Blockchain-enabled Intelligent Transportation Systems: A Distributed Crowdsensing Framework

Zhaolong Ning, Shouming Sun, Xiaojie Wang, Lei Guo, Song Guo, *Fellow, IEEE*, Xiping Hu, Bin Hu, and Ricky Y. K. Kwok, *Fellow, IEEE*

Abstract—Intelligent Transportation System (ITS) is critical to cope with traffic events, e.g., traffic jams and accidents, and provide services for personal traveling. However, existing researches have not jointly considered the user data safety, utility and system latency comprehensively, to the best of our knowledge. Since both safe and efficient transmissions are significant for ITS, we construct a blockchain-enabled crowdsensing framework for distributed traffic management. First, we illustrate the system model and formulate a multi-objective optimization problem. Due to its complexity, we decompose it into two subproblems, and propose the corresponding schemes, i.e., a Deep Reinforcement Learning (DRL)-based algorithm and a DIstributed Alternating Direction mEthod of Multipliers (DIADEM) algorithm. Extensive experiments are carried out to evaluate the performance of our solutions, and experimental results demonstrate that the DRL-based algorithm can legitimately select active miners and transactions to make a satisfied trade-off between the blockchain safety and latency, and the DIADEM algorithm can effectively select task computation modes for vehicles in a distributed way to maximize their social welfare.

Index Terms—Distributed traffic management, blockchain, deep reinforcement learning, vehicular crowdsensing, multi-objective optimization.

1 Introduction

Recently, the number of vehicles is sharply increasing all over the world due to the brought convenience for personal traveling. However, the massive number of vehicles on roads results in many serious problems, e.g., frequent traffic jams, accidents and road collapse. The government and Transportation Management Department (TMD) are paying more and more attention to provide a secure and efficient traffic environment and improve the traveling experience of people. However, traditional traffic management solutions not only have the disadvantage of low efficiency, but also waste a lot of manpower and financial resources.

With the rapid development of wireless communication techniques, Internet of Vehicles (IoV) has attracted increasing attention, and become a key component of Intelligent Transportation System (ITS) [1]. Various advanced technologies (e.g., crowdsensing and edge computing [2]) can be integrated in ITS to accurately sense the traffic events (e.g., traffic congestion and accidents) on roads, and respond to events in time as well as provide services for users. ITS mainly contains three processes, i.e., data collection, data processing and data uploading. Efficiency, safety, and user utility are three important metrics, since the participants in ITS focus on the system Quality of Service (QoS), their data safety and utilities, simultaneously. Therefore, it is necessary to optimize ITS from these three aspects and formulate a multi-objective optimization problem, rather than focusing one of them as the optimization objective while regarding the others as constraints.

Blockchain technique is a distributed ledger and has the characteristics of decentralization, trust, anonymity and tamper-resistant, which is promising to be integrated with ITS to guarantee the data safety of participants [3]. Although some researches have investigated the integration of blockchain and ITS, they mainly focus on data sharing, energy delivery, trust management, blockchain-enabled crowdsensing, and blockchain network architecture [4–9]. However, to the best of our knowledge, existing researches ignore the blockchain safety, the brought latency by blockchain and the trade-off between these two metrics. The challenges of establishing such an ITS can be summarized as follows:

- In order to optimize the data safety, user utility and system latency, these parameters need to be quantified at first. However, few studies focus on the quantification of data safety and user utility in ITS. Besides, the introduction of blockchain makes the quantification of system latency rather difficult. Thus, how to guarantee the user data security and jointly formulate these three factors are challenging.

- *Z. Ning, X. Wang (Corresponding author) and L. Guo (Corresponding author) are with the School of Communications and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China. Email: z.ning@ieee.org, xiaojie.kara.wang@ieee.org and guolei@cqupt.edu.cn.*
- *S. Sun is with the School of Software, Dalian University of Technology, Dalian, China. Email: shoumingsun@outlook.com.*
- *S. Guo is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China. Email: song.guo@polyu.edu.hk.*
- *X. Hu and B. Hu are with the School of Information Science and Engineering, Lanzhou University, Lanzhou, China. Email: huxp@lzu.edu.cn and bh@lzu.edu.cn.*
- *R. Kwok is with the School of Science and Technology, The Open University of Hong Kong, Hong Kong, China. Email: kwok@ouhk.edu.hk.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

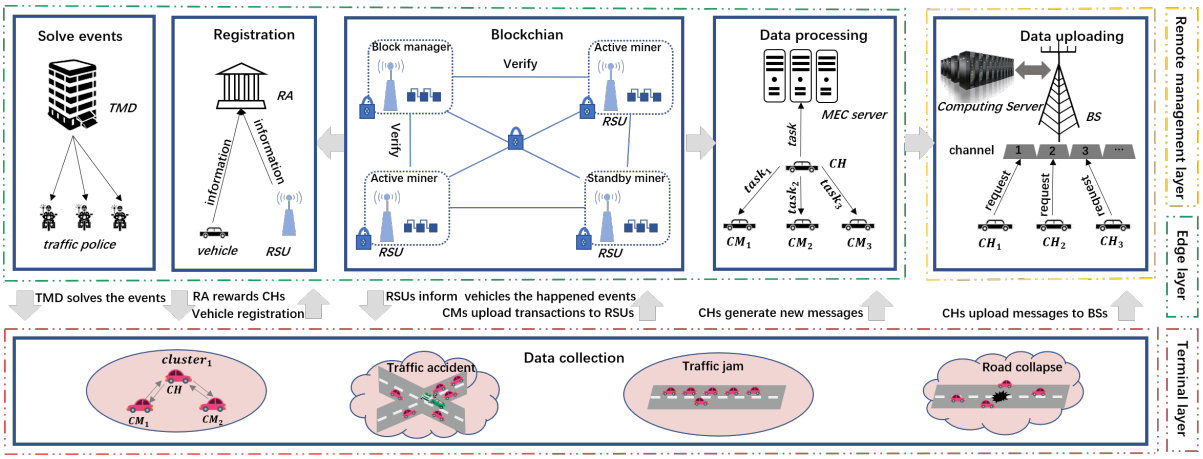


Fig. 1. An illustrative system model of blockchain-enabled ITS.

- To establish a safe, efficient and feasible ITS, a multi-objective optimization problem should be formulated, i.e., minimizing the system latency, maximizing the user data safety and its utility. However, some objectives conflict with others, e.g., user data safety and system latency. Thus, how to solve the multi-objective optimization problem and make a satisfied trade-off between conflicted objectives are extremely challenging.
- ITS contains a large number of participants with limited network resources, e.g., computing, storage and bandwidth resources. Correspondingly, how to optimize the data safety, user utility and system latency simultaneously under limited resources deserves to be well investigated.

This paper intends to construct a safe, efficient and feasible ITS. To the best of our knowledge, we are the first to formulate the blockchain safety and investigate the trade-off between blockchain safety and blockchain latency. The contributions of this paper are fourfold:

- We construct a safe, efficient and feasible ITS based on the decentralized blockchain technique. We formulate a multi-objective optimization problem to minimize the system latency and maximize the user data safety as well as its utility. In order to solve it effectively, we decompose it into two subproblems by the split and integration of optimization objectives.
- We intend to make a satisfied trade-off between the blockchain latency and safety in the first subproblem, i.e., maximizing the blockchain safety and minimizing its latency. Then, a Deep Reinforcement Learning (DRL)-based algorithm is proposed to solve it by legitimately selecting active miners from Road Side Units (RSUs) and transactions from the transaction pool.
- We intend to optimize the user utility in the second subproblem, i.e., maximizing the total utility of all users. In order to solve it, a DIstributed Alternating Direction mEthod of Multipliers (DIADEM) algorithm is proposed, and it can select suitable task computation modes for vehicles in a fully distributed way.
- We conduct various experiments in traffic networks to evaluate the effectiveness of our solution. Experiment results demonstrate that our method can make a satisfied trade-off between the blockchain safety and latency, and maximize the total utility of all users.

The rest of this paper is organized as follows. Section 2 illustrates the system model, and Section 3 shows the formulated multi-objective optimization problem. Section 4 elaborates the decomposition of the formulated problem and the designed algorithms. The performance of our solutions is demonstrated in Section 5. Section 6 reviews the related work, followed by the conclusion in Section 7.

2 System Model

In this section, we elaborate the designed blockchain-based intelligent traffic management system from three aspects, i.e., terminal layer, edge layer and remote management layer, as shown in Fig. 1. Correspondingly, the working process is illustrated in Fig. 2. It is noteworthy that our model can be used in many applications beyond traffic management system, e.g., collecting data from residential facility for energy-saving living environment, collecting medical data from patients for health monitoring, collecting environmental data for intelligent agriculture, and collecting equipment-related data for industrial manufacturing.

2.1 Terminal layer

Terminal layer merely contains vehicles, collecting data from traffic events. Cluster-based data collection and task scheduling have been well investigated to improve the system efficiency [10–12]. It is not our main focus in this paper, since we intend to establish a safe, efficient and practical ITS, and how to formulate the system model and solve the formulated problem is the key point. In this paper, vehicle clustering is involved in two processes, i.e., data collection and processing. In the highly dynamic vehicular environment, the objective of clustering is to achieve relatively stable cluster structures, because frequent cluster reconfiguration generates tremendous communication loads for data collection and processing. During data collection, vehicles are clustered based on their locations, and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

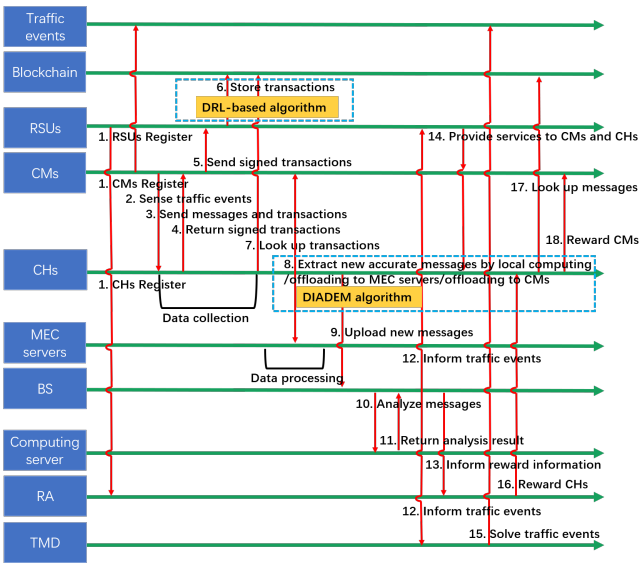


Fig. 2. The working process of blockchain enabled ITS.

details including cluster size, Cluster Head (CH) election and cluster reconfiguration can be found in [10]. Once the CH is elected, other vehicles act as Cluster Members (CMs). During data processing, the cluster of each CH is updated until its task is finished. Due to the advantages of distributed, real-time and robust, the blockchain-enabled vehicular crowdsensing technique can be leveraged for data collection. It contains the following steps: 1) CM creates a message and a transaction, and sends them to its CH. The message records the encountered traffic events, while the transaction records the created message; 2) By auditing the signature of CM generated by the Elliptic Curve Digital Signature Algorithm (ECDSA) in the transaction, CH can verify the integrity of the received message. CH saves the integrated message and signs its signature for the transaction, and returns the transaction to CM; 3) By verifying the signature of CH in the transaction, CM can ensure its message is successfully received by CH. Then, CM transmits the transaction to its nearest RSU for verification and storage in the blockchain to protect its generated message; 4) CH looks up the transactions from the blockchain to ensure its received messages are stored. In addition, each vehicle can only upload one message for the same traffic event during a certain time interval to prevent Distributed Denial of Service (DDoS) attack in the system. If a vehicle uploads multiple messages for the same traffic event, the redundant messages and transactions should be discarded.

2.2 Edge layer

Edge layer contains a Registration Authority (RA), a TMD, RSUs and Mobile Edge Computing (MEC) servers. RA is a credible governmental agency, from which vehicles and RSUs can obtain anonymous accounts through the registration process, and it can reward the participated vehicles. The transmitted privacy information in registration requires to be encrypted by the Rivest-Shamir-Adleman (RSA) algorithm to prevent information leakage. TMD is able to cope with the happened traffic events (e.g., through assigning traffic police). A kind of consortium blockchain [13] is applied in our work due to its advantages of high throughput and decentralization, compared with public and private blockchains. The blockchain application is implemented in RSUs due to the limited resources of vehicles, and RSUs hold the post of miners, which are divided into active and standby ones.

As shown in Fig. 2, after collecting data from CMs, CHs need to process the data and extract an accurate message describing the traffic condition, to reduce the message uploading delay and relieve the problem of false-reporting in traditional crowdsensing. To relieve the computation burden of CHs, MEC technique [14, 15] is integrated in our system. In addition, CHs can also offload the task to their CMs to take full advantage of the computing power of CMs. Therefore, there are three computation modes for CH to extract a new message, i.e., local computing, and offloading the task to MEC servers and CMs, respectively. Then, the CH uploads the new message to BS for awards. After receiving the awards from RA, CH can reward CMs based on their collected messages, since remote management layer communicates with CHs instead of CMs. We consider CHs can obtain rewards with a fixed ratio in the whole received rewards (this ratio can be adjusted to guarantee fairness). The remaining rewards can be distributed by CHs to CMs based on the accuracy degree of their uploaded traffic data. Since we intend to maximize the total utility of all vehicles (i.e., CHs and CMs), how CHs distribute the received rewards from RA is not our main focus, and the total rewards of all vehicles equal to the received rewards of CHs from RA. After calculating the corresponding rewards of each CM, CHs can inform RA the vehicular information as well as the corresponding rewards of CMs, and then RA executes the reward allocation process.

