}(w^j) \tag{3.2.13}$$

   Note that Sum-HL-Cost(w^j) is the Sum of $\mathcal{SC}_i$'s virtual Hanging Links embedding Cost by assuming v^i is hosted in w^j. We recall that T_{w^j} is the age of the physical router w^j when the virtual network $\mathcal{D}$ leaves the backbone network $\mathcal{G}$.

   Then, the mapping costs, $\{\eta_i^j\}, j = 1 \cdots |Clus_s|$, will be communicated to the onlooker bee.

[2] `Basic-PR-VNE` makes use of the multi-commodity flow algorithm [78] in order to load balance the resource usage rate in the $\mathcal{SN}$.

2. `Advanced-PR-VNE-splittable`

   It embeds each virtual link in splittable substrate paths. Thanks to the share load of each virtual link among many substrate paths, the splittable link mapping improves the acceptance rate of $\mathcal{VN}$s and hence $\mathcal{CP}$'s turnover. We formulate the splittable virtual link embedding problem as a Mixed Integer Linear Programming (MILP). Based on this model, the mapping of all $\mathcal{SC}_i$'s hanging virtual links is solved at the same time.

   As explained in Section 3.1.2, the objective is to minimize the cost function defined in equation (3.1.5) applied only on the virtual hanging links of $\mathcal{SC}_i$ (i.e., do not consider all virtual links $E(\mathcal{D})$). Moreover, the solution must respect the capacity constraint to avoid the violation of substrate links' capacity. Formally,

$$\sum_{d\in\{d_j^i\}} \sum_{\mathcal{P}\in \textit{Paths-all}(d)} \left[\delta_e(\mathcal{P}) \cdot b(\mathcal{P},d)\right] \leq C(e), \forall e \in E(\mathcal{G}) \tag{3.2.14}$$

   where $\delta_e(\mathcal{P})$ is the link-path indicator binary variable. $\delta_e(\mathcal{P}) = 1$ if the substrate link e belongs to the substrate path $\mathcal{P}$. Otherwise, $\delta_e(\mathcal{P}) = 0$. We recall that i) $\{d_j^i\}$ is the set of $\mathcal{SC}_i$'s virtual hanging links and ii) $b(\mathcal{P},d)$ is the allocated bandwidth for virtual link d in path $\mathcal{P}$.

   Finally, the solution must respect the requested bandwidth of hanging virtual links. In fact, the total allocated bandwidth assigned to a virtual link d among all the candidate substrate paths must be equal to the requested one. Formally,

$$\sum_{\mathcal{P}\in \textit{Paths-all}(d)} b(\mathcal{P},d) = C(d), \quad \forall d \in \{d_j^i\} \tag{3.2.15}$$

   Hereafter, we will summarize the MILP optimization problem of $\mathcal{SC}_i$ reliable embedding.

$$\texttt{minimize} \quad \Big\{ \sum_{d\in\{d_j^i\}} \sum_{\mathcal{P}\in \texttt{Paths-all}(d)} \big[b(\mathcal{P},d) \cdot \texttt{length}(\mathcal{P}) \cdot (1-\mathcal{R}^p(\mathcal{P})) \cdot \left(1-\bar{C}(\mathcal{P})\right)\big] \Big\}, \mathcal{P} \in \texttt{Paths-all}(d)$$

$$\texttt{subject to:}$$

$$\sum_{d\in\{d_j^i\}} \sum_{\mathcal{P}\in \texttt{Paths-all}(d)} \left[\delta_e(\mathcal{P}) \cdot b(\mathcal{P},d)\right] \leq C(e), \forall e \in E(\mathcal{G})$$

$$\sum_{\mathcal{P}\in \texttt{Paths-all}(d)} b(\mathcal{P},d) = C(d), \quad \forall d \in \{d_j^i\}$$

   Problem 2: Splittable reliable embedding problem of $\mathcal{SC}_i$'s hanging virtual links

   The above MILP problem can be easily solved thanks to standard optimization solver such as CPLEX or GLPK. The computation time of convergence to the optimal solution is polynomial [79].

   The $\mathcal{SC}_i$'s mapping cost in each candidate router $w^j \in Clus_s$ denoted by η_i^j is defined as in unsplittable-based virtual link mapping (see equation 3.2.13)

3.2.3.3 Job of onlooker bee

Once the the mapping cost η_i^j of $\mathcal{SC}_i$ in each candidate of $w^j \in Clus_s$ by the worker bees is performed, the onlooker bee will select the candidate minimizing the cost (i.e., highest value of nectar). In fact v^i will be deployed in w^s if and only if:

$$\eta_i^s = \min_{j=1\cdots|Clus_s|} \left[\eta_i^j\right] \tag{3.2.16}$$

It is straightforward to see that our proposal does not allocate any backup resources. Basing on this preventive behavior, we aim to maximize the acceptance rate while reducing the blackout rate. Note that once an outage occurs in one physical equipment (router or link), all virtual resources embedded in the latter will be impacted. Accordingly, all $\mathcal{VN}$s related to the impacted virtual resources will be released from the $\mathcal{SN}$.

3.3 Performance evaluation

In this section, based on extensive simulations we will assess the performance of our proposal. To do so, first we will describe our virtual network discrete event simulator. Note that it considers the age of physical equipments which impact the reliability offered by the $\mathcal{SN}$. Then, we will define the performance metrics in order to evaluate the effeciency of our proposal and to compare it with most prominent related strategies. Finally, we will illustrate and comment the obtained results.

3.3.1 Simulation environment

We implemented a discrete event reliable $\mathcal{VN}$ mapping simulator. To this aim, as in [52] we make use of `GT-ITM` [80] tool to generate random $\mathcal{SN}$ and $\mathcal{VN}$ topologies. Note that we model the i) $\mathcal{VN}$ requests' arrival by a Poisson Process with rate λ_A and ii) $\mathcal{VN}$ lifetime by exponential distribution with mean μ_L. We set the number of $\mathcal{VN}$ requests to 2000. As in [52], we set the $\mathcal{SN}$ size to 100. In this case, the ratio of core and of access routers are respectively fixed to 80% and 20%. Moreover, we set $\mathcal{VN}$ size randomly following discrete uniform distribution taking values between $[3, 10]$. Similar to the $\mathcal{SN}$, the proportion of access and core virtual routers is fixed respectively to 20% and 80%. It is worth mentioning that in both cases ($\mathcal{VN}$ and $\mathcal{SN}$), each pair of routers is randomly connected with a probability of 0.5. The arrival rate λ_A and the average lifetime μ_L of $\mathcal{VN}$s are respectively set to 4 requests per 100 and 1000 time units. We set the capacity of substrate routers and links (i.e., $B(w)$, $M(w)$ and $\hat{C}(e)$) according to a continuous uniform distribution taking values in $[50, 100]$. Furthermore, we fix the required virtual resources for each router v (i.e., $B(v)$ and $M(v)$) and each link d (i.e., $\hat{C}(d)$) based on a continuous uniform distribution, using values between $[10, 20]$.

Regarding $\mathcal{SN}$ reliability, the parameters a and b of the Weibull distribution (see equation 3.1.1) modeling the MTBF are respectively set to 25×10^4 and 1.5. It is noteworthy that a and b are

Table 3.1: Initial age of $\mathcal{SN}$'s equipments

Group	Min age	Max age	Failure probability
Young	0	50 000	[0, 0.1]
Adult	200 000	230 000	[0.5, 0.6]
Old	400 000	500 000	[0.85, 0.97]

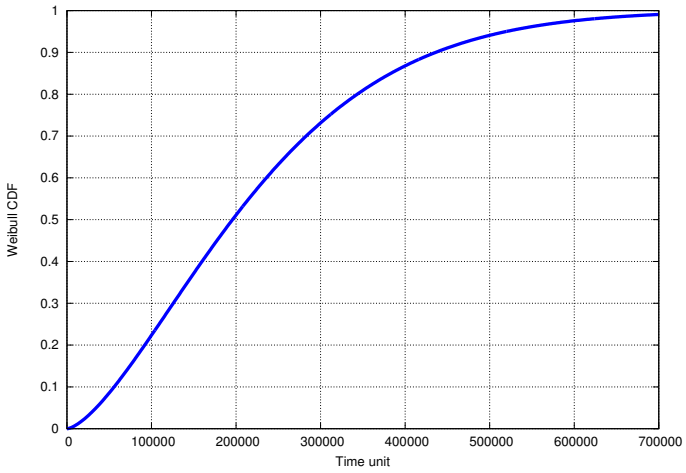


Figure 3.1: CDF of MTBF Weibull distribution

calibrated in such a way to obtain a lifetime for each physical equipment ($\approx 6 \times 10^5$) roughly equal to 10 times the simulation duration ($\approx 6 \times 10^4$ time units). Moreover, the proportion of young equipments in the $\mathcal{SN}$ is fixed to 30%. In our simulations, we will vary the rate of adult equipments in the $\mathcal{SN}$ and consequently the old proportion. Moreover, the initial age of physical equipment follows a uniform distribution within the bounds of its group (i.e., young, adult or old). The initial age bounds are summarized in Table 3.1. The failure probabilities on each group are calculated with the Weilbul distribution. Figure 3.1 illustrates the Cumulative Distribution Function CDF modeling the Mean Time Between failure according to the parameters fixed above (i.e., a and b).

In order to compare `Advanced-PR-VNE`, we implemented the most prominent reliable $\mathcal{VN}$ embedding strategies found in the literature and enumerated in Table 3.2. It is worth pointing out that the confidence level of simulation results is fixed to 95% using 30 samples for each value.

3.3.2 Performance metrics

Hereafter, we define the performance metrics used to gauge our proposal.

- Q is the reject rate of $\mathcal{VN}$ requests during the simulation. In other terms, the rate of $\mathcal{VN}$s

Table 3.2: Related survivable $\mathcal{VN}$ embedding strategies

Algorithm	Reference	Link Mapping
RMap	[65]	Unsplittable
CPP	[67]	Unsplittable
VNP	[67]	Unsplittable
SVNE	[64]	Splittable
ORP	[60]	Splittable
K-Redundant	[63]	Splittable

that have not been mapped in the $\mathcal{SN}$.

- F Evaluates the $\mathcal{VN}$s mapped in the $\mathcal{SN}$ that have been impacted by physical failures during the simulation (i.e., blackout rate). In fact, a $\mathcal{VN}$ is impacted if at least one of its components (i.e., virtual router and/or link) is affected by a physical failure. F is calculated as following: the ratio between the number of impacted $\mathcal{VN}$s per the number of accepted $\mathcal{VN}$ requests during all the simulaion.

- G evaluates the $\mathcal{SN}$ provider's profitability by calculating the total benefit G_1 of all accepted $\mathcal{VN}$ requests minus the paid penalties G_2 to the clients induced from the physical failures during all the simulation lifetime ($\mathrm{G} = \mathrm{G}_1 - \mathrm{G}_2$). G_1 is defined as in [55]. It depends on the amount of requested resources and the lifetime of the virtual network in the $\mathcal{SN}$'s backbone network. We define the penalty G_2 as the reimbursement to the clients du to service's rupture. The cost is proportional to the remaining time of a $\mathcal{VN}$ lifetime heightened by a penalty δ. Formally, for each $\mathcal{VN}$ denoted by $\mathcal{D}$:

$$\mathrm{G}_2(\mathcal{D}) = \frac{(T-\Delta_T)}{T} \times \mathrm{G}_1(\mathcal{D}) \times \delta \tag{3.3.17}$$

where Δ_T is the hosting duration in the $\mathcal{SN}$ before the physical failure and δ is the penalty rate. In our simulations, we set the penalty to 5%.

3.3.3 Simulation results

As explained in section 3.2.3.2, we propose two variants of Advanced-PR-VNE with respect to the virtual link embedding approaches: unsplittable or splittable. Hence, we will first evaluate the performance of Advanced-PR-VNE-unsplittable. Then, we will assess the second variant Advanced-PR-VNE-splittable. Finally, we will summarize the results obtained by discussing the pros and cons of each variant.

3.3.3.1 Unsplittable virtual link embedding approach

In this section, we gauge the performance of `Advanced-PR-VNE-unsplittable` and compare it with the related strategies adopting the unsplittable approach to map virtual links: i) `RMap`, ii) `CPP` and iii) `VNP` (see Table 3.2).

First of all, our evaluation focuses on the reject rate metric Q. It is straightforward to see in Figure 3.2 that both variants (i.e., `Basic` and `Advanced`) of our proposal outperform the related strategies whatever the age of $\mathcal{SN}$'s equipments. We recall that the proportion of young equipment is fixed to 30% and the rate of adult equipment is varied. We notice that the reject rate of $\mathcal{VN}$ requests decreases as the rate of adult equipments increases (i.e., proportion of adult equipments decreases) thanks to the rejuvenation of the $\mathcal{SN}$. For instance, if the rate of adult equipments in the $\mathcal{SN}$ is equal to 30% (i.e., proportions of young and old are respectively of 30% and 40%), `Advanced-PR-VNE-unsplittable` rejects only $37.46 \pm 1.51\%$. However, the rest of strategies including `Basic-PR-VNE` reject at least $91 \pm 1.80\%$ of requests. We notice that if the rate of **old** equipment is equal to 50% (i.e., proportion of adult equipment = 20%), our proposal is slightly better than the related strategies while the reject rate is high ($\geq 79.19 \pm 2.37\%$). The rational behind the previous results consists in the huge number of physical failures which deeply impact the connectivity and the availability of physical resources. Moreover, we notice that when the rate of adult equipments in the $\mathcal{SN}$ is approximately equal to 50%, the performance of both variants (`Basic` and `Advanced`) of our proposal are equivalent. However, if the rate of adults equipments reaches 60%, `Advanced-PR-VNE-unsplittable` outperforms all the strategies. The gain and loss of the related strategies and `Basic-PR-VNE` compared with `Advanced-PR-VNE-unsplittable` are shown in Figure 3.3. As we can see, the loss can reach 347.91%. As explained above, if the rate of adult equipments is approximately equal to 50%, the performance of `Advanced-PR-VNE-unsplittable` and `Basic-PR-VNE-unsplittable` are equivalent. In fact, we notice in Figure 3.3 a tiny mean gain of `Basic-PR-VNE-unsplittable` in terms of reject rate which is approximately equal to 20%.

Figure 3.4 illustrates the rate of $\mathcal{VN}$s impacted by the physical failures. It is straightforward to see that if the proportion of old equipments in the $\mathcal{SN}$ is greater than 30%, `Advanced-PR-VNE-unsplittable` minimizes the rate of $\mathcal{VN}$ impacted by physical failures. We recall that our proposal does not allocate any backup resources and takes into consideration the reliability by prediction during the embedding process. Fo instance, as shown in Figure 3.4, if the rate of old equipment is equal to 50% (i.e., the worst case) physical failures impacts only $15.99 \pm 2.63\%$ of virtual networks deployed by `Advanced-PR-VNE-unsplittable`. However, $21.63 \pm 1.36\%$, $21.40 \pm 1.24\%$, $43.56 \pm 1.66\%$ and $67.46 \pm 3.59\%$ of virtual networks respectively embedded by `Basic-PR-VNE`, `RMap`, `CPP` and `VNP` are impacted. Figure 3.5 illustrates the loss and gain, in terms of F, of the related strategies compared with our proposal `Advanced-PR-VNE-unsplittable`. As we can see, if the network equipment is old (i.e. rate of old equipments is larger than

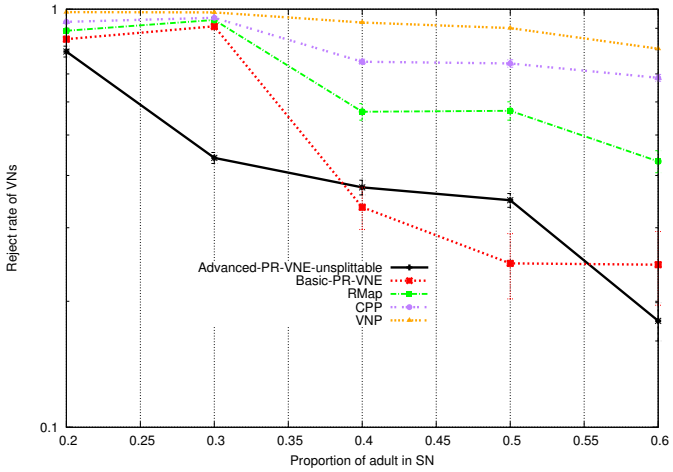


Figure 3.2: Q - Unsplittable based approaches

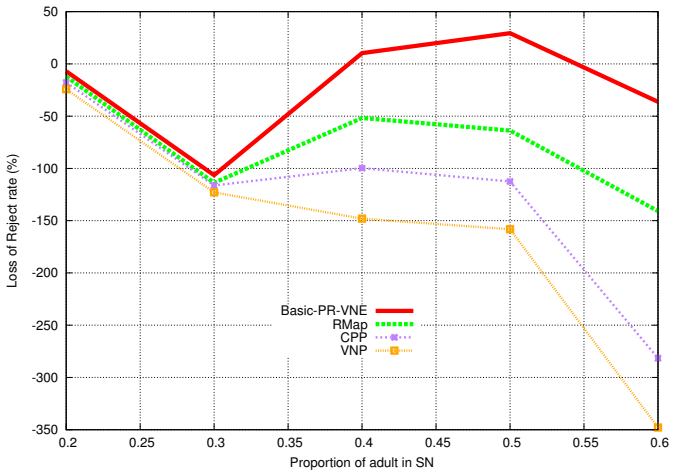


Figure 3.3: Loss of reject rate Q compared with `Advanced-PR-VNE-unsplittable`

30%), our proposal minimizes the blackout of $\mathcal{VN}$s. The loss can reach 760.56% (`CPP` strategy). It is worth pointing out that if the mean age of $\mathcal{SN}$'s equipments is adult or young, the performance of `Advanced-PR-VNE-unsplittable` is approximately equivalent to `Basic-PR-VNE`, `RMap` and `VNP`. The blackout rate of $\mathcal{VN}$s is less than 10%. Indeed, a reliable virtual network embedding strategy must outperform when the network is old and the cost to renew the $\mathcal{SN}$ is expensive. Hence, the idea is to maximize the exploitation of old equipments while the global revenue must be stable and doest not crash down du to the penalties.

Figure 3.6 evaluates $\mathcal{CP}$'s gain G. Note that the latter is proportional to the number of ac-

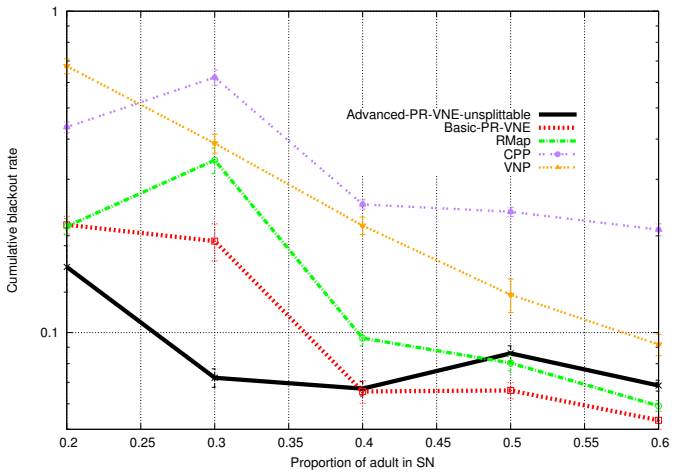


Figure 3.4: F - Unsplittable based approaches

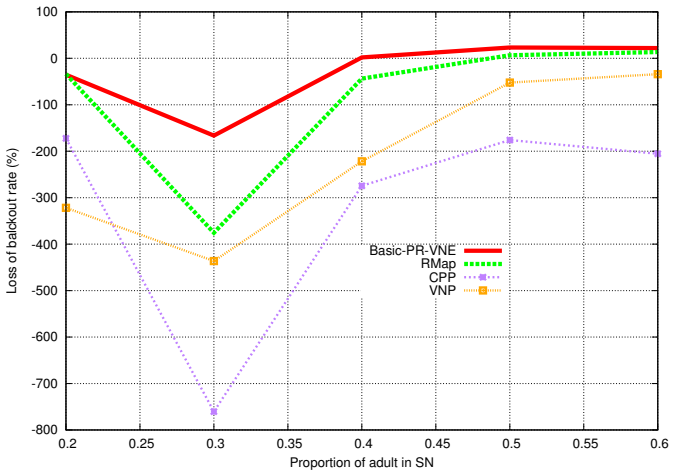


Figure 3.5: Loss of blackout rate F compared with `Advanced-PR-VNE-unsplittable`

cepted $\mathcal{VN}$s and their requested resources. Moreover, G is impacted by failures. It is straightforward to see that both variants of our proposal (i.e., `Advanced-PR-VNE-unsplittable` and `Basic-PR-VNE`) outperforms all the related strategies. This can be explained by i) the maximization of acceptance rate of $\mathcal{VN}$s (see Figure 3.2) and ii) the minimization of impacted $\mathcal{VN}$s by physical failures (see Figure 3.4). Moreover, we notice that `Advanced-PR-VNE-unsplittable` outperforms the basic variant `Basic-PR-VNE` if at least 30% of the equipments are old. This can be explained by the judicious choice of the substrate routers and links during the embedding process thanks to the classification clustering algorithm `K-Means`. We recall that `Basic-PR-VNE` makes

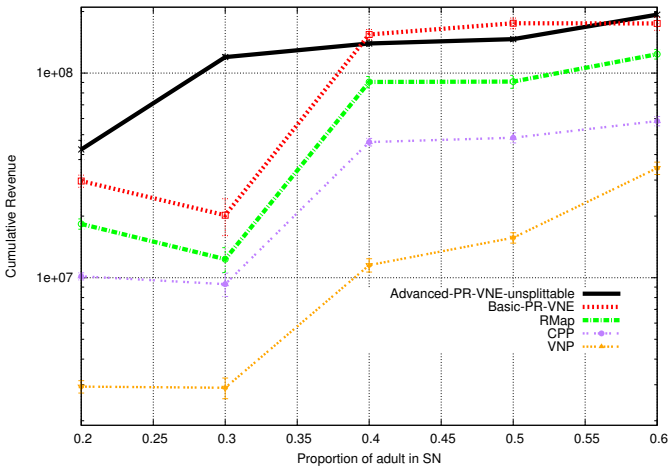


Figure 3.6: G - Unsplittable based approaches

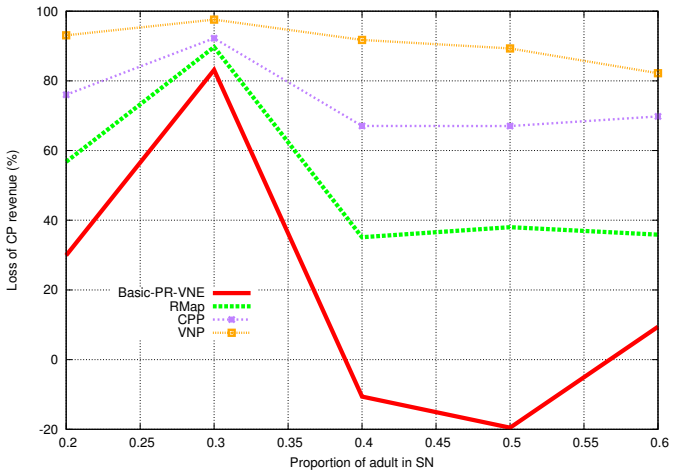


Figure 3.7: Loss of $\mathcal{CP}$'s gain G compared with `Advanced-PR-VNE-unsplittable`

use of the neighborhood of geographic barycenter which is simplistic. In Figure 3.7, we illustrate the loss of revenue compared with `Advanced-PR-VNE-unsplittable`. It is straightforward to see that the related strategies do not maximize the $\mathcal{CP}$'s revenue. For instance, the loss can reach 97.57% when the rate of old equipment is equal to 40%.

3.3.3.2 Splittable virtual link embedding approach

In this section, we assess the performance of the second variant `Advanced-PR-VNE-splittable`. The comparison with related splittable-based approaches: i) `SVNE`, ii) `K-Redundant` and iii) `ORP`

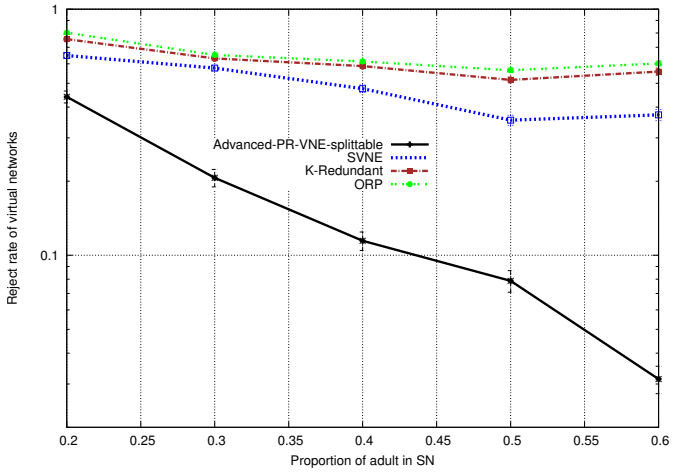


Figure 3.8: Q - Splittable based approaches

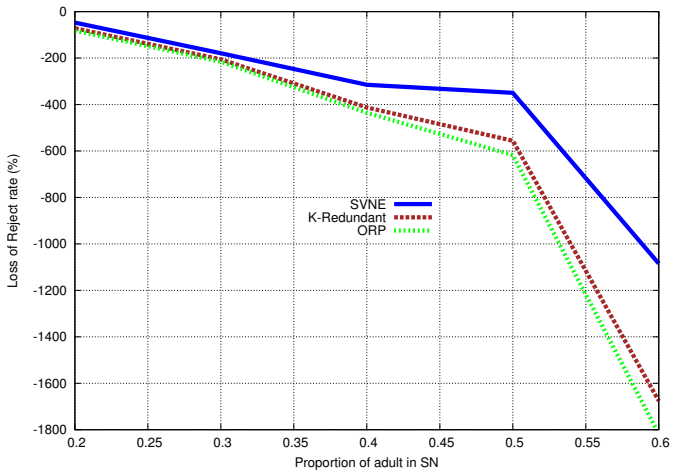


Figure 3.9: Loss of reject rate Q compared with `Advanced-PR-VNE-splittable`

(see Table 3.2) will be performed.

In Figure 3.8, we evaluate the reject rate Q of $\mathcal{VN}$s. We notice that `Advanced-PR-VNE-splittable` outperforms all the related strategies whatever the age of the $\mathcal{SN}$'s equipment. We notice that the reject rate induced by our proposal varies between $3.14 \pm 0.40\%$ and $44.02 \pm 2.44\%$ and is always better than the related strategies. This can be explained by i) the non-allocation of backup resources comparing to related works and i) the load balancing of physical resource usage in the $\mathcal{SN}$ thanks to our proposed objective function (See section 3.2.3.2). It is easy to observe that with our proposal Q decreases as the age of the $\mathcal{SN}$ rejuvenates. This demonstrates

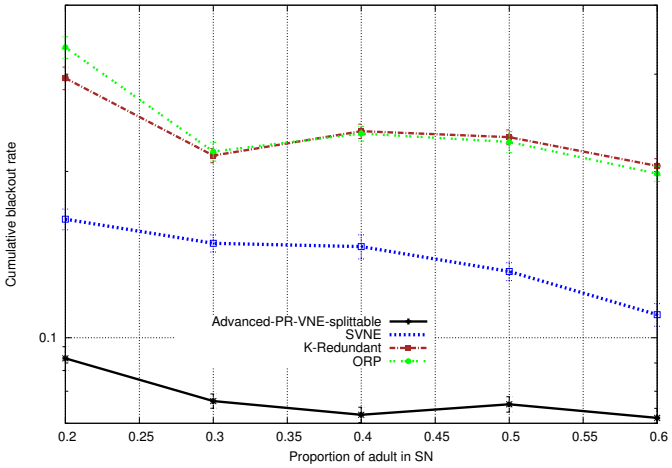


Figure 3.10: F - Splittable based approaches

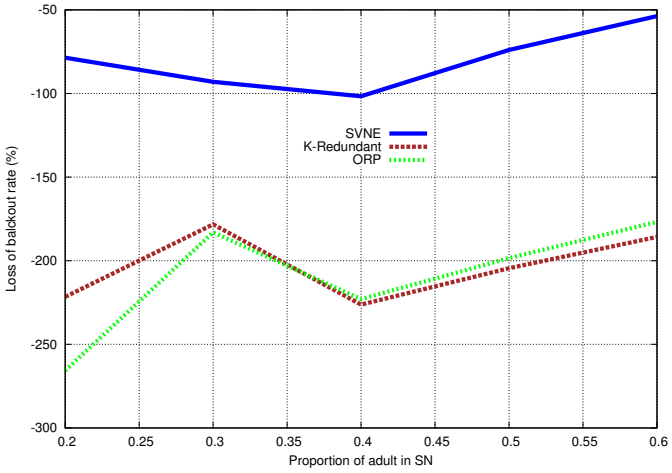


Figure 3.11: Loss of blackout rate F compared with `Advanced-PR-VNE-splittable`

that the preventive approach to consider the reliability during the embedding process and the failure prediction model are relevant. On the other hand, we notice that the related strategies do not avail the rejuvenation of the physical equipments. As shown in Figure 3.8, the related strategies maintain Q approximately stable around 70% which is not suitable. In Figure 3.9, we illustrate the loss rate compared to our proposal in terms of reject rate resulted with the related strategies. The gain of `Advanced-PR-VNE-splittable` is clear. We notice two behaviors. If the $\mathcal{SN}$ is old, the loss rate of our proposal is approximately 70%. If the substate network is young than we observe the loss plummeting that can reach 1800%.

