Joint Offloading and Resource Allocation in Vehicular Edge Computing and Networks

Yueyue Dai, Du Xu,
Key Laboratory of Optical Fiber Sensing and Communications,
University of Electronic Science and Technology of China
Email:{daiyyue,xudu.uestc}@gmail.com

Sabita Maharjan
Simula Research Laboratory
Norway
Email: sabita@simula.no

Yan Zhang
University of Oslo
Norway
Email: yanzhang@ieee.org

Abstract—The emergence of computation intensive on-vehicle applications poses a significant challenge to provide the required computation capacity and maintain high performance. Vehicular Edge Computing (VEC) is a new computing paradigm with a high potential to improve vehicular services by offloading computation-intensive tasks to the VEC servers. Nevertheless, as the computation resource of each VEC server is limited, offloading may not be efficient if all vehicles select the same VEC server to offload their tasks. To address this problem, in this paper, we propose offloading with resource allocation. We incorporate the communication and computation to derive the task processing delay. We formulate the problem as a system utility maximization problem, and then develop a low-complexity algorithm to jointly optimize offloading decision and resource allocation. Numerical results demonstrate the superior performance of our Joint Optimization of Selection and Computation (JOSC) algorithm compared to state of the art solutions.

I. INTRODUCTION

The advancements in Internet of Things (IoT) and wireless technologies have paved a way towards realizing new applications with advanced features. For instance, on-vehicle cameras and embedded sensors, can play a crucial role towards efficient and safe transportation systems. However, the resource-constrained vehicles can be strained by computation-intensive applications, resulting in bottlenecks and making it challenging for the vehicles to ensure the required level of Quality of Service (QoS) [1]. Mobile Edge Computing (MEC) can alleviate the need of heavy computation from the vehicles, yet enable such applications by providing computation capabilities at the edge of the radio access network and in close proximity to mobile users [1], [2].

Computation offloading is a process where mobile users offload their computation-heavy and latency-sensitive tasks to base stations (BSs) for edge execution [2]. There has been considerable amount of work focusing on computation offloading under MEC where each user independently chooses whether to execute the task locally or to offload the task to edge servers, to minimize energy consumption and/or computation latency [3], [4], [5], [6]. In [3] and [4], the offloading problem was studied with an objective to minimize the weighted energy consumption under the constraint on computation latency. The authors in [5] proposed a multi-user MEC system by integrating a multi-antenna Access Point (AP) with an MEC server by jointly optimizing offloading and computing. In [6], a delay-optimal computation offloading algorithm was proposed by jointly considering the offloading decision, the CPU-cycle frequencies, and the transmit power. In general, these schemes consider there is only one MEC server located at BS and all mobile users have to offload their tasks to the only one MEC server to execute computation. With the limitation of the amount of MEC servers, some users' tasks may not be accomplished within the permissible latency threshold.

Vehicular Edge Computing (VEC) is a promising new paradigm that has received much attention lately, as it can extend the computation capability to vehicular network [7]. With the advent of VEC, lightweight but ubiquitous edge resources deployed at nearby Road Side Units (RSUs) can offer high QoS. Further, localized processing is enabled to save backhaul bandwidth and awareness about location is also beneficial to resource allocation.

A few studies investigated computation offloading for vehicle networks. In [8], the authors considered that a vehicular network consisted of multiple VEC servers and each vehicle selected one of them as its target offloading server to maximize the utilities of both the vehicles and the computing servers. The authors in [9] proposed a similar computation offloading strategy. However, these schemes may result in a severe overload as all vehicles greedily select the VEC server with the highest utility. It further leads to low offloading efficiency and prolongs task processing delay. To overcome this shortcoming, we propose an offloading scheme by balancing the load among VEC server, at the same time, to maximize system utility. In addition, as the mobility of vehicles also has a significant impact on task processing delay, we also incorporate the mobility aspect into problem formulation.

We propose integrating offloading and resource allocation in order to carefully address the following critical challenges: 1) VEC server selection for offloading, 2) determining optimal computation resource to maximize system utility. To address these challenges, we consider a VEC network that multiple VEC servers equipped on RSUs locate along the road. Vehicles can select a VEC server to offload their tasks for satisfying stringent latency requirements. We model VEC server selection as a binary decision. By jointly optimizing selection offloading decision and computation resource allocation, we obtain the optimal strategy for maximizing system utility. The

key contributions of our work are as follows:

- We jointly model VEC server selection for offloading, and resource allocation to execute tasks associated with vehicular applications.
- We formulate the joint offloading and resource allocation problem as a system utility maximization problem under the latency constraint.
- We propose a Joint Optimization of Selection and Computation (JOSC) algorithm to find the solution of the optimization problem in a distributed manner and with less overhead.

