

Online LLM Inference Routing for Multi-Agent Systems under Long-Term Budget and Layered Privacy Constraints

Shijing Yuan, *Member, IEEE*, Zihang Li, and Yuming Zhang^(✉), *Member, IEEE*

Abstract The Cloud-Edge-End (CEE) collaborative architecture has become a key paradigm for deploying highly intelligent multi-agent systems with low latency and high energy efficiency. Existing studies on efficient task offloading in CEE, however, either overlook long-term financial budget constraints from commercial API token usage and edge computing leasing, or inadequately handle layered information-sharing and privacy restrictions in decentralized autonomous agents. This paper investigates online inference routing to minimize the computational and communication overhead of large language model (LLM) workloads under long-term budget constraints, agent intent-sharing limitations, and unknown future prompt lengths. We propose a budget-aware online optimization framework that uses Lyapunov optimization to decouple long-term financial constraints, and show that the decoupled problem forms a generalized ordinal potential game. To address heterogeneous information-sharing restrictions, we design an adaptive best response algorithm for cooperative environments with shared decision information, and a multi-agent distributed routing algorithm for limited-information environments, enabling efficient offloading without inter-agent intent sharing. Theoretical analysis provides conditional performance and queue-length bounds and establishes finite convergence of the per-slot solvers. Simulations on distributed inference testbeds show that our scheme reduces the inference overhead of heterogeneous agentic tasks while achieving a favorable trade-off between performance and actual token expenditure.

Keywords Cloud-Edge-End collaboration, large language models, multi-agent systems, online offloading, long-term budget constraints, privacy limitations

1 Introduction

Recent studies on communication-computing integrated wireless networks further highlight the transition from cloud-based collaboration toward sensing- and intelligence-driven coordination across wireless CEE systems [1]. At the infrastructure layer, embedded data processing units and service function chain orchestration provide complementary support for programmable edge-side data processing and service composition [2].

The rapid proliferation of LLMs [3–5] has catalyzed the transformation of end devices into highly intelligent autonomous agent platforms [6]. Central to this paradigm, the CEE collaborative architecture offers immense potential for managing complex multi-agent inference workloads [7, 8]. While data centers dominate the training of massive proprietary LLMs, the ubiquitous influx of user-generated prompt streams necessitates efficient distributed inference to ensure low latency, high energy efficiency, and superior user experiences. In the realm of CEE collaboration, autonomous agents must make critical routing decisions for each reasoning task: executing locally on the device via SLMs, offloading to mid-tier open-source LLMs hosted on nearby edge servers, or querying massive proprietary models via remote cloud APIs.

Recent architectural innovations, such as memory-based expert knowledge injection, have demonstrated the viability of utilizing abundant device ROM to expand model capacity locally without escalating active floating-point operations (FLOPs) or RAM usage. This provides a compelling foundation for flagship-level intelligence on edge devices. However, the heterogeneous nature of agentic tasks—ranging from simple intent recognition to extensive context summarization and complex logical reasoning—often exceeds the bounded capabilities of local SLMs. Consequently, agents must continuously evaluate the necessity of offloading extensive context and computation to robust edge or cloud resources.

Routing inference requests across the CEE hierarchy introduces two critical challenges. First, commercial cloud APIs and edge compute leasing are billed strictly based on token usage or computational volume. These recurrent operational costs impose stringent long-term financial budget constraints on users or enterprises deploying agent platforms. Given the

• S. Yuan is with China Telecom Research Institute (CTRI), and Shanghai Jiao Tong University, Shanghai 200120, China. E-mail: dr.sj.yuan@ieee.org.

• Zihang Li is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: lzh2006@sjtu.edu.cn.

• Yuming Zhang is with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China. E-mail: ymzhang3@bjtu.edu.cn.

* Corresponding author: Yuming Zhang.

Manuscript received: 2026-xx-xx; revised: 20xx-11-xx; accepted: 20xx-00-xx

unpredictable nature of prompt lengths and the prevalence of pay-as-you-go business models, formulating an efficient inference routing strategy that minimizes latency and energy overhead without violating long-term budgets constitutes a formidable research problem [7]. Second, in multi-agent environments, different agents possess varying privacy requirements and information-sharing limitations [6, 9]. While enterprise-managed cooperative agents may freely share their routing intents to globally mitigate edge GPU contention, privacy-centric personal agents may restrict routing-intent disclosure, and some security-critical requests may be forced to remain local because their raw context cannot leave the device.

Existing works on CEE task scheduling primarily focus on short-term latency optimization and largely overlook the intertwined dynamics of long-term token budgets and heterogeneous privacy constraints in modern LLM routing scenarios. To bridge this gap, this paper proposes an online routing and resource allocation framework for heterogeneous multi-agent LLM inference under long-term budget constraints within a CEE architecture. Addressing this requires navigating unpredictable network conditions, fluctuating prompt lengths, and finite edge GPU resources without relying on future knowledge.

By modeling the inference costs as weighted functions of latency and energy consumption, we deploy the Lyapunov optimization method to construct a Budget-aware Online Offloading and Resource Allocation Framework (BOOF). This framework effectively decomposes the long-term stochastic API and leasing budget limits into real-time, single-time-slot routing sub-problems. To tackle the complexities of edge resource contention, each sub-problem is formally modeled as a non-cooperative game. We demonstrate that this formulation constitutes a Generalized Ordinal Potential Game (GOPG), which guarantees the existence of a pure strategy Nash Equilibrium (NE). Furthermore, to accommodate layered information-sharing constraints, we design two distinct routing algorithms: the Best Response Adaptive Offloading (BRAO) algorithm for cooperative environments with intent-sharing capabilities, and the Multi-Agent Distributed Offloading (MADO) algorithm, which enables privacy-sensitive agents to learn stable routing decisions through environmental feedback without inter-agent intent broadcasting.

Our main contributions are summarized as follows:

- Focusing on online LLM inference routing under CEE collaboration, we construct a long-term stochastic optimization problem that minimizes the computational overhead of heterogeneous agentic tasks, fully incorporating practical challenges such as long-term financial budgets, unpredictable token workloads, and multi-layered privacy limitations.
- We propose the BOOF framework, which employs Lyapunov optimization to dynamically decouple the long-term budget constraints into sequential sub-problems. We systematically prove that these sub-problems form a GOPG with verifiable stability properties^①.
- In response to diverse privacy and information-sharing limits, we introduce two tailored algorithms. The BRAO algorithm ensures finite NE convergence in cooperative settings under asynchronous better-response updates, while the MADO algorithm facilitates efficient decentralized routing via projected stochastic policy iteration without inter-agent intent sharing. Theoretical analysis provides conditional performance bounds and convergence properties under explicit feasibility and boundedness assumptions.
- Extensive evaluations using distributed LLM inference testbeds demonstrate that our scheme significantly reduces inference latency and energy costs compared to existing baselines, achieving a robust trade-off between task performance and actual token expenditure.

The remainder of this paper is organized as follows: Section 2 reviews existing related work. Section 3 details the long-term optimization problem. The problem transformation under the BOOF framework and the design of the algorithms are presented in Section 4 and Section 5 respectively. Experimental setup and results are discussed in Section 6. Section 7 concludes this paper.

2 Related Work

Most related work focuses on the joint optimization of task offloading and resource allocation to achieve efficient scheduling in CEE collaborative computing. Addressing the coupled relationship between computation offloading and resource provisioning, various joint optimization algorithms have been proposed. These algorithms typically model the scheduling problem as an optimization framework containing mixed variables and non-convex items.

For instance, Wang et al. [10] and Qiang et al. [11] utilized online optimization frameworks combining Lyapunov optimization and approximation methods to guide computation offloading and bandwidth allocation with unpredictable future information. Yuan et al. [12] proposed an online framework integrating generalized Benders decomposition to simultaneously determine offloading modes and task compression

^① The full source code of BOOF and the simulation environment will be made publicly available upon the acceptance of this paper.

dynamically. Gholivand et al. [13] proposed an end-to-end communication and offloading architecture showing advantages in execution time and energy consumption. Bin et al. [14] transformed the joint optimization problem into a Markov Decision Process utilizing deep reinforcement learning. Yan et al. [15] classified CEE into hierarchical and horizontal architectures, proposing game-based algorithms to optimize offloading strategies. Sun et al. [16] proposed a low-complexity hierarchical heuristic method utilizing a closed-form approach based on the Cauchy-Schwarz inequality for effective resource allocation.

Recently, the deployment of foundation models at the edge has driven new optimization paradigms [3, 5]. Efficient on-device inference architectures, such as memory-based expert knowledge injection, have proposed leveraging ROM bandwidth to expand model capacity without increasing active FLOPs or RAM usage, providing a robust new dimension for local task execution. Furthermore, offloading LLM inference to edge servers presents unique challenges requiring novel solutions. For instance, Chen et al. [7] utilized an active inference approach to address LLM offloading in cloud-edge computing, significantly improving latency and resource efficiency. Li et al. [4] developed a QoS-aware LLM routing framework tailored for edge computing with multiple experts.

Concurrently, multi-agent methodologies have been increasingly adopted to handle decentralized scheduling [6, 9]. In cloud-edge-end scenarios, multi-agent deep reinforcement learning has also been explored for edge caching, demonstrating how decentralized agents can adapt resource placement to dynamic end-edge-cloud interactions [17]. Moreover, Lyapunov optimization continues to be heavily relied upon to balance stability, budget constraints, and task accuracy in modern edge computing scenarios, such as UAV-MEC networks [18] and decentralized task offloading applications [19].

Our proposed framework extends previous work significantly because: 1) we specifically address the modern challenges of multi-agent LLM inference routing, constructing a long-term stochastic optimization problem incorporating financial budget constraints tied to API token usage and edge computing leasing; 2) we introduce an online optimization framework combining Lyapunov methods with potential game theory, utilizing an expenditure deficit queue to decouple budget constraints against unpredictable context workloads; and 3) we propose dual algorithms (BRAO and MADO) tailored for layered information-sharing and intent-sharing limitations in contemporary autonomous agent ecosystems, with explicit convergence properties and performance bounds under

varying degrees of information availability.

3 System Model

The collaborative framework considered in this study addresses the deployment of autonomous agents within a CEE hierarchy. Unlike traditional computational offloading that focuses on generic data packets, this model specifically characterizes the inference process of LLMs. The objective is to minimize the aggregate performance overhead across all agents while maintaining strict adherence to long-term financial constraints imposed by commercial API usage and edge resource leasing.

3.1 System Overview

The system operates over a discrete timeline divided into a series of time slots $\mathcal{T} = \{0, \dots, T-1\}$. We consider a set of agents $\{ED_n\}_{n \in \mathcal{N}}$, where $\mathcal{N} = \{1, \dots, N\}$ denotes the index set of end devices. These agents are supported by a cluster of edge servers $\{ES_m\}_{m \in \mathcal{M}}$ within the set $\mathcal{M} = \{1, \dots, M\}$ and a remote cloud platform. In this architecture, edge servers host mid-tier open-source LLMs to provide low-latency localized inference, while the cloud platform offers access to high-capacity proprietary LLM APIs.

At each time slot t , each agent ED_n generates an inference request HT_n^t characterized by the tuple $\langle S_n^t, C_n^t, \lambda_n^E, \lambda_n^L \rangle$. Here, S_n^t is the total serialized data size of one request and its returned result, measured in bits, and C_n^t is the required CPU cycles per bit. When separate uplink and downlink measurements are available, the effective rate used later is calibrated so that $S_n^t / r_{n,m}^t$ equals the measured complete transfer time. The parameters λ_n^E and λ_n^L are weighting factors reflecting the agent's preference for energy conservation and latency performance, respectively. For each request, the agent must determine a routing strategy $x_n^t = \{x_{n,l}^t, \{x_{n,m}^t\}_{m \in \mathcal{M}}, x_{n,c}^t\}$. The binary variables $x_{n,l}^t$, $x_{n,m}^t$, and $x_{n,c}^t$ indicate whether the inference is executed on the local SLM, an edge-hosted LLM, or a cloud-based proprietary API. To ensure exclusive execution, these variables must satisfy the constraint:

$$x_{n,l}^t + \sum_{m \in \mathcal{M}} x_{n,m}^t + x_{n,c}^t = 1, \quad \forall n \in \mathcal{N}, t \in \mathcal{T}. \quad (1)$$

LLM routing is also subject to a task-dependent feasible action set. Let $q_{n,a}^t$ denote the task-dependent quality score of routing action $a \in \{l, 1, \dots, M, c\}$, and let q_n^t be the minimum acceptable quality for request HT_n^t . Let $\mathcal{A}_{n,\text{loc}}^t$ denote the actions permitted by the task's data-location rule.

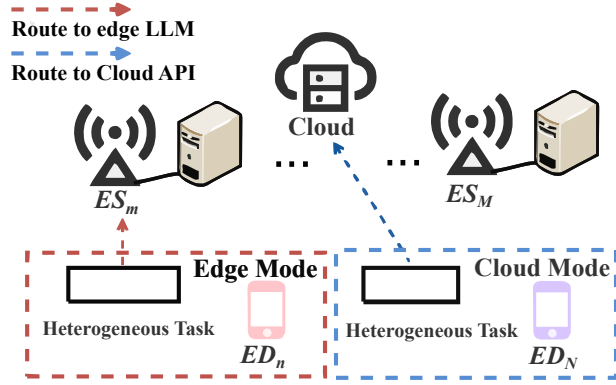


Fig. 1 Schematic diagram of heterogeneous multi-agent LLM inference routing under CEE collaboration.

