UNIVERSIDADE FEDERAL FLUMINENSE

ANSELMO LUIZ ÉDEN BATTISTI

SPEED - SFC Placement in Edge-cloud Environment: a Distributed approach.

NITERÓI

2023

ANSELMO LUIZ ÉDEN BATTISTI

SPEED - SFC Placement in Edge-cloud Environment: a Distributed approach.

Thesis project presented to the Programa de Pós-Graduação em Computação of the Universidade Federal Fluminense as a partial requirement to obtain the Doctor of Science degree in Computer Science. Field: Computer Systems

Orientador:
FLÁVIA COIMBRA DELICATO

Coorientador:
DÉBORA CHRISTINA MUCHALUAT SAADE

NITERÓI

2023

ANSELMO LUIZ ÉDEN BATTISTI

SPEED - SFC Placement in Edge-cloud Environment: a Distributed approach.

Thesis project presented to the Programa de Pós-Graduação em Computação of the Universidade Federal Fluminense as a partial requirement to obtain the Doctor of Science degree in Computer Science. Field: Computer Systems

Aprovada em 04 de 2023.

BANCA EXAMINADORA

Profª. FLÁVIA COIMBRA DELICATO - Orientadora, UFF

Profª. DÉBORA CHRISTINA MUCHALUAT SAADE - Coorientador, UFF

Profª. THAIS VASCONCELOS BATISTA, UFRN

Prof. PAULO DE FIGUEIREDO PIRES, UFF

Niterói

2023

Resumo

Recentemente o paradigma de virtualização dos recursos de computação e rede sobre a infraestrutura física ganhou força. Este paradigma foi adotado pela primeira vez em funções do núcleo da rede, criando o conceito de *Network Function Virtualization*. O processo de seleção do recurso para executar um conjunto de VNFs é chamado de placement. No entanto, virtualizar apenas VNFs individualmente pode ser insuficiente para atender aos requisitos do usuário. Assim, foi criada uma nova abordagem chamada *Service Function Chain*, uma cadeia ordenada de VNFs normalmente associada a um conjunto de SLAs. Com o avanço do paradigma de computação Edge-Cloud, as SFCs puderam ser executados em vários nós gerenciados por provedores de serviços distintos. Neste cenário, um mecanismo de placement distribuído deve ser adotado para alocar as SFCs sem conhecimento completo da infraestrutura. Neste trabalho, propomos uma nova abordagem, denominada SPEED, ela endereça o Problema de Placement de SFCs em um ambiente multidomínio de forma distribuída. A solução engloba os componentes arquiteturais, um novo modelo de mapeamento entre o Problema de Placement de SFCs e a abordagem de Teoria dos Jogos, além de uma nova estratégia de agregação de dados. Os resultados mostram que o framework SPEED é viável e, comparado com outras abordagens, tem um ganho de 20% no número de requisições alocadas.

Palavras-chave: SFC, SFC Placement, Distributed SFC Placement.

Abstract

In the last decades, there has been a trend to virtualize computing and networking resources over the physical infrastructure. This paradigm was first adopted in the network core functions, thus creating the Network Function Virtualization concept. The process of selecting the resource to execute a set of Virtual Network Functions (VNF) is named placement. However, virtualizing only VNFs individually is often not enough to meet the user's requirements. Hence, a new approach called Service Function Chain was created, an ordered chain of VNFs typically associated with a set of required SLAs. With the advance of the Edge-Cloud computing paradigm, SFCs can be executed in multiple nodes managed by independent service providers. In this scenario, a distributed placement mechanism should be created to allocate SFCs without complete knowledge of the infrastructure where the corresponding VNFs will be executed. In this work, we propose a new approach, named SPEED, to solve the SFC Placement Problem over a multi-domain environment in a distributed fashion. The solution encompasses the architectural components, a new mapping model between the SFC Placement Problem and the Game-Theory approach, besides a new data aggregation strategy. The results show that the SPEED is feasible and, compared with other approaches, has a gain of 20% in placed requests.

Keywords: SFC, SFC Placement, Distributed SFC Placement.

List of Figures

List of Table

List of Abbreviations and Acronyms

DAS Data Aggretated Size

DSFCPP Distributed SFC Placement Problem

ETSI European Telecommunications Standards Institute

GQM Goal Question Metric

MANO Management and Orchestration

NE Nash Equilibrium

NFV Network Function Virtualization

NFVO NFV Orchestrator

O-RAN Open RAN

OSM Open Source MANO

PT Placement Time

SF Service Functions

SFC Service Function Chain

SFCPP SFC Placement Problem

SFCPPlan SFC Placement Plan

SFF Service Functions Forwarders

SFP Service Functions Path

SGC Singleton Congestion Game

SLA Service Level Agreement

SPC Service Placement Cost

SPEED SFC Placement in Edge-Cloud Environment: a Distributed Approach

SPR Success Placement Rate

VIM Virtualized Infrastructure Manager

VNF Virtual Network Function

VPN Virtual Private Network

Summary

1 Introduction

In the last decades, there has been a trend to virtualize computing and networking resources over the physical infrastructure (34). This virtualization paradigm was first applied to the network core functions like NATs, Firewalls, and Deep Packet Inspectors, thus creating the Network Function Virtualization (NFV) paradigm (66). NFV decoupling network functions from the underlying dedicated hardware and execute them via software, which are referred as Virtual Network Function (VNF) (66).

Nowadays, even high-level functions directly requested by users, like CDN, video encoders, and antivirus, are also virtualized (51). The adoption of the NFV paradigm reduces the space needed for network hardware, decreases network power consumption, diminishes network maintenance costs, increases the life cycles of network hardware, and increase resource provisioning speed and efficiency, facilitating the management and orchestration of network functions (66, 34, 37, 3).

With the adoption of the NFV paradigm, a new problem named VNF Placement emerged. The VNF Placement defines which computational node will execute the requested VNF (27). This problem can be addresses using centralized and decentralized approaches, besides multiples algorithms like (50, 29, 48, 42).

Virtualizing only VNFs individually is often not enough to meet the user's services demands, for example, the high network traffic needs and the increase of computational utilization, besides the service chaining complexity (40). Thus, these demands have created a new paradigm called Service Function Chain (SFC) (21, 44). The SFC is a chain of multiple and ordered VNFs, associated with a Service Level Agreement (SLA) (7).

Therefore, with the demand to place multiple VNFs executed inside an SFC, the SFC Placement Problem has emerged (64). New challenges were addressed beyond defining which compute node should be used to execute each VNF, the networking connectivity also needs to be created to enable the flow through all the VNFs of the SFC (38, 2, 9).

With the advance of the Edge-Cloud computing paradigm, small to medium size

computing entity that aims to provide extra computing, storage, and networking resources to the applications deployed across IoT devices, edge, and cloud (39) became available. Thus, these compute nodes can cooperate with each other hence, each VNF of a SFC can be deployed and executed in multiple and heterogeneous nodes, potentially managed by different service providers (61).

In an edge-cloud scenario, finding the SFC placement plan adopting a centralized approach is inadequate, given the massive number of nodes in the environment, confidentiality problems and time to create the placement plan (55). Adopting distributed solution for the SFC Placement problem can reduce the operational cost, decrease the delay, optimize the resource consumption and increase the revenue (51). Thus, a distributed mechanism should be created to allocate the SFC without complete knowledge of the underneath infrastructure where the VNFs will be executed (11).

1.1 Research Questions

This thesis is guided by the following research questions:

- **Q1**: Is it possible to adopt a distributed SFC Placement mechanism to execute SFCs in a multi-domain environment?
- **Q2**: The adoption of a distributed SFC Placement mechanism to execute SFCs in a multi-domain environment can reduce the allocation cost compared with a centralized one?

1.2 Main Contributions

The main contributions of this thesis are:

- A novel approach, named SPEED, for solving the SPC placement problem in a multi-domain environment adopting a distributed fashion.
- A new mapping model between the SFC Placement Problem and the Game-Theory approach.
- A new data aggregation approach to share VNFs information.

1.3 Text Structure

In Chapter 2, we present the main concepts and technologies relevant to this thesis. We discuss the Service Function Chain (SFC) technology in Section 2.1, the SFC Placement Problem is analyzed in Section 2.2, the Distributed SFC Placement Problem in Section 2.3, and finally, Game Theory in Section 2.4.

In Chapter 3, we describe and compare studies directly related to our proposals. We analyze the works that proposed solutions that i) Propose solutions to solve the Distributed SFC Placement Problem and ii) Propose architecture or frameworks for environments that address the SFCs in a distributed fashion. We also compare and contrast these studies with our proposals.

In Chapter 4, we present the SPEED (SFC Placement in Edge-Cloud Environment: a Distributed Approach). We briefly present the proposal in Section 4.1. The definition of how the entities will be arranged in the distributed multi-domain environment is depicted in Section 4.2. After that, in Section 4.4, we describe the system model and the problem formulation. Then, we portray in Section 4.5 the heuristics for each step of the proposed solution. Finally, the architectural components and operations of the proposed are presented in Section 4.3.

In Chapter 5 we present the evaluation of our proposed approach. In Section 5.1 we depict the goals, questions and metrics that guides our experiments. In Section 5.2 we depict the environment where the experiments were conducted. In Section 5.3 we present a comparison between the SPEED, Random, and Greedy approaches.

This Chapter 6 we briefly exemplify some work produced the past two years. The section 6.2.1 show the inventions developed related to the VNF and VNF Placement problem. The section 6.2.2 presents articles developed during the period regarding VNF Placement and Distributed VNF Placement. Figure 21 present the achievements' timeline. Section 6.1 show the next activities of this work.

2 Background

This chapter presents the main concepts and technologies relevant to this thesis. We discuss the Service Function Chain (SFC) technology in Section 2.1, the SFC Placement Problem is analyzed in Section 2.2, the Distributed SFC Placement Problem in Section 2.3, and finally, Game Theory in Section 2.4.

2.1 Service Function Chain

This section details the Service Function Chain (SFC) technology. Figure 1 depicts the SFC structure. The SFC can be defined as a network service composed of multiple VNFs (51). The data flow crosses the VNF chain via virtual links (7) and have an SLA (60) that guide the orchestration and life cycle management component during the execution of the SFC.

The architecture of the SFC is defined by the IETF RFC 7665 (19), defining two basic aspects of the SFCs. The first one is the Service Functions (SF) that manages the ingress traffic, and the second one is the Service Functions Forwarders (SFF) that is responsible for steering the packet flow between the SFs according to a pre-defined Service Functions Path (SFP) (62). However, the RFC defines only the scenario where the SFC is deployed on a single domain, and there are no references to multi-domain environment scenarios. Executing SFC over a multi-domain environment is much more complex than executing SFCs in a single domain, especially when the domain where the user is associated and the domain where the elements of the SFC are executed belong to different administrative entities. In this multi-domain scenario, the domain owner would be reluctant to disclose details on their infrastructure with each other infrastructure providers (61, 68).

Each SFC and VNF is characterized by its resource demand, and requirements (36). Typically, these resource demands are defined using file descriptors using the TOSCA specification language (52). TOSCA allows the expressiveness of SFC and VNFs mappings data structures, providing methods for specifying workflows and enabling automatic

lifecycle management tasks from the Management and Orchestration (MANO) tools.

The entities that encompass the SFC will be virtualized and executed over physical resources (edge or cloud nodes). Each element that encompasses the SFC (The SFC itself, VNFs, and virtual links) has different requirements. In Figure 1, we depict an example of SFC that the max delay tolerated is 20 ms, and to execute the VNF 1, the physical node must be reserved at least 256 MB of memory.

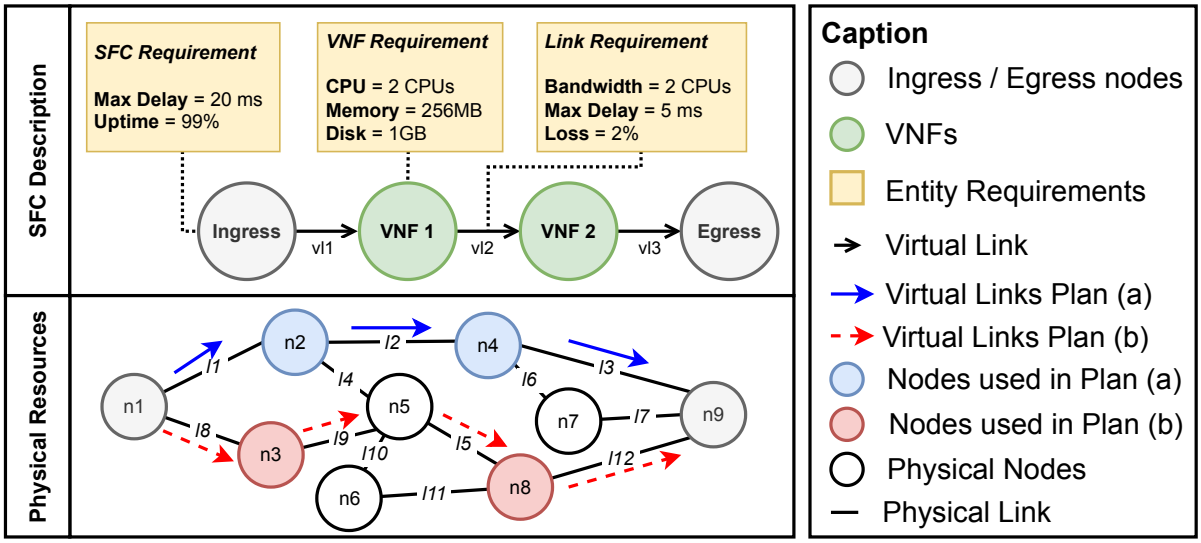

Figure 1: Requirements Associated With a Service Function Chain

To execute an SFC, a MANO platform must be available (33). Different platforms can represent the SFC structure and requirements differently. The ETSI-MANO, ONAP, Cloudify, and Tacker (33) are some of the most known SFC platforms. A key component in the SFC platform is the NFV Orchestrator (NFVO), also known as the NFV network service orchestrator. The NFVO is responsible for orchestrating NFVI resources across multiple Virtualized Infrastructure Manager (VIM), and also executes the lifecycle management of network services to deliver end-to-end connectivity (53, 33).

When the SFC is started, The VNFs that encompass the SFC can be executed in one or a multi-domain environment. The approach proposed in the RFC 7665 defines that elements of the SFC are executed in the same domain where the user is associated (19). The second one is when the elements of the SFC requested by the user are executed in a multi-domain environment (61, 32, 38, 1). In this strategy, parts or all the elements of the SFC are executed by multiple domains. This approach can increase the overall quality of the services provided by the user (61).

2.2 SFC Placement Problem

In this section, we will present the SFC Placement Problem (SFCPP). The SFCPP was already proved as being an NP-Hard problem (25). Hence, to overcome the NP-Hard characteristic, multiple heuristics have been proposed to solve the SFCPP problem, (23, 12, 48, 16, 4).

The objective of solving the SFCPP is to find a valid SFC Placement Plan (SFCPPlan). The SFCPPlan defines which physical resource will execute each element of the SFC (11). Figure 1 also presents two possible ways to allocate the requested SFC. In Plan (a), VNF 1 is executed in node n2 and VNF 2 in n4. In Plan (b), VNF 1 is executed in node n3 and VNF 2 in n8. The virtual link also must be associated with physical links, for example, In Plan (a) the virtual link *vl2* is mapped to the physical link *l2*, and in Plan (b) the virtual link *vl2* is mapped to the physical inks *l9* and *l15*.

The SFCPP can be solved in two ways, i) statically or ii) dynamically (22). In the static approaches, the placement plan is manually defined. In work by Li et al. (31), they point out the following problems with the static approach i) It is impossible to apply only desired network functions based on specific flow, ii) CAPEX and OPEX cost due to purchasing new hardware devices if the existing topology cannot fulfill the requirements and iii) The network devices must be physically connected and manually configured by network operators, which may lead to inconsistent configuration.

In contrast, the dynamic approaches create the placement plan using algorithms. Some advantages of the dynamic approach depicted in (22, 31) are i) Reduces CAPEX and OPEX as the controller steers the traffic over the environment, ii) Scalable, dynamic, flexible, and automatic service allocation and iii) If traffic requirement increases for a particular chain, the ability of only those network functions increases, that are present in that chain. A key disadvantage of the dynamic approach is that the time spend to produce the plan is too long in some algorithms (31).

Solving the SFCPP using the *dynamic approach* is highly investigated and can be addressed using multiple strategies. We depict some of these strategies in Table 1 and Table 2. There are two main categories, the centralized and the decentralized/distributed approaches. In centralized approaches, a single component in the environment knows all the information regarding the physical infrastructure. Based on this data, this component finds the placement plan based on the SFC requested and the algorithm definitions (66). In the decentralized approaches, the knowledge about the infrastructure is segmented

across multiple components. Thus, the request is divided into segments, each segment can be placed independently by a different placement component (11). The decentralized/distributed approaches have been gaining momentum over the last decade because of the advance of 5G networks (18).

