Mobility Prediction-Based Joint Task Assignment and Resource Allocation in Vehicular Fog Computing

Xianjing Wu[1,2], Shengjie Zhao[1,2], Rongqing Zhang[1,3], Liuqing Yang[4]
[1]School of Software Engineering, Tongji University, Shanghai, China.
[2]Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai, China.
[3]National Mobile Communications Research Laboratory, Southeast University, China.
[4]Department of Electrical & Computer Engineering, Colorado State University, CO, USA.
{wanwxj, shengjiezhao, rongqingz@tongji.edu.cn, lqyang@engr.colostate.edu}

***Abstract*—Most recently, vehicular fog computing (VFC) has been regarded as a novel and promising architecture to effectively reduce the computation time of various vehicular application tasks in Internet of vehicles (IoV). However, the high mobility of vehicles makes the topology of vehicular networks change fast, and thus it is a big challenge to coordinate vehicles for VFC in such a highly mobile scenario. In this paper, we investigate the joint task assignment and resource allocation optimization problem by taking the mobility effect into consideration in vehicular fog computing. Specifically, we formulate the joint optimization problem from a Min-Max perspective in order to reduce the overall task latency. Then we decompose the non-convex problem into two sub-problems, i.e., one to one matching and bandwidth resource allocation, respectively. In addition, considering the relatively stable moving patterns of a vehicle in a short period, we further introduce the mobility prediction to design a mobility prediction-based scheme to obtain a better solution. Simulation results verify the efficiency of our proposed mobility prediction-based scheme in reducing the overall task completion latency in VFC.**

***Index Terms*—VFC, task assignment, resource allocation, mobility prediction**

I. Introduction

With the rapid development of network and computing technology, intelligent transportation systems (ITS) come to practical application from theory gradually. Meanwhile, emerging 5G vehicular network applications such as self-driving, object detection, object recognition with complicated data processing and constraint delay, promote the growing demand for more powerful computing capacity, higher transmission rate and lower time delay [1]. This poses challenges to the conventional cloud computing paradigm, which cannot guarantee real-time feedback. In contrast, fog computing extends both computing and caching abilities to network edges, forming the ability of local decision-making and quick response, which are key factors in the vehicular network.

Many recently body of research concerns introducing Fog computing paradigm to vehicular network [2]–[4]. Most of their proposals consider the resources of vehicles as part of the fog computing infrastructures [5]. Different from Vehicular Cloud Computing (VCC), the VFC framework processes vehicles' tasks locally as much as possible instead of sending computation requests to cloud servers. As a significant component of ITS, vehicular fog computing can support complex vehicular services by collaborative vehicle scheduling. The quality of services (QoS) and quality of experience (QoE) can achieve great improvements based on better utilization of each vehicle's communication and computational resources. However, due to the mobility of vehicles, the task assignment and resource allocation (TARA) problem becomes more challenging to solve.

As one important characteristic that we pay more attention to, mobility support has been studied from different perspectives. Due to vehicles' mobility, the relative positions of vehicles change rapidly, leading to rapid changes in the network topology. Hence, the influence of mobility cannot be neglected in VFC. Relationships among the communication capability, connectivity, and mobility of vehicles were unveiled in [5], the results indicate that the faster a vehicle moves, the fewer vehicles it can successfully connect with, leading to poorer communication capacity. In [6], to provide efficient and reliable service to users in VFC, vehicles are divided into three sub-networks based on their turning directions. It is a good solution to formulate vehicular network clusters. The connections between client vehicles and mobile fog nodes may not last until the assigned tasks completed due to the mobility, so task migration can be triggered based on the mobility information [7], [8], or the fog vehicle that cannot provide service to the end should be excluded [9]. Although mobility is taken into account in the above works, they do not involve mobility in their optimization problems, i.e., the mobility feature is more like a trigger condition.

