Efficient Task Offloading for MEC-Enabled Vehicular Networks: A Non-Cooperative Game Theoretic Approach

Md Delowar Hossain, Subina Khanal and Eui-Nam Huh
Department of Computer Science and Engineering, Kyung Hee University
Yongin-si 17104, South Korea
Email:{delowar, subinakhanal, johnhuh}@khu.ac.kr

***Abstract*—Vehicular Edge Computing (VEC) is a new leading technology to enhance the vehicular performance through task offloading where resource-confined vehicles offload their computing task to the vehicular multi-access edge computing (MEC) networks in proximity. However, the environment of vehicular task offloading is extremely dynamic and faces some challenges to determine the location of processing the offloaded task. As a result, to achieve optimal performance by using traditional VEC system is difficult because in advance we don't know the demand of vehicles. Therefore, a non-cooperative game theory-based efficient task offloading (NGTO) scheme is proposed in this study where the offloading decisions are taken either the MEC server or remote cloud server through the game-theoretic approach. To reduce the processing latency of the vehicles' computation tasks and assure the maximum utility of each vehicle, we used a distributed best response offloading strategy. Our proposed strategy accommodates its offloading probability to achieve a unique equilibrium under certain conditions. Detailed performance evaluation affirms that our proposed NGTO scheme can outperform in all scenarios. It can minimize the response time at almost 41.2% and average task failure rate at approximately 56.3% when compared with a local roadside unit computing (LRC) scheme. The reduced response time and task failure rates are approximately 25.2% and 20.4%, respectively, when compared with a collaborative (LRC with cloud via roadside unit) offloading scheme.**

***Keywords*—vehicular edge computing, game theory, task offloading, vehicular networks.**

I. Introduction

With the evolution of intelligent vehicles and the emergence of diverse high demanding vehicular applications such as advanced driver assistance, accident warning, auto navigation, natural language processing, and autonomous driving, are employed to assist both passengers and drivers in a vehicular environment [1]- [4]. For processing those delay constraints computation-intensive applications usually need a lot of extensive computing resources. However, due to the resource-constrained vehicles, it is hard to meet the computation requirements to provide the quality-of-service (QoS) for the above-mentioned applications. Therefore, the evolution of vehicular networks is crying need to overcome these challenges [5]. At present, VEC is considered as a promising solution [5], [6]. By deploying lightweight but ubiquitous multi-access edge computing (MEC) servers on nearby roadside units (RSUs), excessive computation capabilities are provided for resource-confined vehicles. To minimize the processing delay, moving vehicles can easily offload their latency-sensitive computing tasks to the nearest MEC server via RSU. Moreover, to overcome the limitation of resource-constrained vehicles in executing computation-intensive applications, vehicle can offload their computing tasks to the high-powered capability remote cloud server via RSU [7].

Recently, the task offloading in VEC networks has gained widespread attention from many researchers [1], [8]- [13]. To utilize the idle computational resources in dynamic vehicular environments, an autonomous vehicular edge (AVE) framework is introduced for task offloading [1]. The main advantage of this framework is that without centralized control, it can manage idle nearby vehicular resources to enhance the computing capabilities of vehicles. In MEC-enabled vehicular networks, Wang et al. [8] proposed the federated offloading of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication to improve the utilization of computing resources and minimize the total latency. For finding the optimal server to process the offloaded tasks, Hossain et al. [9] proposed a fuzzy decision based FTOM scheme. For offloading delay-sensitive tasks, they preferred local or nearby MEC servers and remote cloud server for resource-demanding delay-tolerant tasks. To utilize the vehicle computing resource and the VEC server, Zhang et al. [10] used a Stackelberg game-theoretic approach for incentive-mechanism based task offloading. Moreover, to share the computing resources, a backup edge server was used in the neighborhood. In a multi-user VEC network, Dai et al. [11] developed a low-complexity JSCO algorithm and proposed an integrated load balancing with an offloading scheme to maximize the system utility. To utilize the idle edge resource and minimize the redundant data, Nguyen et al. [12] proposed a cost-aware collaborative task computing scheme in MEC-enabled internet of vehicles (IoV) applications. Moreover, to enhance the vehicular performance, Zhang et al. [13] introduced MEC-enabled fiber-wireless (FiWi) networks. To achieve efficient task offloading, authors proposed software-defined networking (SDN) based load balancing approach.