We define

$$\psi_n^t = \left\{ a \in \{l, 1, \dots, M, c\} : q_{n,a}^t \geq \underline{q}_n^t, a \in \mathcal{A}_{n,\text{loc}}^t \right\}. \quad (2)$$

The set only removes infeasible actions; it does not change the latency, energy, price, or queue equations below. For example, security-critical requests whose raw context cannot leave the device are modeled by $\mathcal{A}_{n,\text{loc}}^t = \{l\}$ and hence $\psi_n^t = \{l\}$. Less sensitive requests may admit edge or cloud actions when their quality scores satisfy the threshold. The main experiments use the full action set $\psi_n^t = \{l, 1, \dots, M, c\}$ and evaluate routing cost and budget behavior under quality feasibility rather than answer-quality improvement. This paper treats privacy as layered information-sharing and data-placement restrictions rather than a cryptographic privacy guarantee.

3.2 Inference Modes and Performance Metrics

The performance of LLM inference is governed by the interaction between communication latency, computational delay, and the energy consumed during data transmission and local processing. In this subsection, we discuss these three computing modes and their performance characteristics.

3.2.1 Local Mode

In this mode ($x_{n,l}^t = 1$), the agent utilizes its local compute capabilities $f_{n,l}^t$ to execute a memory-augmented SLM. The inference latency $\ell_{n,l}^t$ is defined by the ratio of total required operations to the available clock frequency:

$$\ell_{n,l}^t = \frac{S_n^t C_n^t}{f_{n,l}^t}. \quad (3)$$

The corresponding local energy consumption $e_{n,l}^t$ is modeled based on the effective capacitance of the computing chip $\tilde{\kappa}_{n,l}$ and the squared frequency of the processor:

$$e_{n,l}^t = \tilde{\kappa}_{n,l} S_n^t C_n^t (f_{n,l}^t)^2. \quad (4)$$

3.2.2 Edge Mode

When offloading to ES_m ($x_{n,m}^t = 1$), the request and returned result are transferred over a shared wireless channel. Let $R_{n,m}^t$ be the effective peak rate available to ED_n for the complete request-response transfer when occupying the channel exclusively. The actual effective rate $r_{n,m}^t$ is allocated based on a weight factor γ_n^t as:

$$r_{n,m}^t = \frac{\gamma_n^t}{\sum_{j \in \mathcal{N}} \gamma_j^t x_{j,m}^t} R_{n,m}^t. \quad (5)$$

Similarly, the computational resource $f_{n,m}^t$ provided by the edge GPU cluster $\tilde{f}_m^t$ is shared among concurrent requests:

$$f_{n,m}^t = \frac{\gamma_n^t}{\sum_{j \in \mathcal{N}} \gamma_j^t x_{j,m}^t} \tilde{f}_m^t. \quad (6)$$

The computational delay $\tilde{\ell}_{n,m}^{\text{com},t}$ includes a startup overhead $a_{n,m}^t$ related to key-value cache initialization:

$$\tilde{\ell}_{n,m}^{\text{com},t} = \frac{S_n^t C_n^t}{f_{n,m}^t} + a_{n,m}^t. \quad (7)$$

The communication delay covers the complete request and returned result. With the effective rate $r_{n,m}^t$ measured over that transfer,

$$\tilde{\ell}_{n,m}^{\text{del},t} = \frac{S_n^t}{r_{n,m}^t}. \quad (8)$$

Thus $S_n^t / r_{n,m}^t$ is the end-to-end communication time of one request-response transaction. The corresponding end-device energy uses the effective average power $\tilde{p}_n^{ES}$ during the same transfer:

$$\tilde{e}_{n,m}^t = \tilde{p}_n^{ES} \tilde{\ell}_{n,m}^{\text{del},t}. \quad (9)$$

3.2.3 Cloud Mode

Routing to a cloud API ($x_{n,c}^t = 1$) leverages massive remote resources $\hat{f}_c^t$ but incurs delays across both wired ($\hat{R}_{n,c}^t$) and wireless ($\hat{R}_{n,c}'^t$) segments. The total transmission latency $\hat{\ell}_{n,c}^{\text{del},t}$ is written as $S_n^t / \hat{R}_{n,c}^t + S_n^t / \hat{R}_{n,c}'^t$, where the two rates are effective rates for the complete request and return path. The inference latency $\hat{\ell}_{n,c}^{\text{com},t}$ incorporates the API connection and initial generation overhead $a_{n,c}^t$:

$$\hat{\ell}_{n,c}^{\text{com},t} = \frac{S_n^t C_n^t}{\hat{f}_c^t} + a_{n,c}^t. \quad (10)$$

The energy consumption $\hat{e}_{n,c}^t$ depends on the transmission power $\tilde{p}_n^{\text{cloud}}$ and the total communication time:

$$\hat{e}_{n,c}^t = \tilde{p}_n^{\text{cloud}} \hat{\ell}_{n,c}^{\text{del},t}. \quad (11)$$

The energy terms in the objective are end-device energy evaluated from the terminal viewpoint. Local execution includes the terminal's computation energy, while edge and cloud execution include the terminal energy used to transfer the request and receive the result, together with the monetary

service charge paid by the terminal. Electricity consumed by provider GPUs is a provider-side operating cost and is outside the terminal-centric metric evaluated in this paper. The monetary terms $b_{n,ES}^t$ and $b_{n,c}^t$ are budget variables for remote inference expenditure, not physical measurements of server energy.

3.3 Computational Costs and Financial Constraints

The comprehensive computational cost C_n^t for agent ED_n is defined as the weighted sum of latency and energy metrics across the three possible modes:

$$C_n^t = (\lambda_n^L \ell_{n,l}^t + \lambda_n^E c_{n,l}^t) x_{n,l}^t + \sum_{m \in \mathcal{M}} \left(\lambda_n^L (\tilde{\ell}_{n,m}^{com,t} + \tilde{\ell}_{n,m}^{del,t}) + \lambda_n^E \tilde{c}_{n,m}^t \right) x_{n,m}^t + \left(\lambda_n^L (\hat{\ell}_{n,c}^{com,t} + \hat{\ell}_{n,c}^{del,t}) + \lambda_n^E \hat{c}_{n,c}^t \right) x_{n,c}^t. \quad (12)$$

Beyond performance metrics, the system must account for the financial expenditures associated with external inference services. In the edge mode, the leasing price is sensitive to user density: under proportional sharing, each user receives less bandwidth and compute capacity as the server load grows, so the provider applies a service-quality rebate. The expenditure $b_{n,ES}^t$ for agent ED_n is:

$$b_{n,ES}^t = \sum_{m \in \mathcal{M}} \left[\vartheta_{ES}^t \left(1 - \alpha \sum_{j \in \mathcal{N}} \gamma_j^t x_{j,m}^t \right) C_n^t S_n^t x_{n,m}^t \right], \quad (13)$$

where ϑ_{ES}^t is the base price per unit of computational volume and $\alpha \geq 0$ is chosen so that $1 - \alpha \sum_{j \in \mathcal{N}} \gamma_j^t x_{j,m}^t \geq \delta > 0$ for every feasible load. Thus every reported charge remains positive. In the cloud mode, the expenditure $b_{n,c}^t$ follows a fixed unit-price model with unit price ϑ_{cloud}^t :

$$b_{n,c}^t = \vartheta_{cloud}^t C_n^t S_n^t x_{n,c}^t. \quad (14)$$

The total instantaneous expenditure is $b_n^t = b_{n,ES}^t + b_{n,c}^t$. Each agent is subject to a long-term budget limit B_{max} , which constrains the time-averaged expenditure over the infinite horizon:

$$\lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=0}^{T-1} b_n^t \leq B_{max}, \quad \forall n \in \mathcal{N}. \quad (15)$$

3.4 Problem Formulation

The objective of this study is to determine an online routing policy $\mathbf{x}$ that minimizes the long-term average computational cost for the multi-agent system without prior knowledge of future task parameters or network states. Based on the preceding definitions, the stochastic optimization problem $\mathcal{P}_1$

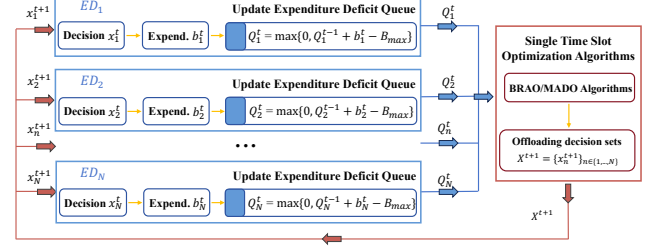


Fig. 2 Expenditure deficit queue update process.

is formulated as:

$$\begin{aligned} \mathcal{P}_1 : \quad & \min_{\mathbf{x}} \lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{n \in \mathcal{N}} C_n^t(x_n^t) \\ \text{s.t.} \quad & C_1 : x_{n,l}^t + \sum_{m \in \mathcal{M}} x_{n,m}^t + x_{n,c}^t = 1, \quad \forall n, t, \\ & C_2 : x_n^t \in \psi_n^t, \quad x_{n,l}^t, x_{n,m}^t, x_{n,c}^t \in \{0, 1\}, \quad \forall n, m, t, \\ & C_3 : \lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=0}^{T-1} b_n^t \leq B_{max}, \quad \forall n. \end{aligned} \quad (16)$$

The complexity of $\mathcal{P}_1$ arises from the inter-temporal coupling of decisions caused by the long-term budget constraint C_3 , the non-convex nature of the objective function under shared resource contention, the quality/privacy feasible set C_2 , and the dynamic variability of inference workloads S_n^t and C_n^t . To address these challenges, we transition to an online optimization framework in the subsequent sections.

4 Budget-aware Online Optimization Framework

In this section, we discuss the transformation of the optimization problem using the Lyapunov approach, the balance of the inference cost and token expenditure, and subsequently propose the budget-aware online optimization framework. This framework utilizes the Lyapunov optimization method to decompose the original long-term optimization problem into a series of single time-slot optimization sub-problems. Under standard feasibility, boundedness, and per-slot approximation conditions, the framework provides asymptotic performance and queue-length bounds for long-term budget control.

4.1 Problem Transformation Using Lyapunov

To decouple long-term financial budget constraints associated with LLM API usage and edge compute leasing, we transform the constraint into a virtual queue stability problem. For each agent ED_n , an *Expenditure Deficit Queue* (Q_n) is defined to track the financial deficit of ED_n . It is initialized at $Q_n = 0$ and updated in each subsequent time slot according to:

$$Q_n^{t+1} = [Q_n^t + b_n^t - B_{max}]^+, \quad (17)$$

where $[\cdot]^+$ denotes the operation of taking the maximum between the variable x and 0 in the real domain, and B_{max} signifies the long-term average maximum budget allocated to the agents. Figure 2 illustrates the update process of the deficit queue based on Lyapunov optimization. The magnitude of Q_n quantifies the cumulative expenditure deficit of ED_n over the historical period $[0, t-1]$. A larger Q_n value indicates a more substantial exceedance of the long-term average budget by the device's actual expenditure, compelling the agent to shift towards cost-effective local SLM processing in subsequent time slots. Theorem 1 states how queue stability implies budget feasibility; the queue bound itself is derived later in Theorem 3.

Theorem 1 (Budget Feasibility under Queue Stability). *If the virtual queue Q_n is mean rate stable for any $n \in \mathcal{N}$, then the corresponding long-term budget constraint is satisfied.*

Proof. Consider the definition of mean rate stability from [20], i.e., $\lim_{t \rightarrow +\infty} \mathbb{E}[Q_n^t]/t = 0$. According to the update rule for Q_n :

$$Q_n^{t+1} \geq Q_n^t + b_n^t - B_{max}, \quad (18)$$

$$\Rightarrow b_n^t - B_{max} \leq Q_n^{t+1} - Q_n^t. \quad (19)$$

Summing from $\tau = 0$ to $t-1$, we obtain:

$$\sum_{\tau=0}^{t-1} (b_n^\tau - B_{max}) \leq Q_n^t - Q_n^0 = Q_n^t. \quad (20)$$

From equation (20), it follows that:

$$\frac{1}{t} \sum_{\tau=0}^{t-1} (b_n^\tau - B_{max}) \leq \frac{Q_n^t}{t}, \quad (21)$$

$$\limsup_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[b_n^\tau - B_{max}] \leq \lim_{t \rightarrow +\infty} \frac{\mathbb{E}[Q_n^t]}{t} = 0. \quad (22)$$

Therefore, it is concluded:

$$\limsup_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[b_n^\tau] \leq B_{max}. \quad (23)$$

This confirms that mean rate stability of Q_n is sufficient for satisfying the long-term budget constraint C_3 . $\square$

Following this, the framework employing Lyapunov optimization transforms the long-term budget constraint into a problem of virtual queue stability, capable of adaptively adjusting inference routing strategies and resource allocation in a dynamic environment.

4.2 Trade-off Between Computational Cost and Expenditure

This subsection explores the balance between minimizing inference latency and energy while controlling finan-

cial expenditures. Initially, a quadratic Lyapunov function is defined to assess the system state, represented by the collection of expenditure deficit queues for all agents, $\Theta(t) = (Q_1^t, Q_2^t, \dots, Q_N^t)$, $n \in \mathcal{N}$, as shown below:

$$L(\Theta(t)) = \frac{1}{2} \sum_{n=1}^N (Q_n^t)^2, \quad (24)$$

where $L(\Theta(t))$ quantifies the congestion level of the expenditure deficit queue system; a higher value indicates greater system instability. For clarity, $L(\Theta(t))$ is abbreviated as $L^t \triangleq L(\Theta(t))$ in this section. Notably, a large L^t suggests that even minor variations in the queue can drastically affect the overall system state due to the squaring effect.

