Network Slicing for Massive Machine Type Communication in IoT-5G Scenario

Rayner Gomes, Dario Vieira, Yacine Ghamri-Doudane, Miguel Franklin de Castro

Network Slicing for Massive Machine Type Communication in IoT-5G Scenario

Rayner Gomes
Federal University of Piauí
Picos, Brazil
rayner@ufpi.edu.br

Dario Vieira
EFREI - Paris
Paris, France
dario.vieira@efrei.fr

Yacine Ghamri-Doudane
Université de La-Rochelle
La-Rochelle, France
yacine.ghamri@univ-lr.fr

Miguel Franklin de Castro
Federal University of Ceará
Ceará, Brazil
miguel@ufc.br

***Abstract*—Network slicing is a key component of the envisioned 5G network. Slices are virtual networks purpose-built for tenants using a shared infrastructure. The slicing process is mathematically known as a virtual network embedding problem (VNE). Despite the plethora of VNE strategies in the literature, they do not take into account the fact that massive data transmission can be carried in a certain period. Embedding a large number of virtual networks to a real physical network over time is an NP-*Hard* problem, and it becomes more complex because of new considerations such as periodicity, amount of data and duration. Thus, we propose the NS4MIoT, a solution to allocate slices resources for each tenant, allowing it to be aware of each transmission's periodicity, amount of data, and duration. NS4MIoT is an approach to increase the quantity of requisition mapped in an envisioned 5G network for IoT massive communication. To validate our solution, we incorporate it into two different embedding algorithms. Furthermore, we compare the same algorithms with and without the NS4MIoT approach, and the outcomes demonstrate an improvement in the mapping rate in all cases when it is incorporated.**

I. Introduction

Fifth-generation mobile networks (5G) has the ambition to endure a range of services and applications and, among them, the Internet of Things is a particular case. In IoT the transmissions rise from a vast quantity of sensors and demands high-capacity networks. The 5G infrastructure must support the estimated 1,000-fold increase in mobile/fixed data traffic while promoting a reduction in energy usage [1]. 5G is the first mobile telecommunication that included the IoT concepts in its design since the beginning. Indeed, the relation between IoT and 5G is symbiotic, since 5G offers an infrastructure that allows paving the way to new IoT applications and, at the same pace, IoT presents new challenges to be faced in 5G.

A standard mechanism adopted in 5G to face the diversity of demands and distinct requirements of services and applications is the Network Slicing (NS). NS has been designed as a key enabler to allow 5G to handle IoT and other vertical markets. The main idea in NS is that a slice is a virtual network deployed for a specific proposal and an exclusive tenant. Using slice, the provider can adjust the use of its infrastructure resources (increasing of flexibility) and create new architectures instead of adopting the traditional architecture philosophy of "one-fits-all" (increasing of dynamicity). Slice has an overloaded definition and, according to [2], it is a composition of the adequately configured network functions, network applications, and underlying infrastructures that are bundled together to meet the requirement of a specific use case or business model.

5G has two new entities: Infrastructure-Provider (InP) and Service-Provider (SP). The InP owns and manages the substrate network (SN) (physical devices: switches, routes, links, so on), while the SP focuses on offering customized services to clients, and it can share many SN. NS is a service provided by a SP which has the objective to create a virtual network (VN) over the substrate network. Under the 5G vision, a virtual network is a slice, and each one is independent. A client requires a VN to a SP through a requisition named as virtual network request (VNR), which is composed by a set of virtual nodes, virtual links, services, and some expected QoS.

The process of mapping VNRs comprises a complex task to associate a set of virtual nodes and links to a set of physical nodes and links, taking into account all network constraints and resources availability. Mathematically, this mapping is known as virtual network embedding (VNE), and the mapping is a combinatorial problem. Effectively allocating the substrate network to VNRs is a vital problem in VNE. In summary, network slicing is the process of creating slices which are requested on-demand by a tenant through a VNR, and VNE is the method adopted to mapping a VNR to a physical substrate. Therefore, network slicing is a technique for resource provisioning and reuse. The main goal of a provider is to optimize its mapping process to increase the acceptance mapping rate (AR), that is, to carry out as many mapping as possible.

In [3], the authors state that a significant amount of devices from IoT applications can trigger a massive data transmission, which can diminish network utility if not carefully handled. Therefore, this unfortunate situation is a current challenge to be faced. Researchers must investigate and design solutions in many study fields. A broad survey in massive machine type communication (MMTC) was carried out by [4], in which the authors grouped related work into three fields to deal with MMTC: (a) pull-based schemes, (b) massive clustering communication and (c) data aggregation (DA). Among them, DA has attracted a lot of research due to its robust ability to manage massive access and network congestion.

The DA ability derives from the aspect that transmissions can be started in a given period. Despite the advantages of DA and the efforts related to the 5G network slicing [5]–

[10], to the best of our knowledge, there is not a VNE algorithm which has included the DA aspects to mapping slices for specific MMTC cases; however, its contemplation can lead to more optimized resource allocation. Therefore, in this work, we present the network slicing for massive IoT communications (NS4MIoT), which is an approach to conduct the mapping process of virtual networks to physical nodes taking into account the inherited aspects of VNE and the discerning features of MMTC-IoT applications such as periodicity, amount of data and duration.

