SCHEMA: Service Chain Elastic Management with Distributed Reinforcement Learning

Anestis Dalgkitsis*, Luis A. Garrido*, Prodromos-Vasileios Mekikis*, Kostas Ramantas*,
Luis Alonso§ and Christos Verikoukis†
*Iquadrat Informatica S.L., Barcelona, Spain
†Telecommunications Technological Centre of Catalonia (CTTC/CERCA), Castelldefels, Barcelona, Spain
§Signal Theory & Communications Dept., Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
Email: {a.dalgkitsis, l.garrido, vmekikis, kramantas}@iquadrat.com, luisg@tsc.upc.edu and cveri@cttc.es

***Abstract*—As the demand for Network Function Virtualization accelerates, service providers are expected to advance the way they manage and orchestrate their network services to offer lower latency services to their future users. Modern services require complex data flows between Virtual Network Functions, placed in separate network domains, risking an increase in latency that compromises the offered latency constraints. This shift requires high levels of automation to deal with the scale and load of future networks. In this paper, we formulate the Service Function Chaining (SFC) placement problem and then we tackle it by introducing SCHEMA, a Distributed Reinforcement Learning (RL) algorithm that performs complex SFC orchestration for low latency services. We combine multiple RL agents with a Bidding Mechanism to enable scalability on multi-domain networks. Finally, we use a simulation model to evaluate SCHEMA, and we demonstrate its ability to obtain a 60.54% reduction of average service latency when compared to a centralised RL solution.**

***Index Terms*—Zero-touch Orchestration, Network Function Virtualization, SFC Placement, Distributed Reinforcement Learning.**

I. Introduction

Wireless mobile networks are evolving at a pace faster than ever before. As networks are expanding from 5G to Beyond 5G and 6G architectures, supporting greater quality communications at a faster rate, lower latency and higher reliability at a much larger scale, the urgency of automated network management and orchestration is emerging amongst providers and the academia.

The introduction of Software-Defined Networking (SDN) and Network Function Virtualization (NFV) propelled the transition from static architectures to complex and easily scalable networks with multiple domains. These technologies allow multiple network services to share the same physical infrastructure and enables new business models. In particular, the Network–as–a–Service business model is expected to play a pivotal role in 5G mobile networks, allowing mobile network operators to tap into new revenue streams. The ease of deployment Mobile-Edge Computing servers in the field, which allows parts of services to be processed at the edge of the network [1], introduced new networks with multiple domains that span across different geographical locations. SDN and NFV, with the help of adaptive network service orchestration, are two of the main technical enablers that will allow operators to cope with the diverse range of service and user requirements. This state will characterize future network applications and services.

This new variety of services will be implemented using network softwarization features that are enabled by technologies such as SDN and NFV. These new services will be built as a series of interconnected Virtual Machines (VMs) or Containers, that together perform a specific function, hence called Virtual Network Functions (VNFs). These VNFs are typically chained together with data flows, forming more complex structures called Service Function Chains (SFCs).

For the deployment of SFCs on the physical infrastructure, the VNFs must be placed in a corresponding network node. However, the underlying physical network can have network domains that are geographically distributed, which yields a service that has a spatial distribution that can extend large distances, even hundreds of kilometres apart. Under these circumstances, the placement of VNFs in the network plays a significant role in the latency, performance and quality of the offered service. All of these will determine the Quality of Experience (QoE) of the end-user.

It is well known that telecommunication applications are extremely sensitive to performance indicators, especially latency, which tends to be influenced by the VNF arrangement. Multi-domain NFV ensures network service providers great flexibility in service deployment and cost optimization. As most carrier networks have Wide Area Networks (WANs) and use multiple domains, many SFCs are deployed over the WAN with VNFs located in different data centres. For reference, the upcoming 5G mobile networks are envisioned to support various services such as machine-type communications and the Internet of Things. Smart management and orchestration can be regarded as the most important service that can considerably reduce response time, especially in networks with multiple domains.

