



# Distributed Auction-Based SFC Placement in a Multi-domain 5G Environment

Evandro L. C. Macedo<sup>2</sup> · Anselmo L. E. Battisti<sup>1</sup> · Juan Lucas Vieira<sup>1</sup> · Julia Noce<sup>3</sup> · Paulo F. Pires<sup>1,3</sup> · Débora C. Muchaluat-Saade<sup>1</sup> · Ana C. B. Oliveira<sup>3</sup> · Flavia C. Delicato<sup>1</sup>

Received: 30 May 2023 / Accepted: 31 August 2023  
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2023

## Abstract

The fifth generation of mobile networks (5G) brings an evolution of network service provisioning through a new communication paradigm, which enables the development of new applications and improves users' experience. With 5G, it is envisioned that networks will provide services accessed by a variety of mobile users, some of such services requiring ultra-low latency and very high reliability. 5G also enables massive connectivity of sensors and actuators in the Internet of Things and Cyber-physical Systems scenarios, standing for massive Machine-Type Communication applications. Network Function Virtualization (NFV) is an emerging and promising solution to deal with such demands for flexible and agile service provisioning by replacing dedicated hardware implementations with software instances based on virtualization. A well-known challenge in NFV is the resource allocation problem. In particular, the Service Function Chain (SFC) placement problem is considered a major challenge, even more, when considering the distributed placement in a multi-administrative domain context. We propose a novel solution for the SFC placement problem called Multi-Domain Distributed Auction-Based SFC Placement Algorithm, which relies on an auction-based strategy. The proposed algorithm targets different 5G application scenarios with a focus on solving the SFC placement problem taking into consideration the dynamic aspects of unpredictable requests and the management of the entire auction process to decide which domain will be selected to meet the required SFC. Evaluations were carried out in a simulated environment aiming at assessing the performance of the algorithm in terms of the profit for each service provider during the allocation of the requested services in the respective domain that the service provider is responsible for. Results showed that adopting an auction-based mechanism, and thus allowing a domain to outsource the provision of requested services, successfully reduced the total cost of the service execution in a multi-domain environment. The auction-based approach increases the service provider profit by around 20% in the tested scenarios. Moreover, the number of services placed increases in comparison to the approach where all the services must be executed in the decision domain (with no auction).

**Keywords** Multi-domain environment · Auction placement · Service function chain · Edge computing

## Introduction

The demands for flexibility and stringent Quality of Service (QoS) requirements are continuously reinforced with the development of the fifth generation of mobile networks (5G). The number of novel services increases, leading to remarkable growth in network traffic. In such a context, Network Function Virtualization (NFV) is an emerging and

promising solution to deal with such demands by replacing dedicated hardware implementations with software instances based on virtualization. By doing so, NFV provides the necessary flexibility to enable agile, cost-effective, and on-demand service delivery models combined with automated management. With NFV, the state-of-the-art is leveraged by the virtualization of computing, networking, and storage, provided by the decoupling of network functions from underlying proprietary hardware, thus allowing software implementations of such network functions on commodity hardware. The requested services are implemented by Virtual Network Functions (VNF) that run on an NFV Infrastructure (NFVI) and are organized in a predefined order

---

This article is part of the topical collection “Dependable Cyber-Physical Systems and Cyber Security” guest edited by Deepak Puthal and Niranjan K. Ray.

---

Extended author information available on the last page of the article

through which application traffic flows traverse. Such an organized set of VNFs is known as Service Function Chaining (SFC), which defines a specific provided service and replaces a chain of physical equipment that provides complete functionality required by a user.

Cloud and Edge Computing are enabler platforms to deploy VNFs on virtual machines of data centers, either far from or close to end users, depending on which applications are considered. For instance, applications that present stringent latency requirements tend to better perform if their required VNFs are deployed at the computing Edge Nodes, given their proximity to users. Edge Computing brings significant benefits such as reducing communication delays, relieving network bandwidth usage, and bringing control decisions to the edge of the network. Moreover, Edge Computing can take advantage of the collaboration among devices to compensate for their lower resource capacity. When integrating NFV and Edge Computing, VNF Instances are deployed in edge nodes. Such a strategy facilitates the QoS fulfillment of time-critical applications while taking advantage of possible idle resources of the various edge devices. Specially in the context of 5G, leveraging the benefits of integrating Multi-Access Edge Computing (MEC) and NFV will enable service providers to meet the requirements of brand-new scenarios of ultra-reliable and low-latency communication (URLLC), enhanced Mobile Broadband (eMBB), and massive Machine-Type Communication (mMTC). In such a way, service providers can fulfill the expected QoS while raising their revenue and reducing costs.

There are many challenges related to resource management in NFV-Edge environments. A well-known challenge in the NFV context is the VNF and SFC placement [1–3]. VNF placement consists of selecting, among a set of candidate processing nodes, the one where a given VNF will be deployed and executed. SFC placement denotes choosing the nodes to run each VNF that composes it and the links between them. The placement problem needs to take into account the characteristics of nodes, network, and the required QoS parameters. This problem is even more challenging when we consider the distributed placement in a multi-domain context.

The Distributed-SFC Placement problem consists of selecting a specific domain among a set of network domains that can meet the user and applications SFC Requests and, consequently, the most suitable edge nodes within such a domain to deploy VNFs. We consider as domain the administrative grouping of networks that compose the infrastructure of a provider. Usually, the NFV Orchestrator (NFVO) is the component responsible for the placement, but in typical architectures (e.g., ETSI [4]), there is no component responsible for choosing which domain will take care of the SFC Request. The placement must be accomplished in such a way that the

QoS requirements of the applications are met, while trying to increase the revenue of service providers by optimizing their resource usage at the edge tier. All these tasks must be done without violating the Service-Level Agreement (SLA).

The main entities involved in the SFC placement problem are: (i) users that make SFC requests; (ii) Service Function Chains (SFCs) that define the sequence of VNFs to be placed in the computing nodes, based on VNF Types (a predefined description of a VNF); (iii) the VNF Instances, which are the VNFs that are executing in the edge environment; (iv) the edge nodes that provide computational resources to execute such VNF Instances; and (v) the domains that can be selected to take care of SFC Requests. The number of entities and their high heterogeneity along with the dynamism of resource availability make this a challenging problem. Moreover, an edge node can execute multiple VNF Instances, and a single VNF Instance can be shared among multiple applications. Additionally, every application request needs to traverse the VNF Instances of the corresponding SFC. For simplicity, in the remainder of the document, we use the term VNF to explicitly refer to the VNF Type (e.g., firewall, proxy, etc.), and the term instance or VNF Instance to refer to an instance of a VNF Type that is running in a node.

We addressed the distributed-SFC placement problem by providing a novel auction-based strategy to solve the problem in an effective way, promoting the scalability of our solution in terms of the number of domains, users, and edge nodes. The proposed algorithm targets different 5G application scenarios with focus on solving the SFC placement problem taking into account the dynamic aspects of unpredictable requests and the management of the entire auction process to decide which domain will be selected to meet the required SFC. Enabling the proposed algorithm and strategy brings up challenges as the need for new components, such as an orchestrator and an auction manager will orchestrate the auction process to establish consensus among the involved domains. While addressing these new challenges, the proposal still maintains the key objectives of minimizing the energy consumption of edge nodes, minimizing the total SFC latency, and increasing the revenue of the service provider by reducing the total infrastructure cost, which is crucial in Edge Computing solutions. The main contributions of our paper are (i) an innovative multi-domain approach to tackle the SFC placement problem in large-scale, distributed scenarios; and (ii) a novel auction-based algorithm to decide which domain will be responsible for placing a requested SFC.

## Related Work

Recent solutions for resource allocation and orchestration problems in NFV consider a variety of approaches, such as combinatorial optimization theory, deep reinforcement

learning, and game theory, among others, including heuristic-based solutions [1, 2, 5–8].

In the paper [9], the authors argue that the NFV paradigm has introduced a new market focused on the supply and distribution of VNFs. Taking advantage of the virtualization of network services, infrastructure providers (InP) can benefit from an NFV market by providing their infrastructures to meet the demands of end users who, in turn, can acquire VNFs following a model similar to the one adopted in cloud computing, called VNFs-as-a-Service (VNFaaS). In this context, they claim that solutions that promote competition between InPs can lead to lower prices, while increasing VNF performance to accommodate specific end-user demands. In this context, the authors explore the use of an auction mechanism associated with blockchain technology to provide an efficient and flexible solution to support fair and auditable competition between providers. They propose adopting a blockchain-based reverse auction, in which sellers compete for buyers, and the auction winner will be the seller that provides the lowest price for the better-tailored service. Such a strategy allows InPs to monetize their infrastructure by receiving requests to host VNFs and give a sealed-bid based on their specific business algorithms. InPs estimate, based on the VNF's and user's demands, the costs and efforts, according to their configurations, to supply infrastructures and resources for each request. Once the bidding process is finished, the auction owner can recommend, based on the best price, the suitable infrastructure to host VNFs to supply the requested demands. The use of blockchain and smart contracts in this context is motivated by the need to provide trust and reliability in the auction process. Such technologies provide immutable, permanent records that allow stakeholders to audit and trust persisted data. This proposal has in common with our work the idea of using the auction mechanism as a negotiation option between the parties. And since the internal details of each provider do not need to be exposed, the proposal can be easily adopted in multi-domain environments. However, unlike our work, the paper is focused on the architecture and detailing of the process of creating and managing bids, taking into account the offers and demands. The specific solution for the placement of VNFs is left for future work. The authors also do not consider the need to chain VNFs to compose SFCs, with the entire allocation process focused on individual VNFs.

In [10], similarly to our work, the authors address the challenges involved in orchestrating chains of VNFs composing SFCs. They share our view that such a problem is even more challenging in multi-domain scenarios, due to issues of confidentiality of the information held by each provider and their different operational strategies. In particular, they focus on information regarding the network topology, and how much the lack of detailed information about it hinders the operation of an SFC orchestrator. In a

multi-domain topology, a vertex cannot know the graph of the entire network; it can only know the physical nodes in its own domain and several boundary vertices in other domains [10]. They then propose a centralized approach to solve such a problem, which is divided into several steps. In the first step, the authors employ the full mesh aggregation (FMA) technique [11] to build an abstract topology composed of the edge vertices in each domain and the inter-domain edges of the substrate network. Based on this FMA topology, they can find the abstract paths that connect the source and destination of the SFC request, which helps to avoid unnecessary forwards in the SFC partition process. Each domain's confidentiality is respected by exposing information about its available resources only in the form of a weight, used in its SFC partition algorithm. After partitioning the SFC into sub-SFCs, they propose a heuristic algorithm for the placement of each VNF and virtual links within each domain. Such an algorithm seeks to optimize the delay while balancing the load on the underlying network. As a final step, from the several abstracted paths, they employ a bidding mechanism to select the path with minimal delay or resource cost as the final SFC placement scheme. A big difference between the proposal presented in [10] and ours is that they adopt a centralized approach. The authors argue that the advantage of the centralized method is that a third party can get a global view instead of just a local perspective when trying to map an SFC into multiple domains, which can improve the SFC deployment scheme. In fact, global solutions can achieve a global optimum, but at the expense of poor scalability. We opted for a decentralized approach to leverage its known advantages of greater scalability, as we believe that with the widespread adoption of NFV technology, the number of entities involved will grow tremendously.

