Minimizing the Cost of 5G Network Slice Broker

Ali Gohar
Dept. of Electrical Engineering and Computer Science
University of Stavanger
Stavanger, Norway
ali.gohar@uis.no

Gianfranco Nencioni
Dept. of Electrical Engineering and Computer Science
University of Stavanger
Stavanger, Norway
gianfranco.nencioni@uis.no

***Abstract*—Network slicing is a key enabler of the fifth-generation (5G) of mobile networks. It allows creating multiple logical networks, i.e. network slices, with heterogeneous requirements over a common underlying infrastructure. The underlying infrastructure is composed of heterogeneous resources, such as network and computational resources. These resources are owned and managed by various Infrastructure Providers (InPs). In network slicing, a new actor, called Slice Broker (SB), purchases resources from the various InPs to create the network slices. In this paper, we address the problem of the allocation of network slices. Our target is to minimize the total cost of SB to acquire the resources from the InPs. The contributions are the following: (i) we define the addressed problem; (ii) we propose a heuristic solution to the problem; (iii) we evaluate the behavior of the proposed heuristic in various scenarios, and we compare it with a benchmark solution. The results show that a cost reduction from 60% to 80% is possible in all scenarios investigated.**

***Index Terms*—5G, Broker, Cost Minimization, Multi-access Edge Computing, Network Slicing, Resource Allocation, Service Function Chain,**

I. INTRODUCTION

The fifth-generation (5G) of the mobile network aims to provide differentiated services on top of a shared infrastructure. Therefore, the 5G mobile networks embrace software-based networking solutions for various network functions, built over virtualization technologies such as Network Function Virtualization (NFV) [1]. Network slicing is a key enabler of 5G mobile networks and allows the creation of network slices. A network slice is an isolated, end-to-end, and customized logical network on top of a shared infrastructure. The Third Generation Partnership Project (3GPP) classifies the network slice into four categories such as Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), Massive Machine Type Communication (mMTC), and Vehicle-to-Everything (V2X) [2].

Differentiated services provided by 5G mobile networks are often composed of interconnected Virtual Network Functions (VNFs). VNFs can be dynamically deployed on a distributed cloud and edge infrastructure. The infrastructure consists of heterogeneous resources such as computing, storage, and network, and is owned or managed by various Infrastructure Providers (InPs). Since 5G aims to provide ultra-low latency services, Multi-access Edge Computing (MEC) is a complementary technology to cloud computing [3]. MEC extends cloud computing capabilities by providing cloud computing capabilities at the network's edges.

Sufficient InP resources must be allocated to VNFs when a network slice is instantiated. It becomes challenging to manage multiple InP resources and meet the requirements of various network slices. Therefore, a new actor that can satisfy network slice requirements while making the most effective use of InP resources is required. In this regard, Slice Broker (SB) is introduced [4]. To create network slices with specific requirements, the SB acquires heterogeneous (computational and network) resources from various InPs. To this purpose, novel algorithms are needed to jointly select different resources from the various InPs and allocate multiple network slices.

In this paper, we focus on minimizing the costs of the SB for acquiring from the InPs the resources needed to create the requested network slices with given guaranteed requirements. The paper has the following main contributions:

- We describe the slice allocation problem of minimizing the costs of the SB, given the InP resources in edge and core networks and the computational and network requirements for the network slices;
- We propose a heuristic solution to the defined problem;
- We evaluate and analyze the performance of the proposed solution under various scenarios.

The rest of the paper is organized as follows. The problem description is presented in Section II. A heuristic algorithm is proposed as a problem solution in Section III. The performance of the heuristic algorithm is evaluated through simulations in Section IV. The paper concludes with final remarks in Section V.

II. PROBLEM DESCRIPTION

In this section, we first highlight the objective of our problem and clearly state the assumptions. Second, we present the type of resources provided by InPs. Finally, the requirements for creating a network slice is discussed.

