Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks

Yueyue Dai, *Student Member, IEEE,* Du Xu, *Member, IEEE,* Sabita Maharjan, *Member, IEEE,* and Yan Zhang, *Senior Member, IEEE*

***Abstract*—The emergence of computation intensive and delay sensitive on-vehicle applications makes it quite a challenge for vehicles to be able to provide the required level of computation capacity, and thus the performance. Vehicular Edge Computing (VEC) is a new computing paradigm with a great potential to enhance vehicular performance by offloading applications from the resource-constrained vehicles to lightweight and ubiquitous VEC servers. Nevertheless, offloading schemes where all vehicles offload their tasks to the same VEC server, can limit the performance gain due to overload. To address this problem, in this paper, we propose integrating load balancing with offloading, and study resource allocation for a multi-user multi-server VEC system. First, we formulate the joint load balancing and offloading problem as a mixed integer non-linear programming problem to maximize system utility. Particularly, we take IEEE 802.11p protocol into consideration for modeling the system utility. Then, we decouple the problem as two subproblems and develop a low-complexity algorithm to jointly make VEC server selection, and optimize offloading ratio and computation resource. Numerical results illustrate that the proposed algorithm exhibits fast convergence and demonstrates the superior performance of our joint optimal VEC server selection and offloading (JSCO) algorithm compared to the benchmark solutions.**

***Index Terms*—Vehicular edge computing (VEC), computation offloading, offloading, load balance, resource allocation, optimization.**

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, thus resulting in bottlenecks and making it challenging for the vehicles to ensure the required level of Quality of Service (QoS) [1], [2]. 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], [3], [4].

This research is partially supported by the National key Research and Development Program of China-No.2016YFB0800105; the projects 240079/F20 funded by the Research Council of Norway; the 111 project (B14039); and the scholarship from China Scholarship Council under Grant 201706070009.

Y. Dai and D. Xu are with the Key Laboratory of Optical Fiber Sensing and Communications, University of Electronic Science and Technology of China, Chengdu, China (email:daiyyue@gmail.com;xudu.uestc@gmail.com).

S. Maharjan is with Simula Metropolitan Center for Digital Engineering and University of Oslo, Norway (email: sabita@simula.no).

Y. Zhang (corresponding author) is with the Department of Informatics, University of Oslo, Norway (email: yanzhang@ieee.org).

Vehicular edge computing is a promising new paradigm that has received much attention lately, as it can extend the computation capability to vehicular network edge [5], [6]. With the advent of VEC, lightweight but ubiquitous edge resources deployed on nearby Road Side Units (RSUs) offer vehicles high QoS. Further, localized processing is enabled to save backhaul bandwidth and awareness about location is also beneficial to resource allocation.

There has been considerable amount of work focusing on computation offloading under cellular network. The authors considered/analyzed the problem of binary offloading decision where each user independently chooses whether to execute the task locally or to offload the task to the edge servers, to minimize energy consumption and/or computation latency [7], [8], [9]. Studies such as [10], [11] and [12] proposed partial offloading where the input data can be arbitrarily divided into two parts for local and edge computation. Compared to binary offloading, it is more reasonable to offload partially to VEC server, since wireless channel resource is limited. Moreover, different from the cellular network, the wireless technology in vehicular environment is generally adopted the contention based IEEE 802.11p standard [13], which makes partial offloading even more reasonable. Motivated by such considerations, in this paper, we dynamically offload parts of computation tasks from vehicles to VEC servers based on IEEE 802.11p.

In conventional MEC network, all mobile users have to offload their tasks to only one MEC server located at the base station. As a result, some users' tasks may not be accomplished within the permissible latency threshold. A vehicular network consists of multiple VEC servers and each vehicle can select one of them as its target offloading server. To satisfy vehicles' requirements, we balance the load among VEC servers and, at the same time, maximize system utility. In addition, as the mobility of vehicles also has a significant impact on task completion latency, we also incorporate mobility into problem formulation.

We propose integrating load balancing with offloading in order to address the following critical challenges: 1) VEC server selection for offloading and load balancing, 2) determining optimal offloading fraction to maximize system utility. To address these challenges, we consider a VEC network, which consists of multiple mobile vehicles and RSUs. Each RSU consists of a VEC server. Mobile vehicles are required to compute different computation tasks within a certain time. Each vehicle can select one VEC server to offload the task to. We model VEC server selection as a binary decision. By jointly optimizing offloading ratio and selection decision, we obtain the optimal strategy for maximizing system utility. Further, computation resource allocation is also taken into consideration. The key contributions of our work are as follows:

- We jointly model VEC server selection for offloading and load balancing to execute tasks associated with vehicular applications.
- We formulate the joint load balancing and offloading problem as a system utility maximization problem under the permissible latency constraint. We adopt comprehensive task processing delay as the performance metric by jointly analyzing transmission model and computation model.
- We propose a joint optimal VEC server selection and offloading (JSCO) algorithm with which we can find the solution of the optimization problem in a distributed manner and with less overhead. Extensive experiments demonstrate the effectiveness of proposed schemes.

The rest of this paper is organized as follows. Related works on offloading are presented in Section II. In Section III, we introduce the system model. In Section IV, we formulate the joint load balancing and offloading problem and propose a low-complexity JSCO algorithm to solve this problem. We evaluate the performance of the JSCO algorithm and provide illustrative results in Section V. We conclude the paper in Section VI.

II. Related work

MEC has recently attracted much attention from both academia and industry, as it brings computation and storage resources to the network edge enabling computing and caching capabilities in close proximity to users, which are in line with the requirements of several new and emerging applications including vehicular applications. Recent works on MEC include those that reduce energy consumption [7], [8], [10], [12], [14], [15], and those that reduce latency [16], [17] by offloading computation-heavy and latency-sensitive tasks to nearby RSUs and/or base stations for remote execution. Studies such as [18], [19] jointly considered content caching and offloading for MEC, and implied that moving popular contents close to end users can reduce replicated transmission and improve users' quality of experience.

The author in [7] proposed to minimize the system energy consumption while also ensuring that latency constraints of the computation tasks are met. With dynamic voltage and frequency scaling techniques, binary offloading [8] and partial offloading [10] for minimizing energy consumption by controlling the CPU-cycle frequencies were proposed. In [12], the offloading problem was studied with an objective to minimize the weighted sum of mobile energy consumption under the constraint on computation latency for a multi-user MEC system. In [14], computation offloading was investigated to minimize energy consumption at the mobile devices. In [15], the authors proposed a multi-user MEC system by integrating a multi-antenna Access Point (AP) with an MEC server such that mobile users can execute their respective tasks locally by themselves or offload them to the AP. In [16], a delay-optimal computation offloading algorithm was proposed by jointly considering the offloading decision, the CPU-cycle frequencies for mobile execution, and the transmit power for computation offloading. The authors in [17] formulated a power-constrained delay minimization problem and proposed an efficient one-dimensional search algorithm to solve it. Several other works such as [9], [11] attempted to find a trade-off between energy consumption and execution delay. In [9], the authors proposed a multi-user computation offloading problem and adopted a game theoretic approach for achieving efficient computation offloading in a distributed manner. In [11], the authors proposed an integrated framework for computation offloading and interference management for a wireless cellular network. The aforementioned computation offloading schemes assume that there is only one MEC server deployed at a base station or an AP. In our scenario of a vehicular network, multiple VEC servers exist along the road, and the vehicles are mobile, thus making the state of the art schemes either infeasible or of limited applicability.