Although, by using the MEC server to extend the computation capacity of vehicular networks, it faces significant challenges because of its confined resources capability. With-

out any collaboration, a distinct MEC server cannot handle a huge number of vehicle offloading requests. Therefore, it degrades the quality of vehicular performance. However, existing research mostly focused on task offloading from the vehicle-to-vehicle, vehicle to the nearest MEC server via RSU, vehicle to the cloud server, and very few research work consider the vehicle with the edge server collaboration. Moreover, they considered static vehicle position or constant speed movement for designing their vehicular mobility model while ignored the enormous computing remote cloud resources [4]. For filling the gap, by utilizing the nearby MEC servers and powerful remote cloud computing resource, we design a game theory-based cloud assisted MEC-enabled vehicular networks which consider the task deadline constraint and a real-life variable vehicle speed movement. To accomplish maximum utility for each vehicle, our proposed scheme accommodates its offloading probability to achieve a unique equilibrium. The contribution of this study are summarized as follows:

- We investigate an efficient task offloading framework for ensuring the QoS within the task deadline constraint in the MEC-enabled vehicular network.
- We used vehicle to roadside unit (V2R) communication mode for local RSU computing. Moreover, the task is offloaded to the cloud via RSU to provide remote cloud server supports.
- To assure the maximum utility of each vehicle, the NGTO approach is proposed in multi-user vehicular networks. Moreover, we adopt the game-theoretic based distributed best response offloading strategy for making the decision of task offloading.
- Finally, the experimental results validate the efficacy of our proposed solution for reducing the response time and task failure rate of navigation, danger assessment, and infotainment applications.

The remaining parts of this paper are organized as follows. In Section II, we have described our proposed system model and problem scenario. Moreover, a non-cooperative game theory-based distributed task offloading algorithm is also illustrated in this section. Afterward, we have carried out an extensive evaluation of our proposed scheme in Section III. Lastly, we conclude this paper in Section IV.

II. System Model

A. Problem Scenario

Task offloading in dynamic multi-user vehicular networks is still a challenging problem because of limited computing resources and vehicle mobility issues. Fig. 1 shows such scenarios, where RSU_1 is overloaded due to many offloading requests from the vehicles and it degrades the QoS. On the other hand, RSU_2 is lightly loaded. For example, $\mathfrak{f}_i^{max}$ represents the maximum computing resource capability of MEC server and $v\omega_1, v\omega_2, ..., v\omega_n$ represents the vehicle workload received by the RSU_1 from N vehicles. Then by using α_i, we can identify the difference between the capacity of the MEC server and the total receives workload.

$$\alpha_i = \mathfrak{f}_i^{max} - \sum_{i=1}^{n} v\omega_i \tag{1}$$

If $\alpha_i \geq 0$, then we can say that the MEC server has enough computing resources to handle new offloading requests after processing all the workloads received from the vehicles. On the other hand, if $\alpha_i < 0$, then the MEC server will need spare computing resources to execute the tasks because of the server overloaded problem. In this situation, RSU_1 can forward the overloaded task to the nearby RSU_2 or remote cloud server. Therefore, it becomes a challenging problem to decide which server is best for offloading the task. To overcome the RSU_1 overloaded problem and minimize the response time, we have proposed an efficient task offloading scheme where the decision of task offloading are taken through a game-theoretic approach.

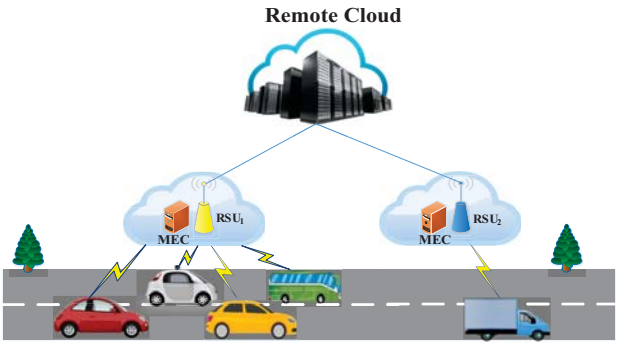

Fig. 1. Overloaded problem in vehicular network.