The main contributions of this work are: (1) NS4MIoT considers three new specificities of MMTC: transmission periodicity, amount of data, and slice duration. The NS4MIoT approach is applied during the resource allocation phase of a network slicing process, and it permits a better allocation of resources, having improvement of the acceptance mapping rate as a consequence; (2) we propose a formal description of network slicing for MMTC; (3) we developed two algorithms based on greedy and genetic algorithm (GA) heuristics to performs NS considering the constraints of MMTC. (4) To validate our solution, we adapted the previous two algorithms with the NS4MIoT; thus, we derived two more versions to evaluate the performance of VNE algorithms with and without NS4MIoT. (5) we compared our approach with other solutions, and the outcomes show the improvement in the AR.

The remainder of the paper is organized as follows. Section II presents the main architecture's elements, the modelling and the problem description. Section III describes the built scenarios regarding the data aggregation solution as the pivot of slices to MMTC. The evaluation results and conclusions are presented in Sections IV, and VI respectively.

II. Modeling and Problem Description

Massive communications (MC) are habitual behaviors in smart scenarios such as smart-cities, smart-homes, and smart-transportation, which makes them intrinsically connected with IoT networks. Dealing with MC brings some challenges to actual networks, and should be managed by 5G proposals. The International Telecommunication Union (ITU) has determined that the addressing of architectural issues coming from the massive number of devices has a high priority [11].

The main problem in supporting the intermittent short burst traffic is that it has to go through the full signaling procedure, which causes the waste of battery life, spectrum, and network capacity [11], [12]. The noticeable feature is that the data of IoT sensors can be stored in *IoT Gateways*, with which IoT Gateways, transmissions can be planned since they are delay-tolerant, and providers can prepare communications at a proper time. The use of *IoT Gateways* is the standard procedure to enhance the scalability and deal with interoperability of heterogeneous access links [13]. IoT Gateways perform the role of data aggregator [14]. Data aggregation (DA) is widely adopted to effectively address the massive connectivity and latency requirement for Machine Type Communication (MTC) applications. DA is the process of gathering the data from multiple IoT devices sited at the extremity of the network to eliminate redundant transmissions and provide fused data [14].

Considering a network with IoT Gateways, we can divide the nodes into three categories: collector, transport, and sink. Collector node is a generic name, and it is used to store data from many sensors. Collector nodes must be on the access network area of Machine Type Communication Devices (MTCD), which is responsible for aggregating data. IoT Gateways perform three network functions [15]: (i) share data from distributed sensors; (ii) connect with sensors with divergent wireless and wire transmission technologies; and (iii) aggregate data depending on the business roles. The transport category covers all nodes belonging to the path that connects a collector node to a sink node. The third node category is sink node, which is the final point where transmission is intended.

A tenant requests a slice through a VNR, which is a structured document that specifies all the features of the desired VN. The mapping process associates a virtual network, described by VNR, to a real infrastructure. Each virtual network has a customized topology formed by virtual nodes; a node can be a collector, transport, or sink node.

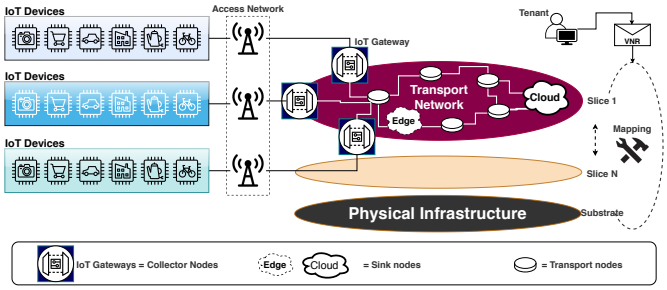

Fig. 1. Three types of nodes in MMTC use case.

Network slicing is a problem mathematically known as Virtual Network Embedding (VNE), and it has received exhaustive attention from researchers. The basic VNE is an *NP-hard* problem, and it can be proven by reducing it to *the multiway separator problem* [16], [17]. Even when node mapping from virtual to physical is given, the problem of optimally allocating a set of virtual links to a single physical path reduces to *the unsplittable multicommodity flow problem*, which is also *NP-hard* [18]. Work [19] describes every reduction of VNE problems to Knapsack problems, and it enriches the literature with some theorems and proofs.

A network infrastructure (Table I) is represented by a non-directed graph denoted by $G = (V, E, \beta)$, in which G represents a physical network, V is a set of vertex which $v_i \in$ V, and represents a real device, an edge $e_i \in E$, where E is a set of edges, and it represents a real link. Each vertex and edge is characterized by capacities $\beta \mathrm{v}_i^c \geq 0$ and $\beta \mathrm{e}_i^b \geq 0$. In the *online operation* of the network, $\beta \mathrm{v}_i^r$ and $\beta \mathrm{e}_i^r$ can play the role of residual capacity, which is the remaining resource of the nodes or links after taking out the current utilization.