A few studies such as [20], [21] investigated the performance of computation offloading for vehicle networks. In [20], the authors proposed a stackelberg game theoretic approach to design an optimal multilevel offloading scheme, which maximizes the utilities of both the vehicles and the VEC servers. In [21], the authors designed an efficient computation offloading strategy through a contract theoretic approach to reduce offloading latency and transmission. Different from the above work, in this paper we propose joint load balancing and offloading such that each vehicle can select one VEC server and offload parts of its computation task for edge execution which can help to enhance the system utility.

III. System Model

A. VEC Network

We consider a unidirectional road, where M RSUs are located along the road, as shown in Fig. 1. Each RSU is equipped with a VEC server, whose computation resources are limited. We denote the id set of these RSUs as $\mathcal{M} = \{1, ..., M\}$. Due to variation in wireless environment, the size of the wireless coverage areas of these RSUs may be different [22]. We divide the road into M segments, with length $\{L_1, L_2, .., L_M\}$ respectively.

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, which 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 be divided into two parts and parallel executed at the vehicles and VEC servers. Specifically, let $\lambda_{ij}(0 \leq \lambda_{ij} \leq 1)$ be the offloading ratio variable which represents the ratio of the offloaded task to the total task. Vehicle i offloads $\lambda_{ij} d_i$ to the VEC server on RSU j and computes the rest $(1 - \lambda_{ij}) d_i$ locally [12]. Denote $x_{ij} \in \{0, 1\}$ as the selection decision variable, i.e., $x_{ij} = 1$ if vehicle i chooses VEC server on RSU j as its target offloading

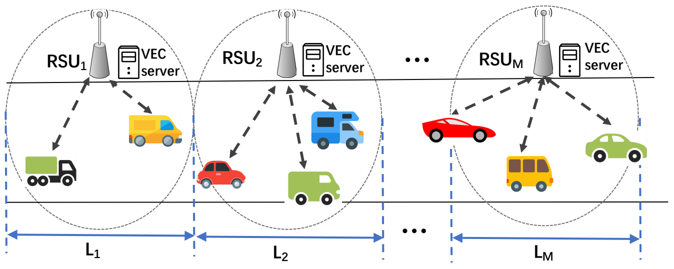


Fig. 1: The VEC offloading in a vehicular network.

server and offloads $\lambda_{ij}d_i$ parts of the task to this server, and $x_{ij}=0$ otherwise.

The processing of a computation-intensive task involves three main components of time which are incured due to mobility of vehicle, wireless transmission, and computation. Thus, the task processing delay can be correspondingly divided into three parts. The first one is the time that vehicle i moves from the starting point to the coverage of RSU j (i.e. $\sum_{k=1}^{j-1}L_j)/v$). The second part is the transmission time that a task is transmitted from vehicle i to RSU j via wireless channel. The third part is the computation time which depends on the allocated computation resource and task size. Further, since the wireless access technique between a vehicle and an RSU is generally based on IEEE 802.11p which includes a contention-based Medium Access Control (MAC) protocol, it is necessary to model the transmission time with taking this protocol into consideration. Next, we present models of transmission time, computation time, task processing delay, and system utility.

B. Transmission Time

The IEEE 802.11p adopts the Carrier Sense Multiple Access protocol with Collision Avoidance (CSMA/CA) [23] for data transmission. According to CSMA/CA, the offloading task may experience one of the following processes at one time slot: 1) successful packet transmission, 2) packet collision, and 3) backoff. To reduce collisions, we also take Request To Send/Clear To Send (RTS/CTS) into account.

We define p_{ij}^s to represent successful transmission probability between vehicle i and RSU j. A successful transmission indicates that only one offloading task is successfully transmitted in a time slot without collision, thus

$$p_{ij}^s = N_j\tau_{ij}(1-\tau_{ij})^{N_j-1} \ , \tag{1}$$

where τ_{ij} is the probability that vehicle i offloads its task to RSU j in a random time slot [24] and $N_j=\sum_{i=1}^{N}x_{ij}, (N_j\le N)$ is the amount of vehicles, chosen by the VEC server on RSU j. We define the successful transmission period as T_{ij}^s, such that

$$T_{ij}^s = \Phi + T_{ij}^p, \tag{2}$$

where $\Phi = H{+}SIFS{+}\delta{+}ACK{+}AIFS{+}\delta{+}RTS{+}SIFS{+}\delta{+}CTS{+}SIFS{+}\delta$ is specific to the MAC protocol (i.e. δ is the propagation delay, $H=PHY_{head}+MAC_{head}$ denotes the overhead of packet header, $SIFS$ is the interval of the short inter-frame space, ACK is the interval of acknowledgement, $AIFS$ is the interval of the arbitration inter-frame spacing, RTS is the RTS interval, CTS is the CTS interval, respectively). T_{ij}^p is the transmission delay taking on the wireless channel, which can be written as

$$T_{ij}^p = \frac{\lambda_{ij}d_i}{B_j log(1+P_iG_{ij})} \tag{3}$$

where B_j is the bandwidth of RSU j, P_i is the transmission power of vehicle i, and G_{ij} is the channel gain between vehicle i and RSU j.

Let p_{ij}^c denote collision probability and T_{ij}^c represent the corresponding collision period. p_{ij}^c indicates that there are at least two offloading tasks transmitted in the same time slot, which can be expressed as

$$p_{ij}^c = 1-(1-\tau_{ij})^{N_j} - N_j\tau_{ij}(1-\tau_{ij})^{N_j-1} \ . \tag{4}$$

T_{ij}^c is expressed as

$$T_{ij}^c = RTS{+}AIFS{+}\delta \ . \tag{5}$$

Let p_{ij}^{idle} denote the probability that the wireless channel is idle. The idle state means that all offloading tasks are not transmitted at the current time slot, thus p_{ij}^{idle} can be written as

$$p_{ij}^{idle} = (1-\tau_{ij})^{N_j} \ . \tag{6}$$

We define R_{ij} as the normalized throughput, which can be obtained using

$$R_{ij} = \frac{p_{ij}^s\lambda_{ij}d_i}{p_{ij}^{idle}\sigma + p_{ij}^sT_{ij}^s + p_{ij}^cT_{ij}^c} \tag{7}$$

where σ is the duration of a timeslot. As throughput presents the amount of data that enters and goes through a system in a time slot, it can also be represented in the form of data rate. Thus, the transmission time for offloading $\lambda_{ij}d_i$ from vehicle i to RSU j can be written as

$$\begin{aligned} T_{ij}^{trans} &= \frac{\lambda_{ij}d_i}{R_{ij}} \\ &= T_{ij}^s + \frac{p_{ij}^{idle}\sigma + p_{ij}^cT_{ij}^c}{p_{ij}^s} \end{aligned} \tag{8}$$

C. Computation Time

We introduce the computation model in terms of the task computation time. For each task D_i, the vehicle can execute parts of D_i locally on its own computing resources and offload the rest to a VEC server to process.