B. Proposed Model

Fig. 2 shows our proposed multi-user MEC-enabled vehicular networks architecture that consists of three layers named, data generation vehicle layer, edge computing networks layer, and remote cloud layer. In the data generation layer, the vehicles are located. This layer is regarded as the first tier of architecture. In the second layer, there are M RSUs are placed which is named as the edge computing networks layer for providing faster task computation. In the proposed architecture, the RSUs are also connected to a metropolitan area network (MAN) to share their computational capacity via task migration. Finally, to provide centralized cloud computing services, the traditional remote cloud server is located in the third tier. We use a fiber communication link to access the remote cloud resources from the RSUs.

The proposed model consist of a set of N vehicles as $N = \{1, 2, 3, \ldots, N\}$. Each vehicle has a set of computation task $T = \{T_i | i = 1, 2, 3, \ldots, T\}$ and each task is represented by the following, $T_i = \{\mathfrak{d}_i, \mathfrak{c}_i, \mathfrak{t}_i^{max}\}$. For task T_i, $\mathfrak{d}_i$ denotes the input task size for the computation; $\mathfrak{c}_i$ is the required computing resource to finish the computation task T_i; and $\mathfrak{t}_i^{max}$ represents the maximum processing latency that the task can tolerate. Moreover, we consider a unidirectional road where M RSUs are deployed equidistantly along the road having the same coverage range, R. We denote the set of M RSUs as

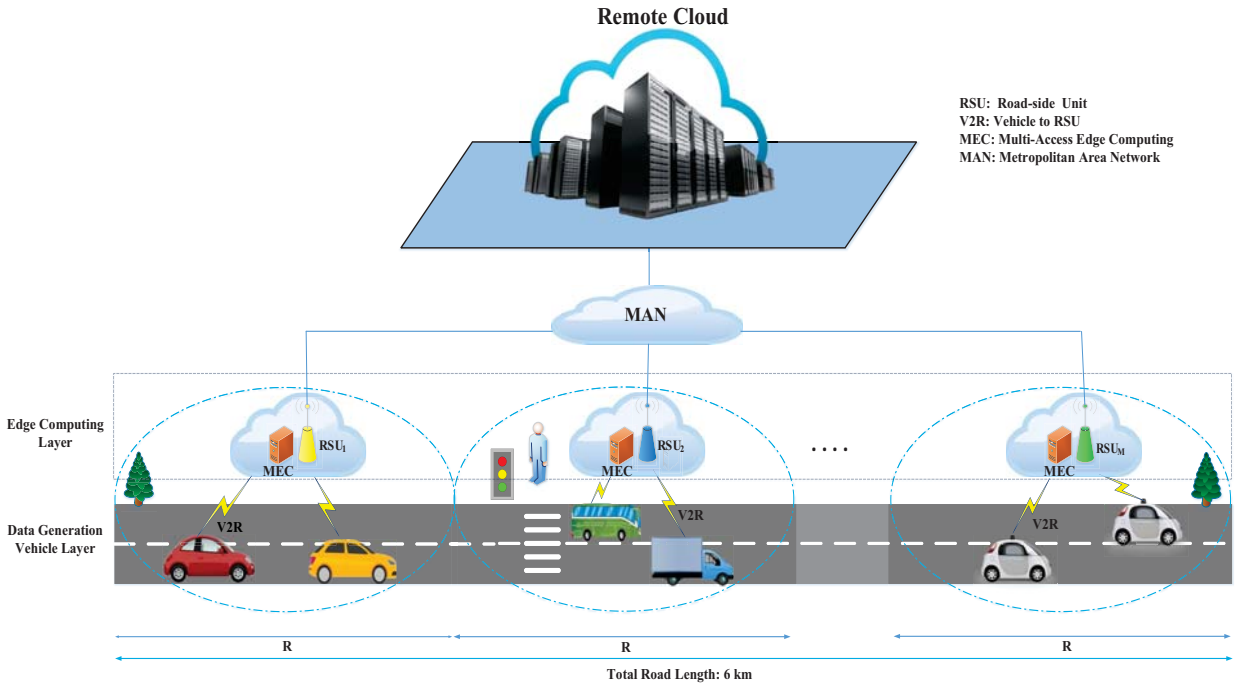

Fig. 2. The proposed MEC-enabled vehicular network.

