information

Article

Resource Allocation in Decentralized Vehicular Edge Computing Network

Hongli Zhang [1,2] and Ying Li [1,2,*]

[1] College of Computer Science and Technology, Qingdao University, Qingdao 266071, China; 2020025820@qdu.edu.cn
[2] Institute of Ubiquitous Networks and Urban Computing, Qingdao University, Qingdao 266071, China
* Correspondence: yingli2016@qdu.edu.cn

Abstract: Computation-intensive vehicle tasks sharply increase with the rapid growth of intelligent vehicles. The technology of Mobile Edge Computing (MEC) has the possibility of assisting vehicles with computation offloading. To solve the problem of computation resource management and guarantee the security of resource transaction, we jointly combine the MEC network and the blockchain networks to build a blockchain based MEC offloading model. The non-cooperative interactions between MEC server and vehicles formulate a two-stage Stackelberg game in an aim to maximize their benefits and information security. We theoretically demonstrate the unique existence of Nash equilibrium, which enables participants to decide their optimal strategies. Finally, the performance of the proposed model is analyzed by conducting simulation experiments. Our proposed model optimizes resource allocation and also improves the security of the whole network.

Keywords: mobile edge computing; blockchain; stackelberg game; resource allocation

check for updates

Citation: Zhang, H.; Li, Y. Resource Allocation in Decentralized Vehicular Edge Computing Network. *Information* **2023**, *14*, 206. https://doi.org/10.3390/info14040206

Academic Editor: Luigi Laura

Received: 1 December 2022
Revised: 19 January 2023
Accepted: 23 March 2023
Published: 27 March 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The development of urban intelligent transportation incurs people's convenient travel. Many innovative vehicle applications have emerged, such as intelligent driving control, onboard online multimedia, image assisted navigation, and so on. Related researchers have conducted a lot of studies and experiments on its main techniques (e.g., gait recognition, Spatial-temporal prediction, and pedestrian flow prediction) [1–5]. The implementation of these technologies not only requires significant computing and caching resources, but also presents the serious challenge on limited energy a vehicle can carry. It is difficult for a vehicle to support such a huge computational workload and energy consumption during driving [6]. For good Quality of Service (QoS), relevant researchers have conducted more detailed and in-depth studies [7–11], which to some extent alleviate the computational workload and energy consumption of vehicles. However, these studies focused on optimizing the models and algorithms for vehicle applications instead of offloading the overloaded computational tasks of vehicles. The development of mobile edge computing (MEC) has led to a significant role in urban smart mobility [12]. It is a promising technology that allows vehicles to offload their local computing tasks to nearby MEC servers. So, the vehicle's own computational workload and energy consumption can be sharply reduced [13]. However, in order to maximize its revenue, the MEC server prefer to price the computing resources it provides to vehicles. According to MEC server's pricing, the vehicles next decide the number of its offloading computation tasks for their cost minization. So, The MEC server and vehicles interact with each other to determine an optimal resource pricing and management for their interest maximization. Zeng et al. [14] analyzed the conflict of interest between MEC servers and vehicles using the Stackelberg game to cooperate MEC servers and vehicles. This approach effectively alleviates the problem of limited computing resources for vehicles, however, it fails to take security issues into account. There is a large amount of data exchange and transaction records in the MEC network. The

entire network is likely to be vulnerable, which even leads to unacceptable errors. The authors in [15,16] introduced an economic computing resource management based on the Stackelberg game model, which also took into account blockchain for network security. However, these works employ proof-of-work (PoW) protocols to achieve consensus in the blockchain. Block mining and synchronization can lead to high energy costs and long latency, making it unsuitable for vehicles with limited computational resources. To address the above challenges, we utilize Stackelberg game to optimal the resource allocation and pricing model in an attempt to maximize each participant's revenue. Moreover, we implement the blockchain by a POS mechanism to guarantee offloading safety, which consumes less energy in comparison with the POW mechanism [17]. Our main contributions are enumerated as follows.

- We introduce MEC to offload the computational tasks of vehicles for alleviating their computational and energy burdens. The security risks are also considered in the computational offloading process by integrating the blockchain network into the MEC network.
- In the blockchain network, the POS mechanism is used instead of the POW mechanism, avoiding the problem that POW will generate huge energy consumption.
- We analyze interactions between participants through the Stackelberg game and prove theoretically to achieve a unique Nash equilibrium. Conflicts of interest of participants are resolved by rationally allocating resource and pricing in the network.

2. Blockchain-Based MEC Network Architecture

In blockchain-based MEC network architecture, both MEC networks and blockchain are built based on computation resources and wireless communication [18]. The combination of blockchain technology into MEC networks can fully utilize their respective advantages. On the one hand, blockchain protects the accuracy, consistency and validity of data in a transparent way [19]. In the MEC network, the data generated by the vehicles and the MEC server during the computational offloading process are encrypted and stored on blocks of the blockchain. It is feasible for privacy protection and security without any third-party control mitigation. Blockchain technology has the practical availability in transaction market across different infrastructures and operators under insecure network conditions. On the other hand, the nodes in the blockchain need to have sufficient computing resources for generating blocks, broadcasting and verifing them through the network. The complete blockchain needs to be stored in all the blockchain nodes for ensuring the accuracy, consistency, and validity of their data. The servers deployed in the MEC network have enough computing and storage resources to meet these computation and caching demands of the blockchain [20,21].

