Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.034994
Article

Tech Science Press

Task Offloading Based on Vehicular Edge Computing for Autonomous Platooning

Sanghyuck Nam[1], Suhwan Kwak[1], Jaehwan Lee[2] and Sangoh Park[1],*

[1]School of Computer Science and Engineering, Chung-Ang University, Dongjak-gu, Seoul, 06974, Korea
[2]Department of Computer Science and Engineering, Kongju National University, Cheonan, 31080, Korea
*Corresponding Author: Sangoh Park. Email: sopark@cau.ac.kr
Received: 03 August 2022; Accepted: 28 October 2022

Abstract: Autonomous platooning technology is regarded as one of the promising technologies for the future and the research is conducted actively. The autonomous platooning task generally requires highly complex computations so it is difficult to process only with the vehicle's processing units. To solve this problem, there are many studies on task offloading technique which transfers complex tasks to their neighboring vehicles or computation nodes. However, the existing task offloading techniques which mainly use learning-based algorithms are difficult to respond to the real-time changing road environment due to their complexity. They are also challenging to process computation tasks within 100 ms which is the time limit for driving safety. In this paper, we propose a novel offloading scheme that can support autonomous platooning tasks being processed within the limit and ensure driving safety. The proposed scheme can handle computation tasks by considering the communication bandwidth, delay, and amount of computation. We also conduct simulations in the highway environment to evaluate the existing scheme and the proposed scheme. The result shows that our proposed scheme improves the utilization of nearby computing nodes, and the offloading tasks can be processed within the time for driving safety.

Keywords: Task offloading; vehicular edge computing; vehicular ad-hoc network; dedicated short-range communication; autonomous platooning

1 Introduction

With the development of information and communication technology, information science technology such as artificial intelligence, the Internet of Things (IoT), and digital twins attract growing attention. In particular, autonomous driving technology receives high attention in various fields such as transportation, logistics, and national defense. An autonomous vehicle performs safe driving without human intervention by recognizing the road and surrounding conditions from various sensors mounted on the vehicle. The global autonomous vehicle market is expected to grow from \$54.2B in 2019 to \$556B in 2026, maintaining an average annual growth rate of 39.47% [1].

One of the ultimate goals for autonomous driving technology and vehicular communication technology is autonomous platooning in which multiple vehicles move in groups. A platoon of vehicles consists of a leader who leads the group and members belonging to the group. The leader communicates with the platoon members and exchanges road conditions, platooning commands, or emergency control such as obstacle avoidance and braking. Vehicle platooning improves the traffic volume on the road by narrowing the distance between vehicles. It also reduces the air resistance of each vehicle, thereby reducing fuel consumption and CO2 emission [2,3]. In addition, since the platoon leader transmits control commands to members simultaneously, it is possible to perform actions faster and safer than human control. Therefore, it is possible to provide a safer road environment by reducing traffic congestion and accident rate through vehicle platooning [4].

Applications related to vehicle platooning are generally computationally intensive, which makes real-time control difficult only with vehicles with low computational power. Performance improvement and energy reduction can be achieved in mobile cloud computing by offloading some computation-intensive tasks to the cloud. However, cloud servers are far from mobile devices resulting in high communication latency and difficulty in processing real-time computational tasks. In fog computing environment, the offloaded tasks are processed in computing nodes near the node that initiated the task offloading [5]. Fog computing is suitable for environments that require computationally intensive and real-time data processing near edge nodes. Existing studies have focused on how to determine which node to offload in a fog computing environment [6–10]. In the studies, the offloading node decision problem was viewed as sequential decision-making considering mobility, communication, and computing environment, and the Markov model or reinforcement learning was employed to solve the problem. These existing studies are not suitable for real-time platoon control applications because they require iterative computations until the convergence to an optimal offloading solution.

Recently, studies are being conducted to provide stable vehicular wireless communication [11–14]. The complexity of the offloading problem can be reduced if stable communication is ensured in task offloading. Therefore, it is necessary to design a task offloading scheme based on the communication techniques of such a vehicle environment. The vehicular safety supporting system (V3S) addressed the bandwidth and latency limitations by proposing a clustering scheme based on the time division multiple access (TDMA) media access control (MAC) protocol. The dynamic channel allocation for each vehicle cluster contributed to the stable and efficient communication of vehicles. In this paper, we propose a task offloading scheme to satisfy the time limit for vehicle safety [15,16] based on V3S. To this end, we model the task offloading problem in the V3S environment to determine which edge computing node to offload the task to, taking into account the amount of the computation, the communication bandwidth, and the delay. The proposed task offloading scheme based on vehicular edge computing utilizes a roadside unit (RSU) as an edge computing node to support autonomous platooning tasks. RSUs can support platooning tasks with low latency at the location closest to the vehicle.

