Virtual Edge: Collaborative Computation Offloading in VANETs

Narisu Cha[1], Celimuge Wu[1(✉)], Tsutomu Yoshinaga[1], and Yusheng Ji[2]

[1] The University of Electro-Communications, Tokyo, Japan
narisu@comp.is.uec.ac.jp, celimuge@uec.ac.jp
[2] National Institute of Informatics, Tokyo, Japan

Abstract. Edge computing can reduce service latency through task offloading. Since computational resources on the edge of the network are scarce, selecting a node with rich computational capability is the key for getting a high-quality service. In this paper, we propose a virtual edge scheme where a node can offload its tasks to a virtual edge node that consists of multiple vehicles in vicinity. The relative vehicle velocity and computational capability are considered in the virtual edge selection. We compare our proposed scheme with several baseline schemes and show the superiority of the scheme.

Keywords: Virtual edge · Edge computing · Computation offloading · VANETs

1 Introduction

Edge computing [1, 2] is a method of selecting nodes that are closer to the end user to offload computation. In the Internet of Vehicles (IoV), in order to improve the network resource utilization efficiency and reduce the response delay, the edge computing has been attracting increasing interest in the past decade due to its capability of conducting computing and data caching near the end users.

In the future, connected vehicle technology will become one of the main important components of an intelligent transportation system. In order to improve traffic safety and alleviate the traffic congestion, various sensors including cameras, radars would be equipped on the vehicles, which generates numerous data. IHS Automotive forecasts that the quantity of vehicles and its in-vehicle equipment could reach nearly two billion on the roads by 2025 and every car will produce up to 30 terabytes of data each day [3]. Due to network bandwidth limitations, traditional cloud-based data processing methods are not suitable for latency-sensitive services in vehicular ad hoc networks (VANETs).

Automatic vehicles on the road can be viewed as edge nodes with computational resources to execute driving tasks such as lane identification, positioning, image processing, and traffic environmental awareness. On the one hand, from the perspective of cost, the owner has the economic capability to purchase a high-performance computer to improve the safety, because the cost of a computer is much cheaper as compared with

S. W. Loke et al. (Eds.): MONAMI 2020, LNICST 338, pp. 79–93, 2020.
https://doi.org/10.1007/978-3-030-64002-6_6

the vehicle. Deploying services and computational resources at the edge of the network is able to avoid transmitting the original data to the cloud and thereby reduce the latency.

Multiple automatic vehicles can collaborate with each other to accomplish some tasks in the scenarios that infrastructure cannot be support adequate resources, such as 3D traffic scene generation, video stream analysis, etc. These applications have the feature that intensive computation can be offloaded to some vehicles in vicinity.

Computation offloading has been studied in mobile edge computing (MEC) including energy-efficient allocation of computing resources [4], binary computation offloading [5], partial computation offloading [6, 8], and stochastic offloading [7]. Most aforementioned works consider that the computation tasks on mobile nodes are aggregated to a node, such as road-side unit (RSU) or base station (BS), which does not consider the use of vehicles as edge nodes.

In [9], Mach and Becvar have reviewed the offloading strategies in the MEC. In [10], Mao et al. concluded the method that uses joint radio and computational resource management to merge the two disciplines of wireless communications and mobile computing seamlessly. Z. Ning et al. have constructed decentralized three-layer vehicular fog computing (FVC) architecture, where moving and parked cars are viewed as fog nodes for processing local traffic data to optimize the network loading balance [11]. Vehicular multi-access networks with collaborative task offloading can guarantee low latency of the application [12]. The optimal policy for the computation offloading to multiple base stations in an ultra-dense sliced RAN is modeled by Markov decision process, and a new algorithm to offload computation based on double deep Q-network is proposed [13]. To tackle the spatial intelligence of vehicular Internet of thing (IoT), the technology of combining decentralized moving edge with multi-tier multi-access edge clustering by Q-learning is discussed in [14]. As automobiles are equipped with stronger processing units, the idea that uses parking or moving vehicles as communication and computation infrastructures has been proposed in [12–17]. Hou et al. [18] found that adjacent moving vehicles have good connectivity between each other and they can form a powerful computation cluster. They described four kinds of application scenarios and three-layer Vehicular fog computing (VFC) paradigm, then verified by real-world vehicular mobility traces and can enlarge available resources and enhance computation capability. There also have been some efforts discussing the joint resource-allocation and computation-offloading [15–23]. However, existing studies do not seriously consider how to efficiently utilize the computing capabilities of moving vehicles in dynamic vehicular networks.

