RSU-Empowered Resource Pooling for Task Scheduling in Vehicular Fog Computing

Chaogang Tang*†, Chunsheng Zhu‡§, Xianglin Wei¶, Wei Chen*†, Joel J. P. C. Rodrigues‖**
*School of Computer Science and Technology, China University of Mining and Technology, 221116, Xuzhou, China
†Mine Digitization Engineering Research Center of the Ministry of Education,
China University of Mining and Technology, Xuzhou 221116, China
‡SUSTech Institute of Future Networks, Southern University of Science and Technology, 518055, Shenzhen, China
§PCL Research Center of Networks and Communications, Peng Cheng Laboratory, 518055, Shenzhen, China
¶The 63rd Research Institute, National University of Defense Technology, 410073, Changsha, China
‖ Federal University of Piauí, Teresina - Pi, Brazil
** Instituto de Telecomunicações, Portugal
{cgtang, Chenw}@cumt.edu.cn, chunsheng.tom.zhu@gmail.com, wei_xianglin@163.com, joeljr@ieee.org

***Abstract*—We in this paper consider a scenario where multiple vehicles jointly provision computing resources to obtain their benefits in the contexts of vehicular fog computing. A community that vehicles can freely join and leave is sponsored by a road side unit (RSU) and thus a resource pool is established such that tasks can be performed by sufficient computing resources. RSU as a coordinator takes in charge of decision making for task scheduling. A permutation of community members is established in advance and updated periodically so as to make the most suitable decision. A task scheduling strategy is proposed from the perspective of service oriented architecture. We have carried out the experiments to investigate our approach and the experimental results have revealed our approach has a great advantage over other approaches in terms of pursuing the values of the community.**

***Index Terms*—RSU, community, task scheduling, vehicular fog computing, permutation**

I. Introduction

Internet of vehicles (IoVs) has gained extensive attention recently both in academia and industry, due to its tremendous benefits such as traffic safety guarantee, location awareness, and convenience in smart cities. In this context, vehicles are envisioned as intelligent agents that are equipped with various sensors and facilities, for the purpose of data packets delivery, information dissemination, and multimedia sharing. Rapidly developing vehicle-to-vehicle (V2V) communication and vehicle-to-infrastructure (V2I) communication technologies have laid the foundation for the vision of IoV. For example, the dedicated short range communications (DSRC) is one of these technologies that is designed based on IEEE 802.11p and intended for V2V and V2I communication. The infrastructure in V2I usually refers to the road side units (RSUs) deployed along the road with multiple capabilities. In this regard, RSUs can serve as access points (APs) to provision wireless radio resources for moving vehicles, and on another hand, they can act as relay stations to forward information for vehicles that cannot directly communicate with each other.

Actually, RSUs have been empowered with more roles in newly emergent computing paradigms such as vehicular fog computing (VFC) and mobile edge computing (MEC). In the context of smart cities, everything is interconnected and the interaction is everywhere, with the aim to deliver sustainable and prosperous future for the citizens [1]. To this end, an immense amount of data is generated by various internet of thing (IoT) devices all the time, which poses a giant challenge for both computing and wireless radio resources. This is because the captured data (e.g., in IoV) should be disseminated and handled in a cost-efficient and time-sensitive way. Offloading data to the cloud centers for processing and analysis renders serious network bandwidth consumption and long response latency. It may not be practical for the applications with strict delay requirements. Accordingly, fog computing and edge computing are proposed as countermeasures to mitigate the pressure on these resources, which extends the computing and storing resources to the edge of network, thus enabling to run computationally intensive applications with strict delay requirements. Furthermore, VFC, derived from fog computing, envisions an enticing computing scenario where tasks can be offloaded and executed at the vehicles that act as fog nodes to contribute the idle computing resources for their own benefits. As such, the response latency can be reduced drastically. RSU in VFC can serve as an immobile fog node with much more powerful computing and storing capabilities, and even as a coordinator responsible for computing resource allocation for the arrived tasks.