The rest of this paper is organized as follows. In Section II, we introduce the system model. In Section III, we formulate the joint offloading and resource allocation problem and propose a low-complexity JOSC algorithm to solve this problem. We evaluate the performance of the JOSC algorithm and provide illustrative results in Section IV. We conclude the paper in Section V.

II. SYSTEM MODEL

We consider a unidirectional road, where M RSUs are located along the road, as shown in Fig. 1. The road is correspondingly divided into M segments, with length $\{L_1, L_2, .., L_M\}$ respectively. Each RSU is equipped with a VEC server, whose computation resources are limited. We denote the id set of these RSUs as $\mathcal{M} = \{1, ..., M\}$.

There are N vehicles arriving at the starting point of the road. The vehicles are running at speed v. Each vehicle has a computation task to be completed with a stringent delay constraint. The tasks include interactive gaming, real-time financial trading, virtual reality and etc. Each computation task can be described as $D_i \triangleq (d_i, c_i, T_i^{max}), i \in \mathcal{N} = \{1, 2, ..., N\}$, where d_i denotes the size of computation input data (e.g. the program codes and input parameters), c_i is the required computation resource for computing task D_i, and T_i^{max} denotes the maximum latency allowed to accomplish the task.

Each task can either be offloaded to a selected VEC server to process, or be executed locally at the vehicle. Denote selection decision variables by x_{ij} and x_{i0} indicating whether the task of vehicle i is processed on the selected VEC server on RSU j (i.e. $x_{ij} = 1$) or locally (i.e. $x_{i0} = 1$). The selection decision variables should satisfy the constraint $\sum_{j=0}^{M} x_{ij} = 1$ which indicates that only one of x_{ij} could be 1.

A. Offloading to VEC Server

If vehicle i chooses to offload task D_i to a selected VEC server to process, the task processing delay can be divided into three parts. The first one is the time taken for vehicle i from the starting point to the coverage of RSU j (i.e. $\sum_{k=1}^{j} L_k)/v$). The second part is the communication time that the task is transmitted from vehicle i to RSU j. The third component is the computation time.

1) Communication Time:

We assume the wireless communication between vehicles and RSUs is based on the Orthogonal Frequency Division
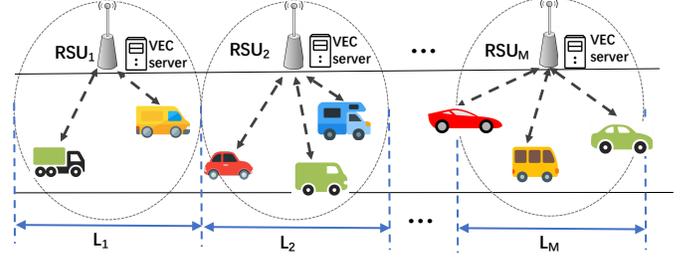

Fig. 1: The VEC offloading in a vehicular network.

Multiple Access (OFDMA). The RSUs in this system are also allocated orthogonal spectrum such that the inter-cell interference among RSUs can be ignored.

Let h_{ij} denote the channel gain and p_{ij} the transmission power for vehicle i. Then the achievable rate, denoted by r_{ij}, is given as:

$$r_{ij} = B \log(1 + \frac{p_{ij} h_{ij}}{N_0}) \tag{1}$$

where N_0 is the noise power and B is the system bandwidth.

The communication time for offloading d_i from vehicle i to RSU j can be written as

$$T_{ij}^{com} = \frac{d_i}{B \log(1 + \frac{p_{ij} h_{ij}}{N_0})} \tag{2}$$

2) Computation Time:

Since multiple vehicles may choose the same VEC server as their offloading target and the computation resource of each VEC server is limited, we need to allocate this computation resource to satisfy all vehicles' latency constraints. Denote F_j as the total computation resource of VEC server on RSU j and f_{ij} is the amount of computation resource that the VEC server assigns to vehicle i. We have $\sum_{i=1}^{N} x_{ij} f_{ij} \leq F_j$. The computation execution time T_{ij}^{vec} will be

$$T_{ij}^{vec} = \frac{c_i}{f_{ij}} \tag{3}$$

Then, the task processing delay for offloading D_i from vehicle i to RSU j can be expressed as

$$T_{ij} = \sum_{k=1}^{j} \frac{L_k}{v} + T_{ij}^{com} + T_{ij}^{vec} \tag{4}$$

B. Local Processing

If vehicle i executes its computation task D_i locally, the task processing delay is determined by its own computation resource. Let f_i denote the computational resource of vehicle i, which varies for different users and can be obtained through offline measurement [10]. The local computation execution delay T_{i0} can now be expressed as

$$T_{i0} = \frac{c_i}{f_i} \tag{5}$$

C. System Utility Function

Different from the task offloading in previous works which aim to optimize energy consumption [3], [4], [5], the task processing time is a critical metric for vehicles. We therefore design a QoS based utility function, i.e., based on the task processing time.