There are three types of vertex. The collector vertex represents a device placed in the access area, and it is placed

where the data is stored for a certain period, in a real infrastructure represented by an IoT Gateway. The collector capacity defines the quantity of data an IoT Gateway can store ($\beta \mathrm{v}_i^c$). The transport vertex allows connecting a collector to a sink vertex. The sink vertex represents a device which is the final destination of data in a real network. To differentiate these type of vertex/nodes/devices, we use a function $\beta \mathrm{v}_i^t$ that returns the values 's' to sink, 't' to transport or 'c' to collector.

There is only one edge between two vertices. The region is represented by $\beta \mathrm{v}_i^{re}$, which can define a geographic localization. A domain $\beta \mathrm{v}_i^d$ indicates an organization to which nodes and link belongs, and an organization can act in many regions.

A virtual network is denoted by an undirected graph $H^i = (N, L, \delta)$ with virtual nodes $n_i \in N$, and links $l_i \in L$. Each collector node has a total of store capacities $\delta \mathrm{n}_i^a$. Each virtual node n_i can be embedded in one physical node from a set of physical nodes V. A virtual node is associated with only one physical node, and a virtual link is only associated with one physical path. A physical path is a set of physical links.

The embedding of H into G consists of a mapping as follows: (i) each virtual node n ∈ N to a physical node $v \in V$; (ii) virtual link (n_i, n_j) to a loop-free physical path, connecting physical nodes v_i and v_j to which the virtual nodes n_i and n_j have been mapped. For a slice user, a slice behaves like a physical network, and no difference should be noticed.

TABLE I
LIST OF SYMBOLS.

Symbols	Definition
$G = (V, E, \beta)$	physical network substrate; V is a set of nodes; E is a set of edges
$v_i \in V$	vertex i
$e_i \in E$	edge i
βv_i^c, βv_i^d, $\beta v_i^r e$, βv_i^t	capacity, domain, region, and type of vertex v_i
βe_i^b, βv_i^r, βe_i^r	bandwidth of edge e_i, residual capacity of vertex v_i, residual capacity of edge e_i
$H = (H^i, \ldots, H^m)$	set of requisitions
$H^i = (N, L, \delta, id, d)$	massive slice requirement
$n_i \in N$	virtual node
$l_i \in L$	virtual link
N	set of virtual nodes
L	set of virtual links
$l_i == (n_i, n_j) \mid [n_i, n_j] \in N$	a virtual link is a pair of virtual nodes
δl_i^b	bandwidth demanded by virtual link
$\delta n_i^t == t$	type of n_i, where t is $sink \vee transport \vee collector$
$\delta \mathrm{n}_i^p$	periodicity of transmission from n_i
$\delta \mathrm{n}_i^c \Rightarrow n_j$	collector node associated with a sink node
$\delta \mathrm{n}_i^a$	amount of data stored in n_i
id	every slice has a unique identifier
d	the maximum cycle to remain (duration)

The VNE is a process to associate each virtual node to a physical node and virtual links to physical links respecting that the sum of the demanded virtual resources is less than of equal the available physical resources. The VNE must maintain the control of used resources. Moreover, VNE algorithms have the goal of finding a feasible embedding with the highest acceptance ratio (AR), also called an embedding factor, which is the amount of successful requested mapping.

Duration d represents the time in which a slice must exist. The symbol $\delta \mathrm{n}_i^p$ is the period of time to perform a transmission, and this information is used by NS4MIoT to schedule the transmission regarding all links shared by the VNRs (Fig. 2). The symbol $\delta \mathrm{n}_i^a$ denotes the amount of data to be carried by a slice in each $\delta \mathrm{n}_i^p$. The parameter δl_i^b is not added into VNR, and it is calculated dividing δn_i^a by δn_i^p

VNE is a process of mapping a VNR to a given physical substrate. Let σ be a mapping function, G a substrate and H a set of VNRs where H^i is a request from H (Table I). The main goal can be defined as (1), and this equation means we seek to maximize the quantity of mapping (σ) in current cycle (c). Constraints of map processing are defined in Table II.

$$max \sum_{i}^{H} \sigma(H^i, G, c) \mid c \in [c_i, ..., c_j] \tag{1}$$

VNE maps l_i in $\phi(l_i)$ (see Table II) considering the amount of data that a collector node must send to its respective sink node into a slot of time. The $\beta\phi(l_i)^b$ must be enough to carry all data into a period. The collect node stores an amount of data and needs to transmit all the data to its respective sink node in a sequence of cycles. In case of a broadcast, multicast, or unicast communication, one collector node can be associated with one sink node, and one sink node can be associated with one or more collector nodes. A cycle is shortest period of time in which an infrastructure can manage a slice. In a practical situation, a cycle can denote n time units (tu). The slice schedule manages the creation of slices in accordance with the arrival of VNRs.

