Incentivizing and Orchestrating Cloud-Edge LLM Speculative Decoding via Auctions

Mingtao Ji[1,2], Lei Jiao[3], Bin Tang[1], Zhihao Qu[1], Chen Chen[4], Baoliu Ye[2]
[1]College of Computer Science and Software Engineering, Hohai University, China
[2]State Key Laboratory of Novel Software Technology, Nanjing University, China
[3]Center for Cyber Security and Privacy, University of Oregon, USA
[4]Department of Computer Science, Nottingham Trent University, UK

***Abstract*—To realize the full potential of cloud-edge speculative decoding for Large Language Models (LLMs), it is essential to incentivize participation from user-owned edge devices. This paper presents a thorough mathematical and algorithmic study of this emerging setting, proposing an auction-based framework to minimize social cost that captures speculative decoding overhead, prompt dispatching latency, and bidding expenses. The resulting optimization problem involves combinatorial decisions, long-term constraints, and NP-hard complexity, while requiring economic guarantees. We design a novel online mechanism that integrates multiple polynomial-time sub-algorithms: a linearization-based prompt dispatcher, an online fractional algorithm with randomized rounding for winning-bid and draft-length selection, and a Myerson-based payment scheme. Our approach is rigorously proven to achieve sub-linear regret and constraint violation while ensuring truthfulness and individual rationality. Extensive evaluations on our NVIDIA-based testbed with real-world traces exhibit that our approach reduces the social cost substantially by 30%∼60% compared to state-of-the-art and heuristic methods.**

I. Introduction

Cloud-edge *speculative decoding* has emerged as a promising paradigm [1] to accelerate Large Language Model (LLM) inference. In this setup, a powerful cloud-based *target LLM* is paired with smaller, faster *draft LLMs* [2] hosted on distributed edge devices. These edge models generate speculative token sequences that are then verified in parallel by the cloud model, reducing the sequential computational burden on the cloud [3], [4] and cutting inference latency without compromising generation quality—since the cloud model still governs the final output. Leveraging distributed edge resources also enhances scalability and allows for robust LLM inference services [5].

Existing edge computing platforms, e.g., Multi-access Edge Computing servers [6]–[8], provide a stable foundation, but they often offer limited, geographically-constrained capacity and are typically provisioned through static pay-as-you-go models [9] that lack the agility to adapt to highly variable LLM inference demand. Meanwhile, the vast computational capacity of third-party user-owned devices (e.g., desktops, tablets, and smart routers) can provide the necessary scale and proximity to end users [10], but introduce dynamic availability and heterogeneous performance, contingent on user participation. A unified market mechanism is thus desirable to dynamically and efficiently integrate dedicated infrastructure and opportunistic consumer devices into a single responsive resource pool.

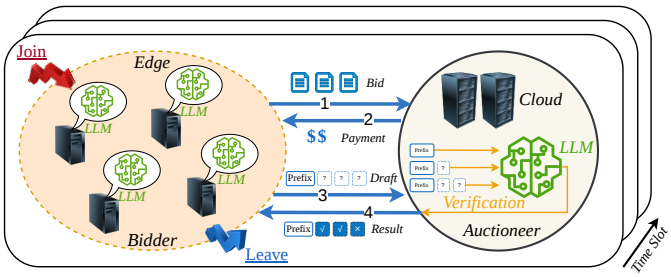


Fig. 1: Auction-Based Cloud-Edge Speculative Decoding

To address this need, we propose an *auction*-based framework. As shown in Fig. 1, edge devices including edge servers can act as the *bidders* and submit bids specifying desired compensation for draft generation, and the cloud can act as the *auctioneer* and select winning bids to minimize the social cost for the entire system. This enables the system to adapt to fluctuating supply and demand, recruit the most cost-effective edge resources with reasonable compensation [11], and realize performance-aware economically-sustainable cloud-edge LLM inference. However, designing such an auction mechanism is a non-trivial task, facing unique and fundamental challenges.

First, this mechanism inevitably entangles the complicated orchestration of distributed LLM speculative decoding in the edge resource procurement. The speculative decoding latency is a non-linear function of the draft length [12] which needs to be decided to strike the following balance: a long draft may cause waste of computation at the edge, while a short draft could lead to frequent interactions with the cloud, incurring excessive communication overhead [4]; some user prompts may not use speculative decoding at all but switch to standard cloud-only LLM inference due to varying network conditions. Also, due to the heterogeneity of each edge device in terms of location and capability, it is critical to dispatch user prompts to edge devices without disproportionate delay or straggler effect. This entails min-max optimization [13] subject to available resources from the auction. Both draft length selection and user prompt dispatching are thus difficult to solve due to the nonsmooth and discrete nature of the problem.

Second, as the system operates continuously [1], [14], this mechanism needs to manage auctions with the conflicting objectives of single-auction economic robustness and long-term

participant retention. For each individual auction, the guarantees of *truthfulness* and *individual rationality* [11] are often desired but are already hard to attain in our circumstance. The former ensures no motivation for each bid to lie about its true cost, and the latter ensures no loss for each bid. Despite the standard Vickrey-Clarke-Groves (VCG) [15] can achieve both, it necessitates finding the exact optimum of the underlying problem; yet, our case is intractable (as elaborated later), and speculative decoding even makes fractional VCG inapplicable [16]. Across auctions, ensuring long-term participation could require guaranteeing a minimum cumulative win rate. This inter-temporal constraint contradicts the per-auction winning-bid selection and payment design that underpins truthfulness and individual rationality, since any rule favoring past losers may distort incentives and open avenues for manipulation.

Unfortunately, prior research exhibits substantial limitations and fails to address the above challenges. Many of those on cloud-edge LLM serving optimization [17]–[22] rarely study speculative decoding. Those on LLM speculative decoding in particular [3]–[5], [12], [23], [24] do not typically address all elements targeted in this work: the uncertain cloud-edge environment, the dynamic online optimization, and the incentive mechanisms. Meanwhile, the vast majority of LLM auction research [25]–[29] applies LLMs to various auction settings, rather than incorporating auctions into LLM system design. See Section VI for detailed discussions on related work.

This paper presents the first investigation into auction-driven orchestration for cloud-edge LLM inference using speculative decoding. We introduce a rigorous mathematical framework with novel algorithms, supported by theoretical analysis and empirical validation. Our contributions are fourfold.

Model: We formulate a long-term social cost optimization problem to capture a series of auctions over time by minimizing the speculative-decoding computation and communication delay between the edge devices and the cloud, the prompt dispatching delay to the edge devices, and the bidding cost of all bids. Our models embed all the aforementioned challenges, making almost no assumption on the input heterogeneity and dynamics, and pursue the dynamic control of winner selection, prompt dispatching, draft configuration, and payment allocation. Our problem turns out to be a non-linear integer program, for which we prove the NP-hardness.

Algorithm: We design a comprehensive polynomial-time online mechanism that coordinates four distinct sub-algorithms to jointly solve our proposed problem: (i) We attack the lexicographic min-max optimization [30] of prompt dispatching in each time slot via separable convex functions and linearization [13], where we further use total unimodularity to seek integer dispatching decisions by standard linear optimization solvers; (ii) Given dispatching decisions, we utilize online learning to handle long-term constraints by alternate primal-dual updates that can be conducted as time progresses, obtaining fractional decisions for winning-bid selection and draft length selection; (iii) We devise a resource-aware randomized rounding strategy to convert such fractional decisions into integers, jointly addressing the combinatorial explosion of the solution space; (iv) Based on our randomization process, we apply an equivalent variant of the Myerson's Lemma [31] to calculate the payment for each winner to attain economic guarantees in expectation.

TABLE I: Notations

Inputs	Meaning
B_j	Resource capacity of edge j
r	Resource requirement per token generation at edge
φ_j	Minimum winning count for edge (bid) j over time
K_i	Total tokens to generate for prompt i
α	Acceptance rate for verification in cloud
Γ_j^{edge}	Time for draft LLM on edge j to generate a single token
$\beta_{j,t}^{\text{up}}$	Upload delay per token from edge j to cloud at t
T^{cloud}	Cloud target LLM forward pass time
$\beta_{j,t}^{\text{down}}$	Download delay per token from cloud to edge j at t
$d_{i,j,t}$	Communication cost for dispatching prompt i to edge j
$\gamma_{\max}$	Maximum allowed draft length
$b_{j,t}$	Bidding price for bid j at t
Decisions	**Meaning**
$x_{j,t}$	Whether edge (bid) j is selected at t
$y_{i,j,t}$	Whether prompt i is assigned to edge j for processing at t
$\gamma_{j,t}$	Draft length for prompts at edge j at t
$p_{j,t}$	Payment to edge j at t

Analysis: We conduct formal performance analysis for our algorithms. First, we analyze the performance of each of our sub-algorithms from different angles, and based on those we prove that our entire approach leads to sub-linear *regret*, i.e., the time-averaged social cost asymptotically converges to the series of single-time-slot optimums in hindsight, and sub-linear *fit*, i.e., the time-averaged cumulative constraint violation vanishes as time elapses. Next, we focus on economic properties and prove that each of our auctions attains *truthfulness* and *individual rationality* in expectation by satisfying the desired necessary and sufficient conditions for randomized auctions.

Experiment: We implement a prototype testbed using NVIDIA GeForce RTX and Jetson platforms, various LLMs (e.g., Qwen and Llama series), and an existing speculative decoding framework [2]. We collect the inputs to our experiments through testbed profiling [32] and real-world data sources such as LLM chats [33], target models' acceptance rates [24], [34], draft models' maximum draft lengths [34], electricity prices [35], network latency [36], [37], and so on. Our results demonstrate that our proposed mechanism (i) reduces social cost by more than 30% compared to state-of-the-arts [4], [12] and up to about 60% compared to heuristics, (ii) adapts well in system environments of varying network conditions and LLM configurations, with consistent, scalable, and robust cost advantages, and (iii) accomplishes the desired economic guarantees in each auction in practice.

II. Modeling and Formulation

A. System Modeling

Our notations are summarized in Table I.

Cloud-Edge LLM Speculative Decoding: We consider a cloud Large Language Model (LLM) inference service that operates over a series of time slots, denoted as $\mathcal{T} = \{1, \ldots, T\}$. In each time slot $t \in \mathcal{T}$, the end users submit a group of prompts, denoted as $\mathcal{I}_t$, for processing. The cloud LLM

inference service uses the auctions, as stated next, to recruit edge devices from an edge device set $\mathcal{J} = \{1, \ldots, J\}$ and conducts the cloud-edge speculative decoding [1], [24], [34] to generate response for each prompt. In the time slot t, for the prompt $i \in \mathcal{I}_t$, suppose the cloud uses the edge device $j \in \mathcal{J}$ to process it; the cloud-edge speculative decoding consists of multiple iterations, with each iteration described as follows.

