Cognitive GenAI-as-a-Service Orchestration in Edge Networks: A Value-Adaptive Multi-Agent Learning Framework

Yan Gao, Shaoyuan Huang, Minglai Shao, Huaming Wu, *Senior Member, IEEE*, and Yansha Deng, *Senior Member, IEEE*

Abstract*—Deploying Generative AI-as-a-Service (GenAIaaS) at the network edge enables ubiquitous intelligence but faces severe stability challenges, primarily stemming from the lack of cognitive awareness regarding stochastic inference latency spikes and the coordination complexity of interdependent microservice chains. To bridge these gaps under strict Quality of Service (QoS) constraints, we propose value-adaptive multi-agent proximal policy optimization (VAMAPPO) as a novel cognitive orchestration framework. Unlike centralized approaches, VAMAPPO treats orchestration as a decentralized cognitive decision process. It integrates a variational inference-based uncertainty quantification mechanism, empowering agents to perceive environmental volatility and dynamically optimize learning steps to prevent performance degradation. To further solve the ambiguity in distinguishing individual contributions within sequential service chains, we design a coordinated advantage function (CAF). By utilizing counterfactual baselines to estimate marginal impacts, CAF effectively disentangles complex inter-agent dependencies. Extensive experiments on real-world network topologies with trace-driven workloads demonstrate that our cognitive framework significantly outperforms state-of-the-art baselines. Specifically, VAMAPPO achieves up to a 97.0% service success rate and reduces response latency by 24.2% while satisfying stringent service level agreement (SLA) constraints, exhibiting superior robustness in zero-shot generalization across unseen edge environments. Code and data are available at *https://github.com/gymorsiback/Value-Adaptive-Multi-Agent-PPO

***Index Terms*—Generative AI services, service orchestration, multi-agent reinforcement learning, quality of service, edge networks.**

I. Introduction

In RECENT years, the paradigm of Artificial Intelligence has shifted from monolithic model deployment to Generative AI-as-a-Service (GenAIaaS) [1]. This paradigm [2] allows mobile users to access foundation models via APIs for diverse applications, ranging from text generation to image synthesis [3]. Unlike traditional inference tasks, real-world GenAIaaS requests typically manifest as complex workflows [4] requiring the composition of multiple interdependent microservices [5]. For instance, generating an illustrated storybook involves a chain: a large language model (LLM) first structures the narrative, followed by prompt refinement agents, and finally, a text-to-image model synthesizes the visual content. Orchestrating these heterogeneous components demands cognitive intelligence—specifically, the capacity to perceive environmental states, reason about uncertainties, and autonomously adapt strategies—to ensure precise coordination.

While cloud-centric frameworks [2], [6] demonstrate the feasibility of multi-model collaboration, migrating these services to the network edge offers significant advantages, including reduced end-to-end latency and enhanced user privacy [7]. Consequently, deploying GenAIaaS at the edge has become a critical trend. However, this transition introduces severe challenges. Edge environments are resource-constrained and bandwidth-limited, an infrastructure that lacks scalability. Crucially, the inference latency of GenAI services is highly stochastic, heavily depending on the fluctuating length of input tokens and the decoding complexity [8]. In distributed edge environments, this uncertainty, coupled with dynamic network conditions, makes static scheduling algorithms ineffective.

Recent research has explored resource scheduling for general edge tasks. Works such as EdgeShard [9] and EdgeAIBus [10] have optimized distributed inference for specific LLM layers. Similarly, multi-agent reinforcement learning (MARL) approaches have been applied to UAV trajectory design [11] and intent-driven task scheduling [12] for 6G networks. While providing methodological baselines, these general-purpose works do not address the stochastic nuances of GenAIaaS. Despite these advancements, existing approaches face two main cognitive limitations in the context of GenAIaaS:

- **Lack of cognitive uncertainty quantification:** Most methods [11], [12] assumed deterministic computational models, failing to quantify the decision risk caused by the stochastic nature of GenAI workloads. This inability to perceive uncertainty often leads to policy instability and service level agreement (SLA) violations during runtime latency spikes.
- **Inefficient credit assignment in service chains:** In a multi-stage service workflow, traditional MARL struggles to identify the distinct impact of individual microservices

Received 29 December 2025; revised 24 March 2026 and 29 April 2026; accepted 18 May 2026. Date of publication 25 May 2026; date of current version 28 May 2026. This work was supported in part by the Natural Science Foundation of Tianjin under Grant 25JCYBJC01540. The associate editor coordinating the review of this article and approving it for publication was J. Kang. *(Corresponding author: Minglai Shao.)*

Yan Gao and Minglai Shao are with the School of New Media and Communication, Tianjin University, Tianjin 300072, China (e-mail: gymorsiback@tju.edu.cn; shaoml@tju.edu.cn).