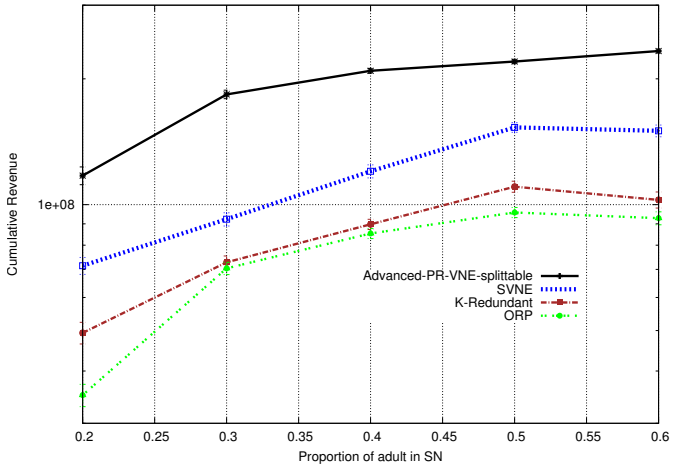


Figure 3.12: G - Splittable based approaches

Figure 3.10 illustrate the blackout rate F of $\mathcal{VN}$s crashed by physical failures. We observe that `Advanced-PR-VNE-splittable` maintains stable the rate of $\mathcal{VN}$ impacted (varies between $7.15 \pm 0.29\%$ and $9.17 \pm 0.45\%$) with respect to the age of the $\mathcal{SN}$'s equipments. This result show that our proposal is not sensitive to the rate of old equipments in the $\mathcal{SN}$. We observe that the related strategies do not sufficiently protect the embedded $\mathcal{VN}$s. As shown in Figure 3.11, the related strategies deteriorate the rate of blackout $\mathcal{VN}$s compared with `Advanced-PR-VNE-splittab- le` between 50% and 250%. We can put forward that our proposal takes into account the physical failures better than related work.

Figure 3.12 illustrates $\mathcal{CP}$'s gain G. It is clear that `Advanced-PR-VNE-splittable` ensures the best turnover since it maximizes the acceptance rate and minimizes the penalties induced by failures. This result is the direct consequence of the performances in terms of: i) reject rate Q and blackout rate F of $\mathcal{VN}$s. We notice that the $\mathcal{CP}$'s revenue increases as the $\mathcal{SN}$ become youngest. In Figure 3.13, it is straightforward to see that the revenue loss of the related strategies is at least 30% and can reach 70% even the physical equipment is old. For instance, if the proportion of adult equipments in the $\mathcal{SN}$ is 30% (i.e., old and youn are respectively 40% and 30%), the revenue loss of i) `SVNE`, ii) `K-Redundant` and iii) `ORP` compared with our proposal `Advanced-PR-VNE-splittable` is respectively i) 49.71%, ii) 60.32% and iii) 61.63%.

3.3.3.3 Summary

Two variants of `Advanced-PR-VNE` with respect to the virtual link mapping approach are proposed and evaluated. Based on extensive simulations, we conclude that the splittable approach outperforms the unsplittable one in terms of i) reject rate, ii) blackout rate of virtual networks and

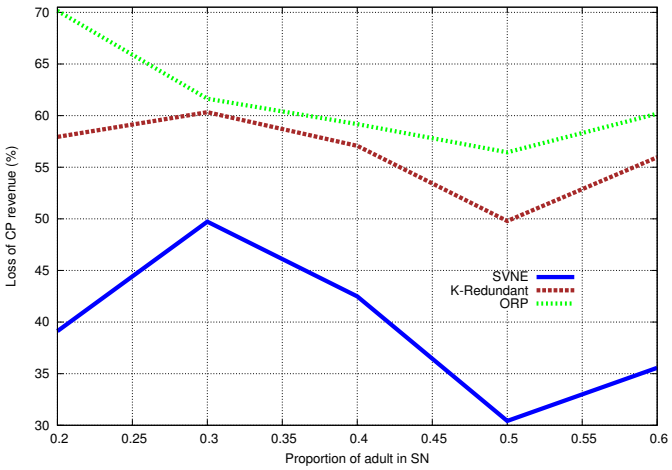


Figure 3.13: Loss of $\mathcal{CP}$'s gain G compared with `Advanced-PR-VNE-splittable`

iii) Cloud provider's revenue. In fact, the above obtained results are in line with our original expectations. Indeed, mapping each virtual link within multiple substrate paths in the $\mathcal{SN}$ favors the avoidance of bandwidth congestion and offers more flexibility to select the most reliable resources.

Unfortunately, $\mathcal{SN}$s mainly make use of shortest-path-based routing protocols such as Open Shortest Path First (OSPF) or Routing Information Protocol (RIP). Indeed, to employ sophisticated splittable routing algorithms, already deployed IP routers must be upgraded. This would significantly increase expenses. Consequently, the unsplittable approach is crucial to satisfy the $\mathcal{CP}$'s clients in short term.

We estimated the convergence time to calculate the mapping of one $\mathcal{VN}$ request. The results obtained are illustrated in Table 3.3. It is straightforward to notice that the unsplittable approach (i.e., `Advanced` and `Basic`) needs more time (i.e., around 3 seconds) to converge in comparison with splittable variant. The latter is tiny and it is equal to 372.979 ± 23 ms. Note that the convergence time of splittable approach is 10 times less than the unsplittable variant. However, the convergence time of `Advanced-PR-VNE-unsplittable` is still reasonable (3 seconds). These results validate the scalability propriety of our proposal and its potential to be exploited in large-scale Cloud network. Finally, we notice that with `Basic-PR-VNE`, the number of hops $\mathcal{H}$ strongly impacts the convergence time. In fact, the number of candidates in the optimization research field is proportional to $\mathcal{H}$.

Note that the simulation results illustrated in Table 3.3 are calculated in a machine characterized by: i) CPU Intel i7 3770 MHz with 4 cores and ii) size of memory is 16 Giga Bytes.

Table 3.3: Convergence time PR-VNE

Variant	Time (ms)
Advanced-PR-VNE-unsplittable	3324.404 ± 37
Basic-PR-VNE, $\mathcal{H}$ = 2 hops	6645.995 ± 21
Basic-PR-VNE, $\mathcal{H}$ = 3 hops	10147.004 ± 34
Advanced-PR-VNE-splittable	372.979 ± 23

3.4 Conclusion

In this chapter, we studied the reliable $\mathcal{VN}$ embedding problem typified as NP-hard. Beside the virtual network complexity, the survivable issue makes the problem more difficult. Accordingly, we proposed a new strategy named Advanced-PR-VNE based on artificial bee colony metaheuristic and classification algorithm. Note that our proposal is preventive approach and does not allocate any backup resources while guaranteeing better performance. The main idea behind our proposal is to promote the use of reliable physical resources during the virtual network embedding process. Two variants are propounded according the virtual link mapping approach: i) unsplittable and ii) splittable. Based on extensive simulations, the proposal outperforms the most prominent related strategies in terms of: i) reject rate of virtual networks, ii) rate of virtual networks impacted by substrate outages and iii) Cloud provider's revenue.

In the next chapter, we will propose a reactive algorithm that re-embeds the impacted virtual resources (i.e., routers and links) in order to ensure the service continuity despite the hardware failure. This propounded algorithm, named CG-VNE, is based on Game Theory framework to embed the virtual networks. Beside the use of the same preventive mechanism like PR-VNE, our new proposal CG-VNE migrates the impacted resources to safe physical equipments once an outage occurs in $\mathcal{SN}$.

Chapter 4

Reactive survivable virtual network mapping

Contents

4.1 Problem statement 76
4.1.1 Virtual and substrate network models 76
4.1.2 Formalization of the reliable $\mathcal{VN}$ embedding problem 77
4.2 Proposal: `Advanced-CG-VNE` 82
4.2.1 Embedding of virtual routers 82
4.2.2 Embedding of virtual links 84
4.2.3 Reliability support 87
4.3 Performance evaluation 87
4.3.1 Simulation environment 87
4.3.2 Performance metrics 89
4.3.3 Simulation results 90
4.4 Conclusion 97

In the previous chapter, we proposed a preventive approach denoted by `PR-VNE` that urges the use of the reliable physical equipments in order to avoid the service interruption due to the hardware failures. However, this strategy lacks of a curative operation ensured by a recovery mechanism that guarantees service continuity. This absence of a reconfiguration mechanism that re-embeds the impacted virtual resources, encourages us to define a new algorithm that helps the Cloud provider to respect the signed SLA with the client.

In this chapter, we tackle the virtual network embedding problem within Cloud's backbone network by taking into consideration the impact of physical equipments' outages. Our main focus is to improve the provider's revenue by i) maximizing the acceptance rate of virtual networks and

ii) minimizing the penalties induced by service disruption due to the hardware outages. This optimization problem is NP-hard with multi-objective and non-linear formalization. To cope with this complexity, we propose an Advanced Coordination Game for Virtual Network Embedding denoted by `Advanced-CG-VNE`. In this mapping game, fictitious players are playing on bahlf the Cloud provider in order to increase the turnover. These decision makers cooperate to converge to a Nash Equilibrium of which we have proven the existence. This Nash Equilibrium matches with a social optimum. Two variants of `Advanced-CG-VNE` are defined according to the virtual links embedding approach. The first one, denoted by `Advanced-CG-VNE-Unsplit`, embeds a virtual link in one substrate path. The second variant, denoted by `Advanced-CG-VNE-splittab- le`, dispatches the required bandwidth of a virtual link among a set of substrate paths. `Advanced-CG-VNE` does not allocate any backup to alleviate the service interruption impact but it adopts a preventive and a reactive approach to handle substrate failures. Extensive simulations show that `Advanced-CG-VNE` outperforms the related work strategies in terms of i) rejection rate of virtual networks, ii) rate of virtual networks impacted by physical failures and iii) Cloud provider's turnover.

The current chapter is organized as follows. In Section 4.1, we will formulate the virtual and substrate network models as well as the reliable $\mathcal{VN}$ mapping problem. Then, we will detail our proposed strategy (i.e., `Advanced-CG-VNE`) in Section 4.2. Afterwards, we will expose the performance evaluation in Section 4.3. Finally, Section 4.4 will conclude this chapter.

4.1 Problem statement

As in the previous chapter, in this section we will recall our model of the reliable $\mathcal{VN}$ embedding problem. For this aim, we defined the virtual and substrate network models. Later on, we will formulate the survivable $\mathcal{VN}$ mapping problem basing on a Game Theory approach. To do so, we will define the: i) players ii) payoff functions and iii) utility functions.

4.1.1 Virtual and substrate network models

On one hand, a $\mathcal{VN}$ request is modelized as an undirected graph, denoted by $\mathcal{D} = (V(\mathcal{D}), E(\mathcal{D}))$ where $V(\mathcal{D})$ and $E(\mathcal{D})$ are respectively the sets of virtual routers and links. Each virtual link $d \in E(\mathcal{D})$ is characterized by its required bandwidth $C(d)$. Furthermore, each virtual router, $v \in V(\mathcal{D})$, is characterized by its i) requested processing power $B(v)$ and memory $M(v)$ and ii) its type: access or core $X(v)$. Note that if $X(v) = 1$, then v is an access virtual router. Otherwise, v is a core virtual router. Besides, each one of the virtual access routers v is characterized by a geographical region denoted by $\mathcal{Z}_v$. It defines the geographical zone in which v may be embedded.

On the other hand, the $\mathcal{SN}$ (i.e., $\mathcal{CP}$'s network) is modelized as an undirected graph, denoted by $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ where $V(\mathcal{G})$ and $E(\mathcal{G})$ are respectively the sets of substrate (i.e., physical)

routers and links. Each substrate router, $w \in V(\mathcal{G})$, is typified by its i) residual processing power $B(w)$, ii) remaining memory capacity $M(w)$, iii) type: access or core $X(w)$, and iv) reliability $\mathcal{R}^n(w,t)$ at a time t. Note that if $X(w) = 1$, then w is an access substrate router. Otherwise, w is a core substrate router. Besides, the initial capacity in terms of processing power and memory are respectively denoted by $\hat{B}(w)$ and $\hat{M}(w)$. Each physical link, $e \in E(\mathcal{G})$, is typified by its i) remaining bandwidth $C(e)$, ii) initial bandwidth capacity $\hat{C}(e)$ and iii) reliability $\mathcal{R}^l(e,t)$ at time t.

Basing on [71] results, we formulate the Mean Time Between Failures (MTBF) in the $\mathcal{SN}$ as a Weibull distribution. Therefore, it is possible to estimate and predict the equipment's reliability at any time. Consequently, the Cumulative Distribution Function (CDF) of the MTBF is expressed as:

$$F(x) = 1 - \exp(-\left(\frac{x}{a}\right)^b),\ x \geq 0 \tag{4.1.1}$$

where a and b are the parameters of the Weibull distribution. It is noteworthy that we assume heterogeneous physical equipments in the $\mathcal{SN}$ basing on different resources' ages. Consequently, each physical equipment in $\mathcal{G}$ has its own initial age denoted by $A(w)$ and $A(e)$ respectively for substrate node w and substrate link e. Accordingly, we classify the equipments within the substrate network into three groups: i) **young**, ii) **adult** and iii) **old**. Thus, we can evaluate the substrate resource's reliability at any instant t. Formally, the reliability of a i) substrate router w is equal to $\mathcal{R}^n(w,t) = F(A(w) + t)$ and ii) substrate link e is equal to $\mathcal{R}^l(e,t) = F(A(e) + t)$. It is worth pointing out that the substrate resource's reliability is inversely proportional to its age. In other terms, the probability of a physical failure increases with time.

4.1.2 Formalization of the reliable $\mathcal{VN}$ embedding problem

As it was aformentionned, the substrate resources in $\mathcal{G}$ are limited. Consequently, the $\mathcal{CP}$'s network can not to host an infinite number of $\mathcal{VN}$ requests. Accordingly, a smart and judicious $\mathcal{VN}$ embedding algorithm is compulsory in order to i) maximize the acceptance rate of $\mathcal{VN}$ requests hence the $\mathcal{CP}$'s turnover is improved and ii) minimize the blackout rate (i.e., the rate of $\mathcal{VN}$ impacted by physical failures) hence $\mathcal{CP}$'s penalty is reduced.

4.1.2.1 Modelization of virtual router mapping

The maximization of the acceptance rate of $\mathcal{VN}$ requests can be achieved by i) minimizing the amount of resources allocated for each $\mathcal{VN}$ request and ii) maximizing the load balancing of the resource usage rate within the $\mathcal{SN}$. To reach the above objectives, the residual and the standard deviation of residual resources should be respectively maximized and minimized. Formally,

$$\begin{gathered}\textbf{maximise}[\min\{C(e)\}, \min\{B(w)\}, \min\{M(w)\}] \\ \textbf{minimise}[std\{C(e)\}, std\{B(w)\}, std\{M(w)\}] \\ e \in E(\mathcal{G}),\ \ w \in V(\mathcal{G})\end{gathered} \tag{4.1.2}$$

Moreover, our objective consists in mapping the virtual **routers** and **links** within the substrate resources offering the highest reliability during the lifetime of each $\mathcal{VN}$.
For each virtual router $v \in V(\mathcal{D})$ in a $\mathcal{VN}$, the reliability objective is expressed as following:

$$\textbf{maximise}\{\mathcal{R}^n(w^v, T_{w^v})\}, w^v \in V(\mathcal{G}) \tag{4.1.3}$$

where w^v denotes the substrate router hosting v and T_{w^v} is the age of w^v when the virtual network $\mathcal{D}$ leaves the physical backbone network $\mathcal{G}$.

4.1.2.2 Virtual router embedding basing on Game Theory

We modelize the survivable $\mathcal{VN}$ mapping problem as a game. In [41], Game Theory is defined as the study of mathematical models interaction between rational decision makers (i.e., players). A game $\Gamma(\mathcal{N}, \mathcal{S}, \mathcal{U})$ is defined basing on the main following parameters:

1. $\mathcal{N} = \{\mathcal{N}_1, \cdots, \mathcal{N}_p\}$: Set of **players**.
2. $\mathcal{S}$ is the **action profile** for all players $\mathcal{N}$. It is equal to $\mathcal{S} = \prod_{\mathcal{N}_i \in \mathcal{N}} \mathcal{S}_{\mathcal{N}_i}$. Note that $\mathcal{S}_{\mathcal{N}_i}$ is the **action set** of player $\mathcal{N}_i$.
3. $\mathcal{U}$ is the vector of **game's utility function**. $\mathcal{U}$ is equal to $\left(\mathcal{U}_{\mathcal{N}_1}, \cdots, \mathcal{U}_{\mathcal{N}_p}\right)$. Note that $\mathcal{U}_{\mathcal{N}_i}$ is the utility function of player $\mathcal{N}_i$ defined as, $\mathcal{U}_{\mathcal{N}_i} : \mathcal{S} \rightarrow \mathbb{R}$.

We modelize the virtual network embedding problem as a **coordination game** [81] where all fictitious players $\{\mathcal{N}_i\}$ collaborate together to reach a **Nash Equilibrium**. In order to motivate fictitious players $\{\mathcal{N}_i\}$ to cooperate together, decision makers share the same utility function. This kind of games is defined as an **Identical Interest Game** (IIG) [82]. Formally, $\forall \mathcal{N}_i, \mathcal{N}_j \in \mathcal{N} : \mathcal{U}_{\mathcal{N}_i} = \mathcal{U}_{\mathcal{N}_j}$. Basing on this game modelization, fictitious players will play on behalf of the $\mathcal{CP}$ in order to improve the turnover.

We recall that our problem is to embed a $\mathcal{VN}$ (i.e., set of virtual routers and links). It is straightforward to see that virtual router mapping strongly depends on virtual link embedding and vice-versa. Thus, we devise a virtual router mapping game that takes into consideration virtual link embedding. We denote by Γ this virtual router embedding game. Hereafter, we define the parameters for our game $\Gamma(\mathcal{N}, \mathcal{S}, \mathcal{U})$.

The set of fictitious decision makers (i.e., players) $\mathcal{N}$ matches with the set the virtual routers $V(\mathcal{D})$. We assume that an initial position in the $\mathcal{SN}$ (i.e., substrate router) is dedicated for each one of the fictitious player (i.e., virtual router). The strategies (i.e., actions set) $\mathcal{S}_{\mathcal{N}_i}$ of a fictitious player $\mathcal{N}_i$ includes the K-hop neighbourhood of the initial physical router hosting the latter player $\mathcal{N}_i$ beside a set of substrate routers that are randomly selected in $V(\mathcal{G})$. It is noteworthy that all substrate routers $s_{\mathcal{N}_i} \in \mathcal{S}_{\mathcal{N}_i}$ have enough physical resources (i.e., $B(\mathcal{N}_i) \leq B(s_{\mathcal{N}_i})$ and

$M(\mathcal{N}_i) \leq M(s_{\mathcal{N}_i})$). Moreover, they satisfy the type constraint (i.e., $X\left(\mathcal{N}_i^1\right) = X\left(s_{\mathcal{N}_i^1}\right)$). The payoff function $\mathcal{U}_{\mathcal{N}_i}$ of one fictitious player $\mathcal{N}_i$ strongly depends on the i) remaining resources (i.e., memory and processing power) of the physical router $w_{\mathcal{N}_i}$, ii) its reliability and iii) the payoff gained by embedding its attached virtual links. Formally we have,

$$\mathcal{U}_{\mathcal{N}_i} = \sum_{d \in \xi_{\mathcal{N}_i}} (\mathcal{L}_d) + \mid \exp\left[R^n(w_{\mathcal{N}_i}, T_{\mathcal{N}_i}) \cdot \nabla^w_{\mathcal{N}_i}\right] - 1 \mid \tag{4.1.4}$$

where $\mathcal{L}_d$ is the payoff earned by mapping the virtual link d, $\xi_{\mathcal{N}_i}$ is the set of attached virtual links to the virtual router player $\mathcal{N}_i$, $R^n(w_{\mathcal{N}_i}, T_{\mathcal{N}_i})$ is the reliability of the substrate router w hosting the virtual router player at time $T_{\mathcal{N}_i}$. Note that $T_{\mathcal{N}_i}$ corresponds to the departure time of the virtual network $\mathcal{D}$, $\nabla^w_{\mathcal{N}_i}$ is the residual resources of substrate router w after the mapping of virtual player $\mathcal{N}_i$. More details regarding the payoff (i.e., $\mathcal{L}_d$) earned by link mapping will be exposed in the next subsection. The utility function is equal to:

$$\mathcal{U} = \sum_{\mathcal{N}_i \in \mathcal{N}} \mathcal{U}_{\mathcal{N}_i} \tag{4.1.5}$$

It is clear that basing on the defined payoff and utility function, the reliability of $\mathcal{VN}$ is maximized. In other words, these functions are proportional to the selected substrate resources reliability.

On the other hand, $\mathcal{U}_{\mathcal{N}_i}$ is proportional to the remaining physical resources in the $\mathcal{CP}$ network. Accordingly, we ensure the $\mathcal{SN}$ load balancing. As a consequence, the acceptance rate of $\mathcal{VN}$ requests will increase. In other terms, load balancing a network avoides congestion, so the $\mathcal{SN}$ have enough physical resources to handle incoming $\mathcal{VN}$ requests.

Once we finalized the definition of our mapping game, we concentrate our focus to the existence and convergence to the **Nash Equilibrium** (**NE**) in our game. In fact, **NE** is defined as a set of actions, one for each player $\mathcal{N}_i$, such that no decision maker has incentive to unilaterally change its strategy (i.e., action). In other words, **NE** is a steady state where each player cannot improve its benefit anymore by individually deviating its choice, assuming that other players keep constant their strategies. Formally, a strategy $\mathcal{S}^* = (s^*_{\mathcal{N}_1}, \ldots, s^*_{\mathcal{N}_p}) \in \mathcal{S}$ is a **NE** if: $\forall\, \mathcal{N}_i \in \mathcal{N},\ \forall\, s_{\mathcal{N}_i} \in \mathcal{S}_{\mathcal{N}_i}$:

$$\mathcal{U}_{\mathcal{N}_i}\left(s^*_{\mathcal{N}_i}, \mathcal{S}^*_{-\mathcal{N}_i}\right) \geq \mathcal{U}_{\mathcal{N}_i}\left(s_{\mathcal{N}_i}, \mathcal{S}^*_{-\mathcal{N}_i}\right)$$

where $\mathcal{S}^*_{-\mathcal{N}_i} = \mathcal{S}^* \backslash s^*_{\mathcal{N}_i}$.

However, the **NE** does not exist for all games. So, hereafter we will explain that our game modelization has at least one Nash Equilibrium.

Basing on the aforementioned model, our $\mathcal{VN}$ mapping game is formulated as an **Identical Interest Game** (IIG). It is worth pointing out that an IIG is a special case of a **Potential Game** [83]. Hereafter, we explain this claim. It is noteworthy that basing on the Potential Game, it is possible to compute the players deviation without using the players'utility function. But first of all, we introduce the Potential Game. Formally, a Potential Game is characterised by a Potential function

ϕ satisfying:

$$\begin{array}{ll} \phi: & \mathcal{S}_{\mathcal{N}_1} \times \cdots \times \mathcal{S}_{\mathcal{N}_p} \rightarrow \quad \mathbb{R} \\ & where \quad \forall\, \mathcal{N}_i \in \mathcal{N},\, \forall\, s^1_{\mathcal{N}_i}, s^2_{\mathcal{N}_i} \in \mathcal{S}_{\mathcal{N}_i}: \\ & \phi\left(s^1_{\mathcal{N}_i}, s^1_{-\mathcal{N}_i}\right) - \phi\left(s^2_{\mathcal{N}_i}, s^2_{-\mathcal{N}_i}\right) = \\ & \mathcal{U}_{\mathcal{N}_i}\left(s^1_{\mathcal{N}_i}, s^1_{-\mathcal{N}_i}\right) - \mathcal{U}_{\mathcal{N}_i}\left(s^2_{\mathcal{N}_i}, s^2_{-\mathcal{N}_i}\right) \end{array} \tag{4.1.6}$$

It is straightforward to see that to prove that an IIG is a Potential Game, we can define the common utility function shared by all players as the potential function.

Let's enumerate two interesting properties of a potential game proven in [83]:

1. Every finite (i.e., actions' cardinal $|\mathcal{S}|$ is finite) potential game has at least one pure strategy to reach a Nash Equilibrium.

2. All Nash Equilibria are either **local** or **global** maximzers of the utility function $\mathcal{U}$.

Accordingly, our reliable $\mathcal{VN}$ mapping model includes at least one **NE**. Besides, basing on our formalization the Nash Equilirium is equal to a **social optimum** which satisfies our main objective. In fact, decision makers (i.e., the defined fictitious players) plays on behalf the Cloud Provider to reach the social optimum.

4.1.2.3 Modelization of virtual link embedding

Basing on the outlined game above, the virtual link mapping is crucial step to reach a reliable embedding of the hole $\mathcal{VN}$. In fact, we propose two approaches: i) unsplittable virtual link embedding and ii) splittable one. We define the following objectives, which should be optimized basing on the virtual link mapping. Formally:

- **Unsplittable virtual link embedding**: For each virtual link $d \in E(\mathcal{D})$,

$$\begin{array}{ll} \textbf{maximize} & \Big\{ \prod_{e^d \in \mathcal{P}} \left(\mathcal{R}^l\left(e^d, T_{e^d}\right)\right) \cdot \quad \prod_{e^d \in \mathcal{P}} \left(\frac{C(e^d)}{\hat{C}(e^d)}\right) \cdot \\ & \prod_{w^d \in \mathcal{P}} \left(\mathcal{R}^n\left(w^d, T_{w^d}\right)\right) \Big\}, \mathcal{P} \in \textit{Paths}(d) \end{array} \tag{4.1.7}$$

  where $\{e^d\}$ and $\{w^d\}$ denote the set of substrate links and routers forming the substrate path $\mathcal{P} \in \textit{Paths}(d)$. $\textit{Paths}(d)$ denotes the set of substrate paths which can host the virtual link d. We recall that T_{w^d} and T_{e^d} are respectively the ages of the physical router w^d and link e^d when the virtual network $\mathcal{D}$ leaves the backbone network $\mathcal{G}$.
  It is straightforward to see that the above gain function quantifies the reliability provided by substrate path's equipments (i.e., routers and links) and residual bandwidth. It is noteworthy that this optimization function urges the use of reliable physical resources and aims also to load balance the substrate network.

- **Splittable virtual link embedding**: For all virtual links (i.e., $E(\mathcal{D})$),

$$\textbf{maximize} \quad \left\{ \sum_{d \in E(\mathcal{D})} \sum_{\mathcal{P} \in \textit{Paths-all}(d)} \left[\left(\frac{\textit{len}\left(\text{Dijkstra}_d\right)}{\textit{len}\left(\mathcal{P}\right)} \right) \cdot \mathcal{R}^p\left(\mathcal{P}\right) \cdot \bar{C}\left(\mathcal{P}\right) \right] \right\}, \mathcal{P} \in \textit{Paths-all}(d) \tag{4.1.8}$$

  where i) $\textit{Paths-all}(d)$ is the set of all substrate paths connecting the substrate routers hosting the virtual link d's extremities, ii) $\textit{len}\left(\mathcal{P}\right)$ is the number of hops of $\mathcal{P}$, iii) $b\left(\mathcal{P}, d\right)$ is the allocated bandwidth for virtual link d in path $\mathcal{P}$, iv) $\mathcal{R}^p\left(\mathcal{P}\right)$ corresponds to the lowest reliability value of the equipments (router and link) forming the path $\mathcal{P}$, and v) $\bar{C}\left(\mathcal{P}\right)$ corresponds to the lowest usage rate value of the substrate links forming the path $\mathcal{P}$. Note that unlike the unsplittable-based approach, splittable virtual link embedding handles all the virtual links of a $\mathcal{VN}$ at once. It is worth noting that Dijkstra_d is the shortest substrate path hosting the virtual link d.

  Note that the above gain function evaluates i) the reliability provided by substrate path's equipments (i.e., routers and links) and ii) remaining bandwidth in physical link.

We recall that the player payoff function includes one of the defined gain functions 4.1.7 and 4.1.8 basing on the virtual link mapping approach (i.e., unsplittable or splittable). In other words, in the payoff function $\mathcal{U}_{\mathcal{N}_i}$ (i.e., equation 4.1.4) $\mathcal{L}_d$ should be substituted by the i) payoff function 4.1.7 for unsplittable link mapping and ii) payoff function 4.1.8 for splittable one.

4.1.2.4 Virtual network mapping constraints

It is noteworthy that the $\mathcal{VN}$ embedding operation should respect some constraints. Hereafter, we present the main constraints.

Regarding virtual routers embedding, for each $\mathcal{VN}$ request (i.e., $\mathcal{D}$), one substrate router hosts at most one virtual node. In other terms, two (i.e., or more) virtual routers cannot be mapped into the same physical router. Formally,

$$\sum_{v \in V(w)} x_{vw} \leq 1, \quad \forall w \in V\left(\mathcal{G}\right) \tag{4.1.9}$$

where x_{vw} is a binary variable indicating whether virtual router $v \in V\left(\mathcal{D}\right)$ is embedded into the physical router $w \in V\left(\mathcal{G}\right)$, $V\left(w\right) \subseteq V\left(\mathcal{D}\right)$ denotes the set of virtual routers that can be embedded in physical router $w \in V\left(\mathcal{G}\right)$.