The structure of this paper is as follows. Section 2 introduces and discusses the vehicular communication environments and the studies on task offloading schemes. Section 3 proposes a novel edge computing architecture and a task offloading scheme to support autonomous platooning based on an efficient vehicular communication environment. Section 4 shows the results of the performance evaluation and analysis to validate that the task offloading can be done within a 100 ms safety time limit. Finally, Section 5 describes our conclusions and future works.

2 Related Work

2.1 Task Offloading in Vehicular Environment

Autonomous platooning requires computation-intensive tasks to be processed with low latency for safe driving, whereas it is difficult to satisfy the latency requirement due to the limitation of computational resources mounted on a vehicle. Mobile cloud computing provides computational resources through servers located remotely, which shows a large transmission delay owing to its geological characteristics. Mobile edge computing can reduce the offloading latency by placing computational resources close to the edge node. However, mobile edge computing did not consider devices with high mobility such as vehicles. Vehicular edge computing paradigm considers vehicular mobility and communications. In this computing paradigm, the RSU acts as an edge server to collect data and provide computing power near the vehicles. In addition, vehicular fog computing architecture is being actively studied to maximize computing power by utilizing spare computing resources between vehicles.

Vehicular edge computing provides an efficient way to serve computationally intensive applications with low latency [17,18]. A study on searching for an efficient edge server selection and task delivery scheme proposed a prediction-based delivery considering the task execution time and vehicle mobility [19]. In the proposed scheme, tasks are transmitted directly to a single-hop edge server or relayed to a multi-hop edge server in consideration of vehicle mobility. A study on cooperative vehicular edge computing framework proposed a hierarchical structure of cloud computing and edge computing to offer improved scalability [20]. Another study focused on addressing insufficient vehicular communication bandwidth for task offloading in vehicular edge computing [21]. The authors proposed to optimize task offloading and wireless frequency allocation simultaneously. In a study to reduce task execution time and energy consumption, the authors proposed an approximation of offloading decisions based on the alternating direction method of multipliers (ADMM) [22] and Lyapunov optimization [23].

Vehicular fog computing cooperatively utilizes the computing resource of edge servers and surrounding vehicles to handle computation-intensive tasks [24]. The allocation problem is more complex than the vehicular edge computing case. This is because both the computation capability and the high mobility constraints must be considered. In previous studies, reinforcement learning was largely adopted to learn the task offloading decision such as multi-armed bandit (MAB) [7] or Markov decision process [8–10]. In these studies based on reinforcement learning, in general, offloading nodes are selected according to the surrounding environment. An offloading policy is learned according to the reward value such as offloading execution time.

2.2 *Vehicular Safety Supporting System (V3S)*

Deidicated short range communication (DSRC) is a communication standard for vehicles to provide intelligent transportation system (ITS) services [25]. DSRC uses a 5.9 GHz frequency band [26] it consists of 1 control channel (CCH) and 6 service channels (SCH) as shown in Fig. 1. The CCH transmits and receives high-priority control and management messages, and the SCH is designed to transmit and receive additional messages for vehicular applications. The channel switching between CCH and SCH is periodically performed to support the exchange of control and service messages.

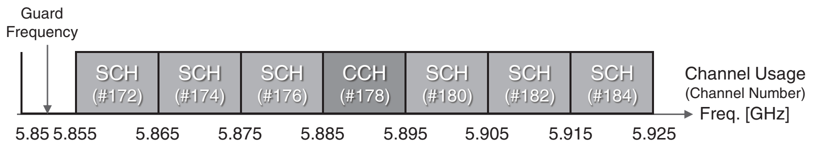

Figure 1: DSRC channels

When performing wireless communication in DSRC, channel switching between CCH and SCH is periodically performed to support the transmission and reception of each message by type. In the standard, the default value of the channel switching period of DSRC in which the CCH period and the SCH period are performed once is defined as 100 ms. Safety-related applications, including basic safety message (BSM), which is vehicle information for a safe traffic environment, should be transmitted within 100 ms to ensure safety for vehicles on the road [16]. DSRC cannot guarantee channel access latency because packet collision increases exponentially as the number of nodes increases. Existing studies have improved the TDMA MAC protocol, clustering, multi-channel utilization, etc. to reduce the instability and packet collision rate of vehicular communication. It showed an improved packet delivery rate and reduced packet collision compared to the 802.11p. However, to support safety applications, research is needed to consider ensuring packet delivery, and characteristics of road and vehicle communication environments. Although studies have been conducted [12,13,27–30] to reduce the collision rate of packets and efficiently utilize bandwidth by utilizing TDMA and clustering, these studies do not maximize bandwidth utilization because they do not efficiently utilize the multi-channel DSRC and/or do not consider road bi-directionality. In a recent previous study, V3S was proposed [14] for safe vehicle communication. V3S solves the bandwidth problem of DSRC by combining cluster, TDMA, and channel allocation.