2.3 Distributed SFC Placement Problem

One key component in any architecture that executes an SFC is the Management and Orchestration (MANO). MANO is responsible for coordinating the composition of multiple VNF that combined forms the SFC. By default, all the VNFs of an SFC are executed in the same domain (26). However, using only one domain to run the SFCs can struggle to deal with the restricted SFCs requirements. Thus, new approaches like the multi-domains one must be addressed.

Typically, the works that address the SFCPP in a distributed fashion considers that the SFC is executed in a multi-domain environment (4), (11), (51). A multi-domain environment can be defined as a set of independent entities with private computing and network resources that share limited information and can cooperate with each other to execute an SFC.

The work (11) presents a multi-domain orchestration survey from an algorithmic perspective by considering cooperative and competitive approaches. They propose a taxonomy that considers network service life cycle tasks. Regarding privacy, they found that most works assume that the shared information does not compromise the privacy and security of the domain participant in the orchestration process. However, these works provide no guarantees to support this claim.

The work (11) also present that the DSFCPP can be subdividing into three subproblems:

- **Shortest Path**: find the shortest path between the ingress and egress zone; if the delay in this path is higher than the max_delay thus the problem does not have a solution.
- **SFC Segmentation**: subdivide the VNFs of the SFC into VNF segments that are placed individually.
- **Segment Assignment**: select the zone in the shortest path that should execute each VNF segment.

All the subproblems described above must be addressed to execute the SFC in a multi-domain environment. Each problem has security and time-consuming issues. In this work, we present in section 4.5 for each one of these problems.

2.4 Game Theory

In this section, we will briefly introduce the main concepts of Game Theory and its relation with the SFC Placement Problem (SFCPP). Game Theory is a theoretical framework for designing interaction among players who desire to achieve some goals (47). The focus of Game Theory it to provide tools for analyzing scenarios in which players make interdependent decisions (43).

Multiple classes of problems can be modeled using Game Theory (43). One type of game was proposed by Rosenthal in 1973 (49) named Congestion Game. In a Congestion Game, each player's outcomes will be influenced by i) the resources it chooses and ii) the number of players that choose the same resource. Equation 2.1 formally describes a Congestion Game:

$$\Gamma = (N, R, (\Sigma_i)_{i\in[n]}, (d_r)_{r\in[R]}) \tag{2.1}$$

The element of the Equation 2.1 can be defined as:

- A finite set of players $N = (1, 2, ..., n)$, where $|N| = n$.
- A finite set of resources $R = (1, 2, ..., r)$, where $|R| = m$, these resources can be modeled as the edges of a directed weighted graph, for example.
- Each player $n \in N$ has a set of strategies that it can play named $\Sigma_i \subseteq 2^R$, where 2^R is the powerset of all possible strategies.
- Each element $r \in R$ has a cost function $d_r : \mathbb{N} \to \mathbb{R}$, where $\mathbb{N}$ is the number of players using the resource and, $\mathbb{R}$ is the total cost.
- The cost for player i, to adopt a finite set of strategies $S = (s_1, s_2, ..., s_n)$ is $c_i(S) = \sum_{r\in S_i} d_r(n_r(S))$, where $n_r(S) = |i \in N | r \in S_i|$, in other words, the sum of the cost of multiple payers adopting the same strategy.

All the congestion games have a Nash Equilibrium (NE) according to (49). NE is a state where no player can improve its outcomes by unilaterally changing one of its

strategies, and the other players maintain their strategies (35, 14). The computational cost to reach the NE in congestion games is n^m, where n is the number of players and m is the number of resources. Reaching the NE in an environment with many players and multiple resources is a computationally complex task (14).

The algorithm complexity to reach the NE is a problem when the number of resources and players is huge. However, there is a subset of the Congestion Game named Singleton Congestion Game (SGC) (14), with lower computational complexity (35). A congestion game is called singleton if, for every player $i \in N$ and every $R \in \Gamma_i$, it holds that $|R| = 1$. In other words, all players must allocate a single resource from a subset of allowed resources (54).

Modeling distributed problems using SFC is not widely adopted. Few works utilize this strategy; in particular, (8) investigate whether better performance is possible when restricted to the load-balancing problem in a game where players can only choose among single resources.

3 Related Work

This chapter describes and compares studies directly related to our proposals. We analyze the works that proposed solutions that i) Propose solutions to solve the Distributed SFC Placement Problem and ii) Propose architecture or frameworks for environments that address the SFCs in a distributed fashion. We also compare and contrast these studies in relation to our proposals.

3.1 Distributed SFC Placement

In this section, we will present some relevant works that deal with the SFC Placement Problem (SFCPP). As explained in Section 2.2, this problem can be addressed in a centralized or a distributed fashion. The centralized approaches are the most common and some works are depicted in Table 1. They have a unique component with full knowledge of the network topology, domains, nodes resources, and capabilities. Some drawbacks of centralized approaches are i) lack of scalability ii) network communication cost to gather data about all the resources iii) privacy and security in a multi-provider environment (51).

To overcome the issues related to centralized approaches, distributed approaches began to be proposed. In a systematic review, Santos et al. (51), present that 60% of the works in this field were recently published (2019 or 2020). In fact, the main platforms adopted by the SFC providers, like Open Source MANO (OSM) or Open RAN (O-RAN), do not offer any distributed SFC Placement solutions (63, 20). The distributed approaches to solve the SFCPP were less common than the centralized one.

Besides, in most of the works presented in (51), the distribution was focused on the scenarios and not on the SFC Placement per se. Furthermore, typically, the distributed approach combines a component that runs a centralized SFC Placement inside the domain. The SFC Placement is composed of two phases, in the first one the VNFs of the SFC are segmented into chunks of VNFs, in the second one, each segment is delivered to the candidate's domains to run the VNFs.

Table 1: Centralized SFC Placement approaches

Study	Objective	Technique	Environment
Ruiz et al. (50)	Mono	Genetic algorithm	Cloud
Kim et al. (23)	Mono	Conventional Lightchain algorithm	N/D
Emu et al. (12)	Mono	Machine Learning	Edge
Garrich et al. (17)	Mono	Greed	Edge
Li et al. (29)	Mono	Greed	Edge
Nguyen et al. (41)	Multi	Markov Chain	Edge & Cloud
Reyhanian et al. (48)	Multi	Alternating Direction Method of Multipliers	Edge
Kiran et al. (24)	Multi	Genetic algorithm	Edge
Pei et al. (45)	Multi	Machine Learning	N/D
Bunyakitanon et al. (9)	Multi	Q-learning scheme	Edge
Pham et al. (46)	Multi	Markov chain	Cloud
Li et al. (30)	Multi	Deep Reinforcement Learning	Cloud
Niu et al. (42)	Multi	Graph-based Particle Swarm Optimization	Edge
Alahmad et al. (2)	Multi	Greed	Cloud
Gao et al. (15)	Multi	Steiner Tree and Markov Decision	Cloud
Mohamad et al. (38)	Multi	Greed	Edge

In the work (10), Chen et al. present a distributed SFC Placement in a multi-domain environment where each domain is independent, shares minimum information about the internal infrastructure, and has its own orchestrator. The SFC Request arrives in any domain, named ingress domain and, the orchestrator of the ingress domain will coordinate the SFC Placement. The orchestrator of the ingress domain builds an aggregated graph including inter-domain links and aggregated nodes. Each domain will be converted into one aggregated node, this node stores the total CPU, Memory available, and the cost of the resources in the domain. The ingress orchestrator employs the k-shortest path algorithm in the aggregated graph and decides how to assign the VNFs to the different domains, they call this process SFC Partition. Subsequently, each partition is sent in parallel, this is the distributed step of the algorithm, to the selected domains and each orchestrator finds a solution (placement plan) within the resources of their own domain and sends deployment results back to the ingress orchestrator. If the process fails due to lack o resources in the domain path selected, the ingress orchestrator will select the next k-shortest path to execute the SFC. They also adopt the concept of VNF affinity and anti-affinity that allows the SFC Request defines that a VNF is forbidden or if it must be

allocated in a certain domain.

However, (10) has several drawbacks. The strategy adopted to create the domain aggregated node implies that the resources of all the nodes have the same cost which is not true in most realistic scenarios, allocate VNFs in nodes that demand more energy will increase the total cost of operation in the domain. Furthermore, if many nodes have few available resources the aggregated information will lead the ingress domain to possibly select domains that actually do not have a node with the resources required to run a VNF. Moreover, the algorithm creates the aggregation graph for each SFC Request which is not feasible for medium/large networks. They also consider that the link requirements between all the VNFs are the same, which is not true in most cases. And finally, the segmentation phase does not use the fallback information to create a better segmentation plan which increases the number of rounds to completely allocate the SFC.

Another relevant work in this field was proposed by Liu et al. (32). They present a distributed SFC Placement and load balance in a multi-domain environment. Each domain is independent and shares with the other domains only the peering nodes and which VNFs it can execute. Also, each domain can execute its own orchestrator configured to meet the service provider's interests. The SFC Request can arrive in any domain, named ingress domain. The ingress domain orchestrator will create segments from the VNF that compose the SFC, at least one domain should have resources to handle the defined segment. After the segmentation phase, each segment is sent to candidate domains, they will use a distributed auction strategy to decide which domain will execute each segment.

Nonetheless, (32) has many disadvantages. The adoption of a distributed auction strategy is typically employed in scenarios where the participants are competing against each other, in cooperative environments, like federations, the objective is to maximize the general wealth rather than the individual gain. Another flaw is that distributed auction (67) converges in 2Δ iterations without changes in the local winning list. As Δ is the domain-level network diameter, which is the number of inter-links included in the shortest path connecting the furthest domain pair, we can infer that in networks with a huge number of domains the number of iterations can be considerable, making unfishable the adoption of this strategy. Finally, they point out that in the load balancing phase, only domains that do not participate in the auction compete, i.e., only domains that were considered not feasible to run the segment can compete to run the segment in the load balance, that is a contrasense

In the work (58), Gang et al. present a novel to solve the distributed SFC Placement in

a multi-domain environment. In the adopted architecture, there is one main orchestrator that will coordinate the placement of all the SFC Requests. All the domains will run the same internal placement component. The resources of each domain are independent and the domain only shares the peering nodes and which VNFs it can execute. The algorithm executed by the main orchestrator is composed of two phases named partitioning and placement phase. The partitioning phase consists in creating sub-SFCs, this partition can be customized to i) minimize the number of domains or ii) improve the load balance of consumed resources in the domains. The placement phase that each domain executes to place the sub-SFC can be customized to i) minimize the delay or ii) improve the load balance in the nodes of the domain. Nonetheless of the innovations proposed in (58), the cornerstone resides in the full-mesh aggregation (28) approach to create the topology of the aggregated graph. It means that direct virtual links must be created between all the domains in the network, in scenarios with a huge number of domains, this can consume valuable resources unnecessarily.

The proposed approach by Gao et al. (16), solves the placement problem in a collaborative edge and cloud computing environment. The environment is composed of low earth orbit satellites (edge nodes), and the cloud, they are organized in a hierarchical structure. The proposed approach is based on the Viterbi algorithm executed in each satellite. When the request arrives, the satellite shares with its neighbors the requested VNFs and they decide if they can handle the request, if not they offload the task to cloud nodes. Despite the collaborative behavior presented in this work, they do not manage the linking between the VNFs in the correct order, thus making unfeasible the approach to deal with network services as depicted by the work.

The approach proposed by (4) is based on a decentralized auction strategy. They seek to find a deployment mapping for each VNF of an SFC compliant with the resource requirements and latency constraints, besides increasing the privacy of each domain. Each domain that participates in the auction sends a bid value for each VNF of the SFC. A centralized component receives the bids and runs an algorithm based on the satisfiability modulo theories (SMT) to match each VNF with each domain. However, the proposed approach rapidly increases the computational demand in environments with more than 20 edge nodes, determining that this approach only can be used for finding a near-optimal solution in tiny environments.

Table 2: Distributed SFC Placement approaches

Study	Distributed Approache	Environment
Chen et al. (10)	The allocation of each VNF segment in each domain is executed in parallel.	Edge-Cloud
Liu et al. (32)	The auction consensus phase that define s which segment will be executed in each domain.	Edge-Cloud
Gang et al. (58)	The allocation of each sub-SFC in each domain is executed in parallel.	Edge-Cloud
Avasalcai et al. (4)	Auction-based with a centralized orchestrator that defines the winner domain for each VNF in the SFC.	Edge-Cloud
Gao et al. (16)	The allocation of each service in each satellite is executed in parallel.	Satellite Edge-Cloud
Our Approach	Create VNF segments while executing Games to find a suitable domain for each VNF.	Edge-Cloud

3.2 Architectures for Distributed Placement ofSFCs

This section presents work that proposed architectures or frameworks that support distributed solutions to the SFC Placement Problem (SFCPP). A new topic, few researchers have addressed this topic.

In the work (20), Huff et al. extended the European Telecommunications Standards Institute model proposing the Multi-SFC Orchestrator, depicted in Figure 2. Figure 3 depicts the proposed architecture, it allows the deployment of VNFs that compose an SFC among multiple domains. Each domain can execute different orchestrators and adopts different network topologies and technologies. Inside the domain, there is an *NFVO Agents* and *VIM Agents*, these components receive commands from the main orchestrator, named Multi-SFC Orchestrator, and translate to the local domain NFV Orchestrator and Virtualized Infrastructure Manager, they are drivers that enable the orchestrators and Virtualized Infrastructure Manager heterogeneity. The flow forwarding between segments ($seg1$ and $seg2$) executed in different domains ($dom1$ and $dom2$) is performed using Virtual Private Network (VPN). After the last VNF of $seg1$ and before the first VNF of $seg2$, new VNFs that run VPN components are added, these VPNs components are connected and the SFC flow will traverse the domains. However, they do not formally define how the segmentation process will be executed in the Multi-SFC Orchestrator components, and this component is unique in the environment, thus creating a single fail point.

The work firstly proposed by Toumi et al. in (62), and extended in (61) presents

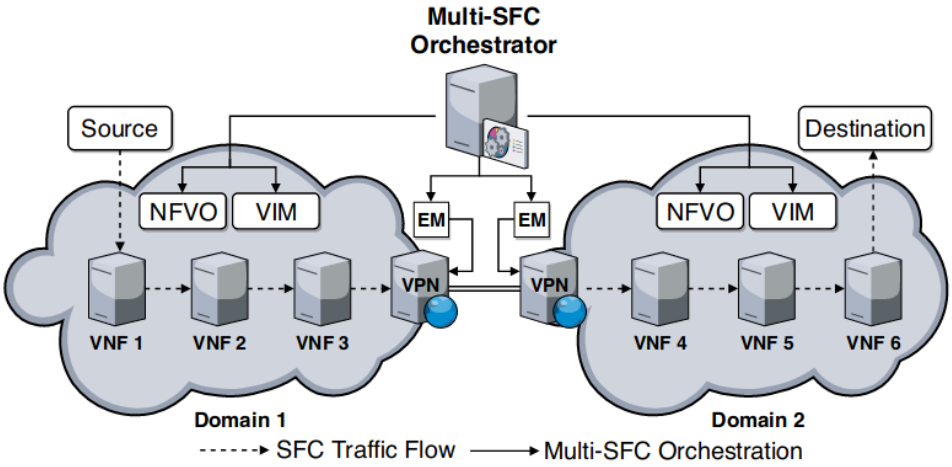

Figure 2: Multi-domain approach proposed by (20)

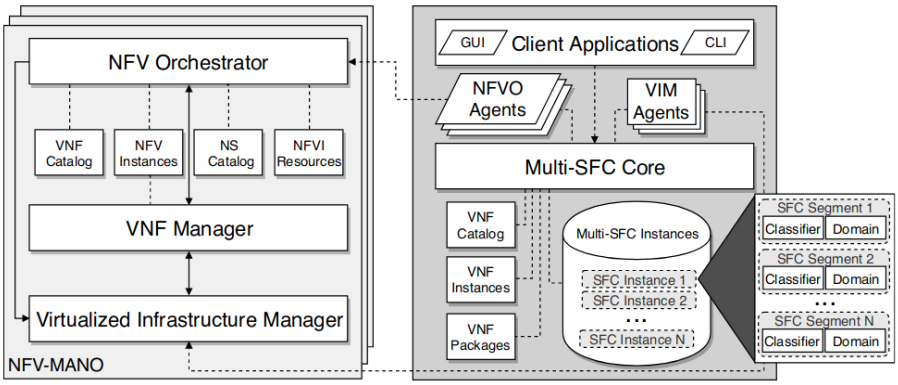

Figure 3: Multi-domain Architecture proposed by (20).

an orchestration architecture that leverages existing standardization efforts, adopting the TOSCA language to specify the network services and VNF Descriptor (59), in order to ensure an end-to-end orchestration of SFCs in a multi-domain environment. Figure 4 depicts the architecture prosed in (61). The proposal works regardless of the internal domain communication protocols. They also propose a multi-domain SFC deployment protocol that configures the network components to meet the end-to-end packet forwarding. Despite proposing an SFC Partitioning workflow, they do not specify how the VNF segments are created.