Moreover, one virtual router cannot be mapped to more than one physical router. In other words, the task(s) ensured by one virtual router cannot be dispatched into many substrate routers.

$$\sum_{w \in W(v)} x_{vw} = 1, \quad \forall v \in V\left(\mathcal{D}\right) \tag{4.1.10}$$

where $W(v) \subseteq V(\mathcal{G})$ denotes the set of the eligible physical routers to host the virtual node $v \in V(\mathcal{D})$. In fact, they satisfy the resource availability requirements like: i) processing power $B(v)$, ii) memory $M(v)$, iii) geographical location $\mathcal{L}(v)$, iv) router type $X(v)$ (i.e., access or core), etc. Formally, a substrate router $w \in V(\mathcal{G})$ can host a virtual router $v \in V(\mathcal{D})$ if it satisfies the following constraints:

$$\forall v \in V(w) \begin{cases} (B(w) - B(v))\, x_{vw} & \geq 0 \\ (M(w) - M(v))\, x_{vw} & \geq 0 \\ (X(w) - X(v))\, x_{vw} & = 0 \end{cases} \tag{4.1.11}$$

Concerning link mapping, each substrate link cannot violate its bandwidth capacities by satisfying requests requirements. Formally,

$$C(e) \geq 0, \forall e \in E(\mathcal{G}) \tag{4.1.12}$$

Basing on the aformentionned model and the problem constraints, it is straightforward to see that the virtual embedding process is a non-linear optimization problem. Indeed, it has been proved in [72] [73] that it is NP-hard problem. Consequently, computing the optimal solution in polynomial time is not guaranteed for a wide substrate network. To skirt this complexity, we propose new reliable virtual network mapping strategy basing on Game Theory denoted by `Advanced-CG-VNE`. Our proposed algorithm will be detailed in the next section.

4.2 Proposal: `Advanced-CG-VNE`

In this section, we present our proposal **Advanced Coordination Game for Virtual Network Embedding** (`Advanced-CG-VNE`) by describing its main stages. We recall that we modelize our mapping problem as an embedding game. Accordingly, we will explain how our proposal converges to the Nash Equilibrium of which we have proved the existence. In order to evaluate the payoff earned by a fictitious decision maker, the virtual link mapping is processed. Basing on the virtual link embedding approach (i.e., unsplittable or splittable), two variants are defined. The first one is based on simulated allocation technique [84] [85] and the other computes the optimal solution basing on Mixed Integer Linear Programming resolution.

4.2.1 Embedding of virtual routers

The following stage begins basing on an initial mapping (i.e., or a virtual node assignement). Then, the actions set for each fictitious player $\mathcal{N}_i$ (i.e., matching with a virtual) is defined. We recall that for a decision maker $\mathcal{N}_i$, its actions set $\mathcal{S}_{\mathcal{N}_i}$ includes the $\mathcal{H}$-hops neighborhood and randomly selected substrate routers. It is worth pointing out that by adding the random subset, we aim to avoid the local optima. In order to ensure the convergence to the Nash Equilibrium, we make use

of the **Best Response** algorithm [86]. At each turn, every player $\mathcal{N}_i$ runs the Best Response algorithm by selecting the strategy from its action set $\mathcal{S}_{\mathcal{N}_i}$ maximizing its payoff function. Once all decision makers select their best strategy, we evaluate the utility $\mathcal{U}$. The above is repeated untill we reach the Nash Equilibrium. It is worth noting that the order of players sequence is randomly computed at each turn. In this way, we urge the computing of the global optimum. Besides, the difference between the $\mathcal{U}$ values is computed at the end of each turn. If this gap is less than a predifined threshold ϵ_Γ then the Nash Equilibrium is reached. Otherwise, the above game will be continuously processed. In order to evaluate the relevance of the substrate router candidate, `Advanced-CG-VNE` simulates the mapping of its attached virtual links. For this aim, we propose two variants: i) `Advanced-CG-VNE-Unsplit` and ii) `Advanced-CG-VNE-Split`. We summarize our mapping game proposal in Algorithm 6.

Algorithm 6: `Advanced-CG-VNE`

Inputs: $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$, $\mathcal{D} = (V(\mathcal{D}), E(\mathcal{D}))$, ϵ_Γ
Output: Mapping of $\mathcal{D}$ in $\mathcal{G}$
Determine the initial embedding of $\mathcal{D}$
Build the player set of Γ: $\mathcal{N} \leftarrow V(\mathcal{D})$
$\mathcal{U}_0 \leftarrow \sum_{\mathcal{N}_i \in \mathcal{N}} \mathcal{U}^0_{\mathcal{N}_i}(\mathcal{N}_i)$
repeat
    Determine the action set $\mathcal{S}_{\mathcal{N}_i}$ of each player $\mathcal{N}_i \in \mathcal{N}$
    $\{\mathcal{N}_j\} \leftarrow$ TurnPlayers ($\mathcal{N}$)
    for $j \leftarrow 1$ *to* $|\mathcal{N}|$ **do**
        for *each* $\mathcal{A}_{j,k} \in \mathcal{S}_{\mathcal{N}_j}$ **do**
            Simulate the mapping of $\mathcal{N}_j$ in $\mathcal{A}_{j,k}$
            Process link mapping for $\mathcal{A}_{j,k}$ action
            Compute the payoff $\mathcal{U}_{\mathcal{N}_j}$ for $\mathcal{A}_{j,k}$ action
        Select the best candidate $\mathcal{A}^*_{j,k} \in \mathcal{S}_{\mathcal{N}_j}$ that maximizes the payoff function $\mathcal{U}_{\mathcal{N}_j}$ (The maximum is equal to $\mathcal{U}_{best}(\mathcal{N}_j)$)
    $\mathcal{U}_{best} \leftarrow \sum_{\mathcal{N}_j \in \mathcal{N}} \mathcal{U}_{best}(\mathcal{N}_j)$
    $\epsilon_{tmp} \leftarrow \mid \mathcal{U}_0 - \mathcal{U}_{best} \mid$
    $\mathcal{U}_0 \leftarrow \mathcal{U}_{best}$
until $\epsilon_{tmp} \leq \epsilon_\Gamma$;

First of all, `Advanced-CG-VNE` makes use of an other mapping strategy to compute the initial embedding configuration. It is worth pointing out that `Advanced-CG-VNE` requires only to set the initial positions of virtual routers set. Then, the relevance of this mapping is evaluated by the utility function value $\mathcal{U}_0$. Afterwards, `Advanced-CG-VNE` computes the actions set $\mathcal{S}_{\mathcal{N}_i}$ of each fictitious player $\mathcal{N}_i$. The players' turn (i.e., TurnPlayers procedure) are generated basing on a discrete uniform distribution. Once the players order is defined, every decision maker simulate the

mapping of the virtual router in the current candidate. Besides, Advanced-CG-VNE simulates the embedding of virtual links basing on one of the aforementioned variants: i) Advanced-CG-VNE-unsplittable and ii) Advanced-CG-VNE-Split. Each one of the fictitious player selects the strategy that maximizes its payoff function basing on Best Response algorithm. At the end of each turn the utility function $\mathcal{U}$ is evaluated. Thus, the Nash Equilibrium condition is checked. This mapping game (i.e., the above process) is repeated till the convergence to the Nash Equilibrium.

4.2.2 Embedding of virtual links

It is straightforward to notice the importance of the link mapping stage to evaluate the relevance of one substrate router candidate in our embedding game. For instance, one candidate may have a good reliability with a huge remaining resources, however, its attached physical links do not have enough bandwidth resources to host the virtual links of the fictitious player. That's why the link mapping stage is decisive. Hereafter, we will describe our two approaches to handle virtual link mapping basing on i) unsplittable and ii) splittable manner.

4.2.2.1 Advanced-CG-VNE-Unsplit

The unsplittable virtual link mapping can be formulated as the atomic (i.e., unsplittable) Multi Commodity Flow problem. Our proposal strongly relies on the simulated allocation technique [87]. We aim to i) load balance the bandwidth consumption and ii) minimize the total reserved (i.e., allocated) link resources by avoiding unnecessary load on bottleneck edges. These two objectives are achived by two key features: i) ordering and preprocessing the virtual link set and ii) adaptive establishment of paths with respect to their relative lengths. In fact, initially the set of virtual link $d \in E(\mathcal{D})$ is sorted basing on their difficulties h_d in decreasing order. Formally,

$$h_d = \frac{C(d)}{len\left(\text{Dijkstra}_d\right)} \tag{4.2.13}$$

Our algorithm maintains two sets: i) the set of actually unsatisfied demand Σ and ii) the set of processed paths Θ. During each step, our approach proceeds to : i) transfer a virtual link to an embedded substrate path or ii) remove path(s) back for later recomputing. Indeed, the former behaviour deals with allocation process and the latter is related deallocation one. It is worth noting that during the iterative path processing, the possible choices may vary basing on the substrate network status. At each turn, we take the top of Σ. Then we compute the path that maximizes equation 4.1.7. In other terms, the selected path P is the best one from a pre-computed paths basing on k-shortest path algorithm. It is noteworthy that we make use of the Yen's algorithm [88] to compute the k-shortest paths. If the embedding of d into P is feasible so the current virtual link (i.e., d) would be embedded into P and both Σ and Θ would be updated. The latter process matches with the allocation stage. Otherwise, the deallocation stage will be executed. First, bottleneck substrate

edges E_B will be identified. E_B is build basing on the substrate edges that i) has not enough bandwidth resources to host d and ii) belongs to one path candidate for the embedding of demand d. Second, the candidate paths $\mathcal{P}_{cand}$ that will be removed from Θ and also released from $\mathcal{G}$ will be computed. Formally $\mathcal{P}_{cand}$ is defined as follows,

$$\mathcal{P}_{cand} = \{P_i \in \Theta | P_i \cap E_B \neq \emptyset\} \tag{4.2.14}$$

Afterwards, a "tournament of deallocation" is processed to select a subset of $\mathcal{P}_{cand}$. The winner of this tournament are the paths having maximum of "badness criterium" B_c. It is noteworthy that this criterium is defined for a path P_j and its corresponding demand d as:

$$B_c\left(P_j\right) = \frac{\mathit{len}\left(P_j\right) - \mathit{len}\left(\text{Dijkstra}_d\right)}{C\left(d\right)} \tag{4.2.15}$$

Beside of this deterministic selection, a random paths are chosen. It is worth pointing out that the number of deallocated paths from $\mathcal{P}_{cand}$ set is variable. In other words, it increases when deallocation stage is continuously repeated and it decreases when the algorithm finds feasible paths. We summarize the atomic multicommodity flow strategy in Algorithm 7.

Algorithm 7: `Unsplittable MCF`

```
1  Inputs: E(D) , G
2  Output: Mapping of E(D) in G
3  Θ ← ∅
4  Σ ← sort E(D) in decreasing order basing on h_d
5  repeat
6      take the first demand d ∈ Σ
7      P ← computePath(d , G)
8      if P is feasible then
9          Map d in P
10         Remove d from Σ
11         Add P to Θ
12     else
13         Identify bottleneck edges set E_B
14         P_cand ← {P_i ∈ Θ | P_i ∩ E_B ≠ ∅}
15         Remove paths from P_cand
16         Update the two lists Θ and Σ
17 until Σ = ∅ ;
18 return Θ
```

4.2.2.2 Advanced-CG-VNE-Split

In this approach, we dispatch the required bandwidth into a set of paths between the two substrate routers hosting the two extremities of the virtual link. Thanks to the flexibility to split the requested bandwidth among many substrate paths, the splittable link embedding enhances $\mathcal{CP}$'s turnover by accepting more $\mathcal{VN}$s. We modelize the splittable virtual link embedding problem as a Mixed Integer Linear Programming (MILP). Based on this model, the mapping of all the attached virtual links to a virtual router is processed at once in polynomial time. It is worth pointing out that we converge to the optimal solution that matches with the social optimum. This social optimum ensures the substrate network load balancing basing on equation 4.1.8.

Moreover, the solution must respect the capacity constraint to avoid the violation of substrate links' capacity. Formally,

$$\sum_{d\in\{d_j^i\}} \sum_{\mathcal{P}\in \textit{Paths-all}(d)} \left[\delta_e\left(\mathcal{P}\right)\cdot b(\mathcal{P},d)\right] \leq C(e), \forall e \in E\left(\mathcal{G}\right) \tag{4.2.16}$$

where $\delta_e\left(\mathcal{P}\right)$ is the link-path indicator binary variable. $\delta_e\left(\mathcal{P}\right) = 1$ if the substrate link e belongs to the substrate path $\mathcal{P}$. Otherwise, $\delta_e\left(\mathcal{P}\right) = 0$. We recall that i) $\{d_j^i\}$ is the set of $\mathcal{SC}_i$'s virtual hanging links and ii) $b\left(\mathcal{P},d\right)$ is the allocated bandwidth for virtual link d in path $\mathcal{P}$.

Finally, the solution must respect the requested bandwidth of hanging virtual links. In fact, the total allocated bandwidth assigned to a virtual link d among all the candidate substrate paths must be equal to the requested one. Formally,

$$\sum_{\mathcal{P}\in \textit{Paths-all}(d)} b\left(\mathcal{P},d\right) = C(d), \quad \forall d \in \{d_j^i\} \tag{4.2.17}$$

Hereafter, we will summarize the MILP optimization problem of reliable virtual links embedding.

$$\textbf{maximize} \left\{ \sum_{d\in E(\mathcal{D})} \sum_{\mathcal{P}\in \texttt{Paths-all}(d)} \left[\left(\frac{\texttt{len}\left(\text{Dijkstra}_d\right)}{\texttt{len}\left(\mathcal{P}\right)} \right) \cdot \mathcal{R}^p\left(\mathcal{P}\right)\cdot \bar{C}\left(\mathcal{P}\right)\right] \right\}, \mathcal{P} \in \texttt{Paths-all}(d)$$

$$\texttt{subject to:}$$

$$\sum_{d\in\{d_j^i\}} \sum_{\mathcal{P}\in \texttt{Paths-all}(d)} \left[\delta_e\left(\mathcal{P}\right)\cdot b(\mathcal{P},d)\right] \leq C(e), \forall e \in E\left(\mathcal{G}\right)$$

$$\sum_{\mathcal{P}\in \texttt{Paths-all}(d)} b\left(\mathcal{P},d\right) = C(d), \quad \forall d \in \{d_j^i\}$$

Problem 3: Splittable reliable virtual links embedding problem

The above MILP problem can be easily solved thanks to standard optimization solver such as CPLEX or GLPK. The convergence time to **the optimal solution** is polynomial [79]. We recall this above gain function value will be used to evaluate the relevance of one router candidate mapping.

4.2.3 Reliability support

`Advanced-CG-VNE` deals with survivability basing on two stages. First, during the mapping process our proposal urges the use of reliable physical resources. In fact, `Advanced-CG-VNE` tries to avoid the embedding in non-reliable resources in order to avert physical failures' impact. It is straightforward to note this **preventive** mechanism from the payoff function in the mapping game. Besides, the link mapping objectives ensure this preventive approach. Consequently, `Advanced-CG-VNE` prevents the service interruption by avoiding the use of non-reliable physical resources (i.e., links and/or routers). Second, once a failure impacts one $\mathcal{VN}$ (or more), `Advanced-CG-VNE` tries to re-embed the affected virtual resources (i.e., virtual routers and/or links) basing on the same mapping process described earlier. This **reactivity** guarantees a better usage of the available physical resources. In fact, the advantage of a reactive mechanism comparing to a proactive one that relies on a dedicated backup is that no extra resources are allocated to a $\mathcal{VN}$. But, additional substrate resources would be used only for the re-mapping process. In this way, we avoid the overprovisioning that can harm the acceptance of new clients requests. In order to guarantee the recovery, we devise a central physical controller [89] which keeps checkpoints of all virtual routers saved in its memory. Accordingly, this central unit can ensure the migration of tasks from the failed physical equipments to the new ones hosting the impacted virtual resources.

4.3 Performance evaluation

Hereafter, we will evaluate the performance of our proposal `Advanced-CG-VNE` by comparing it to prominent related work. For this aim, first, we describe our virtual network discrete event simulator. Note that heterogeneity of substrate equipments is taken into account by considering different ages. Accordingly, reliability will not be same for all physical hardware in $\mathcal{SN}$. Second, we will describe the performance metrics we consider to assess the effeciency of our proposal. Finally, we will present and comment the obtained results.

4.3.1 Simulation environment

We implemented a discrete event reliable $\mathcal{VN}$ mapping simulator. Like [52] we make use of `GT-ITM` [80] tool to generate random $\mathcal{SN}$ and $\mathcal{VN}$ topologies. Note that we model the i) $\mathcal{VN}$ requests' arrival by a Poisson Process with rate λ_A and ii) $\mathcal{VN}$ lifetime by exponential distribution with mean μ_L. The number of $\mathcal{VN}$ requests is fixed to 2000. As in [52], we set the size of our $\mathcal{SN}$ to 100 (i.e., 100 routers). In this case, the ratio of core and of access routers are respectively set to 80% and 20%. Furthermore, we randomly set $\mathcal{VN}$ size basing on a discrete uniform distribution taking values between $[3, 10]$. Similar to the $\mathcal{SN}$, the proportions of access and core virtual routers are fixed respectively to 20% and 80%. It is worth mentioning that for $\mathcal{VN}$ and $\mathcal{SN}$ each pair of

Table 4.1: Initial age of $\mathcal{SN}$'s equipments

Group	Min age	Max age	Failure probability
Young	0	50 000	[0, 0.1]
Adult	200 000	230 000	[0.5, 0.6]
Old	400 000	500 000	[0.85, 0.97]

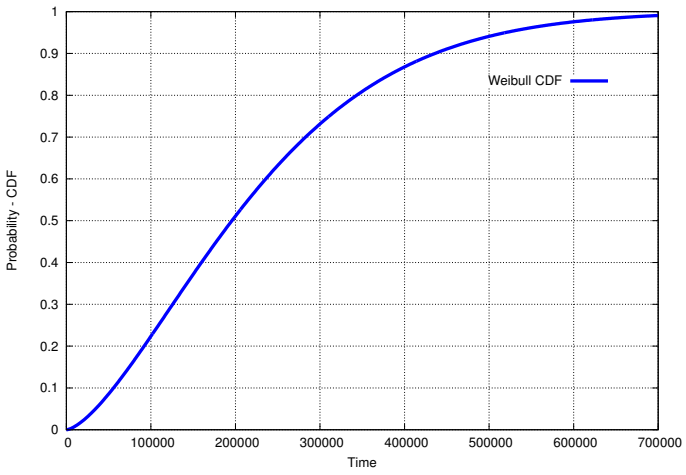


Figure 4.1: CDF of MTBF Weibull distribution

routers is randomly connected with a probability of 0.5. The arrival rate λ_A and the average lifetime μ_L of $\mathcal{VN}$s are respectively set to 4 requests per 100 and 1000 time units. Besides, we set the capacity of substrate routers and links (i.e., $B(w)$, $M(w)$ and $\hat{C}(e)$) according to a continuous uniform distribution taking values in $[50, 100]$. Moreover, we fix the required virtual resources (i.e., $B(v)$, $M(v)$ and $\hat{C}(d)$) based on a continuous uniform distribution, using values between $[10, 20]$.

Regarding $\mathcal{SN}$ reliability, the parameters a and b of the Weibull distribution (see equation 4.1.1) modeling the MTBF are respectively set to 25×10^4 and 1.5. It is noteworthy that a and b are calibrated in such a way to obtain a lifetime for each physical equipment ($\approx 6 \times 10^5$) roughly equal to 10 times the simulation duration ($\approx 6 \times 10^4$ time units). Furthermore, the proportion of young equipments in the $\mathcal{SN}$ is fixed to 30% for simulation scenarii. In our simulations, we will vary the rate of adult (i.e., medium) equipments in the $\mathcal{SN}$ and consequently the old proportion. Moreover, the initial age of physical equipment follows a uniform distribution within the defined bounds of its group (i.e., young, adult or old). The initial age bounds are summarized in Table 4.1. The failure probabilities on each group are computed basing on the Weibull distribution. Figure 4.1 illustrates the Cumulative Distribution Function CDF modeling the Mean Time Between failure according to the aformentionned parameters value (i.e., a and b).

In order to compare `Advanced-CG-VNE`, we implemented the most prominent survivable

Table 4.2: Related survivable $\mathcal{VN}$ embedding strategies

Algorithm	Reference	Link Mapping
RMap	[65]	Unsplittable
CPP	[67]	Unsplittable
VNP	[67]	Unsplittable
SVNE	[64]	Splittable
ORP	[60]	Splittable
K-Redundant	[63]	Splittable

$\mathcal{VN}$ embedding strategies found in the literature. These algorithms are summarized in Table 4.2. It is worth noting that the confidence level of simulation results is fixed to 95% using 30 samples for each value.

4.3.2 Performance metrics

Hereafter, we detail the performance metrics that we use to gauge Advanced-CG-VNE.

- Q is the reject rate of $\mathcal{VN}$ requests during the simulation period. In other words, the rate of $\mathcal{VN}$s that Advanced-CG-VNE could not map in the $\mathcal{SN}$ because of resource shortage.

- F is the rate of embedded $\mathcal{VN}$s that have been impacted by hardware failures during the simulation (i.e., blackout rate). In fact, a $\mathcal{VN}$ is impacted if at least one of its virtual components (i.e., virtual router and/or link) is affected by a physical outage. F is the ratio between the number of impacted $\mathcal{VN}$s per the number of accepted $\mathcal{VN}$ requests.

- G describes the $\mathcal{CP}$'s profitability by calculating the total benefit G_1 of all accepted $\mathcal{VN}$ requests minus the paid penalties G_2 to the clients induced from the physical failures during all the simulation lifetime ($\mathrm{G} = \mathrm{G}_1 - \mathrm{G}_2$). G_1 is defined as in [55]. It depends on the amount of requested resources and the lifetime of the virtual network in the $\mathcal{CP}$'s backbone network. We define the penalty G_2 as the reimbursement to the clients due to service's interruption. The cost is proportional to the remaining lifetime of a $\mathcal{VN}$ heightened by a penalty δ. Formally, for each $\mathcal{VN}$ denoted by $\mathcal{D}$:

$$\mathrm{G}_2(\mathcal{D}) = \frac{(T-\Delta_T)}{T} \times \mathrm{G}_1(\mathcal{D}) \times \delta \tag{4.3.18}$$

  where Δ_T is the hosting period in the $\mathcal{SN}$ before the physical failure and δ is the penalty rate. In our simulations, we set the penalty to 5%.

4.3.3 Simulation results

In section 4.1.1, we mention that two variants of `Advanced-CG-VNE` are proposed basing on the virtual link mapping strategy: unsplittable or splittable. Consequently, we first evaluate the performance of `Advanced-CG-VNE-Unsplit`. Then, we gauge the second variant `Advanced-CG-VNE-Split`. Finally, we conclude this section by a discussion regarding the obtained results and enumerating also the pros and cons of each variant.

4.3.3.1 Unsplittable virtual link embedding approach

Hereafter, we assess the performance of `Advanced-CG-VNE-Unsplit` by comparing it to the related competitor algorithms adopting the atomic approach to embed virtual links: i) `RMap`, ii) `CPP` and iii) `VNP` (see Table 4.2).

First, our evaluation focuses on the reject rate metric Q. It is straightforward to see in Figure 4.2 that our proposal outperforms the related work whatever the age of $\mathcal{SN}$'s equipments. We recall that the proportion of young equipments is fixed to 30% and the rate of adult (and consequently the old) equipments is varying. It is clear that the reject rate of $\mathcal{VN}$ requests decreases when the proportion of adult equipments increases (i.e., proportion of adult equipments decreases) thanks to the rejuvenation of the $\mathcal{SN}$. In fact, having more adult proportion is equivalent to increase the reliability of hardware resources. Consequently, $\mathcal{SN}$ will loose less physical equipments. For instance, if the rate of adult equipments in the $\mathcal{SN}$ is equal to 40% (i.e., proportion of old 30%), `Advanced-CG-VNE-Unsplit` rejects only $36.65 \pm 2.66\%$. However, the other algorithms reject at least $61.91 \pm 0.21\%$ of requests. It is worth mentioning that if the proportion of adult resources is 20% (i.e., rate of old equipment is equal to 50%), the reject rate is important for all algorithms. But, our proposal is always better than the related strategies. This high value is explained by the huge number of hardware outages that directly impact the $\mathcal{SN}$'s connectivity and the availability of physical resources. The loss of the related strategies compared to `Advanced-CG-VNE-Unsplit` is presented in Figure 4.3. It is straightforward to see that the loss is always negative. It is trivial to conclude this, since `Advanced-CG-VNE-Unsplit` reject less than the competitor algorithms for all proportions.

Second, we illustrate the rate of the impacted $\mathcal{VN}$s by the substrate failures in Figure 4.4. We notice that `Advanced-CG-VNE-Unsplit` minimizes the rate of $\mathcal{VN}$ impacted by physical failures. This result is thanks to the reactive mechanism that `Advanced-CG-VNE-unsplit table` adopts. In fact, once a physical failure impacts some virtual resources (i.e., routers and/or links) our algorithm re-embeds these impacted resources into other substrate equipments. This reactivity allows an optimized management of the physical resources available in the $\mathcal{SN}$ by avoiding the reservation of a dedicated backup that may not be used. Besides, the mapping process adopted by `Advanced-CG-VNE-Unsplit` optimizes the physical resources usage. Accordingly, we

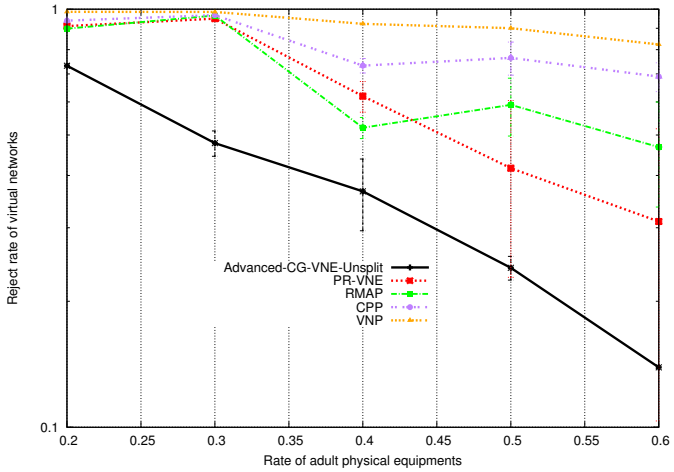


Figure 4.2: Q - Unsplittable based approaches

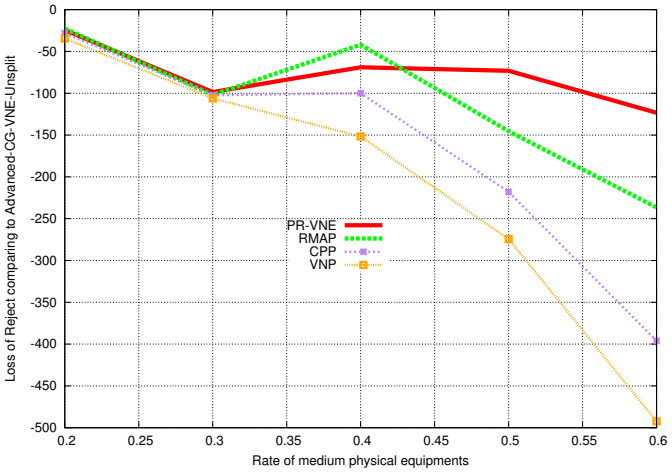


Figure 4.3: Loss of reject rate Q compared with `Advanced-CG-VNE-Unsplit`

have more available resources to re-map impacted virtual routers and links. It is worth pointing out that basing on the earlier result, `Advanced-CG-VNE-Unsplit` accepts more clients than the other algorithms. So, there is high probability to impact more clients since in one impacted substrate equipment the number of hosted virtual resources is greater than other algorithms. Whereas, thanks to the reactive mechanism `Advanced-CG-VNE-Unsplit` has the minimum blackout rate. In fact, it is straightforward to see this in Figure 4.5.

Finally, we evaluate the $\mathcal{CP}$'s gain (i.e., revenue) G in Figure 4.6. We recall that the latter metric is directly proportional to the number of accepted $\mathcal{VN}$s and their requested resources. Fur-

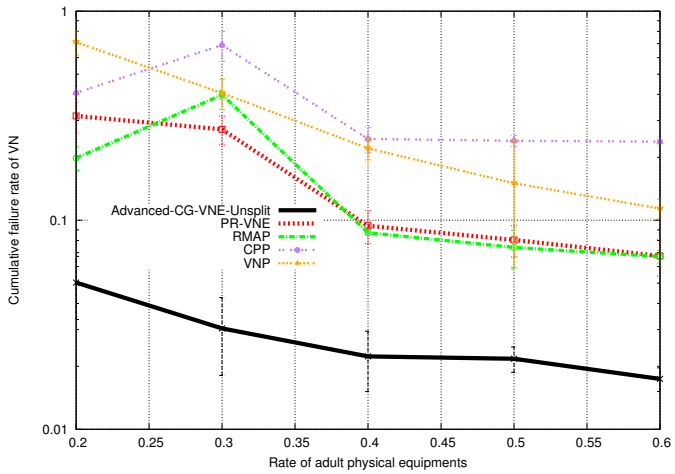


Figure 4.4: F - Unsplittable based approaches

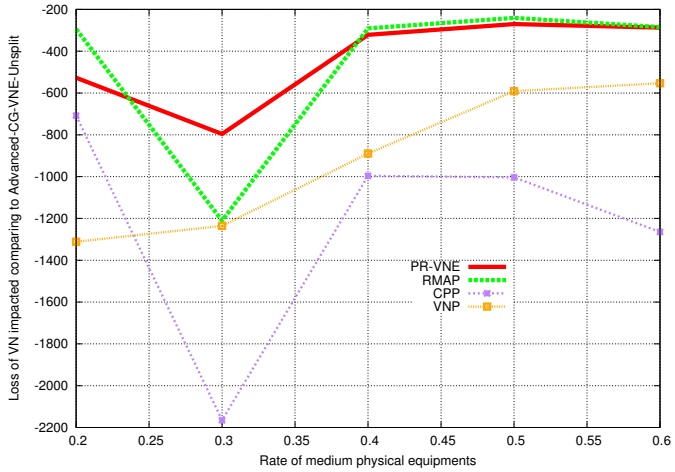


Figure 4.5: Loss of blackout rate F compared with `Advanced-CG-VNE-Unsplit`

thermore, G is inversely proportional to penalties du to SLA violation caused by physical failures. It is straightforward to see that our proposal `Advanced-CG-VNE-Unsplit` outperforms all the related strategies. This logic result can be explained by i) the maximization of acceptance rate of $\mathcal{VN}$s (see Figure 4.2) and ii) the minimization of blackout rate of $\mathcal{VN}$s (see Figure 4.4). In Figure 4.7, we illustrate the loss of revenue compared to `Advanced-CG-VNE-Unsplit`. For instance, the gain of our strategy is greater at least 20 times than the related work.