2.3 Remote management layer

Remote management layer contains BSs, deploying computing servers. The functions of BSs are fourfold, i.e., receiving messages from CHs and verifying their accuracy degree, informing TMD the happened events, informing vehicles the happened events via RSUs, and informing RA the reward information to reward CHs. Due to the limited bandwidth of BSs, the channel is divided into multiple subchannels to improve the bandwidth utilization. In Orthogonal Frequency Division Multiplexing (OFDM) [16], each subchannel only serves one user, leading to large uploading latency when there are a large number of vehicles that prepare to establish connections with the same BS. Thus, Non-Orthogonal Multiple Access (NOMA) [17] that each subchannel can serve multiple users simultaneously is employed in our system.

3 Problem Formulation

This section formulates our problem by considering data safety, user utility and system latency. However, it is rather difficult to formulate this problem from the global perspective, since multiple optimization objectives are correlated with each other among multiple processes and have distinct constraints. For simplicity, we first formulate the system model from the perspective of local optimization, i.e., only considering a certain CH m. Then, we decompose the formulated problem into two subproblems from the global perspective.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

TABLE 1
Main Notations

Notations	Description
$\mathcal{J}, \mathcal{K}, \mathcal{A}$	The set of RSUs, MEC servers and active miners, respectively
$\mathcal{W}$, $\mathcal{Q}$	The set of transactions in the block and the transaction pool, respectively
$\mathcal{M}_k, \mathcal{N}_{k,m}$	The set of CHs and CMs in the cluster with CH m connecting to MEC server k, respectively
F_x, T_x	The blockchain safety and latency when generating block x, respectively
$T_{k,m}^{loc}$	The task delay of CH m connecting to MEC server k for extracting a message in local
$T_{k,m}^{mec}$, $T_{k,m}^{cm}$	The task delay of CH m connecting to MEC server k for extracting a message by offloading to servers and CMs, respectively
$T_{k,m}^{pro}$	The computing delay of CH m connecting to MEC server k in data processing
$C_{k,m}^{loc}$	The task cost of CH m connecting to MEC server k for extracting a message in local
$C_{k,m}^{mec}$, $C_{k,m}^{cm}$	The task cost of CH m connecting to MEC server k for extracting a message by offloading to MEC servers and CMs, respectively
$C_{k,m}^{pro}$	The computing cost of CH m connecting to MEC server k in data processing
$AoI_{k,m}$	The AoI value of the generated message of CH m connecting to MEC server k
$Y_{k,m}^{ra}, U_{k,m}^{ra}$	The obtained reward and utility of CH m connecting to MEC server k from RA, respectively
δ_k, δ_n	The price of MEC server k and CM n for running one cycle CPU, respectively
z_w	The data size of transaction w
$d_{k,m}$	The size of the collected data of CH m connecting to MEC server k
$\delta_{k,m}$	The price of CH m connecting to MEC server k for running one cycle CPU

There are J RSUs and K MEC servers. We denote RSUs by $j \in \mathcal{J} = \{1, ..., J\}$, and MEC servers by $k \in \mathcal{K} = \{1, ..., K\}$. The active miners are represented by $a \in \mathcal{A} = \{1, ..., A\}$, where A is the number of active miners, and the current block manager is denoted by $\mathbb{A}$. Vehicles are divided into different clusters, and each CH connects to its nearest MEC server. CHs are denoted by $m \in \mathcal{M}_k = \{1, ..., M_k\}$, where M_k is the number of CHs connecting to MEC server k. CMs are denoted by $n \in \mathcal{N}_{k,m} = \{1, ..., N_{k,m}\}$, where $N_{k,m}$ is the number of CMs in the cluster with CH m connecting to MEC server k. Major notations are summarized in TABLE 1.

3.1 User data safety

Since RSUs are semi-credible [4] and the traditional center-based storage mode is not safe, it is necessary to introduce blockchain technique to prevent that malicious users alter message records for their personal interests to guarantee user fairness and data security. However, introducing blockchain brings extra latency, and it arises a large delay when a large-scale block is mined and there are enormous number of RSUs. In order to reduce the blockchain latency, miners are divided into active and standby ones, which can reduce the consensus time compared with traditional Practical Byzantine Fault Tolerance (PBFT) scheme. The former takes part in the block generation and consensus, while the latter accepts the consensus result. Active miners are randomly disrupted and act as the block manager in turn. The consensus process of the blockchain is shown in Fig. 3, which involves the following steps: 1) The block manager signs the generated block and broadcasts it to other active miners; 2) After verifying the block manager signature, active miners verify whether transactions are stored in the block by calculating their corresponding hash values and comparing the calculated root of the merkel tree with that in the block head; 3) The signed verification results are broadcasted to other active miners for further comparison; 4) If an active miner finds there are $2f$ active miners with the same verification result (including its verification result), it can reach a consensus with those active miners. Herein, f is the number of malicious active miners in the blockchain; 5) Active miners unicast comparison results to the block manager; 6) For the block manager, if there are $2f + 1$ active miners with the same comparison result (including its comparison result), it regards the blockchain reaches a consensus; 7) Finally, the block manager broadcasts the consensus result to other miners.

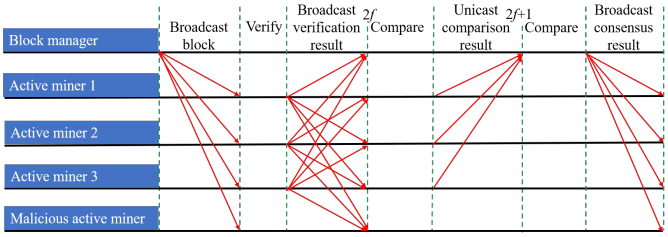


Fig. 3. The consensus process in blockchain.

Since user data is stored in the blockchain, its safety is equivalent to the blockchain safety. We consider each active miner (RSU) can be compromised by malicious attackers with probability λ. According to [18], if the number of malicious active miners is no more than $\lfloor \frac{A-1}{3} \rfloor$, blockchain security can be guaranteed, where A is the number of active miners in $\mathcal{A}$. Thus, we define the blockchain risk as the probability that the number of malicious active miners reaches $\lfloor \frac{A-1}{3} \rfloor + 1$ when generating block x, and the blockchain safety can be represented by:

$$F_x = 1 - \lambda^{\lfloor \frac{A-1}{3} \rfloor + 1}, \tag{1}$$

where block x contains the last collected message of CH m.

3.2 System latency

System latency contains three components, i.e., the latency of data collection, data processing and data uploading.

The time spent in data collection includes the fixed time for CHs to collect data, denoted by τ, and the blockchain latency for block x. Blockchain latency includes the time interval of block generation T_x^{gen} and that of reaching a consensus T_x^{con}, i.e., $T_x = T_x^{gen} + T_x^{con}$. The former contains the transaction verification time of the block manager (i.e., the required time to verify the signatures of CMs and CHs), and the block construction time (i.e., the needed time to calculate the root of the merkel tree), expressed by:

$$T_x^{gen} = 2W \frac{l_g}{f_{\mathbb{A}}^x} + \sum_{i=0}^{\lceil \log_2 W \rceil} (\lceil \frac{W}{2^i} \rceil \frac{l_h}{f_{\mathbb{A}}^x}), \tag{2}$$

where W is the block size, i.e., the number of transactions in block x, and $f_{\mathbb{A}}^x$ is the CPU frequency of block manager $\mathbb{A}$ when generating block x. Symbols l_g and l_h are the required

$$T_x^{con} = \left\{ \begin{array}{l} \psi A \sum\limits_{w \in \mathcal{W}} z_w + \max\limits_{a \in \mathcal{A} \setminus \{\mathbb{A}\}} \frac{l_g}{f_a^x} + \max\limits_{a \in \mathcal{A} \setminus \{\mathbb{A}\}} \{W(2\lceil \log_2 W \rceil + 1) \frac{l_h}{f_a^x}\} + \psi d_x A + \max\limits_{a \in \mathcal{J}} T_a^{app} + \\ \max\{ \max\limits_{a \in \mathcal{A} \setminus \{\mathbb{A}\}} \{(A-2) \frac{l_g}{f_a^x}\}, (A-1) \frac{l_g}{f_{\mathbb{A}}^x} \} + \max\limits_{a \in \mathcal{A} \setminus \{\mathbb{A}\}} \frac{d_x}{r_a^{\mathbb{A}}} + (A-1) \frac{l_g}{f_{\mathbb{A}}^x} + \psi d_x J + \max\limits_{a \in \mathcal{J} \setminus \{\mathbb{A}\}} \frac{l_g}{f_a^x} \end{array} \right\}. \quad (3)$$

CPU cycles for verifying a signature and calculating a hash value, respectively. The consensus time includes broadcasting time $\psi A \sum_{w \in \mathcal{W}} z_w$ of the block to other active miners, verification time $\frac{l_g}{f_a^x}$ of active miners for the signature of block manager, verification time $W(2\lceil \log_2 W \rceil + 1) \frac{l_h}{f_a^x}$ of active miners for the valid of the block, broadcasting time $\psi d_x A$ of the block verification results, comparison time of active miners for the verification results, i.e., $(A-1) \frac{l_g}{f_{\mathbb{A}}^x}$ for the block manager and $(A-2) \frac{l_g}{f_a^x}$ for other active miners, unicast time $\frac{d_x}{r_a^{\mathbb{A}}}$ of the comparison results to block manager, auditing time $(A-1) \frac{l_g}{f_{\mathbb{A}}^x}$ of block manager for the comparison results, broadcasting time $\psi d_x J$ of the consensus result, verification time $\frac{l_g}{f_a^x}$ of other miners for the signature of block manger, and appending time T_a^{app} of the block to blockchain. In the worst case, the consensus time can be expressed as (3).

Herein, ψ is a predefined parameter of the broadcasting time [4], and $\mathcal{W}$ is the set of transactions in block x, denoted by $w \in \mathcal{W} = \{1, ..., W\}$. Symbol d_x is the data size of messages in the consensus process, and f_a^x is the CPU frequency of miner a. Symbol $r_a^{\mathbb{A}}$ is the transmission rate from active miner a to block manager $\mathbb{A}$, and T_a^{app} is the block appending time of miner a. The transactions in the transaction pool are denoted by $w \in \mathcal{Q} = \{1, ..., Q\}$, and z_w is the data size of transaction w. The data size of block x is calculated by $\sum_{w \in \mathcal{W}} z_w$.

After receiving the messages from CMs, CH processes the received data, i.e., extracting a new accurate message from the collected data and uploading the message to BS for rewards. Due to the limited computing capacity of vehicles, CH can offload the corresponding task to its connected MEC server or CMs. For different computation modes, the corresponding task delay can be calculated as follows:

Local computing: The task delay of CH m for extracting the message in local is denoted by:

$$T_{k,m}^{loc} = \frac{l_{k,m}}{f_{k,m}^{loc}}, \quad (4)$$

where $l_{k,m}$ is the required CPU cycles of the task for CH m connecting to MEC server k, equaling to the collected data size multiplies by the computation intensity $\mathfrak{v}$, and $f_{k,m}^{loc}$ is the CPU frequency of CH m.

Offloading to MEC servers: If CH m offloads the task to MEC server k, the task delay includes the transmission delay of the task to MEC server k and the execution delay of MEC server, i.e.,

$$T_{k,m}^{mec} = \frac{d_{k,m}}{r_{k,m}^k} + \frac{l_{k,m}}{f_{k,m}^k}, \quad (5)$$

where $d_{k,m}$ is the size of the collected data for CH m connecting to MEC server k, and $r_{k,m}^k$ is the transmission rate of the task from CH m to MEC server k. Symbol $f_{k,m}^k$ is the CPU frequency of MEC server k that CH m connects to.

Offloading to CMs: The delay of CH m for offloading the task to its CMs equals to the maximum task delay of its CMs, i.e., $T_{k,m}^{cm} = \max\limits_{n=1,...,N_{k,m}} T_{k,m,n}^{cm}$, where $T_{k,m,n}^{cm}$ is the task delay of CM n to offload the partial task of CH m connecting to MEC server k, containing the transmission delay and execution delay, i.e.,

$$T_{k,m,n}^{cm} = \zeta_{k,m,n} (\frac{d_{k,m}}{r_{k,m}^n} + \frac{l_{k,m}}{f_{k,m}^n}), \quad (6)$$

where $\zeta_{k,m,n}$ is the task ratio that CH m connecting to MEC server k offloads the task to CM n. We intend to determine which CM has the maximum task delay for the subsequent calculations of message freshness and user utility, and propose proposition 1. Since the task delay contains the task transmission and execution delay, we need to consider them when allocating tasks to CMs, i.e., equation (7) is leveraged to guarantee the most number of tasks can be offloaded to the CM with the highest CPU frequency, so that it has the maximum task delay among CMs.

Proposition 1. *If binary variable $\gamma_{k,m}$ indicting whether CH m connecting to MEC server k offloads tasks to CMs and task ratio variable $\zeta_{k,m,n}$ in (6) satisfy the constraint in (7), the offloading delay from a CH to its CMs can be uniquely determined and equals to the one of the fastest CM, i.e.,*

$$\gamma_{k,m} (f_{k,m}^{n_1} - f_{k,m}^{n_2})(T_{k,m,n_1}^{cm} - T_{k,m,n_2}^{cm}) \geq 0, \quad \forall k \in \mathcal{K}, m \in \mathcal{M}_k, n_1, n_2 \in \mathcal{N}_{k,m}. \quad (7)$$

The proof can be found in Appendix A of Supplementary File.