A. Objective

In this problem, two actors are considered, InPs and SB. The InPs own resources and provide them to the SB. The SB utilizes the resources provided by the InPs to create slices. In this problem, it needs to be jointly decided:

- Amount of resources that the SB is buying from InP;

G. Nencioni was supported by the Norwegian Research Council through the 5G-MODaNeI project (no. 308909).

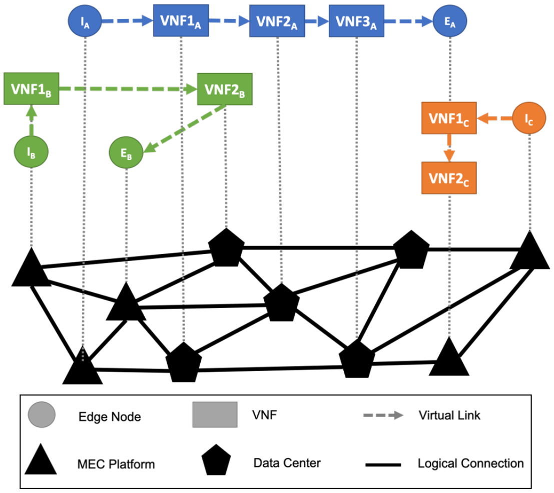

Fig. 1. Example of Network Slice resource allocation

- Allocation of each network slice.

Following assumptions are considered in this problem:

- *Static problem* – There is no dynamism in the request of network slices, nor the acquisition of resources. The SB will acquire the required resources and creates all the network slices at once.
- *Full satisfaction* – All the network slice requests are allocated and the requirements are satisfied.

B. InP resources

Two kinds of resources that are provided by the InP are considered in the problem as follows:

- *Computational* – virtual resources in a computing platform;
- *Network* – logical connections between the computing platforms.

A computing platform can be a *MEC* or a *Data Center (DC)*. The computational resources are usually provided by *cloud providers* or similar [5]. We do not consider radio resources. Following virtual resources are considered in a computing platform:

- *Processing*;
- *Storage*;
- *Intra-connection data rate* – data rate for communication inside the computing platform;
- *Intra-connection delay* – guaranteed maximum latency associated communication inside the computing platform.

The network resources are usually provided by *network operators*. For the logical connections, following resources are considered:

- *Data rate* – guaranteed minimum data rate;
- *Delay* – guaranteed maximum latency.

Note that a logical connection is a physical path between computing platforms set up by the network operator with the guaranteed requirement. We assume that a logical connection can be created between every computing platform. Both cloud providers and network operators offer different configurations at different prices for virtual resources and logical connections, respectively.

C. Network slice

Network service consists of an ordered concatenation of VNFs to form a Service Function Chain (SFC). The composition of SFC seeks to reorganize VNFs in a SFC to enhance allocation of resources [6]. We represent a network slice as a SFC. A SFC is composed of an ingress edge node, a set of interconnected VNFs, and (eventually) an egress edge node. We represent each network slice as follows:

- *End-to-end delay* – between edge nodes;
- *SFC diagram* – list of edge nodes, VNFs and virtual links (which characterize inter-connectivity between VNFs);
- *Virtual resources* (processing and storage) required by each VNF;
- *Data rate* required for the virtual links.

Figure 1 shows an example of network slice allocation. In the bottom part, there are computing platforms and logical connections. In the upper part, there are three network slices. Note that an edge node is corresponding to a MEC platform.

Moreover, two types of network slices are considered:

- *eMBB* – characterized by a predominant bandwidth requirement;
- *URLLC* – characterized by a predominant bandwidth requirement.

We do not consider the mMTC and V2X use cases because both use cases are characterized by a predominant requirement on access connectivity (a massive number of devices has to be able to connect). However, in this paper, we focus on allocating resources in the edge and core networks and not in the access network (i.e., radio resources).

D. Related works

Due to space constraints, we limit our review to the works on resource allocation in network slicing that consider economic objectives [7]–[9]. Regarding the economic objectives, these works focus on maximizing the InP revenue because they do not consider an intermediate element, as the slice broker, which creates the network slices by using resources from multiple InPs. About the resource allocation, [7] allocates radio resources, [9] allocates computational resources, and [8] allocates both radio and computational resources. We do not allocate radio resources, but we focus on computational and network resources in the edge and core networks. Differently from previous works, we consider various requirements for various services belonging to URLLC and eMBB categories. One of the requirements is the end-to-end delay that has not been considered in previous works. To the best of our knowledge, our work is the first to consider multiple configurations available for each InP resource.