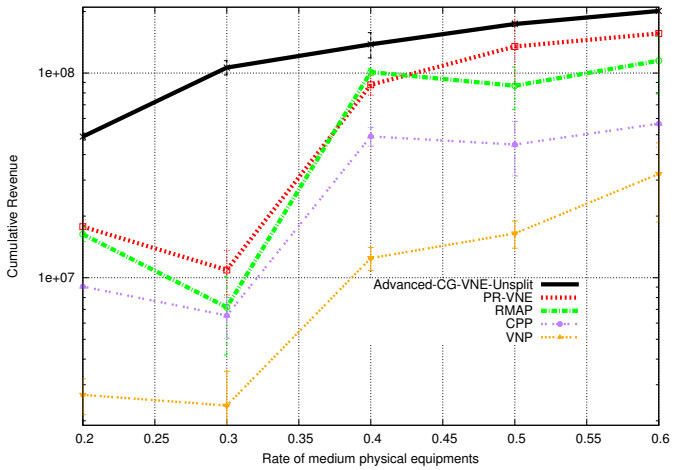


Figure 4.6: G - Unsplittable based approaches

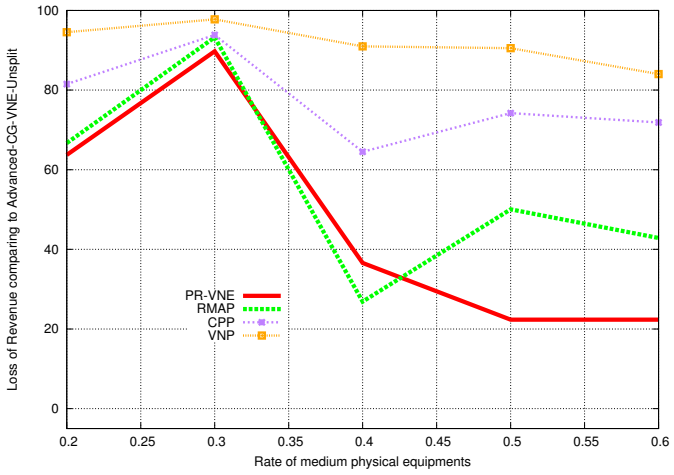


Figure 4.7: Loss of $\mathcal{CP}$'s gain G compared with `Advanced-CG-VNE-Unsplit`

4.3.3.2 Splittable virtual link embedding approach

Hereafter, we evaluate the second variant (i.e., `Advanced-CG-VNE-Split`) performance. For this aim, we compare it to the related splittable-based approaches: i) `SVNE`, ii) `K-Redundant` and iii) `ORP` (see Table 4.2).

In Figure 4.8, we focus on the reject rate Q of $\mathcal{VN}$s. We notice that `Advanced-CG-VNE-splittable` outperforms all the related strategies whatever the age of the $\mathcal{SN}$'s equipment. However, all strategies have roughly the same results when the adult proportion is 20% (i.e., old rate is 50%). Even for this scenario, `Advanced-CG-VNE-Split` slightly outperforms the remaining

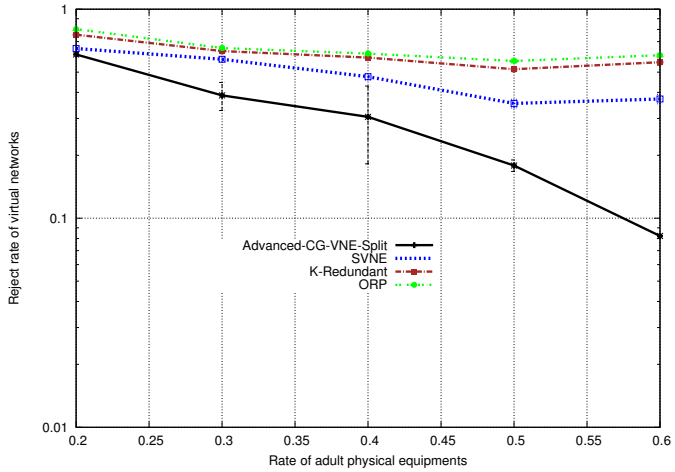


Figure 4.8: Q - Splittable based approaches

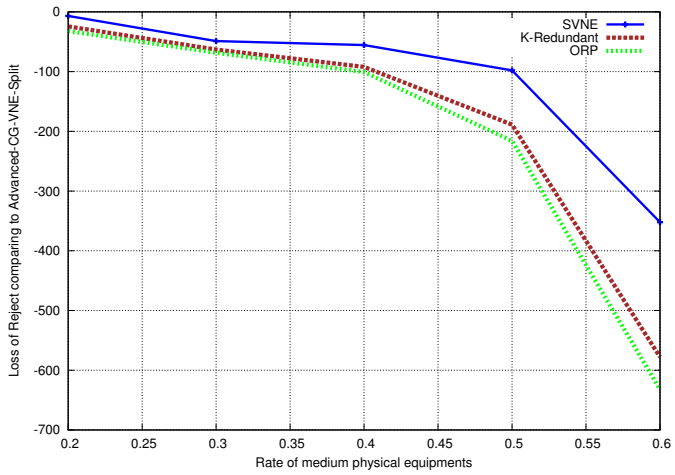


Figure 4.9: Loss of reject rate Q compared with `Advanced-CG-VNE-Split`

approaches. In fact, for this proportion of old equipments (50%) the majority of the $\mathcal{SN}$ falls down quickly because of physical crashes. Accordingly, these algorithms have not enough physical resources to host new clients. On the other hand, `Advanced-CG-VNE-Split` rejects less than 8% when the adult proportion is 60%. These good results can be justified basing on two facts. The first is related to the non-backup reservation. And the second concerns the fair management of the physical resources. In Figure 4.9, we illustrate the loss rate compared to our proposal in terms of reject rate resulted with the related strategies. It is straightforward to observe that with our proposal Q decreases as the age of the $\mathcal{SN}$ rejuvenates.

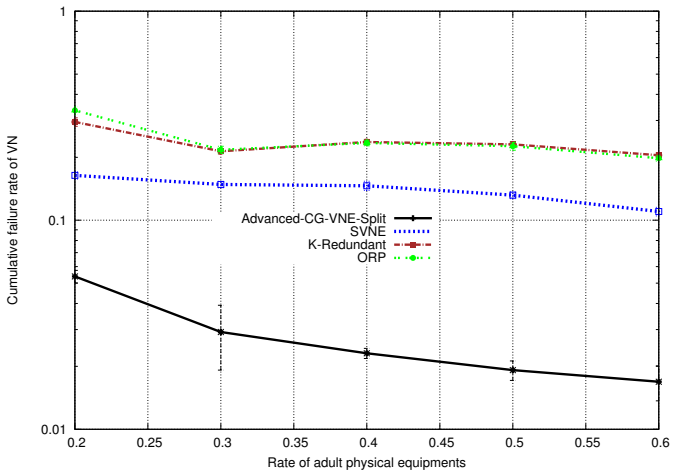


Figure 4.10: F - Splittable based approaches

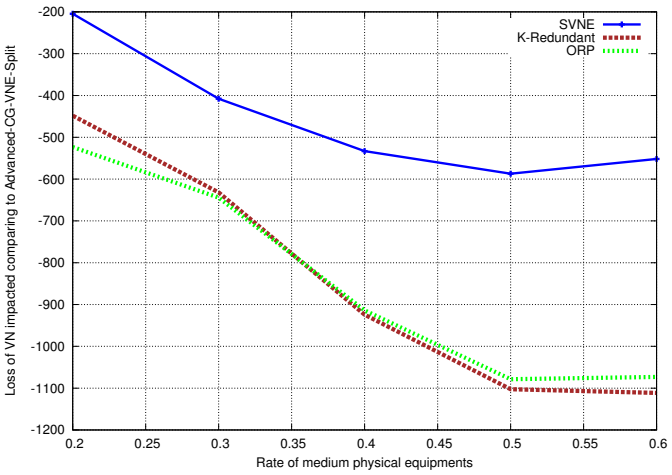


Figure 4.11: Loss of blackout rate F compared with `Advanced-CG-VNE-Split`

In Figure 4.10 we asses the blackout rate F of $\mathcal{VN}$s caused by physical failures. It is clear that `Advanced-CG-VNE-Split` has always the minimum rate of $\mathcal{VN}$ impacted comparing to the related splittable strategies. This result shows that `Advanced-CG-VNE-Split` adopts a good strategy to handle physical failures in the $\mathcal{SN}$. In fact, thanks to the preventive approach we try to avoid as maximum the failures impact. Second, basing on the reative mechanism impacted resources are re-embedded into other physical equipments. Indeed, Figure 4.10 and Figure 4.11 show that reactive method is not only better than the proactive one in term of resource consumption but also in term of failure recovery.

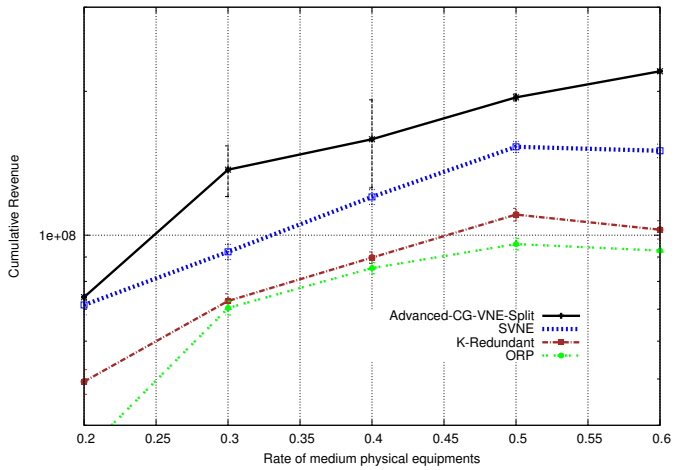


Figure 4.12: G - Splittable based approaches

Figure 4.12 illustrates $\mathcal{CP}$'s gain G. As a logical consequence of the studied metrics (i.e., Q and F), `Advanced-CG-VNE-Split` guarantees the best turnover to the $\mathcal{CP}$. In other words, `Advanced-CG-VNE-Split` ensures the best revenue since it i) maximizes the acceptance rate and ii) minimizes the penalties caused by hardware crashes. We notice that the $\mathcal{CP}$'s revenue increases as the $\mathcal{SN}$ become youngest. This is because when the $\mathcal{SN}$ becomes younger the remaining lifetime of physical resources is longer. Hence, there are more available substrate resources. Besides, the substrate failures will be less frequent, so the penalties would be minimized. In fact, Figure 4.13 supports this point and shows clearly that `Advanced-CG-VNE-Split` is the best choice to the $\mathcal{CP}$ to enhance the benefit comparing to related work: i) `SVNE`, ii) `K-Redundant` and iii) `ORP`.

4.3.3.3 Summary

We evaluate the two variants of our proposed algorithm `Advanced-CG-VNE`. We reacll that the unique difference is the link mapping strategy. In fact, the first one `Advanced-CG-VNE-unsplit table` and the second is `Advanced-CG-VNE-Split`. Basing on simulation results presented above, it is clear that the splittable approach has better performances than the unsplittable one in terms of i) reject rate, ii) blackout rate and iii) $\mathcal{CP}$'s turnover. These simulation results are in concordance with the logical expectations. In fact, the embedding of each virtual link within multiple substrate paths in the $\mathcal{SN}$ ensures more flexibility to select the most reliable resources and favors the avoidance of bandwidth congestion.

Despite of this efficiency, the deployed routers in the current Cloud's backbone do not support the splittable mapping. Thus, making use of a splittable strategy remains impossible till a probable

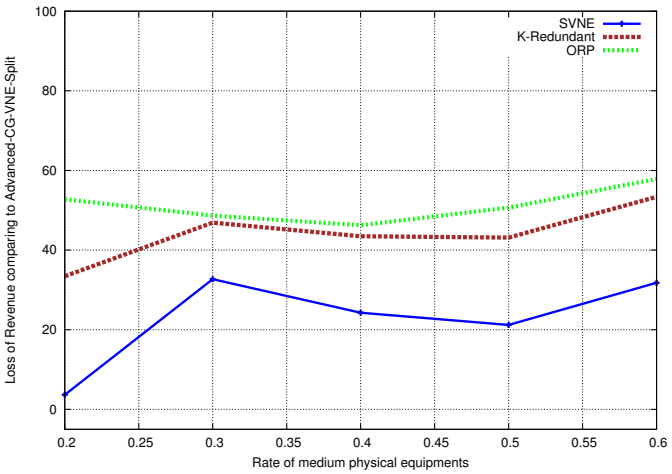


Figure 4.13: Loss of $\mathcal{CP}$'s gain G compared with `Advanced-CG-VNE-Split`

upgrade of the backbone's routers in the near future. It should be highlighted that this technological migration to a new generation of routers will dramatically increase the provider's expenses. Accordingly, the unsplittable approach is highly recommanded to be adopted in order to deal with the reliable virtual network embedding problem right now.

4.4 Conclusion

In this chapter, we tackled the survivable $\mathcal{VN}$ embedding problem typified as NP-hard. Beside the virtual network complexity, the survivability issue makes the problem more difficult. Accordingly, we proposed a new reactive algorithm named `Advanced-CG-VNE` based on game theory. To the best of our knowledge, we are the first to deal with the reactivity in $\mathcal{VN}$ environment in order to handle physical failures. It is noteworthy that our proposal is a preventive approach and does not allocate any backup resources while guaranteeing better performance. The main idea behind our proposal is to promote the use of reliable physical resources during the virtual network embedding process and to re-map the impacted virtual resources by failures. We propose two variants according to the virtual link mapping strategy: i) unsplittable and ii) splittable. Based on extensive simulations, our proposal outperforms the most prominent related strategies in terms of: i) reject rate of virtual networks, ii) blackout rate of virtual networks and iii) Cloud provider's revenue.

In the next chapter, we will focus on the macrospic view where the Cloud provider collects the arrived requests during a defined time window in order to simultaneously handle this set of $\mathcal{VN}$s. In fact, this batch mapping alleviates the provider from a frequent access to the $\mathcal{SN}$ and allows to improve the decision concerning the requests which should be accepted in order to maximize the

revenue. To the best of our knowledge, our proposal named BR−VNE is the first **batch** embedding algorithm taking into consideration the reliability and the physical outages.

Chapter 5

A Batch approach for a survivable virtual network embedding

Contents

5.1 Problem formalization 103
5.1.1 Substrate and virtual networks models 103
5.1.2 Reliable batch-embedding $\mathcal{VN}$ problem 104
5.1.3 Reliable embedding of one virtual network problem 107
5.2 Proposal: BR−VNE strategy 107
5.2.1 Tree search optimization algorithms 108
5.2.2 BR−VNE description 109
5.3 Performance evaluation 111
5.3.1 Simulation environment 111
5.3.2 Performance metrics 112
5.3.3 Simulation results 113
5.4 Conclusion 124

In the previous contributions, we only tackle the online reliable virtual network embedding problem within Cloud's backbone network. Despite the good results obtained for both proposals, our study is not complete due to the absence of a batch consideration. In fact, due to some technical limitations providers do not opt for online algorithms. Indeed, the batch mapping is prefered in the aim to minimize the access to the substrate network as well as to select the most prominent requests to be embedded in order to maximize the turnover.

In this chapter, we tackle the survivable $\mathcal{VN}$ **batch-embedding** problem within the $\mathcal{SN}$ of Cloud's backbone. It is worth noting that in existing literature, almost all the $\mathcal{VN}$ embedding strategies consider the **online** approach which consists in handling a $\mathcal{VN}$ request as soon as it arrives. However, a $\mathcal{CP}$ prefers to periodically handle a bundle of incoming $\mathcal{VN}$ requests in order to i) efficiently exploit the residual resources in the $\mathcal{SN}$ and ii) minimize the frequency of configurations in the Cloud's backbone and hence minimize the service outage probability. In fact, a $\mathcal{CP}$ queues all the incoming client's requests during a predefined time-window, then survivable batch-embedding $\mathcal{VN}$ algorithm will be performed. In doing so, $\mathcal{CP}$ alleviates the high frequency of hardware resources' interruptions which deeply deteriorates the performance of the equipments in the $\mathcal{SN}$.

Besides, dealing with a batch-mapping of $\mathcal{VN}$s may considerably improve the $\mathcal{CP}$' revenue. Indeed, the macroscopic vision of $\mathcal{VN}$s set is exploited by the $\mathcal{CP}$ to select a sub-set of $\mathcal{VN}$ requests maximizing the global turnover. Note that the elected $\mathcal{VN}$s to be embedded do not necessarily follow their arrival time order. Ideally, $\mathcal{CP}$ should embed all received $\mathcal{VN}$s, whereas this may not be feasible sometimes due to the limited residual resources of substate equipments in the Cloud's backbone.

In this chapter, we put forward a novel **Batch Reliable Virtual Network Embedding** (`BR-VNE`) strategy within Cloud's backbone dealing with the survivable mapping of incoming $\mathcal{VN}$ requests during a predefined time-window. The main objectives of `BR-VNE` consists in maximizing the $\mathcal{CP}$'s revenue by i) maximizing the acceptance rate of $\mathcal{VN}$ requests, ii) selecting the $\mathcal{VN}$s generating most profit and iii) minimizing the penalties impacted by physical failures in the Cloud's backbone. Due to the finite resources in the $\mathcal{SN}$, `BR-VNE` defines the sequence of $\mathcal{VN}$s that maximizes the cumulative revenue of $\mathcal{CP}$.

It is straightforward to see that the simplistic way to get the best sequence is to compute all possible combinations (i.e., sequences) of incoming $\mathcal{VN}$ requests. Whereas, this approach is computationally intractable. In order to skirt this exponential complexity, we formulate the search space by a decision tree which will be incrementally built. In the decision tree, each node models the mapping of one $\mathcal{VN}$ request. Hence, a branch in the tree depicts a sequence of $\mathcal{VN}$ embedding requests. Our objective is the generation of the optimal branch in the tree that maximizes the $\mathcal{CP}$'s revenue. Unfortunately, the construction of the whole decision tree is not conceivable, since the size of $\mathcal{VN}$ sequences is exponential. Moreover, the search process in the decision tree to find the optimal branch is computationally intractable. Beside the aforementioned exponential complexity, the mapping of **one** $\mathcal{VN}$ in each node in the decision tree is NP-hard [72] [73]. In summary, the generation of the optimal sequence is a combinational optimization problem which is NP-complete and the mapping of $\mathcal{VN}$ in each node in the sequence is NP-hard.

To overcome the above complexity, our proposal relies on the Monte-Carlo Tree Search [43] optimization method. `BR-VNE` incrementally builds the $\mathcal{VN}$ embedding decision tree and searches for the **best** branch (i.e., sequence of $\mathcal{VN}$s). Initially the decision tree contains only the root node

modeling the state of no $\mathcal{VN}$ request embedded in the $\mathcal{SN}$. BR−VNE operates as following.

First, BR−VNE selects the node among the nodes in the tree that can generate a child and simultaneously satisfy three conditions: i) its father cannot generate more children, ii) its election metric is the maximum among its brothers and iii) recursively the latter rule is satisfied for each ancestor until reaching the root. Note that the election metric controls the balance between i) exploitation of promoting branches and ii) exploration dealing with less promising branches. Then, a child (i.e., $\mathcal{VN}$ request) of the elected node is randomly generated (i.e., embedding simulation of $\mathcal{VN}$) and this process is repeated until i) all the $\mathcal{VN}$ requests are mapped or ii) the embedding process is blocked due to the congestion in the $\mathcal{SN}$. Next, BR−VNE enhances the decision tree with the newly created sub-branch. Afterwards, BR−VNE evaluates the cumulative revenue induced by the new sub-branch starting from the root of the tree to the leaf of the added sub-branch. Finally, the election metric of all the nodes in the latter branch is updated with respect to the accumulative earned income and the visit frequency. The same process is recursively repeated until the predefined computational budget is expired. The selected $\mathcal{VN}$ requests which will be mapped in the Cloud's backbone consist in the $\mathcal{VN}$ forming the branch **maximizing** the cumulative revenue. BR−VNE is validated by extensive simulations and compared with the most prominent related strategies found in existing literature. The results obtained show that our proposal outperforms related approaches in terms of i) rejection rate of $\mathcal{VN}$ requests, ii) rate of $\mathcal{VN}$s impacted by physical failures and iii) $\mathcal{CP}$'s revenue.

The remainder of this chapter is organized as follows. In Section 5.1, we will formulate $\mathcal{VN}$ and $\mathcal{SN}$ models as well as the reliable batch-mapping of $\mathcal{VN}$s. Subsequently, Section 5.2 will describe the details of our proposal BR−VNE. Afterwards, simulation environment and performance evaluation will be presented in Section 5.3. Finally, Section 5.4 will conclude the chapter.

5.1 Problem formalization

In this section, we will describe the reliable batch-embedding $\mathcal{VN}$ problem within Cloud's backbone network. To this aim, we will first define, as in the previous chapters, the substrate and virtual network models. Then, we will formulate the $\mathcal{VN}$ mapping problem.

5.1.1 Substrate and virtual networks models

The $\mathcal{SN}$ is formulated as an undirected weighted graph, denoted by $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ where $E(\mathcal{G})$ and $V(\mathcal{G})$ are respectively the sets of substrate links and routers. Each physical router, $w \in V(\mathcal{G})$, is characterized by its i) remaining memory capacity $M(w)$, ii) residual processing power (i.e., CPU) $B(w)$, iii) type: core or access $X(w)$, and iv) reliability $\mathcal{R}^n(w,t)$ at time t. It is worth pointing out that if $X(w) = 0$, then w is a core substrate router. Otherwise, w is an access physical router (i.e., $X(w) = 1$). Each physical link (i.e., $e \in E(\mathcal{G})$) is characterized by its i) initial bandwidth capacity $\hat{C}(e)$, ii) remaining bandwidth $C(e)$, and iii) reliability $\mathcal{R}^l(e,t)$ evaluated at

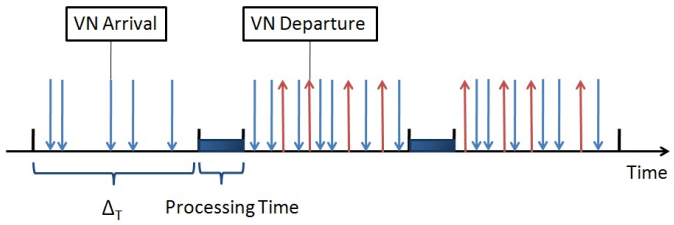


Figure 5.1: Arrival and processing of $\mathcal{VN}$ requests

time t.

We assume the same model as [71] of Mean Time Between Failures (MTBF) in the $\mathcal{SN}$ which follows the Weibull distribution. Accordingly, the equipment's reliability at any time can be estimated. The Cumulative Distribution Function (CDF) of the MTBF is expressed as:

$$F(x) = 1 - \exp(-\left(\frac{x}{a}\right)^b),\ x \geq 0 \tag{5.1.1}$$

where a and b are specific parameters to the Weibull distribution. It is noteworthy that we consider heterogeneous ages classes in $\mathcal{SN}$. Hence, each substrate resource in $\mathcal{G}$ (i.e., physical router or link) has its initial age denoted by $A(w)$ and $A(e)$ respectively for physical router w and physical link e. Consequently, in the $\mathcal{SN}$ we distinguish three groups of physical resources: i) **young**, ii) **adult** and iii) **old**. Therefore, the reliability of a i) substrate link e is equal to $\mathcal{R}^l(e,t) = F(A(e) + t)$ and ii) substrate router w is equal to $\mathcal{R}^n(w,t) = F(A(w) + t)$. It is worth pointing out that the physical equipment's reliability is inversely proportional to its age. Thus, the probability of equipment's outage increases with time.

A $\mathcal{VN}$ request is also modeled as an undirected graph, denoted by $\mathcal{D} = (V(\mathcal{D}), E(\mathcal{D}))$ where $V(\mathcal{D})$ and $E(\mathcal{D})$ are respectively the sets of virtual routers and links. Each virtual router, $v \in V(\mathcal{D})$, is typified by its i) requirement in term of memory $M(v)$ and processing power $B(v)$ and ii) its type: access or core $X(v)$. Note that if $X(v) = 1$, then v is an access virtual router. Otherwise, v is a core virtual router. Furthermore, each virtual access router v is characterized by a geographical zone denoted by $\mathcal{Z}_v$. In fact, $\mathcal{Z}_v$ defines the geographic area where v should be embedded. Moreover, each virtual link $d \in E(\mathcal{D})$ is typified by its required bandwidth $C(d)$.

5.1.2 Reliable batch-embedding $\mathcal{VN}$ problem

We model the arrival rate of $\mathcal{VN}$ requests as a Poisson process with density λ_A. All the incoming $\mathcal{VN}$s during the predefined time window, denoted by Δ_T, are queued and their mapping is deferred to the next processing time slot. At the end of Δ_T, the $\mathcal{VN}$ batch-embedding strategy is launched. Figure 5.1 illustrates the above model. Note that each $\mathcal{VN}$ has a finite lifetime and leaves the $\mathcal{SN}$

by unfreezing the allocated physical resources. Let's denote the set of received $\mathcal{VN}$ requests during Δ_T by $\Omega = \{\mathcal{VN}_1, \cdots \mathcal{VN}_m\}$.

It is worth noting that the order of embedding $\mathcal{VN}$s directly impacts the $\mathcal{SN}$ configuration (i.e., physical residual resources). For instance, if we assume that 4 requests arrived during Δ_T, hence $\Omega = \{\mathcal{VN}_1, \mathcal{VN}_2, \mathcal{VN}_3, \mathcal{VN}_4\}$. In one hand, the mapping of $\mathcal{VN}_1$ may consume huge $\mathcal{SN}$ resources so this would prohibit the embedding of the remaining requests. In the other hand, embedding first $\mathcal{VN}_3$ would allow the mapping of $\mathcal{VN}_4$ and $\mathcal{VN}_2$. However, starting by mapping $\mathcal{VN}_4$ cannot allow the embedding of $\mathcal{VN}_2$ and $\mathcal{VN}_3$. In summary, the embedding order strongly impacts the acceptance rate of $\mathcal{VN}$s and hence the $\mathcal{CP}$'s turnover.

For each Δ_T, the $\mathcal{VN}$ batch-embedding strategy selects the **best sequence** (i.e., order is considered) of $\mathcal{VN}$s providing the maximum of revenue. Let's denote this sequence by $\Theta_b \subseteq \Omega$. Formally, our objective is to generate Θ_b while:

$$\forall \Theta_i \subseteq \Omega, \mathcal{R}(\Theta_b) \geq \mathcal{R}(\Theta_i) \tag{5.1.2}$$

where $\mathcal{R}(\Theta_i)$ is the sum of revenues earned by embedding the sequence of $\mathcal{VN}$s included in Θ_i. Formally,

$$\mathcal{R}(\Theta_i) = \sum_{\mathcal{VN}_i \in \Theta_i} U(\mathcal{VN}_i) \tag{5.1.3}$$

Note that $U(\mathcal{VN}_i)$ is the utility function 5.1.4 defined in the previous chapter and depends on the volume of resources required in terms of bandwidth, processing power, memory and lifetime.

$$U(\mathcal{VN}_i) = \textstyle\sum_{\mathcal{N}_j \in \mathcal{N}} \mathcal{U}_{\mathcal{N}_j} \tag{5.1.4}$$

$$\mathcal{U}_{\mathcal{N}_j} = \textstyle\sum_{d \in \xi_{\mathcal{N}_j}} (\mathcal{L}_d) + \mid \exp\left[R^n(w_{\mathcal{N}_j}, T_{\mathcal{N}_j}) \cdot \nabla^w_{\mathcal{N}_i}\right] - 1 \mid \tag{5.1.5}$$

Note that the embedding of sequence Θ_i is binding by the residual hardware resources in the $\mathcal{SN}$. Thus, Θ_i includes only $\mathcal{VN}$s that are successfully mapped in the same order defined by this sequence (i.e., Θ_i). The mapping of each $\mathcal{VN}$ is ensured by a predefined virtual network embedding strategy. The survivable embedding problem of **one** $\mathcal{VN}$, in each node in $\mathcal{T}$, is modelized in subsection 5.1.3.

To compute the best sequence Θ_b maximizing the $\mathcal{CP}$'s revenue, the ideal and simplistic way is to check by simulation the embedding of all combinations (i.e., all possible orders) with regard to the $\mathcal{SN}$ resources constraints. Unfortunately, the above approach is computationally hard because of its exponential time complexity.

