Mobility-Aware Task Parallel Offloading for Vehicle Fog Computing

Jindou Xie, Yunjian Jia(✉), Zhengchuan Chen, and Liang Liang

School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400044, People's Republic of China
{xjd,yunjian,czc,liangliang}@cqu.edu.cn

Abstract. When applying fog computing paradigm into Internet of vehicle, vehicles are regarded as intelligent devices with computation and communication capability. These moving intelligent devices are often employed to assist various computation-intensive task offloading in vehicle fog computing, which brings real-time responses. However, vehicles mobility and network dynamics make it challenging to offload tasks to ideal target nodes for user-vehicle. In this paper, leveraging the result of vehicle mobility-awareness, we investigate the task offloading problem in vehicle fog computing aiming to minimize service time. Specifically, we consider that a task can be decomposed into subtasks in any proportion and offloaded from user-vehicle to multi service vehicles in parallel via vehicle-to-vehicle (V2V) links. Mobility information of vehicles collected by RSU is modeled to predicted the states of V2V links based on hidden Markov model (HMM). Then, we refine a rule to select target service-vehicles and the size of each subtask according to predicted results. Comparing with random and single-point task offloading, the proposed approach indicates a better performance on amount of finished task and service time in vehicle dense area.

Keywords: Mobility · Task offloading · HMM · Vehicle fog computing · Parallel offloading

1 Introduction

With the development of wireless multimedia service, increasing more mobile applications such as immersive gaming, video streaming and human-computer interaction services demand intensive computation resource for real-time processing and high network bandwidth for data exchange [1]. Fog computing also known as mobile edge computing [2], is regarded as a promising solution for

Supported by the National Key Research and Development Program of China (Grant No. 2016YFE0121100); and the Program for Innovation Team Building at colleges and Universities in Chongqing, China (Grant No. CXTDX201601006); and National Natural Science Foundation of China(Grant No. 61601067).

S. Han et al. (Eds.): AICON 2019, LNICST 287, pp. 367–379, 2019.
https://doi.org/10.1007/978-3-030-22971-9_32

those challenges by sinking cloud function to edge nodes [3]. To this end, various service tasks generated in user side can be offloaded to target edge servers, taking advantage of sufficient resources on edge servers when mobile users suffer from limited device resource [4].

The development of autonomous driving technology illustrates that the future vehicles are equipped with advanced computational resources to facilitate sophisticated artificial intelligence (AI) based on multi-dimension data collected by on-board-unit [5]. The idea of vehicle-as-a-resource has been proposed, which exploits the enormous resources of vehicular network to facilitate new types of services, such as mobile crowd sensing [6] and cooperative caching [7]. Thus it is a good way to share resource that applying fog computing in vehicle-based settings, to form vehicle fog computing [8]. In vehicle fog computing system, the role of vehicle can be a user or a edge server that is named as user-vehicle (User-V) and service-vehicle (Service-V) respectively in this work. Task offloaded from User-V to Service-V relies on successful transmission via vehicle-to-everything (V2X), including vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) over different protocols [9].

In the literature, there are some recent efforts dedicating to task offloading in vehicle fog computing. In [10], the authors pointed to exploiting parked cars for task offloading and cooperative sensing. On the contrary, since parked cars can be regarded as static cloud servers, the moving vehicles produce more opportunities to share resources but also high dynamics of vehicle network, which increase the complexity of offloading decision. Some researches focus on utilizing vehicles on the move endowed with ample computation resources. The authors in [8] extended the fog computing paradigm to conventional vehicular networks, and presented a viewpoint of vehicles as infrastructures, discussing the structure in four types of application scenarios. Authors in [11] concentrated on various types tasks offloading in mobile-edge cloud-based vehicular networks. And optimal strategies for mobile edge server selection and task transmission management are proposed. However, they all consider single service node provides resources for computation-intensive task.

In this paper, we consider multi moving vehicles provide service for user-vehicle in parallel processing different parts of one task to meet low-latency demand. To improve the offloading efficiency, we estimate the states of V2V links based hidden Markov model (HMM) modeling the mobility information collected by RSU. After that, we refine a task parallel offloading scheme considering transmission delay, computation delay, as well as task decomposition and handover cost.