As the mobile networks are getting larger and the demand for computing resources increases, the necessity of automated service management and orchestration is necessary to offer better services to the users. The autonomous placement of SFCs is a key aspect of the Zero-touch network and Service Management (ZSM) in 5G and beyond networking [2]. The European Telecommunications Standards Institute (ETSI) has work evolving around ZSM and self-management on large-

scale networks. Researchers from both academia and the industry are researching solutions under ETSI. Several studies have proposed a solution to this problem, some addressing it through new network architecture designs, such as the ETSI proposals, and some through an algorithmic way [3], as we follow in this paper.

The main contribution of this paper is a ZSM scheme with attention to the scalability of multi-domain networks and the minimization of service latency. We model the complex multi-domain SFC placement problem as a Distributed Markov Decision Process (MDP) space and solve it with a proposed Distributed Reinforcement Learning (RL) algorithm. A bidding mechanism is proposed to enable scalability while maintaining the MDP space definition locally in geographically distributed domains, making the problem tractable in a parallel and asynchronous manner. The main contributions of this paper can be summarized as follows:

- A novel, practical and elastic SCF orchestration scheme, that is focused on scalability on multi-domain network architectures.
- A scalable SCF orchestration algorithm that is able to optimize for low latency, specifically targeted to Ultra-Reliable Low-Latency Communication (URLLC) services.
- We evaluate our proposed algorithm and compare it with existing solutions of the literature. We analyze the performance with focus on providing low-latency services.

The remainder of this paper is organized as follows. Section II provides an extensive discussion of related works of the literature. Section III offers an overview of the System Model. Section IV presents in detail the solution that we have developed. Section V showcases the experimental setup and verifies the performance of the proposed approach through simulations. Finally, Section VI provides a conclusion of this work and our future intentions.

II. Related Works

Autonomous SFC management and orchestration will play an integral role in 5G and beyond networks, as demand for services increases in the future. For this reason, problems such as SFC embedding, VNF placement and orchestration, was at the epicentre of the focus of both academia and industry in the past years.

Although a big fraction of the literature studies VNF placement and Virtual Network Embedding, the majority of these works consider solutions for a single domain approach regarding the placement. In [4], the authors attempt to maximize the revenue by optimizing the placement of the VNFs, by modelling and solving the problem with Integer Linear Programming (ILP). Similarly, authors in [5] develop a heuristic algorithm to optimize the network bandwidth consumption by taking advantage of the SFC placement. On the other hand, it is also possible to study the placement or embedding problem using tabular and Deep RL. One example of this is found in [3] in which the authors use Deep RL to perform the placement of Virtual Network Function - Forwarding Graphs (VNF-FGs) considering the constraints of the underlying infrastructure. Although these works may provide a satisfying solution for the scenario they study, they both ignore the large input space of the SFC placement problem on a modern, multi-domain network.

Single agent algorithms are unable provide efficient solutions for realistic environments with enormous problem and actions spaces due to the exponential growth of complexity in one centralized computing point. Most recently, a few works have tackled the multi-domain VNF embedding issue. Authors in [6] formulate the multi-domain VNF-FG embedding as an ILP problem and introduce a decentralized network utilization optimization scheme. However, their approach presents some limitations regarding the complexity of the algorithm as the load of the network increases. In contrast, Zhang et al. in [7] propose a cooperative multi-agent solution splits the network into a graph where each domain is responsible only for the internal placement of the VNFs, enabling this way scalability and great performance enhancements.

Inspired by the advancements of RL, researchers started adapting it into their solutions. Authors in [8] attempt to both minimize the operation cost of NFV providers and maximize the total throughput of requests with a Policy Gradient algorithm approach that automatically deploy SFCs. Similarly, authors in [9] propose the use of a Deep Deterministic Policy Gradient algorithm for the secure deployment of VNF-FGs in multi-domain networks. Although these approaches are beneficial to the orchestration of the SFCs in multi-domain networks, none of these directly tackle the latency of the offered services.