In order to skirt this complexity, we model the sequences of $\mathcal{VN}$s as a decision tree denoted by $\mathcal{T}$ and defined as follows. Each node in the tree represents the embedding simulation of one $\mathcal{VN}$ request. The link between two nodes in the tree exits if their associated $\mathcal{VN}$ requests are

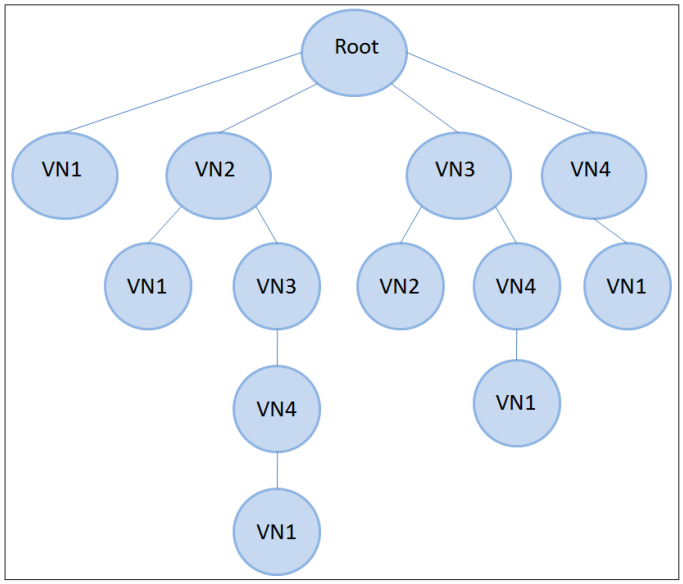


Figure 5.2: Batch $\mathcal{VN}$s' embedding basing on decision tree model

neighbors in a sequence Θ_i. The root node is abstract and represents the non-mapping of any $\mathcal{VN}$ in the Cloud's backbone. In other words, each branch in $\mathcal{T}$ corresponds to one and unique sequence Θ_i. Besides, each node in $\mathcal{T}$ located at depth d means that the mapping of $d-1$ $\mathcal{VN}$ requests is simulated since the level 1 represents the root node. Figure 5.2 illustrates the decision tree $\mathcal{T}$ related to the mapping of the $\Omega = \{\mathcal{VN}_1, \mathcal{VN}_2, \mathcal{VN}_3, \mathcal{VN}_4\}$. As we can see in this figure, from the root node the mapping of each request is possible independently (i.e., the $\mathcal{VN}_i$ request embedding is made basing on the first $\mathcal{SN}$ state and not considering other requests). Then the mapping process is stopped after the $\mathcal{VN}_1$ embedding because of substrate resource lack. On the other hand, after mapping $\mathcal{VN}_2$ the embedding of $\mathcal{VN}_1$ and $\mathcal{VN}_3$ remains possible. However, it is impossible to map $\mathcal{VN}_4$ after $\mathcal{VN}_2$ embedding. That is why a child containing $\mathcal{VN}_4$ is not generated. In this way the generation of new child(ren) from each node is processed. It is clear that having new child is related to embedding new virtual request. It is straightforward to see that the best sequence $\Theta_b = \{\mathcal{VN}_2, \mathcal{VN}_3, \mathcal{VN}_4, \mathcal{VN}_1\}$ (i.e., order is respected) having to the maximum revenue is the one leading to the mapping of the whole requests set Ω.

It is worth pointing out that reliable $\mathcal{VN}$ embedding problem is a non-linear and multi-objective combinatorial optimization (see sub-section 5.1.3). It has been proved in [72] [73] that it is NP-hard problem. Accordingly, handling the mapping of $\mathcal{VN}$ requests set in a batch mode is computationally intractable. Consequently, a smart and judicious batch-embedding $\mathcal{VN}$ strategy is mandatory to build incrementally $\mathcal{T}$ and to converge to the best branch (i.e., sequence Θ_b). In Section 5.2, we will detail our proposal based on the Monte-Carlo Tree Search algorithm.

5.1.3 Reliable embedding of one virtual network problem

The maximization of the acceptance rate of $\mathcal{VN}$ requests can be achieved by i) maximizing the load balancing of the $\mathcal{SN}$ and ii) minimizing the amount of physical resources allocated for each $\mathcal{VN}$ request. To reach these aims, the volume and the standard deviation of residual resources should be respectively maximized and minimized. Formally,

$$\begin{array}{c} \textbf{maximise}[\min\{C(e)\}, \min\{B(w)\}, \min\{M(w)\}] \\ \textbf{minimise}[std\{C(e)\}, std\{B(w)\}, std\{M(w)\}] \\ e \in E(\mathcal{G}), \quad w \in V(\mathcal{G}) \end{array} \tag{5.1.6}$$

Moreover, in order to maximize the reliability, our objective consists in mapping the virtual **routers** and **links** within the substrate resources offering the highest reliability during the lifetime of each $\mathcal{VN}$. For each virtual router $v \in V(\mathcal{D})$ in a $\mathcal{VN}$, the reliability objective is expressed as following:

$$\textbf{maximise}\{\mathcal{R}^n(w^v, T_{w^v})\}, w^v \in V(\mathcal{G}) \tag{5.1.7}$$

where w^v denotes the substrate router hosting v and T_{w^v} is the age of w^v when $\mathcal{D}$ leaves the physical backbone network $\mathcal{G}$.

Besides, the objective of virtual link embedding in unsplittable manner is formally described as follow:

$$\begin{array}{c} \textbf{minimize} \quad \Big\{ \prod_{e^d \in \mathcal{P}} \left(1 - \mathcal{R}^l\left(e^d, T_{e^d}\right)\right) \cdot \prod_{e^d \in \mathcal{P}} \left(1 - \frac{C(e^d)}{\hat{C}(e^d)}\right) \cdot \\ \prod_{w^d \in \mathcal{P}} \left(1 - \mathcal{R}^n\left(w^d, T_{w^d}\right)\right) \Big\}, \mathcal{P} \in \textit{Paths}(d) \end{array} \tag{5.1.8}$$

where $\textit{Paths}(d)$ denotes the set of substrate paths which can host the virtual link d, $\{e^d\}$ and $\{w^d\}$ denote the set of substrate links and routers forming the substrate path $\mathcal{P} \in \textit{Paths}(d)$. As defined above, T_{w^d} and T_{e^d} are respectively the ages of the physical router w^d and link e^d when the virtual network $\mathcal{D}$ leaves the backbone network $\mathcal{G}$. Note that the above cost function quantifies the reliability provided by substrate path's equipments (i.e., routers and links) and residual bandwidth. It is straightforward to see that reliable $\mathcal{VN}$ embedding problem is a non-linear and multi-objective combinatorial optimization.

5.2 Proposal: `BR-VNE` strategy

In this section we detail our proposal named **Batch Reliable Virtual Network Embedding** (`BR-VNE`). As described above, the batch reliable mapping problem of $\mathcal{VN}$s is transformed as a tree search problem. So first of all, we will introduce the main optimization strategies found in existing literature to resolve this kind of problem. Then, we will present the main stages of our proposal `BR-VNE`.

Algorithm 8: BR−VNE psuedo-code

```
Inputs: SN, Ω
Output: Θ_b
Initialization (T)
while Computational Budget do
    n ← Selection-Node-Explore(T)
    B ← Generation-Sub-Branch(T, n)
    T ← T + B
    Update-Relevance(T)
Θ_b ← Selection-Best-Solution(T)
```

5.2.1 Tree search optimization algorithms

The main problem of tree search problem is the scalability. In fact, the size of solution space is finite but increases exponentially with the number of nodes in the tree. Consequently, the problem becomes computationally intractable in large-scale trees. Many research strategies were investigated to tackle its complexity such as: A⋆ [90], Best−First−Search [91], Branch&Bound [92], etc. Unfortunately, computing time and the relevance of the solution are the main weaknesses of the aforementioned algorithms. In fact, most of them are using evaluation functions to guide the navigation process in the tree. Whereas, even if the chosen evaluation function is an admissible under-estimator one, the quality of the generated solution is not appreciated [93]. Moreover, the above methods assume that the tree search is built and the problem consists only in finding the best solution. Unfortunately, it is not easy to generate the global solution tree at the beginning due the exponential size of the solution space.

Our proposal BR−VNE is based on Monte-Carlo Tree Search [43] (MCTS) optimization algorithm. It couples the i) tree building and ii) optimum searching process. This fits to our batch-embedding problem since the decision tree is not fully built. Indeed, MCTS combines the precision of tree search with the generality of random sampling [43]. MCTS has been applied with spectacular success in computer Games such as computer Go [94] as well as some combinatorial optimization problems [43]. Accordingly, MCTS is an efficient tool that guides a decision maker in different kind of problems.

The determination of the best sequence Θ_b of $\mathcal{VN}$s in the set Ω of incoming $\mathcal{VN}$s during the time slot Δ_T is similar to some computer games' family. Fortunately, the latter kind of problem is solved efficiently with the MCTS strategy. The suitable game category that best fits our problem description is the **Single Player** one [93]. Its objective is to maximize the player's payoff regardless to any other opponent. Consequently, the $\mathcal{CP}$ can be considered as the player who is looking to maximize the global benefit. For this reason, our proposal BR−VNE is based on MCTS. Hereafter, we will describe the main BR−VNE's stages.

5.2.2 BR−VNE description

BR−VNE incrementally builds the decision tree $\mathcal{T}$ and searches for the optimized mapping sequence Θ_b matching with the best branch in $\mathcal{T}$. The main stages of BR−VNE are: i) *Tree initialization*, ii) *Selection of node to explore*, iii) *Generation of sub-branch*, iv) *Update of nodes' relevance* and finally v) *Selection of best solution*. Algorithm 8 summarizes the pseudo-code of BR−VNE.

5.2.2.1 Tree initialization

In this stage, the root of decision tree $\mathcal{T}$ is created. This node does not contain any $\mathcal{VN}$ requests. The root node means that no $\mathcal{VN}$ request within Ω is mapped in $\mathcal{SN}$. The root is an abstract node.

5.2.2.2 Selection of node to explore

The main objective of this stage is to select one node within decision tree to expand its branch in the next stage. In fact, the main idea is to make a balance between i) exploitation of promoting branches and ii) exploration dealing with less promising branches. In the other hand, BR−VNE should also check non promising nodes in order to avoid local optima. To do so, we associate to each node $n \in \mathcal{T}$ a relevance function denoted by ψ_n. It quantifies the attractiveness of n to be elected in Θ_b. Formally,

$$\psi_n = \bar{\mathcal{R}}_n + \alpha \cdot \sqrt{\frac{\ln(\hat{\mathcal{V}}_n)}{\mathcal{V}_n}} + \sqrt{\frac{\sum_l {\mathcal{R}_n^l}^2 - \hat{\mathcal{V}}_n \cdot \bar{\mathcal{R}}_n^2 + \beta}{\mathcal{V}_n}} \tag{5.2.9}$$

where i) $\bar{\mathcal{R}}_n$ is the average revenue generated by all the sequences Θ_i transiting over node n, ii) $\mathcal{V}_n$ is the number of sequences passing through n, in other words it is the number of visits, iii) $\hat{\mathcal{V}}_n$ is the number of visits of n's parent (i.e., predecessor), iv) $\mathcal{R}_n^l$ is the revenue generated by the sequence Θ_l transiting over n. Note that $1 \leq l \leq \mathcal{V}_n$. Finally, α and β are constants.

It is worth noting that ψ_n is inspired from [95] which defines the **upper confidence bounds** applied to trees. Thanks to Monte Carlo Tree Search approach, the upper bound can be reached in polynomial time [95]. It is clear that the third term in the above metric quantifies a possible search deviation of node n [93]. It includes the sum of the squared revenues $\mathcal{R}_n^l$ corrected by the expected revenues $\hat{\mathcal{V}}_n \cdot \bar{\mathcal{R}}_n^2$. The big constant β is useful to promote rarely explored nodes. The defined metric 5.2.9 allows BR−VNE to balance the control between exploitation of frequently visited nodes and exploration of rarely visited nodes. Accordingly, local optimums are avoided.

BR−VNE selects one node n_s in $\mathcal{T}$ which satisfies the following constraints:

- n_s can generate new children. That means, there exists at least one $\mathcal{VN}$ request in Ω which is not mapped by all the ancestors of n_s. Formally, assuming that $n_s \in \Theta_i$, where Θ_i defines the sequence from the root till n_s, then:

$$\Omega - \Theta_i \neq \emptyset \tag{5.2.10}$$

- The father of n_s cannot generate other children. That means, the upstream levels are fully explored and cannot add any new sub-branch.

- The relevance metric ψ_{n_s} of n_s is the maximum among its brothers. Formally,

$$\psi_{n_s} = \max_{n \in Brother(n_s)} (\psi_n) \tag{5.2.11}$$

- Recursively the latter rule must be satisfied for each ancestor until reaching the root. In other words, all the nodes in sequences are the best among their small family (i.e., only brothers).

5.2.2.3 Generation of sub-branch

Once n_s is located, a sub-branch denoted by $\mathcal{B}$ is randomly and iteratively generated. Note that randomness guarantees the convergence to the upper bound in polynomial time [95]. We assume that $n_s \in \Theta_i$ and $\mathcal{B} = \emptyset$. Then at each iteration one virtual network request $\mathcal{VN}_r$ is randomly elected in $\Omega - (\Theta_i + \mathcal{B})$. Afterwards, the mapping simulation of $\mathcal{VN}_r$ is launched. If the embedding process is successful, then $\mathcal{VN}_r$ is added to $\mathcal{B}$. The same process is repeated until i) all virtual network requests are mapped (i.e., $\Omega - (\Theta_i + \mathcal{B}) = \emptyset$) or ii) the embedding process fails to map the randomly elected virtual network request due to physical resources shortage. Finally, the new generated sub-branch $\mathcal{B}$ is pasted to the decision tree $\mathcal{T}$ at node n_s. Formally, $\mathcal{T} \leftarrow \mathcal{T} + \mathcal{B}$.

It is worth noting that in this work, we make use of one of our previous online virtual network embedding strategies: `CG-VNE` [40] and `PR-VNE` [39]. In fact, we define four variants of `BR-VNE`: i) `B-CGVNE`[1], ii) `B-PRVNE`, iii) `B-CG-VNE-splittable` and iv) `B-Hybrid` basing on the used online approach. `B-CGVNE` and `B-PRVNE` make use of `CG-VNE`, `PR-VNE` and `Advanced-CG-VNE-splittable`, respectively. On the other hand, `B-Hybrid` alternates the use of `CG-VNE` and `PR-VNE` to embed one $\mathcal{VN}$.

5.2.2.4 Update of nodes' relevance

Once the sub-branch $\mathcal{B}$ is added to the decision tree $\mathcal{T}$, `BR-VNE` updates the relevance function defined in equation 5.2.9 for all nodes in the sequence Θ_i containing $\mathcal{B}$. More precisely for all nodes $n \in \Theta_i$, i) $\bar{\mathcal{R}}_n$ and $\mathcal{R}_n^l$ are updated with respect to the cumulative revenue induced by Θ_i (i.e., the sum of virtual network revenue associated to each node in Θ_i) and ii) the visit frequency register $\mathcal{V}_n$ is incremented. In other words, `BR-VNE` propagates the revenue to all ancestors nodes till the root node in the new created sub-branch $\mathcal{B}$. Consequently, in doing so, the relevance function value of each node ψ_n is updated. Note that ψ_n decreases with respect to the frequency of visits and increases with the benefit induced by node n.

[1] In this section, `BR-VNE` is used to refer `B-CGVNE` variant.

5.2.2.5 Selection of final solution

The above stages 2, 3 and 4 are repeated during a predefined period named computational budget and denoted by δ_C. Note that δ_C can be expressed by i) time period or ii) maximum number of loops. δ_C is calibrated with respect to the i) the arrival rate λ_A density of $\mathcal{VN}$ requests, ii) the duration of batch period Δ_T, and iii) the hardware performance of Cloud provider controller in which the embedding strategy is executed.

Once δ_C period is over, `BR-VNE` looks for the best branch Θ_b in $\mathcal{T}$. In order to minimize the research convergence time to this best sequence, the idea is to check only the leaf nodes in $\mathcal{T}$. In fact, the relevance function ψ of any leaf node interprets more than the induced revenue of its sequence (i.e., $\bar{\mathcal{R}}_n$) since it is visited only once (i.e., $\mathcal{V}_n = 1$). We consequently reduce the convergence time to select the best sequence. Formally, the best leaf node denoted by n_b must satisfy:

$$\psi_{n_b} = \max_{n \in LeafNodes(\mathcal{T})} (\bar{\mathcal{R}}_n) \tag{5.2.12}$$

Then, Θ_b is the best sequence in $\mathcal{T}$ containing n_b. It is worth noting that Θ_b is unique since each node in the tree is attached to one and only one parent.

5.3 Performance evaluation

In this section, we will gauge the performance of our proposed algorithm `BR-VNE` and its variants (i.e., `B-CGVNE`, `B-PRVNE`, `B-Hybrid` and `B-CG-VNE-splittable`) based on extensive simulations. First of all, we describe our discrete event simulator that takes into consideration different level of reliability by distinguishing different classes of substrate equipments' ages. Then, we define the performance metrics to assess our proposal and the related strategies. Afterwards, we calibrate the parameters of `BR-VNE`. Finally, we discuss the effectiveness of our proposal by comparing it with batch and online embedding virtual network strategies.

5.3.1 Simulation environment

We implemented a discrete event reliable $\mathcal{VN}$ mapping simulator. To this aim, we make use of `GT-ITM` [80] tool to generate random $\mathcal{SN}$ and $\mathcal{VN}$ topologies. We model $\mathcal{VN}$ lifetime by exponential distribution with mean μ_L. As in [52], we set the $\mathcal{SN}$ size to 100. In this case, the ratio of access and of core routers are respectively fixed to 20% and 80%. Moreover, we set $\mathcal{VN}$ size based on discrete uniform distribution taking values between $[3, 10]$. We keep the same proportion of access virtual nodes like $\mathcal{SN}$ 20% for each $\mathcal{VN}$. It is worth mentioning that in both cases ($\mathcal{VN}$ and $\mathcal{SN}$), each pair of routers is randomly connected with a probability of 0.5.

We set the number of $\mathcal{VN}$ requests to 2000. The average lifetime μ_L of $\mathcal{VN}$s is set to 1000

seconds. Regarding the arrival rate λ_A of virtual network requests[2], it is taking the following values: 4, 8, 16 and 32 requests per 100 seconds. We calibrate the capacity of substrate nodes and links (i.e., $B(w)$, $M(w)$ and $\hat{C}(e)$) according to a continuous uniform distribution taking values in $[50, 100]$. Furthermore, we set the requested virtual resources (i.e., $B(v)$, $M(v)$ and $\hat{C}(d)$) based on a continuous uniform distribution taking values between $[10, 20]$.

Regarding $\mathcal{SN}$ reliability, the parameters a and b of the Weibull distribution modeling the MTBF are respectively set to 25×10^4 and 1.5. It is noteworthy that a and b are calibrated in order to obtain a lifetime for each physical equipment ($\approx 6 \times 10^5$) roughly equal to 10 times the simulation duration ($\approx 6 \times 10^4$ seconds). Moreover, the proportional of young equipments in $\mathcal{SN}$ is set to 30%. We study the performance with respect to the adult proportion (hence old proportion) in the $\mathcal{SN}$. Besides, physical equipment's initial age follows a uniform distribution within the bounds of its group (i.e., young, adult and old).

It is worth pointing out that each performance value of the implemented strategies is equal to the average of 30 simulations. Furthermore, simulation results are always presented with confidence intervals corresponding to a confidence level of 95%. Note that tiny confidence intervals are not shown in the following figures.

5.3.2 Performance metrics

Hereafter, we define performance metrics used to evaluate `BR-VNE`.

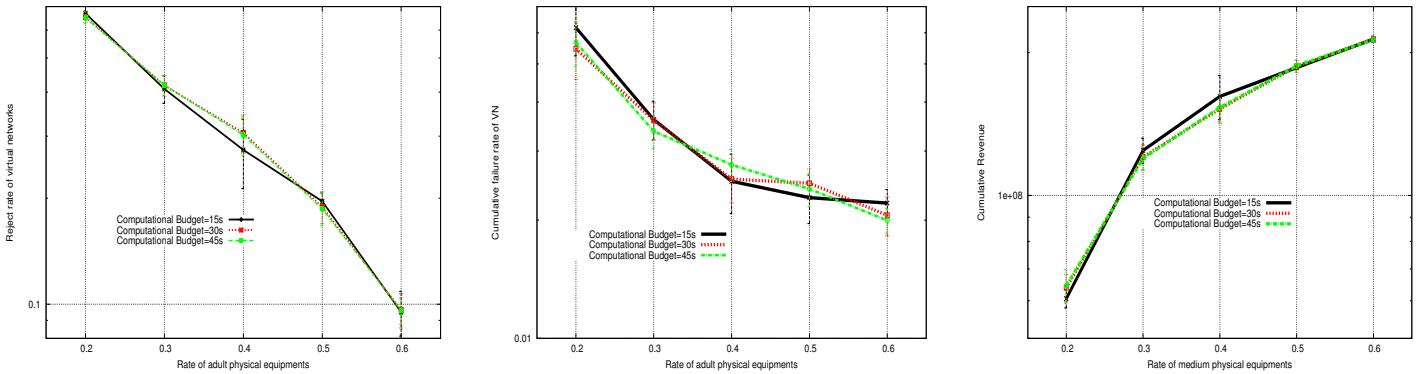
- Q is the reject rate of $\mathcal{VN}$ requests during the simulation. In other terms, the rate of $\mathcal{VN}$s that have not been mapped in the $\mathcal{SN}$.

- F is the rate of deployed $\mathcal{VN}$s that has been impacted by physical failures during the simulation. In fact, a $\mathcal{VN}$ is impacted if at least one of its components (i.e., virtual router and/or link) is impacted by the physical failure.

- G evaluates the $\mathcal{SN}$ provider's profitability by calculating the total benefit G_1 of all accepted $\mathcal{VN}$ requests minus the paid penalties G_2 to the clients induced from the physical failures during all the simulation lifetime ($\mathrm{G} = \mathrm{G}_1 - \mathrm{G}_2$). G_1 is defined as in [55]. It depends on the amount of requested resources and the lifetime of the virtual network in the backbone network $\mathcal{SN}$. We define the penalty G_2 as the reimbursement to the clients. The cost is proportional to the residual time of a $\mathcal{VN}$ heightened by a penalty ρ. Formally, for each $\mathcal{VN}$ denoted by $\mathcal{D}$:

$$\mathrm{G}_2(\mathcal{D}) = \frac{(T - \Upsilon_T)}{T} \times \mathrm{G}_1(\mathcal{D}) \times \rho \tag{5.3.13}$$

[2]In the remaining, claiming, for instance, that $\lambda_A = x$ is equivalent to $\lambda_A = x$ per 100 seconds.



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.3: δ_C - $\lambda_A = 4$

where T is the requested network virtual lifetime, Υ_T is the hosting duration in the $\mathcal{SN}$ before the physical failure, and ρ is the penalty rate. In our simulations, we set the penalty to 50%.

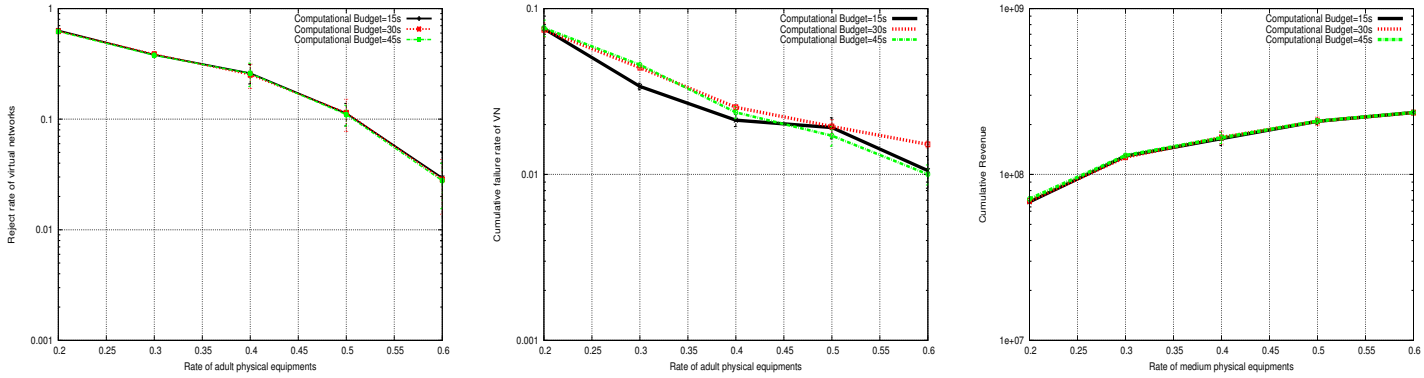
5.3.3 Simulation results

The current subsection is organized as follow. First, we focus on setting the suitable **Computational Budget** parameter δ_C of `BR-VNE`. In fact, it is very important to fix the fastness level of the algorithm while the quality of final solution is not deteriorated. Next, `BR-VNE` will be compared with the virtual network batch-embedding `B-Base` [52] strategy. Afterwards, we compare our proposal `BR-VNE` to the most prominent online survivable $\mathcal{VN}$ algorithm `CG-VNE` [40]. Finally, we compare the splittable variant of our proposal denoted `B-CG-VNE-splittable` to the splittable batch-based approach `WiNE` [55].

5.3.3.1 `BR-VNE` setting parameters

Computational Budget δ_C is a decisive parameter since it impacts directly the `BR-VNE` performance. In one hand, in order to get solution with better quality, δ_C should be fixed to high value. In other words, `BR-VNE` would have enough time to search the best solution by discovering new branches in the decision tree. However, dedicating a long period to computing process may delay the client service as well as consume further $\mathcal{CP}$ resources. On the other hand, setting δ_C to a short period for the mapping process may converge to a local optimum. Thus, $\mathcal{CP}$'s revenue would not be maximized. In the following, we will evaluate the impact of computational budget δ_C on the solution quality. δ_C is fixed respectively to: 15, 30 and 45 seconds.

In Figure 5.3, we set the arrival rate λ_A to 4. Figure 5.3(a) illustrates the reject rate Q of virtual



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.4: δ_C for $\lambda_A = 8$

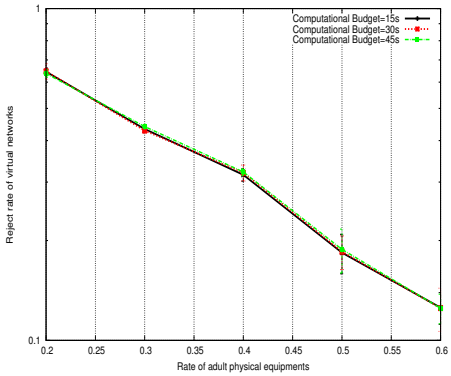
networks. We remark that approximately the same performances are obtained even if δ_C is equal to 45 seconds. We can conclude that BR−VNE quickly converges to the best solution. Figure 5.3(b) illustrates the rate F of virtual networks impacted by physical failures. It is straightforward to see that BR−VNE approximately impacts the same proportions of physical equipments even if the computation budget is increased. Note that the rate of adult equipment varies between 20% and 60% (i.e., rate of old equipment varies between 50% and 10%). We recall the proportion of young equipment is fixed to 30%. Consequently, in Figure 5.3(c), the cumulative revenue of Cloud provider is roughly the same for all values of δ_C. In fact, the figure clearly shows that the three curves are overlapping. So we can conclude that the three values of BR−VNE approximately guarantee the same revenue to the Cloud provider.

Hereby, we describe the performance of BR−VNE for $\lambda_A = 8$. The Figure 5.4 confirms the obtained results when $\lambda_A = 4$. Indeed, curves in the aforementioned figures are overlapping which refelect the fact that results are roughly same for the different values of δ_C. Accordingly, BR−VNE converges to an optimized solution in reasonable time (i.e., 15 seconds).

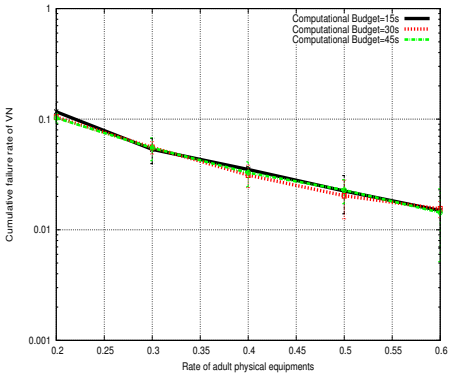
In the following scenario we set the arrival rate λ_A to 16 in order to stress more the performance of our proposal and to check the suitable calibration. Hereafter, we gauge the performance of BR−VNE basing on the predefined metrics (i.e., Q, F and G). It is clear that the overlapped curves in Figure 5.5 show that by setting the calibration parameter δ_C to 15 seconds BR−VNE reaches the same performances as setting δ_C to 45 seconds.

Hereinafter, we set the arrival rate parameter λ_A to 32. It is straightforward to see that results, showed in Figure 5.6 are in concordance with those related to $\lambda_A = 16$, $\lambda_A = 8$ and $\lambda_A = 4$.