The issue of resource allocation in VFC has been extensively investigated in the existing literature such as [2] [3] [4] [5] [6]. However, we notice that few of them have considered the revenue or payoffs of vehicles earned by contributing computing resources. It is worthwhile mentioning that the payoff plays an important role in stimulating vehicles to contribute their computational resources. For example, vehicles prefer not to be involved in task execution if the payoffs are far from their expectation. In this paper, we consider a scenario where multiple vehicles cooperatively provision computing resources to obtain more revenues. For this purpose, vehicles can freely join a community sponsored by an RSU and thus a

978-1-7281-3129-0/20/$31.00 ©2020 IEEE

resource pool is established such that tasks can be performed by sufficient computing resources. Compared to the way vehicles provision computing services individually, at least two advantages can be achieved. On one hand, resource requesters (e.g., mobile devices) do not have the global information on the fog nodes such as the amount of resources, processing rate, and price. Thus, multiple interactions with vehicle fogs in vicinity are required to make the most suitable offloading decision. This process can incur extra latency, which may even tremendously degrade the quality of user experience. Note that for the ease of discussion, we use fog nodes and vehicle fogs exchangeably hereafter.

On the other hand, vehicles in VFC are usually featured by high mobility and the wireless connection is intermittent and unstable. Thus, task offloading and result return can be error prone from the viewpoint of data delivery. Further to this, it may be not successful for task execution at the vehicle fog either, due to some inherent causes such as system-level errors. As a result, reliability as a performance indicator should be considered to comprehensively evaluate the performance of vehicle fogs when they provision computing resources as service providers. However, most of existing works do not mention it when scheduling the tasks in MEC and VFC.

In our scenario, we investigate the task scheduling and resource allocation from the viewpoint of service oriented architecture (SOA) [7] while taking into account the aforementioned issues. Instead of initiating direct interactions between resource requestors and vehicle fogs, the requestor sends the request to RSU, where resource allocation plan for this request is performed and the most suitable vehicle fog is determined. In the next, the information on this vehicle fog is returned back to the requestor. Then, tasks are offloaded and executed at this fog node.

Actually, we strive to maximize the value of community (defined in section III) that takes into account both the payoffs and the reliability of the vehicle fogs. In such way, vehicles are stimulated to join the community for their own benefits. Note that there is a basic assumption that all the vehicles in the same community are willing to follow the rules designed by RSU. For example, all the vehicle fogs that join the community accept the way tasks are allocated, and they are only responsible for task execution when tasks arrive.

The rest of paper is organized as follows. In section II, we review some works on task scheduling in VFC and MEC. In section III, we present our system model and mathematically formulate the objective function. In section IV, we investigate our approach with other two approaches, followed by the experimental results analysis. We conclude our work in section V.

II. Related Works

Recently, much more attention has been paid to applying VFC to various scenarios where vehicles as fog nodes compute their idle computing resources, which can hopefully reduce the respond latency drastically, especially in the contexts of smart cities. Tasks with the strict time requirements from the mobile terminals are explosively increasing, which necessitates the task offloading and execution at the edge of the networks and also stimulates numbers of researches and works in this regard. VFC offers a good solution to this problem and RSU plays an important role in the task scheduling. For instance, RSU is utilized to assist the distributed trust framework with the purpose of privacy preservation in VANETs [8]. Authors in [9] investigate the information delivery in the contexts of connected vehicles. They strive to find the minimal dominating set of vehicles such that other vehicles out of this set but centering the same RSU can communicate with those in this dominating set by one hop. By doing so, the success rate of the information delivery can be improved. Authors in [3] strive to utilize the computing resources of the parking vehicles to provision computing services. They have proposed the corresponding architecture, conducted feasibility analysis, and investigated their thought experimentally. Authors in [2] consider a use case where parking slots are difficult to find in the peaking hours. To improve the parking efficiency, they have proposed a fog computing based parking strategy where an RSU acts as a fog node responsible for task scheduling and decision making.

