Decentralized Convex Optimization for Joint Task Offloading and Resource Allocation of Vehicular Edge Computing Systems

Kaige Tan, *Member, IEEE*, Lei Feng, *Member, IEEE*, György Dán, *Senior Member, IEEE*, and Martin Törngren, *Senior Member, IEEE*

***Abstract*—Vehicular Edge Computing (VEC) systems exploit resources on both vehicles and Roadside Units (RSUs) to provide services for real-time vehicular applications that cannot be completed in the vehicles alone. Two types of decisions are critical for VEC: one is for task offloading to migrate vehicular tasks to suitable RSUs, and the other is for resource allocation at the RSUs to provide the optimal amount of computational resource to the migrated tasks under constraints on response time and energy consumption. Most of the published optimization-based methods determine the optimal solutions of the two types of decisions jointly within one optimization problem at RSUs, but the complexity of solving the optimization problem is extraordinary, because the problem is not convex and has discrete variables. Meanwhile, the nature of centralized solutions requires extra information exchange between vehicles and RSUs, which is challenged by the additional communication delay and security issues. The contribution of this paper is to decompose the joint optimization problem into two decoupled subproblems: task offloading and resource allocation. Both subproblems are reformulated for efficient solutions. The resource allocation problem is simplified by dual decomposition and can be solved at vehicles in a decentralized way. The task offloading problem is transformed from a discrete problem to a continuous convex one by a probability-based solution. Our new method efficiently achieves a near-optimal solution through decentralized optimizations, and the error bound between the solution and the true optimum is analyzed. Simulation results demonstrate the advantage of the proposed approach.**

***Index Terms*—Decentralized convex optimization, hierarchical decomposition, multi-server resource allocation, task offloading, vehicular Edge Computing.**

Manuscript received 18 December 2021; revised 28 May 2022; accepted 5 August 2022. Date of publication 9 August 2022; date of current version 19 December 2022. This work was supported in part by the TECoSA Vinnova Competence Center for Trustworthy Edge Computing Systems and Applications at KTH Royal Institute of Technology and in part by InSecTT (www.insectt.eu), funded by ECSEL Joint Undertaking (JU) under Grant 876038. The JU was supported in part by European Union's Horizon 2020 Research and Innovation Programme and in part by Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, and Turkey. The review of this article was coordinated by Dr. Ming Li. *(Corresponding author: Lei Feng.)*

Kaige Tan, Lei Feng, and Martin Törngren are with the Division of Mechatronics, KTH Royal Institute of Technology, 10044 Stockholm, Sweden (e-mail: kaiget@kth.se; lfeng@kth.se; martint@kth.se).

György Dán is with the Division of Network and Systems Engineering, KTH Royal Institute of Technology, 10044 Stockholm, Sweden (e-mail: gyuri@kth.se).

Digital Object Identifier 10.1109/TVT.2022.3197627

I. Introduction

THE substantial increase in the number of connected vehicles and the latest advances in autonomous driving lead to the emergence of various services and applications in the intelligent transportation system, such as online path planning, real data playback, localization, and perception [1]. These high-complexity applications demand extraordinary computation capacities, but resource-constrained vehicles may not be capable of serving the ever-increasing computational needs of new applications within their latency deadlines [2]. Driven by the evolution of wireless communication, Vehicular Edge Computing (VEC) systems, supported by Mobile Edge Computing (MEC)[1] [4], [5], are recognized as a promising paradigm in the development of vehicular networks. Owing to the close proximity to vehicles [6], VEC systems can provide computing services in Roadside Units (RSUs) with reduced end-to-end transmission delays.

To facilitate VEC in accelerating task completion and saving energy, the development of a vehicle computation offloading policy is crucial. Existing studies focus on the design of optimal offloading strategies to meet different performance requirements, such as low latency [7], high energy efficiency [8], and load balancing [6]. There are mainly two challenges for task offloading: the offloading decision, which determines where the task is to be executed, and the resource allocation, which characterizes how much computation and communication resources are allocated to the tasks. The formulation of this problem arises naturally in an intricate structure, which makes it challenging to obtain an optimal solution, especially when the number of variables grows exponentially in a high-dimensional scenario.

To address this challenge, researchers have proposed many different solution approaches lately. The problem is formulated as a constrained optimization problem in [9], [10], [11], [12], [13] to minimize the offloading delay. To investigate a holistic offloading solution in a multi-server MEC-assisted network, Tran et al. [10] decompose the original problem and find the resource allocation solution by quasi-convex optimization techniques, where the task offloading problem is tackled by the proposed heuristic approach. Tang et al. [12] address the energy-constrained delay minimization problem, which is solved using

[1]We refer to MEC as Mobile Edge Computing in this study, while a more recent interpretation of MEC is Multi-access Edge Computing [3].

the decision tree and dynamic programming. The total network delay is emphasized in [13], where a Lyapunov optimization is used for the development of an online multi-decision making algorithm. Besides the above works that apply optimization techniques, studies on vehicular offloading policy also exploit Reinforcement Learning (RL) algorithms [14], [15], [16], [17], [18]. For instance, Qi et al. [17] consider the data dependency in multiple tasks and apply the deep RL algorithm to find the long-term optimal offloading policy. To minimize the processing delay in VEC networks, Guo et al. [18] design an intelligent task offloading scheme based on deep Q learning, which is a centralized approach that requires all vehicles' information to be collected at a central RSU.

In the aforementioned studies, the vehicle plays the role only of a service client, where the offloading strategy is calculated and determined at the edge device. When the problem is both formulated and solved in a centralized way, the vehicles must send task parameters to the RSU and wait for the offloading decisions returned from the RSU. The potential issues, in terms of additional communication delay, increased computing complexity, and security issues caused by the information exchange, are not fully addressed yet. As an alternative, a decentralized offloading policy is desired.

Recently, a few studies have also focused on exploiting the benefits of decentralized computation offloading in VEC systems [19], [20], [21], [22], [23]. An adaptive learning-based task offloading algorithm is proposed in [19] based on multi-armed bandit theory. It works in a distributed manner and minimizes the average offloading delay. The vehicle-to-vehicle communication is considered in [20], where a decentralized resource allocation mechanism is proposed based on deep RL, and the global information is not required for each vehicle to make its decisions. The consensus ADMM-based energy-efficient resource allocation algorithm is proposed in [21], where the formulated joint problem is decomposed into a set of subproblems and solved in parallel. Jošilo et al. [22] develop a game-theoretical model and allow users to make offloading decisions autonomously. Liu et al. [23] design a user-centric control policy to optimize both delay and energy consumption by formulating the problem as a fully decentralized multi-agent Markov decision process.

Most of the existing decentralized solutions rely on a trained deep RL model or reach global coordination through an iterative way, which still requires high computational power or synchronous updates among vehicles when implemented in a real-time offloading application. In the context of vehicular offloading, the high computation power requirements call for effective energy management at the RSUs, which is an issue not emphasized widely. Most existing studies concentrate on reducing energy consumption at the vehicle but ignore the corresponding analysis at the edge [24]. Some recent studies [25], [26], [27] have investigated this topic by considering the optimization of both vehicles and RSUs. They optimize the energy consumption and the execution time of the holistic vehicular services, including vehicles, RSUs, and base stations. These works justify the need for optimizing both the vehicles and RSUs. In addition, because of the high mobility of vehicles and limited coverage of the RSUs, the vehicle cannot select offloading destinations arbitrarily. To address these problems, we develop a decentralized convex optimization approach that decomposes a holistic Mixed-integer Nonlinear Problem (MINLP) into a hierarchy of convex optimization problems. The decomposition is obtained by the dual decomposition and the probability-based offloading policy.

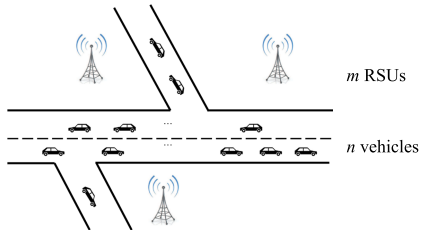

Fig. 1. Illustration of the offloading scenario.

The main contributions are summarized as follows:

- A decentralized task offloading and resource allocation problem in a multi-server VEC system is formulated as an optimization problem. The optimization criteria include the total latency of all tasks and the energy consumption in both vehicles and RSUs.
- A hierarchical decomposition approach is designed to break down the original MINLP into a group of convex subproblems for optimal resource allocation. These subproblems have low complexity and can be efficiently solved at the vehicle side. The vehicles only receive broadcasting messages from RSUs, which enhances user privacy and reduces information exchange delay.
- A convexification procedure is presented to transform the discrete optimization problem for task offloading into a continuous convex one. The integer design variables of deterministic task offloading targets are replaced by probabilities for offloading targets.
- To examine the application of the proposed decomposition approach, we analyze two common RSU deployment scenarios. The task offloading and resource allocation methods are studied and evaluated in both scenarios.

The rest of this paper is organized as follows. In Section II, the system model is presented with the formulation of the joint task offloading and resource allocation problem. Section III describes the hierarchical decomposition approach and solves the resource allocation problem in a single RSU scenario. We extend the solution to the multi-RSUs scenario and investigate the task offloading problem for load forecast coordination in Section IV. The numerical performance evaluation of the proposed methods is given in Section V, and Section VI concludes this paper.

II. System Model

In this section, we present the system model of the VEC network. After that, a vehicle-edge task offloading and resource allocation problem is formulated.

A. Vehicular Computing System

As illustrated in Fig. 1, we consider a highway scenario with m RSUs and n vehicles. Let $\mathcal{M} = \{1, 2, \ldots, m\}$ denote the index set of RSUs and $\mathcal{N} = \{1, 2, \ldots, n\}$ denote the index set of vehicles. We use $j \in \mathcal{M}$ as the index of the RSU and $i \in \mathcal{N}$ as

TABLE I
DEFINITIONS OF NOTATION

Notation	Definition
$\mathcal{M}$	Set of RSUs, indexed by *j*, *k*
m	Number of RSUs
$\mathcal{N}$	Set of vehicles, indexed by i
n	Number of vehicles
ΔT	Task period and the longest acceptable delay
$\mathbf{r}_j$	Geographical location of RSU-*j*
$\mathbf{p}_i$	Geographical location of vehicle-*i*
$\mathcal{U}$	Set of drivable positions of an area
d_j	Coverage radius of RSU-*j*
L_i	Required transmission data of task from vehicle-i
C_i	Required computation resources of task from vehicle-i
ϵ	Delay ratio of context transfer among RSUs
Δd_i	Context transfer delay of task from vehicle-i
θ_j	Energy consumption coefficient of CPU on RSU-j
u	Scaling ratio of CPU frequency
f_j	CPU frequency of RSU-*j*
T^o_{ij}	Computation time at RSU-*j* to finish task from vehicle-i
E^o_{ij}	Computation energy at RSU-*j* to finish task from vehicle-i
p_i	Maximal transmission power of vehicle-*i*
h_{ij}	Channel power gain from vehicle-*i* to RSU-*j*
q_{ij}	Wireless transmission rate from vehicle-i to RSU-j
r	Percentage of maximal transmission power
ω	Communication variable after substitution
B_p	Pre-allocated channel bandwidth of RSU-vehicle link
δ_j	Power of noise at RSU-*j*
E^c_{ij}	Communication energy at vehicle-*i* to connect to RSU-j
ϕ_{ij}	Offloading decision variable
E^o_j	Computation energy consumption limit at RSU-*j*
η_i	Weighting parameter between latency and energy consumption
$\mathcal{V}_j$	Set of vehicles which offloads to RSU-j
$\mathcal{G}$	Graph with RSU nodes to represent the offloading network
$\mathcal{A}$	Adjacency matrix associated to $\mathcal{G}$
a_{jk}	Estimated workload of vehicles driving from RSU-j to RSU-k
$\mathcal{V}_{jk}$	Set of vehicles which contribute to the workload a_{jk}
$\mathbf{P}$	Offloading probability matrix
ΔD_j	Total estimated context transfer delay at RSU-j

the index of the vehicle. The edge computing network consists of RSUs, and each RSU contains an MEC server. The server provides wireless radio access and computation resources to the vehicles. The computing tasks of the vehicles can be offloaded to an RSU so that the driving performance can be improved by reducing the task execution time. The notation used in this paper is summarized in Table I.

