Fog Computing Model and Efficient Algorithms for Directional Vehicle Mobility in Vehicular Network

Yalan Wu, Jigang Wu, *Member, IEEE*, Long Chen, Gangqiang Zhou, and Jiaquan Yan

***Abstract*— Vehicular fog computing (VFC) has become an appealing paradigm to provide services for vehicles and traffic systems. However, high mobility is one of the great challenges to the communication and computation service qualities in VFC. A network model for directional vehicle mobility is proposed in this paper to guarantee the service qualities of vehicles in VFC. In the model, vehicles are configured into three vehicular subnetworks according to their turning directions at the next crossing. For each subnetwork, vehicles communicate with each other via vehicle-to-vehicle communication, and with roadside units via vehicle-to-infrastructure communication. The aim is to minimize the average response time of the tasks originated from vehicles. By carefully choosing neighboring vehicles as task processing helpers, a greedy algorithm is proposed to solve the mentioned optimization problem. Besides, two bipartite matching based algorithms, named BMA_1 and BMA_2, are proposed by exploiting Kuhn-Munkras approach and minimum-cost maximum-flow approach, respectively. Performance of the proposed model and the offloading algorithms are evaluated on the combined simulation platform by open street map, SUMO and NS-3. Simulation results show that, the proposed model outperforms four existing models in terms of average response time, when the five models have similar number of unsuccessful tasks. Moreover, the proposed BMA_1 and BMA_2 are superior to the existing greedy algorithm in terms of the average response time of tasks, and the proposed greedy algorithm significantly accelerates the generation of offloading decisions in comparison to BMA_1, BMA_2 and the existing greedy algorithm.**

***Index Terms*— Vehicular fog computing, vehicular network, task offloading, algorithm.**

I. Introduction

VEHICULAR network has emerged as a result of advancements in widely distributed wireless access network and 5G communication technologies. Moreover, vehicular network becomes an important component of the future intelligent transportation systems [1], [2]. In the systems, vehicles are equipped with components, such as wireless communication modules, on-board units, sensors, electronic control units etc. [3]. Therefore, vehicular network not only brings various mobile services to users [4], [5], such as navigation and entertainments, but also provides the traffic manager much information sensed by vehicles [6], [7]. Many design challenges exist in vehicular network [8], [9], such as multi-objective protocols, channel state information estimation, optimal cooperative relay selection etc. in a highly mobile environment. Besides, highly mobile environment leads to frequent handover of vehicular-to-infrastructure (V2I) communication and vehicle-to-vehicle (V2V) communication [10], [11], and increased data loss [12].

In vehicular network, existing works mainly focus on the aspect of V2V and V2I communications to achieve higher spatial diversity [11], lower transmission delay [13], and higher throughput [14] and so on, via multi-objective protocol [15], channel estimation [16] and collision avoidance [17], etc. For achieving the low transmission delay and high throughput, authors in [10] integrate 5G mobile communication technologies with software defined network as a new vehicular network architecture. In the new architecture, multihop relay network is utilized to reduce the frequent handover between RSUs and vehicles. Authors in [18] propose a hybrid routing scheme to achieve robust communication and max-flow min-cut data dissemination. Authors in [19] propose a software-defined IoT system in urban heterogeneous network to achieve scalable mobility management and robust flow scheduling. All these existing works only consider the aspect of communication in vehicular network, but the computation is not considered.

Generally, vehicles and roadside units (RSUs) are equipped with a large number of computation and storage resources. It is well known that, the computation resources in vehicles or RSUs can improve computing services. Therefore, as a promising paradigm, vehicular fog computing (VFC) is proposed to provide services with less latency for vehicles by integrating the computation resources of near located vehicles and RSUs [20]. Fig. 1 shows the architecture of VFC, where vehicles are treated as potential computation resources. Vehicles in VFC can connect with each other or with RSUs which are placed with edge servers. It is noteworthy that, the resources in vehicles and edge servers are integrated into a fog cloud to provide services for vehicles.

In the existing works, three types of vehicular network models are widely used in VFC to provide various kinds of

Manuscript received March 3, 2019; revised August 22, 2019 and November 26, 2019; accepted January 24, 2020. Date of publication February 10, 2020; date of current version May 3, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1003201, in part by the National Natural Science Foundation of China under Grant 61702115 and Grant 61672171, in part by the Guangdong Natural Science Foundation under Grant 2018B030311007, and in part by the Guangdong Key Research and Development Project of China under Grant 2018B010107003 and Grant 2019B010118001. This article was presented in part at the 2018 IEEE Ubiquitous Intelligence and Computing. The Associate Editor for this article was D. Wu. *(Corresponding author: Jigang Wu.)*

Yalan Wu, Jigang Wu, Long Chen, and Jiaquan Yan are with the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou 510006, China (e-mail: wuyalan93@outlook.com; asjgwucn@outlook.com).

Gangqiang Zhou is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China.

Digital Object Identifier 10.1109/TITS.2020.2971343

1558-0016 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

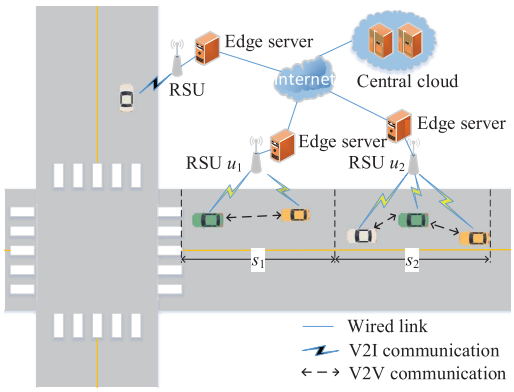


Fig. 1. The architecture of vehicular fog computing.

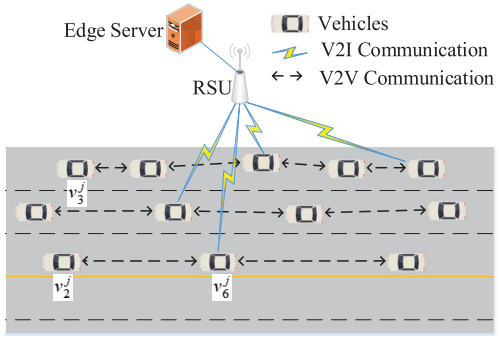


Fig. 2. The directional group mobility vehicular network model.

services. The first one is vehicular network model without cooperative communication among vehicles [12], [21], and it is denoted as individual model in this paper. In the individual model, all on-road vehicles can communicate with RSUs by V2I communication, while vehicles can not communicate with each other. To the best of our knowledge, the vehicles are flexible and independent in the individual model, but V2I connection needs to be reconfigured frequently due to high mobility of vehicles. Moreover, when there are a large amount of vehicles in the road segment, communications between vehicles and RSUs will be congested. It will decrease the bandwidth of V2I connection and waste the computation resources in vehicles.

The second type of vehicular network model is only with collaborative communication among vehicles [22], and it is denoted as OVM in this paper. In OVM, all vehicles in the road segment are integrated into one vehicular network by V2V communication, without establishment of V2I communication. Therefore, vehicles only can cooperate with other vehicles, but they can not cooperate with RSUs. OVM can take full advantage of computation resources in vehicles. However, the V2V communication will be congested, if lots of vehicles appear in the road segment. Meanwhile, the computation resources of RSUs are wasted.

The third type of vehicular network models is with V2I communication and V2V communication [10], [23]. In this type of models, all vehicles in the road segment are integrated into one vehicular network by V2V communication. Meanwhile, in the vehicular network, some selected vehicles, namely gateway vehicles, can communicate with RSU by V2I communication. Specifically, this type of models has two vehicular network models utilized in the existing works. One is the one gateway vehicular network model [10], which is denoted as OGM in this paper, and the other is the multiple gateways vehicular network model [23], which is denoted as MGM in this paper. In OGM, only one selected vehicle, i.e., gateway vehicle, can directly connect with RSU by V2I communication. In addition, other vehicles only can communicate with RSUs through the gateway vehicle, instead of by V2I communication. But, in MGM, all vehicles in the network can directly communicate with RSU by V2I communication. In this type of models, the computation resources in vehicles and RSUs can be utilized. The frequent handover of V2I communication can be partly avoided, compared with individual model. It is worthwhile to point out that the handover is in a short time [24], [25]. Thus, it is reasonable to neglect the handover in this paper. The congestion of V2V and V2I communications can be relatively decreased in this type of models, compared with individual model and OVM. However, the cost in V2V communication increases sharply in this type of models, particularly in OGM, if many vehicles appear in the road segment. This is caused by the increased congestion of V2V communication. Moreover, the vehicular network in this type of models will be broken up when they cross the crossing, as vehicles may turn to different directions at the crossing. This leads to that vehicles lose connection with their vehicular network until the vehicular network is reconfigured.

To the best of our knowledge, the communication delay among the vehicles turning different directions will sharply increase, after the vehicles cross the crossing. This is because the connections between the vehicles are weak, even disconnected. On the basis of traffic rules, the vehicles in three-lane road should drive in the corresponding lanes in the light of their turning directions. That is to say, vehicles in the left (right) lane will turn left (right) at the crossing, and vehicles in the middle lane will go straight at the crossing. Inspired by this traffic rule, a vehicular network model for the directional vehicle mobility, denoted as DVNM, is proposed in this paper. In DVNM, the vehicles are configured into three vehicular subnetworks, according to the lanes they locate on. Vehicles in DVNM can communicate with each other by V2V communication in their subnetworks and with RSUs by V2I communication. But, vehicles in different subnetworks cannot communicate with each other. For example, as shown in Fig. 2, in OGM and MGM, v_2^j can communicate with v_3^j, because they are in the same road segment. But, in DVNM, v_2^j cannot communicate with v_3^j, while v_2^j can communicate with v_6^j. This is because v_2^j and v_3^j belong to different vehicular subnetworks, while v_2^j and v_6^j are in the same subnetwork. Therefore, the computation resources in vehicles are well utilized, as vehicles in the same subnetwork can communicate with each other and offload their tasks to other vehicles in DVNM. Meanwhile, the congestion in terms of communication between vehicles and RSUs is relieved, compared with the individual model. And the number of vehicles

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

in each subnetwork is small and vehicles can communicate with RSU directly, which leads to the decrease in cost of V2V communication, compared with OVM, OGM and MGM. Moreover, each vehicular subnetwork will not be separated by the crossing. Thus, DVNM is stable enough to increase the utilization of computation resources in vehicles and decrease the response time of tasks for the directional vehicle mobility. In addition, we propose three task-offloading algorithms to minimize the total response time of tasks, which aims to improve the performance of task-offloading for DVNM.

The main contributions of this paper are as follows.