In this paper, we propose virtual edge (VE), a light autonomous cooperative-distributed computing platform established by multiple nodes willing to share computational resources. When there is a need for using the computational resources of other vehicles, a node can construct a VE. The purpose is to alleviate the scarcity of computational resources at the node by using other vehicles' resources and to make the end-user have more computational capability. Our main contributions in the paper are:

- Based on mobile cloud computing, we proposed the concept of virtual edge that can be established on-demand to provide more computational resources for a node, for a better resource utilization and delay reduction.

- A method that quickly establishes VE with short delay in VANETs is proposed and then verified. The method discovers available resources at the vicinity of vehicle nodes in tens of milliseconds and generates stable and efficient edge nodes for providing more computational resources.
- The nodes consisting the VE are selected from the one-hop neighbors of the master node (the node with task establishes a VE) where the relative velocity of vehicles is considered in the VE member selection.

The reminder of the paper is as follows. We first describe system architecture of the virtual edge in Sect. 2. Section 3 describes system model and problem formulation and experimental results are shown in Sect. 4. The last section draws our conclusions.

2 System Architecture

Vehicles are viewed as mobile nodes with available computational resources in the VE architecture. The architecture added a layer on the mobile cloud computing architecture, which has three layers: Cloud, Fog/MEC, virtual edge, as shown in Fig. 1.

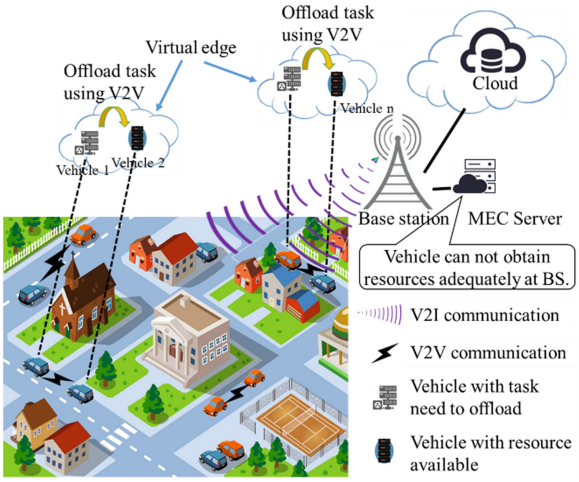

Fig. 1. System architecture.

Cloud layer: Permanent data is stored in the cloud center. For example, Traffic management servers and trusted third authority service are stored in this layer. It is in charge of monitoring global traffic data to optimize the schedule.

Fog/MEC layer: This layer consists of network infrastructures, such as cloudlet server, road-side unit (RSU), smart traffic light, router, access point (AP), and base station (BS). They process local data by collecting, aggregating, controlling and caching, after that, a huge amount of data has been compressed and send it to the cloud.

Virtual Edge: The number of RSUs on the road is limited, and therefore in some cases, it is impossible to connect with the RSU. In this case, the use of available resources for

vehicles for computation offloading is more effective than RSU. The vehicle nodes with tasks establish virtual edge on-demand among the idle nodes with available computational resources in vicinity. The lifetime of the connectivity between vehicles is predicted when VE create. After VE is established, it provides more stable computational capability and more bandwidth to the node with a task.

Vehicle node: A vehicle can be used as a router, relay node, or cache to accomplish the computing tasks. Generally, automatic vehicles have more sensing and computational resources than other vehicles. The node that with insufficient computational resources, needs to offload the computation tasks to virtual edge when the vehicle does not obtain the computing resource adequately from the base station.

Virtual edge is established by vehicle nodes to explore lower cost resources when the computational resources provided from BS/RSU do not satisfy the constraint. The architecture proposed is very suitable for dense vehicle scenes, such as urban areas, and can utilize the computational resource of parked vehicles and others running the same direction. Without modifying the cloud and fog/edge layer, the virtual edge can be used as a complement to expand the function of utilizing resources in vicinity. A task needs to be sent to the cloud, when the task cannot be completed at BS/RSU or virtual edge.

3 System Model and Problem Formulation

3.1 System Model

We consider the V2V communications based on dedicated short range communication (DSRC), such as IEEE 802.11p. Basic safety message (BSM) as defined in SAE J2735, including vehicle information such as position, velocity, direction, acceleration and so on. The standard payload of BSM is 39B, smaller than the length of a frame. In our system model, BSM also contains available computation resources like CPU frequency, caching size, available bandwidth etc. According to IEEE 1609.4, each vehicle needs to broadcast a BSM in a certain interval. Each vehicle maintains a BSM table which contains information of vehicles at the vicinity and then computes companion time between connected others directly. VE establishment process could utilize the information of BSM table and companion time in order for a faster creation of VE topology.