$M = \{1, 2, 3, \ldots, M\}$, where $\{RSU_i | i = 1, ..., M\}$. Each RSU has 300 meters communication range and is equipped with a single MEC server which has limited storage and computing resources for processing tasks. For each MEC server, $MEC_i = \{\mathfrak{f}_i^{max}, \mathfrak{s}_i^{max}\}$, where $\mathfrak{f}_i^{max}$ is the computing resource capability of MEC_i, and $\mathfrak{s}_i^{max}$ is the local storage capacity of MEC_i. We assume that each MEC_i server has one host that operates four VMs. The resource capacity of each VM is 10 giga instructions per second (GIPS). Based on our model, the tasks can be executed either MEC server or remote cloud server. In both cases, we can divide the total processing time into two parts which are: task transmission duration (the time which requires the vehicle to offload the task) and task process duration (the time requires to process the task). So the total processing time to offload and execute the task T_i on the MEC server can be calculated as follows:

$$v_{i,mec}^{total} = v_{i,mec}^{trans} + v_{i,mec}^{exe} \tag{2}$$

Where $v_{i,mec}^{trans}$ is the transmission delay when i^{th} vehicle offloads their task to the MEC server for computing and $v_{i,mec}^{exe}$ denotes the processing latency for executing the task on the MEC server. The execution time $v_{i,mec}^{exe}$ can be calculated as follows:

$$v_{i,mec}^{exe} = \frac{\mathfrak{d}_t}{f_{mec}}(1 - U_{mec}) \tag{3}$$

Here, $\mathfrak{d}_t$ is the size of the task, f_{mec} and U_{mec} are the computation capability and utilization of the MEC server respectively. On the other hand, when the task is processed on the cloud server, we can calculate the total processing time of the task as follows:

$$v_{i,cloud}^{total} = v_{i,cloud}^{trans} + v_{i,cloud}^{exe} \tag{4}$$

Where $v_{i,cloud}^{trans}$ is the transmission delay when i^{th} vehicle offloads their task to the cloud server for computing and $v_{i,cloud}^{exe}$ denotes the processing time for executing the task on the cloud. The execution time $v_{i,cloud}^{exe}$ can be calculated as follows:

$$v_{i,cloud}^{exe} = \frac{\mathfrak{d}_t}{f_{cloud}}(1 - U_{cloud}) \tag{5}$$

Here, f_{cloud} and U_{cloud} are the computation capability and average utilization of the cloud server.

C. Non Cooperative Game Theory Based Task Offloading

We used a non-cooperative game theory-based task offloading approach for vehicular networks. To achieve the maximum utility of each vehicle, our proposed NGTO algorithm is used to adjust the task offloading probability which is given in Algorithm 1. This algorithm uses the best response offloading strategy to ensure that each vehicle can achieve a unique equilibrium under certain conditions. The updated best response offloading strategy is expressed as:

$$O_i = \left[\frac{v_{i,mec}^{total} - v_{i,cloud}^{total}}{2pv_{i,max}(1 - \prod_{i \neq j}(1 - \iota_j O_j))} \right]_0^1 \tag{6}$$

Where p indicates the pricing factor to decide either the task will be offloaded to the remote cloud or the MEC server by adjusting the degree of willingness. $v_{i,max}$ represents the maximum delay that is produced by vehicle i for generating the task. Moreover, ι represents the task arrival rate and the operator $\left[x\right]_0^1$ can cause the probability $O_i \in [0, 1]$.