Therefore, in this paper, we take full consideration of the mobility effect and investigate the joint task assignment and resource allocation optimization problem in terms of overall task latency in vehicular fog computing. Specifically, we formulate the joint optimization problem from a Min-Max perspective, which can better choose and coordinate appropriate vehicles as fog nodes to offload computation tasks. Then, we propose a two-stage scheme to efficiently solve the formulated non-convex problem. In addition, considering the relatively stable moving patterns of a vehicle in a short period, we further introduce the mobility prediction into the proposed scheme to obtain a better task assignment and resource allocation solution. Simulation results verify the efficiency of our proposed mobility prediction-based scheme in reducing the overall task

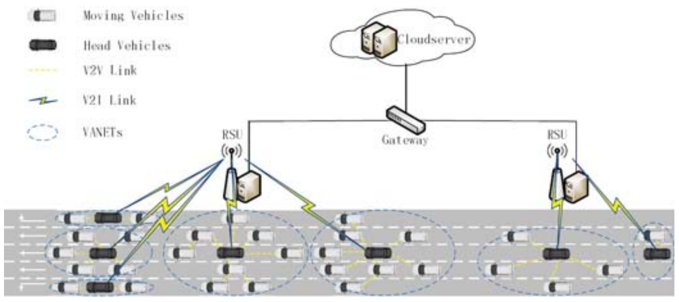


Fig. 1: VFC architecture combined with VANETs

completion latency in VFC.

The remainder of this paper is organized as follows. Section II describes the investigated system model. In Section III, the latency minimization problem is formulated by utilizing the proposed basic frameworks and mobility prediction model. The solution to the nonconvex problem is presented in section IV. The numerical results and conclusions are drawn in Section V and Section VI, respectively.

II. System Model

A. VFC framework

Vehicles are considered as both providers and users of fog service in VFC framework [5], that means a vehicle can request services when it cannot process task by itself, and can also act as a fog node to provide communication and computational resources to other vehicles when it is idle. The former is defined as client vehicle and the latter as fog vehicle. Fig. 1 illustrates the system framework of Vehicular Fog Computing and it consists of the following entities:

- **RSUs:** Roadside Units (RSUs) are base infrastructures that have wide communication ranges and powerful computing ability to form them the coordinators of the VFC networks. RSUs execute TARA algorithm according to information such as task size and complexity, velocity, network topology, utilization of vehicles' resources, etc.
- **Moving vehicles:** Vehicles in the VFC framework can flexibly change their identities between fog service requestor and provider, so every vehicle is assumed to carry on an on-board computer for tasks processing, and onboard dedicated short-range communication (DSRC) to ensure the ability to communicate with other vehicles and RSUs. The vehicles which need computation offloading are defined as client vehicles, and the vehicles work as fog nodes are defined as fog vehicles.
- **Gateway:** Gateway can realize the connection of RSUs and cloud servers, and realize the interconnection of RSUs to ensure the sharing of some information such as vehicles' locations.
- **Head vehicles:** Head vehicles are responsible for coordinating communication between RSUs and other vehicles.

Due to the vehicle mobility, it is unrealistic and inefficient to let all vehicles directly connect to RSUs, so we introduce VANETs to the VFC framework. Vehicles are connected together to form a Vehicular ad-hoc network (VANET) according to speed, direction, distance, and some other factors. Each VANET has a head vehicle responsible for coordinating intra-group V2V communications, i.e., head vehicle is responsible for maintaining a VANET. Only the head vehicle needs to be connected to the RSU through V2I communication while other vehicles in the same VANET connect with each other by V2V communication. Vehicles are interconnected to form a stable combination, even if vehicles are entering/leaving VANETs continuously. This network topology is the physical guarantee of low latency.

B. Description of the TARA optimization

Based on the VFC framework above, we introduce the task assignment and resource allocation problem. In a VANET, when a vehicle generates a task hard to process by itself, it will send task offloading request to the RSU via the head vehicle and the RSU will do the TARA optimization to get an optimal TARA strategy that can minimize the overall latency, and then the RSU will send the strategy back to client vehicles for practical application.