We can safely conclude that there is a gap in the multi-domain SFC placement for URLLC services in the literature awaiting to be filled. In this work, we tackle it directly, since most of the works studying this issue do not focus on minimizing the latency on multi-domain networks. To the best of our knowledge, we are amongst the first to propose a distributed RL-based approach for multi-domain SFC orchestration for URLLC services in this regard.

III. System Model

We consider a 5G network with multiple geographically distributed clouds, called *Domains*. Each domain consists of interconnected servers with computational resources that are able to instantiate, terminate or migrate VNFs to any other domain in the network. User terminals are connected to the domain servers through a millimetre wave (mmWave) 5G Base Station (BS) and request access to a service in the network. The services are comprised of multiple VNFs, located in different servers and domains, forming complex SFC chains between VNFs to execute a user service request.

We split the underlying network into two types of graphs with distinct levels to enable a *divide et impera* approach, separating the problem of SFC placement into smaller subtasks. The higher-level network graph is denoted as *Substrate Network Graph* and is comprised of the network domains and

the intermediate links that connect them. The domain subgraphs containing user devices, servers, their interconnected links and their gateway devices to the rest of the network are indicated as *Domain-Level Network Graphs*.

A. Substrate Network Graph

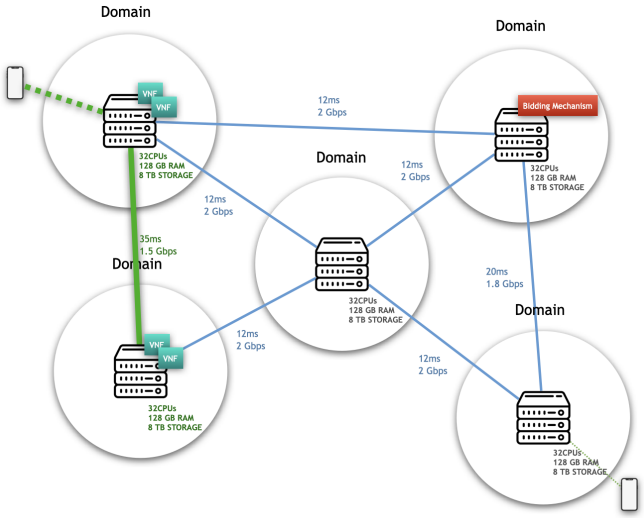

Figure 1. Substrate Network graph.

The *Substrate Network Graph* is defined as a non-directed weighted graph composed of a set of vertexes and links. As G_S we define the substrate network graph that consists of two sets, V_S and E_S:

- As $V_S = \{v_1, v_2, \ldots, v_m\}$ we denote the vertexes and m is the total number of vertexes.
- The $E_S = \{e_1, e_2, \ldots, e_k\}$ is the set of links, with k the total number of links that interconnect the V_S vertexes or domains.

The vertexes represent the computational domains of the network, whereas the links denote the physical links that interconnect them. The links that interconnect all computational domains in the network are defined with physical limitations $l_S = \{delay, bandwidth\}$. We assume that all computational domains have a capacity that can be defined as $c_s = \{cpu, ram, storage\}$, which limits the number of VNFs that each domain can serve and is equal to the sum of all domain server resources.

B. Domain-Level Network Graph

The distributed *Domain-Level Network Graphs* are defined as non-directed weighted graphs, composed of a set of vertexes and links once more. As G_D we define the substrate network graph that consists of two sets, V_D and E_D:

- As $V_D = \{v_1, v_2, \ldots, v_m\}$ we denote the vertexes in the network and m is the number of vertexes.
- The $E_D = \{e_1, e_2, \ldots, e_k\}$ is the set of links, with k the number of links.

