Hindawi
Mobile Information Systems
Volume 2021, Article ID 8821583, 15 pages
https://doi.org/10.1155/2021/8821583

Research Article

Double Auction-Based Two-Level Resource Allocation Mechanism for Computation Offloading in Mobile Blockchain Application

Li Li, Yue Li, and Ruotong Li

State Key Laboratory of Networking and Switching Technology, Beijing Laboratory of Advanced Information Networks, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Li Li; lili66@bupt.edu.cn

Received 10 August 2020; Revised 10 December 2020; Accepted 3 January 2021; Published 19 January 2021

Academic Editor: Alessandro Bazzi

It is increasingly popular that platforms integrate various services into mobile applications due to the high usage and convenience of mobile devices, many of which demand high computational capacities and energy, such as cryptocurrency services based on blockchain. However, it is hard for mobile devices to run these services due to the limited storage and computational capacity. In this paper, the problem of computation offloading that requires sufficient computing resources with high utilization in large-scale users and multiprovider MEC system was investigated. A mechanism based on the combinatorial double auction, G-TRAP, is proposed in this paper to solve the above problem. In the mechanism, resources are provided both in the cloud and at the edge of the network. Mobile users compete for these resources to offload computing tasks by the rule that the edge-level resources will be allocated at first while cloud-level resources could be the supplement for the edge level. Given that the proof-of-work (PoW), the core issue of blockchain application, is resource-expensive to implement in mobile devices, we provide resource allocation service to users of blockchain application as experimental subjects. Simulation results show that the proposed mechanism for serving large-scale users in a short execution time outperforms two existing algorithms in terms of social utility and resource utilization. Consequently, our proposed system can effectively solve the intensive computation offloading problem of mobile blockchain applications.

1. Introduction

With the thriving of mobile communication technology and the popularity of mobile devices, the platform services have been expanded on mobile terminals. However, many services are restricted on mobile devices owing to complex computing or too high latency requirements, e.g., autonomous driving services and mobile blockchain services [1]. Recently, blockchain has been continuously combined with many applications in various fields [2], such as e-commerce, the Internet of things [3], and supply chain. Thence, various research institutions attach great importance to this technology and publish academic achievements related to blockchain.

Technically, the essence of blockchain is a distributed ledger based on asymmetric encryption algorithms, and it consists of a combination of blocks linked by hash functions. To confirm and secure the consistency and validity of transactions in the blockchain network, the participants need to complete a computation-intensive task [4], i.e., the proof-of-work (PoW) puzzle [5], which is characterized by the need of consuming significant central processing unit (CPU) resources. According to [6], the core of the PoW process is that the consensus node computes the PoW solution to the cryptographic puzzle, which is formed based on the root hash value of the Markle tree calculated by a newly generated transaction. The Markle tree will be deeper and the calculation will become more complicated when the number of transactions generated by blockchain increases, which sets an extremely high demand for computing capability of the mobile devices. Although the CPU of mobile devices has developed powerfully, it still can hardly meet the demand of

many computing tasks of applications in a short time. Thus, blockchain applications cannot be widely used in mobile devices, which limits mobile blockchain applications' development.

Mobile edge computing (MEC) [7] provides an exemplary method for solving such problems. To support the offloading of computation-intensive tasks, mobile edge computing can provide mobile users with computing and storage resources to solve limitations related to the hardware [8, 9]. In particular, it overcomes the drawbacks of mobile cloud computing, which could cause long latency and backhaul bandwidth congestion [10–12]. Because the computing capacity and appropriate incentive policies are of great importance to users of mobile blockchain applications, it is vital to decide how to offload the computing tasks of the PoW puzzle to the mobile edge computing in the mobile blockchain system.

Given that the resource allocation mechanism involves an incentive policy to allow participants to maintain long-term resource transactions independently, the auction mechanism can be applied in resource allocation as a market-incentive mechanism [13]. Furthermore, considering the characteristics of the PoW consensus mechanism [5], i.e., low latency and rewarding users for broadcasting a new block successfully, users need computing resources to accomplish more computing tasks in a short time to have more chances for obtaining rewards. Furthermore, the edge computing service providers hope to get revenues by selling resources. Consequently, the auction model can provide a fair competition market for resource trading mentioned above.

There have been several works that focus on using auctions or other economic methods to solve the computation offloading problem of the mobile blockchain system. Jiao et al. first investigated resource management and pricing in the mobile blockchain with an auction model [14]. In [14], the resource allocation problem is modeled to maximize the social welfare of edge computing service providers (ESPs) considering the competition between miners and the utility of blockchain network. In [15], Xiong et al. adopted the two-stage Stackelberg game to maximize both the profits of the ESP and the individual utilities of miners. Li et al. designed a long-term auction-based incentive mechanism POEM+ in [16], where the edge servers were encouraged to share their resources to expedite the PoW process execution of mobile devices with an approximation ratio. However, the deficiency of the aforementioned works is that no algorithm or simulation addressed the scenario of large-scale user groups for the mobile blockchain system.

The lack of edge-level resources has also posed a significant obstacle in computation offloading. In response, Bahreini et al. [17] studied resource allocation and pricing in the two-level edge computing system. They proposed a mechanism combining features from both positions and combinatorial auctions to meet the competition and heterogeneous demand of virtual machine (VM) instances for mobile users. The proposed allocation mechanism in [17] provides heterogeneous types of resources for multilocation resource selection based on location information, which contributes to solving intensive computation offloading problems. Unfortunately, this work only considered resource transactions between a single provider and mobile application users, without considering that there will be multiple service providers at the same time in the market.

With the above observations, the promotion of mobile blockchain applications is proposed as a research object in this paper. Correspondingly, a two-level combinatorial double auction mechanism that includes two levels of resources is proposed to solve the computation-intensive offloading problem of mobile blockchain applications mentioned above. In the mechanism, the providers who can provide cloud-edge two-level service are called cloud-edge computing service providers (C-ESPs). The C-ESP can schedule cloud-level resources to supplement the edge-level resources through the cloud computing center. Moreover, this mechanism considered that mobile users usually prefer edge-level computing close to applications. The main contributions of this paper are summarized as follows:

(i) An allocation mechanism based on the combinatorial double auction is proposed for cloud-edge computing resource allocation. In the mechanism, the computing tasks in generating a new block and broadcasting throughout the whole mobile blockchain network to make consensus can be offloaded to C-ESPs.

(ii) Resource allocation and pricing algorithms are proposed to determine winners and final prices, which satisfy the economic attributes of auction truthfulness, individual rationality, budget balance, and computation efficiency.

(iii) The proposed mechanism is simulated in various situations. The simulation results show that the proposed mechanism is effective with large-scale user group participation and outperforms in terms of resource utilization.

2. Related Work

To overcome the resource constraints of mobile devices, the existing works have widely studied resource allocation problem of computation offloading in MEC systems [18] from different fields. In [18], the authors analyzed that the computation offloading is a critical use case regarding the MEC as it can save energy and/or speed up the process of computation from the user perspective. In [19], the authors proposed an adaptive wireless resource allocation strategy of computation offloading service under a three-layered vehicular edge cloud computing framework, which can ensure endurance mileage of electric vehicles when a large number of computation-intensive applications bring colossal energy consumption to the vehicle. To support resource-intensive applications running on the industrial Internet of things (IIOT) devices, the authors in [20] presented a distributed game-theoretic approach to achieving multihop cooperative computing offloading of the IIOT in the edge computing system.

Second, many researchers focused on the resource allocation of multiple service providers and multiple user scenarios by applying the double auction model. Moreover, the double auction model is an economic model widely used in resource allocation that encourages buyers and sellers to bid simultaneously to reach a deal by the incentive mechanism. And during the auction, buyers can bid on combined goods, and buyers and sellers choose each other based on bids' details. In [21], Singhal et al. proposed a combinatorial double auction model for resource allocation. Buyers bid for a package of resources and pay for it one time, which can improve resource utilization and resource efficiency allocation. In the [22], the authors proposed a breakeven-based double auction (BDA) and a dynamic pricing-based double auction (DPDA) to determine the matched pairs between IIOT mobile devices (MDs) and edge servers, as well as the pricing mechanisms restricted by the actual situation for higher system efficiency. These two algorithms mainly modeled the two-side interaction between MEC servers and IIOT MDs. They described a double auction framework to address interaction and maximized the system efficiency where IIOP MDs request computing services with claimed bids and servers sell service with a reported price. In [23], the authors further adopted the auction model and proposed a resource allocation mechanism based on combined double auction to allocate VM instances of ESPs to miners by considering group buying. Although this article considers multiservice provider and multiuser scenarios, it lacks the expansion of future intensive task offloading.