In V3S-based autonomous platooning, vehicles and infrastructure transmit and receive information using 5.9 GHz DSRC communication. The vehicle cluster uses SCH of the DSRC to share information between cluster members and the cluster leader and communicates between the cluster leader and RSUs through CCH. In addition, in order to prevent interference within 1 km of the DSRC range, each cluster is arranged at an interval of 200 m, and 6 SCHs are sequentially allocated. The SCH is divided into an Up period and a Down period. The cluster member broadcasts the BSM during every SCH Up period, and the cluster leader collects the BSM received from the cluster members to construct an aggregated BSM list. Thereafter, the cluster leader propagates the aggregated BSM list and the platoon operation request to the RSU in the CCH period. A cluster member is allocated a time slot capable of transmitting data in SCH Up every communication period from the cluster header. Therefore, the vehicles in each cluster can transmit packets without collision in their time slots. As described above, V3S showed that stable packet transmission is possible even large number of vehicles are on road through dynamic channel allocation according to clusters and TDMA-based slot allocation. V3S can eliminate the variable of task offloading problems and lower the complexity in the vehicular environment based on stable communication.

2.3 Motivation and Contributions

The existing vehicular edge computing or vehicular fog computing study exhibited high computational complexity of offloading decision-making problems because of limited available communication and computation resources, and the high mobility of the vehicle must all be considered. Therefore, most studies proposed an approximation or prediction based on reinforcement learning to deal with the problem. However, these studies are not suitable for autonomous platooning that requires low latency for task execution because the computation of offloading decision-making algorithms requires sufficient time to converge to a (locally or globally) optimal solution.

Recently, studies on reliable vehicular wireless communication capable of reducing the complexity of task offloading are being conducted such as V3S. However, it is difficult to directly apply the existing task offloading methods because unique characteristics of the networking model of V3S, such as the channel switching period are not considered.

Therefore, in this paper, we propose a task offloading scheme for supporting autonomous platooning based on V3S with considering the characteristics of the underlying network model and autonomous platooning workload. The proposed task offloading method employs a computationally simpler decision-making algorithm for task offloading to efficiently handle task offloading for supporting autonomous platooning.

3 Fog Computing-Based Task Offloading

The proposed vehicular edge computing system in the V3S environment is shown in Fig. 2. The system consists of multiple platooning vehicles and RSUs, and each RSU embeds Edge Servers for serving task offloading requests. RSUs are placed along the road at intervals of 1 km, considering the DSRC communication radius of 500 m. The vehicle platoon shares information between the platoon leader and the member through SCH, while the platoon leader communicates with RSU through CCH.

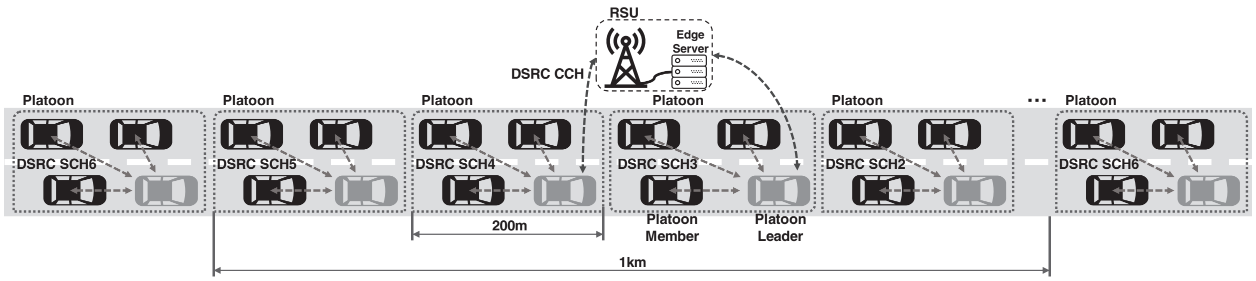

Figure 2: Communications between nodes

In this paper, we divide both the CCH and the SCH period into sub-periods named Up and Down as shown in Fig. 3. The purpose is to give the overall offloading procedure a sufficient time. The DSRC channel switching consists of SCH Up/Down period of 23 ms each, and CCH Up/Down period of 23 ms each, followed by a guard period of 4 ms. In order for the RSU to secure task processing time within 100 ms, it is used alternately in the order of CCH → SCH and the CCH Down period precedes over CCH Up. Therefore, for both directions A and B in Fig. 3, channel switching is performed in the order of CCH Down→ CCH Up → SCH Up → SCH Down. The platoon leader manages its member vehicles and offloads its platooning-related tasks to the RSU edge nodes. When the RSU has finished the computation offloading request, it responds the result to the platoon leader in the next CCH Down period. The platoon leader that received the offloading result propagates to its members in the next SCH Down period. The RSU receives the aggregated BSMs and offloading requests from platoon leaders within its communication range. The RSUs can perform distributed processing of a computation offloading request if it cannot process within 100 ms on its own. The symbols used in the equation for modeling task offloading are defined in Table 1.