Assuming that, there are N vehicles in a VANET, and the generated task is partitioned into N parts, every vehicle is assigned a sub-task. Because each sub-task has a different file size and computation size, the task assignment strategy and resource allocation strategy should be jointly optimized to minimize the latency.

III. Problem Formulation of Latency Optimization Based on Mobility Prediction

In this section, we formulate the latency minimization problem by introducing mobility prediction.

The objective is the overall time delay including communication delay and computing delay, our task is to find an optimal task assignment and resource allocation scheme to achieve the minimum total time delay. The number of vehicles in a VANET is denoted as N, denote the set of vehicles as $\mathcal{N}$, so we have the definition of tasks Γ and fog vehicles V, let us denote $\Gamma = [\Gamma_1, \Gamma_2, ..., \Gamma_N]^T$ and $\mathrm{V} = [V_1, V_2, ..., V_N]^T$. Some basic models and assumptions are given in the following sections.

A. Task Formulation, Partitioning, and Assignment

Tasks generated by vehicles vary according to their computation size Q, file size Δ, and result size $\hat{\delta}$. The computation size Q is not positively related to file size Δ, for example, considering the same picture, the computation size of basic image processing such as image binaryzation is far less than some complex algorithm, for instance, object detection. In addition, the result size $\hat{\Delta}$ is also not positively related to the original file size Δ. Therefore, we independently generate these variable values in simulations conducted in section IV. Since the computation tasks generated by vehicles are varied, the partition of the task is a very complex process with multiple factors, so we cannot simply partition it equally. As some sub-tasks cannot be partitioned anymore, so it's unrealistic to partition a task into N equal parts. For simplification, we

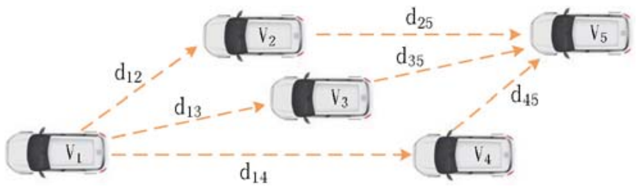


Fig. 2: Communication link routing

define that the task is partitioned into N parts, and they will be assigned to N vehicles, namely, one to one matching. Each part may differ in the size of file and computation, so the client vehicle's sub-tasks computation size set is given as

$$\mathrm{q} = [q_1, q_2, ..., q_N]^T, \sum_{n \in \mathcal{N}} q_n = Q.$$

Similarly, the sub-tasks file size set and results size set are given as

$$\delta = [\delta_1, \delta_2, ..., \delta_N]^T, \sum_{n \in \mathcal{N}} \delta_n = \Delta.$$

$$\hat{\delta} = [\hat{\delta}_1, \hat{\delta}_2, ..., \hat{\delta}_N]^T, \sum_{n \in \mathcal{N}} \hat{\delta}_n = \hat{\Delta}.$$

Therefore, the original task Γ_n is now partitioned into $\Gamma_n = [\Gamma_{n1}, \Gamma_{n2}, ..., \Gamma_{nN}]^T = [(q_{n1}, \delta_{n1}, \hat{\delta}_{n1}), (q_{n2}, \delta_{n2}, \hat{\delta}_{n2}), ..., (q_{nN}, \delta_{nN}, \hat{\delta}_{nN})]^T, n \in \mathcal{N}$.

The task assignment strategy is defined as a $N \times N$ matrix H, whose elements are binary distribution. Each row of the matrix represents the corresponding vehicle's strategy, and it means the corresponding sub-task is offloaded to this vehicle or not. Since it is one to one matching, there can only be one element equals to 1 in each row. The client vehicle's task assignment result S is given as

$$S = \mathrm{H}_{N \times N} \cdot \mathrm{V}_{N \times 1}. \tag{1}$$

B. Communication Link Routing

In our system model, multi-hop communication is adopted to expand the range of VFC service. When the target fog vehicle exceeds the client vehicle's communication radius, multi-hop communication will be triggered. As showed in Fig. 2, there could be multiple communication links between two vehicles in general. We define $\mathrm{r} = [r_1, r_2, ..., r_N]^T$ as the routing strategy, and r_n is set to be 0 when the sub-task doesn't need multi-hop communication. Each r_n is calculated by