Due to lack of universally accepted definition of vehicular fog computing as well as specific use cases, authors in [4] aim to propose a general architecture of vehicular fog computing and definition. Furthermore, they demonstrate a typical use case in the next, followed by several key security and forensic challenges discussions. In the context of smart city, some new applications such as real-time vision processing need a almost realtime respond, which poses a great challenge for the current sensor cloud systems where data are captured by sensors but executed and analyzed in the remote cloud center. This kind of respond latency may not satisfy the realtime requirement. Task scheduling thus becomes critical, especially considering the limited computing resources and energy at the mobile terminal side. Accordingly, authors in [5] designs a cooperative strategy for IoV and endeavor to minimize the makespan with specified conditions. To be more concrete, they have proposed algorithms based on the alternating direction method of multipliers (ADMM) algorithm to optimize the objective function.

In IoV, large quantity of data are supposed to be exchanged and shared by RSU, hopefully to improve the quality of driver experience. Nevertheless, unstable wireless links and high dynamics of vehicles make data delivery prone to fail. Thus, authors in [11] investigate the applicability of the information-centric networking paradigm in vehicular environments. The results have revealed that this kind of networking is tailed to satisfy the challenges.

Other works such as [12] [13] also strive to address similar problems in the task scheduling. However, we found that none of the aforementioned works have considered the benefits of the vehicles acting as fog nodes, while the benefits of the vehicles are very important to stimulate their enthusiasm to contribute the idle computing resources.

III. System Model

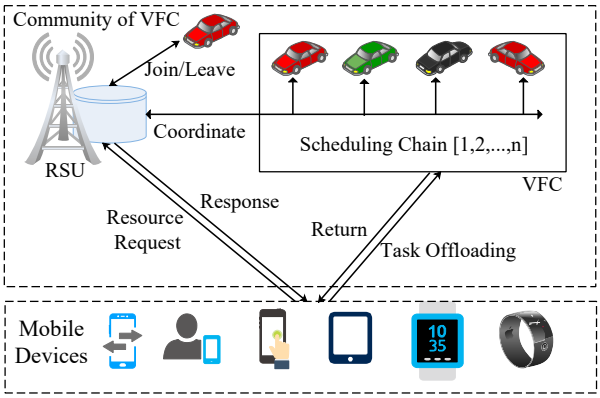

Fig. 1. Architecture of task scheduling in community.

Consider a community C consisting of one RSU R and multiple vehicles within the communication range of R. Denote by $Num(t)$ the number of vehicles in C at time t. Assume that these vehicles are willing to contribute the computing resources and accept the way revenues are distributed in C. A vehicle fog can be simply represented by a tuple $(veh\_ID, R\_avail, s, \gamma, p)$, where veh_ID uniquely identifies the vehicle, R_avail is the available computing resources that can be contributed to C, s denotes the speed of the vehicle, γ represents the reliability of the vehicle and can be defined as the probability that the tasks are performed successfully. p is the price per unit of resource. Note that γ can be obtained from the historical data and an average value of γ can be assigned to the vehicles that serve as fog nodes at the first time. A task offloaded to C for execution can be denoted by a tuple $T = (tsk\_ID, R\_req, bgt, dl)$, where tsk_ID is the identification of the task, R_req denotes the amount of resources required for the task execution, bgt is the budget that the requester can afford for using the resources and dl is the maximal execution time that the requester can tolerate. We assume that the resource requests come from the mobile terminals such as smart phones, smart watches and so on. Thus, the locations of these terminals can be obtained easily, e.g., by built-in GPS. It is vital for the response time calculation when tasks are offloaded and executed.