The vertexes represent the servers that can host VNFs of the SFCs of the local domains, whereas the links denote

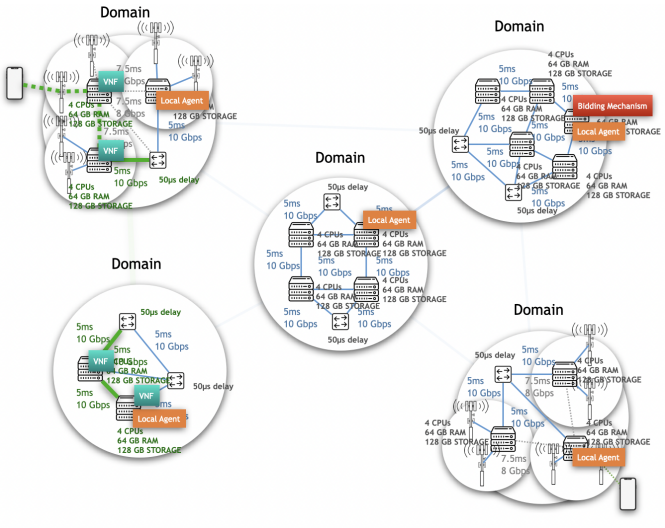

Figure 2. Distributed Domain-Level Network graphs.

the physical links that interconnect them. Similar to the substrate network grapg, the links have physical limitations $v_i = \{delay, bandwidth\}$ and the servers have a limited computational capacity $e_i = \{cpu, ram, storage\}$.

C. Service Function Chains

The VNFs, denoted with f_i, are considered as a cluster of interconnected VMs. Therefore, every VNF has computing requirements that need to be satisfied by the placement. The SFCs are a set of VNFs, denoted as s_i, where f_i are the VNFs: $s_i = \{f_1, f_2, \ldots, f_n\}$. We consider a flow of data between the VNFs of the chain whenever a user is granted access to the service that utilizes the SFC s_i. The flow of data is modelled as a directed graph: $flow_s = \{f_i \rightarrow f_j, f_j \rightarrow f_k\}$. The objective of this work is to identify and perform dynamically an optimized mapping of all SFCs in the network that offers minimal service latency to the users.

D. Delay Modeling

The VNFs f_i are considered as parts of the service that need to be placed in a server v_i in the network. Every VNF f_i can be described as a set of delays regarding the latency simulation, as follows: $f_i = D_{processing,i} + D_{transmission,i}$.

The total service delay that we minimize with our solution can be calculated as follows:

$$SL_{total} = \sum_{i=0}^{u} f_i, \tag{1}$$

where u is the number of users in the network.

IV. Solving the SFC Placement Problem with Distributed Reinforcement Learning

This section formalizes both the SFC orchestration problem and the proposed solution. We define the RL algorithm and the *Bidding Mechanism* that we have designed to solve the problem.

A. Problem Overview

The network topology consists of multiple domains that use SDN and NFV to provide URLLC services to the connected users. The local domains are connected through a WAN that we model as a link. Users are connected to the local domains through a 5G mmWave connection and each one requests a URLLC service from the network. Each service is offered by the network as an SFC with flows between multiple VNFs. Some VNFs are shared amongst the SFCs and they require vertical scaling depending on the number of requests. It is apparent that, due to the limited resources of the domain hardware and the number of hops between the servers, the mapping of the SFCs directly affects the offered URLLC service latency. If there are available resources on the network, the network accepts the incoming service requests.

The proposed system responds dynamically to the traffic load variation by initiating re-configurations in predetermined time-steps, only when required, to avoid additional costs or load to the network. Unlike works in similar literature where there is only one centralized entity responsible for the SFC mapping even for multiple domains, we employ a distributed system of agent that can operate on their own. Instead of utilizing a global network algorithm responsible for the orchestration of the SFC VNFs, we employ a local placement algorithm for each domain to share the enormous problem state space of the placement algorithm. The domains are performing VNF orchestration and VNF clustering internally, without affecting the rest of the SFC chain in the other domains. When there is a need for VNF migration between the domains, an *Auction* is taking place in which each local agent bids with its *Confidence* metric to receive and place locally a VNF from another local domain, as we will later explore in this section. The agents are bidding in the VNF auction through a confidence metric and only the highest bidder receives the VNF to place and orchestrate it locally its the output states.