$$r^* = \arg\min_r \left(\frac{\delta}{e_{ab} B \log_2(1 + \gamma_{ar})} + \frac{\delta}{e_{ab} B \log_2(1 + \gamma_{rb})} \right), \tag{2}$$

where r stands for the label number of relay vehicles, a is the label number of the client vehicle and b refers to the label number of the fog vehicle. e_{ab} is the bandwidth percentage allocated to the link from a to b, and we assume that the first hop and the second hop has the same bandwidth since the V2V communication is defined as half-duplex Communication.

While the allocated percentage of bandwidth is unknown before resource allocation, so the communication link routing is a sub-optimization problem integrated to the overall optimization problem. According to the actual situation and the purpose of simplifying the problem, we assume the maximum number of hops is 2.

C. Communication Model

The max transmission rate C_{ab} of V2V channel between vehicle a and b associated with Shanon formula can be expressed as

$$C_{ab} = W_{ab} \cdot \log_2(1 + \gamma_{ab}), \tag{3}$$

where W_{ab} denotes the bandwidth of the channel. γ_{ab} is the received signal to noise ratio (SNR), and it is given by [9]

$$\gamma_{ab} = \frac{p d_{ab}^{-\alpha} h_{ab}^2}{N_0}, \tag{4}$$

where p denotes the transmission power, d_{ab} is the transmission distance from a to b, h_{ab} denotes the Rayleigh channel coefficient following a complex Gaussian distribution. N_0 is the power of additive white Gaussian noise (AWGN). α is the path-loss exponent.

Hence, the transmission time delay of client vehicle offloading sub-tasks to other fog vehicles can be obtained as

$$T_{ab} = \frac{\delta_{ab}}{C_{ab}}. \tag{5}$$

D. Computing Model

Vehicles vary according to their remaining computing capability *P*, while the computing capability of the same vehicle also varies at different times depending on its current task load. As a common sense, the more powerful the computing capability is, the less computation time will be spent for the same task. Therefore, the computation time T_c is given by

$$T_c = \frac{Q}{P}. \tag{6}$$

We use MIPS, short for Million Instructions Per Second, to measure the computing capacity, accordingly, MI, short for Million Instructions, is used to measure task computation size. Fog vehicle's computing capability is defined as $P = [\beta_1, \beta_2, ..., \beta_n]^T$, so each fog vehicle's computation time is given by

$$T_{cn} = \frac{q_n}{\beta_n}, n \in \mathcal{N}. \tag{7}$$

E. Vehicle Mobility Prediction

High mobility challenges low latency and real-time performance of VFC, to solve this issue, we introduce the vehicle mobility prediction. As shown in Fig. 3, the VANET's topology can change a lot after each fog vehicle completes the computation, while that time's topology will affect the results' backhaul. We can get the topology of that time by adopting Vehicle Mobility Prediction. By doing so, we can predict the network condition information at that time to promote optimal strategy.

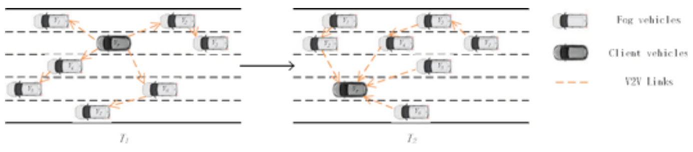


Fig. 3: Vehicle mobility prediction

We need to assess the time from sub-tasks' dissemination to computation completed, which is the key factor for prediction. The time is composed of dissemination part tp_d and computation part tp_c, given as

$$\begin{aligned} tp &= \theta\left(tp_d + tp_c\right), \\ &= \theta\left(\frac{\delta_s}{B\log_2(1+\frac{pd_a^{-\alpha}h^2}{N_0})} + \frac{Q_s}{P_s}\right), \end{aligned} \tag{8}$$

where θ is a constant coefficient, δ_s is the sum of file size, B is the total bandwidth, d_a represents the average distance, Q_s and P_s are the sum of computation size and computing capacity, respectively. This formula is used to measure the approximate completion time of each task.