We consider that vehicles have to solve periodic tasks with the period ΔT. At every time step t, every vehicle $i \in \mathcal{N}$ generates a task. In the offloading system, we assume that the period ΔT is the longest acceptable delay to finish a task. The computing task from vehicle-*i* can be characterized by $[L_i, C_i]$, where L_i is the length of interactive transmission data (in bits) between vehicles and RSUs. We consider that the transmission data consist mainly of the computation instruction, and thus L_i is viewed as a constant. C_i represents the amount of computation resources (in CPU cycles) required to finish the task with the mean value of $\overline{C}_i$ [28].

In this study, a task is viewed as atomic and cannot be split [10]. Hence, a task cannot be computed partially on different computation nodes. Particularly, we focus on the type of computational resource-demanding tasks which have to be accomplished at the edge side [29], such as high-complexity motion planning and control modules. Further potential applications for offloading are, for example, given by cooperative driving automation scenarios [30]. Thus, we only consider the pure offloading case where a task is entirely executed on a single RSU.

B. Traffic Scenario Model

The traffic system in our study is a bidirectional road network. To describe the coverage of communication, we denote by $\mathcal{T} \subseteq \mathbb{R}^2$ the set of two-dimensional geographical locations of every drivable position in the system, and by $\mathbf{p}_i \in \mathcal{T}$ the position of vehicle-i. Locations of RSUs are denoted by $\mathbf{r}_j$. The Euclidean distance between two positions $\mathbf{p}_1$ and $\mathbf{p}_2$ is estimated by the square norm $\|\mathbf{p}_1 - \mathbf{p}_2\|$, $\mathbf{p}_1, \mathbf{p}_2 \in \mathcal{T} \cup \{\mathbf{r}_1, \ldots, \mathbf{r}_m\}$.

We define a circular area in the traffic system as $\mathcal{U}$, which can be viewed as the set of all drivable positions within the area. Suppose the coverage radius of RSU-j is d_j, $\mathcal{U}(\mathbf{r}_j, d_j) \triangleq \{\mathbf{p} \in \mathcal{T} | \|\mathbf{r}_j - \mathbf{p}\| \leq d_j\}$, which represents the set of positions on the road within the communication range of RSU-j. In this study, RSU-*j* is available for receiving a task from a vehicle only when its position is within $\mathcal{U}(\mathbf{r}_j, d_j)$. For simplicity, we consider that the *context transfer* among RSUs is well managed by proactive service migration [31]. We also assume that the vehicles' movements and the offloading workload in an area can be accurately estimated. Therefore, the context transfer delay across RSUs is considered to be known. The context transfer delay is assumed to be proportional to the task computation workload C_i and is estimated to be $C_i\epsilon$, for some $\epsilon > 0$[12]. Also, we define the expected offloading demand $\mathbb{E}(\sum_{\{i|\mathbf{p}_i \in \mathcal{U}(\mathbf{r}_j, d_j)\}} C_i)$ for the area. Note that the prediction method and task migration design are not the focus of this work but will be investigated in the future. State-of-the-art methods are, for instance, premigration [32] and mobility-based services migration prediction (MSMP) [33].

We assume that each drivable position on the road is covered by at least one RSU. If the distance between two RSUs is smaller than the sum of their coverage radii, the corresponding service areas will overlap. When solving the RSU deployment problem in similar scenarios, authors in [34] prove that, in terms of profit maximization, a non-overlapping solution performs no worse than an overlapping one. However, the non-overlapping solution does not consider the demand disparity under different traffic densities. Due to the heterogeneity of the RSU capacity, it may fail to handle the overload situation. To examine our approach in a more complex scenario with high workload, we thus also consider a deployment architecture with overlapped areas. More details of the scenario are given later in Section IV.

C. Computation Model

As mentioned in Section II-A, this study focuses on periodic tasks that cannot be accomplished onboard the vehicles, and should thus be offloaded to an RSU for execution. We consider that the computation capacity of the RSU is provided by a CPU supporting dynamic voltage and frequency scaling (DVFS) [19]. DVFS is a technique to adjust the frequency of a CPU for balancing the energy consumption and task execution time. In practice, the available frequency is restricted to a finite set of values $\mathcal{F} = \{f_{j1}, f_{j2}, \ldots, f_{jN}\}$, referred to as the available clock-frequency vector for RSU-*j*. Similar to [35], [36], [37], we assume that the difference between consecutive frequencies

is so small that the frequency can be approximately treated as a continuous variable. The solution can be viewed as a reference to the performance upper bound in realistic offloading policies. We denote by f_j the maximum nominal CPU frequency of RSU-*j*. Let $u_{ij} \in (0, 1]$ be the scaling ratio of f_j determined for vehicle-*i* to finish the task. Thus, the execution time is

$$T^o_{ij}(u_{ij}) = \frac{C_i}{u_{ij} f_j}, \tag{1}$$

where the superscript o denotes the computation part in the sequel.

An important aspect of computation is the energy consumption of task execution. We focus on the dynamic energy consumption of task execution [38]. Other energy consumption on the RSU, such as energy consumption at idle mode and cooling system, is assumed constant. The power consumption of a CPU is modeled as $\theta_j {f_j}^3$ per CPU cycle, where θ_j is the energy consumption coefficient depending on the chip architecture [8]. Then, the energy consumption of the task-i executed at RSU-*j* is

$$E^o_{ij}(u_{ij}) = \theta_j u^2_{ij} f^2_j C_i. \tag{2}$$

D. Communication Model

The vehicles can transmit data to the RSUs using wireless communications. We denote by p_i the maximal transmission power of vehicle, and by $r_{ij}p_i$ the actual transmission power used by vehicle-i for transmitting a task to RSU-j, where $r_{ij} \in (0, 1]$ is a decision variable. The decision variable r_{ij} allows the vehicle to trade off between energy consumption and transmission delay [23]. Let $h_{ij} = g(\|\mathbf{p}_i - \mathbf{r}_j\|)$ be the *channel gain* from vehicle-*i* to RSU-*j*. The channel gain includes path loss and fading, and is a function of the distance $\|\mathbf{p}_i - \mathbf{r}_j\|$ between the RSU-j and the vehicle-i [9]. We consider that Orthogonal Frequency Division Multiple Access (OFDMA) is used [22]. If a vehicle tries to communicate with an RSU, it occupies a fixed bandwidth B_p. If the system bandwidth is B_j, then at most $\hat{N}_j = \lfloor \frac{B_j}{B_p} \rfloor$ vehicles can offload to RSU-j. Given noise power δ_j, the *transmission rate* can be expressed as

$$q_{ij} = B_p \log_2 \left(1 + \frac{r_{ij} p_i h_{ij}}{\delta_j}\right). \tag{3}$$

We consider that the transmission delay from the RSU to the vehicle is negligible since the result of the computation is typically much smaller than the input data. Thus, given L_i, the transmission delay can be calculated as

$$T^c_{ij}(r_{ij}) = \frac{L_i}{q_{ij}}, \tag{4}$$

where the superscript c denotes the communication part in the sequel. If a context transfer occurs, the extra delay is $C_i\epsilon$.

The energy consumption of data transmission can be expressed based on the transmit power and the transmission delay as

$$E^c_{ij}(r_{ij}) = r_{ij} p_i T^c_{ij}(r_{ij}). \tag{5}$$

We do not account for the communication energy consumption on RSUs.

E. Problem Formulation

Our objective is to minimize the total task response times and communication energy at vehicles while satisfying the computation energy constraints at the RSUs. We focus on a single time step, as a building block for solving the finite horizon version of the problem, which we leave as the subject of future work. In what follows, we refer to *response time* as the total task completion time, including computation, communication, and context transfer delay. Note that the response time is used for simplicity sake, which does not rigorously follow the definition in the real-time scheduling theory [39], since the task preemption and blocking are omitted in this study.

We denote by $\boldsymbol{\phi}$ the offloading decision variables:

$$\phi_{ij} = \begin{cases} 1, & \text{if vehicle-}i \text{ offloads a task to RSU-}j \\ 0, & \text{otherwise.} \end{cases}$$

To estimate the possible context transfer delay, we consider the decision matrix $\boldsymbol{\phi}'$ at the next time step. Note that $\boldsymbol{\phi}'$ may not reflect the actual offloading decision, but is only used to check the offloading availability at the next time step based on the vehicle position. The fundamental assumption is that if vehicle-i offloads a task to RSU-j at the current step-t and is still within $\mathcal{U}(\mathbf{r}_j, d_j)$ at the next time step, then the vehicle maintains the same offload target, i.e., $(\phi_{ij} = 1) \wedge (\|\mathbf{r}_j - \mathbf{p}_i(t + \Delta T)\| \leq d_j) \Rightarrow \phi'_{ij} = 1$. Otherwise, it must start communication with a different RSU, i.e., $(\phi_{ij} = 1) \wedge (\|\mathbf{r}_j - \mathbf{p}_i(t + \Delta T)\| > d_j) \Rightarrow \phi'_{ij} = 0$. Under the latter situation, the context transfer occurs, and the additional communication delay must be counted. Since the road network is fully covered by all RSUs, there must be some $j' \neq j$ such that $\|\mathbf{r}_{j'} - \mathbf{p}_i(t + \Delta T)\| \leq d_{j'}$ and $\phi'_{ij'} = 1$. The exact value of j' is irrelevant for the optimization problem, because we count the context transfer delay for vehicle-i as $\Delta d_i = \sum_{j \in \mathcal{M}} C_i \epsilon \frac{|\phi_{ij} - \phi'_{ij}|}{2}$.

Thus, we can formulate the joint problem of task placement and resource allocation with the objective of minimizing the weighted sum of overall task response times and communication energy consumed at vehicles. η_i in the objective function is the weighting parameter used for the balance between delay and energy saving. The problem is defined in (**P0**):

$$(\mathbf{P0}) \min_{\mathbf{u}, \mathbf{r}, \boldsymbol{\phi}} \sum_{i=1}^{n} \left[\sum_{j=1}^{m} \phi_{ij} \left(T^o_{ij}(u_{ij}) + T^c_{ij}(r_{ij}) + \eta_i E^c_{ij}(r_{ij}) \right) + \Delta d_i \right] \tag{6a}$$

$$\text{s.t.} \quad \frac{\sum_{i=1}^{n} \phi_{ij} T^o_{ij}(u_{ij})}{\Delta T} \leq U_j, \ \forall j \tag{6b}$$

$$\sum_{i=1}^{n} \phi_{ij} E^o_{ij}(u_{ij}) \leq E^o_j, \ \forall j \tag{6c}$$

$$\sum_{j=1}^{m} \phi_{ij}(T_{ij}^{o}(u_{ij}) + T_{ij}^{c}(r_{ij})) \leq \Delta T - \Delta d_i, \ \forall i \tag{6d}$$

$$u_{ij}, r_{ij} \in [0,1], \ \forall i,j \tag{6e}$$

$$u_{ij} + r_{ij} = 0, \ \forall (i,j) \in \{\mathcal{N} \times \mathcal{M} \mid \phi_{ij} = 0\} \tag{6f}$$

$$\sum_{j=1}^{m} \phi_{ij} = 1, \ \forall i \tag{6g}$$

$$\phi_{ij} \in \{0,1\}, \ \forall i,j \tag{6h}$$

$$\sum_{i=1}^{n} \phi_{ij} \leq \hat{N}_j, \ \forall j \tag{6i}$$

$$\phi_{ij}(d_j - \|\mathbf{r}_j - \mathbf{p}_i(t)\|) \geq 0, \ \forall i,j, \tag{6j}$$

with variables $\mathbf{u} = \{u_{ij} | i \in \mathcal{N}, j \in \mathcal{M}\}$, $\mathbf{r} = \{r_{ij} | i \in \mathcal{N}, j \in \mathcal{M}\}$ and $\boldsymbol{\phi} = \{\phi_{ij} | i \in \mathcal{N}, j \in \mathcal{M}\}$.

In the set of constraints, (6b) states the total computation utilization on each RSU, where U_j represents the upper bounds on CPU utilization. Constraint (6c) regulates the energy consumption at the RSU, where E_j^o denotes the energy consumption limit at RSUs. (6d) guarantees the time constraint satisfaction while (6e) determines the ranges of the control variables. For each task, a vehicle can only have one RSU to offload. Thus, the sum of ϕ_{ij} over all RSUs should be 1, which is reflected in (6f) $\sim$ (6h). (6i) defines the upper limit for offloading, and (6j) examines the RSU service availability due to vehicle mobility.