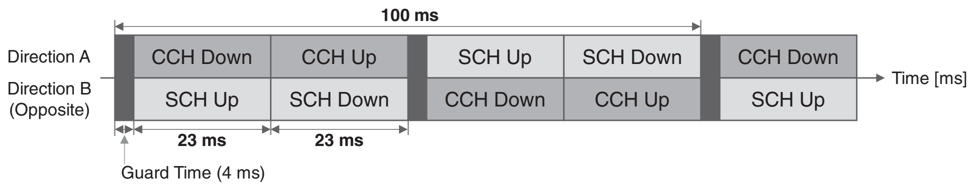

Figure 3: DSRC channel switching model

Table 1: Definition of parameters related to task offloading method

Symbol	Definition
Γ	The entire set of RSUs on the road
r	RSUs belonging to set Γ
n	Minimum number of RSUs that enables platoon offloading tasks to be performed within 100 ms
w_{req}	The amount of computation of a task a platoon leader requested
$w_{p,r}$	The net amount of computation that r can perform in 100 ms
$w_{c,r}$	The computation speed of r
$w_{q,r}$	The amount of computation remaining in r's task queue

(Continued)

Table 1 (continued)

Symbol	Definition
d_{req}	The size of a task a platoon leader requested
$d_{d,r}$	The size of data for the task received from r
b_w	Wire communication bandwidth
b_e	Ethernet communication bandwidth
ε	The ratio of offloading request data size and result data size
$t_{o,r}$	Time taken to deliver distributed offloading task to r
$t_{q,r}$	Time spent executing remaining tasks in the task queue at r
$t_{g,r}$	Time taken to return the result of platoon offloading tasks

When an RSU receives a task offloading request from a platoon, it determines whether distributed processing of multiple RSUs is required. In order to ensure stable and safe platooning, the offloading time must not exceed 100 ms. As shown in Fig. 4, this includes ①; the time to transmit the request to RSU, ② the time to process the request in RSU(s) and ③ the time to receive the result from RSU.

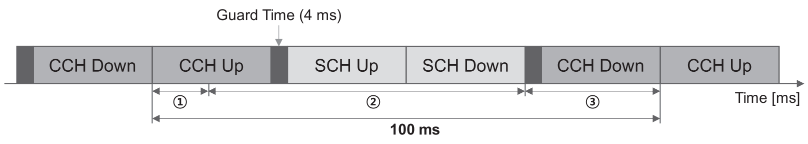

Figure 4: The offloading time

Algorithm 1 Calculate the Number of RSUs for Supporting the Offloaded Task

```
1:  procedure getOffloadingRSUs
2:      $n \leftarrow 1$
3:      if (1) is TRUE then
4:          return $n$
5:      end if
6:      $n \leftarrow n + 1$
7:      $w_{p,r} \leftarrow []$
8:      for each $r \in R \subset \Gamma$ do
9:          Obtain $w_{p,r}$ by (2)
10:     end for
11:     if (6) is TRUE then
12:         $w_{p,r} \leftarrow []$
13:         goto line 4
14:     end if
15:     return $n$
16: end procedure
```

Algorithm 1 is proposed in order for an RSU to determine how much distributed processing is required for the offloaded task received from the platoon leader. The RSU determines if the received request can be processed on its own as shown in line 2, and if not, it increases *n* the number of RSUs processing the task and determines whether such *n* can process the task within the 100 ms limit as in llines 8–14. The *n* is returned if the offloading can be performed within time using *n* RSUs. Through this process, it is possible to provide real-time task offloading for safe autonomous platooning.

$$\frac{w_{req}}{w_{c,r}} \leq 0.073 - \frac{d_{req}}{b_w} - \frac{w_{q,r}}{w_{c,r}} \tag{1}$$

The left side of Eq. (1) represents the time it takes for RSU $r \in \Gamma$ to execute the offloading task alone, and the right side represents the net computation time the *r* can allocate for that task only. The right side, in other words, is the maximum processing time that *r* can allocate minus the task transmission time and minus the processing time to finish the tasks in the tasks queue of *r*. The channel switching model of V3S should be considered because CCH in V3S uses carrier sense multiple access with collision avoidance (CSMA/CA). Therefore, the RSU should be prepared to respond with the offloading result before ③ in Fig. 4 and the maximum computation time of an RSU is from the start of CCH Up to the end of SCH Down, which is as long as 73 ms. The ① in Fig. 4 is determined when a platoon leader requests a task offloading of size d_{req} with the transmission speed of b_w. Excluding the time required for the remaining task to finish in the queue, the net computation time for the offloading task can be obtained. If Eq. (1) is satisfied, the offloading can be performed within the time limit by *r* alone. If Eq. (1) is not satisfied, the task must be divided among the neighboring RSUs.

