Task Offloading with 5G Network Slicing for V2X Communications

George Alkhoury, Sara Berri, Arsenia Chorti

ETIS, UMR 8051 CY Cergy Paris University, ENSEA, CNRS F-95000
george.alkhoury@ensea.fr, sara.berri@ensea.fr, arsenia.chorti@ensea.fr

***Abstract*—Vehicular Edge Computing (VEC) technology allows vehicles demanding significant computation and storage resources to offload their demands to the nearest edge computing node, aiming at reducing data transfer latency and enhancing the Quality of Service (QoS). Moreover, the heterogeneous applications of Vehicle-to-Everything (V2X) communications need an efficient management of the nodes' resources to satisfy the diverse requirements of the vehicles' demands. To this end, network slicing could be a promising solution. Task offloading algorithms in VEC proposed in the literature usually rely on offloading to a 5G base station (gNodeB) or a Road Side Unit (RSU) and do not differentiate between the various vehicular demands. In this paper, we study the task offloading problem with network slicing in V2X communications from vehicles to edge computing nodes hosted at gNodeBs, RSUs, and nearby vehicles. We model the network and formulate the problem as an integer linear program, with the objective of maximizing the volume of offloaded tasks from diverse services. We propose a heuristic algorithm and slicing schemes to find a near-optimal solution to the NP hard optimization problem. The simulation results show that considering offloading at nearby vehicles in addition to RSUs and gNodeBs yields better results in terms of acceptance ratio and resource utilization. Furthermore, it is found that it is beneficial to use an adaptive slicing scheme instead of relying on a fixed slicing; in particular, when the number of slices is large.**

***Index Terms*—5G Network Slicing, Vehicular Edge Computing, Task Offloading, Vehicle-to-Everything Communications (V2X).**

I. Introduction

Vehicle-to-Everything (V2X) communications are among the most promising services of the 5G and comprise four communication modes, namely Vehicle-to-Vehicle (V2V), Vehicle-to-Pedestrians (V2P), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Network (V2N), with heterogeneous applications, use case scenarios, and requirements. Recently, advanced sensing technologies are introduced into modern vehicles to increase and develop the automation levels and pave the way toward efficient V2X communications. A significant number of sensors that generate an enormous amount of data from the surrounding environment are used to perform various safety and autonomous driving tasks like localization and object detection [1]. This amount of data can overload the vehicle's capacity-limited computing and storage units. Therefore, considering Vehicular Edge Computing (VEC) technology and task offloading is more favorable. Indeed, VEC is defined as the integration of the Multi-access Edge Computing (MEC) technology in the traditional vehicular networks [2], whose the aim is to move computing, storage, and caching resources as close as possible to the vehicular users at the edge servers hosted at 5G base stations (gNodeBs) or Road Side Units (RSUs).

978-1-6654-3540-6/22 © 2022 IEEE

Most of the existing works on task offloading in vehicular edge computing do not consider the possibility to offload some tasks into a nearby vehicle, which can help significantly in reducing the data transfer latency for some delay-sensitive vehicular applications and reducing the load on the network resources especially in rush hours, which, in turn, improves the network reliability and availability. Moreover, the V2X use cases have diverse and conflicting computing, storage, latency, reliability, and throughput requirements. To deal with this, network slicing could be integrated to allow the service providers to design one or more network slices to support multiple V2X requirements.

Motivated by the above, in this work, we study the task offloading problem with network slicing in V2X communications from vehicles to edge computing nodes residing at gNodeBs, RSUs, and nearby vehicles. A related problem was studied in [3], [4] where the authors considered a static slicing scheme and the task offloading was possible only at gNodeBs and RSUs. In the present article we extend this work to a more general scenario that includes V2V task offloading and propose novel adaptive slicing schemes. The contributions of this article are as follows:

- We study the problem of task offloading at gNodeBs, RSUs, and nearby vehicles with network slicing. We address jointly the problem of task offloading and network slicing to differentiate the different services.
- We formulate the considered problem as an integer linear program in the presence of multiple edge nodes and propose a heuristic algorithm and slicing schemes to yield a near-optimal solution to the problem.
- Finally, we show through simulations that considering the capability of task offloading into nearby vehicles and adaptive slicing schemes have a large impact on the performance of task offloading. Moreover, we evaluate the impact of the gNodeBs on the global performance.