To ensure that the time-averaged expenditure does not exceed the time-averaged budget while aiming for a low congestion state of L^t , the *single-time-slot Lyapunov drift* is:

$$\Delta(L^t) = \mathbb{E}[L^{t+1} - L^t | \Theta(t)], \quad (25)$$

where $\Delta(L^t)$ indicates the expected change in L^t at the next time step, given the current system state $\Theta(t)$. A smaller $\Delta(L^t)$ suggests enhanced system stability.

Furthermore, to minimize the overall inference costs while ensuring that expenditures adhere to long-term budget constraints, the *Lyapunov drift plus penalty* function is:

$$\Delta(L^t) + V \mathbb{E}[P^t | \Theta(t)], \quad (26)$$

where $P^t = \sum_{n \in \mathcal{N}} C_n^t(x_n^t)$ represents the inference cost penalty term, and V is a control parameter used to balance the trade-off between minimizing inference penalties (latency and energy) and maintaining system stability (budget adherence). A higher V value prioritizes LLM performance, pushing inference costs toward the theoretical minimum at the potential risk of temporary budget deficits. By adjusting V , agents can adaptively balance their preference for high-quality LLM responses against their available financial tokens. Instead of directly minimizing the drift plus penalty term, the approach seeks to minimize its upper bound as detailed in Theorem 2.

Theorem 2 (Upper Bound of Lyapunov Drift Plus Penalty). *For any offloading queue Q_n , the following proposition holds:*

$$\begin{aligned} & \Delta(L^t) + V \mathbb{E}[P^t | \Theta(t)] \\ & \leq B + \sum_{n=1}^N Q_n^t \mathbb{E}[(b_n^t - B_{max}) | \Theta(t)] \\ & \quad + V \mathbb{E}[P^t | \Theta(t)], \end{aligned} \quad (27)$$

where $B = \frac{1}{2} \sum_{n=1}^N \max\{B_{max}^2, (b_{n,max} - B_{max})^2\}$ is a constant satisfying $\frac{1}{2} \sum_n (y_n^t)^2 \leq B$, and let $b_{n,max} = \max_{t \in \mathcal{T}} \{b_n^t\}$ be the maximum expenditure of ED_n across all time slots.

Proof. Denote $y_n^t = b_n^t - B_{max}$. Since $([x]^+)^2 = (\max\{x, 0\})^2 \leq x^2$, we have

$$(Q_n^{t+1})^2 \leq (Q_n^t + y_n^t)^2 = (Q_n^t)^2 + (y_n^t)^2 + 2Q_n^t y_n^t, \quad (28)$$

thus

$$\frac{1}{2} \sum_{n=1}^N (Q_n^{t+1})^2 \leq \frac{1}{2} \sum_{n=1}^N (Q_n^t)^2 + \frac{1}{2} \sum_{n=1}^N (y_n^t)^2 + \sum_{n=1}^N Q_n^t y_n^t. \quad (29)$$

Therefore, for a single time slot, the Lyapunov drift is

$$\begin{aligned} L^{t+1} - L^t &= \frac{1}{2} \sum_{n=1}^N (Q_n^{t+1})^2 - \frac{1}{2} \sum_{n=1}^N (Q_n^t)^2 \\ &\leq \frac{1}{2} \sum_{n=1}^N (y_n^t)^2 + \sum_{n=1}^N Q_n^t y_n^t. \end{aligned} \quad (30)$$

Further derivation yields

$$\begin{aligned} \Delta(L^t) + V\mathbb{E}[P^t | \Theta(t)] &\leq B + \sum_{n=1}^N Q_n^t \mathbb{E}[y_n^t | \Theta(t)] \\ &\quad + V\mathbb{E}[P^t | \Theta(t)]. \end{aligned} \quad (31)$$

This concludes the derivation of the upper bound for Lyapunov drift plus penalty. $\square$

4.3 Online Optimization Framework

Building on the insights from previous analysis, this subsection details BOOF. The primary objective of BOOF is to minimize inference costs by reducing the upper bound of the drift plus penalty term, while concurrently ensuring the stability of the deficit queue system. Finally, the online optimization problem is reformulated as:

$$\mathcal{P}_2 : \min_{\{x\}} \sum_{n \in \mathcal{N}} Q_n^t b_n^t + V \sum_{n \in \mathcal{N}} C_n^t(x_n^t) \quad (32)$$

$$s.t. \quad C_1, C_2, \quad (33)$$

where $t \in \mathcal{T}$, and the long-term budget constraint C_3 is handled by the queue-weighted expenditure term. Continuous updates to Q_n across time slots transform the long-term coupled API and leasing expenditure expectations into a real-time balance between inference latency and deficit control. The comprehensive process of the proposed BOOF framework addressing $\mathcal{P}_1$ is detailed in Algorithm 1.

Although $\mathcal{P}_2$ remains a nonlinear and non-convex optimization problem, BOOF has a certain robustness to the errors in the solution to $\mathcal{P}_2$. Let F represent the per-slot approximation gap between the selected solution and the optimal solution for $\mathcal{P}_2$, and Theorem 3 proves the resulting long-term cost and queue bounds.

Theorem 3 (Asymptotic Performance under Approximate Per-slot Solvers). *Suppose the arrival, workload, cost, and*

Algorithm 1 Online Offloading and Resource Allocation BOOF

Input: $\{Q_n^0\}_{n \in \mathcal{N}} \leftarrow \{0\}_{n \in \mathcal{N}}$, other constant parameters in $\mathcal{P}_2$.

Output: $\mathcal{S} \triangleq \{\{x_n^{t,*}\}_{n \in \mathcal{N}}\}_{t \in \mathcal{T}}$

for $t = 0 \rightarrow T$ **do**

Obtain an approximate optimal set of routing decisions $\{x_n^*\}_{n \in \mathcal{N}}$, and the corresponding resource allocation plan by calling Algorithm 2 or Algorithm 3 depending on the degree of privacy constraints;

Update the virtual queues:

$$Q_n^{t+1} = [Q_n^t + b_n^t - B_{max}]^+, \quad \forall n \in \mathcal{N};$$

end for

Return $\mathcal{S} \triangleq \{\{x_n^{t,*}\}_{n \in \mathcal{N}}\}_{t \in \mathcal{T}}$.

expenditure processes are bounded, the feasible set satisfying C_1 and C_2 is nonempty in every time slot, Slater's condition holds for C_3 , and the chosen solver returns an F -approximate solution to $\mathcal{P}_2$ in each time slot. BOOF achieves the following performance:

$$\lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[p^t] \leq p^{opt} + \frac{B+F}{V}, \quad (34)$$

$$\lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=1}^{T-1} \sum_{n=1}^N Q_n^t \leq \frac{B+F+V(p^{\max} - p^{\min})}{\epsilon}, \quad (35)$$

where $p^t = \sum_{n \in \mathcal{N}} C_n^t(x_n^t)$ represents the penalty cost for the t^{th} time slot, p^{opt} is the optimal time-average objective value, $p^{\min} = p^{opt}$, $p^{\max}$ is a finite upper bound on feasible per-slot penalty, and ϵ is the Slater slack.

Proof. Assume that the original optimization problem $\mathcal{P}_2$ has at least one non-trivial solution, and for $\forall n \in \mathcal{N}$, the expected values of expenditures b_n^t and penalty terms p^t are bounded, i.e., $p^{\min} < p^t < p^{\max}$. Let $w(t)$ be a stationary process with some distribution that satisfies all the aforementioned assumptions, then for any $\delta > 0$, there exists a strategy ξ such that

$$\mathbb{E}[(b_n^t)^\xi - B_{max}] \leq \delta, \quad \forall n \in \mathcal{N}. \quad (36)$$

Since the error introduced by BOOF is at most $F \geq 0$, for all feasible strategies, we have

$$\begin{aligned} \Delta(L^t) + V\mathbb{E}[p^t | \Theta(t)] &\leq B + F + \\ &\sum_{n=1}^N \left(Q_n^t \mathbb{E}[(b_n^t)^\xi - B_{max} | \Theta(t)] \right) + V\mathbb{E}[(p^t)^\xi | \Theta(t)]. \end{aligned} \quad (37)$$

When $\delta \rightarrow 0$, we can derive

$$\Delta(L^t) + V\mathbb{E}[p^t | \Theta(t)] \leq B + F + Vp^{opt}. \quad (38)$$

Transforming equation (38) and taking the expectation, then summing the inequalities over multiple time slots $\tau \in \{0, 1, \dots, t-1\}$, we obtain:

$$\mathbb{E}[L^t] - \mathbb{E}[L^0] + V \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau] \leq (B + F + Vp^{opt})t. \quad (39)$$

Rearranging (39), we derive:

$$\begin{aligned} V \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau] &\leq (B + F + Vp^{opt})t + \mathbb{E}[L^0] - \mathbb{E}[L^t] \\ &\leq (B + F + Vp^{opt})t + \mathbb{E}[L^0]. \end{aligned} \quad (40)$$

Dividing both sides of (40) by Vt , we find:

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau] \leq p^{opt} + \frac{B + F}{V} + \frac{\mathbb{E}[L^0]}{Vt}. \quad (41)$$

As $t \rightarrow +\infty$, this results in:

$$\lim_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau] \leq p^{opt} + \frac{B + F}{V}, \quad (42)$$

thus proving (34) is valid.

Next, we demonstrate the upper bound of the expenditure deficit queue's length as shown in equation (35). Assuming the original optimization problem satisfies Slater's condition [20] to ascertain the backlog item's upper bound, specifically, for $\epsilon > 0$ and function $\psi(\epsilon)$, there exists $p^{min} \leq \psi(\epsilon) \leq p^{max}$ and a strategy ξ satisfying the following conditions:

$$\begin{aligned} \mathbb{E}[(p^t)^\xi] &= \psi(\epsilon), \\ \mathbb{E}[(b_n^t)^\xi - B_{max}] &\leq -\epsilon, \quad \forall n \in \mathcal{N}. \end{aligned} \quad (43)$$

Combining formula (37) with (43), we conclude:

$$\Delta(L^t) + V\mathbb{E}[p^t | \Theta(t)] \leq B + F + \Delta_v, \quad (44)$$

where $\Delta_v = V\psi(\epsilon) - \epsilon \sum_{n=1}^N Q_n^t$. Thus, we have:

$$\begin{aligned} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{n=1}^N Q_n^\tau &\leq \frac{V(\psi(\epsilon) - \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau])}{\epsilon} \\ &\quad + \frac{B + F}{\epsilon} + \frac{\mathbb{E}[L^0]}{t\epsilon}. \end{aligned} \quad (45)$$

Letting $t \rightarrow +\infty$, equation (45) can be rewritten as:

$$\begin{aligned} \lim_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{n=1}^N Q_n^\tau &\leq \\ &\left[\frac{B + F}{\epsilon} + \frac{V\left(\psi(\epsilon) - \lim_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau]\right)}{\epsilon} \right]. \end{aligned} \quad (46)$$

Considering that $\lim_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[p^\tau] \geq p^{opt}$, it follows:

$$\lim_{t \rightarrow +\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{n=1}^N Q_n^\tau \leq \frac{B + F + V(p^{max} - p^{min})}{\epsilon}. \quad (47)$$

Thus, proving that equation (35) holds true as well. $\square$

5 Algorithm Design

To solve the decoupled sub-problem $\mathcal{P}_2$ under long-term constraints, this section designs algorithms suitable for different levels of information sharing and data-placement restrictions, including BRAO and MADO.

5.1 BRAO Algorithm

In this section, we introduce an adaptive routing algorithm based on the best response in a weighted potential game, specifically designed to address the optimization problem $\mathcal{P}_2$ within a single time slot. The BRAO algorithm is particularly suited for cooperative multi-agent scenarios where routing intent can be shared among agents to mitigate resource contention at the edge GPUs. Upon determining the optimal inference routing strategies, the corresponding resource allocations are executed as defined in Section 3.2.

To simplify the notation, we omit the superscript t and reformulate the original problem $\mathcal{P}_2$ as:

$$\mathcal{P}_2' : \min_{\{\mathbf{x}\}} \sum_{n \in \mathcal{N}} \Omega_n(x_n, \mathbf{x}_{-n}), \text{ s.t. } C_1, C_2, \quad (48)$$

where the objective function $\Omega_n(x_n) = \Omega_n$ expands to:

$$\Omega_n = Q_n b_n + VC_n(x_n). \quad (49)$$

Expanding (49), we have:

$$\begin{aligned} \Omega_n = Q_n \cdot &\left(\sum_{m \in \mathcal{M}} \left[\vartheta_{ES} \left(1 - \alpha \sum_{j \in \mathcal{N}} \gamma_j x_{j,m} \right) C_n S_n x_{n,m} \right] \right. \\ &+ \vartheta_{cloud} C_n S_n x_{n,c} \Big) + V \cdot (\lambda_n^L \ell_{n,l} + \lambda_n^E e_{n,l}) x_{n,l} \\ &+ V \cdot \sum_{m \in \mathcal{M}} \left(\lambda_n^L (\tilde{\ell}_{n,m}^{com} + \tilde{\ell}_{n,m}^{del}) + \lambda_n^E \tilde{e}_{n,m} \right) x_{n,m} \\ &+ V \cdot \left(\lambda_n^L (\hat{\ell}_{n,c}^{com} + \hat{\ell}_{n,c}^{del}) + \lambda_n^E \hat{e}_{n,c} \right) x_{n,c}. \end{aligned} \quad (50)$$

It is evident that problem $\mathcal{P}_2'$ presents a non-convex and non-linear challenge due to high coupling of routing variables $\{x_n\}_{n \in \mathcal{N}}$, making a direct solution difficult. However, game theory provides a robust method to address such problems with coupled variables, as it effectively analyzes competitive interactions among agents for finite edge GPU resources. Consequently, we transform $\mathcal{P}_2'$ into game model

$\mathcal{G} = \{\mathcal{N}, \{x_n\}_{n \in \mathcal{N}}, \Omega_n(x_n, \mathbf{x}_{-n})\}$. We then proceed to analyze the Nash equilibrium, explore approximate NE routing strategies, and evaluate the worst-case performance for game $\mathcal{G}$.