III. Hierarchical Decomposition and Resource Allocation Problem Solving

Problem (**P0**) is nonlinear and has the binary variables ϕ_{ij}. It is an MINLP and generally difficult to solve. To address this issue, we propose a hierarchical decomposition approach in this section, which achieves a close-to-optimal solution that can be computed in a decentralized way.

Fig. 2 gives an overview of the hierarchical decomposition approach. The essential idea is to decompose the original problem into several decoupled subproblems, each of which is represented as a block in the figure. They have lower complexities and can be solved efficiently. All methods to perform the decomposition in this study are labeled in grey. The signals on the dashed lines connecting different blocks are solutions from the prior problems, which are used for coordination or problem-solving in sequential problems.

The domain of the variables of (**P0**) is huge and not continuous, and it requires the simultaneous exploration of three groups of decision variables. One way to simplify the problem is to search for the optimal solutions for the three groups of variables sequentially via the Tammer decomposition method [10]. If we first fix an arbitrary task offloading decision, then (**P0**) becomes an optimization problem only with free variables $\mathbf{u}$ and $\mathbf{r}$. Note that the context transfer delay Δd_i is independent of $\mathbf{u}$ and $\mathbf{r}$.

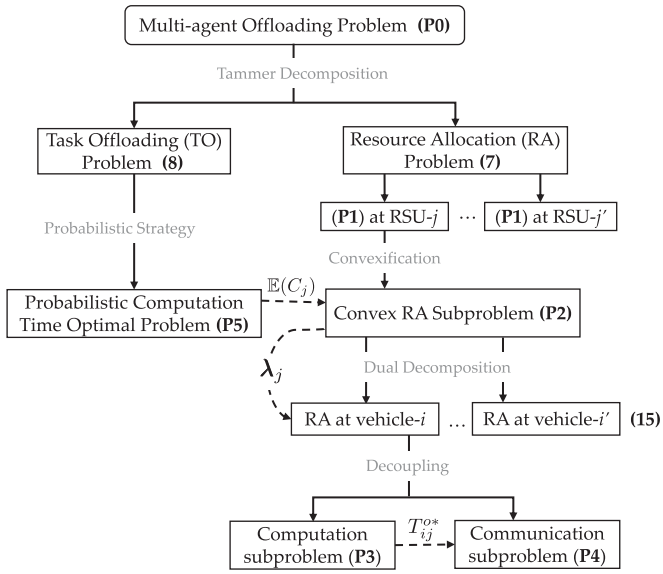

Fig. 2. Hierarchical decomposition for the task offloading and resource allocation problem.

Then (**P0**) is reduced to the subproblem

$$J^*(\boldsymbol{\phi}) = \min_{\mathbf{u},\mathbf{r}} \sum_{j=1}^{m} \sum_{i=1}^{n} \phi_{ij} \left[T_{ij}^{o}(u_{ij}) + T_{ij}^{c}(r_{ij}) + \eta_i E_{ij}^{c}(r_{ij})\right] \tag{7}$$

$$\text{s.t.} \quad (6b) \sim (6f).$$

Note that if the given task offloading decision $\boldsymbol{\phi}$ is infeasible, then no solution of $\mathbf{u}$ and $\mathbf{r}$ may satisfy all constraints of (6b) $\sim$ (6f). In that case, let $J^*(\boldsymbol{\phi}) = \infty$. Since the reduced optimization problem (7) has only $\mathbf{u}$ and $\mathbf{r}$ as free variables, it is called the resource allocation (RA) problem.

Subsequently, the searching for the optimal task offloading decision is named the task offloading (TO) problem

$$\min_{\boldsymbol{\phi}} \quad J^*(\boldsymbol{\phi}) + \sum_{i=1}^{n} \Delta d_i \tag{8}$$

$$\text{s.t.} \quad (6g) \sim (6j).$$

The decomposition of the overall problem (**P0**) into the TO and the RA problems is illustrated at the top of Fig. 2.

By the definition of the RA problem in (7), since the value of $\boldsymbol{\phi}$ is given, we can determine the set of vehicles offloading to each RSU-j as $\mathcal{V}_j = \{i \in \mathcal{N} \mid \phi_{ij} = 1\}$. According to constraint (6g), $\mathcal{V}_j \cap \mathcal{V}_{j'} = \emptyset$ if $j \neq j'$. Then the objective function of the RA problem in (7) becomes:

$$J^*(\boldsymbol{\phi}) = \sum_{j=1}^{m} \min_{\mathbf{u},\mathbf{r}} \sum_{i \in \mathcal{V}_j} \left[T_{ij}^{o}(u_{ij}) + T_{ij}^{c}(r_{ij}) + \eta_i E_{ij}^{c}(r_{ij})\right]. \tag{9}$$

The RA problem is equivalent to the subproblems of finding the optimal resource allocation at each RSU. The individual

problem at RSU-j is defined in (**P1**):

$$(\mathbf{P1}) \quad \min_{\mathbf{u},\mathbf{r}} \sum_{i\in\mathcal{V}_j} \left[T^o_{ij}(u_{ij}) + T^c_{ij}(r_{ij}) + \eta_i E^c_{ij}(r_{ij})\right] \tag{10a}$$

$$\text{s.t.} \quad \frac{\sum_{i\in\mathcal{V}_j} T^o_{ij}(u_{ij})}{\Delta T} \le U_j \tag{10b}$$

$$\sum_{i\in\mathcal{V}_j} E^o_{ij}(u_{ij}) \le E^o_j \tag{10c}$$

$$T^o_{ij}(u_{ij}) + T^c_{ij}(r_{ij}) \le \Delta T'_i, \ \forall i \tag{10d}$$

$$0 < u_{ij} \le 1, \ \forall i \tag{10e}$$

$$0 < r_{ij} \le 1, \ \forall i. \tag{10f}$$

In (**P1**), we simplify the problem by removing $\boldsymbol{\phi}$-related elements. Since only one RSU-j is considered, dimensions of variables are decreased, and $\mathbf{u}, \mathbf{r} \in (0,1]^{|\mathcal{V}_j|}$. Time constraint is replaced by $\Delta T'_i$ in (10d) to compensate context transfer delay with: $\Delta T'_i = \Delta T - \Delta d_i$. Once the decision variable $\boldsymbol{\phi}$ is given, the corrected values $\Delta T'_i$ are known at the vehicles. As illustrated in Fig. 2, the RA problem for the entire transportation system is reduced to the decentralized RA problems at individual RSUs.

Note that (**P0**) and (**P1**) suffer a common disadvantage that they are formulated and solved in a centralized manner, which requires the entire observation of all vehicles connected to the same RSU. Such an approach allocates the decision algorithms at the RSUs, where they acquire the detailed task information from each vehicle and then calculate the solution. This introduces an extra communication burden and may cause security and privacy concerns by its nature. Moreover, the optimization problem may become intractable for large-scale systems since an excessive delay is included. Thus, in Section III-B, we formulate a decentralized strategy to overcome this issue via dual decomposition.

A. Convexification of the RA Problem

The objective function and constraints in (**P1**) are continuous and twice differentiable in their domains. However, by the properties of quasiconvexity (see, e.g., [40] Section 3.4.2), it can be found that $E^c_{ij}(r_{ij})$ in (10a) is quasilinear but not convex. In order to hold convexity, we can transform the problem by substituting the variable r_{ij}. Note that the transmission rate q_{ij} is always a non-zero value, which ensures the feasibility of the substitution. Let us introduce ω_{ij}, where

$$\omega_{ij} = \frac{1}{q_{ij}} \ \Rightarrow \ r_{ij} = \frac{\delta_j(2^{\frac{1}{B_p\omega_{ij}}} - 1)}{p_i h_{ij}}. \tag{11}$$

Then problem (**P1**) can be rewritten as

$$(\mathbf{P2}) \quad \min_{\mathbf{u},\boldsymbol{\omega}} \sum_{i\in\mathcal{V}_j} \left[T^o_{ij}(u_{ij}) + T^c_{ij}(\omega_{ij}) + \eta_i E^c_{ij}(\omega_{ij})\right] \tag{12a}$$

$$\text{s.t.} \quad (10\text{b}), (10\text{c}), (10\text{e}),$$

$$T^o_{ij}(u_{ij}) + T^c_{ij}(\omega_{ij}) \le \Delta T'_i, \ \forall i \tag{12b}$$

$$1 \le 2^{\frac{1}{B_p\omega_{ij}}} \le Z_{ij}, \ \forall i, \tag{12c}$$

with variables $\mathbf{u} = \{u_{ij} | i \in \mathcal{V}_j\}, \boldsymbol{\omega} = \{\omega_{ij} | i \in \mathcal{V}_j\}$. In constraint (12c), $Z_{ij} = \frac{p_i h_{ij} + \delta}{\delta}$.

Lemma 1: (**P2**) is a convex optimization problem.

Proof: Please refer to Appendix A. ■

B. Decomposition of the RA Problem

In (**P2**), the objective function (12a) can be viewed as the sum of the objectives of all vehicles. Also, each vehicle has its own constraints in (10e), (12b), and (12c). However, constraints (10b) and (10c) are coupled among different vehicles, in which the restrictions on CPU utilization and energy consumption can be viewed as the aggregate amount on RSU-j. These coupled constraints hinder us from solving the optimization problem for individual vehicles. To tackle this issue, we form the dual problem by introducing the Lagrange variables $\boldsymbol{\lambda} = (\lambda_1, \lambda_2) \in \mathbb{R}^2_+$ for the inequality constraints (10b) and (10c). For RSU-j, we denote by $\boldsymbol{\lambda}_j$ the Lagrange variables. This results in the Lagrangian function:

$$L(\mathbf{u}, \boldsymbol{\omega}, \boldsymbol{\lambda}_j) = \sum_{i\in\mathcal{V}_j} \left[T^o_{ij}(u_{ij}) + T^c_{ij}(\omega_{ij}) + \eta_i E^c_{ij}(\omega_{ij})\right]$$
$$+\lambda_{j1}\left[\frac{\sum_{i\in\mathcal{V}_j} T^o_{ij}(u_{ij})}{\Delta T} - U_j\right] + \lambda_{j2}\left[\sum_{i\in\mathcal{V}_j} E^o_{ij}(u_{ij}) - E^o_j\right]. \tag{13}$$

Correspondingly, the dual function $g(\boldsymbol{\lambda}_j)$: $\mathbb{R}^2_+ \to \mathbb{R}$, as the infimum value of the Lagrangian function over $\mathbf{u}, \boldsymbol{\omega}$ for $\boldsymbol{\lambda}_j \in \mathbb{R}^2_+$, can be expressed as

$$g(\boldsymbol{\lambda}_j) = \inf_{\mathbf{u},\boldsymbol{\omega}} \left\{ L(\mathbf{u}, \boldsymbol{\omega}, \boldsymbol{\lambda}_j) \middle| (10\text{e}), (12\text{b}), (12\text{c}) \right\}. \tag{14}$$

The dual function can be evaluated separately on each vehicle with the Lagrangian variables. Moreover, the optimization objective comprises two parts: computation and communication. They have an independent structure of the objective and constraint functions, but are coupled by (12b) in (**P2**) as the time constraints. To analyze the problem appropriately, we can further break down the problem into single-commodity [41] for each vehicle. Assume that the optimal computation time of vehicle-i's task at the RSU-j is T^{o*}_{ij}, and $T^{o*}_{ij} < \Delta T'_i$ should always hold; otherwise (**P2**) does not have a feasible solution. Then, the communication delay must satisfy: $T^c_{ij}(\omega_{ij}) \le \Delta T'_i - T^{o*}_{ij}$.