The rest of the paper is organized as follows. We present some related works in Section II. In Section III, we model the network and formulate the problem. In Section IV, we introduce a heuristic algorithm to solve the problem. In Section V, we present some numerical results to assess the performance of the proposed scheme. Finally, the Section VI concludes the paper.

II. Related Works

In this section, we introduce some related works to the problem of task offloading in vehicular edge computing and 5G network slicing.

In [5], the authors envisaged a radio access network (RAN) slicing mechanism to design a customized V2I communication for mission-critical applications such as the Autonomous Tram use case. The slicing problem was formulated considering two approaches, namely RAN Slicing with No Inter Slice Protection (RS-NOISP) and RAN Slicing with Inter Slice Protection (RS-ISP). The work [6] studied RAN network slicing at the level of downlink's resources to support two divergent vehicular services, namely autonomous driving and infotainment service. The proposed slicing algorithm is based on vehicles clustering mechanism and the assignment of the slice leaders. The authors in [7] showed the impact of slice reconfiguration delay on the performance of a vehicular URLLC slice and proposed a proactive resource allocation approach that anticipates the slice needs. In [8], the authors focused on network slicing supported by Non-Orthogonal Multiple Access (NOMA) in non-cellular vehicular networks where vehicles communicate using broadcast via sidelinks. They formulated the problem as a multi-agent Markov decision process by considering two network slices, namely safety and non-safety, and solved it using deep Q-learning (DQL) algorithm. These works considered a small number of slices, which could not necessarily cover a large specter of applications. Therefore, it would be more interesting to consider a general scenario that can include more than two slices and services.

In [1], the authors proposed a learning-assisted hierarchical approach to solve a two-timescale RAN slicing and computing task offloading problem. The main objective was to maximize computing resources and communication utilization in a joint way with diverse QoS requirements in a cloud-enabled autonomous vehicular networks (C-AVN). The considered problem was formulated as a stochastic optimization problem for maximizing computation load balancing with minimum task offloading variations to capture small timescale network dynamics. The authors in [3] studied and handled the task offloading problem from vehicles to 5G base stations and RSUs. The problem was formulated as a linear integer program and a new heuristic algorithm was proposed. Moreover, this work was extended to introduce the load balancing feature [4].

In contrast to previous works, such as [3], [4], we propose to study a more general case by considering the V2V communications via sidelinks and the possibility of offloading into nearby vehicles in addition to RSUs and gNodeBs. Moreover, we will consider dynamic slicing schemes at the edge.

III. Network Model and Problem Formulation

In this section, we will describe the network model and formulate the problem of task offloading with network slicing as an integer linear program.