By comprehensively considering the above three modes, the computing latency of CH m can be expressed by:

$$T_{k,m}^{pro} = \alpha_{k,m} T_{k,m}^{loc} + \beta_{k,m} T_{k,m}^{mec} + \gamma_{k,m} T_{k,m}^{cm}, \quad (8)$$

where binary variables $\alpha_{k,m} = 1$, $\beta_{k,m} = 1$ and $\gamma_{k,m} = 1$ illustrate CH m connecting to MEC server k chooses local computing, offloads the task to MEC servers and to its CMs, respectively.

For data uploading, uplink NOMA is considered in BSs. Since the delay in data uploading is related to the subchannel allocation of BSs, which has been well investigated in [19, 20], how to allocate bandwidth resources of BSs for users is not our main contribution, and we consider the uploading delay, i.e., $T_{k,m}^{upl}$, as a constant value.

The concept of Age of Information (AoI) [21] is introduced to represent the freshness of the generated message of CH m connecting to MEC server k, calculated by:

$$AoI_{k,m} = \tau + T_x + T_{k,m}^{pro} + T_{k,m}^{upl}, \quad (9)$$

where $\tau + T_x$ is the required time in data collection of CH m; T_x is the blockchain latency; $T_{k,m}^{pro}$ is the computing latency

of CH m in data processing; and $T^{upl}_{k,m}$ is the delay in data uploading.

3.3 User utility

In our system, there are two kinds of rewards. One is the mining rewards of RSUs received from blockchain operators for mining, and the other is the rewards of vehicles received from RA for their reported traffic data. Since we focus on the total utility of all vehicles in our system, the mining reward of RSUs is out of the scope of our research. The reward of CH m received from RA is related to the freshness of its uploaded message, i.e., $AoI_{k,m}$ and the accuracy of its uploaded message. Since the reported message of CH m describes the traffic condition, we consider the accuracy of the message has positive correlation with data size $d_{k,m}$, and the received rewards of vehicles are non-negative [11]. Thus, the received reward of CH m can be expressed by:

$$Y^{ra}_{k,m}(d_{k,m}, AoI_{k,m}) = \max\{(AoI^{max} - AoI_{k,m})d_{k,m}, 0\}, \tag{10}$$

where AoI^{max} is the upper bound of the message freshness, and we define 1s $\times$ 1kbits = 1token, where token represents the reward unit. If $AoI_{k,m}$ is larger than AoI^{max}, the extracted message of CH m is considered to be invalid and abandoned by BS, and the reward of CH m is 0.

For different computation modes, the corresponding required task costs are calculated as follows:

Local computing: The task cost of CH m for extracting the message in local is expressed by:

$$C^{loc}_{k,m} = l_{k,m}\delta_{k,m}, \tag{11}$$

where $l_{k,m}$ is the required CPU cycles of the task of CH m connecting to MEC server k, and $\delta_{k,m}$ is the price of CH m connecting to MEC server k for running one cycle CPU (token per cycle).

Offloading to MEC servers: The task cost of offloading to MEC servers contains the uplink transmission cost and the price of MEC server k, i.e.,

$$C^{mec}_{k,m} = \frac{d_{k,m}}{r^{k}_{k,m}}P_{k,m}\mathfrak{r} + l_{k,m}\delta_{k}, \tag{12}$$

where $P_{k,m}$ is the transmit power of CH m connecting to MEC server k, and $\mathfrak{r}$ is the unit price of energy (token per Joule). Symbol δ_k is the price of MEC server k for running one cycle CPU (token per cycle).

Offloading to CMs: The task cost that CH m demands to pay when offloading the task to CMs includes the downlink transmission cost and the price of all CMs, i.e.,

$$C^{cm}_{k,m} = \sum_{n=1}^{N_{k,m}}\left(\frac{\zeta_{k,m,n}d_{k,m}}{r^{n}_{k,m}}P_{k,m}\mathfrak{r} + \zeta_{k,m,n}l_{k,m}\delta_{n}\right), \tag{13}$$

where δ_n is the price of CM n for running one cycle CPU (token per cycle). Correspondingly, the computing cost of CH m in data processing is calculated by:

$$C^{pro}_{k,m} = \alpha_{k,m}C^{loc}_{k,m} + \beta_{k,m}C^{mec}_{k,m} + \gamma_{k,m}C^{cm}_{k,m}. \tag{14}$$

Then, the utility of CH m, i.e., the difference between its reward and computing cost, becomes:

$$U^{ra}_{k,m} = Y^{ra}_{k,m}(d_{k,m}, AoI_{k,m}) - C^{pro}_{k,m}. \tag{15}$$

3.4 Problem formulation

In our system, we intend to minimize the system latency, and maximize the data safety as well as user utility. From the perspective of CH m, the problem can be formulated as:

$$\begin{aligned}
P: \quad & \max_{\mathcal{A}} \quad F_x, \\
& \min_{\mathcal{A},\mathcal{W},\alpha_{k,m},\beta_{k,m},\gamma_{k,m},\zeta_{k,m,n}} \quad AoI_{k,m}, \\
& \max_{\mathcal{A},\mathcal{W},\alpha_{k,m},\beta_{k,m},\gamma_{k,m},\zeta_{k,m,n}} \quad U^{ra}_{k,m}, \\
s.t. \quad & C1: 0 < A \le J, A \in \mathbb{N}, \\
& C2: 0 < W \le Q, W \in \mathbb{N}, \\
& C3: F_x \ge F^{min}_x, \\
& C4: \frac{W}{T_x} \ge \Omega^{min}_x, \\
& C5: T_w + T_x \le T^{max}_w, \forall w \in \mathcal{Q}, \\
& C6: \alpha_{k,m}, \beta_{k,m}, \gamma_{k,m} \in \{0,1\}, \zeta_{k,m,n} \in [0,1], \\
& \quad \forall n \in \mathcal{N}_{k,m}, \\
& C7: \alpha_{k,m} + \beta_{k,m} + \gamma_{k,m} = 1, \\
& C8: \gamma_{k,m}\sum_{n=1}^{N_{k,m}}\zeta_{k,m,n} = \gamma_{k,m}, \\
& C9: \gamma_{k,m}\zeta_{k,m,n}d_{k,m} \le \aleph_{k,m,n}, \forall n \in \mathcal{N}_{k,m}, \\
& C10: \beta_{k,m}l_{k,m} \le \varsigma, \\
& C11: \gamma_{k,m}(f^{n_1}_{k,m} - f^{n_2}_{k,m})(T^{cm}_{k,m,n_1} - T^{cm}_{k,m,n_2}) \ge 0, \\
& \quad \forall n_1, n_2 \in \mathcal{N}_{k,m}, \\
& C12: AoI_{k,m} \le AoI^{max}.
\end{aligned} \tag{16}$$

Herein, constraints $C1$, $C2$, $C6$, $C7$ and $C8$ ensure the validation of decision variables. The minimum requirement for the security of blockchain is shown in $C3$, i.e., F^{min}_x. Constraint $C4$ restricts the blockchain throughput cannot be less than its lower bound Ω^{min}_x. Symbol T_w is the waiting time of transaction w in the transaction pool, and constraint $C5$ restricts the time span of transactions before being stored in the blockchain is no more than its upper bound T^{max}_w. Constraint $C9$ restrains the offloaded task size of CH m to one CM cannot exceed its D2D link capacity $\aleph_{k,m,n}$. MEC servers can communicate with each other, and the overloaded ones are able to offload tasks to other underloaded MEC servers to make full use of their computing resources. The offloaded tasks of CH m to MEC servers cannot exceed their total computing resources as illustrated in $C10$, and $C11$ guarantees the CM with the largest CPU frequency has the maximum task delay when CH m offloads tasks to its CMs. $C12$ restricts the freshness of the uploaded message to reduce the search scope of pareto optimal solutions. The details about the corresponding relationship between decision variables and constraints can be found in TABLE 2. There is no redundancy between minimizing $AoI_{k,m}$ and maximizing $U^{ra}_{k,m}$, since the utility of vehicles is determined by their received rewards and task costs in data processing. Although the received rewards are related to the freshness of messages, task costs are impacted by the mode of task computation, the prices of CMs and MEC servers.

TABLE 2
Relationship between Decision Variables and Constraints

Decision Variables	Constraints
$\mathcal{A}$	C1, C3, C4, C5, C12
$\mathcal{W}$	C2, C4, C5, C12
$\alpha_{k,m}$	C6, C7, C12
$\beta_{k,m}$	C6, C7, C10, C12
$\gamma_{k,m}$	C6, C7, C8, C9, C11, C12
$\zeta_{k,m,n}$	C6, C8, C9, C11, C12

4 Our Solution

In order to solve the formulated multi-objective optimization problem, we first decompose it into two subproblems for decoupling based on the split and integration of the optimization objectives in P. After that, two corresponding schemes are developed, i.e., a DRL-based algorithm and a DIADEM algorithm.

4.1 Problem Decomposition

Proposition 2. *The upper bound of message freshness AoI^{max} can be decomposed into three parts, i.e., AoI^{col}, AoI^{pro} and $T^{upl}_{k,m}$, representing the upper bound of time in data collection and data processing, and the delay in data uploading which is a constant. The division of AoI^{max} is determined by experienced traffic managers based on their different requirements for different processes (i.e., data collection, data processing and data uploading) to guarantee subproblems are solvable. Then, the objectives of our multi-objective optimization problem P can be determined by three variables, i.e., blockchain latency T_x, blockchain safety F_x, and utility $U_{k,m}$ of CH m obtained in data processing, where $U_{k,m} = (AoI^{pro} - T^{pro}_{k,m})d_{k,m} - C^{pro}_{k,m}$.*

The proof can be found in Appendix B of Supplementary File.

By considering the three objectives and constraints in problem P, we decompose it into two subproblems, i.e., $P1$ and $P2$. For problem $P1$:

$$\begin{aligned} P1: \quad & \min_{\mathcal{A},\mathcal{W}} \quad T_x, \\ & \max_{\mathcal{A}} \quad F_x, \\ s.t. \quad & C1 - C5 \ in \ P, \\ & C6: T_x \le T^{max}_x, \end{aligned} \tag{17}$$

where constraint $C6$ in $P1$ guarantees the blockchain latency cannot exceed its upper bound T^{max}_x, and $T^{max}_x = AoI^{col} - \tau$. We can observe that the purpose of problem $P1$ is to make a trade-off between blockchain latency and safety by selecting the sets of active miners $\mathcal{A}$ and transactions $\mathcal{W}$.

Problem $P2$ intends to maximize the utility of CH m in data processing by selecting a proper task computation mode for CH m, i.e.,

$$\begin{aligned} P2: \quad & \max_{\alpha_{k,m},\beta_{k,m},\gamma_{k,m},\zeta_{k,m,n}} U_{k,m}, \\ s.t. \quad & C1: \tau + T_x + \alpha_{k,m} T^{loc}_{k,m} \le T^{max}_{k,m}, \\ & C2: \tau + T_x + \beta_{k,m} T^{mec}_{k,m} \le T^{max}_{k,m}, \\ & C3: \tau + T_x + \gamma_{k,m} T^{cm}_{k,m} \le T^{max}_{k,m}, \\ & C6 - C11 \ in \ P, \end{aligned} \tag{18}$$

where constraints $C1-C3$ in $P2$ limit the upper bound time in data collection and processing for the three computation modes of CH m, and $T^{max}_{k,m} < \tau + T_x + AoI^{pro}$ holds to ensure CH m can obtain a positive task reward in data processing. By resolving problem $P1$, variable T_x can be obtained for the input of problem $P2$.

However, when CH m processes data, many CHs can extract the generated messages by offloading tasks to MEC servers simultaneously. Considering that MEC servers have limited computing power, we intend to maximize the sum utility of all CHs (social welfare), i.e.,

$$\begin{aligned} P2': \quad & \max_{\alpha,\beta,\gamma,\zeta} \sum_{k=1}^{K} \sum_{m=1}^{M_k} U_{k,m}, \\ s.t. \quad & C1: \alpha_{k,m}, \beta_{k,m}, \gamma_{k,m} \in \{0,1\}, \zeta_{k,m,n} \in [0,1], \\ & C2: \alpha_{k,m} + \beta_{k,m} + \gamma_{k,m} = 1, \\ & C3: \gamma_{k,m} \sum_{n=1}^{N_{k,m}} \zeta_{k,m,n} = \gamma_{k,m}, \\ & C4: \tau + T_x + \alpha_{k,m} T^{loc}_{k,m} \le T^{max}_{k,m}, \\ & C5: \tau + T_x + \beta_{k,m} T^{mec}_{k,m} \le T^{max}_{k,m}, \\ & C6: \tau + T_x + \gamma_{k,m} T^{cm}_{k,m} \le T^{max}_{k,m}, \\ & C7: \gamma_{k,m} \zeta_{k,m,n} d_{k,m} \le \aleph_{k,m,n}, \\ & C8: \sum_{k=1}^{K} \sum_{m=1}^{M_k} \beta_{k,m} l_{k,m} \le \varsigma, \\ & C9: \gamma_{k,m}(f^{n_1}_{k,m} - f^{n_2}_{k,m})(T^{cm}_{k,m,n_1} - T^{cm}_{k,m,n_2}) \ge 0, \\ & \quad \forall k \in \mathcal{K}, m \in \mathcal{M}_k, n_1, n_2 \in \mathcal{N}_{k,m}. \end{aligned} \tag{19}$$

For simplicity, we set $\alpha = \{\alpha^k\} = \{\alpha_{k,m}\}$, $\beta = \{\beta_{k,m}\}$, $\gamma = \{\gamma^k\} = \{\gamma_{k,m}\}$, and $\zeta = \{\zeta^k\} = \{\zeta_{k,m,n}\}$, $\forall k \in \mathcal{K}, m \in \mathcal{M}_k, n \in \mathcal{N}_{k,m}$. The two subproblems are correlative in time dimension, and should be solved in order, since the result of the former influences that of the latter. In constraints C4 to C6 of $P2'$, we intend to make full use of the remaining time of previous procedures. There are two advantages: one is that the second subproblem is easier to be solved due to larger feasible regions, and the other is that more time resource can be utilized to obtain suitable solutions.