With T^{o*}_{ij} as an auxiliary parameter, we can decompose the problem by separating the computation and the communication part of each vehicle. Accordingly, the dual function is

$$g(\boldsymbol{\lambda}_j) = \sum_{i\in\mathcal{V}_j} g^o_i(\boldsymbol{\lambda}_j) + \sum_{i\in\mathcal{V}_j} g^c_i, \tag{15}$$

where

$$g^o_i(\boldsymbol{\lambda}_j) = \inf_{u_{ij}} \left\{ T^0_{ij}(u_{ij}) + \lambda_{j1}\left[\frac{T^0_{ij}(u_{ij})}{\Delta T} - \frac{U_j}{|\mathcal{V}_j|}\right] \right.$$
$$\left. +\lambda_{j2}\left[E^0_{ij}(u_{ij}) - \frac{E^0_j}{|\mathcal{V}_j|}\right] \middle| 0 < u_{ij} \le 1 \right\}, \tag{16}$$

$$g_i^c = \inf_{\omega_{ij}} \left\{ T_{ij}^c(\omega_{ij}) + \eta_i E_{ij}^c(\omega_{ij}) \middle| \begin{matrix} T_{ij}^c(\omega_{ij}) \le \Delta T_i' - T_{ij}^{o*} \\ 1 \le 2^{\frac{1}{B\omega_{ij}}} \le Z_{ij} \end{matrix} \right\}. \tag{17}$$

$g_i^o(\boldsymbol{\lambda}_j)$ and g_i^c represent the dual functions of computation and communication subproblems at vehicle-*i*. Note that (17) is independent of the primal coupling constraints (10b) and (10c). By the definition, the corresponding subproblems are

$$(\mathbf{P3}) \ \min_{u_{ij}} \ T_{ij}^o(u_{ij}) + \lambda_{j1} \frac{T_{ij}^o(u_{ij})}{\Delta T} + \lambda_{j2} E_{ij}^o(u_{ij}) \tag{18a}$$

$$\text{s.t.} \ \ 0 < u_{ij} \le 1, \tag{18b}$$

and

$$(\mathbf{P4}) \quad \min_{\omega_{ij}} \quad T_{ij}^c(\omega_{ij}) + \eta_i E_{ij}^c(\omega_{ij}) \tag{19a}$$

$$\text{s.t.} \quad T_{ij}^c(\omega_{ij}) \le \Delta T_i' - T_{ij}^{o*} \tag{19b}$$

$$1 \le 2^{\frac{1}{B_p \omega_{ij}}} \le Z_{ij}. \tag{19c}$$

Problems (**P3**) and (**P4**) are computation and communication subproblems at the vehicle side. Note that the variation of $|\mathcal{V}_j|$ in (16) does not influence the optimal solution. Similar to (**P2**), both subproblems can be verified to be convex, and the duality gap is zero. Hence, the dual optimum obtained by (**P3**) and (**P4**) is equivalent to the primal solution of (**P2**), which enables us to solve the centralized primal problem through the decentralized dual subproblems.

Proposition 1: Given an RSU-j and n vehicles, when the optimal solutions of the dual subproblems in (**P3**) and (**P4**) at vehicles converge to the primal global optimum in (**P2**), the values of Lagrangian variables (i.e., $\boldsymbol{\lambda}_j \in \mathbb{R}_+^2$) are only influenced by the total amount of computation workload (i.e., $\sum_{i \in \mathcal{V}_j} C_i$) and energy constraints (i.e., θ_j and E_j^o), and:

$$\boldsymbol{\lambda}_j = (\lambda_{j1}, \lambda_{j2}) = \left(0, \sqrt{\frac{\theta_j (\sum_{i \in \mathcal{V}_j} C_i)^3}{4 (E_j^o)^3}} \right) \tag{20}$$

Proof: Please refer to Appendix B. ■

Remark 1: Proposition 1 reveals that the associated Lagrangian variable is only influenced by the energy-related parameters and the total amount of task load. Therefore, it enables an RSU to predict Lagrangian variables from the estimated total workload at the RSU.

Even with the closed-form expression of Lagrangian variables, the future workload $\sum_{i \in \mathcal{V}_j} C_i$ is a posteriori knowledge, which is only available after the optimal task offloading decision $\boldsymbol{\phi}$ has been determined. However, as shown in (8), the solution of the TO problem requires the solution of the RA problem. We need to break the circular dependency between the TO and the RA problems. In Section II-B, we assume that the offloading demand of an area can be accurately estimated. Therefore, we apply $\mathbb{E}(C_j)$ as the load forecast on RSU-j to replace $\sum_{i \in \mathcal{V}_j} C_i$ when deriving Lagrangian variables and calculating the corresponding objective function. It is given as

$$\boldsymbol{\lambda}_j = (\lambda_{j1}, \lambda_{j2}) = \left(0, \sqrt{\frac{\theta_j [\mathbb{E}(C_j)]^3}{4 (E_j^o)^3}} \right). \tag{21}$$

Note that in this paper, λ_{j1} is always treated as 0, since the existence of E_j^o implies a more stringent constraint on the task response time, and thus the schedulability limit is viewed as relaxed. If the value of E_j^o is unrealistically large, then λ_{j1} is not zero, and the time constraint becomes dominant. This extreme case is not explored in this study.

A major argument in Appendix B is (35): $\mathbf{u}^* = \frac{\mathbf{1}_{|\mathcal{V}_j|}}{(2\theta_j \lambda_{j2}^*)^{\frac{1}{3}} f_j}$, where $\mathbf{1}_{|\mathcal{V}_j|}$ is an $1 \times |\mathcal{V}_j|$ vector of all ones. The closed-form expression of u_{ij}^* is independent of the task load C_i. It shows that within the task period $[t, t{+}\Delta T)$, the RSU works with a fixed CPU frequency for all tasks from the connected vehicles. With (21) and (35), the optimal computation time $T_{ij}^o(u_{ij}^*)$ in (**P2**) is

$$T_{ij}^o(u_{ij}^*) = C_i \sqrt{\frac{\theta_j \mathbb{E}(C_j)}{E_j^o}}. \tag{22}$$

$T_{ij}^o(u_{ij}^*)$ in (22) corresponds to T_{ij}^{o*} introduced in (19b). Consequently, all terms in (**P4**) can be explicitly expressed. Since (**P4**) does not include Lagrangian variables, its optimal solution ω_{ij}^* can be determined independently by convex optimization. Therefore, we have shown that the optimization problems (**P3**) and (**P4**) can be efficiently solved with the given $\boldsymbol{\lambda}_j$. As depicted in Fig. 2, they give the fully decentralized RA solution for the convex RA subproblem (**P2**), and thus the RA subproblem (**P1**). Accordingly, the computing-intensive optimization at the edge can be circumvented by leveraging the unidirectional communication from RSUs to vehicles with the updated values of Lagrangian variables. Meanwhile, as the resource provider, the RSU coordinates vehicles by predicting and adjusting the values of Lagrangian variables.

In this section, the problem is restricted to the single RSU scenario with the RA solution. In what follows, we extend our solution by considering the multiple RSUs scenario and solve the top-level TO problem in (**P0**).

IV. Load Forecast for Task Offloading

In (21) and (22), the optimal computation resource allocation is influenced by $\mathbb{E}(C_j)$. Therefore, load forecast is essential to vehicles when making the offloading decision. However, even though we assume that the computational workloads of all vehicles in the transportation system can be estimated from a priori knowledge, we still cannot predict the estimated workload on each RSU, because some vehicles may have the flexibility of offloading to multiple RSUs. For instance, if vehicles offload with the pure greedy strategy, then all the vehicles in the service overlapped area will offload towards the RSU with the smallest predicted workload. Consequently, the selected RSU may quickly have a high workload or even become overloaded. All vehicular tasks offloaded to the RSU will suffer longer response

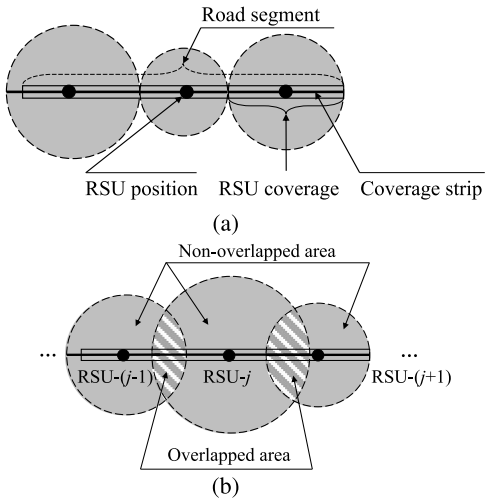

Fig. 3. Comparison between non-overlapped and overlapped architecture, the reference scenario is a straight road segment, adapted from [42].

times or even miss the deadline. The greedy offloading strategy is neither optimal nor stable.

This problem brings the challenge of reaching global coordination on load forecast in multiple RSUs scenarios. To accurately determine the expected offloading demand, as illustrated in Fig. 3, we consider the following two cases with different RSU deployment architectures, categorized based on whether the coverage of RSUs is overlapped or not [42].

A. Non-Overlapped RSUs Scenario

Consider a VEC system with m RSUs. Recall that each RSU is characterized by a deployment location $\mathbf{r}_j$ and a coverage range d_j. For the non-overlapped architecture, the RSU has its distinct service area, and each drivable position is covered by only one RSU, i.e., $\mathcal{U}(\mathbf{r}_{j_1}, d_{j_1}) \cap \mathcal{U}(\mathbf{r}_{j_2}, d_{j_2}) = \emptyset, \forall j_1, j_2 \in \mathcal{M}$. Under this condition, the vehicle only has one feasible offloading destination. Therefore, in non-overlapped scenarios, the original problem (**P0**) degenerates to m independent (**P1**), and the problem can be trivially decoupled. The RA problem can be estimated efficiently by solving (**P3**) and (**P4**), while the TO problem can be omitted. Meanwhile, since there are no overlapped areas, RSU-j can explicitly estimate its load within the covered area by

$$\mathbb{E}(C_j) = \mathbb{E}\left(\sum_{i \in \mathcal{S}} C_i\right), \tag{23}$$

where $\mathcal{S} = \{i \in \mathcal{N} | \, \mathbf{p}_i \in \mathcal{U}(\mathbf{r}_j, d_j)\}$.

The Lagrangian variables can be estimated with a high-accuracy load estimation system, and the proposed decentralized offloading strategy via decomposition can be achieved. Thus, with the non-overlapped deployment architecture, an optimal offloading solution can be realized.

B. Overlapped RSUs Scenario

In the non-overlapped scenario, the single available offloading destination within an area can be a bottleneck. The offloading availability of the RSU gets exacerbated by the increase in traffic density. Therefore, in order to provide alternative offloading solutions, we assume that the coverage of different RSUs can overlap. When an area is covered by several RSUs, the problem is quite complex in general when performing decentralized coordination. As shown in Fig. 3(b), we restrict the scenario that an area can be covered by at most two RSUs. Despite this simplification, the combinatorial structure of the problem is still challenging to solve in polynomial time [10]. The complexity of the exhaustive search is at most $\mathcal{O}(2^n)$, which is hard to perform by enumerating all possible solutions. Moreover, to tackle this problem, existing heuristic search algorithms (e.g., [9], [10]) require global information from vehicles and execute in an iterative manner, which is contradictory to our decentralized ideas. To analyze the problem, we make the following two assumptions:

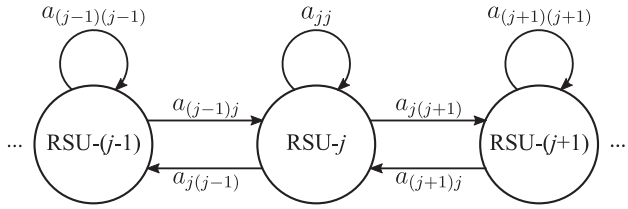

Fig. 4. Graph presentation of the overlapped scenario.

Assumption 1: An overlapped area has a similar distance to adjacent RSUs. For a vehicle in the overlapped area, the optimal values of (**P4**) to adjacent RSUs are similar. Thus, the offloading decision of vehicles in the overlapped area is dominated by the result of (**P3**).

Assumption 2: An overlapped area is located at the boundaries of two RSUs' coverage regions. For vehicles in overlapped areas, context transfer delay only occurs when offloading to the receding RSU.

Assumption 1 is reasonable because OFDMA is used in Section II-D as the communication feature with RSUs. With a fixed task and RA solution, (**P4**) only depends on the channel power gain, which is influenced by the distance. Note that Assumption 2 is also used in [12] for the position offset consideration, and is helpful in estimating the context transfer delay based on vehicle mobility.

