Combinatorial Auction-enabled Dependency-Aware Offloading Strategy in Mobile Edge Computing

Hong Kang, Minghao Li, Sizheng Fan, and Wei Cai*
School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China
{*hongkang1, minghaoli1, sizhengfan*}*@link.cuhk.edu.cn, caiwei@cuhk.edu.cn*

2023 IEEE Wireless Communications and Networking Conference (WCNC) | 978-1-6654-9122-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/WCNC55385.2023.10118890

***Abstract*—Mobile Edge Computing (MEC) enables computation offloading from resource-constrained mobile devices to edge servers in close vicinity, effectively promoting the user experience on emerging interactive multimedia applications such as virtual/augmented reality, mobile gaming, and mobile video editing. However, most contemporary MEC offloading research disregards the interdependencies between partitioned subtasks of application. Also, few studies focused on application topologies have neglected to design effective incentives to encourage edge servers to provide offloading services. In this paper, we propose a dependency-aware offloading algorithm based on a multi-round truthful combinatorial reverse auction (MTCRA) to address the social welfare maximization problem in the paradigm of MEC. Building on the topology of directed acyclic graphs (DAGs) modeled from applications, we discuss the complementarity and substitutability of subtasks in the context of combinatorial auction. Theoretical analysis shows that the presented auction mechanism achieves computing efficiency while maintaining desirable economic features like truthfulness, individual rationality, and budget balance. Simulation results demonstrate that the proposed algorithm achieves high social welfare regarding reduced execution time and good economic benefits for MEC servers.**

***Index Terms*—Mobile edge computing, auction mechanism, computational offloading, dependent application**

I. Introduction

As the demand for computation-intensive multimedia applications grows, such as virtual/augmented reality, mobile gaming, and mobile video editing. The limitations of computing resources and battery capacity due to the physical properties of the mobile device severely affect the user experience. Mobile edge computing (MEC) is a promising technology to overcome the above problems of mobile devices and improve users' quality of experience (QoE). Many previous works [1]–[3] have studied allocating resources and scheduling tasks efficiently in MEC systems. These studies treat the task as an indivisible unit that is offloaded to the edge server or reserved locally. However, mobile applications often consist of interdependent fine-grained subtasks in practice.

Recent research considered the offloading of dependent tasks, leveraging the parallelism within the application to further reduce the overhead of the mobile device. Ding *et al.* proposed an offloading strategy [4] in a resources-restricted multi-user and multi-edge environment to minimize the execution overhead. Chen *et al.* [5] modeled the dependent task offloading problem as a Markov decision process and designed an Actor-Critic mechanism for DAG-based dependent tasks computation offloading. An intelligent computational offloading scheme [6] for dependent IoT applications (CODIA) was framed with a prioritized scheduling strategy and a reinforcement learning (RL)-enabled offloading solution. However, an essential premise of these studies is that the edge servers engage in volunteer service without compensation, which is unrealistic. Therefore, an incentive mechanism should be introduced to encourage MEC servers to provide offloading services with tasks that have dependent relationships.

Realizing that no current work combines incentives well with dependent computation offloading, we propose a combinatorial auction-enabled computing service trading mechanism for dependent tasks in MEC systems. Our MTCRA mechanism is expected to address the following challenges: 1) although the economic effectiveness of the auction theory for edge computing has been shown [7], [8], it is still worthwhile to investigate how to integrate the offloading and scheduling of dependent tasks with the auction mechanism and how dependencies can be expressed in the pricing and bidding phases; 2) how to design a truthful auction mechanism that can maintain long-lasting and fair trading between mobile devices and MEC servers to eliminate the impact of malicious bidding; and 3) how to ensure that it is beneficial for both parties involved, reducing the makespan of the mobile application while ensuring that the utilities of the MEC server are positive.

Combinatorial auction [9] differs from traditional auctions in that bidders can bid on a combination of items (bundle). It is suitable when buyers provide a non-additive measurement of products' value, as known for complementarity and substitutability [9]. To incorporate the consideration of dependency in the pricing of the mechanism, we model the mobile application as a DAG, and define complementarity and substitution of subtasks based on their parameters and topological relationships. Secondly, we utilize a concurrent provider and consumer (CPC) model [10] to decompose the dependencies between subtasks into several groups. According to the characteristics of the CPC model, we propose a bidding strategy for MEC servers to build their bundles in each auction round and maximize their own benefits. Furthermore, the Vickery payment [11] is adopted to ensure the truthfulness of the mechanism. Selfish and malicious bids do not bring additional gains, ensuring fairness in trading.