Fig. 1 depicts the architecture of task scheduling in a community sponsored by R. All the vehicles willing to join the community C register themselves in R, such that R has the global information on its community members. The information plays a critical role in determining the task scheduling strategy. At the beginning, resource requesters such as mobile users send their requests to RSU and wait for the response. RSU receives the requests and determines the suitable vehicle fog based on the scheduling chain. A scheduling chain is a dynamic permutation $\sigma = [\sigma[1], \sigma[2], ..., \sigma[Num(t)]]$ of the community members $\{1,2,...,Num(t)\}$ that dynamically determines the vehicle fog to respond to the request. For example, once R as a coordinator receives a request, it responds to the request by sending the information of vehicle fog $\sigma[1]$ to the requester. Then, tasks can be offloaded to $\sigma[1]$ and executed. If $\sigma[1]$ is not available, e.g., lack of sufficient computing resources, $\sigma[2]$ as a backup is selected to respond to this request. We assume that each vehicle fog can only execute one task each time. Similarly, if $\sigma[2]$ is not available, another backup $\sigma[3]$ is selected. This procedure continues until a vehicle fog is available. Assume that vehicle fog i's availability is α_i. Then, the reliability of the community C is defined as [7]

$$\gamma_C = 1 - \prod_{i=1}^{Num(t)} (1 - \alpha_i \cdot \gamma_i) \tag{1}$$

Usually, there is a cost associated with task execution for each vehicle fog. We assume that the costs on tasks execution in C mainly depend on its community member to which the task is assigned. Denote by c_i the cost on task execution at vehicle fog i. Thus, given the scheduling chain σ, we define the cost of community C on task execution as

$$\mathbb{C}_C(\sigma) = \sum_{i=1}^{Num(t)} \prod_{j=1}^{i} (1 - \alpha_{\sigma[j]} \cdot \gamma_{\sigma[j]}) c_{\sigma[j]} + \beta \tag{2}$$

where $c_{\sigma[j]}$ is the cost of vehicle fog $\sigma[j]$, and β is a constant denoting the extra cost coming from the interactions between C and resource requesters as well as between C and its community member. Denote by $V_C(\sigma)$ the value of community. Usually, there are several ways to define the value of community. The value of the community can totally be independent of its community members or partially dependent on its members. Instinctively, the definition of the value of the community should be related to the price of its members. To ease the discussion, all the vehicle fogs with the same reliability are assumed to have the same price in this paper. Thus, the value of community C for one task execution can be defined as

$$V_C(\sigma) = P_C \cdot \gamma_C - \mathbb{C}_C(\sigma) \tag{3}$$

where P_C is the price of the community dependent on the community members that execute the tasks. In our scenario, we assume that R responds to the user request one by one, following the First-Come-First-Serve (FCFS) principle. Due to the high dynamics of vehicles in VFC, the community members may change frequently, e.g., vehicles leave the community due to out of community range and vehicles join the community for their own benefits. Thus, the global information of community members should be updated periodically such that R can make the most suitable decision.

In this paper, we strive to maximize $V_C(\sigma)$ while considering the payoffs of the community members. To this end, two issues in this process should be addressed, e.g., regarding 1) the scheduling chain definition and 2) the design of the community price. Specifically, the scheduling chain σ should be defined comprehensively and fairly such that the motivation

of vehicles to join the community should be stimulated always. A vehicle that joins the community but does not get the opportunities to execute the tasks may lose enthusiasm and motivation soon. Accordingly, the reputation of the community may degrade, which further hinders other vehicles from joining the community. On another hand, the community price should be also designed appropriately, since high pricing may temper the resource requesters' enthusiasm while low pricing can reduce the benefits of the community and its community members.

A. Scheduling Chain Definition

The scheduling chain denotes the order of vehicle fogs which are designated to respond to the request. Intuitively, the further toward the front of the scheduling chain, the better. This is because the fogs in the front of the scheduling chain may get more opportunities, and each vehicle fog hates being located at the back of the scheduling chain. This common sense poses an issue concerning how to define the scheduling chain while considering the fairness and the benefits of the community members. In this paper, we assume that the scheduling chain can be predefined in each time slot and it needs to be updated periodically due to the high dynamics of the vehicle fogs.