As shown in line 9, the amount of computation $w_{p,r}$ that can be individually performed for *n* RSUs is calculated. The amount of computation that each RSU can perform within the time is shown in Eq. (2). It is obtained by multiplying the maximum computation time minus offloading time, queue processing time, and result transmission time by the computation speed $w_{c,r}$.

$$w_{p,r} = \left(0.073 - \frac{d_{req}}{b_w} - t_{o,r} - t_{q,r} - t_{g,r}\right) \times w_{c,r} \tag{2}$$

$t_{o,r}$ is the time it takes to transfer the offloading task to RSU *r*, as shown in Eq. (3).

$$t_{o,r} = \frac{d_{d,r}}{b_e} \tag{3}$$

If an RSU is already executing other tasks, it queues the later received task and executes in order. Therefore, the tasks in the RSU's task queue should be taken into account. $w_{q,r}$ represents the amount of computations remaining in the task queue of *r* and dividing it by the processing speed $w_{c,r}$, the remaining time to finish the tasks is obtained. In this case, even if there is a remaining task, the computation may be completed while the offloading data is transmitted. In addition, $w_{q,r}$ was received at least the previous CCH Down period, and the task in the task queue may have already been completed, which should also be considered. The $t_{q,r}$ is shown as Eq. (4). If the remaining task processing time is greater than the transmission time, the execution time of the task which has just arrived may increase because the remaining task must be executed first.

$$t_{q,r} = \max\left(0,\ \frac{w_{q,r}}{w_{c,r}} - \left(0.023 + \frac{d_{req}}{d_w}\right) - \frac{d_{d,r}}{b_e}\right) \tag{4}$$

$t_{g,r}$ is the time takes for *r* to transmit the result of executing the divided offloading task as shown in Eq. (5). The time to return the task execution result is obtained by dividing the size of the result by the network transmission bandwidth. The size of the result data may vary depending on the offloading request. In this paper, we assume that the size of the resulting data is linearly proportional to the size of the task request to respond to various platooning applications. Therefore, the proportional coefficient ε is added.

$$t_{g,r} = \frac{d_{req} \times \varepsilon}{b_e} \tag{5}$$

As shown in line 11, it is determined whether the amount of computation of n RSUs can safely perform the task offloading within 100 ms based on Eq. (6). If the sum Eq. (2) of the RSUs' computation capability falls behind the required amount l, then the n RSUs cannot process the offloading task in 100 ms. Lines 12–13 search for the minimum \$n\$ to be able to process the offloading task.

$$w_{req} \leq \sum_{r=1}^{n} w_{p,r} \tag{6}$$

4 Performance Evaluation

To evaluate the performance of the distributed task offloading method proposed in this paper, a simulation of vehicle communication in a virtual road environment was implemented based on the Vehicle in Network Simulation (Veins) framework [31]. The Veins framework is used to simulate vehicle networks in road environments and wireless network environments and is based on Simulation of Urban Mobility (SUMO) [32], a road traffic simulator, and OMNet++ [33], an event-based network simulator. The standard 802.11p was used as the vehicle to everything (V2X) basic communication protocol in simulation, and the highway environment was used.

In the vicinity of a 5-lane highway, RSUs are installed at fixed positions at intervals of 1 km considering the DSRC communication radius of 500 m. Also, to show the distributed processing between RSUs, three RSUs were placed at 500, 1500, and 2500 m points over 3 km and connected via Ethernet, and vehicles travel this highway from left to right. As the proposed method supports simultaneous communication on two-way roads, when the total number of vehicles on a one-way highway is 40, the offloading success rate is theoretically the same as when the total number of vehicles on a two-way highway is 80. In addition, the vehicle's speed does not affect the offloading throughput, so it travels at a constant speed of 30 m/s. Each vehicle also broadcasts a BSM and offloading request every 100 ms, a minimum interval to ensure vehicle safety on the road.

The parameters for the simulation are shown in Table 2. For performance comparison, the existing method of broadcasting the offloading request and BSM is denoted as offloading-NOC, Time-Aware Markov decision process (TMDP) [10] is denoted as offloading-TMDP and the proposed offloading method is denoted as offloading-V3S.