As shown in Fig. 4, we use a directed graph $\mathcal{G}$ consisting of m RSU nodes to represent the offloading network considering the vehicle mobility. Let $\mathcal{V}_{jk}$ be the set of vehicles driving from RSU-j to RSU-k in the overlapped area of the two RSUs, a_{jk} be the estimated total computation workload of all these vehicles, i.e., $a_{jk} = \sum_{i \in \mathcal{V}_{jk}} C_i$. In addition, $\mathcal{A} = [a_{jk}] \in \mathbb{R}^{m \times m}$ is the adjacency matrix of $\mathcal{G}$ and a_{jk} is the weight of the edge from RSU-j to RSU-k. Thus, on the bidirectional road, a_{kj} is distinct from a_{jk} owing to driving directions. If RSU-k and RSU-j have a shared region, then $a_{jk}, a_{kj} \geq 0$, otherwise $a_{jk} = a_{kj} = 0$. The self-loop is denoted by a_{jj} and represents the workload in the exclusive areas.

Finding the binary task offloading decisions for vehicles in the overlapped areas has exponential complexity. To avoid the binary optimization problem, we propose a probabilistic task

offloading strategy. As shown in Fig. 2, the strategy determines the probabilities of allocating tasks to the adjacent RSUs, which helps with the calculation of $\mathbb{E}(C_j)$ to solve (**P2**). The advantage of this approach is to replace the original binary decision problem by a continuous optimization problem.

The task offloading decision is made by optimizing the TO problem in (8), where the cost function $J^*(\boldsymbol{\phi})$ is evaluated from the optimal values of (**P3**) and (**P4**). According to Assumption 1, the cost function $J^*(\boldsymbol{\phi})$ is primarily determined by (**P3**). Furthermore, the solution to (**P3**) is the optimal computation time of a task at an RSU, as estimated by (22). The equation implies that, for all vehicles in the same area, the optimal computation time of a task monotonically increases with the total workload at the connected RSU. In addition, Assumption 2 implies that the offloading probability is also related to the driving direction of the vehicle. Therefore, all vehicles in $\mathcal{V}_{jk}$ which are on the same edge in $\mathcal{G}$ should have an identical offloading probability distribution.

Thus, we denote by $\mathbf{P}=[P_{jk}] \in [0,1]^{m\times m}$ the offloading probability matrix, which has the same dimension as $\mathcal{A}$. The element P_{jk} denotes the probability for vehicles in $\mathcal{V}_{jk}$ offloading tasks to RSU-k. Correspondingly, the probability for these vehicles to offload tasks to RSU-j is $1-P_{jk}$. All elements on the main diagonal of $\mathbf{P}$ are 1, due to the unique offloading choice in the exclusive areas.

From the adjacency matrix $\mathcal{A}$ and the offloading probability matrix $\mathbf{P}$, the offloading workload on RSU-j can be shown as

$$\mathbb{E}(C_j) = \mathcal{A}(:,j)^\top \mathbf{P}(:,j) + \mathcal{A}(j,:)(\mathbf{1}_m - \mathbf{P}(j,:))^\top, \quad (24)$$

where $(:,j)$ and $(j,:)$ are the j-th column and row vectors of the matrix. To count context transfer delay Δd_i of all vehicles offloading to RSU-j, Assumption 2 yields the total estimated context transfer delay at RSU-j, denoted by ΔD_j, through the probability-based workload in overlapped areas:

$$\Delta D_j = \sum_{i\in[\mathcal{V}_{j(j-1)}\cup\mathcal{V}_{j(j+1)}]} \Delta d_i = \epsilon\, \mathcal{A}(j,:)(\mathbf{1}_m - \mathbf{P}(j,:))^\top. \quad (25)$$

We can further approximate the constraint (6i) on the maximal connected vehicles at an RSU by a new constraint on the maximal expected workload at the RSU with

$$\mathbb{E}(C_j) \le \overline{C}_i\, \hat{N}_j, \quad (26)$$

where $\overline{C_i}$ is the mean computational workload of vehicle tasks.

Combining the results in (22) and (24), the optimal computation time for all vehicles offloading on RSU-j, denoted by T_j^{o*}, can be estimated as

$$T_j^{o*} = \sum_{k\in\{j-1,j,j+1\}} \left[\sum_{i\in\mathcal{V}_{kj}} P_{kj} T_{ij}^o(u_{ij}^*) + \sum_{i\in\mathcal{V}_{jk}} [1-P_{jk}] T_{ij}^o(u_{ij}^*) \right] = \sqrt{\frac{\theta_j}{E_j^o}}\, [\mathbb{E}(C_j)]^{\frac{3}{2}}. \quad (27)$$

And the computation time for all vehicles in the system under optimization is

$$\sum_{j\in\mathcal{M}} T_j^{o*} = \sum_{j\in\mathcal{M}} \sqrt{\frac{\theta_j}{E_j^o}}\, [\mathbb{E}(C_j)]^{\frac{3}{2}}. \quad (28)$$

Equation (28) corresponds to the objective function of the computation part in (**P0**), and is determined by the offloading probabilities in the overlapped areas. From Assumption 1, with an optimal RA solution determined by (**P3**) and (**P4**), the TO problem in (8) can be transferred to derive the offloading probability distribution that provides the minimal task computation time plus the context transfer delay

$$(\mathbf{P5}) \;\; J(\mathbf{P}) = \min_{\mathbf{P}\in[0,1]^{m\times m}} \sum_{j\in\mathcal{M}} \left(\sqrt{\frac{\theta_j}{E_j^o}}\, [\mathbb{E}(C_j)]^{\frac{3}{2}} + \Delta D_j \right) \quad (29)$$

$$\text{s.t.} \quad (24) \sim (26).$$

Note that (**P5**) is not identical to (8) since the communication part is ignored. However, the focus of (**P5**) is to derive a probability-based offloading solution, which can be leveraged to estimate offloading workload $\mathbb{E}(C_j)$ through (24). And thus, the Lagrangian variables $\boldsymbol{\lambda}_j$ can be evaluated with (21), which contributes to the decentralized RA approach in Section III.

Problem (**P5**) can be solved at RSUs in a coordinated way before offloading periods. Its convexity can be confirmed similarly to Lemma 1 by verifying the Hessian being positive semidefinite. Thus, the proof is omitted for the sake of brevity. The value of $\mathbf{P}$ can be determined either by convex optimization, or by solving linear equations in the matrix form derived from the first-order optimality condition. Both methods are efficient and give vehicles in the overlapped areas a probability-based TO solution.

Proposition 2: The difference between the optimal solution $J(\mathbf{P}^*)$ of (29) using stochastic offloading and the optimal solution $J^*(\boldsymbol{\phi})$ of (8) using the binary offloading matrix is bounded by the following inequality

$$Pr\left(|J(\mathbf{P}^*) - J^*(\boldsymbol{\phi})| \ge \sum_{j\in\mathcal{M}} \text{var}(C_j)\sigma \sqrt{\frac{E_j^o}{\theta_j \mathbb{E}(C_j)}} \right) \le \frac{1}{\sigma^2}, \quad (30)$$

where $\text{var}(C_j)$ is the variance of estimated workload on RSU-j with the expression of $\text{var}(C_j) = \sum_{k\in\{j-1,j+1\}} [P_{kj}(1-P_{kj})a_{kj} + P_{jk}(1-P_{jk})a_{jk}]$, $\sigma > 1$.

Proof: Please refer to Appendix C. ■

Remark 2: Equation (30) gives a bound on the optimality error in our probability-based offloading approach. σ denotes the number of standard deviations away from the mean. For instance, when $\sigma = 2$, the probability-based offloading approach has more than a 75% chance of being within two standard deviations of the true optimal value $J^*(\boldsymbol{\phi})$. A general trend in the optimality error can be observed. Notice that $\text{var}(C_j)$ is affected by $\mathbf{P}$, which is correlated with delay ratio ϵ and workload distribution $\mathcal{A}$. With a higher value of ϵ and uneven distribution of workload in the exclusive areas (i.e., the variance of main diagonal elements

Algorithm 1: TO Coordination at RSU-j.

Data:Updated location of the vehicle $\mathbf{p}_i$
Result:Probability $\mathbf{P}(j,:)$, $\mathbf{P}(:,j)$ and Lagrangian variable $\boldsymbol{\lambda}_j$

1: Collect the updated adjacency matrix $\mathcal{A}$ through the offloading workload prediction;
2: Obtain offloading probability matrix $\mathbf{P}$ by solving (**P5**);
3: Update $\mathbb{E}(C_j)$ and $\boldsymbol{\lambda}_j$ through (24) and (20);
4: Broadcast $\mathbf{P}(j,:)$, $\mathbf{P}(:,j)$ and $\boldsymbol{\lambda}_j$ to the vehicles in $\mathcal{U}(\mathbf{r}_j, d_j)$;

Algorithm 2: TO and RA at Vehicle-i.

Data:Computation workload C_i, transmission data L_i
Result:Control variable ϕ_{ij}, u_{ij} and ω_{ij}

1: Collect the value of $\mathbf{P}(j,:)$, $\mathbf{P}(:,j)$ and $\boldsymbol{\lambda}_j$ from the adjacent RSUs;
2: Select the probability-based offloading action ϕ_{ij};
3: Obtain u_{ij} and T_{ij}^{o*} by solving (**P3**);
4: Obtain ω_{ij} by solving (**P4**);
5: Send requests to the corresponding RSU;

on $\mathcal{A}$), we can see that P_{jk} approaches either 0 or 1. Such a scenario gives a minor var(C_j), and our task approach tends to reach the optimal point more closely. Especially, when $P_{jk} = \{0,1\}, \forall j,k \in \mathcal{M}$, the overlapped scenario degenerates to the non-overlapped one described in Section IV-A. The influence of ϵ is also examined numerically in Section V-D.

C. Overall Solution Algorithm

Based on the analysis in the above sections, we summarize the overall solution and present the algorithms on vehicles and RSUs, respectively.

Algorithms 1 and 2 present the pseudo-code for the decentralized joint task offloading and resource allocation solution in one task period. The RSUs need to finish the coordination task in Algorithm 1 first and broadcast the updated information to the vehicles in their coverage areas, so that the vehicle can determine the task offloading and resource allocation action when the task arises. Note that all control variables are determined in Algorithm 2 at the vehicles, which enables the decentralized control with high efficiency.

V. Performance Evaluation

We simulate a stretch of a 1 km highway with 10 evenly distributed RSUs. Each RSU is equipped with 10 Nvidia Jetson TX2 NX modules[2] as the edge computing server [11]. The computing capacity of each module is 1 Gcycle/s, and the power consumption when fully utilized is 25 W. Based on these numbers, we can estimate the values of the corresponding parameters shown in Table II. We set the allocated communication bandwidth of each vehicle as $B_p = 2$ MHz, and the average noise power $\delta_j = 2 \times 10^{-13}$W. According to [15], the nominal channel gain is expressed by the free-space path loss model $\overline{h_{ij}} = \overline{g}(||\mathbf{p}_i - \mathbf{r}_j||) = A_d([3 \cdot 10^8]/[4\pi f_c||\mathbf{p}_i - \mathbf{r}_j||])^{d_e}$, where $A_d = 4.11$ as the antenna gain, $f_c = 915$ MHz as the carrier frequency, and $d_e = 2.8$ as the path loss exponent. The channel gains used in our model are then generated based on Rayleigh fading channel model as $h_{ij} = \overline{h_{ij}}\alpha$, where α is the independent random channel fading factor following an exponential distribution with unit mean.

TABLE II
Parameter Value Selection

Parameter	Value
Number of RSUs, m	10
Number of vehicles, n	[10, 80]
Required transmission data, L_i	1 Mbits
Average computation workload, $\overline{C}_i$	{1.5, 1.75, 2.0} Gcycles
Effective switched capacitance of CPU, θ_j	2.5×10^{-28}
CPU frequency of RSU-j, f_j	10 Gcycles/s
Maximal transmission power, p_i	400 mW
Allocated channel bandwidth to vehicle-i, B_p	2 MHz
Average power of noise at RSU-j, δ_j	2×10^{-13} W
Context transfer delay ratio, ϵ	0.05 s/Gcycles

For realistic evaluation, we adopt the real-time energy management control module proposed in [43], where the power and thermal management problem in a connected hybrid electric vehicle (HEV) system is investigated based on a model-based optimization approach. Based on the type of task in [43], the amount of data to be processed is known before starting the execution, and the amount of data to be transmitted is $L_i = 1$ Mbits. From the numerical results shown in [43], we estimated and selected the average required CPU cycle $\overline{C_i}$ of a task by three levels with {1.5, 1.75, 2.0} Gcycles, where each level represents a different planning horizon length. A random value in the range of $[-0.1\overline{C_i}, 0.1\overline{C_i}]$ is added to C_i to simulate the uncertain convergence times for the optimization task. Without loss of generality, the update of a motion planning iteration is set to $2\,s$, and it defines the task period and the completion deadline ΔT. Similar simulation settings can be found in [6], [19], [44], showing that the parameters defined in our task model are realistic and have practical relevance. According to [39], the limit on schedulability is approximately $U_j = 0.7$.