The rest of the paper is organized as follows. Section 2 introduces the system scenario; Sect. 3 gives an account of mobility awareness based on V2V links states prediction with HMM. Section 4 analyzes the task parallel offloading policy for service delay reduction. Then simulations are implemented in Sect. 5 and conclusions are provided in Sect. 6.

2 System Overview

As illustrated in Fig. 1, User-V has little computation resource for remained intensive and delay-sensitive task but sufficient communication resource to establish V2I or V2V links. There are several neighbor vehicles with available computation and communication resource in the same RSU coverage area as User-V, such as vehicles A-E in Fig. 1. These vehicles are denoted as $\mathcal{N} = \{1, 2, \cdots n\}$ with different computation capabilities f_n, and regarded as moving intelligent devices. The RSU can collect mobility information from them and control links between vehicles according to offloading decision.

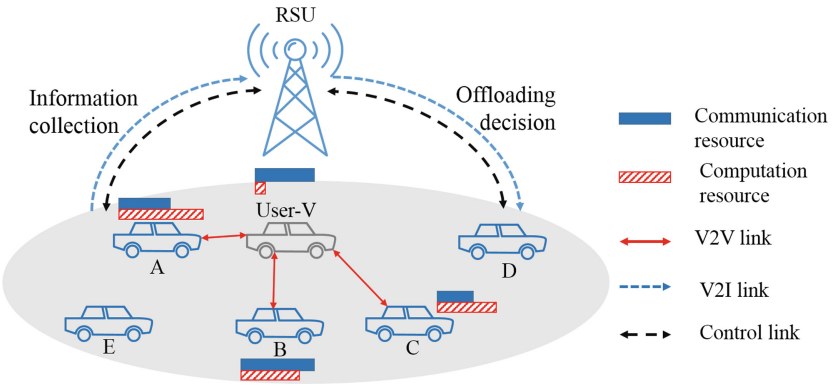

Fig. 1. Illustration of vehicle fog computing system.

We focus on a scenario where User-V with unknown trajectory tends to offload subtasks (parts of task derived from task decomposition in any proportion) to several service vehicles (Service-Vs) for computation resource sharing gain. The objective is to complete more tasks in short service time. At the beginning of offloading process, User-V sends the offloading requests and reports its mobility information to RSU. After received the messages, RSU analyzes all the current and historical mobility information in the coverage area to forecast the states of V2V links between User-V and Service-Vs in the residence time of User-V. Then, RSU selects appropriate vehicles as target edge nodes, formed as set $\boldsymbol{V}_s$ based on available resources. Vehicles in $\boldsymbol{V}_s$ are able to provide service for User-V simultaneously. Afterwards, V2V links between User-V and Service-Vs are established with RSU's assistance and the task is decomposed according to available resources on Service-Vs. User-V transmits different subtasks to the selected nodes with the assistance of RSU. Upon the nodes finishing the corresponding subtasks in parallel, they send the results back to User-V. If task has not been finished by Service-Vs during the time user stayed in the coverage, additional handover cost will be accounted for service time t_f. According to these steps, the task parallel offloading scheme is based on the estimation of

states of V2V links that are dynamic due to vehicle mobility. So we address the challenge of mobility awareness firstly based on HMM.

3 HMM-Based Mobility Awareness

We resort to the tool of HMM to characterize the mobility information of vehicles and predict the states of V2V links, which influence component of $\boldsymbol{V}_s$ and transmission rate. Afterwards, we take into account transmission, computation, as well as task decomposition and handover cost to refine a task parallel offloading approach.

3.1 HMM Description

HMM is a typically dynamic Bayesian Network, and a sequential probability model [12]. It is good at describing a process of observable states sequence generated by a hidden Markov chain. The variables in HMM can be divided into two sets, hidden states and observable states.

- Hidden states sequence: $\{s_1, s_2, \cdots, s_t\}$. $\boldsymbol{S}$ is the state space including S elements, and $s_t \in \boldsymbol{S}$ denotes the t-th state. Besides, the state s_t only relates to the state s_{t-1}, following Markov rules.
- Observable states sequence: $\{o_1, o_2, \cdots, o_t\}$. Observable state space $\boldsymbol{O}$ includes O elements, and $o_t \in \boldsymbol{O}$ indicates the t-th observable state. The values of observable states only depend on the current hidden states value.