The edge device j uses its deployed draft LLM to generate a sequence of $\gamma_{j,t}$ draft tokens auto-regressively, denoted as $[\hat{\tau}_1, \ldots, \hat{\tau}_{\gamma_{j,t}}]$, with the corresponding sequence of probabilities $q(\gamma_{j,t}) = [q_1, \ldots, q_{\gamma_{j,t}}]$, and sends these to the cloud. Upon receipt, the cloud uses its target LLM to perform a single parallel forward pass over the $\gamma_{j,t}$ tokens and produce its own probabilities $\rho(\gamma_{j,t}) = [\rho_1, \ldots, \rho_{\gamma_{j,t}}]$, which often incurs a fixed verification time due to parallelization. The cloud afterwards conducts a sequential acceptance test, which only accepts a token if the corresponding probability in $\rho(\gamma_{j,t})$ is no less than that in $q(\gamma_{j,t})$, until the first rejection occurs or until all draft tokens are accepted. In that position, the cloud adopts or appends a new token from the target LLM. The cloud further sends all accepted tokens with the one new token to the edge device j to initiate the next iteration.

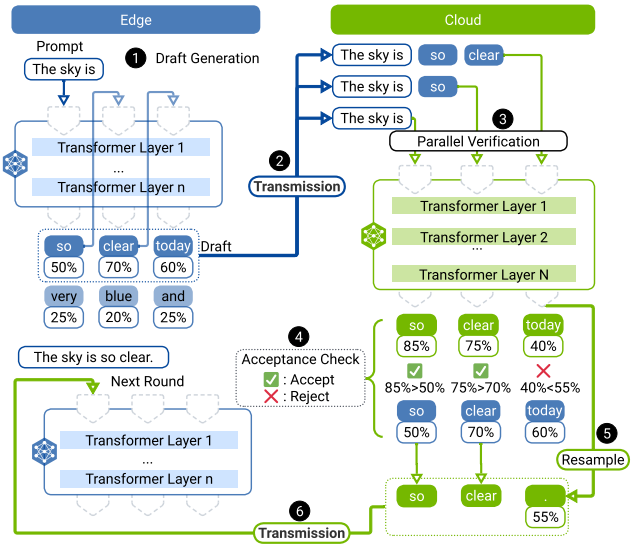


Fig. 2: Cloud-Edge Speculative Decoding

We provide an indicative illustration of one cloud-edge speculative-decoding iteration as in Fig. 2. The existing tokens the edge device has is "`The sky is`", e.g., from the result of the last iteration, and the draft length is set to 3.

❶ **Draft Generation:** The edge draft LLM generates the draft tokens auto-regressively, e.g., `["so", "clear", "today"]`, and their probabilities, e.g., `[50%, 70%, 60%]`.

❷ **Data Transmission:** The edge device transfers both the draft tokens and the probabilities to the cloud.

❸ **Parallel Verification:** The cloud target LLM executes a single parallel forward pass, generating the corresponding probabilities, e.g., `[85%, 75%, 40%]` for `["so", "clear", "today"]`.

❹ **Token Acceptance Check:** The cloud compares `[85%, 75%, 40%]` to `[50%, 70%, 60%]` sequentially until the first rejection or the end. A draft token is accepted only if the corresponding probability in the former is no less than that in the latter.

❺ **Resampling upon Rejection:** In the position of rejection, or the position after the last draft token (if all draft tokens are accepted), the target LLM's distribution is used to resample a corrected token. In this specific example, after accepting `"so"` and `"clear"`, the token `"today"` is rejected and resampled as `"."`.

❻ **Feedback Loop:** The sequence of the accepted tokens with the one corrected token, i.e., `["so", "clear", "."]`, is transferred to the edge device. "`The sky is`" is updated to "`The sky is so clear.`", enabling the next iteration to start.

Now, consider the computation and communication delay of generating the response to the user prompt i via cloud-edge speculative decoding using the edge device j at the time slot t. We use K_i to denote the number of the output tokens in the response. We also use α to denote the token acceptance rate of the cloud target LLM, and thus the expected number of tokens returned from the cloud in a single speculative-decoding iteration is $\frac{1-\alpha^{\gamma_{j,t}+1}}{1-\alpha}$ [34], where $\gamma_{j,t}$ is draft length per iteration set for the draft LLM deployed on the edge device j. The expected number of iterations to generate K_i tokens is $K_i \cdot \frac{1-\alpha}{1-\alpha^{\gamma_{j,t}+1}}$. Each iteration consists of edge-side draft generation, uplink transmission, cloud-side verification, and downlink transmission. That is,

$$\Gamma_{i,j,t} \triangleq K_i \frac{1-\alpha}{1-\alpha^{\gamma_{j,t}+1}} [\Gamma_j^{\text{edge}} \gamma_{j,t} + \beta_{j,t}^{\text{up}} \gamma_{j,t} + T^{\text{cloud}} + \beta_{j,t}^{\text{down}} \frac{1-\alpha^{\gamma_{j,t}+1}}{1-\alpha}],$$

where Γ_j^{edge} is the generation time per token; T^{cloud} is the one-pass cloud verification time; and $\beta_{j,t}^{\text{up}}$ and $\beta_{j,t}^{\text{down}}$ denote the uplink communication delay and the downlink communication delay per token, respectively.

Edge Device Auctions: The cloud LLM inference service acts as the auctioneer, and the edge devices act as the bidders. At each time slot t, the auctioneer conducts one auction: (i) Bid collection: Each edge device j submits a bid $\{b_{j,t}, B_j, \Gamma_j^{\text{edge}}\}$, where $b_{j,t}$ is the bidding price, i.e., the amount of money this edge device wants to charge, for draft generation; B_j represents the resource capacity; and Γ_j^{edge} denotes the per-token generation latency. (ii) Winner selection: Upon collecting all bids $\mathbf{b}_t \triangleq \{b_{1,t}, \ldots, b_{J,t}\}$, the cloud LLM service decides the winning bids by solving a social cost minimization problem, as stated next. (iii) Prompt dispatching and execution: The cloud LLM service dispatches user prompts to the corresponding edge devices according to the winning bids, and conducts speculative decoding to generate the response to each user prompt. (iv) Payment calculation: The cloud LLM service computes the payment $p_{j,t}$ (which is not necessarily equal to $b_{j,t}$) for each winning bid j and delivers this payment to the

corresponding edge device. As in standard auction settings, a bid is a commitment, i.e., the edge device is committed to contributing its resource if selected as a winner.

Control Decisions: At each time slot t, the cloud LLM inference service makes the following control decisions in real time: $x_{j,t} \in \{1, 0\}$ indicates whether or not the bid from the edge device j is selected as a winning bid for the auction at t; $y_{i,j,t} \in \{1, 0\}$ represents whether or not the user prompt i is assigned to the edge device j for processing at t; $\gamma_{j,t} \in \{0, 1, 2, \ldots, \gamma_{\max}\}$ specifies the number of the draft tokens to generate per speculative-decoding iteration by the edge device j's draft LLM at t, where 0 denotes no speculative decoding (i.e., no user prompts are sent to this device) and $\gamma_{\max}$ denotes a practical upper bound for the draft length [1], [34]; $p_{j,t} \geq 0$ represents the payment paid to the bidder j at t.

Cost of Auctioneer: The cost of the auctioneer consists of three components: (i) the total speculative decoding latency, i.e., $\sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}_t} y_{i,j,t}\Gamma_{i,j,t}$; (ii) the maximum prompt dispatching latency that needs to be minimized, i.e., $\max_{i\in\mathcal{I}_t, j\in\mathcal{J}} y_{i,j,t}d_{i,j,t}$, where $d_{i,j,t}$ is the latency for dispatching the prompt i to the edge device j at the time slot t; (iii) the total payment made by the auctioneer and paid to the winning bidders, i.e., $\sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} x_{j,t}p_{j,t}$.

Cost of Bidders: The cost of the bidders has two components: (i) the total bidding cost, i.e., $\sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} x_{j,t}b_{j,t}$; (ii) the total payment received from the auctioneer, treated as negative cost, i.e., $-\sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{J}} x_{j,t}p_{j,t}$.

B. Social Cost Minimization Problem

Problem Formulation: The *social cost* refers to the total cost incurred in the entire system, i.e., the sum of the auctioneer's cost and the bidders' cost. Based on our models, we formulate the social cost minimization problem $\mathscr{P}$:

$$\min \sum_{t\in\mathcal{T}} \{ \sum_{j\in\mathcal{J}} \{ \sum_{i\in\mathcal{I}_t} y_{i,j,t}\Gamma_{i,j,t} + x_{j,t}b_{j,t}\} + \max_{i\in\mathcal{I}_t, j\in\mathcal{J}} y_{i,j,t}d_{i,j,t}\}$$

$$C_1: \sum_{i\in\mathcal{I}_t} y_{i,j,t} K_i \frac{1-\alpha}{1-\alpha^{\gamma_{j,t}+1}} \gamma_{j,t} r \leq B_j x_{j,t}, \forall j \in \mathcal{J}, t \in \mathcal{T},$$

$$C_2: \sum_{t\in\mathcal{T}} x_{j,t} \geq \varphi_j, \forall j \in \mathcal{J},$$

$$C_3: \sum_{j\in\mathcal{J}} y_{i,j,t} = 1, \forall i \in \mathcal{I}_t, t \in \mathcal{T},$$

$$C_4: \gamma_{j,t} \leq \gamma_{\max} \min\{\sum_{i\in\mathcal{I}_t} y_{i,j,t}, 1\}, \forall j \in \mathcal{J}, t \in \mathcal{T},$$

$$\text{var. } x_{j,t} \in \{0,1\}, y_{i,j,t} \in \{0,1\}, \gamma_{j,t} \in \{0, 1, 2, \ldots, \gamma_{\max}\},$$
$$\forall i \in \mathcal{I}_t, j \in \mathcal{J}, t \in \mathcal{T}.$$

The objective function captures the social cost. Constraint C_1 enforces the edge capacity, where r is the resource overhead per draft token generation and B_j is the resource capacity of the edge device j. Constraint C_2 ensures that each edge device wins for at least φ_j times cumulatively across all auctions, encouraging long-term participation [38]. Constraint C_3 ensures that every prompt is dispatched. Constraint C_4 ensures that an edge device is used when there is at least one prompt dispatched to this edge device.

Note that the payments are automatically canceled in the social cost, which is not uncommon in auction-related research, but the payments still need to be decided. Also, each term in the objective function can be associated with a *weight*, which is not displayed in the above for brevity but can be set and controlled by the auctioneer based on its own needs to navigate the optimization trade-offs among the different terms.

Problem Challenges: First, minimizing $\max_{i,j} y_{i,j,t}d_{i,j,t}$ over discrete $y_{i,j,t}$ confronts a combinatorial solution space, and the $\Gamma_{i,j,t}$ function of discrete $\gamma_{j,t}$ is nonlinear and non-convex. Second, Constraint C_2 couples $x_{j,t}$ over time, making it difficult to minimize $\sum_t x_{j,t}b_{j,t}$ by deciding $x_{j,t}$ at the time slot t on the fly, given that all future $b_{j,t}$ are unknown. Third, our problem is intractable even if we consider only one time slot and neglect the above two aspects, not to mention that we aim to solve the long-term problem in an online manner.

Theorem 1. The problem $\mathscr{P}$ is NP-hard, due to reduction from the *bin packing* problem.