Shaoyuan Huang is with the College of Intelligence and Computing, Tianjin University, Tianjin 300072, China (e-mail: hsy_23@tju.edu.cn).

Huaming Wu is with the Center for Applied Mathematics, KL-AAGDM, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Yansha Deng is with the Department of Engineering, King's College London, WC2R 2LS London, U.K. (e-mail: yansha.deng@kcl.ac.uk).

Digital Object Identifier 10.1109/TCCN.2026.3696248

on the aggregate service utility. This ambiguity results in suboptimal coordination and misaligned local objectives.

To bridge these gaps, we propose a cognitive orchestration framework powered by value-adaptive multi-agent proximal policy optimization (VAMAPPO). This framework treats the orchestration of interdependent GenAI services as a decentralized cognitive decision process. At its core, VAMAPPO introduces a variational inference-based mechanism to endow agents with cognitive awareness, allowing them to explicitly quantify the uncertainty in policy evaluation. By modeling network parameters as probability distributions, agents can dynamically adjust their learning steps based on environmental perception, thereby guaranteeing robustness against latency volatility. Parallel to uncertainty management, to solve the coordination challenge in service chains, we design a coordinated advantage function (CAF) that utilizes counterfactual reasoning to distinguish and optimize the impact of each agent on the global service utility. The main contributions are summarized as follows:

- We propose VAMAPPO, a novel framework that integrates variational inference into multi-agent policy optimization. By introducing stochastic parameterization, the framework enables agents to perceive and robustly adapt to the stochastic variability in inference latency, preventing performance degradation in highly dynamic edge networks.
- To resolve the complexity of interdependent microservice chains, we design a coordinated advantage function. Utilizing counterfactual reasoning, CAF disentangles the specific contribution of each agent from the global service utility. This mechanism effectively supports the resolution of the credit assignment problem in sequential workflows, ensuring that upstream decisions are optimized in concert with downstream outcomes.
- To ensure efficient and stable cognitive learning in bandwidth-constrained edge environments, we introduce an adaptive clipping mechanism coupled with gradient compression. This approach minimizes communication overhead while dynamically adjusting the trust region based on historical policy updates to balance exploration and stability.
- We evaluate our framework using trace-driven workloads derived from real-world cluster data and profiled GenAI model characteristics. Extensive experiments across diverse network topologies demonstrate that our approach reduces response time by 24.2% and maintains a 97.0% service success rate under stringent SLA constraints, outperforming state-of-the-art baselines.

The remainder of this paper is organized as follows: Section II reviews related work; Section III models the system; Section IV details the VAMAPPO framework; Section V analyzes experimental results; and Section VI concludes.

II. Related Work

This section reviews recent advancements in two key areas: the orchestration of Generative AI services at the edge and intelligent service scheduling in next-generation networks. To clearly distinguish our contributions, we provide a comprehensive functional comparison between the proposed VAMAPPO framework and existing state-of-the-art approaches in Tab. I.

A. Edge Intelligence Services

As LLMs and AIGC models become ubiquitous, orchestrating them as services has attracted significant attention [16]. Early frameworks [2], [6] utilize LLMs as central controllers to dispatch tasks to expert models. However, these cloud-centric approaches often incur high latency. To address this, recent research focuses on collaborative edge-cloud inference.

EdgeShard [9] proposes a collaborative edge computing framework that optimizes LLM inference by sharding model layers across distributed devices, significantly reducing memory footprints. Similarly, EdgeAIBus [10] introduces an AI-driven framework for joint container management and model selection, enabling efficient provisioning of heterogeneous edge services. In the context of complex workflows, Navardi et al. [17] propose a meta-reasoning approach for edge-cloud collaborative LLM planning, focusing on autonomous navigation tasks. Furthermore, Yao et al. [13] present a diffusion-based MARL approach to enhance the QoS of LLM services through edge-cloud collaboration, addressing the high-dimensional action space in service dispatching [18].

While these works have successfully optimized resource allocation for specific AI tasks, e.g., layer offloading or container placement, they predominantly focus on static or predictable workload models. Most existing frameworks lack explicit mechanisms to quantify the runtime uncertainty inherent in GenAI inference, e.g., token-length dependent latency variations, which is critical for maintaining strict SLA guarantees in dynamic edge environments.

B. Intelligent Service Scheduling in Edge Networks

Beyond AI models, the broader problem of intelligent task scheduling in 6G and edge networks provides the foundational methodology for our work. MARL has emerged as a powerful tool for solving decentralized resource allocation problems.

