Bridging Incentives and Dependencies: An Iterative Combinatorial Auction Approach to Dependency-Aware Offloading in Mobile Edge Computing

Hong Kang, *Student Member, IEEE,* Minghao Li, Lehao Lin, Sizheng Fan, *Student Member, IEEE,* Wei Cai, *Senior Member, IEEE*

Abstract—As mobile applications grow increasingly computation-intensive, the challenges arising from the limitations of mobile devices in terms of computing resources and battery life become more pronounced. Mobile Edge Computing (MEC) provides a promising avenue to address these challenges and enhance user experience. While existing studies have extensively explored resource allocation and task scheduling in MEC, most treat tasks as monolithic entities, overlooking the nuanced subtasks/components that often make up mobile applications. This paper endeavors to bridge the gap between the need for incentive mechanisms and the offloading of dependent computation tasks in MEC. Drawing inspiration from auction theory, we introduce a novel multi-stage iterative combinatorial double auction (MICDA) mechanism, specifically tailored for dependent tasks in a cloud-edge-end cooperative computing scenario. Through theoretical analysis, the MICDA mechanisms demonstrate truthfulness, individual rationality, budget balance, and computational efficiency. Comprehensive experiment results further confirm its superior performance in improving application makespan and social welfare compared to other existing offloading strategies. This work validates the effective integration of dependency-aware computation offloading and auction mechanisms in overcoming economic and computational challenges in MEC systems, thereby paving the way for their potential application in broader real-world scenarios.

Index Terms—Mobile Edge Computing, Auction Theory, Incentive Mechanism, Computation Offloading, Dependent Application

1 Introduction

With the escalating demand for computation-intensive mobile applications, including autonomous driving [2], virtual and augmented reality [3], [4], and advanced mobile gaming [5]—the constraints imposed by the inherent physical properties of mobile devices, particularly in terms of computing resources and battery capacity, critically impinge on users experience. Mobile Edge Computing (MEC) emerges as a pivotal solution poised to surmount the aforementioned challenges, thereby enhancing the users' Quality of Experience (QoE) [6]–[9]. Numerous prior studies [10]–[12] have delved into the efficient allocation of resources and task scheduling within MEC systems. However, a commonality among these studies is the treatment of tasks as indivisible units, either offloaded to edge servers or executed locally. This oversimplification often neglects the fact that mobile applications frequently comprise a series of interdependent fine-grained subtasks.

For tasks that are dependent, a subtask must wait for all of its predecessors to complete before it can be executed. Any delay in a subtask may result in an extended makespan for the entire application [13]–[16]. For users, longer execution times lead to greater energy consumption as the device continues to consume CPU cycles and memory while waiting for results. This increased waiting time and energy usage, in turn, can adversely affect the user's QoE [7]–[9]. Recent studies have shifted their focus towards the offloading of such dependent tasks. This approach seeks to harness the innate parallelism within applications, thereby further attenuating the operational overhead borne by mobile devices. For instance, Ding *et al.* [14] proposed an offloading method for resource-limited multi-user and multi-edge settings to minimize execution overhead. Chen *et al.* [15] approached dependent task offloading as a Markov Decision Process, introducing an Actor-Critic method for DAG-based tasks. In a similar vein, an offloading scheme for dependent IoT applications, CODIA [16], combined prioritized scheduling with a reinforcement learning approach. Notwithstanding their merits, a critical assumption underpinning these studies is the benevolent participation of edge servers without any remuneration—an assumption that borders on impracticality. Consequently, there is an impending need for an incentive mechanism to galvanize MEC servers into facilitating offloading services, especially for tasks exhibiting dependent relationships.

Auction theory [17] is an economic field studying the distribution of services or goods via bidding processes. Its

- *Part of this paper was presented in WCNC 2023 [1]. (Corresponding author: Wei Cai.)*
- *Hong Kang, Lehao Lin, Sizheng Fan and Wei Cai are with School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China. E-mail: hongkang1@link.cuhk.edu.cn, lehaolin@link.cuhk.edu.cn, sizhengfan@link.cuhk.edu.cn, and caiwei@cuhk.edu.cn*
- *Minghao Li is with School of Computer Science and Engineering, Nanyang Technological University, Singapore. Email: minghao002@e.ntu.edu.sg*

Corresponding authors: Wei Cai

relevance has grown considerably in spectrum allocation, online advertising, or cloud computing resources, as auctions have become more central in resource allocation within these areas [18]. There are a few reasons auction theory holds significant value in computer science. Initially, auctions can distribute resources in a manner that's equitable and effective. This is critical in computer science because resources like bandwidth, storage space, and computational power are often limited. Secondly, auctions can assist in designing incentive-compatible mechanisms. In other words, participants are motivated to bid truthfully for items, which can yield best or at least not worse utilities.

While prior research has made significant strides in computation offloading and dependent tasks, a conspicuous gap remains: the integration of incentive mechanisms with dependent computation offloading in MEC systems. To bridge this gap, we introduce a novel combinatorial auction-enabled computing service trading mechanism tailored specifically for dependent tasks in MEC environments. This endeavor is underpinned by several intricate challenges that our proposed framework seeks to address: a) While auction theory's benefits in edge computing are recognized [19], [20], it's unclear how to combine this with offloading and scheduling of dependent tasks. Specifically, how do task dependencies affect the pricing and bidding stages in auctions? b) We need a reliable auction mechanism that promotes long-term and fair interactions between mobile devices and MEC servers, preventing negative effects from dishonest bidding. c) The mechanism should benefit both sides: it should reduce the time taken by mobile applications and ensure that MEC servers also gain from the process.

Distinct from single-item auction mechanisms, combinatorial auctions enable bidders to place bids on a combination of items, often referred to as a *'bundle'* [21]. Such an approach is particularly appropriate when the valuation of products by buyers is non-additive, known as complementarity and substitutability [21]. To incorporate dependency considerations into the pricing mechanism, we model the mobile application as a Directed Acyclic Graph (DAG), representing the complementarity and substitutability of subtasks based on their parameters and topological relationships. A multi-stage iterative combinatorial double auction (MICDA) mechanism is proposed to address the issue of component scheduling and resource allocation under the Cloud-Edge-End cooperative computing scenario, with a focus on dependency-aware offloading. This can be viewed as a multi-seller multi-buyer procurement auction, where edge/cloud servers act as service providers (sellers) and users' mobile devices function as service requesters (buyers) for offloading services, extending the scope of our previous work [1]. Network service operators could potentially serve as auctioneers, motivated by the desire to improve the overall benefit of their network service.

The MICDA mechanism is comprised of two stages that are integrated with a concurrent provider and consumer (CPC) model [13] to address the dependency relationships among the subtasks of mobile applications. The mechanism employs value query and demand query to discover values, assign components, and facilitate convergence in iterative auctions. In the first stage, a winner determination algorithm inspired by [22] is utilized to determine the winners, and the market clearing bid density is computed to calculate the payments for mobile devices and the revenues for service providers. The second stage involves computing the fractional optimal solution using linear relaxation and the ellipsoid method [23]. A hypergraphs-based proxy cost model is employed to enable the demand oracle, used in the ellipsoid method, to efficiently query the cost of service providers. This ensures that the winner determination problem in the second stage of MICDA can be solved in polynomial time. The Lavi-Swamy [24] decomposition technique is applied to transform the optimal solution obtained from the linear programming relaxation into a probability distribution over a polynomial-sized support set. Based on this probability distribution, an integer solution is randomly selected. Furthermore, the Vickrey–Clarke–Groves (VCG) payment rule [25]–[27] is adopted in the second stage to calculate the revenue of service providers. Theoretical analysis has demonstrated that the MICDA mechanism achieves individual rationality, truthfulness, and weakly budget balanced. Moreover, the mechanism operates efficiently within polynomial time, reduce the overhead concern.

The main contribution can be concluded below:

1) We propose a multi-stage iterative combinatorial double auction (MICDA) mechanism. This novel approach integrates dependency-aware task offloading with a combinatorial auction mechanism in Cloud-Edge-End cooperation.
2) The proposed mechanism is subjected to thorough theoretical analysis to establish its compliance with essential economic properties. These properties comprise truthfulness, individual rationality, weak budget balance, and polynomial time complexity.
3) We conducted comprehensive experiments to evaluate the effectiveness and performance of the MICDA mechanism. The proposed mechanism demonstrates significant superiority compared to offloading strategies that do not consider the dependency within mobile applications. It also exhibits certain advantages over other dependency-aware algorithms in performance and efficiency.

The remainder of the thesis is organized as follows: Section 2 provides an overview of the relevant literature and highlights the existing gaps. In Section 3, we introduce the associated system architecture and modeling. Section 4 delves into problem formulation. Section 5 presents a detailed overview of the proposed mechanism, along with specifics of each component. Section 6 theoretically validates the anticipated properties of the mechanism. Through simulation experiments, Section 7 validates the performance of the mechanism, comparing it with existing methods. Finally, Section 8 concludes this work.

2 Related Work

This section delves into two primary facets of the field: Dependency-aware Task offloading and Incentive Mechanisms for Mobile Edge Computing.

TABLE 1
Summary of Symbols

Symbol	Description
MD	Set of mobile devices
SP	Set of service providers
$\mathbb{G}$	Set of application DAGs
$\mathcal{G}_i$	Specific application DAG on md_i
$\mathcal{V}_i$	Component/subtask set for application $\mathcal{G}_i$
$v_{i,k}$	Component/subtask in $\mathcal{G}_i$
$\sigma_{i,k}$	Data size of component $v_{i,k}$
$\omega_{i,k}$	Workload of component $v_{i,k}$
$\Theta_{i,k}$	Returned data size for component $v_{i,k}$
$\mathbb{E}$	Set of DAG edges
ι_i^*	Critical path in application DAG $\mathcal{G}_i$
Φ_i^*	Set of provider components
Φ_i	Set of consumer components
$R_{i,j}$	Data rate from md_i to sp_j
B_i, B_j	Bandwidths for mobile devices and edge servers
$\rho_i, h_{i,j}$	Communication power, channel gain
$d_{i,j}$	Distance between md_i and sp_j
a, N_0	Path loss factor and AWGN
R_c	Rate between cloud server and relay edge servers
$\hat{T}_{i,k}^j$	Earliest start time of component $v_{i,k}$
$\tilde{T}_{i,k}^j$	Finish time of component $v_{i,k}$ on sp_j
$t_{i,k}^{j,e}$	Execution time of $v_{i,k}$ on sp_j
$t_{i,k}^{j,j'}$	Return Data transmission time of $v_{i,k}$ from sp_j to sp_j'
$\tilde{t}_{i,k}^{j,tr}$	Transmission time of $v_{i,k}$ itself from sp_j to sp_j'
$E_{i,k}^{j,e}$	Execution Energy cost of $v_{i,k}$ on sp_j
$E_{i,k}^{j,j'}$	Transmission Energy cost of $v_{i,k}$ from sp_j to sp_j'
$E_{i,k}^j$	Total energy for $v_{i,k}$ on sp_j
η_i	Service ratio of md_i
ϑ_i	Valuation of mobile devices
c_i^j	Cost function of service providers
U_i	User utility in the offloading scheme
U_j	Utility for service providers
SW	Social welfare
vd_i	Valuation density of mobile device md_i
cd_i^j	Cost density for md_i provided by sp_j
$x_{j,k}^i$	Decision variable for offloading $v_{i,k}$ of md_i to sp_j
y_{j,S_i^j}	Decision variable for offloading set S_i^j to sp_j
$\mathcal{B}_j$	Bid of service provide sp_j in an auction round
Ω	the total workload of a selected components set

2.1 Dependency-aware Task offloading

The importance of dependency-aware task offloading strategies in mobile edge computing has been increasingly acknowledged by the research community. Noteworthy contributions in this realm include: **Cloud Gaming Platforms:** A decomposed cloud gaming platform was introduced by Cai et al. [28], which supports flexible migrations of gaming components, aiming at cognitive resource management. **Hardware Parameters Optimization:** Ding et al. [14] proposed an offloading strategy to optimize several hardware parameters of user equipment, aiming to minimize execution overheads. **MDP-based Approaches:** Chen et al. [15] modeled the dependent task offloading challenge as a Markov decision process (MDP). They subsequently designed an Actor-Critic mechanism tailored for DAG-based multiple dependent tasks computation offloading. **IoT Applications:** Xiao et al. [16] formulated an intelligent computational offloading scheme for dependent IoT applications (CODIA) with a prioritized scheduling strategy and an AI-based offloading mechanism. **Multi-tier Frameworks:** Shen et al. [29] proposed EdgeMatrix, a multi-tier edge-cloud computing framework. This framework leverages a Networked Multi-agent Actor-Critic algorithm and a multi-task mechanism to efficiently address resource heterogeneity and service orchestration. **Mobility-aware Offloading:** Zhao et al. [30] introduced a vehicular edge computing model to optimize task offloading, reducing response times and energy consumption. Their approach, using deep reinforcement learning and task prioritization, shows improved efficiency and success rates over existing methods.