The execution of SFCs in a multi-domain environment is a new trend. This section presented some approaches to executing the SFCs in a multi-domain environment. The approaches rely on a single central orchestrator that has the knowledge about where each segment of the SFC will be executed. In the next section, we will present our proposed approach to execute the SFCs in a multi-domain environment in a distributed fashion.

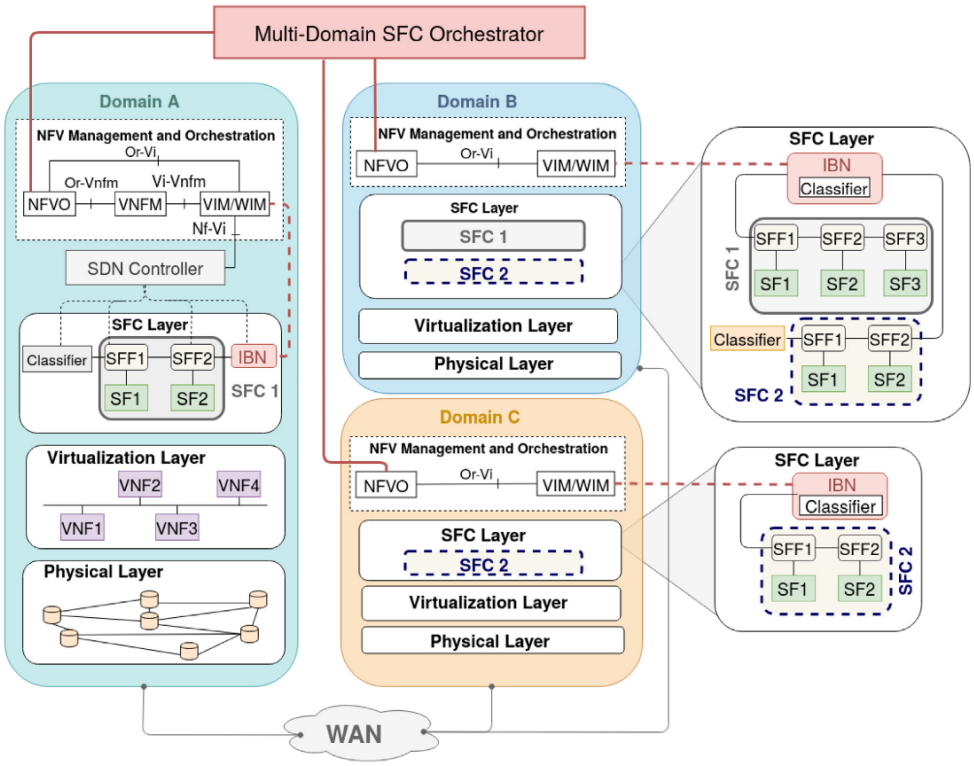

Figure 4: Multi-domain Architecture proposed by (61).

4 Proposal

This chapter presents our proposal, named SPEED (SFC Placement in Edge-Cloud Environment: a Distributed Approach). We briefly present the proposal in Section 4.1. The definition of how the entities will be arranged in the distributed multi-domain environment is depicted in Section 4.2. After that, in Section 4.4, we describe the system model and the problem formulation. Then, we portray in Section 4.5 the heuristics for each step of the proposed solution. Finally, the architectural components and operations of the proposed are presented in Section 4.3.

4.1 SPEED, a Brief Overview

This section presents an overview of the SPEED framework. This framework solves the SFC Placement Problem (SFCPP) in a distributed fashion. The environment where SPEED is executed is composed of unities that provide computation and network resources, they can be edge datacenters, network providers, autonomous systems, etc. We named these entities as zones in this work. The name zones and domain in this work can be used interchangeably, having the same meaning. Tree pillars support the proposed framework:

1. **Hierarchical Topology**: This virtual structure encompasses all the zones in a tree-like topology.
2. **Data Aggregation**: The process of aggregating the data over the tree-like topology. The data generated in the low lever zones is aggregated in the top zones.
3. **Distributed Placement**: The algorithms and components used to find a suitable placement plan in a distributed fashion.

Our proposed framework finds a suitable placement plan, if it exists, for a requested service based on the SFC requirements and the resources available in a multi-domain

environment. We consider that the environment is composed of multiple zones. The zones are hierarchically organized. The data used to execute the placement process is aggregated across the topology.

We split the proposed solutions into multiple steps. For each step, we provided a heuristic that covers all the required aspects. In particular, the core of the presented approach is the step that selects which zone will execute each VNF Segment. The VNF Segment are chunks of VNFs that encompass the SFC. This step was modeled using the SGC technique, and multiple games are executed in parallel to find a suitable placement plan.

We also provide detailed architectural components that indicate the required elements to execute the proposed framework in real scenarios. Besides, we depict the system operations required to execute our proposed framework.

4.2 Environment Entities

This section presents the environment where the SFCs will be executed in a distributed fashion. Figure 5 depicts an abstract view that embraces i) the entities, ii) zone types, and iii) hierarchical topology. We named zones the main elements of our topology. There are three types of zones, Access, Compute, and Aggregation. Each zone type has a specific meaning in the proposed approach, and each will be described below.

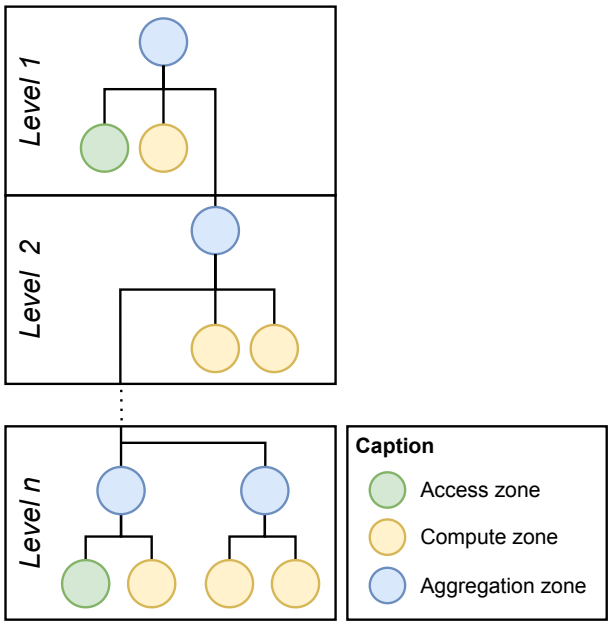

Figure 5: Environment Topology Overview.

The **Access zones** provide network connectivity for the users. In a 5G scenario, these zones are the RAN components. Multiple Access zones can belong to the same parent zone. VNFs are not executed in these zones.

The **Compute zones** are the elements that provide computational resources. They can represent either a MEC-Hosts or a Cloud-Hosts. The VNFs are only executed inside the Compute zones. Multiple types of VNFs can be executed in each Compute zone.

The **Aggregation zones** are abstracted elements adopted in our approach that aggregated the data about the descending zones. The component that manages the Aggregation zone topology and data can be executed in a border gateway. A network domain (autonomous system) is one element that can be mapped as an Aggregation zone.

All the zones are subjected to one Aggregation zone, except the higher-level zone (root). Thus, the elements of the distributed environment are arranged in a multi-level structure creating a tree-like topology. Each level determines how the aggregate data is compartmentalized. The Compute, Access, and Aggregation zones sent their internal data only to the Aggregation zone where they belong. The Aggregated data flow from the bottom to the top-level Aggregation zones. The top-level Aggregation zones have aggregated information about their own Compute zone and all the bottom-level Aggregation zones.

4.2.1 Data Aggregation over Topology

In this section, we present how the data flows in the topology according to the proposed solution. As presented in Section 2.3, the size of the edge-cloud environment (number of zones, nodes, and links) encompassing the infrastructure where the VNFs of the SFCs can be executed is broad. Thus, maintaining all the information about the topology in one single node, typically adopted in centralized orchestrator approaches like (58), faces technological and business model challenges, making this approach difficult to be adopted in real environments (51). Therefore, storing partial information (Aggregated data) in multiple zones is an already adopted strategy of hierarchical routing (28), and it is a more plausible approach to accomplish the SFC Placement in a distributed multi-domain environment.

In our proposed approach, the **Aggregation zones** process and store all the data sent by the underlying child zones. Adopting data aggregation strategies, the Aggregation zone will provide partial information to its parent zone. The aggregated data in each Ag-

gregation zone is created over i) the data sent by the underlying Aggregate and Compute zones, ii) the data already stored in the Aggregation zone, and iii) data gathered by the Aggregation zone, like network status and services status. Table 3 depict which raw data type the underlying zone will provide to the parent zone.

Table 3: Data aggregation fields.

Name	Description
Zone	The name of the zone. The zone can be an **Aggregation zone** or a **Compute zone**.
VNF	Define the VNF Type that can be executed in the zone.
Gateway	The Gateway where the packets will flow from to access the zone.
Cost	The cost to execute the VNF in the zone.
Delay	For **Compute zone**, the delay is based on the network the propagation delay from the Gateway until the zone. For **Aggregation zone**, the delay is the sum of propagation delay from the Gateway until the zone plus the time already consumed between the Aggregation zone and the child zone where the VNF will be executed.

The Aggregation zone updates the aggregated data in response to the changes reported by the child zones. Furthermore, the Aggregation zone can also update the aggregated data when a child zone fails to execute the request for executing a set of VNF. When the aggregated data is updated, the Aggregation zone will inform the parent zone about its new status.

The child zone will send data to the parent zone when the zone is configured in the environment or when the data previously sent becomes invalid. The data stored in the zone became invalid in the following situations, i) when a previously available VNF can no anymore be executed in the zone, ii) when a VNF previously unavailable became available in the zone, and iii) when a new VNF type became available. The information about the available resources in each physical node inside a Compute zone will not be sent to the parent zone. Only information about each VNF that can be executed in the zone will be sent. This approach (not sharing restricted information about the node resources and topology) increases data privacy, thus improving the security of the overall business model and infrastructure (65).

Figure 6 depicts an example of the environment and the data aggregation over time. The environment comprises 3 Aggregation zones, named $A1$, $A2$, and $A3$. The Aggregation zone $A2$ has two children, the Compute zone $C1$ and $C2$. The Compute zones inside $A1$ and $A2$, besides all the Access zones, will not be depicted to increase diagram

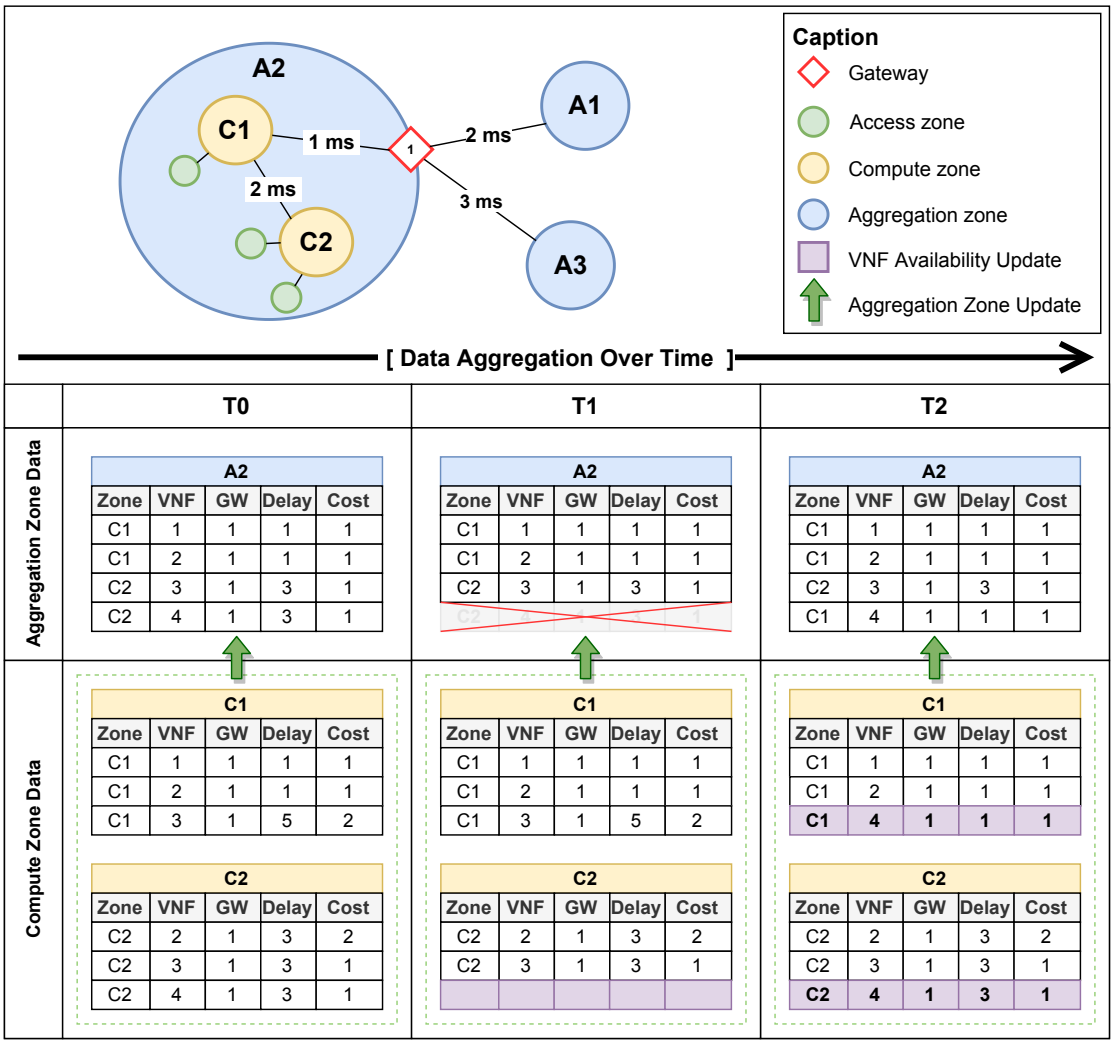

Figure 6: Data aggregation in a distributed environment over time.

plainness.

The aggregated data stored inside each zone, either Compute or Aggregation, changes over time. The Aggregation zone $A2$ stores aggregated data from $C1$ and $C2$. In time $T0$, $A2$ stores the aggregated information that the best Compute zone to execute a VNF of type "4"is the Compute zone $C2$. In time $T1$, $C2$ inform $A2$ that the VNF of type "4"can not be executed anymore; thus, $A2$ updates its aggregated data removing the possibility of VNFs of type "4"being executed because none of $A2$ child's can execute this VNF anymore. In time $T2$, the Compute zone $C1$ and $C2$ will inform $A2$ that they can execute VNFs of type "4", and $A2$ updates the aggregated data pointing that the Compute zone $C1$ now is the best zone to execute VNFs of type "4"because the delay from $C1$ to the Gateway 1 is lower than $C2$.

4.3 Architectural Components

This section presents the architecture that enables the proposed SPEED approach, which extends the components and interface specifications from ETSI (13). Figure 7 depicts the architectural components. Inside each Aggregation zone, previously presented in Figure 5, exists a VNF-MANO and the SPEED component. The VNF-MANO executes the VNFs management and orchestration tasks. The SPEED component will coordinate the zones to execute the VNFs required by the SFC.