Vehicle i executes $(1-\lambda_{ij})d_i$ parts of its computation task D_i locally. Let f_i denote the computational resource of vehicle i, which varies for different users and can be obtained through offline measurement [25]. The local computation time T_{ij}^{loc} can now be expressed as

$$T_{ij}^{loc} = \frac{(1-\lambda_{ij})c_i}{f_i} \tag{9}$$

For the VEC server computing approach, vehicle i offloads $\lambda_{ij}d_i$ parts of its computation task to the VEC server on RSU

j via wireless access. Since N_j vehicles choose to offload their tasks to VEC server on RSU j and the computation resource of the VEC server is limited, we need to allocate this computation resource among N_j vehicles. 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 time T_{ij}^{vec} on VEC server will be

$$T_{ij}^{vec} = \frac{\lambda_{ij} c_i}{f_{ij}} \tag{10}$$

D. Task Processing Delay

Since each task can be divided into two parts and parallel executed locally and at the VEC server simultaneously, the task processing delay is determined by the maximum value of the two parallel executed parts. If the task processing delay is determined by the local executed part, task processing delay is equal to the local computation time since the local executed part does not experience the processes of movement and wireless transmission. If the task processing delay is determined by the VEC server executed part, task processing delay consists of three parts: movement time, transmission time, and computation time on VEC server. Thus, the task processing delay is determined by the

$$T_i = \sum_{j=1}^{M} x_{ij} \max\{T_{ij}^{loc}, \sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{trans} + T_{ij}^{vec}\} \tag{11}$$

E. System Utility Function

Different from the task offloading in previous works which aim to optimize energy consumption [7], [8], [10], [12], [14], [15], the task processing delay is a critical metric for vehicles. We therefore design a QoS based utility function, i.e., based on the task processing delay. Here, vehicles are allowed to offload its task to one VEC server at one time.

Due to the fact that the task processing delay is short, moving vehicles may have high satisfaction, the utility function as a satisfaction function which 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 [26], [27], the logarithmic utility is known as proportional fairness, which is able to achieve load balancing [28]. Therefore, the utility function is defined as

$$U_i = \alpha \log(1 + \beta - T_i) \tag{12}$$

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.

IV. Problem Formulation and Solution

In this section, we formulate the joint load balancing and offloading 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, $\mathbf{f} = \{f_{ij}\}$ as the computation resource vector, and $\lambda = \{\lambda_{ij}\}$ as offloading ratio vector, respectively. We formulate the optimization problem as follows:

$$P1 : \max_{\mathbf{x},\mathbf{f},\lambda} \quad \sum_{i=1}^{N} U_i$$

$$s.t. \quad T_i \leqslant T_i^{max}, \quad \forall\, i \in \mathcal{N} \tag{13a}$$

$$\sum_{j=1}^{M} x_{ij} = 1, \qquad \forall\, i \in \mathcal{N} \tag{13b}$$

$$0 \leqslant \sum_{i=1}^{N} x_{ij} f_{ij} \leqslant F_j, \ \forall\, j \in \mathcal{M} \tag{13c}$$

$$0 \leqslant \sum_{j=1}^{M} x_{ij} \lambda_{ij} \leqslant 1, \ \forall\, i \in \mathcal{N} \tag{13d}$$

$$0 \leqslant \lambda_{ij} \leqslant 1 \qquad \forall\, i \in \mathcal{N}, \ j \in \mathcal{M} \tag{13e}$$

$$x_{ij} \in \{0,1\}, \quad \forall\, i \in \mathcal{N}, \ j \in \mathcal{M} \tag{13f}$$

The first constraint (13a) guarantees that the task processing delay cannot exceed the maximum allowed latency T_i^{max}. Constraints (13b) and (13f) state each vehicle offloads its task to one and only one VEC server. Constraints (13c) ensures 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. Constraints (13d) and (13e) present the total offloaded task of vehicle i cannot exceed to 1. The key challenge in solving this problem is the integer constraint $x_{ij} \in \{0,1\}$, which makes P1 a mixed-integer non-linear programming problem and this is in general non-convex and NP-hard [29].

It is quite challenging to solve P1 which has non-convex constraint (13a) and integer constraint (13f). In order to solve P1, we first transform it into an equivalent form as shown in Lemma 1.

Lemma 1. *The optimization problem P1 can be transformed into the following equivalent problem:*

$$P1 : \max_{\mathbf{x},\mathbf{f},\lambda} \quad \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log(1 + \beta - x_{ij} t_{ij})$$

$$s.t. \quad \sum_{j=1}^{M} x_{ij} t_{ij} \leqslant T_i^{max}, \quad \forall\, i \in \mathcal{N} \tag{14a}$$

$$T_{ij}^{loc} \leqslant t_{ij}, \qquad \forall\, i \in \mathcal{N}, j \in \mathcal{M} \tag{14b}$$

$$\sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{trans} + T_{ij}^{vec} \leqslant t_{ij}, \ \forall\, i \in \mathcal{N}, j \in \mathcal{M} \tag{14c}$$

$$(13b), (13c), (13e)(13f)$$

Proof. we introduce an additional auxiliary variable t_{ij} to transform T_i into $\sum_{j=1}^{M} x_{ij} t_{ij}$, which can be expressed as

$$t_{ij} = \max\{T_{ij}^{loc}, \sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{trans} + T_{ij}^{vec}\} \tag{15}$$

Since $t_{ij} = \max\{T_{ij}^{loc}, \sum_{k=1}^{j-1} \frac{L_k}{v} + T_{ij}^{trans} + T_{ij}^{vec}\}$, constraint (13a) can equivalently transform into constraints

(14a), (14b), and (14c). According to the constraints of (13b) and (13f), $U_i = \alpha \log(1+\beta-\sum_{j=1}^{M} x_{ij}t_{ij}) = \sum_{j=1}^{M} \alpha \log(1+\beta - x_{ij}t_{ij})$. Moreover, according to the constraints of (13b), (13e), and (13f), (13d) can be omitted as it is naturally satisfied. Specifically, since there is one and only one x_{ij} that can be equal to 1, due to $0 \leqslant \lambda_{ij} \leqslant 1$, $\sum_{j=1}^{M} x_{ij}\lambda_{ij} = \lambda_{ij}$ cannot exceed 1, so (13d) is naturally satisfied. □

It is still challenging to solve P1 due to highly complex coupling among optimization variables and mixed combinatorial feature. Therefore, we further decouple selection decision, offloading ratio and computation resource allocation into two subproblems (i.e. selection decision, optimization of offloading ratio and computation resource) to develop a low-complexity algorithm. That is, we determine $\mathbf{x}$ under given $\mathbf{f}$ and λ and then $\mathbf{f}$ and λ under obtained $\mathbf{x}$, and repeat this process until convergence.

A. Selection Decision

The selection decision problem for a given $\mathbf{f}$ and λ from P1 takes the form

$$\begin{aligned} P1.1 : \max_{\mathbf{x}} \quad & \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log(1+\beta - x_{ij}t_{ij}) \\ s.t. \quad & (13b), (13c), (13f), (14a) \end{aligned} \tag{16}$$

In P1.1, all the indicator variables x_{ij}s are binary, while the objective function is non-linear with respect to x_{ij}. Thus it is also a mixed integer non-linear programming problem [30], which is NP-hard.

To solve P1.1 with low complexity, we propose an approximation algorithm. First, we construct a subset $\mathcal{B}_i$ with a latency threshold. Then we relax P1.1 as a continuous non-linear programming problem and solve it using standard convex method. Finally, we use a rounding method by constructing a bipartite graph based on the continuous problem solution to obtain a feasible solution.

To ensure each task of vehicles can be accomplished as soon as possible, we construct an available RSU subset. According to (15) and given λ and $\mathbf{f}$, we construct subset $\mathcal{B}_i$

$$\mathcal{B}_i = \mathcal{M} \cap \{j | \frac{t_{ij}}{t_{ij}^*} \leqslant \rho\} \tag{17}$$

where $t_{ij}^* = \min_{j\in\mathcal{M}} t_{ij}$ and ρ is a threshold. After we adopt this threshold, a limited number of RSUs are taken into consideration for a vehicle.