1) We propose a vehicular network model DVNM for the directional vehicle mobility to provide services for vehicles in VFC. The proposed model can efficiently utilize the computation resources in vehicles and decrease the communication congestion of V2I and V2V. And it is stable enough in the crossing scenario. In addition, the problem of minimizing the average response time of the tasks originated from vehicles for the proposed model is formulated in this paper.
2) To minimize the average response time of tasks, we propose a greedy algorithm by carefully choosing neighboring vehicles as task processing helpers. Moreover, we present two bipartite matching algorithms by exploiting Kuhn-Munkras and minimum-cost maximum-flow approaches, respectively.
3) Open street map, SUMO and NS-3 are combined to evaluate the performance of DVNM and the offloading algorithms. Simulation results show that, DVNM is superior to individual model, OVM, OGM and MGM in terms of average response time of tasks when the five models have similar number of unsuccessful tasks. Meanwhile, the utilization of computation resources in vehicles in DVNM is very close to that in OGM, and it is better than that in MGM for most cases. The proposed greedy algorithm outperforms two bipartite matching algorithms and the existing greedy algorithm in terms of running time, while the two bipartite matching algorithms outperform the existing greedy algorithm in terms of the average response time of tasks.

The rest of this paper is organized as follows. The related works in VFC are discussed in section II. We describe the system model DVNM in section III. Then the performance of task-offloading is analyzed in section IV. The optimization problem is formulated in section V. The corresponding task-offloading algorithms are proposed in section VI. In section VII, we show the simulation results and the analysis for the presented model and algorithms. Finally, we conclude the paper in the section VIII.

II. Related Work

In VFC, three types of vehicular network models are widely employed to provide the delay-sensitive computing services, by alleviating the resource scarcity of vehicles. They consist of the model only with V2I communication, the model only with V2V communication, and the models with V2V and V2I communications.

A. Vehicular Network Model Only With V2I Communication

The vehicular network model only with V2I communication is denoted as individual model in this paper. For the individual model, some existing works focus on the performance improvement in terms of response time of tasks or energy consumption. For reducing the latency of computing services, authors in [12] propose a group-based network mobility management scheme by establishing multiple tunnels simultaneously. The task scheduling schemes are proposed in [21] to support mass data dissemination in real time. For energy saving, authors in [26] propose a distributed solution based on consensus alternating direction method of multipliers. A novel two-stage algorithm based on resource allocation and precoding design is proposed in [27] to minimize the total power consumption in the downlink. The renewable energy is considered to maximize the number of served vehicles by scheduling the downlink communication in [28]. However, in individual model, the communications between vehicles and RSUs are congestion and the computation resources of vehicles are wasted.

B. Vehicular Network Model Only With V2V Communication

The vehicular network model only considering V2V communication is denoted as OVM in this paper. Some existing works in OVM focus on the performance improvement for the delay-sensitive computing services or saving energy. For the delay-sensitive computing services, authors in [5] propose an adaptive algorithm based on the multi-armed bandit theory to minimize the average offloading delay. For reducing energy consumption, authors in [22] propose a task scheduling algorithm by exploiting Lyapunov optimization techniques to minimize the overall energy consumption while reducing average service delay and delay jitter. Besides, some exiting works for OVM focus on other aspects. Authors in [2] propose an algorithm to calculate the capability of VFC by utilizing the time-varying graph to describe the structure of VFC. Authors in [29] propose a VFC-aware parking reservation auction in the smart VFC system that combines parked vehicle assistance and smart parking to encourage the moving vehicles carried with CPU resources by monetary rewards for service offloading. However, in OVM, the communications between vehicles are congestion and the computation resources of RSUs are wasted.

C. Vehicular Network Models With V2I and V2V Communications

The vehicular network models with V2I and V2V communications take advantage of individual model and the model OVM. This type of models contains two vehicular network models, i.e., OGM and MGM, and they are widely used in the existing works in VFC. In OGM, existing works mainly reduce the delay of tasks. A vehicle-centric framework is proposed in [13] to guarantee low-latency communication among V2V networks underlaying V2I network. A task allocation scheme is proposed in [30] to minimize the overall delay violation probability, such that tasks can be executed within a requested time. In addition, authors in [31] propose

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

techniques based on machine learning and coded computing to address and exploit mobility of vehicles, for the purpose of minimizing the offloading delay and guaranteeing security and privacy. Besides, some existing works in OGM focus on other aspects. Authors in [32] propose an approach by introducing radio resource management methodology to improve cellular spectral efficiency. Authors in [33] propose a fog-assisted congestion avoidance scheme to reduce messaging overhead and congestion. In MGM, the existing works focus on different aspects, such as the deployment, management and routing scheme. Authors in [34] and [23] propose a software defined networking based architecture to get the global knowledge. They combine the architecture with fog computing for deployment and management in vehicular ad hoc network. However, in this type of models, the vehicular network will be broken up when vehicles cross the crossing, which results in that the connections between vehicles are weak, even disconnected.

It is worthwhile to point out that, the mentioned existing works mainly focus on the performance improvement in different aspects on the basis of the existing vehicular network models. However, the scenario of the crossing in road has not been considered in the existing vehicular network models, which leads to the performance bottleneck in VFC. In addition, the well utilization of computation resources in vehicles and RSUs can greatly improve the efficiency in computing services, which is not considered in most existing works in V2V and V2I communications. Therefore, this paper investigates the fog computing model integrating the computation resources in vehicles and RSUs, and task-offloading algorithms in the scenario including the crossing to provide computing services with low delay.

III. System Model

We consider the scenario that a road is comprised of crossings and the straight road segments. As shown in Fig. 1, a road is divided into several road segments, and a RSU is deployed in each road segment. Let $S = \{s_1, s_2, \cdots, s_c\}$ and $U = \{u_1, u_2, \cdots, u_c\}$, S and U are the sets of road segments and RSUs, where c is the number of road segments. Each road segment s_i is under the communication coverage by RSU u_i for $i \in \{1, 2, \cdots, c\}$. In the model, the RSUs are Internet-connected and placed with an edge sever to serve the vehicles on the road. The connection among RSUs is wired, so that RSUs can communicate with each other through infrastructure-to-infrastructure (I2I) communication. Vehicles can connect with RSU by V2I communication supported by the long term evolution (LTE), if they are in the communication coverage of RSU. Meanwhile, vehicles can connect with each other by V2V communication supported by 802.11p.

Let $V = \{V^1, V^2, \cdots, V^c\}$, where V^j is the set of vehicles in the road segment s_j for $j \in \{1, 2, \cdots, c\}$. Let $V^j = \{v_1^j, v_2^j, \cdots, v_{|V^j|}^j\}$, where $|V^j|$ is the number of vehicles in s_j and v_i^j is the ith vehicle in s_j for $i \in \{1, 2, \cdots, |V^j|\}$. As we known, the vehicles in the road segment which are close to the crossing will drive to different directions with high probability. This results in that the vehicular network is far from steady, which brings the decrease of the quality of servers for users.

TABLE I
Notations in System Model

Notations	Meaning
S	The set of road segments in the road.
s_i	The ith road segment.
c	The number of road segments.
U	The set of RSUs in the road.
u_i	The ith RSU.
V	The set of vehicles in the road.
V^j	The set of vehicles in s_j.
$\|V^j\|$	The number of vehicles in s_j.
v_i^j	The ith vehicle in s_j.
N_l^j	The left direction vehicular networks in s_j.
N_s^j	The straight direction vehicular networks in s_j.
N_r^j	The right direction vehicular networks in s_j.

For example, in Fig. 2, the task originated from vehicle v_2^j is offloaded to the vehicle v_3^j to process before the vehicles in V^j cross over the crossing. But v_2^j and v_3^j has already driven through the crossing and to different directions, when processing of the task is completed. It is time-consuming that v_2^j receives the results of execution for the task from v_3^j. This is because v_2^j is far away from v_3^j, the connection between v_2^j and v_3^j is weak, even disconnected, which leads to the sharp increase in the communication time. This motivates us to construct a vehicular network with stability for providing high quality of server.

This paper is under the following reasonable assumptions. The vehicles obey to the traffic rules and the directional mobility. In other words, vehicles only can go straight or turn to right / left, and they can not suddenly change lanes, when vehicles approach the crossing. Inspired by the traffic rule that the vehicles in three-lanes road drive in the corresponding lanes according to their turning directions, when they are close to the crossings. Therefore, we propose a vehicular network for the directional vehicle mobility, denoted as DVNM, in which the vehicles with the same turning direction are configured into a subnetwork, as shown in Fig. 2. Now we describe the configuration of DVNM.

Let N_l^j, N_s^j and N_r^j be the vehicular subnetwork of left, straight and right lanes in s_j, respectively. The vehicles in s_j are classified into three categories according to their turning directions of the next crossing, and the vehicles in each category are assigned into the same vehicular subnetwork. In detail, vehicle v_i^j will be assigned to N_r^j if it turns right at the next crossing. Similarly, vehicle v_i^j will be assigned to N_s^j (N_l^j) if it goes straight (turns left) at the next crossing. All vehicles in their subnetworks can connect with other vehicles which is in the same subnetwork by V2V communication. Meanwhile, all vehicles can connect with RSU directly by V2I communication. Note that, vehicles in different subnetworks cannot connect with each other by V2V communication directly.

Table I summarizes the notations in the system model. The configuration algorithm for DVNM can be formally described

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 CONFIG: Configuration of Directional Group Mobility Vehicular Networks

Input: Set of vehicles V^j, RSU u_j
Output: N_r^j, N_s^j, N_l^j

```
N_r^j := null; N_s^j := null; N_l^j := null
for i := 1 to |V^j| do
  if v_i^j turns right at the next crossing then
    N_r^j := N_r^j ⊎ v_i^j;
  else
    if v_i^j goes straight at the next crossing then
      N_s^j := N_s^j ⊎ v_i^j;
    else
      N_l^j := N_l^j ⊎ v_i^j;
    end if
  end if
end for
return N_r^j, N_s^j, N_l^j;
```

as Algorithm 1. For simplicity, we use $N \biguplus v$ to indicate that the vehicle v is assigned to network N and it is configured the protocol 802.11p and LTE to establish the V2V and V2I communication in Algorithm 1.

Note that the DVNM with some minor adjustment can adapt to different intersection scenarios, such as T-junction, two-lanes or more than three-lanes roads. For the different scenarios, the size and the number of subnetworks need to be appropriately adjusted. In this paper, we focus on the DVNM in crossing scenario which is common in the metropolis. Besides, the subnetworks in DVNM is configured or initialized at the road segment which is adjacent to the crossing. Then, the vehicles in the same subnetwork keep connecting with each other until they drive to the road segment which is adjacent to the next crossing. In addition, for the case of that vehicles drive to the coverage of the other RSU during the execution of tasks, the feedbacks of the tasks are offloaded from the original RSU executing the task to the current RSU covering the vehicles. Then, the current RSU offloads the feedbacks of the tasks to the vehicles.