In summary, the following contributions are made in this paper: 1) we propose a multi-round truthful combinatorial

*corresponding author

reverse auction to address the social welfare maximization problem for dependent task offloading in the MEC paradigm. To the best of our knowledge, this is the first work combining the dependency-aware task offloading with the combinatorial auction; 2) we apply theoretical analysis to demonstrate that the proposed MTCRA mechanism satisfies economic properties, including truthfulness, individual rationality, budget balance, and computation efficiency; 3) we perform extensive experiments to evaluate the effectiveness and performance of the proposed mechanism. The simulation results show that MTCRA performs substantially superior to baseline algorithms in application makespan and social welfare.

II. System Model and Problem Formulation

A. System Model

As the system shown in Fig. 1, we consider the MEC servers denoted as $\mathbb{S} = \{s_1, s_2, ..., s_m\}$, are heterogeneous and eligible to provide computing service to the mobile device s_l in the network. With regard to the heterogeneity of a MEC server, it is defined explicitly as $s_i = \{f_i, B_i, d_i\}$. The parameters represent the CPU frequency, network bandwidth and distance from s_i to local device s_l, respectively. For ease of understanding, we assume that there is one CPU core available for execution on both entities. Likewise, the paradigm where a MEC server has parallel processing capabilities may be readily expanded.

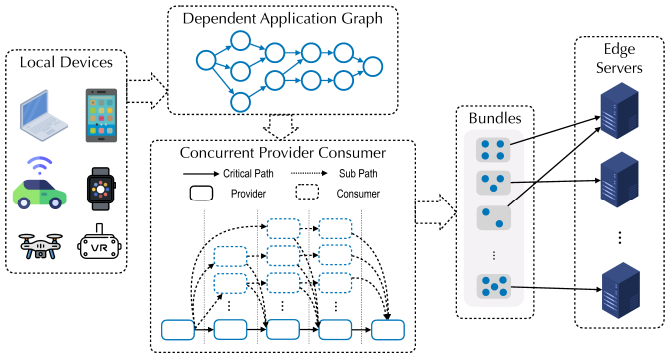


Fig. 1. Framework of the MTCRA offloading system for MEC computing

B. Application Model

A mobile application is a collection of fine-grained subtasks or components. Considering the topological structure and dependencies between tasks, we build the application as a directed acyclic graph $\mathbb{G} = \{\mathbb{V}, \mathbb{E}\}$. The set of components $\mathbb{V} = \{v_1, v_2, ..., v_n\}$ denotes the decomposed tasks in dependency-aware. For each $v_j = \{\sigma_j, \omega_j, \Theta_j\}$, where σ_j and ω_j indicate the data size and computation work load (required CPU cycles) of v_j, respectively. $\Theta_j = \{\theta_{jj'}, ...|e(v_j, v_{j'}) \in \mathbb{E}\}$ is about the size of returned data that must be transmitted to the connected components. For example, components v_k and v_j are linked by a directed edge $e(v_k, v_j) \in \mathbb{E}$, showing the dependency between them, which v_j can not start execution until v_k is finished. Here, v_k is defined as the predecessor of v_j, written as $pred(v_j) = \{v_k \in \mathbb{V}|e(v_k, v_j) \in \mathbb{E}\}$. In contrast, v_j is defined as the successor of v_k, denoted as $succ(v_k) = \{v_j \in \mathbb{V}|e(v_k, v_j) \in \mathbb{E}\}$. Each component v_j can only be carried out once, either locally or on MEC servers. Let $x_{i,j} \in \{0, 1\}$ denote whether the task v_j is offloaded to MEC server s_i. X is a $(m+1) \times n$ matrix to record $x_{i,j}$, and the index zero indicate local execution condition.

A path of a DAG contains a sequence of interdependent components $(e(v_k, v_{k+1}) \in \mathbb{E})$, written as $\lambda = \{v_a, v_b, ..., v_z\}$. The longest path in a DAG is defined as λ^*, also nominated as the critical path [10], which contains the largest workload. In the ideal parallelism maximization condition, the application's overall execution time should be the execution time of subtasks on the critical path (λ^*). Non-critical components ($v_k \notin \lambda^*$) can take advantage of gaps in the execution of critical components and execute in parallel on other MEC servers. A concurrent provider and consumer (CPC) model [10] shown in Fig.1 can be applied to describe the relationship as mentioned above. The critical components are identified as providers (Φ^*), who offer the time capacity that non-critical components specified as consumers (Φ) can utilize to execute concurrently on other MEC servers. The providers are constructed by assembling components in λ^* sequentially until non-critical components cause potential inferences related to precedence limitations.