Once $\mathcal{B}_i$ is determined, we relax the binary variable x_{ij} into a real value in $[0,1]$. We define $S(\mathbf{x}) = \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log(1+\beta - x_{ij}t_{ij})$. The original problem P1.1 is relaxed as follows:

$$\begin{aligned} \max \quad & S(\mathbf{x}) \\ s.t. \quad & \sum_{j\in\mathcal{B}_i} x_{ij}t_{ij} \leqslant T_i^{max}, \quad \forall\, i \in \mathcal{N} && \text{(18a)} \\ & \sum_{j\in\mathcal{B}_i} x_{ij} = 1, \qquad \forall\, i \in \mathcal{N} && \text{(18b)} \\ & x_{ij} \geqslant 0, \qquad \forall j \in \mathcal{B}_i, \quad \forall i \in \mathcal{N} && \text{(18c)} \\ & x_{ij} = 0, \qquad \forall j \notin \mathcal{B}_i, \quad \forall i \in \mathcal{N} && \text{(18d)} \\ & (13c) \end{aligned}$$

Since the sum of x_{ij} is already upper bounded by 1 in (18b), we remove the upper bound 1 of x_{ij} and obtain constraint (18c).

Lemma 2. *Problem (18) is a convex optimization problem.*

Proof. The second-order derivative of $S(\mathbf{x})$ with respect to x_{ij} is

$$\frac{\partial^2 S(\mathbf{x})}{\partial x_{ij}^2} = -\frac{\alpha\, {t_{ij}}^2}{\ln(2)(1+\beta - x_{ij}t_{ij})^2} \leqslant 0 \tag{19}$$

The objective function $S(\mathbf{x})$ is concave. Combining with the linear convex constraints, problem (18) is a convex optimization problem. □

Problem (18) can be solved by an optimization solver and we can obtain an optimal fractional solution, denoted as $\mathbf{x}' = \{x'_{ij} | x'_{ij} \in [0,1]\}$. This solution is an upper bound of the original problem P1.1, because it is obtained by expanding the solution space. Unfortunately, the solution of problem (18) is usually an infeasible solution to P1.1 as it is fractional. Therefore, we adopt a rounding method from [31] to obtain a feasible solution for the original problem P1.1. In this rounding method, a bipartite graph is constructed according to the solution of problem (18), 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 using Algorithm 1.

Theorem 1. *The approximation algorithm based on non-linear programming and the rounding method ensures a $(1+\rho)$-approximation of the optimal solution.*

Proof. The details of proof can be found in [31]. □

In step 2, we use the Hungarian algorithm [32] 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 P1.1. According to Theorem 1, the solution produced by the rounding approximation algorithm is at most $(1+\rho)$ times greater than the optimal solution.

Algorithm 1 Construct the edges of bipartite graph $\mathcal{G}$

```
1: Set $\mathcal{E} \leftarrow \varnothing$.
2: if $J_j \leqslant 1$ then
3:     There is only one node $v_{j1}$ corresponding to base station $j$.
4:     for each $x'_{ij} > 0$ do
5:         Add edge $(u_i, v_{j1})$ into $\mathcal{E}$ and set the weight of this edge as $e_{ij1} = x'_{ij}$.
6:     end for
7: else
8:     Find the minimum index $i_s$ such that $\sum_{i=1}^{i_s} x'_{ij} \geqslant s$.
9:     if $i = i_{s-1}+1, .., i_s - 1$, and $x'_{ij} > 0$ then
10:        Add edge $(u_i, v_{js})$ into $\mathcal{E}$ with weight $e_{ijs} = x'_{ij}$.
11:    else if $i = i_s$ then
12:        Add edge $(u_i, v_{js})$ into $\mathcal{E}$ with weight $e_{ijs} = 1 - \sum_{i=1}^{i_s-1} x'_{ij}$. This ensures that the total weight of edges connecting $v_{js}$ is at most 1.
13:    else
14:        Add edge $(u_i, v_{j(s+1)})$ into $\mathcal{E}$ with weight $e_{ij(s+1)} = \sum_{i=1}^{i_s} x'_{ij} - s$.
15:    end if
16: end if
```

B. Joint Optimization of Computation Resource and Offload Ratio)

Substituting (2), (3), (8)-(10) into $P1$, the joint optimization of computation resource and offloading ratio problem for a given $\mathbf{x}$ from P1 takes the form

$$P1.2: \max_{\mathbf{f},\lambda} \quad \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log(1+\beta - x_{ij}t_{ij})$$

$$s.t. \quad \frac{(1-\lambda_{ij})c_i}{f_i} \leqslant t_{ij}, \qquad \forall\, i \in \mathcal{N} \tag{20a}$$

$$\sum_{k=1}^{j-1}\frac{L_k}{v} + T_{ij}^{p} + \Phi + \frac{p_{ij}^{idle}\sigma + p_{ij}^{c}T_{ij}^{c}}{p_{ij}^{s}} + \frac{\lambda_{ij}c_i}{f_{ij}} \leqslant t_{ij}, \ \forall\, i \in \mathcal{N} \tag{20b}$$

$$(13c), (13e), (14a)$$

Since t_{ij} is non-differentiable with respect to f_{ij} and λ_{ij}, we first approximate t_{ij} as

$$t_{ij} \leqslant \frac{(1-\lambda_{ij})c_i}{f_i} + \sum_{k=1}^{j-1}\frac{L_k}{v} + T_{ij}^{p} + \Phi + \frac{p_{ij}^{idle}\sigma + p_{ij}^{c}T_{ij}^{c}}{p_{ij}^{s}} + \frac{\lambda_{ij}c_i}{f_{ij}} = \lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) + \Lambda_{ij} \tag{21}$$

where $\zeta_{ij} = -\frac{c_i}{f_i} + \frac{d_i}{B_j log(1+P_iG_{ij})}$ and $\Lambda_{ij} = \frac{c_i}{f_i} + \sum_{k=1}^{j-1}\frac{L_k}{v} + \Phi + \frac{p_{ij}^{idle}\sigma + p_{ij}^{c}T_{ij}^{c}}{p_{ij}^{s}}$. Moreover, $\lambda_{ij}(\frac{c_i}{f_{ij}} + \zeta_{ij}) + \Lambda_{ij}$ is the upper bound of t_{ij}. Substituting (21) into P1.2, P1.2 can be bounded with the worst case delay as

$$\max_{\mathbf{f},\lambda} \quad \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)$$

$$s.t. \quad \sum_{j=1}^{M} x_{ij}\left(\lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) + \Lambda_{ij}\right) \leqslant T_i^{max}, \quad \forall\, i \in \mathcal{N} \tag{22a}$$

$$(13c), (13e)$$

Constraint (14a) is equivalent to (22a). (20a) and (20b) are naturally satisfied so we omit them.

Lemma 3. *Problem (22) is a non-linear and non-convex problem.*

Proof. See Appendix A. □

Problem (22) is a non-linear and non-convex program according to Lemma 3. Moreover, computation resource variables and offloading ratio variables are highly coupled with each other in both the objective function and constraints. Therefore, to solve we further decouple the two variables to develop low-complexity algorithms. Specifically, we obtain $\mathbf{f}$ under given λ and then obtain λ under given $\mathbf{f}$, and repeat this process until convergence.