Reducing energy consumption is a key challenge in many computational fields [12]. In [13], the authors focus on the reduction of energy consumption when mapping SFC requests to services in a multi-domain environment. As characteristics that contribute to the complexity of an online, multi-domain environment, the authors highlight the fact that information regarding the arriving SFC requests cannot be obtained in advance and that the complete information regarding each domain of the substrate network might not be available due to privacy concerns. To save energy, the authors set a goal of minimizing power consumption by activating the least amount of servers that can meet the service demands. First, they formulate the SFC orchestration problem as an ILP that aims to minimize computing and networking power consumption, which is calculated according to the energy consumption of a server depending on its workload and the number and utilization of its network ports. The constraints of the problem describe the restrictions related to the resource capacity of the nodes and links and other VNF-related constraints in the multi-domain

environment, such as enforcing that the order of the VNFs in the SFC is maintained when VNFs are allocated to different domains. Due to the NP-hardness of the ILP approach, the authors also propose a lower complexity heuristic to achieve sub-optimal solutions. To guide the orchestration of SFC requests, the authors model the multi-domain environment with two types of graphs. The domain-level function graph (DLFG) is an undirected weighted graph where each node represents a domain, and each edge represents the connection between domains in the substrate network. Each node also has as an attribute the number of VNF types hosted in each domain. This graph is used by the orchestrator in the heuristic to find which paths can be used between domains considering the service source-destination pair and the VNF types that are hosted in each domain. The privacy of each domain is preserved since the graph does not contain the quantity or where these VNFs are located. The domain-intra function graph (DIFG) is a directed weighted graph maintained by each domain in which vertexes represent computing and forwarding nodes and edges represent the connectivity between the nodes. This graph is used by each domain orchestrator to search for possible partitions of the SFC in the domain. After these sub-SFCs are enumerated, a main orchestrator selects the appropriate solution to reduce energy consumption. The gains in terms of energy usage reduction are achieved by prioritizing mapping VNFs to servers that are already active during the SFC mapping. However, the approach has the drawback of concentrating the load among fewer nodes, which might make it more challenging to maintain the SLA when service demands increase after the placement. This approach might also lead to uneven wear of servers' hardware. Different from our work which follows a decentralized approach, the authors consider a hierarchical approach that might also suffer from the scalability issues mentioned previously, since the SFC placement decisions depend on a main orchestrator.

Another relevant work that solved the SFC placement problem in a distributed fashion was proposed by Liu et al. [14]. They present a distributed-SFC Placement and a load balance component named GDM. The proposed solution is executed in a multi-domain environment and is based on an auction approach. Each domain is independent and shares information with the other domains only about the border nodes and the nodes where each VNF type can be executed. Also, each domain can execute its own orchestrator deployed 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 each defined segment. After the segmentation phase, each segment is sent to candidate domains, and they will use a distributed auction strategy to decide which domain will execute each segment.

Nonetheless, the strategy proposed in [14] has many disadvantages. The adoption of a distributed auction strategy is typically employed in scenarios where the participants are competing against each other, however, in cooperative environments, like federations, where the objective is to maximize the general wealth rather than the individual gain, this strategy could not be applied. Another flaw is that distributed auction according to [15] converges in  $2\Delta$  iterations without changes in the local winning list. As  $\Delta$  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 unfeasible 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, which is a counter sense, because if the domains was not selected in the first part of the process probably, it also will be not feasible in the second part too.

The authors in [16] developed a new double-auction approach that is used for both service function chain routing and NFV price adjustment to maximize the profits of NFV broker, customers, and service providers. They propose a Game Theory approach through a noncooperative game where the participants play a game with each other for their own benefits. Although the profit in the double-auction method is higher than that in the single-auction method, in this proposal, authors lack a distributed approach which is more appropriate for dynamic edge environments, since they use a centralized SFC Broker. Buyers request SFCs with the bidding price, suppliers provide their services with the asking price, and the broker decides the transaction value.

The approach proposed by [17] 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 can only be used for finding a near-optimal solution in tiny environments. Table 1 resumes the two most relevant related work and our approach.



## A Multi-domain Distributed Auction-Based SFC Placement Algorithm

In this section, we describe the proposed solution. Section “[System Model and Notations](#)” presents the system considered in the solution, while Sect. “[Proposed Architecture for Distributed-SFC Placement](#)” describes the proposed architecture. Section “[Auction Components and Operation](#)” describes the auction-based multi-domain SFC placement strategy.

### System Model and Notations

Here, we describe the system model considered in our solution. The model encompasses the environment and the elements where our proposed algorithm will be executed to solve the SFC Placement Problem in a distributed fashion. The SFC Placement is performed in an edge-cloud environment. Each user is associated with one node in the edge tier, and each node belongs to a single domain.

The Distributed-SFC Placement Problem can be mapped to an undirected graph  $G = (H, L)$ , as described in Table 2. The set  $H = \{h_1, h_2, \dots, h_n\}$  represents the hosts (or nodes) where VNF Instances will be executed. The nodes can also act as network packet forwarders (network gateways). Each node  $h_n$  has the following attributes:  $d_h$  defines the associated domain; the total resources are represented by  $R_h = \{c_h, m_h\}$ , where  $c_h$  defines the CPU and  $m_h$  defines the memory capacity; a list of VNF images  $IM_h = \{vi_1, vi_2, \dots, vi_n\}$ , that can be instantiated at the node, where  $vi_n \in VI$ , which is the set of VNF images.

The set  $L = \{l_1, l_2, \dots, l_n\}$  denotes the physical links. For every link  $l \in L$ , between two adjacent nodes  $h_1$  and  $h_2$ , we use  $bw_{h_1h_2}$  to denote its bandwidth capacity, and  $dl_{h_1h_2}$  to denote its delay. The set  $U = \{u_1, u_2, \dots, u_n\}$  is the set of users that make SFC Requests. A user has attribute  $node_u$  that represents the access node where the user is associated with, the user can be associated with only one access node at a time. The set  $D = \{d_1, d_2, \dots, d_n\}$  denotes the domains of the environment. Each domain has an attribute  $nodes_d$

that defines all the nodes pertaining to the domain, and an attribute  $neighbor_d$  that is a set with all neighbor domains that can be reached by any gateway of domain  $d$  directly.

Besides the network topology, we also model the VNFs that may provide common network functions (e.g., firewall and proxy functions), or high-level services, such as image processing, cache, video streamer, etc. A VNF  $f \in F$  demands a minimal number of computational resources defined as  $cpu_f$  and  $mem_f$  that represent the minimum number of CPUs and memory in the node to execute VNF  $f$ .

For each service requested by the users, one auction is created. Auctions are modeled as follows. The set  $A = \{a_1, a_2, \dots, a_n\}$  denotes the set of auctions. Each auction  $a_i$  has the following attributes. The  $decision\_domain_a$  is the domain that coordinates the auction and is responsible for providing the service for the user by directly executing the service, or relying on another domain. The  $time_a$  is the time when the auction starts. The  $duration_a$  is the auction duration, which represents the time that the decision domain will wait for the bids. The  $sfc\_request_a$  is the service requested by the user.

The  $bid\_request_i$  represents a request from the decision domain asking the other candidate domains that meet the requirements to participate in the auction  $i$ . It is defined by the tuple  $\{sfc\_request\_vnfs, ingress\_participant\_gw, consumed\_time, remain\_time\}$ , where the  $sfc\_request\_vnfs$  is the set of VNFs that compose the SFC requested by the user,  $ingress\_participant\_gw$  is the gateway of the participant domain where the decision domain will establish a connection, and  $consumed\_time$  is the elapsed time to reach the  $ingress\_participant\_gw$  from the SFC Request source node that is subtracted from the maximum service delay defined in its SLA to find the  $remain\_time$ . The  $remain\_time$  represents the remaining delay that the participant domain will have to process the SFC packets—i.e., the participant domain must find a placement plan with a packet processing delay within the  $remain\_time$ . The winner will be one of the participant domains, while the destination domain contains the SFC’s egress node. Figure 1 illustrates such definitions.

The  $bid\_responses_i$  represent the bid responses created by each participant domain in response to  $bid\_request_i$

**Table 1** Auction strategies to solve the SFC Placement problem

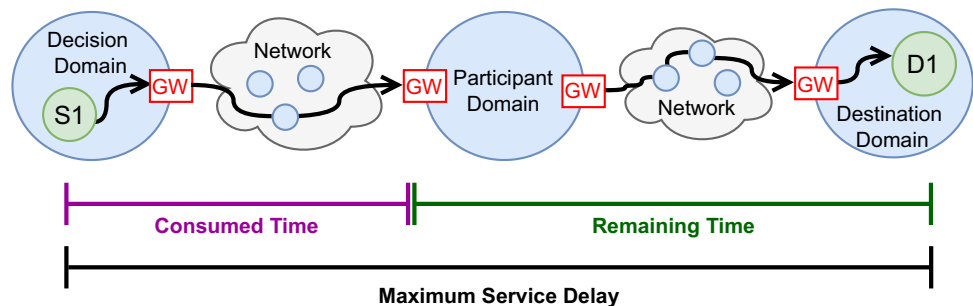
Paper	Auction approach
Liu et al. [14]	A GDM (General Distributed Method) for cross-domain SFC embedding maintains the privacy and autonomy of domains, besides giving fair competition among domains while balancing loads among domains
Avasalcari et al. [17]	A distributed auction strategy based on an auction house (dispatcher). For each application, a list of auction bidders is created. The bidders bid for parts of the application. The dispatcher finds an allocation of tasks (via MIP) to the bidding nodes that satisfy the given SLA
Our approach	A blind auction based on the first-price sealed-bid auction strategy that associates each SFC Request to each service provider domain, thus increasing the speed of finding a suitable placement plan in a multi-domain environment while preserving the autonomy of each domain