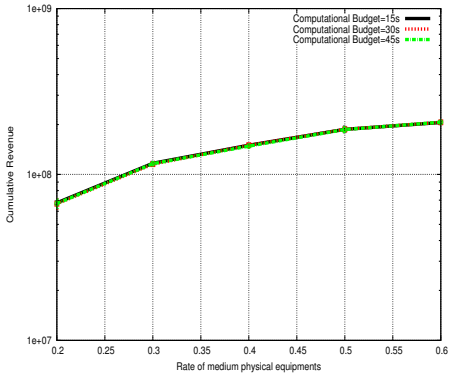
Consequently, our proposal BR−VNE is transparent regarding the arrival rate of λ_A. Thus, BR−VNE develops promizing branches and converges to an optimized solution quickly. Accordingly, in the remainder simulations, we make use of the fastest value (i.e., $\delta = 15$ seconds) to



(a) Reject rate: Q

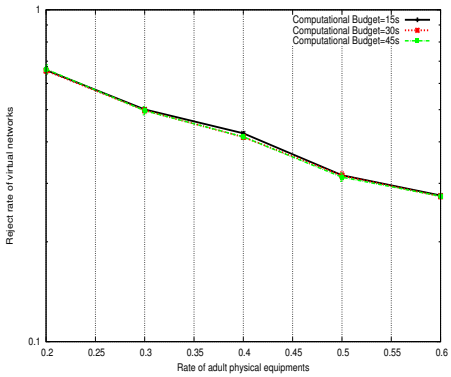


(b) Blackout rate: F

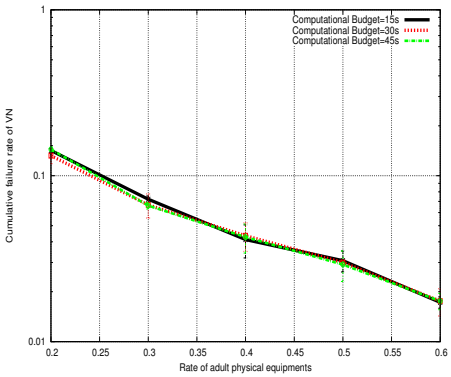


(c) Revenue: G

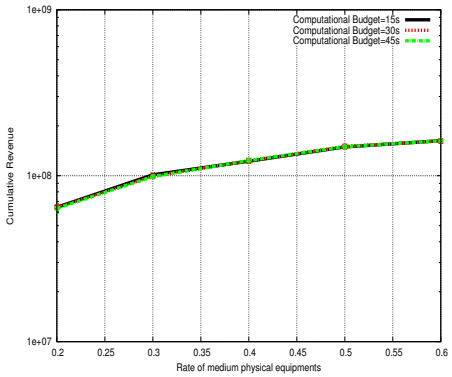
Figure 5.5: δ_C for $\lambda_A = 16$



(a) Reject rate: Q



(b) Blackout rate: F



(c) Revenue: G

Figure 5.6: δ_C for $\lambda_A = 32$

compare our proposal to most prominent related strategies.

5.3.3.2 Comparison with batch-embedding approach

We compare our proposal `BR-VNE` with its variants to the best related work strategy dealing with batch-embedding $\mathcal{VN}$ `B-Base` [52]. To do so, we vary the arrival rate λ_A.

Figure 5.7(a) shows that `BR-VNE` variants outperforms `B-Base` in term of reject rate Q of $\mathcal{VN}$s. In fact, `B-Base` is mapping the set of incoming $\mathcal{VN}$ requests during Δ_T following the order based on the revenue gained from each $\mathcal{VN}$ request. Unfortunately, this greedy approach cripples the acceptance process of $\mathcal{VN}$ compared to our proposal. Indeed, we remark that in Figure 5.7(a), `BR-VNE` minimizes Q with at least 18.72% compared with `B-Base`. We can conclude that our proposal is a successful strategy which fairly manages the $\mathcal{CP}$'s backbone resources. On the other

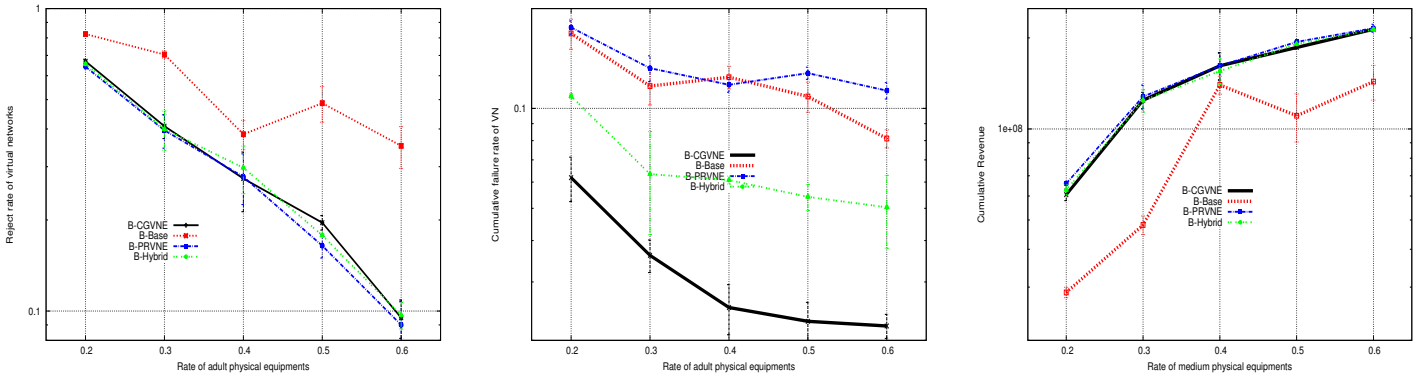
hand, we note that B−PRVNE, B−CGVNE and B−Hybrid have the same reject rate. Whereas, B−PRVNE outperforms the remaining variants when adult rate is 50% and 60%. This result can be explained by the fact that B−CGVNE and B−Hybrid remap impacted virtual resources by physical crashes. So, B−CGVNE and B−Hybrid consume more substrate resources. Thus B−PRVNE has more free resources to handle new requests.

Figure 5.7(b) illustrates the comparison of BR−VNE variants and B−Base in term of blackout rate F. We recall that B−Base does not consider physical failures. It is worth pointing out that B−Base and B−PRVNE have same performances basing on blackout rate metric. Besides, B−Base outperforms B−PRVNE when the adult proportion is 50% and 60%. In fact, both of them do not implement a recovery mechanism to deal with physical failures. Moreover, B−PRVNE clearly outperforms B−Base basing on Q metric. Consequently, when a physical failure occurs it impacts more clients for B−PRVNE strategy comparing to B−Base algorithm. On the other hand, B−CGVNE and B−Hybrid outperform B−Base basing on the blackout rate F. In fact, thanks to the reactive mechanism adopted by B−CGVNE and B−Hybrid, it re-embeds impacted virtual resources by physical outages as soon as a failure occurs in the $\mathcal{VN}$. Besides, all impacted virtual resources will be released for B−Base algorithm. It is worth noting that even if B−Base releases more $\mathcal{VN}$ from the $\mathcal{SN}$ which means the latter is less over-loaded, but the reject rate is more important than our algorithms (i.e., B−CGVNE and B−Hybrid) as illustrated in Figure 5.7(a). This explains the best performances of B−CGVNE and B−Hybrid compared to B−Base.

Obviously, the revenue earned by BR−VNE variants is better than B−Base. Figure 5.7(c) illustrates the cumulative gain G. In fact, it is a direct consequence since B−CGVNE and B−Hybrid outperform B−Base in term of i) reject rate Q and ii) blackout rate F metrics. Note that if the rate of adult equipment is equal to 20% and hence old equipment is 50%, BR−VNE improves the revenue of B−Base by 110.35%. Furthermore, B−PRVNE leads to better turnover comparing to B−Base thanks to accepting more clients.

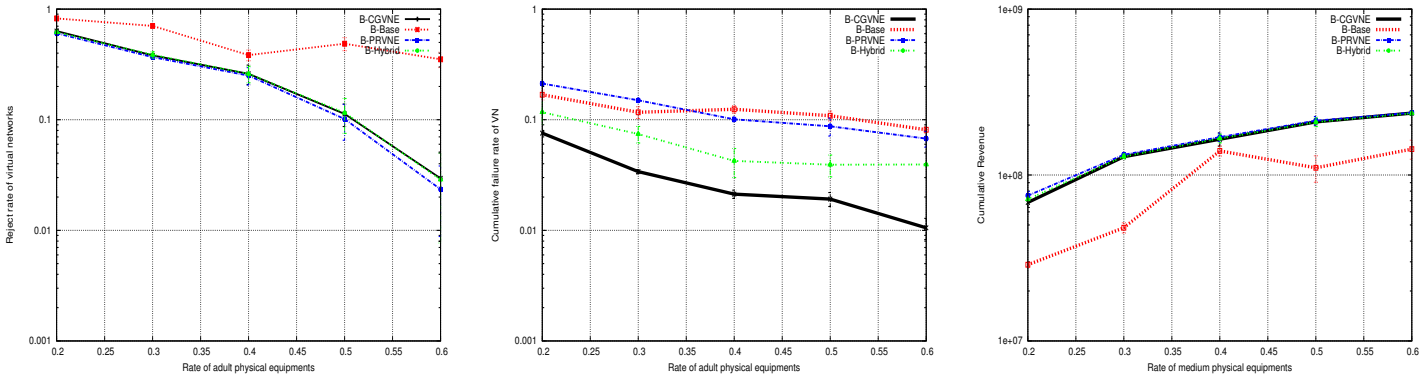
Hereby, we set the arrival rate λ_A to 8. We notice that BR−VNE variants have roughly same performance regarding reject rate (see Figure 5.8(a)). However, B−PRVNE lightly outperforms B−CGVNE and B−Hybrid when adult equipment proportion is 50% and 60%. This is justified by the reactive behaviour of B−CGVNE and B−Hybrid. In Figure 5.8(b), B−PRVNE has the worst blackout rate comparing to the remaining variants of BR−VNE (i.e., B−CGVNE and B−Hybrid). This result explains why B−PRVNE has better results basing on Q metric. In fact, B−CGVNE and B−Hybrid consumes more physical resources comparing to B−PRVNE by reactively remapping the impacted virtual resources Consequently, B−PRVNE, B−CGVNE and B−Hybrid lead to the same turnover. This can be easily seen in Figure 5.8(c).

Hereafter, we stress more the simulation scenario by setting the arrival rate λ_A to 16. It is straightforward to see B−CGVNE outperforms all remaining strategies basing on reject rate metric. (i.e., Q). In fact, Figure 5.9(a) clearly highlights it. This positive result shows that using Game



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.7: BR−VNE versus B−Base - $\lambda_A = 4$

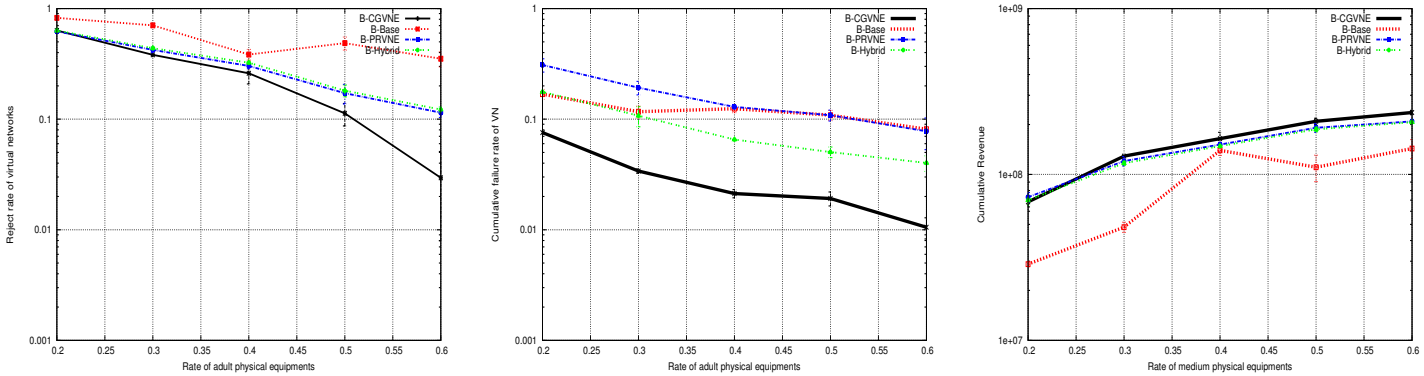


(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.8: BR−VNE versus B−Base - $\lambda_A = 8$

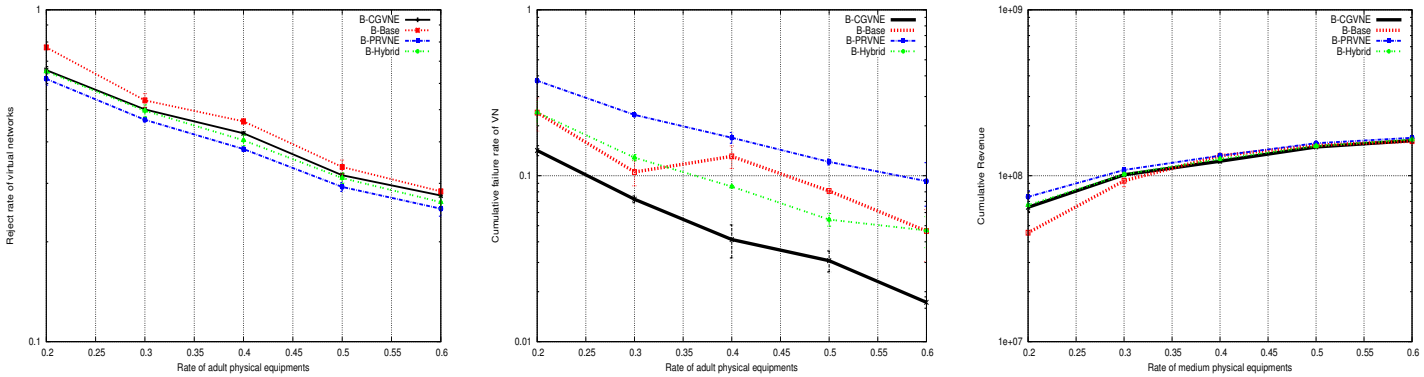
Theory tool to devise a batch algorithm leads to the best acceptance rate. Despite of having the best acceptance rate, B−CGVNE has the minimum blackout rate (see Figure 5.9(b)). Consequently, coupling Game Theory and reactive mechanism induce the best results in term of reject rate and blackout rate. Accordingly, as depicted by Figure 5.9(c) B−CGVNE offers the best turnover for the Cloud Provider.

Hereafter, we fix the arrival rate parameter λ_A to 32. We notice in Figure 5.10(a) that B−PRVNE has the best acceptance rate. This is thanks to the preventive behaviour of the latter variant. Besides, B−PRVNE does not implement any recovery mechanism, so it does not allocate extra physical resources. However, B−PRVNE has the worst blackout rate comparing to B−CGVNE and B−Hybrid (see Figure 5.10(b)). This can be justified by the reactive mechanism adopted by the two latter variants. Accordingly, BR−VNE variants ensure the best revenue basing on Figure 5.10(c). It is clear



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.9: BR−VNE versus B−Base - $\lambda_A = 16$



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

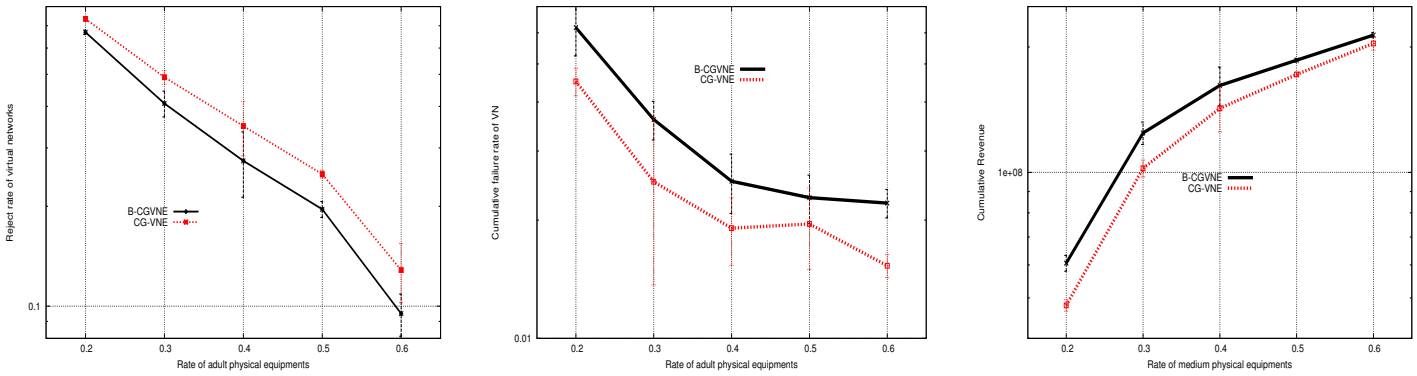
Figure 5.10: BR−VNE versus B−Base - $\lambda_A = 32$

that B−PRVNE lightly outperforms B−CGVNE and B−Hybrid. But, having more blackout may impact the reputation of Cloud Provider. Thus, B−CGVNE and B−Hybrid realize the best trade-off between i) accepting more clients and ii) impacting less clients.

5.3.3.3 Comparison with online-reliable unsplittable strategy

Hereafter, we compare our proposal B−CGVNE variant to the most prominent online-reliable embedding virtual network CG−VNE. In fact, we have shown in [40] that CG−VNE outperforms the related online-embedding $\mathcal{VN}$ algorithms addressing survivability.

Figure 5.11(a) shows that B−CGVNE outperforms CG−VNE. In fact, basing on the macroscopic view of incoming $\mathcal{VN}$s requests during the time window Δ_T, B−CGVNE optimizes the choice of the accepted $\mathcal{VN}$s basing on Monte Carlo Tree search optimization algorithm. This global vision



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

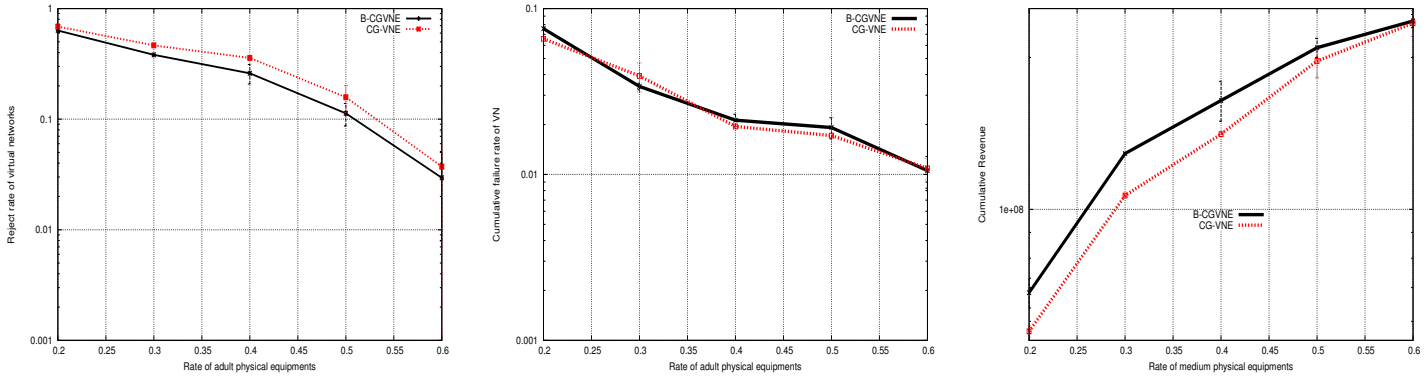
Figure 5.11: B−CGVNE versus CG−VNE - $\lambda_A = 4$

of incoming requests is missing for CG−VNE which is an online strategy. Hence, it may accept some requests crippling the embedding in the $\mathcal{SN}$. Consequently, it is straightforward to see that B−CGVNE outperforms CG−VNE in term of reject rate Q. At least, the improvement is equal to 8.87%. Figure 5.11(b) depicts the comparison of blackout rate metric F between B−CGVNE and CG−VNE. The latter figure shows that CG−VNE outperforms our proposal. This can be explained by two reasons. First, B−CGVNE accepts more $\mathcal{VN}$ requests than CG−VNE. Hence, when a physical failure occurs in an equipment (i.e., router and/or link) within $\mathcal{SN}$, the probability to deteriorate $\mathcal{VN}$s is higher. Second, since CG−VNE and B−CGVNE adopt the same reactive mechanism in order to re-embed the impacted $\mathcal{VN}$s, our proposal has less chance to find free physical resources because the residual resources of $\mathcal{SN}$ are less in B−CGVNE since more $\mathcal{VN}$s are accepted. Fortunately, the global revenue of Cloud provider is optimized with our proposal B−CGVNE as illustrated in Figure 5.11(c). Thanks to batch approach, $\mathcal{CP}$ improves the turnover. In fact the balance between the revenue of accepted $\mathcal{VN}$ requests and the penalties generated by B−CGVNE is positive and better than the online strategy CG−VNE. Note that if the rate of adult equipment is equal to 20% and hence old equipment is 50%, B−CGVNE outperforms the revenue of CG−VNE by 26.25%.

Hereby, we set the arrival rate λ_A to 8. We notice in Figure 5.12(a) that B−CGVNE outperforms the online strategy CG−VNE in term of reject rate Q. This positive result motivates more the batch handling of incoming requests. Even the blackout rate of B−CGVNE illustrated in Figure 5.12(b) is roughly similar to CG−VNE despite of accepting more $\mathcal{VN}$ requests. As a consequence, it is clear in Figure 5.12(c) that B−CGVNE makes the best revenue for the $\mathcal{CP}$.

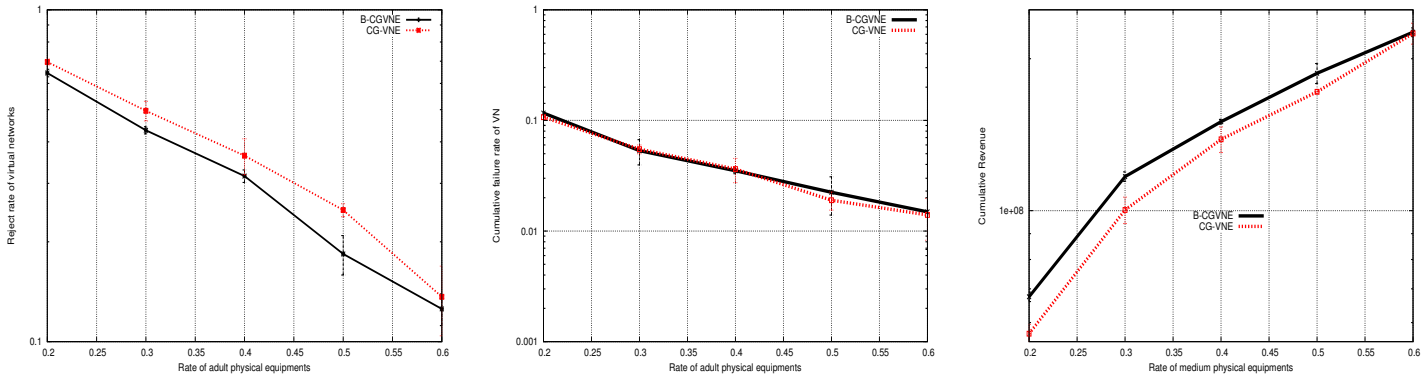
Hereinafter, we set the arrival rate λ_A to 16. Figure 5.13 describes the performance of the two algorithms. It is straightforward to see that the results are similar to those described when setting the λ_A value to 8.

In the following, we describe the performance of the two algorithms when we set the arrival



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.12: B−CGVNE versus CG−VNE - $\lambda_A = 8$



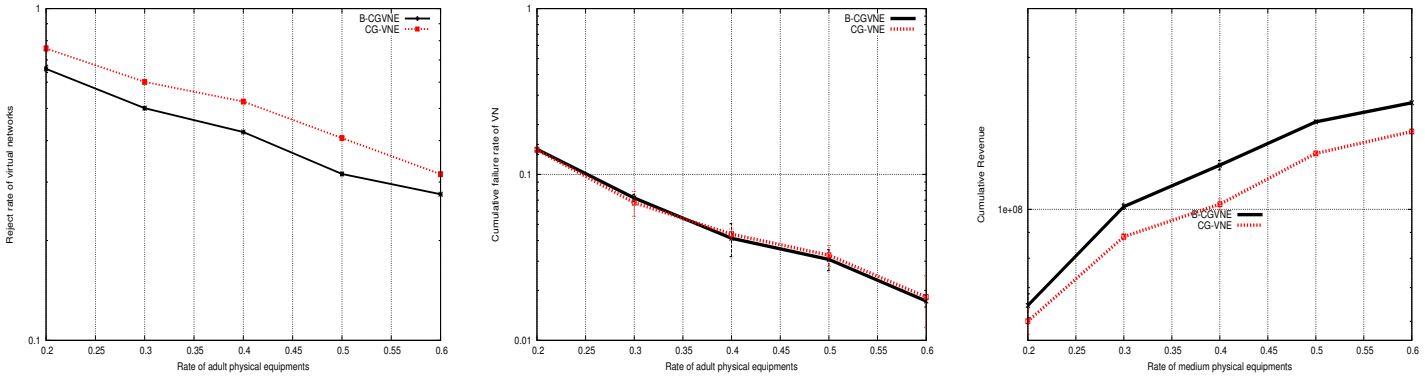
(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.13: B−CGVNE versus CG−VNE - $\lambda_A = 16$

rate λ_A to 32. We notice in Figure 5.14(a) that B−CGVNE outperforms CG−VNE in term of reject rate Q. We remark in Figure 5.14(b) both algorithms have roughly same performances despite that one physical outage may impact more clients for B−CGVNE since acceptance rate for the latter is more the CG−VNE. Indeed, thanks to the reactive mechanism B−CGVNE minimizes the blackout rate. Accordingly, B−CGVNE provides the best revenue to the $\mathcal{CP}$. This clearly deduced from Figure 5.14(c).

5.3.3.4 Comparison with online-splittable strategy

In this section, we gauge the performance of our batch splittable approach B−CG−VNE−splittable, which makes use of Advanced−CG−VNE−splittable algorithm, by comparing it to another batch-based strategy called WiNE [55]. It is noteworthy that the latter strategy embeds



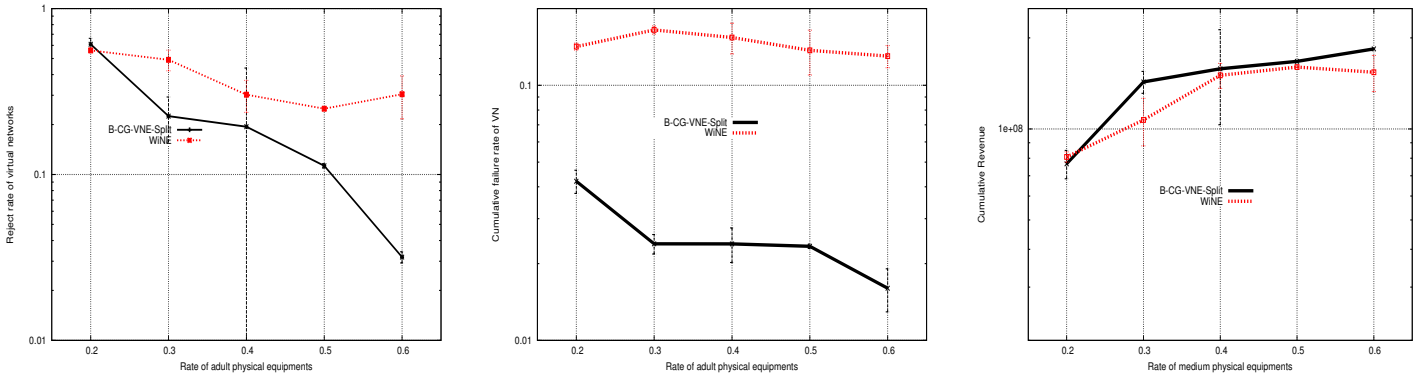
(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.14: B-CGVNE versus CG-VNE - $\lambda_A = 32$

the virtual link in splittable manner and it does not consider the reliability. Hereafter, we vary the arrival rate λ_A from 4 to 32 per 100 seconds and compare the aforementioned algorithms (i.e., B-CG-VNE-splittable and WiNE).

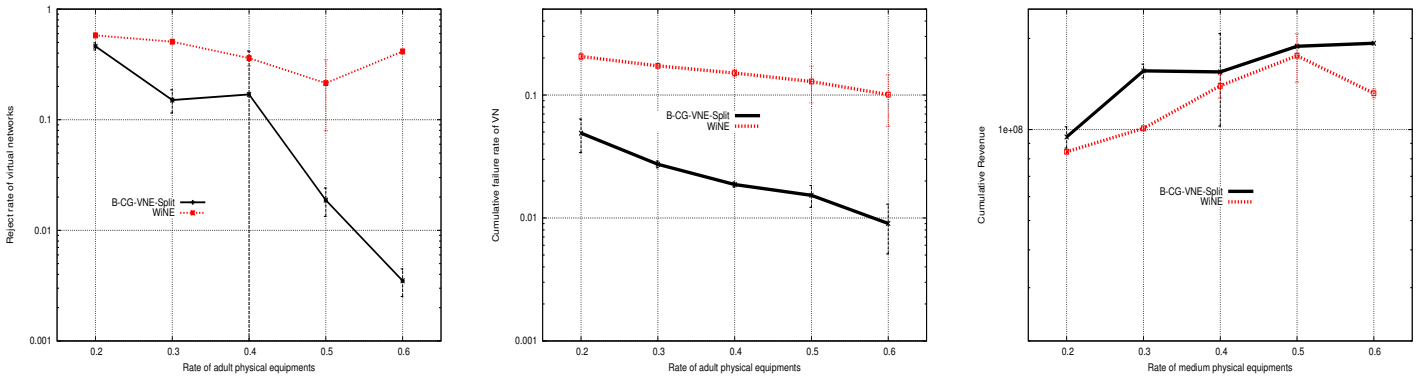
Figure 5.15(a) describes the reject rate of the B-CG-VNE-splittable and WiNE. It is clear that our proposal outperforms WiNE for all the age proportion except when the adult percentage is 20%. At this proportion, the hardware equipments in the $\mathcal{SN}$ have a low reliability and outages frequently occur in the Cloud backbone. Consequently, basing on the reactive mechanism adopted by B-CG-VNE-splittable, our proposal will re-map the impacted virtual resources. So, more physical capabilities will be consumed and the network usage will be higher. Hence, the incoming request may be rejected due to resource shortage. Despite this reactivity, B-CG-VNE-splittable outperforms WiNE when the adult's proportion is 30%, 40%, 50% and 60%. These good results show that our game theory framework leads to an optimized batch $\mathcal{VN}$ mapping. Figure 5.15 represents the performance evaluation of the two algorithms basing on the blackout metric. We notice that the large gap between the two curves proves that B-CG-VNE-splittable succeeds to guarantee some level of reliability. In fact, thanks to the reactive mechanism our proposal reduce the impact of physical failures. As logical consequence of the above results regarding reject rate and blackout rate metrics, B-CG-VNE-splittable outperforms WiNE algorithm in term of earned revenue. This positive result is depicted by Figure 5.15(c).