Algorithm 1 NGTO Algorithm

1: Initialization: vehicle index i, task arrival rate vector ι, pricing vector p, offloading probability vector O.
2: **for all** task $t \in \mathcal{T}$ **do**
3: Estimate transmission delay $v_{i,mec}^{trans}$ of t^{th} task for MEC offloading // Due to WLAN delay
4: Estimate transmission delay $v_{i,cloud}^{trans}$ of t^{th} task for Cloud offloading // Due to WAN delay
5: $v_{i,mec}^{exe} = \frac{\mathfrak{d}_t}{f_{mec}}(1 - U_{mec})$ // Execution time of i^{th} vehicle of task t on MEC server
6: $v_{i,cloud}^{exe} = \frac{\mathfrak{d}_t}{f_{cloud}}(1 - U_{cloud})$ // Execution time of i^{th} vehicle of task t on Cloud server
7: $v_{i,mec}^{total} = v_{i,mec}^{trans} + v_{i,mec}^{exe}$ // Calculate total processing time of task on MEC server
8: $v_{i,cloud}^{total} = v_{i,cloud}^{trans} + v_{i,cloud}^{exe}$ // Calculate total processing time of task on Cloud
9: Update the best-response offloading strategy:
$$O_i = \left[\frac{v_{i,mec}^{total} - v_{i,cloud}^{total}}{2pv_{i,max}(1 - \prod_{i \neq j}(1 - \iota_j O_j))} \right]_0^1$$
10: **end for**
11: Vehicle i decides whether to offload its task with a probability O_i

III. Performance Evaluation

In this section, we evaluate the effectiveness of our proposed NGTO scheme for MEC-enabled vehicular networks through the EdgeCoudSim simulator [14]. To validate the performance of different scenarios, our proposed scheme compared with two existing task offloading schemes namely; local RSU computing (LRC) and collaborative offloading between the LRC and cloud via RSU. In the local RSU computing scheme, the computation tasks of vehicles are offloaded to the MEC server for processing which is located in nearby places. On the other hand, in the case of the collaborative offloading scheme, the offloaded tasks are processed between the LRC with a remote cloud server via RSU.

A. Simulation Setup

During simulation, we consider a real-life scenario, which has a 6 km length unidirectional road represented in Fig. 3. We divide the road into segments. In each segment, the vehicles have to maintain various speeds so that we can easily differentiate the density of the traffic on the road. In this simulation environment, we equidistantly deploy 20 RSUs along the road and each RSU has 300 meters communication range. A high-speed network infrastructure is built through the conventional fiber network to connect all the RSUs. In this experiment, we have used 1000 vehicles that are independently and uniformly distributed to random locations. Moreover, we have considered three different vehicle speeds for vehicular networks scenarios such as 25, 50, and 100 km/h. For example, the running speed of the vehicle is 25 km/h for the hotspot locations that is represented by using blue color in Fig. 3. Generally, the vehicles offload their computing tasks to the nearby RSU as well as remote cloud via RSU. Based on Lin et al.'s measurements for the vehicle to RSU communication [15], in this study, we have used IEEE 802.11p protocol WLAN to transmit the computation tasks between the vehicles and the RSUs.

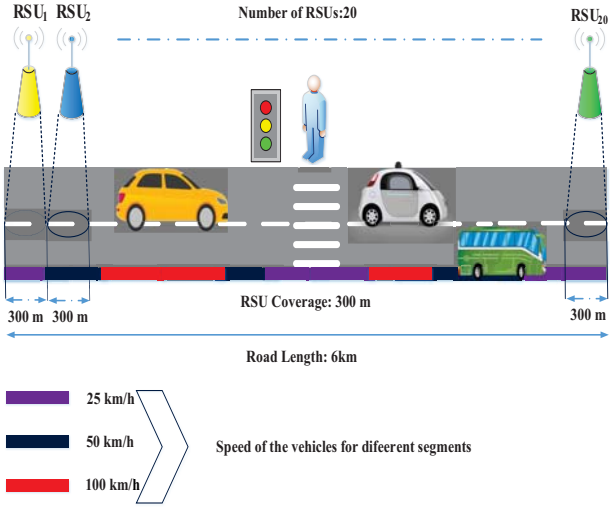

Fig. 3. Vehicular mobility model for the simulation.

Moreover, to generate the different tasks, each vehicle run three different applications during the experiments: navigation application (NA), danger assessment application (DA), and infotainment application (IA). Among them, navigation application and danger assessment application are latency-sensitive, and the infotainment application is latency-tolerant. In our simulation, we employed a delay sensitivity value for differentiating the sensitivity of various applications. We used a higher value of delay sensitivity for latency-sensitive applications and a lower value for latency-tolerant applications. Therefore, we assigned 0.8 for the value of delay sensitivity of danger assessment application and 0.25 for infotainment application. The important simulation parameters and the key characteristic parameters of different applications that are used during the simulation are given in Table I and Table II respectively.