An HMM can be defined by three parameters:

- $\mathbf{A}$: transmission matrix $\mathbf{A} = [a_{ij}]_{S\times S}$, where a_{ij} is the transmission probability of taking transmission from state s_i to s_j,

$$a_{ij} = P\left(s_{t+1} = s_j | s_t = s_i\right), 1 \leq i, j \leq S. \tag{1}$$

- $\mathbf{B}$: confusion matrix. According to the model, the probabilities of observable states are obtained from current states values, denoted as $\mathbf{B} = [b_{ij}]_{S\times O}$. b_{ij} indicates the probability of observation o_j when state s_i is at time t,

$$b_{ij} = P\left(o_t = o_j | s_t = s_i\right), 1 \leq i \leq S, 1 \leq j \leq O. \tag{2}$$

- $\boldsymbol{\pi}$: Initial state probability $\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_S)$ indicates the probability of initial state,

$$\pi_i = P\left(s_1 = s_i\right), 1 \leq i \leq S. \tag{3}$$

The HMM can be written in a compact notation $\theta = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$.

3.2 HMM-Based V2V-Link States Prediction

We predict the states of V2V links between User-V and Service-Vs based on the mobility information collected by RSU, i.e. the relative distance across User-V and Service-Vs. V2V links states contain main two variables, the connectivity between User-V and Service-Vs and relative distance between them.

There are many uncertain impact factors in state of connectivity except relative distance because of sophisticated network environment, such as Service-V authorization and transmission obstacle. Therefore, we apply HMM to estimate the fuzzy relationship between relative distance and connectivity according to historical mobility information, and to predict both of them in the following period, which helps amount of available resource estimation and transmission delay prediction. In the process, the residence time of User-V in the coverage area is divided into T isometric slots, and the length of each slot is T_0. The hidden states value of Service-Vs is $s_t = \{1, 0\}, 0 < t \leq T$, describing whether the vehicle could communicate with User-V via V2V links in the slot t. The observable values in slot t are one of O levels of relative distance noted as $o_t \in \boldsymbol{O} = \{o_1, o_2, \cdots o_O\}$, as shown in Fig. 2.

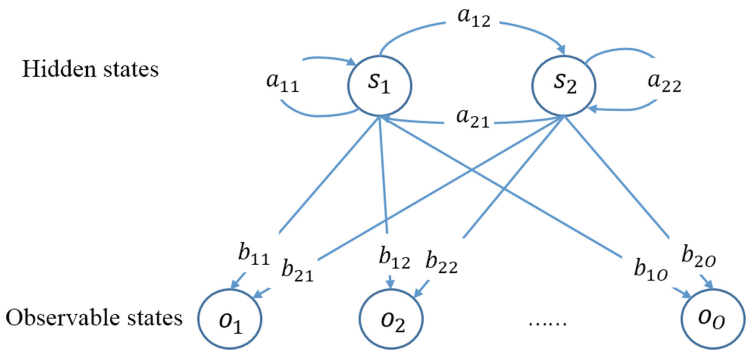

Fig. 2. Framework of the HMM with hidden states $\{s_1 = 0, s_2 = 1\}$, observable values include O levels of relative distance.

In this way, we can learn the transmission matrix $\mathbf{A}$, confusion matrix $\mathbf{B}$, initial state probability $\boldsymbol{\pi}$ from the historical relative distance information in the coverage area using Baum-Welch algorithm in HMM. Then, we can find the most possible hidden states value and observable states value of Service-Vs in every slot based on the derived HMM model parameters $\theta = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ through forward algorithm. To this end, it is easy to know the transmission rate according to the sequence of relative distance (observable states) $\boldsymbol{O}$. Then we can choose appropriate vehicles as edge servers who can communicate with User-V via V2V links and offer abundant communication resource (high transmission rate). A solution scheme of HMM model based on the Baum-Welch algorithm and forward algorithm is represented as follows.