III. PROBLEM SOLUTION

A heuristic solution can find an approximate solution in less time when exact solutions are computationally expensive. A

Algorithm 1 Heuristic solution

```
Input: $\mathcal{S}$, $F$, $\mathbf{W}$, $G$, and $\mathbf{R}$;
Output: $\mathbf{A}$ and $\mathbf{K}$;
1: $\mathbf{Q} \leftarrow$ Random sorting of the slice request in $\mathcal{S}$;
2: $\mathbf{A} = \emptyset$;
3: $\mathbf{K} = \emptyset$;
4: for $i$ from 1 to $|\mathbf{Q}|$ do
5:     $(\mathbf{A}[Q[i]], \mathbf{K}) \leftarrow$ Allocate the slice request $Q[i]$;
6: end for
```

Algorithm 2 Allocate the slice request $Q[i]$: **MCA**

```
Input: $Q[i]$, $F^{Q[i]}$, $\mathbf{W}[Q[i]]$, $G$, $\mathbf{R}$, $\mathbf{A}$, and $\mathbf{K}$;
Output: $A[Q[i]]$ and $\mathbf{K}$;
1: $\mathbf{B} \leftarrow$ Create a list the feasible allocations of $F^{Q[i]}$ in $G$;
2: $\hat{\mathbf{a}} = \emptyset$;
3: $\hat{\mathbf{K}} = \emptyset$;
4: $\hat{p} = \infty$;
5: for $j$ from 1 to $|\mathbf{B}|$ do
6:     if Capacity constraints can be met in $\mathbf{B}[j]$ then
7:         $\mathbf{T} \leftarrow$ Update the selected configuration given $\mathbf{B}[j]$;
8:         $h \leftarrow$ Compute price of $\mathbf{B}[j]$;
9:         if $h < \hat{p}$ then
10:            $\hat{p} = h$;
11:            $\hat{\mathbf{a}} = \mathbf{B}[j]$;
12:            $\hat{\mathbf{K}} = \mathbf{T}$;
13:        end if
14:    end if
15: end for
16: $\mathbf{A}[Q[i]] = \hat{\mathbf{a}}$;
17: $\mathbf{K} = \hat{\mathbf{K}}$;
18: return $(\mathbf{A}[Q[i]], \mathbf{K})$
```

heuristic solution algorithm is proposed to solve the presented problem.

Algorithm 1 presents the heuristic solution. Following parameters are considered in the algorithm:

- $\mathcal{S}$ is the set of network slice requests;
- $F = (E \cup V, L)$ is a graph representing the SFC for one slice request, where E are the ingress/egress edge nodes, V are the VNFs, and L are the virtual links;
- $\mathbf{W}$ is representing the requirements for each slice request in $\mathcal{S}$: end-to-end delay, tuple (processing, storage) for each VNF, and data rate for each virtual link;
- $G = (M \cup D, C)$ is a graph representing InP topology, where M are the MEC platforms, D are the data center, and C are the logical connections;
- $\mathbf{R}$ is representing the resources for each configuration for each element in G: tuple (delay, data rate, processing, storage) for each computing platform and tuple (delay, data rate) for each logical connection;
- $\mathbf{A}$ is representing the VNF allocation for each slice request in $\mathcal{S}$;
- $\mathbf{K}$ is representing the selected configuration for each element in G;

Algorithm 1 creates a list of network slice requests that are randomly sorted and, then allocates each network slice request. The allocation of a slice request (see line 5 in Algorithm 1) can be implemented by using different strategies. For this purpose, we present two strategies: Minimum Cost Allocation (MCA) and Random Slice Allocation (RSA). Algorithm 2 presents MCA, a strategy that minimizes the costs associated with the allocation of the slice request. In line 1, the "feasible" means meeting the requirement on the end-to-end delay and being possible given the connectivity of G. In line 6, it is checked that, given $\mathbf{B}[j]$, it is possible to allocate $\mathbf{W}[Q[i]]$ allocation in G (the capacities are the maximum resources in $\mathbf{R}$, previous allocations $\mathbf{A}$ need to be considered). In line 7, the configurations are updated (from $\mathbf{K}$) by eventually increasing to the upper configuration. Note that if the residual capacity is enough, there is no need to increase the configuration. In line 8, the price is computed by considering the difference in price between the previous configuration $\mathbf{K}$ and the current one $\mathbf{T}$ for each element.