IV. Performance Analysis for Task-Offloading

In this paper, we evaluate the task-offloading performance in terms of the average response time of the offloaded tasks, i.e., the computation delay and communication delay.

In the vehicular network, some vehicles have computation tasks during a certain time period. These generated tasks are denoted as $R = \{r_1, r_2, \cdots, r_n\}$, where n is the number of total tasks in vehicular network during a certain time period. For each task r_i, we use $r_i = \{b_0, b_1, t_i, o_i, d_i\}$ to represent the details of a task, for $i = \{1, 2, \cdots, n\}$. Here, b_0 and b_1 are the sizes of uploaded and downloaded data for r_i, and t_i is the time that r_i arrives. Note that the uploaded data is the original size of task r_i, and the downloaded data is the result feedback after the execution of r_i is finished. In addition, o_i is the vehicle generated the task r_i and d_i is the executed place for r_i. That is to say, task r_i is offloaded to d_i from o_i to process, where $o_i \in V$ and $d_i \in V \cup U$. In this paper, one vehicle can generate only one task.

Let V_R and D be the sets of vehicles generated the tasks and the corresponding place to execute tasks, respectively, i.e., $V_R = \{o_1, o_2, \cdots, o_n\}$ and $D = \{d_1, d_2, \cdots, d_n\}$. In addition, for the task r_i we define p_i^0 and p_i^1 as follows, where $i = \{1, 2, \cdots, n\}$.

$$p_i^0 = \begin{cases} 1, & d_i \in V, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

and

$$p_i^1 = \begin{cases} 1, & d_i \in U, \\ 0, & \text{otherwise}. \end{cases} \tag{2}$$

A. Computation Delay

Let ω be the computation intensity in CPU cycles per bit which is the required CPU cycles to compute one bit input data. Thus, the required CPU cycles for the task is $b_0\omega$. Each computation task can be only executed either locally on another vehicle in its subnetwork or remotely on an edge server through task-offloading. Note that the computation capability of vehicles is assumed to be identical for all vehicles in this paper. Therefore, let f_0 and f_1 be the CPU clock frequency of vehicle and edge server deployed in RSU, respectively. Assume t_i^c is the computation delay for the task r_i executed in a vehicle or RSU. Therefore, t_i^c is defined as follows.

$$t_i^c = \begin{cases} \dfrac{b_0\omega}{f_0}, & p_i^0 = 1, \\ \dfrac{b_1\omega}{f_1}, & p_i^1 = 1. \end{cases} \tag{3}$$

B. Communication Delay

Assume that $L_{t_i}(o_i, d_i)$ and $l_{t_i}(o_i, d_i)$ indicate the channel loss and the distance between node o_i and d_i at the time t_i, respectively. For V2V communication, the channel loss is approximately equal to deterministic path loss [35]. Therefore, $L_{t_i}(o_i, d_i)$ [35], [36], in dB, is defined as

$$L_{t_i}(o_i, d_i) = \begin{cases} A_o(l_{t_i}(o_i, d_i))^{-\alpha}, & p_i^0 = 1, \\ 103.4 + 24.2\log_{10}(l_{t_i}(o_i, d_i)), & p_i^1 = 1. \end{cases} \tag{4}$$

where A_o is suitable path loss coefficient and $\alpha > 1$ is the path loss exponent.

In the paper, the channel model for V2I and V2V communication is based on orthogonal frequency division multiplexing (OFDM). For V2I communication, one RSU could serve more than one vehicle simultaneously, by dividing the bandwidth into several equal channels. For V2V communication, one vehicle could serve only one vehicle simultaneously, so the channel will not be divided. In addition, we assume that there are not interference in different vehicular subnetworks in the paper. We denote the channel bandwidth for V2V communication as B_0 and for V2I communication as B_1 at the time t_i. Therefore, the channel bandwidth for task r_i, denoted as B_μ, is defined as,

$$B_\mu = \begin{cases} B_0, & p_i^0 = 1, \\ B_1, & p_i^1 = 1. \end{cases} \tag{5}$$

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

Let P be the transmission power of vehicle, which is assumed to be same for each vehicle in the paper. N_o indicates the noise power. At the time t_i, the uplink transmission rate that node o_i uploads the task or data to d_i is denoted as $C^u_{t_i}(o_i, d_i)$, and the downlink transmission rate that o_i downloads data from d_i is denoted as $C^d_{t_i}(o_i, d_i)$. According to [5] and [35], the uplink transmission rate can be defined as

$$C^u_{t_i}(o_i, d_i) = B_\mu \log_2(\mathcal{F}(L, E, i)), \tag{6}$$

where

$$\mathcal{F}(L, E, i) = 1 + \frac{PL_{t_i}(o_i, d_i)}{N_o + \sum_{e \in E_{t_i} \setminus \{o_i\}} (PL_{t_i}(e, d_i))}, \tag{7}$$

and E_{t_i} indicates the set of nodes connected with d_i via one wireless channel at the time t_i.

Let $t(i)$ be the time when result feedback of r_i starts to return from d_i to o_i for task r_i. Therefore, $t(i)$ can be calculated by,

$$t(i) = \frac{b_0}{C^u_{t_i}(o_i, d_i)} + t^c_i. \tag{8}$$

Thus, the downlink transmission rate at the time $t(i)$ for task r_i is defined as

$$C^d_{t(i)}(o_i, d_i) = B_\mu \log_2(\mathcal{F}(L, F, i)), \tag{9}$$

where

$$\mathcal{F}(L, F, i) = 1 + \frac{PL_{t(i)}(o_i, d_i)}{N_o + \sum_{e \in F^{t(i)} \setminus \{d_i\}} (PL_{t(i)}(o_i, e))}, \tag{10}$$

and $F^{t(i)}$ is the set of nodes connected with o_i via one wireless channel at the time $t(i)$.

Assume that t^m_i is the communication delay for task r_i, which is defined as,

$$t^m_i = \frac{b_0}{C^u_{t_i}(o_i, d_i)} + \frac{b_1}{C^d_{t(i)}(o_i, d_i)}. \tag{11}$$

C. Response Time

Let T be the average response time for all offloaded tasks during a certain time period. Therefore, the average response time for the tasks in R is calculated by

$$T = \frac{1}{n} \sum_{i=1}^{n} (t^c_i + t^m_i). \tag{12}$$

V. Problem Formulation

In this paper, we consider that the task in each vehicle is indivisible and a vehicle can process only one task simultaneously. In addition, the number of the tasks processed in RSU is not limited, because an edge server which owns rich computation resources is placed with each RSU.

Definition 1: For a vehicle or RSU in DVNM, we say that,

1) *the vehicle or RSU is resource-rich, if it has idle computation resources to help the others to process the task.*
2) *Otherwise, the vehicle or RSU is resource-poor.*

Assume that $y(k)$ represents the state of node k in terms of computation resources, for $k \in U \cup V$. $y(k)$ is defined as follows.

$$y(k) = \begin{cases} 0, & k \text{ is resource-poor,} \\ 1, & k \text{ is resource-rich.} \end{cases} \tag{13}$$

Assume that $g(k)$ represents the loaded state of node k, for $k \in U \cup V$.

Definition 2: The loaded state of the node k is the number of the tasks which are offloaded to k from the other vehicles.

It is well known that the response time of offloaded tasks will directly affect the quality of server. Therefore, the decisions for task-offloading, which determines where the tasks is offloaded to, is critical to minimize the average response time of tasks. Assume that $\mathbb{N} = \{1, 2, \cdots, n\}$. The decision problem is formulated as,

$$(\mathcal{P}) \quad \min_{D} \ T \tag{14}$$

$$\text{s.t. } p^0_i + p^1_i = 1, \quad \forall i \in \mathbb{N}, \tag{15}$$

$$g(d_i) \le 1, \quad \forall d_i \in D \cap V^j, \tag{16}$$

$$y(d_i) = 1, \quad \forall d_i \in D, \tag{17}$$

$$d_i \in (N^j_\tau \setminus \{o_i\}) \cup \{u^j\}, \quad \forall o_i \in V_R \cap N^j_\tau, \tag{18}$$

$$p^0_i \in \{0, 1\}, p^1_i \in \{0, 1\}, \quad \forall i \in \mathbb{N}, \tag{19}$$

$$\tau \in \{l, s, r\}, d_i \in D. \tag{20}$$

The objective given in (14) is to minimize the average response time of tasks. Constraint (15) ensures that each task must be offloaded to one vehicle or one RSU to be processed, and a task can not be offloaded to a vehicle and a RSU simultaneously. Constraint (16) ensures that the vehicle can process only one task. Constraint (17) ensures that destinations where tasks are executed are resource-rich. Constraint (18) ensures that the task in o_i only can be offloaded to vehicles in the same vehicular subnetworks or RSU.

Note that the problem $\mathcal{P}$ is a stochastic optimization problem, where binary task-offloading, uplink transmission rate and download transmission rate need to be determined in each time slot. This is a highly challenging problem with a large amount stochastic information to be handled. The problem $\mathcal{P}$ can be solved in polynomial time, if the uplink transmission rate, the download transmission rate and the location of vehicles are fixed and the time that tasks arrive is given in advance. However, in realistic traffic system, the location of vehicles is rapidly changing as time goes on. Therefore, the uplink transmission rate and the download transmission rate are varying across time. Moreover, the tasks arrive without pre-knowledge. Meanwhile, the information is difficult to be predicted due to the high mobility of vehicles. Therefore, without pre-knowledge of above information, it is difficult to determine which decision is optimal for task-offloading.

VI. Task-Offloading Algorithms

In problem $\mathcal{P}$, the arrive time of tasks is not given in advance and it is difficult to be predicted. Therefore, in this section, we present three task-offloading algorithms to determine where the tasks are offloaded to at the time when tasks arrive.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

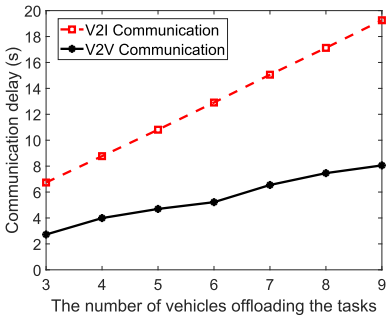


Fig. 3. The comparison of V2V and V2I communications in NS-3 for different numbers of vehicles offloading tasks simultaneously.

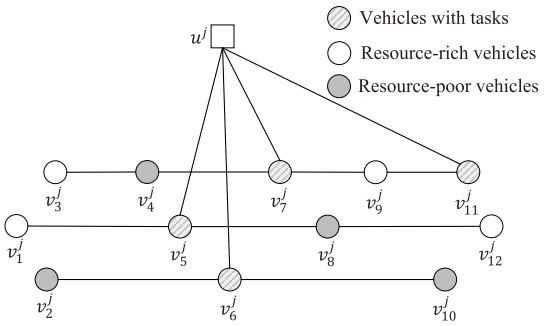


Fig. 4. An example for simplified direction-based vehicular network model.