Baum-Welch algorithm starts from initial mode parameters, adjusting them based on the given sequence of observation to maximize probability of the sequence. Firstly, we definite Q-function $Q(\theta, \bar{\theta})$ based on the log likelihood function of HMM $\log P(\mathbf{O}, \mathbf{S}|\theta)$, where $\mathbf{O}$ is the observable states sequence, $\mathbf{S}$ denotes the hidden states sequence, and $\log P(\mathbf{O}, \mathbf{S}|\theta) = \pi_{s_1} b_{s_1}(o_1) a_{s_1 s_2} b_{s_2}(o_2) \cdots a_{s_{T-1} s_T} b_{s_T}(o_T)$.

$$\begin{aligned} Q(\theta, \bar{\theta}) &= \sum_{S} \log P(\mathbf{O}, \mathbf{S}|\theta) P(\mathbf{O}, \mathbf{S}|\bar{\theta}) \\ &= \sum_{S} \log \pi_{s_1} P(\mathbf{O}, \mathbf{S}|\bar{\theta}) \\ &\quad + \sum_{S} \left(\sum_{t=1}^{T-1} \log a_{s_t s_{t+1}} \right) P(\mathbf{O}, \mathbf{S}|\bar{\theta}) \\ &\quad + \sum_{S} \left(\sum_{t=1}^{T} \log b_{s_t}(o_t) \right) P(\mathbf{O}, \mathbf{S}|\bar{\theta}), \end{aligned} \tag{4}$$

where $\bar{\theta}$ is current estimated mode parameters, and θ is the goal of parameters. We can obtain $\theta = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ by maximizing Q-function. Specifically, we can use Lagrange multiplier subjected to $\sum_i \pi_{s_i} = 1, \sum_j a_{ij} = 1, \sum_i b_{ij} = 1$ to maximize three items [13] in (4) respectively to estimate the goal parameters:

$$\pi_{s_i} = \frac{P(\mathbf{O}, s_1 = s_i|\bar{\theta})}{P(\mathbf{O}, \mathbf{S}|\bar{\theta})}, \tag{5}$$

$$a_{ij} = \frac{\sum_{t=1}^{T-1} P(\mathbf{O}, s_t = s_i, s_{t+1} = s_j|\bar{\theta})}{\sum_{t=1}^{T-1} P(\mathbf{O}, s_t = s_i|\bar{\theta})}, \tag{6}$$

$$b_{ij} = \frac{\sum_{t=1}^{T-1} P(\mathbf{O}, s_t = s_j|\bar{\theta}) L(o_t = o_i)}{\sum_{t=1}^{T-1} P(\mathbf{O}, s_t = s_j|\bar{\theta})}. \tag{7}$$

Given the mode $\theta = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, the likelihood of V2V links connectivity (hidden states) and relative distance (observable states) are calculated with forward algorithm as soon as the User-V steps into the RSU coverage. Forward likelihood means the possibility of a condition where observation sequence is $o_1, o_1, \cdots o_t$, the state value is s_i at slot t in the HMM θ, noted as

$$\alpha_t(s_i) = P(o_1, o_2, \cdots o_t, s_t = s_i|\theta). \tag{8}$$

Possibility of all the possible observation sequence during the residence time will be calculated iteratively while possibilities of hidden states are available from the hidden Markov chain. We choose the most possible hidden state $\boldsymbol{S}$ and observation sequence $\boldsymbol{O}$ as the predicted results to help target Service-Vs selection.

4 Proposed Task Parallel Offloading Scheme

Based on the results of connectivity and relative distance prediction with HMM, we can start task parallel offloading aiming to reduce service time which is defined as follows.

4.1 Service Time Model

In the procedure of task parallel offloading, the service time introduces task transmission delay, computation delay, as well as task decomposition and handover cost.

- **Transmission Delay via V2V Link**
  The task data is transmitted from User-V to the Service-Vs through the wireless V2V channel, let $H_n(o_n^t)$ describe the channel gain between the User-V and Service-V n at slot t. Given the transmission power of User-V P_{tx}, the maximum achievable transmission rate is given by:

$$r(n,t) = W\log_2\left(1 + \frac{H_n(o_n^t)P_{\text{tx}}}{\sigma^2 + I_n}\right), \tag{9}$$

  where W is the channel bandwidth, σ^2 is the noise power and I_n is the intra-cell interference power. Then, λ_t is the subtask size. The transmission delay for sending the subtask to selected node n is