Participants in the Blockchain-Based MEC Network Architecture

In this paper, the blockchain network is used as the backbone of the decentralized computing resource allocation network [22]. Each participant (the MEC server equipped in the base station, the requesting vehicle on the road) maps itself as a node in the blockchain network, as shown in Figure 1. The MEC server deployed in the base station has sufficient computing resources to offload their computing tasks of the nearby requesting vehicles. Meanwhile, the MEC servers are considered consensus nodes equipped with roughly synchronized clocks to perform the consensus process in the blockchain network. All recent transaction data are written into a block and stored, broadcast, and verified by the consensus node. Each block stores a certain number of transaction records generated at fixed intervals and connects to the previous block in the block list for forming an ordered blockchain. Due to the requesting vehicles are constrained by energy and computing resources [6], their computing tasks are offloaded to a nearby MEC server for task execution. In return, the MEC server obtains proper economic compensation from requesting vehicles. All requesting vehicles participate in the blockchain network as light nodes and only perform

actions, such as requesting task offloading, transmitting computation tasks, and receiving computation results, as opposed to consensus nodes mapped by the MEC server.

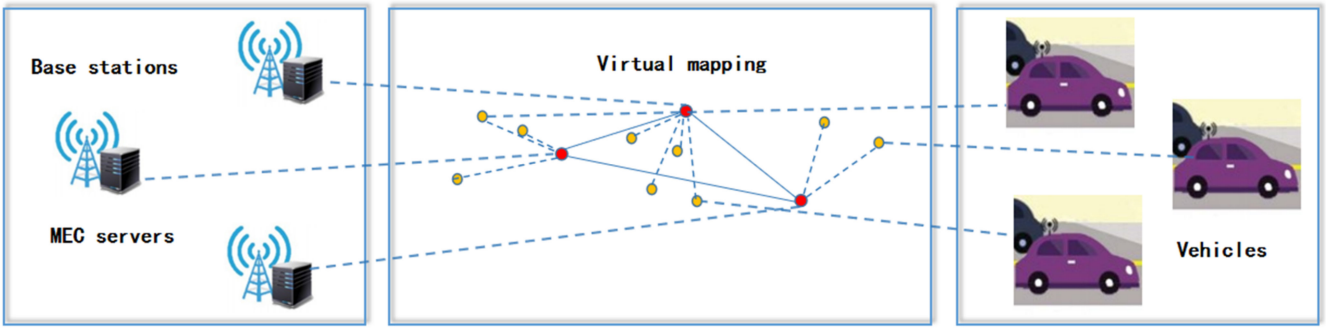

Figure 1. Blockchain-based MEC Network. The requesting vehicles and MEC servers in the MEC network are mapped as different nodes in the blockchain, where the MEC servers are mapped as nodes of performing block generation and validation, and the requesting vehicles are mapped as light nodes of only performing actions.

3. System Model

This section presents the blockchain-based MEC model shown in Figure 1. The notations of frequently used symbols are explained in Table 1.

Table 1. The main notation used in this paper.

Parameter	Meaning
c_j	The MEC server indexed by j in the proposed model
$v_i, i \in N_j$	The requesting vehicle v_i connected to MEC server c_j
$b_{i,j}$	The bandwidth between vehicle v_i and MEC server c_j
$h_{i,j}$	The channel gain vehicle between vehicle v_i and MEC server c_j
σ_{ij}	The noise variances of vehicle v_i to MEC server c_j
w_i	The total computational workload required for the task of vehicle v_i
t_i	The uplink transmission time of the task submitted by vehicle v_i
X_i	The computational workload required per unit of task data
$p_{i,j}$	The transmit power between vehicle v_i and MEC server c_j
s_j	The computation resource pricing of MEC server c_j
P_j	The probability of selecting server c_j as a block producer
k_i, K_j	The energy coefficient of vehicle v_i and MEC server c_j
f_i, F_j	The CPU computing capability of of vehicle v_i and MEC server c_j
$R_{i,j}$	The transmission rate between vehicle v_i and MEC server c_j
R_v, R_c	The block generation reward and block consensus reward
$E_{i,j}$	The total energy consumed by vehicle $v_i, i \in N_j$
E_j	The total energy consumed by MEC server c_j

It is assumed that in this model, there are a total of m MEC servers in the MEC network, and each server is deployed at the appropriate location. The collection of MEC servers is denoted as $\{c_1, c_2, \ldots, c_j, \ldots, c_m\}$. Each MEC server connects different number of requesting vehicles within its communication range. We use $N_j = \{v_1, v_2, \ldots, v_i, \ldots, v_{n_j}\}$ to denote the set of requesting vehicles, connecting to MEC server c_j. Assume that the computing task $A_i : (w_i, T_i, X_i)$ is generated by the requesting vehicle v_i, $i \in N_j$, where w_i is the total workload to finish the task A_i, T_i is the time delay that vehicle v_i can tolerate, and X_i is the computational workload required for per unit of task data.

3.1. Requesting Vehicle Model

3.1.1. Communication Model

Requesting vehicles and MEC server communicate via a wireless channel. Assume that $p_{i,j}$ is the transmit power of requesting vehicle v_i on a subchannel of MEC server c_j. So, the wireless channel rate $R_{i,j}$ is defined as [23].

$$R_{i,j} = b_{i,j} \log_2 \left(1 + \frac{p_{i,j} h_{i,j}}{\sigma_{i,j}^2} \right) \tag{1}$$

where $b_{i,j}$ is the bandwidth of the subchannel between the requesting vehicle v_i and MEC server c_j, $h_{i,j}$ and $\sigma_{i,j}$ represent the channel gain on the subchannel and the additional white Gaussian noise power, respectively [17].