Definition 1 (Nash equilibrium). Let $\Omega_n(x_n) \triangleq \Omega_n(x_n, \mathbf{x}_{-n})$. If a strategy configuration $\mathbf{x}^* = \{x_n\}_{n \in \mathcal{N}}$ for game $\mathcal{G}$ satisfies the condition that no agent can reduce their own $\Omega_n(x_n, \mathbf{x}_{-n})$ by unilaterally changing their strategy, then this configuration is termed the NE, defined formally as:

$$\Omega_n(x_n^*, \mathbf{x}_{-n}^*) \leq \Omega_n(x_n, \mathbf{x}_{-n}^*), \quad \forall n \in \mathcal{N}. \quad (51)$$

Definition 2 (Best response). Under the given set of other agents' strategies $\mathbf{x}_{-n}$, the best response for ED_n refers to the strategy that should be adopted by agent n to minimize their own $\Omega_n(x_n, \mathbf{x}_{-n})$.

$$x'_n = \arg \min_{x_n \in \psi_n} \Omega_n(x_n, \mathbf{x}_{-n}), \quad (52)$$

where ψ_n is the feasible action set induced by C_1 and C_2 for agent n in the current time slot.

In an end-edge-cloud collaborative LLM inference scenario, achieving the NE implies that all agents have identified optimal strategies for model routing under the current system state, balancing edge GPU contention and API costs.

Definition 3 (Generalized ordinal potential game). Based on the research findings in [21], a GOPG is a type of non-cooperative game in which each player's strategy choice not only affects their own payoff but also the performance of the entire system. In a GOPG, there exists a global potential function that captures the impact of any single player's strategy change on the system's overall performance.

In the context of this paper, the potential function $\Phi(\mathbf{x})$ quantifies the impact of all participants' strategy changes on the total cost function $\sum_{n \in \mathcal{N}} \Omega_n(x_n, \mathbf{x}_{-n})$. If for any $x'_n, x_n \in \psi$ the following condition is satisfied:

$$\Omega_n(x'_n, \mathbf{x}_{-n}) < \Omega_n(x_n, \mathbf{x}_{-n}) \quad (53)$$

$$\Rightarrow \Phi(x'_n, \mathbf{x}_{-n}) < \Phi(x_n, \mathbf{x}_{-n}) \quad (54)$$

then a reduction in the potential function due to strategy change by an agent directly decreases the value of (48). Consequently, any routing strategy set $\mathbf{x}$ that locally minimizes the potential function is capable of achieving the NE.

Theorem 4 ($\mathcal{G}$ is a Generalized Ordinal Potential Game). $\mathcal{G}$ is a GOPG, and there exists a pure strategy NE $\{x_n^{NE}\}_{n \in \mathcal{N}}$.

Proof. According to (50), we have

$$\Omega_n(x_n, \mathbf{x}_{-n}) = \lambda_n^{L'} \frac{C_n''}{f_n} + \lambda_n^{E'} \tilde{\kappa}_{n,l} C_n'' f_{n,l}^2, \quad (55)$$

$$\begin{aligned} \Omega_n(x_{n,c} = 1, \mathbf{x}_{-n}) &= \lambda_n^{L'} \left(\frac{C_n''}{\hat{f}_c} + a_{n,c} + \frac{S_n}{\hat{R}_{n,c}} + \frac{S_n}{\hat{R}'_{n,c}} \right) \\ &\quad + \lambda_n^{E'} \hat{p}_n^{\text{cloud}} \left(\frac{S_n}{\hat{R}_{n,c}} + \frac{S_n}{\hat{R}'_{n,c}} \right) \\ &\quad + \vartheta_{\text{cloud}} C_n', \end{aligned} \quad (56)$$

where $C_n' = Q_n S_n C_n$, $C_n'' = S_n C_n$, $\lambda_n^{L'} = V \lambda_n^L$, $\lambda_n^{E'} = V \lambda_n^E$, and $a_{n,m} = a_{n,ES}$ for $m \in \mathcal{M}$ are constants unrelated to optimization.

Next, we specialize the potential construction to the equal-weight proportional-sharing case used by the main experiments, namely $\gamma_n = 1$ for all n . Define

$$N_m(\mathbf{x}) \triangleq \sum_{n \in \mathcal{N}} x_{n,m}, \quad (57)$$

and write the edge and residual local and cloud costs as

$$\Omega_{n,m}(\mathbf{x}) = D_{n,m} N_m(\mathbf{x}) + G_n, \quad (58)$$

$$\bar{\Omega}_{n,l} = \Omega_n(x_{n,l} = 1, \mathbf{x}_{-n}) - G_n, \quad (59)$$

$$\bar{\Omega}_{n,c} = \Omega_n(x_{n,c} = 1, \mathbf{x}_{-n}) - G_n. \quad (60)$$

Here

$$D_{n,m} = \lambda_n^{L'} \left(\frac{C_n''}{\hat{f}_m} + \frac{S_n}{R_{n,m}} \right) + \lambda_n^{E'} E_{n,m} - \alpha H_n, \quad (61)$$

$$E_{n,m} = \frac{\tilde{p}_n^{ES} S_n}{R_{n,m}}, \quad (62)$$

$$G_n = \lambda_n^{L'} a_{n,m} + H_n, \quad (63)$$

$$H_n = \vartheta_{ES} C_n', \quad (64)$$

and the edge discount coefficient is restricted so that $D_{n,m} > 0$. Under the positivity conditions $\bar{\Omega}_{n,l} > 0$, $\bar{\Omega}_{n,c} > 0$, and $D_{n,m} > 0$, a potential function is

$$\begin{aligned} \Phi(\mathbf{x}) &= \sum_{m \in \mathcal{M}} \sum_{k=1}^{N_m(\mathbf{x})} \log k \\ &\quad + \sum_{n \in \mathcal{N}} I_{\{x_n=l\}} \log \bar{\Omega}_{n,l} \\ &\quad + \sum_{n \in \mathcal{N}} \sum_{m \in \mathcal{M}} I_{\{x_n=m\}} \log D_{n,m} \\ &\quad + \sum_{n \in \mathcal{N}} I_{\{x_n=c\}} \log \bar{\Omega}_{n,c}. \end{aligned} \quad (65)$$

We verify the four unilateral moves.

Case 1: agent n switches from local to edge server m . Then

$$\begin{aligned} &\Phi(x_{n,m} = 1, \mathbf{x}_{-n}) \\ &\quad - \Phi(x_{n,l} = 1, \mathbf{x}_{-n}) \\ &= \log(D_{n,m} N_m(x_{n,m} = 1, \mathbf{x}_{-n})) \\ &\quad - \log \bar{\Omega}_{n,l}, \end{aligned} \quad (66)$$

which has the same sign as

$$\begin{aligned} & \Omega_{n,m}(x_{n,m} = 1, \mathbf{x}_{-n}) - \Omega_{n,l} \\ &= D_{n,m} N_m(x_{n,m} = 1, \mathbf{x}_{-n}) \\ & \quad - \bar{\Omega}_{n,l}. \end{aligned} \quad (67)$$

Case 2: agent n switches from local to cloud. Then

$$\begin{aligned} & \Phi(x_{n,c} = 1, \mathbf{x}_{-n}) - \Phi(x_{n,l} = 1, \mathbf{x}_{-n}) \\ &= \log \bar{\Omega}_{n,c} - \log \bar{\Omega}_{n,l}, \end{aligned} \quad (68)$$

which has the same sign as $\Omega_{n,c} - \Omega_{n,l} = \bar{\Omega}_{n,c} - \bar{\Omega}_{n,l}$.

Case 3: agent n switches from edge server m to edge server m' . Then

$$\begin{aligned} & \Phi(x_{n,m'} = 1, \mathbf{x}_{-n}) \\ & \quad - \Phi(x_{n,m} = 1, \mathbf{x}_{-n}) \\ &= \log(D_{n,m'} N_{m'}(x_{n,m'} = 1, \mathbf{x}_{-n})) \\ & \quad - \log(D_{n,m} N_m(x_{n,m} = 1, \mathbf{x}_{-n})), \end{aligned} \quad (69)$$

which has the same sign as the corresponding cost difference.

Case 4: agent n switches from edge server m to cloud. Then

$$\begin{aligned} & \Phi(x_{n,c} = 1, \mathbf{x}_{-n}) \\ & \quad - \Phi(x_{n,m} = 1, \mathbf{x}_{-n}) \\ &= \log \bar{\Omega}_{n,c} \\ & \quad - \log(D_{n,m} N_m(x_{n,m} = 1, \mathbf{x}_{-n})), \end{aligned} \quad (70)$$

which again has the same sign as the cost difference.

Therefore,

$$\Omega_n(x'_n, \mathbf{x}_{-n}) < \Omega_n(x_n, \mathbf{x}_{-n}) \implies \Phi(x'_n, \mathbf{x}_{-n}) < \Phi(x_n, \mathbf{x}_{-n}). \quad (71)$$

This indicates that the game $\mathcal{G}$ is a GOPG under equal-weight allocation. Heterogeneous γ_n values are retained in the numerical study but are not covered by the closed-form potential above. $\square$

Further, to achieve the NE set of routing strategies $\mathbf{x}^{NE}$ under this setting, we propose the BRAO algorithm suitable for scenarios where routing decision information is shared among agents. BRAO uses an asynchronous better-response update: in each inner iteration, all agents evaluate their best responses using the shared routing profile, but only one agent with the largest positive cost reduction updates its decision. This single-agent update rule is important because synchronous updates may move several agents to the same edge server simultaneously and can create oscillations under resource contention. According to [21], finite potential games converge under sequential better-response dynamics. Consequently, after a finite number of accepted updates, BRAO reaches an NE or stops at the iteration cap H . The overall process of the BRAO algorithm is shown in Algorithm 2.

Algorithm 2 BRAO Algorithm to obtain the NE of the Decoupled Sub-problem $\mathcal{P}_2$

Input: Initialize all agents' strategies

$\{x_n^{(0)}\}_{n \in \mathcal{N}} \leftarrow \{x_{n,l} = 1\}_{n \in \mathcal{N}}$, iteration index $h \leftarrow 0$, tolerance $\eta \geq 0$.

Output: NE strategy set $\mathbf{x}^{NE} \triangleq \{x_n^{NE}\}_{n \in \mathcal{N}}$.

while $h < H$ **do**

$h \leftarrow h + 1$

Compute resource allocations of the three computational modes described in Section 3.2;

for $n \in \mathcal{N}$ **do**

Compute x'_n according to (52) and $\Delta_n = \Omega_n(x_n^{(h-1)}, \mathbf{x}_{-n}^{(h-1)}) - \Omega_n(x'_n, \mathbf{x}_{-n}^{(h-1)})$;

end for

Choose $n^* = \arg \max_{n \in \mathcal{N}} \Delta_n$ with deterministic tie-breaking;

if $\Delta_{n^*} > \eta$ **then**

Update $x_{n^*}^{(h)} \leftarrow x'_{n^*}$ and keep $x_j^{(h)} \leftarrow x_j^{(h-1)}$ for all $j \neq n^*$;

end if

else

Break;

end if

end while

Return the final strategy set $\{x_n^{NE}\}_{n \in \mathcal{N}}$.

The BRAO algorithm is suitable for addressing single time-slot inference routing problems under long-term budget constraints, allowing cooperative agents to exploit shared decision information while preserving a monotone potential descent.

5.1.1 Convergence Analysis

In dynamic multi-agent environments, algorithms must adapt quickly to changes in prompt lengths and API network conditions. We analyze BRAO under the asynchronous update rule in Algorithm 2.