These nodes within single-hop communication range show different levels of capabilities, as shown in Fig. 2. In the scenario, vehicles only communicate with each other through V2V communication technology, such as IEEE 802.11p. Assume that vehicle v_2 has a task and needs to be offloaded to the other vehicles. v_1, v_3, v_4, v_5 reside within v_2's communication range. With the movement of vehicles, the connection between vehicles becomes unstable. The duration of the connection between v_2 and v_1 is shorter, because two vehicles are driving in the opposite direction. The connection between v_2 and v_5 is an unstable connection, because v_5 is at the edge of v_2's communication range and the speed difference is large. When selecting a cooperative node, it is important that consider not only the duration time of the connection, but also available computing resources.

The collection $V(t)$ refer to the set of vehicles at time t, where $V(t) = \{v_1, v_2, \ldots, v_{|V(t)|}\}$, $|V(t)|$ is the total number of vehicles at time t. $V_{ve,v}(t)$ refer to the members of VE established by node v at time t and $V_{ve,v}(t) \subseteq V(t)$. Unless otherwise specified, the vehicle set is denoted by $V(t)$.

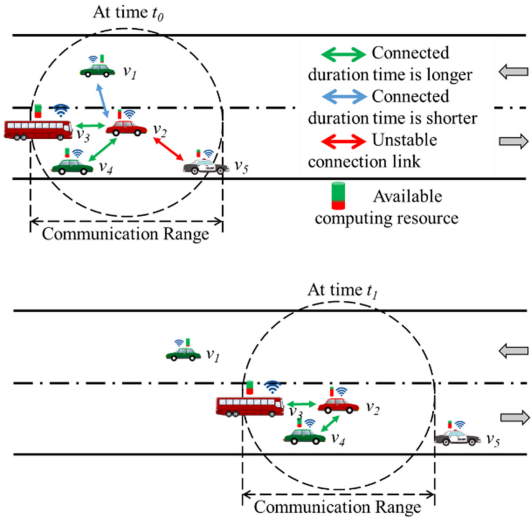

Fig. 2. An example for collaborative node selection; we need to consider the duration of the connection, and the available resources in the selection process.

3.2 Motion Model

Due to the mobility of IoV, establishing a stable VE faces great challenges. In order to return the results of computation offloading before connections between vehicles are broken, the prediction of VE duration time becomes very important. According to the interaction between vehicles, vehicle motion models can be categorized into three types in the literature: macroscopic, mesoscopic and microscopic. Macroscopic and mesoscopic models focused on the analysis results of overall performance by the parameters of the traffic, such as density, flow, arrival rate. Microscopic model describes distance headway of vehicles by the interaction with neighboring vehicles, driving behavior of individual vehicles. In this section, the motion model is a simple motion model belonging to microscopic, similar to in [24]. Below we first introduce three definitions, then introduce the concept of companion time, and use it to calculate the companion time between any two nodes, and finally using companion time to predict the duration time of VE.

Definition 1: Given V is the set of vehicle, $v, u \in V$, we say v and u are connected directly if v and u are reachable within single-hop communication range, and are denoted by $dist(v, u) < r$, where r refers to the communication range of the vehicle.

Definition 2: Given V is the set of vehicle, $\forall v, u \in V$, $\exists v_1, v_2, \ldots, v_n \in V$, $\langle v, v_1\rangle, \langle v_1, v_2\rangle, \ldots, \langle v_{n-1}, v_n\rangle, \langle v_n, u\rangle$ are all connected directly, then it is called that V is interconnected.

Definition 3: Given V is the set of vehicle, in the future period T, if $\forall t \in [0, T], \forall v, u \in V$, node v and u are connected, it is called that the duration time of V is at least T.

Following, the companion time of two nodes with direct connection is discussed. Each vehicle is only responsible for computing/updating the companion time for directly connected vehicles. The vector $\vec{P}_v = (p_{x,v}, p_{y,v}, p_{z,v})$, $\vec{v}_v = (v_{x,v}, v_{y,v}, v_{z,v})$, $\vec{a}_v = (a_{x,v}, a_{y,v}, a_{z,v})$, refer to position, velocity, acceleration of vehicle v. The vector $\vec{P}_u = (p_{x,u}, p_{y,u}, p_{z,u})$, $\vec{v}_u = (v_{x,u}, v_{y,u}, v_{z,u})$, $\vec{a}_u = (a_{x,u}, a_{y,u}, a_{z,u})$, refer to position, velocity, acceleration of vehicle u. Three elements represent values in x, y, z directions, respectively. Assuming that the acceleration is a constant before it is updated, the relative position of two vehicles is

$$\vec{P} = \vec{P}_u - \vec{P}_v = (p_{x,u} - p_{x,v}, p_{y,u} - p_{y,v}, p_{z,u} - p_{z,v}) \tag{1}$$