Furthermore, a few recent computing offloading studies have brought edge and cloud computing together to support computation-intensive and latency-critical applications. The architecture of MEC can be further extended to serve users in different scenarios. The authors in [24] modeled the data-intensive service as a W-DAG (weight directed acyclic graph) to investigate the task placement problem for supporting data-intensive services with guaranteed QoS, i.e., shorter service delay in a cloud-edge system according to the business logic analysis. In [25], the authors derived a closed-form optimal task splitting strategy as a function of the normalized backhaul communication capacity and the normalized cloud computation capacity under the collaboration between cloud computing and edge computing. Fu et al. proposed an economical and flexible framework for IIOT by integrating the function of data preprocessing, storage, and retrieval based on both fog computing and cloud computing [26]. In [27], Chen et al. proposed a multiuser multitask computation offloading framework for a green mobile edge cloud computing system where the dynamics of energy arrivals at the mobile edge cloud and task arrivals at different mobile devices are jointly considered.

With the investigation of these research studies on multiposition resource allocation and the computation offloading in mobile blockchain, a cloud-edge two-level double auction mechanism is proposed for intensive computation offloading scenarios in mobile blockchain applications. The rest of the paper is organized as follows. The system model and problem formulation are described in Section 3. The mechanism is presented in Section 4. Section 5 introduces the simulation setup as well as discusses the simulation results. Finally, the conclusion is given in Section 6.

3. System Model and Problem Formulation

The considered system provides K types of VM instances from cloud-edge two levels. In the system, users and C-ESPs submit bids to a central auctioneer separately to buy or sell VM instances. When the edge-level computing server cannot satisfy large-scale offloading requests from mobile users, resources are scheduled and supplied for edge computing nodes from cloud-level servers to form a cloud-edge two-level auction market. Moreover, a total revenue optimization problem of the system is formulated by jointly considering the profit of users and C-ESPs. Compared to single-level edge computation offloading, this model meets more needs of multiple users and improves resource utilization.

3.1. System Model. This auction system consists of mobile blockchain networks, computing service providers, and the auctioneer. In our mobile blockchain network, the mobile devices continuously run the consensus protocol to confirm and secure distributed data or information transmission in the background. Therefore, mobile devices need to solve the PoW problem that involves calculating a nonce output value that satisfies a given condition. However, it is hard for a mobile user to continuously run such challenging programs that require a large volume of CPU computing resources. It is considered that mobile users could offload the task of solving the PoW problem to nearby cloud-edge computing servers deployed by C-ESPs to get over the computing limitation problem of mobile devices. In particular, C-ESPs announce to the auctioneer services for the supply of VM instances and bid information. The mobile users submit their resource demands and bid information to the auctioneer. After receiving the demands and bids, the auctioneer selects winners including users and C-ESPs according to the allocation algorithm and matching mobile users and C-ESPs as the allocation matrix $X = \{x_{ij}\}$, the detail of which will be presented in Section 3.2.

As shown in Figure 1, the cloud-edge two-level auction system includes C-ESPs, mobile blockchain application users, and auctioneers. C-ESPs sell VM instances to users, and users purchase the VM instance for completing the mathematical puzzles of PoW in blockchain applications. Assume that VM instances can be differentiated by diverse computing resources, such as memory, CPU capacity, and storage. It is considered that each C-ESP provides multiple VM instances at two levels, edge ($l = 1$) and cloud ($l = 2$), where l denotes the location of VM instances. Each user demands a bundle of diverse sorts of instances to complete computing tasks.

3.1.1. Defining Bid Information of Users. C-ESPs and users submit bid information to the auctioneer, and the bid of user i is determined by a 2-tuple $(\vec{d}_i, v_i^{\text{user}})$. In the tuple, $\vec{d}_i$ is the

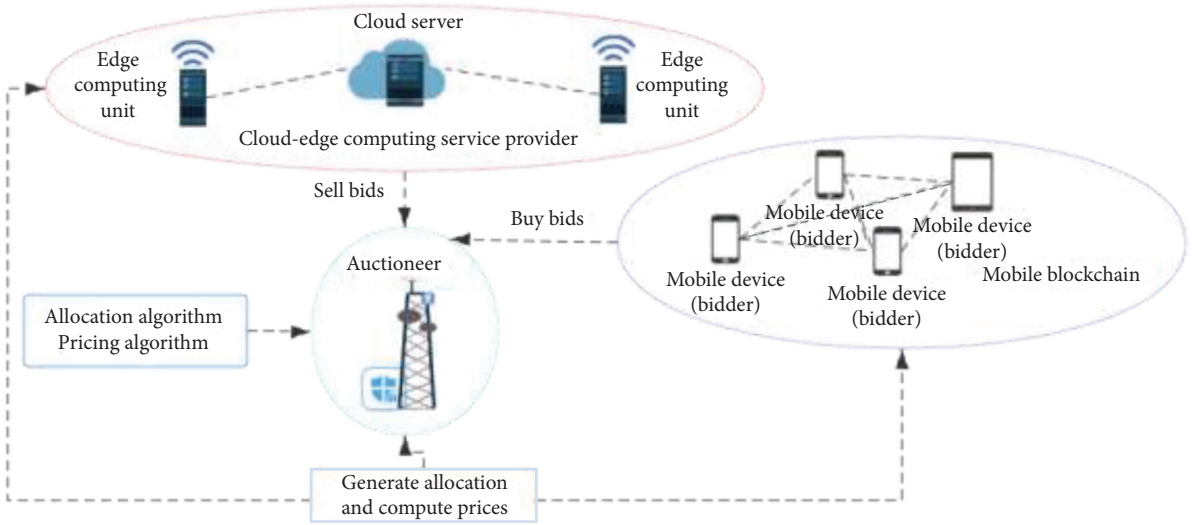


Figure 1: Cloud-edge two-level auction system for mobile blockchain.

demand vector, which is defined as $\vec{d_i} = (d_i^1, d_i^2, \ldots, d_i^k)$. d_i^k denotes the demand of user i for the instance k, and v_i^{user} signifies the reward valuation obtained by users after solving the PoW problem and broadcasting the result in the whole blockchain network.

3.1.2. Calculating Reward Valuation as the Bid of Users. In the mobile blockchain network, users compete to be the first to solve PoW problem with correct nonce value to receive rewards that are considered the value, v_i^{user}, offered by users in this auction system, thence calculating valuation for rewards of all users at first. Assume that the computing power of each VM instance is 1 and each user can successfully purchase required resources. With the allocation x_{ij} and demand d_i^k, the proportion of the computing capacity of the user i in the whole blockchain network can be represented as follows:

$$\gamma_i = \frac{x_{ij}\sum_{i=1}^{N} d_i^k}{\sum_{i=1}^{N}\sum_{j=1}^{M}\left(x_{ij}\sum_{k=1}^{K} d_i^k\right)}, \quad \gamma_i > 0 \text{ and } \sum_{i=1}^{N}\gamma_i = 1. \tag{1}$$

According to the reference in [14], the generation of new blocks follows a Poisson distribution process with a constant rate $1/\lambda$ throughout the whole blockchain network. Before the tournament, users collect unconfirmed transactions into their blocks. The first mobile user to successfully make the block achieve consensus can get the reward consisted of a fixed reward R and a flexible fee r. The flexible reward is proportionate to the number of transactions s included in the block, and the proportionality coefficient is denoted as r. Thus, the potential rewards of user i can be expressed as follows:

$$v_i^{\text{user}} = (R + r \cdot s)P(\gamma_i, s), \tag{2}$$

where $P(\gamma_i, s)$ represents the probability where the user i is the first to work out the computing task and generate a new block. However, the user will not be rewarded if the new block cannot reach consensus on the entire network in time, and such blocks are called orphaned blocks [14]. The broadcasting time τ is proportional to the transaction volume contained in a new block, and the proportionality coefficient can be denoted as ξ; then, $\tau = \xi \cdot s$. Moreover, blocks with large transaction volumes require more propagation time, which may lead to high latency and fail to reach consensus, so blocks with a larger transaction size are more likely to become orphaned blocks. The time of generating a new block obeys Poisson distribution, so the orphaning probability can be approximated as follows:

$$P_{\text{orphan}(\tau)} = 1 - e^{-(1/\lambda)\xi \cdot s}. \tag{3}$$

The probability that a user creates a new block and rewarded can be described as follows:

$$P(\gamma_i, s) = \gamma_i\left(1 - P_{\text{orphan}(\tau)}\right). \tag{4}$$

The potential reward can be rewritten as follows:

$$v_i^{\text{user}} = (R + r \cdot s) \cdot \gamma_i \cdot e^{-(1/\lambda)\xi \cdot s}. \tag{5}$$