To evaluate the efficiency of the proposed **Decentralized Offloading** approach, we compare its performance against the following baselines.

1) **Random Offloading**: Each vehicle is randomly assigned an available offloading decision. Then each RSU independently performs optimization on communication and computation resources [45].
2) **Myopic Offloading**: Each vehicle is offloaded to the nearest RSU. Then each RSU independently performs resource optimization [9].
3) **Enumeration method**: An exhaustive search is performed on all possible offloading decisions to find the global optimum. Only scenarios with limited vehicles are

[2][Online]. Available: https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra%20Linux%20Driver%20Package%20Development%20Guide/power_management_tx2_32.html

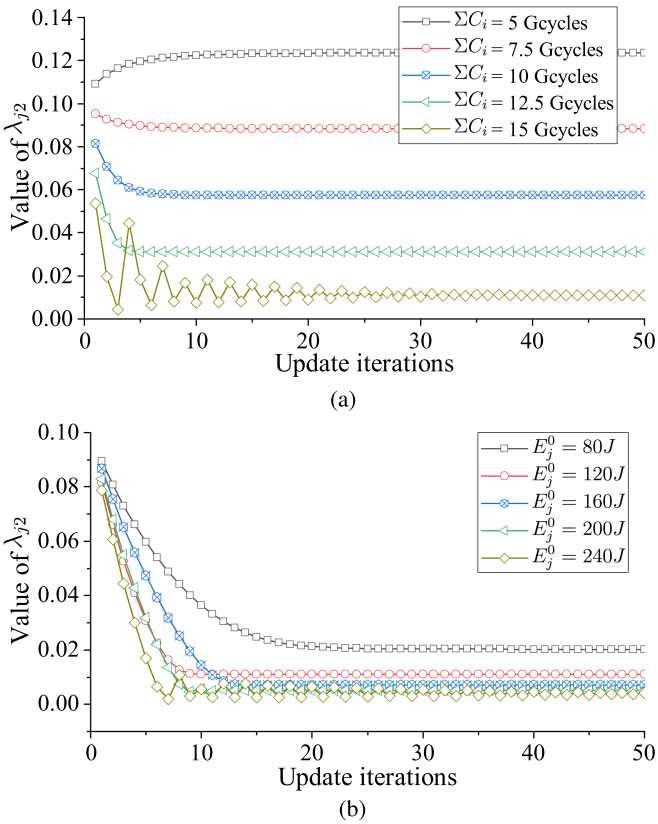

Fig. 5. Comparison of λ_{j2} under different conditions, $SS_{k_0} = 0.001$ when $E_j^o > 120$ J, otherwise $SS_{k_0} = 0.002$. (a) λ_{j2} with varied $\sum C_i$, $E_j^o = 40$ J. (b) λ_{j2} with varied E_j^o, $\sum C_i = 10$ Gcycles.

evaluated for this method due to its high computational complexity.

For each set of parameters, we randomly generate 50 traffic scenarios. Only overlapped scenarios are evaluated and shown, since the non-overlapped ones can be trivially decoupled. All the optimization algorithms in the case study were executed on a laptop PC with Intel(R) Core(TM) i5-8300H CPU @2.30 GHz, 4 cores, and 16.0 GB installed memory (RAM). After simulations, the average performance is shown.

A. Analysis of Lagrangian Variables

In Proposition 1, we study the Lagrangian variable and give a closed-form expression. To validate the correctness, we first compare values with the converged solution obtained by the primal-dual gradient method, which is often applied in the decomposition-based approach [46]. Note that our problem in (**P1**) can also be adequately solved with the gradient method. However, such approaches generally require a relatively long time to converge to high accuracy iteratively; hence, we utilize the primal-dual gradient method only for validation.

As shown in Fig. 5, when assigned with different energy constraints (i.e., E_j^o) and task loads (i.e., $\sum_{i=1}^n C_i$), the Lagrangian variable λ_{j2} varies. The initial value of λ_{j2} is set to 0.1 with the initial step size $SS_{k_0} = 0.002$. To better illustrate the figure, we

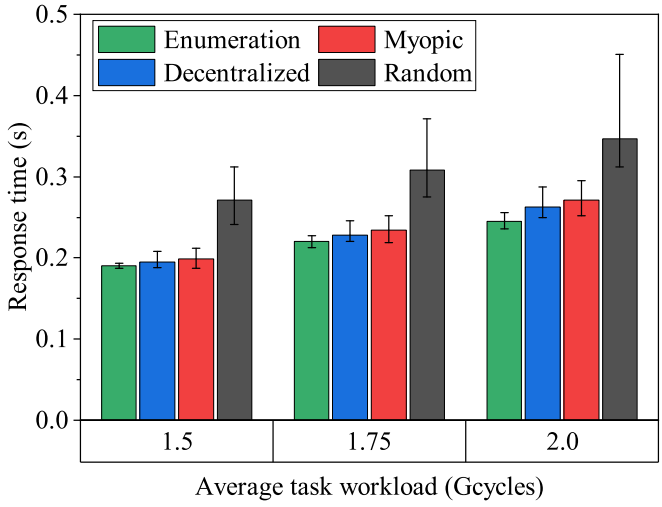

Fig. 6. Comparison of the response time per task for different approaches, with error bars representing the ranges of performance in 50 random scenarios and $E_j^o = 150$ J.

also halve the initial step size when $E_j^o > 120$ J to avoid oscillation and reach faster convergence. To guarantee the convergence, we apply a diminishing step size with $SS_k = SS_{k_0} k^{-0.5+\gamma}$, where k is the iteration number and $\gamma = 0.3$ is a positive constant. A larger γ gives a larger step size and may lead to stronger oscillation. The stopping criterion is based on ϵ-suboptimal [40], and the relative tolerance is set to $\epsilon_{rel} = 1 \times 10^{-4}$. Some of the curves have larger overshoots than others, mainly because the same initial value of λ_{j2} is applied. By examining the values from (20), it can be confirmed that our derived values coincide with the results from the primal-dual gradient method.

B. Optimization Result and Runtime Comparison

To evaluate the optimality of our proposed method, we compare its performance with other methods mentioned above. Since the **Enumeration method** searches all possible offloading decisions, and its runtime grows exponentially with the increase of vehicle number, the number of vehicles is set to 30, and the number in overlapped areas is limited up to 15. The response time is shown in Fig. 6. We respectively set $\overline{C}_i = 1.5$, 1.75 and 2.0 Gcycles, and report the task response time in 50 scenarios for each approach. With the increase of the task workload, the task response time becomes longer. Our proposed **Decentralized Offloading** performs closely to **Enumeration method**, which searches over all possible offloading decisions and has the best performance. Compared to **Myopic Offloading**, **Decentralized Offloading** achieves slightly shorter task response times in a decentralized way.

The average runtime per scenario finished by each algorithm is reported in Table III. The **Enumeration method** consumes the longest time, around 1000 times longer than **Decentralized Offloading**, even for this light traffic scenario. **Decentralized Offloading** runs faster than **Myopic Offloading**, mainly because the resource allocation step is performed decentralized in parallel at vehicles instead of at the RSU.

TABLE III
RUNTIME COMPARISON

Algorithm	Runtime (s)
Random Offloading	0.152
Decentralized Offloading	0.190
Myopic Offloading	0.432
Enumeration method	212.6

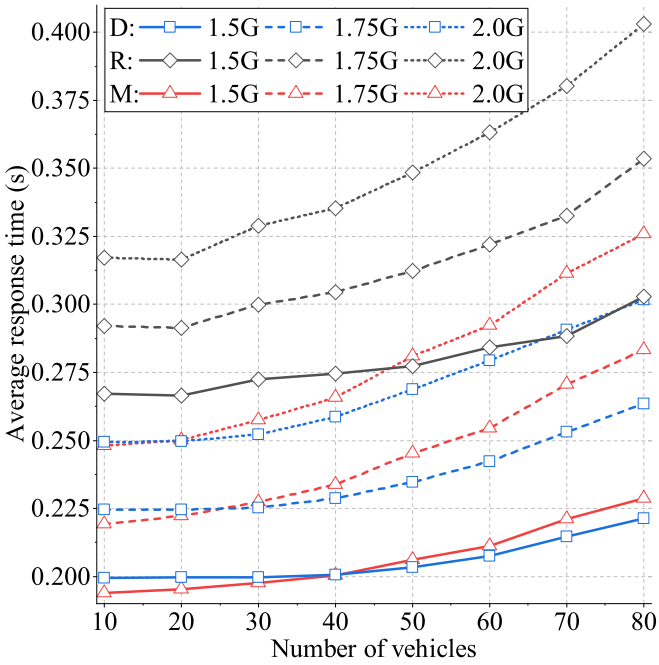

Fig. 7. Comparison of average response time per task with different numbers of vehicles, $E_j^o = 250$ J. The average workload $\overline{C}_i$ increases from 1.5 Gcycles to 2.0 Gcycles. (D: Decentralized, M: Myopic, R: Random).

C. Effect of the Number of Vehicles

Fig. 7 shows the offloading behavior with different numbers of vehicles. When the number of vehicles is less than 40, the average response times of **Decentralized Offloading** and **Myopic Offloading** differ slightly. In general, **Myopic Offloading** has an advantage over **Decentralized Offloading** when the number of vehicles is lower than 20, mainly because in light traffic situations, every vehicle can share adequate computing resources. In this regard, communication time has a more significant influence on the performance compared to computation time. **Myopic Offloading** offloads vehicle tasks to the nearest RSU; thus, the communication delay is minimized. However, when more vehicles offload under the same constrained computation resources, **Decentralized Offloading** has a better performance with a shorter completion time. Especially when the number exceeds 50, vehicles in overlapped areas can collaboratively select the offloading destination and prompt load balancing among RSUs.

Since the energy, computation, and communication resources are constrained in the above scenarios, with the increase of vehicles, RSUs cannot serve all tasks and satisfy the timing requirement simultaneously. Service outage [11] happens when a task cannot be completed by the selected RSU, which is

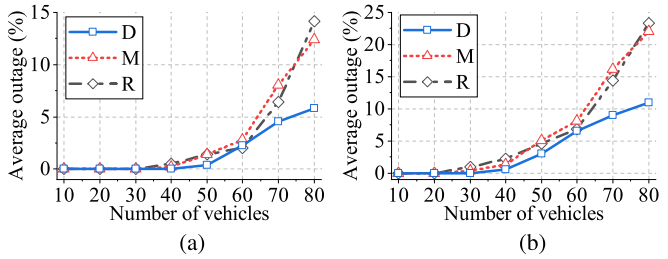

Fig. 8. Comparison of average service outage with different number of vehicles, with $E_j^o = 250$ J. (D: Decentralized, M: Myopic, R: Random). (a) $\overline{C}_i = 1.75$ Gcycles. (b) $\overline{C}_i = 2.0$ Gcycles.

evaluated by the number of failure cases over the total number of vehicles. We consider it as the indicator to assess the quality of service (QoS) loss. Fig. 8 shows the service outage with different numbers of vehicles. Scenarios with $\overline{C}_i = 1.75$ and 2.0 Gcycles are plotted as representatives. When the number of vehicles increases, RSUs fail to handle all tasks within the time constraint. The stochastic vehicle locations and unbalanced traffic distribution can exacerbate the problem. **Random Offloading** and **Myopic Offloading** have similar outage rates with the absence of demand coordination and balancing. With the load forecast coordination, **Decentralized Offloading** outperforms others significantly, especially when the traffic density is high. This shows that our approach can scale well with the number of vehicles.