Despite these advancements, there remains a gap in the literature pertaining to the integration of effective incentive mechanisms in dependency-aware task offloading. The prevailing assumption is that all offloading service providers operate altruistically, which diverges from practical scenarios. Thus, the union of dependency relationships within application components and a fitting incentive mechanism remains an area ripe for exploration.

2.2 Incentive Mechanisms for Mobile Edge Computing

The design of incentive mechanisms for mobile edge computing scenarios, aiming to galvanize edge nodes to deliver superior services, has been extensively studied. Notable methodologies encompass game theory, auction theory, and other mechanisms that facilitate efficient resource allocation and augment social welfare. Key contributions in this domain include: **Game-Theoretic Solutions:** He et al. [31] presented a game-theoretic solution to the edge user allocation problem, emphasizing the Nash equilibrium to optimize system costs and improve performance metrics. Sun et al. [32] developed a hierarchical framework for vehicular edge computing that enhances server resource utilization and vehicle service satisfaction. Their solution, BARGAIN-MATCH, combines bargaining for resource allocation and matching for task offloading, achieving optimal system utility and efficiency, particularly under heavy workloads. **Reinforcement Learning:** A decentralized algorithm for computation offloading, which marries game theory and deep reinforcement learning, was proposed in [33]. This approach has demonstrated superiority over existing methodologies. **Risk-Aware Policies:** Bai et al. [34] introduced a Risk-aware Computation Offloading policy, utilizing a Bayesian Stackelberg game to balance service delays and manage risks. **Auction Mechanisms:** Ma et al. [35] proposed an auction mechanism tailored for industrial IoT in mobile edge computing. Similarly, Zhou et al. [36] presented RACORAM, a reverse auction mechanism for computation offloading. Chen et al. [37] introduced COMSA, which integrates auction mechanisms with network resource allocation to incentivize resource contribution. **Contract Theoretic Approaches:** Diamanti et al. [38] utilized multi-dimensional contract theory for incentive mechanisms, guiding user preferences towards fog layers according to delay tolerance. They utilize Stackelberg game for task offloading and power allocation emphasizes the need to account for user heterogeneity and applications' varied delay sensitivities.

A glaring omission in the aforementioned studies is the consideration of task divisibility and the inherent topological relationships between components. Recognizing and capitalizing on these relationships can significantly enhance

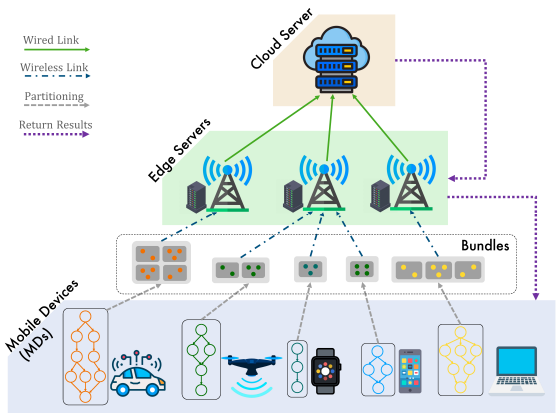

Fig. 1. Framework of the MICDA offloading system for Cloud-Edge-End collaboration, including mobile devices, edge servers, and cloud server.

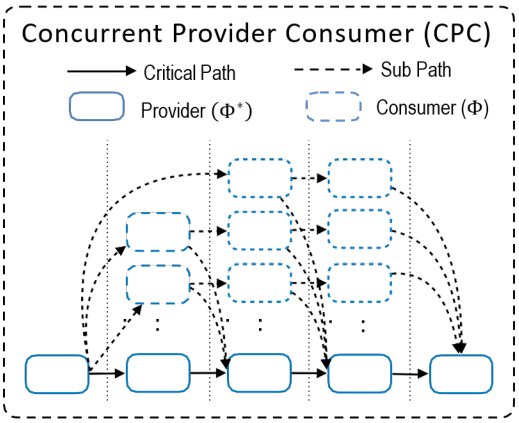

Fig. 2. Indication for Concurrent Provider and Consumer model

application execution and overall system efficiency. This underscores the pressing need to devise incentive mechanisms tailored for dependency-aware offloading within the mobile edge computing framework.

3 System Model

3.1 System Framework for Cloud-Edge-End Scenario

As illustrated in Fig, 1, the system comprises three entities: mobile devices, edge servers, and cloud servers, each situated at different tiers within the system. The lowest tier comprises users' mobile devices, denoted as $MD = \{md_0, md_1, ..., md_m\}$. These devices are required to run various applications for different scenarios. Due to their physical limitations and computational capabilities, it is necessary to offload computations to higher-tier devices (edge/cloud servers) to enhance user experience. Depending on the complexity of user scenarios, the modeling of applications is detailed in the following section. The middle tier represents edge servers, and the top tier is cloud servers, which are concluded as Service Providers $SP = \{sp_0, sp_1, ...sp_n\}$. The edge servers are strategically positioned in proximity to the network edges of base stations, facilitating direct wireless communication with mobile devices. Additionally, the cloud server, referred to as sp_0 in our model, operates as a higher-tier entity characterized by its superior computational capabilities. Considering the heterogeneity among different tiers and devices, each entity in the system has distinct positions, communication bandwidth, communication power, CPU frequency, and the number of CPU cores.

3.2 Application Dependency Model

Consider a system of m mobile devices, each hosting applications with diverse configurations and computational requirements. Each application is represented by a DAG $\mathbb{G} = \{\mathcal{G}_0, \mathcal{G}_1, ..., \mathcal{G}_m\}$, derived from its inherent characteristics and topology. For any application, $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}$, where $\mathcal{V}_i$ is the set of components, considering their dependencies. $v_{i,k}$ is defined by $\{\sigma_{i,k}, \omega_{i,k}, \Theta_{i,k}\}$, with $\sigma_{i,k}$ and $\omega_{i,k}$ signifying the data size and computational workload, respectively. $\Theta_{i,k}$ represents the data size to be relayed to associated components. If there exists a directed edge $e(v_{i,k}, v_{i,k'}) \in \mathcal{E}_i$ between components $v_{i,k}$ and $v_{i,k'}$, then $v_{i,k'}$ is dependent on $v_{i,k}$ and can only commence post the completion of $v_{i,k}$. This relationship is denoted as $pred(v_{i,k'}) = \{v_{i,k} \in \mathcal{V}_i | e(v_{i,k}, v_{i,k'}) \in \mathcal{E}_i\}$ and $succ(v_{i,k}) = \{v_{i,k'} \in \mathcal{V}_i | e(v_{i,k}, v_{i,k'}) \in \mathcal{E}_i\}$. Each $v_{i,k}$ is executed once, either locally or on a remote server. A path of a DAG ($\mathcal{G}_i$) contains a sequence of interdependent components $(e(v_{i,k}, v_{i,k+1}) \in \mathbb{E})$, written as $\lambda = \{v_{i,a}, v_{i,b}, ..., v_{1,z}\}$. The longest path in a DAG is defined as ι_i^*, also nominated as the critical path [13], 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 (ι_i^*). Non-critical components ($v_{i,k} \notin \iota_i^*$) can take advantage of gaps in the execution of critical components and execute in parallel on other places.

Acknowledging the inherent complexity of managing these dependencies within MEC environments, our study leverages the Concurrent Provider and Consumer (CPC) model, as introduced by [13]. The visualization indication is shown in Fig. 2. The critical components are identified as providers (Φ_i^*), who offer the time capacity that non-critical components specified as consumers (Φ_i) 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 the generated providers, the corresponding consumers for each provider is assigned by two principles. First, the associated consumers ($\Phi_{i,l}$) of a provider ($\Phi_{i,l}^*$) must execute simultaneously with it, which means there is no topological dependency (ancestors or descendants) between components in $\Phi_{i,l}$ and $\Phi_{i,l}^*$ [39]. Second, the consumer's execution may postpone the next provider's earliest start time and finish time of the whole application.

By leveraging the CPC model [13], it provides a comprehensive framework to represent the dependency relationships among DAG components, offering detailed insights into both direct and indirect impacts of non-critical nodes on critical path components. This granular understanding is essential for tailoring our offloading mechanisms to accommodate the complex dependency dynamics inherent in mobile applications. Secondly, utilizing the CPC model allows us to elucidate how different offloading strategies in-

fluence the overall application makespan and user utility. By considering the interplay between critical and non-critical components, the model guides the formulation of offloading decisions of different entities in the mechanism.

3.3 Communication Model

In this work, the transmission channel is modeled based on the Rayleigh fading channel [40], considering the densely populated urban areas' effect on radio signals. The bandwidth of orthogonal channels allocated to mobile devices is B_i, and B_j to the edge servers. The data transmission rate from the mobile device md_i for the edge server sp_j is

$$R_{i,j} = B_i \log_2 \left(1 + \frac{\rho_i h_{i,j}}{d_{i,j}^a N_0}\right), \tag{1}$$

where ρ_i represents the transmission power of the mobile device md_i. The parameters $h_{i,j}$ and $d_{i,j}$ are the channel gain of the wireless channel as well as the distance between md_i and sp_j, respectively. The exponent a is a path loss factor, and N_0 shows the additive white Gaussian noise (AWGN). The communication model between MEC servers can be referred to as Eq. (1). We disregard the download time, which is negligible compared to the upload time.

Additionally, mobile devices are required to transfer data to their nearest edge servers (base station), which then forward the data to the cloud through a fiber communication link. The transmission rate between the cloud server and the relay edge servers is assumed to be the same, denoted as R_c.

3.4 Computation model

Because of the dependency among components in the application $\mathcal{G}_i$, all of the immediate predecessors must be finished before the execution of component v_j. Thus, the earliest start time of component v_j's execution be formulated as

$$\hat{T}_{i,k}^{j} = \max_{v_{i,k'} \in \text{pred}(v_{i,k})} \left(\max\left\{\tilde{T}_{i,k'}^{j}, \tilde{T}_{i,k'}^{i}\right\}\right) + \tilde{t}_{i,k}^{j,tr}, \tag{2}$$

where $v_{i,k'}$ is one of the predecessor of $v_{i,k}$. $\tilde{T}_{i,k'}^{j}$ and $\tilde{T}_{i,k'}^{i}$ indicate the finish time of the component $v_{i,k'}$ executed on the service provider sp_j or the mobile device locally. Besides, $\tilde{t}_{i,k}^{j,tr}$ is the completion time of component $v_{i,k}$ transmitted from the mobile device to the service provider sp_j, which is

$$\tilde{t}_{i,k}^{j,tr} = \begin{cases} 0, & \text{if } sp_j = md_i \\ \frac{\sigma_k}{R_{i,j'}} + \frac{\sigma_k}{R_c}, & \text{if } sp_j = sp_0 \\ \frac{\sigma_k}{R_{i,j}}, & \text{otherwise.} \end{cases} \tag{3}$$

When the component $v_{i,k}$ is executed locally, the transmission time of $v_{i,k}$ itself is eliminated. If the component is executed on the cloud, the transmission delay of component $v_{i,k}$ consists of wireless transmission delay between md_i and the earliest edge server $sp_{j'}$ and the fiber transmission delay from $sp_{j'}$ to the cloud.

The CPU frequency of the service provider sp_j is f_j (cycles/second), and the execution time of the component $v_{i,k}$ is

$$t_{i,k}^{j,e} = \frac{\omega_{i,k}}{f_j}. \tag{4}$$

As for the local execution, the f_j can be replaced by the CPU frequency (f_i) of md_i. After the component $v_{i,k}$ is completed, the results should transmit to the position ($sp_{j'}$) where its successors are processed. Based on the communication model in Eq. (1), we have

$$t_{i,k}^{j,j'} = \begin{cases} 0, & \text{if } j' = j \\ \sum_{v_{i,k'} \in \text{succ}(v_{i,k})} \frac{\theta_{kk'}}{R_{j,j'}}, & \text{otherwise} \end{cases} \tag{5}$$

If both the $v_{i,k}$ and its successors are executed on the same entity, the transmission delay $t_{i,k}^{j,j'}$ of $v_{i,k}$'s execution result is zero. Otherwise, the result transmission delay is set to the returned data size divided by the transmission rate. Hence, the finish time of the component $v_{i,k}$ on sp_j is concluded as

$$\tilde{T}_{i,k}^{j} = \hat{T}_{i,k}^{j} + t_{i,k}^{j,e} + t_{i,k}^{j,j'}. \tag{6}$$

Regarding energy consumption, the execution energy cost of component $v_{i,k}$ performed on a service provider sp_j is

$$E_{i,k}^{j,e} = \omega_{i,k} \kappa_j f_j^2, \tag{7}$$

where κ_j [41] is the chip architecture's coefficient factor. After that, the transmission energy cost is calculated as

$$E_{i,k}^{j,j'} = \rho_j \cdot t_{i,k}^{j,j'}, \tag{8}$$

where ρ_j is the transmission power of the service provider. So far, we conclude the total energy consumed for component $v_{i,k}$ finished on the service provider sp_j is

$$E_{i,k}^{j} = E_{i,k}^{j,e} + E_{i,k}^{j,j'}. \tag{9}$$

Similarly, for mobile devices, energy consumption can be referred to the equation mentioned above by substituting the configurations (κ_j, ρ_j) with the attributes related to mobile devices.