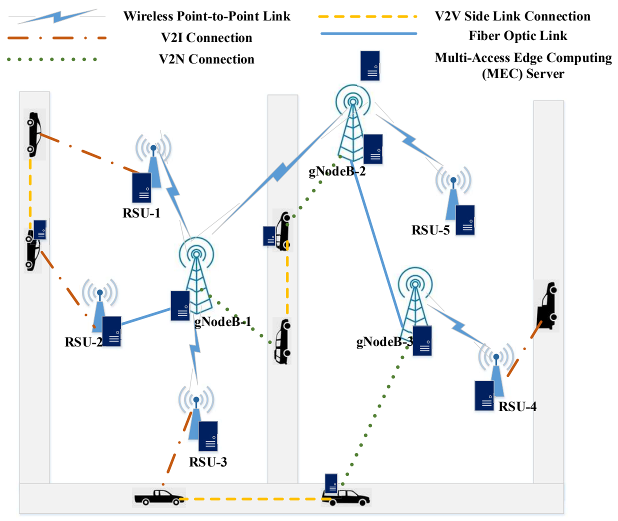

Figure 1: Network Model

A. Network Model

We consider the environment of a dense urban area travelled by a set of vehicles $\mathcal{V} = \{1, \dots, V\}$, and fully covered by a set of gNodeBs $\mathcal{G} = \{1, \dots, G\}$ and a set of RSUs $\mathcal{R} = \{1, \dots, R\}$. Each RSU is connected to one gNodeB through one wired connection (e.g. fiber optic) or wireless connection (e.g. millimeter-wave point-to-point link), and the gNodeBs are connected through similar wired or wireless connections. Each vehicle has one connection with RSU or gNodeB via V2I or V2N, respectively, and may have a connection with a nearby vehicle via V2V communication. Furthermore, the RSUs and gNodeBs are equipped with a MEC server, representing the VEC servers, and provide computing edge feature. In addition, in this paper, we consider that the vehicles might be equipped with a small edge server as well, and consequently offer the possibility of task offloading to other nearby vehicles. Therefore, to account for this feature, we represent the set of vehicles by means of two subset: the subset of vehicles that offer edge computing feature $\mathcal{V}_{\text{VEC}} = \{1, \dots, V_{\text{VEC}}\}$ and the subset of vehicles that do not offer this feature $\mathcal{V}_{\text{NVEC}} = \{1, \dots, V_{\text{NVEC}}\}$, where $V_{\text{VEC}} + V_{\text{NVEC}} = V$ and $\mathcal{V} = \mathcal{V}_{\text{VEC}} \cup \mathcal{V}_{\text{NVEC}}$. Define by $\mathcal{N} = \{1, \dots, N\}$ the set of nodes composed of the sets of RSUs, gNodeBs and VEC enabled vehicles, i.e. $\mathcal{N} = \mathcal{R} \cup \mathcal{G} \cup \mathcal{V}_{\text{VEC}}$. The sketch of the network model is represented by Fig. 1. The VEC servers on the nodes are assumed to be virtualized and sliced into a set of slices, denoted in the sequel as $\mathcal{S} = \{1, \dots, S\}$, following network function virtualization (NFV) and network slicing guidelines. We denote the computing power and the storage capacity of a node $i \in \mathcal{N}$ that is devoted to an instantiated slice $j \in \mathcal{S}$ as $C_{i,j}$ and $M_{i,j}$, respectively. On the other hand, the vehicles might ask for task offloading, we denote by $\mathcal{D} = \{1, \dots, D\}$ the set of all the demands. Each demand $k \in \mathcal{D}$ has a set of related requirements to be fulfilled defined as follows: computing power c_k, storage capacity m_k, latency d_k, data rate w_k, and the requested slices. To indicate the requested slice j by a demand k, we define the requested slice matrix $\mathbf{Z}$, where $z_{k,j} = 1$ if the demand k requests the slice j and $z_{k,j} = 0$ otherwise. Moreover, we introduce two binary parameters $\delta_{k,i}$ and $\gamma_{k,i}$ to decide whether the latency and data rate requirements of the demand k could be respected by the node i or not, respectively:

$$\delta_{k,i} = \begin{cases} 1, & \text{if } d_k \geq H_{k,i}; \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

$$\gamma_{k,i} = \begin{cases} 1, & \text{if } w_k \leq I_{k,i}; \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where: d_k and w_k are the requested latency and data rate of the demand k, respectively; $H_{k,i}$ and $I_{k,i}$ are the offered latency and data rate by the node i in response to a demand k, respectively.

B. Problem Formulation

In this subsection, we formulate the problem of task offloading based on network slicing. The objective is to maximize the number of satisfied task offloading demands throughout the entire network given the task offloading demand's requirement and the total capacities in terms of computing power and storage at the VEC nodes including gNodeBs, RSU, and VEC enabled vehicles. For this purpose, we formulate the problem as an integer linear program. We introduce a binary variable y_k that indicates the successful offloading of the demand k ($y_k = 1$) or not ($y_k = 0$). We define the binary variable $x_{k,i,j}$ that indicates if the task offloading demand k is satisfied by the node i using the resources devoted to the slice j ($x_{k,i,j} = 1$) or not ($x_{k,i,j} = 0$).