Similarly, relative velocity and acceleration of two vehicles are

$$\vec{a} = \vec{a}_u - \vec{a}_v = (a_{x,u} - a_{x,v}, a_{y,u} - a_{y,v}, a_{z,u} - a_{z,v}) \tag{2}$$

$$\vec{v} = \vec{v}_u - \vec{v}_v = (v_{x,u} - v_{x,v}, v_{y,u} - v_{y,v}, v_{z,u} - v_{z,v}) \tag{3}$$

The relative trajectory of node u is

$$\vec{S} = \frac{1}{2}\vec{a}t^2 + \vec{v}t + \vec{P} \tag{4}$$

Then, the length of relative trajectory can be expressed as

$$\vec{S} \cdot \vec{S} = r^2 \tag{5}$$

where r is the radius of the vehicle's communication range, $(\cdot)$ is an inner product.

Obtain from equations above

$$\left(\frac{1}{2}\vec{a}t^2 + \vec{v}t + \vec{P}_u - \vec{P}_v\right) \cdot \left(\frac{1}{2}\vec{a}t^2 + \vec{v}t + \vec{P}_u - \vec{P}_v\right) = r^2 \tag{6}$$

Expand the above formula we get

$$\begin{aligned}
&\left(\frac{1}{2}(a_{x,u} - a_{x,v})t^2 + (v_{x,u} - v_{x,v})t + p_{x,u} - p_{x,v}\right)^2 + \\
&\left(\frac{1}{2}(a_{y,u} - a_{y,v})t^2 + (v_{y,u} - v_{y,v})t + p_{y,u} - p_{y,v}\right)^2 + \\
&\left(\frac{1}{2}(a_{z,u} - a_{z,v})t^2 + (v_{z,u} - v_{z,v})t + p_{z,u} - p_{z,v}\right)^2 = r^2
\end{aligned} \tag{7}$$

In the equation above, variable t is an unknown and others are known. The value of t can be solved, and it is the companion time between node v and u. Companion time of two nodes v and u t multiple relay nodes is

$$t_{\langle v,u\rangle} = \left\{t_{x_i,x_{i+1}} | i \in \{1, n-1\}, x_1 = v, x_n = u,\ dist(x_i, x_{i+1}) < r\right\} \tag{8}$$

The duration time of VE is determined by the minimum companion (duration) time between the master node and other nodes, which can be expressed as

$$t_{dur,ve} = \left\{ t_{\langle v,u \rangle} \right\} \tag{9}$$

where node v is master node, $V_{ve,v}$ is the set of nodes included VE. In order to avoid disconnection of nodes frequently, the companion time must be greater than the threshold δ when the node is selected, $t_{\langle v,u \rangle} \geq \delta$. The recommended value of δ

$$\delta = \frac{r}{v_{road}} \tag{10}$$

where v_{road} is the maximum allowable velocity of road.

3.3 Communication Model

The efficiency of the offloading process is affected by the stability of the connection between vehicles in VANETs. We use a simple model to estimate the time of transmissions by dividing the number of the hops with the data size. An example is presented in Fig. 3, where v_2 is the vehicle with a task, and the dashed circle shows the communication range of the node. The hops between v_2 and v_4 varies from 2 to 1, and then becomes to 2. The throughput between v_2 and v_4 varies with the routing length. Figure 4 illustrates the result that the throughput changes for various number of hops under TCP (RENO) flow in the simulation, where the vehicle nodes are running at the velocity of 20 km/hour along the straight road.

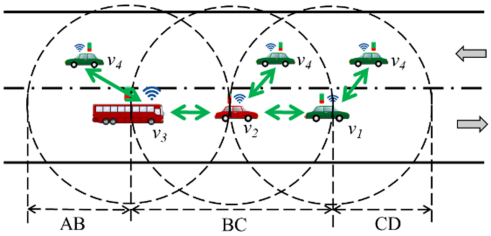

Fig. 3. The throughput varies with the number of hops between source and destination.