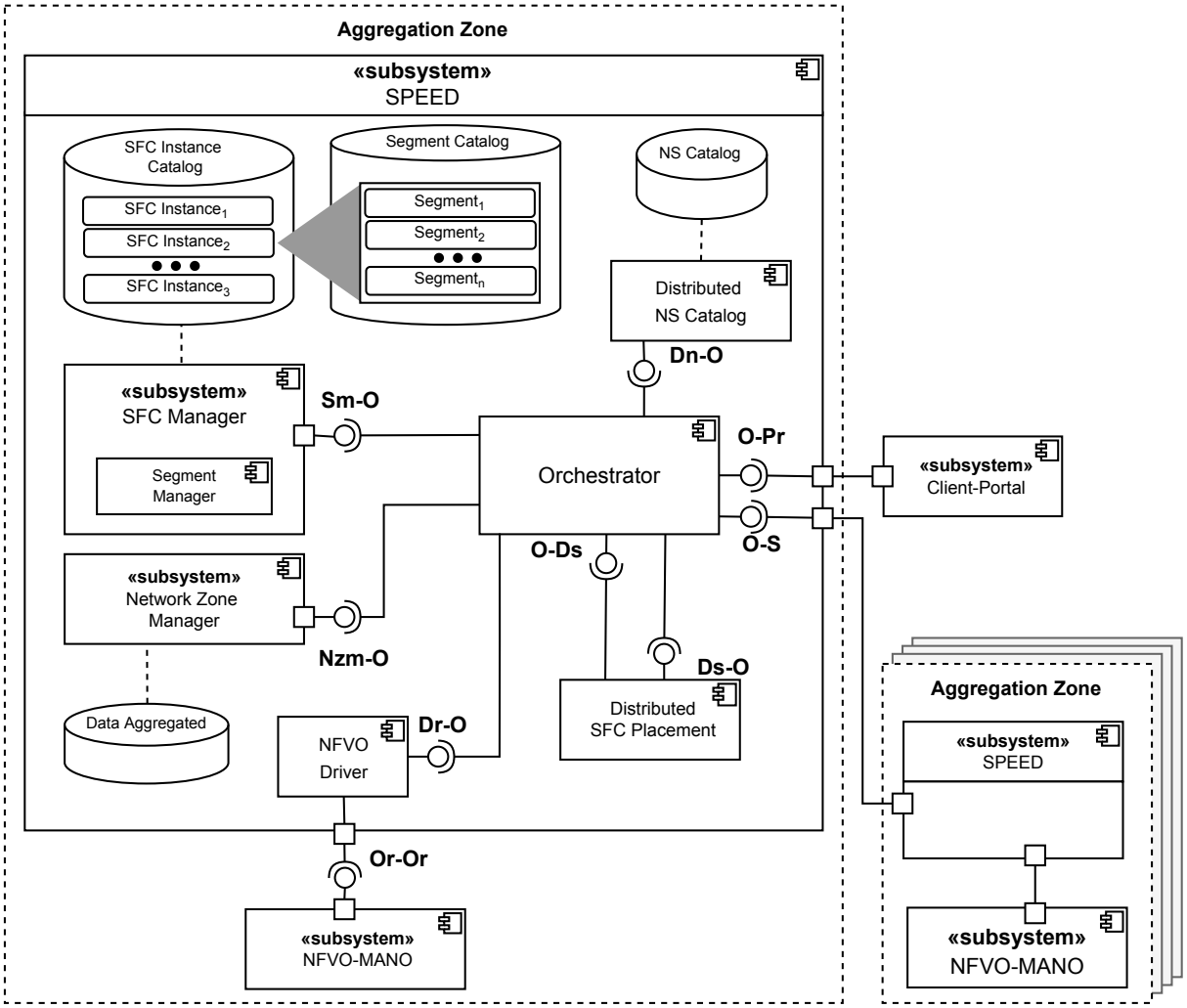

Figure 7: Architecture components of our approach.

The proposed architecture aims to allow the SFC Placement in a distributed fashion. All the zones in the environment are members of the same federation, with the same objectives, meaning that all zones are aware of the default behavior of the other zones. As a result, all zones cooperate to ensure that the federation's global objectives are met.

The **Network Zone Manager** subsystem is responsible for managing the relationship

between the parent and the chields zone. The zone's topology is defined manually by the zone owner or automatically using some heuristic. In this work, we do not cover this environmental coordination aspect. This subsystem provides the *Nzm-O* interface that allows the orchestrator to collect information about the zones.

The **Distributed NS Catalog** component is responsible for managing the descriptors of the SFCs that can be requested in the zone. It embraces the SFC whose VNFs can be executed in the zone and the SFCs that need coordination between zones to be executed. It provides the *Dn-O* interface that allows the orchestrator component to manage which SFC can be executed in the zone.

The *VNFO Driver* component is responsible for translating the commands from the Orchestrator to the VNF-MANO. There will be one *VNFO Driver* for each type of VNF-MANO available. As each zone has its own VNF-MANO, this strategy allows the interoperability of our proposal in heterogeneous environments. The *VNFO Drive* installed in a zone must be compatible with the VNF-MANO configured in the zone. It provides the *Dr-O* interface that allows the Orchestrator component to communicate with the NFVO-MANO in the zone.

The *SFC Manager* component is responsible for managing the SFC Instances lifecycle. When the zone is defined as the Zone Manager of an SFC Request, the SFC Instance associated is created in this zone component. This component provides the *Sm-O* interface that allows the orchestrator component to manage the SFC Instances.

The *SFC Instance Catalog* and *Segment Catalog* store data about the SFC and VNF segments executed in a child zone. Each segment is associated with one SFC Instance. The SFC Manager component consumes and updates data in this two storages.

The *Orchestrator* component is responsible for receiving the SFC Request from the Client-Portal via *O-Pr* interface. It provides the *O-S* interface that allows the orchestrator component to communicate with the SPEED component executed in another zone. This interface allows the necessary coordination between zones to execute the SFC Instance in a multi-domain environment using a distributed fashion approach.

4.4 System Model

In this work, we modeled our SFC Placement as an online problem, meaning that the SFC Requests are processed individually when it arrives in the system. We propose a new

system model for the Distributed SFC Placement Problem (DSFCPP) based on a multi-level and multi-domain hierarchy. In the model, the environment is composed of edge and cloud nodes. These nodes are part of a federation, the federation is divided into zones (domains) hierarchically organized. Each node is associated with only one Aggregation zone. Each user is associated with one Access zone. The user cannot move over time. The SFC Requests (demands) are executed both in edge or cloud nodes accordingly to the resources available and restrictions imposed by commercial rules, user restrictions, and request needs. Table 4 depicts all the symbols and decision variables of the proposed system model.

The network is modeled as an undirected graph $G = (\Omega, L)$. The $\Re = \{r_1, r_2,..., r_n\}$ set denotes all the SFC Requests that arrive in the system. Each request $r \in \Re$ is an n-tuple defined as $r = (src, dst, N, VL, max\_delay, affinity, \tau)$, where src and dst are the ingress and egress zones respectively. The src is an Access zone, and dst is a Compute zone. N is the sorted list of VNFs, $n_i \in VNF$, that compose the SFC. $VL = \{vl_1, vl_2,..., vl_n\}$ is the sorted list of virtual links $vl \in VL$ that will be mapped to physical links and connect the VNFs. The virtual link vl_1 associates the user with the first VNF, and the virtual link vl_n associates the final VNF with the egress node. Each vl_i has the attribute bw that defines the required bandwidth by the virtual link. The max_delay represents the max network delay tolerable between the src and dst. And finally, the $affinity$ is defined in Equation 4.2, this equation prevents or imposes the deployment of a specific VNF in a zone ω. The τ parameter defines when the SFC Request arrives in the system.

The $VNF = \{n_1, n_2,..., n_n\}$ set denotes all the VNFs in the system. Each $n \in VNF$ has the attributes cpu_n that define the required CPU in the Compute zone and mem_n that define the memory needed in the Compute zone.

The set $\Omega = \{\omega_1, \omega_2,..., \omega_n\}$ represents the network zones. Each zone $\omega_i \in \Omega$ has the following attributed: $chields_\omega = \{\omega_1, \omega_2,..., \omega_n\}$ that comprise the subordinated zoned, $siblings_\omega = \{\omega_1, \omega_2,..., \omega_n\}$ that embrace other zones with same parent zone, $N_\omega = \{n_1, n_2,..., n_n\}$ that represent the VNFs that can be executed by at least one subordinated zone, $n_i \in VNF$, $parent_\omega$ that defines its parent zone, if $parent == NULL$ it means that the zone is the higher element in the hierarchy, $type_\omega$, defined in Equation 4.1, identify if it is an aggregated or a Compute zone, β_ω^n represents the cost of executing a VNF n in zone ω. The Compute zones will only execute VNFs and the $chields_\omega$ list is empty.

$$type = \begin{cases} 0, & \text{if is an Aggregation zone} \\ 1, & \text{if is a Compute zone} \end{cases} \tag{4.1}$$

The set $L = \{l_1, l_2, ..., l_n\}$ represents the links between the zones. For every link $l_i \in L$, between two adjacent zones ω_1 and ω_2, we use bw_l and $delay_l$ to denote its bandwidth capacity and delay respectively. The parameter Ψ_l^v represents the cost of executing the virtual link v in the physical link l. Links between siblings Compute zones have infinity bw, $delay$, and $cost = 0$.

$$affinity_{\omega_i,\omega_j,n} = \begin{cases} 1, & \text{indicate that } \omega_i \text{ allows VNF } n \text{ to be executed in } \omega_j \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

4.4.1 Problem Formulation

In this section, the SFC Placement Problem (SFCPP) is formulated. In the begging, we introduce the decision variables of the model, shown in Table 4, and the interdependencies between them; after, we also discuss the constraints and objective function of the model. The objectives aim at minimizing: i) the cost of computational resources, ii) the cost of network resources, and iii) the SFC network delay.

The SFC Requests will consume computational and network resources to be executed. The service provider owns the computational and networking resources, and it what to lease the resources. The allocation of the resources to execute the service will have a cost for the final user. The cost is computed based on the total of computational resources in edge and cloud nodes, network bandwidth in the links, and traffic routing through the VNF chain.

Thus, our problem formulation requires two decision variables. The decision variable x_{ij} that defines if the VNF n_i will be executed in the zone ω_j and, the decision variable y_{ij} that defines if the virtual link v_i will be executed in the link l_j.

Thus, given a set of SFC Requests, our goal is to minimize the overall cost. The total placement cost, named $PlacementCost$, consists of two components, the computational resource cost, named $CompCost$, and the network resource cost, named $NetCost$.

$$PlacementCost = CompCost + NetCost \tag{4.3}$$

The Equation 4.4 defines how the computational resource cost is computed.

$$CompCost = \sum_{n_i \in N} \sum_{\omega_j \in \Omega} x_{ij} * \beta_{\omega_j}^{n_i} \tag{4.4}$$

The Equation 4.5 defines how the network resource cost is computed.

$$NetCost = \sum_{v_i \in V} \sum_{l_j \in L} y_{ij} * \psi_{l_j}^{v_i} \tag{4.5}$$

Given the zones, links, and resources available. The objective function is defined in Equation 4.6 is to find the placement plan for the VNFs and the subsequent traffic route that minimizes the deployment cost.

$$minimize\ PlacementCost \tag{4.6}$$

The problem is subject to a number of constraints depicted above. Equation 4.7 defines that the VNF only can be executed in Compute zones. Equation 4.8 defines that the VNF only can be executed in zone ω when the VNF is available. Equation 4.9 defines that the VNF only can be executed in zone ω when there are no affinity restrictions.

$$\sum_{n_i \in N} \sum_{\omega_j \in \Omega} x_{ij} * type_{\omega_j} \tag{4.7}$$

$$\sum_{n_i \in N} \sum_{\omega_j \in \Omega} x_{ij} * N_{\omega_j}^{n_i} \tag{4.8}$$

$$\sum_{n_i \in N} \sum_{\omega_i \in \Omega} \sum_{\omega_k \in \Omega} x_{ij} * affinity_{\omega_i, \omega_k, x_{ij}} \tag{4.9}$$

Table 4: Notation

Notation	Description
$G = (\Omega, L)$	Undirected graph of the physical network.
Ω	The set of network zones.
L	The physical links between the zones.
VNF	Set that denotes all the VNFs in the system.
$chields_{\omega}$	List with the child zones of the zone ω.
$siblings_{\omega}$	List with the siblings' zones of the zone ω.
$parent_{\omega}$	The parent zone of ω.
N_{ω}	Set of VNFs that can be executed by at least one subordinated of zone ω.
r	The SFC Request definition.
src	The node source of the dataflow node.
dst	The node destination of the dataflow.
N	The VFN that composes the requested SFC.
VL	The Virtual Links that bind the VNFs.
bw	The required network bandwidth.
max_delay	The max tolerable delay by the SFC.
$affinity_{VNF}$	Determine if a VNF must be deployed in a zone or if it is forbidden to be deployed in a zone.
$type$	The zone type. It can be Access, Compute, or Aggregation zone.
$CompCost$	The computing cost to execute the VNFs.
$NetCost$	The network cost to steering the traffic.

4.5 Proposed Approach

This section presents the details about the proposed approach that composes SPEED, our distributed novel to solve the SFC Placement Problem. Figure 8 depicts all the activities of the proposed solution. We partitioned the problem into four steps i) Manager Zone Selection, ii) VNF Segmentation, iii) Execution Zone Selection, iv) VNFs Allocation at Compute zone, and v) Network Path Build.

The **Manager Zone Selection** is the phase where the system selects the Aggregation zone that coordinates the distributed SFC Placement. This selection considers the VNFs, the source, and the destination of the SFC Requested. This process is straightforward to compute once the environment is modeled as a tree-like topology (Figure 5).

The **VNF Segmentation** consists of creating VNF Segments based on i) the VNFs requested by the SFC and ii) the VNFs of an already created VNF Segment. This process allows the VNFs to be placed in the most suitable Zone based on their requirements. Each segment will be placed into a different Compute zone.

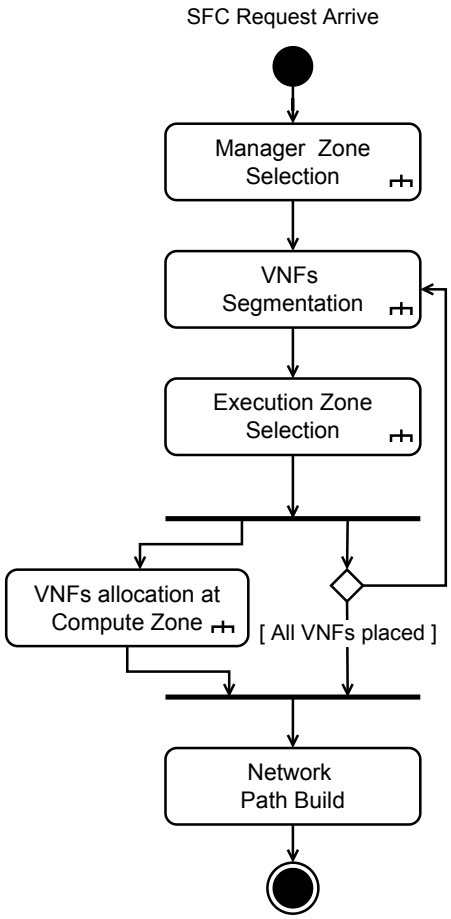

Figure 8: Approach Steps Overview.

In the **Execution Zone Selection** phase, we select which zone will handle each VNF Segment created. We modeled this step as a Singleton Congestion Game. The VNF Segmentation and the Execution Zone Selection will be executed multiple times until all the VNFs requested are placed.

The **VNFs allocation at Compute zone** is the VNF placement executed inside the selected Compute zone. The **Network Path Build** will create the network connectivity between all the VNFs of the service across multiple domains.

4.5.1 Manager Zone Selection

The **Manager Zone Selection** is the process of selecting the Aggregation zone that coordinates the distributed SFC Placement. The Manager zone is (from the bottom to top in the topology) the most inferior Aggregation zone between the source and destination of the requested service. All the VNFs of the service requested must be placed in a computed zone underneath the selected zone. Thus, the selected zone must have encompassed at

least one child zone capable of executing each VNF requested in the service.

Figure 9 depicts the zone topology, and VNF availability in a hypothetical scenario composed of three Aggregation zones and three Compute zones. Considering two services $r1$ and $r2$. The VNFs of $r1$ are VNF 1 and VNF 2, and the VNFs of $r2$ are VNF 1 and VNF 3. Both services have the source in $C2$ and destination in $C3$. The zone selected to manage $r1$ is $A4$ once $C2$, and $C3$ are children of $A4$, and the child of $A4$ can execute all the requested VNFs. However, the zone selected to manage the service $r2$ must be $A1$ once $C1$ is the unique zone in the topology that can execute VNF 3.

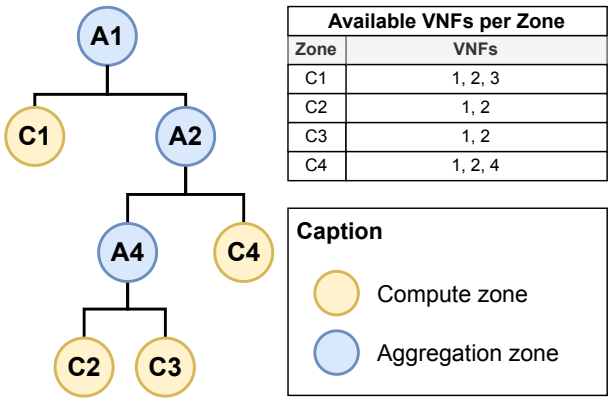

Figure 9: Example of Zone Manager Selection.