TABLE II
CONSTRAINTS

Constraints	Meanings
$H^i(id) \neq H^j(id) \forall H \in VNRs$	two requisitions from the set of requisitions VNR do not have the same id
$\beta \mathrm{v}_i^t == \delta \mathrm{n}_i^t$	the type of virtual node and real node must be the same
$\beta \mathrm{v}_i^d == \delta \mathrm{n}_i^d$	the domain of virtual node and real node must be the same
$\beta \mathrm{v}_i^r \geq 0$	the residual node capacity must be greater than or equal to zero
$\beta \mathrm{e}_i^r \geq 0$	the residual link capacity must be greater than or equal to zero
$\phi l_i \neq \emptyset$	let ϕ be a path mapped by a virtual link l_i, where a path is set of $[e_n, ..., e_m] \in V$ to connect a collector node to sink node
$\beta\phi {l_i}^b \leq \delta\phi {l_i}^b$	a bandwidth requested by a virtual link must be less than all bandwidth between all physical links mapped
$\delta n_i^c == n_j \Rightarrow \delta n_i^t == collector \wedge \delta n_j^t == sink \wedge n_i \neq n_j \wedge [n_i, n_j] \in N$	each collector node has an associated sink node
$\delta l_i == (n_a, n_b) \vee (n_b, n_a) \iff \delta(n_a, n_b) == \delta(n_b, n_a)$	a virtual link is defined by a tuple of two virtual nodes, where $n_i \neq n_j$
$\lvert l_i \cap L \rvert = 1$	between two virtual nodes there is at most one virtual link
$\beta\phi(l_i)^b \leq \delta n_i^a / \delta n_i^p$	the path between collector node and sink node must be enough to carry all data into of a period

To illustrate the core of the problem faced in this project, take a link (x,y) as the member of three virtual links of requisitions: VNR_1, VNR_2 and VNR_3. Let $\beta_1 = \beta, \beta_2 = \frac{\beta}{2}, \beta_3 = \frac{\beta}{2}$ be a bandwidth required respectively by these requisitions, and β denotes the total of remained bandwidth

of link (x,y). if we do not consider the periodicity of these requisitions, it is impossible to map all requests without corrupting the limit of bandwidth of the link (x,y). At first glance, it appears to be a designation or activity selection problem, however, it is not because the periodicity of transmission does not have a fixed initial and final time. These values can be changed by the provider, but a periodicity must be assured.

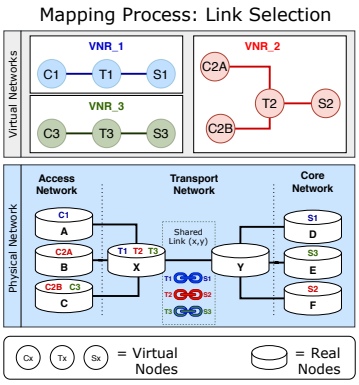

Fig. 2. Shared Link by multiple VNR.

For MMTC, some transmissions can be delayed or advanced as long as their periodicity is respected. We consider network infrastructure with IoT Gateways. An IoT Gateway controls the links, and it can store and prioritise the streams in each of them. Our solution works together with the VNE algorithm during the VNR acceptance process. Additionally, in the same link, all VNRs with the same periodicity (p) can be executed sequentially without the possibility of a time conflict. In the same link, any transmission with a periodicity different from p will eventually have a conflicted time slot. The sequencing of transmissions saves bandwidth consumption for any link which has VNRs with the same periodicity. We consider network infrastructure following Software-Defined Networking (SDN) mechanisms [20]. Therefore, there is a controller which has control of all links. The SDN allows to obtain a holistic vision of a whole network. Besides underpinning our premises, the solution calls for all IoT Gateways to be informed about sequential of transmissions. SDN switches do not have a buffer memory to store-and-forward all packets coming from MTCD [21]; thus, coordinating IoT Gateways plays an important rule in our solution.

The solution is performed while each virtual link is selected for a real link. We can obtain the intersections of all links which are disputed by the VNRs. After we have discovered all VNRs with links that have intersections, we can detect if there is a possibility of optimization. For all links that have no dispute with any requisition, the resource allocation can be performed trivially, that is, the link resources' capacity is diminished, and the real link is mapped to the virtual link.

Otherwise, for each link disputed by various VNRs, two sequential phases are executed. The first phase is carried out in the following sequence: **(a)** group all requisitions based on their periodicity, where $g^1, g^2, \ldots, g^n \in G$, g denotes a group, and n is the periodicity of requisitions; **(a1)** if the sum of bandwidth demanded in a group is greater than the link's capacity (β), then remove the requisition with the greatest demands; **(a2)** repeat (a1) until the sum of bandwidth demands is less than or equal to the link's capacity; **(b)** for each group, retrieve the maximum bandwidth requested, and for that, recover from the group the VNR with the greatest bandwidth requested; **(c)** calculate the score of each group, which is a tuple composed by $\tau = (t, b, l)$, where t means the total of requisitions in the group, b means the maximum bandwidth required for all requisitions of the group, and l means the total of bandwidth remained ($\beta - b$). At the end of every sequence (a) to (c), we have $T = \{\tau^1, \ldots, \tau^n\}$.

The second phase is the selection process, which has the goal of allowing the highest possible quantity of VNRs, and it is carried out in the following sequence: **(a)** order T by values t considering all $\tau \in T$; **(b)** select a τ^x that represents a group g^x that has the greatest t; **(b1)** if there is still a tie, select the τ with the lesser b, if there is a tie, select one randomly; **(c)** decrements the value b stored in t^x from the link bandwidth **(d)** accept all VNRs of group g^x; **(e)** remove g^x from G; **(f)** as the bandwidth of the link was reduced, repeat the process (a) to (c) as mentioned in the previous paragraph; however, disregarding all requisitions in the g^x; **(g)** repeat the sequence (a) to (e) until $|T| \leq 0$. All steps realized in this section can be performed in a linear time, $\Theta(|H|\dot{|}E|)$.