The mobility model we used is the model of motion with constant acceleration. With the prediction time, combined with the mobility model, we can get the network topology after tp seconds as shown in Fig. 3.

F. Formulation of the latency minimization problem based on TARA optimization

In this section, we formulate the latency minimization problem based on TARA optimization by utilizing the models and assumptions proposed above.

The communication link for transmitting nth sub-task is defined as $l_n^{send/rec}$. We define $E(\cdot)$ as the bandwidth allocated percentage vector, and $e(l_n)$ indicates the bandwidth percentage that allocated to the l_n link, $E(\mathscr{L})$ indicates the overall bandwidth allocation vector, $E(\mathscr{L}) = [e(l_1), e(l_2), ..., e(l_N)]^T$. And it meets the following constraint:

$$\sum_{n\in\mathcal{N}} e(l_n) = 1, e(l_{nc}) = 0, \tag{9}$$

where nc is the label number of the client vehicle, and $e(l_{nc}) = 0$ means the percentage of bandwidth that allocated to the client vehicle is 0, because local processing don't need transmissions.

With the bandwidth allocation vector and aforementioned models, we can formulate the total time delay T_n of each sub-task. T_n consists of three parts: transmission delay of sending data, computation time and transmission delay of sending back results, and it is given as

$$T_n = T_n^{send} + T_n^{comp} + T_n^{rec}, n \in \mathcal{N}, \tag{10}$$

with

$$T_n^{comp} = \frac{Q_n}{\mathrm{H}_n \cdot P}, n \in \mathcal{N}, \tag{11}$$

$$T_n^{send} = \frac{\delta_n}{C(l_n^{send})}, n \in \mathcal{N}, \tag{12}$$

$$C(l_n^{send}) = \begin{cases} e(l_n^{send})B\log_2\left(1+\dfrac{pd_{S(n)}^{-\alpha}h^2}{N_0}\right), & r_n = 0, \\ 1/\left(1/C(l_{ncr}) + 1/C(l_{rs})\right), & r_n \neq 0, \end{cases} \tag{13}$$

where $C(l_{ncr}) = e(l_n^{send})B\log_2\left(1+\frac{pd_{(nc,r_n)}^{-\alpha}h^2}{N_0})\right)$, and $C(l_{rs}) = e(l_n^{send})B\log_2\left(1+\frac{pd_{(r_n,s(n))}^{-\alpha}h^2}{N_0}\right)$. H_n is the nth row of H matrix, d is the vector stored distances from client vehicle to fog vehicles. $\gamma_{r_1^*}$ and $\gamma_{r_2^*}$ are the SNR of the first hop and second hop of the chosen link, respectively. $d_{(nc,r_n)}$ indicates the distance from V_{nc} to V_{r_n}, so the same of $d_{(r_n,s(n))}$. The expression of T_n^{rec} is similar to T_n^{send}, so we don't write it in detail.

Then the optimization problem is formulated as

$$\xi 1: \min_{\mathcal{A}} \max_{n\in\mathcal{N}} T_n$$

s.t.

$$\begin{aligned} & r_n \text{ s.t. } (2) \\ & \sum_{n\in N} e(l_n^{send/rec}) = 1, \\ & e(l_{nc}^{send/rec}) = 0, \\ & \|\mathrm{H}_n\|_1 = \|\mathrm{H}_m\|_1 = 1, h_{nm} \in \{0,1\}, \forall n,m \in \mathcal{N}. \end{aligned} \tag{14}$$

Where $\mathcal{A} \triangleq \left\{\mathrm{H}, E(\mathscr{L}^{send}), E(\mathscr{L}^{rec}), \mathrm{r}^{send}, \mathrm{r}^{rec}\right\}$ is the set of optimization variables, r^{send} and r^{rec} are the set of relay vehicles label numbers, respectively. H_n and H_m are the nth row and mth column of matrix H, respectively.