1) Optimization of Computation Resource:

The computation resource allocation problem for the given λ and $\mathbf{x}$ is

$$P1.2.1: \max_{\mathbf{f}} \quad \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right) \tag{23a}$$

$$s.t. \quad (13c), (22a)$$

P1.2.1 is a convex problem as $\frac{\partial^2 U(\mathbf{f})}{\partial \mathbf{f}^2} \leqslant 0$. We use Lagrangian method to solve this problem. The Lagrangian function is

$$\mathcal{L}(\mathbf{f},\theta,\psi,\omega) = \sum_{i=1}^{N}\sum_{j=1}^{M} \alpha \log\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right) - \sum_{i=1}^{N}\theta_i\left(\sum_{j=1}^{M} x_{ij}\left(\lambda_{ij}\left(\frac{c_i}{f_{ij}} + \zeta_{ij}\right) + \Lambda_{ij}\right) - T_i^{max}\right) - \sum_{j=1}^{M}\psi_j\left(\sum_{i=1}^{N} x_{ij}f_{ij} - F_j\right) + \sum_{i=1}^{M}\sum_{j=1}^{M}\omega_{ij}x_{ij}f_{ij} \tag{24}$$

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

$$\mathcal{D}(\theta,\psi,\omega) = \max\ \mathcal{L}(\mathbf{f},\theta,\psi,\omega) \tag{25}$$

and the dual problem of P1.2.1 is

$$\begin{aligned} \min \quad & \mathcal{D}(\theta,\psi,\omega) \\ s.t. \quad & \theta \succ 0,\ \psi,\ \omega \succeq 0 \end{aligned} \tag{26}$$

Since P1.2.1 is convex, there exists a strictly feasible point, so Slater's condition holds, leading to strong duality [33]. This allows us to solve the primal problem P1.2.1 via the dual problem (26). The dual problem (26) can be solved using the gradient method. As the Lagrange function is differentiable, the gradients of the Lagrange multipliers can be obtained as

$$\begin{aligned}\frac{\partial\mathcal{L}(\mathbf{f},\theta,\psi,\omega)}{\partial\theta_i}&=-\sum_{j=1}^{M}(x_{ij}\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})+x_{ij}\Lambda_{ij})+T_i^{max}\\ \frac{\partial\mathcal{L}(\mathbf{f},\theta,\psi,\omega)}{\partial\psi_j}&=-\sum_{i=1}^{N}x_{ij}f_{ij}+F_j\\ \frac{\partial\mathcal{L}(\mathbf{f},\theta,\psi,\omega)}{\partial\omega_{ij}}&=x_{ij}f_{ij}\end{aligned} \tag{27}$$

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

$$\begin{aligned}\theta_i(t+1)&=\left[\theta_i(t)+\kappa_1\frac{\partial\mathcal{L}}{\partial\theta_i}\right]^+\\ \psi_j(t+1)&=\left[\psi_j(t)+\kappa_2\frac{\partial\mathcal{L}}{\partial\psi_j}\right]^+\\ \omega_{ij}(t+1)&=\left[\omega_{ij}(t)+\kappa_3\frac{\partial\mathcal{L}}{\partial\omega_{ij}}\right]^+\end{aligned} \tag{28}$$

where $\kappa_1,\kappa_2,\kappa_3>0$ 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,

$$\begin{aligned}\frac{\partial\mathcal{L}(\mathbf{f},\theta,\psi,\omega)}{\partial f_{ij}}=&\frac{\alpha\, x_{ij}\lambda_{ij}c_i}{f_{ij}^2\ln(2)(1+\beta-x_{ij}\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})-x_{ij}\Lambda_{ij})}\\ &+\frac{\theta_i\, x_{ij}\lambda_{ij}c_i}{f_{ij}^2}-\psi_j x_{ij}+x_{ij}\omega_{ij}=0\end{aligned} \tag{29}$$

2) Optimization of Offloading Ratio:

From $P1.2$, the offloading ratio problem for the given $\mathbf{f}$ and $\mathbf{x}$ is

$$\begin{aligned}P1.2.2:\max_{\lambda}\quad&\sum_{i=1}^{N}\sum_{j=1}^{M}\alpha\log(1+\beta-x_{ij}\lambda_{ij}(\frac{c_i}{f_{ij}}\\ &\qquad+\zeta_{ij})-x_{ij}\Lambda_{ij})\\ s.t.\quad&(13e),(22a)\end{aligned} \tag{30}$$

It is clear that the objective function of problem (30) is log-concave with respect to λ_{ij} and constraint (22a) is linear. Thus the optimization problem (30) is convex. We use primal-dual Lagrangian method to solve this problem.

The Lagrangian function is given as follows

$$\begin{aligned}\mathcal{L}_i'(\lambda,\nu,\mu,\eta)=&\sum_{j=1}^{M}\alpha\log(1+\beta-x_{ij}\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})-x_{ij}\Lambda_{ij})\\ &-\nu_i(\sum_{j=1}^{M}x_{ij}(\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})+\Lambda_{ij})-T_i^{max})\\ &-\sum_{j=1}^{M}\mu_{ij}(\lambda_{ij}-1)+\sum_{j=1}^{M}\eta_{ij}\lambda_{ij}\end{aligned} \tag{31}$$

where ν, μ, and η are the Lagrangian multipliers. Based on the KKT condition [33], the following complementary Slackness conditions must be satisfied

$$\begin{aligned}&\nu_i(\sum_{j=1}^{M}x_{ij}(\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})+\Lambda_{ij})-T_i^{max})=0\\ &0\le\lambda_{ij}\le1\\ &\mu_{ij}(\lambda_{ij}-1)=0\\ &\eta_{ij}\lambda_{ij}=0\\ &\nu_i>0,\mu_{ij}\ge0,\eta_{ij}\ge0\end{aligned} \tag{32}$$

Using the first-order derivative optimality condition, we write

$$\begin{aligned}\frac{\partial\mathcal{L}_i'(\lambda,\nu,\mu,\eta)}{\partial\lambda_{ij}}=&-\frac{\alpha\, x_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})}{\ln(2)(1+\beta-x_{ij}\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})-x_{ij}\Lambda_{ij})}\\ &-\nu_i x_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})-\mu_{ij}+\eta_{ij}=0\end{aligned} \tag{33}$$

Based on (33), we obtain the expression of λ_{ij}, as in (34).

To satisfy the slackness condition of $\mu_{ij}(\lambda_{ij}-1)=0$ and $\eta_{ij}\lambda_{ij}=0$, we consider four possible cases:

Case 1: $\mu_{ij}=0$ and $\eta_{ij}=0$, Then, $0\le\lambda_{ij}\le1$. Using $\nu_i(\sum_{j=1}^{M}x_{ij}(\lambda_{ij}(\frac{c_i}{f_{ij}}+\zeta_{ij})+\Lambda_{ij})-T_i^{max})=0$ and $\frac{\partial\mathcal{L}_i'(\lambda,\nu,\mu,\eta)}{\partial\lambda_{ij}}=0$, we get

$$\nu_i=-\frac{\alpha}{\ln(2)(1+\beta-T_i^{max})} \tag{35}$$

$$\lambda_{ij}=-\frac{x_{ij}\Lambda_{ij}-T_i^{max}}{x_{ij}}(\frac{c_i}{f_{ij}}+\zeta_{ij})^{-1} \tag{36}$$

Case 2: $\mu_{ij}=0$ and $\eta_{ij}>0$. Then, $\lambda_{ij}=0$.

Case 3: $\mu_{ij}>0$ and $\eta_{ij}=0$. Then, $\lambda_{ij}=1$. In this case, vehicle i will offload the whole tasks to VEC server on RSU j.

Case 4: $\mu_{ij}>0$ and $\eta_{ij}>0$. This case has no λ_{ij} that satisfies all KKT conditions.