To compare the performance of MCA, we propose RSA. In RSA, the slices are allocated without considering the cost of the InP resources. In RSA, a random feasible allocation of $F^{Q[i]}$ to G is selected. If the selected allocation meets the capacity requirements, such allocation will be returned; otherwise, another random feasible allocation is selected, and so on.

IV. EVALUATION

To evaluate the performance of the proposed solution, we have created a simulation in C++. In the following, we first present the simulations settings and then discuss the simulation results.

A. Simulation settings

In this subsection, we characterize the InP and the network slice. InP is composed of computational and network resources, and the network slice is characterized by a SFC, which requires computational and network resources.

1) InP resources and prices: For InP resources, we consider computing platforms and logical connections. The computing platforms are further divided into MEC and DC. We assume that the network operator can provide a logical connection from every MEC to every DC (and vice versa) and between every DC. Instead, the MEC can have a logical connection between them only if they belong to the same cluster (i.e., they are close to each other). For both computing platforms and logical connections, the resources have discrete values and are grouped in "configurations".

TABLE I
CONFIGURATIONS FOR DC

Configuration	Processing [vCPU]	Storage [GiB]	ICR [Gb/s]	ICD [ms]	Price [€/h]
1	36	7×10^3	10	1	1.521
2	42	4×10^3	10	1	1.954
3	64	14×10^3	25	1	4.173

TABLE II
CONFIGURATIONS FOR MEC

Configuration	Processing [vCPU]	Storage [GiB]	ICR [Gb/s]	ICD [ms]	Price [€/h]
1	4	150	25	1	0.272
2	8	300	25	1	0.544
3	16	400	10	1	0.768

TABLE III
CONFIGURATIONS FOR LOGICAL CONNECTIONS INTERCONNECTING MEC

Configuration	Data rate [Gb/s]	Delay [ms]	Price [€/h]
1	1	1.3	0.205
2	10	1.9	0.288
3	10	3.2	0.260

TABLE IV
CONFIGURATIONS FOR LOGICAL CONNECTIONS INTERCONNECTING MEC AND DC

Configuration	Data rate [Gb/s]	Delay [ms]	Price [€/h]
1	1	1.5	0.164
2	4	2.7	0.247
3	8	3.5	0.288

TABLE V
CONFIGURATIONS FOR LOGICAL CONNECTIONS INTERCONNECTING DC

Configuration	Data rate [Gb/s]	Delay [ms]	Price [€/h]
1	10	2.8	0.247
2	20	2.1	0.342
3	25	4.7	0.288

Table I presents the list of configurations for DC. Each configuration is described by the amount of processing (in number of virtual CPUs), amount of storage, intra-connection data rate, intra-connection delay, and price. The intra-connection data rate and intra-connection delay are reported as ICR and ICD in the table. The configurations are taken from Amazon Elastic Compute Cloud (EC2) instances provided by Amazon Web Services (AWS) [10]. AWS specifies the intra-connection data rate between various EC2 computing platforms to vary between 5000, 10000, and 25000Mb/s, respectively. AWS does not specify the intra-connection delay. AWS supports low-latency computing; therefore, it is expected that the intra-connection delay to vary between 1 ms and 7 ms [11].

Table II lists configurations for the MEC. The configurations for MEC are different from cloud-service-provider owned DC. MEC has a multi-core CPU with low clock frequency for low power consumption. Moreover, MEC supports multi-threading, workload distribution, and location awareness. Hence, the MEC has hardware characteristics that do not exist when compared with cloud-service-provider-owned DC.

Table III lists the configurations for logical connections interconnect MEC. Each configuration is described by the capacity of the logical connections, delay, and the price. The configurations for logical connections, such as the capacity and price, are taken from [12]. The delay of the various logical connections on average varies between 2 ms and 5 ms [13].

Table IV lists configurations for logical connections that are interconnecting MEC and DC, and Table V lists configurations for logical connections that interconnect DC.

2) Network slices: Table VI lists 5G network slice categories services for vertical industries with their corresponding and associated SFC [14] [15]. The VNFs here in SFC refers to Network Address Translator (NAT), Firewall (FW), WAN Optimization Controller (WOC), Intrusion Detection Prevention System (IDPS), Video Optimization Controller (VOC) and Traffic Monitor (TM). The requirements of each VNF in terms of data rate, delay, and hardware (processing and storage) is calculated from [14].