Based on generated providers, we assign the corresponding consumers for each provider based on two principles. First, the associated consumers (Φ_k) of a provider (Φ_k^*) must be able to execute simultaneously with it, which means there is no topological dependency (ancestors or descendants) between components in Φ_k and Φ_k^* [12]. Second, the consumer's execution may postpone the next provider's earliest start time and finish time of the whole application.

C. Communication Model

In this work, the transmission channel is modeled based on the Rayleigh fading channel [13], considering the densely populated urban areas effect on radio signals. The bandwidth of orthogonal channels allocated to MEC servers are B_i, and B_l to the mobile device. The data transmission rate from the mobile device for the MEC server s_i is

$$R_{l,i} = B_l \log_2 \left(1 + \frac{\rho_l h_{l,i}}{d_{l,i}^{\alpha} N_0}\right), \tag{1}$$

where ρ_l represents the transmission power of the mobile device s_l. The parameters $h_{l,i}$ and $d_{l,i}$ are the channel gain of the wireless channel as well as the geographical distance between s_l and s_i, respectively. The exponent α is a pathloss factor, and N_0 shows the additive white Gaussian noise (AWGN). The communication model between MEC servers can be referred to Eq. (1). We disregard the download time, which is negligible in comparison with upload time.

D. Local Computation Model

Because of the dependency among subtasks in the application $\mathbb{G}$, all of immediate predecessors must be finished before

the execution of task v_j. Thus, the earliest start time of task v_j executed locally can be formulated as

$$T_l^{j-} = \max_{v_{j'} \in \text{pred}(v_j)} \max\left\{T_i^{j'*}, T_l^{j'*}\right\}, \tag{2}$$

where $v_{j'}$ is one of the predecessor of v_j. $T_i^{j'*}$ and $T_l^{j'*}$ indicate the finish time of the task $v_{j'}$ executed on the MEC sever s_i or the mobile device locally.

We assume that the CPU frequency of the mobile device is f_l (cycles per second), the local execution time of the task v_j is $t_l^{j,e} = \frac{\omega_j}{f_l}$. After the task v_j is completed, the results should transmit to the position (s_i) where its successors processed. Based on the communication model in Section II-C, we have

$$t_{l,i}^{j,tr} = \begin{cases} 0, & i = l \\ \sum_{v_{j'} \in succ(v_j)} \frac{\theta_{jj'}}{R_{l,i}}, & \text{otherwise} \end{cases} \tag{3}$$

When both v_j and $pred(v_j)$ are executed locally, $t_{l,i}^{j,tr}$ should be zero. Instead, $t_{l,i}^{j,tr}$ is set to the returned data size divide the transmission rate. Moreover, the finish time of task v_j performed locally is obtained by

$$T_l^{j*} = T_l^{j-} + t_l^{j,e} + t_{l,i}^{j,tr}. \tag{4}$$

The execution energy cost of task v_j performed locally is $E_l^{j,e} = \omega_j \kappa_l f_l^2$, where κ_l [14] is the chip architecture's coefficient factor for the mobile device. The cost of the transmission energy is calculated as $E_{l,i}^{j,tr} = \rho_l^{j,tr} \times t_{l,i}^{j,tr}$, where $\rho_l^{j,tr}$ is the transmission power of the mobile device. So far, we conclude the total energy consumed for task v_j finished on the mobile device is

$$E_l^j = E_l^{j,e} + E_{l,i}^{j,tr}. \tag{5}$$

E. Edge Computation Model

The offloaded tasks is processed in accordance with the First-Come-First-Serve (FCFS) and in a non-preemptive way on MEC servers. Due to interdependent restrictions, if task v_j is offloaded to MEC server s_i and task $v_{j'} \in pred(v_j)$ is offloaded $s_{i'}$, then we can calculate the time v_j is ready to execute:

$$T_i^{j-} = \max_{v_{j'} \in \text{pred}(v_j)} \max\left\{T_i^{j'*}, T_l^{j'*}\right\} + t_j^{tr*}, \tag{6}$$