Recent studies have demonstrated the efficacy of MARL in diverse network scenarios. Guan et al. [11] utilize a Multi-Agent PPO method to optimize cooperative UAV trajectories for emergency communications, demonstrating robust coordination capabilities. For intent-driven networks, Wang et al. [12] propose an adaptive task scheduling framework for 6G. Specific to edge computing orchestration, Han et al. [14] applied MAPPO to manage communication-dependent computing tasks, establishing a strong baseline for decentralized decision-making with continuous action spaces. Furthermore, addressing the dynamics of augmented reality (AR) applications, Qian and Coutinho [15] proposed the META (METAPPO) orchestrator, which incorporates task migration mechanisms to optimize resource utilization across networked edge servers. In the domain of network function virtualization, Wen et al. [19] develop an orchestration strategy for network function chains, while Zhang et al. [20] focus on resilient task scheduling in vehicular edge computing.

TABLE I
FUNCTIONAL COMPARISON OF THE PROPOSED VAMAPPO WITH STATE-OF-THE-ART APPROACHES

Reference	Focus Domain	Decentralized Arch.	Cognitive Uncertainty Awareness	Sequential Credit Assignment	Key Technique
HuggingGPT [2]	Cloud AI Services	–	–	–	Centralized LLM Controller
EdgeShard [9]	LLM Layers	✓	–	–	Distributed Heuristics
EdgeAIBus [10]	Heterogeneous Containers	–	–	–	AI-driven Joint Optimization
Yao et al. [13]	AIGC Services	✓	(∼)	–	Diffusion-based MARL
MAPPO [14]	General Edge Tasks	✓	–	–	Standard Multi-Agent PPO
METAPPO [15]	AR Rendering Tasks	✓	–	–	Optimization-based MARL
VAMAPPO (Ours)	**GenAI Microservice Chains**	✓	✓	✓	**Variational Inference + Counterfactual CAF**

Notes: ✓: Supported; –: Not supported or not explicitly addressed; (∼): Partially supported (implicitly handled by diffusion noise).

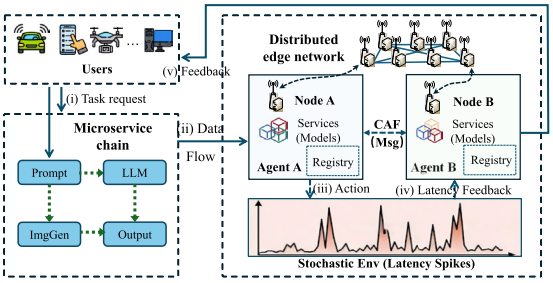


Fig. 1. Overview of the QoS-aware GenAIaaS orchestration framework. User requests from diverse terminals (i) are modeled as interdependent microservice chains. The data flow (ii) is orchestrated by distributed VAMAPPO agents co-located with edge nodes. Agents take actions (iii) based on local states and coordinate via lightweight CAF messages. The environment provides feedback on stochastic latency spikes (iv), closing the RL loop and ensuring QoS-aware results are delivered back to users (v). Offline QoS registries are utilized to minimize runtime profiling overhead.

Although methods like MAPPO and METAPPO demonstrate the superiority of MARL in managing network resources, they typically rely on deterministic policy evaluations or standard advantage functions. In the specific context of GenAIaaS microservice chains, traditional MARL struggles to: 1) adapt quickly to the high variance of generative tasks, the stability issue, and 2) accurately assign credit to individual microservices within a sequential chain, the coordination issue. Our VAMAPPO framework addresses these limitations by integrating variational inference for uncertainty quantification and counterfactual baselines for precise credit assignment.

III. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a collaborative edge computing environment designed for GenAIaaS. The system orchestrates complex content generation requests from mobile users by composing distributed GenAI microservices.

A. Heterogeneous Service Network Model

Fig. 1 illustrates the proposed GenAIaaS orchestration architecture. Unlike centralized cloud paradigms, the system comprises a set of geographically distributed edge service nodes, denoted as $\mathcal{S} = \{1, \ldots, S\}$. These nodes form a heterogeneous network topology graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{E}$ represents the set of communication links enabling inter-service data transmission.

Each edge node $s \in \mathcal{S}$ is characterized by its specific hardware configuration, e.g., tensor cores and VRAM capacity, and a service capability set, denoted as $\mathcal{M}_s \subseteq \mathcal{M}$. Here, $\mathcal{M} = \{1, \ldots, M\}$ represents the universal set of GenAI service types, e.g., LLaMA-7B, Stable Diffusion, or ControlNet. A binary indicator $\mathbb{I}(m \in \mathcal{M}_s)$ equals 1 if service instance m is deployed and active on node s, and 0 otherwise.