TABLE I
Simulation Parameters

Parameters	Value
Number of repetitions	100
Number of vehicles	1000
Number of RSUs	20
Network delay model	MMPP/M/1 queue model
Number of VMs per RSU	4
Number of VMs per Cloud	20
VM processing speed per RSU	10 GIPS
VM processing speed in the Cloud	75 GIPS
Range of RSU (WLAN)	300 meters
WLAN/MAN bandwidth	10/1000 Mbps
WAN bandwidth	50 Mbps
WAN propagation delay	150 ms

TABLE II
EXPERIMENTAL PARAMETERS FOR THREE APPLICATIONS

Parameters	Application Types		
	Navigation Application	*Danger Assessment Application*	*Infotainment Application*
Usage (%)	30	35	35
Inter-arrival time of tasks (sec)	3	5	15
Maximum delay requirement (sec)	0.5	1	1.5
Delay sensitivity (sec)	0.5	0.8	0.25
Upload data size (KB)	20	40	20
Download data size (KB)	20	20	80
Average task length (GI)	3	10	20
RSU VM utilization (%)	6	20	40
Cloud VM utilization (%)	1.2	4	8

B. Simulation Results

To verify the importance of our proposed scheme, we performed an experiment showing the average response time with various numbers of vehicles which is shown in Fig. 4. From Fig. 4, it is observed that in all three schemes, the average response time tends to increase because of increasing the number of vehicles and the LRC scheme provides more response time than others. This is because of facing the congestion by MEC server to handle a large number of vehicles. When it comes to the collaborative and our proposed approach, the task will distribute between the LRC and the cloud via RSU. Therefore, the response time is not enhance compared to the LRC scheme. For example, when the number of vehicles is 500, then the response time for LRC, collaborative, and our proposed schemes are 0.87, 0.68, and 0.51 s, respectively. Therefore, by comparing with the other two approaches, our introduced NGTO scheme provides a lower response time than others. Because our proposed scheme can maximize the utility of each vehicle than its competitors. Therefore, it can reduce the average response time at almost 41.2% when compared with the LRC scheme and 25.2% when compared with the collaborative offloading scheme.

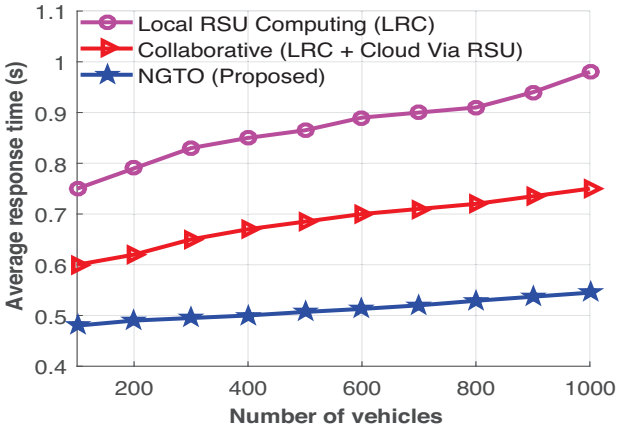

Fig. 4. Average response time in terms of number of vehicles.

One of the significant performance criteria for offloading is the task failure rate. If the VM utilization is too high, then the task is difficult to handle and it will be failed. Fig. 5 shows the effect of RSU VM capacity versus number of vehicles for the above-mentioned three offloading schemes. During the simulation, we employed four VMs in each MEC server. Due to the confined VM capacity, the LRC scheme faces congestion after 400 vehicles, collaborative scheme getting congested after 500 vehicles, and our proposed NGTO scheme can handle 1000 vehicles without congestion. Because of using a non-cooperative game theory-based offloading approach, our proposed scheme can utilize the MEC servers more efficiently than its competitors.