$$d_t(n,t) = \frac{\lambda_t}{r(n,t)}. \tag{10}$$

  In addition, result-return transmission delay, packet loss and control data transmission are not considered in this work. And the following analysis and the proposed solutions are still applicable with the consideration.
- **Computation Delay**
  In slot t, we employ computation capabilities f_t^n to describe the CPU frequency of Service-V n. If Service-V n is allocated to compute a subtask of size λ_c and computation intensity γ, given the allocated CPU frequency f_t^n, the computation delay is

$$d_c(n,t) = \frac{\lambda_c\gamma}{f_n^t}. \tag{11}$$

  We assume there is no change on Service-Vs computation capabilities during the residence time, recorded as $f_n^t = f_n, t = 1, 2, \cdots, T$.
- **Task Decomposition Cost**
  Because task offloading can be parallelized, task has to be decomposed into several subtasks, which produces decomposition cost related with the number of target Service-Vs. Each additional new target Service-V brings extra decomposition delay c_d. We can cumulate the whole task decomposition cost by

$$C_d = c_d \sum_n z(n), \tag{12}$$

where $z(x)$ is an indicator function with $z(x) = 1$ if event $x \in \boldsymbol{V}_s$ is true and $z(x) = 0$ otherwise. c_d is a constant that expresses decomposition cost brought by every new target Service-V.

– **Handover Cost**
If the task is not completed during the residence time, User-V leaving this coverage area switches to a different RSU. The rest of task has to offload to other vehicles via V2V links assisted by another RSU, which brings handover procedure and task migration cost

$$C_h = c_h(\lambda - \sum_{n \in \boldsymbol{V}_s} \lambda_n), \tag{13}$$

where λ is the size of initial task, λ_n is the size of accomplished by Service-V n, and c_h is a constant, the handover cost per task volume.

Thus, the service time can be derived from $t_f = d_t + d_c + C_d + C_h$.

4.2 Task Parallel Offloading Scheme

Up to now, we can refine an offloading approach for task service time t_f reduction. Connectivity of V2V links between User-V and Service-Vs in every slot is obtained, recorded as

$$\boldsymbol{S}_{n\times T} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1T} \\ s_{21} & s_{22} & \cdots & s_{2T} \\ \vdots & & & \\ s_{n1} & s_{n2} & \cdots & s_{nT} \end{bmatrix}.$$

The value of s_{ij} is 0 or 1. $s_{ij} = 1$ means Service-V i can communicate with User-V in slot j, otherwise there is no connectivity between them. Similarly, we can get the relative distance from User-V to Service-Vs $\boldsymbol{O}_{n\times T}$. Thus, the transmission rate matrix according to (9), denoted as

$$\boldsymbol{r}_{n\times T} = \begin{bmatrix} r_{11} & \cdots & r_{1T} \\ r_{21} & \cdots & r_{2T} \\ \vdots & & \\ r_{n1} & \cdots & r_{nT} \end{bmatrix},$$

where r_{ij} expresses the transmission rate between User-V and Service-V i in$j$-th slot. The available transmission resources can be derived as

$$\boldsymbol{R}^{\text{tr}}_{n\times T} = \boldsymbol{S} \circ \boldsymbol{r} \triangleq \begin{bmatrix} s_{11} \times r_{11} & \cdots & s_{1T} \times r_{1T} \\ \vdots & & \\ s_{n1} \times r_{n1} & \cdots & s_{nT} \times r_{nT} \end{bmatrix}. \tag{14}$$

Meanwhile, Service-Vs are equipped with diversity computation capabilities those keep steady during the residence time, so we write $\boldsymbol{f}_{n\times 1} = (f_1, f_2, \cdots, f_n)^T$.