III. Artefacts and Scenarios for Evaluation

The hardware used has one CPU i7-7700HQ with quad-core, each core with two threads, with max frequency at 3.8GHz, 6MB cache; 8GB RAM of 2,400MHz; SSD 500MB M.2 with sequential read and write performance levels of up to 3,500MB/s and 3,300MB/s, respectively. The hardware's features are irrelevant to increase or diminish the accepted ratio (AR) of requisitions mapped. Any hardware enhancement will influence only the processing time and not the quantity of mapping success. The hardware specification is timely to show how much our solution impacts the execution time.

The datasets were created using two processes. The first one aimed to select nodes and edges, and the second one aimed to add collector nodes and label the transport and sink nodes. Two primary datasets were created, and two new versions were derived from them, each one changing the link capacities, totalling four datasets: 1A, 1B, 2A, 2B.

The datasets were provided by the website the Internet Topology Zoo[1]. The topology of Dataset 1 was built using the BT-Europe model. Once the dataset creation processing is finished, it comprises 112 nodes distributed in 90 collectors, 18 transports and 4 sink nodes, and with 125 links. Dataset 2 was based on the collection of 261 datasets from the same site. Its topology was build using all datasets with nodes, latitude and longitude from the USA. After the filter process, it is composed of 1,725 collectors, 345 transports, 68 core nodes, a total of 2,138 nodes, and 2,395 links.

The second dataset building process consists of labelling nodes and creating new access nodes into the dataset. Transport and core label were oriented based on the degree's node

[1]www.topology-zoo.org/, accessed at May 15, 2020

generated in the first process. All nodes with a degree greater than the degree's median multiplied by 1.5 are tagged as core, otherwise as transport. All transport nodes are the root of a region, and five access nodes were added to each region.

The sequence of tests was executed using two different datasets, which differ from each other in the variation of bandwidth and node capacity. The existence of these datasets consist of the attempt to make the solution search harder.

The Gaussian distribution was used to select the nodes storage capacity and link bandwidth values. The Dataset 2 has a median node capacity of 48; there are 1,062 nodes with a capacity of less than 48 and 1,076 nodes with a storage capacity of greater than or equal to 48. There are 778, 807 and 810 links with 3, 4, 5 of bandwidth respectively.

There are two groups of three sets of VNRs: (i) set 1 has 50 requisitions; (ii) set 2 has 100 requisitions; and (iii) set 3 has 200 requisitions. The sets in groups 1 and 2 are based on Dataset 1 and Dataset 2, respectively. Each set is kept the same for each different mapping algorithm. One VNR is composed of (a) VNR identification, (b) virtual nodes demands (vnd), and (c) links. Each VNR has one unique identification. A vnd is a sequence of virtual nodes with their demands.

In the VNR description, the field links represent the virtual network topology. The topology formation is guided by the rules: (a) there is only one link between two nodes; (b) all collector nodes are associated with a sink node; (c) a sink node can be associated with one or more collector nodes; (d) the data from a collector node can reach its destiny sink node through one or more transport nodes. There is no bandwidth's link information, and the necessary bandwidth is calculated by the algorithms to convey all data. These link's features are another particularity regarded and embedded in our work, and adapted to MMTC.

IV. Use Case and Evaluations

The evaluations were performed using four datasets. Dataset A is Dataset 1 with its bandwidth's links equal to 100. Dataset B is the Dataset 1 described in the same table, but with its bandwidth's links equal 10. Dataset C is Dataset 2 with its bandwidth's links equal to 100. Finally, Dataset D is Dataset 2 described in the same table with its bandwidth's links equal to 4. Keeping the same topology and changing only bandwidth aims to make the searching process more difficult.

For each dataset, one algorithm is executed with three sets of requisitions. Sets 1, 2 and 3 represent distinct vnr-datasets with differences regarding the number of requisitions, size of virtual nodes, topologies and distribution of storage demands. The sets have 50, 100, and 200 requisitions respectively. The same set is used for different algorithms, thus, we can guarantee the efficiency of each algorithm once the dataset and requisition are kept equal for different algorithms.

Each evaluation is carried out as follows: **(a)** a set of requisitions is recovered; **(b)** an embedding algorithm is executed; **(c)** the accepted VNRs are mapped or released based on the embedding algorithms; **(d)** some resources are released based on the duration of each requisition; and **(e)** repeat the steps (a) to (d) until there are no requisitions to be served.

Table III details the summary of results obtained after the execution of each algorithm 10-fold, and it shows the results using datasets A, B, C and D respectively. Consequently, each algorithm was performed 120 times, using three different requisition sets and four variations of datasets. In total, we have 480 executions.

We denote the VNE algorithms as follows: a) VNA-1, the greedy approach to VNE; b) VNA-2, the greedy approach with the NS4MIoT approach embedded; c) VNA-3, the genetic algorithm approach to VNE; and d) VNE-4, the genetic algorithm with the NS4MIoT approach embedded. Thus, VNA-2 and VNA-4 are the VNA-1 and VNA-3 updated with NS4MIoT approach. In this way, we demonstrated that our solution could be embedded in different approaches.