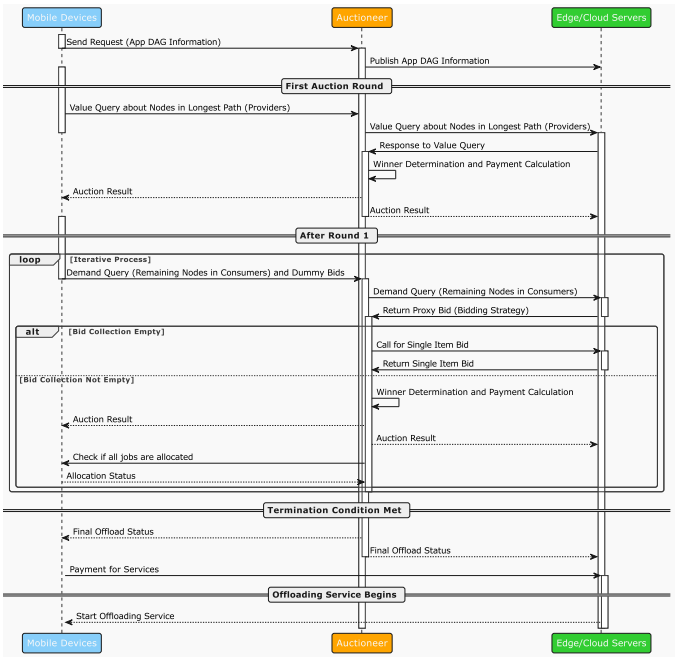

Fig. 3. Sequence diagram of the proposed auction mechanism

4 Problem Formulation

The utility functions for both entities are formulated initially. The Satisfaction level [42], [43] of mobile devices is defined as the overall service time ratio. Let $v_{i,k'}$ be the sink component of $\mathcal{G}_i$. The overall service ratio η_i of md_i is calculated by:

$$\eta_i = \frac{\tilde{T}^i_{i,k'}}{\tilde{T}^*_{i,k'}}, \tag{10}$$

where $\tilde{T}^i_{i,k'}$ and $\tilde{T}^*_{i,k'}$ represent the overall makespan of $\mathcal{G}_i$ executed locally and the value attained by the proposed mechanism, respectively. Furthermore, the valuation of mobile devices is related to their satisfaction level and the energy cost to execute the application locally.

$$\vartheta_i = \eta_i \cdot \sum_{v_{i,k} \in \mathcal{V}_i} E^i_{i,k} \tag{11}$$

Hence, the utility of the user by participating in the offloading scheme is defined as the differences between their valuation and the payment made for the service:

$$U_i = \sum_{j=0}^{n} \sum_{k=0}^{|\mathcal{V}_i|} (\vartheta^j_{i,k} - p^j_{i,k}) x^i_{j,k}, \tag{12}$$

where $x^i_{j,k}$ indicates the decision or allocation status of whether the component $v_{i,k}$ of md_i is offloaded to the service provider sp_j. In addition, $p^j_{i,k}$ represents the determined payment value the mobile device pays to service provider sp_j from md_i for processing the component $v_{i,k}$.

As for offloading services providers, including the mobile edge servers and the cloud servers, their utility is directly correlated to the profit made through the proposed mechanism, which can be defined as the revenue they received from the mobile devices minus the cost of providing offloading services.

$$U_j = \sum_{i=0}^{m} \sum_{k=0}^{|\mathcal{V}_i|} (r^j_{i,k} - c^j_{i,k}) x^i_{j,k} \tag{13}$$

Here, the cost ($c^j_{i,k}$) is equivalent to the energy expenditure $E^j_{i,k}$. So far, we can conclude the objective of this model is to maximize social welfare (SW) by taking into account the utilities of both parties in the auction.

$$\text{SW}: \quad \sum_{i=0}^{m} U_i + \sum_{j=0}^{n} U_j \tag{14}$$

By substituting Eq. (13) and Eq. (12), we can derive the objective function with decision at component level:

$$\begin{aligned}
\max \quad & \sum_{j=0}^{n} \sum_{i=0}^{m} \sum_{k=0}^{|\mathcal{V}_i|} (\vartheta^j_{i,k} + r^j_{i,k} - p^j_{i,k} - c^j_{i,k}) x^i_{j,k} \\
\text{s.t.} \quad & \text{C1}: \tilde{T}^j_{i,k} \le \hat{T}^j_{i,k'}, \forall e(v_{i,k}, v_{i,k'}) \in \mathcal{E}_i, \\
& \text{C2}: \tilde{T}^j_{i,k} \cdot x^i_{j,k} \le \tilde{T}^l_{i,k}, \forall md_i \in MD, \forall v_{i,k} \in \mathcal{V}_i, \\
& \text{C3}: \sum_{j=0}^{n} \sum_{k=0}^{|\mathcal{V}_i|} x^i_{j,k} \le 1, \forall md_i \in MD, \\
& \text{C4}: \sum_{j=0}^{n} \sum_{j=0}^{m} x^i_{j,k} \le |\mathcal{V}_i|, \forall md_i \in MD, \\
& \text{C5}: x^i_{j,k} \in \{0,1\}, \forall md_i \in MD, \forall v_{i,k} \in \mathcal{V}_i.
\end{aligned} \tag{15}$$

Here, C1 exhibits the dependency among application components, asserting that every component $v_{i,k}$ only begins its execution when predecessors are finished. C2 is a time constraint ensuring that the edge execution time of the component matches or surpasses the local execution of md_i, which allows mobile devices to keep the computation locally. Constraints C3, C4 and C5 ensure a component $v_{i,k}$ is offloaded to only one service provider or executed locally.

Algorithm 1: MICDA Mechanism Overview

Input: Set of mobile devices MD, set of service providers SP, Set of mobile applications $\mathbb{G}$
Output: Allocation $\mathbf{y}$, Payment p_i, Revenue r_j

Initialization:
Broadcast requests to SP
Public DAG information
while *Not all components auctioned* **do**
  if *Initial Round* **then**
    `// Stage I:`
    MDs send value queries for components in all provider groups Φ^*
    Auctioneer broadcasts value queries to SPs
    SPs respond with bids $\mathcal{B}_j(S_i)$
    Auctioneer runs χ_1 to determine winner
    Auctioneer calculates payment by Algorithm 4
  else
    `// Stage II:`
    while $\exists md_i \in MD, M_i \neq \emptyset$ **do**
      MDs send demand queries for M_i to the auctioneer
      Auctioneer broadcasts to SPs
      SPs construct bundles S^j_i and submit their bids by Algorithm 2
      Auctioneer runs χ_2 to determine winner
      Auctioneer calculates payments and revenues by Algorithm 4
    end
  end
end

5 Auction Mechanism

5.1 The Framework of MICDA

The proposed MICDA mechanism consists of two stages, the sequence diagram of the whole process of the mech-

anism is shown in Fig. 3. Iterative combinatorial auctions utilize two types of queries to reveal the values and preferences of bidders, namely demand queries and value queries [18]. Both of them will be detailed later, and different query methods will be used at different stages of MICDA.

Within the proposed mechanism, network service operators act as auctioneers, aiming to enhance the overall benefit of their network service. They orchestrate the auction, starting with the preparation phase where mobile devices submit offloading requests, including information about the applications to be offloaded and geographical location. The auctioneer then auctioneer collects all requests from mobile devices and broadcasts them to edge or cloud servers. As auctioneers, they are pivotal in aggregating demand and supply, thereby optimizing the outcome of the mechanism.

In the initial auction round (Stage I), mobile devices send value queries about all components in providers to the auctioneer. These components are the path with the maximum workload in the mobile application and represent the theoretically shortest execution time of the application. Therefore, these components have the highest priority to be determined for mobile users and will also contribute to the advancement of subsequent auctions. It is worth noting that to ensure the properties of individual rationality (non-negative profits) and convergence, mobile devices are allowed to participate in bidding (dummy bids) and become winners. The auctioneer collects all mobile device value queries and sends a summary to service providers. Service providers send bids to the auctioneer based on the components in the query. The auctioneer selects the winner based on the received bids and determines the payment price, broadcasting the auction results to all participants.

After the initial round, the auction enters the second stage. At this stage, mobile devices send demand queries to the auctioneer at the beginning of the round, i.e., all remaining nodes that have not been auctioned off temporarily and their estimated values for these components. The auctioneer sends these demand queries to service providers. When service providers receive these queries, they construct a bundle with the maximum value under the constraints according to the bidding strategy and send the proxy valuation for these components. After receiving all bids, the auctioneer selects the winner and calculates the price based on payment rules.

The termination condition of the auction is that all components of all mobile devices have been auctioned off, regardless of whether the winner is the mobile device itself or a cloud or edge server. The specific bidding strategy, winner determination, payment rule, and other steps in the mechanism will be introduced later.

5.2 Desired Property

MICDA is desired the following economic properties:

- **Individual Rationality:** For all entities participating in the proposed mechanism, their utility should be non-negative, representing $U_i, U_j \geq 0$.
- **Truthfulness (Truthful in expectation)** A mechanism is truthful in expectation if each agent maximizes their expected benefit by reporting their true value/cost, regardless of other agents' report.
- **(Weak) Budget Balance:** The payments from all buyers (MD) are greater than or equal to the returns of all sellers (SP), meaning the auctioneer does not need to subsidize the operation of the mechanism with additional funds. This Weak Budget Balance ensures the sustainability of MICDA.
- **Computation Efficiency:** The proposed mechanism can be completed in polynomial time.

5.3 Query

In this section, the two ways of queries [18] used in the MICDA mechanism are shown:

5.3.1 Value query

A value query asks a service provider sp_j to report its cost for providing a specific bundle of items or services. In mathematical terms, the bundle of components that a mobile device md_i desires to offload are S_i with local computation cost ($c_i^j(S_i)$) and makespan ($\tilde{T}_i(S_i)$). Then a value query to the sp_j is asking for the value of $c_i^j(S_i)$ and $\tilde{T}_i^j(S_i)$.

5.3.2 Demand query

A general demand query asks a service provider sp_j to report the bundle of items or services they are willing to provide given a certain price vector. A price vector assigns a price to each item in the auction. In mathematical terms, the price vector p_i represents the valuation of the mobile device md_i on each component in a ground set M_i. Then a demanding query to sp_j is asking for the set S_i^j that maximizes $p_i(S_i^j) - c_i^j(S_i^j)$, which is the revenue from the bundle S_i^j at prices $p_i(S_i^j)$. Considering the characteristics of our problem, a completion time constraint is adopted to regulate the finish time of each component in the bundle can not exceed the finish time of its' corresponding providers.

5.4 Hypergraph-based proxy cost model

Recalling the definition of complementarity between items in combinatorial auction [1], [18], the complementarity between subtasks of a DAG can be concluded. If a subtask and its predecessor are handled by the same service provider, the communication cost can be eliminated. Therefore, the total benefit of the service provider handling the component will increase, which is greater than handling either of the two components separately. A cost function based on hypergraphs [23] is used to model the complementarity between components.

Assume that in each round, the set of subtasks/components in the demand query initiated by the mobile device md_i is M_i. A service provider sp_j constructs a bundle of components $S_i^j \in M_i$ following the time-constrained demand query detailed in Section 5.5.1. The service provider sp_j can use a hypergraph-r cost model $H_i^j = (S_i^j, \bar{\mathcal{E}}_i^j)$ to represent the cost of any subset of components from S_i^j. Besides, $\bar{e} \subseteq \bar{\mathcal{E}}_i^j$ contains most r components, which are independent of each other. The weight w_e represents the communication cost to deliver their execution result. What's more. a component $v_{i,k}$ in S_i^j

has a corresponding execution cost w_k. Then, for a subset of goods $s \subseteq S_i^j$, the cost is derived by

$$c_i^j(s) = \sum_{v_{i,k} \in s} w_k + \sum_{\bar{e}:\bar{e} \subseteq s} w_{\bar{e}}. \tag{16}$$

This function represents the total cost (including execution cost and communication cost) if he wins all components in s. The parameter r limits the maximum number of vertices each edge in $\bar{\mathcal{E}}_i^j$ can contain. In this way, the cost can be described with $\mathcal{O}(m^r)$ parameters.