3.1.3. Defining Bid Information of C-ESPs. The bid of C-ESP j is specified as a 3-tuple $(\vec{q_{lj}}, \vec{p_j}, \pi_j)$, where $\vec{q_j} = (q_{lj}^1, q_{lj}^2, \ldots, q_{lj}^k)$ and $\vec{p_j} = (p_j^1, p_j^2, \ldots, p_j^k)$. Moreover, q_{lj}^k indicates the number of instances provided by C-ESP j from the lth level, p_j^k denotes the ideal price of instance k determined by C-ESP j, and π_j is the basic energy consumption unit for using VM instances provided by C-ESP j. Besides, K sorts of VM instances are distinguished by computing resource configuration in this model, and the weight of each kind of VM instance is defined as ω_k. The energy consumption function of VM instances is defined as $\cos t(\omega_k) = \omega_k \cdot \pi_j$ based on ω_k, which means that the cost of using computation resource will be higher with the higher resource configuration of the VM instance.

3.1.4. Calculating Bids of C-ESPs with Communication Consumption. The distribution of C-ESPs and users is shown in Figure 2, where the communication transmission between C-ESPs and users comprises the path loss, which is assumed to include in the bid of C-ESPs. Specifically, h' is defined as the threshold of the distance between users and C-ESPs. When the distance between C-ESPs and users exceeds h', the bid will increase in proportion based upon the excess. For C-ESP j, the price of instance k charged for user i can be expressed by

$$p_{ij}^k = \begin{cases} p_j^k, & h_{ij} \le h', \\ p_j^k \dfrac{h_{ij}}{h'}, & h_{ij} > h', \end{cases} \tag{6}$$

where h_{ij} denotes the distance between user i and C-ESP j, and we calculate the average prices of all users. Therefore, the bid of instance k determined by C-ESP j is formulated as

$$\overline{p_j^k} = \frac{\sum_{i=1}^{n} p_{ij}^k}{n}. \tag{7}$$

3.2. Problem Formulation. The previous section has described information of C-ESPs and users as the input to the system. This section mainly defines the output of the system, i.e., the results of resource allocation and pricing, and optimizes this allocation problem as much as possible.

The mechanism proposed (detailed in Section 4) in this work is a combinatorial double auction mechanism with resource location information. The rules are as follows: (i) each user can purchase a bundle of VM instances from no more than one C-ESP; (ii) selling VM instances of cloud level will cost more than the edge level due to the consumption caused by the resource scheduling; (iii) moreover, the bids of users and C-ESPs should be processed by the auctioneer separately. Assume that there are M C-ESPs and N users, and their matched results can be denoted as a $M \times N$ matrix X, whose elements can be denoted as x_{ij}. The value of x_{ij} can be indicated as follows:

$$x_{ij} = \begin{cases} 1, & \text{if seller } j \text{ serves buyer } i, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

According to auction rule (i), the first restriction can be formulated as follows:

$$\sum_{j=1}^{M} x_{ij} < 1, \quad \forall 1 \le i \le N. \tag{9}$$

In addition, the supply of each C-ESP is limited. When allocating VM instances to users, the auctioneer will make sure that the remaining resources of the current C-ESP meet the requirements of the current user. Namely, the total demands of matched users cannot exceed the capacity of C-ESP. The capacity constraint of C-ESPs can be formulated as follows:

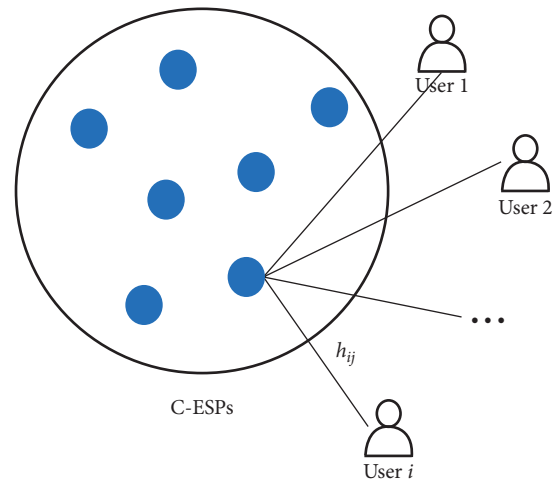


Figure 2: C-ESP distribution structure with users.

$$\sum_{i=1}^{N} x_{ij} d_i^k \le \sum_{l=1}^{2} q_{lj}^k, \quad 1 \le j \le M, 1 \le k \le K. \tag{10}$$

We define the winner set W containing the user who can successfully match C-ESP based on above two constraints.

Along with the allocation, the auctioneer also calculates the payment of users and C-ESPs. The final charge of user i is denoted as p_i that user i actually pays to the provider. p_j is employed to denote the payment of C-ESP j as the revenue obtained from matched users for serving. U_j and U_i represent the utility of C-ESPs and users, respectively, and the utility of the user i is defined as follows:

$$U_i = \sum_{j=1}^{M} x_{ij} \left(v_i^{\text{user}} - p_i\right), \tag{11}$$

and the utility of the C-ESP j is defined as follows:

$$U_j = p_j - \left(v_j^{\text{C-ESP}} - c_j\right). \tag{12}$$

For C-ESP j, $v_j^{\text{C-ESP}}$ is the total offer of instances supplied for matched users. c_j denotes the cost incurred on the process that C-ESP j provides service to users. As an edge computing service provider, the costs generated by the edge-level are within its bid considerations while the excess consumption from the cloud level will affect the C-ESP revenue. Therefore, we assume that the additional resource expenses will only be brought by operating cloud-level VM instances. y_i^k represents the number of the instance k purchased by user i from cloud level of C-ESP j. The utility of C-ESP can be rewritten as follows:

$$U_j = p_j - \sum_{i=1}^{N} x_{ij} \left(\sum_{k=1}^{K} d_i^k \cdot p_j^k - \sum_{k=1}^{K} p_j^k \cdot \cos t\left(\omega_k\right) \cdot y_i^k \right). \tag{13}$$

In this paper, we attach importance to economic efficiency and aim to maximize the total utility of users and C-ESPs. Therefore, the allocation of VM instances problem in the proposed mechanism can be formulated as follows:

$$\max\left(\sum_{i=1}^{N} U_i + \sum_{j=1}^{M} U_j\right),$$
$$\text{s.t.} \sum_{j=1}^{M} x_{ij} \le 1, \quad \forall 1 \le i \le N, \tag{14}$$
$$\text{s.t.} \sum_{i=1}^{i=1} x_{ij} d_i^k \le \sum_{l=1}^{2} q_{lj}^k, \quad 1 \le j \le M, 1 \le k \le K,$$

where two constraints mean that the user can match with no more than one C-ESP and the capacity of C-ESPs is limited. This resource allocation problem can be proved as the NP-hard problem, which cannot reach the optimal in a limited time. According to [13], it can be simplified to the winner determination problem. Table 1 shows the notation used in this paper.

Table 1: Notation.