From the user's perspective, a request is modeled as a sequential microservice chain, represented by an ordered set of stages $\mathcal{I} = \{1, \ldots, I\}$. As shown in Fig. 1, stage $i \in \mathcal{I}$ corresponds to a specific generative sub-task that transforms an input tensor of size $D_{in}^{(i)}$ into an output tensor of size $D_{out}^{(i)}$. The orchestration objective is to map these interdependent stages onto appropriate edge nodes while satisfying strict SLA constraints under environmental uncertainty. For reader convenience, Tab. II provides a comprehensive list of notations used throughout this paper.

B. Communication Model

The execution of a microservice chain involves continuous data transmission between edge service nodes. We adopt a directed link-based transmission model to strictly avoid the redundant delay calculation observed in prior works. Let binary decision variable $x_{i,s} \in \{0, 1\}$ indicate whether service stage i is assigned to edge node s. The communication latency for stage i, denoted as $T_{comm}^{(i)}$, is defined specifically as the time required to transmit the output data of the predecessor stage $i-1$ from its host node to the node assigned for stage i. Mathematically, this is expressed as:

$$T_{comm}^{(i)} = \sum_{s \in \mathcal{S}} \sum_{s' \in \mathcal{S}} x_{i-1,s} \cdot x_{i,s'} \cdot \frac{D_{out}^{(i-1)}}{R_{s,s'}}, \quad \forall i \in \{2, \ldots, I\}, \tag{1}$$

where $D_{out}^{(i-1)}$ represents the output data size of the previous stage, and $R_{s,s'}$ denotes the effective average data transmission rate between node s and node s'. We consider two special boundary cases:

Initial upload ($i = 1$): $T_{comm}^{(1)}$ represents the transmission latency for uploading the user's initial request data, e.g., prompt text or reference image, to the first edge node.

TABLE II
SUMMARY OF KEY NOTATIONS

Notation	Definition	Notation	Definition
System Model & Problem Formulation			
$\mathcal{S}$	Set of edge service nodes	$\mathcal{F}_m(\cdot)$	Deterministic profiling function for model m
$\mathcal{M}$	Universal set of GenAI models	$\delta_{s,t}$	Stochastic runtime noise factor
$\mathcal{I}$	Microservice chain	Q_m	Registered quality score of model m
$\mathcal{M}_s$	Local service capability set of node s	$T_{comm}^{(i)}$	Communication latency for stage i
$D_{in}^{(i)}$	Input tensor data size for stage i	$T_{comp}^{(i)}$	Computation latency for stage i
$R_{s,s'}$	Effective data transmission rate	$T_{max}^{(i)}$	SLA soft deadline for stage i
$x_{i,s}$	Binary variable: assignment of stage i to node s	$\mathcal{U}_{total}^{(i)}$	Composite service utility, SLA + Quality
$y_{i,m}$	Binary variable: selection of model m for stage i	ω	Weight balancing quality and latency
VAMAPPO Algorithm			
s_t^a	Composite state vector for agent a	α_t^a	Value-adaptive modulation factor
$\mathbf{a}_t^a$	Action vector, node and service selection	w_t^a	Policy enhancement weight
r_t^a	Reward function based on SLA utility	U_t^a	Decision uncertainty metric
θ^a	Policy network weights	Δ_t^a	Environmental volatility metric
μ, σ	Variational parameters, Mean, Std Dev	C_t^a	Inter-agent consistency metric
ϕ^a	Critic network parameters	A_{coord}^a	Coordinated Advantage Function (CAF)
π_θ	Stochastic policy distribution	$A_{inf}^{a\to b}$	Counterfactual influence of a on b
V_ϕ	Value function approximation	$\mathcal{N}_{out}(a)$	Set of successor agents in the chain
$\mathcal{D}$	Experience replay buffer	ϵ_t	Adaptive clipping parameter
$\mathcal{D}_{KL}$	KL divergence regularization term	ρ^*	Gradient compression ratio

Co-location ($s = s'$): If consecutive microservices are orchestrated on the same edge node, the data transfer occurs via intra-node memory copy, e.g., GPU VRAM copy. Since this speed is orders of magnitude faster than network transmission, we assume $T_{comm}^{(i)} \approx 0$ in such cases.

This model simplifies the physical layer complexities, e.g., channel fading, into an effective bandwidth abstraction $R_{s,s'}$, allowing the orchestration framework to focus on logical data flow optimization.

C. Profiling-Based Computation Model With Uncertainty

Unlike traditional web services often modeled by M/M/1 queuing theory [21], the inference latency of GenAI services is computationally intensive and highly dependent on input characteristics, rather than following simple exponential distributions. To capture this reality, we propose a Data-Driven Computation Model grounded in offline profiling.

Let binary decision variable $y_{i,m} \in \{0, 1\}$ indicate whether microservice stage i utilizes GenAI model m. We model the computation latency $T_{comp}^{(i)}$ as a composite of a deterministic baseline and a stochastic runtime variance.