The information registered in R can be leveraged to design the scheduling chain. Since all the fogs in R have the same purpose, i.e., pursuing their own benefits by contributing the computing resources. The main difference is that they may have different computing resources, vehicle speed, the reliability and the price per unit of resource, as denoted in the defined tuple $(veh\_ID, R\_avail, s, \gamma, p)$. We can refer to these attributes of the fog vehicles as the quality of service (QoS) attributes. To be more specific, we denote the QoS of each vehicle i by a quad $\mathcal{Q}_i = (R\_avail, s, \gamma, p)$ with each element $\mathcal{Q}_i^j (1 \leq j \leq 4)$ having the same meaning as defined earlier. Thus, we can define a utility function for all the vehicle fogs in R which incorporates all the QoS attributes into a single value by a simple additive weigh technology. It is worth mentioning that there exist two different kinds of QoS attributes for each vehicle fog in R, i.e., the positive ones and the negative ones. Usually, for a positive attribute, larger value of the attribute denotes higher quality while for a negative attribute, larger value of the attribute denotes lower quality. For example, in this paper, R_avail and γ are the positive ones while s and p are the negative ones. For a general case, assume that the number of positive and negative attributes are m and n, respectively. Denote by $\mathcal{U}(i)$ the utility of the ith vehicle fog in R and

$$\mathcal{U}(i) = \sum_{j=1}^{m} w_j \cdot \frac{\mathcal{Q}_i^j - \min(\mathcal{Q}_t^j)}{\max(\mathcal{Q}_t^j) - \min(\mathcal{Q}_t^j)} + \sum_{j=1}^{n} w_j \cdot \frac{\max(\mathcal{Q}_t^j) - \mathcal{Q}_i^j}{\max(\mathcal{Q}_t^j) - \min(\mathcal{Q}_t^j)}, \forall t \in [1, Num(t)] \tag{4}$$

where $\sum_{j=1}^{m} w_j = 1$, $\sum_{j=1}^{n} w_j = 1$ and $w_j (\in (0,1))$ denotes the preference towards different QoS attribute. Generally speaking, R_avail is the most important attribute since it determines whether the task can be performed successfully. R can not allocate the request to vehicle, say i, if the computing resources available cannot satisfy the requirement of the request even though other three QoS attributes of i are much better than others in the community. Accordingly, the weight over R_avail should be larger than other ones. R ranks the vehicle fogs in the descending order of utility values and thus the scheduling chain can be established in this order.

B. Community Price Definition

The other issue we need to address is how to define the price of the community P_C. Inappropriate pricing can either temper the resource requesters' enthusiasm or reduce the payoffs of the community members. However, pricing exactly the same as the fog node declared will neglect the benefits of the community itself. Generally speaking, the pricing should be a little higher than the price the fog node has declared, while taking into account the characteristics of the fog nodes responsible for performing the tasks. For simplicity, we define P_C as follows

$$P_C = \tilde{p} + \triangle \tag{5}$$

where $\tilde{p}$ is the price of the final vehicle fog that responds to the user request successfully and $\triangle$ is the absolute deviation and given as

$$\triangle = \frac{1}{Num(t)} \cdot \sum_{i=1}^{Num(t)} \sqrt{|\mathcal{Q}_i^4 - \overline{p}|} \tag{6}$$

where $\mathcal{Q}_i^4$ is the price of the ith fog node in the community and $\overline{p}$ is the average price of all the vehicle fogs in the community. Thus, the price of the community is higher than arbitrary vehicle fog in the community.