B. Definition of the Reinforcement Learning agents

We formalize the intra-domain VNF placement and orchestration problem as an MDP by representing the problem environment through *Actions*, *States* and *Rewards*. In this specific problem we define them as follows:

- As *State* we define the intra-domain computational resources of the servers that are able to host the SFC VNFs (the available CPU cores, GBs of RAM and storage), the available bandwidths and latencies of the domain links.
- The *Action* of the local domain agents is the *Confidence metric* which is a float number indicating the *willingness* to receive the VNF of the SFC that is offered through the auction.
- As *Reward* we define the optimization function of the RL algorithm. The *Reward* is common between all local domain agents to enable cooperation as stated in [10]. The goal of SCHEMA is the dynamic orchestration of SFC with a distributed algorithm while keeping the service latency low and it is defined as follows:

$$\text{reward} = \sum_{i=0}^{n} \frac{T_{min}}{SL_{max}}, \tag{2}$$

where n is the total number of VNFs, T_min is the minimum service throughput and SL_{max} is the maximum service latency.

To build the local domain RL agents, we utilize a Deep Q-Network (DQN) agent as defined by the work of Mnih et al. in [11]. The agents interact with the network through the *Observations*, *Actions* and *Rewards* of the network. The goal of the agents is to select placements or *Actions* that maximize the *Reward* and thus, minimizing the service latency. We use a Deep Neural Network (DNN) to approximate the optimal action-value function, also known as Q-value function:

$$Q^*(s,a) = \max_{\pi} E[r_t + \gamma_{t+1} + \gamma^2 r_{t+2} + \ldots | s_t = s, a_t = a, \pi] \tag{3}$$

The Q-value function can be defined as the maximum sum of all rewards r_t, discounted by the parameter γ at each time-step t. The maximum sum of all rewards r_t is achieved by a behavioral policy $\pi = P(a|s)$, after an *Observation* s and taking an *Action* a.

To avoid instability during the training of the agents, we employ the *Experience Replay* technique, which randomizes the *Observations* and removes the correlation between them during the early training phase to force the agent to embrace exploration. We store the experiences $e_t = (s_t, a_t, r_t, s_{t+1})$ of the agent at each time-step t in the dataset $D_t = e_t, \ldots, e_t$, that we later use to retrieve them. We apply Q-learning value updates on mini-batches of experience $(s, a, r, s9)$, $U(D)$, drawn uniformly at random from the dataset D_t of stored experiences to perform learning for the agent. The Q-learning update during iteration i utilizes the following loss function:

$$\begin{aligned} L_i(\theta_i) = \epsilon_{(s,a,r,s')\ U(D)}[(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) \\ - Q(s, a; \theta_i))^2], \end{aligned} \tag{4}$$

where γ denotes the discount factor that determines the agent's horizon, θ the parameters of the Q-network during iteration i and θ_i^- are the network parameters used to compute the target value at iteration i.

C. Confidence Metric

Every local domain is served by a VM instance of the aforementioned RL agent and is responsible for the internal re-configuration, by selecting the host that the VNF that will be migrated. In addition to the placement *Action*, a *Confidence* metric can be extracted as a percentage of how confident the agent is for the chosen placement decision. The *Confidence* metric is obtained before applying the *arguments of the maxima*, also known as *argmax* function, in the output vector of the DNN, which is described with the following equation:

$$\underset{x \in D}{\text{argmax}} f(x) = \{x | f(x) \geq f(y) \forall y \in D\}, \quad (5)$$

where $f(x)$ is the set of inputs x from the DNN output D that achieve the highest function value. The *Confidence* metric is extracted from the set D as $max(D)$, whereas $f(x)$ denotes the placement.