**Table 2** System model notations

Notation	Description
$G = (H, L)$	Undirected graph of the physical network
$H = \{h_1, h_2, \dots, h_n\}$	Set of hosts where the VNFs will be executed
$L = \{l_1, l_2, \dots, l_n\}$	Set of virtual links
$A = \{a_1, a_2, \dots, a_n\}$	The auctions
$U = \{u_1, u_2, \dots, u_n\}$	All the users that requests SFCs
$D = \{d_1, d_2, \dots, d_n\}$	The domains of the system
$D_h \in D$	The domains where node $h$ is associated
$R_h = \{c_h, m_h\}$	Set of resources of node $h$
$c_h$	CPU capacity of node $h$
$m_h$	Memory capacity of node $h$
$IM_h = \{vi_1, vi_2, \dots, vi_n\}$	Set of VNF images available at node $h$
$bw_{h_1, h_2}$	Bandwidth of link between $h_1$ and $h_2$
$dl_{h_1, h_2}$	Transmission delay of link between $h_1$ and $h_2$
$node_u$	Access node where user $u$ is associated
$nodes_d$	Set of nodes of domain $d$
$neighbor_d$	Neighbor domains of $d$
$cpu_f / mem_f$	The minimum amount of CPUs/memory required by VNF $f$
$decision\_domain_a$	Domain that coordinates auction $a$
$time_a$	Time when auction $a$ starts
$duration_a$	Duration of auction $a$
$bid\_request_a$	Set of bid requests of auction $a$
$sfc\_request\_vnfs$	Set of VNFs that compose the requested SFC
$ingress\_participant\_gw$	Gateway of the participant domain to where the decision domain will send packets of the SFC Request
$consumed\_time$	Time already consumed by the decision domain to transfer packets from the source of the SFC Request
$remain\_time$	Represents the time that the participant domain will have to execute the SFC
$bid\_responses_a$	Set of Bid Responses of auction $a$
$Value$	Bid value that the participant domain will charge to execute the SFC Requested
$egress\_participant\_gw$	Gateway to where the packets processed through the SFC Instance will be delivered
$sfc\_consumed\_time$	Total time spent to execute the requested SFC
$placement\_plan_{da}$	The placement plan created by domain $d$ to auction $i$
$ingress\_link$	Link from the SFC Request source node to the winner domain
$vnf\_plan$	The placement plan for each VNF of the SFC Requested
$vinst$	The VNF Instance

and is defined by the tuple  $\{value, egress\_participant\_gw, sfc\_consumed\_time\}$ , where the *value* is the bid value that the participant domain will charge to execute the requested

SFC, *egress\\_participant\\_gw* is the gateway where the response packets produced by the SFC Instance will be

**Fig. 1** Illustration of consumed and remaining times

delivered, the *sfc\_consumed\_time* is the total time spent to execute the requested SFC.

To compute the value of the bid response, each participant domain creates a placement plan. After the auction conclusion, only the plan of the winner's domain is executed. For each Bid Request, a different placement plan is created. The plan consists of an estimation of the resources required to execute the SFC Request by the respective domain infrastructure. The plan contains private information of the participant domain and is not sent to the decision domain. The plan is defined as  $placement\_plan_{da} = \{ingress\_link, vnf\_plan\}$ , where the *ingress\_link* defines the link between the *ingress\_participant\_gw* and the node where the first VNF Instance of the SFC Request will be executed, and  $vnf\_plan = \{vnf\_plan_1, vnf\_plan_2, \dots, vnf\_plan_n\}$  is a set that defines the placement of each VNF instance required to execute the SFC request. The attribute  $vnf\_plan_i = \{vinst_i, h_j, l_k\}$ , where  $vinst_i$  is the VNF Instance,  $h_j$  is the node where the VNF Instance is planned to be executed, and  $l_k$  is the link planned to be used to communicate with the next VNF Instance of the SFC.

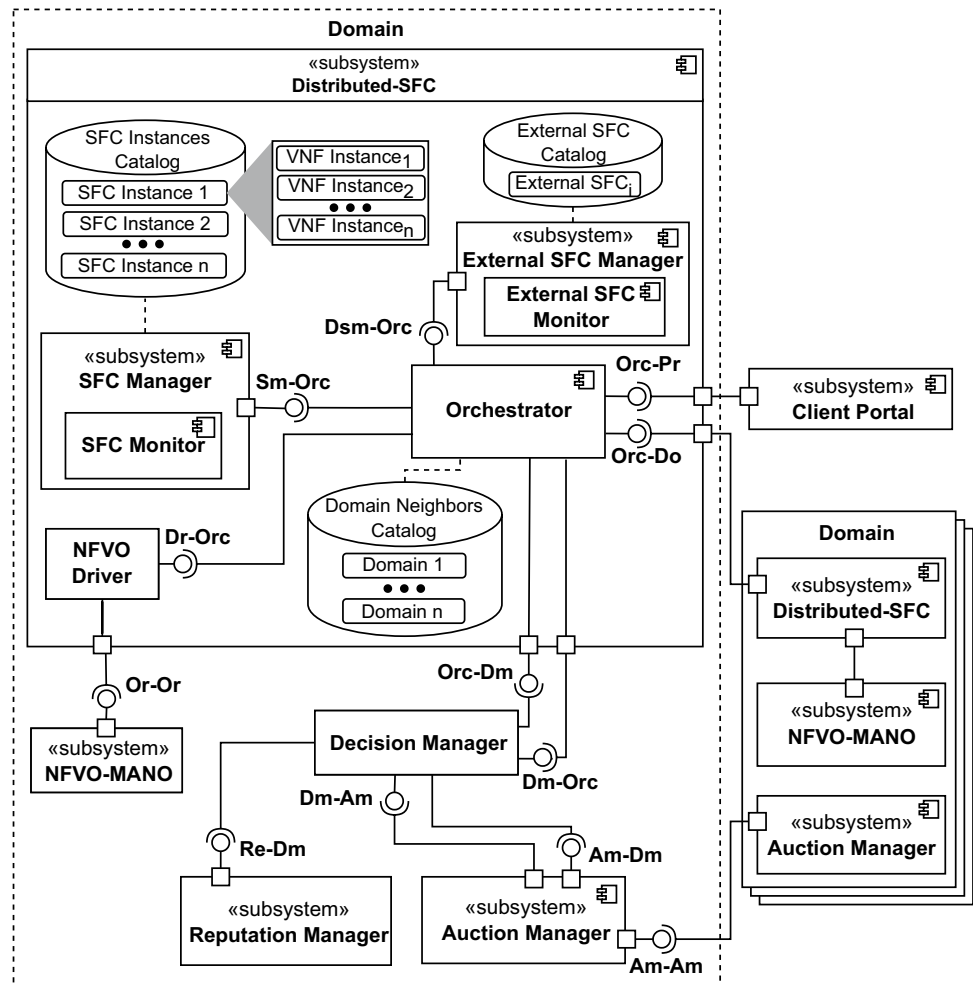
## Proposed Architecture for Distributed-SFC Placement

This section describes the proposed architecture for Distributed-SFC Placement encompassing a set of components and activities of the system operation. The architecture components represent the necessary entities that must be present in the environment to execute the SFC Placement in a distributed fashion. The system operation depicts how these entities will interact with each other to accomplish the SFC Placement. The proposed architecture is integrated to the components and interfaces specified of the ETSI reference architecture [4]. Figure 2 depicts the architectural components and their respective interfaces.

The environment in which our architecture is executed is composed of multiple domains. Such domains are independent, though they cooperate with each other to execute end-users' services' requests. The components and subsystems are described as follows.

The **Client Portal** component is used by end users to request the execution of SFCs. Each client portal is

**Fig. 2** Architecture components of our approach



connected to one domain through the Distributed-SFC subsystem. Our approach is independent on the specific technology used to implement the client portal. Therefore, this component can be implemented as a Web Portal, Mobile APP, or even dedicated software. Each SFC Request is individually managed by the environment.

In distributed systems, an orchestrator component is typically responsible for coordinating and managing the interactions and execution of various components or services within the system. Its main role is to ensure that different components work together to achieve the system's goals. Management and orchestration are key elements of the ETSI network functions virtualization (NFV) architecture. In the context of the ETSI architecture, NFV orchestrator (NFVO) is a key component of the NFV-MANO (network functions virtualization management and network orchestration) architectural framework [18], which helps standardize the functions of virtual networking to increase interoperability of software-defined networking (SDN) elements. The NFVO performs resource orchestration and network service orchestration, as well as other functions. The role of orchestration in edge computing systems is also crucial, and the ETSI MEC reference architecture [19] includes the MEC orchestrator component (MEO). MEC orchestration is the core functionality in MEC system-level management. The MEO is responsible for, among other functions, to keep an overall view of the MEC system based on the deployed edge hosts, available resources, available services, and topology; to perform the on-boarding of application packages, including checking their requirements and if necessary adjusting them to comply with operator policies, to select suitable edge nodes for application instantiation based on constraints, such as latency, available resources, and available services; and trigger application instantiation, termination, and relocation as needed when supported by the infrastructure.

In our proposed architecture, we consider an orchestrator component in charge of all the aforementioned responsibilities. Moreover, to implement the envisioned approach, the **Orchestrator** is also responsible for storing the SFC descriptors for the services that are available to be requested by the end users. It is in charge of coordinating the other components of the architecture and all the operations within the domain, such as handling new service requests received from users. The Orchestrator component also keeps the Domain Neighbors Catalog. This Catalog contains information about all the domains that are located within a hop away from the edge routers of the domain in question.

In the ETSI MANO [4], there are two catalogs. The NS Catalog stores all usable Network Service Descriptors (NSD) and the VNF Catalog store data about all usable VNF Descriptors (VNFD). In our proposed solution, we include a new catalog named **Domain Neighbors Catalog** that maintains information about all domains that are direct

neighbors of the domain in question, i.e., all domains that can be reached by one hop from its edge routers. The stored information includes the domain name, the edge routers through which the domain can be reached, and the respective estimated delays to such routers.

The **External SFC Catalog** stores the SFC entity that arrived at the current domain (decision domain) but was placed in another domain (Winner Domain) after the auction process. The decision domain keeps all necessary information from the original SFC Request of the user in the External SFC component, and the data consist of the VNFs list and each VNF Descriptor, the origin and destination of the packets, and the maximum tolerable delay. These data are necessary to enable end-to-end communication from the data source to the destination described in the SFC Request.

The **External SFC Manager** component is responsible for managing all External SFC components that are related to each user request. The **External SFC Monitor** is responsible for monitoring and collecting data about the External SFC of each SFC Request, similar to what the SFC Monitor does for the SFC Requests that are placed within the decision domain. These data can be used by the decision domain to compute the reputation of the other domains through the Reputation Manager component.

Our proposed architecture is agnostic regarding the platform used to execute the requested SFCs. The **NFVO-Driver** component is responsible for translating the commands from the Distributed-SFC component to the NFV-MANO executed in the domain. There is one NFVO-Driver for each type of NFV-MANO platform implementation. As each domain has its own NFV-MANO, this approach allows the interoperability of our proposal in heterogeneous environments. The NFVO-Driver installed in a domain must be compatible with the NFV-MANO configured in the domain.

The **SFC Manager** component is responsible for managing the SFC Instances that the decision domain Distributed-SFC component is responsible for. The decision domain will coordinate with other domains to execute the SFC Placement and request the inter-domain links. The **Decision Manager** component is responsible for receiving the SFC Request and invoking the appropriate component that is used to decide in which domain the requested SFC should be executed. Multiple approaches can be used in this phase; in our proposed approach, we will adopt a distributed auction strategy.