Theorem 5 (Finite Convergence of BRAO). *For a fixed time slot with finite feasible action sets $\{\psi_n\}_{n \in \mathcal{N}}$, BRAO terminates after finitely many accepted updates. When $\eta = 0$ and exact costs are used, the termination point is a pure-strategy NE. When $\eta > 0$, the termination point is an additive η -approximate NE: no agent can reduce its own cost by more than η through a unilateral feasible deviation. In both cases, the number of accepted updates is at most*

$$\prod_{n \in \mathcal{N}} |\psi_n| - 1. \quad (72)$$

Proof. By Theorem 4, every accepted unilateral better response strictly decreases the potential under equal-weight

allocation, and Algorithm 2 updates at most one agent per iteration. The joint action set is finite, so a strictly decreasing potential path cannot revisit a previous routing profile. Hence BRAO terminates after finitely many accepted updates, and the number of accepted updates is at most the number of distinct joint profiles minus one. When $\eta = 0$, termination implies that no agent has a cost-reducing unilateral move, which is exactly the NE condition. When $\eta > 0$, termination implies that every unilateral improvement is at most η , which is the stated approximate-NE condition. The ordinal potential guarantees only that cost decreases and potential decreases share the same sign; it does not identify the magnitude of the potential drop with η . $\square$

5.1.2 Price of Anarchy Analysis

We analyze the Price of Anarchy (PoA) to evaluate the extent to which individual agent behaviors diverge from centralized optimal scheduling. Let $\bar{x}$ denote any NE solution, and x^* denote the optimal routing strategy.

$$PoA = \frac{\max_{\bar{x}} \sum_{n \in \mathcal{N}} \Omega_n(\bar{x})}{\sum_{n \in \mathcal{N}} \Omega_n(x^*)}. \quad (73)$$

Theorem 6 (Price of Anarchy Analysis). *For the generalized ordinal potential game $\mathcal{G}$, the PoA calculated using (73) satisfies:*

$$1 \leq PoA \leq \frac{\sum_{n=1}^N \max \{\Omega_n^{\max}, \Omega_{n,l}, \Omega_{n,c}\}}{\sum_{n=1}^N \min \{\Omega_n^{\min}, \Omega_{n,l}, \Omega_{n,c}\}}, \quad (74)$$

where $\Omega_n^{\max} \triangleq \max_{m \in \mathcal{M}} \{D_{n,m} \sum_{j \in \mathcal{N}} \gamma_j + G_n\}$, $\Omega_n^{\min} \triangleq \min_{m \in \mathcal{M}} \{D_{n,m} \gamma_n + G_n\}$, $\Omega_{n,l} \triangleq \Omega_n|_{x_{n,l}=1}$, and $\Omega_{n,c} \triangleq \Omega_n|_{x_{n,c}=1}$.

Proof. For $\bar{x} = \{\bar{x}_n, \bar{x}_{-n}\}$, where the agent's strategy $\bar{x}_n$ is one of three modes: local mode $\Omega_n|_{x_{n,l}=1}$, cloud mode $\Omega_{n,c} \triangleq \Omega_n|_{x_{n,c}=1}$, and edge mode. When the agent routes to ES_m , the worst-case scenario is all other agents also choose ES_m . Thus, in the worst-case scenario, we have $\Omega_n^{\max} \triangleq \max_{m \in \mathcal{M}} \{D_{n,m} \sum_{j \in \mathcal{N}} \gamma_j + G_n\}$. Hence,

$$\Omega_n(\bar{x}_n, \bar{x}_{-n}) \leq \max \{\Omega_n^{\max}, \Omega_{n,l}, \Omega_{n,c}\}. \quad (75)$$

Conversely, when the agent routes to ES_m and only n chooses ES_m , the best-case scenario results in $\Omega_n^{\min} \triangleq \min_{m \in \mathcal{M}} \{D_{n,m} \gamma_n + G_n\}$. Thus,

$$\Omega_n(x^*) \geq \min \{\Omega_n^{\min}, \Omega_{n,l}, \Omega_{n,c}\}. \quad (76)$$

Combining the above inequalities (75) and (76), we obtain:

$$1 \leq PoA \leq \frac{\sum_{n=1}^N \max \{\Omega_n^{\max}, \Omega_{n,l}, \Omega_{n,c}\}}{\sum_{n=1}^N \min \{\Omega_n^{\min}, \Omega_{n,l}, \Omega_{n,c}\}}. \quad (77)$$

$\square$

Algorithm 3 Distributed MADO Algorithm

Input: Constants for edge and cloud servers in computational cost (12), learning-rate sequence $\{\alpha_t\}$, and exploration sequence $\{\varepsilon_t\}$.
Output: Routing decision vector $\mathcal{X}^* = \{x_n^*\}_{n \in \mathcal{N}}$.
Initialize each agent's policy vector p_n^0 uniformly over its feasible action set ψ_n .
while $t < T$ and the stopping criterion is not met **do**
 for $n \in \mathcal{N}$ **do**
 Form $\tilde{p}_n^t = (1 - \varepsilon_t)p_n^t + \varepsilon_t u_n$, where u_n is uniform on ψ_n ;
 Sample x_n^t according to $\tilde{p}_n^t$;
 Receive resource allocation and reward φ_n^t under current decisions;
 Set $r_n^t = \text{clip}_{[0,1]}(1 - \beta(\Omega_n^t + \varsigma^t H_n^t))$;
 Update $p_n^{t+1} = \Pi_{\Delta(\psi_n)}[p_n^t + \alpha_t r_n^t (e_{x_n^t} - p_n^t)]$;
 end for
end while
Return $x_n^* = \arg \max_{a \in \psi_n} p_{n,a}^T$ for each agent.

5.2 MADO Algorithm

The BRAO algorithm requires the sharing of routing decision information, which may be restricted in privacy-sensitive or limited-information deployments. To address this, we introduce the MADO algorithm, designed to achieve efficient inference routing without requiring agents to disclose their intent or decision profiles to other agents. The central coordinator only returns aggregate resource feedback and the realized cost needed for policy updates; we do not claim cryptographic privacy against an adversarial coordinator.

5.2.1 Algorithm Process

As illustrated in Algorithm 3, each agent maintains a probability vector $p_n = (p_{n,0}, p_{n,1}, \dots, p_{n,M}, p_{n,M+1})$ associated with its feasible routing decisions. Here, $p_{n,0}$ represents the probability of computing locally via SLMs; $p_{n,m}$, $m \in \mathcal{M}$, denotes the probability of routing to ES_m ; and $p_{n,M+1}$ indicates the probability of calling a remote cloud API. To prevent premature exclusion of feasible actions, agent n samples from the exploration distribution

$$\tilde{p}_n^t = (1 - \varepsilon_t)p_n^t + \varepsilon_t u_n, \quad (78)$$

where u_n is uniform over ψ_n and $\varepsilon_t \in [0, 1]$. A lower computational cost yields a higher reward. We use the clipped and normalized reward

$$r_n^t = \text{clip}_{[0,1]}(1 - \beta(\Omega_n^t + \varsigma^t H_n^t)), \quad (79)$$



and update the policy by

$$\mathbf{p}_n^{t+1} = \Pi_{\Delta(\psi_n)}[\mathbf{p}_n^t + \alpha_t r_n^t (\mathbf{e}_{x_n^t} - \mathbf{p}_n^t)], \quad \forall n \in \mathcal{N}, \quad (80)$$

where $\alpha_t \in (0, 1]$ is the learning rate, $\mathbf{e}_{x_n^t}$ is the unit vector of the sampled action, $\Pi_{\Delta(\psi_n)}$ denotes Euclidean projection onto the probability simplex over ψ_n , and $\text{clip}_{[0,1]}(z) = \min(1, \max(0, z))$. Before projection, the update is the convex combination $(1 - \alpha_t r_n^t) \mathbf{p}_n^t + \alpha_t r_n^t \mathbf{e}_{x_n^t}$, so projection is inactive in exact arithmetic and protects probability validity. Agents update their probability vectors based on realized cost feedback observed from the dynamic environment, making MADDO suitable for deployments where routing intent is not shared among agents.

5.2.2 Convergence Analysis

We first state a conditional result for a stationary learning phase and define every auxiliary symbol before it is used. For each agent $n \in \mathcal{N}$, let ψ_n be its finite feasible action set and let $\mathbf{p}_n^t = (p_{n,a}^t)_{a \in \psi_n}$ be the routing probability vector at iteration t . Write $\mathbf{W}^t = (\mathbf{p}_1^t, \dots, \mathbf{p}_N^t)$ for the joint policy profile, and let $\Delta(\psi_n)$ denote the probability simplex over ψ_n . The exploration mixture, clipped reward, learning rate, and projection operator are those defined in the update rule above. Assume finite fixed feasible action sets, bounded rewards $r_n^t \in [0, 1]$, Lipschitz expected reward maps

$$\bar{r}_n(\mathbf{W}) := \mathbb{E}[r_n^t \mid \mathbf{W}^t = \mathbf{W}],$$

and a strict Lyapunov function U for the projected mean-field ordinary differential equation (ODE) whose largest invariant set is the stationary Nash-equilibrium set $\mathcal{E}$.

Theorem 7 (Conditional convergence of MADDO). *Suppose $\alpha_t = (t+1)^{-\kappa}$ and $\varepsilon_t = (t+1)^{-\rho}$, where*

$$\frac{1}{2} < \kappa \leq 1, \quad 0 < \rho \leq 1, \quad \kappa + \rho > 1.$$

Then every feasible action is sampled infinitely often almost surely, $\mathbf{W}^t$ remains in $\prod_{n \in \mathcal{N}} \Delta(\psi_n)$, and

$$\text{dist}(\mathbf{W}^t, \mathcal{E}) \rightarrow 0 \quad \text{almost surely,}$$

where $\text{dist}(\mathbf{W}, \mathcal{E}) = \inf_{\mathbf{W}' \in \mathcal{E}} \|\mathbf{W} - \mathbf{W}'\|_2$. If the stationary game has a unique strict Nash equilibrium $\mathbf{W}^{NE}$, then $\mathbf{W}^t \rightarrow \mathbf{W}^{NE}$ almost surely.

Proof. We prove the claim in five steps.

Step 1 (simplex invariance). Because $0 < \alpha_t \leq 1$ and $0 \leq r_n^t \leq 1$, the unprojected recursion is the convex combination

$$(1 - \alpha_t r_n^t) \mathbf{p}_n^t + \alpha_t r_n^t \mathbf{e}_{x_n^t} \in \Delta(\psi_n),$$

where $\mathbf{e}_{x_n^t}$ is the unit vector of the sampled action x_n^t . The Euclidean projection $\Pi_{\Delta(\psi_n)}$ therefore keeps $\mathbf{p}_n^t \in \Delta(\psi_n)$ for every $t \geq 0$.

Step 2 (persistent exploration). The exploration mixture satisfies

$$\tilde{p}_{n,a}^t = (1 - \varepsilon_t) p_{n,a}^t + \varepsilon_t \frac{1}{|\psi_n|} \geq \frac{\varepsilon_t}{|\psi_n|}$$

for every feasible action $a \in \psi_n$. Since $\sum_t \varepsilon_t = \infty$, every feasible action is sampled infinitely often almost surely.

Step 3 (mean-field drift and noise decomposition). Define the zero-exploration mean drift $h = (h_n)_{n \in \mathcal{N}}$ by

$$h_n(\mathbf{W}) = \sum_{a \in \psi_n} p_{n,a} \bar{r}_n(\mathbf{W}; a) (\mathbf{e}_a - \mathbf{p}_n),$$

where $\bar{r}_n(\mathbf{W}; a) = \mathbb{E}[r_n^t \mid \mathbf{W}^t = \mathbf{W}, x_n^t = a]$. The joint recursion then admits the projected stochastic-approximation form

$$\mathbf{W}^{t+1} = \mathbf{W}^t + \alpha_t [h(\mathbf{W}^t) + \mathbf{M}^{t+1} + \mathbf{b}^t],$$

where $\mathbf{M}^{t+1}$ is a bounded martingale-difference noise term induced by action sampling and reward randomness, and $\mathbf{b}^t \rightarrow 0$ is the exploration bias caused by ε_t . The step-size conditions imply

$$\sum_t \alpha_t = \infty, \quad \sum_t \alpha_t^2 < \infty, \quad \sum_t \alpha_t \|\mathbf{b}^t\| < \infty.$$

Hence the interpolated process tracks the projected ODE $\dot{\mathbf{W}} = h(\mathbf{W})$.

Step 4 (Lyapunov decrease along the ODE). By assumption, U is continuous on the compact joint simplex, nondecreasing along every ODE trajectory, and strictly increasing outside $\mathcal{E}$. Compactness and LaSalle's invariance principle therefore imply that every ω -limit set of the ODE is contained in $\mathcal{E}$.

Step 5 (transfer to the stochastic iterates). Projected stochastic-approximation arguments transfer the ODE limit-set characterization to the discrete process. Consequently,

$$\text{dist}(\mathbf{W}^t, \mathcal{E}) \rightarrow 0 \quad \text{almost surely.}$$

If the stationary game has a unique strict Nash equilibrium $\mathbf{W}^{NE}$, the same argument yields $\mathbf{W}^t \rightarrow \mathbf{W}^{NE}$ almost surely. $\square$

The theorem applies only to a stationary learning phase. In the nonstationary experiments, workloads, prices, and feasible sets change over time; therefore the fixed-ODE assumptions do not hold. We use a fixed practical learning rate in those experiments and report empirical stabilization under the stopping rule $\|\mathbf{W}^{t+1} - \mathbf{W}^t\|_1$ below a threshold for several consecutive iterations. No unconditional global-optimality claim is made for the dynamic traces.

6 Experiments and Analysis

In this section, we assess the effectiveness of the proposed scheme through comprehensive simulations. Initially, we

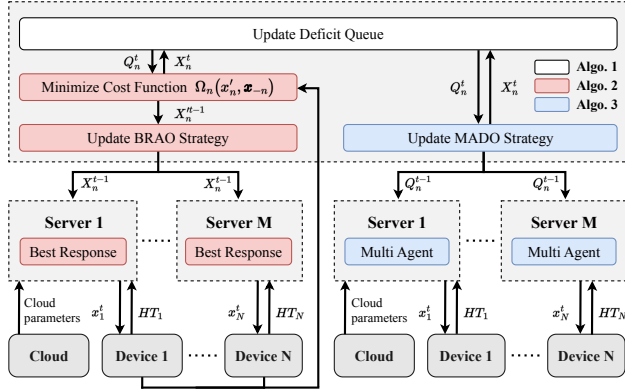


Fig. 3 Workflow of our proposed algorithms.

describe the system prototype's design and our testbed's implementation tailored for distributed LLM inference. This is followed by an explanation of the parameter settings, workloads, and baseline schemes utilized for the experimental assessment. Subsequently, we analyze the performance of the proposed scheme, focusing on aspects such as convergence, inference cost, optimality gap, and budget balance based on experimental results. We also examine the impact of critical parameters, including the MADO's hyperparameters, the tasks' heterogeneous performance weights, inference service unit prices for cloud and edge servers, and the time-averaged maximum budget on the computational cost.