where t_j^{tr*} is the completion time of the task v_j itself is transmitted to s_i from the mobile device, represented as $t_j^{tr*} = \frac{\sigma_j}{R_{i,i'}}$. Similar to Eq. (4), the completion time of v_j executed remotely on s_i can be expressed as

$$T_i^{j*} = T_i^{j-} + t_i^{j,e} + t_{i,i'}^{j,tr}. \tag{7}$$

The computation completion time t_i^j is derived by $t_i^{j,e} = \frac{\omega_j}{f_i}$. In addition to the transmission time of v_j's result can be referred to Eq. (3). The total energy cost of computing component v_j on s_i can be referred to Eq. (5). where the detailed calculation of $E_i^{j,e}$ and $E_i^{j,tr}$ can be refereed to $E_l^{j,e}$ and $E_l^{j,tr}$ by changing the index l to i, respectively. Meanwhile, the transmission energy of the mobile device for transferring the subtask v_j to the MEC server s_i is:

$$E_l^j = \rho_l^{j,tr} \times t_j^{tr*}. \tag{8}$$

F. Utility Function

The power consumption of the mobile device can be formulated as

$$\varepsilon_l = \frac{\sum_{j=1}^{n} E_l^{j,e}}{\sum_{j=1}^{n} t_l^{j,e}} = \kappa_l f_l^3. \tag{9}$$

Then, the valuation of the mobile device is derived by the energy retained to finish the subtask v_j through the proposed mechanism, as shown below:

$$\vartheta_l^j = \varepsilon_l (T_l^{j*} - T_i^{j*}) - E_l^j. \tag{10}$$

Thus, the utility of user can be formulated as its valuation minus the payment to MEC servers who are auction winners at each auction round:

$$U_l = \sum_{i=1}^{m} \sum_{j=1}^{n} (\vartheta_l^j - p_i^j) x_{ij}, \tag{11}$$

where p_i^j is the final price that the mobile device pay to MEC server s_i for processing the subtask v_j. As for MEC servers, they submit bids truthfully, which are directly related to the energy consumption of finishing components offloaded to them. The utility can be represented as the revenue from participating in the auction by subtracting the energy cost when providing offloading service:

$$U_i = \sum_{j=1}^{n} (p_i^j - E_i^j) x_{ij}. \tag{12}$$

G. Problem Formulation

Based on the above mentioned model, our goal is to maximize the social welfare with joint consideration for utilities of both entities in the auction. Because the MTCRA mechanism is budget balanced, the total payment from the mobile device is equal to the revenue received by all MEC servers. Consequently, the overall objective function becomes:

$$\begin{aligned} \max \quad & \sum_{i=1}^{m} \sum_{j=1}^{n} (\vartheta_l^j - E_i^j) x_{ij} \\ \text{s.t.} \quad & C1: T_i^{j*} - T_i^{j-} \le t_l^{j,e}, \forall s_i \in \mathbb{S}, \forall v_j \in \mathbb{V}, \\ & C2: T_i^{j*} \le T_i^{j'-}, \forall e(v_j, v_{j'}) \in \mathbb{E}, \\ & C3: \sum_{j=1}^{N} x_{ij} \le 1, \forall s_i \in \mathbb{S}, \\ & C4: x_{ij} \in \{0,1\}, \forall s_i \in \mathbb{S}, \forall v_j \in \mathbb{V}. \end{aligned} \tag{13}$$

Here, C1 is time constraint. The execution time of the component on MEC servers must be at least as good as the local execution. Otherwise, it will be finished locally. C2 demonstrates the constraint of dependent relationship among subtasks of the application. Every component v_j can be started to execute until its predecessors are finished. The constraints (C3 and C4) restricted that a component can only be offloaded to one edge servers or executed locally. The above mentioned problem is NP-hard, which is easily proved by a polynomial-time reduction from the known 0-1 knapsack problem [15].

III. Auction Mechanism

We take auction theory to tackle the aforementioned social welfare maximization problem. This section illustrates the desired economic properties, the bidding strategy, winner selection, and payment determination of the mechanism.