3.1.2. Energy Consumption Model

We here neglect the downlink transmission time of a task since the the data size of task result is small. Assume that the uplink transmission time of task A_i is t_i. It is easily seen that $t_i < T_i$. So, the size of the task A_i is offloaded to MEC server c_j for execution as $R_{i,j} t_i$ [24]. The energy cost of requesting vehicle v_i to complete a computing task A_i mainly consists of local computing energy consumption and uplink transmission energy consumption. Thus, the total energy consumption $E_{i,j}$ is

$$E_{i,j} = E_{i,j}^{local} + E_{i,j}^{trans} = k_i f_i^2 (w - R_{i,j} t_i X_i) + p_{i,j} t_i \tag{2}$$

where k_i is the energy coefficient that is dependent on the chip architecture of vehicle v_i, f_i is the CPU computing capability of the vehicle v_{i}, and $p_{i,j}$ is the transmit power from the vehicle v_i to the MEC server c_j [25]. $E_{i,j}^{local}$ represents the energy consumption generated by the local computation of the requesting vehicle v_i, and $E_{i,j}^{trans}$ represents the energy consumption generated by uplink data transmission from vehicle v_i to MEC server c_j.

3.1.3. Payment Model

The requesting vehicle needs to pay the MEC server for the corresponding computational resources while submitting computational task to the MEC server for execution. It is supposed that s_j is the unit cost of the resource in MEC server c_j, and $p_{i,j}$ can represent the processing power allocated by MEC server c_j to the requesting vehicle v_i. Then, the $M_{i,j}$ is defined to represent the payment paid by requesting vehicle v_i to MEC server c_j [14],

$$M_{i,j} = s_j p_{i,j} \tag{3}$$

3.1.4. Cost Function

The cost function of the requesting vehicle v_i can be is denoted as

$$U_i(p_{i,j}) = \lambda_i E_{i,j} + \varphi_i M_{i,j} \tag{4}$$

The total cost of the requesting vehicle v_i consists of two terms in Equation (4). The first term is the energy consumption of requesting vehicle v_i and the second term is the fee paid to the MEC server c_j. The parameter s_j is the resource pricing of MEC server c_j. The parameters λ_i and φ_i are the energy consumption decision weight and the payment cost decision weight, constrained by $\lambda_i + \varphi_i = 1$ [14]. When the requesting vehicle is concerned with its own energy consumption, the parameter λ_i is set to a larger value and the parameter φ_i is set to a smaller value. Conversely, when the vehicle is concerned with the computation offloading, φ_i will set to a larger value and λ_i is set to a smaller value.

To facilitate the subsequent work, we insert Equations (2) and (3) into Equation (4) and expand them to obtain a more detailed description of the cost function, which is

$$U_i(p_{i,j}) = \lambda_i\left(k_i f_i^2(w_i - R_{i,j}t_i X_i) + p_{i,j}t_i\right) + \varphi_i s_j p_{i,j} \tag{5}$$

3.2. MEC Server Model

3.2.1. Consensus Model

Considering the fact of the POW mechanism that nodes operates an irreversible hash function to derive the solution of the hash value of a particular block by a large number of calculations, it incurs excessive waste of resources in the blockchain network [26]. We thus choose the PoS consensus mechanism with low energy consumption to implement the blockchain network [27]. Without loss of generality, the choice of a block producer (MEC server) for a new block depends on the number of its "shares". If the block is successfully added to the blockchain, the selected MEC server will be rewarded. Conversely, if the block producer has fraudulent transactions or blocking, it will lose future equity and rights to participate in the consensus process. To motivate MEC servers to provide computing services with the requesting vehicles, we calculate the "shares" of each MEC server by the service benefit it received in the most recent period. Thus, the probability of selecting a server as a block producer in the next period is determined by the ratio of its service reward to the total service reward of other servers in the current period, denoted as

$$P_j = \frac{S_j}{\sum_{k=1,k\neq j}^{m} S_k} \tag{6}$$

3.2.2. Block Reward Model

The blockchain nodes involved in block generation and consensus can all obtain certain rewards from the blockchain, which are the rewards R_v for generating blocks and the rewards R_c for verifying blocks, respectively. Since only one blockchain node can be elected as the block generator for each block, we are able to estimate the block generation probability of each blockchain node according to Equation (6). Accordingly, the reward expectation of MEC server c_j as a blockchain node is denoted,

$$P_j R_v + \left(1 - P_j\right) R_c \tag{7}$$

where $\left(1 - P_j\right) R_c$ indicates that nodes fails to elect as the generator, and instead act as the block validator.

3.2.3. Energy Consumption Model

The total energy consumption generated by the MEC server c_j in the process of providing computing services to the requesting vehicles in its communication range can be expressed as

$$E_j = \xi K_j F_j^2 \sum_{i=1}^{n_j} R_{i,j} t_i X_i \tag{8}$$

where ξ is the weight to balance the energy consumption and the service benefit, K_j is the energy coefficient, and F_j is the CPU computing capability of the MEC server c_j. $\sum_{i=1}^{n_j} R_{i,j} t_i X_i$ represents the total workloads from all requesting vehicles.