Since each task can either be offloaded to a selected VEC server to process, or be executed locally, the task processing time can be expressed as

$$T_i = \sum_{j=1}^{M} x_{ij}T_{ij} + x_{i0}T_{i0} = \sum_{j=0}^{M} x_{ij}T_{ij} \tag{6}$$

Due to the fact that moving vehicles may have a higher satisfaction with a smaller T_i, the utility function should monotonically decrease with T_i. Moreover, because of the limitation of computation resource, offloading can be less efficient and result in overload, if all vehicles select the same VEC server to offload their task to. The utility function also should balance the load among VEC servers. According to [11], the logarithmic utility is known as proportional fairness, which is able to achieve load balancing. Therefore, we define the utility function as

$$U_i(x_{ij}, f_{ij}) = \alpha \log(1 + \beta - T_i) \tag{7}$$

where α is a satisfaction parameter, β is used to normalize the satisfaction to be nonnegative. The higher the α, the more gain of satisfaction. The system utility function can be written as $U = \sum_{i=1}^{N} U_i$.

III. PROBLEM FORMULATION AND SOLUTION

In this section, we formulate the joint offloading and resource allocation scheme as an optimization problem. The objective is to maximize the system utility. Define $\mathbf{x} = \{x_{ij}\}$ as the vector of VEC server selection decision and $\mathbf{f} = \{f_{ij}\}$ as the computation resource vector, respectively. We formulate the optimization problem as follows:

$$\max \quad U(\mathbf{x}, \mathbf{f}) = \sum_{i=1}^{N} \alpha \log(1 + \beta - T_i) \tag{8a}$$

$$s.t. \quad T_i \leqslant T_i^{max}, \quad \forall\, i \in \mathcal{N} \tag{8b}$$

$$\sum_{j=0}^{M} x_{ij} = 1, \qquad \forall\, i \in \mathcal{N} \tag{8c}$$

$$\sum_{i=1}^{N} x_{ij}f_{ij} \leqslant F_j, \quad \forall\, j \in \mathcal{M} \tag{8d}$$

$$0 \leqslant f_{ij} \leqslant F_j \qquad \forall\, i \in \mathcal{N},\, j \in \mathcal{M} \tag{8e}$$

$$x_{ij} \in \{0,1\}, \quad \forall\, i \in \mathcal{N},\, j \in \mathcal{M} \cup \{0\} \tag{8f}$$

The first constraint (8b) guarantees that the task processing time cannot exceed the maximum allowed latency T_i^{max}. Constraints (8c) and (8f) state each vehicle offloads its task to one and only one VEC server. Constraints (8d) and (8e) ensure that the sum of the computation resource assigned to all tasks, which choose the VEC server on RSU j, does not

exceed the total computation capacity of this VEC server. The key challenge in solving this problem is the integer constraint $x_{ij} \in \{0,1\}$, which makes (8) a mixed-integer non-linear programming problem and this is in general non-convex and NP-hard [12].

In order to solve (8), we first transform it into an equivalent form as shown in Lemma 1.

Lemma 1. *The optimization problem (8) can be transformed into the following equivalent problem:*

$$\max \quad U(\mathbf{x}, \mathbf{f}) = \sum_{i=1}^{N} \sum_{j=0}^{M} \alpha \log(1 + \beta - x_{ij}T_{ij}) \tag{9a}$$

$$s.t. \quad \sum_{j=1}^{M} x_{ij}\left(\Lambda_{ij} + \frac{c_i}{f_{ij}}\right) \leqslant T_i', \quad \forall\, i \in \mathcal{N} \tag{9b}$$

$$(8c), (8d), (8e), (8f)$$

where $\Lambda_{ij} = \sum_{k=1}^{j} \frac{L_k}{v} + T_{ij}^{com} - \frac{c_i}{f_i}$ and $T_i' = T_i^{max} - T_{i0}$.