Algorithm 1. HMM-based task parallel offloading algorithm

Baum-Welch algorithm mode initial $\theta^{(0)} = (\mathbf{A}^{(0)}, \mathbf{B}^{(0)}, \boldsymbol{\pi}^{(0)})$;
iteratively cumulate a_{ij}, b_{ij}, π_i, based on (5), (6) and (7);
up on convergency, get the goal parameters $\theta = \theta^{(n)} = (\mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \boldsymbol{\pi}^{(n)})$;
forward algorithm value initial $\alpha_1(s_i) = \pi_i b_i(o_1)$
for each $n \in \mathcal{N}$ **do**
  for each $t \in [1, T-1]$ **do**
    iteratively compute $\alpha_{t+1}(s_i)$ based on (8);
    $P(\mathbf{O}|\theta) = \sum_i \alpha_T(s_i)$;
    $\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0) \times \mathbf{A}^t$;
  end for
end for
$\boldsymbol{O}_{n\times T} = \arg\max_{\mathbf{O}} P(\mathbf{O}|\theta)$;
$\boldsymbol{S}_{n\times T} = \arg\max_{i} \boldsymbol{\pi}(t)$;
Compute $\boldsymbol{r}_{n\times T}, \boldsymbol{R}^{\mathrm{tr}}_{n\times T}$ based on (9), (14) respectively;
for each $t \in [1, T]$ **do**
  if $\max_n \lambda_s(n,t) > f_u \times T_0$ **then**
    for each $n \in \mathcal{N}$ **do**
      if $\lambda_f < \lambda$ **then**
        compute $\lambda_s(n,t)$ based on (16);
        accumulate λ_f based on (17);
        update target node set $\boldsymbol{V}_s$;
      else
        record the time of traverse stop $T_f = t$;
        break;
      end if
    end for
  end if
end for
if $\lambda_f < \lambda$ **then**
  compute handover cost C_h based on (13);
  considering handover cost, service time $t_f = t + C_h$;
else
  service time $t_f = T_f$;
end if
return service time t_f, amount of finished-task λ_f.

In slot t, if amount of task finished by Service-Vs is less than that of local computation, it is unnecessary to offload for User-V. Considering this condition, we select offloading-slots based on the local computation capability f_u and the maximum available Service-Vs resource in specific slot. The length of one slot is fixed recorded as T_0, so for $\{(n,t)|R^{\mathrm{tr}}(n,t) > 0\}$ we can write

$$T_0 - c_d z(n) = \frac{\lambda_s(n,t)}{R^{\mathrm{tr}}(n,t)} + \frac{\lambda_s(n,t)\gamma}{f_n}. \tag{15}$$

To simplify the cumulation, let $\lambda_t = \lambda_c = \lambda_s$. According to (15), the amount of task finished by Service-V n in slot t can be obtained by

$$\lambda_s(n,t) = \frac{[T_0 - c_d z(n)]\, R^{\mathrm{tr}}(n,t) f_n}{R^{\mathrm{tr}}(n,t)\gamma + f_n}. \tag{16}$$

When we meet a new slot t, we can decide whether to offload in the slot based on $\max \lambda_s(n,t)$ and $f_u T_0$. If $\max\limits_{n} \lambda_s(n,t) > f_u T_0$, subtasks will be offloaded to Service-Vs in this slot; if not, they will be computed by User-V.

Obviously, Eq. (16) presents that greater $R(n,t) = R^{\mathrm{tr}}(n,t) f_n$ lead more finished task by Service-Vs. Thus, we choose target Service-Vs following the declining order list of $R(n,t)$ until T_f when task is finished or User-V leaves the RSU coverage. $R(n,t) = 0$ means that Service-V n has no chance to be picked into the target node set in slot t. So far, we can accumulate the amount of finished task λ_f and total service time t_f through traversing available resources in each slot,

$$\lambda_f = \sum_{t=1}^{T_f} \sum_{n \in \boldsymbol{V}_s} \lambda_s(n,t). \tag{17}$$

If $\lambda_f < \lambda$, C_h should be taken into total service time; otherwise $C_h = 0$,

$$t_f = d_t + d_c + C_d + C_h = T_f + C_h. \tag{18}$$

For clarity, the detailed steps of HMM-based task parallel offloading algorithm (HMM-P) at the beginning of residence time are summarized as Algorithm 1.

5 Simulation Results

In this section, we evaluate the proposed task parallel offloading algorithm by comparing its performances with two algorithms, i.e., random offloading algorithm and HMM-S algorithm. In the random offloading algorithm, User-V selects a Service-V randomly in each slot to offload subtasks. In the HMM-S algorithm, User-V selects only one Service-V in each available slot based on the prediction of V2V links states according to HMM.