The assessment is made comparing the number of VNRs mapped split in Maximum (Max), Minimal (Min), and Average VNRs Mapped, following the goal of work (1). The average VNRs Mapped is an integer value, greater than zero, which is the number of VNRs mapped in a slot of time (cycle). The quantify of memory or CPU is not presented because they have no interference in the result. What we aim to increment the number of requisitions mapped. The table has the column time to show the time spent by each execution.

TABLE III
Summary of Results.

Data-set	VNR Set	Approach	Max VNRs Mapped	Min VNRs Mapped	Average VNRs Mapped	Time (s)
A	Set 1	VNA-1	8	6	7	0.97
		VNA-2	40	39	45	2.15
		VNA-3	18	13	14.9	14.9
		VNA-4	48	48	48	37.09
	Set 2	VNA-1	18	15	16.5	1.97
		VNA-2	68	65	66.5	3.79
		VNA-3	34	26	29.85	25.42
		VNA-4	91	91	91	101.34
	Set 3	VNA-1	23	22	22.5	2.84
		VNA-2	51	51	51	7.44
		VNA-3	38	31	34.15	25.88
		VNA-4	54	54	54	198.142
B	Set 1	VNA-1	44	35	39.5	1.13
		VNA-2	50	50	50	1.26
		VNA-3	44	32	38.2	11.21
		VNA-4	50	50	50	23.36
	Set 2	VNA-1	89	81	85	2.40
		VNA-2	97	97	97	2.51
		VNA-3	85	67	77.3	19.42
		VNA-4	97	97	97	44.56
	Set 3	VNA-1	59	57	58	3.47
		VNA-2	59	59	59	3.52
		VNA-3	59	56	57.7	24.20
		VNA-4	60	60	60	71.50
C	Set 1	VNA-1	6	5	5.5	14.73
		VNA-2	11	11	11	30.59
		VNA-3	11	6	7.95	223.78
		VNA-4	16	16	16	771.17
	Set 2	VNA-1	13	12	12.5	30.42
		VNA-2	22	22	22	128.51
		VNA-3	20	17	18.65	398.01
		VNA-4	33	33	33	1736.13
	Set 3	VNA-1	25	24	24.5	41.45
		VNA-2	42	42	42	313.96
		VNA-3	44	38	40	714.18
		VNA-4	64	64	64	3743.31
D	Set 1	VNA-1	10	8	9	9.98
		VNA-2	14	14	14	15.48
		VNA-3	14	7	9.9	161.24
		VNA-4	17	17	17	506.44
	Set 2	VNA-1	25	24	24.5	22.26
		VNA-2	32	32	32	48.05
		VNA-3	28	18	23.5	344.06
		VNA-4	35	35	35	1,267.95
	Set 3	VNA-1	46	42	44	30.79
		VNA-2	59	59	59	91.34
		VNA-3	53	44	47.4	510.78
		VNA-4	65	65	65	2,196.49

VNA-1 is a mapping based on a greedy approach. For

each VNR, a real node is selected with more resources for each virtual node, and after selecting all virtual nodes, a path connecting collector nodes and sink nodes based on the topology of the virtual network is discovered. Table III evidences that VNA-1 is the fastest in all evaluations.

VNA-2 is VNA-1 updated with the NS4MIoT solution. As a consequence, VNA-2 obtained 181,25%, 105%, 58% of gains on acceptance ratio (AR) in sets 1, 2, and 3 respectively, considering all datasets. These gains have added an overhead increasing the execution time in 65,81%, 133,82%, and 255,27%, respectively.

VNA-3 is based on a genetic algorithm (GA), which is a heuristic solution. This heuristic was chosen because it has the advantages [22]–[24]: (i) allowing the search for an acceptable solution in a wide search solution space; (ii) not restricting the search process to local search spaces; (iii) being flexible enough to be adapted to various scenarios.

VNA-3 follows the fundamental steps of a GA: (1) population creation; (2) population offspring; (3) population mutation; (4) repeat steps 1-3 n times; and (5) recover the better individual. Our chromosome is a sequence of a possible map, and our fitness function returns the quantify of VNRs mapped. The population creation is created using VNA-1. As expected, VNA-3 tends to obtain better results than VNA-1. VNA-3 has a gain of 40,80%, 29,24%, and 30,56% in set 1, 2, and 3 based on the same datasets when compared with VNA-1.

VNA-4 is VNA-3 updated with the NS4MIoT solution. As a consequence, VNA-4 obtained 106,82%, 89,06%, 39,81% of gains on AR in sets 1, 2, and 3, respectively, considering all datasets. These gains have a price, so the NS4MIoT adds an overhead increment of time in 179,00%, 258,21%, and 403,81% sets 1, 2 and 3, respectively, considering all datasets.