The **Auction Manager** component is responsible for coordinating the auction process that will define which domain will execute the requested SFC. The Auction Manager operation can be summarized as follows: For each requested SFC within a domain, the Auction Manager of that domain creates a new auction. The Auction Manager then sends auction requests to the Auction Manager component of participant domains, seeking bids from them. After receiving the auction response with the bid from the



participant domains, the Auction Manager, that creates the auction, will proceed to determine the auction winner based on the bids received. The auction strategy adopted in this proposal is fully discussed in Sect. “[Auction Components and Operation](#)”. The **Reputation Manager** component creates a domain reputation system. In this system, each domain will increase its reputation whenever the behavior is trustful (after winning an auction and successfully executing the VNFs, for example), and the reputation will decrease otherwise (after winning an auction the domain refuses to execute the VNFs). In Sect. “[Auction Components and Operation](#)”, we briefly introduce the concept of reputation and how it influences the auction. Nevertheless, many approaches can be adopted to create a distributed reputation system, but the implementation details of such components are out of the scope of this proposal.

Finally, the **SFC Monitor** component will collect data about the VNF Instances that compose an SFC. These data can be used by components like auto-scaling and migration to deal with increasing or decreasing workload in the SFC.

The **Or-Or** interface is defined by ETSI in the specification named ETSI GS NFV-IFA 030 [18]. This interface provides the functionality required to enable multiple OSM components to exchange information and manage Network Services hosted in another OSM infrastructure. This interface mainly supports the Network Service Descriptor Management interface, the Network Service Lifecycle Management interface, and the NS Performance Management interface.

The **Am-Am** interface enables the Auction Manager components to exchange auction data. It enables the auction participants to receive the information related to an ongoing auction—sent by the auctioneer—and allows bids to be placed.

The **Orc-Do** interface enables Orchestrator components of different domains to exchange data about the current network state and compute resources. It also allows domains to indicate their intent of participating in auctions.

## Auction Components and Operation

In this section, we present the auction components and the main operations of the system to execute the Distributed-SFC Placement by employing the Distributed Auction strategy.

We propose an auction-based strategy to solve the SFC Placement problem over multiple domains. In our scenario, the domains compete against each other to win the auction, which enables the winner domain to execute the requested SFC and receive the payment for leasing the network and computing resources. We consider the blind auction strategy to associate an SFC Request to a service provider domain. Another name for this type of auction

is first-price sealed-bid auction (FPSBA) [20]. In a blind auction, all participants simultaneously submit sealed bids, where a participant only knows the value of their own bid. The participant with the highest or the lowest bid value (depending on the auction type) wins the auction. The advantages of the adoption of the blind auction in the presented environment reside in the fact that the auctioneer has a priori knowledge about how long the auction will last, and the sellers will compute the bid only once.

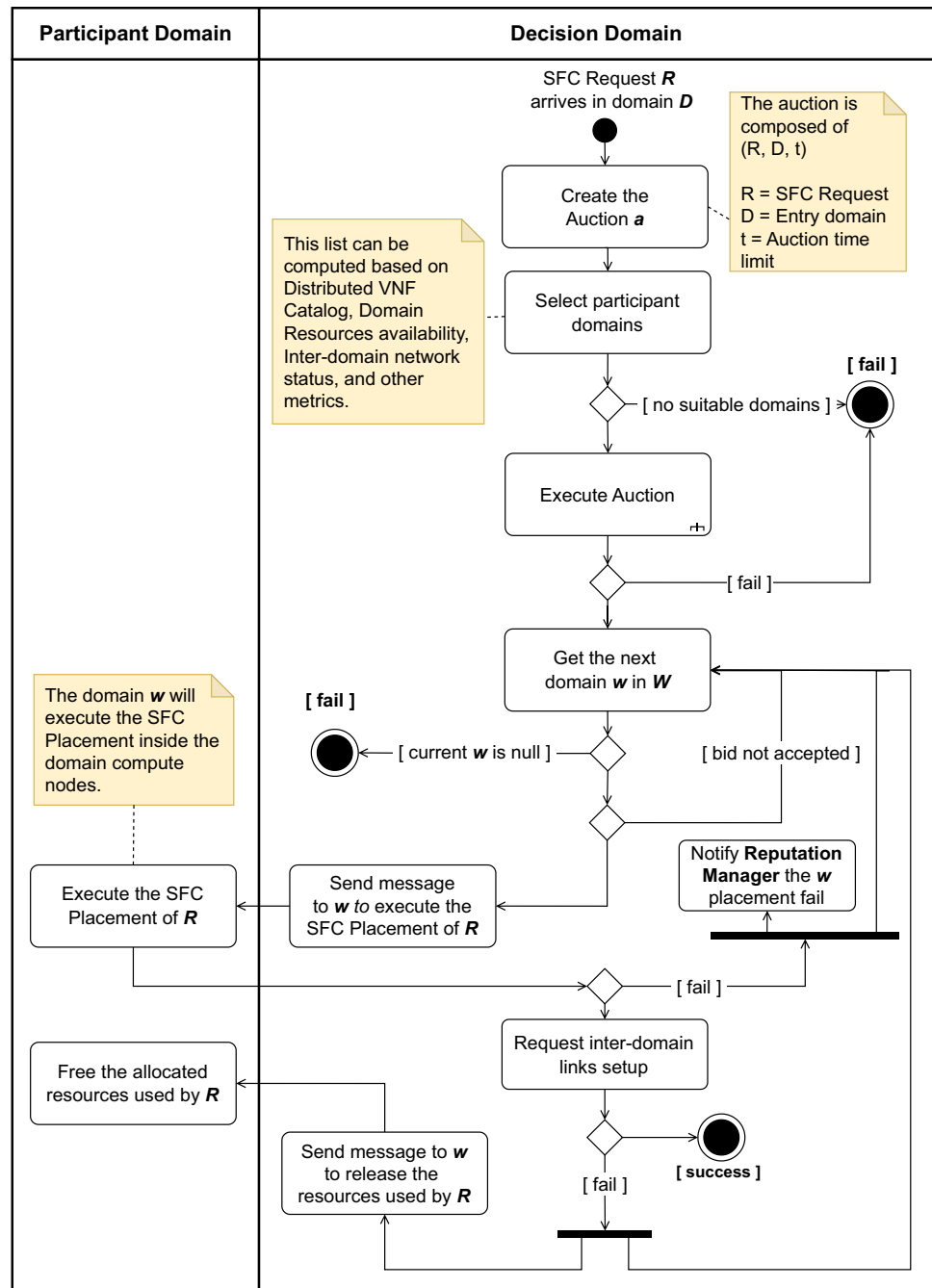
In a typical auction process, there are three stakeholders: (i) the sellers, (ii) the buyers, and (iii) the auctioneer, besides an asset to sell. In the Distributed-SFC Placement problem, the auctioneer is the decision domain. The sellers are the domains that want to sell their computational and network resources, which will be used to run the VNFs composing the requested services. The buyer is the decision domain that will pay for the other domains to execute the requested SFC so as to provide services for the end user. The decision domain can also play the seller role when it sells services for end users. The assets that are commercialized are SFCs, which will be entirely placed within the infrastructure of the winner domain. The auction is considered successfully finished if the SFC Request gets at least one bid. If no bids are received, then the SFC Request fails. At the end of the auction, the decision domain is capable of deciding the path through the winning domain from the gateway of the end user (src) to the destination (dst).

In our approach, we divided the operations into two logical phases: (i) the Distributed-SFC Placement phase to cope with the coordination process, and (ii) the Distributed Auction phase to encompass the auction process.

The Distributed-SFC Placement coordination phase is responsible for receiving the SFC Request, coordinating among the other domains to decide which one will be responsible for executing the VNFs of the requested service, the intra-domain links to connect the VNFs and, finally, creating the inter-domain links. Figure 3 depicts all the aforementioned activities.

As depicted in Fig. 4, the end user requests the creation of the SFC via the Client Portal, which consequently sets an SFC Request and delivers it to the Orchestrator component of the decision domain (the domain that the user is associated with). In our blind auction strategy, the decision domain creates the auction and waits for bids during a fixed time period.

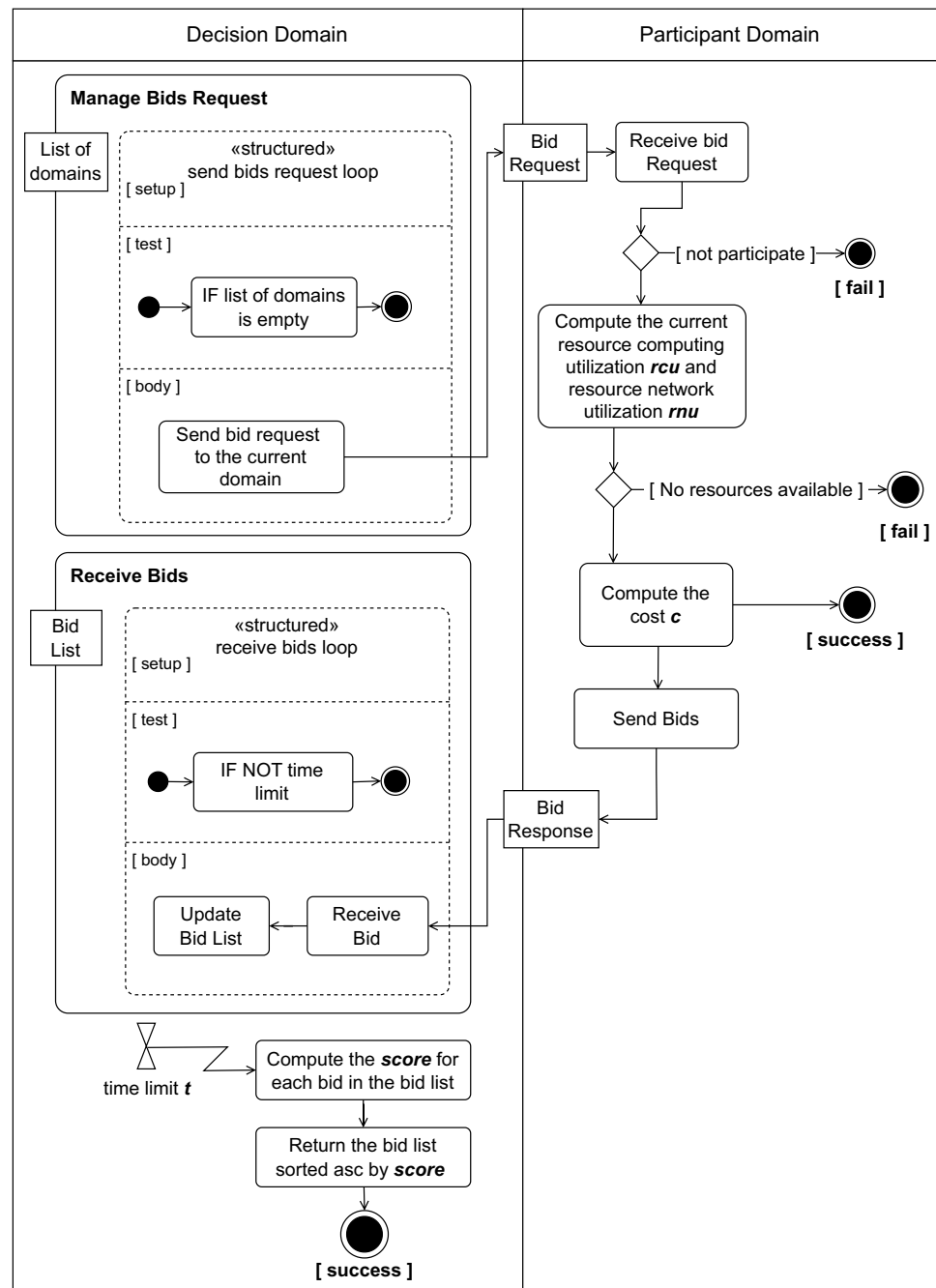
The orchestrator component verifies which domain can execute the requested SFC and creates a list of domains that are allowed to participate in the auction, called participant domains. The decision domain can discard domains for several reasons, such as (i) domains that do not have the VNFs required by the SFC request, (ii) domains that have a low reputation, (iii) domains that suffer from security issues, etc.