Proof. Consider $T = 1$ and thus remove the subscript t from all notations. We set the following special values to the inputs: $b_j = 1, \forall j$; $B_j = B, \forall j$; $r = 1 + \alpha$; $d_{i,j} = 0, \forall i, \forall j$; $\varphi_j = 0, \forall j$. We also set $\gamma_{\max} = 1$, and $\Gamma_j^{\text{edge}} = \beta_j^{\text{up}} = \beta_j^{\text{down}} = 0, \forall j$. We have $\Gamma_{i,j} = K_i T^{\text{cloud}}$ if $\gamma_j = 0$, and $\Gamma_{i,j} = K_i \frac{T^{\text{cloud}}}{1+\alpha}$ if $\gamma_j = 1$. We now consider the bin packing setting, where the set $\mathcal{I}$ represents the items and the set $\mathcal{J}$ represents the bins.

Regarding $\mathscr{P}$'s Constraints: With $\gamma_j = 1$, C_1 changes to $\sum_i y_{i,j}K_i \leq Bx_j$, which is the standard bin capacity constraint; with $\gamma_j = 0$, C_1 changes to $0 \leq Bx_j$, which always holds. C_2 always holds. C_3 is the standard item assignment constraint in the bin packing problem. For C_4, we can set $\gamma_j = 0$ if $\sum_i y_{i,j} = 0$, and set $\gamma_j = 1$ if $\sum_i y_{i,j} \geq 1$.

Regarding $\mathscr{P}$'s Objective: We show that we can set T^{cloud} appropriately, so that for any bin j that is open, we always have $\gamma_j = 1$, rather than $\gamma_j = 0$, in $\mathscr{P}$'s optimum. To that end, we consider any j, where $\sum_i y_{i,j} \geq 1$. Now, if we choose $\gamma_j = 0$, the objective with respect to j becomes $\sum_i y_{i,j}K_i T^{\text{cloud}}$; if we choose $\gamma_j = 1$, it becomes $1 + \sum_i y_{i,j}K_i \frac{T^{\text{cloud}}}{1+\alpha}$. Our goal is to ensure $\sum_i y_{i,j}K_i T^{\text{cloud}} > 1 + \sum_i y_{i,j}K_i \frac{T^{\text{cloud}}}{1+\alpha}$. Solving this inequality, we have $T^{\text{cloud}} > \frac{1+\alpha}{\alpha \min_i K_i}$. That is, we can just set T^{cloud} accordingly, so that $\mathscr{P}$ becomes bin packing. □

Fourth, with all aforementioned complexities, we still need to decide the payments to satisfy the desired economic guarantees as stated later, going beyond standard auction settings.

III. Mechanism Design

To solve $\mathscr{P}$, we design an online algorithmic framework, i.e., **Algorithm 0**, which dynamically invokes multiple other algorithms at each time slot: **Algorithm 1** decides the prompt dispatching; **Algorithm 2**, which further invokes **Algorithm 3**, selects the winning bids for the auction and the draft length for each corresponding edge device; **Algorithm 4** calculates the payment for each winning bid. Each of these algorithms focuses on only part of the problem, and they work jointly to address all the control decisions for each time slot.

A. Algorithm 1: Dispatching Prompts

For each time slot t, we extract problem $\mathscr{P}_{t,1}$ from $\mathscr{P}$ as

Algorithm 0 Online Edge Speculative Decoding (OESD)

for $t = 1, 2, ..., T$ **do**
  Invoke **Alg. 1** to decide prompt dispatching;
  Invoke **Alg. 2∼3** to select winning bids and draft lengths;
  Conduct cloud-edge LLM speculative decoding;
  Invoke **Alg. 4** to calculate payment;
end for

Algorithm 1 Prompt Dispatching Algorithm, $\forall t$

1: Extract $\mathscr{P}_{t,1}$ from $\mathscr{P}$, and transform $\mathscr{P}_{t,1}$ to $\mathscr{P}_{t,1}^{LP}$;
2: Get $\bar{\boldsymbol{y}}_t$ by solving $\mathscr{P}_{t,1}^{LP}$ via a linear program solver [39];
3: **return** $\bar{\boldsymbol{y}}_t$;

$$\begin{aligned}
\min \quad & \mathcal{P}_{t,1} \triangleq \max_{i\in\mathcal{I}_t, j\in\mathcal{J}} y_{i,j,t} d_{i,j,t} \\
s.t. \quad & \textstyle\sum_{i\in\mathcal{I}_t} y_{i,j,t} \le \frac{B_j}{K_{\max}\cdot\gamma_{\max}\cdot r}, \forall j \in \mathcal{J}, \quad (1)\\
& \textstyle\sum_{j\in\mathcal{J}} y_{i,j,t} = 1, \forall i \in \mathcal{I}_t, \quad (2)\\
\text{var.} \quad & y_{i,j,t} \in \{0,1\}, \forall i, j.
\end{aligned}$$

Constraint (2) comes from Constraint C_3 of $\mathscr{P}$. For Constraint (1), we note the following, based on Constraint C_1 of $\mathscr{P}$:

$$\textstyle\sum_{i\in\mathcal{I}_t} y_{i,j,t} K_i \frac{1-\alpha}{1-\alpha^{\gamma_{j,t}+1}} \gamma_{j,t} r \le K_{\max}\cdot\gamma_{\max}\cdot r \sum_{i\in\mathcal{I}_t} y_{i,j,t},$$

which is due to $K_{\max} \triangleq \max_{i\in\mathcal{I}_t} K_i$, and $\frac{1-\alpha}{1-\alpha^{\gamma_{j,t}+1}} \le 1$. In the rest of this section, we demonstrate that we equivalently transform the non-convex min-max integer program $\mathscr{P}_{t,1}$ to a linear program $\mathscr{P}_{t,1}^{LP}$, which can then be solved easily via any standard linear program solver. This is actually **Algorithm 1**.

This transformation process is to change the problem while preserving the optimal solution. Since the $\max\{\cdot\}$ function is non-differentiable and discontinuous, locating its optimum is non-trivial. We first transform $\mathscr{P}_{t,1}$ to $\mathscr{P}_{t,1}^{C}$ through the sum of separable convex functions (i.e., exponential functions in our case) and then transform $\mathscr{P}_{t,1}^{C}$ to $\mathscr{P}_{t,1}^{LP}$ through linearization via "λ-representation" [13]. We elaborate as follows.

From $\mathscr{P}_{t,1}$, we set $\eta = |\mathcal{I}_t| * |\mathcal{J}| + 1$, and derive $\mathscr{P}_{t,1}^{C}$:

$$\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{I}_t}\sum_{j\in\mathcal{J}} \eta^{y_{i,j,t} d_{i,j,t}} \\
\text{s.t.} \quad & (1),\ (2),\\
\text{var.} \quad & y_{i,j,t} \in \{0,1\}, \forall i, j.
\end{aligned}$$

From $\mathscr{P}_{t,1}^{C}$, we derive $\mathscr{P}_{t,1}^{LP}$:

$$\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{I}_t}\sum_{j\in\mathcal{J}} (\eta^{d_{i,j,t}} - 1) y_{i,j,t} \\
\text{s.t.} \quad & (1),\ (2),\\
\text{var.} \quad & 0 \le y_{i,j,t} \le 1, \forall i, j.
\end{aligned}$$

Solving the linear program $\mathscr{P}_{t,1}^{LP}$ will lead to the optimal integral solution to $\mathscr{P}_{t,1}$, due to our **Lemma 1** as shown later. Its proof also exhibits the details of the above transformation.

Complexity: Algorithm 1 invokes a standard linear program solver, e.g., the interior-point method, and its complexity can be $\mathcal{O}((|\mathcal{I}_t||\mathcal{J}|)^2 \log(1/\kappa))$ [40], where $|\mathcal{I}_t||\mathcal{J}|$ represents the number of decisions and κ is the solution precision.

Algorithm 2 Bid and Draft Length Selection Algorithm, $\forall t$

Input: Previous decision $\bar{\boldsymbol{x}}_{t-1}$ and $\bar{\boldsymbol{\gamma}}_{t-1}$; and $\boldsymbol{\lambda_t}, \omega, \theta$;
1: Obtain $\widetilde{\boldsymbol{\gamma}}_t$ via solving the problem (3);
2: Obtain $\bar{\boldsymbol{\gamma}}_t$ via invoking **Algorithm 3**;
3: Given $\bar{\boldsymbol{\gamma}}_t$, obtain $\widetilde{\boldsymbol{x}}_t$ by solving the problem (3);
4: Obtain $\bar{\boldsymbol{x}}_t$ via invoking **Algorithm 3**;
5: Update dual as in (4);
6: **return** $\bar{\boldsymbol{x}}_t$ and $\bar{\boldsymbol{\gamma}}_t$.

B. Algorithm 2: Selecting Winning Bids and Draft Lengths

Given $\bar{y}_{i,j,t}$ returned by **Algorithm 1**, we extract the problem $\mathscr{P}_2$ from the problem $\mathscr{P}$ as

$$\begin{aligned}
\min \quad & \textstyle\sum_{t\in\mathcal{T}} \mathcal{P}_{t,2} \triangleq \sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{J}} \{\sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t}\Gamma_{i,j,t} + x_{j,t} b_{j,t}\} \\
s.t. \quad & \forall j: g_{j,1} = \textstyle\sum_t \{\gamma_{j,t} r \sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t} K_i - B_j x_{j,t}\} \le 0,\\
& \forall j: g_{j,2} = \textstyle\sum_t \{\frac{1}{T}\varphi_j - x_{j,t}\} \le 0,\\
& \forall j,t: h_{j,t} = \gamma_{j,t} \le \gamma_{\max} \textstyle\sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t},\\
\text{var.} \quad & x_{j,t}, \in [0,1], \gamma_{j,t} \in [0, \gamma_{\max}].
\end{aligned}$$

Note that the constraints here are from or transformed from Constraints C_1, C_2, and C_4 of $\mathscr{P}$. While C_4 is converted to $\gamma_{j,t} \le \gamma_{\max} \sum_{i\in\mathcal{I}_t} y_{i,j,t}$ equivalently, C_1 is converted to the long-term version, similar to the original C_2. This is because, as shown later, we actually allow the violation of such long-term constraints, but the time-averaged violation approaches zero as time goes to infinity. Also note that $\mathscr{P}_2$ is defined in the real domain, rather than the integer domain.

To solve $\mathscr{P}_2$, we introduce new notations for the purpose of concise presentation. We use $\boldsymbol{g}_t \triangleq [g_{1,1}, g_{2,1}, ..., g_{j,1}, ..., g_{1,2}, g_{2,2}, ..., g_{j,2}, ...]$ and $\boldsymbol{h}_t \triangleq [h_{1,t}, h_{2,t}, ..., h_{j,t}, ...]$ to represent the aggregation of constraints. $\mathscr{P}_2$ is expressed as

$$\begin{aligned}
\min_{X_t\in\mathbb{X}} \quad & \textstyle\sum_{t=1}^{T} f_t(\boldsymbol{X}_t) \\
\text{s.t.} \quad & \textstyle\sum_{t=1}^{T} \boldsymbol{g}_t(\boldsymbol{X}_t) \preceq \boldsymbol{0}; \boldsymbol{h}_t(\boldsymbol{X}_t) \preceq \boldsymbol{0}, \forall t,
\end{aligned}$$

where $\boldsymbol{X}_t = [\boldsymbol{x}_t^\top, \boldsymbol{\gamma}_t^\top]^\top$ is the aggregation of decision variables and $\mathbb{X} = [0,1]^J \times [0, \gamma_{\max}]^J$ represents the domain.