The reason for using a hypergraph-based proxy cost model in this scenario is that the complementarity between components can be represented by the weight of the edges (i.e., communication cost is 0), and the number and size of the edges can be limited by the parameter r. In this way, a concise model can be used to describe limited complementarity without considering all possible combinations of application components of mobile devices. It can also support efficient queries, which are beneficial for both auctioneers and bidders. Using the hypergraph cost model, auctioneers can obtain the cost information of SP by directly interacting with this proxy model without busy querying with the bidders. This can reduce the number and complexity of information exchanges.

5.5 Bidding Strategy

5.5.1 Bundle Construction

In the initial round of the auction, we elaborated that the execution time of components on the critical path (ι_i^*), also known as providers (Φ_i^*) in the CPC model, representing the ideal makespan of the whole application. Therefore, components in providers have the highest priority for mobile devices being auctioned. Also, the service providers adopt a myopic best-response bidding strategy for the whole mechanism, which means they 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 service providers should bid for bundles of all critical components (providers) in the first round. At this juncture, the interaction between mobile devices and service providers can be conducted in the form of a value query, as the demand and supply are aligned. Furthermore, there is no need to consider the internal topological relationships and latency of the components.

After the initial round, due to the heterogeneity and limited resources of service provider devices, the bundles in the service provider's bid cannot encompass all components within the entire consumer set (components in providers are auctioned in the first round). Additionally, the submitted bundles must take into account the dependency nature of different components and meet latency constraints. As a result, we employ a 'demand query' to represent the auction interaction in the second phase. The selection and construction of the bundle are formulated as a 0-1 knapsack problem for each consumer set $\Phi_{i,l}$. Concurrently, the corresponding cost is calculated and incorporated into the Hypergraph-based proxy cost model $H_i^j = (S_i^j, \bar{\mathcal{E}}_i^j)$ which is then submitted to the auctioneer. The specific bundle construction process is as follows:

Algorithm 2: Bidding Strategy

Input : Demand query M_i from MD
Output: Bid $\mathcal{B}_j$

Initialize $\mathcal{B}_j = \emptyset$;
if *auction round == 0* **then**
  for *each* $md_i \in MD$ **do**
    $S_i^j = \Phi_i^*$, the critical path providers;
    $\mathcal{B}_j = \mathcal{B}_j \oplus (S_i^j, c_i^j(S_i^j))$;
  end
else
  if *Additional case* **then**
    $S_i^j = max\{\omega_{i,k} | v_{i,k} \in S_i\}$;
    $\mathcal{B}_j = \mathcal{B}_j \oplus (S_i^j, c_i^j(S_i^j))$;
  else
    for *each* $md_i \in MD$ **do**
      foreach $\Phi_{i,l}$ *in* Φ_i *(reverse order)* **do**
        foreach *component* $v_{i,k} \in \Phi_{i,l}$ **do**
          Calculate value density $\frac{E_{i,k}^j}{\tilde{t}_{i,k}^{j,tr} + t_{i,k}^{j,e} + t_{i,k}^{j,j'}}$;
        end
        Sort components by value density in descending order;
        Initialize empty bundle $s_{i,l}^j$;
        foreach *component* $v_{i,k}$ *in sorted order* **do**
          if *adding* $v_{i,k}$ *to* $s_{i,l}^j$ *does not exceed* $\tau_{i,l}$ **then**
            Add $v_{i,k}$ to $s_{i,l}^j$;
          end
        end
        Concatenate $s_{i,l}^j$ to S_i^j;
      end
      Construct $H_i^j = (S_i^j, \bar{\mathcal{E}}_i^j)$;
      $\mathcal{B}_j = \mathcal{B}_j \oplus H_i^j$;
    end

In each $\Phi_{i,l}$, 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.

Hence, we should limit the total execution time of the selected components in each $\Phi_{i,l}$ within the execution time ($\tilde{\tau}_{i,l}^w$) of all components of its provider, indicated in constraint (C6). After denoting the winner of the first round of md_i's service request as sp_w, we have $\tilde{\tau}_{i,l}^w = \sum_{v_{i,k} \in \Phi_{i,l}^*} \tilde{t}_{i,k}^{w,tr} + + t_{i,k}^{w,e} + t_{i,k}^{w,w'}$. Then, the bundle construction in Φ_k can be represented as:

$$\begin{aligned} \max \quad & \sum_{v_{i,k} \in \Phi_{i,l}} c_{i,k}^j \cdot x_{j,k}^i \\ \text{s.t.} \quad & \text{C6}: \sum_{v_{i,k} \in s_{i,l}^j} (\tilde{t}_{i,k}^{j,tr} + + t_{i,k}^{j,e} + t_{i,k}^{j,j'}) x_{j,k}^i \leq \tilde{\tau}_{i,l}^w, \\ & \text{C7}: x_{j,k}^i \in \{0,1\}, \forall v_{i,k} \in \Phi_{i,l}, \forall md_i \in MD \end{aligned} \tag{17}$$

Due to the NP-hardness of the problem mentioned

above, we use a greedy algorithm for handling it in polynomial time. The components in consumer $\Phi_{i,l}$ are sorted by their value density $\frac{E^j_{i,k}}{\tilde{t}^{j,tr}_{i,k}+t^{j,e}_{i,k}+t^{j,j'}_{i,k}}$. The components with higher value density have higher priority to add to the bundle until the total time exceeds $\tau_{i,l}$.

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. The set of components selected in the lth consumer set of service provider sp_j is $s^j_{i,l}$, and the corresponding service cost is written as $c^j_i(s^j_{i,l})$. Additionally, the complete bundle submitted by sp_j in the current auction round is the concatenation of the results from all consumer sets:

$$S^j_i = s^j_{i,1} \cup s^j_{i,2} \cup ... \cup s^j_{i,l} \tag{18}$$

After obtaining the complete bundle and its corresponding cost, the service provider sp_j can proceed to construct the hypergraph-based proxy cost model $H^j_i = (S^j_i, \bar{\mathcal{E}}^j_i)$ as illustrated in Section 5.4.

5.5.2 Bidding language

To enhance the efficiency and flexibility of combinatorial auctions, bidding languages can be employed [17], [18]. In terms of efficiency, bidding languages can help buyers express their valuations for different combinations of goods more accurately, thereby improving social welfare and the quality of resource allocation. In terms of flexibility, bidding languages can make it easier for buyers to handle complex constraints and preferences. Therefore, service providers can simultaneously use exclusive-OR (XOR) bidding language to express bids $\mathcal{B}_j$ for different mobile devices.

$$\mathcal{B}_j = (S^j_1, \bar{\mathcal{E}}^j_1) \oplus ... \oplus (S^j_i, \bar{\mathcal{E}}^j_i) \tag{19}$$

XOR ($\oplus$) bids allow bidders to offer different prices for different combinations of components from different mobile devices and are required to get one of them.

5.5.3 Additional case

Considering that if the computation resources in the whole network are under demand, constraint C6 in Eq. (17) is not met, which leads to the collection of bids from service providers becoming an empty set. To ensure the convergence of the mechanism, the auctioneer calls a single-item bid that can ignore constraint C6. According to the myopic best-response bidding strategy, the services provider will bid for the most valuable components $S^j_i = \max\{\omega_{i,k} | v_{i,k} \in S_i\}$ in the remaining consumer nodes of a mobile device.

5.6 Winner Determination

5.6.1 Winner Determination for Stage I

The winner determination algorithm ($\boldsymbol{\chi}$) in the MICDA mechanism can be divided into two categories, shown as Algorithm 3. In the first category (χ_1), which is applied for the initial stage of the auction and the additional case introduced in Section 5.5.3. In the former case, the valuation query requires SP to return their bid prices for all subtasks in providers of each mobile device. Therefore, all bundles from each SP are identical, so we can treat them as a single-item auction. In the latter case, all SP is required to submit a single-item bid in order to overcome the convergence problem when the computation resources are under demand. Thus, they can be tackled in a same way.

Algorithm 3: Winner Determination Algorithms

Input : Bids from MD and SP
Output: Allocation $\mathbf{y}$

```
if χ1 then
    Calculate vd_i for each md_i ∈ MD by Eq. 20 ;
    Calculate cd_i^j for each sp_j by Eq21 ;
    Sort vd_i in decreasing order;
    Sort cd_i^j in increasing order;
    Accept pairs until vd_(k') ≥ cd_(k') · η_(k');
else if χ2 then
    Calculate vd_i by Eq. 20;
    Rank vd_i in descending order by Eq. 22;
    for each md_i do
        Linear relaxation and formulate primal
         problem by Eq. 23;
        Define dual problem by Eq. 24;
        Solve dual problem by Ellipsoid method y* ;
        Decompose fractional solution ;
        Random rounding to get y;
    end
```

In this section, the winner determination problem (χ_1) of the initial stage and the additional case is elaborated and solved. The current active mobile devices set is denoted as MD^*. After receiving bids containing the bundle S_i from the mobile devices, the valuation density of active mobile devices vd_i is calculated firstly by

$$vd_i = \frac{\vartheta_i(S_i)}{\sqrt{\Omega_i}}, \tag{20}$$

where $\Omega_i = \sum_{v_{i,k} \in S_i} \omega_k$. As for the service provider sp_j, the cost density for md_i is defined as:

$$cd^j_i = \frac{c^j_i(S_i)}{\sqrt{\Omega_i}} \cdot \frac{1}{\eta_i}. \tag{21}$$

The effective cost density of sp_j is defined as $cd^j_i = \max\{\, cd^j_i \mid md_i \in MD^* \,\}$ After that, the auctioneer sorts the valuation density of $MD*$ in decreasing order and the effective cost density of SP in an increasing order, accepting the pair of buyer and seller in sequence until the last pair $(vd_{(k')}, cd_{(k')})$ has $vd_{(k')} \geq cd_{(k')} \cdot \eta_{k'}$. It should be noted that the k' is the rank, not the actual id. All mobile devices that are rejected by χ_1 keep the components (S_i) locally.

5.6.2 Winner Determination for Stage II

In this section, the winner determination algorithm (χ_2) for the second stage of the MICDA mechanism is illustrated. After collecting the demand query from MD and receiving the bids of each SP, the auctioneer calculates the total valuation density (vd_i) of the demand query from each mobile device by Eq. (20). Subsequently, the auctioneer will rank the total valuation density (vd_i) of all mobile devices in descending order, from the highest to the lowest.

$$\frac{\vartheta_1(M_1)}{\sqrt{\Omega_1}} \geq \frac{\vartheta_2(M_2)}{\sqrt{\Omega_2}} \geq ... \geq \frac{\vartheta_n(M_n)}{\sqrt{\Omega_n}} \tag{22}$$

Then, a greedy approach is employed to select the highest-ranking candidate as the recipient for choosing a service provider. Based on the aforementioned bidding strategy of service providers, each service provider will bid to multiple mobile devices. And since the components of each mobile device are heterogeneous and unrelated, winners will be selected for them in sequence. Due to the XOR bidding strategy, none of their other bids will be accepted once a sp_j becomes a provisional winner in a auction round.

5.6.3 Linear Relaxation

An allocation rule called Maximal-In-Distributional-Range (MIDR) [44] is implemented for the second winner determination algorithm (χ_2) of the proposed mechanism. The idea behind this rule is to pre-determine a distribution of feasible allocations before receiving the reported valuations from players. Then, based on the reported valuations, the distribution with the maximum expected social welfare is selected. Finally, a feasible allocation is randomly drawn from the selected distribution. This rule can induce players to report their valuations truthfully. It can form a truthful expectation mechanism if combined with a VCG [25]–[27] type of payment.