3.2.4. Utility Function

The utility function of the MEC server c_j can be expressed as

$$U_j(s_j) = \sum_{i=1}^{n_j} s_j p_{i,j} + P_j R_v + \left(1 - P_j\right) R_c - \xi K_j F_j^2 \sum_{i=1}^{n_j} R_{i,j} t_i X_i \tag{9}$$

$\sum_{i=1}^{n_j} s_j p_{i,j}$ is the service benefit of providing computing resources to all of requesting vehicles.

4. Stackelberg Game Analysis

4.1. Description of the Problem

In the system model, a conflict of interest exists between the MEC server and the requesting vehicles. Participants will choose their own best strategies to maximize their interests. The details are described in the following two subsections.

4.1.1. Cost Minimization for Requesting Vehicles

The cost of requesting vehicles is shown in Equation (5). Each requesting vehicle decides an offloading strategy based on the resource pricing of the MEC server to minimize its cost while satisfying its requirements for QoS. So, the optimal problem of a requesting vehicle is described as

$$p_{i,j}^* = arg \min_{p_{i,j}>0} U_i(p_{i,j}) \tag{10}$$

4.1.2. Utility Maximization of MEC Servers

Since the offloading strategies of each requesting vehicle is determined on the basis of the MEC server's resource pricing, the MEC server needs to set a reasonable resource price to increase its revenue and improve its competitiveness in block generation. Correspondingly, the optimal problem of the MEC server is described as

$$s_j^* = arg \max_{s_j>0} U_j(s_j) \tag{11}$$

4.2. Formulation of Two-Stage Stackelberg Game

Stackelberg game divides the players into leader and follower to model behaviors of the two types of players. Both sides make their decisions in stages. The leader first acts in a stage and then the followers should respond to the leader's action in the other stage. Specially, the leader initiates a pricing strategy, on which the followers choose their response strategies and then submit their response strategies to the leader. Such strategy-making processes of both sides repeat until achieving Nash equilibrium.

We here build a two-stage Stackelberg game for describing the interactions of resource allocation and pricing among requesting vehicles and its connected MEC server. Equations (10) and (11) jointly formulate such game $\mathbb{G} = \{N_j, \{p_{i,j}\}_{i \in N_j}, \{U_i(p_{i,j})\}_{i \in N_j}\}$. The MEC server acts as the leader of requesting vehicles to decide the resource pricing. Requesting vehicles act as the followers to determine their resource allocation strategies. The backward induction is used to analyze such a Stackelberg game. While the requesting vehicle requests computing resources from the MEC server, the MEC server first announces its resource pricing. The requesting vehicle then decides the size of the task to be offloaded to MEC's server. This process continues until Nash equilibrium reaches. The optimal strategies of leader and followers accord to the Nash equilibrium point [28,29].

Definition 1. *Nash equilibrium in $\mathbb{G}$: Suppose that the optimal strategies of the requesting vehicle v_i and the MEC server c_j are denoted as $\left\{p_{i,j}^*, s_j^*\right\}$. Let $\boldsymbol{p}_j^* = \left(p_{1,j}^*, \ldots, p_{i,j}^*, \ldots, p_{n_j,j}^*\right)$ represent the optimal strategy vector of requesting vehicles connecting to MEC server c_j. Nash equilibrium implies that neither the leader nor the follower can further increase their gains by unilaterally changing the strategy. Therefore, requesting vehicle and MEC server's strategies always satisfy*

$$U_i\left(p_{i,j}^*, \boldsymbol{p}_{-i,j}^*, s_j^*\right) \leq U_i\left(p_{i,j}, \boldsymbol{p}_{-i,j}^*, s_j^*\right) \tag{12}$$

$$U_j\left(p_{i,j}^*, s_j^*\right) \geq U_j\left(p_{i,j}^*, s_j\right) \tag{13}$$

where $\boldsymbol{p}^*_{-i,\,j} = \left(p^*_{1,\,j}, \ldots, p^*_{i-1,\,j}, p^*_{i+1,\,j}, \ldots, p^*_{n_j,\,j}\right)$ is the optimal strategy vector of requesting vehicles except vehicle v_i.

4.2.1. Vehicle-Level Game Analysis

To efficiently complete the local computing tasks and minimize its energy consumption, the requesting vehicle decides how many computing tasks are offloaded to the MEC server based on its energy reserves and the MEC server's resource pricing. In this section, we explore the optimal strategy for a requesting vehicle v_i. Substituting (1) into (5), the cost function is rewritten as

$$U_i(p_{i,j}) = \lambda_i \left(k_i f_i^2 \left(w_i - b_{i,j} \log_2 \left(1 + \frac{p_{i,j} h_{i,j}}{\sigma_{i,j}^2} \right) t_i X_i \right) + p_{i,j} t_i \right) + \varphi_i s_j p_{i,j} \tag{14}$$

Ultimately, the cost function $U_i(p_{i,j})$ of the requesting vehicle v_i with respect to $p_{i,j}$ can be obtained,

$$U_i(p_{i,j}) = \lambda_i k_i f_i^2 w_i - \lambda_i k_i f_i^2 b_{i,j} \log_2 \left(1 + \frac{p_{i,j} h_{i,j}}{\sigma_{i,j}^2} \right) t_i X_i + \lambda_i p_{i,j} t_i + \varphi_i s_j p_{i,j} \tag{15}$$