Formally, the task offloading problem with network slicing is formulated as follows:

$$\max \sum_{k \in \mathcal{D}} y_k, \tag{3}$$

Subject to:

$$\sum_{k \in \mathcal{D}} x_{k,i,j}.c_k \leq C_{i,j} \qquad \forall i \in \mathcal{N}, j \in \mathcal{S}, \tag{4}$$

$$\sum_{k \in \mathcal{D}} x_{k,i,j}.m_k \leq M_{i,j} \qquad \forall i \in \mathcal{N}, j \in \mathcal{S}, \tag{5}$$

$$y_k \leq \sum_{\substack{i \in \mathcal{N} \\ j \in \mathcal{S}}} \delta_{k,i}.\gamma_{k,i}.x_{k,i,j} \qquad \forall k \in \mathcal{D}, \tag{6}$$

$$\sum_{i \in \mathcal{N}} x_{k,i,j} \leq z_{k,j} \qquad \forall k \in \mathcal{D}, j \in \mathcal{S}, \tag{7}$$

$$\sum_{\substack{i \in \mathcal{N} \\ j \in \mathcal{S}}} x_{k,i,j} \leq 1 \qquad \forall k \in \mathcal{D}, \tag{8}$$

$$y_k \in \{0,1\}, x_{k,i,j} \in \{0,1\} \qquad \forall k \in \mathcal{D}, i \in \mathcal{N}, j \in \mathcal{S}. \tag{9}$$

The objective in (3) is to maximize the total number of successful task offloading demands. The constraints (4) and (5) are introduced to ensure that the total amount of demanded computing power and storage, respectively, cannot exceed the total capacities of the slice at the selected node. The constraints (6) and (7) guarantee that the demand is considered satisfied only if its required latency, data rate and slice are well satisfied by the selected node. The constraints (8) ensure that the demand is processed by only one slice instantiated at one node. Finally, the constraints (9) indicate the binary decision variables of the problem. Note that in this formulation the set of nodes $\mathcal{N}$ includes the gNodeBs, RSUs and VEC enabled vehicles.

IV. The Proposed Algorithm

Previously, we proposed a linear programming formulation for the task offloading problem with network slicing. This problem can be reduced from the well-known NP-hard Multi-dimensional Bin Packing problem. Thus, the proposed problem is also NP-hard. Therefore, in this section we propose a heuristic algorithm to solve the problem.

The proposed algorithm will have a general network-wide view and will coordinate and orchestrate the network resources among the slices and handle the task offloading operations for edge-based vehicular applications; thus, it follows the SDN approach in control centralization and separating the control plane from the data plane and sticks to the NFV approach in virtualizing the data plane of the network and decoupling the virtual network functions from the underlying physical infrastructure.

Initially, we initialize the involved parameters, which form the input of the algorithm. Moreover, the positions of the nodes gNodeBs, RSUs, and VEC enabled vehicles are used to define the set of all possible links between these nodes denoted by $\mathcal{L} = \{1, \ldots, L\}$. To determine if a vehicle is VEC enabled or not, we introduce a binary parameter to indicate whether the vehicle that initiated the demand k belongs to $\mathcal{V}_{\text{VEC}}$, $v_k = 1$, or not $v_k = 0$. According to v_k, the steps to be executed for each demand $k \in \mathcal{D}$ are described in the following :

1) The vehicle, which initiated the demand, belongs to $\mathcal{V}_{\text{VEC}}$ ($v_k = 1$) and the local server has enough resources at the demanded slice to satisfy the requirements : in this case, the vehicle offloads the task internally, and the algorithm updates the binary variables y_k, $x_{k,i,j}$, and the vehicle's resources (lines 5-8); then, it moves to the next demand. Indeed, by considering this approach, the vehicle saves the latency and data rate required to offload the task to another node and consequently decreases the load on the network.

2) The vehicle does not belong to $\mathcal{V}_{\text{VEC}}$ ($v_k = 0$) or there are not enough resources internally : here, the algorithm determines a metric for each node, defined as a normalized weighted score of the latency and data rate with weights $\omega \in [0,1]$, and $(1-\omega)$, respectively.
Subsequently, the nodes are sorted in decreasing order based on the score and form a new ordered set $\mathcal{N}_o$. Then, the algorithm selects the first node, denoted by i_0, and checks if it satisfies the demand's requirements in terms of computing power, storage,

latency, and data rate. If all the constraints are satisfied, the task offloading demand is accepted and the available resources on i_0 are updated (lines 18-21). In the case where the node i_0 does not satisfy the requirements, the algorithm determines the set of the links starting from this node, $\mathcal{L}_{i_0}$, and checks the next node, i_1, of the first link; if it does not satisfy either the demand's requirements, the considered link is removed from the set $\mathcal{L}_{i_0}$ and the algorithm repeats the process (lines 25-29). At last, if the demand is not satisfied by any of the involved nodes in the links $\mathcal{L}_{i_0}$, the algorithm removes the node i_0 from $\mathcal{N}_o$, chooses the next node and repeats all the steps (from line 17). A demand is considered dropped when there is no node that can satisfy its requirements.

