Admission Control and Resource Allocation in 5G Network Slicing

William F. Villota Jácome[1], **Oscar M. Caicedo Rendon**[2], **Nelson L. S. da Fonseca** [1]

[1]Institute of Computing – University of Campinas (UNICAMP)

[2]University of Cauca (UNICAUCA)

wfernando@lrc.ic.unicamp.br, nfonseca@ic.unicamp.br,

omcaicedo@unicauca.edu.co

Abstract. *This paper summarizes the research in the master thesis entitled "Admission Control and Resource Allocation in 5G Network Slicing". We propose two solutions, SARA and DSARA, based on Reinforcement Learning algorithms to learn the admission policy that optimizes the profit of providers. Resource allocation considers the QoS requirements. Results show the outstanding performance of our solutions to 5G Network Slicing in relation to profit and resource utilization.*

Resumo. *Este artigo resume a tese de mestrado intitulada "Admission Control and Resource Allocation in 5G Network Slicing". Propomos duas soluções, SARA e DSARA, que usam algoritmos de Reinforcement Learning para aprender a política de admissão que otimiza o lucro dos provedores. A alocação de recursos considera os requisitos de QoS. Os resultados mostram o bom desempenho de nossas soluções para o Fatiamento de Rede 5G em relação ao lucro e à utilização de recursos.*

1. Introduction

5G is key to provide a myriad of services with different Quality of Service (QoS) requirements. To accomplish this, 5G requires the flexibility and modularity provided by Network Slicing which involves Software-defined Networking (SDN) and Network Function Virtualization (NFV) to provide programmability for creating several Network Slices (NSLs) on-demand over a shared infrastructure. NSLs are virtual networks composed of Virtual Network Functions (VNF) customized to meet particular needs.

Network Slice Providers (NSPs) receive several NSL requests (NSLRs) belonging to one of three types: Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low Latency Communication (URLLC), or Massive IoT (MIoT). As substrate resources are finite and NSLRs have particular QoS requirements, NSPs face the challenge of controlling their admission to increase the overall profit and improve network resource utilization. This challenge involves: an admission control (AC) mechanism, a decision process to permit or restrict the access to a system (e.g., substrate network) considering one or more criteria like profit; and a resource allocation (RA) mechanism that includes the allocation of network core and edge resources.

Intelligent AC and RA for 5G NSLRs are critical to optimize NSP profit and network resource utilization. Machine Learning (ML) techniques like Reinforcement Learning (RL) and Deep RL (DRL) are efficient in solving decision-making problems. Since no data is available a priori about the 5G Network Slicing dynamic, RL and DRL are appropriate to learn from the information generated in such dynamic. Network Slicing has repetitive decisions that produce data that can be leveraged to train RL/DRL-agents.

We propose two solutions, SARA and DSARA, to jointly and intelligently perform AC and RA for 5G NSLRs aiming at optimizing the NSP profit and resource utilization. SARA is based on an RL-agent that learns the NSLRs that increases the NSP profit. NSLRs are collected on batches during time windows favoring profit maximization. SARA performs node and link mapping to allocate substrate resources to NSLs by considering its service type (*i.e.,* eMBB, URLLC, and MIoT) and node type (core or edge) for supporting QoS requirements. DSARA is based on DRL for extending SARA to cope with large scenarios with long convergence time. The DRL-agent of DSARA approximates the admission policy function, enabling to learn from less interactions with the environment. This thesis introduces the following original contributions:

- An RL-based algorithm for AC of 5G NSLRs that increases the profit of service providers and network resource utilization.
- A DRL-based algorithm for AC of 5G NSLRs to further improve the profit of providers in large scenarios while obtaining fast convergence.

The relevance of our contributions is a first step towards an automated and intelligent operation of 5G networks.

2. Related Work

Several papers have addressed AC for 5G NSLRs [Han et al. 2020], [Bega et al. 2019], and [Sciancalepore et al. 2019]. The aforecited papers make admission decisions on individual NSLRs which prevents the selection of a set of them that can potentially optimize a given objective in a specific time window. Also, these papers do not consider different types of requests according to standardized 5G use cases, neglecting the diversity of QoS requirements of 5G service types, and most of them focus only on edge nodes.