Notation	Description
N	Number of C-ESPs
M	Number of C-ESPs
$\mathbf{v_i}$	Valuation of user i
$\mathbf{K}$	Types of VM instances
$\mathbf{d_i^k}$	Demand of user i for the instance k
$\mathbf{q_{lj}^k}$	Quantity of the k^{th} instance C-ESP j provides
$\mathbf{p_j^k}$	Price of instance k determined by C-ESP j
π_j	Basic cost for C-ESP j
ω_k	Weight of instance k
$\mathbf{x_{ij}}$	Relationship matrix of users and C-ESPs
$\mathbf{y_i^k}$	Number of kth instances from cloud level
$\mathbf{U_i}$	Utility of user i
$\mathbf{U_j}$	Utility of C-ESP j
$\mathbf{c_j}$	The extra cost of C-ESP j
$\mathbf{v_j^{C-ESP}}$	Offering by C-ESP j for allocated instances
$\mathbf{p_i}$	Payment of user i
$\mathbf{p_j}$	Payment of C-ESP j
α_l	Weight of VM instance for level l

4. The Resource Auction Mechanism: Greedy Two-Level Resource Allocation and Pricing (G-TRAP)

In this section, a corresponding auction mechanism is proposed to solve the VM instance allocation and pricing problem. The proposed mechanism, G-TRAP, can be divided into two parts as follows. In the first part, a greedy mechanism is designed to complete VM instances allocation since it is unlikely to find a polynomial-time optimal solution of the allocation problem mentioned in Section 3. Moreover, this allocation mechanism trades resource in the two-level resource system under the consideration of combining location information and weights of VM instances, which can improve resource utilization. The second part is pricing, in which an improved Vickrey–Clarke–Groves (VCG) mechanism [28] is applied to calculate the payments of participants to ensure the truthfulness of the auction.

4.1. Allocation Algorithm. The first algorithm contained in G-TRAP is the allocation algorithm. Given bids of C-ESPs and users, our objective is to maximize the total utility of participants during allocation. The resource allocation is an NP-hard problem, so the greedy algorithm is customized to solve the problem in this work.

4.1.1. Calculating Bid Density and Ranking. In the first stage, the bids of C-ESPs and users are sorted based on bid densities. Considering that users and C-ESPs prefer to trade in edge-level resources rather than cloud-level resources, we define the size of resources owed by C-ESP j, s_j, related to the type and the location of VM instances. The bid density of the C-ESP can be calculated as follows:

$$bd_j = \frac{v_j}{\sqrt{s_j}},$$
$$s_j = \sum_{l=1}^{2} \sum_{k=1}^{K} \alpha_l \cdot \omega_k \cdot q_{lj}^k. \tag{15}$$

Similarly, the user's bid density can be expressed as follows:

$$bd_i = \frac{v_i}{\sqrt{s_i}},$$
$$s_i = \sum_{k=1}^{K} \omega_k \cdot d_i^k, \tag{16}$$

where α_l is defined as the preference factor for level l and set to distinguish the weight of resources between cloud level and edge level and v_i/v_j is the valuation of bidders. Then, C-ESPs are sorted in ascending order, and users are sorted in descending order based on the bid density.

4.1.2. Allocating in Order. In the second stage, users are assigned to the C-ESPs in sorted order. When a user matches a C-ESP, the C-ESP will firstly check whether there are enough resources at the edge level or not. If (1) the bid density of user i is not less than C-ESP j's and (2) the number of K sorts of instances required by user i does not exceed the remaining resource at the edge level of C-ESP j, C-ESP j and user i will match each other. Suppose only condition (1) is satisfied and the remaining of edge level cannot meet the demands of user i. In this case, C-ESP j will assign cloud-level instances as complements to user i according to the constraints. Then, C-ESP j updates the capacity if sufficient resources are available to the current user. Besides, we denote y_i^k as the number of k^{th} VM instances purchased by user i from the cloud level.

It is explained here that each user can only purchase all needed VM instances from one C-ESP. The user i will continue to match with the next C-ESP ranked behind C-ESP j if the two fail to match. The element x_{ij} of matrix X is marked after matching successfully, and user i is added to the winner set W. The specific details of the allocation are shown in Algorithm 1.

4.2. Pricing Schema and Calculating Total Utility. After matching C-ESPs and users through the allocation algorithm, the payment of users and the revenue of C-ESPs will be calculated separately. We employ the improved VCG price mechanism to pricing. VCG auction always charges the winner the highest bid of losers [28]. In this paper, we define the VCG price vcg_p of a user as the product of the size of resources obtained by user i and the biggest bid density among losers:

$$\text{vcg}_{pi} = \sqrt{s_i} \cdot bd_{\max\_\text{loser}}. \tag{17}$$

To make sure individual personality and maximize the effectiveness of C-ESP, the final charge price of user i is the maximum value of VCG price, the offer of matched C-ESP, and the compromise price. The compromise price is defined as the average value of the next lower bid density of C-ESP and VCG price. The final payment of users can be expressed as follows:

$$p_i = \max\left\{bd_{\max\_\text{loser}}, bd_j, \frac{bd_{j+1} + \text{vcg}_{pi}}{2}\right\} \cdot \sqrt{s_i}. \tag{18}$$

After calculating the charge of users, the payment of C-ESPs can be figured out as follows:

$$p_j = x_{ij}\left(\sum_{i=1}^{N} p_i - c_j\right). \tag{19}$$

The total utility includes the utility of users and C-ESPs. The details of pricing are shown in Algorithm 2.

4.3. Properties of G-TRAP. In the following, it is shown that G-TRAP is actually an incentive mechanism to solve the distribution problem defined above and it satisfies the economic properties include truthfulness, individual rationality, budget balance, and computing efficiency [29].

Definition 1. (truthfulness). An auction is truthful if any participant's utility is maximized for the actual valuation of bidders. Four properties, including monotonicity, criticalness, exactness, and participation, form the sufficiency for a mechanism with truthfulness.

Theorem 1. *Algorithm G-TRAP is truthful.*

Proof 1. From the allocation algorithm, it can be seen that users are first sorted in descending order according to bid density and join in allocation. Therefore, users can improve their ranking by raising the bids or reducing the configuration of purchased resources to make allocation more likely. Accordingly, the winner decision algorithm of our mechanism is monotonous. Secondly, the VCG price is calculated by multiplying the size of resources and the maximum bid density among losers who would win if the winner does not participate in the auction. Obviously, our mechanism is of exactness and participation. Therefore, our mechanism is truthful. □

Definition 2. (individual rationality). Individual rationality means that each participant's utility is non-negative, i.e., $U_i > 0, U_j > 0, \quad \text{for } \forall i, j \in W$.

Theorem 2. *Algorithm G-TRAP is individually rational.*

Proof 2. Charges of users are determined by the largest among VCG price, the offer of matched C-ESP, and the average value of the next lower bid density of C-ESP and VCG price. VCG price is the product of $\sqrt{s_i}$ and $v_{\max\_\text{loser}}/\sqrt{s_{\max\_\text{loser}}}$ which is the highest bid density of losers. Thus, VCG price is generally lower than the user's valuation. As shown in the 6th to the 12th line of pricing algorithm, the value of p_i is certainly less than user's valuation, which means that user utility must be non-negative. According to the 5th line of allocation algorithm and the 5th and 8th lines of pricing algorithm, it can be concluded that the return of C-ESP must be greater than its valuation, and individual rationality of C-ESP is also proved. □

Definition 3. (budget balance). Budget balance means that the final payment of users is not less than the sum of C-ESP's consumption.

Theorem 3. *Algorithm G-TRAP is budget balanced.*

Proof 3. The payments of users are paid to C-ESP, and all of the actual income of C-ESP comes from users. Therefore, there is no auctioneer charged, which means the algorithm is budget balanced. □

Definition 4. (computing efficiency). If an algorithm can terminate in polynomial time, it can be considered as computing efficient.

Theorem 4. *Algorithm G-TRAP is computing efficient.*

Proof 4. For G-TRAP, sorting takes $O(N \log N)$ time, and allocation takes $O(NMK)$ time. If the number of users far exceeds the number of C-ESPs, the maximum time complexity of allocation algorithm is $O(NK)$ and $O(MK)$, otherwise. In the algorithm of pricing and calculating total utility, pricing takes $O(N)$ time and calculating total utility takes $O(NM)$ time. As a result, the time complexity of G-TRAP is $O(NMK + N \log N)$. □