This process fails if the root zone cannot execute the VNFs of the requested service. We prioritize using the lowest Aggregation to reduce the use of most external inter-domain links. Typically, child zones of the same Aggregation zone will cooperatively share the infrastructure.

4.5.2 SFC Segmentation Strategy

This section presents our strategy to solve the SFC Segmentation Problem (56). The SFC segmentation problem addresses the separation of the VNFs that compose the SFC into chunks, named VNF Segments or segments. Different segmentation plans can be created from the same VNFs encompassing the SFC. The placement of segments into multiple domains is a time costly task. Thus, adopting a strategy to pick the best segmentation plan regarding the environment is necessary to reduce the time to place the requested service (32).

The problem of combining the VNFs into segments can be mapped in the context of combinatorial mathematics. Finding all the possibilities of the SFC segments plan is similar to the "stars and bars"problem presented in (57). The "stars and bars"can be used

to solve counting problems, such as how many ways there are to put n indistinguishable balls into k distinguishable bins. The original problem does not maintain the order of the elements; however, our VNFs are distinguishable; thus, we changed this characteristic to fit the "stars and bars"approach to our problem. A valid SFC segmentation plan must comply with the following conditions (32):

1. A segmentation plan is composed of one or more segments;
2. Each VNF should be allocated inside exactly one segment;
3. The VNFs ordination in the SFC Request must be maintained inside the segment;
4. The ordination of the segments must maintain the same VNF ordination of the SFC Request.
5. *The number of segments must be lower than the number of child zones.*

In our proposed solution, we introduced *item 5* as a new restriction. This restriction is not presented in (32). This applies only to our proposal because we use aggregation data. All plans with more segments than child zones imply the plan is invalid once each segment must be associated with a different zone.

The number of possible segment plans given a set of $VNFs$ can be identified as 2^{n-1}, where n is the number of VNFs. Thus, an SFC composed of three VNFs can be segmented into four possible ways, and Figure 10 depicts all four valid segmentation plans.

Multiple objectives can be pursued during the segmentation phase. In this work, we aim to reduce the bandwidth consumption at the intra-zones links. Thus, during the segmentation phase, we seek a segmentation plan that could be placed in fewer zones, and the VNFs that produce fewer output data is used as a point to split the SFCs.

Figure 10 depicts four possible plans for segmenting an SFC with 3 VNFs. In *plan A*, all the VNFs will be executed in the same zone, thus not using any link between zones. However, this plan can fail due to the lack of a resource in a single zone. The *plan B* has two segments, Segment 1 with the VNFs V0, V1, and Segment 2 with the VNFs V2. Each segment can be executed in different zones, increasing the chances of being a feasible plan. Regarding intra-domain bandwidth consumption, *plan C* is worse than *plan B* once the second one consumes 8 Mbps against 4 Mbps consumed in *plan B*. Finally, *plan D* will consume 12 Mbps of intra-zone bandwidth but probably will be the plan with a high chance of being placed once each VNF can be executed in different zones.

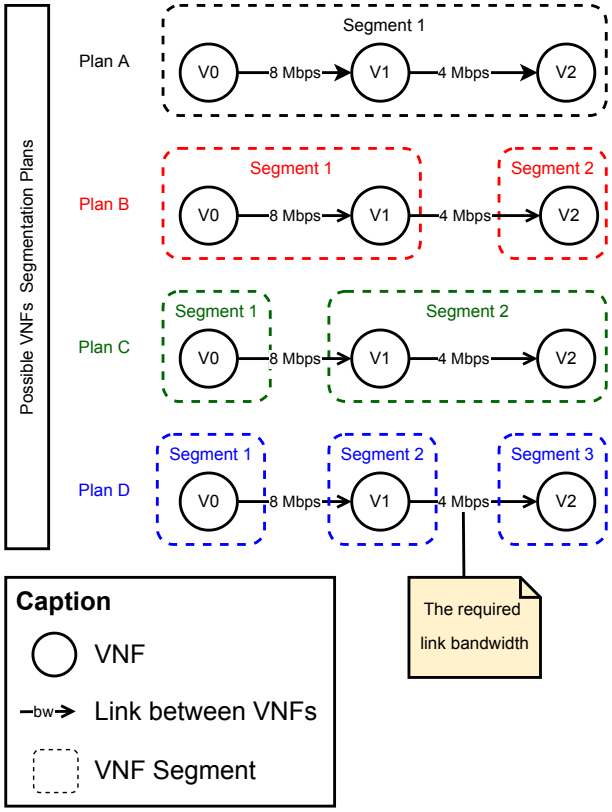

Figure 10: Possible VNFs Segmentation plans for an SFC with 3 VNFs.

Figure 11 depicts our heuristic to create and select the VNF Segmentation Plan. The proposed heuristic creates a segmentation plan that reduces the consumption of intra-domain bandwidth. If the plan were unfeasible, we split the previously created segments into new ones. If all the segment has only one VNF and the plan fails them, the segmentation phase will be considered failed. If no child zones can handle any VNF Segment, thus the allocation of the services will be rejected.

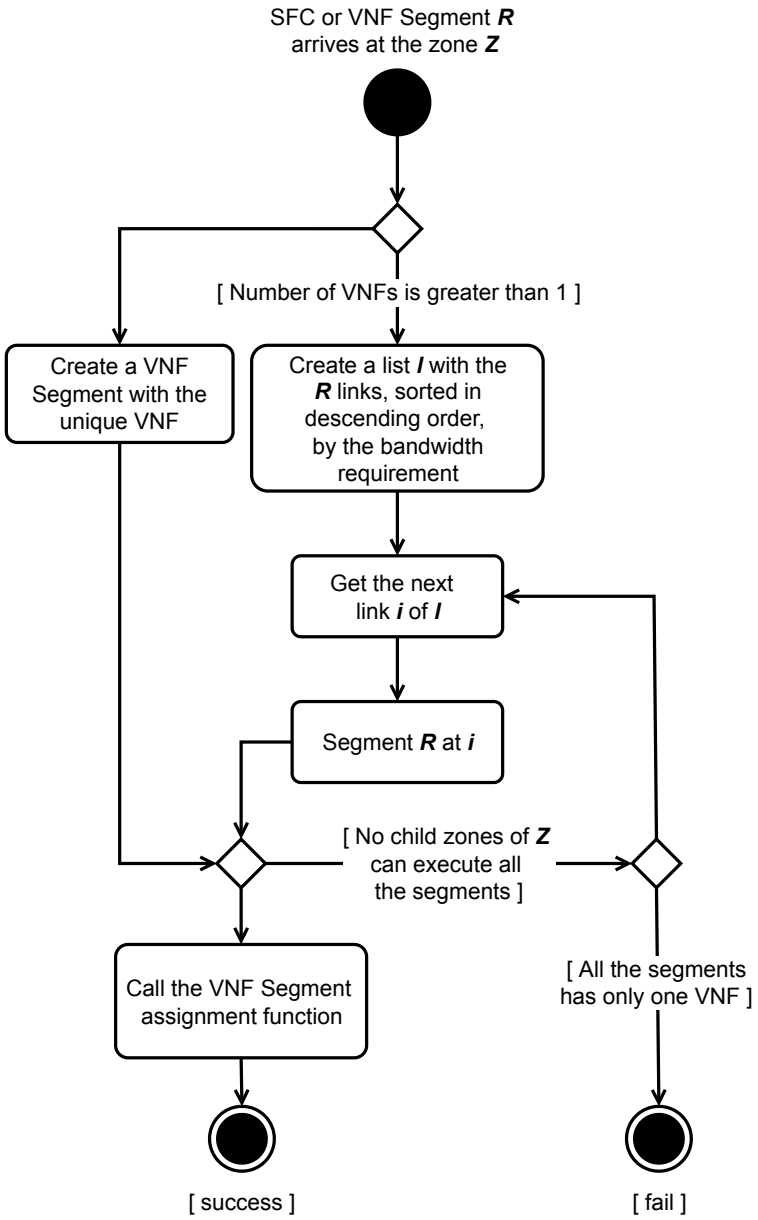

Figure 11: SFC Segmentation Heuristic.

4.5.3 Heuristic for Zone Selection as a Singleton Congestion Game

In Section 2.4, we defined the main concepts of the Game Theory, Congestion Games, and Singleton Congestion Game (SGC). In this section, we present our heuristic that maps the problem of associating each VNF Segment to a child zone and the SGC technique. We mapped the SFC Problem with the SGC as follows:

- The set of players N is composed of VNF Segments.
- The set of resources R is composed of the bind between the *Aggregation zone* and its child zones.

- The set of strategies that each player $n \in N$ is the graph's edges that connect the *Aggregation zone* with the child zones.
- The cost function will is the cost of executing the VNF Segment into a particular child zone.

Algorithm 1 depicts an implementation of how the Singleton Congestion Game can decide which child zone should be selected to execute each VNF Segment. Each VNF Segment s can be associated with a different child zone. During the same game, the zone selected for the previous VNF Segment is used as input to select the next zone, thus increasing the chance of creating a feasible placement plan. The algorithm stops when all the VNF segments do not change the select child zone based on the zones selected by the other VNF Segments.

Figure 12 portrays an example of how our proposed approach can solve the SFCPP in a distributed fashion. The topology comprises 1 Access zone, named R1, 8 Compute zones, named C1 to C8, and 5 Aggregation zone, named A1 to A5. The zones A2, A3, and A4, are children of A1. Zone A3 has the Compute zone C3 and Aggregation zone A4 as child zones. For simplicity's sake, we omit each Aggregation zone's aggregated data. The description of how the aggregated data is computed can be found in Section 4.2.1.

The SFC Request arrives in zone R1; thus, R1 is the source of the packet flow, and the destination is zone C4. The Aggregation zone that can reach both R1 and C4 is A1. Therefore, zone A1 will be selected as the responsible for coordinating (Coordination Zone) the distributed process to allocate the requested service in the environment. The complete process to select the coordinating zone can be found in Section 4.5.1

The first step of the process is to separate the VNFs of the SFC into VNF Segments. The service requested contains 3 VNFs, named V0, V1, and V2. The link between V1 and V2 requires a lower bitrate (1 Mbps) compared to the link between V0 and V1 (8 Mbps). Therefore, the SFC Segmentation process splits the SFC Requested into 2 VNF Segments, named Seg.1, which encompasses the VNFs V0 and V1, and Seg.2, which contains only the VNF V2. The VNF Segments are created based on the compute and network resources available in the subjacent zone, using the aggregated data as the data source. For this example, all the Compute zones can execute all the required VNFs.

The process to find the suitable Compute zone to execute each VNF Segment is founded through playing multiple Singleton Congestion Game (SGC). The game objective is to identify which child zone should execute the VNFs inside the VNF Segment (player).

Algorithm 1: Zone Selection heuristic based on Singleton Congestion Game Approach.

Input: $VNFSegmnts$, $ParentZone$
Result: Mapping between each VNF Segment with one of the $ParentZone$ child zone
Initialization:
$equilibrium \leftarrow False$
$previousZone_s \leftarrow False$
$childZones \leftarrow getChildZones(ParentZone)$
while $equilibrium = False$ **do**
  $changed \leftarrow False$
  for each segment s **in** $VNFSegmnts$ **do**
    $selZone_s \leftarrow suitableZone(s, childZones, prevZone_{s-1}, prevZone_{s+1})$
    if $selZone_s \neq prevZone_s$ **then**
      $changed \leftarrow True$;
    end
    $prevZone_s \leftarrow selZone_s$
  end
  if $changed \neq False$ **then**
    $equilibrium \leftarrow True$
  end
end

The first game is executed by A1 (Coordination Zone), and the players are the VNF Segment Seg.1 and Seg.2. In this example, Seg.1 will be forwarded to zone A3 and Seg.2 to zone A4. Now, Zone A3 and A4 will execute, in parallel, games to determine whether each child zone should execute the VNFs of the segment. This cycle will continue until each VNF Segment reaches a Compute zone. When the Compute zone is selected, the VNFs of the segment will be placed inside the zone. All the VNFs inside the VNF Segment are always allocated in the same Compute zone. The same VNF Segment can be part of many games until a Compute zone being reach.

As selecting each zone for each VNF is executed in parallel, the manager zone must be informed of the process by the child zones. When a VNF Segment reaches a Compute zone, this zone sends a message to the coordination zone informing that the VNFs of the segment will be executed in that zone. The distributed allocation only will be considered finished when all the VNFs requested are assigned to one Compute zone. After the execution of the games, zone A1 (Coordination Zone) creates network connectivity between all the zones used to execute the VNFs.

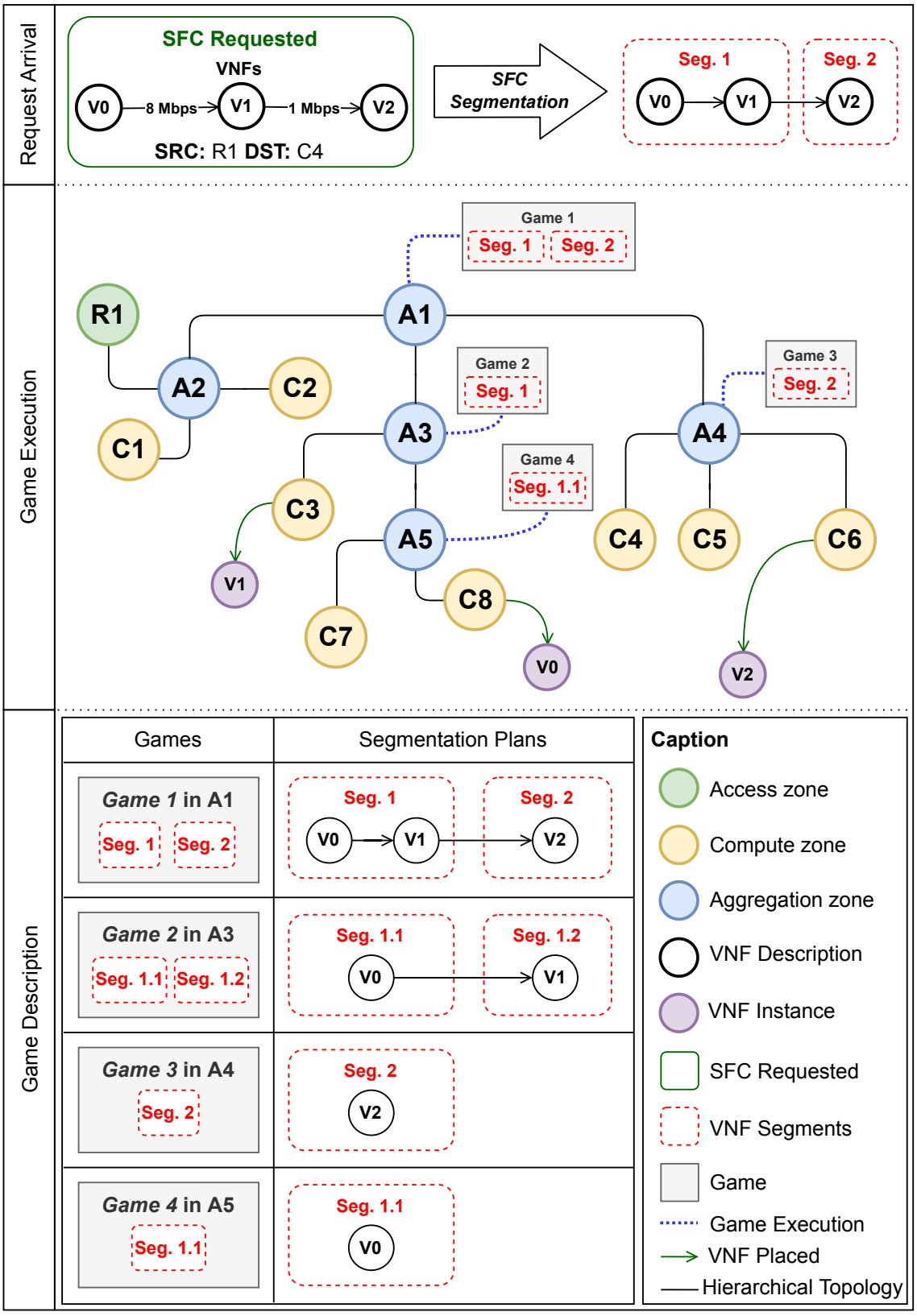

Figure 12: Execution example of our approach solving the SFCPP.

4.6 Operation

This section presents essential aspects of the architectural operation. The data aggregation process is presented in Section 4.6.1. We discuss how the SFC Request arrives in the system in Section 4.6.2.

4.6.1 Data Aggregation

In our proposed approach, data aggregation is the process of combining and processing the data of the subordinated zones. This allows for each zone to only expose to its parent zone public information. Even with this strategy the system was capable of executing the SFC placement in a distributed fashion.