Then, we calculate the first derivative of $U_i(p_{i,j})$ with respect to $p_{i,j}$,

$$\frac{\partial U_i(p_{i,j})}{\partial p_{i,j}} = -\frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i h_{i,j}}{\left(\sigma_{i,j}^2 + p_{i,j} h_{i,j} \right) \ln 2} + \lambda_i t_i + \varphi_i s_j \tag{16}$$

Based on (16), the second derivative of $U_i(p_{i,j})$ is obtained,

$$\frac{\partial^2 U_i(p_{i,j})}{\partial p_{i,j}^2} = \frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i h_{i,j}^2}{\left(\sigma_{i,j}^2 + p_{i,j} h_{i,j} \right)^2 \ln 2} \tag{17}$$

It is obviously find that $\frac{\partial^2 U_i(p_{i,j})}{\partial p_{i,j}^2}$ is always positive. Thus the cost function U_i is strictly convex, which implies the uniqueness of optimal solution $p^*_{i,j}$. Let Equation (16) be equal to 0,

$$\frac{\partial U_i(p_{i,j})}{\partial p_{i,j}} = -\frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i h_{i,j}}{\left(\sigma_{i,j}^2 + p_{i,j} h_{i,j} \right) \ln 2} + \lambda_i t_i + \varphi_i s_j = 0 \tag{18}$$

By solving (18), we obtain the optimal strategy of requesting vehicle v_i as

$$p^*_{i,j} = \frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i}{\left(\lambda_i t_i + \varphi_i s_j \right) \ln 2} - \frac{\sigma_{i,j}^2}{h_{i,j}} \tag{19}$$

4.2.2. Server-Level Game Analysis

In order to rationally allocate the computational resources it holds and maximize its service revenue, the MEC server needs to adjust its resource pricing based on the amount of computational resources requested by the current requesting vehicle. In this subsection, we will explore the optimal strategy of the MEC server.

Based on the service benefit $\sum_{i=1}^{n_j} s_j p_{i,j}$ of the MEC server c_j, the probability of successfully selecting c_j as a block generator is rewritten as

$$P_j = \frac{\sum_{i=1}^{n_j} s_j p_{i,j}}{\sum_{k=1, k \neq j}^{m} \sum_{i=1}^{n_k} s_k p_{i,k}} \tag{20}$$

Therefore, the utility function of the MEC server c_j with POS mechanism is denoted as

$$U_j(s_j) = \sum_{i=1}^{n_j} s_j p_{i,j} + \frac{\sum_{i=1}^{n_j} s_j p_{i,j}}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}} R_v + \left(1 - \frac{\sum_{i=1}^{n_j} s_j p_{i,j}}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}}\right) R_c - \xi K_j F_j^2 \sum_{i=1}^{n_j} b_{i,j} \log_2\left(1 + \frac{p_{i,j} h_{i,j}}{\sigma_{i,j}^2}\right) t_i X_i \tag{21}$$

Substituting $p_{i,j}^*$ into Equation (21), the utility function in (21) is rewritten as

$$U_j(s_j) = \left(1 + \frac{(R_v - R_c)}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}}\right) \sum_{i=1}^{n_j} s_j \left(\frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i}{(\lambda_i t_i + \varphi_i s_j)\ln 2} - \frac{\sigma_{i,j}^2}{h_{i,j}}\right) + R_c - \xi K_j F_j^2 \sum_{i=1}^{n_j} b_{i,j} \log_2\left(1 + \frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i h_{i,j}}{(\lambda_i t_i + \varphi_i s_j)\sigma_{i,j}^2 \ln 2}\right) t_i X_i \tag{22}$$

We calculate the first derivative of $U_j(s_j)$ on s_j,

$$\frac{\partial U_j(s_j)}{\partial s_j} = \left(1 + \frac{(R_v - R_c)}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}}\right) \sum_{i=1}^{n_j} \left(\frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i}{(\lambda_i t_i + \varphi_i s_j)\ln 2} - \frac{\sigma_{i,j}^2}{h_{i,j}} - s_j \frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i \varphi_i}{(\lambda_i t_i + \varphi_i s_j)^2 \ln 2}\right) + \xi K_j F_j^2 \sum_{i=1}^{n_j} \frac{b_{i,j} t_i X_i \varphi_i}{(\lambda_i t_i + \varphi_i s_j)\ln 2} \tag{23}$$

Moreover, the second derivative of $U_j(s_j)$ on s_j is calculated as,

$$\frac{\partial^2 U_j(s_j)}{\partial s_j^2} = -\left(1 + \frac{(R_v - R_c)}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}}\right) \sum_{i=1}^{n_j} \left(\frac{2\lambda_i^2 k_i f_i^2 b_{i,j} t_i^2 X_i \varphi_i}{(\lambda_i t_i + \varphi_i s_j)^3 \ln 2}\right) - \xi K_j F_j^2 \sum_{i=1}^{n_j} \frac{b_{i,j} t_i X_i \varphi_i^2}{(\lambda_i t_i + \varphi_i s_j)^2 \ln 2} \tag{24}$$