Table VII, shows requirements for a video streaming slice belonging to the eMBB category for multimedia consumption. We consider ultra-high-definition video streaming with a data rate of 85Mb/s having a delay between 100ms and 500ms for 100 devices. Table VIII shows the requirements for real-time remote patient care and monitoring network slice belonging to the eMBB category. We consider the network slice with 100 devices with each device requiring a data rate between 1Mb/s and 10Mb/s and end-to-end delay of less than 100ms [16] [15]. Table IX shows the requirements for fault management in distributed power generation slice belonging to the URLLC category. The network slice supports 1000 devices and has an end-to-end delay of less than 30ms, with each device requiring a data rate of 1Mb/s. Table X shows the requirements for a robot tooling network slice belonging to the URLLC category. It has an end-to-end delay of less than 10 ms. Each robot requires a data rate of 10Mb/s to 50Mb/s [17].

B. Simulation results

In this subsection, we analyze the performance of the proposed algorithm by varying the number of computing platforms and slice requests. Note that we consider only one instance in each scenario because almost all the simulation settings are deterministic. The only parameter that can change is the location of the edge nodes for the slice requests. Given the almost full-mesh topology that connects the computing platforms, we assume that the impact of a different location on the result is negligible.

1) Varying computing platforms: In this evaluation, the total number of slice requests is 8. All slice requests are uniformly distributed among MECs. Moreover, slice requests

TABLE VI
5G NETWORK SLICE CATEGORIES

Services	Category	Service Function Chain
Video Streaming	eMBB	NAT-FW-TM-VOC-IDPS
Patient Monitoring	eMBB	NAT-TM-WOC-IDPS
Power Grid Monitoring	URLLC	TM-IDPS
Robot Tolling	URLLC	NAT-TM

TABLE VII
REQUIREMENTS FOR VIDEO STREAMING SLICE

Delay [ms]	300	
Virtual Links	Data rate [Mb/s]	
(E, NAT)	850	
(NAT, FW)	680	
(FW, TM)	680	
(TM, VOC)	680	
$(VOC, IDPS)$	510	
$(IDPS, E)$	510	
VNF	Processing [vCPU]	Storage [GiB]
NAT	2.4	4.7
FW	7.8	8.3
TM	2.3	4.1
VOC	8.3	8.3
$IDPS$	4.6	2.3

TABLE VIII
REQUIREMENTS FOR PATIENT MONITORING SLICE

Delay [ms]	100	
Virtual Links	Data rate [Mb/s]	
(E, NAT)	200	
(NAT, TM)	160	
(TM, WOC)	160	
$(WOC, IDPS)$	112	
$(IDPS, E)$	112	
VNF	Processing [vCPU]	Storage [GiB]
NAT	6.7	3.3
TM	6.6	3.3
WOC	3.3	9.7
$IDPS$	3.3	6.6

are requested equally such that the URLLC requests are the 50% of the total number of slice requests.

Figure 2 shows the total cost of SB, by varying the total number of computing platforms such that the MECs are always the 25% of the total number of computing platforms. The behavior of MCA is almost constant as the total cost for MCA varies between 2.2€/h and 2.8€/h. The total cost of RSA increases with concave function as the total cost varies between 5.6€/h and 16.6€/h. With a lower number of computing platforms, the total cost for MCA is 60% less than when compared to RSA. With a higher number of computing platforms, the total cost difference between MCA and RSA is 83%. MCA results in a slight variation in total cost with the increasing number of computing platforms, because MCA reduces the number of computing platforms allocated to each slice request. Instead, RSA results in increasing the total cost since RSA allocates each slice request to a larger pool of computing platforms.