4.2 DRL-based Algorithm for P1

Problem $P1$ is a multi-objective optimization problem, and no single global solution exists (i.e., the solution of $P1$ is a pareto optimal solution set) [22]. It is worth noticing that there are two decision variables (i.e., $\mathcal{A}$ and $\mathcal{W}$) to deduce the minimum value of T_x, while the maximum value of F_x is only determined by $\mathcal{A}$. However, the above two objectives are in conflict with each other due to decision variable $\mathcal{A}$. In order to evaluate the quality of the pareto optimal solution, we define the Optimal Improvement Ratio (OIR) as follows:

$$OIR_x = \frac{T^{max}_x - T_x}{T^{max}_x} + \mathfrak{d}\frac{F_x - F^{min}_x}{F^{min}_x}, \tag{20}$$

where $\mathfrak{d}$ is a constant to guarantee the first and second parts in (20) in the same order of magnitude. Then, problem $P1$ is transformed into $P1'$, i.e.,

$$P1': \quad \max_{\mathcal{A},\mathcal{W}} OIR_x, \tag{21}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

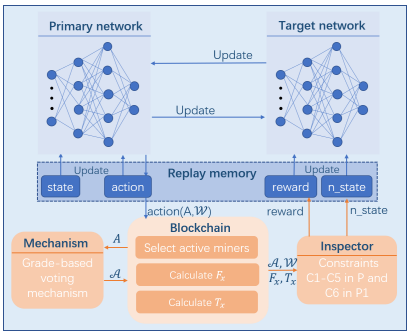


Fig. 4. The framework of DRL-based algorithm for P1.

$$s.t. \quad C1 - C5 \; in \; P, \\ C6 \; in \; P1.$$

Herein, the restrictions of blockchain safety and blockchain latency as shown in constraints C3 and C5 in $P1'$ can decrease the search scope of pareto optimal solutions and accelerate the solving process of the first subproblem.

For transaction selection, we need to consider the data size, waiting time in the transaction pool and blockchain environment. However, it is rather challenging to select suitable active miners from RSUs, since the reliability and computing power of RSUs need to be evaluated. To solve problem $P1'$, a DRL-based algorithm is proposed to select transactions and active miners, and the corresponding framework is shown in Fig. 4.

The state space for $P1'$ contains transaction data size $\mathcal{Z}$, transaction waiting time $\mathcal{T}$, blockchain latency T_x, blockchain safety F_x, and blockchain throughput Ω_x, i.e.,

$$S'^{(t)} = [\mathcal{Z}, \mathcal{T}, T_x, F_x, \Omega_x]^{(t)}, \tag{22}$$

where $\mathcal{Z} = \{z_w\}$, $\mathcal{T} = \{T_w\}, w \in \mathcal{Q}$, and $\mathcal{Q}$ is the transaction set in the transaction pool. The action space is:

$$A'^{(t)} = [\Xi, \Delta]^{(t)}, \tag{23}$$

where Ξ is the number of selected active miners, and can be expressed as $\Xi \in \{1, ..., J\}$ by using the limited fractional method [23]. The selected transaction indicator is $\Delta = \{\Delta_w\}$, $\Delta_w \in \{0, 1\}$, and $\sum_{w=1}^{Q} \Delta_w = W$. Herein, $\Delta_w = 1$ represents transaction w is contained in block x, otherwise $\Delta_w = 0$. The reward function is defined based on the objective of $P1'$ (i.e., maximizing OIR_x) as well as its corresponding constraints, denoted by:

$$R'^{(t)} = \begin{cases} OIR_x, & \text{if } C1 - C5 \text{ in } P \text{ and } C6 \text{ in } P1 \\ & \text{are satisfied,} \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

After receiving the action, the blockchain selects proper active miners through the proposed grade-based voting mechanism. Then, the blockchain environment is updated, and the corresponding reward as well as the next state (i.e., n_state) can be obtained from the inspector based on the constraints in $P1'$. The related information of this decision epoch (i.e., $(state||action||reward||n\_state)$) is stored in the memory of experience replay. The primary network is trained and updated based on the target network and a batch of random samples from the experience replay memory per $\mathfrak{L}$ decision epochs. The target network is updated when the primary network is updated per $\mathfrak{J}$ times. Finally, the optimal number of active miners Ξ^* and the selected transactions in Δ^* can be obtained.

At each decision epoch, how to select the specified active miners from RSUs needs to be resolved. For existing consensus mechanisms (e.g., Delegated Proof-of-Stake (DPoS) [24]), active miners are selected by major shareholders based on the shares they hold. However, a serious defect exists in the consensus, i.e., malicious shareholders may vote attackers as active miners for their own purposes. In order to cope with this issue, the grade-based voting mechanism is proposed by considering the credit and computing power of miners (RSUs). The former enhances the security of blockchain, and the latter improves the blockchain latency as well as throughput.

In the following, we specify the presented grade-based voting mechanism. When a CM uploads its transaction to the nearest RSU, a task for Proof-of-Computing (PoC) is uploaded to measure the computing power of the RSU. The CM looks up its transaction in the blockchain after data collection. The relay RSU is considered to be creditable if the transaction of the CM is stored in the blockchain, since the malicious RSU abandons the uploaded transaction. The computing power of one RSU is evaluated by the time span that the CM receives the task result of PoC. By jointly considering the computing power and credit of RSUs, the CM gives an interaction grade to the relay RSU for this interaction, i.e.,

$$g = \begin{cases} g^1, & \text{credible and with strong computing power,} \\ g^2, & \text{credible and with weak computing power,} \\ g^3, & \text{otherwise,} \end{cases} \tag{25}$$

where $g^1 > g^2 > g^3 > 0$. At the end of each blockchain epoch (i.e., each active miner has worked as a block manager once), vehicles need to vote new active miners based on their interactions with RSUs. The interaction set of CM n with RSU j is denoted by $\vartheta \in \Theta = \{1, ..., \theta\}$, and the interaction grade of ϑ is g_ϑ. CM n gives a local grade to RSU j based on all of their interactions in Θ by considering the freshness of interactions and the importance of the interaction grade. Since recent interactions are more significant than previous ones for RSU evaluation, we evaluate the freshness of interactions by defining the conception of Age of inTeraction (AoT), i.e.,

$$AoT_\vartheta = t_\mathfrak{y} - t_\vartheta, \tag{26}$$

where $t_\mathfrak{y}$ is the current voting time of blockchain, and t_ϑ represents the interaction timestamp of interaction ϑ. Then, the sum of level interaction grades is calculated by:

$$G^j_{n,\varrho} = \sum_{\vartheta=1}^{\theta} e^{-AoT_\vartheta} g_\vartheta \mathfrak{k}_\vartheta, \varrho \in \{1, 2, 3\}, \tag{27}$$

where $\mathfrak{k}_\vartheta = 1$ if g_ϑ equals to g^ϱ. Otherwise, $\mathfrak{k}_\vartheta = 0$.

We introduce various weights, i.e., $\varkappa_1, \varkappa_2$ and $\varkappa_3$, to evaluate the importance of different level interaction grades (i.e., g^1, g^2 and g^3). Herein, the weights should satisfy $\varkappa_1 + \varkappa_2 + \varkappa_3 = 1$ and $0 < \varkappa_3 < \varkappa_2 < \varkappa_1 < 1$, since the trusted RSUs have higher grades than the malicious ones.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

The local grade given by CM n to RSU j is stored in the blockchain, denoted by:

$$G_n^j = \frac{\sum_{\varrho=1}^{3} \varkappa_\varrho G_{n,\varrho}^j}{\sum_{\varrho=1}^{3} G_{n,\varrho}^j}. \tag{28}$$

In addition, we consider the situation that some malicious voters (CMs) arbitrarily give false grades to RSUs due to their private interests. Those malicious CMs puzzle the blockchain about the credit of RSUs. In order to tackle this situation, we need to evaluate the reliability of the local grades that CMs give to RSUs. The set of CMs interacting with RSU j is denoted by $\varpi \in \Pi = \{1, ..., \pi\}$, and CM ϖ broadcasts its local grade G_ϖ^j to other CMs after giving a local grade to RSU j. We define the following equation to obtain the reliability of CM n:

$$\rho_n^j = 1 - \frac{|G_n^j - \frac{1}{\pi-1}\sum_{\varpi\in\Pi\setminus\{n\}} G_\varpi^j|}{\max\{|G_n^j|, |\frac{1}{\pi-1}\sum_{\varpi\in\Pi\setminus\{n\}} G_\varpi^j|\}}. \tag{29}$$

We notice that in (29), if a CM gives a false local grade to an RSU, the difference between its local grade and the average local grade of other CMs is huge, and the reliability of its local grade is near to 0. In addition, even if some CMs are malicious voters, the influence on the reliability of CM n, i.e., ρ_n^j, can be negligible. The reason is that, the number of malicious voters is much less than that of honest voters, and the local grades of malicious voters have an inappreciable effect on the average local grade of all CMs (excepting CM n). The voting grade of RSU j is expressed by:

$$\mathcal{G}_j = \frac{1}{\pi}\sum_{\varpi=1}^{\pi}(1-\iota_\varpi)\rho_\varpi^j G_\varpi^j, \tag{30}$$

where ι_ϖ denotes the probability of the failed link transmission of CM ϖ. Based on voting grades, proper active miners can be selected from RSUs. Malicious RSUs and vehicles can be easily detected based on our grade-based voting mechanism.

4.3 DIADEM Algorithm for P2'

After solving problem $P1$, we can obtain blockchain latency T_x to solve problem $P2'$.

Proposition 3. *Problem $P2'$ is NP-hard.*

The proof can be found in Appendix C of Supplementary File.

We first transform it into a convex optimization problem. Then, a DIADEM algorithm is developed to find the optimal solution. Binary variables α, β and γ in $P2'$ are relaxed to real numbers within a closed interval from 0 to 1, indicating the selective probability of CHs for the three task computation modes (i.e., local computing, offloading to MEC servers and offloading to CMs). To make $P2'$ convex, we define $\tilde{\zeta} = \{\tilde{\zeta}^k\} = \{\zeta_{k,m,n}\}$ to represent the product term of γ and ζ, and $\tilde{\zeta}_{k,m,n} = \gamma_{k,m}\zeta_{k,m,n}, k \in \mathcal{K}, m \in \mathcal{M}_k, n \in \mathcal{N}_{k,m}$. If CH m does not offload its task to CMs (i.e., $\gamma_{k,m} = 0$), the ratio of tasks offloaded to its CMs equals to 0. Therefore, if $\gamma_{k,m} > 0$, $\zeta_{k,m,n} = \frac{\tilde{\zeta}_{k,m,n}}{\gamma_{k,m}}$ holds. Otherwise, $\zeta_{k,m,n} = 0$.

After that, problem $P2'$ is transformed into $P2''$, i.e.,

$$P2'': \quad \max_{\alpha,\beta,\gamma,\tilde{\zeta}} \sum_{k=1}^{K}\sum_{m=1}^{M_k} U''_{k,m}, \tag{31}$$

$$\begin{aligned}
s.t. \quad & C1: \sum_{n=1}^{N_{k,m}} \tilde{\zeta}_{k,m,n} = \gamma_{k,m},\\
& C2: \tau + T_x + \alpha_{k,m}T_{k,m}^{loc} \le T_{k,m}^{max},\\
& C3: \tau + T_x + \beta_{k,m}T_{k,m}^{mec} \le T_{k,m}^{max},\\
& C4: \tau + T_x + \tilde{\zeta}_{k,m,n}(\frac{d_{k,m}}{r_{k,m}^n} + \frac{l_{k,m}}{f_{k,m}^n}) \le T_{k,m}^{max},\\
& C5: \tilde{\zeta}_{k,m,n}d_{k,m} \le \aleph_{k,m,n},\\
& C6: \sum_{k=1}^{K}\sum_{m=1}^{M_k}\beta_{k,m}l_{k,m} \le \varsigma,\\
& C7: (f_{k,m}^{n_1} - f_{k,m}^{n_2})[\tilde{\zeta}_{k,m,n_1}(\frac{d_{k,m}}{r_{k,m}^{n_1}} + \frac{l_{k,m}}{f_{k,m}^{n_1}}) -\\
& \quad \tilde{\zeta}_{k,m,n_2}(\frac{d_{k,m}}{r_{k,m}^{n_2}} + \frac{l_{k,m}}{f_{k,m}^{n_2}})] \ge 0, \forall n_1, n_2 \in \mathcal{N}_{k,m},\\
& C8: \alpha_{k,m}, \beta_{k,m}, \gamma_{k,m}, \tilde{\zeta}_{k,m,n} \in [0,1].
\end{aligned}$$

Herein, $U''_{k,m}$ is an equivalent form of $U_{k,m}$ (i.e., replacing $\gamma_{k,m}\zeta_{k,m,n}$ with $\tilde{\zeta}_{k,m,n}$). In problem $P2''$, we do not consider constraint $C2$ in $P2'$, since α, β and γ are binary variables in $P2'$. However, they are real numbers in $P2''$, requiring to be recovered from real numbers to binary variables. Constraint $C2$ in $P2'$ should be guaranteed during the variable recovery.