3.4 Computation Model

We consider that node v has a computation task $D = \left\{ d_1, d_2, \ldots, d_{|V_{ve,v}(t)|} \right\}$, that can be divided into multiple blocks. Block j is sent from vehicle v to vehicle i over IEEE 802.11p for the execution. Element d_i refer to the cycle number of block i. The variable f_i refers to the available computation resource (i.e., CPU-cycle frequency) on the vehicle i. When the computation task is offloaded onto the vehicle, the processing time t_i can be expressed as

$$t_i = \frac{d_j}{f_i} \tag{11}$$

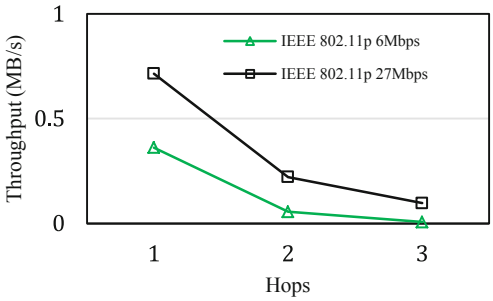

Fig. 4. The throughput of TCP over IEEE 802.11p under the different bitrates.

where $j = 1, 2, \ldots, \left|V_{ve,v}(t)\right|$ and $\forall v_i \in V_{ve,v}$. Since multiple vehicles can process the computation task Simultaneously after received the task, the computation time of VE can be obtained by

$$t_{ve,v} = \left\{t_i | i = 1, 2, \ldots, \left|V_{ve,v}(t)\right|\right\} \tag{12}$$

3.5 Utility Function

Utility function describes the impact of the number of VE topology changes during task execution on the overall performance. The more the number, the more the impact on the task. The value t_e is VE establishment time that is shown as Fig. 5. In the case where the number of nodes is fixed, t_e can be considered as a constant. To complete computation, the duration time of VE must be larger than the processing time of computation. Therefore, the utility function can be expressed as

$$\left\{t_{dua,ve} - t_{ve,v}\right\} \tag{13}$$

where $2^{|V(t)|}$ is the power set of V(t), which is the set including all subsets of $V(t)$.

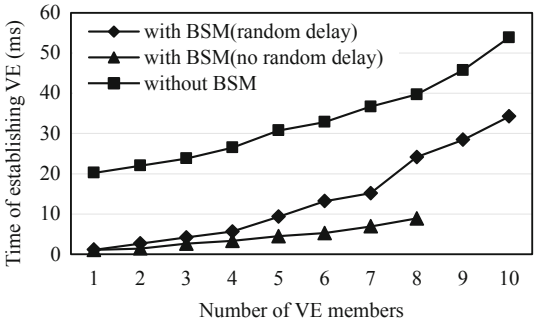

Fig. 5. Time of establishing VE [the method that Basic Safety Message (BSM) without random delay fail to establish VE at 9, 10].

3.6 Complexity

We adopt the utility function above as our objective function, and it is formulated as

$$\left\{t_{dua,ve} - t_{ve,v}\right\}$$

$$\text{s.t. } t_{dua,ve} \geq t_{ve,v}, \forall V_{ve} \in 2^{|V(t)|}$$

$$t_{\langle v,u\rangle} \geq \delta, \forall u \in V_{ve,v}, \forall V_{ve} \in 2^{|V(t)|} \tag{14}$$

The first constraint guarantees that the results can be returned to the vehicle v. The second constraint guarantees that the member of VE did not disconnect frequently.

We assume that $a_{\rightarrow} = \{a_i | a_i \in \{0, 1\}, i = 1, 2, \ldots, |V(t)|\}$ where element a_i refer to the vehicle i is whether the member of VE or not. To find the optimal solution of the problem, it is required to search the entire space and the size of the space is $2^{|V(t)|}$. Due to the complexity of the problem, we proposed a simplified method of establishing VE.

The computational capability is used to indicate the potential computational power that VE can provide.

$$A_{cc} = \sum\nolimits_{i=1}^{n} C_i * T_i \tag{15}$$

where n is times of VE topology changes during the completion of the task. Total computation time T is divided into n stage and T_i is duration time of VE at i th stage. C_i is the computational power of VE at i th stage and it is represented by CPU frequency. The computational power of the VE at each stage is represented by the following formula.

$$C_i = c_i^0 + B * \sum\nolimits_{j=1}^{m} c_i^j \tag{16}$$

where m is the slave node count in VE, B is the ratio of network transmission rate and memory transfer rate, and c_i^j is the computation power of slave node j at stage i. c_i^0 is the computational power of master node at stage i.

$$\begin{aligned} A_{cc} &= \sum\nolimits_{i=1}^{n} \left(c_i^0 + B * \sum\nolimits_{j=1}^{m} c_i^j \right) * T_i \\ &= c_i^0 * T + B * \sum\nolimits_{i=1}^{n} \left(\left(\sum\nolimits_{j=1}^{m} c_i^j \right) * T_i \right) \end{aligned} \tag{17}$$

The problem is to select the best VE members to minimize the objective function. The fewer the number of VE topology changes, the smaller the penalty. When the slave node in VE is disconnected, it needs to reselect the nodes for replacement. The fewer the count to replace nodes, the smaller the communication overhead generated by the migration data. The number of members of VE directly affects the computational capability of the VE. However, a large member node will increase the potential risk of the VE's disconnection.