6.1 Prototype Design and Testbed Implementation

System prototype: Under the end-edge-cloud architecture, end devices including smartphones and Jetson Orin edge boards serve as the execution environments for personalized AI agents. To overcome local RAM limitations, these devices deploy memory-augmented SLMs (e.g., MobileL-LaM) utilizing flash-based ROM expert knowledge injection architectures, enabling them to process lightweight agentic tasks efficiently. Our edge servers handle intermediate inference requests and resource allocation, comprising nodes equipped with entry-level GPUs (e.g., NVIDIA RTX 4090) hosting mid-tier open-source LLMs (e.g., LLaMA-3-8B). Tasks demanding highly complex reasoning or massive context windows are offloaded to the cloud, specifically AWS Bedrock and OpenAI API services hosting proprietary large models. The exchange of heterogeneous token workloads among the end, edge, and cloud servers is managed using the HTTP or gRPC protocols.

Testbed design: To enhance the prototype, we expanded the basic testbed using the Amazon EC2 service by adding multiple instances of end devices and edge servers. Furthermore, Apache Storm [22] functions as a distributed stream platform

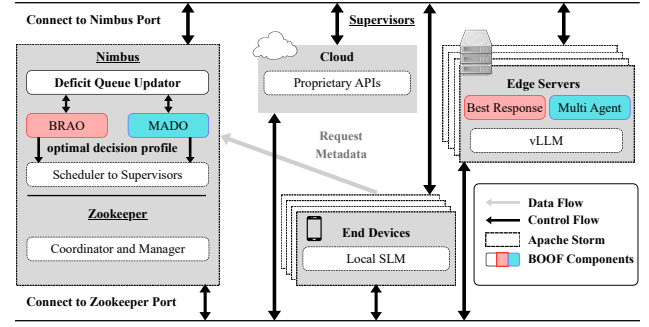


Fig. 4 Framework prototype implementation and testbed design.

to manage continuous LLM inference request streams. The spouts act as the prompt ingestion endpoints for continuous user queries, while the bolts act as the LLM execution wrappers (leveraging vLLM or llama.cpp) that process context decoding and auto-regressive token generation across the CEE hierarchy. The integration of Apache Storm into our architecture entails positioning its core components to schedule and manage inference tasks across various computational nodes. The role alignment between Apache Storm and our scheme, as shown in Figure 4, is outlined as follows: 1) Nimbus is initially deployed on the BOOF edge server, serving as the central coordinator for request distribution and system monitoring. 2) Zookeeper, responsible for configuration management, is also located on the BOOF edge server. 3) Supervisor nodes, operating on the end devices, additional edge servers, and the cloud instances, execute the specific model inference tasks assigned by Nimbus.

System workflow: In our testing platform in Figure 4, autonomous agents on end devices extract request metadata (estimated prompt tokens, max new tokens, and performance weights). The procedure initiates with agents transmitting the lightweight metadata tuple $(S_n, C_n, \lambda_n^E, \lambda_n^L)$ to Nimbus when the corresponding privacy rule permits it. Based on the degree of information sharing, Nimbus employs one of the two algorithms (BRAO and MADO) and derives a routing profile that governs inference execution. In MADO, each agent's policy probabilities remain local and only realized cost feedback is returned. Based on the decision: if local execution is chosen, the agent loads the SLM weights into RAM and processes locally; if edge or cloud execution is chosen, the context is serialized and offloaded. For edge execution, a gRPC stream is established, the edge GPU performs the pre-filling phase (incurring overhead $a_{n,m}$), and generated tokens are streamed back auto-regressively to the user. This configuration ensures a robust system where agentic tasks are efficiently distributed across the CEE structure.

Heterogeneous tasks processing: To simulate a heterogeneous multi-agent environment, we delineate three types of task processing that are effectively distributed across the hierarchy. Local processing is ideal for privacy-sensitive, context-light tasks (e.g., intent recognition and simple conversational turns). Leveraging ROM-augmented architectures, local devices efficiently perform these without significant RAM bottlenecks. Edge processing provides a balance of responsiveness and robust computational power, handling Retrieval-Augmented Generation (RAG) and moderate-length context summarization in near real-time. Cloud processing offers the capacity for intensive reasoning and massive context windows (e.g., complex mathematical reasoning, and large-scale code generation). Each task is assigned a minimum model-quality requirement, and routing options that do not meet this requirement are removed from the feasible set before optimization; the experiments therefore evaluate cost and budget behavior under quality feasibility rather than claiming answer-quality improvement.

Data management with spouts and bolts: Our system architecture employs spouts and bolts to orchestrate the processing of prompt streams. Utilizing spout components, real-time telemetry data is collected from end devices and edge servers, encompassing device status, inference throughput, and network bandwidth utilization. Through integrated cloud service spouts, cloud API usage and computational resources are monitored. The energy consumption of local devices is monitored through built-in tools like Intel's Power Gadget [23]. Energy and latency data for edge devices are collected using Prometheus [24] integrated with the edge servers' own orchestration systems. Cloud server expenditures are calculated from usage metrics recorded by API dashboards [25].

6.2 Parameter Settings and Baseline Schemes

This section outlines the parameter settings and baseline schemes employed in our CEE collaborative scenario, simulating multiple end-users, multiple edge nodes, and one remote cloud node. End devices with limited long-term financial budgets need to handle heterogeneous inference tasks with different performance preferences.

In the experimental setup, the number of agents varied from 10 to 70. The request-response data volume S_n , measured as the total serialized size of a request and its returned result, was randomly distributed within the range of [0, 100] MB, and the computational density C_n was sampled from [1000, 23000] cycles per bit to reflect varying model complexities. According to the literature [26], the local computational

resources of each end device, $f_{n,l}$, followed a random distribution between [0.5, 4] GHz, and the energy coefficient was set at $\tilde{\kappa}_{n,l} \in [1 \times 10^{-28}, 5 \times 10^{-27}]$. The number of edge servers ranged from 4 to 15, with computational capacities, $f_{n,m}$, spanning [5, 20] GHz. As per references [12, 27], the transmit and receive power for end-users in edge mode was calibrated around $\tilde{p}_n^{ES} = 1,224.78$ mJ per s. The cloud servers' computational capacities, $f_{n,c}$, ranged from [10, 20] GHz, and the power consumption in cloud mode was set at $\tilde{p}_n^{cloud} = 2,605$ mJ per s. Referring to real 5G network datasets [28], the wired data transmission rate in cloud mode, $\hat{R}_{n,c}$, and the wireless transmission rate, $\hat{R}'_{n,c}$, fluctuated within [20, 50] Mbps and [4.35, 6.85] Mbps, respectively. The uplink and downlink rates in edge mode randomly varied between [0.85, 6.45] Mbps. For heterogeneous tasks across different timeslots, the latency weight, λ_n^L , and energy weight, λ_n^E , were normalized within [0, 1] with $\lambda_n^E + \lambda_n^L = 1$. This section compared two levels of information sharing: a cooperative scenario using BRAO, which leverages shared routing decisions among agents; and a limited-information scenario employing MADO, where agents independently optimize routing strategies based on environmental feedback without revealing intent to other agents. To ensure fairness, all computational cost measurements were averaged over multiple trials.

To evaluate the effectiveness of the proposed algorithms regarding convergence, inference cost, and budget balance, we consider the following baselines:

- Local Mode (LM): All inference tasks are processed locally using SLMs without the aid of cloud or edge resources.
- Edge-End Mode (EEM) [29]: Excludes cloud nodes, mandating that tasks are either completed locally or routed to the edge.
- Budget-myopic (BM) [30]: Operates under stringent budget constraints per timeslot to satisfy long-term limits without foreseeing future API pricing or network conditions.
- MASL[31]: An algorithm based on multi-agent stochastic learning that allows end-devices to adaptively select channels in dynamic environments; however, MASL lacks comprehensive action exploration.
- First-available Offloading (FO): Routes each request to the first feasible non-local service within the instantaneous budget, serving as a simple rule-based baseline.
- Centralized Oracle (Oracle): Enumerates all feasible joint routing profiles for small-scale cases and provides a lower bound for the optimality-gap comparison.

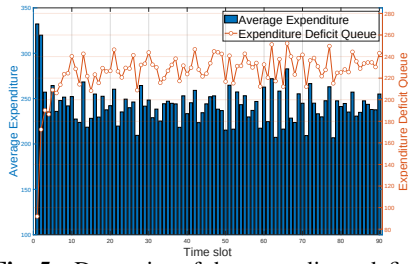


Fig. 5 Dynamics of the expenditure deficit queue

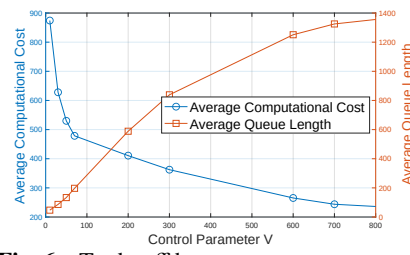


Fig. 6 Trade-off between average computational cost and expenditure

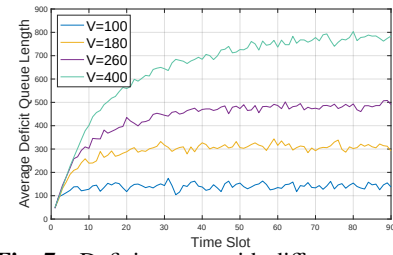


Fig. 7 Deficit queue with different control parameters

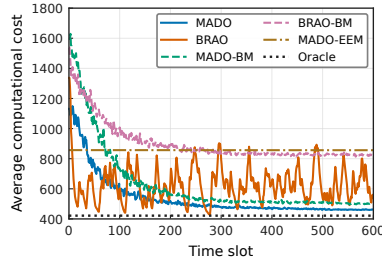


Fig. 8 Average computational cost under different approaches (Oracle baseline)

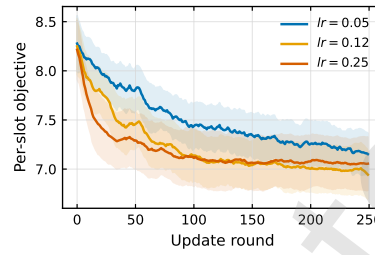


Fig. 9 Impact of learning rate on convergence speed of MADO algorithm

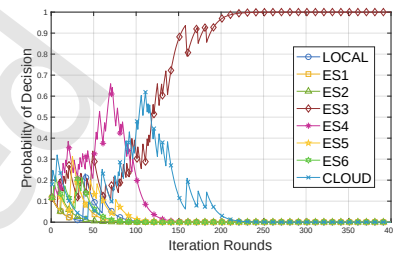


Fig. 10 Convergence process of decisions of MADO algorithm

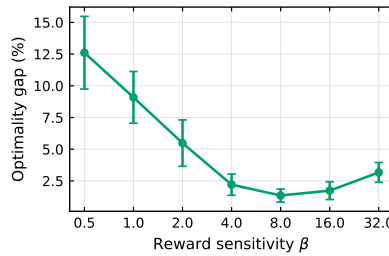


Fig. 11 Impact of the hyperparameter β on the optimality gap of MADO algorithm

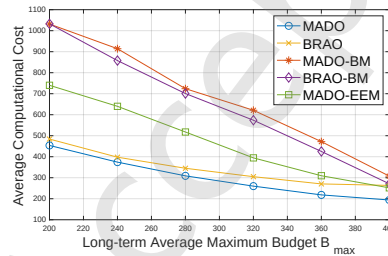


Fig. 12 Impact of the time-averaged maximum budget on average computational cost

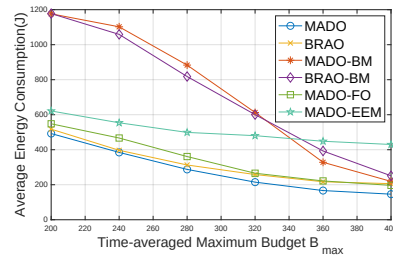


Fig. 13 Impact of time-averaged maximum budget on average energy consumption

6.3 Simulation Experiments and Performance Analysis

Dynamics of budget virtual queue: The experimental results first elucidate the evolving pattern of the expenditure deficit queue under the BRAO algorithm over an extended period. As depicted in Figure 5, the dynamics of the expenditure deficit queue illustrate how long-term API and leasing budget constraints affect inference routing strategies. To navigate these financial limits, BOOF employs Lyapunov optimization to establish an expenditure deficit queue. Figure 5 demonstrates that significant monetary expenditure in timeslot t leads to an increased deficit in timeslot $t + 1$, consequently forcing the agent to rely more on local SLMs or cheaper edge models to reduce subsequent costs. This highlights how the framework adaptively manages the financial budget via the deficit queue to meet long-term constraints.

Trade-off between computational cost and expenditure: Figures 6 and 7 explore the influence of the control parameter V , the latter indicating the backlog of financial deficits. Notably, an increase in V correlates with reduced average inference latency and energy costs, alongside a longer average

length of the deficit queue, suggesting poorer stability. This relationship aligns with the theoretical analysis in Theorem 3, illustrating a trade-off between the inference costs and the deficit queue at $[O(1/V), O(V)]$. By selecting an optimal V value, an effective balance between system performance and budget constraints can be maintained. Figure 7 further confirms that the average length of the deficit queue converges quickly over time, validating the effectiveness of the proposed BOOF framework.