Proposition 4. *Problem $P2''$ is a convex optimization problem.*

The proof can be found in Appendix D of Supplementary File.

The scale of problem $P2''$ is related to the numbers of MEC servers, CHs and CMs. In order to solve $P2''$ with a large scale, a centralized algorithm requires the computing center to collect the local information of all MEC servers, leading to extra network costs and latency. Thus, a distributed algorithm is required in each MEC server to reduce the decision costs and latency of CHs. However, the coupling among MEC servers cannot be removed in problem $P2''$, since β is a global variable in constraint $C6$. To handle this issue, we introduce a local copy of global variable β for MEC server k, i.e., $\tilde{\beta}^k$, and define $\tilde{\beta} = \{\tilde{\beta}^k\} = \{\tilde{\beta}_{i,m}^k\}$. Herein, i denotes MEC server i, $i = 1, ..., K$, and $k \in \mathcal{K}$. Symbol m represents CH m, and $m = 1, ..., M_i$. In order to guarantee local copy $\tilde{\beta}^k$ is consistent with global variable β, equation $\tilde{\beta}_{i,m}^k = \beta_{i,m}, \forall k \in \mathcal{K}, i \in \mathcal{K}, m \in \mathcal{M}_i$ should be satisfied. Then, problem $P2''$ can be transformed into $P2'''$, i.e.,

$$P2''': \quad \max_{\alpha,\tilde{\beta},\gamma,\tilde{\zeta};\beta} \sum_{k=1}^{K}\sum_{m=1}^{M_k} U'''_{k,m}, \tag{32}$$

$$\begin{aligned}
s.t. \quad & C1: \sum_{n=1}^{N_{k,m}} \tilde{\zeta}_{k,m,n} = \gamma_{k,m},\\
& C2: \tau + T_x + \alpha_{k,m}T_{k,m}^{loc} \le T_{k,m}^{max},\\
& C3: \tau + T_x + \tilde{\beta}_{k,m}^k T_{k,m}^{mec} \le T_{k,m}^{max},\\
& C4: \tau + T_x + \tilde{\zeta}_{k,m,n}(\frac{d_{k,m}}{r_{k,m}^n} + \frac{l_{k,m}}{f_{k,m}^n}) \le T_{k,m}^{max},\\
& C5: \tilde{\zeta}_{k,m,n}d_{k,m} \le \aleph_{k,m,n},
\end{aligned}$$

$$U_{k,m}''' = \left\{ \begin{array}{l} AoI^{pro} d_{k,m} - \alpha_{k,m}(\frac{l_{k,m} d_{k,m}}{f_{k,m}^{loc}} + l_{k,m}\delta_{k,m}) - \tilde{\zeta}_{k,m,n'} d_{k,m}(\frac{d_{k,m}}{r_{k,m}^{n'}} + \frac{l_{k,m}}{f_{k,m}^{n'}}) - \\ \sum_{n=1}^{N_{k,m}} \tilde{\zeta}_{k,m,n}(\frac{d_{k,m} P_{k,m}\mathfrak{r}}{r_{k,m}^{n}} + l_{k,m}\delta_n) - \tilde{\beta}_{k,m}^{k}[(\frac{d_{k,m}}{r_{k,m}^{k}} + \frac{l_{k,m}}{f_{k,m}^{k}}) d_{k,m} + \frac{d_{k,m}}{r_{k,m}^{k}} P_{k,m}\mathfrak{r} + l_{k,m}\delta_k] \end{array} \right\}, \tag{33}$$

$$\begin{aligned} &C6: \sum_{i=1}^{K}\sum_{m=1}^{M_i} \tilde{\beta}_{i,m}^{k} l_{k,m} \le \varsigma, \forall k \in \mathcal{K}, \\ &C7: (f_{k,m}^{n_1} - f_{k,m}^{n_2})[\tilde{\zeta}_{k,m,n_1}(\frac{d_{k,m}}{r_{k,m}^{n_1}} + \frac{l_{k,m}}{f_{k,m}^{n_1}}) - \\ &\quad \tilde{\zeta}_{k,m,n_2}(\frac{d_{k,m}}{r_{k,m}^{n_2}} + \frac{l_{k,m}}{f_{k,m}^{n_2}})] \ge 0, \forall n_1, n_2 \in \mathcal{N}_{k,m}, \\ &C8: \alpha_{k,m}, \gamma_{k,m}, \tilde{\zeta}_{k,m,n} \in [0,1], \\ &C9: \tilde{\beta}_{i,m}^{k} \in [0,1], \forall k, i \in \mathcal{K}, m \in \mathcal{M}_i, \\ &C10: \tilde{\beta}_{i,m}^{k} = \beta_{i,m}, \forall k, i \in \mathcal{K}, m \in \mathcal{M}_i. \end{aligned}$$

Herein, $U_{k,m}'''$ is an equivalent form of $U_{k,m}''$ (i.e., replacing $\beta_{k,m}$ with $\tilde{\beta}_{k,m}^{k}$), and can be calculated by (33). CM n' has the largest CPU frequency of CH m. It is worth noticing that MEC servers are decoupled from each other due to the introduction of local copy $\tilde{\beta}^k$. For the sake of convenience, we define the feasible region of constraints $C1-C9$ in $P2'''$ for MEC server k as $\mathcal{X}_k = \{\alpha^k, \tilde{\beta}^k, \gamma^k, \tilde{\zeta}^k | C1-C9 \text{ in } P2'''\}$. The overall difference between the computing cost and the reward of all the CHs connecting to MEC server k in data processing can be expressed by:

$$\Gamma_k = \left\{ \begin{array}{ll} \sum_{m=1}^{M_k} -U_{k,m}''', & \text{if } \{\alpha^k, \tilde{\beta}^k, \gamma^k, \tilde{\zeta}^k\} \in \mathcal{X}_k, \\ \infty, & \text{otherwise.} \end{array} \right. \tag{34}$$

Finally, problem $P2'''$ is transformed into:

$$\begin{aligned} P2^{(4)}: \quad & \min_{\alpha,\tilde{\beta},\gamma,\tilde{\zeta};\beta} \sum_{k=1}^{K} \Gamma_k(\alpha^k, \tilde{\beta}^k, \gamma^k, \tilde{\zeta}^k), \\ s.t. \quad & C10 \ in \ P2'''. \end{aligned} \tag{35}$$

Since the Alternating Direction Method of Multiplier (ADMM) method [25] can effectively solve the complicated convex optimization problem by decomposing the problem into multiple simple subproblems with good convergence, we design an ADMM-based distributed method to solve problem $P2^{(4)}$. Similar to [26], the augmented Lagrangian function of problem $P2^{(4)}$ can be denoted by:

$$\begin{aligned} &\mathcal{L}_\epsilon(\alpha, \tilde{\beta}, \gamma, \tilde{\zeta}, \beta, \Lambda) = \sum_{k=1}^{K} \Gamma_k(\alpha^k, \tilde{\beta}^k, \gamma^k, \tilde{\zeta}^k) + \\ &\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{m=1}^{M_i} [\Lambda_{i,m}^{k}(\tilde{\beta}_{i,m}^{k} - \beta_{i,m}) + \frac{\epsilon}{2}(\tilde{\beta}_{i,m}^{k} - \beta_{i,m})^2], \end{aligned} \tag{36}$$

where $\Lambda_{i,m}^{k}$ is the corresponding Lagrange multiplier, and $\Lambda = \{\Lambda^k\} = \{\Lambda_{i,m}^{k}\}, \forall k \in \mathcal{K}, i = 1, ..., K, m = 1, ..., M_i$. Symbol ϵ is a constant to represent the penalty parameter, influencing the convergence speed of ADMM.

The local variable iteration process is expressed by:

$$\{\Phi^k\}^{(t+1)} = \arg\min_{\{\Phi^k\}} \left\{ \underbrace{\begin{array}{c} \Gamma_k(\Phi^k) + \sum_{i=1}^{K}\sum_{m=1}^{M_i} [\Lambda_{i,m}^{k}]^{(t)} \{\tilde{\beta}_{i,m}^{k} - [\beta_{i,m}]^{(t)}\} + \\ \frac{\epsilon}{2}\sum_{i=1}^{K}\sum_{m=1}^{M_i} \{\tilde{\beta}_{i,m}^{k} - [\beta_{i,m}]^{(t)}\}^2 \end{array}}_{\Psi(\Phi^k)} \right\}. \tag{37}$$

Herein, $\{\Phi^k\}$ is the set of local variables, i.e., $\{\Phi^k\} = \{\alpha_{k,m}, \tilde{\beta}_{i,m}^{k}, \gamma_{k,m}, \tilde{\zeta}_{k,m,n}\}$, and $k = 1, ..., K$. The iterative process in (37) can be achieved in each MEC server by solving the following problem:

$$\begin{aligned} P2^{(5)}: \quad & \min_{\Phi^k} \Psi(\Phi^k), \\ s.t. \quad & \{\alpha^k, \tilde{\beta}^k, \gamma^k, \tilde{\zeta}^k\} \in \mathcal{X}_k. \end{aligned} \tag{38}$$

It can be decomposed into two subproblems. For the first one,

$$\begin{aligned} P2_1^{(5)}: \quad & \min_{\alpha^k, \gamma^k, \tilde{\zeta}^k} \varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k), \\ s.t. \quad & C': \alpha_{k,m} + \gamma_{k,m} = 1, \forall m \in \mathcal{M}_k, \\ & C1, C2, C4, C5, C7, C8 \ in \ P2''', \end{aligned} \tag{39}$$

where $\varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k)$ is expressed by:

$$\varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k) = \left\{ \begin{array}{l} \sum_{m=1}^{M_k} \{\alpha_{k,m}(\frac{l_{k,m} d_{k,m}}{f_{k,m}^{loc}} + l_{k,m}\delta_{k,m}) + \\ \tilde{\zeta}_{k,m,n'} d_{k,m}(\frac{d_{k,m}}{r_{k,m}^{n'}} + \frac{l_{k,m}}{f_{k,m}^{n'}}) + \\ \sum_{n=1}^{N_{k,m}} \tilde{\zeta}_{k,m,n}(\frac{d_{k,m} P_{k,m}\mathfrak{r}}{r_{k,m}^{n}} + l_{k,m}\delta_n)\} \end{array} \right\}. \tag{40}$$

Since $AoI^{pro} d_{k,m}$ is a constant, we do not consider it in the optimization objective of problem $P2_1^{(5)}$, and two computation modes are considered, i.e., local computing and offloading to CMs. Constraint C' in $P2_1^{(5)}$ ensures the validation of decision variables α^k and γ^k. It is obvious that problem $P2_1^{(5)}$ is a linear programming problem, which can be solved in polynomial time.

For the second subproblem $P2_2^{(5)}$, each MEC server needs to determine the optimal selection for CHs about whether to offload tasks to MEC servers. In order to encourage CHs to offload tasks to MEC servers, we design an incentive mechanism C'' in $P2_2^{(5)}$, i.e.,

$$C'' = \left\{ \begin{array}{c} \sum_{m=1}^{M_k} \tilde{\beta}_{k,m}^{k} \Upsilon_{k,m} = \varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k), \text{if} \\ \sum_{m=1}^{M_k} \Upsilon_{k,m} > \varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k), \\ \sum_{i=1}^{K}\sum_{m=1}^{M_i} \tilde{\beta}_{i,m}^{k} l_{k,m} = \varsigma, \quad \text{otherwise,} \end{array} \right. \tag{41}$$

and

$$\Upsilon_{k,m} = (\frac{d_{k,m}}{r^k_{k,m}} + \frac{l_{k,m}}{f^k_{k,m}})d_{k,m} + \frac{d_{k,m}}{r^k_{k,m}}P_{k,m}\mathfrak{r} + l_{k,m}\delta_k, \quad (42)$$

where the first part and the rest in (42) are the time cost (token) and computing cost (token) of CH m by selecting the mode of offloading to MEC servers, respectively. Symbol $\varphi_1(\alpha^k, \gamma^k, \tilde{\zeta}^k)$ is a constant after solving $P2_1^{(5)}$, which can be regarded as the sum costs of time and computing of CH m by selecting local computing and offloading to CMs.