1) Deterministic Baseline: First, the baseline inference time is determined by the specific complexity of model m and the input workload size $D_{in}^{(i)}$ (where $D_{in}^{(i)} = D_{out}^{(i-1)}$). We define a profiling function $\mathcal{F}_m(\cdot)$ derived from offline benchmarks:

$$T_{base}^{(i,m)} = \mathcal{F}_m(D_{in}^{(i)}), \tag{2}$$

where $\mathcal{F}_m$ captures the heterogeneous characteristics of different GenAI tasks. For example, for LLMs, $\mathcal{F}_m$ is typically linear regarding the number of input tokens and generated tokens; for diffusion models, it correlates with image resolution and denoising steps.

2) Stochastic Runtime Uncertainty: In real-world distributed edge environments, inference performance is subject to significant fluctuations due to background process interference, thermal throttling, and resource contention [8]. To address the reviewer's concern regarding "runtime spikes," we explicitly model this service uncertainty:

$$T_{comp}^{(i)} = \sum_{s\in\mathcal{S}} \sum_{m\in\mathcal{M}} x_{i,s} \cdot y_{i,m} \cdot \left[T_{base}^{(i,m)} \cdot (1 + \delta_{s,t}) \right], \tag{3}$$

where $\delta_{s,t} \geq 0$ represents the runtime noise factor on node s at the current scheduling time step t. This factor is modeled as a time-variant stochastic variable to simulate sudden spikes in GPU usage.

The introduction of $\delta_{s,t}$ transforms the scheduling problem from a deterministic optimization into a stochastic decision process, necessitating the uncertainty-quantification mechanism in our proposed VAMAPPO algorithm.

3) Total Response Latency: Finally, the total response latency for stage i, denoted as T_i, is the sum of the data arrival time and the service execution time:

$$T_i = T_{comm}^{(i)} + T_{comp}^{(i)}, \tag{4}$$

note that $T_{comm}^{(i)}$ accounts for the data transfer from the previous stage, ensuring a logical sequential execution flow.

D. Service Capability Profiling and Registration

To enable intelligent model selection without incurring prohibitive runtime overhead, we decouple the quality evaluation from the online scheduling process. We propose a two-phase mechanism: offline profiling and online registration.

1) Offline Capability Benchmarking: Upon service onboarding, each GenAI model $m \in \mathcal{M}$ undergoes a rigorous evaluation using standardized datasets to derive a unified quality score $Q_m \in [0, 1]$. We employ specialized metrics tailored for different service modalities:

Text-to-Text services: We adopt the Bradley-Terry (BT) model to estimate the intrinsic strength of LLMs [22] based on pairwise preference data. As utilized in Chatbot Arena [23], the win probability between model m_i and m_j is modeled as

$P(m_i \succ m_j) = \frac{e^{\xi_i}}{e^{\xi_i}+e^{\xi_j}}$, where ξ is the capability coefficient fitted via maximum likelihood estimation.

Text-to-Image services: We utilize vision-language models, e.g., MiniGPT-4, as automated evaluators to assess generated images across three dimensions: fidelity, text-image alignment, and aesthetics [24].

Image-to-Video services: We employ physics-aware metrics, using SSIM for frame fidelity and RAFT optical flow algorithms to quantify motion smoothness and consistency [25].

The raw metrics from these evaluations are normalized to the unit interval $[0, 1]$ to ensure comparability across heterogeneous tasks.

2) QoS Registry and O(1) Lookup: The computed scores are stored in a lightweight QoS registry table replicated across edge nodes. Let Q_m denote the registered quality score of model m. During the online orchestration phase, the VAMAPPO agent retrieves Q_m directly from the local registry when evaluating action probabilities.

This design reduces the evaluation complexity to $O(1)$, effectively addressing the potential bottleneck of evaluating complex generative models in real-time. Consequently, the orchestration latency depends solely on the inference table lookup, independent of the complexity of the underlying GenAI models.

E. SLA-Based QoS Utility Formulation

To quantify the trade-off between inference quality and response timeliness, we model the system objective using utility theory.

1) Quality Utility: The inference quality of a microservice stage is determined by the capability of the selected GenAI model. Leveraging the offline QoS registry, the quality utility for stage i, denoted as $U_{qual}^{(i)}$, is defined as the normalized score of the deployed model:

$$U_{qual}^{(i)} = \sum_{s\in\mathcal{S}} \sum_{m\in\mathcal{M}} x_{i,s} \cdot y_{i,m} \cdot Q_m, \tag{5}$$

where $Q_m \in [0, 1]$ is the registered capability score.