A. Overview and Design Goal

The proposed mechanism is a multi-round combinatorial auction ends with all components are placed on the winning edge servers or mobile devices, as shown in Algorithm 1. The MEC server s_i participate in each auction round as the bidder and submit a bundle of components b_i maximizing valuation under the time constraint and the corresponding price p_i to the auctioneer. After collecting all bids from MEC servers, the auctioneer selects the winning servers greedily and determines the final price for the winners in each round of the auction. Meanwhile, the following economic properties are expected:

- **Individual Rationality (IR):** For every MEC servers who participate in the proposed auction mechanism, their utility should be non-negative.
- **Truthfulness (TF):** Also known as individual compatibility (IC), all agents will bid according to their true costs and disregard the bids of other competitors to maximize their utilities. For any MEC server s_i, $u_i(c_i) \geq u_i(b_i)$, where c_i holds for true cost of s_i and $b_i \neq c_i$.
- **Budget Balance (BB):** The payment from users equals the total payment received by MEC servers, then the total payment of the participants in the auction is zero.
- **Computation Efficiency (CE):** The proposed mechanism can be completed in polynomial time.

Combining the topology and parameters of components in $\mathbb{G}$, we define their complementarity and substitutability in Definition 1 and 2, which correspond to the properties of items in the combinatorial auction. Before the start of the MTCRA mechanism, we further merge the subtasks has dependency relationship in a same consumer set Φ_k considering Definition 1.

Definition 1. *(Complementarity of substasks)* $\forall v_j, v_{j'} \in \mathbb{V}, \exists e(v_j, v_{j'}) \in \mathbb{E}$*, then* v_j *and* $v_{j'}$ *are complementary to each other. That is, higher social welfare is achieved when both components on the same position because the cost of data transfer between them will be eliminated.*

Definition 2. *(Substitutability of substasks)* $\forall \phi_k \in \Phi_k, \exists \sum_{v_a \in \phi_k} \omega_a \geq \sum_{v_b \in \Phi_k^* } \omega_b$*, then the components in* ϕ_k *are substitutable to each other when the whole subset* ϕ_k *is offloaded to one MEC server. Mobile devices become less profitable because it extends the overall application completion time. Also, the MEC server who submits a bundle like* ϕ_k *gets less chance to win the auction.*

B. Bidding Strategy

This section demonstrates how MEC servers construct their bundles and submit bids $\mathcal{B}_i = \{b_i, p_i\}$ in each auction round. The execution time of components on the critical path (λ^*), also known as providers (Φ^*) in the CPC model, represents the ideal makespan of the whole application. Therefore, components in providers have the highest priority for mobile devices being auctioned. Also, rational MEC servers will not be willing to allow themselves to be idle, so they should bid for a bundle of components with the highest workload, which is exactly the providers. By valuation analysis from both entities, we conclude that the MEC servers should bid for bundles of all critical components (providers) in the first round.

After the first round, we formulate the bundle construction problem as a 0-1 knapsack problem for each consumer set Φ_k. With the help of the merged CPC model, we divide non-critical components of the application into several consumer sets. In each Φ_k, components could concurrently execute with corresponding providers. Besides, components in each consumer set do not cause a potential delay to the corresponding provider directly by definition of the CPC model, but it affects the start time of the components in the following provider and the makespan of the application, which leads to Definition 2.

Hence, we should limit the total execution time of the selected components in each Φ_k within the execution time of all components of its provider (τ_k), indicated in constraint (C5). After denoting the winner of the first round as s_w, we have $\tau_k = \sum_{v_j \in \Phi_k^*} t_j^{tr*} + t_w^{j,e} + t_{w,w'}^{j,tr}$. Then, the bundle construction in Φ_k can be represented as:

$$\begin{aligned} \max \quad & \sum_{v_{j'} \in \Phi_k} p_i x_{ij'} \\ \text{s.t.} \quad & C5: \sum_{v_{j'} \in b_i} (t_{j'}^{tr*} + t_i^{j',e} + t_{i,i'}^{j',tr}) x_{ij'} \leq \tau_k, \\ & C6: x_{ij'} \in \{0,1\}, \forall v_{j'} \in \Phi_k, \forall s_i \in \mathbb{S} \end{aligned} \tag{14}$$

Due to the NP-hardness of the problem mentioned above, we use a greedy algorithm for handling it in polynomial time. The merged nodes in consumer Φ_k are sorted by their value

Algorithm 1: MTCRA mechanism

input : $\mathbb{G} = \{\mathbb{V}, \mathbb{E}\}, \mathbb{S} = \{s_1, s_2, ..., s_m\}$
output: Offloading decision X