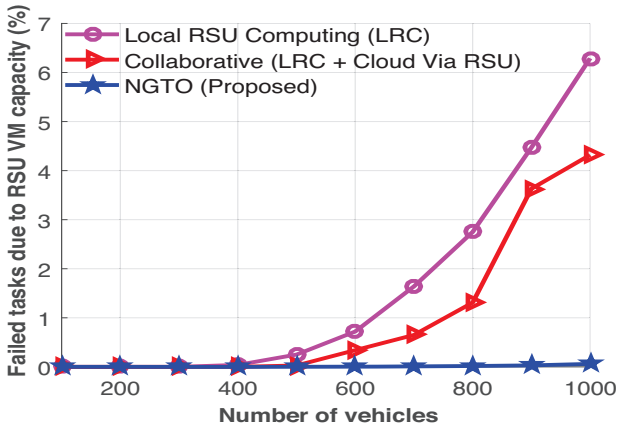

Fig. 5. Average failed task due to the RSU VM capacity.

Moreover, in Fig. 6, we investigate the average task failure rate for various vehicles. If the system is lightly loaded, we observe a lower task failure rate for all schemes. However, the situation changes as the vehicle's number increases. For example, the task failure rate is rapidly increased from 0.57% at 500 devices to 10.7% at 1000 devices in the LRC scheme; from 0.28% at 500 devices to 5.9% at 1000 devices in the collaborative scheme; and from 0.25% at 500 devices to 4.8% at 1000 devices in our proposed NGTO scheme. Therefore, after analyzing Fig. 6, our proposed NGTO scheme can reduce at approximately 56.3% and 20.4% task failure rates when compared with LRC and collaborative schemes respectively. Because our proposed system makes better decisions to maximize the utility of each vehicle about sending the tasks to the MEC servers as well as the remote cloud.

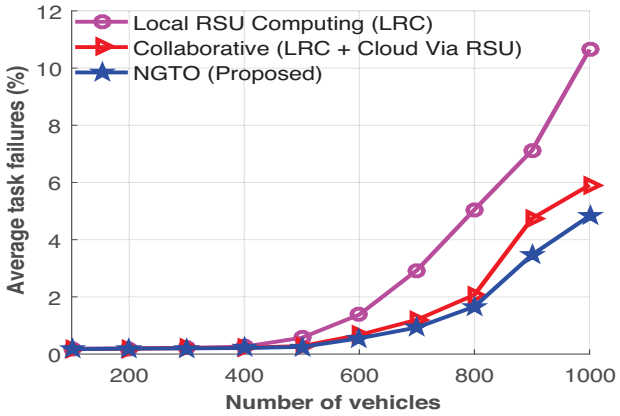

Fig. 6. Average task failure rate for varying number of vehicles.

Finally, we have done another experiment to investigate the impact of different vehicular speeds to measure the task failure rate which is shown in Fig. 7. During the experiment, we have used 500 vehicles. From analyzing Fig. 7, it is observed that, when the average vehicular speed is lower, the task failure rate is also lower in all schemes. But the situation is worse when average vehicular speed increases. For example, when the average vehicular speed is 100 km/h, the task failure rate of LRC, collaborative, and our proposed NGTO schemes are 0.63%, 0.47%, and 0.42% respectively. Throughout the above analysis, we can conclude that our proposal outperforms others.

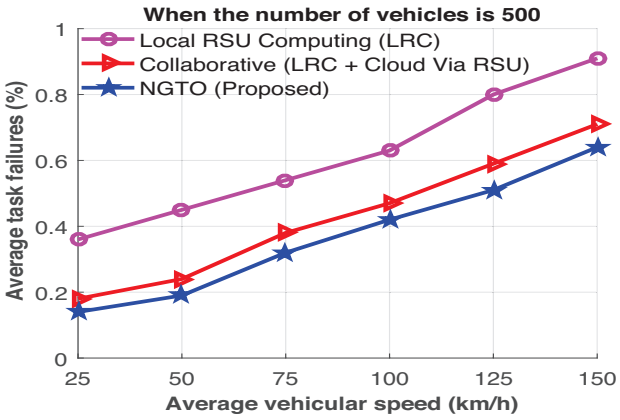

Fig. 7. Average task failure rate for different vehicular speed.