Because in VANETs, the establishment of VE is affected by many factors such as traffic conditions, signal coverage, maximum speed limit of the road, and road layout. In this paper, the simulator is used to explain the impact of many influencing factors on generating VE in VANETs.

4 Performance Evaluations

The discrete event simulator OMNeT++ and the mobility generator SUMO are used to build realistic vehicular scenarios. We compare our scheme with three benchmarks. Simulation process consists of two parts: establishment of VE and VE topology changes. The parameters in the simulation are listed in Table 1. A 2800 m $\times$ 2800 m area of Tokyo is used as the map in the simulator. Speed limit on the road is 40 km/hour (14 m/s). There are two opposite directions of traffic flows and their arrival rate is 0.2 vehicle/second. the vehicle begins to establish VE when the task is arrived, after the warm-up process. Vehicles are classified into three categories: master vehicle, slave vehicle and ordinary vehicle. A master vehicle is a vehicle with a task and the slave is without task. Master and slave vehicles participate in establishing VE and task processing while ordinary vehicles do not.

Table 1. Parameters in the simulator

Parameter	Value
Max velocity of vehicle	14 m/s
Antenna frequency	5.9 GHz
Interval of sending BSM	1 s
BSM timeout	2 s
V2V data rate	27 Mbps
Memory data rate	600 Mb/s
Communication range	250 m
Warm-up time	200 s
Simulation time	320 s

For the slave node selection, we have developed four methods: random, the minimum distance first, the maximum available resources first, and the longest drive-through time first.

- Random (rand): randomly select vehicles within the single-hop range.
- Minimum distance first (minDist): setting a higher priority for the vehicles with the minimum distance to master vehicle.
- Maximum available resources first (maxRes): setting a higher priority for the nodes with the maximum available resources.
- Longest drive-thru time first (ourProp): The master node is regarded as a fixed point. The relative speed between the master and slave node is obtained by vector product between the speed of the master node and the slave node. The relative speed is used to calculate the drive-thru time through the one-hop range of the master node.

4.1 VE Establishment

As shown in Fig. 6 that the duration time of VE becomes shorter with the increase of the number of VE members as the more the nodes, the shorter the connection link lifetime between nodes. Although, as the number of slave nodes increases, the computation power obtained according to weighted accumulation above method will increase, as shown in Fig. 7, but the duration time will decrease so that the computational capacity will decrease. Therefore, it is important to find the most suitable VE size according to the vehicular environment.

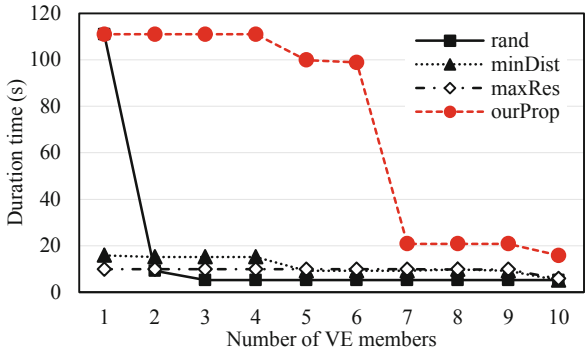

Fig. 6. Duration time of VEs which are consisted of different numbers of members while without VE maintenance.

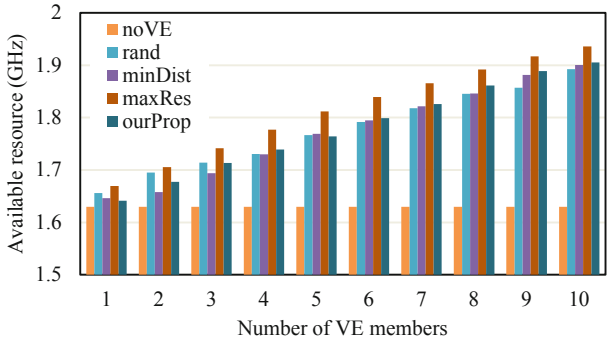

Fig. 7. Computational resources of VEs which are consisted of different numbers of members while without VE maintenance.

4.2 VE Maintenance

Due to the continuity of the cloud service, VE topology change is required in order to satisfy the requested services. The VE will be invalid, when a slave node leaves the communication range of the master node. At this time, the master node automatically maintains the VE, resulting in the change of VE members.

Several metrics are used to compare the performance of schemes in the simulation. In the ideal state, the duration time of the VE equals the service time. However, due to unstable node connection and VE re-establishment time, the duration time of VE is less than the service time, as shown in Fig. 8. Unexpected disconnection is caused by the following reasons: the slave node leaves the communication range of the master node; the failure to receive BSM due to packet losses. The comparison of expected disconnection times for different edge selection methods is shown in Fig. 9.