**Fig. 3** Distributed-SFC placement coordination heuristic

With the list of participant domains, the Distributed Auction phase starts, with the Orchestrator component asking the Auction Manager component to execute the auction process, aiming to select the winner domain. The Auction Manager of the decision domain asks for a Bid Response from each Auction Manager of the participant domains through a Bid Request. Each participant's domain decides whether they will participate or not in the auction. If it decides to participate, it will compute the respective cost to execute the SFC Request and send it to the decision domain as a single sealed bid. The calculations of the

cost of a participant domain are discussed in Sect. “**Cost Calculations**”.

When the time limit of the auction duration is reached, the decision domain computes the respective scores of the submitted bids. The score metric is computed using the bid value and the reputation of each participant domain. Domains that give the lowest bids and have a higher reputation will have more chances to be ranked first. Conversely, even if a domain submits the lowest bid value, if it presents a bad reputation, its chances of being ranked first decrease. Then, the list of bids is sorted from the lowest score (the

**Fig. 4** Distributed auction heuristic

best) to the highest score (the worst). The first domain in the list offers the best score and it is the selected winner domain, which wins the chance to execute the SFC Placement algorithm. If the SFC Placement process fails, the next domain in the list is selected. This process continues until the SFC is successfully placed, or the list of selected domains becomes empty, thus indicating that the SFC Placement failed, and the request could not be met. A participant domain will only be notified about the auction result when it is selected as the winner domain. If the participant domain chose to reserve resources to satisfy the auction demand, the participant

domain must adopt some strategy to release the resources based on a time interval or other metric.

As soon as the selected winner domain successfully executes the placement, the decision domain coordinates the allocation of the inter-domain's links from the source to the destination domain, passing through the domain that successfully executed the SFC Placement. If the link allocation fails, the domain that executes the SFC will deallocate the used resources, and the next domain in the participant's list will have the chance to execute the SFC Placement. Otherwise, the respective external SFC component is created to

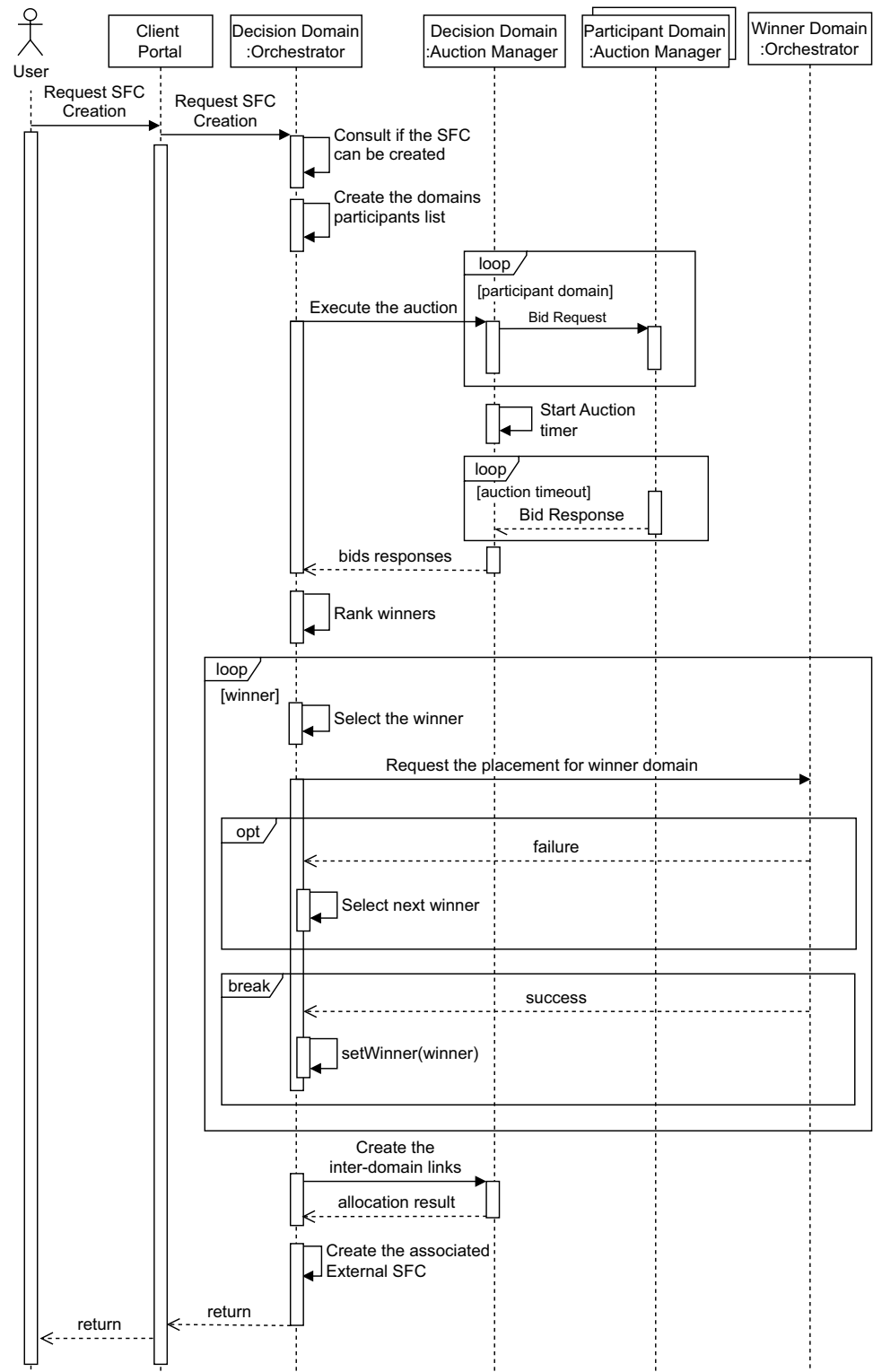
be responsible for monitoring and delivering the data flow to the SFC Instance externally placed.

Figure 5 depicts the sequence of interactions between the components involved during the life-cycle of a user request, i.e., an SFC Request.

### Cost Calculations

A participant domain defines a cost that represents how much money it will charge to run the requested SFC. The price is computed based on the resource usage and if the

**Fig. 5** Sequence diagram of a user request (SFC Request)



requested SFC can share resources with already deployed VNFs. The following variables are used in the cost calculation. The  $rcu$ ,  $0 < rcu \leq 1$ , defines the percentage of the computational resource utilization in the domain. The  $rnu$ ,  $0 < rnu \leq 1$ , defines the percentage of the network resource utilization in the domain.

The  $SFC_{CCost}$  (Eq. 1) is computed based on the required resources in the SFC Request and the computation cost specified by the domain. The  $SFC_{NCost}$  (Eq. 2) is computed based on the required network resources in the SFC Request and the network cost specified by the domain

$$SFC_{CCost} = CPU_{Required} * CPU_{Cost} + MEM_{Required} * MEM_{Cost} \quad (1)$$

$$SFC_{NCost} = Link_{Required} * Link_{Cost} \quad (2)$$

The discount  $\delta$  is computed based on domain load and the number of VNFs that can be shared with the already deployed ones. The service provider can define a max percentage value for the discount  $0 \leq \delta_{max} \leq 100$ . The discount is computed according to Eq. (3)

$$\delta = (SFC_{CCost} * rcu) * \min(\delta_{max}, nVNF_{PS} * \delta_{percent}). \quad (3)$$

The final bid cost, as shown in Eq. (4), defines the value that the domain will ask to execute the requested SFC. Using this approach, the domain will apply a discount to the price if it has sufficient free resources or if it can share VNF Instances; otherwise, it will weigh the price by increasing it as it has fewer resources available to execute new VNF Instances

$$c = (SFC_{CCost} * rcu + SFC_{NCost} * rnu) - \delta. \quad (4)$$

To find the winner, the auctioneer will compute the score based on the reputation and the bid of each seller. The domain with the lowest score wins the auction. In the case of a tie with more than one bid with the same score, the domain that submitted the bid first is the winner. Persisting the tie, the winner domain is randomly selected. Hence, the score of each domain  $d_n$  that submitted the bid  $b_n$  for the auction is calculated according to Eq. (5), where the  $InterDomainLinkCost(src, dn, dst)$  is the cost of all links that composes the network path from the source  $src$  to the destination  $dst$  passing through domain  $dn$

$$\begin{aligned} S(dn, a) \\ = bn * (1/R(d_n)) + InterDomainLinkCost(src, d_n, dst). \end{aligned} \quad (5)$$

The reputation of a domain,  $Reputation(d_i, d_n)$ , is a number defined within the interval  $(0, 1]$ . The reputation is a metric that translates which level of confidence of domain  $d_n$  is observed by domain  $d_i$ . A reputation with a value near 0 (zero) means that the domain is not reliable. Conversely, the maximum reputation with value 1 (one) means that the

domain is totally reliable. A domain  $d_i$  uses a global reputation value related to  $d_n$  to weigh the respective bid value of  $d_n$ . The higher the reputation is, the lower the weight of the bid and, consequently, the lower the score (more chances to win the auction with a high revenue). Hence, a domain that presents a low reputation against the other domains will need to decrease its bid to be more competitive. The reputation of each domain is updated at the end of the execution of the placement algorithm by the winning domain of the respective auction. If the placement fails, the reputation of such domain decreases; otherwise, it increases. The reputation information is managed by the Reputation Manager component and can be calculated based on any available reputation algorithm.