Table 2: Environmental Parameters for performance evaluation

Parameters	Value
Length of the road (m)	3,000
Number of vehicles	3, 5, 10 ~ 100
Velocity of the vehicles (m/s)	30
Length of each vehicle (m)	4
b_{eth} $(Gbps)$	10
n	3
d_{req} (KB)	1

The offloading throughput according to the change in the number of vehicles on the road is shown in Fig. 5. This offloading throughput is the result of counting processed tasks within 100 ms of all offloaded tasks which successfully received by requested vehicle. The x-axis is the total number of vehicles generated on the road. As the number of vehicles increased, the offloading throughput of offloading-NOC decreased rapidly. This is because offloading-NOC increases the offloading execution time and decreases the number of transmitted and received offloading packets due to the rapid increase in packet collisions and rapid network saturation when only broadcasting is used. From (a) to (f) of Fig. 5, as network bandwidth increases, the data transmission time decreases, and it can be seen that the offloading success rate is improved. And offloading-TDMP also shows similar result but better than offloading-NOC. This is because offloading-TDMP considers vehicle mobility to select offload destination, and probability of receiving processed task from RSU is increased. In the case of offloading-V3S, which implements the proposed V3S-based offloading method, it showed a high success rate in all bandwidths. Through this, it shows that task offloading can be performed stably when the proposed method is applied.

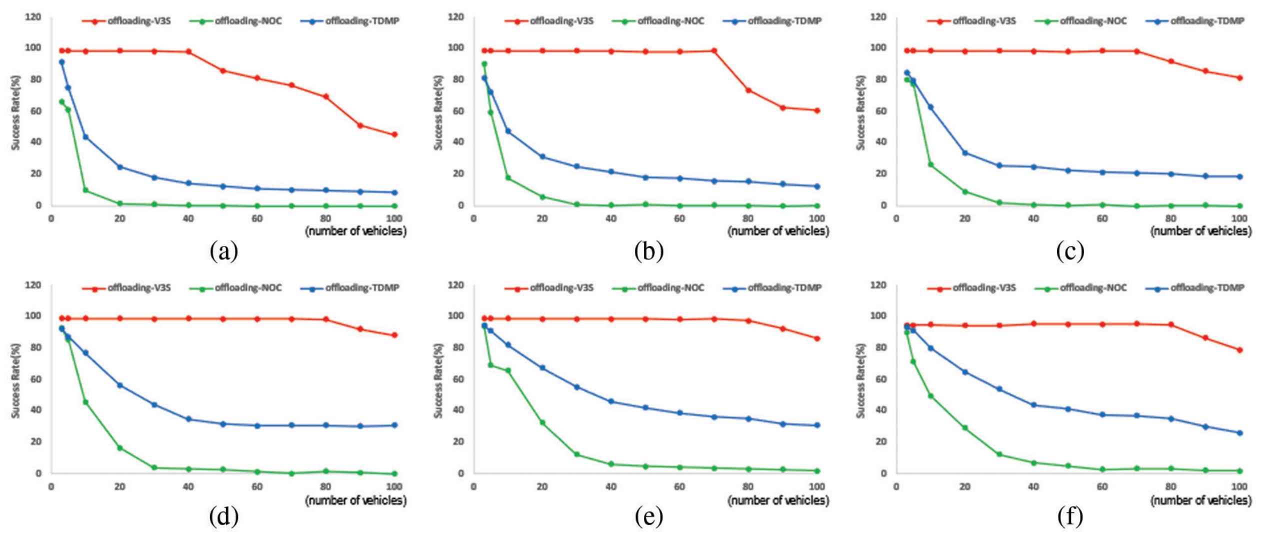

Figure 5: Offloading throughput according to the change in the number of vehicles on the road. (a) 6 Mbps, (b) 9 Mbps, (c) 12 Mbps, (d) 18 Mbps, (e) 24 Mbps, (f) 27 Mbps

The average offloading execution time according to the change in the number of vehicles on the road is shown in Fig. 6. This is the average value of offloading execution time received from each vehicle. In the case of offloading-NOC and offloading-TDMP, it can be seen that the average offloading execution time fluctuates while vehicles increases. As both method rely underlaying DSRC to communicate, latency can be jittery as DSRC uses contention based MAC. On the other hand, in the case of the proposed offloading-V3S, offloading was performed consistently around 80 ms regardless of bandwidth and number of vehicles. This is because the offloading request is transmitted to the RSU in the CCH Up period, the calculation is performed, and the result is returned back to the vehicle in the CCH Down period. That is, in the proposed method, the execution time of a successful offloading request is higher than that of the existing method, but more offloading requests can be stably processed within 100 ms.