C. Problem Formulation

As mentioned earlier, the community R responds to the request one by one. We further assume that there is no concurrent user requests at each time unit. To model the arrival of the task, we use a binary variable $\phi(t)$ to denote whether there is a user request at time t. If there is a request arriving in R at t, $\phi(t) = 1$; otherwise $\phi(t) = 0$. The information on community members is updated every time slot τ. The vector $\overline{\phi} = [\phi(1), \phi(2), ..., \phi(\tau)]$ can be generated randomly to model the random arrival of the tasks. Then we can mathematically formulate our objective function as follows:

$$(\mathcal{P}) \quad Maximize: \quad \sum_{t=1}^{\tau} \phi(t) \cdot V_C(\sigma) \tag{7}$$

s.t.

$$\phi(t) \cdot RS(t) \leq dl_t \tag{8}$$

$$\phi(t) \cdot \mathbb{C}_C(\sigma) \leq bgt_t \tag{9}$$

$$\tau \leq Num(k) \quad \forall k \in [1, \tau] \tag{10}$$

$$\phi(t) \in \{0, 1\} \tag{11}$$

where inequation (8) denotes that the task arriving at time t (if any) should satisfy that the response time denoted by $RS(t)$ should be smaller than the corresponding deadline of the task. Similarly, inequation (9) represents that the cost on task performing in the community should not exceed the corresponding budget of the task. Inequation (10) underlines that the update frequency should be smaller than the number of vehicle fogs during the period. Inequation (10) can guarantee that the number of tasks is always smaller than the number of vehicle fogs. Thus, it can reduces the probability that the arriving tasks find no idle vehicle fog available. Whereas it is still possible that the task scheduling is not successful due to the constraint conditions violation. In this case, R repeatedly scans the vehicle fogs following the sequence of the scheduling chain until one vehicle fog is available that satisfies the constraint condition.

Problem $\mathcal{P}$ strives to maximize the benefits of the community, but it considers the payoffs of its community members, for the reason that the permutation σ is established based on $\mathcal{U}$ with an emphasis on the benefit of each vehicle fog. As for the calculation of the response time of task performing, we can refer to our previous work [14]. Due to the space limitation, we do not detail it any longer in this paper.

To be more specific, the procedure of the task scheduling in R is shown in Algorithm 1. At the beginning, task set and vehicle fog set is established based on $\overline{\phi}$ and $\{\mathcal{Q}_i\}$ respectively. For each task in the task set, R repeatedly scans the scheduling chain until a vehicle fog is available that satisfies the requirements of the task including the deadline and budget. Then the task is offloaded to this vehicle fog and executed. Meanwhile the costs on task performing are calculated based on equation 5. We use $\mathcal{V}$ in the algorithm to denote the accumulated community value and $\mathcal{V}$ is returned finally.

Algorithm 1: Task Scheduling in R Based on Scheduling Chain

Input: R, $Num(t)$, τ, $\{\mathcal{Q}_i | i \in [1, Num(t)]\}$
Output: Community value $\mathcal{V}$

```
Generate the vector φ̄ randomly;
Construct the task set {T} based on φ̄;
Construct the vehicle fogs based on {Q_i};
Calculate the utility value of each fog in R using eq.4;
Define σ based on utility values of vehicle fogs;
V = 0;
for each T_j in task set do
    for each vehicle fog f_i in σ do
        while f_i is available do
            if Q_i^1 ≥ T_j.R_req then
                Estimate the response time RS(T_j);
                Calculate the cost cst using eq.5 and 6;
                if RS(T_j) ≤ T_j.dl and cst ≤ T_j.bgt then
                    Task T_j is executed at f_i;
                    Update the status of fog f_i;
                    Calculate the community value v
                    using eq.3;
                    V = V + v;
                    break;
                end
            end
        end
    end
end
return V;
```

TABLE I
PARAMETER SETTINGS

Notations	Meanings	Values
γ	Reliability	[0.4, 0.9]
$Num(t)$	Number of vehicle fogs at t	[10, 20]
p	Price per unit of computing resources	[4,10]
R_avail	Available computing resources	[50,100]
R_req	Computing resources required	[30, 70]
c_i	The cost of fog i for performing one task	[1,3]