The most relevant columns of Table III are (a) Average VNRs Mapped and (b) Time. The number variation of VNRs mapped on VNA-1 and VNR-2 is due to its randomized choices when the VNA-Greedy had to choose one node among various with the same capacity. Depending on the selected node, the link will be different. The difference of results of VNA-3 and VNA-4 is occasioned by the uncertainty of AG, a typical effect of heuristics algorithms. All evaluations using NS4MIoT (VNA-2 and VNA-4) obtained greater results when compared with their competitor VNA-1 and VNA-2, thus, their number of accepted requisitions were greater. Fig. 3 illustrates the differences of average VNRs mapped between the algorithms with and without the NS4MIoT approach.

V. Related Work

Table IV presents the main contributions of related works and highlights our contributions. Work [14] is the most similar to our work. Its scenario is based on DA to deal with massive connectivity. Radio Access Network aspects are out of context. Similarly, our considerations are set from the data-aggregator devices to the core of the network. Also, we share the idea that the location of aggregator devices is a constraint when taking into account the end-to-end aspect of slices. Unlike our work, work [14] does not consider the periodicity of transmission,

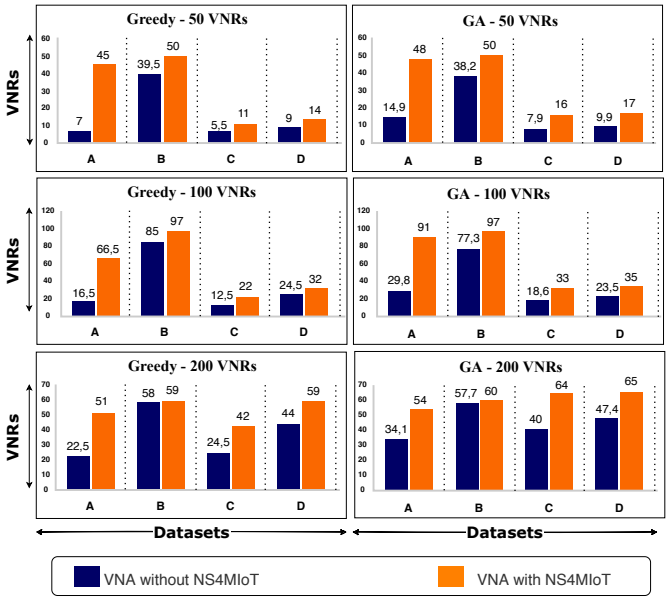

Fig. 3. Results of the four algorithms using four datasets

and it does not aim to improve average VNR acceptance ratio; instead, it aims to reduce latency.

TABLE IV
Related Work

Ref.	Main Contribution	MTC	DA	Period
[25]	the first work to apply Artificial Bee Colony Meta-Heuristic in VNE problems.	No	No	No
[26]	based on Algorithm Genetic, its fitness function aims to maximise the quantity of requestS which can be accepted together.	No	No	No
[27]	the first work to leverage SDN to probe resource usage in the underlying infrastructure in a thin grained manner.	No	No	No
[14]	the most similar to our work, as it also regards the data aggregation method. It does not aim to improve the VNR acceptance ratio.	Yes	Yes	No
[24]	investigates wireless resource allocation scheme for the provision of point-to-point and point-to-multipoint end-to-end virtual links with bandwidth requirements.	No	No	No
[28]	a genetic algorithm in parallel to reduce processing time to embedding VNRs.	No	No	No
[29]	avoids using traditional node rank to select nodes, and performs node and link selection at the same time. The algorithm is inspired by a dynamic programming model.	No	No	No
NS4-MIoT	embedding process regarding the periodicity of transmissions.	**Yes**	**Yes**	**Yes**

VI. Final Considerations and Future Works

Overprovisioning resources implies less profit for the provider when new slices cannot be created due to resource depletion. The results revealed that our approach obtained a better rate of mapping when embedded in the algorithms. We have embedded the NS4MIoT approach in two distinct mapping algorithms, obtaining two new ones. We performed exhaustive tests with 480 evaluations. We used three sets of requisitions, with 50, 100 and 200 VNRs, and four different datasets. In all assessments, our approach obtained better results, as it improves the algorithms to map more requisitions.

Although we showed the benefits of our solutions, it comes with an increase in run-time. This cost can be faced by enhancing computational capacities. It is crucial to highlight that the strengthening of computational capabilities does not improve the quantity of mapping, which means that the average VNRs mapped will not change, and only the run-time can be reduced. However, the goal of slicing providers is to accept as many requisitions as possible. To minimise the run-time incremented by our approach, parallel and distributed programming can be applied, and it is envisioned as future work.

NS is a powerful way to handle complex application's requirements. It is a potential source of profit for the infrastructure provider; thus, as future work, we can consider financial costs to decide when the creation of one slice is more profitable. All the researched works bring the perspective of providers; however, bringing the financial costs from the tenants' perspective is a prominent opportunity.

Real slice management takes time. Slice deployment involves a workflow of actions that trigger a set of services, such as: (i) reconfiguring routes; (ii) updating flowtables in SDN switches; (iii) starting services associated with each slice; and other services. All these actions take time, and time actions are not easily deterministic, since each administrative domain can use different hardware and software platforms. Therefore, as future work, we will consider the time to deploy a slice in each domain to decide the best real path and endpoints to be associated with a slice.

Currently, NS4MIoT does not include services. A set of services can be demanded into a VNR. As a VNR in Network Slice as a Service (NSaaS) describes a virtual network to be used for an individual business market, it is highly typical that a set of services is also requested. Thus, as future work, we will consider that a VNR also includes a set of services.