TABLE IX
REQUIREMENTS FOR POWER GRID MONITORING SLICE

Delay [ms]	30	
Virtual Links	Data rate [Mb/s]	
(E, TM)	200	
$(TM, IDPS)$	200	
$(IDPS, E)$	200	
VNF	Processing [vCPU]	Storage [GiB]
TM	0.6	0.3
$IDPS$	3.3	0.6

TABLE X
REQUIREMENTS FOR ROBOT TOLLING SLICE

Delay [ms]	5	
Virtual Links	Data rate [Mb/s]	
(E, NAT)	500	
(NAT, TM)	400	
(TM, E)	280	
VNF	Processing [vCPU]	Storage [GiB]
NAT	16.7	8.3
TM	16.6	8.3

Figure 3 shows the total cost of SB when the number of the computing platform is always 28. By increasing the number of MECs, the total cost for the MCA increases slightly as it varies between 2.2€/h and 2.5€/h. However, the total cost of RSA decreases from 15.6€/h to 13.2€/h. With a lower number of MECs, the total cost for MCA is 85% less as compared to RSA and, with a higher number of MECs the cost difference is 80%. The slight increase in the total cost for MCA is due to the spread of slice requests to other MECs. Moreover, total cost decreases for RSA because while increasing the number of MECs, the number of DCs is reduced. This results in RSA to select a few DCs to allocate slice requests.

2) Varying slice requests: The performance of the proposed algorithm is evaluated while varying the number of slice requests. In this evaluation, the total number of the computing platform is 28, with MECs being 30% of the total number of the computing platforms. All slice requests are uniformly distributed among MECs.

Figure 4 shows the cost by SB by varying the total number of slice requests. URLLC requests are always the 50% of the total number of the slice requests. With an increasing number of slice requests, the total cost increases for both MCA and RSA. The increase in total cost for MCA is linear as the total cost for MCA varies between 1.4€/h and 5.8€/h. A similar trend is shown by RSA. However, the total cost of RSA varies between 8.9€/h and 31.3€/h. The increase in total cost for both RSA and MCA is because higher configurations of the computing platform are selected to accommodate new slice requests. In all the cases, MEC obtains a cost that is constantly 80% lower than RSA. The reason is that MCA uses a lower number of computing platforms to allocate the slice requests regardless of the number of slice requests.

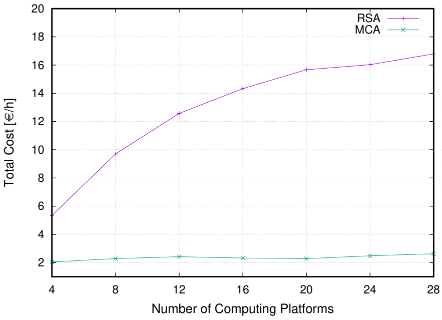

Fig. 2. Varying the number of computing platforms

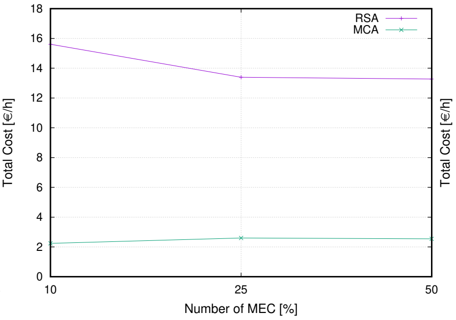

Fig. 3. Varying the percentage of MECs over the total number of computing platforms

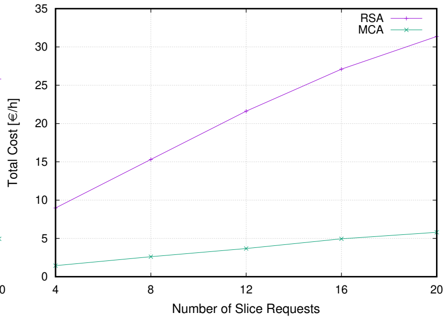

Fig. 4. Varying the total number of slice requests

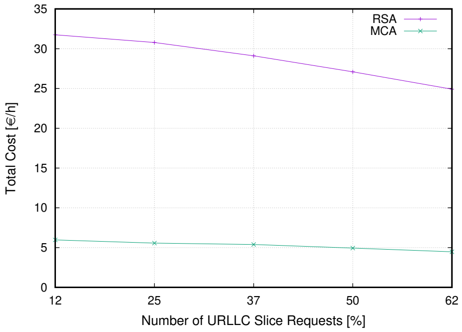

Fig. 5. Varying the percentage of URLLC requests over the total number of requests