Initialization: construct CPC model (Φ^**,* Φ*)*
while $v_j \in \mathbb{V}$ *and* $\sum_{i=0}^{m} x_{ij} = 0$ **do**
  /* Bidding Submission */
  for $s_i \in \mathbb{S}$ **do**
    $\mathcal{B}_i \leftarrow$ BundleCons(Φ^*, Φ, s_i)
    $\mathfrak{B}$.append($\mathcal{B}_i$)
  end
  /* Winner Selection */
  for $\mathcal{B}_i \in \mathfrak{B}$ **do**
    $r_i \leftarrow$ Eq. (16)
    $bd_i \leftarrow (\frac{p_i}{\sqrt{\sum_{v_j \in b_i} \omega_i}} r_i)$
  end
  $bd_{w^*}, bd_{w^-} \leftarrow$ sort bd_i by ascending order
  remove b_w from Φ, Φ^*
  /* Payment determination */
  $p_w^* \leftarrow \frac{\sqrt{\Omega_i}}{r_w} bd_{w^-}$
  /* Updating offloading matrix */
  for $v_j \in b_w$ **do**
    $x_{wj} = 1$
  end
end

density $\frac{E_i^{j'}}{t_{j'}^{tr*}+t_i^{j',e}+t_{i,i'}^{j',tr}}$. The components with higher value density have higher priority to add to the bundle until the total time exceeds τ_k.

The whole process of bundle construction starts with the last set of consumers since it is possible to incorporate considerations for the complementarity of different components refer to Definition 1. The set of components selected in the kth consumer set of MEC server s_i is b_i^k, and the corresponding bid price is written as p_i^k. Additionally, the complete bundle submitted by s_i in the current auction round is the concatenation of the results from all consumer sets:

$$b_i = b_i^1 \cup b_i^2 \cup ... \cup b_i^K \tag{15}$$

The bid price for bundle b_i is calculated by $p_i = \sum_{k=1}^{K} p_i^k$. The winning MEC server will not bid again (consistent with Definition 2) until all of them become winners, and there are still components that have not been auctioned. At that point, the MEC server picks the component with the highest value among the remaining components.

C. Winner Selection

In each auction round, the collected bids from participants are defined as $\mathfrak{B} = \{\mathcal{B}_1, ..., \mathcal{B}_m\}$. A greedy winner selection algorithm is adopted by extending the $\sqrt{M}$-approximation method in [9]. Firstly, the auctioneer calculates the bid density $bd_i = \frac{p_i}{\sqrt{\Omega_i}} \times r_i$ for each bid $\mathcal{B}_i$. Where $\Omega_i = \sum_{v_j \in b_i} \omega_i$, representing the total workload of bundle b_i. Apart from that, r_i measures the performance ratio of the components in this bundle when executed on MEC servers s_i and mobile devices.

$$r_i = \frac{\sum_{v_j \in b_i} t_j^{tr*} + t_i^{j,e} + t_{i,i'}^{j,tr}}{\sum_{v_j \in b_i} t_l^{j,e}} \tag{16}$$

Furthermore, the auctioneer sorts the calculated bid density form each bid in the increasing order: $bd_1 \leq bd_2 \leq ... \leq bd_x$. Here, x is the number of MEC servers participant in. Based on the above list, the first MEC server is selected as the winner of current round of auctions. The components in its bundle are decided to offloaded to this sever.

D. Payment Rule

The final step is determining the payment for winners. Vickery payment [11] is considered to maintain the truthfulness property for our mechanism. Particularly, we designate s_w as the selected winners of the current auction round. Meanwhile, s_{w^-} is the winner without s_w in the same round. Now, we have the final payment $p_w^* = \frac{\sqrt{\Omega_i}}{r_w} \times bd_{w^-}$.

E. Analysis of Economic Properties

In this section, we demonstrate the desired economic properties described in Section III-A.

Theorem 1. *The proposed MTCRA mechanism is truthful.*

Theorem 2. *MTCRA achieves individual rationality.*

Theorem 3. *MTCRA mechanism is budget balanced.*

Theorem 4. *MTCRA achieves computation efficiency.*

The proofs of the above Theorems are presented in [16].

IV. Evaluation

In this section, we conduct extensive simulations to evaluate the performance of the proposed MTCRA algorithm.