5. Simulation and Performance Analysis

5.1. Numerical Simulation Setup. In previous research studies on mobile blockchain application computation offloading [14–16, 21, 23], the number of users tested in the simulations did not exceed 900. There were no tests for a large-scale user group. To build an intensive computation offloading scenario, we expend the number of users gradually increases from 1 to 5000 during the simulation process while keeping the number of C-ESP as 200 and the number of K types of VM instances unchanged. Moreover, the effect of the cloud-edge capacity ratio of the algorithm is also tested based on 5000 users. Note that the simulation results

```
(i) Input: bid information of C-ESPs M and users N
(ii) Calculate bid density of user i ⟶ mb[i] and calculate bid density of C – ESP j ⟶ eb[j]
(iii) Sort users in nonincreasing order of their mb[i] and sort C-ESPs in nondecreasing order of their eb[i]
(iv) X ⟵ 0, i ⟵, W ⟵ 0, l ⟵ 0, y ⟵0
(v) for i = 1, . . . , n do
(vi) l ⟵1, j ⟵ 0
(vii)   while (j ≤ m) and (f[i] = 0) do
(viii)     if eb[j] ≤ mb[i] then
(ix)       access ⟵ true
(x)        for k = 1, . . . , K do
(xi)         if (q[[l − 1] jk] + q[ljk]) < d[ik] then
(xii)          access ⟵ false
(xiii)         break
(xiv)      if access then
(xv)         for k = 1, . . . , K do
(xvi)        if l = 2 then
(xvii)         if d[ik] ≥ q[1jk] then
(xviii)          d[ik] ⟵ d[ik] − q[1jk]
(xix)            q[1jk] ⟵0
(xx)             q[2jk] ⟵q[2jk] − d[ik]
(xi)             y[ik] ⟵ y[ik] + d[ik]
(xii)          else
(xiii)           q[1jk] ⟵q[1jk] − d[ik]
(xiv)        else
(xv)           q[ljk] ⟵ q[ljk] − d[ik]
(xvi)        x[ij] ⟵1
(xvii)       W[i] ⟵user[i]
(xviii)      f[i] ⟵ 1
(xix)      else
(xx)         if l = 1 then
(xxi)          l ⟵l + 1
(xxii)       else
(xxiii)        j ⟵ j + 1
(xxiv)         continue
(xxv)        end if
(xxvi)     end if
(xxvii)    end if
(xxviii)  end while
(xxxi) end for
(xxx) Output: set of winners W, relation matrix X
```

Algorithm 1: G-TRAP: resource allocation algorithm.

obtained in this section are based on an average over multiple iterations.

Two comparing algorithms proposed in existing work [23] are added to evaluate the performance difference between G-TRAP and edge-only computing systems.

STGA: this scheme is a combinatorial double auction [23] called step greedy algorithm. Computing tasks will match edge computing providers (only supply edge computing resources) according to the bid size. After matching, the utility in this scheme calculated follows the VCG mechanism.

SMGA: compared with the STGA, the only difference is that the smooth greedy algorithm [23] depends on a curve discount function to solve the allocation problem between a group of users and edge computing providers (only supply edge computing resources). The evaluation results show that it performs better than STGA.

In order to test the three algorithms legally, we use the same input file. All data in the figures are obtained by simulation on MATLAB R2019b.

5.1.1. Calculating Rewards of Users in Mobile Blockchain Network. Firstly, we calculate the rewards of the PoW consensus algorithm, which depends on whether the mobile user broadcasts a block successfully in the mobile blockchain as the bid of each user. According to Section 3.1.2, the reward of successfully adding to the chain is divided into two parts, including fixed bonus R and variable $r \cdot s$. In the simulation, we set $R = 20$ and $r = 0.04$ and assume that the size of block s follows the uniform distribution $U(0, 1000)$. Hence, each

(i) for $i = 1, \ldots, n$ do
(ii) vcg_$p \leftarrow$ find max$\{mb[i] \not\subset W\}$
(iii) for $i = 1, \ldots, n$ do
(iv) if user$[i]$ in W then
(v) if vcg$_p \leq mb[i]$ then
(vi) $pi \leftarrow \max\{\text{vcg\_}p, eb[j], (eb[j+1] + \text{vcg\_}p)/2\}^* \sqrt{(s[i])}$
(vii) else
(viii) if $((eb[j+1] + \text{vcg}_{pi})/2) \leq mb[i]$ then
(ix) $pi \leftarrow \max\{eb[j], (eb[j+1] + \text{vcg\_}pi)/2\}^* \sqrt{(s[i])}$
(x) else
(xi) $pi \leftarrow eb[j]^* \sqrt{(s[i])}$
(xii) end if
(xiii) end if
(xiv) end if
(xv) $u[i] \leftarrow vi - \text{pi}$
(xvi) $u[j] \leftarrow u[j] + pi - (\text{v[j]} - \cos \text{t[j]})$
(xviii) end for
(xix) Calculate sum of $u[i]$ and sum of $u[j]$
(xx) **Output:** payment of users, total utility

Algorithm 2: G-TRAP: pricing and total utility algorithm.

user who successfully adds a new block in the public chain will gain $20 + 0.04 \cdot s$.

5.1.2. Calculating the Bids of C-ESPs Based on the Distribution of the Network. The network model is assumed to be a topological graph, where the locations of C-ESPs and users are generated randomly. According to the nodes' distribution, we assume that the distance between the user i and C-ESP j, h_{ij}, is randomly generated in the range of [40, 60], and the threshold h' is fixed to 50. The bid of C-ESPs that includes communication consumption is averagely estimated based on the ratio of h_{ij} and h'.

5.1.3. Setting Related Parameters of Auction. Assume that there are four types of VM instances demanded by users and the amount of VM instances required is set to obey the uniform distribution of $U(1, 5)$. Based on Amazon's Elastic Compute Cloud (EC2) M3 series instances, we consider that these four types of VM instances provided by C-ESPs are defined as medium, large, xlarge, and 2xlarge, each equipped with a combination of three types of resources, including vCPU, memory, and storage. The resources are sold by cloud level and edge level of C-ESP. Then, we assume that the number of VM instances for each C-ESP is subject to a uniformly distributed dataset of $U(30, 35)$.

The unit price of C-ESP is randomly generated during the range of $[0.1 \cdot W_k - 0.05, 0.1 \cdot W_k + 0.05]$ in which W_k is the weight corresponding to each VM instance of C-ESP, and $0.1 \cdot W_k$ denotes the basic unit price of k^{th} instance. We assign the weight α_l to differentiate instance in edge-level (α_2) and cloud-level (α_1) and set four combinations of α_l which are $(\alpha_1 = 0.2, \alpha_2 = 0.8)$, $(\alpha_1 = 0.4, \alpha_2 = 0.6)$, $(\alpha_1 = 0.7, \alpha_2 = 0.3)$, and $(\alpha_1 = 0.9, \alpha_2 = 0.1)$ to construct sets of VM instances. The basic energy consumption unit π_j is set with three different value ranges ($[0, 0.5]$, $[0.5, 1]$, $[1, 1.5]$) and C-ESPs will pay different resource costs for diverse types of VM instances. The distribution of parameters of VM instances generated in simulations is shown in Table 2.

5.1.4. Setting Cloud-Edge Ratio. We define a parameter, called cloud-edge capacity ratio, C^r, which is the ratio of the capacity for each VM instance at the cloud and the capacity at the edge level. The values of C^r are expressed as $C^r = 1, (6/4), (7/3), (8/2), (9/1)$.

5.2. Analysis of Results

5.2.1. Impact of Reward Parameters for Successful Broadcasting of Users. Firstly, we consider the impact of fixed bonus R and the transaction fee r for broadcasting a block on the mechanism. It is observed from Figure 3 that if the blockchain network raises R and r, the total utility and the percentage of winners will be increased nearly in proportion. The reason is that users can obtain more rewards by completing computing tasks as expected and the bids increase correspondingly.

5.2.2. Impact of the Number of Mobile Users **N**. We next investigate the effects of user amount on the performance of G-TRAP with different sets of VM instances of C-ESPs. For this purpose, we keep the number of resources provided to users unchanged and increase the number of users continually. And we also compare the performance of G-TRAP against that of STGA and SMGA in [23].

The total utility of G-TRAP is shown in Figure 4(a), which shows that as the number of users increases, the total utility also increases. Moreover, Figure 4(a) shows that the greater relative weight of VM instances of edge level brings more total utility. It is due to the fact that users and C-ESP

Table 2: Types of VM instance used in the simulations.