Hereby, we set the λ_A value to 8 in order to check the resistance of the two strategies to a higher arrival rate. Figure 5.16(a) shows the comparison results in term of reject rate. It is straightforward to see that B-CG-VNE-splittable outperforms WiNE approach for all the physical equipments proportions. We notice in Figure 5.16(b) that B-CG-VNE-splittable has less blackout rate comparing to WiNE strategy. This is thanks to the re-embedding of the impacted



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.15: `B-CG-VNE-splittable` versus `WiNE` - $\lambda_A = 4$

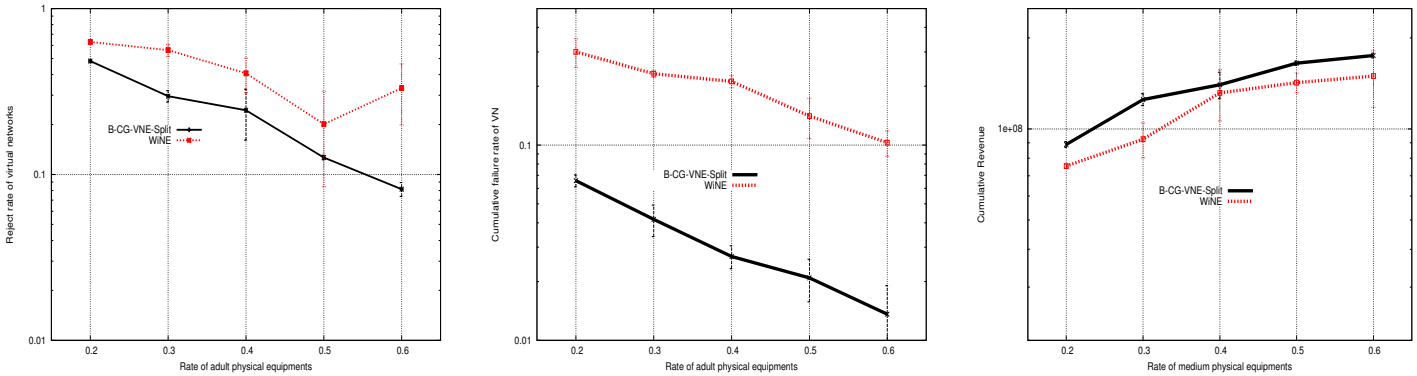


(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.16: `B-CG-VNE-splittable` versus `WiNE` - $\lambda_A = 8$

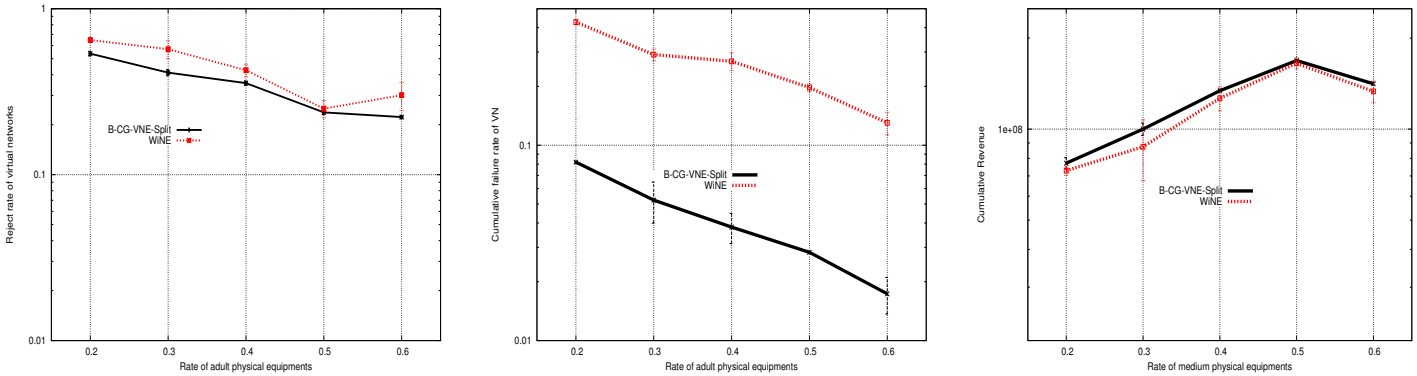
virtual resources when a physical failure occurs. In such way, the recovery mechanism adopted by `B-CG-VNE-splittable` minimizes the damages caused by the hardware outages. According to the above results, `B-CG-VNE-splittable` ensures a better turnover to the Cloud provider comparing to `WiNE` strategy. Indeed, as depicted by Figure 5.16(c) `B-CG-VNE-splittable` outperforms `WiNE` algorithm in term of revenue metric (i.e., G).

In this scenario, we stress more the input by setting the arrival rate λ_A value to 16. We remark in Figure 5.17(a) that `B-CG-VNE-splittable` achieves a better acceptance rate comparing to `WiNE`. Hence, we conclude that our proposal keeps its good results even if we stress it more. Figure 5.17(b) describes the blackout rate for the two algorithms. We notice that our approach has an optimized results basing on the impacted virtual resources comparing to `WiNE` results. As a logical consequence, `B-CG-VNE-splittable` outperforms `WiNE` basing on turnover metric G.



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.17: `B-CG-VNE-splittable` versus `WiNE` - $\lambda_A = 16$



(a) Reject rate: Q (b) Blackout rate: F (c) Revenue: G

Figure 5.18: `B-CG-VNE-splittable` versus `WiNE` - $\lambda_A = 32$

Indeed, Figure 5.17(c) proves that the revenue earned by our proposal is enhanced comparing to the turnover of `WiNE` approach.

In the last scenario, we set the arrival rate λ_A to 32. Figure 5.18(a) shows that `B-CG-VNE-splittable` outperforms `WiNE` strategy basing on reject rate metric. This proves that coupling i) Monte-Carlo Tree Search for decision tree building and searching and ii) game theory framework for a $\mathcal{VN}$ embedding leads to an optimized batch mapping. We notice in Figure 5.18(b) that despite accepting more clients, our proposal achieves optimized results comparing to `WiNE` strategy in term of blackout rate. Consequently, `B-CG-VNE-splittable` ensures higher revenue comparing to `WiNE` algorithm. In fact, Figure 5.18(c) clearly shows the optimized result achieved by our proposal.

5.4 Conclusion

In this chapter, we studied the problem of survivable batch-embedding virtual network. It is formulated as decision tree building and searching problems which are NP-hard. We proposed a novel batch-embedding strategy named BR−VNE based on Monte Carlo Tree Search optimization algorithm. The latter can reach the upper confidence bounds in polynomial time. The proposal aims to maximize the Cloud provider's revenue and minimize the blackout rate of virtual networks caused by $\mathcal{SN}$ failures. Based on extensive simulations, our proposal outperforms the most prominent related work algorithms.

Chapter 6

Conclusion

Contents

6.1 Summary of contributions 127
6.2 Future work 129
6.2.1 Short-term perspectives 129
6.2.2 Mid-term perspectives 130
6.3 Publications 131

This chapter will conclude the thesis and describe some perspectives to the elaborated scientific work. Section 6.1 will summarize our proposed algorithms tackling the reliable virtual network embedding problem. Later on, we will discuss the perspectives and research directions that we are planning to considere as extension to our work in Section 6.2. Finally, Section 6.3 will give an overview of the list of publications that have been achieved during this thesis.

6.1 Summary of contributions

In this thesis, we have addressed the reliable virtual network embedding problem within the Cloud backbone. We have mainly focused on how to maximize the Cloud provider revenue by maximizing the acceptance rate of customers' requests while minimizing the impact of the unplanned physical outages. In other terms, the main objectives of Cloud provider is to improve the turnover by rejecting the minimum of requests as well minimize the penalties induced by Service Level Agreements (SLA) violation due to the network failures. The exposed problem is NP-Hard with multi objective modelization. To cope with the problem intractability, we proceed to its resolution in different steps and three reliable virtual network mapping strategies were proposed while making use of different theoretical approaches. Our contributions are fourfold, hereby we will summarize them.

First, we provided a detailed overview of the reliable virtual network embedding within Cloud network. Our survey exposed a taxonomy of the different survivable strategies according the following criteria: i) centralized or distributed, ii) proactive or reactive, iii) the type of the protected resource (i.e., router and/or link, geographic region), iv) backup use and v) curative or preventive.

Second, in order to overcome the complexity of the survivable virtual network embedding problem, we made use of the Artificial Bee Colony metaheuristic to define our Preventive Virtual Network Embedding strategy denoted PR−VNE. It should be mentionned that PR−VNE deals with reliability by avoiding the instantiation into the non-reliable resources. However, it does not dedicate physical resource backup and does not react when a network outage occurs. Based on extensive simulations, PR−VNE outperforms the most prominent strategies in terms of: i) reject rate of virtual networks, ii) blackoutrate and iii) Cloud provider's revenue.

Third, in order to overcome the absence of a recovery mechanism with PR−VNE algorithm, we define a reactive approach that re-embeds the impacted virtual resources (i.e., router and/or link) by a hardware outage. This proposal called Coordination Game for Virtual Network Embedding (CG−VNE) strongly relies on a coordination game. In fact, during the mapping game, fictitious players collaborate together in order to enhance the provider revenue by improving the embedding quality. CG−VNE tackles the reliability issue basing on two stages: i) preventing the use of non-reliable network equipments and ii) re-embedding the impacted virtual resources by hardware outages in safe resources. Our solution does not rely on backup resources to optimize the use of the network capabilities. However, we define a central controller that periodically receives snpashots from the substrate routers. By doing so, re-mapping the impacted virtual resources will be transparent for the end-user since the migration will be ensured without insignificant service interruption. Extensive simulations show that CG−VNE outperforms the related work strategies in terms of i) rejection rate of virtual networks, ii) rate of clients impacted by physical failures and iii) Cloud provider's turnover.

Finally, we address the batch reliable embedding problem that considers the mapping of a set of virtual networks instead of processing one request as soon as it arrives. In this context, we define a Batch Reliable Virtual Network Embedding (BR−VNE) approach that relies on artificial intelligence approaches. Indeed, the new reliable batch embedding strategy named BR−VNE is based on Monte-Carlo Tree Search algorithm. Our main focus is how to define the best order mapping sequence that optimizes the network resources usage and maximizes the reliability. Consequently, the provider is improved. It is noteworthy that BR−VNE deals with physical failures by using the same reactive strategy adopted in CG−VNE. Based on extensive simulations, the obtained results show that BR−VNE outperforms the related work in terms of i) acceptance rate of virtual network requests, ii) Cloud provider's revenue and iii) rate of virtual network requests impacted by physical failures within the Cloud's backbone.

6.2 Future work

Hereafter, we expose the future work related to our thesis. In fact, we categorize the perspectives in two groups i) short-term and ii) medium-term.

6.2.1 Short-term perspectives

As a short-term planned work, we aim to consolidate our simulation results by implementing and deploying a testbed that supports our proposed strategies: i) `PR-VNE`, ii) `CG-VNE` and iii) `BR-VNE`. Indeed, we achieved the first steps of our contributions' validation by: i) proposing optimized analytical models and strategies and ii) developping a simulator that includes not only our propounded approaches but also the related competitor algorithms. The second step will focus on the experimentation by implementing a real platform that emulates the Cloud's backbone and especially the reliable virtual network embedding process and the communication protocols.

Furthermore, we also aim to evaluate the impact of the migration process, triggered after a hardware failure, on the service continuity. In fact, basing on the reactive mechanism adopted by our strategies once a physical outage occurs in the Cloud's backbone, the impacted virtual routers and their attached virtual links should be migrated to safe substrate resources. Hence, from the provider perspective this migration operation should be transparent to the customer and should not cause any service degradation in order to respect the SLA. Consequently, the central controller should have a recent snapshot from each physical router in the backbone. Thus, we intend to evaluate the network load induced by the snpashot synchronization operations and the recovery period impacting the service continuity.

Moreover, since that our proposed strategies strongly rely on the central controller we have to focus on the communication between the latter and the remaining substrate routers in the Cloud's backbone. In fact, the central controller adopts the reliable mapping strategy and should have a global view on the substrate network. This macro view is ensured by exchanging control messages with the physical routers. We aim to devise a synchronization algorithm that ensures this kind of communication in order to optimize the bandwidth consumption. This new algorithm can be inspired from OpenFlow protocol [96]. In fact, the latter is hugely used to define the communication strategy between the controller and the remaining elements in some of the new network architectures.

Finally, the energetic consumption in Cloud-based environment is continuously raising issues to the provider because of the high expenses required by the cooling systems. Hence, optimizing the energy consumption is becoming a mandatory criterium to evaluate the performance of the new proposed algorithms and especially those tackling problems related to data centers and Cloud computing. In this context, we aim in the near future to define a reliable green virtual network embedding algorithm that deals with: i) SLA constraint, ii) fault-tolerence and iii) energy optimiza-

tion.

6.2.2 Mid-term perspectives

During this thesis, we focused on the reliable virtual network mapping topic within the Cloud's backbone (i.e., inter data centers). However, the bandwidth allocation problem within a data center (i.e., intra data center) is an important issue faced by Cloud providers. In fact, a recent Cisco study [9] showed that the traffic circulating within a data center is estimated to be $6,389$ EB by the end of 2018. Accordingly, this huge volume of bandwidth, transiting within a data center, should be well managed in order to avoid congestion and bottelneck network region. Besides, due to the bandwidth sharing within the data center, any hardware outage may simultaneously cripple many customers' services. Moreover, since our proposals are independent from the reliability formalization, they can be directly applied or lightly updated to any underlying reliability model of the data center network. All these elements, motivate us to deal with the survivable bandwidth allocation within a data center.

Furthermore, the Software Defined Network (SDN) [97] [98] [99] is a new paradigm that defines a new network architecture to cope with the current Internet ossification problem. Ensuring a reliable connection between different entities is a challenging problem that can be considered as an important projection of the elaborated research work in this thesis. In fact, this new network architecture includes: i) a controller that supports all the management policies and ii) the forwording elements that can be defined as light routers without any routing protocol implemention. In this context, the controller is the responsible of the path computing and the bandwidth allocation. Accordingly, the survivable bandwidth allocation in SDN is a suitable perspective to apply our proposed strategies that guarantee some level of reliability.

6.3 Publications

Hereafter, we summarize the publications that have been elaborated during this thesis.

- **Journals**

  1. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, "A Novel Preventive Reliable Embedding Scheme for Virtual Networks within Cloud's Backbone", **submitted to** IEEE Transactions on Network and Service Management (**TNSM**).

- **Conferences**

  1. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, "PR-VNE: Preventive Reliable Virtual Network Embedding Algorithm in Cloud's Network", in IEEE Global Communications Conference (**Globecom**), Atlanta - USA, December 9-13, 2013.
  2. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, "A Reliable Virtual Network Embedding Algorithm based on Game Theory within Cloud's backbone", in IEEE International Conference on Communications (**ICC**), Sydney - Australia, June 10-14, 2014.
  3. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, "A Batch Approach for a Survivable Virtual Network Embedding based on Monte-Carlo Tree Search", in IEEE/IFIP Integrated Management (**IM**) Conference, Ottawa - Canada, Mai 11-15, 2015.

References

[1] P. Papadimitriou, I. Houidi, W. Louati, D. Zeghlache, C. Werle, R. Bless, and L. Mathy, "Towards large-scale network virtualization," *Springer Wired/Wireless Internet Communication*, vol. 7277, pp. 13–25, 2012.

[2] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the Internet impasse through virtualization," *IEEE Computer*, vol. 38, pp. 34 – 41, 2005.

[3] J. Turner and D. Taylor, "Diversifying the Internet," *IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 2, pp. 760 – 756, 2005.

[4] A. Feldmann, "Internet clean-slate design: What and why?" *SIGCOMM Computer Communication Review*, vol. 37, pp. 59 – 64, 2007.

[5] "Offshore providers without a cloud strategy will risk their long-term future," Gartner Inc, Tech. Rep., 2013. [Online]. Available: http://www.gartner.com/newsroom/id/2562415

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," *Technical Report Identifier: EECS-2009-28*, 2009.

[7] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," in *Proceedings of the 2009 conference on Hot topics in cloud computing*, ser. HotCloud'09, 2009.

[8] "Calculating the cost of data center outages," Ponemon Institute, Tech. Rep., 2011. [Online]. Available: http://goo.gl/SC6Ve

[9] Cisco, "Cisco global cloud index: Forecast and methodology, 2013 - 2018," *Cisco white paper*, pp. 1–41, 2014.

[10] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, pp. 50–55, 2009.

[11] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems journal*, vol. 25, pp. 599–616, 2009.

[12] NIST, "The NIST Definition of Cloud Computing, SP 800-145," *http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf*.

[13] I. Fajjari, N. Aitsaadi, and P. Guy, "Cloud networking: An overview of virtual network embedding strategies," *IEEE Global Information Infrastructure Symposium*, pp. 1–7, 2013.

[14] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," *SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 67–74, 2010.

[15] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Gener. Comput. Syst.*, vol. 28, no. 3, pp. 583–592, 2012.

[16] M. Mahjoub, A. Mdhaffar, R. B. Halima, and M. Jmaiel, "A comparative study of the current Cloud Computing technologies and offers," *International Symposium on Network Cloud Computing and Applications*, pp. 131–134, 2011.

[17] S. Babu, M. Hareesh, J. Martin, S. Cherian, S.Cherian, and Y. Sastri, "System performance evaluation of para virtualization, container virtualization and full virtualization using xen, openvz and xenserver," *IEEE International Conference on Advances in Computing and Communications*, vol. 4, pp. 247–250, 2014.

[18] VMware, "Understanding full virtualization, paravirtualization, and hardware assist," *VMware white paper*, pp. 1–17, 2007.

[19] "VMware," *http://www.vmware.com*.

[20] "VirtualBox," *http://www.virtualbox.org*.

[21] "OpenVZ," *http://openvz.org*.

[22] "Xen," *http://www.xenproject.org*.

[23] D. Clark, K. Sollins, J. Wroclawski, and T. Faber, "Addressing reality: An architectural response to real-world demands on the evolving internet," *SIGCOMM Computer Communication Review*, pp. 247–257, 2003.

[24] P. Ferguson and G. Huston, "What is a VPN?" *Technical report, Cisco Systems*, 1998.

[25] E. Rosen and Y. Rekhter, "BGP/MPLS VPNs," *RFC 2547*, 1999.

[26] ——, "BGP/MPLS IP Virtual Private Networks (VPNs)," *RFC 4364*, 2006.

[27] N. Chowdhury, "Identity management and resource allocation in the network virtualization environment," *PhD thesis - University of Waterloo*, 2008.

[28] N. M. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, pp. 862–876, 2010.

[29] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges," *IEEE International Conference on Advanced Information Networking and Applications*, pp. 27–33, 2010.

[30] C. Weinhardt, A. Anandasivam, B. Blau, N. Borissov, T. Meini, W. Michalk, and J. Stosser, "Cloud Computing ? A Classification," *Springer Business and Information Systems Engineering*, vol. 1, pp. 391–399, 2009.

[31] A. Osterwalder, "The business model ontology ? a proposition in a design science approach," *PhD thesis - University of Lausanne*, 2004.

[32] C. Technologies Inc, "The avoidable cost of downtime," *Research Report*, 2010.

[33] D. G. Andersen, "Theoretical approaches to node assignment," 2002, http://www.cs.cmu.edu/?dga/papers/andersen-assign.

[34] J. Kleinberg, "Approximation algorithms for disjoint paths problems," *PhD thesis - MIT*, 1996.

[35] S. G. Kolliopoulos and C. Stein, "Improved approximation algorithms for unsplittable flow problems," *In Proc. IEEE Symposium on Foundations of Computer Science*, 1997.

[36] A. Fischer, J. Botero, M. T. Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys and Tutorials*, vol. 15, pp. 1888–1906, 2013.

[37] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (abc) algorithm," *Elsevier Applied Soft Computing*, vol. 8, pp. 687–697, 2008.

[38] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (abc) algorithm and applications," *Springer, Artificial Intelligence Review*, vol. 42, pp. 21–57, 2012.

[39] O. Soualah, I. Fajjari, N. Aitsaadi, and A. Mellouk, "PR-VNE: Preventive Reliable Virtual Network Embedding Algorithm in Cloud's Network," *IEEE Global Communications Conference (Globecom)*, pp. 1303–1309, 2013.

[40] ——, "A Reliable Virtual Network Embedding Algorithm based on Game Theory within Cloud's backbone," *IEEE International Conference on Communications (ICC)*, pp. 2975–2981, 2014.

[41] R. B. Myerson, *Game Theory: Analysis of Conflict*, ser. Harvard University Press. Cambridge, Massachusetts, 1997.

[42] O. Soualah, I. Fajjari, N. Aitsaadi, and A. Mellouk, "A batch approach for a survivable virtual network embedding based on Monte-Carlo Tree Search," *IEEE/IFIP Integrated Management (IM) Conference*, 2015.

[43] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A Survey of Monte Carlo Tree Search Methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, pp. 1–43, 2012.

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

[45] C. Wang, Y. Yuan, C. Wang, X. Hu, and C. Zheng, "Virtual bandwidth allocation game in data centers," *IEEE International Conference on Information Science and Technology*, 2012.

[46] Y. Zhou, Y. Li, G. Sun, D. Jin, L. Su, and L. Zeng, "Game theory based bandwidth allocation scheme for network virtualization," *IEEE Global Communications Conference*, 2010.

[47] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNE-AC: Virtual Network Embedding Algorithm based on Ant Colony Metaheuristic," *IEEE ICC*, 2011.

[48] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, pp. 243–278, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397505003798

[49] T. Stützle and H. H. Hoos, "MAX-MIN ant system," *Future Gener. Comput. Syst.*, vol. 16, pp. 889–914, 2000.

[50] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNR Algorithm: A Greedy Approach For Virtual Networks Reconfigurations," in *GLOBECOM*. IEEE, 2011, pp. 1–6.

[51] ——, "Adaptive-VNE: A Flexible Resource Allocation For Virtual Network Embedding Algorithm," *IEEE GLOBECOM*, pp. 2640–2646, 2012.

[52] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 17–29, 2008.

[53] A. Jarray and A. Karmouch, "Decomposition Approaches for Virtual Network Embedding With One-Shot Node and Link Mapping," *IEEE/ACM Transactions on Networking*, 2014.

[54] X. Chang, B. Wang, and J. Liu, "Network State Aware Virtual Network Parallel Embedding," *IEEE Performance Computing and Communications Conference (IPCCC)*, pp. 193–194, 2012.

[55] M. Chowdhury, M. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, pp. 206–219, 2012.

[56] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive virtual network provisioning," *ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures (VISA)*, pp. 41–48, 2010.

[57] I. Houidi, W. Louati, D. Zeghlache, and S. Baucke, "Virtual resource description and clustering for virtual network discovery," *IEEE ICC Workshop*, 2009.

[58] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," *ICC*, pp. 565–578, 2008.

[59] G. 5000, *https://www.grid5000.fr/*.

[60] W.-L. Yeow and C. W. U. C. Kozat, "Designing and embedding reliable virtual infrastructures," *ACM SIGCOMM workshop on Virtualized Infrastructure Systems and Architectures (VISA)*, pp. 33–40, 2010.

[61] G. Koslovski, W.-L. Yeow, C. Westphal, T. T. Huu, J. Montagnat, and P. Vicat-Blanc, "Reliability support in virtual infrastructures," *International Conference on Cloud Computing Technology and Science (CLOUDCOM)*, pp. 49–58, 2010.

[62] G. Koslovski, P. V.-B. Primet, and A. S. Charao, "Vxdl: Virtual Resources and Interconnection Networks Description Language," *GridNets*, 2008.

[63] H. Yu, V. Anand, C. Qiao, and G. Sun, "Cost efficient design of survivable virtual infrastructure to recover from facility node failures," *IEEE International Conference on Communications (ICC)*, pp. 1–6, 2011.

[64] M. Rahman and R. Boutaba, "SVNE: Survivable virtual network embedding algorithms for network virtualization," *IEEE Transactions on Network and Service Management (TNSM)*, pp. 1–14, 2012.

[65] W. Yan, S. zhi Chen, X. Li, and Y. Wang, "RMap: An algorithm of virtual network resilience mapping," *International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, pp. 1–4, 2011.

[66] A. Atlas and A. Zinin, "Basic specification for ip fast reroute: Loop-free alternates," *RFC 5286*, 2008.

[67] X. Liu, C. Qiao, and T. Wang, "Robust Application Specific and Agile Private (ASAP) networks withstanding multi-layer failures," *IEEE Optical Fiber Communication*, 2009.

[68] H. Yu, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures," *IEEE Global Telecommunications Conference (GLOBECOM)*, pp. 1–6, 2010.

[69] H. Yu, C. Qiao, A. Vishal, and X. Liu, "On the survivable virtual infrastructure mapping problem," *IEEE ICCCN*, 2010.

[70] G. S. H. Yu, L. Li, V. Anand, and H. Di, "The framework and algorithms for the survivable mapping of virtual network onto a substrate network," *IETE Technical Review*, 2011.

[71] A. P. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. N. Chuah, Y. Ganjali, and C. Diot, "Characterization of Failures in an Operational IP Backbone Network," *IEEE/ACM Transactions on Networking*, vol. 16, pp. 749–762, 2011.

[72] J. Kleinberg, "Approximation algorithms for disjoint paths problems," *PhD thesis - MIT*, 1996.

[73] S. G. Kolliopoulos and C. Stein, "Improved approximation algorithms for unsplittable flow problems," *In Proc. IEEE Symposium on Foundations of Computer Science*, 1997.

[74] R. Xu and D. C. W. II, "Survey of Clustering Algorithms," *IEEE Transactions on neural networks*, vol. 16, pp. 645–678, 2005.

[75] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. Vol. 1*, pp. 281–297, 1967.

[76] S. Ghosh, "Optimizer augmented clustering: a study over iterative k-means," *PhD thesis*, 2011.

[77] R. Xu and D. W. II, "Survey of Clustering Algorithms," *IEEE Transcations on Neural Networks*, 2005.

[78] T. Leong, P. Shor, and C. Stein, "Implementation of a combinatorial multicommodity flow algorithm," *Discrete Mathematics and Theoretical science*, 1992.

[79] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network flows: Theory, algorithms, and applications," *Prentice Hall*, 1993.

[80] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," *Proceedings of IEEE INFOCOM*, pp. 594–602, 1996.

[81] R. Cooper, *Coordination Games*, ser. Business and Economics. Cambridge University Press, 1999.

[82] D. Monderer and S. Shapley, "Fictitious play property for games with identical interests," *Journal of economic theory*, pp. 258–265, 1996.

[83] ——, "Games and economic behavior," *Journal of economic theory*, vol. 14, pp. 124–143, 1996.

[84] P. Gajowniczek and M. Pioro and A. Arvidsson, "VP reconfiguration through simulated allocation," *13th Nordic Teletraffic Sem*, pp. 251–260, 1996.

[85] M. Pioro and P. Gajowniczek, "Simulated allocation: A suboptimal solution to the multicommodity flow problems," *11th UK Teletraffic Symposium*, 1994.

[86] V. Srivastava and all, "Using game theory to analyze wireless ad hoc networks," *IEEE Commun. Surveys Tutorials*, pp. 46–56, 2005.

[87] B. G. Jozska and Z. Kiraly and G. Magyar and A. Szentesi, "An efficient algorithm for global path optimization in mpls networks," *Springer Optimization and Engineering*, vol. 2, pp. 321–347, 2001.

[88] J. Y. Yen, "Finding the K Shortest Loopless Paths in a Network," *Management Science*, vol. 17, pp. 712–716, 1971.

[89] C. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. Lam, and M. Rosenblum, "Optimizing the migration of virtual computers," *Proceedings of the 5th symposium on Operating systems design and implementation, ACM SIGOPS Operating Systems Review*, pp. 377–390, 2002.

[90] N. Huyn, R. Dechter, and J. Pearl, "Probabilistic analysis of the complexity of A*," *Artificial Intell*, vol. 15, pp. 241 – 254, 1980.

[91] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2003.

[92] J. Clausen, "Branch and Bound Algorithms - Principles and Examples," *Department of Computer Science, University of Copenhagen*, pp. 1–30, 1999.

[93] M. P. Schadd, M. H. Winands, M. J. Tak, and J. W. Uiterwijk, "Single-player Monte-Carlo tree search for SameGame," *A Special Issue on Artificial Intelligence in Computer Games: AICG*, vol. 34, pp. 3–11, October 2012.

[94] K.-H. Chen, D. Du, and P. Zhang, "Monte-Carlo Tree Search and Computer Go," *Springer - Advances in Information and Intelligent Systems*, vol. 251, pp. 201–225, 2009.

[95] L. Kocsis and C. Szepesvári, "Bandit Based Monte-Carlo Planning," in *Proceedings of the 17th European Conference on Machine Learning*, ser. ECML'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 282–293.

[96] "OpenFlow Specification," *Open Networking Foundation - OpenFlow v1.4*, 2013.

[97] B. Astuto, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys and Tutorials*, vol. 16, pp. 1617–1634, 2014.

[98] S. Sezer, "Are we ready for SDN? Implementation challenges for software defined networks," *IEEE Communications Magazine*, 2013.

[99] A. Leon-Garcia, H. Bannazadeh, and Q. Zhang, "Openflow and SDN for Clouds," *Cloud Services, Networking, and Management*, pp. 129–152, 2015.