From above analysis, we obtain the optimal solution λ_{ij} as

$$\lambda_{ij}=\begin{cases}-\frac{x_{ij}\Lambda_{ij}-T_i^{max}}{x_{ij}}(\frac{c_i}{f_{ij}}+\zeta_{ij})^{-1} & \text{if } \mu_{ij}=0 \text{ and } \eta_{ij}=0.\\ 0 & \text{if } \mu_{ij}=0 \text{ and } \eta_{ij}>0.\\ 1 & \text{if } \mu_{ij}>0 \text{ and } \eta_{ij}=0.\end{cases} \tag{37}$$

C. Joint Algorithm for Selection Decision, Computation Resource, and Offloading Ratio

Based on above analysis, we present our joint algorithm for selection decision, computation resource, and offloading ratio (JSCO), which is summarized in Algorithm 2. By JSCO, P1 can be solved in a distributed manner. Specifically, we decouple P1 into selection decision and optimization of offloading ratio and computation resource. First, based on given $\mathbf{f}$ and λ, each vehicle obtains its selection decision using relaxation and rounding method. Then under the obtained selection decision and only by exchanging with the neighboring vehicles

$$\lambda_{ij} = -\frac{1}{x_{ij}}(-\frac{\alpha\, x_{ij}}{\ln(2)}(\frac{c_i}{f_{ij}} + \zeta_{ij})(\nu_i\, x_{ij}(\frac{c_i}{f_{ij}} + \zeta_{ij}) + \mu_{ij} - \eta_{ij})^{-1} - 1 - \beta + x_{ij}\Lambda_{ij})(\frac{c_i}{f_{ij}} + \zeta_{ij})^{-1}5 \quad (34)$$

Algorithm 2 **J**oint Optimization for **S**election, **C**omputation, and **O**ffloading algorithm (JSCO)

```
1: Initialization:
     • Set selection decision $\mathbf{x}^{(0)}$, computation resource $\mathbf{f}^{(0)}$, and offloading ratio $\lambda^{(0)}$;
     • Set $k = 0$;
2: repeat
3:     /*Selection Decision*/
4:     Based on $\mathbf{f}^{k-1}$ and $\lambda^{k-1}$, each vehicle obtains the selection decision by solving P1.1 with rounding;
5:     /* Optimization of Computation Resource
       and Offloading Ratio*/
6:     repeat
7:         Update Lagrange mulitpliers $\theta(t+1)$, $\psi(t+1)$, and $\omega(t+1)$ based on (28);
8:         Vehicles calculate $\mathbf{f}^{(k)}$ based on (29);
9:     until Convergence
10:    Each vehicle calculates $\lambda^{(k)}$ based on (37);
11:    k = k+1;
12: until Convergence
```

which are served by the same VEC server, each vehicle uses Lagrangian method to obtain $\mathbf{f}$ and λ and repeat this process until convergence.

At each iteration of Algorithm 2, the computational complexity for solving problem P1.1 is $\mathcal{O}((1+a+b)a^2\sqrt{b+1} + |\mathcal{V}||\mathcal{E}|)$. Specifically, the computational complexity for solving problem (18) is only polynomial in the number of variables and constraints. The complexity required to solve (18) is thus $\mathcal{O}((1+a+b)a^2\sqrt{b+1})$, where $a = N * M$ is the number of decision variables, $b = 2N + NM$ is the number of linear constraints. The complexity of rounding is polynomial in the number of nodes and edges, that is $\mathcal{O}(|\mathcal{V}||\mathcal{E}|)$. Therfore, the computational complexity for solving problem P1.1 is $\mathcal{O}((1+a+b)a^2\sqrt{b+1} + |\mathcal{V}||\mathcal{E}|)$. For t iterations, the complexity of the inner loop of Algorithm 2 is $\mathcal{O}(t)$. Thus, the total complexity of Algorithm 2, for k iterations, is $\mathcal{O}(k(2+t+(1+a+b)a^2\sqrt{b+1} + |\mathcal{V}||\mathcal{E}|)$.

V. Simulation Results

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

A. Simulation Parameters

We consider a unidirectional road, where 5 RSUs are randomly located along a 100-meter road. 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 $\{c, c+\Delta, c+2\Delta, c+3\Delta, c+4\Delta\}$, where $c = 5$ GHz and $\Delta = 5$ GHz. There are 40 arriving vehicles on the road, and they are running at a constant speed 120 km/hr. Each vehicle has a computation task. We consider the input data size, required computation resources, and maximum latency constraint of each computation task are uniformly distributed in the range of U$[100, 300]$ KB, U$[0.5, 1.5]$ GHz, and U$[8, 10]$ s, respectively. The computation resource of each vehicle is 1 GHz. The wireless communication technique between vehicle and RSU is based on the IEEE 802.11p, and the communication parameters used in our simulations are summarized in Table I [24], [34]. The channel model is based on Rayleigh fading model.

TABLE I: Communication Parameters

Parameter	Value
Bandwidth of each RSU	10 MHz
Transmission power of each vehicle	24dBm
PHY header	64 μs
MAC header	43 μs
ACK	101 μs
Slot (σ)	13 μs
SIFS	32 μs
AIFS	58 μs
RTS	144 μs
CTS	120 μs
δ	2 μs
CW_{min}	15 slots
CW_{max}	1023 slots
The maximum backoff stage(V)	5
Retransmission limit (z)	4

To verify the performance of our joint algorithm JSCO, we introduce the following benchmark schemes,

- SO: The Selection Optimization scheme (SO) selects the best VEC server with maximal system utility for each vehicle, under a given computation resource and offloading ratio.
- CO: The Computation Offloading scheme (CO) jointly optimizes computation resource and offloading ratio, as in [14]. Herein, all vehicles offload their tasks to the nearest VEC server as CO does not consider VEC server selection.
- BFS: The Brute Force Scheme (BFS) explores all cases of VEC server selections, feasible computation resource and offloading ratio, and then chooses the VEC server with maximal system utility for each vehicle, which finds the global optimum.

B. Convergence

In this subsection, we evaluate the convergence of the proposed JSCO algorithm under different initial points. We first plot the convergence evolution of the inner loop of Algorithm 2, i.e. the loop from step 6 to step 9 in the proposed JSCO algorithm. From Fig. 2, we observe that the allocated computation resource on each vehicle can achieve converge within 10 iterations and it converges almost simultaneously under different initial points. The convergence of the proposed

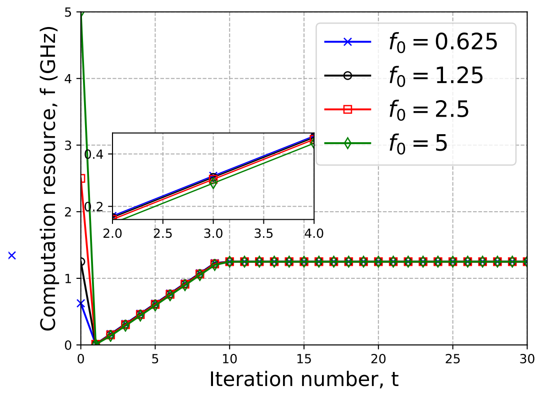


Fig. 2: Convergence of the computation resource allocation under different initial points with $N = 40, M = 5$.

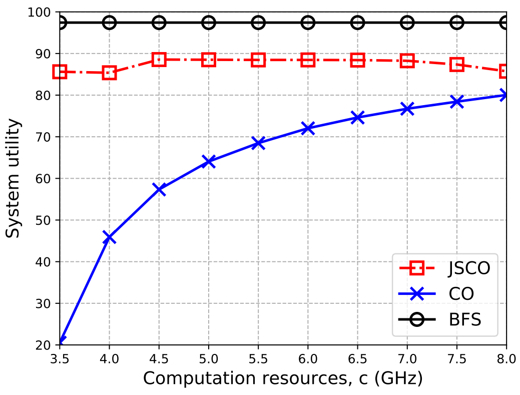


Fig. 4: Comparison of system utility of computation resources under different schemes with $N = 40, M = 5$.