The total estimated computational time complexity of the proposed algorithm to perform the offloading of D demands is $O(D \times (N \log(N) + (N \times L))$.

V. Performance Evaluation

In this section, we assess the performance of the proposed algorithm, and compare its performance with a state of the art algorithm [3], that handles the task offloading problem without considering offloading at nearby vehicles and assuming a static slicing scheme. We used Python 3.6 as a development language and NetworkX library to create, manipulate, and visualize our physical network. We present the simulation settings in Section V-A, the proposed slicing schemes in Section V-B, and some results in Section V-C.

A. Simulation Settings

The physical network is generated by considering a dense urban area of size 1Km $\times$ 1Km. It is composed of 40 fixed nodes, 10 gNodeB and 30 RSUs, and 1500 vehicles (500 of them are VEC-Enabled vehicles) that are distributed randomly across the considered area. Each VEC enabled vehicle is connected with the nearest node either gNodeB or RSU, which is supposed to offer the best communication channel to the vehicle. We assume that V2V communications occur via the PC5 interface, while V2I and V2N communications occur via the 5G New Radio (NR) interface. Concerning the slicing settings, we consider a variant number of slices from 0 (i.e., no slicing) up to 6 slices. Note that the mobility is not considered.

To define the data rate that a certain gNodeB, RSU, or a VEC enabled vehicle i offers to a demand k initiated by another vehicle we use the Shannon achievable rate expressions in fading environments [9].

$$I_{k,i} = W \times \log\left(1 + \frac{P_v \times g_{k,i}}{\sigma^2}\right), \tag{10}$$

where $I_{k,i}$ is the capacity in the link between node i and the vehicle that initiated the demand k, W is the channel bandwidth, P_v is the transmission power of vehicles transceivers, σ^2 is the variance ambient Gaussian noise, and $g_{k,i}$ is the channel gain between the vehicle and the node i in the urban micro-cell wireless channel environment [10], which is defined as follows:

$$\begin{aligned} g_{k,i} = 40\log(d'_{k,i}) + 9.45 - 17.3\log(h_n) - 17.3\log(h_v) \\ + 2.7\log(f_c/5), \end{aligned} \tag{11}$$

Algorithm 1: Proposed algorithm for task offloading with slicing

```
   Require: $\mathcal{N}$, $\mathcal{V}$, $\mathcal{S}$, $\mathcal{D}$, $\mathcal{L}$, $\mathbf{C}$, $c$, $\mathbf{M}$, $m$, $w$, $d$, $v$
1  for $k$ from 1 to $D$ do
2      Set $y_k = 0$, $x_{k,i,j} = 0$ ; $i \in \mathcal{N}, j \in \mathcal{S}$, and $j^\star$ the demanded slice
3      if $v_k = 1$ then
           // Internal checking
4          if $c_k \leq C_{i,j^\star}$ AND $m_k \leq M_{i,j^\star}$ then
5              $y_k = 1$ // The demand is satisfied
6              $x_{k,i,j^\star} = 1$
7              $C_{i,j^\star} = C_{i,j^\star} - c_k$
8              $M_{i,j^\star} = M_{i,j^\star} - m_k$
9          else
10             go to 12
11     else
12         Calculate the data rate $I_{k,i}$ and the latency $H_{k,i}$ offered by each node $i \in \mathcal{N}$ to the demand $k$.
13         Score the nodes $i \in \mathcal{N}$ according to the metric : $\omega H_{k,i} + (1-\omega) I_{k,i}$.
14         Sort the set $\mathcal{N}$ based on the score, $\mathcal{N}_o$
15         while $\mathcal{N}_o \neq \phi$ AND $y_k = 0$ do
16             Select the top scored node in the set $\mathcal{N}_o$, $i_0$
17             if $c_k \leq C_{i_0,j^\star}$; AND $m_k \leq M_{i_0,j^*}$ AND $\delta_{k,i_0} = 1$ AND $\gamma_{k,i_0} = 1$ then
18                 $y_k = 1$ // The demand is satisfied
19                 $x_{k,i_0,j^*} = 1$
20                 $C_{i_0,j^*} = C_{i_0,j^*} - c_k$
21                 $M_{i_0,j^*} = M_{i_0,j^*} - m_k$
22             else
23                 List all the links starting with $i_0$, $\mathcal{L}_{i_0} = \bigcup_{-0 \in \mathcal{N}} (i_0, i_{-0})$.
24                 while $\mathcal{L}_{i_0} \neq \phi$ AND $y_k = 0$ do
25                     Select the first link in $\mathcal{L}_{i_0}$, $l_0 = (i_0, i_1)$
26                     if $c_k \leq C_{i_1,j^*}$ AND $m_k \leq M_{i_1,j^*}$ AND $\delta_{k,i_1} = 1$ AND $\gamma_{k,i_1^*} = 1$ then
27                         Execute 18-21 on the node $i_1$.
28                     else
29                         $\mathcal{L}_{i_0} = \mathcal{L}_{i_0} \setminus \{l_0\}$
30             $\mathcal{N}_o = \mathcal{N}_o \setminus \{i_0\}$
```