A. Greedy Algorithm

This sub-section presents a greedy algorithm, denoted as GA, to minimize the average response time of tasks by giving preference to offloading tasks to the nearest and resource-rich vehicles in their vehicular subnetworks at the time that tasks arrive.

To the best of our knowledge, the communication delay between vehicles is far smaller than that between vehicle and RSU. This is because the distance between vehicle and RSU is much larger than that between vehicles in general. Moreover, the channel bandwidth for V2I communication is shared with vehicles in three subnetworks, while the channel bandwidth for V2V communication is only shared with vehicles in a subnetwork. Thus, the bandwidth for V2V communication is relatively enough for vehicles, compared with that of V2I communication. We now provide the simulation results, as shown in Fig. 3, for the comparison of communication delay in V2V and V2I communications in NS-3, for different numbers of vehicles which offload their tasks simultaneously. In this simulation, the length of road segment is set to 200m. In the road segment, the number of vehicles is set to 40, and the vehicles are evenly distributed on each lane. Besides, each task in vehicles is set to 1MB. The simulation results are averaged by the results of 10 repeated simulations. In addition, the communication delay of V2V (V2I) communication is the communication delay that all tasks are offloaded to other vehicles (RSUs). As shown in Fig. 3, the communication delay of V2V communication is lower than that of V2I communication for all simulated cases. Therefore, we prefer to offload tasks to the nearest and resource-rich vehicles in their vehicular subnetworks to reduce the average response time of tasks.

Let R' represent the set of the tasks which arrive at current time t, where $R' \subseteq R$. For each task r_i, GA works in the following way, where $r_i \in R'$.

1) If there are resource-rich vehicles in the vehicular subnetwork, the task in v_i^j is decided to be offloaded to the vehicle $v_{i'}^j$, which is the nearest vehicle among resource-rich vehicles.
2) Otherwise, the task in v_i^j is offloaded to u^j.

Note that RSUs have the information in roads which is covered by themselves, and the information includes positions of vehicles in the road and states of computation resources in vehicles. Therefore, the decisions for task-offloading will be performed by GA quickly. It is noteworthy that the typical greedy algorithm is customized for the objective function of the proposed problem. Specifically, the typical greedy algorithm for our proposed problem offloads tasks to the vehicles or RSUs by considering the response time for each task. However, it is time-consuming to calculate the response time for each task for all possible offloading positions. Therefore, our proposed greedy algorithm makes the offloading decisions more efficiently than the typical greedy algorithm.

TABLE II
DECISIONS FOR TASK-OFFLOADING

Tasks r_i	r_1	r_2	r_3	r_4
Vehicles with tasks o_i	v_5^j	v_6^j	v_7^j	v_{11}^j
Offloading decisions d_i	v_1^j	u^j	v_9^j	v_3^j

To better understand algorithm GA, we employ an example to show the execution of GA. Fig. 4 shows an example for simplified DVNM corresponding to the example of DVNM shown in Fig. 2 at the time t. In Fig. 4, there are three kinds of vehicles, i.e., the vehicles with tasks, resource-rich vehicles and resource-poor vehicles. v_3^j, v_4^j, v_7^j, v_9^j and v_{11}^j connect with each other to form the vehicular subnetwork N_r^j. Similarly, N_s^j comprises v_1^j, v_5^j, v_8^j and v_{12}^j, N_l^j comprises v_2^j, v_6^j and v_{10}^j. Note that two non-neighboring vehicles in the same subnetwork can communicate with each other directly by V2V communication. GA always chooses the nearest resource-rich vehicle in the corresponding subnetwork for the task. For example, there are two tasks in v_7^j and v_{11}^j in N_r^j, and two resource-rich vehicles v_3^j and v_9^j. Firstly, GA assigns the task in v_7^j to v_9^j, because v_9^j is the nearest resource-rich vehicle in N_r^j for v_7^j. Secondly, GA assigns the task in v_{11}^j to v_3^j, because v_3^j is the nearest resource-rich vehicle in N_r^j for v_{11}^j after assignment of the task in v_7^j. Similarly, we can obtain decisions for all the tasks in this moment, which is shown in the table II.

The pseudocode of GA is shown in Algorithm 2. As shown in Algorithm 2, GA runs in $O(1)$ for each task to find the resource-rich nearest vehicle, as global information is given. Therefore, GA runs in $O(n)$ to make decisions for all task-offloading.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2 GA: Greedy Algorithm

Input: R', N_l^j, N_s^j, N_r^j;
Output: D.

$D := \emptyset$.
for $r_i \in R'$ **do**
  while r_i appears **do**
    if $o_i \in N_l^j$ **then**
      if there are resource-rich vehicles in N_l^j **then**
        $d_i :=$ the nearest vehicle to o_i in these resource-rich vehicles;
      else
        $d_i := u_j$;
      end if
    else if $o_i \in N_s^j$ **then**
      if there are resource-rich vehicles in N_s^j **then**
        $d_i :=$ the nearest vehicle to o_i in these resource-rich vehicles;
      else
        $d_i := u_j$;
      end if
    else
      if there are resource-rich vehicles in N_r^j **then**
        $d_i :=$ the nearest vehicle to v_x^j in these resource-rich vehicles;
      else
        $d_i := u_j$;
      end if
    end if
    $D := D \cup \{d_i\}$;
  end while
end for
return D.

B. Bipartite Matching Algorithms

The sub-section proposes two bipartite matching algorithms to make decisions for task-offloading, which aim to minimize the average response time for all tasks appeared at the time that tasks arrive.

Now, we construct a weighted bipartite graph, i.e., $G(R', P, E)$, to indicate the corresponding vehicular network for making the decisions for task-offloading at current time t, where R' and P are two disjoint and independent sets, respectively.

Definition 3: We say that a node is available if the node is a resource-rich vehicle or a RSU.

Let $P = \{p_1, p_2, \cdots, p_m\}$ be the set of available nodes in $V^j \cup u_j$, where m is the number of the available nodes. Note that p_m in P refers in particular to a RSU. Thus, $p_1, p_2, \cdots, p_{m-1}$ are resource-rich vehicles and p_m is a RSU. $R' = \{r_1, r_2, \cdots, r_{n'}\}$ that is the set of the tasks which arrive at current time t, where $R' \subseteq R$ and n' is the number of tasks at the time t. Meanwhile, $E = \{e_1, e_2, \cdots, e_z\}$ that is the set of all possible connections between nodes in R' and P, where z is the number of the edges in G. In other words, an edge exists only if the vehicle o_i and p_j are in the same vehicular network

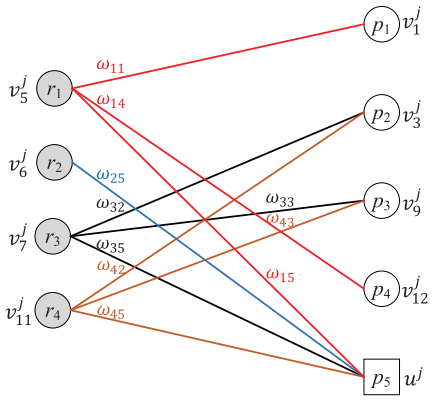


Fig. 5. An application of bipartite matching for vehicular network.

or $j = m$, for $i \in \{1, 2, \cdots, n'\}$ and $j \in \{1, 2, \cdots, m\}$. The weight of the edge in G, denoted as ω_{ij}, is the response time of that task r_i is offloaded to connected the node p_j. We use $X = (x_{ij})$ to record the matching results, i.e., the decisions for task-offloading generated by the bipartite matching algorithms for $i \in \{1, 2, \cdots, n'\}$ and $j \in \{1, 2, \cdots, m\}$. x_{ij} is defined as follows,

$$x_{ij} = \begin{cases} 1, & \text{task } r_i \text{ is offloaded to } p_j, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

In the problem $\mathcal{P}$, one task can be offloaded to only one vehicle or one RSU, and all tasks should be assigned a certain node to process. Also, one vehicle can afford only one task to execute. Thus, one node in R' or $P \setminus \{p_m\}$ can be matched once and only once. Assume that $\mathbb{N}' = \{1, 2, \cdots, n'\}$, and $\mathbb{M} = \{1, 2, \cdots, m\}$. Therefore, the problem $\mathcal{P}$ at current time t can be reduced to the following $\mathcal{P}'$.

$$(\mathcal{P}') \quad \min_X \ \frac{1}{n'} \sum_{i=1}^{n'} \sum_{j=1}^{m} \omega_{ij} \cdot x_{ij} \tag{22}$$

$$\text{s.t.} \ \sum_{j=1}^{m} x_{ij} = 1, \quad \forall i \in \mathbb{N}', \tag{23}$$

$$\sum_{i=1}^{n'} x_{ij} \leq 1, \quad \forall j \in \mathbb{M} \setminus \{m\}, \tag{24}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i \in \mathbb{N}', j \in \mathbb{M}. \tag{25}$$

The objective of $\mathcal{P}'$ is given in (22) to minimize total response time of tasks at current time t. Constraint (23) ensures that one task in R' can be matched exactly once. Constraint (24) ensures that one node in $P \setminus \{p_m\}$ can be matched exactly once.

Fig. 5 shows a simplified transformation from DVNM in Fig. 4 to bipartite matching at current time. R' is composed of all tasks at current time, i.e., $R' = \{r_1, r_2, r_3, r_4\}$, where r_1, r_2, r_3 and r_4 are the tasks originated from v_5^j, v_6^j, v_7^j and v_{11}^j. P is composed of resource-rich vehicles at current time and RSU, i.e., $P = \{p_1, p_2, p_3, p_4, p_5\}$, where p_1, p_2, p_3, and p_4 represent vehicles v_1^j, v_3^j, v_9^j and v_{12}^j, and p_5 represents the RSU u^j. E in G consists of all possible connections between

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

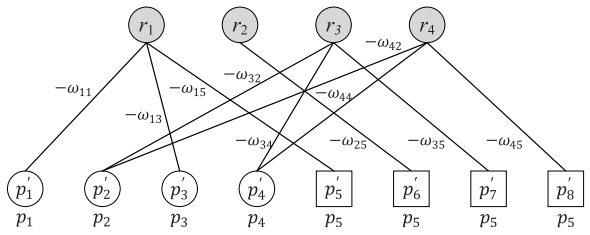


Fig. 6. An example of the construction of G' in BMA_1.