A. Experimental Setting

We consider a network topology in a $60m \times 60m$ region, where the MEC servers are randomly distributed. The communication parameters of MEC servers are: $B_i = 20$MHz, $\alpha = 2.5$, $N_0 = 10^{-9}$, $\rho_i = 3$W and $h_i = 10^{-3}$ [4]. Besides, the CPU frequency $f_i = [3GHz, 3.5GHz]$ and $\kappa_i = [10^{-27}, 10^{-26.5}]$. As for the mobile device: $B_l = 20$MHz, $f_l = 2.5$GHz, and $\kappa_l = 10^{-26}$. The DAGs used in the evaluation are randomly generated by fixing the parallelism and changing the depth gradually. The average $\sigma_j = 100$KB, $\omega_j = 80$M cycles/bit, and $\Theta_j = 8$KB.

To evaluate the feasibility and performance of the proposed algorithm, we set the following algorithms as comparisons:

- AOLC: All components (subtasks) of the application are executed on the mobile device locally.
- AONEC: All components of the application are offloaded to the nearest MEC server of the mobile device.
- Random: The component is randomly offloaded to a MEC server or the mobile device itself.
- Greedy: Topological sorting the DAG firstly, then greedily select the most time-saving MEC server.
- VK: Topological sorting the DAG firstly, then apply iterative reverse Vickery auction [11] for offloading components with the time constrain.

B. Experimental Results

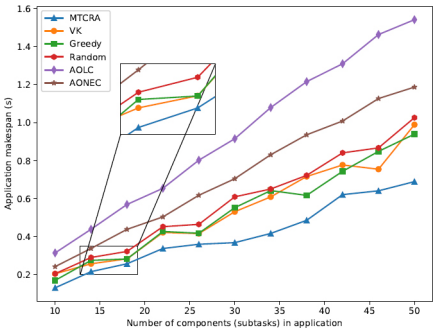


Fig. 2. Application makespan with the different number of components

1) Application makespan: We firstly fix the number of MEC servers to examine the overall makespan of the application. The parallelism of generated DAGs is limited to four with $n \in [10, 50]$. Fig. 2 shows the makespan performance of the proposed MTCRA mechanism and the above-mentioned compared methods with a diverse number of subtasks (n) on three MEC servers. The MTCRA algorithm achieves the shortest makespan than other baseline algorithms as the increase of n. Although the advantages are not apparent with small n, it increases as the DAG becomes more complex. Due to the topology of DAG being superficial at the small n, the dependencies between components have less impact on the offloading strategy. As the number of components increases,

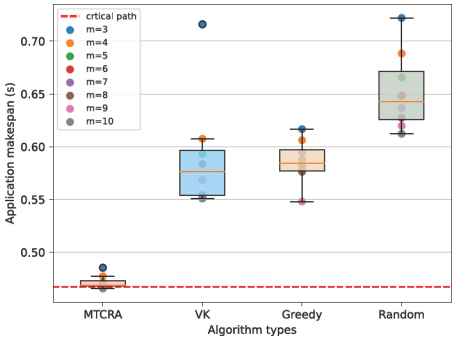
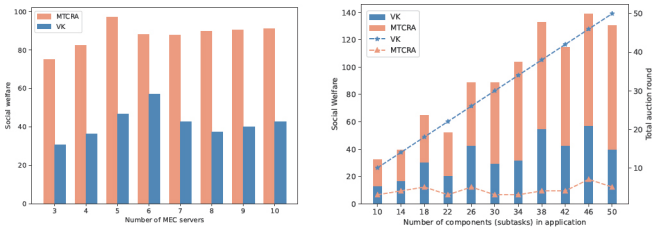


Fig. 3. Application makespan with different MEC numbers

this effect becomes more assertive, demonstrating effectiveness of dependency awareness in the MTCRA algorithm.

Besides, we keep n constant while changing the number of MEC servers ($m \in [3, 10]$) in evaluating the application makespan for different algorithms. Fig. 3 shows that the MTCRA consistently achieves the shortest latency. MTCRA's improvement in makespan associated with a higher number of MEC servers is not as apparent as in the other algorithms. Because it explores the parallelism of DAG, good performance has been achieved in the case of smaller m, so there is less room for improvement. The red line in Fig. 3 reflects average makespan of the critical path in the MTCRA algorithm, which representing the theoretical shortest execution time. The MTCRA's makespan is already approach to it.



(a) SW with m (b) SW with n

Fig. 4. Social welfare evaluation between MTCRA and VK

2) Social welfare: To evaluate the social welfare performance, we firstly preserve n at 50 while varying the number of MEC servers from 3 to 10. Fig. 4(a) shows that the MTCRA mechanism outperforms VK in all groups. The reason is that the MTCRA mechanism considers MEC both servers' makespan and bid prices in the winner selection criteria. This allows MTCRA to achieve better makespan performance while paying the similar payment.