Proof. See Appendix A. ∎

It is still challenging to solve (9) which has non-linear objective function and integer constraint (8f). Therefore, we decouple selection decision and computation resource allocation into two subproblems to develop a low-complexity algorithm. That is, we determine $\mathbf{x}$ under given $\mathbf{f}$ and then $\mathbf{f}$ under obtained $\mathbf{x}$, and repeat this process until convergence.

A. Selective Decision

The selection decision problem for a given $\mathbf{f}$ from (9) takes the form

$$\max \quad U(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j=0}^{M} \alpha \log(1 + \beta - x_{ij}T_{ij}) \tag{10}$$

$$s.t. \quad (8c), (8d), (8f), (9b)$$

In (10), all the indicator variables x_{ij}s are binary, while $U(\mathbf{x})$ is a non-linear function with respect to x_{ij}. Thus it is a Mixed Integer Non-Linear Programming problem (MINLP), which is usually NP-hard.

To solve this problem with low complexity, we propose an approximation algorithm. First, we construct a subset $\mathcal{B}_i$ with a latency threshold. Then, we relax the original MINLP problem as a non-linear programming problem (RNLP) and solve it using standard convex method. Finally, we use rounding to obtain a feasible solution.

To ensure each task of vehicles can be accomplished respecting its latency deadline, given $\mathbf{f}$, we construct an available subset $\mathcal{B}_i$

$$\mathcal{B}_i = \mathcal{M} \cap \{j | \frac{T_{ij}}{T_{ij}^*} \leqslant \rho\} \tag{11}$$

where $T_{ij}^* = \min_{j \in \mathcal{M}} T_{ij}$ and ρ is a threshold. Usually only a limited number of RSU will satisfy the above constraint (11).

For the RSU that cannot meet the threshold, let $x_{ij} = 0, j \notin \mathcal{B}_i$. Thus, the complexity of (10) will be greatly reduced.

Once $\mathcal{B}_i$ is determined, we relax the binary variable x_{ij} into a real value in $[0, 1]$. Then the MINLP problem (10) is relaxed into an RNLP as follows:

$$\max \quad U(\mathbf{x}) = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{B}_i} \alpha \log(1 + \beta - x_{ij} T_{ij}) \quad (12a)$$

$$s.t. \quad \sum_{j \in \mathcal{B}_i} x_{ij}(\Lambda_{ij} + \frac{c_i}{f_{ij}}) \leqslant T_i', \quad \forall\, i \in \mathcal{N} \quad (12b)$$

$$x_{i0} + \sum_{j \in \mathcal{B}_i} x_{ij} = 1, \quad \forall\, i \in \mathcal{N} \quad (12c)$$

$$x_{ij} \geqslant 0, \quad \forall j \in \mathcal{B}_i \cup \{0\}, \quad \forall i \in \mathcal{N} \quad (12d)$$
$$(8d)$$

where $x_{ij} = 0, j \notin \mathcal{B}_i$.

Since the sum of x_{ij} is already upper bounded by 1 in (12c), we remove the upper bound 1 of x_{ij} and obtain the constraint (12d). The objective function $U(\mathbf{x})$ is concave. Combining with the linear convex constraints, problem (12) is a convex optimization problem and we can solve (10) to obtain an optimal fractional solution, which is an upper bound of the original MINLP problem, because it is obtained by expanding the solution space.

We denote the fractional solution of (12) as $\mathbf{x}' = \{x'_{ij} | x'_{ij} \in [0, 1]\}$. The solution of RNLP is usually an infeasible solution to the original MINLP problem as it is fractional. Therefore, we adopt a rounding method from [13] to obtain a feasible solution for the MINLP problem. In this rounding method, a bipartite graph is constructed according to the RNLP solution, which is constructed as an undirected bipartite graph.

The rounding technique consists of the following two steps: 1) construct a weighted bipartite graph to establish the relationship between vehicles and RSUs, 2) find a maximum matching to obtain the integer solution based on the bipartite graph.

In step 1, we construct the weighted bipartite graph $\mathcal{G}(\mathcal{U}, \mathcal{V}, \mathcal{E})$ to establish the relationship between vehicles and RSUs. The set $\mathcal{U}$ represents the vehicles in the network. The set $\mathcal{V} = \{v_{js} : j \in \mathcal{B}_i, s = 1, ..., J_j\}$, where $J_j = \lceil \sum_{i=1}^{N} x'_{ij} \rceil$ implies VEC server on RSU j serves J_j vehicles. The nodes $\{v_{js:s=1,..,J_j}\}$ correspond to RSU j.