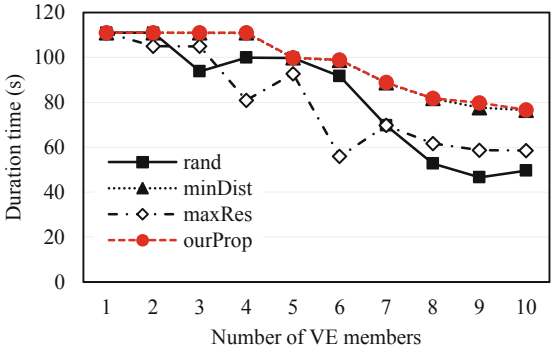

Fig. 8. Duration time of VE which are consisted of different numbers of members.

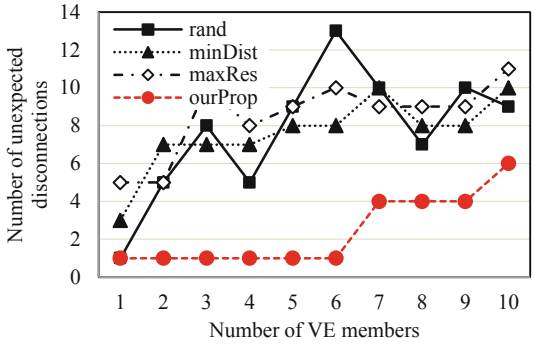

Fig. 9. The number of unexpected disconnections which are consisted of different numbers of members.

Figure 10 and 11 illustrates the number of failures for various number of VE members and the available computational capability, respectively. Available computational capability in the case of VE maintenance is calculated by weighted accumulation. The

potential computational capability of VE is stronger than one single node without VE. VE re-creation failed due to the difficulty in finding the enough slave nodes at vicinity. As the slave nodes count in VE increases, the computational capability of VE can be enhanced, but it also causes some difficulties. For example, the prediction of VE's duration becomes more difficult, and the probability of missing VE member increases, as shown Fig. 9 and Fig. 10. Unexpected disconnection during the task execution is harmful to collaborative task offloading. It is very important to maximize computational capability by increasing the number of the slave nodes while minimizing the number of unexpected disconnections. It can be seen from Fig. 9, VE with 6 slave nodes achieves the largest computational resources with the smallest unexpected disconnections. A similar result can be seen from Fig. 10 where VE with 4 slave nodes achieves the largest computational resources with the smallest number of VE re-creations. Due to the lack of vehicles, VE cannot be established successfully, and transmitting tasks to the cloud for execution will take longer delay.

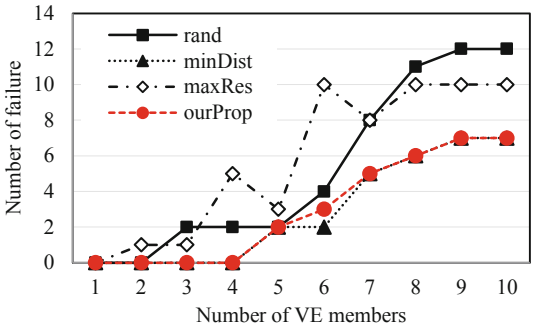

Fig. 10. Number of failures of VE re-formation which are consisted of different numbers of members.

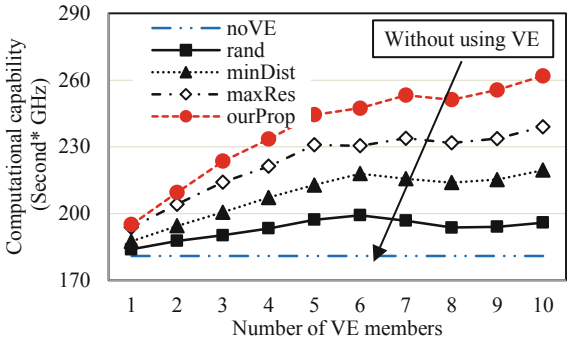

Fig. 11. Obtaining stable services is impracticable in VANET, since the connection is unstable. The product of connection lifetime and computing capability (represented by CPU frequency) is induced to describe the available computational capability of VE.

5 Conclusion

We proposed the concept of virtual edge based on mobile edge computing to solve the shortage of computing capability at the edge of the network. We also proposed a VE selection method where the relative velocity between vehicles is considered. We used computer simulations to evaluate the proposed method by comparing with other baseline approaches. The simulation results show that the proposed method can generate more stable and efficient edge nodes as compared with existing baselines.

Acknowledgment. This research was supported in part by ROIS NII Open Collaborative Research 2020-20S0502, and JSPS KAKENHI grant numbers 18KK0279, 19H04093 and 20H00592.