The second subproblem of $P2_2^{(5)}$ is expressed by:

$$\begin{aligned} P2_2^{(5)}: \quad & \min_{\tilde{\beta}^k} \varphi_2(\tilde{\beta}^k), \quad (43)\\ s.t. \quad & \text{Constraint } C'', \\ & C3, C6, C9 \ in \ P2''', \end{aligned}$$

where objective function $\varphi_2(\tilde{\beta}^k)$ is denoted by:

$$\varphi_2(\tilde{\beta}^k) = \left\{ \begin{array}{l} \sum\limits_{m=1}^{M_k} \tilde{\beta}^k_{k,m}[(\frac{d_{k,m}}{r^k_{k,m}} + \frac{l_{k,m}}{f^k_{k,m}})d_{k,m} + \frac{d_{k,m}}{r^k_{k,m}}P_{k,m}\mathfrak{r} + \\ l_{k,m}\delta_k] + \sum\limits_{i=1}^{K}\sum\limits_{m=1}^{M_i}\{[\Lambda^k_{i,m}]^{(t)}\{\tilde{\beta}^k_{i,m} - [\beta_{i,m}]^{(t)}\} + \\ \frac{\epsilon}{2}\{\tilde{\beta}^k_{i,m} - [\beta_{i,m}]^{(t)}\}^2\} \end{array} \right\}. \quad (44)$$

We can observe that problem $P2_2^{(5)}$ is a quadratic programming problem and can be resolved in polynomial time.

The iteration process of global variable β is denoted by:

$$\{\beta\}^{(t+1)} = \arg\min_{\{\beta_{i,m}\}} \left\{ \underbrace{\sum_{k=1}^{K}\sum_{i=1}^{K}\sum_{m=1}^{M_i}\{[\Lambda^k_{i,m}]^{(t)}\{[\tilde{\beta}^k_{i,m}]^{(t+1)} - \beta_{i,m}\} + \frac{\epsilon}{2}\{[\tilde{\beta}^k_{i,m}]^{(t+1)} - \beta_{i,m}\}^2\}}_{\phi(\{\beta_{i,m}\})} \right\}. \quad (45)$$

Similar to the update of local variables, we need to solve a quadratic optimization problem to update global variable β, which can be expressed as:

$$P2^{(6)}: \quad \min_{\{\beta_{i,m}\}} \phi(\beta_{i,m}). \quad (46)$$

Its objective function is strictly convex due to the added quadratic regularization term in the augmented Lagrangian in (36). The optimal solution of problem $P2^{(6)}$ can be obtained by making the first-order derivative of the objective function equal to 0, since there is no constraint for global variable β. Then the optimal solution becomes:

$$[\beta_{i,m}]^{(t+1)} = \frac{1}{\epsilon K}\sum_{k=1}^{K}[\Lambda^k_{i,m}]^{(t)} + \frac{1}{K}\sum_{k=1}^{K}[\tilde{\beta}^k_{i,m}]^{(t+1)}. \quad (47)$$

The update of global variable β needs the values of Λ^k and $\tilde{\beta}^k$ of all MEC servers. We assume each MEC server maintains a global variable β^k and updates it by broadcasting variables Λ^k and $\tilde{\beta}^k$ to other MEC servers after updating local variable Φ^k. Note that, since all MEC servers have the same sets of $\{\Lambda^k\}$ and $\{\tilde{\beta}^k\}$ after broadcasting, condition $\beta^1 = ...\beta^k = ...\beta^K = \beta$ always holds in equation (47).

The iteration process of Lagrange multipliers Λ can be updated in each MEC server, which can be expressed by:

$$[\Lambda^k_{i,m}]^{(t+1)} = [\Lambda^k_{i,m}]^{(t)} + \epsilon\{[\tilde{\beta}^k_{i,m}]^{(t+1)} - [\beta_{i,m}]^{(t+1)}\}. \quad (48)$$

For ADMM algorithm, a reasonable stopping criterion should meet two conditions, i.e., the primal and dual residuals are small enough [27]. In our work, it should satisfy:

$$\left\|[\tilde{\beta}^k]^{(t+1)} - [\beta^k]^{(t+1)}\right\|_2 \leq \mathfrak{z}^{pri}, \forall k \in \mathcal{K}, \quad (49)$$

$$\left\|[\beta^k]^{(t+1)} - [\beta^k]^{(t)}\right\|_2 \leq \mathfrak{z}^{dua}, \forall k \in \mathcal{K}. \quad (50)$$

Herein, both $\mathfrak{z}^{pri}$ and $\mathfrak{z}^{dua}$ are feasibility tolerances for primal and dual feasibility conditions, respectively.

Proposition 5. *The proposed DIADEM algorithm satisfies the residual convergence, objective convergence and dual variable convergence simultaneously.*

The proof can be found in Appendix E of Supplementary File.

Since the variables of problem $P2'$ (i.e., α, β, γ) are relaxed from binary variables to real numbers within a closed interval from 0 to 1, a recovery algorithm is further proposed to recover α, β and γ to binary variables after solving problem $P2^{(4)}$. Variables α and γ are first recovered by comparing the values of $\alpha^*_{k,m}$ and $\gamma^*_{k,m}$ with 0.5. After that, we check the validation of α and γ, and recover variable β by verifying $C4$ as well as $C8$ in $P2'$ and comparing the first-order partial derivatives of the augmented Lagrangian function (i.e., $\frac{\partial \mathcal{L}_\epsilon}{\partial \alpha^*_{k,m}}$ and $\frac{\partial \mathcal{L}_\epsilon}{\partial \tilde{\zeta}^*_{k,m,n}}$ with $\frac{\partial \mathcal{L}_\epsilon}{\partial \beta^*_{k,m}}$). Finally, variable ζ is recovered as described in the second paragraph of this subsection. In the binary variable recovery algorithm, we do not check the constraints about γ, i.e., $C3, C6, C7$ and $C9$ in $P2'$. The reason is that $\tilde{\zeta}_{k,m,n} = \gamma_{k,m}\zeta_{k,m,n}$ holds, and those constraints can be satisfied in the solving process of problem $P2^{(4)}$, even after recovering variables γ and ζ. The pseudo-codes of the DIADEM algorithm and binary variable recovery algorithm can be found in Appendixes F and G of Supplementary File, respectively.

Proposition 6. *Our proposed algorithms can be solved with polynomial time.*

The proof can be found in Appendix H of Supplementary File.

5 Performance Evaluation

In this section, we evaluate the performance of our algorithms. The objectives of two subproblems are to optimize the overall system objectives, and they are involved in two processes, i.e., data collection and data processing, respectively. The two subproblems are correlative in time dimension, and are solved in order. Overall, the formulated multi-objective optimization problem can obtain the optimal solution when each subproblem gets its optimal solution. Therefore, we evaluate the performance of the proposed two algorithms independently, and the corresponding parameter settings can be found in Appendix I of Supplementary File.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

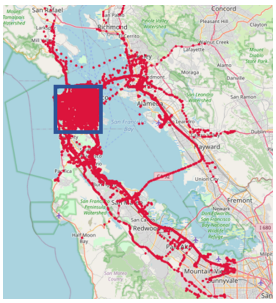


Fig. 5. The mobility traces of 300 vehicles in one hour.

5.1 Simulation Setup

We evaluate the presented methods in a 64 bit Window 10 operating system computer, which has a 16.0 GB RAM, an Intel(R) Core(TM) i7-8700 CPU with 3.20 GHz frequency, and an NVIDIA GeForce GTX 1050.

For subproblem $P1$, the corresponding optimization objectives are to maximize the blockchain safety and minimize the blockchain latency. The evaluation metrics are blockchain safety, blockchain latency, the average computing power of the selected active miners, and the rate of reliable active miners. Five representative schemes are leveraged for comparison:

- DRL-based Performance Optimization Framework (we call it DPOF) [23]: A DRL-based solution is leveraged to select active miners and block sizes to maximize the blockchain throughput by considering the decentralization of active miners, blockchain latency and blockchain security.
- Multi-Weight Subjective Logic (MWSL) scheme [4]: The reputation of each RSU is evaluated by leveraging a multi-weight subjective logic method based on the interaction frequency, the timeliness and effect of the interactions between RSUs and vehicles. Then, active miners are selected from RSUs based on their reputation.
- Traditional Subjective Logic (TSL) scheme [28]: A linear function is designed to calculate the reputation of RSUs, by which vehicles vote active miners.
- DPoS scheme [24]: Vehicles evaluate the reputation of RSUs by leveraging local opinions based on the interactions between vehicles and RSUs to select active miners.
- Computing power-based scheme: Active miners are selected from RSUs based on their computing power.

For subproblem $P2'$, the corresponding optimization objective is to maximize the total utility of all users by selecting proper task computation modes, and evaluation metrics are the social welfare and the number of CHs selecting different task computation modes. Five representative methods are leveraged to compare with our DIADEM algorithm:

- Optimal solution: Traversing all possible solutions and searching out the optimal one with the maximum total utility to illustrate the upper bound of the social welfare.
- Local computing: Each CH processes its task locally.
- Offloading to MEC servers: Each CH processes its task by offloading to MEC servers.

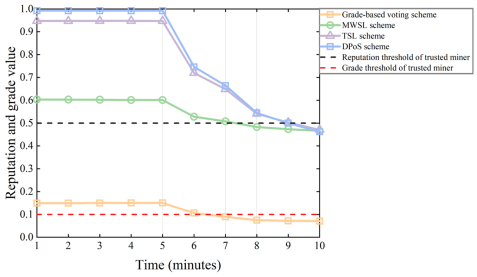


Fig. 6. The reputation and grade value of a malicious RSU.

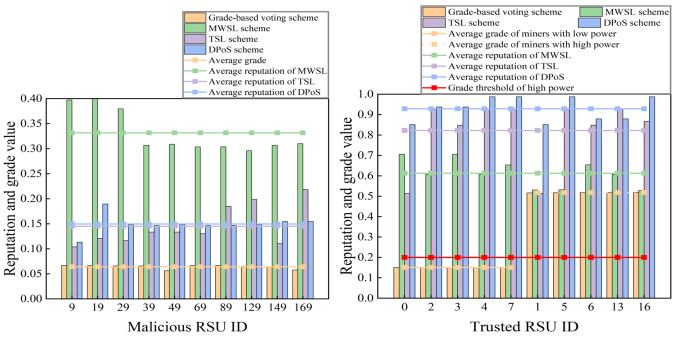


Fig. 7. The reputation and grade values for: (a) Ten malicious RSUs, and (b) Ten trusted RSUs.

- Offloading to CMs: Each CH processes its task by offloading to its CMs.
- LCO to CMs: Each CH processes its task by local or offloading to its CMs (i.e., the subprocess of our DIADEM algorithm) to maximize the total utility.

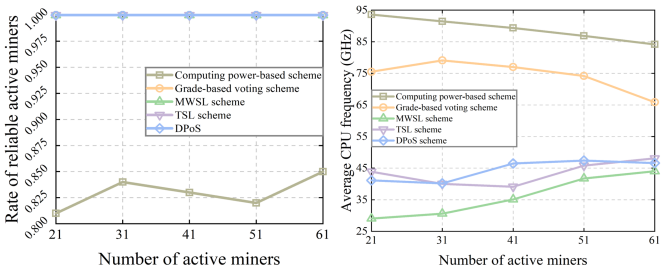
5.2 Performance of Problem P1

We first evaluate the performance of the grade-based voting mechanism based on the real-world dataset of San Francisco Yellow Cab [4]. The dataset contains the moving traces of 536 taxis in one month in 2011. We randomly select 300 vehicles and mainly focus on the traces of vehicles in the dense trajectory area, whose latitude and longitude are from 37.708 to 37.8 and -122.502 to -122.408, respectively. The mobility traces of these vehicles in one hour are illustrated in Fig. 5, and the research area is surrounded by the blue box. We consider there are 200 RSUs randomly distributed in the research area, and 10 malicious RSUs pretend to behave well in the beginning 5 minutes.

Fig. 6 shows the given reputation and the grade value by different schemes to a malicious RSU when it pretends to behave well in the beginning 5 minutes. We can observe that our grade-based voting scheme can detect the malicious RSU at the 7 minutes, while the MWSL, TSL and DPoS schemes can detect it at the 8, 10 and 10 minutes, respectively. Therefore, our grade-based voting scheme can detect malicious RSUs more promptly than other methods.

Figs. 7(a) and 7(b) illustrate the given reputation and the grade value for malicious and trusted RSUs, respectively. From Fig. 7(a), we can observe that all the four schemes can identify malicious RSUs, because the reputation values of malicious RSUs given by MWSL, TSL and DPoS are lower than the reputation threshold of trusted miner (i.e., 0.5), and the grade value given by the grade-based voting scheme is lower than the grade threshold of trusted miner (i.e., 0.1). However, our grade-based voting scheme has a better

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing



(a) (b)

Fig. 8. The performance for different numbers of active miners: (a) Rate of reliable active miners, and (b) Average CPU frequency.

TABLE 3
Experimental results for different number of RSUs

Metric	Method	Number of RSUs			
		20	30	40	50
Number of selected active miners	Our solution	**17**	**28**	**38**	**47**
	DPOF	15	21	32	40
Block size (KB)	Our solution	**100.1**	**132.3**	**134.4**	**137.9**
	DPOF	124.0	140.0	156.0	160.0
Blockchain latency (s)	Our solution	**1.1179**	**1.5627**	**1.8619**	**2.1791**
	DPOF	1.2392	1.6085	1.9095	2.3753

stability than the other methods for 10 malicious RSUs, since its grade variance is smaller than the corresponding reputation variances in MWSL, TSL and DPoS. In Fig. 7(b), the trusted RSUs numbered by 0, 2, 3, 4, 7 have weak computing power, while the ones numbered by 1, 5, 6, 13, 16 have strong computing power. Although all the five schemes can distinguish the trusted RSUs, our scheme can also evaluate the computing power of RSUs. This is because the grade values of the RSUs with strong computing power are higher than the grade threshold of high computing power (i.e., the red line in Fig. 7(b)), and vice versa.