where $d'_{k,i}$ is the Euclidean distance in meters between the vehicle and the node i, h_n is the node's antenna height, h_v is the vehicle's antenna height, and f_c is the frequency of the communication channel. The latency $H_{k,i}$ in the communication between the vehicle that initiated the demand k and the node i, is calculated by means of (12).

$$H_{k,i} = \frac{w_k}{I_{k,i}}, \tag{12}$$

Table I: Simulation parameters

Parameter	Value
Number of gNodeBs	10
Number of RSUs	30
Number of vehicles	1500
Number of VEC enabled vehicles	500
Number of slices	$\{2, 4, 6\}$
Computing power	gNodeB:300, RSU:100, V_{VEC}:30
Storage Capacity	gNodeB:300, RSU:100, V_{VEC}:30
c_k	random between 1 and 10
m_k	random between 1 and 10
d_k	random between 5 and 10
w_k	random between 10 and 50
W	V2I and V2N:50 MHz V2V:20 MHz
f_c	V2I and V2N:3 GHz, V2V:5.9 GHz
P_v	25 dBm
h_v	1.5 m
h_n	10 m
σ^2	7 dB
ω	0.7

where w_k is the requested data rate by the demand k and $I_{k,i}$ is the offered data rate by the node i to that vehicle. Table I summarizes the simulation parameters.

B. Proposed resource allocation schemes for slices

The proposed offloading algorithm is executed given a slicing scheme. Therefore, to provide effective performance evaluation, we propose two fixed slicing schemes (FSS and FSS') and an adaptive slicing scheme (ASS), we describe in the following.

1) Fixed slicing schemes (FSS and FSS'): The resources in terms of computing power and storage of the nodes are divided among the slices according to a fixed distribution defined as the percentages of the total resources. In this paper, we consider two different fixed schemes : *i)* The first one, called FSS, allocates the resources to the slices according to the distribution in Table II. This distribution is used at all the nodes, including RSUs, gNodeBs, and VEC enabled vehicles. *ii)* The second one, called FSS', is based on modifying the resources allocated to the defined slices in VEC enabled vehicles to guarantee enough resources per slice as these nodes are equipped with small capacities. Thus, we consider that the vehicles have different resource allocation and the distribution is determined according to the Table III. Regarding the RSUs and gNodeBs, FSS' uses the resource allocation defined in Table II.

2) Adaptive slicing scheme (ASS): In this scheme, we assume that the offloading demands can be collected in advance and used to guide the resource allocation for slices. Based on that assumption, the percentage of the allocated resources per slice is determined according to the requested resources by all the vehicles. The proportion of the storage capacity and computing power of the slice j at node i correspond to $\frac{\sum_{k\in\mathcal{D}} z_{k,j}.c_k}{\sum_{k\in\mathcal{D}} c_k}$ and $\frac{\sum_{k\in\mathcal{D}} z_{k,j}.m_k}{\sum_{k\in\mathcal{D}} m_k}$, respectively. Indeed, this scheme allows one to determine the capacity of slices such that the total number of satisfied demands is maximized.