Based on Equation (24), the second derivative of $U_j(s_j)$ is obviously negative. So, the utility function U_j is strictly concave. Since the concave function has a unique global optimal solution, the equilibrium point of the game is s_j^*. In order to obtain the optimal strategy for the MEC server c_j, it is necessary to find the solution of the following equation,

$$\frac{\partial U_j(s_j)}{\partial s_j} = \left(1 + \frac{(R_v - R_c)}{\sum_{k=1,k\neq j}^{m}\sum_{i=1}^{n_k} s_k p_{i,k}}\right) \sum_{i=1}^{n_j} \left(\frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i}{(\lambda_i t_i + \varphi_i s_j)\ln 2} - \frac{\sigma_{i,j}^2}{h_{i,j}} - s_j \frac{\lambda_i k_i f_i^2 b_{i,j} t_i X_i \varphi_i}{(\lambda_i t_i + \varphi_i s_j)^2 \ln 2}\right) + \xi K_j F_j^2 \sum_{i=1}^{n_k} \frac{b_{i,j} t_i X_i \varphi_i}{(\lambda_i t_i + \varphi_i s_j)\ln 2} = 0 \tag{25}$$

We see that s_j^* and $p_{i,j}^*$ are the optimal strategies of the MEC server and the requesting vehicle, respectively. The Equations (16) and (23) guarantee the unique Nash equilibrium. It is feasible for each requesting vehicle to obtain its optimal strategy according to (19). However, because of the non-linearity of (25), it is difficult to directly obtain the optimal solution s_j^*. Instead, we employ the gradient descent method to iteratively approximate the optimal solution s_j^* [30–33].

5. Simulation Experiments

In this paper, simulation experiments are conducted using python to evaluate our proposed scheme. We begin with the parameter setting and then display the numerical simulation results for illustration.

5.1. Parameter Setting

In this simulation scenario, 10 MEC servers are deployed in the system. Each server covers 60–100 requesting vehicles, and the delay tolerance interval for offloading tasks is 0.7–0.9 ms. The bandwidth of each subchannel is 5 MHz, the noise power of the subchannel is 60 dB, and the channel gain on the subchannel is 54–57 dBm. The resource pricing of the server is subject to 0.4–1.0. The weights λ_i and φ_i are randomly generated, respectively, and their sum is 1. The main parameters we used in the simulation are shown in Table 2.

Table 2. The simulation parameters. m is the number of MEC servers, n_j is the number of requesting vehicles covered by the jth server, $b_{i,j}$, $h_{i,j}$ and σ_{ij} are used to describe the current wireless channel environment, t_i is the time delay tolerated by the vehicle v_i, λ_i and φ_i are the weight parameters used to balance the energy consumption and resource cost of the requesting vehicles, and ξ is the weight parameter for the energy consumption of the MEC servers.

Parameter	Value	Parameter	Value
m	10	t_i	0.7–0.9 ms
n_j	60–100	λ_i	0.5–0.65
$b_{i,j}$	5 MHz	φ_i	0.35–0.5
$h_{i,j}$	54–57 dBm	ξ	0.02
σ_{ij}	60 dB		

5.2. Analysis of Results

We use the gradient descent method to find the optimal strategy for the MEC servers. Figure 2a shows how to find the optimal resource pricing in the algorithm. Each MEC server decides the resource pricing based on its current network environment and the number of requesting vehicles. As the number of iterations increases, the prices set by MEC servers gradually approximate their own optimal strategies. It is seen from Figure 2a that the proposed algorithm converges to the optimal value. It indicates that each MEC server is able to find its optimal resource pricing for maximizing its revenue.

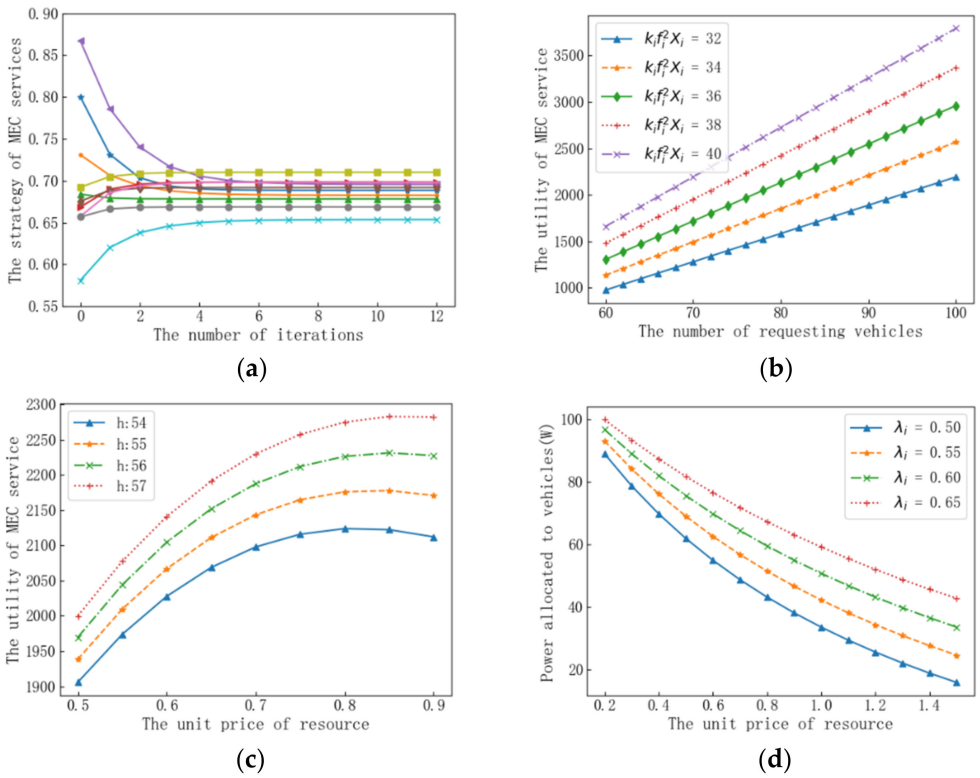

Figure 2. (**a**) is an iterative process of finding the optimal strategy using the gradient descent method for 10 MEC servers, (**b**) is the effect of the number of vehicles and the computational cost of vehicles on the MEC server utility, (**c**) is the impact of resource pricing and channel conditions on MEC server utility and (**d**) is the impact of resource pricing and energy consumption on transmit power.