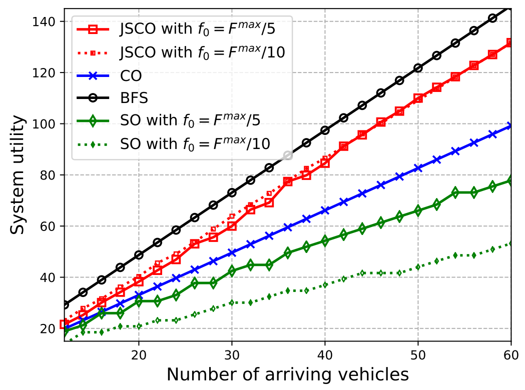


Fig. 3: Comparison of system utility of the number of arriving vehicles at the starting point under different schemes with $M = 5$.

JSCO algorithm is similar to the results of Fig. 2 so we omit it here. Based on the above results, we can conclude that the proposed JSCO algorithm for solving the hard problem P1 is usually efficient.

C. System Utility

In this subsection, we compare the performance of the proposed JSCO algorithm with other three schemes, i.e., SO, CO, and BFS.

Fig. 3 plots the comparison of system utility with respect to the number of arriving vehicles at the starting point. It also illustrates the performance of JSCO and SO under different amount of initial computation resource. From Fig. 3, we can draw several observations. First, the performance of the proposed JSCO significantly outperforms the two benchmark policies, CO and SO. The reason is that the proposed JSCO algorithm jointly optimizes VEC server selection, computation resource, and offloading ratio while CO only optimizes computation resource and offloading ratio, and SO only optimizes VEC server selection. Second, with the variety of the initial value of computation resource, JSCO presents a more stable performance compared to SO. This is because the proposed JSCO algorithm makes the optimization of computation resource such that the initial value of computation resource has less impact on the performance of the proposed JSCO algorithm. Further, the system utility gap between JSCO and CO becomes large as the number of vehicles increases, which is because CO does not consider VEC server selection. Therefore, when the number of arriving vehicles becomes large, CO results in a quite severe overload and low system utility as it just makes all vehicles offload their tasks to the nearest VEC server to process. However, with the optimization of VEC server selection, the proposed JSCO algorithm can not only optimize computation resource and offloading ratio but also balance the amount of vehicles on each server to improve system utility. In addition, BFS gets the optimal performance by enumerating all possible solutions. Although BFS is with the highest system utility, compared to JSCO, the time cost of BFS is very huge which is proportional to the number of candidate solutions.

Fig. 4 shows the comparison of system utility with respect to computation resources under different schemes. We can observe that the system utility of the proposed JSCO algorithm is very close to the optimal solution (i.e. the system utility of BFS). Further, the system utility of the proposed JSCO algorithm can keep a relatively high and stable value even when the computation resource is small. This is because, the proposed JSCO algorithm can make full use of the computation resource of all VEC servers, and not only the nearest VEC server. Therefore, when the computation capacity of the nearest VEC server is small and cannot support all vehicles' computation tasks, CO leads to an inferior performance but JSCO can shift the load of the nearest VEC server to other VEC servers and optimize the computation resource of other VEC servers thus keeping a high and stable system utility.

D. Load Balance

In this subsection, we focus on the performance on load balance under different schemes. Fig. 5 depicts the number of vehicles on each server. Here, we consider that VEC server 1 is the closest server, whereas VEC server 5 is the farthest. From Fig. 5, it can be observed that the proposed JSCO makes

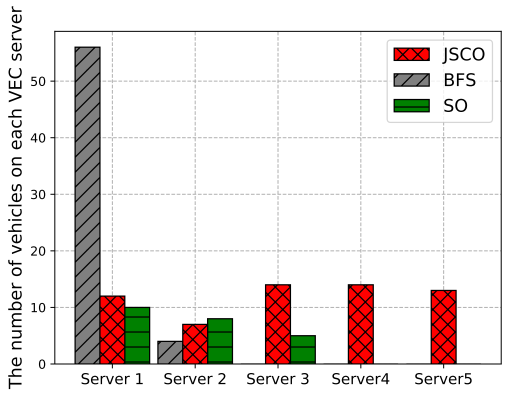


Fig. 5: Comparison of load balance under different schemes with $N = 60$ and $M = 5$.

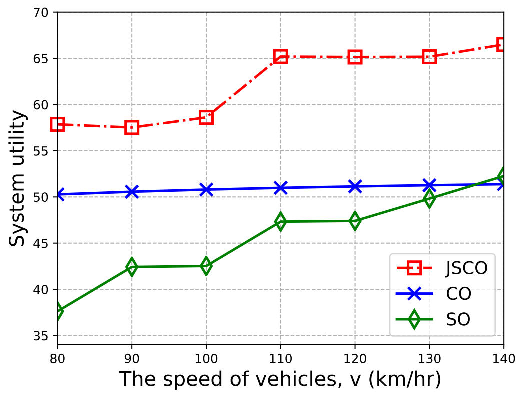


Fig. 6: Comparison of system utility under different schemes with $N = 40, M = 5$.

the most fair load allocation keeping the number of vehicles served by each VEC server close. Since BFS and SO greedily choose the VEC server with the maximal system utility, they lead to an unbalanced assignment of load. On the contrary, JSCO balances the number of vehicles served by each VEC server as much as possible by taking load balancing into VEC server selection and it keeps a higher system utility by jointly considering computation resource and offloading ratio.

E. Impact of Speed

In this subsection, we examine the impact of speed on system utility. Fig. 6 shows the comparison of system utility with respect to the speed of vehicles for the above the proposed JSCO, CO, and SO. From Fig. 6, we can see that the system utility of the proposed JSCO and SO increases with the increase in speed. A higher speed implies that a special vehicle will pass by more VEC servers in a certain timeslot. Thus, JSCO and SO can make a better VEC server selection as the set of available VEC server becomes large. Moreover, since JSCO takes the optimization of computation resource and offloading ratio into account while SO only considers VEC server selection, the system utility of JSCO is obviously better compared to SO. On the other hand, CO is not sensitive to the increment of speed because it does not consider VEC server selection.

VI. Conclusions

In this paper, we proposed a joint load balancing and offloading problem for maximizing system utility in vehicular edge computing network. We adopted a comprehensive task processing delay to formulate the system utility, which jointly considered transmission and computation time. A low-complexity algorithm, JSCO, was then developed, by jointly optimizing selection decision, offloading ratio, and computation resource. Numerical results demonstrated that the proposed JSCO not only significantly outperforms the benchmark policies in terms of system utility, but also performs well on load balance compared to the other counterparts. For the future work, we are going to consider a more general case that vehicles may arrive and depart dynamically at a unidirectional road. In this case, the mobility patterns might be generalized as a stochastic model in the problem formulation.

Appendix A
Proof of Lemma 3

It is clear that problem (22) is non-linear. Then, we check its convexity.