Table II: Allocation of resources among slices in the FSS

S \ Slice	1	2	3	4	5	6
2	60%	40%	-	-	-	-
4	40%	30%	15%	15%	-	-
6	35%	25%	20%	10%	5%	5%

Table III: Allocation of resources among slices in the FSS'

50% of the VEC enabled vehicles						
S \ Slice	1	2	3	4	5	6
2	60%	40%	-	-	-	-
4	50%	50%	0%	0%	-	-
6	34%	33%	33%	0%	0%	0%
50% of the VEC enabled vehicles						
2	60%	40%	-	-	-	-
4	0%	0%	50%	50%	-	-
6	0%	0%	0%	34%	33%	33%

C. Comparison to state of the art

In this subsection, we compare the performance of our algorithm with the performance of the algorithm proposed in [3] that does not consider V2V offloading and works with a fixed slicing scheme according to the parameters of FSS. The proposed algorithm in the present paper is implemented with the three proposed slicing schemes, FSS, FSS' and ASS.

Fig. 2 shows how the acceptance ratio scales with the number of slices S; 0 slice means that the offloading algorithm is run without the slicing feature. Each value in the results is an average of ten simulation runs. We observe that the proposed algorithm always performs better than the proposed one in [3], and provides a significant improvement around $5\% - 27\%$. Moreover, from the results of our algorithm with FSS' or ASS, we can clearly see that the acceptance ratio could be notably improved when we consider an adaptive distribution model of resource allocation among the slices. The results of our algorithm with FSS' are quite similar to those with ASS, and much better than FSS and the proposed algorithm in [3] with a fixed slicing scheme. This is because, with FSS' the total amount of resources allocated for each slice at the VEC enabled vehicles is augmented, which, in turn, increases the capability of these vehicles to fulfill the requirements of the incoming demands. Furthermore, when the number of slices increases, namely 4 and 6 slices, the results are less significant. The reason is that, in such cases the allocated resources on each slice, as defined in Tables II and III, will decrease, especially at the VEC enabled vehicles due to their initial limited resources compared to the available resources at gNodeBs and RSUs. Thus, this degradation reduces the capability of the nodes to fulfill the task offloading demands successfully resulting in a steep decline in the acceptance ratio.

Fig. 3 shows the CPU and memory utilization at the gNodeBs and RSUs. We can see that the proposed algorithm can reduce the resources' utilization by nearly $12\% - 24\%$ compared to the state of the art algorithm [3], because the VEC enabled vehicles satisfy a part of the demands. This will significantly enhance

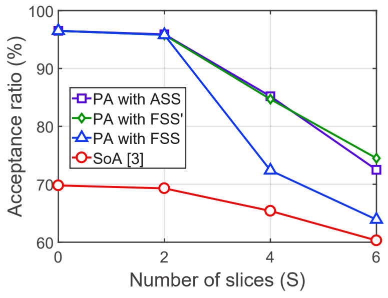

Figure 2: Comparison in terms of acceptance ratio between the proposed algorithm (PA) and the state of the art algorithm (SoA) [3]. We represent for different slicing schemes.

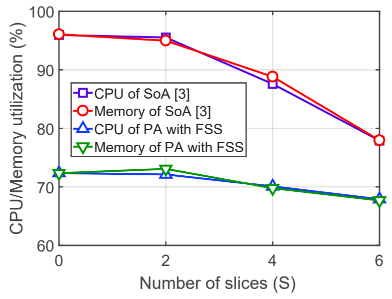

Figure 3: Comparison in terms of CPU and memory utilization between PA and SoA [3].

the network performance during peak hours and satisfy more demands that require huge resources.

To conclude, we can clearly see that it is possible to implement network slicing and benefit from its advantages in considering different services while guaranteeing a certain performance level.