D. Bidding Mechanism

We propose a multi-domain, distributed and non-cooperative scheme to enable scalability in the SFC placement problem while keeping the benefits of using RL. Scaling the SFC placement with RL from one to multiple domains without increasing exponentially the complexity requires the introduction of alternative techniques instead of just scaling the DNNs.

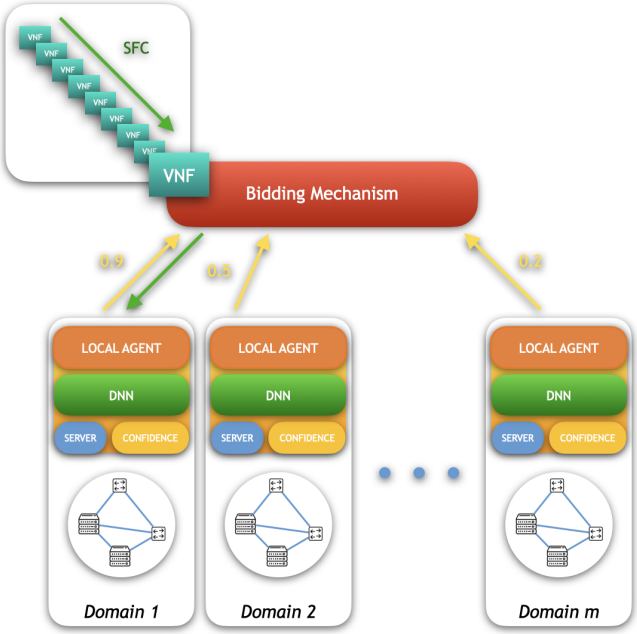

Figure 3. Overview of the Bidding Mechanism.

We introduce the *Bidding Mechanism*, an entity that performs an *Auction* of all SFC VNFs at every time-step in a serialized manner. Each local domain agent place a bid with its placement *Confidence* metric to receive a VNF of the given SFC and only the highest bidder can receive it to perform an internal placement, as the output of the agent denotes. Domains have only knowledge of their own resources making them autonomous to the intra-domain placement procedure. Local domain agents do not communicate with each other to determine the optimal solution, but rather use the global *Reward* to introduce cooperation between the agents, as proposed by Mao et al. in [10].

V. Simulation Results & Evaluation

In this section, we conduct a simulation study in various traffic scenarios and multi-domain networks to evaluate the performance of the proposed scheme.

A. Simulation Setup

The simulation of the virtualized network entities was simulated with *Python* language using a custom OpenAI Gym enviironment. The RL agents were built using *TensorFlow* [12] and the high-level *Keras API* open-source library [13].

For the simulation results presented in the following subsection, we have performed multiple simulation scenarios with groups of $U = \{100, 500, 1000, 1500, 2000\}$ users and $D = \{3, 5, 7, 9, 11\}$ domains. The inter-domain links were assigned a random latency following a normal distribution $l_S^{\text{lat}} \in [2, 3]$ ms. The intra-domain network is composed of servers with 32 Cores, RAM 128 GB, 1TB storage and the intra-domain links are assigned a random latency value between $l_D^{\text{lat}} \in [1, 2]$ ms. Users are connected through a 5G mmWave wireless link to a BS connected directly to a server of the inter-domain network with a 2% to 10% loss. The SFC VNFs have 1 to 2 CPU cores, 2 to 4 GB RAM and 1 to 80 GB storage as computational resource requirement for placement.

B. Baselines

The baselines considered to evaluate SCHEMA are a DQN based single agent solution and a random placement method. The parameters of the DQN DNN were scaled appropriately to match the size of the network of each scenario.

C. Performance Evaluation

In our simulations, we examine the performance of the proposed algorithm by comparing both the average user service latency and the average number of service rejections that occur during every simulation scenario.