Algorithm 2 solves $\mathscr{P}_2$ as follows. Solving $\mathscr{P}_2$ in the above is equivalent to solving this min-max formulation:

$$\begin{aligned}
\min_{\boldsymbol{X}} \max_{\boldsymbol{\lambda}} \quad & \textstyle\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{X}_t, \boldsymbol{\lambda}_t) \triangleq \sum_{t=1}^{T} (f_t(\boldsymbol{X}_t) + \boldsymbol{\lambda}_t^\top \boldsymbol{g}_t(\boldsymbol{X}_t)) \\
\text{s.t.} \quad & \boldsymbol{h}_t(\boldsymbol{X}_t) \preceq \boldsymbol{0}, \boldsymbol{X}_t \in \mathbb{X}, \forall t,
\end{aligned}$$

where $\boldsymbol{\lambda}_t$ denotes the Lagrangian multiplier vector at time slot t. Through these multipliers, the long-term constraints are incorporated into the optimization objective. Now, we adopt the following primal-dual updates. That is, the decision $\boldsymbol{X}_t$ for the time slot t can be obtained by solving this problem:

$$\begin{aligned}
\min_{\boldsymbol{X}_t} \quad & \textstyle\sum_{t=1}^{T} (\nabla f_{t-1}(\widetilde{\boldsymbol{X}}_{t-1})(\boldsymbol{X}_t - \widetilde{\boldsymbol{X}}_{t-1}) \\
& + \boldsymbol{\lambda}_t^\top \boldsymbol{g}_{t-1}(\boldsymbol{X}_t) + \tfrac{\|\boldsymbol{X}_t - \widetilde{\boldsymbol{X}}_{t-1}\|}{2\omega}) \quad (3)\\
\text{s.t.} \quad & \boldsymbol{h}_t(\boldsymbol{X}_t) \preceq \boldsymbol{0}, \quad \boldsymbol{X}_t \in \mathbb{X}, \quad \forall t,
\end{aligned}$$

where $\frac{\|\boldsymbol{X}_t - \widetilde{\boldsymbol{X}}_{t-1}\|}{2\omega}$ denotes a regularization term that controls the variation between consecutive decisions; ω is a pre-defined step size; and $\|\cdot\|$ represents L_2 norm. Subsequently, based

Algorithm 3 Decision Rounding Algorithm, $\forall t$

Input: Fractional decisions $\widetilde{\boldsymbol{x}}_t$ and $\widetilde{\boldsymbol{\gamma}}_t$;
1: **for** $j \in \mathcal{J}$ **do**
2: **if** $\widetilde{\gamma}_{j,t}$ or $\widetilde{x}_{j,t}$ is integer **then**
3: $\bar{x}_{j,t} \leftarrow \widetilde{x}_{j,t}$, $\bar{\gamma}_{j,t} \leftarrow \widetilde{\gamma}_{j,t}$;
4: **end if**
5: **end for**
// **Rounding draft lengths**
6: **for** $j \in \mathcal{J}$ with fractional $\widetilde{\gamma}_{j,t}$ **do**
7: With probability $\widetilde{\gamma}_{j,t} - \lfloor\widetilde{\gamma}_{j,t}\rfloor$: $\bar{\gamma}_{j,t} \leftarrow \lceil\widetilde{\gamma}_{j,t}\rceil$;
With probability $\lceil\widetilde{\gamma}_{j,t}\rceil - \widetilde{\gamma}_{j,t}$: $\bar{\gamma}_{j,t} \leftarrow \lfloor\widetilde{\gamma}_{j,t}\rfloor$;
8: **end for**
// **Rounding bids selection**
9: **for** $j \in \mathcal{J}$ with fractional $\widetilde{x}_{j,t}$ **do**
10: **if** $\sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t} K_i \frac{1-\alpha}{1-\alpha^{\bar{\gamma}_{j,t}+1}} \bar{\gamma}_{j,t} r > 0$ **then**
11: $\bar{x}_{j,t} \leftarrow 1$;
12: **else**
13: $\bar{x}_{j,t} \leftarrow 0$;
14: **end if**
15: **end for**
16: **return** $\bar{\boldsymbol{x}}_t$ and $\bar{\boldsymbol{\gamma}}_t$;

on post-deployment feedback, the Lagrangian multipliers for $t+1$ can be updated as follows:

$$\boldsymbol{\lambda}_{t+1} = \left[\boldsymbol{\lambda}_t + \theta \nabla_{\boldsymbol{\lambda}} \mathcal{L}_t(\widetilde{\boldsymbol{X}}_t, \boldsymbol{\lambda}_t)\right]^+, \forall t, \tag{4}$$

where $[\,\cdot\,]^+ \triangleq \max\{\cdot, 0\}$, and θ is a pre-defined step size. Through alternately executing (3) and (4), the algorithm makes decisions online, solely based on information available up to the time slot $t-1$ without needing any inputs beyond t.

Complexity: Analogous to **Algorithm 1**, the complexity of both Lines 1 and 3 of **Algorithm 2** is $\mathcal{O}(|\mathcal{J}|^2 \log(1/\kappa))$. Line 2 or 4 invokes **Algorithm 3**, which takes $\mathcal{O}(|\mathcal{J}|)$, as shown next. Line 5 could also take $\mathcal{O}(|\mathcal{J}|)$, if implemented as a loop.

C. Algorithm 3: Converting Fractional Decisions to Integers

Algorithm 3 rounds the fractional decisions $\widetilde{\boldsymbol{\gamma}}_t$ and $\widetilde{\boldsymbol{x}}_t$ into integer values. We describe them, respectively, as follows.

Rounding $\widetilde{\boldsymbol{\gamma}}_t$: Lines 1∼5 preserve existing integer decisions, as they are not fractions and not to be rounded. For each remaining fractional $\widetilde{\gamma}_{j,t}$, Line 7 applies a randomized rounding procedure: with probability $\widetilde{\gamma}_{j,t} - \lfloor\widetilde{\gamma}_{j,t}\rfloor$, the value is rounded up to $\lceil\widetilde{\gamma}_{j,t}\rceil$, and with probability $\lceil\widetilde{\gamma}_{j,t}\rceil - \widetilde{\gamma}_{j,t}$, it is rounded down to $\lfloor\widetilde{\gamma}_{j,t}\rfloor$. This ensures that the expected value of each integral draft length equals its fractional value:

$$\begin{aligned}\mathbb{E}[\bar{\gamma}_{j,t}] &= (\widetilde{\gamma}_{j,t} - \lfloor\widetilde{\gamma}_{j,t}\rfloor) \cdot \lceil\widetilde{\gamma}_{j,t}\rceil + (\lceil\widetilde{\gamma}_{j,t}\rceil - \widetilde{\gamma}_{j,t}) \cdot \lfloor\widetilde{\gamma}_{j,t}\rfloor \\ &= \widetilde{\gamma}_{j,t}\lceil\widetilde{\gamma}_{j,t}\rceil - \lfloor\widetilde{\gamma}_{j,t}\rfloor\lceil\widetilde{\gamma}_{j,t}\rceil + \lfloor\widetilde{\gamma}_{j,t}\rfloor\lceil\widetilde{\gamma}_{j,t}\rceil - \widetilde{\gamma}_{j,t}\lfloor\widetilde{\gamma}_{j,t}\rfloor \\ &= \widetilde{\gamma}_{j,t}(\lceil\widetilde{\gamma}_{j,t}\rceil - \lfloor\widetilde{\gamma}_{j,t}\rfloor) = \widetilde{\gamma}_{j,t}.\end{aligned}$$

Rounding $\widetilde{\boldsymbol{x}}_t$: After similarly preserving existing integer values in Lines 1∼5, **Algorithm 3** evaluates, for each fractional $\widetilde{x}_{j,t}$, whether $\sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t} K_i \cdot \frac{1-\alpha}{1-\alpha^{\bar{\gamma}_{j,t}+1}} \cdot \bar{\gamma}_{j,t} \cdot r > 0$ holds. If this holds, then $\bar{x}_{j,t}$ is set to 1 and is otherwise set to 0. Lines 9∼15 lead to $\mathbb{E}[\bar{x}_{j,t}] \geq \widetilde{x}_{j,t}$, $\forall j$, $\forall t$. This holds due to the following: (i) Lines 10∼11 set $\bar{x}_{j,t} = 1$, and $\bar{x}_{j,t} = 1 \geq \widetilde{x}_{j,t}, \forall j$; (ii) Line 13 keeps $\bar{x}_{j,t} = \widetilde{x}_{j,t} = 0$. In both cases, we have $\bar{x}_{j,t} \geq \widetilde{x}_{j,t}$ and $\mathbb{E}[\bar{x}_{j,t}] \geq \widetilde{x}_{j,t}$.

Complexity: Algorithm 3 takes $\mathcal{O}(|\mathcal{J}|)$, due to its execution as loops. It could be reduced to $\mathcal{O}(1)$ if parallel execution is available and adopted.

Algorithm 4 Payment Calculation Algorithm, $\forall t$

Input: Bidding prices $b_{j,t}$ and $\mathbf{b}_{-j,t}, \forall j$;
1: **for** $j \in \mathcal{J}$ **do**
2: **if** $\bar{x}_{j,t} = 1$ **then**
3: $p_{j,t} \leftarrow b_{j,t}\widetilde{x}_{j,t}(b_{j,t}, \mathbf{b}_{-j,t}) + \int_{b_{j,t}}^{\chi_t} \widetilde{x}_{j,t}(s, \mathbf{b}_{-j,t})\mathrm{d}s$;
4: **else**
5: $p_{j,t} \leftarrow 0$;
6: **end if**
7: **end for**
8: **return** $p_{j,t}, \forall j$;

D. Algorithm 4: Calculating Payment

Algorithm 4 calculates the payment $p_{j,t}$ for each bidder j. If a bid or edge device is not selected, Line 5 sets its payment to zero. For a winning bid j, Line 3 calculates the payment following **Theorem 3**, as stated later, where $b_{j,t}$ is the bidding price of the bid j; $\mathbf{b}_{-j,t}$ denotes the bidding prices of all other bids except j; $\widetilde{x}_{j,t}(\cdot)$ is the fractional winning-bid selection decision before rounding, which also serves as the probability that the bid j is selected as a winner for a given bidding price; χ_t is an upper limit for the bidding price. We choose to follow this theorem to calculate the payment, as payment this way is guaranteed to attain truthfulness and individual rationality.

Complexity: Algorithm 4 takes $\mathcal{O}(|\mathcal{J}|^3 \varrho \log(1/\kappa))$. This is because, for each of the $|\mathcal{J}|$ iterations, we consider ϱ bidding prices, where ϱ is the number of discretization intervals for the numerical integration over $[b_{j,t}, \chi_t]$; for each of such ϱ bidding prices, we run **Algorithm 2** to obtain the corresponding $\widetilde{x}_{j,t}$.