The most important procedure for constructing $\mathcal{G}$ is to set the edges and the edge weight between $\mathcal{U}$ and $\mathcal{V}$. The edges in $\mathcal{G}$ are constructed according to the following method.

Construct the edges of bipartite graph $\mathcal{G}$:

If $J_j \leqslant 1$, there is only one node v_{j1} corresponding to RSU j. For each $x'_{ij} > 0$, add edge (u_i, v_{j1}) and set the weight of this edge as x'_{ij}.

Otherwise,

1) Find the minimum index i_s such that $\sum_{i=1}^{i_s} x'_{ij} \geqslant s$.

---

2) For $i = i_{s-1} + 1, .., i_s - 1$, and $x'_{ij} > 0$, add edge (u_i, v_{js}) with weight x'_{ij}.

3) For $i = i_s$, add edge (u_i, v_{js}) with weight $1 - \sum_{i=1}^{i_s-1} x'_{ij}$. This ensures that the total weight of edges connecting v_{js} is at most 1.

4) If $\sum_{i=1}^{j_s} x'_{ij} > s$, add edge $(u_i, v_{j(s+1)})$ with weight $\sum_{i=1}^{i_s} x'_{ij} - s$.

Based on the above steps, we construct a weighted bipartite graph $\mathcal{G}(\mathcal{U}, \mathcal{V}, \mathcal{E})$. In step 2, we use the Hungarian algorithm [14] to find a maximum profit matching whose total profit is the maximum among all matchings. Note that, the *profit* of each edge is defined to be $\alpha \log(1 + \beta - T_{ij})$. According to the matching, we obtain the integer selection decision. Specifically, if the edge (u_i, v_{js}) is in the matching, set $x_{ij} = 1$; otherwise, $x_{ij} = 0$. This maximum matching indicates a feasible solution for the MINLP problem. According to [13], the solution produced by the rounding approximation algorithm is at most $(1 + \rho)$ times greater than the optimal solution.

B. Optimization of Computation Resource

The computation resource allocation problem for a given $\mathbf{x}$ is

$$\max \quad U(\mathbf{f}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \alpha \log(1 + \beta - x_{ij}(\frac{c_i}{f_{ij}} + \Lambda_{ij})) \quad (13)$$
$$s.t. \quad (8d), (8e), (9b)$$

where $\sum_{i=1}^{N} \alpha \log(1 + \beta - x_{i0} T_{i0})$ is omitted from (13), as it is a constant.

Lemma 2. *Problem (13) is a convex optimization problem.*

Proof. See Appendix B. $\square$

Since (13) is a convex problem, we use Lagrangian method to solve this problem. The Lagrangian function is

$$\mathcal{L}(\mathbf{f}, \theta, \psi, \varrho, \omega) = \sum_{i=1}^{N} \sum_{j=1}^{M} \alpha \log(1 + \beta - x_{ij}(\frac{c_i}{f_{ij}} + \Lambda_{ij})) - \\ \sum_{i=1}^{N} \theta_i (\sum_{j=1}^{M} x_{ij}(\Lambda_{ij} + \frac{c_i}{f_{ij}}) - T_i') - \\ \sum_{j=1}^{M} \psi_j (\sum_{i=1}^{N} x_{ij} f_{ij} - F_j) - \sum_{i=1}^{N} \sum_{j=1}^{M} \varrho_{ij}(f_{ij} - F_j) \\ + \sum_{i=1}^{M} \sum_{j=1}^{M} \omega_{ij} f_{ij} \quad (14)$$

where θ, ψ, ϱ, and ω are the Lagrangian multipliers. The Lagrange dual function is then given by:

$$\mathcal{D}(\theta, \psi, \varrho, \omega) = \max \quad \mathcal{L}(\mathbf{f}, \theta, \psi, \varrho, \omega) \quad (15)$$

and the dual problem of (13) is

$$\min \quad \mathcal{D}(\theta, \psi, \varrho, \omega) \\ s.t. \quad \theta \succ 0, \; \psi, \; \varrho, \; \omega \succeq 0 \quad (16)$$

Since (13) is convex, there exists a strictly feasible point where Slater's condition holds, leading to strong duality [12]. This allows us to solve the primal problem (13) via the dual problem (16). The dual problem (16) can be solved using the gradient method. As the Lagrange function is differentiable, the gradients of the Lagrange multipliers can be obtained as