## Evaluation

This section presents the performed experiments to evaluate the proposed auction-based SFC placement algorithm. First, we describe the experimental plan, then the configurations, scenarios, and simulation tools used in the experiments, and finally the experiment results.

## Experimental Plan

We adopted the Goal Question Metric (GQM) [21] to plan the experiments and to guide the evaluation of the approaches. GQM model is a hierarchical (three-level) structure that, at each level, refines the granularity of what is relevant to provide reliable insights on a phenomenon. A goal represents which phenomenon should be analyzed, and each goal can be represented as one or more Questions. The Questions characterize the object of measurement regarding viewpoints. A set of Metrics is associated with every question. In turn, Metrics represent the quantitative level of GQM and help provide a quantitative answer. Table 3 presents the goal used for evaluating the approaches.

Table 4 presents the related Questions for the defined Goals, and Table 5 presents the corresponding Metrics used to answer such questions.

GQM defines a measurement model that structures the experiment evaluation. The conceptual level (Goal), operational level (Questions), and quantitative level (Metrics), described above, were defined considering the auction-based approach as the object of study, and evaluation as the purpose of the experiment.

## Simulation Setup

For the auction experiment, we chose the 5G Best Effort scenario to model the edge-cloud computing environment, entities, nodes, and links to run the experiments. The



**Table 3** Defined goals

Goal	Description
G1	Analyze our decentralized auction-based algorithm, executed in a multi-domain environment, for the purpose of evaluating its performance in terms of the profit for each service provider during the allocation of the requested services in the respective domain that the service provider is responsible for

**Table 4** Defined questions

Question	Description	Goal
Q1	Does the proposed auction-based approach reduce the total service execution cost in contrast with always executing the service in the decision domain?	G1

**Table 5** Defined metrics

Metric	Description	Question
Service Provider Profit	Denotes the difference between the cost of executing the service requested in the decision domain (on the responsibility of a service provider) and the cost paid by the decision domain to outsource the execution to a domain of another service provider	Q1

**Table 6** Parameters of the simulated scenario for the auction evaluation based on [6, 7]

Entity	Parameters	Value
Simulation	Auction duration	100 ms
	Number of domains	10
	Executions per scenario	10
Domain	Number of nodes	5 or 10
	Number of gateways	1
Links	Bandwidth (Gbps)	10
	Delay (ms)	1.13
Nodes	CPU capacity	10,000
	Mem capacity	30,000
SFCs	Max Delay (ms)	30
	Number of VNFs in the SFC	3
VNFs	CPU demand	300
	Mem demand	300

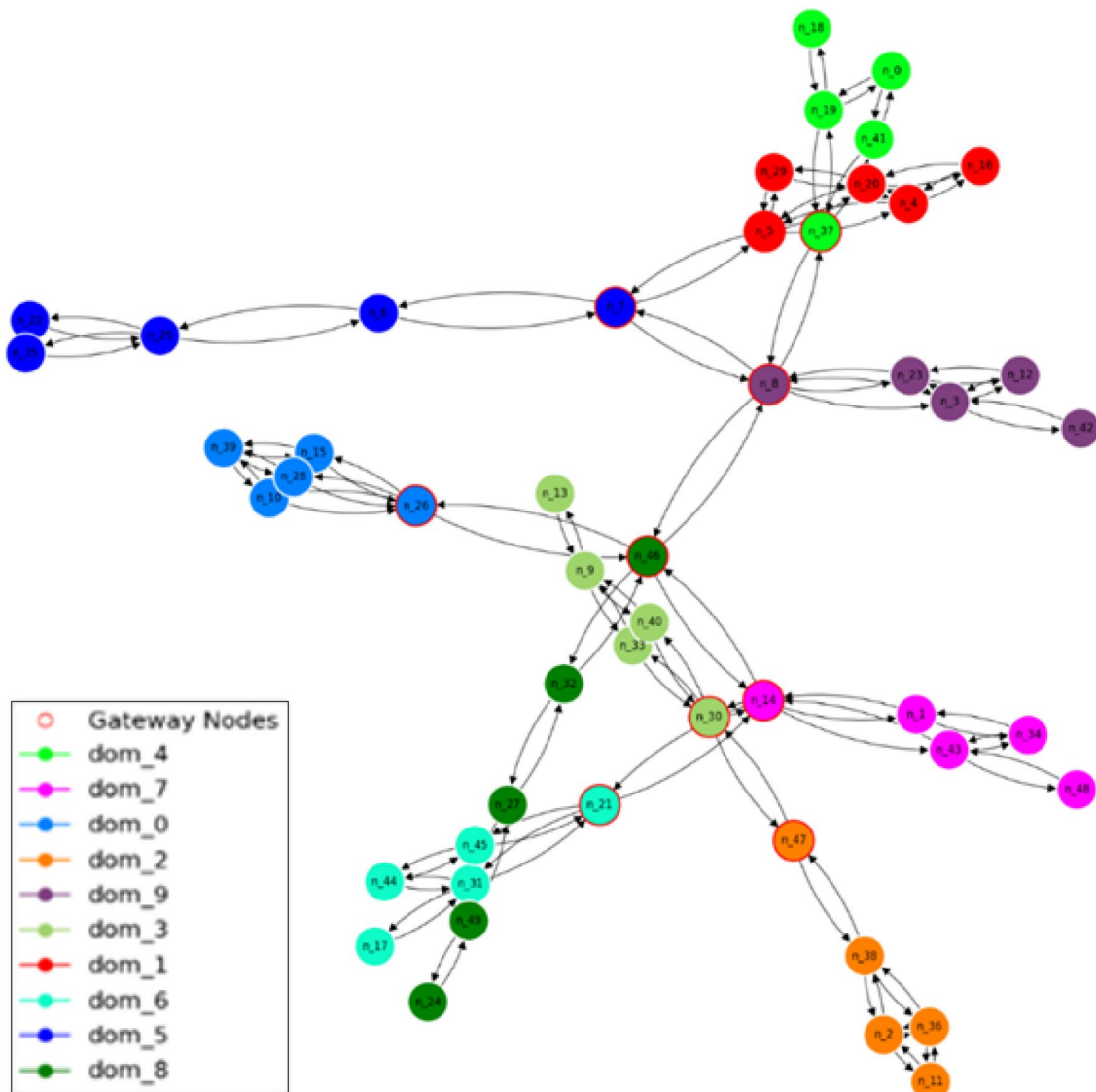
requirements in such a scenario are more flexible and less restrictive than other scenarios, such as ultra-reliable low-latency communication (URLLC) and enhanced Mobile Broadband (eMBB). The chosen scenario represents mixed applications with a variety of entities and attributes. Table 6 presents the parameters defined to create the edge computing environment. The simulation parameters used in our experiments are aligned with what is considered to be suitable best-effort parameters by other works in the literature [6, 7]. In [6], the authors modeled the application traffic based on the 3GPP traffic models. Regarding the topology size,

we assumed a small-scale infrastructure per domain—with a maximum number of 10 nodes per domain. This configuration is similar to the small-scale substrate network presented in [10], in which each domain has up to dozens of nodes. The authors in [10] mentioned that the characteristics of their network could reflect that of an operator such as Orange Lab (<https://5glab.orange.com/en/orange-5g-the-labs/>). Different types of data centers are considered [10], comprising nodes from the network core, cloud, and edge. In the edge, they assume the presence of 15 data centers, each one having 10 servers. Based on this reference, our objective was to force a scenario in which there was not an abundance of resources within each domain, to stimulate and take more advantage of an outsourcing strategy.

Figure 6 shows an example of a multi-domain environment topology executed during the experiments. The nodes with a red line are the border gateways that are responsible for the interconnection between multiple domains.

## Simulation Tool

We create a simulation tool tailored to NFV-Edge scenarios. The core of the simulation relies on an extension of the SimPy simulator. SimPy is a Python library that implements a framework for event-driven simulations. In the simulator, active elements, such as Nodes, Links, VNF Instances, and VNF Requests, are modeled as processes. Such processes are Python generators responsible for creating events (for example, a packet arriving in the VNF queue) and, regarding a timeout value, yielding them for processing.



**Fig. 6** Example of the multi-domain environment

When an event is yielded, the process that yields such an event is suspended and waits for the processing of the event to be finished. For example, the execution of an auction will create an event yield, and the generator waits for the bid responses to arrive from the participant domains before the selection of the winner domain.

The required time for processing all the events is stored in files. For example, the time required for bid responses to arrive at the decision domain, and the auction logs are stored in separate files. Those data files can be further processed and analyzed to generate insights about the auction-based approaches.

## Results

The experimentation plan was based on the GQM model. The main goals of the performed experiments were: (i) to analyze the proposed auction-based approach to evaluate their performance—in terms of service provider's profit in the context of a multi-domain environment; and (ii) to assess the performance of the proposed VNF migration procedure in reducing SLA violations of the services executing in a single domain. The analysis of the total cost for execution of the services requested is directly related to the system's ability to meet the service provider's profit. Also, the analysis

of the SLA violations is related to the ability of maintaining the agreed service delay even under user mobility scenarios.

The following results are related to the question Q1: *Does the proposed auction-based approach reduce the total service execution cost in contrast with always executing the service in the decision domain?* One of the most important goals of our auction-based mechanism is selecting the best domain to execute the service requested by the user with the lowest cost possible, without violating the defined Service-Level Agreement (SLA). Thus, the aim of this question is to compare the proposed approach based on auction and another approach that does not use any auction process regarding the total cost for executing the requested service in a multi-domain environment.

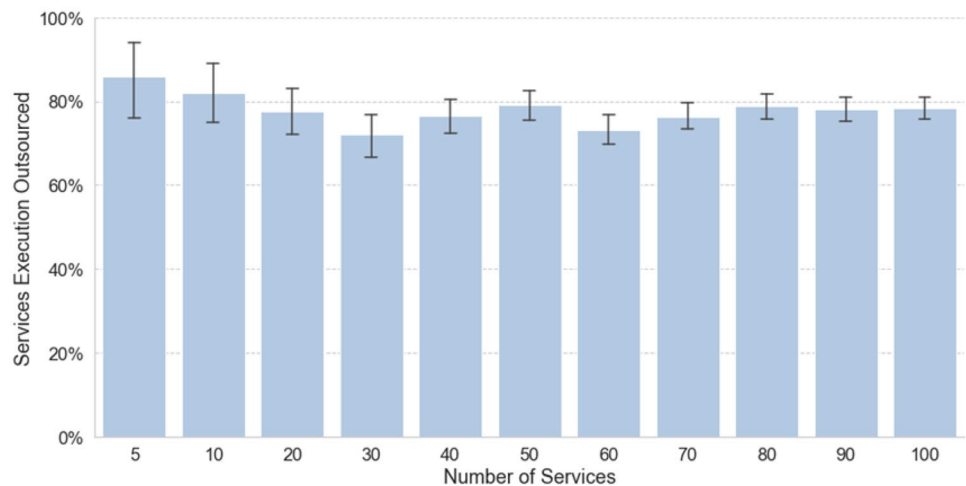
In Figs. 7 and 8, the x-axis represents the number of requested services. It is important to mention that, in our approach, for each requested service, one auction will be executed. Figure 7 shows that 80% of the services were

outsourced. When the auction was not enabled, all the services were executed in the decision domain (denoting the domain that receives the service request—the entry point of the request).