IV. Performance Analysis

A. Regret and Fit Analysis

We first analyze the performance of our sub-algorithms, i.e., **Algorithms 1∼3**, from different aspects and then define and analyze the regret and the fit for our main algorithm, i.e., **Algorithm 0**. Note that lemmas and theorems are what we state and prove in this paper, while propositions are obtained from external literature without proofs shown in this paper.

Lemma 1. Algorithm 1 ensures the optimal integral solution $\bar{\boldsymbol{y}}_t^*$ to the integer program $\mathscr{P}_{t,1}$ equals the optimal fractional solution $\widetilde{\boldsymbol{y}}_t^*$ to the linear program $\mathscr{P}_{t,1}^{LP}$.

Proof. See Appendix A. The proof states and is also based on **Proposition 2** (i.e., exponential sum) and **Proposition 3** (i.e., "λ-representation"). □

Proposition 1. Algorithm 2 ensures the *regret* and the *fit* of the problem $\mathscr{P}_2$ grow sub-linearly with time [41], [42], i.e.,

$$\begin{gathered}\textstyle\sum_{t=1}^{T}\{\mathcal{P}_t(\widetilde{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t)\} - \sum_{t=1}^{T}\{\mathcal{P}_t(\widetilde{\boldsymbol{x}}_t^*, \bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t^*)\} \leq \mathcal{O}(T^{\tau_1}), \\ \textstyle\|\sum_{t=1}^{T} \boldsymbol{g}_t(\widetilde{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t)\| \leq \mathcal{O}(T^{\tau_2}),\end{gathered}$$

where $\tau_1, \tau_2 \in (0,1)$; and $\{\widetilde{\boldsymbol{x}}_t^*, \widetilde{\boldsymbol{\gamma}}_t^*\}$ is the optimal solution to $\mathscr{P}_{t,2}$ at t given $\bar{\boldsymbol{y}}_t$.

Lemma 2. Algorithm 3 ensures the expectation of the computation and communication latency of cloud-edge speculative decoding is bounded as follows:

$$E[\Gamma_{i,j,t}(\bar{\gamma}_{j,t})] \leq \Gamma_{i,j,t}(\widetilde{\gamma}_{j,t}) + \tfrac{M}{8},$$

where M is a constant.

Proof. See Appendix B. The proof is based on $E[\bar{\gamma}_{j,t}] = \widetilde{\gamma}_{j,t}$, Taylor's Theorem, and Cauchy-Schwarz Inequality [43]. □

In the following, we formally define *regret* and *fit*, and show that **Algorithm 0** incurs only sub-linear regret and fit for the problem $\mathscr{P}$, which is a preferred result for online algorithms.

Definition 1 (Regret)**.** Consider the optimization problem:

$$\min_{\mathcal{X}_t \in \mathbb{X}} \textstyle\sum_{t=1}^{T} \mathcal{P}_t(\mathcal{X}_t) \quad \text{s.t.} \textstyle\sum_{t=1}^{T} \boldsymbol{g}_t(\mathcal{X}_t) \preceq \mathbf{0}.$$

The *regret* of an online algorithm is defined as

$$\boldsymbol{Reg}(T) \triangleq \textstyle\sum_{t=1}^{T} \mathcal{P}_t(\bar{\mathcal{X}}_t) - \sum_{t=1}^{T} \mathcal{P}_t(\mathcal{X}_t^*),$$

where $\bar{\mathcal{X}}_t$ is the decision produced by the online algorithm for the time slot t, and $\mathcal{X}_t^*$ is the optimal decision for the time slot t, i.e., $\mathcal{X}_t^* \triangleq \arg\min_{\mathcal{X}_t \in \mathbb{X}} \mathcal{P}_t(\mathcal{X}_t)$, subject to $\boldsymbol{g}_t(\mathcal{X}_t) \preceq \mathbf{0}$.

Definition 2 (Fit)**.** The *fit* incurred by an online algorithm measures the cumulative constraint violation:

$$\boldsymbol{Fit}(T) \triangleq ||[\textstyle\sum_{t=1}^{T} \boldsymbol{g}_t(\bar{\mathcal{X}}_t)]^+||,$$

where $\bar{\mathcal{X}}_t$ is the decision produced by the online algorithm for the time slot t, and $[\cdot] \triangleq \max\{0, \cdot\}$.

Theorem 2. The *regret* and the *fit* of **Algorithm 0** grow sub-linearly, i.e.,

$$\boldsymbol{Reg}(T) \leq \mathcal{O}(T^{\tau_1}), \tau_1 \in (0,1); \boldsymbol{Fit}(T) \leq \mathcal{O}(T^{\tau_2}), \tau_2 \in (0,1).$$

Proof. See Appendix C. The proof exploits the connections between the integer domain and the continuous domain, and utilizes **Lemma 1**, **Proposition 1**, and **Lemma 2**. □

B. Economic Guarantees Analysis

Definition 3 (Utility)**.** The bid j's utility in the auction at t is

$$\upsilon_j(b_{j,t}, \mathbf{b}_{-j,t}) = \begin{cases} p_{j,t}(b_{j,t}, \mathbf{b}_{-j,t}) - b'_{j,t} E(\bar{x}_{j,t}(b_{j,t}, \mathbf{b}_{-j,t})), \\ \quad \text{if } \bar{x}_{j,t} = 1, \\ 0, \quad \text{otherwise}, \end{cases}$$

where $p_{j,t}(b_{j,t}, \mathbf{b}_{-j,t})$ is the received payment; $b'_{j,t}$ is the true cost of the bid j; $b_{j,t}$ refers to the bidding price of the bid j; $\mathbf{b}_{-j,t}$ refers to the bidding prices submitted by all the other bids; $\bar{x}_{j,t}$ is the winning-bid selection decision, which depends on bidding prices of all bids; and the expectation E is due to randomization.

Definition 4 (Individual Rationality)**.** An auction is *individually rational* if and only if for every bid j, its utility is always non-negative, i.e., $\upsilon_j(b_{j,t}, \mathbf{b}_{-j,t}) \geq 0$.

Definition 5 (Truthfulness)**.** An auction is *truthful* if and only if for every bid j, bidding its true cost achieves the maximum utility, i.e., $\upsilon_j(b'_{j,t}, \mathbf{b}_{-j,t}) \geq \upsilon_j(\hat{b}_{j,t}, \mathbf{b}_{-j,t})$, where $b'_{j,t}$ is the true cost of the bid j and $b'_{j,t} \neq \hat{b}_{j,t}, \forall \hat{b}_{j,t}$.

Theorem 3. A randomized auction is truthful and individually rational in expectation if and only if it meets three conditions, for any participating bid j: (i) $\bar{x}_{j,t}(b_{j,t}, \mathbf{b}_{-j,t})$ is monotonically non-increasing in $b_{j,t}$; (ii) $\int_0^\infty E(\bar{x}_{j,t}(s, \mathbf{b}_{-j,t}))\mathrm{d}s < \infty$; (iii) the payment is in the form of $p_{j,t} = b_{j,t} E(\bar{x}_{j,t}(b_{j,t}, \mathbf{b}_{-j,t})) + \int_{b_{j,t}}^{\chi_t} E(\bar{x}_{j,t}(s, \mathbf{b}_{-j,t}))\mathrm{d}s$. Our auctions meet these conditions.

Proof. See Appendix D. These conditions are a generalization of Myerson's Lemma [31] to randomized auctions in single-parameter (i.e., bidding price) settings, but are not automatically met. In the proof, we show our design meets them. □

V. Experimental Evaluations

A. Evaluation Setup

Testbed Implementation: We construct a testbed using an existing speculative decoding framework [2], which consists of a high-performance server equipped with an NVIDIA GeForce RTX 4080 GPU as the cloud server, and Jetson NX and Nano as the distinct edge devices. We employ Qwen-3 (8B/14B) and Llama-2 (13B) as the cloud target models, and a set of lightweight draft models on the edge devices, including Qwen-3 (0.6B/1.4B/4B), Llama-2 (160M), Qwen-2 (0.5B/1.5B), and Qwen-2.5 (0.5B/1.5B). We use five public datasets [32] from previous research for testbed profiling: SpecBench, C4 (en), OpenAssistant, WikiText-2, and MTBench. Table II summarizes our measured latency parameters T^{cloud} and Γ_j^{edge}.

LLM Parameters: We simulate three hundred 15-minute-long time slots. For the LLM workload, the token demand per prompt ranges in 200∼450, based on the WildChat [33] data. The speculative draft acceptance probability, α, is configured in 0.03∼0.9 [24], [34]. Via measurements from our testbed, the per-token generation latency on edge devices (Γ_j^{edge}) ranges in 15∼50 ms/token, while the cloud verification time (T^{cloud}) falls in 20∼100 ms. The maximum draft length, $\gamma_{\max}$, is constrained to 1∼24 [34]. Further, the prompt dispatch latency (i.e., $d_{i,j,t}, d_{i,c,t}$) comes from real sources [36].

Auction Configuration: The minimum selection count φ_j is set to 10∼100 [11]. The bidding price for each device is calculated by combining power consumption data from our testbed, with public electricity prices from CAISO [35]. The resource capacity of the edge devices (B_j) is set to 16∼64 GB, with the per-token resource requirement (r) profiled directly on our testbed. For cloud-edge communication, the unit upload delay ($\beta_{j,t}^{\text{up}}$) and the unit download delay ($\beta_{j,t}^{\text{down}}$) are set to 0.25∼0.5 s/token from public network dataset [36].

Algorithms: We implement all algorithms in Python. To comprehensively evaluate our proposed approach **OESD** (Online Edge Speculative Decoding), we compare it against multiple benchmark approaches:

- **Random** makes decisions for bid selection, prompt dispatching and draft length in a randomized manner.

TABLE II: Parameters of Speculative Decoding

Model Pair (Target/Draft)	T^{cloud} (ms)	Γ_j^{edge} (ms)	Len. γ	Lat. ms/tok	TP tok/s
Qwen 3-8B/0.6B	37.6	8.46	3	24.60	53
			4	24.51	54
			5	25.43	53
Qwen 3-8B/1.7B	37.6	11.67	3	28.60	35
			4	30.49	32
			5	31.33	32
Qwen 3-14B/4B	55.41	20.24	3	40.63	30
			4	41.84	31
			5	45.69	30
Llama2-13B/160M	48.50	10.77	3	22.64	54
			4	20.52	60
			5	21.07	59

Len. denotes draft length; Lat. denotes average latency.

- **Greedy** adopts a myopic strategy to make decisions, aiming to minimize the immediate cost in each current time slot without considering long-term performance.
- **Fast** [12] calculates the draft length based on the acceptance probability and the computation cost ratio [12], while making the remaining decisions via **Greedy**.
- **SOTA** [4] decides the draft length accounting for the communication cost between the edge and the cloud. The rest decisions also follow **Greedy**.