Numerous papers have addressed RA in NFV [Zhang et al. 2019], [Agarwal et al. 2019], and [D'Oro et al. 2020]. These papers focus on mapping NSLs without controlling the admission of requests. Performing AC and RA jointly in 5G core network slicing is critical to optimize resource utilization and maximize the NSP profit. RL and DRL are efficient tools for solving decision-making problems modeled as Markov Decision Processes. Moreover, the Network Slicing process has repetitive decisions and produces a large quantity of data to train RL/DRL-algorithms.

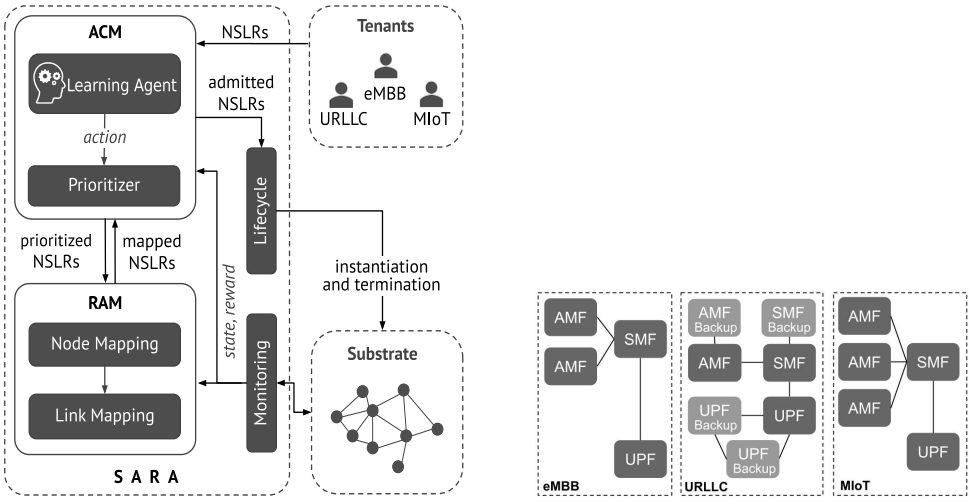
SARA and DSARA are model-free RL and DRL algorithms; they do not make assumptions about the environment (*i.e.,* substrate network), they learn continuously by exploring it without prior knowledge. The originality of our proposal comes from the fact that no other work performs jointly AC and RA based on RL or DRL, differentiates core and edge nodes, and considers the typical 5G use cases.

3. SARA and DSARA Architecture

3.1. Overview

Network Slicing involves repetitive decisions that produce amounts of valuable data to train ML algorithms. RL algorithms are efficient for solving decision-making problems. Since no data is available a priori about the 5G Network Slicing dynamic, RL is appropriate to learn from the instantaneous information generated in the slicing process. SARA and DSARA perform AC and RA for 5G NSLRs launched by different tenants. AC is based on RL and DRL agents which learn from the slicing dynamic the admission policy that maximizes the NSP's profit. RA includes node mapping and link mapping steps for meeting latency, bandwidth, processing and reliability requirements of each type of service.

3.2. Modules



(a) Architecture of SARA (b) NSL graphs

Figure 1 Architecture and NSL graphs

The architecture (Figure 1a) is composed of four modules (Admission Control Module - ACM, Resource Allocation Module - RAM, Lifecycle Module, and Monitoring Module) that interact with substrate network and the NSLRs of different tenants. The substrate network, based on the ETSI recommendation [Etsi 2013], is composed of several Points of Presence (NFVI-PoPs). NFVI-POPs are nodes that offer processing capacity and can be either high capacity data centers (core nodes) or small ones close to end users (edge nodes). Core nodes are appropriate for the 5G Control Plane that requires high processing and bandwidth capacities. Edge nodes are adequate for 5G User Plane VNFs which need to be located close to the end-users. We model the 5G core network substrate as a labeled and weighted undirected graph: $SN = \{N, L\}$. N is the set of nodes, $N = \{n_1, n_2...n_m\}$, and L is the set of links, $L = \{(n_1, n_2), (n_1, n_3)...(n_l, n_m)\}$. The processing capacity of node n_i is $CPU(n_i)$. The bandwidth of link (n_i, n_j) is $BW(n_i, n_j)$.