We next observe the effect of the number of requesting vehicles on the server's utility. Figure 2b shows that the average effect of all servers increases with the increase in requests for vehicles. Besides, with the increasing energy consumption of requesting vehicles, it has more significant influence on the MEC server utility becomes more pronounced.

We are also exploring the effect of the MEC server's resource price on its utility. Figure 2c shows the simulation results. It is found that the utility of the MEC server starts increasing while the resource price improves. However, its benefit then decrease as the resource price increases, exceeding the optimal strategy point of the Stackelberg game.

Lastly, we investigate the effect of the MEC server's resource pricing on the transmit power of the requesting vehicle. The results are shown in Figure 2d. While the resource price gradually increases, the requested transmit power decreases. It indicates that a high resource price incurs a low computing resource allocation. As users attach importance to energy consumption, more computing tasks would be offloaded to the MEC server. Such behavior is aimed to reduce its own energy consumption.

6. Conclusions

To improve the resource utilization and security of the MEC network, this paper combines the MEC network with the blockchain network to propose a task offloading system. Especially, the PoS mechanism in the blockchain network is employed to ensure the data's safety in the MEC network. The offloading interactions between MEC servers and requesting vehicles formulate a two-stage Stackelberg game to find their optimal strategies for revenue maximization. We also prove to achieve Nash equilibrium, which indicates the uniqueness of optimal strategies for MEC servers and requesting vehicles. The performance of our proposed system model is discussed by analyzing the variable factors in simulation experiments. Such system is built to assist MEC servers with their computation resource management. In the future work, we will furthermore explore the dynamic the computation offloading in the blockchain-based MEC network.

Author Contributions: Conceptualization, H.Z.; methodology, H.Z.; software, H.Z.; validation, H.Z. and Y.L.; formal analysis, H.Z.; investigation, H.Z.; resources, H.Z.; data curation, H.Z.; writing—original draft preparation, H.Z.; writing—review and editing, H.Z.; visualization, H.Z.; supervision, Y.L.; project administration, Y.L.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by, National Natural Science Foundation of China under Grant number 61802216, National Key Research and Development Plan Key Special Projects under Grant number 2018YFB2100303, Shandong Province colleges and universities youth innovation technology plan innovation team project under Grant number 2020KJN011, Shandong Provincial Natural Science Foundation under Grant number ZR2020MF060, Program for Innovative Postdoctoral Talents in Shandong Province under Grant number 40618030001.

Data Availability Statement: Data used in this article can be made available by the corresponding authors on reasonable request.

Acknowledgments: Hongli Zhang thanks Zhihao Xu from Qingdao University for his constructive comments about the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, A.; Li, J.; Ahmed, M. SpiderNet: A spiderweb graph neural network for multi-view gait recognition. *Knowl.-Based Syst.* **2020**, *206*, 106273. [CrossRef]
2. Zhao, A.; Dong, J.; Li, J.; Qi, L.; Zhou, H. Associated spatio-temporal capsule network for gait recognition. *IEEE Trans. Multimed.* **2021**, *24*, 846–860. [CrossRef]
3. Lv, Z.; Li, J.; Dong, C.; Wang, Y.; Li, H.; Xu, Z. DeepPTP: A deep pedestrian trajectory prediction model for traffic intersection. *KSII Trans. Internet Inf. Syst.* **2021**, *15*, 2321–2338.
4. Xu, Z.; Li, J.; Lv, Z.; Wang, Y.; Fu, L.; Wang, X. A graph spatial-temporal model for predicting population density of key areas. *Comput. Electr. Eng.* **2021**, *93*, 107235. [CrossRef] [PubMed]
5. Lv, Z.; Li, J.; Li, H.; Xu, Z.; Wang, Y. Blind travel prediction based on obstacle avoidance in indoor scene. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 5536386. [CrossRef]
6. Somov, A.; Giaffreda, R. Powering IoT devices: Technologies and opportunities. *IEEE IoT Newsl.* **2015**, *367*, 368.
7. Lv, Z.; Li, J.; Dong, C.; Xu, Z. DeepSTF: A deep spatial–temporal forecast model of taxi flow. *Comput. J.* **2023**, *66*, 565–580. [CrossRef]