R' and P, i.e., $E = \{e_1, e_2, \cdots, e_{10}\}$. For example, there are three edges connected with P for task r_3. This is because that the vehicle v_7^j with r_3 is in N_r^j, and there are two resource-rich vehicles, i.e., v_3^j and v_9^j, in N_r^j. Therefore, r_3 in v_7^j only can connect with p_2, p_3 and p_5, i.e., v_3^j, v_9^j and u^j. The weight ω_{ij} is the response time of that r_i is offloaded to p_j, for $i = 1, 2, 3, 4$ and $j = 1, 2, 3, 4, 5$. For example, ω_{45} indicates the response time of that r_4 is offloaded to p_5.

Two bipartite matching algorithms are designed to solve the problem $\mathcal{P}'$. One algorithm, denoted as BMA_1, exploits the well known Kuhn-Munkras approach [37]. Another one, denoted as BMA_2, adopts minimum-cost maximum-flow approach [38]. Now we describe BMA_1 and BMA_2 in detail.

BMA_1 works in the following way. Kuhn-Munkras approach exploited in BMA_1 works for one-to-one matching. Therefore, a new graph must be constructed firstly due to that each RSU can afford more than one task. In BMA_1, a new graph $G' = (R', P', E')$ is constructed firstly. In G', P' consists of P and $n' - 1$ new nodes, where each new node is a virtual RSU. The difference between E and E' is that the edges connected to p_m in G are assigned to a set of the nodes which consists of p_m and all $n' - 1$ new nodes, such that each node in the set corresponds to one task. Meanwhile, the weight of edge, denoted as ω'_{ij}, is negative value of response time of that task r_i is offloaded to p_j.

Fig. 6 shows an example of the construction of G'.

1) Obviously, R' in G' is the same as that in G. P' consists of P and three new nodes p'_6, p'_7 and p'_8.
2) The original edges connected r_i in R' with p_5 are assigned to the set of nodes which is composed of p'_5, p'_6, p'_7 and p'_8, i.e., r_1, r_2, r_3, r_4 connect with p'_5, p'_6, p'_7 and p'_8 respectively, instead of all the tasks in R' connect with p_5.
3) Meanwhile, the weight of edges is the negative value of the original one. For example, $\omega'_{22} = -\omega_{22}$.

Then, BMA_1 employs Kuhn-Munkras approach to obtain the maximum weight matching in G'. Finally, the matching results are the decisions for task-offloading in problem $\mathcal{P}'$.

Theorem 1: The decisions for task-offloading generated by BMA_1 are optimal for the current state.

Lemma 1: The matching obtained by Kuhn-Munkras approach is a maximum weight perfect matching in G'.

Proof: In Problem $\mathcal{P}$, all tasks are required to make the decisions for task-offloading. This is equal to that the matching in G' must be a perfect matching. According to lemma 1, the decisions for task-offloading obtained by Kuhn-Munkras approach are perfect matching with maximum weight of the edges. In addition, the weight of edge is the negative value of the response time of the task that is offloaded to the corresponding connected node. Therefore, the decisions for task-offloading obtained by BMA_1 are the optimal one at the current state. □

The pseudocode of BMA_1 is shown in Algorithm 3.

Algorithm 3 BMA_1: Bipartite Matching Algorithm Based on Kuhn-Munkras Approach

Input: $G(R', P, E)$;
Output: X.
Create a new bipartite graph $G'(R', P', E')$;
for $i := 1$ to $m + n' - 1$ **do**
  if $i \leq m$ **then**
    $p'_i := p_i$;
  else
    $p'_i := p_m$;
  end if
end for
for $i := 1$ to z **do**
  if e_i is connection between a task r_k and p_m **then**
    if there are vertices that are not connected by one edge from p'_m to $p'_{m+n'-1}$ **then**
      e'_i is the connection between r_k and new vertex that is not connected by one edge from p'_m to $p'_{m+n'-1}$;
      $\omega'_{km} := -\omega_{km}$;
    else
      e'_i is the original connection of e_i
      $\omega'_{km} := -\omega_{km}$;
    end if
  else
    /* e_i is connection between task r_k and p_j */
    e'_i is the original connection of e_i;
    $\omega'_{kj} := -\omega_{kj}$;
  end if
end for
X := KM(G'); /*Employ Kuhn-Munkras approach to find a maximum weighted matching in G' and return the matching results;*/
return X.

Assume that η is the number of the vertexes in P', i.e., $\eta = m + n' - 1$. As shown in Algorithm 3, BMA_1 runs in $O(\eta)$ to create a new set P' in bipartite graph G', and $O(z)$ to create a new set of edge E'. Thus, BMA_1 runs in $O(\eta + z)$ to create G''. In addition, Kuhn-Munkras approach runs in $O(\eta^3)$. Therefore, BMA_1 runs in $O(\eta^3)$ to make the decisions for task-offloading at the time t.

Now we describe the algorithm BMA_2. Initially, a new graph $G''(Q, E'')$ is constructed in BMA_2. G'' is constructed as follows.

1) In G'', Q includes not only R' and P, but also a source vertex s and a terminal vertex t. Hence, $Q = R' \cup P \cup \{s\} \cup \{t\}$.
2) A directed edge from s to each r_i is added to E'' for $r_i \in R'$, and a directed edge from each p_j to vertex t is added to E'' for $p_j \in Q$. The original edges in E

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

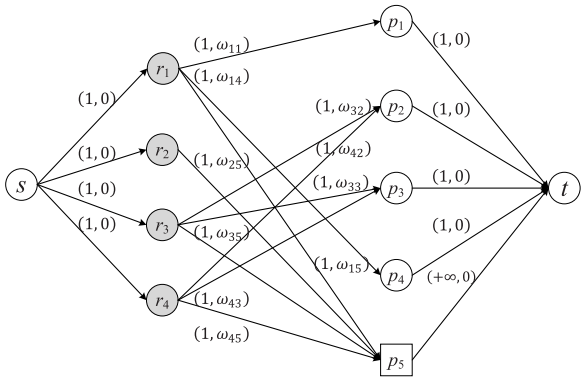


Fig. 7. An example of the construction of G'' in BMA_2.

are changed to the directed edges from R' to P. These directed edges are added to E''.

3) The weights of edges in G'' are the same as that in G, except the edges connected with s or t. The weights of the edges which connect with s or t are 0.
4) The capacities of all edges in E'' are 1, except the directed edge from p_m to t. The capacity of the directed edge from p_m to t is $+\infty$, as u^j has no limit in capacity.

Fig. 7 shows an example of construction of $G''(Q, E'')$. For Q, there are two new vertexes s and t which are added into G'' compared with G. Thus, $Q = \{s, r_1, r_2, r_3, r_4, p_1, p_2, p_3, p_4, p_5, t\}$. For E'', all edges in G are changed to the directed edges from R' to P, and the directed edges from s to R' and from P to t are added. For example, the original edge connected with r_1 and p_5 is changed into the directed edge from r_1 to p_5. In the notation (i, j) on each edge in Fig. 7, i and j indicate the capacity and the weight of the edge, respectively. The capacity is 1 for each edge in G'', except for the edge from p_5 to t. The capacity of the edge from p_5 to t is $+\infty$. All weights of the edges from r_i to p_j are the original weight ω_{ij} for $i = 1, 2, 3, 4$ and $j = 1, 2, 3, 4, 5$. Meanwhile, the weights of the edges connected with s or t are 0.

After the construction of G'', BMA_2 employs minimum-cost maximum-flow approach to generate a minimum weight perfect matching in G'' for solving the problem $\mathcal{P}'$.

The pseudocode of BMA_2 is shown in Algorithm 4. In BMA_2, ω''_{ij} represents the weight of the edge from r_i to p_j. Similarly, ω''_{si} represents the weight of the edge from s to r_i, and ω''_{jt} represents the weight of the edge from p_j to t, for $i \in \{1, 2, \cdots, n'\}$ and $j \in \{1, 2, \cdots, m\}$. We use $c(i, j)$ to indicate the capacity of the edge from p_i to q_j. Similarly, $c(s, i)$ indicates the capacity of the edge from s to p_i, and $c(j, t)$ indicates the capacity of the edge from p_j to t, for $i \in \{1, 2, \cdots, n'\}$ and $j \in \{1, 2, \cdots, m\}$.

Assume that ℓ_e and ℓ_v are the number of edges and vertexes in G'', respectively. As shown in Algorithm 4, BMA_2 runs in $O(\ell_e)$ to create a graph G''. Meanwhile, minimum-cost maximum-flow approach runs in $O({\ell_v}^3)$. Therefore, BMA_2 runs in $O({\ell_v}^3)$ to make the decisions for task-offloading at the time t.

Algorithm 4 BMA_2: Bipartite Matching Algorithm Based on Minimum-Cost Maximum-Flow Approach

Input: $G(R', P, E)$;
Output: X.

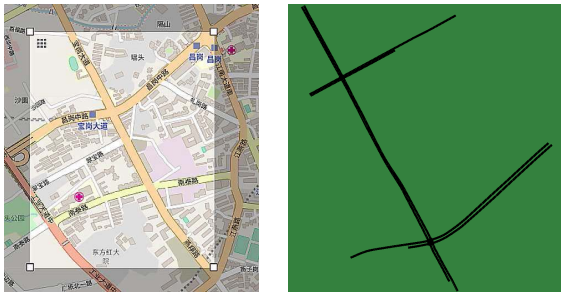
```
Create a new bipartite graph G''(Q, E''), and new vertexes s and t;
Q := R' ∪ P ∪ {s} ∪ {t};
for each node k in Q do
  if k = s then
    for i := 1 to n' do
      Establish directed edge from s to r_i;
      ω''_si := 0;
      c(s, i) := 1;
    end for
  else if k = t then
    for j := 1 to m do
      Establish directed edge from p_j to t;
      ω''_jt := 0;
      if j = z then
        c(j, t) := +∞;
      else
        c(j, t) := 1;
      end if
    end for
  else
    Change edge connected with r_i and p_j to directed edge from r_i to p_j.
    ω''_ij := ω_ij;
    c(i, j) := 1;
  end if
end for
X := MCMF(G''); /*Employ Minimum-cost Maximum-flow approach to find a minimum weighted and maximum matching in G'' and return the matching results;*/
return X.
```

VII. Experimental Results and Analysis

In this section, the open street map, simulation of urban mobility (SUMO) and NS-3 [39] are combined to co-simulate the realistic scenario for evaluating the performance of task-offloading on our proposed network model and the corresponding algorithms. Now, we describe the setting of simulation in detail.