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_i} &= -(\sum_{j=1}^{M} x_{ij}(\Lambda_{ij} + \frac{c_i}{f_{ij}}) - T_i') \\
\frac{\partial \mathcal{L}}{\partial \psi_j} &= -(\sum_{i=1}^{N} x_{ij} f_{ij} - F_j) \\
\frac{\partial \mathcal{L}}{\partial \varrho_{ij}} &= -(f_{ij} - F_j) \\
\frac{\partial \mathcal{L}}{\partial \omega_{ij}} &= f_{ij}
\end{aligned}
\tag{17}
$$

By applying the gradient method, the Lagrange multipliers are calculated iteratively as follows:

$$
\begin{aligned}
\theta_i(t+1) &= \left[\theta_i(t) + \kappa_1 \frac{\partial \mathcal{L}}{\partial \theta_i}\right]^+ \\
\psi_j(t+1) &= \left[\psi_j(t) + \kappa_2 \frac{\partial \mathcal{L}}{\partial \psi_j}\right]^+ \\
\varrho_{ij}(t+1) &= \left[\varrho_{ij}(t) + \kappa_3 \frac{\partial \mathcal{L}}{\partial \varrho_{ij}}\right]^+ \\
\omega_{ij}(t+1) &= \left[\omega_{ij}(t) + \kappa_4 \frac{\partial \mathcal{L}}{\partial \omega_{ij}}\right]^+
\end{aligned}
\tag{18}
$$

where $\kappa_1, \kappa_2, \kappa_3, \kappa_4 > 0$ are the gradient steps, t represents the gradient number, and $[\cdot]^+$ denotes $\max(0, \cdot)$.

Taking the first-order derivative of $\mathcal{L}$ with respect to f_{ij} and setting the result to zero, we obtain,

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial f_{ij}} =& \frac{\alpha \, x_{ij} c_i}{f_{ij}^2 \ln(2)\left(1 + \beta - x_{ij}\left(\frac{c_i}{f_{ij}} + \Lambda_{ij}\right)\right)} \\
&+ \frac{\theta_i \, x_{ij} c_i}{f_{ij}^2} - \psi_j x_{ij} - \varrho_{ij} + \omega_{ij} = 0
\end{aligned}
\tag{19}
$$

C. Joint Algorithm for Selection Decision and Computation Resource

Based on above analysis, we present our joint algorithm for selection decision and computation resource (JOSC), which is summarized in Algorithm 1. According to JOSC, (9) can be solved in a semi-distributed manner. Specifically, based on given $\mathbf{f}$, each vehicle obtains its selection decision by solving relaxation problem (12) and using rounding method. Then under the obtained selection decision and by exchanging computation resources among neighboring vehicles, each vehicle uses Lagrangian method to obtain $\mathbf{f}$ and repeat this process until convergence.

At each iteration of Algorithm 1, the computational complexity of solving convex problem (12) is only polynomial in the number of variables and constraints. The complexity required to solve (12) is thus $\mathcal{O}((1 + a + b)a^2\sqrt{b+1})$,

where $a = N * M'$ is the number of decision variables, $b = 3N + M + M'$ is the number of linear constraints and $M' = |\mathcal{B}_i|$ ($|\cdot|$ denotes the cardinality of a set). The complexity of rounding is polynomial in the number of nodes and edges, that is $\mathcal{O}(|\mathcal{V}||\mathcal{E}|)$. For t iterations, the complexity of the inner loop of Algorithm 1 is $\mathcal{O}(t)$. Thus, the total complexity of Algorithm 1, for k iterations, is $\mathcal{O}(k(t + (1 + a + b)a^2\sqrt{b+1} + |\mathcal{V}||\mathcal{E}|)$.

Algorithm 1 Joint Optimization for Selection and Computation algorithm (JOSC)

1: Initialization:
2:    Set selection decision $\mathbf{x}^{(0)}$ and computation resource $\mathbf{f}^{(0)}$ to arbitrary values;
3:    Set the number of iteration as $k = 0$;
4: **repeat**
5:    Based on $\mathbf{f}^{k-1}$, each vehicle obtains its selection decision by solving (12) and using rounding method;
6:    **repeat**
7:        Update $\theta(t+1)$, $\psi(t+1)$, $\varrho(t+1)$ and $\omega(t+1)$ based on (18);
8:        Calculate $\mathbf{f}^{(k)}$ using (19);
9:    **until** Convergence.
10:    k = k+1;
11: **until** Convergence.

IV. NUMERICAL RESULTS

In this section, we present extensive simulation results to evaluate the performance of the proposed algorithm.