We further investigate the performance of the proposed offloading algorithm with Wireless Access in Vehicular Environment (WAVE) system. Indeed, this provides a possibility to compare our results to existing works that do not consider offloading to edge server hosted at 5G base station and network slicing, invoked in works such as [11]. The new simulation parameters are as follows : the channel bandwidth is equal to 10 MHz, and the channel frequencies are set to 5.85 GHz and 5.89 GHz for V2I and V2V, respectively; the VEC vehicles' computation and storage capacities are set to 10 units; the requested data rate is randomly selected between 1 and 20, since we considered different technology that is supposed to condition less data rate than 5G and its applications. All the other parameters remain the same. In the network model, the set of gNodeB is replaced by RSUs. Table IV, provides the acceptance ratio for different RSUs' computation and storage capacities (CP). It is seen that the difference in terms of performance is small (CP: from 0 unit to 150 units), which means that, in such scenario, the algorithm might rely on the VEC enabled vehicles to fulfil the demands, and the acceptance ratio could not be highly improved due to the other constraints,

Table IV: Acceptance Ratio (AR) with WAVE environment

CP	0	25	50	75	100	125	150
AR	53.2%	59.7%	61.7%	61.9%	62%	62.1%	62.1%

such as data rate that is limited in this scenario. From these results, we can clearly see that the scheme that offloads tasks at RSUs or nearby vehicles always performs worse than the one that includes gNodeBs and network slicing whose results vary $63\% - 96\%$ as shown in Fig. 2.

VI. Conclusion

In this paper, we have studied the task offloading problem in vehicular edge computing that supports 5G network slicing. The goal is to offload the tasks at gNodeBs, RSUs and vehicles using the appropriate slice, so as to maximize the accepted task offloading demands. We formulated the problem as an integer linear program, and proposed a heuristic and slicing schemes to provide an approximate solution. The results have shown significant improvement in terms of acceptance ratio and resource utilization by including some offloading enabled vehicles in addition to gNodeBs and RSUs, and there is a significant performance increase by considering advanced slicing schemes.

In future works, we will investigate the problem of task offloading with 5G network slicing in a high mobility vehicular environment.

References

[1] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. Shen, "Joint RAN Slicing and Computation Offloading for Autonomous Vehicular Networks: A learning-Assisted Hierarchical Approach," *IEEE Open Journal of Vehicular Technology*, vol. 2, pp. 272–288, 2021.

[2] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular Edge Computing and Networking: A Survey," *Mobile Networks and Applications*, vol. 26, no. 3, pp. 1145–1168, 2021.

[3] S. Berri, K. Hejja, and H. Labiod, "Slicing-Based Offloading in Vehicular Edge Computing," in *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*. IEEE, 2021, pp. 1–7.

[4] K. Hejja, S. Berri, and H. Labiod, "Network Slicing with Load-Balancing for Task Offloading in Vehicular Edge Computing," *Vehicular Communications*, pp. 1–16, 2021.

[5] D. Tamang, S. Martiradonna, A. Abrardo, G. Mandó, G. Roncella, and G. Boggia, "Architecting 5G RAN Slicing for Location Aware Vehicle to Infrastructure Communications: The Autonomous Tram Use Case," *Computer Networks*, vol. 200, pp. 1–11, 2021.

[6] H. Khan, P. Luoto, S. Samarakoon, M. Bennis, and M. Latva-Aho, "Network Slicing for Vehicular Communication," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, pp. 1–14, 2021.

[7] N. Naddeh, S. B. Jemaa, S. E. Elayoubi, and T. Chahed, "Proactive RAN Resource Reservation for URLLC Vehicular Slice," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*. IEEE, 2021, pp. 1–5.

[8] Z. Mlika and S. Cherkaoui, "Network Slicing for Vehicular Communications: A Multi-Agent Deep Reinforcement Learning Approach," *Annals of Telecommunications*, vol. 76, no. 9, pp. 665–683, 2021.

[9] W. C. Lee, "Estimate of Channel Capacity in Rayleigh Fading Environment," *IEEE transactions on Vehicular Technology*, vol. 39, no. 3, pp. 187–189, 1990.

[10] P. Kyosti, "WINNER II Channel Models," *IST, Tech. Rep. IST-4-027756 WINNER II D1. 1.2 V1. 2*, 2007.

[11] S. Raza, W. Liu, M. Ahmed, M. R. Anwar, M. A. Mirza, Q. Sun, and S. Wang, "An Efficient Task Offloading Scheme in Vehicular Edge Computing," *Journal of Cloud Computing*, vol. 9, no. 1, pp. 1–14, 2020.