Convergence analysis of different schemes: Figure 8 illustrates the convergence process of average computational costs for various schemes within a single timeslot. The computational cost is highest under the local computing mode, primarily because local SLMs, despite memory-augmentation, have limited compute capacity for highly complex reasoning tasks, resulting in increased delays. BRAO decreases rapidly when shared routing decisions are available; the remaining oscillations mainly come from resource contention and, in synchronous variants, agents reacting to the same edge-load signal at the same time. MADO may show a lower average cost in some dynamic simulations because its stochastic policy update smooths contention and keeps exploring alter-

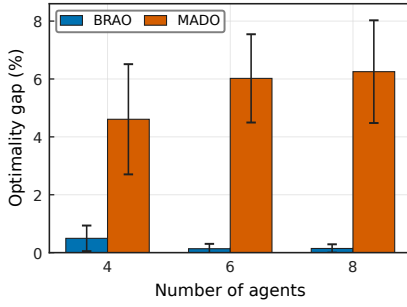


Fig. 14 Optimality gap of BRAO and MADO relative to a centralized oracle

natives. This empirical behavior should not be interpreted as an information-theoretic dominance over BRAO, and it is not used as a substitute for the small-scale Oracle comparison.

Figure 14 reports a small-scale optimality-gap comparison against a centralized Oracle that enumerates all feasible joint routing profiles. The panel answers the reviewer request for an Oracle baseline and uses the name Oracle rather than OPTIMAL. Over 20 fixed random seeds, the mean gaps and 95% confidence intervals are $0.49\% \pm 0.44\%$ (BRAO) and $4.61\% \pm 1.90\%$ (MADO) for $N = 4$; $0.14\% \pm 0.17\%$ and $6.02\% \pm 1.53\%$ for $N = 6$; and $0.14\% \pm 0.14\%$ and $6.25\% \pm 1.77\%$ for $N = 8$. In stationary cooperative slots, BRAO is closer to the Oracle because it uses the shared profile and sequential best responses, while MADO remains a limited-information solver.

Impact of learning rate on convergence speed of MADO:

Figure 9 shows the impact of varying learning rates $lr \in \{0.05, 0.12, 0.25\}$ on MADO. A larger learning rate such as $lr = 0.25$ reaches a stable region in fewer iterations, but it also exhibits larger oscillation amplitude and is more likely to overshoot a near-optimal routing probability. A smaller learning rate such as $lr = 0.12$ converges more slowly, yet its trajectory is more stable and its terminal average cost is lower. We therefore report both the number of rounds needed to enter a stable region and the terminal average cost, and use $lr = 0.12$ in subsequent experiments as a practical operating point that balances speed against overshoot and long-run stability, rather than as a claim of global optimality.

Convergence process of MADO: Figure 10 illustrates the convergence process of a single agent's routing decisions under the MADO algorithm. Initially, MADO explores a range of offloading options extensively, followed by a rapid convergence in later iterations, ultimately culminating in the adoption of the most cost-effective inference layer.

Impact of the hyperparameter β of MADO: Figure 11 reports the sensitivity of MADO to the reward scaling parameter β under the original reward form $\varphi_n^t = 1 - \beta(\Omega_n^t + \zeta^t H_n^t)$.

The operating value is selected from the range used in the archived experiments, where moderate β improves discrimination among actions without collapsing exploration. While MADO is useful in limited-information environments, it requires hyperparameter tuning; BRAO is easier to configure when cooperative decision sharing is available.

Impact of time-averaged maximum budget: Figure 12 examines the effects of the time-averaged maximum budget B_{max} . Higher values of B_{max} facilitate lower average computational costs by allowing agents more diverse routing options, thereby enabling frequent offloading to powerful cloud models for reduced latency. Among the evaluated schemes, MADO achieves the lowest average cost in this dynamic setting, highlighting its adaptability in volatile network and pricing environments, while Figure 14 shows that BRAO is closer to the centralized oracle in small stationary slots. Figures 13 and 15 further show average end-device energy and monetary expenditure under varying B_{max} . The queue mechanism keeps long-term average expenditure aligned with the budget limit in the main dynamic simulations. Figures 13 and 15 report average energy and expenditure under varying B_{max} , while Figures 26 and 28 report the corresponding finite-horizon residuals.

Number of beneficial users: In this experiment, beneficial users are agents experiencing reduced average costs relative to local computation mode. Figure 16 displays the number of beneficial users across different schemes. MADO yields the highest number of beneficial users in the tested dynamic workloads. Specifically, at an agent count of 70, the number of beneficial MADO users increased significantly compared to MADO-EEM, suggesting that decentralized exploration helps more agents find useful edge or cloud opportunities.

Impact of task size: Figure 17 examines the impact of varying request–response data sizes. As S_n increases, computational and communication costs rise across all schemes because larger workloads demand more resources. Due to its adaptability, MADO achieves the lowest average computational cost in this experiment, closely followed by BRAO. Figures 18 and 19 highlight that MADO and BRAO maintain lower average energy consumption and monetary expenditure than budget-myopic baselines for large-request LLM inference workloads.

Trade-off between the maximum budget and the average computational cost under different control parameters:

Figure 20 illustrates the trade-off between average computational overhead and B_{max} under various values of V . For a given B_{max} , a higher V results in a lower average computational overhead, underscoring the impact of V on balancing

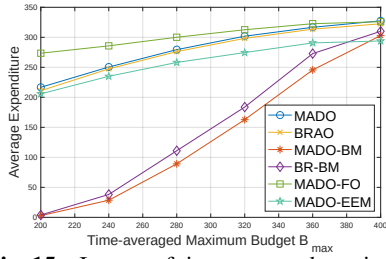


Fig. 15 Impact of time-averaged maximum budget on average expenditure

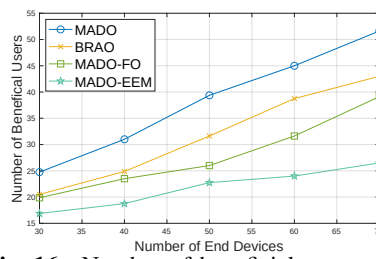


Fig. 16 Number of beneficial users under different approaches

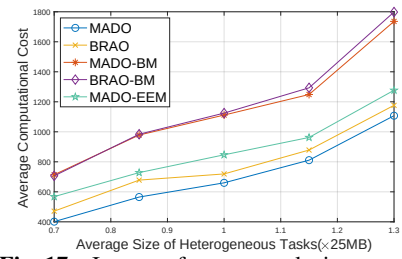


Fig. 17 Impact of average task size on average computational cost

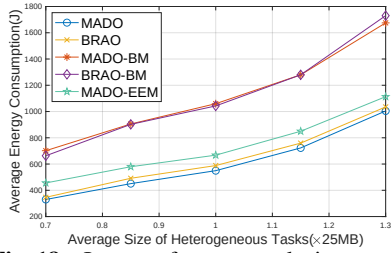


Fig. 18 Impact of average task size on average energy consumption

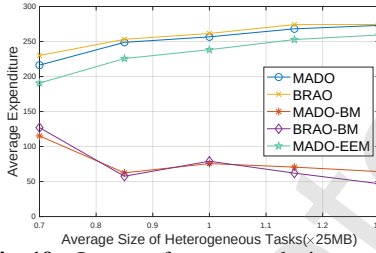


Fig. 19 Impact of average task size on average expenditure

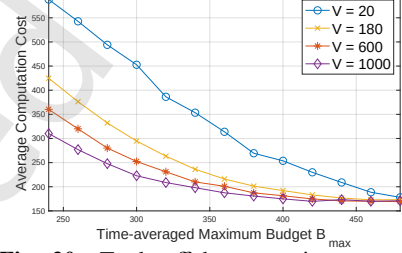


Fig. 20 Trade-off between time-averaged maximum budget and average computational cost with different control parameters

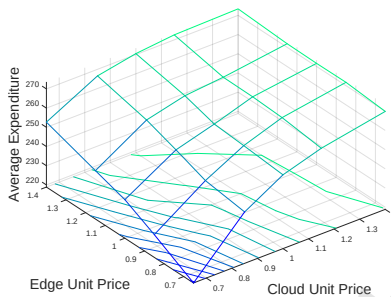


Fig. 21 Impact of service unit price on average expenditure for cloud and edge

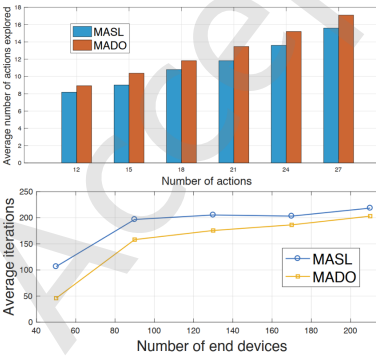


Fig. 22 Comparison of scalability and number of actions explored

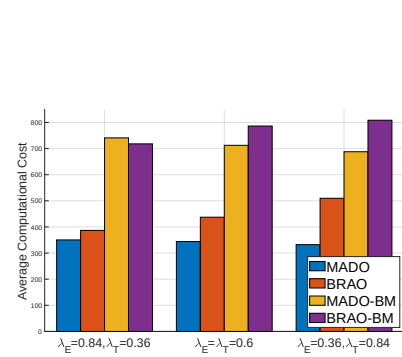


Fig. 23 Impact of performance preference weights on average computational cost

inference latency and monetary expenditure. An increase in B_{max} leads to a decrease in overhead across all values of V , as higher budgets enable more frequent utilization of powerful edge and cloud models.

Figure 21 assesses the impact of the inference service unit prices. Since the edge price uses a load-dependent service-quality discount with a positive admissible range, higher base edge or cloud prices reduce the frequency of external routing and increase local execution. The performance of the BRAO algorithm demonstrates that it can manage average computational costs and monetary expenditures by dynamically adjusting routing strategies in response to fluctuations in API and leasing prices.

Scalability and action exploration: Figure 22 compares the number of explored actions and scalability of MADO and MASL. MADO's stochastic sampling visits a broader set of feasible actions early in learning, while MASL tends to

exploit a narrower subset of actions.

Impact of performance preference weights: Figure 23 presents the overhead as performance preferences vary. Different agents have distinct sensitivities (λ^E for energy and cost, λ^L for latency). Remarkably, the MADO algorithm shows insensitivity to weight changes. This resilience stems from the algorithm's capacity to allow agents to autonomously learn optimal policies through environmental feedback. Conversely, other schemes exhibit significant fluctuations in average costs, highlighting the stability of the proposed schemes when managing diverse agentic requirements in real-time CEE deployments.

Multiple random seeds: Figure 24 reports the final per-slot objectives of LM, EEM, BM, BRAO, and MADO over 20 fixed random seeds, with 95% confidence intervals. BRAO attains 6.44 ± 0.20 and MADO attains 6.72 ± 0.21 , both below LM (7.27 ± 0.20), EEM (9.63 ± 0.15), and BM (10.73 ± 0.47).

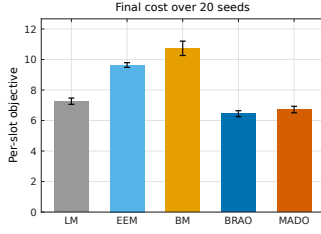


Fig. 24 Final cost over 20 seeds.

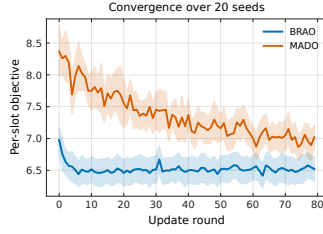


Fig. 25 Mean convergence over 20 seeds.

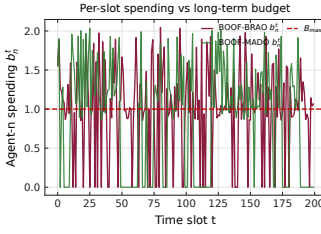
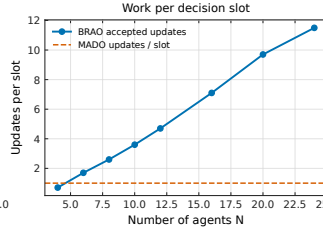
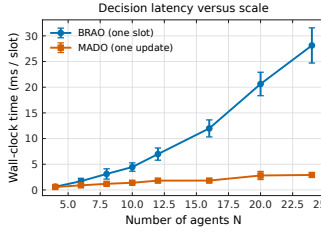
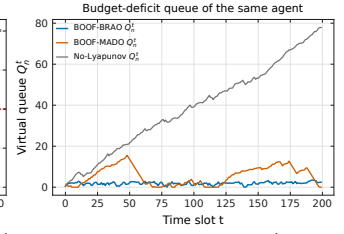
Fig. 26 Per-slot spending b_n^t versus $B_{\max}$ and queue Q_n^t .

Fig. 27 Decision latency and per-slot update count versus the number of agents.

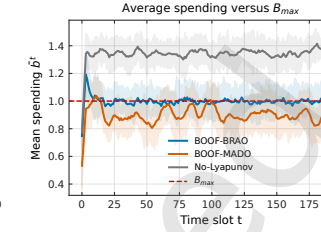


Fig. 28 Finite-horizon budget residuals and normalized queue backlogs.