Figs. 8(a) and 8(b) demonstrate the rate of reliable active miners and the average computing power (i.e., CPU frequency) of active miners, respectively. Fig. 8(a) shows that the computing power-based scheme can select some malicious RSUs as active miners, while all the selected active miners of the other methods are reliable. The reason is that it only considers the CPU frequency of RSUs when selecting active miners. However, the other methods can distinguish malicious RSUs and select reliable ones. From Fig. 8(b), we can find that the average CPU frequency of the active miners selected by our grade-based voting scheme is higher than that of MWSL, TSL and DPoS. This is because our scheme can evaluate the computing power of RSUs, while others can merely evaluate the reputation of RSUs. By jointly considering Figs. 8(a) and 8(b), we can conclude that the performance of our grade-based voting scheme is better than that of other schemes, since it can select reliable RSUs with the maximum computing power as active miners.

TABLE 3 illustrates the number of selected active miners (equivalent to the blockchain safety based on equation (1)), the block size and the blockchain latency for various numbers of RSUs. It is obvious that our solution can select more active miners, generate smaller blocks and lower blockchain latency than DPOF. The reason is that the optimization objective of DPOF is to maximize the blockchain throughput by considering the decentralization of active miners, blockchain latency and blockchain security, and it prefers to select less active miners and more transactions to maximize the blockchain throughput. Therefore, our DRL-based algorithm can guarantee a more secure and lower latency blockchain than DPOF.

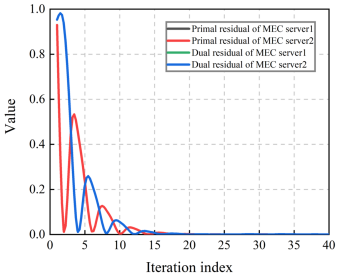
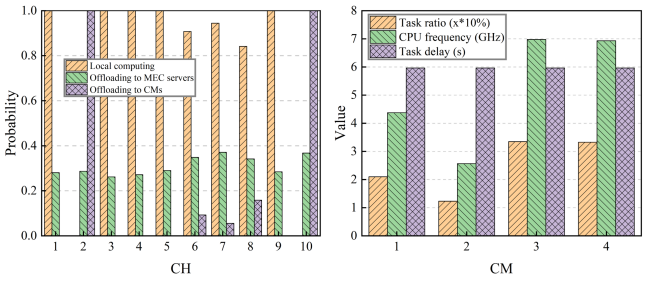


Fig. 9. Convergence performance of DIADEM algorithm.



(a) (b)

Fig. 10. Experimental results of DIADEM algorithm: (a) Probability of selecting the three computing fashions of CHs, and (b) Task ratio, CPU frequency and task delay of CMs in CH 10.

5.3 Performance of Problem P2'

The convergence performance of DIADEM in different MEC servers is shown in Fig. 9. We can observe that MEC servers 1 and 2 have the same convergence tendency, since they have the same primal and dual residuals. The difference between the primal and dual residuals of each MEC server decreases with the increase of iterations. Fig. 9 demonstrates our DIADEM algorithm can reach convergence within 20 iterations.

It is obvious that CHs mainly select computation modes from local computing and offloading to CMs, as shown in Fig. 10(a). The reason is that the price provided by the MEC server is significantly high, leading to the probability of offloading to MEC servers lower than 0.5 for each CH. CHs 1, 3, 4, 5 and 9 choose local computing with probability 1, which indicates they can meet all the constraints of local computing in binary variable recovery. For CHs 6, 7, and 8, they do not satisfy constraint $C4$ in $P2'$. Offloading to CMs is selected by CHs to meet the required constraints, since they can obtain larger social welfare compared to that obtained by offloading to MEC servers. Task ratio $\tilde{\zeta}_{k,m,n}$, CPU frequency $f^n_{k,m}$ and task delay $T^{cm}_{k,m,n}$ of CMs in CH 10 are demonstrated in Fig. 10(b). To meet constraint $C3$ in $P2'$, the total task ratio of all CMs equals to the probability that CH 10 selects the mode of offloading to CMs. When CHs offload their tasks to CMs, they prepare to reduce their task delay. Finally, all CMs have the same task delay, since the CM with a higher CPU frequency has a larger task ratio.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3079984, IEEE Transactions on Mobile Computing

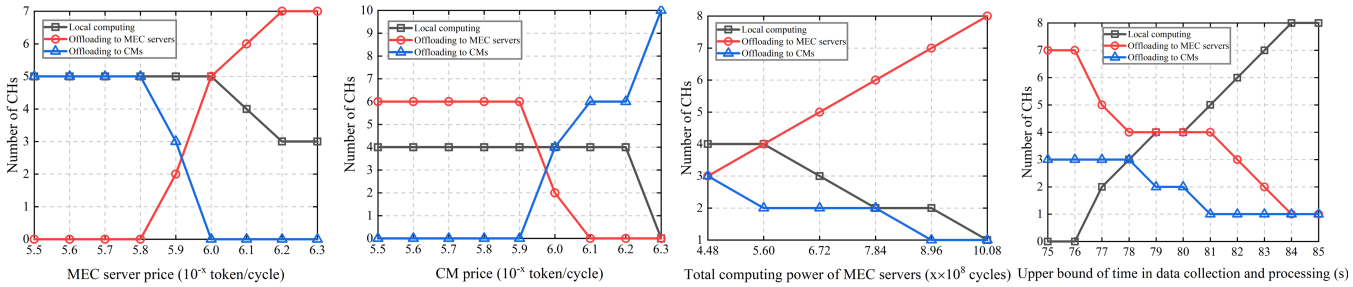


Fig. 11. Experimental performance of DIADEM algorithm for different metrics: (a) MEC server price, (b) CM price, (c) Total computing power of MEC servers, and (d) Upper bound of time in data collection and processing.

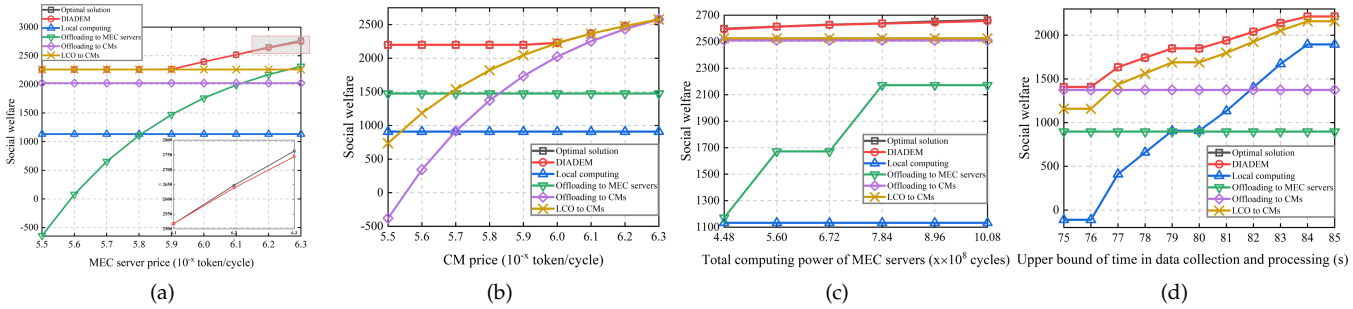


Fig. 12. Social welfare comparison for different metrics: (a) MEC server price, (b) CM price, (c) Total computing power of MEC servers, and (d) Upper bound of time in data collection and processing.

The experimental results of DIADEM for MEC server price δ_k, CM price δ_n, total computing power ς of MEC servers, and upper bound of time $T_{k,m}^{max}$ in data collection and processing are shown in Fig. 11. From Fig. 11(a), where δ_n=10^{-6} token, $T_{k,m}^{max}$=81 seconds and ς=$8.96 * 10^8$ cycles, we can observe that when δ_k is no less than $10^{-5.8}$, CHs mainly choose local computing and offloading to CMs, since the price of MEC server is rather expensive and brings an enormous task computing cost. When δ_k is within $10^{-5.8}$ and $10^{-6.0}$, the number of CHs selecting the mode of offloading to CMs decreases, while that of selecting the mode of offloading to MEC servers increases, since offloading to MEC servers brings larger social welfare than that brought by offloading to CMs. When δ_k is smaller than $10^{-6.0}$, the mode of offloading to MEC servers is better than that of local computing. After δ_k reaches $10^{-6.2}$, even reducing the value of δ_k, the number of CHs selecting the mode of offloading to MEC servers does not change, due to the limited total computing power of MEC servers.

For various δ_n, the experimental results of DIADEM are illustrated in Fig. 11(b), where δ_k=$10^{-5.9}$ token, $T_{k,m}^{max}$ =81 seconds, ς=$8.96 * 10^8$ cycles. When δ_n is no less than $10^{-5.9}$, CHs prefer to choose the modes of local computing and offloading to MEC servers, because offloading to CMs needs to pay expensive prices. When δ_n is between $10^{-5.9}$ and $10^{-6.1}$, the mode of offloading to CMs begins to increase social welfare. When δ_n is less than $10^{-6.2}$, offloading to CMs brings more social welfare than local computing, and all CHs select offloading to CMs when δ_n reaches $10^{-6.3}$. Fig. 11(c), where δ_n=$10^{-6.25}$, δ_k=$10^{-6.2}$ token, $T_{k,m}^{max}$=81 seconds, illustrates the experimental results for different ς. It is obvious that the number of CHs offloading to MEC servers increases with the increase of ς, while that of selecting local computing and offloading to CMs decreases with the increase of ς. The reason is that offloading to MEC servers enhances social welfare, and ς limits the upper bound of the number of CHs offloading to MEC servers. Similar to Fig. 11(c), the number of CHs selecting local computing increases with the enhancement of $T_{k,m}^{max}$, while that of offloading to MEC servers and CMs decreases in Fig. 11(d), where δ_n=$10^{-5.8}$ token , δ_k=$10^{-5.75}$ token, ς=$8.96*10^8$ cycles, because choosing local computing brings larger social welfare than the other two modes.

The social welfare for different metrics is illustrated in Fig. 12. For different δ_k, the corresponding social welfare is shown in Fig. 12(a). We can observe that the social welfare of local computing, offloading to CMs, and the combination of local computing and offloading to CMs (i.e., LCO to CMs) is constant. The reason is that the varied δ_k has no influence on those three methods. The social welfare of LCO to CMs is larger than that of local computing and offloading to CMs, since only selecting local computing leads to the task computing delay of some CHs violating constraint $C4$ in $P2'$, i.e., their task computing delay makes the time in data collection and processing exceeds the upper bound $T_{k,m}^{max}$, and the expensive price of CMs needs to be paid by offloading to CMs. The social welfare of offloading to MEC servers increases with the decrease of δ_k. When δ_k equals to $10^{-5.5}$, the social welfare of offloading to MEC servers is negative. Although task computing delay is low to obtain a larger reward, CHs need to pay for the MEC server with a high price, resulting in an enormous task cost. When δ_k is no less than $10^{-5.8}$, LCO to CMs obtains the same social welfare with that of DIADEM and the optimal solution. The reason is that the price of the MEC server is rather high, and CHs select computation modes from local computing and offloading to CMs, rather than offloading to MEC servers. When δ_k is less than $10^{-5.8}$, the social welfare of LCO to CMs is lower than that of DIADEM and the optimal solution, because offloading to MEC servers can generate larger social welfare. When the MEC server price is $10^{-6.3}$

token per cycle, the performance gap between DIADEM and the optimal solution is 19.26 token, and the performance difference between DIADEM and the optimal solution decreases with the increase of MEC server prices. The social welfare for different δ_n is demonstrated in Fig. 12(b). Similar with Fig. 12(a), the social welfare of local computing and offloading to MEC servers is constant. Obviously, the social welfare of offloading to CMs and LCO to CMs increases with the decrease of δ_n. When δ_n is more than $10^{-6.1}$, the performance of DIADEM and the optimal solution is better than that of LCO to CMs. The reason is that some CHs select offloading to MEC servers and generate larger social welfare in DIADEM and the optimal solution. The social welfare of LCO to CMs equals to that of DIADEM and the optimal solution, when δ_n is no more than $10^{-6.1}$, because all CHs choose task computation modes from local computing and offloading to CMs. When δ_n equals to $10^{-6.3}$, offloading to CMs can generate the same social welfare with that in DIADEM, optimal solution and LCO to CMs. The reason is that all CHs select offloading to CMs. From Fig. 12(c), we can observe that local computing, offloading to CMs and LCO to CMs contribute constant social welfare for various ς. Our DIADEM has better performance than that of the other four methods (expect the optimal solution), because it can select a suitable computing mode for each CH. The social welfare of DIADEM and the optimal solution increases with the increase of ς, since a larger ς allows more CHs to select the mode of offloading to MEC servers. The method of offloading to MEC servers can obtain larger social welfare with the increase of ς. The reason is that if existing MEC servers cannot meet the computing requirements of all CHs, additional MEC servers need to be deployed, resulting in extra costs. . For different $T_{k,m}^{max}$, the social welfare of various methods is shown in Fig. 12(d). We notice that the social welfare obtained by offloading to CMs and offloading to MEC servers is constant for different $T_{k,m}^{max}$. The reason is that the delay for data collection and processing is lower than 75 seconds due to the task decomposition in CMs and the strong computing capacities of MEC servers. The performance of other methods can be improved with the increase of $T_{k,m}^{max}$. The reason is that a larger $T_{k,m}^{max}$ allows more CHs to select the mode of local computing with larger social welfare. The performance of DIADEM is better than that of local computing and LCO to CMs, because DIADEM can choose a suitable task computation mode for each CH by considering all constraints in $P2'$. Therefore, DIADEM has better performance compared with LCO to CMs, demonstrating the effectiveness of our proposed incentive mechanism.