A. Simulation Setup

We consider a unidirectional road, where 5 RSUs are randomly located along a 100-meter road. There are 40 arriving vehicles on the road, and they are running at a constant speed of 120 km/hr. The bandwidth of each RSU is 1.25 KHz. The transmission power of each vehicle is 100 mW and the noise power is $10^{-10} mW$. We set the channel gain $h_{ij} = d_{ij}^{-4}$, where d_{ij} is the distance between vehicle i and RSU j.

Each RSU is equipped with a VEC server. The computation resources of the VEC servers from the beginning of the road to the other end are $\{5, 10, 15, 20, 25\}$ GHz. Each vehicle has a computation task. The input data size, required computation resources, and maximum latency constraint of each computation task are uniformly distributed in the range of $[100, 300]$ KB, $[0.5, 1.5]$ GHz, and $U[8, 10]$ s, respectively. The computation capability of each vehicle is 1 GHz.

To verify the performance of our proposed JOSC algorithm, we introduce the following benchmark schemes,

- GS: Given a feasible computation resource, the Greedy Selection optimization scheme (GS) greedily picks out the best VEC server with maximal utility for each vehicle.
- RA: The Resource Allocation scheme (RA) optimizes computation resource, as in [4]. In this scheme, all vehicles offload their tasks to the nearest VEC server.
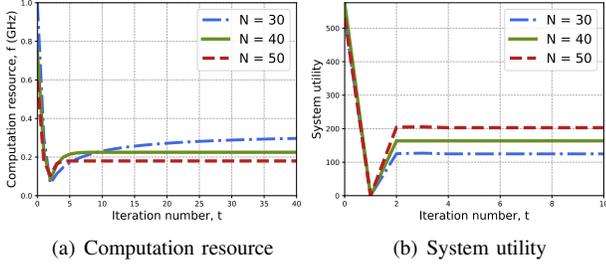
(a) Computation resource   (b) System utility

Fig. 2: Convergence of JOSC under different N.

B. Performance Analysis

Fig. 2 shows the convergence of our proposed JOSC algorithm. Specifically, Fig. 2(a) plots the convergence of the inner loop of Algorithm 1, i.e. the loop from step 6 to step 9 in JOSC. We observe that the computation resource can achieve converge very fast and a smaller value of N results in more computation resource. From Fig. 2(b), we can see that the convergence of system utility can achieve converge within 4 iterations and a larger value of N leads to higher system utility.

Fig. 3 shows system utility with respect to the number of arriving vehicles at the starting point under different schemes. We can draw several observations from Fig. 3. First, the system utility of the proposed JOSC is obviously higher than GS and RA since it jointly optimizes VEC server selection and computation resource allocation. Second, the system utility of JOSC and GS increases rapidly as the number of arriving vehicles increases while the system utility of RA starts decreasing when the number of arriving vehicles is larger than 45. The reason is that RA does not consider VEC server selection which prolongs task processing time and decreases system utility. On the contrary, JOSC and GS improves system utility by performing selection decision. Moreover, the system utility gap between JOSC and GS increases with increasing the number of vehicles. This is justified since the JOSC adopts the optimal policies of selection decision and computation resource but GS only considers selection decision.

Fig. 4 depicts load balancing performance of JOSC, RA and GS for the same profile (i.e. $N = 40, M = 4$). We can see that the performance of JOSC is significantly better than that of GS and RA and RA does not completely balance the load among servers. While GS greedily chooses the VEC server which provides maximal utility, RA offloads all tasks to the nearest VEC server (i.e. Server 1) which leads to an unbalanced assignment of resources. On the contrary, JOSC balances the number of vehicles served by each server and keeps a higher system utility.

V. CONCLUSIONS

In this paper, we proposed a joint offloading and resource allocation approach for maximizing system utility in vehicular edge computing networks. We incorporated the components due to both communication and computation in the definition
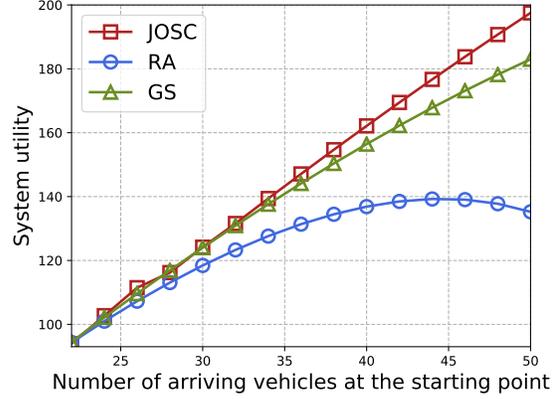