2) Latency Utility and End-to-End SLA: While the ultimate user experience depends on the end-to-end latency $T_{e2e} = \sum_{i\in\mathcal{I}} T_i$, optimizing a sparse terminal reward in multi-agent reinforcement learning often leads to slow convergence. To facilitate stable distributed learning, we decompose the global SLA requirement into stage-wise soft deadlines $T_{max}^{(i)}$. The latency utility for stage i is defined as a sigmoid function:

$$U_{lat}^{(i)} = \frac{1}{1 + \exp\left(\kappa \cdot (T_i - T_{max}^{(i)})\right)}, \tag{6}$$

where T_i is the stage response latency derived in Eq. (4), and $\kappa > 0$ controls the penalty steepness. Maximizing the cumulative stage-wise utility aligns with satisfying the global SLA, while providing dense reward signals for the agents.

3) Composite Service Utility: The comprehensive QoS utility for stage i is formulated as the weighted sum of quality and latency utilities:

$$\mathcal{U}_{total}^{(i)} = \omega \cdot U_{qual}^{(i)} + (1-\omega) \cdot U_{lat}^{(i)}, \tag{7}$$

where $\omega \in [0, 1]$ is a preference parameter. This utility function serves as the reward r_i in the VAMAPPO formulation.

F. Problem Formulation

Based on the defined models, we formulate the QoS-aware GenAIaaS orchestration problem. Our objective is to find the optimal service composition policy $\{\mathbf{X}, \mathbf{Y}\}$ that maximizes the cumulative service utility across the entire microservice chain, subject to resource and deployment constraints.

The optimization problem is formulated as follows:

$$\mathcal{P}1: \max_{\mathbf{X},\mathbf{Y}} \sum_{i\in\mathcal{I}} \mathbb{E}_\delta \left[\mathcal{U}_{total}^{(i)}\right] \tag{8a}$$

$$\text{s.t.} \quad \sum_{s\in\mathcal{S}} x_{i,s} = 1, \quad \sum_{m\in\mathcal{M}} y_{i,m} = 1, \quad \forall i \in \mathcal{I}, \tag{8b}$$

$$x_{i,s} \cdot y_{i,m} \le \mathbb{I}(m \in \mathcal{M}_s), \quad \forall i, s, m, \tag{8c}$$

$$x_{i-1,s} \cdot x_{i,s'} \le \mathbb{I}((s, s') \in \mathcal{E}), \quad \forall i, s, s', \tag{8d}$$

$$\sum_{i\in\mathcal{I}_{active}} x_{i,s} \cdot \mathcal{R}_{req}(m) \le \mathcal{C}_{cap}(s), \quad \forall s \in \mathcal{S}, \tag{8e}$$

$$x_{i,s}, y_{i,m} \in \{0, 1\}, \quad \forall i, s, m. \tag{8f}$$

1) Constraint Analysis: The constraints defined above enforce the physical and logical feasibility of the orchestration plan. Specifically, (8b) enforces assignment atomicity by mapping each microservice stage to exactly one edge node and one specific GenAI model instance. (8c) dictates service availability, restricting task dispatch to nodes where the required service m is explicitly active in the local capability set $\mathcal{M}_s$. (8d) guarantees topological connectivity, requiring valid communication links in graph $\mathcal{G}$ whenever consecutive stages are assigned to different nodes. (8e) imposes resource capacity limits, ensuring that the aggregate demand of active microservices on node s does not exceed its physical threshold $\mathcal{C}_{cap}(s)$. (8f) restricts the decision variables to binary values, characterizing the discrete nature of the selection problem.

2) Complexity and Solution Strategy: Problem $\mathcal{P}1$ constitutes a stochastic combinatorial optimization problem that generalizes the NP-hard multidimensional knapsack problem (MKP). The computational intractability stems from two primary challenges. First, the system suffers from combinatorial explosion, where the search space expands exponentially with the number of service nodes and model variants ($O(S^I \cdot M^I)$). Second, (8a) incorporates inherent stochasticity through the expectation $\mathbb{E}_\delta$ over the runtime uncertainty factor $\delta_{s,t}$ (as defined in Eq. (3)). Consequently, traditional static solvers, e.g., mixed-integer linear programming (MILP), are ill-suited to adapt to these dynamic environmental spikes. To address these limitations, we transform $\mathcal{P}1$ into a decentralized Networked MDP and propose the VAMAPPO algorithm to learn a value-adaptive policy robust to system stochasticity.