IV. Solution of The Latency Minimization Problem

Since the original problem ($\xi 1$) is nonconvex and hard to solve, we try to decompose it into several subproblems and solve them. We deploy a two-stage solution for this problem.

A. Stage 1: Solution of Task Assignment Subproblem

By constructing each sub-task's preference list based on predicted mobility information, we formulate the task assignment sub-problem as a matching problem. The preference of $Task_n$ towards V_m is calculated by

$$\begin{aligned} p_{n,m} =& \frac{T_{send}}{T_{sum}} \cdot \mathrm{N}\left(\frac{d_m}{\delta_n}\right) + \frac{T_{comp}}{T_{sum}} \cdot \mathrm{N}\left(\frac{Q_n}{\beta_m}\right) + \\ & \frac{T_{rec}}{T_{sum}} \cdot \mathrm{N}\left(\frac{dt_m}{\hat{\delta}_n}\right), \end{aligned} \tag{15}$$

Where $\mathrm{N}(\cdot)$ represents the normalized process to uniform dimension. T_{sum} is the assessed approximate completion time of all the three parts.

Because the total time delay is determined by δ, $\hat{\delta}$ and Q, so we construct a new variable O to represent the complexity of sub-tasks. O_n is given as

$$O_n = \frac{T_{send}}{T_{sum}} \cdot \mathrm{N}\left(\delta_n\right) + \frac{T_{comp}}{T_{sum}} \cdot \mathrm{N}\left(Q_n\right) + \frac{T_{rec}}{T_{sum}} \cdot \mathrm{N}\left(\hat{\delta}_n\right). \tag{16}$$

After the above construction, we can assign tasks one by one according to task complexity and corresponding preference list, and easily get the task assignment result by executing Algorithm 1.

Algorithm 1 Solution of Task Assignment subproblem

```
Input: Computing capacity: $P$, Computation size: $Q$, File size: $\Delta$, results size: $\hat{\Delta}$
 1: Initialize: vehicle position $\Phi$, velocity: v, acceleration: a
Output: Task Assignment strategy: $TA$
 2: initial $\forall n, TA(n) = 0$;
 3: Compute complexity $O$ of sub-task according to (16);
 4: repeat
 5:     Find the most complex task $n^*$, $n^* = \arg\min_n O(n)$;
 6:     Compute $n^*$ sub-task's preference list $p_n^{list}$ according to (15);
 7:     for $i = 1$ to $n$ do
 8:         $i^* = \arg\min_i p_i^{list}$;
 9:         if $\exists n \in N, TA(n) = i^*$ then
10:             Delete $p_{i^*}^{list}$ from $p^{list}$
11:         else
12:             $TA(n^*) = i^*$
13:         end if
14:     end for
15:     Delete $O(n^*)$ from $O$;
16: until $\forall n \in \mathcal{N}, TA(n) \neq 0$
```

B. Stage 2: Solution of the Resource Allocation Subproblem

After getting the strategy TA, we could solve the routing sub-problem according to (2) by setting $e_{ab} = B/N$, where B is the overall bandwidth. After that, the original optimization problem is transformed into

$$\xi 2 : \min_{E^{send}, E^{rec}} \max_{n \in \mathcal{N}} T_n$$

s.t.

$$\begin{aligned} &\sum_{n \in \mathcal{N}} E_n^{send} = 1, \\ &\sum_{n \in \mathcal{N}} E_n^{rec} = 1, \\ &E_n^{send} > 0, n \in \mathcal{N}, \\ &E_n^{rec} > 0, n \in \mathcal{N}. \end{aligned} \tag{17}$$