An NSLR is described by $nslr = \{s\_type, T_o, G\}$. s_type is the 5G use case eMBB, URLLC, or MIoT. T_o is the operational time. $G = \{F, V\}$ is a labeled and weighted undirected graph representing an NSL. F is the set of VNFs, and V is the set of virtual links connecting them. $cpu(vnf_i)$ is the processing capacity required by vnf_i and

$type(vnf_i)$ is the node type vnf_i requires. $bw(vnf_i, vnf_j)$ is the bandwidth demanded by the virtual link (vnf_i, vnf_j). Figure 1b depicts the NSL graphs of typical 5G use cases. The graphs follow the 5G Control Plane (CP) and User Plane (UP) Separation (CUPS). Each graph includes essential CP VNFs (Access and Mobility Management Function, AMF) and Session Management Function, SMF) and UP VNFs (User Plane Function, UPF). We model the graphs regarding the features of 5G use cases.

The Monitoring Module gets resource availability information (structured as states and rewards) from network substrate and delivers it to ACM and RAM. ACM performs the admission of NSLRs by employing an agent (based on either RL or DRL) and a Prioritizer. The agent chooses a normalized weight value for each type of service. The Prioritizer uses these values to sort the NSLRs for establishing the order they should allocate resources. The weights lead to the maximum profit, i.e., the agent learns to select the action that maximizes the profit considering the information on states and rewards from the environment.

$$EP(n_i) = CPU(n_i) \times \sum_{l \in adj(n_i)} BW(l_j) \tag{1}$$

RAM allocates resources to NSLRs by performing the mapping $M : G = \{F, V\} \rightarrow SN' = \{N', L'\}$, while meeting the requirements of each type of service. N' is a subset of N and L' is a subset of L. RAM performs two steps, node mapping and link mapping. Node mapping maps the VNFs of an NSLR onto nodes in the substrate network, while meeting processing, latency and reliability requirements. The service type s_type allows choosing the type of node in SN. All nodes in SN, are ordered according to their embedding potential EP, given by Equation 1). EP determines the capacity of a node to embed a VNF considering its available capacities $CPU(n_i)$ and $BW(adj(n_i))$. Link Mapping maps virtual links onto paths (sets of links) with the lowest number of hops to minimize the bandwidth utilization that allows to admit new NSLRs. The allocation information for accepted NSLRs is passed to the Lifecycle module.

4. RL/DRL-based Admission Control

4.1. SARA-agent

The AC mechanism of SARA (detailed in Section 3.3 in [Villota 2020]) is based on Q-learning, an RL algorithm. RL algorithms are described by state space S, action space A, reward function R, and exploration method. An RL-agent takes an action (a_t), under a state (s_t), that is applied on the environment that returns a reward (r_t) and the next state (s_{t+1}) to the RL-agent. In Q-learning, the agent uses a lookup table (Q-table), to store the quality value of each action (*i.e.,* the Q-value).

- **State Space**. A state $s \in S$ is defined by $\{cpu(E), cpu(C), bw(L)\}$. $cpu(E)$ and $cpu(C)$ are the available processing capacity in the set of edge (E) and core nodes (C), respectively. $bw(L)$ is the available bandwidth in the set of links (L). The substrate resource capacity is discretized in ten equal intervals. The number of states is 10^3.
- **Action Space**. An action $a \in A$ is denoted by $a = \{w_{embb}, w_{urllc}, w_{miot}\}$. w_{embb}, w_{urllc}, and w_{miot} are admission weights for each service type. In every step, the RL-agent chooses the action a that returns the maximum profit.

- **Reward**. The reward received by the agent after taking an action is computed by Equation 2. $p(nsl_i)$ is the amount of money earned by NSP for selling the NSL minus the operational cost. $maxP(SN, T)$ is the maximum profit the NSP could receive when using all the resources in the substrate (SN) in a certain period (T).

$$R = \frac{\sum_{i=0}^{k} p(nsl_i)}{maxP(SN, T)} \tag{2}$$

- **Exploration**. RL-agent uses the epsilon-greedy method which allows the selection of either the current expected optimal action with probability $1 - \varepsilon$ or a random action with probability ε.