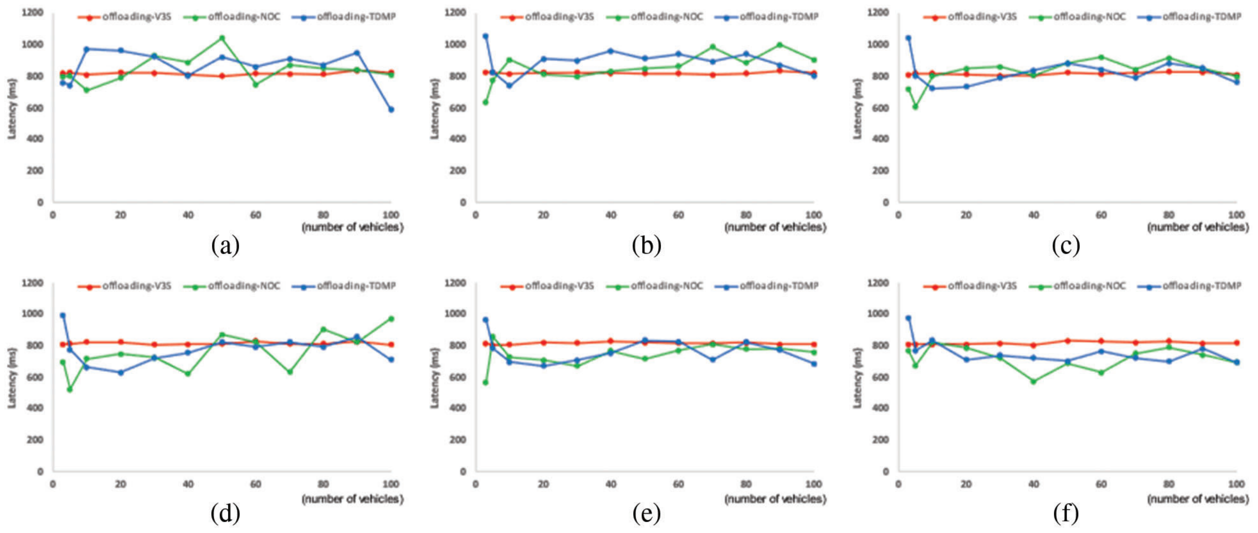

Figure 6: Offloading execution time according to the change in the number of vehicles on the road. (a) 6 Mbps, (b) 9 Mbps, (c) 12 Mbps, (d) 18 Mbps, (e) 24 Mbps, (f) 27 Mbps

5 Conclusion

With the development of autonomous driving technology, the potential of autonomous platooning is increasing. In order to support safe autonomous platooning, it is necessary to provide a sufficient amount of computational power, but it is difficult to be satisfied with the computational power of the vehicle itself. To solve this problem, task offloading methods using vehicular communications are being studied. However, existing methods for solving task offloading suggests an approximation method or a reinforcement learning-based prediction method which has a long error convergence time, making it difficult to support platooning large number of vehicles in real-time changing road environments.

In this paper, to support safe autonomous platooning, the task offloading problem was modeled based on V3S communication, and a task offloading method that satisfies 100 ms latency limit was proposed which can reduce the task offloading problem complexity and make faster offloading decisions. The simulation results of the highway scenario showed that the proposed task offloading method can stably process a larger amount of computational tasks within 100 ms compared to the existing method. However, the proposed method has a limitation in that it has to search all surrounding nodes until offloading can be performed within the time limit. Therefore, in future research, it is necessary to study a method for more efficiently searching for computing nodes in order to improve the performance of the task offloading method. In addition, research is needed to ensure safe platooning tasks by using heterogeneous communication in a saturated DSRC network.

Funding Statement: This work was supported in part by the Chung-Ang University Research Scholarship Grants in 2021, and in part by R&D Program for Forest Science Technology (Project No. “2021338B10-2223-CD02) provided by Korea Forest Service (Korea Forestry Promotion Institute).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Jadhav, *Global Autonomous Vehicle Market by Level of Automation, Component, and Application: Global Opportunity Analysis and Industry Forecast*, Dublin, Dublin, Ireland: Research and Markets, 2018. [Online]. Available: https://www.researchandmarkets.com/research/lpzwz6.

[2] J. Axelsson, T. Bergh, A. Johansson, B. Mårdberg, P. Svenson *et al.,* "Truck platooning business case analysis," 2020.

[3] A. Winder, *ITS4CV–ITS for Commercial Vehicles*, Brussels, Belgium: ERTICO, 2016. [Online]. Available: http://erticonetwork.com/wp-content/uploads/2016/09/ITS4CV-Report-final-2016-09-09.pdf.

[4] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund and S. Tsugawa, "Overview of platooning systems," in *Proc. of the 19th ITS World Congress*, Vienna, Austria, 2012.

[5] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. of the 1st ed. of the MCC Workshop on Mobile Cloud Computing*, Helsinki, Finland, pp. 13–16, 2012.

[6] A. M. A. Hamdi, F. K. Hussain and O. K. Hussain, "Task offloading in vehicular fog computing: State-of-the-art and open issues," *Future Generation Computer Systems*, vol. 133, pp. 201–212, 2022.

[7] Y. Sun, X. Guo, J. Song, S. Zhou, Z. Jiang *et al.,* "Adaptive learning-based task offloading for vehicular edge computing systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3061–3074, 2019.

[8] Y. Wang, K. Wang, H. Huang, T. Miyazaki and S. Guo, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 976–986, 2018.

[9] Q. Wu, H. Liu, R. Wang, P. Fan, Q. Fan *et al.,* "Delay-sensitive task offloading in the 802.11p-based vehicular fog computing systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 773–785, 2019.