B. Evaluation Results

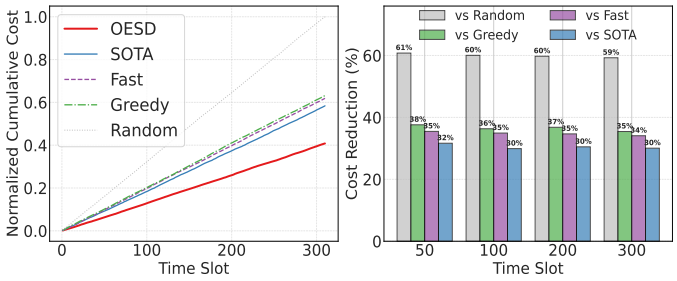
Social Cost: Fig. 3(a) depicts the cumulative social cost incurred by different algorithms. Our proposed OESD consistently maintains the lowest cost trajectory. This performance advantage stems from OESD's holistic optimization framework. While alternatives such as Fast and SOTA optimize draft length based on acceptance probabilities, they focus the problem only partially; they overlook the heterogeneity of distributed edge devices, often leading to higher social cost.

Fig. 3(b) quantifies the cost reductions achieved by OESD. The results show significant improvements: compared to Random, OESD achieves a cost reduction of 59%∼61%; against Greedy, OESD yields a gain of 35%∼38%, while providing an improvement of 34%∼35% over Fast; even against the advanced SOTA algorithm, OESD maintains a competitive edge, reducing the social cost by 30%∼32%.

Scalability: Fig. 4(a) illustrates the social cost under varying dispatch latency, analyzed across five distinct settings of $(125, 250)$, $(250, 500)$, $(500, 700)$, $(700, 725)$, and $(725, 1000)$ ms, respectively. The results indicate that OESD consistently outperforms others across all network conditions, achieving an average cost reduction of 20%∼50%. This highlights OESD's ability to effectively optimize prompt dispatching even in high-latency environments.

Fig. 4(b) presents the social cost under different token generation limits, categorized into the groups of $(200, 250)$, $(250, 300)$, $(300, 350)$, $(350, 400)$, $(400, 450)$. We observe that the social cost exhibits only mild fluctuations as the token generation length increases, indicating system robustness.

Fig. 4(c) depicts the impact of the edge-to-cloud upload latency. Unlike the token limit, variations in upload latency cause significant shifts in the social cost, highlighting the system's sensitivity to uplink bandwidth. Despite these fluctuations, OESD adapts agilely to network conditions, achieving an average improvement of 13.9% compared to others.



(a) Dynamic Social Cost (b) Cost Reduction

Fig. 3: Social Cost Comparison

Fig. 4(d) analyzes performance under varying cloud LLM generation times, spanning from $(20, 35)$ to $(85, 100)$ ms. OESD demonstrates substantial gains in this scenario, achieving a 58.1% improvement over Fast and a 33.9% improvement over SOTA. This confirms that OESD effectively accommodates heterogeneous cloud generation latencies.

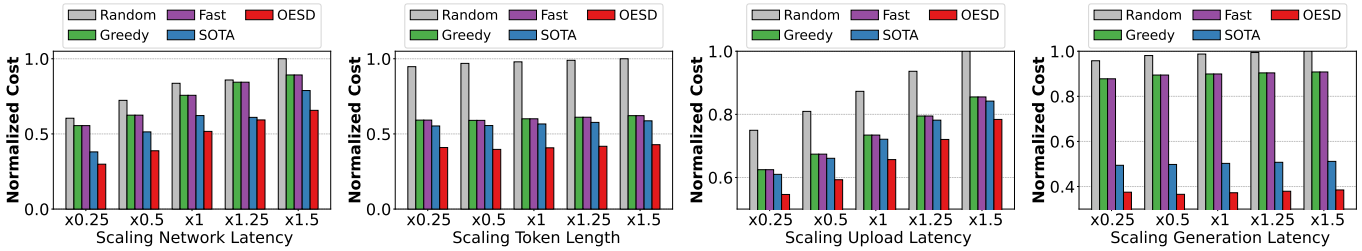
Individual Rationality and Truthfulness: Fig. 5(a) illustrates the bidding prices and the corresponding payments for different bids during a representative time slot. It is evident that the payment received by each winning edge device exceeds its bidding price, confirming non-negative utility and thus individual rationality. Fig. 5(b) analyzes a bid's utility as a function of the different reported bidding prices. The utility is maximized if and only if the bidder reports the true cost (i.e., 0.1 in this experiment), confirming truthfulness.

VI. Related Work

Cloud-Edge LLM Serving: Zhang *et al.* [18] propose EdgeShard, which partitions models across distributed devices to enhance latency and throughput. You *et al.* [19] present JPPO, utilizing model-based prompt compression and reinforcement-learning-driven power control to accelerate wireless LLM services. Li *et al.* [20] design a cloud-edge scheduling framework using combinatorial neural bandits to prioritize edge requests. Yang *et al.* [21] propose PerLLM, employing constraint-satisfaction bandits to optimize personalized inference and reduce energy costs. You *et al.* [22] also develop JPPO++, which integrates iterative denoising compression to maintain output quality.

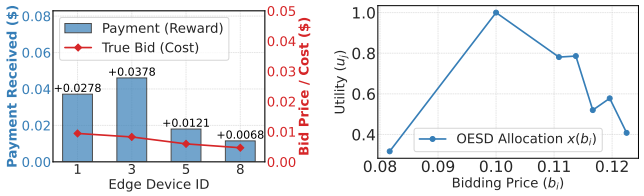
This group of research exploits distributed execution for collaborative LLM inference, addressing bandwidth and other resource overhead, fundamentally different from speculative decoding. Their methods are typically orthogonal and inapplicable to speculative-decoding optimization.

LLM Speculative Decoding: Leviathan *et al.* [12] pioneer speculative decoding to accelerate inference by utilizing a small draft model to generate candidate tokens for parallel verification. Venkatesha *et al.* [23] incorporate early exits and preemptive drafting mechanisms to reduce latency and cloud API costs. Addressing communication bottlenecks, Zheng *et al.* [3], [4] propose DSSD, which optimizes transmission



(a) Impact of Dispatching (b) Impact of Length Limit (c) Impact of Upload Latency (d) Impact of Generation

Fig. 4: Social Cost in Varying Environments



(a) Payment vs. Cost (b) Utility

Fig. 5: Individual Rationality and Truthfulness

by sending only token indices instead of full probability distributions. Regarding dynamic orchestration, Hao *et al.* [24] introduce a threshold mechanism based on a weighted cost function to balance quality and cost, while Xu *et al.* [5] propose LLMCad, utilizing confidence scores to adaptively control the verification frequency.

This group of works focus on speculative decoding; yet, they often study it in isolation and miss either cloud-edge environments, dynamic online optimization, or rigorous performance analysis. To the best of our knowledge, none of them has studied speculative decoding incentives.

LLM Auctions: Avinava *et al.* [25] introduce LLMs to dynamically generate ad summaries based on allocated prominence, maximizing social welfare and incentive compatibility. Zhao *et al.* [26] propose a generative mechanism that embeds ad allocation directly into LLM responses via iterative preference optimization. Cai *et al.* [27] develop RTBAgent, which dynamically adjusts bidding strategies by multi-memory mechanisms to adapt to market fluctuations. Huang *et al.* [28] use LLMs as proxies for preference elicitation, reducing communication costs while ensuring efficient allocation. Zhang *et al.* [29] utilize LLMs to simulate complex socioeconomic dynamics and market phenomena for Dutch auctions.

These studies employ LLMs for various types and settings of auctions, and few of them, if not none, has incorporated auctions into LLM serving design as in our work, not to mention that they typically do not consider speculative decoding.

VII. Conclusion

Our work in this paper advances cloud-edge collaborative LLM inference by enabling the system to dynamically recruit and use dynamic heterogeneous edge resources for speculative decoding. We formulate a long-term social cost minimization problem, and propose a holistic online mechanism to jointly control prompt dispatching, edge device selection, draft length configuration, and payment allocation, while overcoming the various algorithmic challenges. We prove that our approach possesses sub-linear regret and fit with guaranteed economic properties, and conduct solid experiments on our real-world testbed to validate the superiority of our design in practice.

Acknowledgment

This work was supported in part by the Basic Research Program of Jiangsu (No. BK20253011), the National Natural Science Foundation of China (No. 62441225 and No. 62572171), the Jiangsu Provincial Special Project for the Transformation of Scientific and Technological Achievements under Grant BA2023030, the Fundamental Research Funds for the Central Universities (No. B250201250), the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM118), the "111 Center" (No. B26023), and the U.S. National Science Foundation (CNS-2047719 and CNS-2225949). Corresponding authors are Lei Jiao (ljiao2@uoregon.edu), Bin Tang (cstb@hhu.edu.cn), and Zhihao Qu (quzhihao@hhu.edu.cn).

References

[1] Y. Venkatesha, S. Kundu, and P. Panda, "Fast and cost-effective speculative edge-cloud decoding with early exits," *arXiv:2505.21594*, 2025.
[2] J. Park, S. Cho, and D. Han, "Specedge: Scalable edge-assisted serving framework for interactive LLMs," in *NeurIPS*, 2025.
[3] C. Zheng and T. Yang, "Communication-efficient collaborative llm inference via distributed speculative decoding," *arXiv:2509.04576*, 2025.
[4] J. NING, C. ZHENG, and T. Yang, "Dssd: Efficient edge-device deployment and collaborative inference via distributed split speculative decoding," in *ICML Workshop*, 2025.
[5] D. Xu, W. Yin, and *et al*, "Llmcad: Fast and scalable on-device large language model inference," *arXiv:2309.04255*, 2023.
[6] Y. Jin, Z. Qian, S. Guo, S. Zhang, L. Jiao, and S. Lu, "runData: Redistributing data via piggybacking for geo-distributed data analytics over edges," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 1, pp. 40–55, 2021.
[7] Y. Jin, L. Jiao, Z. Qian, S. Zhang, and S. Lu, "Learning for learning: Predictive online control of federated learning with edge provisioning," in *IEEE INFOCOM*, 2021.
[8] Y. Zeng, R. Zhou, L. Jiao, Z. Han, J. Yu, and Y. Ma, "Efficient online dnn inference with continuous learning in edge computing," in *IEEE/ACM IWQoS*, 2024.
[9] "Edge Computing Hosting," https://www.cdnetworks.com/products/edge-computing-hosting/, 2026.
[10] M. Ji, H. Zhao, L. Jiao, S. Zhang, X. Li, Z. Qian, and B. Ye, "Edge ai inference as a service via dynamic resources from repeated auctions," *IEEE Transactions on Mobile Computing*, vol. 24, no. 9, pp. 7947–7964, 2025.
[11] Z. Wang, L. Gao, and J. Huang, "Socially-optimal mechanism design for incentivized online learning," in *IEEE INFOCOM*, 2022.
[12] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," in *ICML*, 2023.