6 Related Work

In this section, we mainly review the state-of-the-art researches for blockchain-enabled resource management.

Tseng *et al.* proposed a blockchain-based architecture to manage the resource of heterogeneous Internet of things, in which devices are able to interact with others to reduce energy consumption, communication and computation overhead [29]. In order to establish a secure remote user authentication for industrial 4.0 applications, a blockchain-based framework was proposed by leveraging an attribute-based signature scheme, a multi-receiver encryption method and a mac generation algorithm [30]. The authors in [31] considered an MEC-based public blockchain, where miners offload their computation tasks to MEC servers. The interactions between the service provider and miners are modeled as a stochastic Stackelberg game, and the reinforcement learning technique is leveraged to make decisions for the service provider. In order to enable the data of owners can be outsourced to cloud for secure storage, a privacy preserving calculation toolkit was proposed in [32]. Shen *et al.* developed a reliable data sharing platform, including data owners, miners and third parties in a cloud environment, by leveraging the blockchain technique to guarantee data security [13].

Xu *et al.* proposed a blockchain-enabled big data sharing framework to reduce the required storage and computing resources for network edges [5]. The authors in [33] proposed a credit-based incentive mechanism by using the improved model of population dynamics to prevent the attack of node compromising and encourage vehicles to forward packages. Jiang *et al.* investigated how to leverage the blockchain technique in ITS by considering the distributed and secure data storage [6]. The authors in [4] proposed an improved delegated proof-of-stake consensus mechanism with a two-stage soft security solution for secure data sharing among vehicles. In the former, a reputation management method is designed by leveraging the weight subjective logic model to select active miners. In the latter, the standby miners with high reputation are stimulated to verify the block by using contact theory.

Different from these studies, we consider the safety and latency of blockchain, and strive for a satisfied trade-off between blockchain safety and latency, while guaranteeing the requirement of blockchain throughput.

7 Conclusion

In this paper, we put forward a secure, efficient and distributed ITS system, and formulated it as a multi-objective optimization problem, i.e., minimizing the system latency, maximizing the data safety and user utility. In order to solve the formulated problem, we decomposed it into two subproblems, and proposed two corresponding algorithms. The DRL-based algorithm can make a satisfied trade-off between blockchain security and latency, and the DIADEM algorithm is able to choose task computation modes for vehicles in a distributed way. Extensive experiments demonstrate the effectiveness of our algorithms, i.e., the DRL-based algorithm can reach higher blockchain safety and lower blockchain latency, and the DIADEM algorithm can obtain larger social welfare than benchmark methods.

8 Acknowledgements

This work was supported by the National Key Research and Development Program of China under Grant 2018YFE0206800, the Natural Science Foundation of China under Grants 62025105, 62001073, 61971084 and 61872310, the Chongqing Talent Program under Grant CQYC2020058659, the National Natural Science Foundation of Chongqing under Grant cstc2019jcyjmsxmX0208, the

funding from Hong Kong RGC Research Impact Fund (RIF) under Project R5060-19, in part by General Research Fund (GRF) under Project 152221/19E.

REFERENCES

[1] Z. Ning, P. Dong, X. Wang, X. Hu, J. Liu, L. Guo, B. Hu, R. Kwok, and V. C. Leung, "Partial computation offloading and adaptive task scheduling for 5G-enabled vehicular networks," *IEEE Transactions on Mobile Computing*, 2020.

[2] Z. Ning, P. Dong, X. Wang, X. Hu, L. Guo, B. Hu, Y. Guo, T. Qiu, and R. Y. Kwok, "Mobile edge computing enabled 5G health monitoring for Internet of medical things: A decentralized game theoretic approach," *IEEE Journal on Selected Areas in Communications*, 2020.

[3] P. K. Sharma, N. Kumar, and J. H. Park, "Blockchain-based distributed framework for automotive industry in a smart city," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4197–4205, 2018.

[4] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. Zhao, "Toward secure blockchain-enabled Internet of vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2906–2920, 2019.

[5] C. Xu, K. Wang, P. Li, S. Guo, J. Luo, B. Ye, and M. Guo, "Making big data open in edges: A resource-efficient blockchain-based approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 870–882, 2018.

[6] T. Jiang, H. Fang, and H. Wang, "Blockchain-based Internet of vehicles: distributed network architecture and performance analysis," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4640–4649, 2018.

[7] X. Liu, R. H. Deng, Y. Yang, H. N. Tran, and S. Zhong, "Hybrid privacy-preserving clinical decision support system in fog–cloud computing," *Future Generation Computer Systems*, vol. 78, pp. 825–837, 2018.

[8] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, and H. Qi, "Personalized privacy-preserving task allocation for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1330–1341, 2018.

[9] A. Dua, N. Kumar, A. K. Das, and W. Susilo, "Secure message communication protocol among vehicles in smart city," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4359–4373, 2017.

[10] Z. Wang, L. Liu, M. Zhou, and N. Ansari, "A position-based clustering technique for ad hoc intervehicle communication," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 201–208, 2008.

[11] X. Wang, Z. Ning, X. Hu, L. Wang, B. Hu, J. Cheng, and V. C. Leung, "Optimizing content dissemination for real-time traffic management in large-scale internet of vehicle systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1093–1105, 2018.

[12] X. Wang, Z. Ning, S. Guo, and L. Wang, "Imitation learning enabled task scheduling for online vehicular edge computing," *IEEE Transactions on Mobile Computing*, 2020.

[13] M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, and M. Guizani, "Blockchain-based incentives for secure and collaborative data sharing in multiple clouds," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1229–1241, 2020.

[14] J. Liu, J. Wan, D. Jia, B. Zeng, D. Li, C.-H. Hsu, and H. Chen, "High-efficiency urban traffic management in context-aware computing and 5G communication," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 34–40, 2017.

[15] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 94–100, 2017.

[16] L. Diez, A. Garcia-Saavedra, V. Valls, X. Li, X. Costa-Perez, and R. Aguero, "Lasr: A supple multi-connectivity scheduler for multi-rat OFDMA systems," *IEEE Transactions on Mobile Computing*, 2018.

[17] M. Salehi, H. Tabassum, and E. Hossain, "Meta distribution of sir in large-scale uplink and downlink NOMA networks," *IEEE Transactions on Communications*, vol. 67, no. 4, pp. 3009–3025, 2018.

[18] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *Proc. 3rd USENIX Symp. Operating Syst. Des. Implementation*, 1999, pp. 173–186.

[19] B. Liu, C. Liu, and M. Peng, "Resource allocation for energy-efficient mec in noma-enabled massive iot networks," *IEEE Journal on Selected Areas in Communications*, 2020.

[20] J. Zhao, Y. Liu, K. K. Chai, Y. Chen, and M. Elkashlan, "Joint subchannel and power allocation for noma enhanced d2d communications," *IEEE Transactions on Communications*, vol. 65, no. 11, pp. 5081–5094, 2017.

[21] X. Wang, Z. Ning, S. Guo, M. Wen, and V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, 2021.

[22] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[23] M. Liu, R. Yu, Y. Teng, V. Leung, and M. Song, "Performance optimization for blockchain-enabled industrial Internet of things (IIoT) systems: A deep reinforcement learning approach," *IEEE Transactions on Industrial Informatics*, 2019.

[24] G. Xu, Y. Liu, and P. W. Khan, "Improvement of the DPoS consensus mechanism in blockchain based on vague sets," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4252–4259, 2020.

[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[26] L. Chen, F. R. Yu, H. Ji, G. Liu, and V. C. Leung, "Distributed virtual resource allocation in small-cell networks with full-duplex self-backhauls and virtualization," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 7, pp. 5410–5423, 2015.

[27] M. Liu, F. R. Yu, Y. Teng, V. C. Leung, and M. Song, "Computation offloading and content caching in wireless blockchain networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11 008–11 021, 2018.

[28] X. Huang, R. Yu, J. Kang, Z. Xia, and Y. Zhang, "Software defined networking for energy harvesting Internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1389–1399, 2018.

[29] L. Tseng, L. Wong, S. Otoum, M. Aloqaily, and J. B. Othman, "Blockchain for managing heterogeneous Internet of things: A perspective architecture," *IEEE Network*, vol. 34, no. 1, pp. 16–23, 2020.

[30] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "Bsein: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *Journal of Network and Computer Applications*, vol. 116, pp. 42–52, 2018.

[31] A. Asheralieva and D. Niyato, "Learning-based mobile edge computing resource management to support public blockchain networks," *IEEE Transactions on Mobile Computing*, 2019.

[32] X. Liu, R. Deng, K.-K. R. Choo, Y. Yang, and H. Pang, "Privacy-preserving outsourced calculation toolkit in the cloud," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[33] J. Zhou, X. Dong, Z. Cao, and A. V. Vasilakos, "Secure and privacy preserving protocol for cloud-based vehicular DTNs," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1299–1314, 2015.

Zhaolong Ning received the M.S. and PhD degrees from Northeastern University, Shenyang, China. He was an associate professor with Dalian University of Technology, China. Currently, he is a full professor with Chongqing University of Posts and Telecommunications, China. He has published over 100 scientific papers in international journals and conferences. His research interests include Internet of things, mobile edge computing, and artificial intelligence.

Shouming Sun received B.S. degree from Dalian University of Technology, Dalian, China, in 2018. He is currently working toward the M.S. degree in the same university. His research interests include mobile edge computing, artificial intelligence and blockchain.

Xiaojie Wang received the PhD degree from Dalian University of Technology, China, in 2019. After that, she was a postdoctor in the Hong Kong Polytechnic University. Currently, she is a distinguished professor with the College of Communication and Information Engineering, the Chongqing University of Posts and Telecommunications, Chongqing, China. Her research interests are wireless networks, mobile edge computing and machine learning. She has published over 40 scientific papers in international journals and conferences, such as IEEE JSAC, IEEE TMC, IEEE TPDS and IEEE COMST.

Lei Guo received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2006. He is currently a Full Professor with Chongqing University of Posts and Telecommunications, Chongqing, China. He has authored or coauthored more than 200 technical papers in international journals and conferences. He is an editor for several international journals. His current research interests include communication networks, optical communications, and wireless communications.

Song Guo (F'20) received the PhD degree in computer science from the University of Ottawa and was a professor with the University of Aizu. He is a full professor with the Department of Computing, The Hong Kong Polytechnic University. His research interests include big data, cloud computing and networking, and distributed systems with more than 400 papers published in major conferences and journals. He is now on the editorial board of the IEEE Transactions on Emerging Topics in Computing, the IEEE Transactions on Sustainable Computing, the IEEE Transactions on Green Communications and Networking, and the IEEE Communications. He currently serves as an officer for several IEEE ComSoc Technical Committees and a director in the ComSoc Board of Governors.

Xiping Hu is a professor with Lanzhou University, China. He was the co-founder and CTO of Bravolol Limited in Hong Kong. He has around 90 papers published and presented in prestigious conferences and journals. His research areas consist of distributed intelligent systems, crowdsensing, social networks, and cloud computing. He holds a PhD from The University of British Colombia, Vancouver, Canada.

Bin Hu is currently a professor in Lanzhou University and a guest professor in ETH Zurich, Switzerland. He was a recipient of many research awards, including the 2014 China Overseas Innovation Talent Award, the 2016 Chinese Ministry of Education Technology Invention Award, the 2018 Chinese National Technology Invention Award, and the 2019 WIPO-CNIPA Award for Chinese Outstanding Patented Invention. He is also the TC Co-Chair of computational psychophysiology in the IEEE Systems, Man, and Cybernetics Society (SMC), the TC Co-Chair of cognitive computing in IEEE SMC, and the Vice-Chair of the TC 9.1. Economic, Business, and Financial Systems on Social Media at the International Federation of Automatic Control (IFAC). He is also a Member-at-Large of the ACM China Council and the Vice-Chair of the China Committee of the International Society for Social Neuroscience. He serves as the Editor-in-Chief for IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS and an Associate Editor for IEEE TRANSACTIONS ON AFFECTIVE COMPUTING.

Ricky Y. K. Kwok (F'14) received a B.Sc. degree in Computer Engineering from the University of Hong Kong in 1991, and the M.Phil. and Ph.D. degrees, both in Computer Science, from the Hong Kong University of Science and Technology (HKUST) in 1994 and 1997, respectively. His research focus has been on designing efficient communication protocols and robust resources management algorithms toward enabling large scale distributed mobile computing. In these research areas, he has authored one textbook, co-authored another two textbooks, and published more than 200 technical papers in various leading journals, research books, and refereed international conference proceedings. He is a Fellow of the HKIE, the IEEE, and the IET. From March 2006 to December 2011, Ricky served on the Editorial Board of the Journal of Parallel and Distributed Computing as a Subject Area Editor in Peer-to-Peer Computing. He also served as an Associate Editor for the IEEE Transactions on Parallel and Distributed Systems from January 2013 to December 2016.