D. Effect of RSU Capacity and User Preference

In this section, we fix the number of vehicles to 40, so that the service outage problem when the task workload is high can be mitigated. We try to examine the offloading performance with varied RSU capacities in terms of CPU frequency f_j and energy consumption constraint E_j^o. Results are checked under different task computation workloads $\overline{C}_i$. Two types of RSU configuration, *homogeneous* and *heterogeneous* servers [10], are evaluated and compared. In the homogeneous scenario, all servers have the same CPU speed of 10 Gcycles/s, while the CPU speeds in the heterogeneous scenario are randomly selected from $\{5, 10, 15\}$ Gcycles/s. With the change of CPU speed, energy constraint E_j^o is adjusted accordingly with $\{125, 250, 375\}$ J to keep the same level of stringency.

Average response times with the increase of $\overline{C}_i$ are shown in Fig. 9. The performance obtained from the homogeneous scenario is better than that in the heterogeneous scenario, mainly because the latter scenario has an unbalanced deployment of computation resources. Meanwhile, there is an increasing gap between **Decentralized Offloading** and others with the growth of task workload in both scenarios, and it is more apparent in the heterogeneous scenario. This is because when vehicles are close to RSUs with low computation capacities, **Myopic Offloading** selects the nearest one but **Decentralized Offloading** may choose a further RSU to balance the workload. Thus the latency performance gets improved.

The effect of user preference η_i in (6a) and context transfer delay ratio ϵ are also studied, and the results are shown in

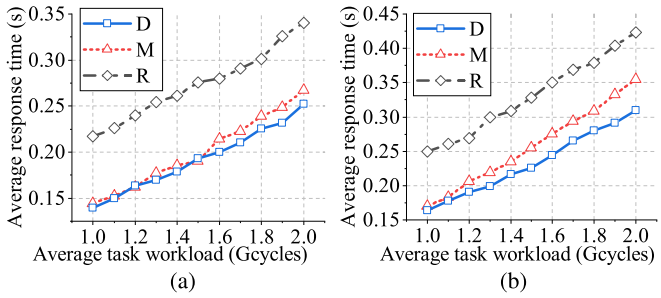

Fig. 9. Comparison of average response time per task with different task workloads, with $E_j^o = 250$ J and 40 vehicles. (D: Decentralized, M: Myopic, R: Random). (a) Homogeneous RSUs. (b) Heterogeneous RSUs.

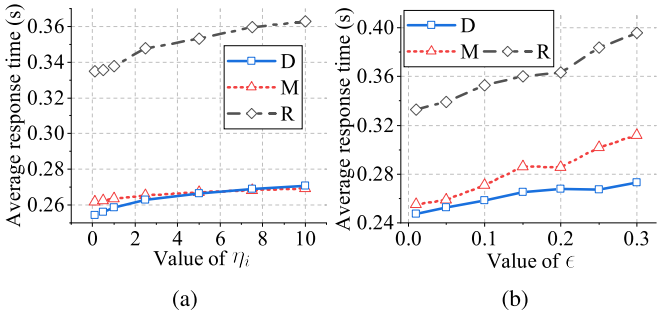

Fig. 10. Comparison of average response time with different η_i and ϵ, with $E_j^o = 250$ J, $\overline{C}_i = 2.0$ Gcycles and 40 vehicles. (D: Decentralized, M: Myopic, R: Random). (a) Comparison of average response time with different η_i. (b) Comparison of average response time with different ϵ.

Fig. 10. We vary the values of weighted-sum parameter η_i in Fig. 10(a). With the increase of η_i, the energy-saving demand on communication is emphasized; thus, the advantage of **Decentralized Offloading** over **Myopic Offloading** is decreased. Fig. 10(b) shows the impact of the context transfer delay ratio ϵ on the response time. Both **Myopic Offloading** and **Random Offloading** experience a near proportional increase with a higher ϵ, while **Decentralized Offloading** differs with a sub-linear growth. It nearly reaches stabilized after $\epsilon > 0.2$, mainly because the solution of the task offloading optimization problem tends to avoid offloading tasks to the receding RSU to reduce the chance of context transfer.

E. Effect of Energy Constraint

In Fig. 11, the influence of the energy constraint on optimization performance is analyzed. We fix the number of vehicles to 50 and $\overline{C}_i = 2.0$ Gcycles. The values on the horizontal axis decrease along the positive direction, indicating a more constrained scenario. Fig. 11(b) compares the average energy consumption at the RSUs, which is an indicator of load balancing at the edge devices. Since there are 10 RSUs, from (1) and (2), the average energy consumption per RSU is 250 J. In Fig. 11(b), when $E_j^o = 500$ J, all tasks can be executed with the maximal frequency (i.e, $u_{ij} = 1$), and the average energy consumption reaches 250 J for all three policies. When the energy constraint becomes more stringent, owing to the uneven workload distribution among RSUs, tasks cannot be equally

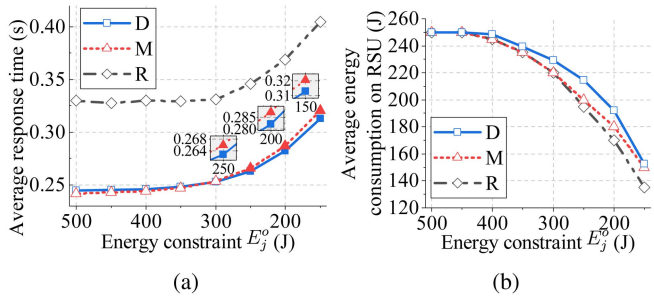

Fig. 11. Comparison of time consumption and actual energy consumption with different energy constraints, with $\overline{C}_i = 2.0$ Gcycles and 50 vehicles. (D: Decentralized, M: Myopic, R: Random). (a) Time consumption per task. (b) Energy consumption per RSU.

solved with the same frequency, and the average utilization gradually decreases. Among the three policies, **Decentralized Offloading** has the best load balancing performance due to the advantage in the task offloading. Fig. 11(a) evaluates the average task response time when the energy constraint varies. When the energy limit is high, all tasks can be computed with adequate resources, and the average response time is short. In such a scenario, **Myopic Offloading** performs slightly better than **Decentralized Offloading** because **Myopic Offloading** has a lower communication latency. However, when E_j^o gets smaller, **Decentralized Offloading** shows better performance in terms of load balancing with a more rapid response time.

VI. Conclusion and Future Work

This study investigates the optimal computation task offloading and resource allocation problem in vehicular edge computing systems. A decentralized strategy is proposed to minimize the overall response time while guaranteeing the deadline and energy limitations. The original MINLP is converted into tractable convex subproblems through hierarchical decomposition, and it enables the problem to be solved in a decentralized way at the vehicle side. Besides, we study the coordination problem among RSUs with two deployment architectures. A global coordination on load estimation in multiple RSUs scenarios is reached. The probability-based approach provides a near-optimal solution with high efficiency, and the simulation results verify that our approach outperforms baseline methods. In future work, the system's reliability will be investigated with the consideration of disturbances and stochastic behaviors. We will also investigate how to deal with offloading of safety-related tasks.

Appendix A
Proof of Lemma 1

In (**P1**), the non-convexity manifests due to $E_{ij}^c(r_{ij})$. After the substitution in Section III-A, r_{ij} is expressed by ω_{ij}. From (11), we have

$$E_{ij}^c(\omega_{ij}) = r_{ij} p_i T_{ij}^c = \frac{\delta_j L_i \omega_{ij} (2^{\frac{1}{B p \omega_{ij}}} - 1)}{h_{ij}}. \quad (31)$$

By taking the second-order derivative of $E_{ij}^c(\omega_{ij})$, we have

$$\frac{d^2 E_{ij}^c(\omega_{ij})}{d\,{\omega_{ij}}^2} = \frac{\ln^2 2\ \delta_j L_i}{h_{ij} {B_p}^2} \frac{2^{\frac{1}{B_p \omega_{ij}}}}{{\omega_{ij}}^3}. \tag{32}$$

From (10f), we can get the feasible range of ω_{ij}:

$$\omega_{ij} \in \left(0, \left(B_p \log_2\left(\frac{p_i h_{ij}}{\delta_j} + 1\right)\right)^{-1}\right]. \tag{33}$$

Since ω_{ij} is positive, we know (32) is always positive. Besides $E_{ij}^c(r_{ij})$, other r_{ij} related elements, such as $T_{ij}^c(r_{ij})$ in (10a) and (10d), will be linear to ω_{ij} after the substitution. Therefore, in (**P2**), the objective function and the inequality constraints are all convex. Thus, (**P2**) is a convex problem. It allows us to solve the primal via the dual.

Appendix B
Proof of Proposition 1

We focus on the computation subproblem (**P3**) since the communication part is not correlated with $\boldsymbol{\lambda}_j$. The convexity of (**P3**) can be examined similarly as in Lemma 1 by examining the Hessian as positive semidefinite. Let $\mathbf{u}^*$ be the optimal solution to the primal problem and $\boldsymbol{\lambda}_j^*$ be the optimal solution to the dual problem. The Karush-Kuhn-Tucker (KKT) conditions for the corresponding problem are

$$\frac{d(\sum_{i\in\mathcal{V}_j} T_{ij}^o(u_{ij}^*))}{d\mathbf{u}^*} + \lambda_{j1}^* \frac{d\left(\sum_{i\in\mathcal{V}_j} T_{ij}^o(u_{ij}^*)/\Delta T\right)}{d\mathbf{u}^*} + \lambda_{j2}^* \frac{d\left(\sum_{i\in\mathcal{V}_j} E_{ij}^o(u_{ij}^*)\right)}{d\mathbf{u}^*} = 0, \tag{34a}$$

$$(10b), (10c),$$

$$\lambda_{j1}^* \left(\frac{\sum_{i\in\mathcal{V}_j} T_{ij}^o(u_{ij}^*)}{\Delta T} - U_j\right) = 0 \tag{34b}$$

$$\lambda_{j2}^* \left(\sum_{i\in\mathcal{V}_j} E_{ij}^o(u_{ij}^*) - E_j^o\right) = 0. \tag{34c}$$

Equations (34b) and (34c) state the complementary slackness conditions. In our problem, the energy constraint is lower than the nominal power, which means it is always tighter than the utilization constraint. Thus, the energy consumption will exceed E_j^o before the utilization reaches the maximal threshold U_j. We first assume the strong inequality holds for (10b) by considering it as an inactive constraint. Since it does not bind, we can conclude that $\lambda_{j1}^* = 0$. Meanwhile, since (10c) is an active constraint, to satisfy complementary slackness conditions, we have: $\sum_{i\in\mathcal{V}_j} E_{ij}^o(u_{ij}) - E_j^o = 0$. From (34a), we can obtain

$$\mathbf{u}^* = \frac{\mathbf{1}_{|\mathcal{V}_j|}}{(2\theta_j \lambda_{j2}^*)^{\frac{1}{3}} f_j}. \tag{35}$$

Since $u_i \in (0, 1], \forall u_i \in \mathbf{u}$, we have

$$0 < \frac{\mathbf{1}_n}{(2\theta_j \lambda_{j2}^*)^{\frac{1}{3}} f_j} \leq 1 \ \Rightarrow\ \lambda_{j2}^* > \frac{1}{2\theta_j {f_j}^3}. \tag{36}$$

Also, to make sure our assumption above holds, $\mathbf{u}^*$ should satisfy the strong inequality in (10b):

$$\sum_{i\in\mathcal{V}_j} \frac{C_i}{u_{ij}^* f_j} < U_j \Delta T \ \Rightarrow\ \lambda_{j2}^* < \frac{1}{2\theta_j}\left(\frac{U_j \Delta T}{\sum_{i\in\mathcal{V}_j} C_i}\right)^3. \tag{37}$$

Therefore, the dual subproblem can be expressed as

$$\max_{\lambda_{j2}}\ g(\lambda_{j2}) = 3 \cdot (2^{-\frac{2}{3}}) \sum_{i\in\mathcal{V}_j} C_i (\theta_j \lambda_{j2})^{\frac{1}{3}} - E_j^o \lambda_{j2}$$
$$\text{s.t.}\quad (36), (37), \tag{38a}$$

with variable $\lambda_{j2} \in \mathbb{R}_+$. Thus, the optimal dual variable can be found by: $\nabla g(\lambda_{j2}^*) = 0 \ \Rightarrow\ \lambda_{j2}^* = \sqrt{\frac{\theta_j (\sum_{i\in\mathcal{V}_j} C_i)^3}{4 (E_j^o)^3}}$.