[13] R. Meyer, "A class of nonlinear integer programs solvable by a single linear program," *SIAM Journal on Control and Optimization*, vol. 15, no. 6, pp. 935–946, 1977.
[14] M. Ji, L. Jiao, and *et al*, "Online scheduling of federated learning with in-network aggregation and flow routing," in *IEEE SECON*, 2024.
[15] N. Nisan and A. Ronen, "Computationally feasible vcg mechanisms," *Journal of Artificial Intelligence Research*, vol. 29, pp. 19–47, 2007.
[16] L. Zhang, Z. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," in *IEEE INFOCOM*, 2014.
[17] G. Qu, Q. Chen, and *et al*, "Mobile edge intelligence for large language models: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2025.
[18] M. Zhang, X. Shen, and *et al*, "Edgeshard: Efficient llm inference via collaborative edge computing," *IEEE Internet of Things Journal*, vol. 12, no. 10, pp. 13 119–13 131, 2025.
[19] F. You, H. Du, and *et al*, "Jppo: Joint power and prompt optimization for accelerated large language model services," in *IEEE ICC*, 2025.
[20] Y. Li, J. Guo, and *et al*, "Cloud-edge system for scheduling unpredictable llm requests with combinatorial bandit," *IEEE Transactions on Services Computing*, vol. 18, no. 6, pp. 3567–3580, 2025.
[21] Z. Yang, Y. Yang, and *et al*, "Perllm: Personalized inference scheduling with edge-cloud collaboration for diverse llm services," *arXiv:2405.14636*, 2024.
[22] F. You, H. Du, and *et al*, "Jppo++: Joint power and denoising-inspired prompt optimization for mobile llm services," *arXiv:2412.03621*, 2025.
[23] Y. Venkatesha, S. Kundu, and P. Panda, "Fast and cost-effective speculative edge-cloud decoding with early exits," *Transactions on Machine Learning Research*, 2025.
[24] Z. Hao, H. Jiang, and *et al*, "Hybrid slm and llm for edge-cloud collaborative inference," in *EdgeFM Workshop*, 2024.
[25] A. Dubey, Z. Feng, and *et al*, "Auctions with llm summaries," in *ACM KDD*, 2024.
[26] C. Zhao, Q. Hu, and *et al*, "Llm-auction: Generative auction towards llm-native advertising," *arXiv:2512.10551*, 2025.
[27] L. Cai, J. He, and *et al*, "Rtbagent: A llm-based agent system for real-time bidding," in *WWW*, 2025.
[28] D. Z. Huang, "Llm-based proxies for preference elicitation in combinatorial auctions," in *Harvard University Engineering and Applied Sciences*. Bachelors Thesis, 2024.
[29] W. Zhang, C. Zang, and *et al*, "Bid: Behavioral agents in dynamic auctions," in *ICLR Workshop*, 2025.
[30] M. Ji, Y. Jin, and *et al*, "Orchestrating in-network aggregation for distributed machine learning via in-band network telemetry," *Journal of Computer Science and Technology*, vol. 40, no. 1, pp. 196–214, 2025.
[31] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.
[32] "Specedge Dataset," https://github.com/kaist-ina/specedge, 2026.
[33] W. Zhao, X. Ren, and *et al*, "Wildchat: 1m chatgpt interaction logs in the wild," *arXiv:2405.01470*, 2024.
[34] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," *arXiv:2211.17192*, 2023.
[35] "California ISO," https://www.caiso.com/, 2026.
[36] "Planetlab," http://www.planet-lab.org/, 2026.
[37] M. Ji, Z. Qian, and B. Ye, "When cpn meets ai: Resource provisioning for inference query upon computing power network," in *IEEE ICPADS*, 2023.
[38] Y. Zhang, L. Jiao, and *et al*, "Toward market-assisted ai: Cloud inference for streamed data via model ensembles from auctions," *IEEE Transactions on Networking*, vol. 33, no. 2, pp. 793–806, 2025.
[39] "Welcome to CVXPY 1.1," https://www.cvxpy.org, 2025.
[40] S. Mizuno and F. Jarre, "Global and polynomial-time convergence of an infeasible-interior-point algorithm using inexact computation." *Mathematical Programming*, vol. 84, no. 1, 1999.
[41] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Transactions on Signal Processing*, vol. 65, no. 24, pp. 6350–6364, 2017.
[42] Y. Jin, L. Jiao, and *et al*, "Scheduling in-band network telemetry with convergence-preserving federated learning," *IEEE/ACM Transactions on Networking*, vol. 31, no. 5, pp. 2313–2328, 2023.
[43] R. Bhatia and C. Davis, "A cauchy-schwarz inequality for operators with applications," *Linear Algebra and Its Applications*, vol. 223, pp. 119–129, 1995.

APPENDIX

A. Proof of Lemma 1

We introduce the following two propositions:

Proposition 2. For $\min_{\boldsymbol{x}} \max_{i=1,\dots,k} f_i(\boldsymbol{x})$, where each $f_i(\boldsymbol{x})$ takes an integer value, the optimal solution of this optimization problem is the same as that of $\min_{\boldsymbol{x}} \sum_{i=1}^{k} \eta^{f_i(\boldsymbol{x})}$, where $\eta > k$ is an integer [13].

Proposition 3. Any convex function $f(x)$ can be rewritten as $f(x) = \sum_{h\in\mathcal{H}} f(h)\lambda_h$, where $\sum_{h\in\mathcal{H}} h\lambda_h = x$, $\sum_{h\in\mathcal{H}} \lambda_h = 1$, and $\lambda_h \geq 0$, $\forall h \in \mathcal{H}$. Here, $\mathcal{H}$ is the set of all possible values x can take [13].

Using **Proposition 2**, we transform $\max_{i\in\mathcal{I}_t, j\in\mathcal{J}} y_{i,j,t}d_{i,j,t}$ to $\sum_{i\in\mathcal{I}_t}\sum_{j\in\mathcal{J}} \eta^{y_{i,j,t}d_{i,j,t}}$. Consider **Proposition 3**. In our case, $\mathcal{H} = \{0, 1\}$; we introduce $\lambda^0_{i,j} \geq 0$ and $\lambda^1_{i,j} \geq 0$. Thus, from $\mathscr{P}_{t,1}$, we have

$$\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{I}_t}\sum_{j\in\mathcal{J}}\sum_{h\in\mathcal{H}} \eta^{h\cdot d_{i,j,t}}\lambda^h_{i,j} \\
\text{s.t.} \quad & \textstyle\sum_{h\in\mathcal{H}} h\lambda^h_{i,j} = y_{i,j,t},\ \forall i,j, \\
& \textstyle\sum_{h\in\mathcal{H}} \lambda^h_{i,j} = 1,\ \forall i,j, \\
& (1),\ (2), \\
\text{var.} \quad & 0 \leq \lambda^h_{i,j} \leq 1, \forall h \in \mathcal{H}, \forall i, \forall j,
\end{aligned}$$

which is further equivalent to

$$\begin{aligned}
\min \quad & \textstyle\sum_{i\in\mathcal{I}_t}\sum_{j\in\mathcal{J}} (\eta^{d_{i,j,t}} - 1)\lambda^1_{i,j} \\
\text{s.t.} \quad & \lambda^1_{i,j} = y_{i,j,t},\ \forall i,j, \\
& (1),\ (2), \\
\text{var.} \quad & 0 \leq \lambda^1_{i,j} \leq 1, \forall i, j.
\end{aligned}$$

The first constraint of the above problem allows us to directly remove $\lambda^1_{i,j}$ and keep $y_{i,j,t}$ as the decision variable. After such elimination, we obtain $\mathscr{P}^{LP}_{t,1}$.

Note that the constraint matrix of $\mathscr{P}^{LP}_{t,1}$ is *totally unimodular*. For Constraint (1) in $\mathscr{P}_{t,1}$, we can replace $\frac{B_j}{K_{\max}\gamma_{\max}r}$ by $\lfloor\frac{B_j}{K_{\max}\gamma_{\max}r}\rfloor$ on its right-hand side without impacting the optimal solution, because its left-hand side is guaranteed to be an integer. Now, as the right-hand sides of Constraints (1) and (2) are integers, the total unimodularity holds. Thus, the optimal solution to $\mathscr{P}^{LP}_{t,1}$ will automatically be integral.

B. Proof of Lemma 2

It can be verified that $\Gamma_{i,j,t}(\gamma_{j,t})$ for $\gamma_{j,t} \in [0, \gamma_{\max}]$ is convex. For the upper bound, we have the following derivations. Let $\tilde{\gamma}_{i,t} = n + \delta$, where n is the integer part $\lfloor\widetilde{\gamma}_{j,t}\rfloor$, and δ is the fractional part $\tilde{\gamma}_{i,t} - \lfloor\widetilde{\gamma}_{j,t}\rfloor$. Then, we define the gap:

$$\begin{aligned}
&J(\tilde{\gamma}_{i,t}) = \mathbb{E}[\Gamma_{i,j,t}(\bar{\gamma}_{i,t})] - \Gamma_{i,j,t}(\tilde{\gamma}_{i,t}) \\
&= (1-\delta)\Gamma_{i,j,t}(n) + \delta\Gamma_{i,j,t}(n+1) - \Gamma_{i,j,t}(n+\delta)
\end{aligned}$$

As $\Gamma_{i,j,t}(\cdot)$ is twice differentiable, we apply Taylor's theorem around $\tilde{\gamma}_{i,t}$:

$$\begin{aligned}
&\Gamma_{i,j,t}(n) = \Gamma_{i,j,t}(\tilde{\gamma}_{i,t}) + \Gamma'_{i,j,t}(\tilde{\gamma}_{i,t})(n - \tilde{\gamma}_{i,t}) \\
&+ \tfrac{1}{2}\Gamma''_{i,j,t}(\xi_1)(n - \tilde{\gamma}_{i,t})^2, \xi_1 \in [n, \tilde{\gamma}_{i,t}]. \qquad (5)
\end{aligned}$$

$$\Gamma_{i,j,t}(n+1) = \Gamma_{i,j,t}(\tilde{\gamma}_{i,t}) + \Gamma'_{i,j,t}(\tilde{\gamma}_{i,t})(n+1-\tilde{\gamma}_{i,t}),$$
$$+ \tfrac{1}{2}\Gamma''_{i,j,t}(\xi_2)(n+1-\tilde{\gamma}_{i,t})^2, \xi_2 \in [\tilde{\gamma}_{i,t}, n+1]. \quad (6)$$

We substitute $n-\tilde{\gamma}_{i,t} = -\delta$ and $n+1-\tilde{\gamma}_{i,t} = 1-\delta$ into (5):

$$\begin{aligned} J(\tilde{\gamma}_{i,t}) &= \mathbb{E}[\Gamma_{i,j,t}(\bar{\gamma}_{i,t})] - \Gamma_{i,j,t}(\tilde{\gamma}_{i,t}) \\ &\overset{7a}{=} \tfrac{1}{2}(1-\delta)\delta[\delta\Gamma''_{i,j,t}(\xi_1) + (1-\delta)\Gamma''_{i,j,t}(\xi_2)] \\ &\leq \tfrac{1}{2}(1-\delta)\delta[\delta M + (1-\delta)M] = \tfrac{1}{2}(1-\delta)\delta M \overset{7b}{\leq} \tfrac{1}{8}M, \end{aligned} \quad (7)$$

where (7a) holds since $\Gamma_{i,j,t}(\cdot)$ is convex, i.e., $\Gamma''_{i,j,t}(\xi_1) \geq 0$, $\Gamma''_{i,j,t}(\xi_2) \geq 0$, and $M \triangleq \max_{\xi' \in [n,n+1]} \Gamma''_{i,j,t}(\xi')$; (7b) holds since $\delta + (1-\delta) = 1$, and from Cauchy-Schwarz Inequality [43], we have $\delta(1-\delta) \leq \frac{1}{4}$.