The problem ($\xi 2$) is still nonconvex due to the min-max formulation. This is easy to handle by introducing a slack variable ζ , and let $\max_{n \in \mathcal{N}} T_n = \zeta$. Then the problem ($\xi 2$) is transformed as

$$\xi 3 : \min_{E^{send}, E^{rec}, \zeta} \zeta$$

s.t.

$$\begin{aligned} &T_n \leq \zeta, \\ &\sum_{n \in N} E_n^{send} = 1, \\ &\sum_{n \in N} E_n^{rec} = 1, \\ &E_n^{send} > 0, n \in \mathcal{N}, \\ &E_n^{rec} > 0, n \in \mathcal{N}. \end{aligned} \tag{18}$$

Now, the problem ($\xi 2$) has been transformed into a standard convex optimization problem ($\xi 3$), which can be solved in polynominal time using standard CVX tools such as SeDuMi. After that, the TARA strategy is completely generated, and RSU will send it back to the client vehicle.

V. Simulation Results

In order to evaluate the performance of the proposed latency optimization scheme, we conduct the simulations in comparison with other schemes in the urban vehicular scenario. The mobility of vehicles doesn't change dramatically usually, so we deploy the same motion model in the simulation as used in mobility prediction. The simulation parameters are showed in Table I.

TABLE I: Simulation Parameters

Parameters	**Value**
Number of Vehicles	5-25
Transmission power of vehicles	30dBm
Noise power	-114dBm
Bandwidth	30 MHz
Path loss exponent α	3
Velocity	3-25 m/s
Acceleration	-3-3 $\mathrm{m/s^2}$
Radius of vehicles' communication coverage	200 m
FNG coverage	400 m
Computation size	100-500 MI
Computing capacity of each vehicle	10-25 MIPS
Data size of tasks	100-500 Mb
Data size of results	100-500 Mb

The schemes are described as follows:

1) Locally processing: As the name suggests, tasks are not offloaded to other fog vehicles, but are handled locally.

2) Random Task Assignment: The task assignment matrix is generated randomly, so the results can be either good or bad.

3) Task Assignment Only based on Computation size: This scheme means that the larger the computation size of the sub-task is, the more powerful the computing capacity of fog vehicle allocated to the sub-task is.

4) Task Assignment Without Mobility Prediction: The only difference of this scheme from the proposed one is that it does not use mobility prediction information to assign tasks, but use the information the RSU get before executing Algorithm 1, just as shown in the left figure of Fig. 3.

All the above schemes except locally processing use the same resource allocation solution as the proposed scheme.

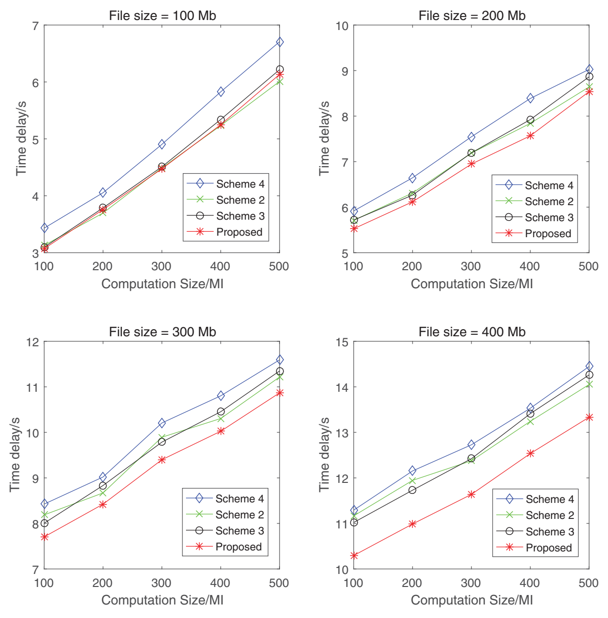
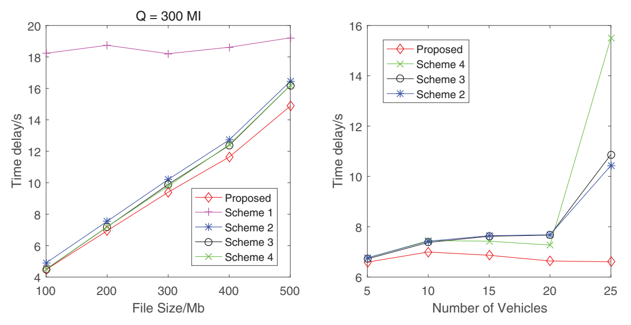