IV. Experimental Investigation

A. Assumptions and Experimental Settings

In this section, we have carried out the experiments to evaluate the task scheduling strategy based on the scheduling chain. A few of assumptions have been made as follows. First, the number of vehicle fogs in R at each time period is randomized. Second, each vehicle fog is able to execute at least one task but three tasks at most due to their high mobility. Third, the processing speed is assumed to be the same for all the vehicle fogs in R. Last but least, we assume that resource consumers tend to select the communities instead of individual vehicle fogs. Usually, the community with a good reputation always attracts much more attention and receive large number of user requests, even in reality. Specifically, parameters involved in this experiment are listed in Table I.

B. Strategy Comparison and Results Analysis

Two approaches are compared to our approach in the evaluation. One is the random approach, denoted by "$Random$", which establishes the scheduling chain in a random way. Specifically, in "$Random$", for each position of the scheduling chain, it chooses the vehicle fog randomly. Thus, we can see that the scheduling chain is established without the consideration of the benefits of vehicle fogs. The other is the price preferred approach, denoted by $Price\_order$, which establishes the scheduling chain in the descending order of the price of vehicle fogs in the community. We can see that this approach considers the revenues of the community members, but it neglects the idle computing resources of each vehicle fog.

The experimental result is shown in Fig. 2, where the x-coordinate represents the number of user requests and y-coordinate represents the average value of the community C. From the figure, we can easily observe that our approach in this paper denoted by "$Utility\_based$" is much better than "$Random$" and "$Price\_order$". By comparison, "$Price\_order$" is the worst in terms of optimizing the average value of the community in the three approaches. It is understandable and acceptable for the reason that on one hand, it neglects the effects of QoS attribute R_avail on the fogs selection. According to the aforementioned descriptions, in the vector $\mathcal{Q}_i = (R\_avail, s, \gamma, p)$, $\mathcal{Q}_i^1$(i.e., R_avail) is the most important attribute when tasks are scheduled to the fogs for execution. In other words, the fog node cannot be selected by R if R_avail is not qualified for the user requirement no matter how better s, α and p are. On the other hand, the vehicle fogs and tasks are generated in the random way, which increases the chances of constraints violation.

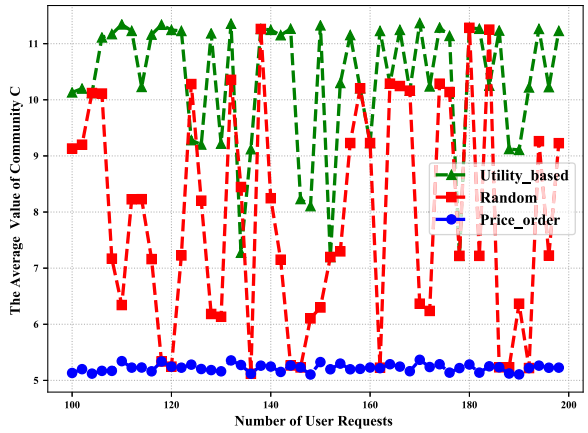

Fig. 2. The comparison among different approaches.

As for the average value of the community, our approach averagely improves the performance by 31.99% and 98.09%, compared to "$Random$" and "$Price\_order$", respectively. Accordingly, the results can reveal that our approach indeed has a great advantage over other two approaches with regards to community value optimization.

V. Conclusion

Currently, few of works in IoV and VFC have considered the revenue or payoffs of vehicles earned by contributing computing resources. It is worthwhile mentioning that the payoff plays an important role in stimulating vehicles to contribute their computational resources. We in this paper empower RSU with an important role from the viewpoint of SOA. Namely, RSU accounts for request processing and decision making for task scheduling, which avoids multiple interactions between vehicle fogs and resource requesters, and thus improves the efficiency of task scheduling. Further to this, we take into consideration the reliability of each vehicle fog that however usually is neglected in most of existing literature. We define a permutation of community members based on a utility function to maximize the value of the community. The results have shown that the strategy proposed in this paper outperforms other approaches.