Type	Weight	Price range
Medium	1	[0.1 W_1−0.05, 0.1 W_1]
Large	3	[0.1 W_2−0.05, 0.1 W_2]
Xlarge	5	[0.1 W_3−0.05, 0.1 W_3]
2Xlarge	7	[0.1 W_4−0.05, 0.1 W_4]

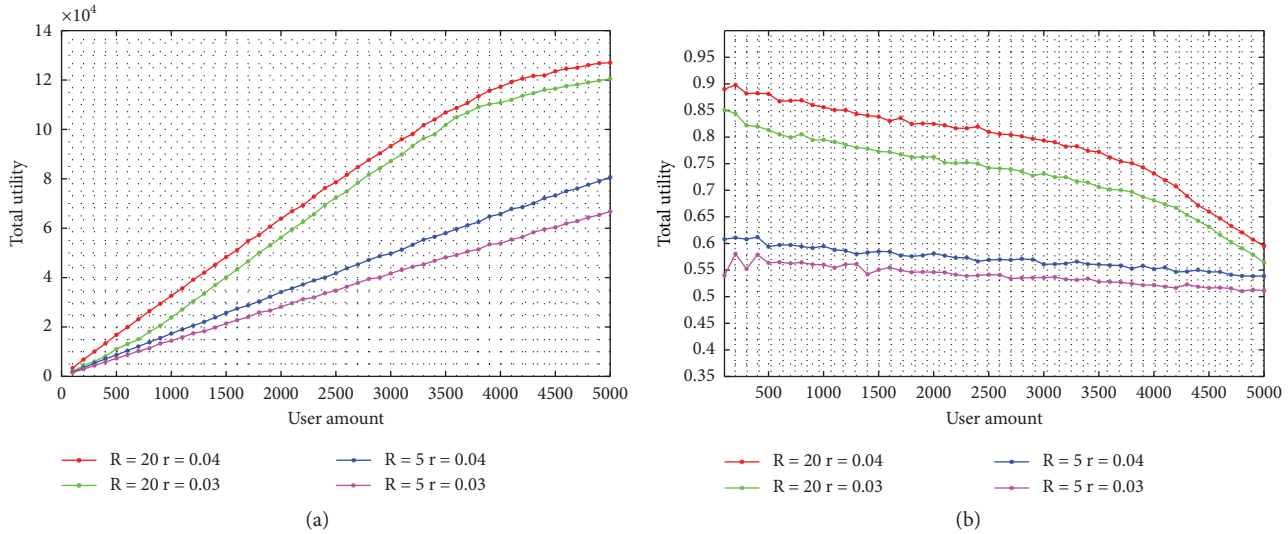


Figure 3: The effect of **R** and **r** for PoW consensus mechanism on total utility and winning ratio. (a) Total utility. (b) User winning ratio.

prefer to trade VM instances of the edge level. As shown in Figure 5(a), the total utility of STGA and STMA is far less than that of G-TRAP since they only provide edge-level resources and offer group buying discounts without applying cloud-level resources.

Figure 4(b) shows the percentage of winners with different sets of instances. When the number of users does not exceed 4300, the percentage of winners exceeds 70%, up to 90%, and not less than 55% at worst. It can be considered that G-TLRA performs well in intensive scenarios of computation offloading. Again, the winning ratio is highest with the setting of $\alpha_1 = 0.9$ and $\alpha_2 = 0.1$. As shown in Figure 5(b), the winning ratio of G-TRAP is much higher than that of STGA and STMA. In the STGA and STMA, each user can purchase VM instances from one C-ESP, and the resources of edge level are insufficient, which reduces the probability of users successfully obtaining a bundle of VM instances. However, C-ESPs in G-TRAP can serve more users by providing more VM instances from cloud level.

It can be seen from Figure 4(c) that when the number of users increases, the resource utilization of G-TRAP first reaches 1 with the setting of $\alpha_1 = 0.9$ and $\alpha_2 = 0.1$. Thus, G-TRAP performs better with the rise in the weight of edge-level resources. However, STGA and STMA cannot achieve the full utilization of resources, as shown in Figure 5(c). In STGA and STMA, the provider sells VM instances of the edge level where remaining resources cannot satisfy the demands of resources requested from the user behind. In summary, G-TRAP is more suitable for various scenarios of intensive computation offloading than STGA and STMA.

5.2.3. Impact of Communication Distance of Users. In Figure 6, we vary the distance range between users and C-ESPs as [40, 70], [40, 80], and [40, 90] and then compare with previous results. As expected, the larger communication distance range will lower the percentage of winners and lower the resource usage ratio. This is because that the longer uploading distances of tasks cause more communication loss, and the bids of C-ESPs will increase, which lowers the ranking of several C-ESP and reduces the matching between C-ESPs and users in turn.

5.2.4. Impact of Cloud-Edge Capacity Ratio. Further, we examine the impact of cloud-edge ratio C^r on the performance of G-TRAP. The total number of VM instances of C-ESPs is set to follow the uniform distribution of $U(0, 2000)$, and the number of users is set to 5000; then, we change the value of C^r in G-TRAP.

Figure 7(a) shows the effects of C^r on the execution time of G-TRAP with different sets of VM instances. It can be seen that the execution time of G-TRAP is insensitive to variations in C^r and is less than 0.018 seconds, which can

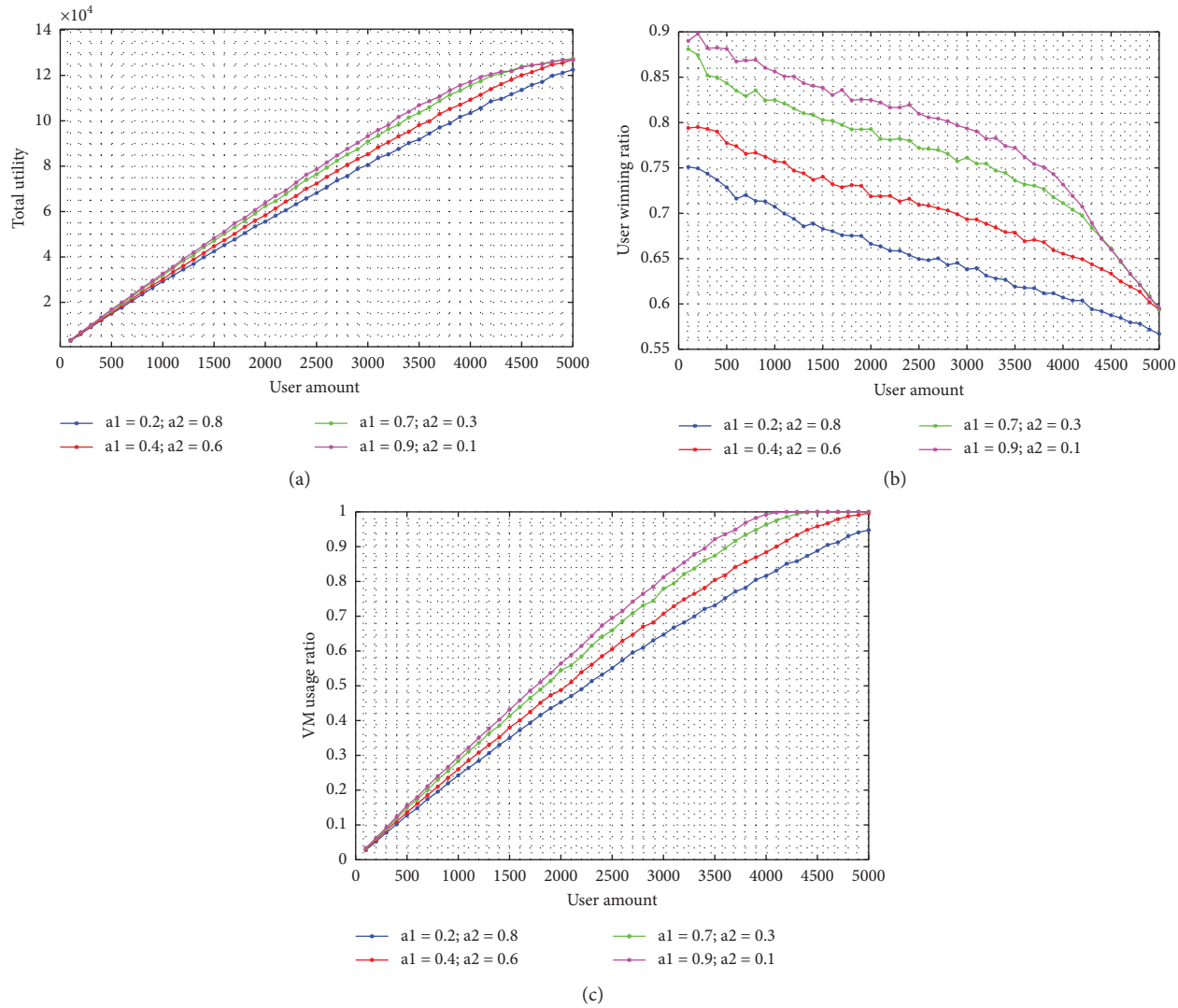


Figure 4: The effect of the total number of users on total utility, winning ratio of users, and VM usage ratio. (a) Total utility. (b) User winning ratio. (c) VM usage ratio.