4.2. DSARA-agent

SARA could produce even higher profit values if more information is provided (more features to represent the states and more actions to explore). However, such enhancement is achieved at the cost of two limitations: a longer convergence time since the Q-learning-agent has to experience more state-action pairs many times before learning, and larger memory capacity to store Q-values. Deep Q-learning (DQN), a DRL algorithm deals with such limitations by using a function approximator (Neural Network (NN), generally) for generalizing the knowledge learned from some already visited states to other similar states. An NN is a set of interconnected neurons organized into layers that applies a weighted sum to its input and passes its output to the next neuron. Neurons learn by adjusting their weights based on examples. An NN allows learning from less interactions with the environment. To extend SARA to cope with larger scenarios, we replace the Q-learning-agent by a DQN-agent. This new approach is named DSARA.

Figure 2 depicts our DRL-agent that includes two NNs (target and evaluation), Replay Memory (*i.e.,* to store past experiences for training), and loss calculation and weights updates. Evaluation NN estimates Q-values while target NN estimates the target Q-values. Training is performed by reducing the difference between Q-values and target Q-values. The elements that specify our DRL-based solution are:

- **State Space**. The notation $\{cpu(E), cpu(C), bw(L), n_e, n_u, n_m\}$ represents a state in S. $cpu(E)$ and $cpu(C)$ are the available processing capacity in edge (E) and core nodes (C), respectively. $bw(L)$ is the available bandwidth in links (L). n_e, n_u, and n_m are the number of NSLs in operation of type eMBB, URLLC, and MIoT, respectively.
- **Action Space**. Every action in A is denoted by $\{w_{embb}, w_{urllc}, w_{miot}\}$. w_{embb}, w_{urllc}, and w_{miot} are the weights for each type of service
- **Reward**. The same considered in SARA (See section 3).
- **Exploration**. We use an epsilon-greedy method with decaying ε (Equation 3) which enables reducing the exploration parameter progressively along the episodes.

$$\varepsilon_t = \varepsilon_{max} - (nsteps_t \times dr) \tag{3}$$

- **Evaluation NN** estimates the Q-value $Q(s_t, a_t)$ for each state-action pair.
- **Target NN** returns the Target Q-value $Q^+(s_t, a_t)$ for each state-action pair by Equation 4. Target Q-values are used for the loss calculation when training the Evaluation NN.