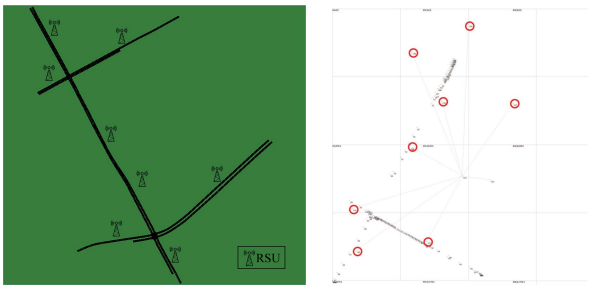
1) **Road Topology.** As shown in Fig. 8(a), a road topology of a street in Guangzhou used in simulation is obtained by open street map. After a simplified processing, the data is imported from map into SUMO. The road topology in SUMO is shown in Fig. 8(b), with a surface of 1800×1200 m^2 consisting of two crossings and some straight road segments.
2) **Vehicle Trajectory.** In simulation, we generate the movements of vehicles by utilizing "randomTrips.py" in SUMO, which is well-known open-source traffic simulator to model realistic vehicle scenario. Note that, as shown in Fig. 9, there are 8 RSUs, which are placed in the road side for simulation. Fig. 9(a) shows the

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.



(a) The road topology obtained by open street map. (b) The road topology in SUMO.

Fig. 8. The road topology exploited in simulation.



(a) Locations of RSUs in SUMO. (b) Locations of RSUs in NS-3.

Fig. 9. The locations of RSUs in simulation.

TABLE III
DEFAULT VALUES OF SYSTEM PARAMETERS

System parameters	Default values
B_0	10 MHz
B_1	30 MHz
f_0	2 GHz
f_1	4 GHz
P	0.1 W
N_o	10^{-13} W
A_o	-21.06 dBm
α	1.68
Length of road segment	250 m
Each time period in NS-3	50s

TABLE IV
DATASETS

Datasets	CASE-1	CASE-2	CASE-3	CASE-4	CASE-5
$b_0\omega$ (M cycles)	200	400	600	800	1000
b_0(KB)	200	400	600	800	1000
b_1(KB)	40	80	120	160	200

locations of RSUs in the road topology, and Fig. 9(b) shows the corresponding locations of RSUs in the NS-3. In addition, each vehicle has equal probability to locate in different lanes, and it is prohibited to turn around in the crossing. Meanwhile, the acceleration and deceleration of each vehicle are set to the default values, i.e., the acceleration is 2.6 m/s^2 and the deceleration is 4.5 m/s^2, as mentioned in SUMO.

3) **Network Simulation.** For the simulation in the network, we simulate the communication in network by NS-3, which is a open-source discrete event network simulator. The vehicle trajectory is exploited to NS-3. Meanwhile, the V2V communication is implemented by protocol 802.11p and V2I communication is implemented by protocol LTE. In addition, the communication among RSUs is implemented by the protocol of point-to-point (P2P) with the maximum transmission rate of 100 Gb/s. Note that, in simulation, if the vehicle drives to the coverage of the next RSU before its task is completed in the original RSU, the feedbacks of the task will be sent to the vehicle via the RSU where the vehicle is in the coverage.
4) **Default Parameters.** The default parameters of simulation are the same as that in [5], [35], [40], which are shown in Table III. Meanwhile, the datasets of simulation are the same as that in [11], which are shown in Table IV. Note that each dataset indicates a kind of data size for each task.

A. Comparisons of Network Models

In this subsection, the performance of task-offloading on our proposed network model DVNM is compared with four existing vehicular network models. The first one is the individual model, denoted as IDM, which is the vehicular network model only with V2I communication. The second one is OVM, which is the vehicular network model only with V2V communication. The other two vehicular network models are OGM and MGM, which are the vehicular network models with V2V and V2I communications. In detail, only one selected vehicle can directly communicate with RSUs in OGM, while all vehicles can directly communicate with RSUs in MGM.

Meanwhile, BMA_1 is employed to generate the offloading decisions in OVM, OGM, MGM and DVNM. The task-offloading ratio, denoted as ρ, is the proportion of the vehicles with tasks in all vehicles in a road segment at the time t. ρ is defined as,

$$\rho = \frac{n}{|V^j|} \times 100\%, \tag{26}$$

where n is the number of tasks in s_j and $|V^j|$ is the number of vehicles in s_j. Note that the number of resource-rich vehicles is a half of the number of vehicles without the task that needs to offloaded in the simulation, i.e., the number of resource-rich vehicles is $(|V^j| - n)/2$.

Meanwhile, the vehicle density in a road segment is defined as follows.

Definition 4: The vehicle density in a road segment is

1) *low vehicle density, if* $|V^j| = 20 \pm 5$,
2) *middle vehicle density, if* $|V^j| = 40 \pm 5$,
3) *high vehicle density, if* $|V^j| = 60 \pm 5$.

Fig. 10 shows the performance comparison among IDM, OVM, OGM, MGM and DVNM in terms of the average response time of tasks on different cases of datasets for middle vehicle density and $\rho = 50\%$. As shown in Fig. 10, the average

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

TABLE V
THE PROPORTION OF UNSUCCESSFUL TASKS IN SIMULATION

$\rho(\%)$	Low vehicle density (%)					Middle vehicle density (%)					High vehicle density (%)				
	φ_i	φ_v	φ_o	φ_m	φ_d	φ_i	φ_v	φ_o	φ_m	φ_d	φ_i	φ_v	φ_o	φ_m	φ_d
10	13.33	0.00	11.11	0.00	11.11	14.29	0.00	8.82	0.00	5.71	0.00	11.10	12.50	5.09	2.04
20	13.33	0.00	9.68	0.00	12.50	9.23	15.73	23.44	7.22	9.09	0.00	13.23	16.49	5.37	0.00
30	11.11	16.70	6.38	0.00	9.30	13.40	30.28	21.88	8.88	10.20	0.00	18.98	18.25	2.98	1.47
40	4.26	29.10	10.34	5.47	5.45	12.70	39.52	34.75	10.34	7.81	0.00	35.64	20.08	6.16	0.56
50	9.23	47.46	12.31	4.44	7.69	15.92	57.01	40.34	13.87	11.32	0.00	54.54	44.34	5.09	0.90
60	12.20	63.83	14.63	9.34	9.76	21.16	70.11	45.50	12.81	10.94	0.00	69.95	50.00	5.37	0.37
70	13.83	75.45	19.15	9.25	5.32	26.77	77.46	55.00	15.27	16.36	0.00	78.56	62.35	2.98	0.00
80	18.75	84.66	21.50	11.53	10.28	25.50	86.65	60.55	19.62	22.44	0.00	87.82	62.85	1.55	0.28
90	10.83	92.10	20.00	11.68	9.17	28.98	92.99	60.00	25.23	26.32	7.67	93.08	63.12	1.54	0.00

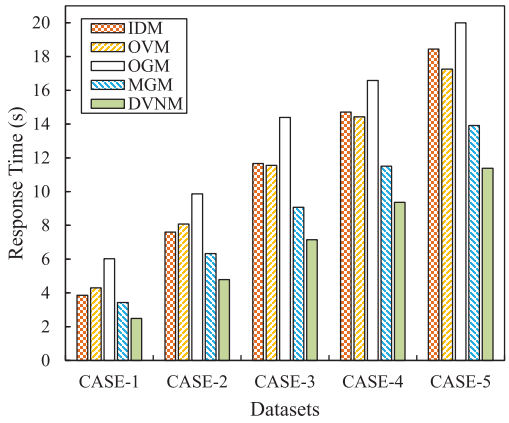


Fig. 10. The average response time for different datasets.

response time of tasks for DVNM is always less than that of IDM, OVM, OGM and MGM for all cases of datasets. Meanwhile, the average response time of tasks on DVNM grows slower than that on IDM, OVM, OGM and MGM with the increase of data size for each task. For example, in CASE-1, the average response time of tasks is 2.49s, 3.86s, 4.30s, 6.02s and 3.43s, for DVNM, IDM, OVM, OGM and MGM, respectively. The average response time on DVNM is reduced by 1.36s, 1.81s, 3.53s and 0.94s, respectively, in comparison to IDM, OVM, OGM and MGM. However, in CASE-5, the average response time of tasks is 11.38s, 18.44s, 17.25s, 20.00s, and 13.91s, for DVNM, IDM, OVM, OGM and MGM, respectively. The average response time on DVNM is reduced by 7.06s, 5.87s, 8.62s and 2.53s, respectively, in comparison to IDM, OVM, OGM and MGM. Therefore, the performance of task-offloading on DVNM is better than that on IDM, OVM, OGM and MGM for all datasets. Meanwhile, the performance of DVNM clearly increases with the increasing data size of each task.

Assume that φ represents the proportion of unsuccessful tasks in simulations, which is defined as follows.

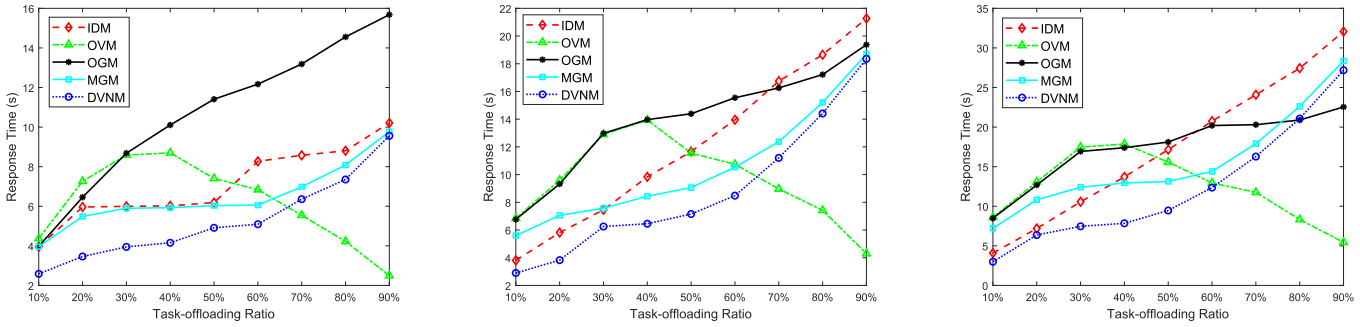
$$\varphi = \frac{\tau}{n} \times 100\%, \tag{27}$$

where τ is the number of tasks which are lost or uncompleted in a time period. We use φ_i, φ_v, φ_o, φ_m and φ_d to represent the proportion of unsuccessful tasks in the simulation for IDM, OVM, OGM, MGM and DVNM, respectively.

Fig. 11 shows the performance for IDM, OVM, OGM, MGM and DVNM in terms of the average response time of tasks, for different task-offloading ratios and vehicle densities with dataset of CASE-3.