8. Xu, Z.; Li, J.; Lv, Z.; Dong, C.; Fu, L. A classification method for urban functional regions based on the transfer rate of empty cars. *IET Intell. Transp. Syst.* **2022**, *16*, 133–147. [CrossRef]
9. Li, H.; Lv, Z.; Li, J.; Xu, Z.; Yue, W.; Sun, H.; Sheng, Z. Traffic Flow Forecasting in the COVID-19: A Deep Spatial-Temporal Model Based on Discrete Wavelet Transformation. *ACM Trans. Knowl. Discov. Data* **2022**, *17*, 64. [CrossRef]
10. Xu, Z.; Lv, Z.; Li, J.; Sun, H.; Sheng, Z. A Novel Perspective on Travel Demand Prediction Considering Natural Environmental and Socioeconomic Factors. *IEEE Intell. Transp. Syst. Mag.* **2022**, *15*, 136–159. [CrossRef]
11. Cheng, Z.; Rashidi, T.H.; Jian, S.; Maghrebi, M.; Waller, S.T.; Dixit, V. A Spatio-Temporal autocorrelation model for designing a carshare system using historical heterogeneous Data: Policy suggestion. *Transp. Res. Part C Emerg. Technol.* **2022**, *141*, 103758. [CrossRef]
12. Lin, C.; Li, Y.; Ahmed, M.; Song, C. Piece-wise pricing optimization with computation resource constraints for parked vehicle edge computing. *Peer-to-Peer Netw. Appl.* **2023**. [CrossRef]
13. Shah-Mansouri, H.; Wong, V.W.; Huang, J. An incentive framework for mobile data offloading market under price competition. *IEEE Trans. Mob. Comput.* **2017**, *16*, 2983–2999. [CrossRef]
14. Zeng, F.; Chen, Q.; Meng, L.; Wu, J. Volunteer assisted collaborative offloading and resource allocation in vehicular edge computing. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 3247–3257. [CrossRef]
15. Xiong, Z.; Zhang, Y.; Niyato, D.; Wang, P.; Han, Z. When mobile blockchain meets edge computing. *IEEE Commun. Mag.* **2018**, *56*, 33–39. [CrossRef]
16. Xiong, Z.; Feng, S.; Niyato, D.; Wang, P.; Han, Z. Optimal pricing-based edge computing resource management in mobile blockchain. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), IEEE, Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
17. Liu, Y.; Yu, F.R.; Li, X.; Ji, H.; Leung, V.C.M. Decentralized resource allocation for video transcoding and delivery in blockchain-based system with mobile edge computing. *IEEE Trans. Veh. Technol.* **2019**, *68*, 11169–11185. [CrossRef]
18. Goldsmith, A. *Wireless Communications*; Cambridge University Press: Cambridge, UK, 2005.
19. Kotobi, K.; Bilen, S.G. Secure blockchains for dynamic spectrum access: A decentralized database in moving cognitive radio networks enhances security and user access. *IEEE Veh. Technol. Mag.* **2018**, *13*, 32–39. [CrossRef]
20. Tan, Z.; Yu, F.R.; Li, X.; Ji, H.; Leung, V.C.M. Virtual resource allocation for heterogeneous services in full duplex-enabled SCNs with mobile edge computing and caching. *IEEE Trans. Veh. Technol.* **2017**, *67*, 1794–1808. [CrossRef]
21. Liang, C.; He, Y.; Yu, F.R.; Zhao, N. Video rate adaptation and traffic engineering in mobile edge computing and caching-enabled wireless networks. In Proceedings of the 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, Canada, 24–27 September 2017; pp. 1–5.
22. Wang, W.; Niyato, D.; Wang, P.; Leshem, A. Decentralized caching for content delivery based on blockchain: A game theoretic perspective. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
23. Hao, Y.; Chen, M.; Hu, L.; Hossain, M.S.; Ghoneim, A. Energy efficient task caching and offloading for mobile edge computing. *IEEE Access* **2018**, *6*, 11365–11373. [CrossRef]
24. You, C.; Huang, K.; Chae, H.; Kim, B.-H. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2016**, *16*, 1397–1411. [CrossRef]
25. Zhang, W.; Wen, Y.; Guan, K.; Kilper, D.; Luo, H.; Wu, D.O. Energy-optimal mobile cloud computing under stochastic wireless channel. *IEEE Trans. Wirel. Commun.* **2013**, *12*, 4569–4581. [CrossRef]
26. Liang, Y.; Li, Y.; Guo, J.; Li, Y. Resource Competition in Blockchain Networks under Cloud and Device Enabled Participation. *IEEE Access* **2022**, *10*, 11979–11993. [CrossRef]
27. Kiayias, A.; Russell, A.; David, B.; Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2017; pp. 357–388.
28. Li, Y.; Li, J.; Ahmed, M. A three-stage incentive formation for optimally pricing social data offloading. *J. Netw. Comput. Appl.* **2020**, *172*, 102816. [CrossRef]
29. Qiao, Y.; Li, Y.; Li, J. An economic incentive for D2D assisted offloading using stackelberg game. *IEEE Access* **2020**, *8*, 136684–136696. [CrossRef]
30. Zhao, A.; Wang, Y.; Li, J. Transferable Self-Supervised Instance Learning for Sleep Recognition. *IEEE Trans. Multimed.* **2022**. [CrossRef]
31. Xu, Z.; Lv, Z.; Li, J.; Shi, A. A Novel Approach for Predicting Water Demand with Complex Patterns Based on Ensemble Learning. *Water Resour. Manag.* **2022**, *36*, 4293–4312. [CrossRef]
32. Lv, Z.; Li, J.; Dong, C.; Li, H.; Xu, Z. Deep learning in the COVID-19 epidemic: A deep model for urban traffic revitalization index. *Data Knowl. Eng.* **2021**, *135*, 101912. [CrossRef]
33. Cheng, Z.; Jian, S.; Rashidi, T.H.; Maghrebi, M.; Waller, S.T. Integrating household travel survey and social media data to improve the quality of od matrix: A comparative case study. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 2628–2636. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.