$$Q^+(s_t, a_t) = R_t + \gamma \cdot maxQ(s_{t+1}, a) \tag{4}$$

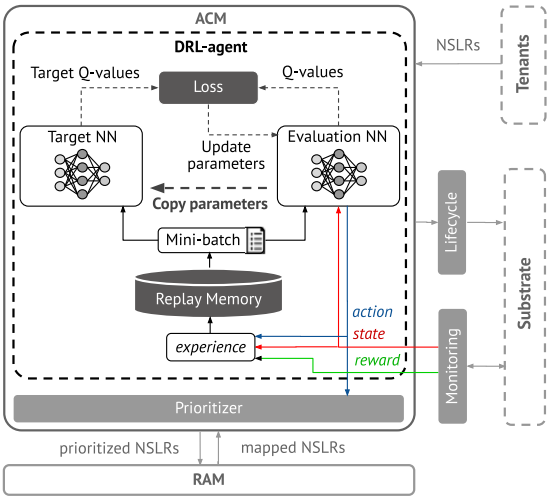


Figure 2 DRL-agent of DSARA

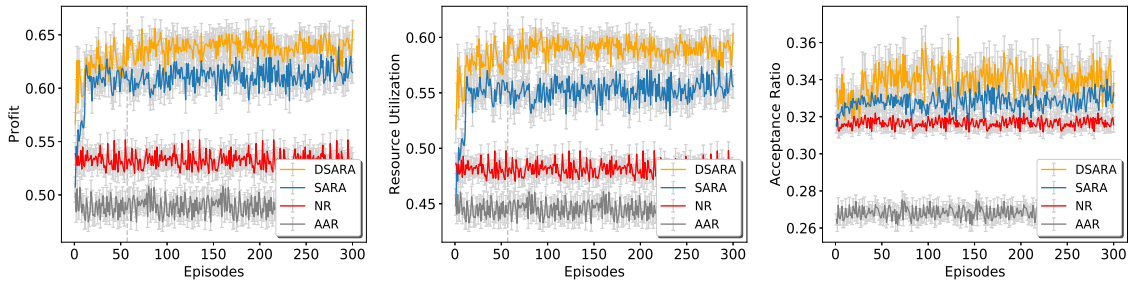
- **Loss** is calculated by the difference between the Q-value estimated by the Evaluation NN and the Target Q-value obtained by the Target NN (Equation 5). Target NN is fixed temporarily and updated periodically with the trained parameters of the Evaluation NN for learning stability reasons.

$$Loss = (Q^{+}(s_t, a_t) - Q(s_t, a_t))^2 \tag{5}$$

5. Evaluation

The performance evaluation of SARA and DSARA considers profit, resource utilization, and acceptance ratio. Benchmarks are Always Admit Requests (AAR) and Node Ranking (NR) heuristics that admit NSLRs as they arrive if resource capacity is available. NR ranks nodes according to their embedding potential, while ARR did not differentiate nodes. Experiments include topologies of 16, 32, and 64-nodes generated by using the Barabasi-Alberth algorithm. We develop the architectural modules and the environment (discrete event simulator) in Python 3 executed on an Ubuntu 16.04 LTS desktop with Intel Core i5-4570 CPU and 15.5 GB RAM. We use NetworkX library to manipulate NSLRs graphs and topologies. Evaluation and Target NNs were set with 6 neurons in the input, 150 neurons in hidden layer, and 30 neurons in the output. The NSLRs operation time follows an exponential distribution with a mean 12 time units. The arrivals for the three types of NSLRs follow the Poisson process. The total arrival rate was varied from 1 to 100 requests per time unit. Time window last 2 time units. Simulation assumptions and parameters were set according to values used in the literature. We conducted 33 repetitions to obtain results with 95% confidence level.

Figures 3a, 3b, and 3c show that SARA and DSARA outperformed the baselines. SARA and DSARA profits increased from episodes 1 to 12 and from episodes 1 to 55, respectively. After convergence, DSARA profit is 3%, 10%, and 14.3% greater than the profits of SARA, NR, and AAR, due to its DQN-algorithm that quickly learns the most profitable NSLRs. The utilization of SARA and DSARA increased rapidly from episodes 1 to 12 and from episodes 1 and 55, respectively. After converging, DSARA obtained 12%, 9%, and 5% higher resource utilization than AAR, NR, and SARA. Also, DSARA got 2%, 8%, and 12% higher acceptance ratio than SARA, NR, and AAR, respectively.



(a) Profit (b) Resource Utilization (c) Acceptance Ratio

Figure 3 Results for 20 requests per time unit in the 16-node topology

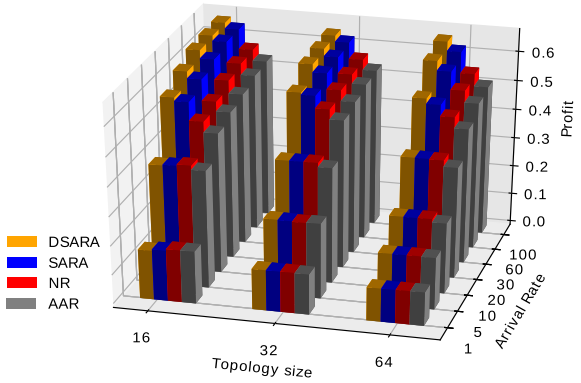


Figure 4 Profit for different loads and topologies

Values produced by NR and AAR did not evolve as they do not learn from the environment. For the sake of visual interpretation, we plotted 300 episodes which is more than the sufficient to observe that the agents maintain their performance after convergence (when agents have learned the optimal policy).