RSU coverage area is assumed as a circle with 200 m radius. The vehicles' trajectory are generated randomly. Wireless channel gain is modeled as

$H_n o_n^t = 127 + 30\log({o_n}^t)$. Besides, channel bandwidth $W = 20\,\text{MHz}$, noise power $\sigma^2 = 10^{-8}\,\text{W}$, intra-interface $I_n \propto d_0^{-4}$ (d_0 is the average distance between vehicles and neighbouring small base station.), and transmission power $P_{\text{tx}} = 200\,\text{mW}$. Computation intensity is $\gamma = 3000$ (cycle/bit) and vehicles' CPU frequency are uniformly distributed $f_n \in [1, 2]\,\text{GHz}$. Similarly, the data size of task is $\lambda \in [0.5, 2]\text{Mb}$. Moreover, we set a observable state every 40m, so there is $O = 10$.

The number of vehicles in the same RSU coverage area with User-V is referred to vehicle density n. The resident time of User-V is divided into 8 slots, written as $T = 8$, and length of each slot is $T_0 = 1$ to simplify calculation. We formulate 1000 times for average value of λ_f and t_f.

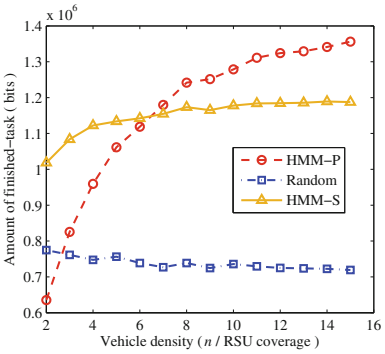

Fig. 3. The amount of finished-task in different vehicle density conditions.

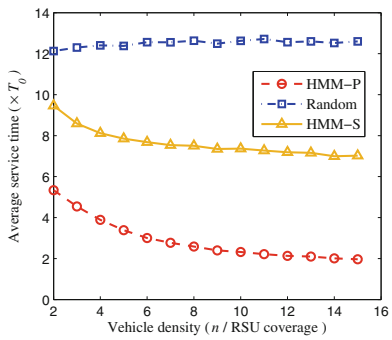

Fig. 4. The service time in different vehicle density conditions.

We discuss amount of task finished by Service-Vs in different vehicle density condition, and more vehicles in the coverage means the area is more dense. Figure 3 compares the amount of task finished by HMM-P, random, and HMM-S. As can be seen, the proposed approach finishes most task than other two methods in vehicle dense area. And the amount of task finished by HMM-S is most when vehicle density is small, while the figure of random algorithm keeps the least in all conditions. The amount of task finished by HMM-P and HMM-S both increase, and the figure for HMM-P takeovers that of HMM-S at $n = 7$. Because in HMM-P there are few vehicles not constrained with inequality $\max_n \lambda_s(n, t) > f_u T_0$, and task is most likely to be computed locally in vehicle sparse area. Meanwhile, service time of three methods in diversity conditions is analysed in Fig. 4 that shows opposite situations compared with Fig. 3. Although the service time of HMM-P and HMM-S both decrease, greater changes occur in that of HMM-P with the increase of vehicle density. Service time of random algorithm remains largest around $12T_0$, while that of HMM-P remains lowest and it is less than $6T_0$. Moreover, the data of HMM-S starts from approximately $10T_0$ and decrease to $8T_0$ at $n = 5$. The simulation results show advantages of

HMM-P in vehicle dense condition since User-V can finished more task in less service time assisted by Service-Vs.

Figure 5 shows the effect of slot granularity on amount of finished task. During the fixed residence time, the more slots, the smaller T_0 is, which means the analysis process is more fine-grained. Amount of finished task with HMM-P maintain steady at most, while that of random algorithm experience a significant increase. And the figure for HMM-S remains steady after increasing dramatically. At the same time, the effect of slot granularity on service time is discussed in Fig. 6. Service time of three schemes are in decrease, but in different degrees. That figure of HMM-P shows a modest decrease from approximately $8T_0$ between $T = 2$ and $T = 15$, while average service time for random algorithm and HMM-S decreased dramatically. We can infer the reason is more fine-grained analysis gives more chances to change Service-V and share resources which increase the possibility of ideal target node finding. This result indicates fine-grained analysis is more suitable for dynamic network.

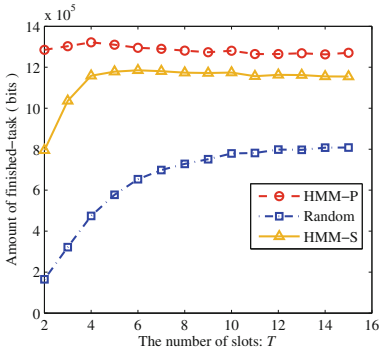

Fig. 5. The effect of slot length on amount of finished-task.