[10] X. Zhang, J. Zhang, Z. Liu, Q. Cui, X. Tao *et al.,* "MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3296–3309, 2020.

[11] M. Klapez, C. A. Grazia and M. Casoni, "Minimization of ieee 802.11p packet collision interference through transmission time shifting," *Journal of Sensor and Actuator Networks*, vol. 9, no. 2, 2020. https://doi.org/10.3390/jsan9020017.

[12] X. Duan, Y. Zhao, K. Zheng, D. Tian, J. Zhou *et al.,* "Cooperative channel assignment for VANETs based on dual reinforcement learning," *Computers, Materials & Continua*, vol. 66, no. 2, pp. 2127–2140, 2021.

[13] M. Ahmad, A. Hameed, F. Ullah, I. Wahid, A. Khan *et al.,* "Adaptation of vehicular ad hoc network clustering protocol for smart transportation," *Computers, Materials & Continua*, vol. 67, no. 2, pp. 1353–1368, 2021.

[14] J. Lee, S. Kwak and S. Park, "Cluster-based stable bsm dissemination system for safe autonomous platooning," *Computers, Materials and Continua*, vol. 71, no. 1, pp. 321–338, 2022.

[15] DSRC Committee, "Dedicated Short Range Communications (DSRC) Message Set Dictionary," in *SAE Standard J2735_200911*, Warrendale, PA, USA: SAE International, 2009. [Online]. Available: https://www.sae.org/standards/content/j2735_200911/.

[16] A. L. Svenson, G. Peredo and L. Delgrossi, "Development of a basic safety message for tractor-trailers for vehicle-to-vehicle communications," in *24th Int. Conf. on the Enhanced Safety of Vehicles (ESV)*, Gothenburg, Sweden, 2015.

[17] G. Qiao, S. Leng, K. Zhang and Y. He, "Collaborative task offloading in vehicular edge multi-access networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 48–54, 2018.

[18] X. Liu, S. Xu, C. Yang, Z. Wang, H. Zhang *et al.,* "Deep reinforcement learning empowered edge collaborative caching scheme for internet of vehicles," *Computer Systems Science and Engineering*, vol. 42, no. 1, pp. 271–287, 2022.

[19] K. Zhang, Y. Mao, S. Leng, Y. He and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 36–44, 2017.

[20] K. Wang, H. Yin, W. Quan and G. Min, "Enabling collaborative edge computing for software defined vehicular networks," *IEEE Network*, vol. 32, no. 5, pp. 112–117, 2018.

[21] J. Du, F. R. Yu, X. Chu, J. Feng and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1079–1092, 2018.

[22] Z. Zhou, J. Feng, Z. Chang and X. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5087–5099, 2019.

[23] T. Cui, Y. Hu, B. Shen and Q. Chen, "Task offloading based on lyapunov optimization for mec-assisted vehicular platooning networks," *Sensors*, vol. 19, no. 22, pp. 4974, 2019.

[24] Y. Zhang, C. -Y. Wang and H. -Y. Wei, "Parking reservation auction for parked vehicle assistance in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3126–3139, 2019.

[25] Y. J. Li, "An overview of the DSRC/WAVE technology," in *Int. Conf. on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, Huston, TX, USA, pp. 544–558, 2010.

[26] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[27] M. S. Almalag, S. Olariu and M. C. Weigle, "Tdma cluster-based mac for vanets (tc-mac)," in *2012 IEEE Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, San Francisco, CA, USA, pp. 1–6. 2012.

[28] R. Zhang, X. Cheng, L. Yang, X. Shen and B. Jiao, "A novel centralized TDMA-based scheduling protocol for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, 1, pp. 411–416, 2014.

[29] C. Tripti and R. Manoj, "An asynchronous multi-channel MAC for improving channel utilization in VANET," *Procedia Computer Science*, vol. 115, pp. 607–617, 2017.

[30] N. Shahin and Y. -T. Kim, "Scalable TDMA cluster-based MAC (STCM) for multichannel vehicular networks," in *2017 19th Asia-Pacific Network Operations and Management Symp. (APNOMS)*, Seoul, Korea, pp. 13–18, 2017.

[31] D. Krajzewicz, J. Erdmann, M. Behrisch and L. Bieker, "Recent development and applications of SUMO-simulation of urban MObility," *International Journal on Advances in Systems and Measurements*, vol. 5, pp. 128–138, 2012.

[32] M. Behrisch, L. Bieker, J. Erdmann and D. Krajzewicz, "SUMO–simulation of urban MObility: An overview," in *Proc. of the Third Int. Conf. on Advances in System Simulation (SIMUL)*, Barcelona, Spain, pp. 55–60, 2011.

[33] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. of the 1st Int. Conf. on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Marseille, France, pp. 1–10, 2008.