IV. Value-Adaptive Multi-Agent Proximal Policy Optimization

To effectively tackle the stochastic combinatorial nature of Problem $\mathcal{P}1$, we propose the VAMAPPO framework. Unlike traditional heuristic scheduling, VAMAPPO transforms the

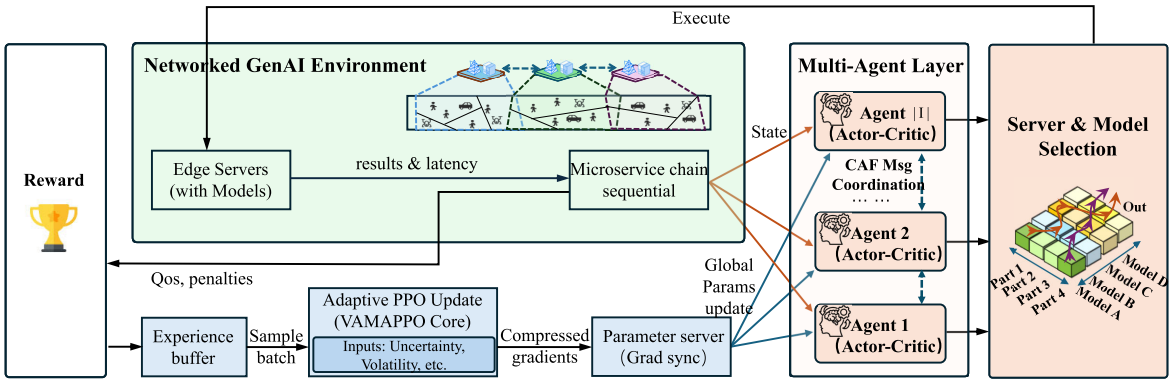


Fig. 2. The schematic workflow of the proposed VAMAPPO framework.

static orchestration problem into a dynamic networked MDP. Central to this framework is the value-adaptive mechanism, which fundamentally shifts the learning paradigm from static updates to uncertainty-aware modulation. This mechanism empowers agents to autonomously gauge the "value" of current experiences—dynamically amplifying learning intensity for high-confidence, high-advantage transitions while dampening updates during stochastic latency spikes to prevent overfitting to noise. By integrating this adaptive capability, VAMAPPO enables decentralized agents to learn robust orchestration policies that maximize long-term service utility. As illustrated in Fig. 2, the VAMAPPO framework operates as a hierarchical closed-loop system.

A. MDP Formulation for Service Orchestration

We decompose the global microservice chain orchestration into decentralized sequential decision-making tasks. Each microservice stage $i \in \mathcal{I}$ is managed by an orchestration agent a_i. The MDP tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$ is defined as follows:

1) State Space Mapping: The state s_t^a for agent a (responsible for stage i) at time step t encapsulates local service requirements and global environmental awareness. To support value-adaptive learning, we design a composite state vector:

$$s_t^a = \{\mathbf{o}_{req}^a, \mathbf{o}_{net}^t, \mathbf{o}_{prev}^t\}, \tag{9}$$

where $\mathbf{o}_{req}^a = \{D_{in}^{(i)}, T_{max}^{(i)}, \mathcal{M}_{req}\}$ characterizes the current microservice request, including input tensor size $D_{in}^{(i)}$, SLA soft deadline $T_{max}^{(i)}$, and the set of compatible GenAI models $\mathcal{M}_{req}$. $\mathbf{o}_{net}^t = \{\mathcal{M}_s, Q_m, \hat{R}_{s,s'}\}_{\forall s,m}$ represents the system observability, containing the service capability Set $\mathcal{M}_s$ of each edge node, the quality scores Q_m retrieved from the offline registry, and the estimated link bandwidth $\hat{R}_{s,s'}$. Finally, $\mathbf{o}_{prev}^t = \{x_{i-1,s'}\}$ denotes the decision of the predecessor agent, which is critical for calculating communication latency $T_{comm}^{(i)}$. This design ensures agents have sufficient information to trade off between communication overhead $\mathbf{o}_{prev}^t$ and inference quality Q_m.

2) Action Space Transformation: The continuous decision variables in $\mathcal{P}1$ are mapped to a parameterized policy output. The action space $\mathcal{A}^a$ is defined as a joint distribution over edge nodes and service instances:

$$\mathcal{A}^a = \left\{ (\mathbf{a}_{node}, \mathbf{a}_{serv}) \mid \sum_{s \in \mathcal{S}} a_{node}^s = 1, \sum_{m \in \mathcal{M}} a_{serv}^m = 1 \right\}, \tag{10}$$

where $\mathbf{a}_{node}$ and $\mathbf{a}_{serv}$ correspond to the probability distributions for variables $\mathbf{X}$ and $\mathbf{Y}$, respectively.

To guarantee valid orchestration, we enforce action masking: probabilities are set to zero if service m is not deployed on node s or if node s is unreachable from the predecessor.