The *zone manager* component runs in each Aggregation zones and periodically collects data about the subordinated zones. This data represents the VNFs that can be executed in the child zones. The data from all child zones are aggregated and informed to the parent zone. This propagation will allow SPEED to create the placement plan. Figure 13 depicts the sequence diagram to collect information about the child zones and send the aggregated data to the parent zone.

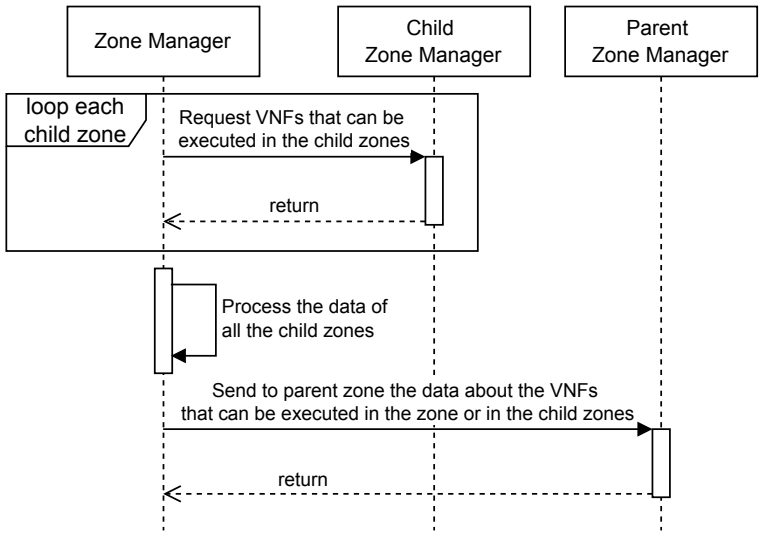

Figure 13: Aggregate data from child zones diagram.

All the Aggregation zone must execute the procedure to collect data about their child zones. This task can be executed in a fixed interval, configured by the zone administrator, or every time a child zone reports some change in the VNFs availability. The data aggregated is stored in the Data Aggregated Storage in the SPEED component.

4.6.2 SFC Request Arrival

The main objective of the proposed approach is to allocate an SFC in a multi-domain environment in a distributed fashion. Figure 14 depicts the actions and the main components involved during the placement of an SFC. Firstly, the User requests the SFC via the Client Portal. The Request arrives in the *Orchestrator* component of the zone where the User is associated. Then, the *Zone Manager* component is responsible for finding the zone manager κ where the source and destination of the requested SFC are subordinated. Thus, the zone κ will be responsible for coordinating the distributed placement of the Request SFC.

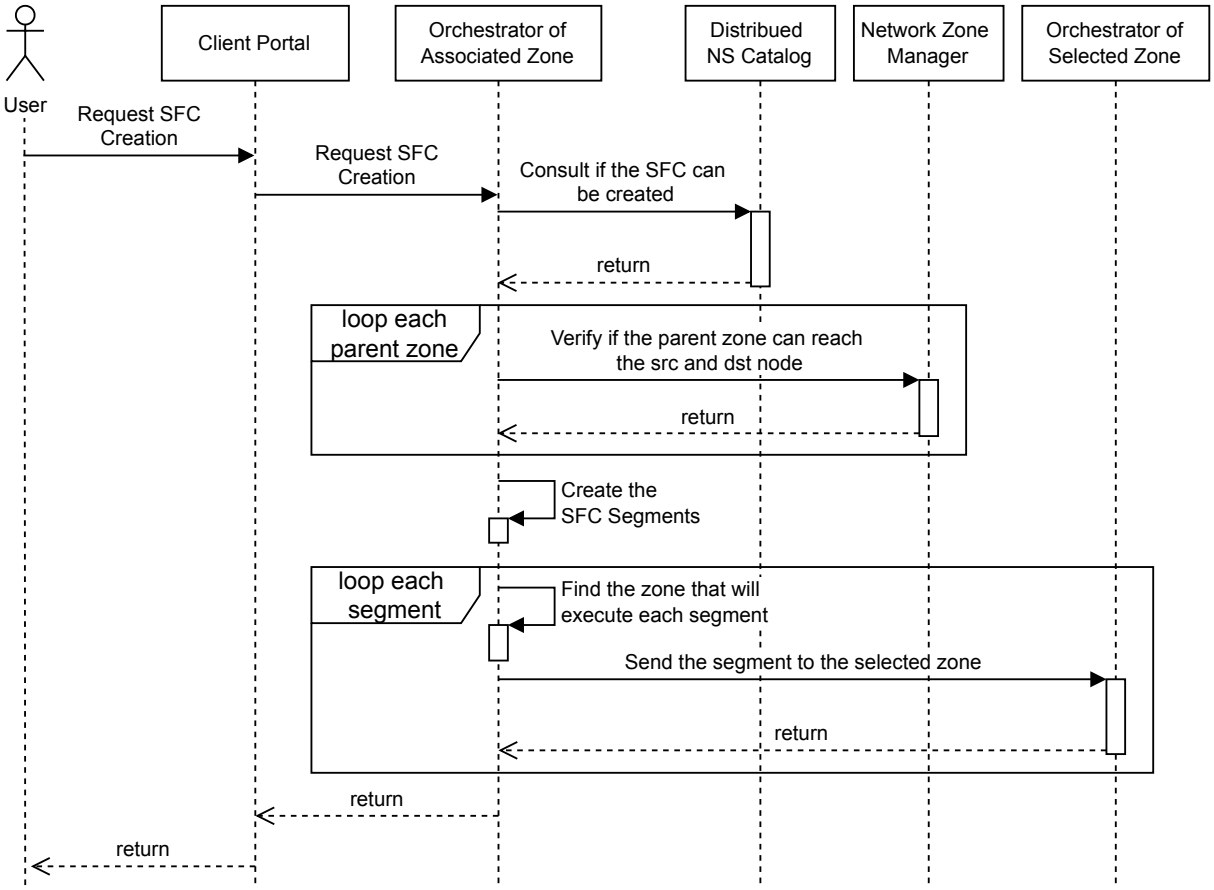

Figure 14: SFC Request diagram.

The zone κ will create the SFC segments based on the Algorithm 1. After that, each segment will be delivered to the selected zone, and the zone will execute the placement of the VNF in its Compute zone. After the placement of each VNF, the zone κ is informed by the computed zone.

5 Evaluation

This chapter presents the evaluation of our proposed approache. In Section 5.1 we depict the goals, questions and metrics that guides our experiments. In Section 5.2 we depict the environment where the experiments were conducted. In Section 5.3 we present a comparison between the SPEED, Random, and Greedy approaches.

5.1 GQM

For planning the preliminary experiments of this thesis, we adopted the Goal Question Metric (GQM) (5) approach. The GQM model is a hierarchical structure, composed of three levels of entities, named *Goals, Questions*, and *Metrics*. Each level refines the granularity of what is relevant to provide reliable insights into a phenomenon.

A **goal** represents which phenomenon should be analyzed. Each goal can be decomposed into one or more **questions**. A **metric** dependent variables, or correlations between two or more variables. The question should be answered based on the metrics. Table 5, 6, and 7 presents the goals, the questions, and the metrics respectively. This will guide the evaluation of our work.

Table 5: Goals definition

Goal	Goal Description
G1	Analyze our proposed approach (SPEED) in a multi-domain environment, for the purpose of evaluating its performance in terms of allocation cost, and service allocation rate in comparison with approaches proposed in the literature.

Table 6: Questions

Question	Question description
Q1	Does the adoption of SPEED reduce the allocation cost compared with a traditional approach?
Q2	Does the adoption of SPEED increase the allocation rate compared with a traditional approach?

Table 7: Summary of the metrics used to answer the questions.

Metric	Metric description	Question
SPC	Service Placement Cost (SPC) is the cost to allocate the requested service.	Q1
SPR	Success Placement Rate (SPR) is the number of requested services divided by the number of services placed.	Q2
PT	Placement Time (PT) is the total time for finding the placement solution.	Q2
DAS	Data Aggretated Size (DAS) is the total aggregated data consumed for finding the placement solution.	Q2

5.2 Topology Overview

This section presents the environment where the initial experiments were conducted. Figure 15 depicts the relationship between the Compute Zones and the Aggregation Zones. These entities are organized into a tree-like topology.

Figure 16 presents the physical topology. The topology represents the compute nodes inside each domain and the physical links. The requested services will be executed in this environment. The experimental scenario was composed of 10 domains. Each domain has five compute nodes. There are 10 different types of VNFs and SFCs.

5.3 Performance Evaluation

This section presents a comparison between the SPEED, Random, and Greedy approaches. The Random approach randomly selects a domain to execute the VNFs of the requested service. The Greedy approach selects the zone with the minimum delay to place the next VNF of the requested service. We compare the metrics Service Placement Cost (SPC), Success Placement Rate (SPR).

We create scenarios with 10, 20, 30, 50, and 100 services requested. The services

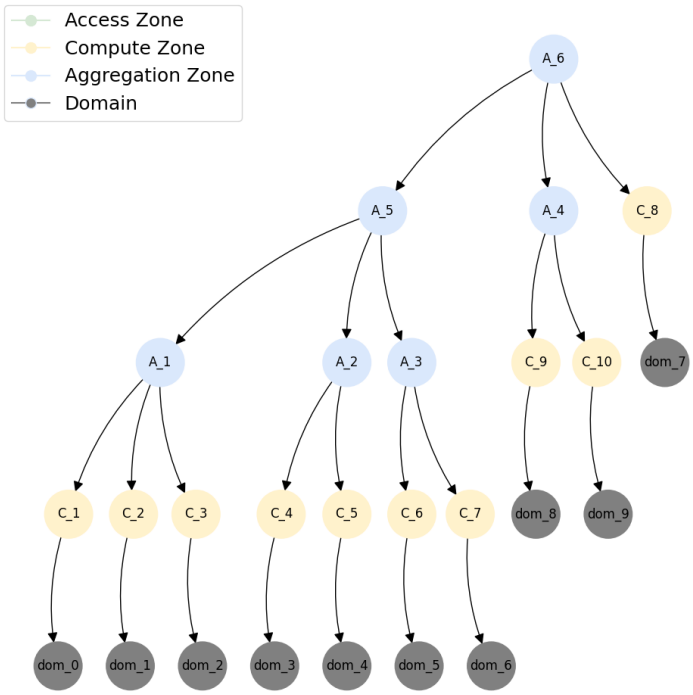

Figure 15: EnvironHierarquical Topology.

arrive in an interval following a Poisson distribution. For each scenario, we executed the simulation 10 times. We implement the simulation using the SimPy framework[1]. The source code of the simulation can be found in the project repository [2].

Figure 17 compares the different SFC Request Placement Methods. Higher bars represent better results. The SPEED approach can split the requested VNFs of a service into multiple domains. Thus, the number of suitable execution plans increases compared to approaches that allocate all the VNFs to the same domain. We found a suitable plan for almost all the requested services, and in comparison with a greedy approach, we increased the number of successfully placed by 20%.

Figure 18 compares the average cost to execute the requested service. Lower bars represent better results. Even with a higher success placement rate, the average cost to execute each service was lower using the SPEED approach. Thus, our proposed approach increases the number of requests placed and decreases the cost to execute each service.

Figure 19 presents data regarding the time used for the SPEED approach to find a suitable placement plan. In the described scenario, took an average of 21ms with an std of 2.1ms. Most part of this time is network delay. Therefore, the time can variate based

[1]https://simpy.readthedocs.io/en/latest/
[2]https://github.com/anselmobattisti/speed

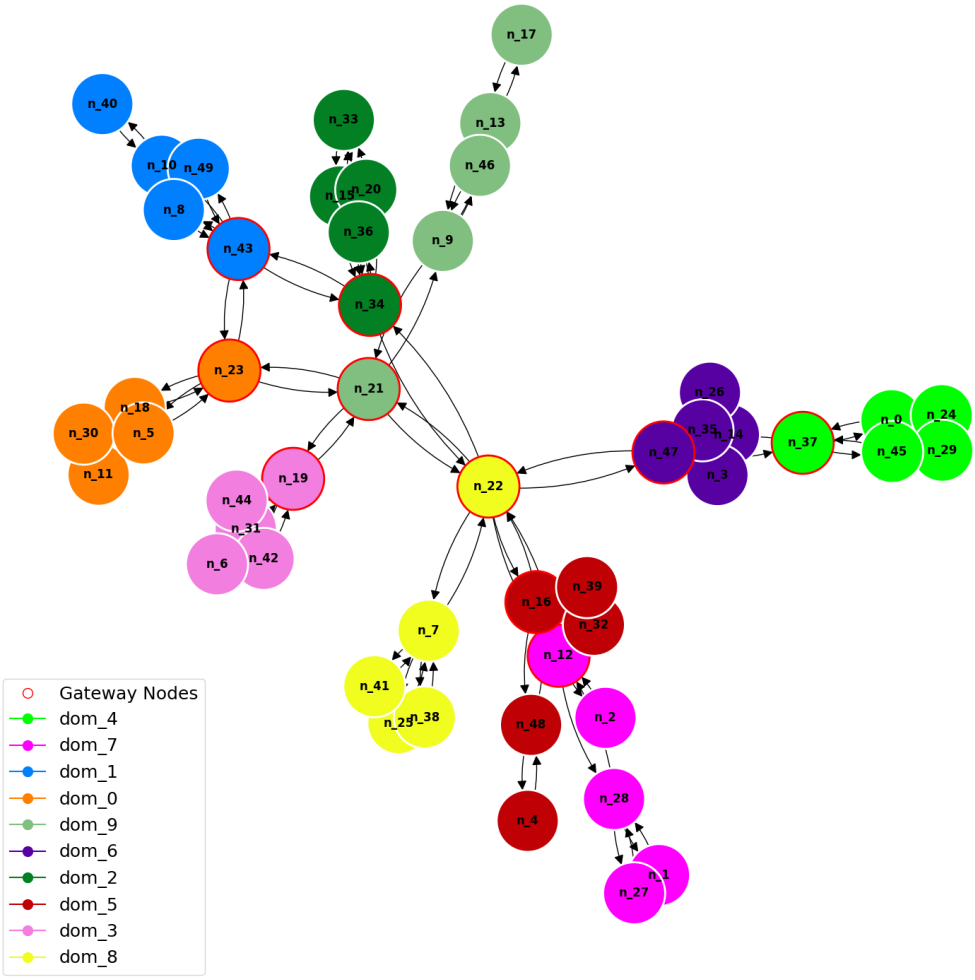

Figure 16: Domains Topology.

on the number of levels in the topology and the network delay between the zones.

Figure 20 presents data regarding the size of data shared between the aggregation zones. The data shared grows proportionally to the number of requested services. We adopt a reactive approach in terms of data sharing, we request the info about the child zones only when the zone is selected to participate in a game to execute a VNF Segment.

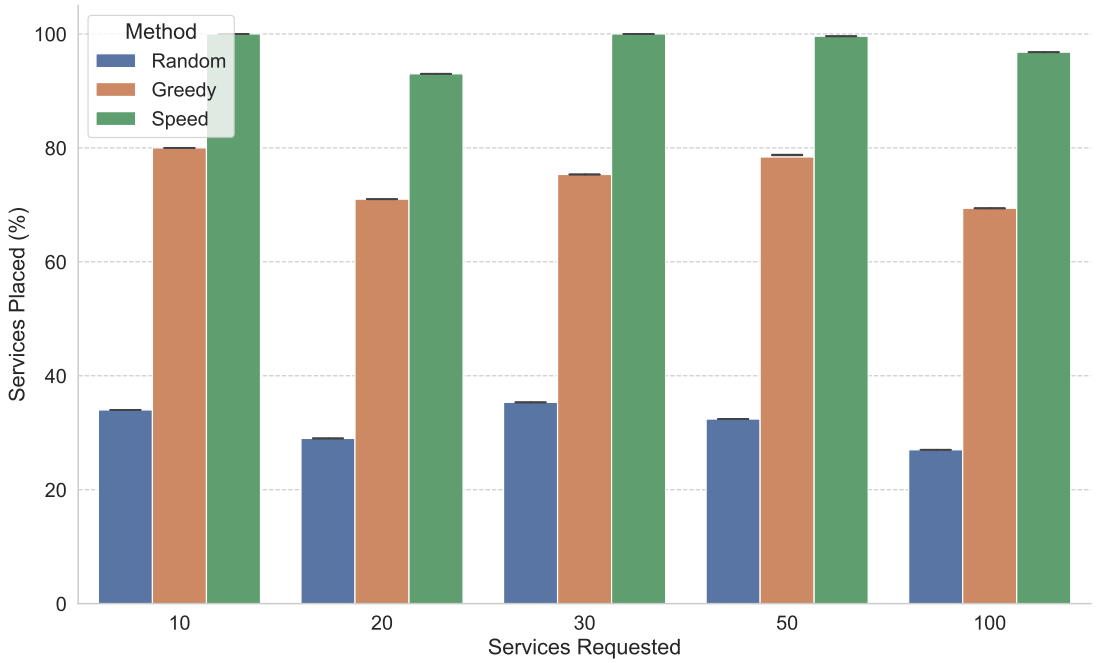

Figure 17: Services Placement Success Rate.