Résumé détaillé

Inroduction

De nos jours, aussi bien les industriels que les instituts académiques se fient de plus en plus aux services offerts par le Cloud computing. Ce nouveau concept a offert des outils permettant de palier aux problèmes liés à l'architecture traditionnelle d'Internet. Comme il a été défini par le National Institue of Standards and Technology (NIST), le Could computing est un modèle qui permet l'accès aux ressources déployées sur le réseau à la demande. Les ressources sont partagées et ses capacités sont configurables en fonction des besoins. Le client peut bénéficier d'une flexibilité importante avec un effort minimal de gestion. La technologie principale qui a assuré cette ampleur du recours au Cloud est la virtualisation. En fait, grâce à cette technologie, le fournisseur du service Cloud expose des capacités physiques (i.e., stockage, calcul, réseau) ou logiciels pour être exploités par des clients. En effet, dans un environnement totalement virtualisé, le fournisseur peut servir plusieurs clients simultanément tout en assurant l'isolation et la transparence entre les différents services hébergés dans la même ressource physique.
L'architecture de base de ce genre de ce système complètement virtualisé peut être décrite par un réseau coeur (*backbone*) qui détient des routeurs supportant la virtualisation. Ce type de routeur permet d'offrir plusieurs implémentations de la pile protocolaire réseau afin de servir le maximum de demandes clients. Parallèlement aux routeurs, les liens physiques connectants les différents noeuds du réseau peuvent supporter simultanément plusieurs trafics issus de différents utilisateurs. Ce réseau dorsal assure l'interconnexion des data centers localisés dans des zones géographiques lointaines. En se basant sur une architecture Cloud, le fournisseur du service peut définir plusieurs types de services qui sont répertoriés selon NIST en trois grandes classes, à savoir :

- Software as a Service (SaaS) : Le fournisseur de service expose ses capacités logicielles aux clients. Les applications exposées sont accessibles via différentes interfaces, clients légers, navigateur Web, terminaux mobiles, etc.

- Platform as a Service (PaaS) : le consommateur peut déployer sur l'infrastructure du four-

nisseur ses propres applications, dans la mesure de compatibilité logicielle. L'utilisateur gère mais ne contrôle pas l'infrastructure Cloud sous-jacente (i.e., réseau, serveurs, systèmes d'exploitation, bases de données, stockage). Par contre, il gère ses applications installées et a la possibilité de configurer l'environnement logiciel sur lequel il travaille.

- Infrastructure as a Service (IaaS) : Le fournisseur de ce type de service expose son infrastructure pour être utilisée par des clients qui peuvent être des industriels, des instituts académiques ou même des utilisateurs finaux. Un client a la possibilité de provisionner des capacités de traitement, de stockage, de réseau en unité élémentaire et pour un temps préalablement défini. L'utilisateur ne gère et ne contrôle pas l'infrastructure Cloud sous-jacente, mais a le contrôle sur les systèmes d'exploitation, les bases de données et les applications déployées.

Dans le cadre de cette thèse, on s'intéresse aux services réseaux offerts par les founisseurs Cloud. Plus particulièrement, on focalise notre recherche sur un type de service, nommé Network as a Service (NaaS), qui est une sous-classe de la classe IaaS. Dans ce contexte, le fournisseur expose ses capacités réseaux, à savoir les routeurs et les liens physiques, pour le déploiement des services clients. En d'autres termes, les clients s'approvisionnent des capacités réseaux afin de définir à la volée et pour une période prédéfinie une topologie de réseau virtuel qui satisfait leurs exigences. De point de vue fournisseur de service NaaS, ce dernier a intérêt à recourir à une technique d'instanciation de réseaux virtuels qui lui permet de maximiser son gain et de mutualiser les ressources physiques. Cette technique doit respecter l'accord de niveau de service (i.e., Service Level Agreement) qui constitue un contrat entre le client et le fournisseur de service pour définir les exigences du premier et l'engagement du deuxième. Les interruptions de service dûes aux pannes des équipements physiques induisent des penalités au fournisseur. Par conséquent, le revenu du fournisseur sera dégradé. Théoriquement, il a été prouvé que le processus de mapping (i.e., l'instanciation) des réseaux virtuels est un problème de type NP-dur. De même, la considération de la fiabilité comme nouvelle contrainte rajoute une compléxité au problème de base. Dans les travaux de recherche menés au cours de cette thèse, nous avons essayé de répondre aux questions relatives à l'instanciation des réseaux virtuels tout en prenant en compte la fiabilité afin de proposer des mécanismes tolérants aux pannes. Dans un premier temps, nous avons introduit un algorithme préventif, nommé PR-VNE[1], qui favorise l'utilisation des équipements physiques fiables mais sans aucun mécanisme de reconfiguration. Dans un second temps, nous avons enrichi cet aspect préventif de notre première contribution par une réactivité qui assure une re-instanciation des ressources virtuelles touchées par les pannes. Cet enrichissement a été proposé dans le cadre d'un nouvel algo-

[1]O. Soualah, I. Fajjari, N. Aitsaadi and A. Mellouk, "PR-VNE : Preventive Reliable Virtual Network Embedding Algorithm in Cloud's Network", in IEEE Global Communications Conference (**Globecom**), 2013.

rithme basée sur la théorie des jeux, nommé CG-VNE[2]. En fin, nous avons considéré le traitement par lot des requêtes en considérant la fiabilité. Cette nouvelle approche, nommée BR-VNE[3], est basée sur la stratégie Monte-Carlo Tree Search permettant de calculer, en temps polynomial, la séquence des réseaux virtuels maximisant le revenu du fournisseur.

La suite de ce chapitre est organisé comme suit. Dans la première section, on décrit les différents algorithmes de l'état de l'art étudiant le mapping des réseaux virtuels tout en considérant la fiabilité. Ensuite, dans la deuxième section on explique notre première contribution PR-VNE. Puis, on expose notre deuxième algorithme CG-VNE dans la troisième section. Dans la quatrième section, on décrit notre troisième contribution BR-VNE. La cinquième section conclu ce chapitre et décrit les principales perspectives.

État de l'art

Dans ce chapitre, on présente initialement un aperçu sur les algorithmes traitant la problèmatique de l'instanciation des réseaux virtuels. Ces algorithmes ne tiennent en compte ni la fiabilité ni la tolérance aux pannes. Les propositions dans ce contexte peuvent être répertoriées selon deux grandes classes : i) statique et ii) dynamique. La première classe d'algorithme ne procède pas à la reconfiguration réseau après l'instanciation des requêtes. Cependant, la deuxième classe tient compte du départ des réseaux virtuels, de la libération des ressources physiques et des changements de l'état du réseau physique sous-jacent pour reconfigurer les capacités résiduelles. En outre, ce chapitre apporte principalement une synthèse des méthodes d'instanciation des réseaux virtuels en considérant la fiabilité. Ces algorithmes peuvent être classés selon plusieurs critères. Principalement, les critères considérées sont : i) centralisé ou distribué, ii) type de ressource protégée, iii) manière d'instancier les liens virtuels, iv) preventive et v) proactive ou réactive. Il est à noter qu'une seule méthode [56] propose une approche distribuée pour assurer un certain degré de fiabilité. L'incovénient majeur de cette classe de stratégie est le besoin de déployer des routeurs autonomes ce qui n'est pas le cas des équipements réseaux actuels. Les autres méthodes se déploient sur une unité (i.e., machine serveur) centrale, appelée contrôleur central, qui a une vue générale sur l'état du réseau d'où le nom centralisé pour ces approches. Le rôle de ce contrôleur central est d'adopter une politique de gestion de l'acceptation des requêtes clients arrivées. Les méthodes centralisées peuvent être classées aussi en se basant sur le type de la ressource protégée contre les pannes. Dans la littérature, on trouve des méthodes qui assurent la protection : i) des routeurs [60] [61] [63], ii) des liens [65] [64], iii) des routeurs et des liens [67] et iv) des zones géographiques [70] [68]. La plupart des algorithmes centralisés se basent sur des ressources de secours préalablement allouées afin d'assurer la reprise

[2]O. Soualah, I. Fajjari, N. Aitsaadi and A. Mellouk, "A Reliable Virtual Network Embedding Algorithm based on Game Theory within Cloud's backbone", in IEEE International Conference on Communications (**ICC**), 2014.

[3]O. Soualah, I. Fajjari, N. Aitsaadi and A. Mellouk, "A Batch Approach for a Survivable Virtual Network Embedding based on Monte-Carlo Tree Search", in IEEE/IFIP Integrated Management (**IM**) Conference, 2015.

et la continuité de service suite à une panne. Néanmoins, la réservation des ressources secondaires (i.e., de secours) impose un surcoût au niveau de l'utilisation des capacités réseaux, notamment la bande passante, de sorte que la qualité du processus d'acceptation des requêtes est détériorée. En d'autres termes, les ressources physiques qui pourraient servir pour héberger des nouvelles requêtes seront utilisées comme ressources de secours, génerant ainsi l'éventuel acceptation des nouveaux clients. Cette classe d'algorithme est nommée la classe proactive où la méthode prend l'initiative pour faire face aux pannes. Les méthodes qui se focalisent à remédier uniquement les pannes surgissant aux niveaux des routeurs ou des liens physiques ont pour inconvénient majeur celui de tenir en compte qu'un seul type de panne. Par contre, aucune garantie ne justifie qu'une panne peut impacter la ressource non protégée. Par ailleurs, des travaux ont proposé des algorithmes pour palier à cet inconvénient en introduisant des méthodes qui traitent aussi bien les pannes des liens que celles surgissant au niveau des routeurs. Malheuresement, les approches proposées dans ce contexte souffrent d'une utilisation excessive de ressources physiques ce qui détériore directement le taux d'acceptation des réseaux virtuels. Concernant les méthodes fournissant une protection régionale contre les pannes, les auteurs de ces travaux ont manqué de bien justifier leurs hypothèses. De plus, les algorithmes traitant ce genre de panne ont un surcoût considérable sur l'utilisation des ressources du réseau.
Il est à noter que toutes les approches tolérantes aux pannes traitent les requêtes clients dès leur arrivée. Aucune d'entre elles n'a proposé un traitement par lot (i.e., batch) des réseaux virtuels.

Approche preventive pour l'instanciation des réseaux virtuels

Dans ce chapitre, nous avons proposé un algorithme préventif nommé Preventive Reliable Virtual Network Embedding (`PR-VNE`) qui veille à éviter au maximum les ressources (i.e., routeurs et liens) réseaux non fiables (i.e., ayant une grande probabilité de subir une panne). L'objectif de cette proposition est de maximiser le revenu du fournisseur du service Cloud par l'amélioration du taux d'accéptation et la réduction des penalités causées par les pannes physiques. Cette proposition se base principalement sur la métaheuristique de la colonie d'abeille (Artificial Bee Colony - ABC). La nouveauté principale de cet algorithme est qu'il ne résèrve pas de ressource de secours mais il évite lors du mapping les ressources non fiables. Si une panne surgit alors `PR-VNE` ne procède à aucun traitement de reconfiguration des ressources virtuelles impactées. `PR-VNE` expolite l'aspect distribué de la métaheuristique de la colonie d'abeille. Initialement, les routeurs virtuels d'accès sont déployés en respectant leur type. De même, les liens virtuels reliant deux routeurs virtuels d'accès sont instanciés. Le deploiment des liens se fait à la base de l'algorithme du plus court chemin tout en proposant une nouvelle métrique pour prendre en compte la fiabilité et les capacités résiduelles des liens physiques. Cette nouvelle métrique favorise les ressources ayant une grande fiabilité et assure un équilibrage de charge. Une fois l'étape de l'instanciation des routeurs virtuels

d'accès ainsi que leurs connexions est achevée, le reste de la requête virtuelle est exploité pour la diviser en des topologies virtuelles élémentaires. En appliquant le principe de diviser pour régner, des topologies élémentaires, appelées Solution Components, sont générées. Ces topologies ont pour centre un noeud virtuel central ainsi qu'un sous ensemble de ses liens sous-jacents qui seront utilisés par la suite dans le processus de mapping. Ces topologies étoiles seront, par la suite, mappées séquenciellement dans le même ordre de leur génération. Par conséquent, le traitement qu'on va décrire ci-dessous concerne chacune de ces topologies élémentaires. Pour chaque topologie étoile, les routeurs du réseau physique sont partitionnés (i.e., clustering process) selon l'algorithme K-Means. Cette tâche est assurée par les abeilles de type scout, qui ont pour rôle de chercher les potentiels routeurs physiques pouvant héberger le centre de la topologie élémentaire. Ce partitionnement se base sur des coordonnées virtuelles que nous avons défini aux routeurs physiques pour pouvoir les grouper dans des boules homogènes dans $\Re^5$. Les routeurs appartenant au même cluster (i.e., partition) ont un dégré de similarité élévé exprimé par une distance euclidienne courte entre ces noeuds physiques. Après avoir procédé à la répartition du réseau physique, le rôle des abeilles de type worker arrive suite à la décision de la reine (i.e., onlooker). En fait, la reine sélectionne le partitionnement le plus intéressant du point de vue de mapping pour l'exploitation postérieure des abeilles de type worker. Cette séléction se fait principalement à la base des capacités résiduelles aux niveaux des noeuds physiques, la distance vers le routeur physique hébergeant le voisin (i.e., routeur) virtuel du centre de la topologie élémentaire et la fiabilité des ressources physiques. En effet, ce type d'abeille va simuler le mapping du noeud virtuel en question dans les routeurs physiques situés dans le partitionnement choisi. Finalement, la reine se base sur la qualité de l'instanciation de chacune des abeilles de type worker pour sélectionner la meilleure solution. Les critères de cette sélection favorisent des ressources physiques ayant la plus grande fiabilité afin d'éviter au maximum l'impact des pannes des équipements tout en assurant un équilibrage de charge du réseau.
Parallèlement, nous avons proposé une variante qui respecte exactement ces étapes mais instancie les liens virtuels de manière à partager la demande requise de la bande passante sur plusieurs chemins physiques. Cette approche de traitement des liens virtuels fournit plus de flexibilité et permet d'avoir plus de chance pour instancier les requêtes virtuelles. Afin de trouver une solution optimale, une formalisation en programme linéaire a été élaborée et la résolution est assurée par le solveur GLPK. De même, cette modélisation respecte les objectifs préalablement mentionnés, à savoir favorisation des ressources ayant une grande fiabilité tout en assurant l'équilibrage de charge.

Afin de valider notre nouvelle proposition, nous avons procédé à la simulation pour se comparer aux différentes stratégies de l'état de l'art. Les tests étaient concluants en dépit de l'absence d'un mécanisme curatif qui reconfigure le réseau suite aux pannes surgissant au niveau des équipements physiques.

Approche réactive pour l'instanciation des réseaux virtuels

Un algorithme réactif est ainsi proposé pour reconfigurer les ressources virtuelles impactées par les pannes afin de minimiser les penalités. Pour palier à la limitation de PR−VNE, le nouvel algorithme nommé Coordination Game for Virtual Network Embedding (CG−VNE) adopte aussi bien le mécanisme préventif que le réactif pour minimiser les impacts des pannes. CG−VNE est conçu en se basant sur la théorie des jeux. En fait, cette dernière fournit un ensemble d'outils mathématiques aidant à la modélisation et à la résolution de plusieurs types de problèmes. Nous avons conçu un jeu collaboratif pour l'instanciation des réseaux virtuels où les joueurs coopèrent entre eux afin d'atteindre un optimum social qui maximise le revenu du fournisseur tout en minimisant le taux de rejet et l'impact des pannes sur les requêtes hébergées dans le réseau physique. Nous avons conçu deux jeux imbriqués pour l'instanciation des réseaux virtuels. Le premier est défini pour le mapping des routeurs virtuels et le second est pour celui des liens virtuels. En effet, pour évaluer la qualité du mapping d'un noeud virtuel, nous avons besoin de quantifier la pertinance du mapping de ses liens virtuels attachés. On va décortiquer ces deux jeux dans la suite. Initialement, chaque routeur virtuel est instancié dans un routeur physique. Chaque routeur virtuel est associé à un joueur fictif qui est le responsable du mapping du noeud virtuel en question. Ensuite, l'ensemble d'actions (appelées aussi stratégies) est défini pour chacun des routeurs virtuels. Il est à noter que le joueur choisira l'action qui maximise son profit à chaque itération. Vu qu'il y a une intéraction continue entre les joueurs, le profit de chaque joueur est directement influencé par le choix des autres. Dans notre proposition, les joueurs partagent la même fonction de profit de telle manière à assurer une collaboration implicite entre eux. Ce genre de jeu est appelé Identical Interest Game (IIG) ou Jeu à Intérêt Identique. Cette collaboration assure la convergence vers un optimum social qui maximise le revenu du fournisseur du service Cloud. En effet, nous avons démontré l'existence de l'équilibre de Nash (Nash Equilibrium - NE) qui correspond à l'optimum social dans notre formalisation. Dans notre jeu de mapping des routeurs virtuels, les actions constituent l'ensemble de voisinage du routeur physique en question. Afin de converger vers l'équilibre de Nash, on opte pour l'utilisation de la stratégie Best Response Algorithm où chaque joueur sélectionne l'action qui maximise sa fonction de profitabilité. En d'autres termes, chacun des joueurs choisit séquenciellement l'action qui maximise son profit. Dans cette étape, l'ordre de l'intervention des joueurs change d'une itération à une autre afin d'éviter la convergence vers un optimum local. Par conséquent, la qualité du mapping final sera meilleure. Concernant l'instanciation des liens virtuels, on définit un joueur fictif pour chaque lien virtuel. L'ensemble d'actions conçues pour ce type de joueur sont les K-plus courts chemins (i.e., K-shortest paths) ayant assez de bande passante pour accueillir le lien virtuel considéré. Semblablement au jeu défini pour le mapping des routeurs, celui conçu pour le mapping des liens virtuels est un Jeu à Intérêt Identique (IIG). Dans ce jeu, seuls les liens virtuels attachés à un routeur virtuel participent au jeu et non pas la totalité des liens virtuels dans la requête client. Pareillement au jeu

défini pour l'instanciation des routeurs, celui conçu pour les liens change aléatoirement l'ordre des liens virtuels afin d'éviter une convergeance vers un optimum local.
CG−VNE assure la fiabilité en se basant sur deux étapes. La première est au moment de l'instanciation du réseau virtuel et la seconde lors des pannes. En effet, les fonctions de profitabilité définies pour les deux jeux favorisent l'utilisation des équipements les plus fiables. De cette façon, l'impact des pannes sera minimisé. Ce mécanisme rejoint la prévention adoptée par PR−VNE. Lorsqu'une panne surgit dans le réseau au niveau des routeurs ou des liens physiques, on utilise la même procédure de mapping que celle adoptée par CG−VNE. Il est à souligner qu'aucune ressource de secours n'est préalablement réservée pour assurer la reconfiguration afin d'eviter de surcharger le réseau avec le backup. Avec cette démarche réactive on ne paralise pas le processus d'acceptation des nouvelles requêtes et on assure la continuité de service tout en minimisant l'impact des pannes physiques. Parallèlement à cette variante de CG−VNE, une autre variante a été proposée tout en respectant le framework susmentionné. Cette nouvelle variante instancie les liens virtuels sur plusieurs chemins physiques de manière à améliorer le taux d'acceptation des requêtes. En fait, le lien virtuel qui ne peut être hébergé sur un seul chemin à cause du manque des ressources réseaux, peut être accueilli en divisant la bande passante demandée sur plusieurs chemins. Cette nouvelle variante se base sur une modélisation en un programme linéaire et une résolution par le solveur GLPK. Cette formalisation a pour objectif d'assurer l'équilibrage de charge tout en favorisant l'utilisation des capacités réseaux avec une grande fiabilité.

Pour évaluer notre modèle théorique et mesurer les performances de CG−VNE, nous avons procédé par simulation pour se comparer aux différentes stratégies de l'état de l'art. Malgré le non recours à des ressources de secours, les résultats obtenus sont plus performantes du point de vue taux de rejet et taux de requêtes virtuelles impactées par les pannes. Par conséquent, le revenu déduit par le fournisseur Cloud adoptant CG−VNE est maximisé par rapport aux autres stratégies e l'état de l'art.

Traitement par lot de l'instanciation des réseaux virtuels

Dans ce chapitre, nous avons proposé un algorithme qui traite un lot (i.e., batch) de requêtes de réseaux virtuels et non pas une seule. On suppose que les requêtes ne sont pas traitées dès leur arrivée mais elles seront sauvegardées jusqu'à la fin d'une fenêtre de temps. Par la suite, ces requêtes seront traitées simultanément par notre proposition. Cet algorithme nommé Batch Reliable Virtual Network Embedding (BR−VNE) tient en considération la fiabilité pour proposer un mécanisme de tolérance aux pannes. La problématique liée au traitement par lot des requêtes est de trouver le sous-ensemble de requêtes à accepter afin de maximiser le revenu du fournisseur. De même, l'ordre de mapping des réseaux virtuels impacte aussi bien l'acceptation finale des clients que l'état du réseau physique. Ce dernier paramètre, à savoir l'état du réseau, influence directement le traitement

des prochaines requêtes. Par conséquent, BR−VNE choisit le meilleur sous-ensemble avec l'ordre de mapping ce qui revient à définir une séquence (i.e., l'ordre est pris en compte) de requêtes à accepter. Nous rappelons dans ce contexte que le mapping d'une seule requête est NP-dur, alors afin de minimiser la compléxité de la problèmatique de l'instanciation par lot, nous supposons qu'il y a un algorithme performant (e.g., CG-VNE) qui assure le mapping d'une seule requête. La question qui se pose maintenant est comment trouver la bonne séquence optimale, si possible, à partir du lot de requêtes arrivées. La recherche de la meilleure séquence revient à dénombrer toutes les combinaisons possibles tout en prenant en considération l'ordre. Cette recherche exhaustive explose d'une manière exponentielle en fonction du nombre de requêtes à traiter. Notre modélisation en arbre de recherche décisionel pour la sélection de la meilleure séquence nous conduit à l'exploitation des performances de l'algorithme Monte-Carlo Tree Search (MCTS). En fait, ce dernier a prouvé ses capacités à trouver la solution optimale pour de nombreux jeux résolus par l'ordianteur. Le point fort de cette approche est qu'il construit l'arbre décisionel en développant les branches les plus prometteuses, vu que la construction de la totalité de l'arbre rejoint la recherche exhaustive. La racine de l'arbre correspond à l'état du réseau avant le traitement du lot actuel des requêtes. Chaque fils généré traduit le mapping d'une requête virtuel. Une feuille dans cet arbre reflète le mapping de la totalité des requêtes client ou un état de blocage dû à un manque de ressources physiques. Dans le dernier cas définissant une feuille, il est clair qu'un sous-ensemble stricte a été instancié et non pas la totalité. D'une façon itérative, l'arbre se construit en développant les branches prometteuses. En effet, il y a deux étapes cruciales. La première concerne la sélection qui guide la navigation dans l'arbre car MCTS assure la recherche et la construction au même temps. Tandis que la deuxième est relative au développement des branches ou sous-branches. En arrivant à un noeud ayant la totalité de ses fils déjà développés, la question qui se pose est quel noeud choisir pour continuer la navigation. Le mécanisme de sélection défini par BR−VNE apporte une réponse à cette question en prenant en considération l'exploitation des branches prometteuses ou l'exploration de celles ayant moins d'intérêt et qui sont moins visitées. Par contre, en arrivant à un noeud avec des fils non développés, BR−VNE génère aléatoirement une nouvelle branche en choisissant à chaque niveau un seul noeud. Cette géneration se continue jusqu'au mapping de la totalité des réseaux virtuels ou l'arrivée à une situation de blocage. Cet état représentera la feuille de cette branche. Ce processus de navigation et de géneration se répète jusqu'à l'expiration d'une durée pré-définie ou atteindre le nombre maximal d'itérations. Finalement, la séquence choisie pour le mapping correspond à la branche offrant un maximum de profit. Ce profit est quantifié à la base de l'état du réseau et du revenu tiré par le fournisseur. BR−VNE est le premier algorithme traitant un lot de requêtes tout en prenant en compte la fiabilité afin de fournir une framework tolérante aux pannes. De plus, il est transparent vis-à-vis de la technique du mapping en ligne sous-jacente et il n'exige aucune contrainte sur l'algorithme utilisé. Par conséquent, il peut être utilisé avec tout algorithme en ligne traitant une seule requête à la fois.

Il est à souligner, qu'à notre connaissance, aucun travail scientifique traitant un lot de requête n'a focalisé sur la garantie de la fiabilité. Alors, pour l'évaluation des performances de BR−VNE nous l'avons comparé avec : i) les méthodes prenant en compte la fiabilité, ii) les méthodes traitant un lot de requête. BR−VNE est implémenté en utilisant les deux variantes de PR−VNE et CG−VNE. Les tests étaient concluants et notre nouvel approche réussit à maximiser le revenu du fournisseur en minimisant le taux de rejet et le taux de réseaux virtuels impactés par les pannes.

Conclusion et travaux futurs

Cette partie récapitule les travaux développés dans le cadre de cette thèse et fournit quelques directions futures de recherche. Nos contributions durant cette thèse s'articulent principalement sur quatre piliers, à savoir :

- Un état de l'art sur les algorithmes d'instanciation des réseaux virtuels tenant compte de la fiabilité. En effet, nous avons classé ces algorithmes en se basant sur différents critères comme : i) la stratégie distribuée ou centralisée, ii) le type de ressource protogée, iii) l'utilisation des backups. Il est à souligner que toutes les méthodes de l'état de l'art traitent les requêtes reçues une par une. Aucune proposition ne traite les requêtes client par lot.

- Notre deuxième contribution, nommée Preventive Reliable Virtual Network Embedding (PR−VNE), consiste à introduire un algorithme préventif qui favorise l'utilisation des ressources réseaux les plus fiables. PR−VNE se base principalement sur la métaheuristique issue de la colonie d'abeille (i.e., *Artificial Bee Colony* - ABC) pour définir le mécanisme d'instanciation. PR−VNE ne reconfigure pas le réseau lors des pannes pour re-instancier les ressources virtuelles impactées par cette interruption de service. Même si PR−VNE n'adopte pas d'approche curative, les résultats obtenus prouve qu'il assure un meilleur revenu pour le fournisseur du service Cloud.

- Afin de palier à la limitation de PR−VNE, la troisième contribution introduit un mécanisme curatif pour remédier aux pannes. Le nouvel algorithme nommé Coordination Game for Virtual Network Embedding (CG−VNE) se base principalement sur la théorie des jeux. Cette dernière nous a permis de concevoir un jeu collaboratif où les joueurs jouent au profit du fournisseur. Nous avons démontré l'existence d'un optimum social qui maximise le revenu du fournisseur du service Cloud. Cet optimum coïncide avec l'équilibre de Nash, qui représente un état d'équilibre où tous les joueurs ne peuvent pas améliorer leus revenus unilatéralement. De même, nous avons proposé un algorithme qui converge vers cet équilibre de Nash (i.e., optimum social) en temps polynomial. Lorsqu'une panne surgit dans le réseau physique, CG−VNE reconfigure les ressources (i.e., routeurs et liens) virtuelles dans des équipements

sains. Les simulations ont montré que les performances de CG−VNE comparées avec celles des méthodes concurrentes de l'état de l'art.

- Notre quatrième contribution traite l'instanciation d'un lot de requêtes et non pas une seule tout en considérant la fiabilité. L'objectif majeur de notre proposition, nommée Batch Reliable Virtual Network Embedding (BR−VNE), est de maximiser le profit du fournisseur de service Cloud tout en maximisant le taux d'accéptation et en minimisant le taux des requêtes impactées par les pannes. Pour le traitement par lot des requêtes, il faut chercher la meilleure séquence parmi les requêtes stockées. En d'autres termes, cela revient à chercher un sous-ensemble de requêtes tout en spécifiant l'ordre d'instanciation. Nous modélisons cette problèmatique par un arbre décisionel, qui constitue notre éspace de recherche. Nous concevons notre solution en se basant sur l'algorithme Monte-Carlo Tree Search qui nous permet non seulement de construire progressivement l'arbre mais aussi de développer et de chercher les branches les plus prometteuses. Afin d'évaluer les performances de BR−VNE, nous considérons deux classes de méthodes en ligne, qui tiennent en considération la fiabilité et celles qui traitent un lot de requêtes. Il est à noter que nous avons été les premiers à proposer un algorithme batch considérant de la fiabilité. Les tests étaient concluants et la méthode maximise bien le revenu du fournisseur par rapport aux méthodes proposées de l'état de l'art.

Comme perspectives à nos travaux de thèse, il nous semble adéquat de procéder à éxperimentation réelle afin de mieux valider nos résultats qui, pour l'instant, ne le sont que par les simulations.
De même, on peut pousser d'avantage les simulations pour définir des nouveaux scenarii de tests.
Du moment que la consommation énergétique représente l'un des problèmes dans les data center, nous envisageons de proposer un algorithme qui tient en considération la contrainte énergétique.

Publications

- **Revue Internationale**

  1. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, “A Novel Preventive Reliable Embedding Scheme for Virtual Networks within Cloud’s Backbone”, **soumis à** IEEE Transactions on Network and Service Management (**TNSM**).

- **Congrès Internationaux**

  1. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, “PR-VNE : Preventive Reliable Virtual Network Embedding Algorithm in Cloud’s Network”, in IEEE Global Communications Conference (**Globecom**), Atlanta - USA, December 9-13, 2013.
  2. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, “A Reliable Virtual Network Embedding Algorithm based on Game Theory within Cloud’s backbone”, in IEEE International Conference on Communications (**ICC**), Sydney - Australia, June 10-14, 2014.
  3. Oussama Soualah, Ilhem Fajjari, Nadjib Aitsaadi and Abdelhamid Mellouk, “A Batch Approach for a Survivable Virtual Network Embedding based on Monte-Carlo Tree Search”, in IEEE/IFIP Integrated Management (**IM**) Conference, Ottawa - Canada, Mai 11-15, 2015.