As for a DAG-based application, the overall makespan depends on the longest path's finish time. Considering that all bundles submitted through the time-constrained demand query ellaborated in Section 5.5.1 will not cause extra makespan of the application. Therefore, the valuation of the mobile device in stage II can be regarded as a constant, the social welfare problem is about minimizing the cost of the offloading services of the mobile devices. We transition from a component-level analysis to a bundle-level abstraction. This shift aligns with the combinatorial auction setting. By applying linear relaxation to the social welfare maximization objective function shown in Eq. (15), we achieve a streamlined formulation. Consequently, the winner determination linear programming (LP) problem for a mobile device md_i with remaining unallocated components M_i in an auction round can be formulated as:

$$\begin{aligned} \min \quad & \sum_{j,S_i^j} y_{j,S_i^j}(c_i^j(S_i^j)) \\ \text{s.t.} \quad & \text{C8}: \sum_{j} \sum_{v_{i,k}\in S_i^j} y_{j,S_i^j} \le 1, \quad \forall S_i^j \subseteq M_i \\ & \text{C9}: \sum_{S_i^j} y_{j,S_i^j} \le 1, \quad \forall sp_j \in SP \\ & \text{C10}: y_{j,S_i^j} \ge 0, \quad \forall sp_j \in SP, S_i^j \subseteq M_i \end{aligned} \tag{23}$$

The dual linear program of the primary problem can be defined, acknowledging that the primary linear program is a relaxation from constraint C10. However, there's an issue: the linear program has an exponential number of variables, making it impossible to solve in polynomial time. On the other hand, the constraints are relatively few, hinting that its dual problem has fewer variables but exponentially many constraints. This setup suggests that the dual problem might be more manageable despite its complexity. The formulation of the dual problem is as follows:

$$\begin{aligned} \max \quad & \sum_{j} u_j + \sum_{k} p_k \\ \text{s.t.} \quad & \text{C11}: \sum_{v_{i,k}\in S_i^j} p_k - c_i^j(S_i^j) \ge u_j, \quad \forall j, S_i^j \\ & \text{C12}: u_j, p_k \ge 0 \end{aligned} \tag{24}$$

Recognizing that the dual problem possesses exactly $n+m$ variables, but an exponential number of constraints. A potential approach to solve this problem could involve the ellipsoid method, which can solve this problem in polynomial time. Nevertheless, this doth rely upon a separation oracle capable of operating within polynomial time. Here a demand oracle is played as the separation oracle for a cost $c_i^j(S_i^j)$: given a price p_k for each subtask $v_{i,k} \in M_i$, compute a bundle S_i^j by

$$S_i^j \in \arg\max \left\{ \sum_{v_{i,k}\in S_i^j} p_k - c_i^j(S_i^j) \right\}. \tag{25}$$

The auctioneer can calculate such operation in polynomial time by the hypergraph-based proxy cost model introduced in section 5.4 [23]. After solving the dual problem, the solution of the primal problem(Eq. (23)) can be calculated by making use of the complementary slackness [45].

5.6.4 Decomposing the fractional solution

By the dualization and the ellipsoid algorithm with the given demand oracle, the fractional solution of Eq. (23) can be resolved. To achieve a MIDR mechanism, we use a technique called Lavi-Swarmy reduction [24]. This approach allows for the computation of distribution over integer solutions with the support of polynomial size, achievable in polynomial time. By obtaining the optimal fractional solution y^*, we can execute a decomposition of the vector $\frac{y^*}{\alpha}$ into a convex combination of feasible integral solutions. This can be represented as $\frac{y^*}{\alpha} = \sum_{l\in\mathcal{I}} \lambda_l \mathbf{y}_\ell$, where the sum is over a polynomial number of feasible integral solutions. After getting the probability λ_l, a random rounding procedure is taken to get the integral solution ($\mathbf{y}$) of Eq. (23). The solution ($\mathbf{y}$) is the result of the selected winners and allocation results of χ_2 in each auction round.

5.7 Payment Rule

5.7.1 Payment Rule for Stage I

As for the winner determination rule (χ_1) in Section 5.6.1, the final pair of $vd_{(k')} \ge cd_{(k')}$ is accepted. Additionally, the clearing price density inspired from [22] is defined as

$$pd = \frac{vd_{(k'+1)} + \eta_{(k'+1)} \cdot cd_{(k'+1)}}{2} \tag{26}$$

Thus, for an accepted pair of md_i and sp_j. The payment of md_i should be made is calculated as

$$p_i = \begin{cases} pd \cdot \sqrt{\Omega_i}, & \text{if} \quad vd_{(k)} \ge pd \ge cd_{(k)} \cdot \eta_{(k)} \\ vd_{(k)} \cdot \sqrt{\Omega_i}, & \text{otherwise} \end{cases} \tag{27}$$

Algorithm 4: Payment Rules

Input : Allocation $\mathbf{y}$, bids $\mathcal{B}_j$
Output: Payments p_i, r_j
if *Stage I* **then**
  Calculate pd by Eq. 26;
  for *each accepted MD i, SP j pair* **do**
    Calculate p_i by Eq. 27;
    Calculate r_j by Eq. 28;
  end
else if *Stage II* **then**
  for *each winner SP j* **do**
    Calculate fractional payment r_j^* by Eq. 31;
    Decompose r_j^* to get r_j
  end
  for *each winner MD i* **do**
    Calculate p_i by Eq. 34;
  end

The revenue of sp_j is also derived based on the clearing price density by

$$r_j = \begin{cases} pd \cdot \sqrt{\Omega_i}, & \text{if } vd_{(k)} \geq pd \geq cd_{(k)} \cdot \eta_{(k)} \\ cd_{(k)} \cdot \sqrt{\Omega_i} \cdot \eta_i, & \text{otherwise} \end{cases} \tag{28}$$

For other entities who participate but not be accepted by the winner determination rule have zero revenue or payment.

5.7.2 Payment rule for Stage II

In the second stage of the proposed MICDA mechanism, VCG payment rule [25]–[27] is utilized to calculate the revenue of SP. Consider a scenario where the service provider, denoted as sp_j, does not participate in Stage II of MICDA mechanism. In such a case, SW achieved in each auction round depends on the total cost of winning SP, which is formulated as:

$$\min_{S_i^{j'} \subseteq M_i} \sum_{j' \neq j} c_i^{j'}(S_i^{j'}). \tag{29}$$

When sp_j is included in the auction, the social welfare generated by the set of service providers excluding sp_j may decrease compared to the case when sp_j was absent. The total cost in this situation is given by:

$$S_i^{j*} \in \arg\min_{S_i^j \subseteq M_i} \sum_j c_i^j(S_i^j). \tag{30}$$

The disparity in social welfare generated for the other service providers when sp_j is absent versus present is:

$$r_j^* = \min_{S_i^{j'} \subseteq M_i} \sum_{j' \neq j} c_i^{j'}(S_i^{j'}) - \sum_{j' \neq j} c_i^j(S_i^{j*}), \tag{31}$$

where the valuation $\vartheta_i(S_i^j)$ of md_i is constant and canceled by subtraction. This amount is precisely the payment that the VCG mechanism (procurement version) paid for sp_j. Thus, one can interpret the sp_j's revenue as the collective externality they impose on the other bidders.

Additionally, the payment rule for the linear relaxation and decomposition-based winner determination requires additional procedures to ensure truthfulness. Suppose the optimal fractional solution of Eq. (23) is y^*. By the VCG payment rule we elaborated above, we can calculate a fractional payment r_j^*. The final payment is obtained by dividing the integrality gap α of the decomposition:

$$r_j = \frac{r_j^*}{\alpha} \tag{32}$$

When it comes to the payment for mobile devices (MDs), we adopt the concept of critical payment as outlined in [21]. The current provisional candidate (md_i^*), which also represents the MD with the highest rank according to Eq. (22), is denoted as having a value density of vd_i^*. The MD with the second-highest rank, excluding md_i^*, is labeled as $md_{i^-}^*$. With this in mind, the critical payment (p_i^*) for md_i^* can be calculated as follows:

$$p_i^* = vd_{i^-}^* \cdot \sqrt{\Omega_i} \tag{33}$$

Furthermore, taking into account the dummy bids of the mobile devices and the budget balance of MICDA, the final payment for each md_i can be derived by

$$p_i = \max(p_i^*, r_j). \tag{34}$$

6 Theorical Analysis of MICDA mechanism

6.1 Truthfulness

Lemma 1. *Stage I of MICDA mechanism is truthful.*

Proof. The truthfulness of mobile devices is established first. Let md_i denote a mobile device with a set of components M_i. The true value of md_i is denoted by $\vartheta_i(M_i)$ and the corresponding value density by vd_i. The untruthful bid price of md_i is denoted as $b_i(M_i)$ and the bid density bd_i.

Untruthful bidding does not yield additional utility to mobile devices in two distinct scenarios:

Case 1: $bd_i < vd_{(k)}$: Here, md_i is not accepted by the auctioneer, resulting in a utility $U_i = 0$. If $vd_i \leq vd_{(k)}$, then the truthful bid also results in zero utility. However, if $vd_i > vd_{(k)}$, the utility of truthful bidding is $u_i = \vartheta_i(M_i) - \min(pd, vd_{(k)}) \cdot \sqrt{\Omega} \geq 0$. Hence, untruthful bidding reduces the utility of md_i.

Case 2: $bd_i \geq vd_{(k)}$: In this case, md_i is accepted for trade, and the utility is $U_i = \vartheta_i(M_i) - \min(pd, vd_{(k)}) \cdot \sqrt{\Omega}$. If $vd_i \geq vd_{(k)}$, untruthful bidding achieves the same utility as truthful bidding. However, if $vd_i < vd_{(k)}$, then $U_i < 0$ and untruthful bidding reduces the utility of md_i.

From these cases, it is concluded that untruthful bidding does not provide additional utility for mobile devices, and truthful valuation is the dominant strategy.

Next, the truthfulness of service providers in Stage I of the MICDA mechanism is proven. Let sp_j be a service provider with a bundle of components $S_i \subseteq M_i$. The true cost for a mobile device md_i is $c_j^i(S_i)$. The untruthful ask price for md_i for S_i is denoted as a_i^j, and the untruthful ask price density as ad_i^j. Truthfulness is proven for each case of auction results:

Case 1: $ad_i^j > cd_{(k)}$: In this case, sp_j is not accepted by the auctioneer, so $U_j = 0$. If $cd_i^j > cd_{(k)}$, truthful bidding is still not accepted. However, if $cd_i^j \leq cd_{(k)}$, then $U_j = [\max(pd, vd_{(k)}) - cd_i^j] \cdot \sqrt{\Omega} \cdot \eta_i \geq 0$. Hence, untruthful bidding reduces the utility of sp_j.

Case 2: $ad_i^j \leq cd_{(k)}$: Here, sp_j is accepted by the auctioneer and provides service. If $cd_i^j \leq cd_{(k)}$, then sp_j is still accepted when bidding truthfully. However, if $cd_i^j > cd_{(k)}$, then the utility of untruthful bidding is $U_j = [\max(pd, vd_{(k)}) - cd_i^j] \cdot \sqrt{\Omega} \cdot \eta_i < 0$. Hence, untruthful bidding reduces the utility of sp_j.

From these cases, it is concluded that untruthful bidding does not provide additional utility for the service provider in Stage I of MICDA. In conclusion, it is proven that both entities, the mobile devices, and the service providers, are incentivized to report truthful valuation or cost in Stage I of the MICDA mechanism.

□

Lemma 2. *Stage II of MICDA mechanism is truthful.*

Proof. Consider a mobile device, denoted as md_i, with a true valuation of the remaining components set md_i as $\vartheta_i(M_i)$. If md_i bid as $\vartheta_{i'}(M_{i'}) \leq \vartheta_i(M_i)$, then the calculated value density $vd_{i'} \leq vd_i$, which leads to a lower rank in the sorted list (Eq. (22)). Hence, the ranking algorithm for mobile devices in the winner determination χ_2 of Stage II is monotone. Additionally, the critical payment rule, as introduced in Section 5.7.2, is adopted. Therefore, the Stage II of MICDA is proven to be truthful for mobile devices [21].