References

1. Hassan, N., Yau, K.L.A., Wu, C.: Edge computing in 5G: a review. IEEE Access **7**, 127276–127289 (2019)
2. Feng, J., Liu, Z., Wu, C., Ji, Y.: AVE: autonomous vehicular edge computing framework with ACO-based scheduling. IEEE Trans. Veh. Technol. **66**(12), 10660–10675 (2017)
3. Huang, C., Lu, R., Choo, K.R.: Vehicular fog computing: architecture, use case, and security and forensic challenges. IEEE Commun. Mag. **55**(11), 105–111 (2017)
4. You, C., Huang, K., Chae, H., Kim, B.-H.: Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Trans. Wireless Commun. **16**(33), 1397–1411 (2016)
5. Bi, S., Zhang, Y.: Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading. IEEE Trans. Wireless Commun. **17**(6), 4177–4190 (2018)
6. Khalili, S., Simeone, O.: Inter-layer per-mobile optimization of cloud mobile computing: a message-passing approach. Trans. Emerg. Telecommun. Technol. **27**(6), 814–827 (2016)
7. Jiang, Z., Mao, S.: Energy delay tradeoff in cloud offloading for multi-core mobile devices. IEEE Access **3**, 2306–2316 (2015)
8. Wang, Y., Sheng, M., Wang, X., Wang, L., Li, J.: Mobile-edge computing: partial computation offloading using dynamic voltage scaling. IEEE Trans. Commun. **64**(10), 4268–4282 (2016)
9. Mach, P., Becvar, Z.: Mobile edge computing: a survey on architecture and computation offloading. IEEE Commun. Surveys Tutor. **19**(3), 1628–1656 (2017)
10. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Survey Tutor. **19**(4), 2322–2358 (2017)
11. Ning, Z., Huang, J., Wang, X.: Vehicular fog computing: enabling real-time traffic management for smart cities. IEEE Wirel. Commun. **26**(1), 87–93 (2019)
12. Qiao, G., Leng, S., Zhang, K., He, Y.: Collaborative task offloading in vehicular edge multi-access networks. IEEE Commun. Mag. **56**(8), 48–54 (2018)
13. Chen, Xianfu, Zhang, Honggang, Celimuge, Wu, Mao, Shiwen, Ji, Yusheng, Bennis, Mehdi: Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning. IEEE Internet Things J. **6**(3), 4005–4018 (2019)
14. Wu, C., Liu, Z., Zhang, D., Yoshinaga, T., Ji, Y.: Spatial intelligence toward trustworthy vehicular IoT. IEEE Commun. Mag. **56**(10), 22–27 (2018)
15. Whaiduzzaman, M., Sookhak, M., Gani, A., Buyya, R.: A survey on vehicular cloud computing. J. Netw. Comput. Appl. **40**, 325–344 (2014)

16. Olariu, S., Eltoweissy, M., Younis, M.: Towards autonomous vehicular clouds. ICST Trans. Mobile Commun. Appl. **11**(7–9), 1–11 (2011)
17. Olariu, S., Khalil, I., Abuelela, M.: Taking VANET to the clouds. Int. J. Pervasive Comput. Commun. **7**(1), 7–21 (2011)
18. Hou, X., Li, Y., Chen, M., Wu, D., Jin, D., Chen, S.: Vehicular fog computing: a viewpoint of vehicles as the infrastructures. IEEE Trans. Veh. Technol. **65**(6), 3860–3873 (2016)
19. Wang, C., Liang, C., Yu, F.R., Chen, Q., Tang, L.: Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. IEEE Trans. Wireless Commun. **16**(8), 4924–4938 (2017)
20. Kumar, K., Liu, J., Lu, Y.-H., Bhargava, B.: A survey of computation offloading for mobile systems. Mobile Netw. Appl. **18**(1), 129–140 (2013)
21. Huynh, L.N.T., Pham, Q., Pham, X., Nguyen, T.D.T., Hossain, M., Huh, E.: Efficient computation offloading in multi-tier multi-access edge computing systems: a particle swarm optimization approach. Appl. Sci. **10**(1), 203 (2020)
22. Yan, J., Bi, S., Zhang, Y., Tao, M.: Optimal task offloading and resource allocation in mobile-edge computing with inter-user task dependency. IEEE Trans. Wireless Commun. **19**(1), 235–250 (2020)
23. Xu, X., et al.: An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Futur. Gener. Comput. Syst. **96**, 89–100 (2019)
24. Zhang, D., Zhang, T., Liu, X.: Novel self-adaptive routing service algorithm for application of VANET. Appl. Intell. **49**, 1866–1879 (2019)