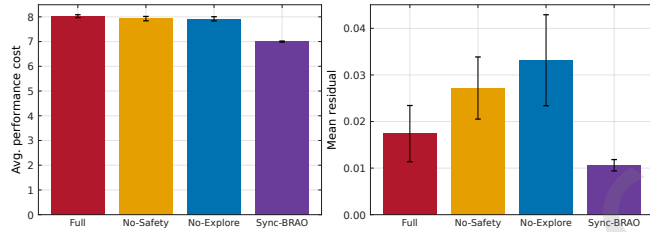
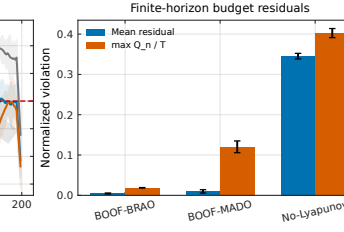
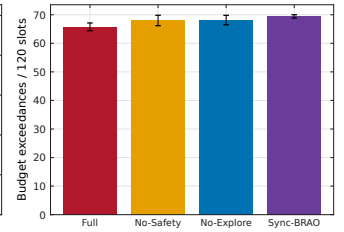
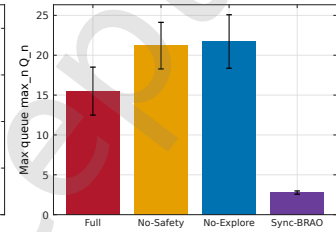


Fig. 29 Four-metric ablation of BOOF and MADO components over 20 seeds.



The ranking is therefore stable across seeds rather than an artifact of a single realization. Figure 25 further shows the mean per-slot objective trajectories of BRAO and MADO with 95% confidence bands, confirming that both solvers enter a stable low-cost region within the reported horizon.

Budget-violation measurements: The long-term constraint is $\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T b_n^t \leq B_{\max}$, not a per-slot cap $b_n^t \leq B_{\max}$. Figure 26 therefore plots one agent's instantaneous spending b_n^t against $B_{\max} = 1$ together with the same agent's virtual queue Q_n^t . Instantaneous spikes above the red line are expected; what matters is that BOOF-BRAO and BOOF-MADO keep Q_n^t bounded, whereas a myopic No-Lyapunov controller lets the deficit grow. Figure 28 summarizes the finite-horizon residual $[\bar{b}_n - B_{\max}]_+$. Over $T = 200$ slots the mean residuals are 0.0045 (BOOF-BRAO), 0.0098 (BOOF-MADO), and 0.345 (No-Lyapunov). The small positive residuals of the proposed solvers are consistent with a finite-horizon Lyapunov bound and do not indicate systematic budget violation.

Solver overhead versus the number of agents: Wall-clock decision latency was measured on a laptop with an Intel Core i7-12700H CPU (2.30 GHz), 32 GB RAM, Windows 11, Python 3.12 and NumPy 2.2.4, using a single thread and `time.perf_counter`. Figure 27 reports milliseconds

per slot. BRAO grows from 0.29 ms at $N = 4$ to 9.71 ms at $N = 24$, consistent with sequential best-response search. MADO remains below 1 ms per slot across the same range because each slot performs one local probability update.

Ablation: Figure 29 isolates the Lyapunov safety term, exploration, and synchronous BRAO updates. Removing the queue (No-Safety) slightly lowers the performance cost but raises the mean residual from 0.017 to 0.027 and the maximum queue from 15.5 to 21.2. Disabling exploration similarly inflates the deficit queue. Synchronous BRAO attains the lowest cost and the tightest queue among the tested variants, which is consistent with shared-information best response. The ablation therefore supports keeping BOOF in the loop: the safety term is the component that enforces the long-term budget.

In summary, the BRAO algorithm swiftly reaches an NE in environments where routing intent is shared and is closer to the oracle in small stationary cases. Conversely, the MADO algorithm autonomously updates its offloading policy via interactive feedback, exhibiting high adaptability in dynamic limited-information environments where inter-agent intent sharing is restricted.

7 Conclusion

In this work, we investigate the efficient online inference offloading problem for multi-agent systems in CEE architectures, focusing on minimizing computation costs under long-term operational budget constraints and layered information-sharing restrictions. We develop a framework utilizing Lyapunov optimization and potential game theory to decouple long-term stochastic API and leasing budget constraints into single-time-slot decisions. We propose BRAO for cooperative agents with decision-sharing capabilities, and MADO for privacy-sensitive environments where intent and routing sharing are restricted. Under explicit boundedness, feasibility, and approximation assumptions, our scheme provides performance bounds and convergence properties. Finally, we discuss the integration of our framework into real-world distributed LLM inference systems. Extensive experiments demonstrate that our scheme effectively reduces the computational and communication overhead of heterogeneous agentic tasks while achieving a favorable trade-off between inference performance and long-term financial expenditure. The proposed approach is currently evaluated in CEE scenarios with a single centralized cloud provider; in the future, we plan to explore multi-cloud collaboration and task-quality-aware mixture-of-experts (MoE) routing.

Acknowledgment

This work was supported in part by China Telecom under Grant No. T-2025-15. The authors would also like to thank the anonymous reviewers for their valuable comments and constructive suggestions, which have helped to improve the quality of this paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Electronic Supplementary Material

The source code and additional experimental data associated with this study will be made available upon the final acceptance and publication of this manuscript.

Reference

- [1] M. Peng, N. Jiang, and S. Yan, Communication-computing integrated wireless networks: from cloud-based collaboration to integrated sensing and intelligent coordination, *Journal on Communications*, vol. 47, no. 05, pp. 242–265, 2026.
- [2] R. Xing, S. Guo, X. Qiu *et al.*, Service function chain orchestration mechanism on embedded DPU, *Journal on Communications*, vol. 47, no. 04, pp. 13–28, 2026.
- [3] X. Zhang, J. Nie, Y. Huang, G. Xie, Z. Xiong, J. Liu, D. Niyato, and X. Shen, Beyond the cloud: Edge inference for generative large language models in wireless networks, *IEEE Transactions on Wireless Communications*, vol. 24, no. 1, pp. 643–658, 2025.
- [4] J. Yang, Q. Wu, Z. Feng, Z. Zhou, D. Guo, and X. Chen, Quality-of-service aware llm routing for edge computing with multiple experts, *IEEE Transactions on Mobile Computing*, vol. 24, no. 12, pp. 13 648–13 662, 2025.
- [5] E. Dinçer and Z. Kilimci, Real-time and offline large language models on edge devices: a systematic review, *PeerJ Computer Science*, vol. 12, p. e3769, 05 2026.
- [6] Z. Qu, W. Wang, Z. Yu, B. Sun, Y. Li, and X. Zhang, Llm enabled multi-agent system for 6g networks: Framework and method of dual-loop edge-terminal collaboration, *IEEE Communications Magazine*, vol. 63, no. 3, pp. 54–60, 2025.
- [7] Y. He, J. Fang, F. R. Yu, and V. C. Leung, Large language models (llms) inference offloading and resource allocation in cloud-edge computing: An active inference approach, *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 11 253–11 264, 2024.
- [8] R. Ren, Y. Wu, X. Zhang, J. Ren, Y. Shen, S. Wang, and K.-F. Tsang, Retrieval-augmented generation for mobile edge computing via large language model, arXiv:2412.20820, 2024. [Online]. Available: <https://arxiv.org/abs/2412.20820>
- [9] A. Qayyum, A. Albaser, J. Qadir, A. Al-Fuqaha, and M. Abdallah, Llm-driven multi-agent architectures for intelligent self-organizing networks, *IEEE Network*, vol. 40, no. 4, pp. 258–267, 2026.
- [10] C. Wang, S. Zhang, Y. Chen, Z. Qian, J. Wu, and M. Xiao, Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics, in *IEEE Conference on Computer Communications (INFOCOM)*, 2020, pp. 257–266.
- [11] Q. Liu and T. Han, Dare: Dynamic adaptive mobile augmented reality with edge computing, in *IEEE International Conference on Network Protocols (ICNP)*. IEEE, 2018, pp. 1–11.
- [12] S. Yuan, J. Li, H. Chen, Z. Han, C. Wu, and Y. Zhang, Jira: Joint incentive design and resource allocation for edge-based real-time video streaming systems, *IEEE Transactions on Wireless Communications*, vol. 22, no. 5, pp. 2901–2916, May 2023.
- [13] R. Gholivand and Z. Movahedi, A cloud-ran based end-to-end computation offloading in mobile edge computing, *Computer Communications*, vol. 175, no. 1, pp. 193–204, 2021.
- [14] B. Dai, J. Niu, T. Ren, and M. Atiquzzaman, Toward mobility-aware computation offloading and resource allocation in end-edge-cloud orchestrated computing, *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 19 450–19 462, 2022.
- [15] Y. Ding, K. Li, C. Liu, and K. Li, A potential game theoretic approach to computation offloading strategy optimization in

- end-edge-cloud computing, *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1503–1519, 2021.
- [16] C. Sun, H. Li, X. Li, J. Wen, Q. Xiong, X. Wang, and V. C. Leung, Task offloading for end-edge-cloud orchestrated computing in mobile networks, in *IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2020, pp. 1–6.
- [17] H. Wang, B. Chang, and J. Luo, Edge caching strategy based on multi-agent deep reinforcement learning in cloud-edge-end scenarios, *Journal on Communications*, vol. 46, no. 06, pp. 153–167, 2025.
- [18] F. Khoramnejad, A. Syed, W. S. Kennedy, and M. Erol-Kantarci, Stability and accuracy-aware learning for task offloading in uav-mec-assisted smart farms, *IEEE Transactions on Network and Service Management*, vol. 21, no. 5, pp. 5647–5661, 2024.
- [19] K. Yamada, T. Hara, and S. Kasahara, Repeated stochastic game and lyapunov optimization for mining task offloading in decentralized applications, *IEICE Transactions on Communications*, vol. E107-B, no. 12, pp. 936–944, 2024.
- [20] Neely and M. J, Stochastic network optimization with application to communication and queueing systems, *Synthesis Lectures on Learning, Networks, and Algorithms (SLLNA)*, vol. 3, no. 1, pp. 10–146, 2010.
- [21] Y. H. Chew, B.-H. Soong *et al.*, *Potential Game Theory*. Springer, 2016.
- [22] M. Senn, Apache Storm, 2023, accessed: 2023-March-18. [Online]. Available: <https://storm.apache.org/>
- [23] Gadget, Intel power gadget tools, 2023, accessed: 2023-July-07. [Online]. Available: <https://www.intel.com/content/www/us/en/developer/articles/tool/power-gadget.html>
- [24] Prometheus, <https://prometheus.io/>, accessed: 2023-July-03.
- [25] CloudWatch, Amazon cloudwatch alarms, 2023, accessed: 2023-July-07. [Online]. Available: <https://docs.aws.amazon.com/cloudwatch/>
- [26] S. Yuan, B. Dong, H. Lv, H. Liu, H. Chen, C. Wu, S. Guo, Y. Ding, and J. Li, Adaptive incentive for cross-silo federated learning in iiot: A multiagent reinforcement learning approach, *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 15 048–15 058, 2024.
- [27] Y. Chen, J. Zhao, Y. Wu, J. Huang, and X. S. Shen, Qoe-aware decentralized task offloading and resource allocation for end-edge-cloud systems: A game-theoretical approach, *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 769–784, 2024.
- [28] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan, Beyond throughput, the next generation: A 5g dataset with channel and context metrics, in *ACM Multimedia Systems (MMSys)*, 2020, pp. 303–308.
- [29] S. Yuan, J. Li, and C. Wu, Jora: Blockchain-based efficient joint computing offloading and resource allocation for edge video streaming systems, *Journal of Systems Architecture: Embedded Software Design*, vol. 133, no. 40, pp. 1–12, 2022.
- [30] E. Wang, H. Wang, P.-M. Dong, Y.-B. Xu, and Y.-J. Yang, Distributed game-theoretical d2d-enabled task offloading in mobile edge computing, *Journal of Computer Science and Technology*, vol. 37, no. 4, pp. 919–941, 2022.
- [31] J. Zheng, Y. Cai, Y. Wu, and X. Shen, Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach, *IEEE Transactions on Mobile Computing*, vol. 18, no. 4, pp. 771–786, Apr. 2019.

Author biography



Shijing Yuan (Member of IEEE) received his Ph.D. in Computer Science and Technology from Shanghai Jiao Tong University in 2024, honored as an Outstanding Doctoral Graduate. He is with the China Telecom Research Institute. His research focuses on large-scale LEO satellite network architecture, 6G multimedia communications, Edge AI, and blockchain. He leads several national and ministerial-level research projects, including grants from MIIT and the Zhejiang International Youth Talent Fund. Dr. Yuan has authored over 20 first-authored papers in top-tier IEEE journals/conferences such as IEEE Transactions on Computers, IEEE Transactions on Mobile Computing, and IEEE INFOCOM. He actively serves the community as a Youth Editorial Board Member for Telecommunications Science, a TPC member for IEEE WCNC 2026 Workshop, a former Session Chair for IEEE ICC 2025, and a regular reviewer for multiple prestigious IEEE/Elsevier journals.



Zihang Li is currently pursuing the B.E. degree in electronic and computer engineering with Shanghai Jiao Tong University, Shanghai, China. He joined Global College, Shanghai Jiao Tong University in 2024. His research interests include large language models, multi-agent systems, and online optimization in edge computing.



Yuming Zhang (Member, IEEE) received the Ph.D. degree in telecommunications and information system from the National Engineering Laboratory for Next Generation Internet Interconnection Devices, Beijing Jiaotong University, Beijing, China, in 2021. He served as a Post-Doctor with the School of Computer Science, Beijing University of Posts and Telecommunications, from 2021 to 2024. He is currently a Lecturer with the School of Electronic and Information Engineering, Beijing Jiaotong University. His current research interests include UAV networks, mobile edge computing, queue theory, and machine learning.