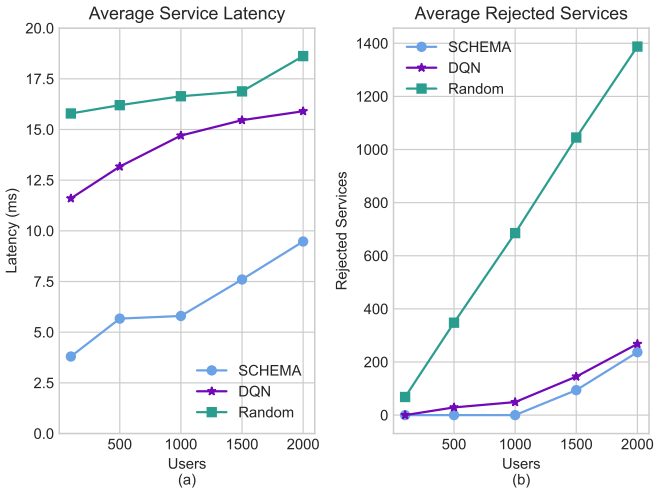

Figure 4. (a) Average service latency of accepted services for 3 domains. (b) Rejected services by the number of users for 3 domains.

As we can observe in Fig. 4, increasing the number of users in the network also increases the average service latency due to insufficient computing resources in servers within the local domains. Specifically, in the case of 3 domains, our proposed scheme in Fig. 4a was able to outperform both baseline solutions by offering considerably lower service latency by

almost 60.54% in the case of 1000 users. Accordingly, in Fig. 4b SCHEMA rejected less user services compared to the DQN by 54.25% in the case of 1500 users, demonstrating better SFC placements for the same infrastructure. As the number of users increases we can see reaching the limit of insufficient resources, near the 1600 users.

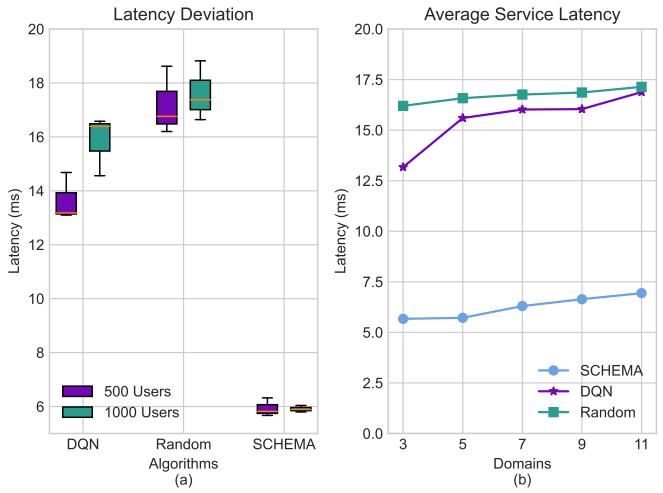

Figure 5. (a) Latency variation in 500 and 1000 user scenarios for 3 domains. (b) Average service latency of 500 users for 3 domains.

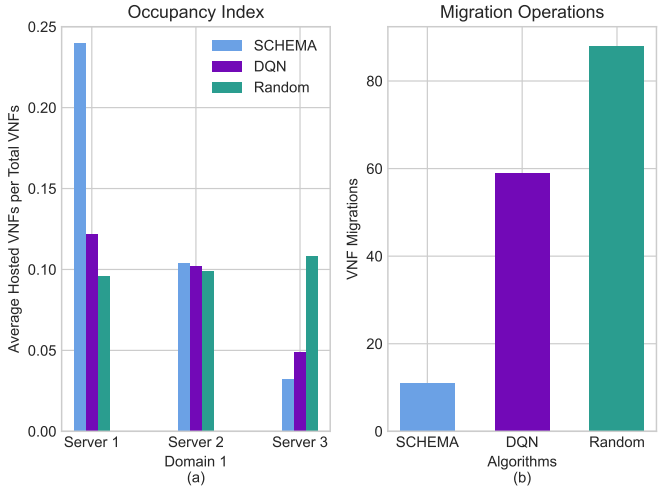

Figure 6. (a) VNF Occupancy index or the average hosted VNFs per total VNFs of 100 simulation iterations for 500 users. (b) Migration operations of simulation iterations and 50 SFCs with 125 VNFs.