Fig. 4: Time delay of four schemes



(a) (b)

Fig. 5: Time delay versus file size and number of vehicles

In Fig. 4, we compare the latency performance of the proposed scheme with the other three schemes. In each subfigure, the data size (both tasks size and results size) is fixed, and the computation size is set from 100MI to 500 MI. Because position, velocity and most of the other parameters are all randomly generated, so each latency result is obtained by averaging 100 times of simulation, this can truly reveal the actual performance of each scheme. From Fig. 4, we can find that compared with the other three schemes, the proposed scheme has no significant performance improvement when the file size is small, but when the file size increases, the performance improves greatly. For scheme 4, although it is not much different from the proposed scheme, but because it utilizes the original position information instead of predicted information, it is counterproductive on the latency performance since the network topology changes rapidly, that's the reason why scheme 4 is the worst in the simulation.

In Fig. 5(a), when the computation size is fixed, the performance of the proposed scheme becomes more significant with the increase of file size.

In Fig. 5(b), we show the effect of the number of vehicles on the latency performance. With the total bandwidth fixed, the increase of the number of vehicles reduces the bandwidth allocated to each link, and the proposed scheme can better provision resources to maximize performance compared with other three schemes.

VI. Conclusions

In this paper, we proposed a scheme for task assignment and resource allocation optimization in VFC based on mobility prediction. By integrating predicted mobility information into the latency minimization problem, we formulated the joint optimization to realize the optimal task assignment and resource allocation strategy. We employed a two-stage solution to solve the problem, simulation results verified that compared with other schemes, our proposed strategy based on mobility prediction can effectively reduce the delay, which is a key factor in VFC.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grants 61936014 and 61901302, and in part by the National Key Research and Development Project under Grant 2019YFB2102300, 2019YFB2102301, and 2017YFE0119300. This work was also supported in part by the open research fund of National Mobile Communications Research Laboratory, Southeast University (No. 2020D01), in part by the Scientific Research Project of Shanghai Science and Technology Committee under Grant 19511103302, and in part by the National Science Foundation under Grant CPS-1932413 and ECCS-1935915.

References

[1] H. A. Khattak, S. U. Islam, I. U. Din, and M. Guizani, "Integrating fog computing with vanets: A consumer perspective," *IEEE Communications Standards Magazine*, vol. 3, no. 1, pp. 19–25, Mar. 2019.

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

[3] S. S. Shah, M. Ali, A. W. Malik, M. A. Khan, and S. D. Ravana, "vfog: A vehicle-assisted computing framework for delay-sensitive applications in smart cities," *IEEE Access*, vol. 7, pp. 34 900–34 909, 2019.

[4] M. Sookhak *et al.*, "Fog vehicular computing: Augmentation of fog computing using vehicular cloud computing," *IEEE Vehicular Technology Magazine*, vol. 12, no. 3, pp. 55–64, Sep. 2017.

[5] X. Hou *et al.*, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[6] Y. Wu, J. Wu, G. Zhou, and L. Chen, "A direction-based vehicular network model in vehicular fog computing," in *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, Oct. 2018, pp. 585–589.

[7] C. Zhu *et al.*, "Folo: Latency and quality optimized task allocation in vehicular fog computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4150–4161, Jun. 2019.

[8] J. Li *et al.*, "Service migration in fog computing enabled cellular networks to support real-time vehicular communications," *IEEE Access*, vol. 7, pp. 13 704–13 714, 2019.

[9] Z. Zhou *et al.*, "Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3113–3125, Apr. 2019.