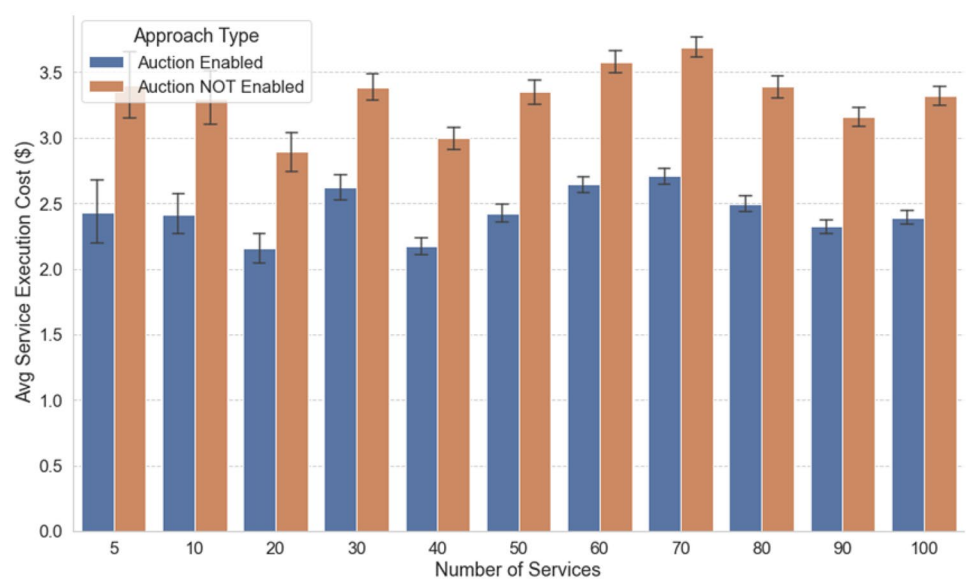
Figure 8 compares the Service Provider Profit Metric in two different approaches (with the auction and without auction) to select the winner domain. As expected, the “Auction” approach achieved the best result regarding the lower average cost for the service execution. The y-axis represents the average cost to execute the requested service. This behavior shows that adopting an auction-based mechanism is a possible approach to reducing the total cost of the service execution in a multi-domain environment. The auction-based approach increases the service provider profit by at least 18% in the tested scenarios.

We also performed one auxiliary experiment to identify if the number of services placed increased by adopting the auction strategy. In this experiment, we reduced the number

**Fig. 7** Number of services outsourced by the decision domain



**Fig. 8** Comparison of Average Service Cost in different approaches



of resources to force some nodes to run out of resources over time. Figure 9 shows that, when the auction is enabled, the number of services placed increases in comparison to the approach where all the services must be executed in the decision domain (auction disabled).

As expected, according to Fig. 10, the total service providers' profit increases as the number of services executed increases. It means that our approach can find suitable domains to execute the services, even under the increase of services to be allocated.

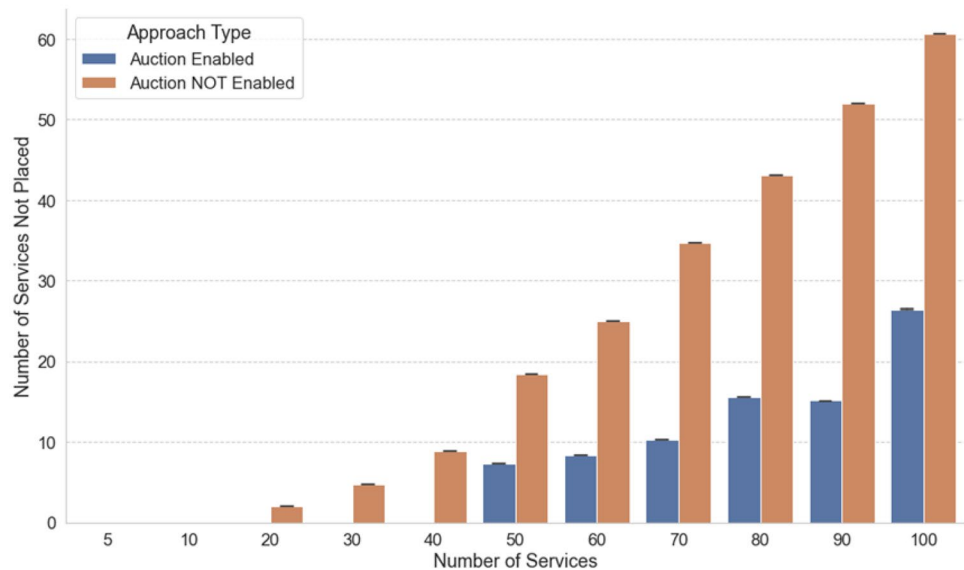
Figure 11 illustrates the execution of services in various scenarios with different numbers of domains. The domains are heterogeneous regarding the size of the infrastructure (number of nodes and links) and the cost of the infrastructure. In this experiment, we kept the number of requested

services fixed (50 services) while varying the number of domains.

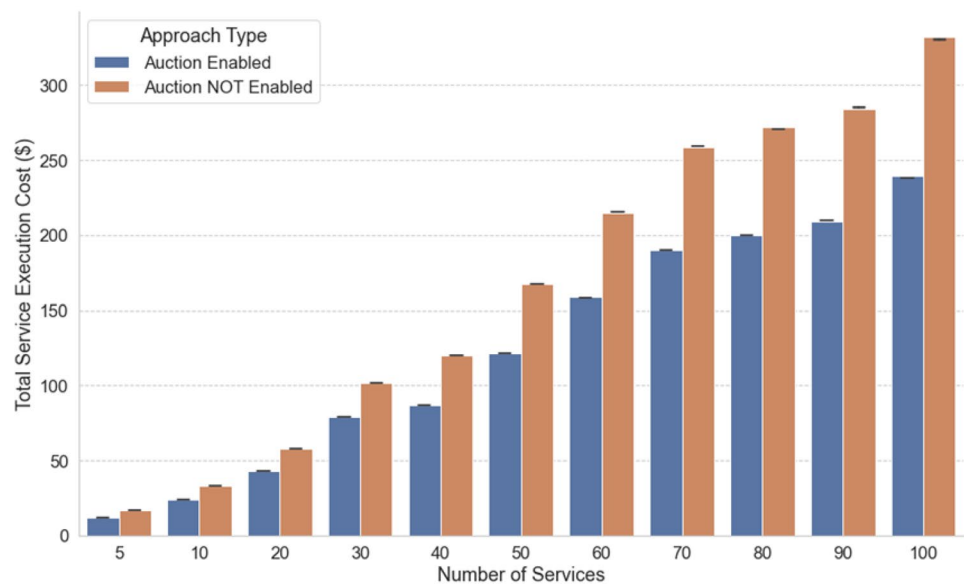
We found that the number of available domains where the SFC can be executed has a positive impact on the total cost paid to execute the SFC. Figure 11a shows that scenarios with a higher number of domains lead to a decrease in the total cost of execution, in contrast with scenarios with fewer domains. Thus, the decision domain must create partnerships to offer competitive prices for its users.

Figure 11b depicts that, with a greater number of domains, the execution of the requested services can be outsourced more frequently to domains with cheaper nodes. Thus, we can infer that the auction strategy proposed in this paper can identify more suitable domains to

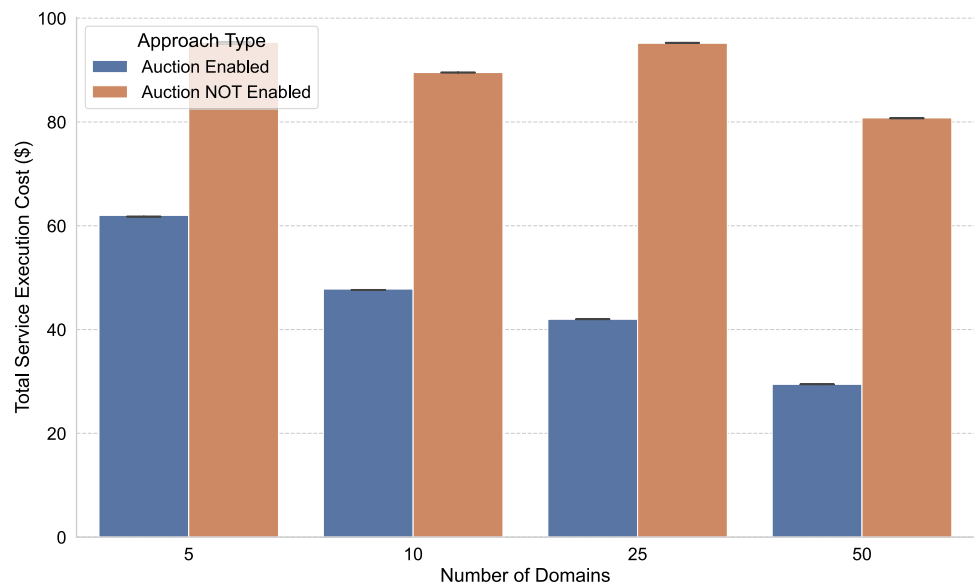
**Fig. 9** Comparison of the number of services placed



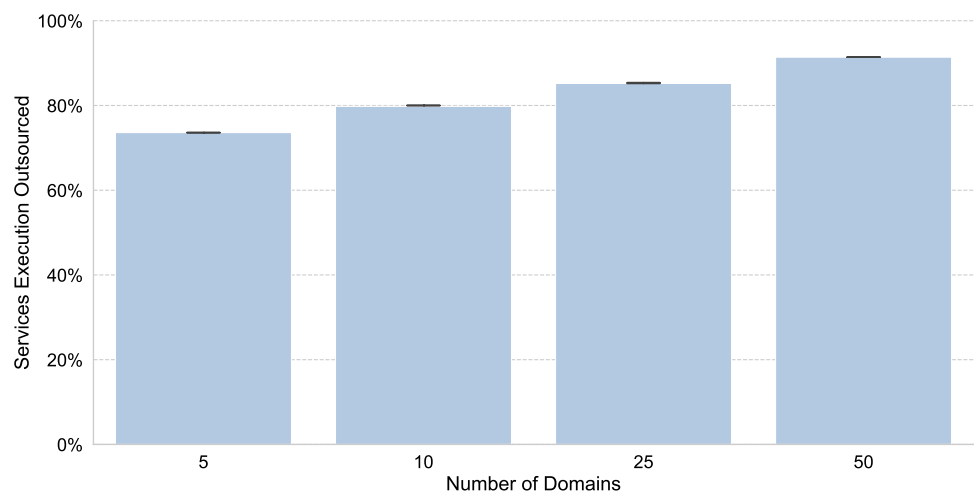
**Fig. 10** Comparison of Total Service Cost in different approaches



**Fig. 11** Comparison of Total Service Cost with different number of domains



(a) Total Service Cost.