Moreover, Fig. 5a outlines the service latency variation for the cases of 500 and 1000 users. After training, SCHEMA was able to conceive a better placement than the DQN, leading to consistently lower service latency by almost 63.33% in the case of 5 domains. It is evident that the DQN is heavily affected by the number of the users comparing the height difference in the boxes. With the help of Fig. 6a, we can conclude that SCHEMA gravitated towards consolidating the SFC VNFs in the same server to further reduce the number of hops to the end user. In Fig. 6b we have further evidence that due to the introduction of the *Bidding Mechanism*, the local agent of the depicted Domain in 6a was keeping the same placement to avoid inter-domain SFC re-configurations.

VI. Conclusion

In this paper, we have studied the elastic multi-domain SFC placement problem. We introduced SCHEMA, a distributed SCF orchestration scheme that utilizes the domains of the network to perform VNF placements locally, without affecting the rest of the chain. We have introduced the *Bidding Mechanism* that the local domain agents utilize to acquire and host VNFs internally, with their local resources. The distributed agents learn how to cooperate through a common reward and perform SFC placements in multiple domains while keeping the service latency low for URLLC services. The results confirm superior performance in multiple scenarios, maintaining the same levels of efficiency in multiple numbers of domains as compared to a single RL agent solution.

References

[1] J. Gil Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[2] Y. Goto, "Standardization of automation technology for network slice management by etsi zero touch network and service management industry specification group (zsm isg)," *NTT Technical Review*, vol. 16, pp. 39–43, 09 2018.

[3] P. T. A. Quang, Y. Hadjadj-Aoul, and A. Outtagarts, "A deep reinforcement learning approach for vnf forwarding graph embedding," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1318–1331, 2019.

[4] S. Sahhaf, W. Tavernier, M. Rost, S. Schmid, D. Colle, M. Pickavet, and P. Demeester, "Network service chaining with optimized network function embedding supporting service decompositions," *Computer Networks*, vol. 93, pp. 492–505, 2015, cloud Networking and Communications II. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912861500359X

[5] Z. Ye, X. Cao, J. Wang, H. Yu, and C. Qiao, "Joint topology design and mapping of service function chains for efficient, scalable, and reliable network functions virtualization," *IEEE Network*, vol. 30, no. 3, pp. 81–87, 2016.

[6] P. T. A. Quang, A. Bradai, K. D. Singh, G. Picard, and R. Riggio, "Single and multi-domain adaptive allocation algorithms for vnf forwarding graph embedding," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 98–112, 2019.

[7] Q. Zhang, X. Wang, I. Kim, P. Palacharla, and T. Ikeuchi, "Service function chaining in multi-domain networks," in *2016 Optical Fiber Communications Conference and Exhibition (OFC)*, 2016, pp. 1–3.

[8] Y. Xiao, Q. Zhang, F. Liu, J. Wang, M. Zhao, Z. Zhang, and J. Zhang, "Nfvdeep: Adaptive online service function chain deployment with deep reinforcement learning," in *2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS)*, 2019, pp. 1–10.

[9] P. T. A. Quang, A. Bradai, K. D. Singh, and Y. Hadjadj-Aoul, "Multi-domain non-cooperative vnf-fg embedding: A deep reinforcement learning approach," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 886–891.

[10] H. Mao, Z. Gong, and Z. Xiao, "Reward design in cooperative multi-agent reinforcement learning for packet routing," *arXiv preprint arXiv:2003.03433*, 2020.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[12] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[13] F. Chollet *et al.*, "Keras," https://keras.io, 2015.