Figure 4 depicts the profits obtained for different topologies and arrival rates. SARA's profit was greater than that achieved by AAR and NR from 7, 15, and 25 requests in the 16, 32, and 64-node topologies, respectively. DSARA's profit was higher than that achieved by SARA from 10, 20, and 30 requests in the 16, 32, and 64-node topologies, respectively. This occurs since the more requests, the more experiences from which the agents can learn. DSARA got 3.2%, 3%, and 3.6% higher profit than SARA in the 16, 32, and 64-node topologies. These gains could represent a significant monetary difference. SARA obtained the wider confidence interval for the 16-node topology and 30 requests per time unit: 95% CI $[0.625, 0.634]$.

6. Conclusion

We proposed SARA and DSARA to jointly and intelligently perform admission control and resource allocation for 5G NSLRs of eMBB, URLLC, and MIoT use cases. SARA uses an RL-agent to learn the admission policy that optimizes profit and resource utilization. DSARA uses a DRL-agent to cope with scenarios where the number of state-action pairs is large. These algorithms are model-free; they do not make assumptions about the substrate network as do optimization-based approaches.

Evaluation was performed in terms of profit, resource utilization, and acceptance ratio for different topology sizes and request arrival rates. SARA obtained up to 7% and

11.3% higher profit than the generated by the NR and AAR in the 16-node topology, 5.6% and 9% in the 32-node topology, and 7.1% and 11.2% in the 64-node topology. These results corroborate that SARA is suitable for managing AC and RA of 5G NSLRs for optimizing the profit of NSPs. DSARA achieved up to 3.2%, 3%, and 3.6% higher profit values than SARA did for 16, 32, and 64-node topologies, respectively. The DRL-agent of DSARA is proper to cope with large scenarios where the use of typical RL-based approaches as SARA may become impractical.

Future work includes: enhancing AC and RA mechanisms with latest developments in the RL field and more sophisticated RA strategies [Spinnewyn et al. 2018], extending our solutions to support end-to-end slices (*i.e.,* 5G radio access and core), and implementing adaptive scaling for NSLs at run-time to guarantee QoS variations.

7. Publications

- Casas-Velasco, D. M., **Villota-Jácome W. F.**, Fonseca, N.L.S., Caicedo O. M. Delay Estimation in Fogs based on Software-defined Networking. In 2019 IEEE Globecom.
- **Villota-Jácome W. F.**, Caicedo O. M., Fonseca, N.L.S. Admission Control for 5G Network Slicing based on Reinforcement Learning. Submitted to IEEE TNSM.

References

Agarwal, S., Malandrino, F., Chiasserini, C. F., and De, S. (2019). Vnf placement and resource allocation for the support of vertical services in 5g networks. *IEEE/ACM Transactions on Networking*, 27(1):433–446.

Bega, D., Gramaglia, M., Banchs, A., Sciancalepore, V., and Costa-Pérez, X. (2019). A machine learning approach to 5g infrastructure market optimization. *IEEE Transactions on Mobile Computing*, 19(3):498–512.

D'Oro, S., Bonati, L., Restuccia, F., Polese, M., Zorzi, M., and Melodia, T. (2020). Sl-edge: Network slicing at the edge. *arXiv preprint arXiv:2005.00886*.

Etsi, N. (2013). Etsi gs nfv 002 v1. 1.1 network functions virtualization (nfv). *Architecture and Framework: ONF*.

Han, B., Sciancalepore, V., Costa-Perez, X., Feng, D., and Schotten, H. D. (2020). Multiservice-based network slicing orchestration with impatient tenants. *IEEE Transactions on Wireless Communications*, 19(7):5010–5024.

Sciancalepore, V., Costa, X., and Banchs, A. (2019). Rl-nsb: Reinforcement learning-based 5g network slice broker. *IEEE/ACM Trans. on Networking*, 27(4):1543–1557.

Spinnewyn, B., Isolani, P. H., Donato, C., Botero, J. F., and Latré, S. (2018). Coordinated service composition and embedding of 5g location-constrained network functions. *IEEE TNSM*, 15(4):1488–1502.

Villota, W. (2020). Admission control and resource allocation in 5g network slicing.

Zhang, Q., Liu, F., and Zeng, C. (2019). Adaptive interference-aware vnf placement for service-customized 5g network slices. In *IEEE INFOCOM*, pages 2449–2457. IEEE.