(b) Outsourced Services.

execute the requested service, resulting in a reduction in the total execution cost.

## Conclusion

The main contribution of this work is a solution for SFC placement that addresses the challenges posed by large-scale, distributed, and multi-domain environments. The main objective pursued by the proposal was to provide a resource allocation solution that takes into account highly distributed large-scale environments that involve multiple administrative domains. We believe that such environments will be typical in emerging edge-cloud scenarios.

Our decentralized auction-based solution meets these requirements. In it, relevant information about provider resource availability is used to make a market-based decision, while sensitive information for each provider is kept private. An auction is created for each service request, and the distribution of the place where the auction will be running distributes the load of its realization, helping to deal with the high demands of requests. The fact that the auction has a defined duration time avoids waiting too long for an allocation decision in case of an increasing number of auction participants. These factors contribute to the overall scalability of the solution. However, the fact that we use an auction-based approach does not directly address environmental fluctuations. Dealing with such fluctuations



is addressed by the fact that we do not assume a static resource allocation (and consequently placement of SFCs/VNFs) solution. Instead, we consider that the service placement plan within each administrative domain will be redone from time to time, to deal with the dynamism of the environment in terms of incoming requests and availability of resources. However, this dynamic placement within each domain is beyond the scope of our solution. We addressed this problem in other works by our research team, for example, in [22] where we describe our proposed online placement approach to handle continuously arriving service requests in NFV.

We highlight the following novel aspects of our work: (i) Handling SFC placement in multiple administrative domains: our solution considers the placement of SFCs in a decentralized manner, through the interaction of various possible domains, and it includes a proposal for a business model to be followed by the participating entities. (ii) Decentralized orchestration: our architecture encompasses all the components to enable multiple domains to participate in all necessary steps for resource allocation, placement, and execution of service chains on edge nodes, in a decentralized way. (iii) Auction-based method for SFC Placement: adopting an auction-based approach provides more economical efficiency for suppliers according to customer demands, flexible allocation of SFCs, and finer targeting of customers.

We adopted the auction model to take advantage of its features in the multi-domain environment considered. Providers in each domain are interested in providing their resources for the placement of SFCs. There are several auction models, many of them based on the existence of a centralized broker. Considering the large scale and distribution of the considered scenario, we propose a decentralized auction, where the entity responsible for initiating the auction varies, being the domain that received a given SFC request. As future work, we highlight investigating the incorporation of mechanisms for managing the reputation of the nodes participating in the auction.

**Acknowledgements** This work was partially funded by DELL EMC, CAPES, CNPq, FAPERJ, and FAPESP. Professors Débora Muchaluat-Saade, Flavia Delicato, and Paulo Pires are CNPq Fellows. Professors Débora Muchaluat-Saade and Flavia Delicato are also FAPERJ Fellows.

**Funding** This study was partially funded by DELL EMC (ID FEC 4717), FAPESP (Grant No. 2015/24144-7), and FAPERJ (Grant No. 200.972/2022).

## Declarations

**Conflict of Interest** The authors declare that they have no conflict of interest.

## References


1. Yang S, Li F, Trajanovski S, Yahyapour R, Fu X. Recent advances of resource allocation in network function virtualization. *IEEE Trans Parallel Distrib Syst*. 2020;32(2):295–314. <https://doi.org/10.1109/TPDS.2020.3017001>.
2. Benkacem I, Taleb T, Bagaa M, Flinck H. Optimal VNFs placement in CDN slicing over multi-cloud environment. *IEEE J Sel Areas Commun*. 2018;36(3):616–27. <https://doi.org/10.1109/JSAC.2018.2815441>.
3. Reyhanian N, Farmanbar H, Mohajer S, Luo Z-Q. Joint resource allocation and routing for service function chaining with in-subnetwork processing. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4990–4994 (2020). <https://doi.org/10.1109/ICASSP40776.2020.9054706>. IEEE
4. ETSI: ETSI Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Architectural Framework Specification; 2021. [https://www.etsi.org/deliver/etsi\\_gs/nfv/001\\_099/006/02\\_01\\_01\\_60/gs\\_nfv006v020101p.pdf](https://www.etsi.org/deliver/etsi_gs/nfv/001_099/006/02_01_01_60/gs_nfv006v020101p.pdf)
5. Zhao D, Liao D, Sun G, Xu S. Towards resource-efficient service function chain deployment in cloud-fog computing. *IEEE Access*. 2018;6:66754–66. <https://doi.org/10.1109/ACCESS.2018.2875124>
6. Mountaser G, Condoluci M, Mahmoodi T, Dohler M, Mings I. Cloud-ran in support of urllc. 2017;1–6. <https://doi.org/10.1109/GLOCOMW.2017.8269135>
7. Alves Esteves JJ, Boubendir A, Guillemin F, Sens P. Heuristic for edge-enabled network slicing optimization using the “power of two choices”, pp. 1–9 (2020). <https://doi.org/10.23919/CNSM50824.2020.9269099>
8. Abu-Lebdeh M, Naboulsi D, Glietho R, Tchouati CW. NFV orchestrator placement for geo-distributed systems. In: *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, pp. 1–5 (2017). <https://doi.org/10.1109/NCA.2017.8171391>. IEEE
9. Franco MF, Scheid EJ, Granville LZ, Stiller B. BRAIN: Blockchain-based reverse auction for infrastructure supply in virtual network functions-as-a-service. In: *2019 IFIP Networking Conference (IFIP Networking)*, pp. 1–9 (2019). <https://doi.org/10.23919/IFIPNetworking.2019.8816843>. IEEE
10. Sun G, Li Y, Liao D, Chang V. Service function chain orchestration across multiple domains: a full mesh aggregation approach. *IEEE Trans Netw Serv Manage*. 2018;15(3):1175–91. <https://doi.org/10.1109/TNSM.2018.2861717>.
11. Gao C, Cankaya HC, Jue JP. Survivable inter-domain routing based on topology aggregation with intra-domain disjointness information in multi-domain optical networks. *J Opt Commun Netw*. 2014;6(7):619–28. <https://doi.org/10.1364/JOCN.6.000619>.
12. Hamzaoui I, Duthil B, Courboulay V, Medromi H. A survey on the current challenges of energy-efficient cloud resources management. *SN Comput. Sci*. 2020;1(2). <https://doi.org/10.1007/s42979-020-0078-9>
13. Sun G, Li Y, Yu H, Vasilakos A.V, Du X, Guizani M. Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. *Future Gener Comput Syst*. 2019;91:347–60. <https://doi.org/10.1016/j.future.2018.09.037>
14. Liu Y, Zhang H, Chang D, Hu H. GDM: a general distributed method for cross-domain service function chain embedding. *IEEE Trans Netw Serv Manage*. 2020;17(3):1446–59. <https://doi.org/10.1109/TNSM.2020.2993364>.
15. Zavlanos M.M, Spesivtsev L, Pappas GJ. A distributed auction algorithm for the assignment problem. In: *2008 47th IEEE Conference on Decision and Control*, pp. 1212–1217 (2008). <https://doi.org/10.1109/CDC.2008.4739098>

16. Borjigin W, Ota K, Dong M. In broker we trust: a double-auction approach for resource allocation in NFV markets. *IEEE Trans Netw Serv Manag*. 2018;15(4):1322–33. <https://doi.org/10.1109/TNSM.2018.2882535>
17. Avasalcai C, Tsigkanos C, Dustdar S. Decentralized resource auctioning for latency-sensitive edge computing. In: 2019 IEEE International Conference on Edge Computing (EDGE), pp. 72–76 (2019). <https://doi.org/10.1109/EDGE.2019.00027>
18. ETSI: Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Multiple Administrative Domain Aspect Interfaces Specification (2018). [https://www.etsi.org/deliver/etsi\\_gs/nfv-ifa/001\\_099/030/03.01.01\\_60/gs\\_nfv-ifa030v030101p.pdf](https://www.etsi.org/deliver/etsi_gs/nfv-ifa/001_099/030/03.01.01_60/gs_nfv-ifa030v030101p.pdf)
19. ETSI: MEC 003—V3.1.1—Multi-access Edge Computing (MEC); Framework and Reference Architecture. Technical report, ETSI, Valbonne/França (2022). [https://www.etsi.org/deliver/etsi\\_gs/MEC/001\\_099/003/03.01.01\\_60/gs\\_MEC003v030101p.pdf](https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/03.01.01_60/gs_MEC003v030101p.pdf)
20. Zheng S, McAven L, Mu Y. First price sealed bid auction without auctioneers. In: Proceedings of the 2007 International Conference on Wireless Communications and Mobile Computing, pp. 127–131 (2007)
21. Caldiera VRBG, Rombach HD. The goal question metric approach. *Encyclop Softw Eng*, pp. 528–532 (1994)
22. Vieira JL, Battisti ALE, Macedo ELC, Pires PF, Muchaluat-Saade DC, Delicato FC, Oliveira ACB. Dynamic and mobility-aware vnf placement in 5g-edge computing environments. In: 2023 IEEE 9th International Conference on Network Softwarization (NetSoft), pp. 53–61 (2023). <https://doi.org/10.1109/NetSoft57336.2023.10175437>

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

## Authors and Affiliations

Evandro L. C. Macedo<sup>2</sup> · Anselmo L. E. Battisti<sup>1</sup> · Juan Lucas Vieira<sup>1</sup> · Julia Noce<sup>3</sup> · Paulo F. Pires<sup>1,3</sup> · Débora C. Muchaluat-Saade<sup>1</sup> · Ana C. B. Oliveira<sup>3</sup> · Flavia C. Delicato<sup>1</sup> 

✉ Flavia C. Delicato  
fdelicato@gmail.com

Evandro L. C. Macedo  
evandro@ravel.ufrj.br

Anselmo L. E. Battisti  
anselmo@midiacon.uff.br

Juan Lucas Vieira  
juanlucasvieira@id.uff.br

Julia Noce  
julia.noce@dell.com

Paulo F. Pires  
paulo.pires@dell.com

Débora C. Muchaluat-Saade  
debora@midiacon.uff.br

Ana C. B. Oliveira  
ana.oliveira@dell.com

<sup>1</sup> MídiaCom Laboratory, Universidade Federal Fluminense, Niterói, Brazil

<sup>2</sup> High-Speed Networks Laboratory, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil

<sup>3</sup> Dell EMC Research Center, Rio de Janeiro, Brazil