Fig. 11(a) shows the performance for IDM, OVM, OGM, MGM and DVNM in terms of the average response time of tasks, on low vehicle density. As shown in Fig. 11(a), the average response time of tasks for DVNM is always less than that of IDM, OGM and MGM in low vehicle density for all task-offloading ratios. In addition, DVNM outperforms OVM in terms of the average response time of tasks, when the task-offloading ratio ρ is less than 60%. For example, for the case of that $\rho = 30\%$, the average response time of tasks on DVNM decreases by 33.08%, 34.15%, 54.02% and 54.51%, respectively, compared with MGM, IDM, OVM and MGM. Meanwhile, the response time of tasks on DVNM is close to that of IDM and MGM, but far less than that of OGM, when the task-offloading ratio is 90%. This is because the tasks are almost offloaded to RSUs in MGM and DVNM, as few vehicles are resource-rich. Therefore, the response time of tasks for MGM and DVNM are approximate to that of IDM. Especially, all tasks are offloaded to RSU when the task-offloading ratios are 100%, which implies that the average response time of tasks on MGM and DVNM is identical to that on IDM. But, in OGM, the tasks are offloaded to RSU by the gateway vehicle, which leads to long response time for offloading tasks to RSU. Besides, in Fig. 11(a), the average response time of tasks on OVM is less than that of other models, when $\rho \geq 70\%$. This is because only the computation resources in vehicles are utilized in OVM, and the number of vehicles is less than the number of tasks when $\rho > 30\%$. Therefore, there are not enough computation resources in OVM for completing all tasks. As a result, a large number of tasks can not be completed. From Table V, the proportion of unsuccessful tasks for OVM is large, when $\rho > 30\%$.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.



(a) Low vehicle density. (b) Middle vehicle density. (c) High vehicle density.

Fig. 11. The average response time for different task-offloading ratios and vehicle densities.

For example, for the case of that $\rho = 90\%$, the proportion of unsuccessful tasks for OVM is larger than 92.00% on different vehicle densities. This implies that almost all tasks are not executed. Therefore, we can conclude that IDM, OGM, MGM and DVNM outperform OVM in terms of the response time of tasks, if the five models have similar proportion of unsuccessful tasks, on the basis of the response time of tasks in the cases of that $\rho \leq 30\%$.

Fig. 11(b) and Fig. 11(c) show the performance comparison among IDM, OVM, OGM, MGM and DVNM in terms of the average response time of tasks, on middle vehicle density and high vehicle density, respectively. As shown in Fig. 11(b) and Fig. 11(c), DVNM performs better than IDM, OVM, OGM and MGM in most cases in terms of the average response time, both on middle vehicle density and high vehicle density. For example, the average response time is 6.45s, 8.43s, 9.84s, 13.93s and 13.97s for DVNM, MGM, IDM, OVM and OGM, respectively, on the middle vehicle density. Thus, the task-offloading performance on response time for DVNM is improved by 23.49%, 34.45%, 53.68% and 53.83%, compared with MGM, IDM, OVM and OGM, respectively. As shown in Fig. 11(c), the average response time for DVNM is larger than that of OGM, when $\rho > 70\%$. This is because a large number of tasks are lost or not completed in OGM within the simulation time, as shown in Table V. For example, in the case of that $\rho = 90\%$ and high vehicle density, the proportion of unsuccessful tasks for OGM reaches to 63.12%. However, the proportions of unsuccessful tasks for IDM, MGM and DVNM are 7.67%, 1.54% and 0.00% in this case, respectively. Therefore, the response time on OGM is relatively small in a time period in some cases, while the performance for task-offloading is not better than that of IDM, MGM or DVNM if all tasks are successfully completed.

In conclusion, DVNM can perform better than IDM, OVM, OGM and MGM, in terms of the average response time of tasks, on different vehicle densities and task-offloading ratios in dataset of CASE-3, if the five models have similar proportion of unsuccessful tasks.

Let ϑ be the ratio of resource utilization of vehicles in a road segment, which is defined as follows.

$$\vartheta = \frac{\hbar_v}{\hbar_a} \times 100\%, \tag{28}$$

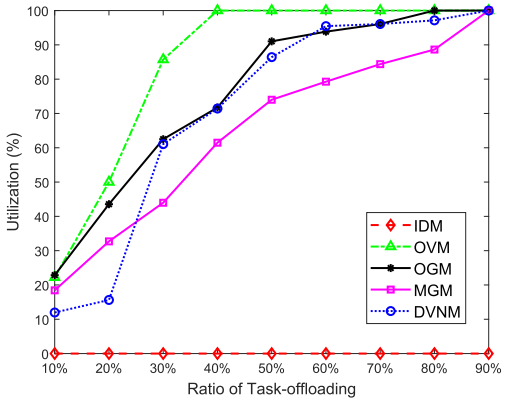


Fig. 12. The resource utilization of vehicles in different task-offloading ratios.

where $\hbar_v$ is the number of tasks which are processed in vehicles and $\hbar_a$ is the number of resource-rich vehicles in the road segment.

Fig. 12 shows the performance for IDM, OVM, OGM, MGM and DVNM in terms of resource utilization of vehicles in different task-offloading ratios on middle vehicle density with dataset of CASE-3. As shown in Fig. 12, OVM, OGM, MGM and DVNM outperform IDM in terms of the resource utilization of vehicles for all cases. In addition, the ratios of resource utilization of vehicles for DVNM are very close to that on OGM and larger than that for MGM, for the cases of $\rho \geq 30\%$. For example, for the case of $\rho = 40\%$, the value of ϑ for DVNM is 71.43%, for OGM is 71.72%, and for MGM is 61.48%. For the cases of $\rho \leq 30\%$, the value of ϑ for DVNM is slightly less than that for OGM, and the value of ϑ for DVNM is not always larger than that for MGM. This is caused by lacking resource-rich vehicles to meet task requests in a certain subnetwork on DVNM, or the resource-rich vehicles are far away from the vehicles generated tasks. Note that, in Fig. 12, the value of ϑ for IDM is always 0.00% for all cases, as all the tasks are offloaded to RSUs, instead of vehicles. In addition, the value of ϑ for OVM is 100%, when $\rho \geq 30\%$. This is because OVM only utilizes the computation resources in vehicles. Therefore, OVM performs better than other models in terms of the resource utilization of vehicles. In summary, DVNM, OGM, OVM and MGM take full advantage of the computation resources in vehicles in comparison with IDM.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

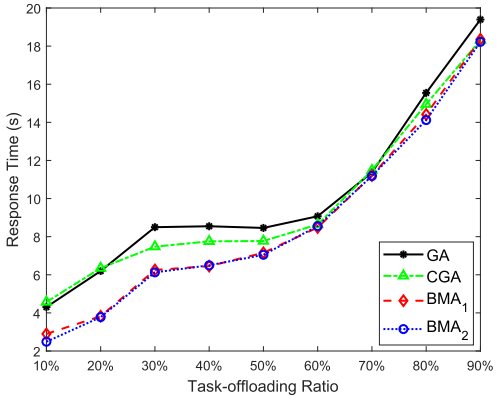


Fig. 13. The average response time for different task-offloading algorithms in DVNM.

In conclusion, DVNM outperforms IDM, OVM, OGM and MGM in terms of the average response time of tasks, if the five models have similar proportion of unsuccessful tasks. In addition, the computation resources in vehicles are well utilized in DVNM, compared with IDM and MGM. Meanwhile, the proportion of unsuccessful tasks for DVNM is smaller than that of IDM, OVM, OGM and MGM in most cases.

B. Comparisons of Task-Offloading Algorithms

In this subsection, we evaluate the performance of task-offloading algorithms on our proposed network model DVNM. Let CGA indicate the greedy algorithm proposed in [41]. In order to make fair comparison, CGA is customized for DVNM. Specifically, CGA firstly calculates the response time for each task that is offloaded to each resource-rich vehicle and RSU. Meanwhile, a minimum response time of each task is regarded as the value of the task. Then, CGA makes the offloading decision for the task with minimum value among all tasks. In other words, the task with minimum value is decided to be offloaded to the place where the response time of the task is minimum. Finally, CGA removes the task with minimum value and the corresponding execution place, then these steps repeat until that the offloading decisions are made for all tasks.

Fig. 13 shows the performance comparison among the algorithms GA, CGA, BMA_1 and BMA_2 in terms of average response time of tasks for different task-offloading ratios in DVNM with dataset of CASE-3 and middle vehicle density. As shown in Fig. 13, BMA_1 and BMA_2 perform better than GA and CGA in terms of response time of tasks. In addition, the average response time for BMA_1 and BMA_2 is very similar for all cases. Moreover, GA outperforms CGA when $\rho \leq 20\%$. The response time of tasks for GA is slightly larger than that of CGA, when $\rho \geq 20\%$. Besides, the average response time of tasks of algorithm GA is close to that of the algorithms BMA_1 and BMA_2, especially in the case of that $\rho \geq 60\%$. For example, for the case of $\rho = 70\%$, the average response time of tasks is 11.3696s for GA, and it is 11.2066s for BMA_1 and 11.1826s for BMA_2. The difference of the performance is no more than 1.7%.

Table VI shows the performance comparison for the algorithms in terms of running time in DVNM with dataset of CASE-3 and middle vehicle density. It is clear that, the running time of GA is far smaller than CGA, BMA_1 and BMA_2 for all cases. The running time for BMA_1 is slightly smaller than that of CGA for most cases. In addition, BMA_1 and CGA run faster than BMA_2 for all cases.

TABLE VI
THE RUNNING TIME OF TASK-OFFLOADING ALGORITHMS

$\rho(\%)$	Running time (μs)			
	GA	CGA	BMA_1	BMA_2
10	26.63	82.00	103.88	411.88
20	39.38	123.20	171.00	567.00
30	43.13	168.90	159.75	704.63
40	47.25	192.20	268.25	802.88
50	62.75	256.10	206.63	931.38
60	54.13	277.80	240.13	1050.75
70	61.00	298.30	240.88	1113.00
80	70.57	379.40	222.50	1198.00
90	62.50	338.50	179.63	1238.75

In conclusion, algorithm GA has significantly accelerated CGA, BMA_1 and BMA_2, while the increasing of response time of tasks is acceptable. In addition, BMA_1 and BMA_2 perform better than GA and CGA in terms of the average response time of tasks.

VIII. CONCLUSION

A network model named DVNM for directional vehicle mobility has been proposed in this paper, in order to provide better services for vehicles and traffic systems. In DVNM, vehicles have been configured into three stable and efficient directional fog cells according to their turning directions of the next crossing. For each fog cell, vehicles can communicate with each other by V2V communication and with RSU by V2I communication. The problem of minimizing the average response time of tasks has been formulated in this paper. Meanwhile, a greedy algorithm GA, together with two bipartite matching algorithms BMA_1 and BMA_2, have been proposed for solving the optimization problem. A combined simulation platform has been completed for co-simulating the practical traffic scenario and network communication. Simulation results show that, the proposed model DVNM is superior to the existing models IDM, OVM, OGM and MGM in terms of the average response time of tasks, when the five models have similar proportion of unsuccessful tasks. For example, for the cases of task-offloading ratio of 50% and middle vehicle density, the average response time of tasks in DVNM can be saved by 37.49%, 36.74%, 47.38%, and 21.07% in comparison to IDM, OVM, OGM and MGM, respectively. Meanwhile, the computation resources of vehicles have been well utilized in DVNM. The proposed algorithms BMA_1 and BMA_2 can perform better than the GA also proposed in this paper and CGA, in terms of the average response time of tasks, while GA can significantly accelerate CGA, BMA_1 and BMA_2.