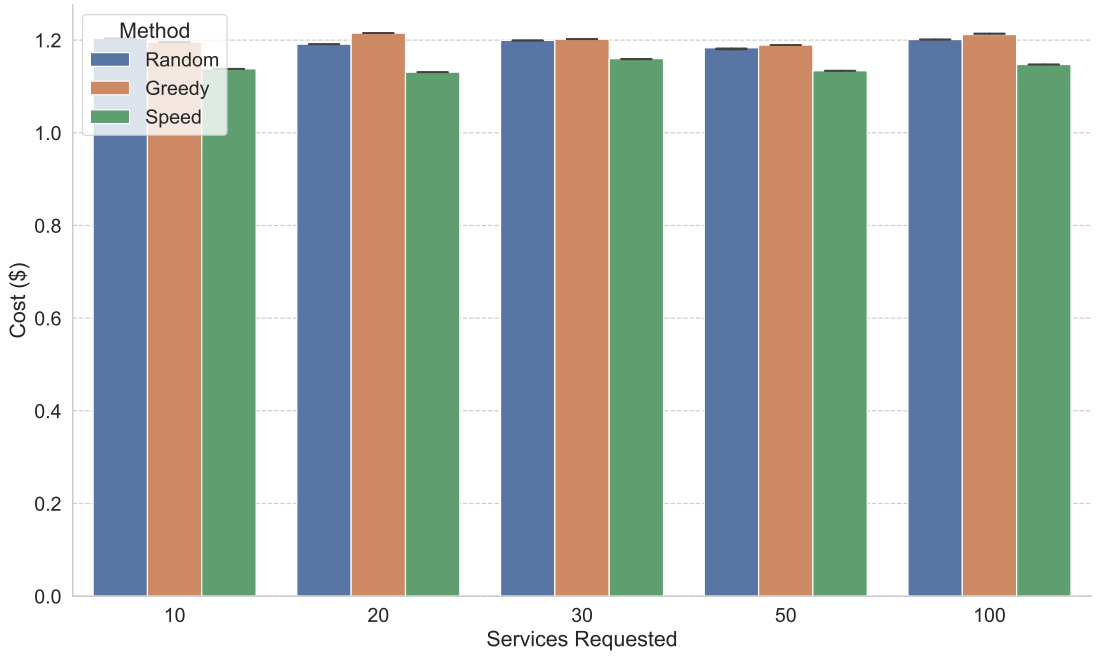

Figure 18: Service Average Execution Cost.

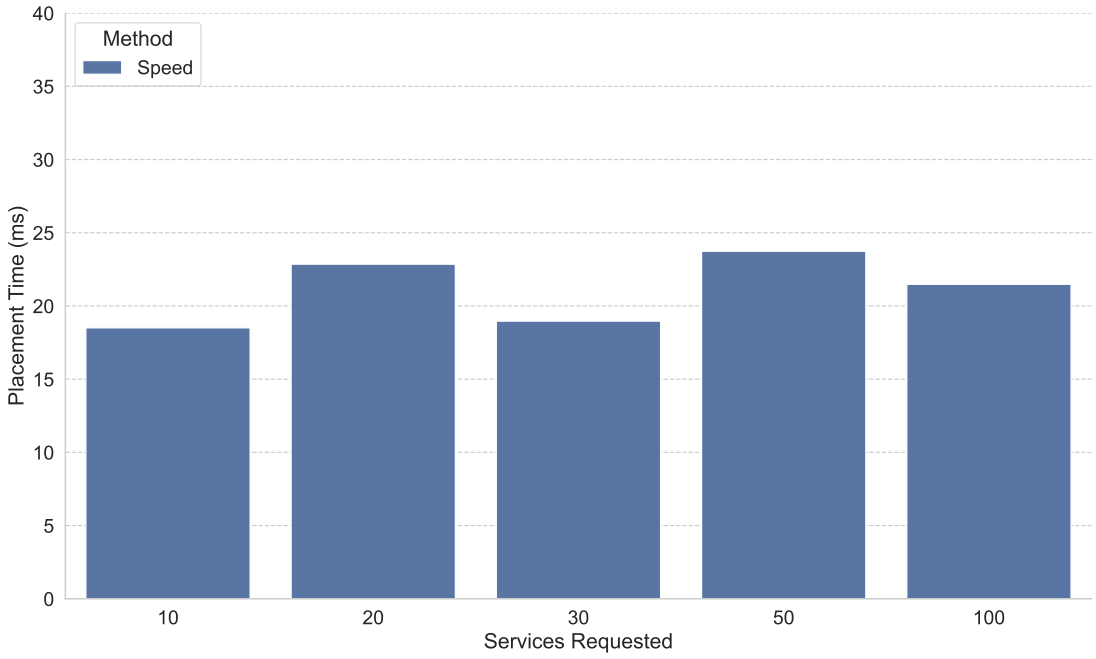

Figure 19: Placement Time.

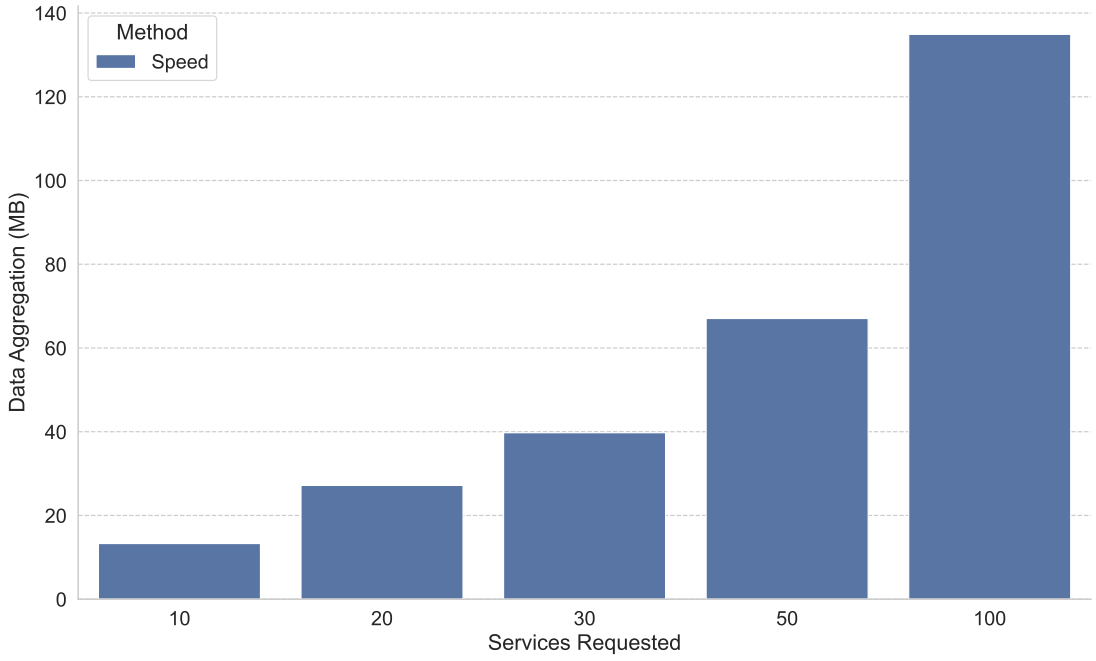

Figure 20: Data aggregation size.

6 Final Remarks

6.1 Work Plan

This section presents the work plan of this thesis. Table 8 depicts the tasks and dates associated. We show the activities already done until the present date. We also show the next steps to conclude the project.

Table 8: Tasks and schedules period

Task	Period	Status
Literature review.	10/21 - 06/22	✓
Simulation framework implementation.	04/22 - 08/22	✓
Algorithm implementation.	07/22 - 10/22	✓
Experiment execution.	10/22	✓
Thesis proposal writing.	01/22 - 02/23	✓
Article submission (B4 or higher).	12/23	✓
Proposal defense.	03/23	✓
Select and implement the environment for the testbed.	04/23 - 08/23	-
Execute the experiments in the testbed.	08/23 - 12/23	-
Thesis writing.	01/24 - 06/24	-
Article submission (B1 or higher).	06/24	-
Article published (B2 or higher).	06/24	-

One key task depicted in Table 8 is the execution of the experiments in the selected testbed. Currently, I am working as a member of the project OpenRAN@Brasil. The project aims to bootstrap the development of 5G technologies in Brazil. A project central objective is creating a testbed to be used by academic research. Thus, helping the development of the OpenRAN@Brasil testebed will facility the execution of the experiments of this thesis.

6.2 Current Production

This section briefly exemplify some projects done over the past two years. The section 6.2.1 show the inventions developed related to the VNF and VNF Placement problem. The section 6.2.2 presents articles developed regarding VNF Placement and Distributed VNF Placement. Figure 21 present the achievements' timeline.

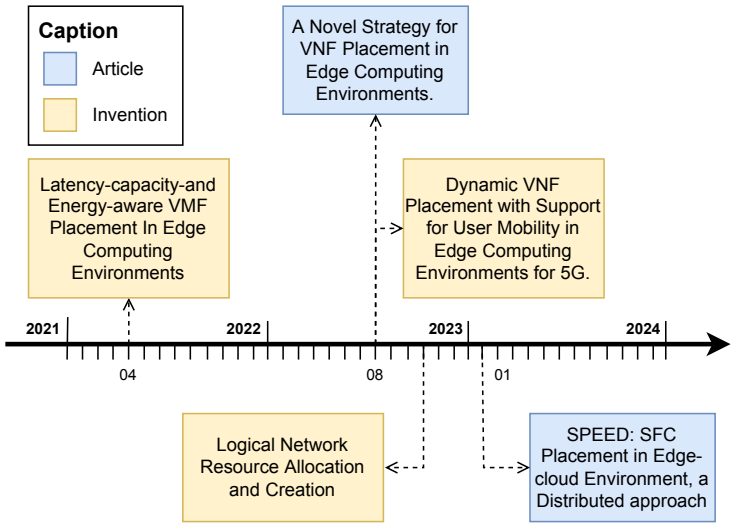

Figure 21: Achievements timeline.

6.2.1 Inventions

Title:
Latency-capacity-and Energy-aware VMF Placement In Edge Computing Environments.

Objective: In this invention, we addressed the VNF placement problem by providing a novel integer linear programming (ILP) optimization model and a novel heuristic method in order to solve the problem in a non-optimal but faster way. In our definition, the multi-objective optimization problem aims to (i) minimize the energy consumption in the edge nodes, (ii) minimize the total latency, and (iii) reduce the total cost of the infrastructure. In addition, we explore the sharing of VNFs, a feature that is still little explored in the current literature.

Date: 04/21

Project Name: Framework para gerenciamento e execução de Funções de Rede Virtualizadas em infraestruturas 5G.

Company Name: DELL

Title:
Dynamic VNF Placement with Support for User Mobility in Edge Computing Environments for 5G.
Objective: In this invention, we address the VNF placement problem by providing a novel graph-based model and a Dynamic Smart VNF Placement algorithm taking into consideration the dynamic aspects of unpredictable requests and user mobility. Such a model and algorithm have the following unique features: (i) optimizing resource utilization through an energy-efficient two-level resource sharing mechanism (VNF and SFC Instance sharing); (ii) minimizing the cost for the infrastructure provider while respecting QoS requirements; (iii) dynamic (online) placement, and (iv) user mobility aware placement solution
Date: 08/22
Project Name: Alocação de recursos distribuída e orientada para a mobilidade em sistemas 5G.
Company Name: DELL

6.2.2 Papers

This section presents the papers developed during the execution of this thesis. The first one was a novel to allocate the VNFs into an environment addressing the VNF Sharing and cost of resource minimization, this paper result in insights to this thesis. The second one is a paper depicting the main idea of this thesis.

Title:
A Novel Strategy for VNF Placement in Edge Computing Environments.
Journal: Future Internet
Date: 08/22
Status: Published | (6)

Title:
SPEED: SFC Placement in Edge-cloud Environment, a Distributed approach.
Conference: SBRC-2023
Date: 01/23
Status: Submitted

REFERENCE

1 ABUJODA, Ahmed; PAPADIMITRIOU, Panagiotis. DistNSE: Distributed network service embedding across multiple providers. **2016 8th International Conference on Communication Systems and Networks, COMSNETS 2016**, IEEE, p. 1–8, 2016. DOI: 10.1109/COMSNETS.2016.7439948.

2 ALAHMAD, Yanal; AGARWAL, Anjali; DARADKEH, Tariq. Cost and Availability-Aware VNF Selection and Placement for Network Services in NFV. In: 2020 International Symposium on Networks, Computers and Communications, ISNCC 2020. [S. l.]: IEEE, 2020. p. 1–6. ISBN 9781728156286. DOI: 10.1109/ISNCC49221.2020.9297190. Disponível em: <https://ieeexplore.ieee.org/document/9297190/>.

3 ALAM, Iqbal et al. A Survey of Network Virtualization Techniques for Internet of Things Using SDN and NFV. en. **ACM Computing Surveys**, v. 53, n. 2, p. 1–40, mar. 2021. ISSN 0360-0300, 1557-7341. DOI: 10.1145/3379444. Disponível em: <https://dl.acm.org/doi/10.1145/3379444>. Acesso em: 16 fev. 2023.

4 AVASALCAI, Cosmin; TSIGKANOS, Christos; DUSTDAR, Schahram. Decentralized Resource Auctioning for Latency-Sensitive Edge Computing. **Proceedings - 2019 IEEE International Conference on Edge Computing, EDGE 2019 - Part of the 2019 IEEE World Congress on Services**, p. 72–76, 2019. ISBN: 9781728127088. DOI: 10.1109/EDGE.2019.00027.

5 BASILI, Victor R. **Software modeling and measurement: the Goal/Question/Metric paradigm**. [S. l.: s. n.], 1992. p. 24. DOI: 10.1016/S0164-1212(99)00035-7.

6 BATTISTI, Anselmo Luiz Éden et al. A Novel Strategy for VNF Placement in Edge Computing Environments. **Future Internet**, v. 14, n. 12, 2022. ISSN 1999-5903. DOI: 10.3390/fi14120361. Disponível em: <https://www.mdpi.com/1999-5903/14/12/361>.

7 BHAMARE, Deval et al. A survey on service function chaining. en. **Journal of Network and Computer Applications**, v. 75, p. 138–155, nov. 2016. ISSN

10848045. DOI: 10.1016/j.jnca.2016.09.001. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1084804516301989>. Acesso em: 21 jan. 2023.

8 BILÒ, Vittorio; VINCI, Cosimo. On the Impact of Singleton Strategies in Congestion Games. en, 14 pages, 2017. DOI: 10.4230/LIPICS.ESA.2017.17. Acesso em: 12 nov. 2022.

9 BUNYAKITANON, Monchai et al. End-to-End Performance-Based Autonomous VNF Placement with Adopted Reinforcement Learning. **IEEE Transactions on Cognitive Communications and Networking**, v. 6, n. 2, p. 534–547, 2020. ISSN 23327731. DOI: 10.1109/TCCN.2020.2988486.

10 CHEN, Chen et al. Distributed federated service chaining for heterogeneous network environments. **ACM International Conference Proceeding Series**, Association for Computing Machinery, 2021. DOI: 10.1145/3468737.3494091.

11 CISNEROS, Josue Castaneda et al. A Survey on Distributed NFV Multi-Domain Orchestration from an Algorithmic Functional Perspective. en. **IEEE Communications Magazine**, v. 60, n. 8, p. 60–65, ago. 2022. ISSN 0163-6804, 1558-1896. DOI: 10.1109/MCOM.002.2100950. Disponível em: <https://ieeexplore.ieee.org/document/9779641/>. Acesso em: 24 out. 2022.

12 EMU, Mahzabeen; YAN, Peizhi; CHOUDHURY, Salimur. Latency aware VNF deployment at edge devices for IoT services: An artificial neural network based approach. In: IEEE. 2020 IEEE International Conference on Communications Workshops (ICC Workshops). [S. l.: s. n.], 2020. p. 1–6.

13 ETSI. **GS MEC 003 - V3.1.1 - Multi-access Edge Computing (MEC); Framework and Reference Architecture**. v. 1. [S. l.], 2022. p. 1–29.

14 GAIRING, Martin; SCHOPPMANN, Florian. Total Latency in Singleton Congestion Games. In: DENG, Xiaotie; GRAHAM, Fan Chung (Ed.). **Internet and Network Economics**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. v. 4858. Series Title: Lecture Notes in Computer Science. p. 381–387. ISBN 978-3-540-77104-3. DOI: 10.1007/978-3-540-77105-0_42. Disponível em: <http://link.springer.com/10.1007/978-3-540-77105-0_42>. Acesso em: 19 set. 2022.