In our problem, to meet the schedulability requirement, the total computation task cannot exceed an upper bound of: $\sum_{i\in\mathcal{V}_j} C_i < [E_j^o (U_j \Delta T)^2 \theta_j^{-1}]^{\frac{1}{3}}$, which is the maximal capacity of an RSU under the energy constraint. Also, (36) gives a lower bound of λ_{j2}. The satisfaction of both constraints is examined in Section V.

Appendix C
Proof of Proposition 2

In the graph $\mathcal{G}$, RSU-j connects to RSU-(j-1) and RSU-(j+1). Expand (24) with only the nonzero elements, we have

$$\mathbb{E}(C_j) = a_{jj} + a_{(j-1)j} P_{(j-1)j} + a_{(j+1)j} P_{(j+1)j} + a_{j(j-1)}[1 - P_{j(j-1)}] + a_{j(j+1)}[1 - P_{j(j+1)}]. \tag{39}$$

Besides the self edge a_{jj}, the offloading workload on RSU-j is from four connected edges. With the probability-based offloading policy, it can be seen that, each of them follows the binomial distribution, e.g., $a_{(j-1)j} P_{(j-1)j} \sim \mathcal{B}(a_{(j-1)j}, P_{(j-1)j})$. (39) gives the total offloading expected demand on RSU-j, and its summation follows the Poisson binomial distribution, with the mean value of $\mu(C_j) = \mathbb{E}(C_j)$, and the variance is calculated as

$$\begin{aligned}\text{var}(C_j) &= \mathcal{A}(:, j)^\top [\mathbf{P}(:, j) \circ (1 - \mathbf{P}(:, j))] \\ &+ \mathcal{A}(j, :)[\mathbf{P}(j, :) \circ (1 - \mathbf{P}(j, :))]^\top \\ &= \sum_{k\in\{j-1, j+1\}} [P_{kj}(1 - P_{kj}) a_{kj} + P_{jk}(1 - P_{jk}) a_{jk}],\end{aligned} \tag{40}$$

where operator $\circ$ denotes Hadamard product.

The ideal computation delay in $J^*(\boldsymbol{\phi})$ is represented by (27) with the expected workload $\mathbb{E}(C_j)$, and the actual offloading workload C_j is expressed as a Poisson binomial distribution.With Assumption 1, an approximate optimal value difference between $J(\mathbf{P}^*)$ and $J^*(\boldsymbol{\phi})$ can be written as

$$J(\mathbf{P}^*) - J^*(\boldsymbol{\phi}) \simeq \sum_{j\in\mathcal{M}} \sqrt{\frac{\theta_j \mathbb{E}(C_j)}{E_j^o}} \left(C_j - \mathbb{E}(C_j)\right). \tag{41}$$

From *Bienaymé-Chebyshev* inequality [47], we can find an estimate of the optimal value error:

$$Pr\left(|J(\mathbf{P}^*) - J^*(\boldsymbol{\phi})| \geq \sum_{j\in\mathcal{M}} \text{var}(C_j)\sigma\sqrt{\frac{E_j^o}{\theta_j \mathbb{E}(C_j)}}\right) \leq \frac{1}{\sigma^2}, \tag{42}$$

where $\sigma > 1$ is a parameter to represent the number of standard deviations from the mean value.

REFERENCES

[1] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proc. IEEE*, vol. 107, no. 8, pp. 1697–1716, Aug. 2019.

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

[3] J. M. G. Sánchez et al., "Edge computing for cyber-physical systems," *ACM Trans. Cyber-Phys. Syst.*, Jun. 2022, doi: 10.1145/3539662.

[4] I. Jang, S. Choo, M. Kim, S. Pack, and G. Dan, "The software-defined vehicular cloud: A new level of sharing the road," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 78–88, Jun. 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, Jun. 2017.

[6] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2092–2104, Feb. 2019.

[7] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.

[8] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Trans. Cloud Comput.*, vol. 7, no. 1, pp. 196–209, Jan.-Mar. 2019.

[9] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.

[10] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[11] Y.-J. Ku and S. Dey, "Sustainable vehicular edge computing using local and solar-powered roadside unit resources," in *Proc. IEEE Veh. Technol. Conf.*, 2019, pp. 1–7.

[12] D. Tang, X. Zhang, and X. Tao, "Delay-optimal temporal-spatial computation offloading schemes for vehicular edge computing systems," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2019, pp. 1–6.

[13] Z. Ning et al., "Intelligent edge computing in internet of vehicles: A joint computation offloading and caching solution," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2212–2225, Apr. 2021.

[14] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.

[15] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[16] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019.

[17] Q. Qi et al., "Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4192–4203, May 2019.

[18] H. Guo, J. Liu, J. Ren, and Y. Zhang, "Intelligent task offloading in vehicular edge computing networks," *IEEE Wireless Commun.*, vol. 27, no. 4, pp. 126–132, Aug. 2020.

[19] Y. Sun et al., "Adaptive learning-based task offloading for vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3061–3074, Apr. 2019.

[20] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019.

[21] Z. Zhou, J. Feng, Z. Chang, and X. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, May 2019.

[22] S. Jošilo and G. Dan, "Joint management of wireless and computing resources for computation offloading in mobile edge clouds," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1507–1520, Oct.–Dec. 2021.

[23] T. Liu, Y. Zhang, Y. Zhu, W. Tong, and Y. Yang, "Online computation offloading and resource scheduling in mobile-edge computing," *IEEE Internet of Things J.*, vol. 8, no. 8, pp. 6649–6664, Apr. 2021.

[24] Z. Ning, J. Huang, X. Wang, J. J. Rodrigues, and L. Guo, "Mobile edge computing-enabled internet of vehicles: Toward energy-efficient scheduling," *IEEE Netw.*, vol. 33, no. 5, pp. 198–205, Sep./Oct. 2019.

[25] Z. Lv, D. Chen, and Q. Wang, "Diversified technologies in internet of vehicles under intelligent edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2048–2059, Apr. 2021.

[26] X. Xu et al., "Edge content caching with deep spatiotemporal residual network for IoV in smart city," *ACM Trans. Sens. Netw.*, vol. 17, no. 3, pp. 1–33, 2021.

[27] X. Wang et al., "Future communications and energy management in the internet of vehicles: Toward intelligent energy-harvesting," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 87–93, Dec. 2019.

[28] W. Chang, Y. Xiao, W. Lou, and G. Shou, "Offloading decision in edge computing for continuous applications under uncertainty," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 6196–6209, Sep. 2020.

[29] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[30] *Taxonomy and Definitions for Terms Related to Cooperative Driving Automat. for On-Road Motor Veh.*, Cooperative Driving Automation (CDA) Committee, Standard, SAE J3216 Standard, Jul. 2021.

[31] Z. Rejiba, X. Masip-Bruin, and E. Marín-Tordera, "A survey on mobility-induced service migration in the fog, edge, and related computing paradigms," *ACM Comput. Surv.*, vol. 52, no. 5, pp. 1–33, 2019.

[32] W. Bao et al., "Follow me fog: Toward seamless handover timing schemes in a fog computing environment," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 72–78, Nov. 2017.

[33] A. Nadembega, A. S. Hafid, and R. Brisebois, "Mobility prediction model-based service migration procedure for follow me cloud to support QoS and QoE," in *Proc. IEEE Int. Conf. Commun.*, 2016, pp. 1–6.

[34] Z. Gao, D. Chen, S. Cai, and H.-C. Wu, "OptDynLim: An optimal algorithm for the one-dimensional RSU deployment problem with nonuniform profit density," *IEEE Trans. Ind. Inform.*, vol. 15, no. 2, pp. 1052–1061, Feb. 2019.

[35] 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, Oct. 2016.

[36] W. Zhang, Y. Wen, J. Cai, and D. O. Wu, "Toward transcoding as a service in a multimedia cloud: Energy-efficient job-dispatching algorithm," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2002–2012, Jun. 2014.

[37] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[38] A. Uchechukwu et al., "Energy consumption in cloud computing data centers," *Int. J. Cloud Comput. Serv. Sci.*, vol. 3, no. 3, pp. 31–48, 2014.

[39] E. A. Lee and S. A. Seshia, *Proc. Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. Cambridge, MA, USA: MIT Press, 2017.

[40] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[41] L. Xiao, M. Johansson, and S. P. Boyd, "Simultaneous routing and resource allocation via dual decomposition," *IEEE Trans. Commun.*, vol. 52, no. 7, pp. 1136–1144, Jul. 2004.

[42] Z. Gao, D. Chen, S. Cai, and H.-C. Wu, "Optimal and greedy algorithms for the one-dimensional RSU deployment problem with new model," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7643–7657, Aug. 2018.

[43] X. Gong, J. Wang, B. Ma, L. Lu, Y. Hu, and H. Chen, "Real-time integrated power and thermal management of connected HEVs based on hierarchical model predictive control," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 3, pp. 1271–1282, Jun. 2021.

[44] X. Zhang, J. Zhang, Z. Liu, Q. Cui, X. Tao, and S. Wang, "MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3296–3309, Mar. 2020.
[45] W. Zhang, Y. Wen, and D. O. Wu, "Collaborative task execution in mobile cloud computing under a stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 14, no. 1, pp. 81–93, Jan. 2015.
[46] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Anal. and Optim.*, vol. 1, Belmont, MA, USA: Athena Scientific, 2003.
[47] C.-H. Chen and K.-T. Feng, "Statistical distance estimation algorithms with RSS measurements for indoor LTE-A networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1709–1722, Feb. 2017.

Kaige Tan (Member, IEEE) received the B.Sc. degree in mechatronics from the Harbin Institute of Technology, Harbin, China, in 2018, and the M.Sc. degree in mechatronics from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2020, where, he is currently working towards the Ph.D. degree with the Department of Machine Design. His research interests include optimization, reinforcement learning, and optimal filtering.

Lei Feng (Member, IEEE) received the B.S. and M.S. degrees in mechatronics from Xi'an Jiaotong University, Xi'an, China, in 1998 and 2001, respectively, and the Ph.D. degree in electrical & computer engineering from the Systems Control Group, University of Toronto, Toronto, ON, Canada, in 2007. In 2012, he joined the Mechatronics and Embedded Control System Division, KTH Royal Institute of Technology, Stockholm, Sweden, where he is currently an Associate Professor. His research interests mainly include energy management control of mechatronic systems, autonomous driving, verification and control synthesis of cyber-physical systems, and supervisory control of discrete event systems.

György Dán (Senior Member, IEEE) received the M.Sc. degree in computer engineering from the Budapest University of Technology and Economics, Budapest, Hungary, in 1999, the M.Sc. degree in business administration from the Corvinus University of Budapest, Budapest, Hungary, in 2003, and the Ph.D. degree in telecommunications from KTH Royal Institute of Technology, Stockholm, Sweden, in 2006. He was a Consultant of access networks, streaming media, and videoconferencing from 1999 to 2001. He was a Visiting Researcher with the Swedish Institute of Computer Science, Kista, Sweden, in 2008, a Fulbright Research Scholar with the University of Illinois, Champaign, IL, USA, Urbana-Champaign from 2012 to 2013, and an Invited Professor with EPFL from 2014 to 2015. He is currently a Professor with the KTH Royal Institute of Technology. His research interests include the design and analysis of content management and computing systems, game theoretical models of networked systems, and cyber-physical system security and resilience. He has been an Area Editor of *Computer Communications* since 2014 and the IEEE TRANSACTIONS ON MOBILE COMPUTING since 2019.

Martin Törngren (Senior Member, IEEE) received the Ph.D. degree in mechatronics from the KTH Royal Institute of Technology, Stockholm, Sweden, in 1995. Since 2002, he has been a Professor of embedded control systems with the Division of Mechatronics and Embedded Control Systems, KTH Royal Institute of Technology. He is also a pioneer in bridging the gaps between the fields of automatic control and real-time distributed systems, with research in recent years focused on architectural design and safety of autonomous trustworthy cyber-physical systems. He was a Visiting Scholar for several periods in abroad, including UC Berkeley, Berkeley, CA, USA, and the Stevens Institute of Technology, Hoboken, NJ, USA. His research interests mainly include networking and multidisciplinary research. He created the Innovative Center for Embedded Systems, a KTH-industry competence network launched in 2008. He is also the Director of Swedish Research Center on Trustworthy Edge Computing Systems and Applications, TECoSA.