For the service providers, the expected revenue of sp_j is given by $\mathbb{E}_{\lambda_l}[r_j^*] = \frac{r_j^*}{\alpha}$. This can be expanded as:

$$\begin{aligned}\mathbb{E}_{\{\lambda_l\}_{l\in\mathcal{I}}}\left[r_j^*\right] &= \sum_{l\in\mathcal{I}} \lambda_l \cdot \left[c_i^j\left(z^l\right)/c_i^j\left(y^*\right)\right] \cdot r_j^* \\ &= \left[r_j^*/c_i^j\left(y^*\right)\right] \cdot \sum_{l\in\mathcal{I}} \lambda_l \cdot c_i^j\left(z^l\right) \\ &= \left[r_j^*/c_i^j\left(y^*\right)\right] \cdot c_i^j\left(x^*/\alpha\right) = r_j^*/\alpha\end{aligned} \tag{35}$$

Suppose sp_j reports an untruthful cost $\_c_i^j$, resulting in the allocation in LP 23 as $\_y^*$. Given that the VCG payment is applied and the fractional revenue is truthful [18], it follows that:

$$r_j^*(c_i^j, c_i^{-j}) - c_i^j(y^*) \geq r_j^*(\_c_i^j, c_i^{-j}) - \_c_i^j(\_y^*) \tag{36}$$

Combining Equations 35 and 36, it is obtained that:

$$\begin{aligned}&\left[\sum_{l\in\mathcal{I}} \lambda_l \cdot c_i^j\left(y^*\right)\right] - \mathbb{E}_{\{\lambda_l\}_{l\in\mathcal{I}}}\left[r_j^*(c_i^j, c_i^{-j})\right] \geq \\ &\left[\sum_{l\in\mathcal{I}} \lambda_l \cdot c_i^j\left(\_y^*\right)\right] - \mathbb{E}_{\{\lambda_l\}_{l\in\mathcal{I}}}\left[r_j^*(\_c_i^j, c_i^{-j})\right]\end{aligned} \tag{37}$$

The left side represents the utility obtained by sp_j when a truthful cost is reported, while the right side represents the utility of untruthful bidding. Thus, the truthfulness of service providers is proven, and the lemma follows. □

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

Proof. By invoking Lemma 1 and Lemma 2, the truthfulness of the proposed MICDA mechanism in Stage I and Stage II has been established respectively. Therefore, it can be concluded that the MICDA mechanism is truthful.

□

6.2 Individual Rational

Lemma 3. *Stage I of MICDA mechanism is Individual Rational.*

Proof. For any participant not accepted by the MICDA mechanism, no payment or return is required, and therefore, their utility is 0. For participants who are accepted by the mechanism, the utility of md_i is given by the payment function of mobile devices in Eq. (27):

$$U_i = \begin{cases}(vd_i - pd) \cdot \sqrt{\Omega_i}, & \text{if } vd_{(k)} \geq pd \geq cd_{(k)} \cdot \eta_{(k)} \\ (vd_i - vd_{(k)}) \cdot \sqrt{\Omega_i}, & \text{otherwise}\end{cases} \tag{38}$$

In the first case of Eq. (38), $vd_i \geq vd_{(k)}$, so the utility $U_i \geq 0$. In the second case, $vd_{(k)} \geq pd$ and $vd_i \geq pd$, so $U_i \geq 0$ also holds.

The utility of sp_j is given by the revenue function of service providers in Eq. (28):

$$U_j = \begin{cases}(pd - cd_i^j \cdot \eta_i) \cdot \sqrt{\Omega_i}, & \text{if } vd_{(k)} \geq pd \\ & \text{and } pd \geq cd_{(k)} \cdot \eta_{(k)} \\ (cd_{(k)} - cd_i^j) \cdot \sqrt{\Omega_i}, \cdot \eta_i & \text{otherwise}\end{cases} \tag{39}$$

In the first case, $cd_{(k)} \geq cd_i^j$, so the utility of sp_j is non-negative. In the second case, $pd \geq cd_i^j \cdot \eta_i$ is obtained by the ranking algorithm, hence $U_j \geq 0$. Therefore, the individual rationality of the MICDA mechanism in Stage I is proven. □

Lemma 4. *Stage II of MICDA mechanism is Individual Rational.*

Proof. For the mobile devices in Stage II, the dummy bid with a reserve price and the threshold payment ensures the utility of the mobile device $U_i \geq 0$, which means their participant will not bring negative utility. So, the individual rationality of mobile devices is ensured.

For the service providers, the VCG payment rule is applied to calculate the revenue of sp_j, which is:

$$r_j = c_i^j(S_i^{j^*}) - \left(\sum_{j'\neq j} c_i^j(S_i^{j*}) - \min_{S_i^{j'}\subseteq M_i} \sum_{j'\neq j} c_i^{j'}(S_{j'})\right) \tag{40}$$

By the truthfulness of MICDA Stage II of Lemma 2, it follows that $\sum_{j'\neq j} c_i^j(S_i^{j*}) \geq \min_{S_i^{j'}\subseteq M_i} \sum_{j'\neq j} c_i^{j'}(S_i^{j'})$. Hence, $r_j \geq c_i^j(S_i^{j*})$, proving the individual rationality of service providers. □

Theorem 2. *The MICDA mechanism is Individually rational.*

Proof. By invoking Lemma 3 and Lemma 4, the individual rationality of the MICDA mechanism is established. □

6.3 (Weak) Budget Balance

Theorem 3. *The MICDA mechanism is Weak Budget Balance.*

Proof. In Stage I of the MICDA mechanism, for each accepted mobile device and service provider pair (md_i, sp_j), the net of payment and revenue is:

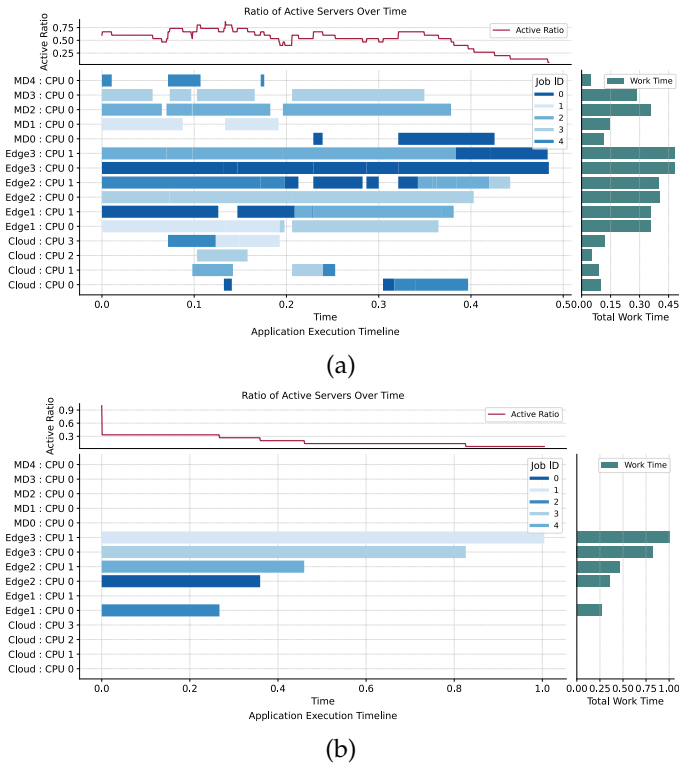

Fig. 4. Eventplots of (a) MICDA and (b) KM matching ($m = 5$ and $n = 3$).

$$net_{i,j} = \begin{cases} 0, & \text{if } vd_{(k)} \geq pd \\ & \text{and } pd \geq cd_{(k)} \cdot \eta_{(k)} \\ (vd_{(k)} - cd_{(k)} \cdot \eta_i) \cdot \sqrt{\Omega_i}, & \text{otherwise} \end{cases} \tag{41}$$

By the ranking criteria in Section 5.6.1, $\eta_k \geq \eta_i$. Hence, $vd_{(k)} \geq cd_{(k)} \cdot \eta_i$, which supports $net_{i,j} \geq 0$ in MICDA Stage I. In the second stage of MICDA, utilizing the critical payment described in Eq. (34), we can establish that $p_i \geq r_j$. Consequently, $net_{i,j} = p_i - r_j \geq 0$.

Hence, the net value $net_{i,j} \geq 0$ in the Stage II of the MICDA mechanism. In conclusion, $net_{i,j} = p_i - r_j \geq 0$ in both stages of the proposed MICDA mechanism. Hence, a weak budget balance is established. □

6.4 Computation Efficiency

Theorem 4. *The MICDA mechanism is Computation Efficiency.*

Proof. In the initialization phase, building the CPC model has a time complexity of $\mathcal{O}((|\mathcal{V}_i|+|\mathcal{E}_i|))$ for each $\mathcal{G}_i$. In Stage I of the MICDA mechanism, the value query for each SP operates in $\mathcal{O}(|\Phi|)$, where $|\Phi|$ is the size of the longest path in $\mathcal{G}_i$. The winner determination in Stage I is completed in $\mathcal{O}(m\log(m + n\log(n))$. Therefore, Stage I is calculated in polynomial time.

The total auction round of MICDA is smaller than $m|\mathcal{V}_i|$. In Stage II, the bundle construction time complexity is $\mathcal{O}(m^2L|\mathcal{V}_i|^2(log|\mathcal{V}_i|)$, where L is the number of consumers in Φ_i. The demand oracle used in the Ellipsoid method on the hypergraph-based cost proxy model operates in polynomial time [23]. Then, the dual problem of winner determination Eq. (23) can be solved in polynomial time with an efficient demand oracle [46]. Therefore, all parts of Stage II are proven to finish in polynomial time.

Hence, both stages in MICDA run in polynomial time, which proves the computational efficiency of the proposed MICDA mechanism. □

7 Evaluations

This section is dedicated to a comprehensive evaluation of the proposed MICDA algorithm through numerous simulations. The description commences with an introduction to the experimental parameters and the algorithms employed for comparison. Subsequently, simulation outcomes are presented, followed by an analysis aiming to demonstrate the performance attributes of the proposed mechanism.

7.1 Experimental Settings

7.1.1 Parameter Setting

This evaluation involves a network topology within a 60m×60m area, featuring randomly dispersed MEC servers. The communication parameters for these servers include a bandwidth of $B_i = 20$ MHz, path loss exponent of $a = 2.5$, noise power spectral density of $N_0 = 10^{-9}$, transmission power of $\rho_i = 3$ W, and channel gain of $h_i = 10^{-3}$, as suggested by [14]. The transmission rate between the edge server and the cloud is 10MB/s. Furthermore, the CPU frequency, f_i, ranges between 3 GHz and 3.5 GHz, and the effective switched capacitance, κ_i, lies between 10^{-27} and $10^{-26.5}$. The number of CPUs is 2 for edge servers. As for the cloud, the CPU frequency is set to 5 GHz and the CPU number is set to 4. As for the mobile device, the bandwidth is set at $B_l = 20$ MHz, there is only one CPU and the frequency is ranged between 1 GHz and 2 GHz. The effective switched capacitance is set to $\kappa_l = 10^{-26}$.

The DAG utilized for the evaluation are generated randomly by maintaining the level of parallelism constant while gradually modifying the depth. The generation algorithm is modified from Dai [47]. The average input data size, σ_j, is 100 KB, the number of CPU cycles required to process a unit bit, ω_j, is 80 M cycles/bit, and the local output data size, Θ_j, is 8 KB. To simulate the diversity of actual demands, the components of mobile application range from 10 to 50. During the simulation, each mobile device is equipped randomly with several subtasks in their DAG configuration.

7.1.2 Comparsion Methods

To evaluate the feasibility and performance of MICDA mechanism, we set two categories of the algorithm as comparisons, dependency-aware and holistic methods, respectively. Below is the explanation of each offloading algorithm.

- $\mathbf{Random_D}$: The component is randomly offloaded to a service provider or the mobile device itself.
- $\mathbf{Greedy_D}$: Topological sorting the DAG first, then sorting the request of mobile devices by their valuations in decreased order. For each mobile device following the sorted sequence, offload the components of each mobile device greedily to the most time-saving entities without considering any monetary cost or social welfare.