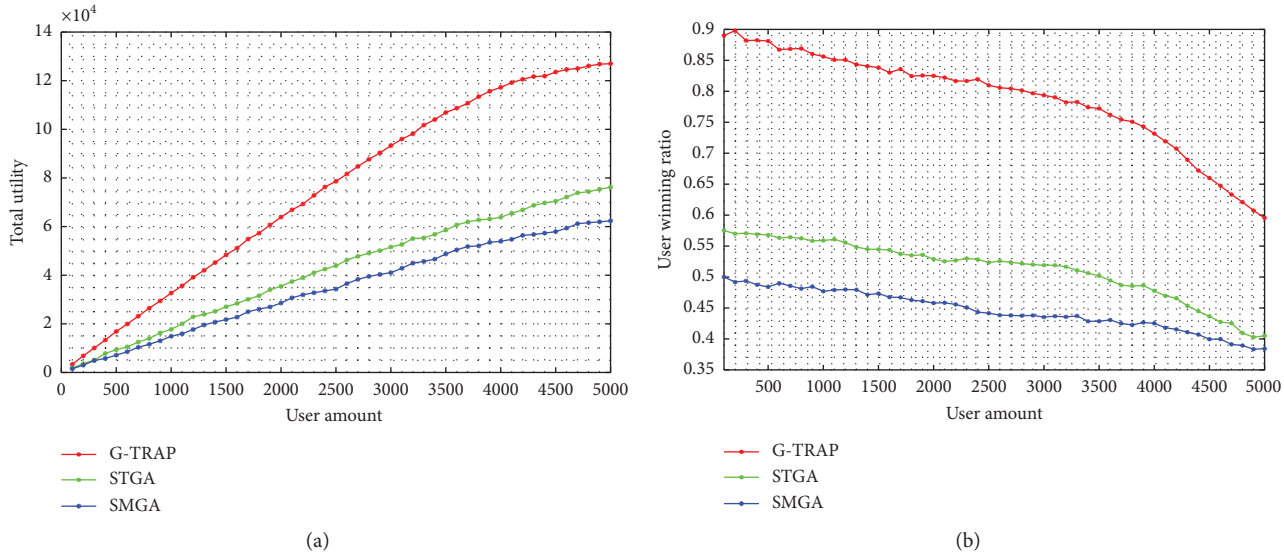
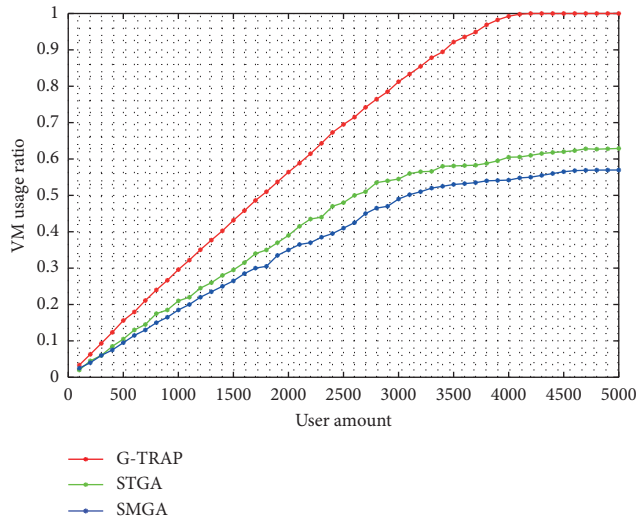
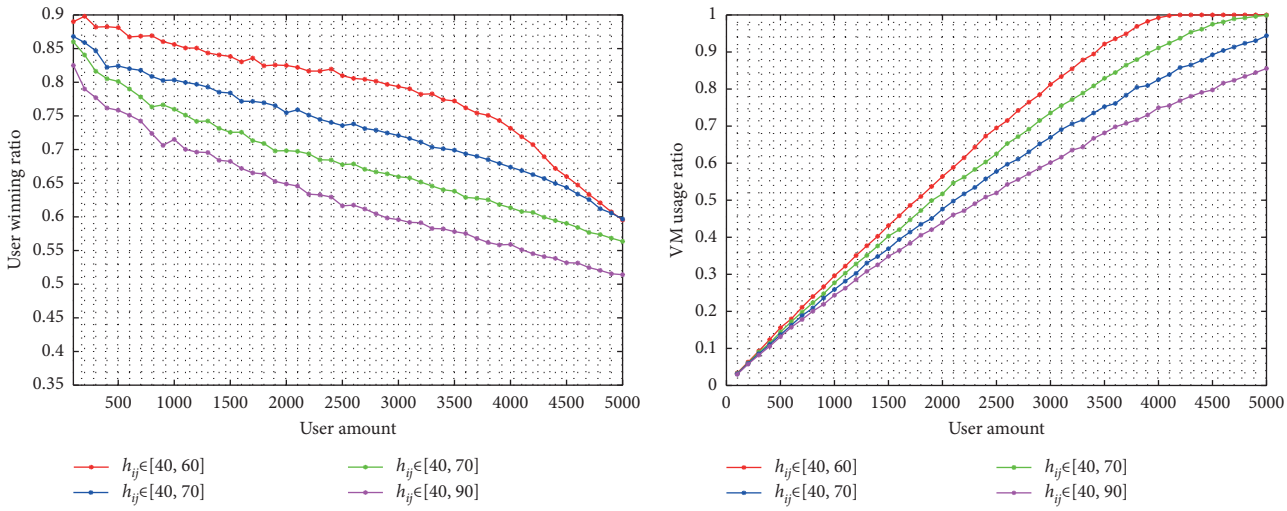


Figure 5: Continued.



(c)

Figure 5: The comparison of total utility, winning ratio of users, and VM usage ratio of G-TRAP with STGA and STMA. (a) Comparison of total utility. (b) Comparison of user winning ratio. (c) Comparison of VM usage ratio.



(a) (b)

Figure 6: The effect of communication distance on user winning ratio and VM usage ratio. (a) User winning ratio. (b) VM usage ratio.

verify that the computing efficiency of G-TRAP is well when serving a large-scale user group.

Figure 7(b) shows the impacts of C^r on the number of served users. When C^r is 1, the value of α_1 and α_2 has little effect on the number of users matched with C-ESP successfully, while the higher C^r is, the larger the difference of served user amount with different sets of α_1 and α_2 is. Moreover, the larger difference between α_1 and α_2 brings a greater change in the number of served users. With the increasing of C^r, the served user amount of the last two sets increases well, that is $(\alpha_1 = 0.7, \alpha_2 = 0.3)$ and $(\alpha_1 = 0.9, \alpha_2 = 0.1)$, while decreasing in the first two sets. This is because that the total bids of C-ESPs remains unchanged when C^r increases, and the total amount of resources raises because of the increase in ratio between α_1 and α_2. As a result, the bid density of C-ESP decreases so that it matches more users. In a word, increasing the difference between α_1 and α_2 and the value of C^r can encourage more users to participate in the auction.

Figure 7(c) shows the effects of the total utility of G-TRAP with varying settings of instances. Similarly, the total utility does not change much with different sets of α_1 and α_2 when C^r is 1. As C^r increases and α_1 is less than 0.5, the bid density of C-ESP increases, which can lead to a reduction in the number of served users and reduce the total utility.