It is well known that there are many challenge problems in transportation systems. This paper mainly focuses on the performance improvement for the task-offloading with the directional vehicle mobility. However, when the vehicle changes lanes in the road segment which is adjacent to the

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

crossing, the stability of vehicular network model will be broken. Therefore, our future work will include system model for large tasks, stability of vehicular network model and mobility management of vehicles, in which the vehicles may change lanes, turn around in the road, and other relatively realistic scenarios.

REFERENCES

[1] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[2] X. Kui, Y. Sun, S. Zhang, and Y. Li, "Characterizing the capability of vehicular fog computing in large-scale urban environment," *Mobile Netw. Appl.*, vol. 23, no. 4, pp. 1050–1067, Aug. 2018.

[3] S. Abdelhamid, H. S. Hassanein, and G. Takahara, "Vehicle as a resource (VaaR)," *IEEE Netw.*, vol. 29, no. 1, pp. 12–17, Jan. 2015.

[4] H. Zhang, Q. Zhang, and X. Du, "Toward vehicle-assisted cloud computing for smartphones," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5610–5618, Dec. 2015.

[5] Y. Sun *et al.*, "Adaptive learning-based task offloading for vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3061–3074, Apr. 2019.

[6] J. Chen, G. Mao, C. Li, W. Liang, and D. Zhang, "Capacity of cooperative vehicular networks with infrastructure support: Multiuser case," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1546–1560, Feb. 2018.

[7] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape—Architectures, enabling technologies, applications, and development areas," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2391–2406, Aug. 2018.

[8] P. Papadimitratos, A. La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 84–95, Nov. 2009.

[9] E. Ahmed and H. Gharavi, "Cooperative vehicular networking: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 996–1014, Mar. 2018.

[10] X. Ge, Z. Li, and S. Li, "5G software defined vehicular networks," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 87–93, Jul. 2017.

[11] T. Taleb and K. Letaief, "A cooperative diversity based handoff management scheme," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1462–1471, Apr. 2010.

[12] M.-S. Kim and S. Lee, "Enhanced network mobility management for vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1329–1340, May 2016.

[13] S.-C. Hung, X. Zhang, A. Festag, K.-C. Chen, and G. Fettweis, "Vehicle-centric network association in heterogeneous vehicle-to-vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5981–5996, Jun. 2019.

[14] N. Gupta and V. A. Bohara, "An adaptive subcarrier sharing scheme for OFDM-based cooperative cognitive radios," *IEEE Trans. Cogn. Commun. Netw.*, vol. 2, no. 4, pp. 370–380, Dec. 2016.

[15] C. Chen, L. Liu, T. Qiu, K. Yang, F. Gong, and H. Song, "ASGR: An artificial spider-Web-based geographic routing in heterogeneous vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1604–1620, May 2019.

[16] M. M. Awad, K. G. Seddik, and A. Elezabi, "Low-complexity semi-blind channel estimation algorithms for vehicular communications using the IEEE 802.11p standard," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1739–1748, May 2019.

[17] M. Kilicarslan and J. Y. Zheng, "Predict vehicle collision by TTC from motion using a single video camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 522–533, Feb. 2019.

[18] D. Wu, Y. Zhang, L. Bao, and A. C. Regan, "Location-based crowdsourcing for vehicular communication in hybrid networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 837–846, Jun. 2013.

[19] D. Wu *et al.*, "Towards distributed SDN: Mobility management and flow scheduling in software defined urban IoT," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1400–1418, Jun. 2020.

[20] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 87–93, Feb. 2019.

[21] X. Chen and L. Wang, "Exploring fog computing-based adaptive vehicular data scheduling policies through a compositional formal method–PEPA," *IEEE Commun. Lett.*, vol. 21, no. 4, pp. 745–748, Apr. 2017.

[22] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, and J. Wang, "DEBTS: Delay energy balanced task scheduling in homogeneous fog networks," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2094–2106, Jun. 2018.

[23] J. Chen *et al.*, "Service-oriented dynamic connection management for software-defined Internet of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2826–2837, Oct. 2017.

[24] J.-M. Chung, M. Kim, Y.-S. Park, M. Choi, S. Lee, and H. S. Oh, "Time coordinated V2I communications and handover for WAVE networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 3, pp. 545–558, Mar. 2011.

[25] L. Banda, M. Mzyece, and G. Noel, "IP mobility support: Solutions for vehicular networks," *IEEE Veh. Technol. Mag.*, vol. 7, no. 4, pp. 77–87, Dec. 2012.

[26] Z. Zhou, J. Feng, Z. Chang, and X. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, May 2019.

[27] J. Shi, Z. Yang, H. Xu, M. Chen, and B. Champagne, "Dynamic resource allocation for LTE-based vehicle-to-infrastructure networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5017–5030, May 2019.

[28] W. S. Atoui, W. Ajib, and M. Boukadoum, "Offline and online scheduling algorithms for energy harvesting RSUs in VANETs," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6370–6382, Jul. 2018.

[29] Y. Zhang, C.-Y. Wang, and H.-Y. Wei, "Parking reservation auction for parked vehicle assistance in vehicular fog computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3126–3139, Apr. 2019.

[30] T. Liu, L. Sun, R. Chen, F. Shu, X. Zhou, and Z. Han, "Martingale theory-based optimal task allocation in heterogeneous vehicular networks," *IEEE Access*, vol. 7, pp. 122354–122366, 2019.

[31] S. Zhou, Y. Sun, Z. Jiang, and Z. Niu, "Exploiting moving intelligence: Delay-optimized computation offloading in vehicular fog networks," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 49–55, May 2019.

[32] I. De La Iglesia, U. Hernandez-Jayo, E. Osaba, and R. Carballedo, "Smart bandwidth assignation in an underlay cellular network for Internet of vehicles," *Sensors*, vol. 17, no. 10, p. 2217, Sep. 2017.

[33] S. Yaqoob, A. Ullah, M. Akbar, M. Imran, and M. Shoaib, "Congestion avoidance through fog computing in Internet of vehicles," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 10, pp. 3863–3877, Oct. 2019.

[34] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane, "Software defined networking-based vehicular ad hoc network with fog computing," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage.*, May 2015, pp. 1202–1207.

[35] M. Abdulla, E. Steinmetz, and H. Wymeersch, "Vehicle-to-vehicle communications with urban intersection path loss models," in *Proc. IEEE Global Commun. Conf. Workshops*, Dec. 2016, pp. 1–6.

[36] *Study on Small Cell Enhancements for E-UTRA and E-UTRAN; Higher Layer Aspects (Release 12)*, document TR 36.842, 3GPP, 2014.

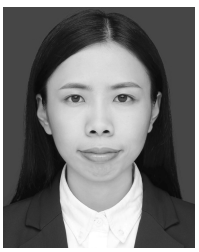
[37] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Res. Logistics*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[38] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network flows: Theory, algorithms and applications," *J. Oper. Res. Soc.*, vol. 45, no. 11, pp. 461–494, Nov. 1994.

[39] W. Liu, X. Wang, W. Zhang, Y. Lin, and P. Chao, "Coordinative simulation with SUMO and NS3 for vehicular ad hoc networks," in *Proc. 22nd IEEE Asia–Pacific Conf. Commun.*, Oct. 2016, pp. 337–341.

[40] G. Araniti, C. Campolo, M. Condoluci, A. Iera, and A. Molinaro, "LTE for vehicular networking: A survey," *IEEE Commun. Mag.*, vol. 51, no. 5, pp. 148–157, May 2013.

[41] W. Xia, T. Q. S. Quek, J. Zhang, S. Jin, and H. Zhu, "Programmable hierarchical C-RAN: From task scheduling to resource allocation," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 2003–2016, Mar. 2019.



Yalan Wu received the B.Sc. degree from the Guangdong University of Technology (GDUT), China, in 2016, where she is currently pursuing the Ph.D. degree in computer science and technology. She was an Intern Student with Nanyang Technological University for the joint project of high performance architecture from July 2017 to September 2017. Her research interests include vehicular networks, mobile computing, fault tolerant computing, and high performance architecture. She was awarded as the Best Student from the School of Computer Science and Technology, GDUT.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.

Jigang Wu (Member, IEEE) received the B.Sc. degree from Lanzhou University, China, in 1983, and the Ph.D. degree from the University of Science and Technology of China in 2000. He was with the Center for High Performance Embedded Systems, Nanyang Technological University, Singapore, from 2000 to 2010, as a Research Fellow. He was the Dean and a Tianjin Distinguished Professor with the School of Computer Science and Software, Tianjin Polytechnic University, China, from 2010 to 2015. He is currently a Distinguished Professor with the School of Computer Science and Technology, Guangdong University of Technology. He has published more than 260 articles in the IEEE TC, IEEE TPDS, IEEE TVLSI, IEEE TNNLS, IEEE TSMC JPDC, IEEE PARCO, IEEE JSA, and international conferences. His research interests include network computing and reconfigurable architecture. He serves for China Computer Federation as a Technical Committee Member in the branch committees, High Performance Computing, Theoretical Computer Science, and Fault Tolerant Computing.

Long Chen received the B.E. degree in computer science from Anhui University, Hefei, China, in 2011, and the Ph.D. degree from the School of Computer Science and Technology, USTC, in 2016. He is currently an Associate Professor with the School of Computer Science and Technology, Guangdong University of Technology. He has published several journal articles in the IEEE TRANSACTIONS ON SERVICES COMPUTING and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. His main research interests are cognitive radio networks, network economics, and mobile computing.

Gangqiang Zhou received the B.E. degree in communication from the Hunan University of Science and Technology, China, in 2013, and the master's degree from the Guangdong University of Technology, China, in 2018. He is currently pursuing the Ph.D. degree with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His main research interests include cloud computing, incentive mechanisms, and vehicular networks.

Jiaquan Yan received the B.Sc. degree from the Guangdong University of Technology (GDUT), China, in 2019, where he is currently pursuing the master's degree in computer science and technology. His main research interests include mobile computing, incentive mechanisms, and vehicular networks.

Authorized licensed use limited to: Universitaetsbibliothek der RWTH Aachen. Downloaded on August 18,2022 at 10:22:01 UTC from IEEE Xplore. Restrictions apply.