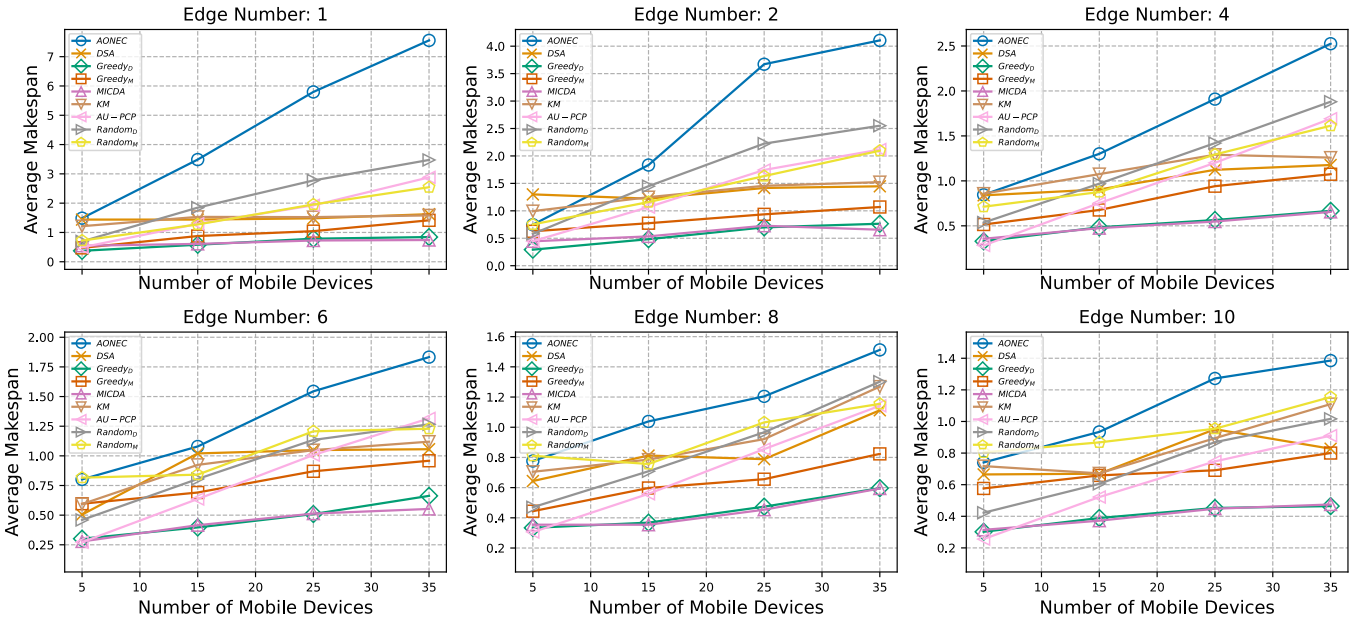

Fig. 5. Comparison of average makespan for different edge and mobile device numbers

- **DSA**: Topological sorting of the DAG firstly, then applying a second-price double sequential auction (DSA) to assign the components of mobile devices.
- **AU-PCP**: The AU-PCP algorithm employs an online auction-based approach for offloading dependent tasks in MEC environments. It utilizes a greedy winner selection to maximize user valuation and incorporates a heuristic for task assignment, enhancing offloading efficiency.

For the holistic type of offloading algorithm, the application is regarded as an entire computation task to offload:

- **AONEC**: For any mobile device, the whole application is offloaded to the nearest services provider.
- **Random$_{\mathrm{M}}$**: The application is offloaded randomly to an Edge/Cloud server for any mobile device.
- **Greedy$_{\mathrm{M}}$**: The mobile devices are sorted by the total workload and sequentially allocate the corresponding component to the most time-saving SP.
- **KM**: Utilizing the Kuhn-Munkres [48], [49] algorithm for maximum weight matching, similar to the methodology by Yao et al. [50]. Edge weights are adapted from network potential to social welfare of pairing mobile devices with service providers.

7.2 Experimental results

7.2.1 Overview

The importance of dependency in computation offloading tasks in mobile edge computing scenarios is illustrated from three aspects (application makespan, active ratio of the system, average workload) through Fig. 4. We specifically compare the MICDA mechanism proposed by us and the matching mechanism of the KM algorithm as dependency-aware and holistic offloading algorithms, respectively.

Firstly, under the same experimental conditions, the dependency-aware MICDA mechanism achieves a smaller

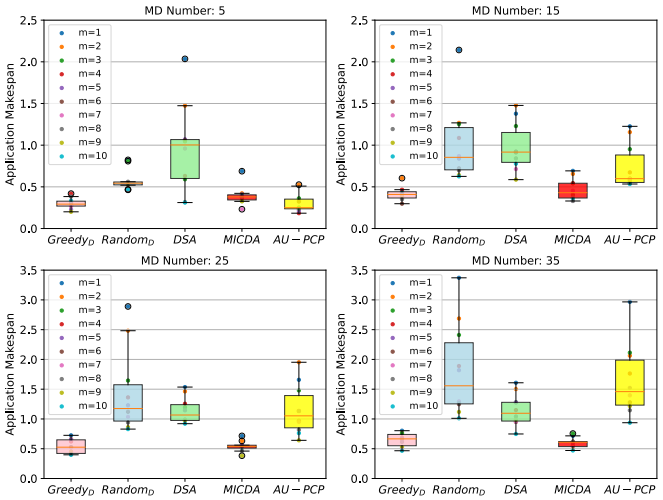

Fig. 6. Boxplot comparison of makespan by five dependency-aware offloading algorithms

makespan than the KM mechanism, with the former being less than half of the latter in terms of makespan. Moreover, through the spike raster plot, it can be seen that MICDA fully utilizes the topological relationships in mobile application subtasks and the parallel computing capabilities.

Secondly, it can be seen that the proportion of active entities in the entire system under the MICDA mechanism is much higher than that under the KM mechanism. This means that more entities can participate in offloading services without idling and wasting computational resources. The active ratio of the MICDA mechanism remains at a high level, exceeding 75 percent at its peak and maintaining above 50 percent for the majority of the time. In contrast, the KM algorithm only reaches a maximum of 30 percent and is continuously decreasing.

Lastly, for MICDA mechanism, the workload of the en-

tire system is borne by more entities, rather than being concentrated on a few service providers as in the KM algorithm. Considering the above three points, the dependency-aware mechanism is superior to the direct offloading mechanism, as the makespan of the application program is reduced by more than half. For service providers, their computational capabilities and resources are more effectively allocated, reducing their idle time. Empirically, the dependency-aware MICDA mechanism improves the benefits of both parties involved, contributing to better social welfare. This will also be confirmed in the subsequent experimental analysis.

7.2.2 Makespan

In this section, we compare the performance of all offloading algorithms based on dependency and blindness in terms of the average makespan of all mobile applications. Fig 5 indicates that the methods based on dependency awareness are significantly superior to the holistic methods. Although the disparity diminishes as the number of mobile edge servers increases and the overall computational resources of the system become more abundant, this trend still underscores the significance of incorporating dependency awareness into MEC offloading. For each algorithm, increasing the number of edge servers will help the average makespan of mobile devices. For the AONEC algorithm, mobile devices always offload their computation to the nearest edge device. However, in densely populated areas, this approach can lead to a workload imbalance, with some edge devices being heavily utilized while others remain idle, thus underutilizing system resources. Interestingly, for the two variants of the random algorithm, incorporating dependency awareness did not yield an improvement in makespan performance. This observation emphasizes the importance of well-designed offloading strategies. Compared to the $Greedy_M$ algorithm, the KM algorithm demonstrates weaker makespan performance, primarily because it focuses on maximizing the social welfare of the matching rather than solely considering makespan. For each subplot, we fixed the number of edge nodes and changed the number of mobile devices. The AU-PCP algorithm achieves an impressive average makespan when the network has abundant overall offloading computational resources. Although this advantage diminishes as the overall network workload increases, it remains competitive with any holistic offloading strategy. This reflects the effectiveness of its dependency-aware heuristic assignment algorithm. The $Greedy_D$ algorithm and the MICDA mechanism achieved similar and the best results. However, we can see that as the number of mobile devices increases (the total computing demand increases) under multiple edge number settings, the MICDA algorithm will have a more obvious advantage. This proves that in situations where the supply-demand relationship of computing resources is more tense, the MICDA mechanism can better utilize the topological structure of DAG and achieve a more effective components assignment.

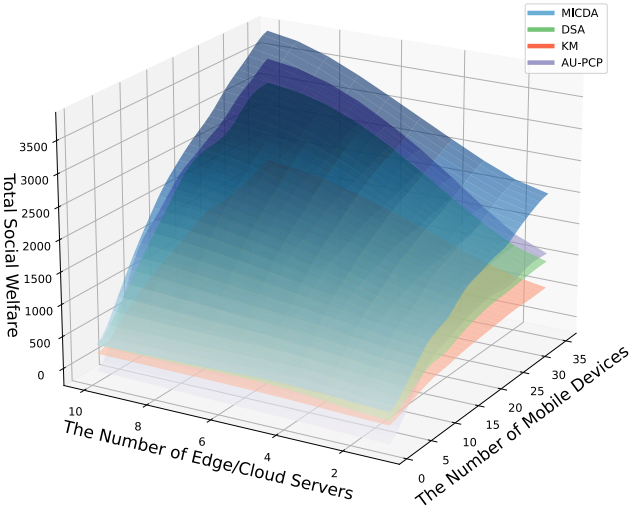

Fig. 7. Comparison of social welfare achieved by different offloading mechanisms

In addition, we specifically compare five dependency-aware algorithms in Fig. 6. When the workload is relatively small (MD number = 5), the average effect of different algorithms under different numbers of edges will be close. This gap becomes more and more obvious as the workload of the entire system increases. This reflects that even for dependency-aware algorithms, different offloading strategies can have a significant impact. While the AU-PCP algorithm demonstrates superior average makespan performance under conditions of lower overall network workload compared to the DSA algorithm, it tends to be outperformed by DSA as the workload increases. This occurs because both algorithms operate online offloading auction mechanisms; however, AU-PCP prioritizes processing all DAG components of a single user first. DSA, on the other hand, operates on a finer granularity, focusing on the most critical DAG component for all users at the current moment. This allows for a more effective exploitation of the dependencies between different components and the computational resources of the service providers. In the case of the DSA algorithm, its performance is inferior to the $Greedy_D$ method, which might be attributed to its consideration of offloading cost. The DSA algorithm tends to prioritize offloading to service providers with the lowest cost that still satisfies the constraints. While this approach takes into account cost-effectiveness, it might not yield the best makespan performance, leading to a lower overall efficiency compared to the $Greedy_D$ algorithm. Moreover, the MICDA mechanism has proven the effectiveness of its offloading strategy under different parameters.

7.2.3 Social Welfare

Fig. 7 illustrates the social welfare achieved by the KM, MICDA, and DSA mechanisms. The social welfare of the KM mechanism under different numbers of mobile devices and MEC servers is lower than the other mechanisms that consider dependency awareness. The MICDA achieves the highest social welfare compared to the DSA and AU-PCP mechanism because its offloading strategy better utilizes the topological structure of the DAG and the computing resources of the entire system. By comparison, the AU-PCP mechanism, while being an innovative approach, operates on a one-sided mechanism that, despite taking user valuation into consideration, fails to account for the cost and preferences of service providers. This limitation hinders its

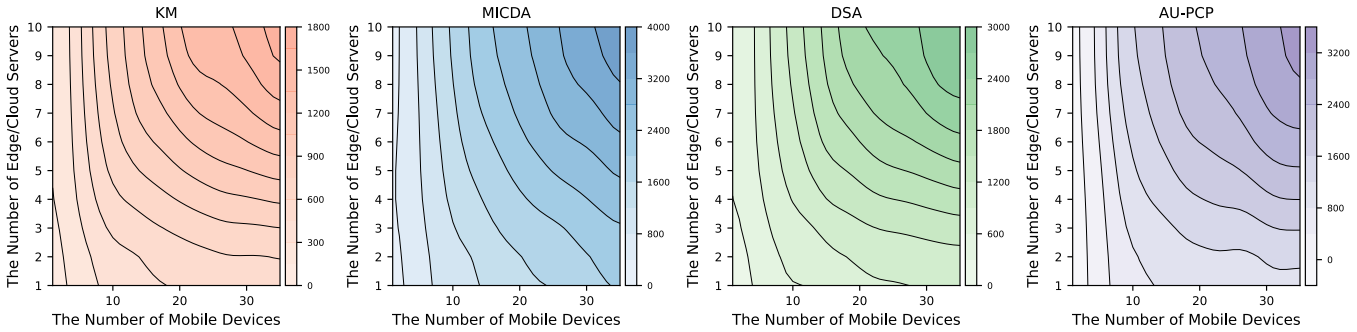

Fig. 8. The contour plot of social welfare achieved by different offloading mechanisms

ability to achieve higher levels of social welfare. As for the DSA algorithm, which primarily focuses more on costs in user's valuation, whereas the MICDA and AU-PCP mechanisms comprehensively consider both factors related to makespan and cost. This also leads to lower social welfare.

Fig. 8 presents a comparative contour plot that maps social welfare achieved by different offloading mechanisms to the number of mobile devices against the number of edge/cloud servers. The KM mechanism shows a relatively gradual increase in social welfare as the number of edge/cloud servers increases. This suggests that while the KM algorithm does benefit from additional computational resources, the improvement in social welfare is not as substantial when the number of mobile devices is high. This again reflects the weakness of the holistic strategy with the workload increase. In contrast, the MICDA mechanism exhibits a more pronounced increase in social welfare with the increase in both mobile devices and edge/cloud servers. This indicates a robust scalability of the MICDA mechanism, demonstrating significant improvement in managing higher workloads and efficiently utilizing the computation resources of service providers. Both the DSA and AU-PCP mechanisms exhibit sensitivity to the allocation of computational resources, as indicated by the closeness of contour lines for DSA and the steep gradient for AU-PCP. This implies that while both mechanisms are capable of achieving high social welfare, their performance is particularly sensitive to the balance between the DAG workloads and computation resource supply. Specifically, they seem to thrive under conditions of abundant resource availability, but may not scale as efficiently with an increase in demand without a corresponding increase in resources. This demonstrates that the proposed MICDA mechanism achieves higher resource utilization efficiency due to its better exploitation of task dependencies. It can realize greater social welfare even when computational resources are relatively scarce.