REFERENCES

[1] A. Gupta and R. K. Jha, "A Survey of 5G Network: Architecture and Emerging Technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.

[2] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network Store," in *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture - MobiArch '15*. New York, New York, USA: ACM Press, 2015, pp. 8–13.

[3] S. Bera, S. Misra, and A. V. Vasilakos, "Software-Defined Networking for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 4662, no. c, pp. 1–15, 2017.

[4] T. Salam, W. U. Rehman, and X. Tao, "Data Aggregation in Massive Machine Type Communication: Challenges and Solutions," *IEEE Access*, vol. 7, pp. 41 921–41 946, 2019.

[5] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, N. Sastry, O. Holland, S. Tayade, B. Han, D. Bega *et al.*, "Network slicing to enable scalability and flexibility in 5g mobile networks," *IEEE Communications magazine*, vol. 55, no. 5, pp. 72–79, 2017.

[6] H. Zhang, N. Liu, X. Chu, K. Long, A. H. Aghvami, and V. C. Leung, "Network Slicing Based 5G and Future Mobile Networks: Mobility, Resource Management, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 138–145, 2017.

[7] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[8] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.

[9] T. Taleb, B. Mada, M. I. Corici, A. Nakao, and H. Flinck, "PERMIT: Network Slicing for Personalized 5G Mobile Telecommunications," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 88–93, 2017.

[10] S. Sharma, R. Miller, and A. Francini, "A Cloud-Native Approach to 5G Network Slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 120–127, 2017.

[11] International Telecommunications Union, "Report on Standards Gap Analysis," 2016. [Online]. Available: {https://goo.gl/qXlsm9}

[12] R. Morabito and N. Beijar, "A framework based on sdn and containers for dynamic service chains on iot gateways," in *Proceedings of the Workshop on Hot Topics in Container Networking and Networked Systems*, 2017, pp. 42–47.

[13] M. M. Alam, H. Malik, M. I. Khan, T. Pardy, A. Kuusik, and Y. Le Moullec, "A survey on the roles of communication technologies in iot-based personalized healthcare applications," *IEEE Access*, vol. 6, pp. 36 611–36 631, 2018.

[14] Y. Xu, G. Feng, L. Liang, S. Qin, and Z. Chen, "MTC data aggregation for 5G network slicing," *2017 23rd Asia-Pacific Conference on Communications: Bridging the Metropolitan and the Remote, APCC 2017*, vol. 2018-Janua, no. 61631004, pp. 1–6, 2018.

[15] Z. Zhu, R.-G. Huang *et al.*, "Study on the iot architecture and access technology," in *2017 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES)*. IEEE, 2017, pp. 113–116.

[16] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *Computer Communication Review*, vol. 38, no. 2, pp. 19–29, 2008.

[17] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[18] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The Algorithmic Aspects of Network Slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 2017.

[19] H. Wu, F. Zhou, Y. Chen, and R. Zhang, "On Virtual Network Embedding: Paths and Cycles," *IEEE Transactions on Network and Service Management*, vol. 4537, no. c, pp. 1–14, 2020.

[20] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167, 2020.

[21] A. Mondal, S. Misra, and I. Maity, "Buffer size evaluation of openflow systems in software-defined networks," *IEEE Systems Journal*, vol. 13, no. 2, pp. 1359–1366, 2019.

[22] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," *IEEE/IFIP NOMS 2014*, 2014.

[23] B. Han, J. Lianghai, and H. D. Schotten, "Slice as an Evolutionary Service: Genetic Optimization for Inter-Slice Resource Management in 5G Networks," *IEEE Access*, vol. 6, no. c, pp. 33 137–33 147, 2018.

[24] L. Chen, S. Abdellatif, A. F. Simo Tegueu, and T. Gayraud, "Embedding and re-embedding of virtual links in software-defined multi-radio multi-channel multi-hop wireless networks," *Computer Communications*, vol. 145, no. July, pp. 161–175, 2019.

[25] I. Pathak and D. P. Vidyarthi, "An interactive Artificial Bee Colony based virtual network embedding," *2015 IEEE UP Section Conference on Electrical Computer and Electronics, UPCON 2015*, 2016.

[26] Z. Zhou, X. Chang, Y. Yang, and L. Li, "Resource-Aware virtual network parallel embedding based on genetic algorithm," *Parallel and Distributed Computing, Applications and Technologies, PDCAT Proceedings*, vol. 0, pp. 81–86, 2016.

[27] L. R. Bays, L. P. Gaspary, R. Ahmed, and R. Boutaba, "Virtual network embedding in software-defined networks," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, no. Noms. IEEE, apr 2016, pp. 10–18.

[28] K. T. Nguyen and C. Huang, "An intelligent parallel algorithm for online virtual network embedding," in *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*. IEEE, 2019, pp. 1–5.

[29] G. Kibalya, J. Serrat, J. L. Gorricho, H. Yao, and P. Zhang, "A novel dynamic programming inspired algorithm for embedding of virtual networks in future networks," *Computer Networks*, vol. 179, no. December 2019, p. 107349, 2020.