Fig. 3: Comparison of system utility of the number of arriving vehicles at the starting point under different schemes.
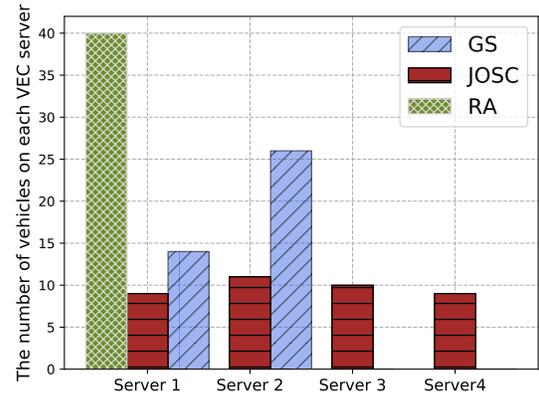

Fig. 4: Comparison of load balance under different schemes with $N = 40$ and $M = 4$.

of the task processing delay. We provided a low complexity, JOSC, by jointly optimizing selection offloading decision and computation resource. Numerical results demonstrated that the proposed JOSC not only significantly outperforms the benchmark policies in terms of system utility, but also performs better on balancing the load distribution, compared to the other counterparts.

APPENDIX A
PROOF OF LEMMA 1

According to constraints (8c) and (8f), the objective function $\sum_{i=1}^{N} \alpha \log(1 + \beta - \sum_{j=0}^{M} x_{ij} T_{ij})$ is equivalent to $\sum_{i=1}^{N} \sum_{j=0}^{M} \alpha \log(1 + \beta - x_{ij} T_{ij})$.

Substituting (3)-(6) into (8b), with $x_{i0} = 1 - \sum_{j=1}^{M} x_{ij}$, we

have

$$\sum_{j=1}^{M} x_{ij}(T_{ij} - T_{i0}) \leqslant T_i^{max} - T_{i0}$$

$$\sum_{j=1}^{M} x_{ij}\left(\sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{com} + \frac{c_i}{f_{ij}} - \frac{c_i}{f_i}\right) \leqslant T_i^{max} - T_{i0} \quad (20)$$

$$\sum_{j=1}^{M} x_{ij}\left(\Lambda_{ij} + \frac{c_i}{f_{ij}}\right) \leqslant T_i'$$

where $\Lambda_{ij} = \sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{com} - \frac{c_i}{f_i}$ and $T_i' = T_i^{max} - T_{i0}$. Thus, we obtain constraint (9b).

APPENDIX B
PROOF OF LEMMA 2

The second-order derivative of $U(\mathbf{f})$ with respect to f_{ij} is

$$\frac{\partial^2 U(\mathbf{f})}{\partial f_{ij}^2} = - \frac{2\alpha\, x_{ij}c_i}{f_{ij}^3 \ln(2)} \left(1 + \beta - \frac{x_{ij}c_i}{f_{ij}} - x_{ij}\Lambda_{ij}\right)^{-1}$$
$$- \frac{\alpha\, x_{ij}^2 c_i^2}{f_{ij}^4 \ln(2)} \left(1 + \beta - \frac{x_{ij}c_i}{f_{ij}} - x_{ij}\Lambda_{ij}\right)^{-2} \quad (21)$$

As $\alpha > 0$, $x_{ij}c_i \geqslant 0$, and $\left(1 + \beta - \frac{x_{ij}c_i}{f_{ij}} - x_{ij}\Lambda_{ij}\right) > 0$,

we have $\frac{\partial^2 U(\mathbf{f})}{\partial f_{ij}^2} \leqslant 0$. Thus the objective function $U(\mathbf{f})$ is concave. Combining with the linear convex constraints, (13) is a convex optimization problem.

REFERENCES

[1] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, 2017.

[2] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 2017.

[3] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, 2017.

[4] Y. Yu, J. Zhang, and K. B. Letaief, "Joint subcarrier and cpu time allocation for mobile edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2016.

[5] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, 2017.

[6] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, 2016.

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

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

[9] K. Zhang, Y. Mao, S. Leng, A. Vinel, and Y. Zhang, "Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks," in *Proc. 8th Int. Workshop Resilient Netw. Design Modeling (RNDM)*, 2016.

[10] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing." in *Proc. 2nd USENIX Conf. Hot Topics Cloud Computing*, vol. 10, 2010.

[11] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Emerging Telecommun.*, vol. 8, no. 1, pp. 33–37, 1997.

[12] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[13] D. B. Shmoys and É. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, no. 1-3, pp. 461–474, 1993.

[14] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Res. Logist. (NRL)*, vol. 2, no. 1-2, pp. 83–97, 1955.