C. Proof of Theorem 2

For t, we use $\bar{\mathcal{X}}_t$ to denote the integer decision produced by our online approach, and $\mathcal{X}_t^*$ to denote the optimal integer decision; we use $\widetilde{\mathcal{X}}_t$ to denote the fractional decision produced by our online approach, and $\widetilde{\mathcal{X}}_t^*$ to denote the fractional optimal decision. Now, we define the fractional regret and fit:

$$\widetilde{\boldsymbol{Reg}}(T) = \textstyle\sum_{t=1}^T \mathcal{P}_t(\widetilde{\mathcal{X}}_t) - \sum_{t=1}^T \mathcal{P}_t(\widetilde{\mathcal{X}}_t^*),$$
$$\widetilde{\boldsymbol{Fit}}(T) = ||\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\widetilde{\mathcal{X}}_t)||.$$

For *regret*, we have the following derivations:

$$\begin{aligned} \boldsymbol{Reg} &= E[\textstyle\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\mathcal{X}_t^*) \\ &\overset{8a}{=} \textstyle\sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) - \sum_t \mathcal{P}_t(\mathcal{X}_t^*) + E[\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) \\ &\overset{8b}{\leq} \textstyle\sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t^*) + E[\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) \\ &\overset{8c}{=} \widetilde{\boldsymbol{Reg}} + \mathcal{K}(k_t) \overset{8d}{\leq} \mathcal{O}(T^{\tau_1}), \end{aligned}$$

where $\tau_1 \in (0,1)$ is a constant. (8a) holds as we introduce the redundant terms. (8b) holds because the integer optimum of a minimization problem is always no less than the fractional optimum of the relaxed problem. From (8b) to (8c), on one hand, we apply **Proposition 1**; on the other hand, as proven right next, we will have $E[\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) \leq \mathcal{K}(k_t)$. Finally, (8d) holds as $\mathcal{K}(k_t)$ is constant.

Now, to prove $E[\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) \leq \mathcal{K}(k_t)$, we conduct the following. As defined before, $\mathcal{P}_{t,1}(\cdot)$ and $\mathcal{P}_{t,2}(\cdot)$ represent the objectives of $\mathscr{P}_{t,1}$ and $\mathscr{P}_2$. We further define

$$\mathcal{P}_t(\cdot) = \mathcal{P}_{t,1}(\cdot) + \mathcal{P}_{t,2}(\cdot), \;\; \mathcal{P}_{t,2}(\cdot) = \mathcal{P}'_{t,2}(\cdot) + \mathcal{P}''_{t,2}(\cdot), \quad (8)$$

where $\mathcal{P}'_{t,2}(\bar{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t) \triangleq \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}_t} \bar{y}_{i,j,t}\Gamma_{i,j,t}$, $\mathcal{P}''_{t,2}(\boldsymbol{x}_t) \triangleq \sum_{j\in\mathcal{J}} x_{j,t}b_{j,t}$. Then, we have

$$\begin{aligned} E[\textstyle\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] &\leq E[\textstyle\sum_t\{\mathcal{P}_{t,1}(\bar{\boldsymbol{y}}_t) + \mathcal{P}_{t,2}(\bar{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \bar{\boldsymbol{\gamma}}_t)\}] \\ &\overset{9a}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\bar{\boldsymbol{y}}_t) + E[\mathcal{P}_{t,2}(\bar{\boldsymbol{x}}_t, \bar{\boldsymbol{y}}_t, \bar{\boldsymbol{\gamma}}_t)]\} \\ &\overset{9b}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + E[\mathcal{P}_{t,2}(\bar{\boldsymbol{x}}_t, \widetilde{\boldsymbol{y}}_t, \bar{\boldsymbol{\gamma}}_t)]\} \\ &\overset{9c}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + E[\mathcal{P}'_{t,2}(\widetilde{\boldsymbol{y}}_t, \bar{\boldsymbol{\gamma}}_t)] + E[\mathcal{P}''_{t,2}(\bar{\boldsymbol{x}}_t)]\} \\ &\overset{9d}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + \mathcal{P}'_{t,2}(\widetilde{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t) + \frac{M}{8} + E[\mathcal{P}''_{t,2}(\bar{\boldsymbol{x}}_t)]\} \\ &\overset{9e}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + \mathcal{P}'_{t,2}(\widetilde{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t) + \frac{M}{8} + k_t\}. \end{aligned} \quad (9)$$

(9a) holds because **Algorithm 1** is deterministic; (9b) holds due to **Lemma 1**; (9c) holds because of (8); (9d) holds by **Lemma 2**; (9e) holds since $\sum_t \sum_j \bar{x}_{j,t}b_{j,t} \leq \sum_t b_{max,t}|\mathcal{J}| \triangleq \sum_t k_t$, where $k_t \triangleq b_{max,t}|\mathcal{J}|$ and $b_{max,t} = \max\{b_{j,t}, j \in \mathcal{J}\}$.

$$\begin{aligned} &E[\textstyle\sum_t \mathcal{P}_t(\bar{\mathcal{X}}_t)] - \sum_t \mathcal{P}_t(\widetilde{\mathcal{X}}_t) \overset{10a}{\leq} \textstyle\sum_t\{\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + \mathcal{P}'_{t,2}(\widetilde{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t) \\ &+ \frac{M}{8} + k_t - (\mathcal{P}_{t,1}(\widetilde{\boldsymbol{y}}_t) + \mathcal{P}'_{t,2}(\widetilde{\boldsymbol{y}}_t, \widetilde{\boldsymbol{\gamma}}_t) + \mathcal{P}''_{t,2}(\widetilde{\boldsymbol{x}}_t))\} \\ &= \textstyle\sum_t\{\frac{M}{8} + k_t - \mathcal{P}''_{t,2}(\widetilde{\boldsymbol{x}}_t)\} \overset{10b}{\leq} \sum_t\{\frac{M}{8} + k_t\} \triangleq \mathcal{K}(k_t). \end{aligned} \quad (10)$$

(10a) holds due to (8) and (9). (10b) holds due to $\mathcal{P}''_{t,2}(\widetilde{\boldsymbol{x}}_t) \geq 0$.

For *fit*, we have the derivations as follows:

$$\begin{aligned} \boldsymbol{Fit}(T) &= ||[E[\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\bar{\mathcal{X}}_t)]]^+|| \overset{11a}{\leq} ||E[\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\bar{\mathcal{X}}_t)]|| \\ &\overset{11b}{=} ||\textstyle\sum_{t=1}^T E[\boldsymbol{g}_t(\bar{\mathcal{X}}_t)]|| \overset{11c}{\leq} ||\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\widetilde{\mathcal{X}}_t)|| \triangleq \widetilde{\boldsymbol{Fit}}(T) \\ &\overset{11d}{\leq} \mathcal{O}(T^{\tau_2}), \end{aligned}$$

where $\tau_2 \in (0,1)$ is a constant. (11a) holds because $||[\cdot]^+|| \leq ||\cdot||$; (11b) holds because the expectation of the sum equals the sum of individual expectations; (11c) holds due to the derivation below; (11d) holds due to **Proposition 1**.

$$\begin{aligned} &E[\bar{x}_{j,t}] \geq \widetilde{x}_{j,t} \Rightarrow -E[\bar{x}_{j,t}] \leq -\widetilde{x}_{j,t} \\ &\Rightarrow \textstyle\sum_{t=1}^T\{\frac{1}{T}\varphi_j - E[\bar{x}_{j,t}]\} \leq \sum_{t=1}^T\{\frac{1}{T}\varphi_j - \widetilde{x}_{j,t}\} \\ &\Rightarrow ||E[\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\bar{\mathcal{X}}_t)]|| \leq ||\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\widetilde{\mathcal{X}}_t)|| \\ &\Rightarrow E[\textstyle\sum_{t=1}^T \boldsymbol{g}_t(\bar{\mathcal{X}}_t)] \leq \sum_{t=1}^T \boldsymbol{g}_t(\widetilde{\mathcal{X}}_t). \end{aligned}$$

D. Proof of Theorem 3

We show that the three conditions are satisfied respectively.

(i) Let $\Lambda^t(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\gamma}_t)$ denote the objective function of the problem $\mathscr{P}$. Given the inputs of all other bids, i.e., $\mathbf{b}_{-j,t}$, we adjust the bid j's bidding price and observe the corresponding decision $x_{j,t}$ and the objective value. Hence, the function can be written as $\Lambda_j^t(x_{j,t}, b_{j,t} \mid \mathbf{b}_{-j,t})$. Assume that $\widetilde{x}_{j,t}^0$ and $\widetilde{x}_{j,t}^1$ are the optimal decisions corresponding to the bidding prices $b_{j,t}^0$ and $b_{j,t}^1$, respectively. Then we have

$$\Lambda_j^t(\widetilde{x}_{j,t}^0, b_{j,t}^0|\mathbf{b}_{-j,t}) \leq \Lambda_j^t(\widetilde{x}_{j,t}^1, b_{j,t}^0|\mathbf{b}_{-j,t}),$$
$$\Lambda_j^t(\widetilde{x}_{j,t}^1, b_{j,t}^1|\mathbf{b}_{-j,t}) \leq \Lambda_j^t(\widetilde{x}_{j,t}^0, b_{j,t}^1|\mathbf{b}_{-j,t}).$$

Summing up these two inequalities, we have $(b_{j,t}^0 - b_{j,t}^1) * \widetilde{x}_{j,t}^0 \leq (b_{j,t}^0 - b_{j,t}^1) * \widetilde{x}_{j,t}^1$. Without loss of generality, assuming $b_{j,t}^0 - b_{j,t}^1 > 0$, we get $\widetilde{x}_{j,t}^0 \leq \widetilde{x}_{j,t}^1$. Then, consider three cases: a) If $\widetilde{x}_{j,t}^0 = 0$ and $\widetilde{x}_{j,t}^1 = 0$, after rounding we have $\bar{x}_{j,t}^0 = \bar{x}_{j,t}^1 = 0$; b) If $\widetilde{x}_{j,t}^0 = 0$ and $\widetilde{x}_{j,t}^1 \in (0,1]$, after rounding we have $\bar{x}_{j,t}^0 = 0 \leq 1 = \bar{x}_{j,t}^1$; c) If $\widetilde{x}_{j,t}^0 \in (0,1]$ and $\widetilde{x}_{j,t}^1 \in (0,1]$, after rounding we have $\bar{x}_{j,t}^0 = \bar{x}_{j,t}^1 = 1$. That is, in all cases, we have $\bar{x}_{j,t}^0 \leq \bar{x}_{j,t}^1$.

(ii) Note that we have

$$\int_{b_{j,t}}^{\infty} \bar{x}_{j,t}(s, \mathbf{b}_{-j,t})\mathrm{d}s = \int_{b_{j,t}}^{\chi_t} \bar{x}_{j,t}(s, \mathbf{b}_{-j,t})\mathrm{d}s \leq \chi_t < \infty.$$

(iii) Our **Algorithm 4** directly follows the third condition stated in the theorem to calculate the payment.