The second-order derivatives of $U(\mathbf{f}, \lambda)$ with respect to $\mathbf{f}$ and λ are

$$\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \mathbf{f}^2} = -\frac{2\alpha\, x_{ij}\lambda_{ij}c_i}{\ln(2) f_{ij}^3 \left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)} - \frac{\alpha\, {x_{ij}}^2{\lambda_{ij}}^2{c_i}^2}{\ln(2) f_{ij}^4 \left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)^2} \tag{38}$$

As $\alpha > 0$, $x_{ij}\lambda_{ij}c_i \geqslant 0$, and $\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right) > 0$, we have $\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \mathbf{f}^2} \leqslant 0$.

$$\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \lambda^2} = -\frac{\alpha\, {x_{ij}}^2 \left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right)^2}{\ln(2)\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)^2} \leqslant 0 \tag{39}$$

$$\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \mathbf{f}\partial \lambda} = \frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \lambda \partial \mathbf{f}} = \frac{\alpha\, x_{ij}c_i}{f_{ij}^2 \ln(2)\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)} + \frac{\alpha\, {x_{ij}}^2\lambda_{ij}c_i}{{f_{ij}}^2 \ln(2)\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right)\left(1+\beta - x_{ij}\lambda_{ij}\left(\frac{c_i}{f_{ij}}+\zeta_{ij}\right) - x_{ij}\Lambda_{ij}\right)^2} \geqslant 0 \tag{40}$$

Since the following inequation is indefinite, problem (22) is non-convex [35]

$$\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \mathbf{f}^2}\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \lambda^2}-\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \mathbf{f}\partial \lambda}\frac{\partial^2 U(\mathbf{f},\lambda)}{\partial \lambda\partial \mathbf{f}}=$$
$$2\,\frac{\alpha^2 {x_{ij}}^3 {c_i}^2 \zeta_{ij}\lambda_{ij}}{(\ln(2))^2\, {f_{ij}}^4\left(1+\beta-\frac{\lambda_{ij}c_i x_{ij}}{f_{ij}}-x_{ij}\lambda_{ij}\zeta_{ij}-x_{ij}\Lambda_{ij}\right)^3}$$
$$+\,2\,\frac{\alpha^2 {x_{ij}}^3 {\zeta_{ij}}^2 \lambda_{ij} c_i}{(\ln(2))^2\, {f_{ij}}^3\left(1+\beta-\frac{\lambda_{ij}c_i x_{ij}}{f_{ij}}-x_{ij}\lambda_{ij}\zeta_{ij}-x_{ij}\Lambda_{ij}\right)^3}$$
$$-\,\frac{\alpha^2 {x_{ij}}^2 {c_i}^2}{(\ln(2))^2\, {f_{ij}}^4\left(1+\beta-\frac{\lambda_{ij}c_i x_{ij}}{z}-x_{ij}\lambda_{ij}\zeta_{ij}-x_{ij}\Lambda_{ij}\right)^2} \tag{41}$$

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*, vol. 5, pp. 6757–6779, 2017.

[2] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[3] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[4] 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.

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

[6] X. Huang, R. Yu, J. Kang, and Y. Zhang, "Distributed Reputation Management for Secure and Efficient Vehicular Edge Computing and Networks," *IEEE Access*, vol. 5, 2017.

[7] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[8] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, 2013.

[9] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2016.

[10] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, 2016.

[11] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, 2017.

[12] 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.

[13] "Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 6: Wireless access in vehicular environments," *IEEE Stand.*, 2010.

[14] 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.

[15] 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.

[16] 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.

[17] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2016.

[18] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 4924–4938, 2017.

[19] Z. Tan, F. R. Yu, X. Li, H. Ji, and V. Leung, "Heterogeneous services provisioning in small cell networks with cache and mobile edge computing," [online]. Available: https://arxiv.org/pdf/1706.09542.pdf.

[20] 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.

[21] 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.

[22] Y. Zhang, R. Yu, S. Xie, W. Yao, Y. Xiao, and M. Guizani, "Home m2m networks: architectures, standards, and qos improvement," *IEEE Commun. Mag.*, vol. 49, no. 4, 2011.

[23] M. van Eenennaam, L. Hendriks, G. Karagiannis, and G. Heijenk, "Oldest packet drop (opd): a buffering mechanism for beaconing in ieee 802.11 p vanets," in *Proc. IEEE Veh. Netw. Conf. (VNC)*. IEEE Communications Society, 2011.

[24] C. Han, M. Dianati, R. Tafazolli, and R. Kernchen, "Throughput analysis of the ieee 802.11 p enhanced distributed channel access function in vehicular environment," in *Proc. Veh. Technol. Conf. Fall (VTC-Fall)*, 2010.

[25] 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.

[26] Q. Ye, O. Y. Bursalioglu, H. C. Papadopoulos, C. Caramanis, and J. G. Andrews, "User association and interference management in massive mimo hetnets," *IEEE Trans. Commun.*, vol. 64, no. 5, pp. 2049–2065, 2016.

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

[28] L. Li, M. Pal, and Y. R. Yang, "Proportional fairness in multi-rate wireless lans," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2008.

[29] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[30] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education, 2006.

[31] 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.

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

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

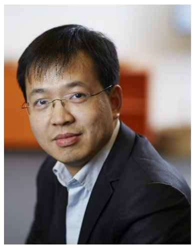
[34] I. . W. Group *et al.*, "802.11p-part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 6: Wireless access in vehicular environments," *IEEE Std*, vol. 802, no. 11, 2010.

[35] Y. Dauphin, H. de Vries, and Y. Bengio, "Equilibrated adaptive learning rates for non-convex optimization," in *Proc. Advances in Neural Information*. Curran Associates, Inc., 2015.

Yueyue Dai received the B.Sc. degree in communication and information engineering from the University of Science and Technology of China (UESTC), in 2014, where she is currently pursuing the Ph.D. degree. She is now a visiting Ph.D. student with the University of Oslo, Norway. Her current research interests include wireless network, mobile edge computing, Internet of Vehicles, blockchain, and deep reinforcement learning.

Du Xu is a professor at University of Electronic Science and Technology of China (UESTC) in Chengdu, China. He received a B.S., an M.S. and a Ph.D. from the South-East University and UESTC in 1990, 1995 and 1998, respectively. His research interests include network modelling and performance analysis, switching and routing, network virtualization and security. He presided over many advanced research projects, including NSFC, National 863 Plans and National key Research and Development Program of China.

Sabita Maharjan (M'09) received the Ph.D. degree in networks and distributed systems from the Simula Research Laboratory, and University of Oslo, Norway, in 2013. She is currently a Senior Research Scientist at the Simula Metropolitan Center for Digital Engineering, Norway, and an Associate Professor at the University of Oslo. Her current research interests include wireless networks, network security and resilience, smart grid communications, Internet of Things, machine-to-machine communication, software-defined wireless networking, and the Internet of Vehicles.



Yan Zhang is Full Professor at the Department of Informatics, University of Oslo, Norway. He received a PhD degree in School of Electrical & Electronics Engineering, Nanyang Technological University, Singapore. He is an Associate Technical Editor of IEEE Communications Magazine, an Editor of IEEE Network Magazine, an Editor of IEEE Transactions on Green Communications and Networking, an Editor of IEEE Communications Surveys & Tutorials, an Editor of IEEE Internet of Things Journal, an Editor of IEEE Vehicular Technology Magazine, and an Associate Editor of IEEE Access. He serves as chair positions in a number of conferences, including IEEE GLOBECOM 2017, IEEE VTC-Spring 2017, IEEE PIMRC 2016, IEEE CloudCom 2016, IEEE ICCC 2016, IEEE CCNC 2016, IEEE SmartGridComm 2015, and IEEE CloudCom 2015. He serves as TPC member for numerous international conference including IEEE INFOCOM, IEEE ICC, IEEE GLOBECOM, and IEEE WCNC. His current research interests include: next-generation wireless networks leading to 5G, green and secure cyber-physical systems (e.g., smart grid, healthcare, and transport). He is IEEE VTS (Vehicular Technology Society) Distinguished Lecturer. He is also a senior member of IEEE, IEEE ComSoc, IEEE CS, IEEE PES, and IEEE VT society.