Figure 5 shows the cost of varying URLLC slice requests. The total number of slice requests is always 16. Both MCA and RSA show a decrease in the total cost as the number of URLLC requests increases. The decreases in total cost for MCA is linear as the total cost for MCA varies between 5.9€/h and 4.4€/h. The total cost for RSA varies between 31.7€/h and 24.9€/h. This is because eMBB slice requests require a higher amount of resources (except for the delay) than URLLC slice requests. With a lower number of URLLC slice requests, the total cost for MCA is 81% less as compared to RSA. With a higher number of slice requests, the cost reduction of using MCA further increases by 1%. This is due to the characteristics of the slice categories. The eMBB slice types require a higher data rate and are expected to support a higher number of devices as compared to URLLC. URLLC slice types instead focus on high reliability and remarkably low delay. The eMBB slice types require more InP resources as compared to URLLC.

V. CONCLUSIONS

In this paper, we minimize the cost of the SB for buying resources from InPs to accommodate all slice requests while satisfying performance guarantees. Heuristic solution algorithm MCA is proposed and compared with RSA. We conduct evaluations to analyze the behavior of the proposed algorithm by considering a realistic context. The number and the composition of the computing platforms and the slice of requests are varied in the evaluation. From evaluations, we conclude that the cost reduction of MCA with respect to RSA is mainly affected by the number of computing platforms and varies from 60% to 80%. The total cost reduction is not affected by the composition of the computing platforms (MEC or DC), neither by the number or composition of the slices requests.

REFERENCES

[1] G. Nencioni, R. G. Garroppo, A. J. Gonzalez, B. E. Helvik, and G. Procissi, "Orchestration and Control in Software-Defined 5G Networks: Research Challenges," *Wireless Communications and Mobile Computing*, vol. 2018, p. 6923867, Aug. 2018, publisher: Hindawi.

[2] "System architecture for the 5g system (release 16), 3gpp," Mar 2020.

[3] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, and C. S. Hong, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1359–1374, 2020.

[4] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5g network slice broker," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32–39, 2016.

[5] G. Mirjalily and Z. Luo, "Optimal network function virtualization and service function chaining: A survey," *Chinese Journal of Electronics*, vol. 27, no. 4, pp. 704–717, 2018.

[6] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, 2014, pp. 7–13.

[7] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, and X. Costa-Pérez, "A machine learning approach to 5g infrastructure market optimization," *IEEE Transactions on Mobile Computing*, vol. 19, no. 3, pp. 498–512, 2020.

[8] M. Jiang, M. Condoluci, T. Mahmoodi, and L. Guijarro, "Economics of 5G Network Slicing: Optimal and Revenue-based allocation of radio and core resources in 5G," p. 25.

[9] L. Yala, P. A. Frangoudis, and A. Ksentini, "Latency and availability driven vnf placement in a mec-nfv environment," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–7.

[10] "Amazon S3 Simple Storage Service Pricing - Amazon Web Services." [Online]. Available: https://aws.amazon.com/s3/pricing/

[11] "Low-latency computing with AWS Local Zones – Part 1," Jun. 2020, section: Best Practices. [Online]. Available: https://aws.amazon.com/blogs/compute/low-latency-computing-with-aws-local-zones-part-1/

[12] Clare, "Top 41 Leased Line Providers: 2020 Price," businessfibre.co.uk. [Online]. Available: https://businessfibre.co.uk/leased-lines/

[13] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

[14] A. Chiha, M. Van der Wee, D. Colle, and S. Verbrugge, "Network Slicing Cost Allocation Model," *Journal of Network and Systems Management*, vol. 28, no. 3, pp. 627–659, Jul. 2020.

[15] ATIS, "IOT Categorization - Exploring the Need for Standardizing Additional Network Slices," ATIS, White Paper ATIS-I-0000075, 2019. [Online]. Available: https://api.govwhitepapers.com/wp-content/uploads/2020/06/ATIS-I-0000075.pdf

[16] HITInfrastructure, "IoT Sensors Critical to Successful Health IT Infrastructure," Mar. 2017. [Online]. Available: https://hitinfrastructure.com/news/iot-sensors-critical-to-successful-health-it-infrastructure

[17] A. Zahemszky, "5G E2E Technology to Support Verticals URLLC Requirements," NGMN Alliance, White Paper, 2020.