IV. Conclusion

Multi-access edge computing in vehicular networks is one of the prominent thought where vehicles offload their computing tasks to the RSU for rapid computation. However, one of the challenging issues in dynamic vehicular networks is task offloading where vehicles need to find out their efficient strategies for offloading their task in real-time. In this paper, we proposed a NGTO approach for efficient task offloading in MEC-enabled vehicular networks. For making the decision of task offloading, we used the game-theoretic based distributed best response offloading strategy. Furthermore, to assure the maximum utility of each vehicle under certain conditions, our proposed scheme accommodates its offloading probability to achieve a unique equilibrium. To evaluate our proposed scheme, we used navigation, danger assessment, and infotainment applications. Extensive simulation results affirm the best performance of the proposed NGTO scheme for reducing the response time and task failure rate in all scenarios compared to local RSU computing and collaborative scheme.

Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2017-0-00294, Service mobility support distributed cloud technology). Professor Eui-Nam Huh is the corresponding author.

References

[1] J. Feng, Z. Liu, C. Wu, and Y. Ji, "Ave: Autonomous vehicular edge computing framework with aco-based scheduling," IEEE Trans. Veh. Technol., vol. 66, no. 12, pp. 10660–10675, Jun. 2017.

[2] W. Sun, J. Liu, and H. Zhang, "When Smart Wearables Meet Intelligent Vehicles: Challenges and Future Directions," IEEE Wireless Communications, vol. 24, no. 3, pp. 58–65, Jun. 2017.

[3] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in Proc. IEEE Int. Con. Commun. (ICC), Paris, France, May 2017, pp. 1–6.

[4] X. Cheng, C. Chen, W. Zhang, and Y. Yang, "5G-enabled cooperative intelligent vehicular (5GenCiv) framework: When Benz meets Marconi," IEEE Intell. Syst., vol. 32, no. 3, pp. 53–59, May 2017.

[5] 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 Veh. Technol. Mag., vol. 12, no. 2, pp. 36–44, Jun. 2017.

[6] H. Guo, J. Zhang, and J. Liu, "FiWi-enhanced vehicular edge computing networks: Collaborative task offloading," IEEE Veh. Technol. Mag., vol. 14, no. 1, pp. 45–53, Mar. 2019.

[7] A. Alahmadi, A. Q. Lawey, T. E. El-Gorashi, and J. M. Elmirghani, "Distributed processing in vehicular cloud networks," in Proc. IEEE 8th Int. Con. Netw. Future, London, U.K., Nov. 2017, pp. 22–26.

[8] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated Offloading Scheme to Minimize Latency in MEC-Enabled Vehicular Networks," 2018 IEEE Globecom Workshops (GC Wkshps), 2018, pp. 1-6.

[9] M. D. Hossain, T. Sultana, M. A. Hossain, M. I. Hossain, Nguyen, N. T. Huynh, J. Park, and E. N. Huh, "Fuzzy Decision-Based Efficient Task Offloading Management Scheme in Multi-Tier MEC-Enabled Networks," Sensors vol. 21, no. 4: 1484, Feb. 2021.

[10] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in Proc. IEEE Int. Conf. Commun. (ICC), 2017, pp. 1–6.

[11] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks," IEEE Internet of Things Journal, vol. 6, no. 3, pp. 4377-4387, June 2019.

[12] T. D. T. Nguyen, V. Nguyen, V. -N. Pham, L. N. T. Huynh, M. D. Hossain, and E. N. Huh, "Modeling Data Redundancy and Cost-Aware Task Allocation in MEC-Enabled Internet-of-Vehicles Applications," IEEE Internet of Things Journal, vol. 8, no. 3, pp. 1687-1701, Feb. 2021.

[13] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task Offloading in Vehicular Edge Computing Networks: A Load-Balancing Solution," IEEE Transactions on Vehicular Technology, vol. 69, no. 2, pp. 2092-2104, Feb. 2020.

[14] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of Edge Computing systems," Transactions on Emerging Telecommunications Technologies, vol. 29, no. 11, pp.1–17, 2018.

[15] W.-Y. Lin, M.-W. Li, K.-C. Lan, and C.-H. Hsu, "A comparison of 802.11 a and 802.11 p for V-to-I communication: A measurement study," in Quality, Reliability, Security and Robustness in Heterogeneous Networks, Berlin, Germany: Springer, 2012, pp. 559–570.