Fig. 4(b) illustrates the social welfare performance for MTCRA and VK algorithms by changing $n \in [10, 50]$ and setting m to 9. The social welfare achieved by MTCRA is nearly 50% better than VK when dealing application with the same configuration. In addition, the comparison about total auction round of two algorithms is presented in Fig. 4(b). The total auction round of VK increases linearly with n since components are auctioned singly following the topological sequence. In addition, the total auction round for the MTCRA mechanism fluctuates around the application's parallelism, which improves the communication efficiency of auction mechanisms. Hence, the MTCRA mechanism achieved higher SW and communication efficiency than VK.

V. Conclusion

This paper proposes a multi-round combinatorial auction (MTCRA) for dependency-aware task offloading in a MEC scenario. We successfully fuse the dependency relationship among components in DAG-based mobile applications and the intrinsic properties of combinatorial auction. Theoretical analysis is given to prove the essential economic properties of truthfulness, individual rationality, budget balance, and computation efficiency. Extensive simulation results show the effectiveness of the MTCRA mechanism on both performance utilities and social welfare maximization.

Acknowledgement

This work was supported by National Natural Science Foundation of China under Project 61902333.

References

[1] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM transactions on networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[2] H. Zhang, S. Fan, and W. Cai, "Decentralized resource sharing platform for mobile edge computing," in *International Conference on 5G for Future Wireless Networks*. Springer, 2020, pp. 101–113.

[3] S. Fan, H. Zhang, Z. Wang, and W. Cai, "Mobile devices strategies in blockchain-based federated learning: A dynamic game perspective," *IEEE Transactions on Network Science and Engineering*, 2022.

[4] Y. Ding, C. Liu, X. Zhou, Z. Liu, and Z. Tang, "A code-oriented partitioning computation offloading strategy for multiple users and multiple mobile edge computing servers," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4800–4810, 2019.

[5] J. Chen, Y. Yang, C. Wang, H. Zhang, C. Qiu, and X. Wang, "Multi-task offloading strategy optimization based on directed acyclic graphs for edge computing," *IEEE Internet of Things Journal*, 2021.

[6] H. Xiao, C. Xu, Y. Ma, S. Yang, L. Zhong, and G.-M. Muntean, "Edge intelligence: A computational task offloading scheme for dependent iot application," *IEEE Transactions on Wireless Communications*, 2022.

[7] H. Qiu, K. Zhu, N. C. Luong, C. Yi, D. Niyato, and D. I. Kim, "Applications of auction and mechanism design in edge computing: A survey," *IEEE Transactions on Cognitive Communications and Networking*, 2022.

[8] S. Fan, H. Zhang, Y. Zeng, and W. Cai, "Hybrid blockchain-based resource trading system for federated learning in edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2252–2264, 2021.

[9] D. Lehmann, L. I. Oćallaghan, and Y. Shoham, "Truth revelation in approximately efficient combinatorial auctions," *Journal of the ACM (JACM)*, vol. 49, no. 5, pp. 577–602, 2002.

[10] S. Zhao, X. Dai, I. Bate, A. Burns, and W. Chang, "Dag scheduling and analysis on multiprocessor systems: Exploitation of parallelism and dependency," in *2020 IEEE Real-Time Systems Symposium (RTSS)*, 2020, pp. 128–140.

[11] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *The Journal of finance*, vol. 16, no. 1, pp. 8–37, 1961.

[12] Q. He, N. Guan, Z. Guo *et al.*, "Intra-task priority assignment in real-time scheduling of dag tasks on multi-cores," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2283–2295, 2019.

[13] B. Sklar, "Rayleigh fading channels in mobile digital communication systems. i. characterization," *IEEE Communications Magazine*, vol. 35, no. 9, pp. 136–146, 1997.

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

[15] M. R. Garey, "A guide to the theory of np-completeness," *Computers and intractability*, 1979.

[16] K. Hong, L. Minghao, S. Fan, and W. Cai, "Supplemental material: Combinatorial auction-enabled dependency-aware offloading strategy in mobile edge computing," https://www.dropbox.com/s/cdin0h88g7por9h/WCNC2023_Supplemental_Material.pdf?dl=0, 2022.