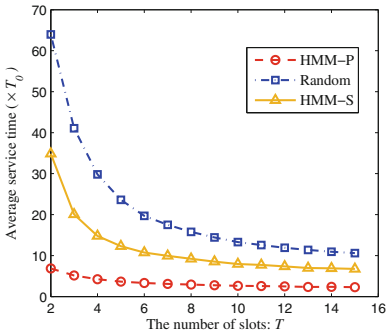

Fig. 6. The effect of slot length on service time.

6 Conclusion

In this paper, the task parallel offloading approach is proposed in vehicle fog computing system through HMM predicting V2V link states between vehicles. We consider the scenario where multi service-vehicles provide resources for User-V in parallel based on task decomposition, and refine an offloading node selection rule for minimizing the task service time. The proposed approach presents good performance on service time and finished-task amount compared with random-selection and HMM-S in vehicle dense area. Besides, the effect of model parameters on results is discussed, and it indicates appropriate value of number of slots can result in better performance.

References

1. Han, S., Xu, S., Meng, W., Li, C.: Dense-device-enabled cooperative networks for efficient and secure transmission. IEEE Netw. **32**(2), 100–106 (2018)
2. Shirazi, S.N., Gouglidis, A., Farshad, A., Hutchison, D.: The extended cloud: review and analysis of mobile edge computing and fog from a security and resilience perspective. IEEE J. Sel. Areas Commun. **35**(11), 2586–2595 (2017)
3. Peng, M., Yan, M., Zhang, K., Wang, C.: Fog-computing-based radio access networks: issues and challenges. IEEE Netw. **30**(4), 46–53 (2016)
4. Zhou, Z., Liao, H., Gu, B., Huq, K.M.S., Mumtaz, S., Rodriguez, J.: Robust mobile crowd sensing: when deep learning meets edge computing. IEEE Netw. **32**(4), 54–60 (2018)
5. Zhou, Z., Gao, C., Xu, C., Zhang, Y., Mumtaz, S., Rodriguez, J.: Social big-data-based content dissemination in Internet of vehicles. IEEE Trans. Ind. Inf. **14**(2), 768–777 (2018)
6. Zhou, Z., et al.: When mobile crowd sensing meets UAV: energy-efficient task assignment and route planning. IEEE Trans. Commun. **66**(11), 5526–5538 (2018)
7. Duan, P., Jia, Y., Liang, L., Rodriguez, J., Huq, K.M.S., Li, G.: Space-reserved cooperative caching in 5G heterogeneous networks for industrial IoT. IEEE Trans. Ind. Inform. **14**(6), 2715–2724 (2018)
8. Hou, X., Li, Y., Chen, M., Wu, D., Jin, D., Chen, S.: Vehicular fog computing: a viewpoint of vehicles as the infrastructures. IEEE Trans. Veh. Tech. **65**(6), 3860–3873 (2016)
9. Zhou, Z., Yu, H., Xu, C., Zhang, Y., Mumtaz, S., Rodriguez, J.: Dependable content distribution in D2D-based cooperative vehicular networks: a big data-integrated coalition game approach. IEEE Trans. Intell. Transp. Syst. **19**(3), 953–964 (2018)
10. Eckhoff, D., Sommer, C., German, R., Dressler, F.: Cooperative awareness at low vehicle densities: how parked cars can help see through buildings. In: Proceedings IEEE GLOBECOM, pp. 1–6, December 2011
11. Zhang, K., Mao, Y., Leng, S., He, Y., Zhang, Y.: Mobile-edge computing for vehicular networks: a promising network paradigm with predictive off-loading. IEEE Veh. Tech. Mag. **12**(2), 36–44 (2017)
12. Yang, J., Xu, Y., Chen, C.S.: Hidden Markov model approach to skill learning and its application to telerobotics. IEEE Trans. Robot. Autom. **10**(5), 621–631 (1994)
13. Qiao, S., Shen, D., Wang, X., Han, N., Zhu, W.: A self-adaptive parameter selection trajectory prediction approach via hidden Markov models. IEEE Trans. Intell. Transp. **16**(1), 284–296 (2015)