7.2.4 *Efficiency*

Auction round is defined as the process in that bidders submit a bid until he/she receives an allocation result or one mutual interaction between bidders and auctioneer. Fig. 9 shows the number of auction rounds required for the DSA, MICDA and AU-PCP mechanisms to complete under different numbers of mobile devices (which also represent the number of components). MICDA is more efficient than DSA. As the number of items increases, the efficiency difference between MICDA and DSA becomes more apparent. This is because the MICDA mechanism not only uses combinatorial auctions but also uses bidding language, leveraging the XOR bidding language to allow service providers bid across multiple mobile devices in each auction round. The proposed method breaks through the limitation of single-minded service providers. What's more, the number of auction rounds required by the AU-PCP mechanism grows linearly with the number of mobile devices because it processes user requests iteratively, one round at a time. This approach yields higher efficiency compared to the MICDA mechanism when the number of users is relatively low. However, as the number of mobile devices increases, the required auction rounds for AU-PCP exceed those of the MICDA mechanism.

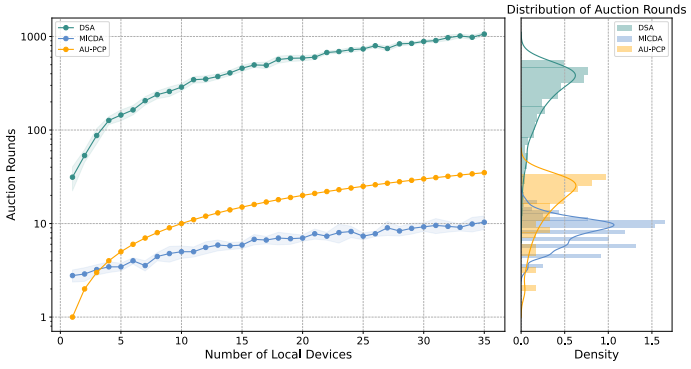

Fig. 9. Comparison of auction rounds of different mechanisms

8 Conclusion

In summary, we addressed a pressing gap in the realm of computation offloading in MEC systems: the integration of incentive mechanisms with dependent computation offloading. Recognizing the interconnection of subtasks in mobile applications, we introduced the multi-stage iterative combinatorial double auction (MICDA) mechanism—a pioneering approach that melds dependency-aware task offloading with combinatorial auction mechanisms. Our rigorous theoretical analysis validated the mechanism's adherence to pivotal economic properties, including truthfulness, individual rationality, weak budget balance, and polynomial time complexity. Simulation experiments further highlighted its superiority in enhancing application makespan and improving

social welfare, outperforming baseline algorithms. Regarding future directions, we envision deploying our mechanism on real testbeds to tackle the challenges associated with implementing system practices. This step will enable us to further refine and validate the feasibility and performance of our proposed mechanism.

REFERENCES

[1] H. Kang, M. Li, S. Fan, and W. Cai, "Combinatorial auction-enabled dependency-aware offloading strategy in mobile edge computing," in *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2023, pp. 1–6.

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

[3] J. Ren, Y. He, G. Huang, G. Yu, Y. Cai, and Z. Zhang, "An edge-computing based architecture for mobile augmented reality," *IEEE Network*, vol. 33, no. 4, pp. 162–169, 2019.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[5] X. Zhang, H. Chen, Y. Zhao, Z. Ma, Y. Xu, H. Huang, H. Yin, and D. O. Wu, "Improving cloud gaming experience through mobile edge computing," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 178–183, 2019.

[6] H. Jiang, X. Dai, Z. Xiao, and A. K. Iyengar, "Joint task offloading and resource allocation for energy-constrained mobile edge computing," *IEEE Transactions on Mobile Computing*, 2022.

[7] H. Hu, W. Song, Q. Wang, R. Q. Hu, and H. Zhu, "Energy efficiency and delay tradeoff in an mec-enabled mobile iot network," *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 15 942–15 956, 2022.

[8] J. Feng, L. Liu, X. Hou, Q. Pei, and C. Wu, "Qoe fairness resource allocation in digital twin-enabled wireless virtual reality systems," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 11, pp. 3355–3368, 2023.

[9] S.-T. Hong and H. Kim, "Qoe-aware computation offloading to capture energy-latency-pricing tradeoff in mobile clouds," *IEEE Transactions on mobile computing*, vol. 18, no. 9, pp. 2174–2189, 2018.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] P. R. Milgrom, *Putting auction theory to work*. Cambridge University Press, 2004.

[18] T. Roughgarden, "Algorithmic game theory," *Communications of the ACM*, vol. 53, no. 7, pp. 78–86, 2010.

[19] 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.

[20] 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.

[21] 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.

[22] R. P. McAfee, "A dominant strategy double auction," *Journal of economic Theory*, vol. 56, no. 2, pp. 434–450, 1992.

[23] I. Abraham, M. Babaioff, S. Dughmi, and T. Roughgarden, "Combinatorial auctions with restricted complements," in *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2012, pp. 3–16.

[24] R. Lavi and C. Swamy, "Truthful and near-optimal mechanism design via linear programming," *Journal of the ACM (JACM)*, vol. 58, no. 6, pp. 1–24, 2011.

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

[26] E. H. Clarke, "Multipart pricing of public goods," *Public choice*, pp. 17–33, 1971.

[27] T. Groves, "Incentives in teams," *Econometrica: Journal of the Econometric Society*, pp. 617–631, 1973.

[28] W. Cai, H. C. Chan, X. Wang, and V. C. Leung, "Cognitive resource optimization for the decomposed cloud gaming platform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 2038–2051, 2015.

[29] S. Shen, Y. Ren, Y. Ju, X. Wang, W. Wang, and V. C. Leung, "Edgematrix: A resource-redefined scheduling framework for sla-guaranteed multi-tier edge-cloud computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 3, pp. 820–834, 2022.

[30] L. Zhao, E. Zhang, S. Wan, A. Hawbani, A. Y. Al-Dubai, G. Min, and A. Y. Zomaya, "Meson: A mobility-aware dependent task offloading scheme for urban vehicular edge computing," *IEEE Transactions on Mobile Computing*, 2023.

[31] Q. He, G. Cui, X. Zhang, F. Chen, S. Deng, H. Jin, Y. Li, and Y. Yang, "A game-theoretical approach for user allocation in edge computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 515–529, 2019.

[32] Z. Sun, G. Sun, Y. Liu, J. Wang, and D. Cao, "Bargain-match: A game theoretical approach for resource allocation and task offloading in vehicular edge computing networks," *IEEE Transactions on Mobile Computing*, 2023.

[33] Y. Zhan, S. Guo, P. Li, and J. Zhang, "A deep reinforcement learning based offloading game in edge computing," *IEEE Transactions on Computers*, vol. 69, no. 6, pp. 883–893, 2020.

[34] Y. Bai, L. Chen, L. Song, and J. Xu, "Risk-aware edge computation offloading using bayesian stackelberg game," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 1000–1012, 2020.

[35] L. Ma, X. Wang, X. Wang, L. Wang, Y. Shi, and M. Huang, "Tcda: Truthful combinatorial double auctions for mobile edge computing in industrial internet of things," *IEEE Transactions on Mobile Computing*, vol. 21, no. 11, pp. 4125–4138, 2021.

[36] H. Zhou, T. Wu, X. Chen, S. He, D. Guo, and J. Wu, "Reverse auction-based computation offloading and resource allocation in mobile cloud-edge computing," *IEEE Transactions on Mobile Computing*, 2022.

[37] X. Chen, G. Zhu, H. Ding, L. Zhang, H. Zhang, and Y. Fang, "End-to-end service auction: A general double auction mechanism for edge computing services," *IEEE/ACM Transactions on Networking*, vol. 30, no. 6, pp. 2616–2629, 2022.

[38] M. Diamanti, P. Charatsaris, E. E. Tsiropoulou, and S. Papavassiliou, "Incentive mechanism and resource allocation for edge-fog networks driven by multi-dimensional contract and game theories," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 435–452, 2022.

[39] 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.

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

[41] 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.

[42] J. Li, W. Liang, W. Xu, Z. Xu, X. Jia, W. Zhou, and J. Zhao, "Maximizing user service satisfaction for delay-sensitive iot applications in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 5, pp. 1199–1212, 2021.
[43] J. Li, W. Liang, W. Xu, Z. Xu, X. Jia, A. Y. Zomaya, and S. Guo, "Budget-aware user satisfaction maximization on service provisioning in mobile edge computing," *IEEE Transactions on Mobile Computing*, 2022.
[44] S. Dobzinski and S. Dughmi, "On the power of randomization in algorithmic mechanism design," *SIAM Journal on Computing*, vol. 42, no. 6, pp. 2287–2304, 2013.
[45] L. Blumrosen and N. Nisan, "On the computational power of iterative auctions," in *Proceedings of the 6th ACM Conference on Electronic Commerce*, 2005, pp. 29–43.
[46] M. Grötschel, L. Lovász, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, pp. 169–197, 1981.
[47] X. Dai, "dag-gen-rnd: A randomized multi-DAG task generator for scheduling and allocation research," Mar. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6334205
[48] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
[49] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, pp. 32–38, 1957.
[50] Y. Yao, Y. Qin, W. Feng, P. Li, X. Xu, X. Xu, and X. Liang, "Kfto: Kuhn-munkres based fair task offloading in fog networks," *Computer Networks*, vol. 195, p. 108131, 2021.

Hong Kang [S'22] received the B.Eng. in Electronic Information Engineering from The University of Electronic Science and Technology of China and University of Glasgow in 2021. He is currently working towards M.Phil. with the School of Science and Engineering at The Chinese University of Hong Kong, Shenzhen, China. He is working as an Research Assistant in Human-Crypto Society Laboratory. His research interests include blockchain, mechanism design and edge intelligence.

Minghao Li received the B.Eng. degree in software engineering from Xiamen University, China in 2020 and the M.Phill. degree in Computer and Information Engineering from The Chinese University of Hong Kong, Shenzhen, China in 2022. He is currently a second-year Ph.D. student at the College of Computer and Data Science, Nanyang Technological University, Singapore. His research interests include data center digital twins, cloud-edge-end systems, and AI-empowered data center operations, and sustainable computing.

Lehao Lin received the B.Eng. degree in Computer Science and Engineering from The Chinese University of Hong kong, Shenzhen, China, in 2021. He is currently working towards the Ph.D. degree in Computer and Information Engineering with The Chinese University of Hong Kong, Shenzhen, China. He is working as a research assistant with Human-Crypto Society Laboratory. His current research interests include blockchain, human-centered computing and multimedia.

Sizheng Fan [S'20] received the B.Eng. degree in Automation from Beijing Institute of Technology, China in 2018, and the Ph.D. degree in Computer and Information Engineering at The Chinese University of Hong Kong, Shenzhen, China. He is working as an Research Assistant in Human-Crypto Society Laboratory. His current research interests include blockchain, federated learning, game theory and crowdsourcing. He is a student member of the IEEE and the CCF.

Wei Cai [S'12-M'16-SM'22] received the B.Eng. degree in Software Engineering from Xiamen University, China in 2008, the M.S. degree in Electrical Engineering and Computer Science from Seoul National University, Korea, in 2011, and the Ph.D. degree in Electrical and Computer Engineering from The University of British Columbia (UBC), Vancouver, Canada, in 2016. From 2016 to 2018, he was a Postdoctoral Research Fellow with UBC. He is currently an Assistant Professor of Computer Engineering with the School of Science and Engineering at The Chinese University of Hong Kong, Shenzhen. He is serving as the director of the Human-Crypto Society Laboratory, as well as the director of the CUHK(SZ)-White Matrix Joint Metaverse Laboratory. He has co-authored more than 100 journal and conference papers in the areas of distributed/decentralized systems and interactive multimedia. His recent research interests are mainly in the topic of human-centered computing for metaverse, including blockchain, Web3, digital games, and computational art. He is now serving as an associate editor for IEEE Transactions on Computational Social Systems (TCSS), IEEE Transactions on Cloud Computing (TCC), ACM Transactions on Multimedia Computing, Communications and Applications (TOMM), program co-chair for ACM NOSSDAV'23, TPC member for top conferences including ACM MM, MMSys, WWW, NOSSDAV, and guest editor for many leading journals including ACM TOMM, IEEE Multimedia, TNSE, TCSS, etc. He was a recipient of 6 Best Paper Awards. He is a senior member of the IEEE and a member of the ACM.