15 GAO, Tao et al. Cost-Efficient VNF Placement and Scheduling in Public Cloud Networks. **IEEE Transactions on Communications**, v. 68, n. 8, p. 4946–4959, 2020. ISSN 15580857. DOI: 10.1109/TCOMM.2020.2992504.

16 GAO, Xiangqiang; LIU, Rongke; KAUSHIK, Aryan. Virtual Network Function Placement in Satellite Edge Computing with a Potential Game Approach. **IEEE Transactions on Network and Service Management**, p. 1–15, 2022. ISSN 19324537. DOI: 10.1109/TNSM.2022.3141165. arXiv: 2012.00941.

17 GARRICH, Miquel et al. IT and Multi-layer Online Resource Allocation and Offline Planning in Metropolitan Networks. **Journal of Lightwave Technology**, v. 38, n. 12, p. 3190–3199, 2020. ISSN 15582213. DOI: 10.1109/JLT.2020.2990066.

18 HALABIAN, Hassan. Distributed Resource Allocation Optimization in 5G Virtualized Networks. en. **IEEE Journal on Selected Areas in Communications**, v. 37, n. 3, p. 627–642, mar. 2019. ISSN 0733-8716, 1558-0008. DOI: 10.1109/JSAC.2019.2894305. Disponível em: <https://ieeexplore.ieee.org/document/8635559/>. Acesso em: 6 out. 2022.

19 HALPERN, Joel; PIGNATARO, Carlos. **Service function chaining (SFC) architecture**. [S. l.], 2015.

20 HUFF, Alexandre; VENÂNCIO, Giovanni; DUARTE JR., Elias P. Multi-SFC: Orquestração de SFCs Distribuídas sobre Múltiplas Nuvens em Múltiplos Domínios com Múltiplas Plataformas NFV. In: ANAIS do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2019). [S. l.]: Sociedade Brasileira de Computação - SBC, 2019. p. 777–790. DOI: 10.5753/sbrc.2019.7402. Disponível em: <https://sol.sbc.org.br/index.php/sbrc/article/view/7402>.

21 KAUR, Karamjeet; MANGAT, Veenu; KUMAR, Krishan. A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture. **Computer Science Review**, v. 38, p. 100298, 2020. ISSN 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2020.100298. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1574013720303981>.

22 _____. A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture. en. **Computer Science Review**, v. 38, p. 100298, nov. 2020. ISSN 15740137. DOI: 10.1016/j.cosrev.2020.100298. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1574013720303981>. Acesso em: 16 nov. 2022.

23 KIM, Sang Il; SUNG KIM, Hwa. A VNF Placement Method Considering Load and Hop count in NFV Environment. **International Conference on Information**

Networking, IEEE, 2020-Janua, p. 707–712, 2020. ISSN 19767684. DOI: 10.1109/ICOIN48656.2020.9016492.

24 KIRAN, Nahida et al. VNF Placement and Resource Allocation in SDN/NFV-Enabled MEC Networks. In: 2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW). [S. l.: s. n.], 2020. p. 1–6. DOI: 10.1109/WCNCW48565.2020.9124910.

25 KUO, Tung-Wei et al. Deploying chains of virtual network functions: On the relation between link and server usage. en. In: IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications. San Francisco, CA, USA: IEEE, abr. 2016. p. 1–9. ISBN 978-1-4673-9953-1. DOI: 10.1109/INFOCOM.2016.7524565. Disponível em: <http://ieeexplore.ieee.org/document/7524565/>. Acesso em: 17 nov. 2022.

26 _____. Deploying chains of virtual network functions: On the relation between link and server usage. **IEEE/ACM Transactions On Networking**, IEEE, v. 26, n. 4, p. 1562–1576, 2018.

27 LAGHRISSI, Abdelquoddouss; TALEB, Tarik. A Survey on the Placement of Virtual Resources and Virtual Network Functions. **IEEE Communications Surveys and Tutorials**, IEEE, v. 21, n. 2, p. 1409–1434, 2019. ISSN 1553877X. DOI: 10.1109/COMST.2018.2884835.

28 LEE, Whay C. Topology aggregation for hierarchical routing in ATM networks. **Computer Communication Review**, v. 25, n. 2, p. 82–92, 1995. ISSN 01464833. DOI: 10.1145/210613.210625.

29 LI, Defang et al. Virtual network function placement and resource optimization in NFV and edge computing enabled networks. **Computer Networks**, Elsevier B.V., v. 152, p. 12–24, 2019. ISSN 13891286. DOI: 10.1016/j.comnet.2019.01.036. Disponível em: <https://doi.org/10.1016/j.comnet.2019.01.036>.

30 LI, Shuopeng et al. An Attention Based Deep Reinforcement Learning Method for Virtual Network Function Placement. **2020 IEEE 6th International Conference on Computer and Communications, ICCC 2020**, p. 1005–1009, 2020. DOI: 10.1109/ICCC51575.2020.9345041.

31 LI, Taixin; ZHOU, Huachun; LUO, Hongbin. A new method for providing network services: Service function chain. en. **Optical Switching and Networking**, v. 26, p. 60–68, nov. 2017. ISSN 15734277. DOI: 10.1016/j.osn.2015.09.005. Disponível

em: <https://linkinghub.elsevier.com/retrieve/pii/S1573427715000764>. Acesso em: 16 nov. 2022.

32 LIU, Yi et al. GDM: A General Distributed Method for Cross-Domain Service Function Chain Embedding. **IEEE Transactions on Network and Service Management**, v. 17, n. 3, p. 1446–1459, 2020. ISSN 19324537. DOI: 10.1109/TNSM.2020.2993364.

33 MAMUSHIANE, Lusani et al. Overview of 9 Open-Source Resource Orchestrating ETSI MANO Compliant Implementations: A Brief Survey. **2019 IEEE 2nd Wireless Africa Conference, WAC 2019 - Proceedings**, IEEE, p. 1–7, 2019. DOI: 10.1109/AFRICA.2019.8843421.

34 MAO, Yuyi et al. A Survey on Mobile Edge Computing: The Communication Perspective. **IEEE Communications Surveys and Tutorials**, v. 19, n. 4, p. 2322–2358, 2017. ISSN 1553877X. DOI: 10.1109/COMST.2017.2745201. arXiv: 1701.01090.

35 MASKIN, Eric. Nash Equilibrium and Welfare Optimality. **Review of Economic Studies**, v. 66, p. 23–38, 1999. Reprinted in J.J. Laffont (ed.), The Principal Agent Model: The Economic Theory of Incentives, London: Edward Elgar, 2003.

36 MECHTRI, Marouen; BENYAHIA, Imen Grida; ZEGHLACHE, Djamal. Agile service manager for 5G. In: p. 1285–1290. ISBN 9781509002238. DOI: 10.1109/NOMS.2016.7503004.

37 MIJUMBI, Rashid et al. Network function virtualization: State-of-the-art and research challenges. **IEEE Communications Surveys and Tutorials**, v. 18, n. 1, p. 236–262, 2016. ISSN 1553877X. DOI: 10.1109/COMST.2015.2477041. arXiv: 1509.07675.

38 MOHAMAD, Amir; HASSANEIN, Hossam S. On Demonstrating the Gain of SFC Placement with VNF Sharing at the Edge. In: 2019 IEEE Global Communications Conference (GLOBECOM). [S. l.: s. n.], 2019. p. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9014106.

39 MORABITO, Roberto et al. Consolidate IoT Edge Computing with Lightweight Virtualization. **IEEE Network**, v. 32, n. 1, p. 102–111, 2018. DOI: 10.1109/MNET.2018.1700175.

40 MOSTAFAVI, Seyedakbar; HAKAMI, Vesal; SANAEI, Maryam. Quality of service provisioning in network function virtualization: a survey. en. **Computing**, v. 103, n. 5, p. 917–991, mai. 2021. ISSN 0010-485X, 1436-5057. DOI: 10.1007/s00607-021-00925-x. Disponível em: <https://link.springer.com/10.1007/s00607-021-00925-x>. Acesso em: 26 nov. 2022.

41 NGUYEN, Duong Tuan et al. Placement and Chaining for Run-Time IoT Service Deployment in Edge-Cloud. **IEEE Transactions on Network and Service Management**, IEEE, v. 17, n. 1, p. 459–472, 2020. ISSN 19324537. DOI: 10.1109/TNSM.2019.2948137.

42 NIU, Meng; CHENG, Bo; CHEN, Jun Liang. GPSO: A graph-based heuristic algorithm for service function chain placement in data center networks. **Proceedings - 2020 IEEE 13th International Conference on Services Computing, SCC 2020**, p. 256–263, 2020. DOI: 10.1109/SCC49832.2020.00041.

43 OWEN, G. **Game Theory**. [S. l.]: Emerald Group Publishing Limited, 2013. ISBN 9781781905074. Disponível em: <https://books.google.com.br/books?id=OfnLkgEACAAJ>.

44 PATTARANANTAKUL, Montida; VORAKULPIPAT, Chalee; TAKAHASHI, Takeshi. Service Function Chaining security survey: Addressing security challenges and threats. en. **Computer Networks**, v. 221, p. 109484, fev. 2023. ISSN 13891286. DOI: 10.1016/j.comnet.2022.109484. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1389128622005187>. Acesso em: 16 fev. 2023.

45 PEI, Jianing et al. Optimal VNF Placement via Deep Reinforcement Learning in SDN/NFV-Enabled Networks. **IEEE Journal on Selected Areas in Communications**, IEEE, v. 38, n. 2, p. 263–278, 2020. ISSN 15580008. DOI: 10.1109/JSAC.2019.2959181.

46 PHAM, Chuan et al. Traffic-Aware and Energy-Efficient vNF Placement for Service Chaining: Joint Sampling and Matching Approach. **IEEE Transactions on Services Computing**, v. 13, n. 1, p. 172–185, 2020. ISSN 19391374. DOI: 10.1109/TSC.2017.2671867.

47 RASMUSEN, E. **Games and Information: An Introduction to Game Theory**. [S. l.]: Wiley, 2006. ISBN 9781405136662. Disponível em: <https://books.google.com.br/books?id=5XEMuJwnBmUC>.

48 REYHANIAN, Navid et al. JOINT RESOURCE ALLOCATION AND ROUTING FOR SERVICE FUNCTION CHAINING WITH IN-SUBNETWORK PROCESSING Department of Electrical and Computer Engineering , University of Minnesota , Twin Cities , MN , USA Huawei Canada Research Center , Ottawa , Canada Shenzhen. **ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**, IEEE, p. 4985–4989, 2020.

49 ROSENTHAL, Robert W. A class of games possessing pure-strategy Nash equilibria. **International Journal of Game Theory**, v. 2, n. 1, p. 65–67, 1973. ISSN 1432-1270. DOI: 10.1007/BF01737559. Disponível em: <https://doi.org/10.1007/BF01737559>.

50 RUIZ, Lidia et al. Genetic algorithm for holistic VNF-mapping and virtual topology design. **IEEE Access**, v. 8, p. 55893–55904, 2020. ISSN 21693536. DOI: 10.1109/ACCESS.2020.2982018.

51 SANTOS, Guto Leoni et al. Service Function Chain Placement in Distributed Scenarios: A Systematic Review. **Journal of Network and Systems Management**, Springer US, v. 30, n. 1, p. 1–39, 2022. ISSN 15737705. DOI: 10.1007/s10922-021-09626-4. Disponível em: <https://doi.org/10.1007/s10922-021-09626-4>.

52 SARAIVA DE SOUSA, Nathan F. et al. Network Service Orchestration: A survey. **Computer Communications**, Elsevier B.V., v. 142-143, April, p. 69–94, 2019. ISSN 1873703X. DOI: 10.1016/j.comcom.2019.04.008. arXiv: 1803.06596. Disponível em: <https://doi.org/10.1016/j.comcom.2019.04.008>.

53 SARRIGIANNIS, Ioannis et al. Application and Network VNF migration in a MEC-enabled 5G Architecture. **IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD**, 2018-Septe, p. 1–6, 2018. Publisher: IEEE ISBN: 9781538661512. ISSN 23784873. DOI: 10.1109/CAMAD.2018.8514943.

54 SBABOU, Samir; SMAOUI, Hatem; ZIAD, Abderrahmane. Equilibrium Analysis in Singleton Congestion Games. en, p. 17.

55 SON, Jungmin; BUYYA, Rajkumar. Latency-aware Virtualized Network Function provisioning for distributed edge clouds. **Journal of Systems and Software**, v. 152, p. 24–31, 2019. Publisher: Elsevier Inc. ISSN 01641212. DOI: 10.1016/j.jss.2019.02.030.

56 SOUALAH, Oussama et al. A link failure recovery algorithm for Virtual Network Function chaining. **Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on Integrated Network and Service Management**, May, p. 213–221, 2017. DOI: 10.23919/INM.2017.7987282.

57 STANLEY, Richard P. Enumerative Combinatorics Volume 1 second edition. **Cambridge studies in advanced mathematics**, 2011.

58 SUN, Gang et al. Service Function Chain Orchestration Across Multiple Domains: A Full Mesh Aggregation Approach. **IEEE Transactions on Network and Service Management**, IEEE, v. 15, n. 3, p. 1175–1191, 2018. ISSN 1932-4537. DOI: 10.1109/TNSM.2018.2861717. Disponível em: <https://ieeexplore.ieee.org/document/8485473/%20https://ieeexplore.ieee.org/document/8423711/>.

59 TOSCA, OASIS. **TOSCA Simple Profile for Network Functions Virtualization (NFV) Version 1.0, Committee Specification Draft 04**. [S. l.]: Technical Report, 2017. Disponível em: <https://docs.oasis-open.org/tosca/tosca-nfv/v1.0/csd04/tosca-nfv-v1.0-csd04.html>.

60 TOUMI, Nassima; BAGAA, Miloud; KSENTINI, Adlen. **Hierarchical Multi-Agent Deep Reinforcement Learning for SFC Placement on Multiple Domains**. [S. l.].

61 TOUMI, Nassima et al. On cross-domain Service Function Chain orchestration: An architectural framework. **Computer Networks**, Elsevier B.V., v. 187, December 2020, p. 107806, 2021. ISSN 13891286. DOI: 10.1016/j.comnet.2021.107806. Disponível em: <https://doi.org/10.1016/j.comnet.2021.107806>.

62 _____. Towards Cross-Domain Service Function Chain Orchestration. **2020 IEEE Global Communications Conference, GLOBECOM 2020 - Proceedings**, 2020-January, 2020. DOI: 10.1109/GLOBECOM42002.2020.9348028.

63 VAQUERO, Luis M. et al. Research challenges in nextgen service orchestration. **Future Generation Computer Systems**, v. 90, p. 20–38, 2019. ISSN 0167739X. DOI: 10.1016/j.future.2018.07.039. arXiv: 1806.00764.

64 WANG, Shuyi; CAO, Haotong; YANG, Longxiang. A Survey of Service Function Chains Orchestration in Data Center Networks. en. In: 2020 IEEE Globecom Workshops (GC Wkshps. Taipei, Taiwan: IEEE, dez. 2020. p. 1–6. ISBN 978-1-72817-307-8. DOI: 10.1109/GCWkshps50303.2020.9367463. Disponível em: <https://ieeexplore.ieee.org/document/9367463/>. Acesso em: 16 fev. 2023.

65 XU, Qi et al. Low Latency Security Function Chain Embedding Across Multiple Domains. en. **IEEE Access**, v. 6, p. 14474–14484, 2018. ISSN 2169-3536. DOI: 10.1109/ACCESS.2018.2791963. Disponível em: <https://ieeexplore.ieee.org/document/8254344/>. Acesso em: 26 set. 2022.

66 YI, Bo et al. A comprehensive survey of Network Function Virtualization. **Computer Networks**, v. 133, p. 212–262, 2018. ISSN 13891286. DOI: 10.1016/j.comnet.2018.01.021.

67 ZAVLANOS, Michael M; SPESIVTSEV, Leonid; PAPPAS, George J. A distributed auction algorithm for the assignment problem. In: PROCEEDINGS of the IEEE Conference on Decision and Control. [S. l.: s. n.], 2008. p. 1212–1217. ISBN 9781424431243. DOI: 10.1109/CDC.2008.4739098.

68 ZHANG, Jianwei; ZHAO, Chenwei. Q-SR: An extensible optimization framework for segment routing. **Computer Networks**, Elsevier B.V., v. 200, p. 108517, October 2021. ISSN 13891286. DOI: 10.1016/j.comnet.2021.108517. Disponível em: <https://doi.org/10.1016/j.comnet.2021.108517>.