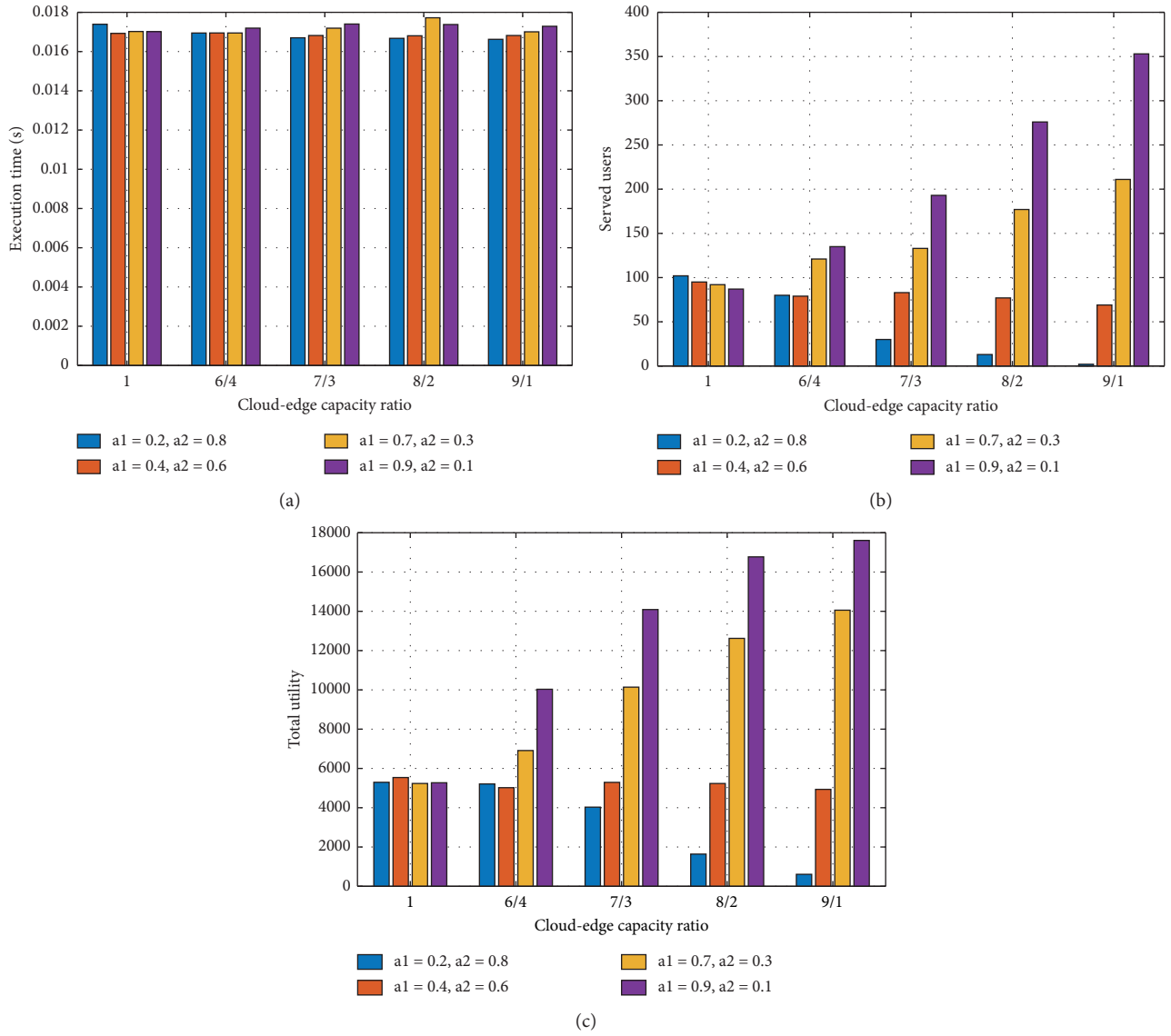


FIGURE 7: The effect of cloud-edge capacity ratio, $\mathbf{C}^{\mathbf{r}}$, on the execution time, served user amount, and total utility. (a) Execution time. (b) Served users. (c) Total utility.

6. Conclusions

In this paper, an efficient resource allocation with a pricing mechanism, G-TRAP, is proposed to support the offloading of compute-intensive tasks generated by mobile blockchain from mobile devices to the cloud-edge computing system. In this mechanism, the two-level resource allocation system provides users more bidding options, which increases the matching probability between users and C-ESPs. And the improved VCG pricing mechanism guarantees the individual rationality of participants and maximizes the benefits of C-ESPs as much as possible. Moreover, the proposed mechanism is proved to satisfy the economic properties of the auction model by theoretical analysis. The simulated results prove that this mechanism can run with short execution time and the near optimality of resource utilization, which make it a suitable candidate for deployment in current and future mobile blockchain systems and other service applications involving massive computing tasks.

Data Availability

The simulation data of our algorithm implementation, which is used to support the findings of this study, are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the project of National Science and Technology Major Project of the Ministry of Science and Technology of China (no. 2018ZX03001014-003).

References

[1] K. Suankaewmanee, D. T. Hoang, D. Niyato, S. Sawadsitang, P. Wang, and Z. Han, "Performance analysis and application of mobile blockchain," in *Proceedings of the 2018 International Conference on Computing, Networking and Communications (ICNC)*, pp. 642–646, Maui, HI, USA, March 2018.

[2] Y. Zou, T. Meng, P. Zhang, W. Zhang, and H. Li, "Focus on blockchain: a comprehensive survey on academic and application," *IEEE Access*, vol. 8, pp. 187182–187201, 2020.

[3] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[4] M. Liu, F. R. Yu, Y. Teng, V. C. M. 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. 11008–11021, 2018.

[5] S. Nakamoto, *Bitcoin: A Peer-To-Peer Electronic Cash System*, 2008, https://bitcoin.org/bitcoin.pdf.

[6] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: analysis and applications," in *Advances in Cryptology - EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Part II*, pp. 281–310, Springer, Berlin, Germany, 2015.

[7] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[8] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5506–5519, 2018.

[9] S. Yu, R. Langar, X. Fu, L. Wang, and Z. Han, "Computation offloading with data caching enhancement for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11098–11112, 2018.

[10] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its rolein the internet of things," in *Proceedings of the MCC Workshop on Mobile Cloud Computing*, Helsinki, Finland, August 2012.

[11] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, *Mobile Edge Computing Introductory Technical White Paper*, ETSI, Sophia Antipolis, France, 2014.

[12] Y. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing: a key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[13] V. Krishna, *Auction Theory*, Academic Press, Cambridge, MA, USA, 2009.

[14] Y. Jiao, P. Wang, D. Niyato, and Z. Xiong, "Social welfare maximization auction in edge computing resource allocation for mobile blockchain," in *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.

[15] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal pricing-based edge computing resource management in mobile blockchain," in *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.

[16] Y. Li, J. Wu, and L. Chen, "POEM+: pricing longer for mobile blockchain computation offloading with edge computing," in *Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 162–167, Zhangjiajie, China, August 2019.

[17] T. Bahreini, H. Badri, and D. Grosu, "An envy-free auction mechanism for resource allocation in edge computing systems," in *Proceedings of the 2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pp. 313–322, Seattle, WA, USA, October 2018.

[18] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[19] X. Li, Y. Dang, M. Aazam, X. Peng, T. Chen, and C. Chen, "Energy-efficient computation offloading in vehicular edge cloud computing," *IEEE Access*, vol. 8, pp. 37632–37644, 2020.

[20] Z. Hong, W. Chen, H. Huang, S. Guo, and Z. Zheng, "Multi-hop cooperative computation offloading for industrial IoT-edge-cloud computing environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2759–2774, 2019.

[21] R. Singhal and A. Singhal, "A combinatorial economical double auction resource allocation model (CEDARA)," in *Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pp. 497–502, Faridabad, India, February 2019.

[22] W. Sun, J. Liu, Y. Yue, and H. Zhang, "Double auction-based resource allocation for mobile edge computing in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4692–4701, 2018.

[23] X. Liu, J. Wu, L. Chen, and C. Xia, "Efficient auction mechanism for edge computing resource allocation in mobile blockchain," in *Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 871–876, Zhangjiajie, China, August 2019.

[24] X. Li, Z. Lian, X. Qin, and J. Abawajyz, "Delay-aware resource allocation for data analysis in cloud-edge system," in *Proceedings of the 2018 IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pp. 816–823, Melbourne, Australia, December 2018.

[25] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019.

[26] J.-S. Fu, Y. Liu, H.-C. Chao, B. K. Bhargava, and Z.-J. Zhang, "Secure data storage and searching for industrial IoT by integrating fog computing and cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4519–4528, 2018.

[27] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 726–738, 2019.

[28] Y. Narahari, *Game Theory and Mechanism design*, pp. 231–273, China Remin University Press (CRUP), Beijing, China, 2017.

[29] G. Gao, M. Xiao, J. Wu, H. Huang, S. Wang, and G. Chen, "Auction-based VM allocation for deadline-sensitive tasks in distributed edge cloud," *IEEE Transactions on Services Computing*, 2019, In press.