For the future work, we plan to design more efficient strategy to scheduling the tasks in the contexts of vehicular fog computing while considering the benefits of the community members.

Acknowledgment

This work was partially supported by the project "PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019)", by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/EEA/50008/2020 and by Brazilian National Council for Scientific and Technological Development (CNPq) via Grant No. 309335/2017-5. Chunsheng Zhu is the corresponding author.

References

[1] J.M. Shapiro, "Smart Cities: Quality of Life, Productivity, and the Growth Effects of Human Capital," in The Review of Economics and Statistics, vol. 88, no.2, pp. 324–335, 2006.

[2] C. Tang, X. Wei, C. Zhu, W. Chen and J. J. P. C. Rodrigues, "Towards Smart Parking Based on Fog Computing," in IEEE Access, vol. 6, pp. 70172–70185, 2018.

[3] X. Hou, L. Yong, C. Min, W. Di, D. Jin, and C. Sheng, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," IEEE Transactions on Vehicular Technology, vol. 65, no. 6, pp. 3860–3873, 2016.

[4] H. Cheng, R. Lu, and K. K. R. Choo, "Vehicular fog computing: Architecture, use case, and security and forensic challenges," IEEE Communications Magazine, vol. 55, no. 11, pp. 105–111, 2017.

[5] Y. Deng, Z. Chen, X. Yao, S. Hassan, and J. Wu, "Task scheduling for smart city applications based on multi-server mobile edge computing," IEEE Access, vol. 7, pp. 14410–14421, 2019.

[6] C. Xu, J. Lei, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," IEEE/ACM Transactions on Networking, vol. 24, no. 5, pp. 2795–2808, 2016.

[7] A. Liu, Q. Li, L. Huang, S. Ying and M. Xiao, "Coalitional Game for Community-Based Autonomous Web Services Cooperation," in IEEE Transactions on Services Computing, vol. 6, no. 3, pp. 387–399, July-Sept. 2013.

[8] S. Wang, N. Yao, "A RSU-aided distributed trust framework for pseudonym-enabled privacy preservation in VANETs, " in Wireless Networks, vol. 25, no. 3, pp. 1099C1115, 2019.

[9] A. Chinnasamy, B. Sivakumar, P. Selvakumari, A. Suresh, "Minimum connected dominating set based RSU allocation for smartCloud vehicles in VANET," vol. 22, s. 5, pp. 12795–12804, September 2019.

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

[11] M. Amadeo, C. Campolo, A. Molinaro, M. Amadeo, C. Campolo, A. Molinaro, M. Amadeo, C. Campolo, and A. Molinaro, "Information-centric networking for connected vehicles: A survey and future perspectives," IEEE Communications Magazine, vol. 54, no. 2, pp. 98–104, 2016.

[12] A. Ghosh, V. V. Paranthaman, G. Mapp, O. Gemikonakli, and J. Loo, "Enabling seamless v2i communications: Toward developing cooperative automotive applications in vanet systems," Communications Magazine IEEE, vol. 53, no. 12, pp. 80–86, 2015.

[13] J. Feng, L. Zhi, C. Wu, and Y. Ji, "Ave: Autonomous vehicular edge computing framework with aco-based scheduling," IEEE Transactions on Vehicular Technology, vol. PP, no. 99, pp. 1–1, 2017.

[14] C. Tang, X. Wei, C. Zhu, Y. Wang, W. Jia, "Mobile Vehicles As Fog Nodes For LatencyOptimization In Smart Cities," IEEE Transactions on Vehicular Technology, accepted, 2020, doi: 10.1109/TVT.2020.2970763.