3) SLA-Aware Reward Function: The reward function drives the agent to optimize the composite utility defined in Eq. (7). Unlike prior works using heuristic metrics, we align the reward strictly with the SLA Utility:

$$r_t^a = \mathcal{U}_{total}^{(i)} - \sum_k \eta_k \cdot P_k, \tag{11}$$

where $\mathcal{U}_{total}^{(i)}$ is the weighted sum of quality and sigmoid-based latency utility. P_k represents penalty terms for soft constraint violations (e.g., extreme load imbalance), and η_k is the penalty coefficient. Crucially, the latency utility $U_{lat}^{(i)}$ in $\mathcal{U}_{total}^{(i)}$ provides a dense, differentiable signal that naturally penalizes SLA violations, e.g., $T_i > T_{max}^{(i)}$, without requiring hard clipping, facilitating stable gradient descent.

B. Coordinated Advantage Function for Microservice Chains

In decentralized orchestration for interdependent microservice chains, standard independent advantage estimation fails to capture complex inter-agent dependencies. A latency spike in an upstream service, e.g., prompt engineering, can cascade, causing SLA violations in downstream services, e.g., image generation, even if downstream agents act optimally locally. This leads to the severe credit assignment problem, where agents cannot distinguish their actual contribution to the global service utility.

To address this, we propose the coordinated advantage function (CAF). Unlike traditional approaches, CAF incorporates a "social influence" term derived from counterfactual reasoning. Formally, the coordinated advantage for agent a at time t is defined as:

$$A_{coord}^a(s_t, \mathbf{a}_t) = (1-\lambda) A_{local}^a(s_t, a_t^a) + \lambda \sum_{b \in \mathcal{N}_{out}(a)} \omega_{a \to b} A_{inf}^{a \to b}(s_t, \mathbf{a}_t), \tag{12}$$

Algorithm 1 Calculation of Coordinated Advantage Function (CAF)

Require: Current state s_t, joint action $\mathbf{a}_t$, agent index a
Require: Local rewards $r^a_{t:T}$, Centralized Critic Q^b for successor b
Hyperparameters: Coordination factor λ, GAE parameter λ_{GAE}, Discount γ
Compute TD errors: $\delta^a_k = r^a_k + \gamma V_{\phi^a}(s^a_{k+1}) - V_{\phi^a}(s^a_k)$ for $k \in [t, T]$
Estimate local advantage: $A^a_{local}(s_t, a^a_t) = \sum_{l=0}^{T-t-1} (\gamma\lambda_{GAE})^l \delta^a_{t+l}$
Identify successor agents $\mathcal{N}_{out}(a)$ in the microservice chain
for each successor $b \in \mathcal{N}_{out}(a)$ **do**
    Get structural weight $\omega_{a\to b}$ based on chain dependency
    Counterfactual Reasoning:
    Sample baseline action: $\bar{a}^a_t \sim \mathbb{E}_{\bar{a}\sim\pi^a}[\bar{a}|s^a_t]$
    Estimate influence using Centralized Critic Q^b:
    $A^{a\to b}_{inf} = Q^b(s_t, a^a_t, \mathbf{a}^{-a}_t) - \mathbb{E}_{\bar{a}}[Q^b(s_t, \bar{a}, \mathbf{a}^{-a}_t)]$
end for
$A^a_{coord} = (1-\lambda)A^a_{local} + \lambda \sum_{b\in\mathcal{N}_{out}(a)} \omega_{a\to b} A^{a\to b}_{inf}$
Return: Coordinated advantage estimate A^a_{coord}

where $A^a_{local}(s_t, a^a_t)$ is the standard generalized advantage estimation (GAE) based on agent a's own SLA utility reward r^a_t, $\mathcal{N}_{out}(a)$ denotes the set of immediate successor agents in the microservice chain $\mathcal{I}$, $\lambda \in [0, 1]$ is the coordination factor controlling the balance between self-interest and social welfare. Crucially, $\omega_{a\to b}$ denotes the dynamic structural weight defined as $\omega_{a\to b} = \tanh\left(\frac{D^{(a)}_{out}}{\hat{R}_{s_a,s_b}\tau}\right)$, where $D^{(a)}_{out}$ represents the transmission data size, $\hat{R}_{s_a,s_b}$ is the real-time link bandwidth, and τ is a time normalization constant. The pseudo-code is provided in Alg. 1.

C. Distributed Implementation and Stability Mechanisms

To deploy VAMAPPO in bandwidth-constrained edge networks, we address two practical challenges: communication overhead during gradient synchronization and training instability caused by latency stochasticity.

1) Counterfactual Influence Estimation: The innovation lies in the influence term $A^{a\to b}_{inf}$, which measures the marginal contribution of agent a's action a^a_t to agent b's cumulative reward. We quantify this using a counterfactual baseline:

$$A^{a\to b}_{inf}(s_t, \mathbf{a}_t) = \underbrace{Q^b(s_t, a^a_t, \mathbf{a}^{-a}_t)}_{\text{Actual Value}} - \underbrace{\mathbb{E}_{\bar{a}\sim\pi^a}\left[Q^b(s_t, \bar{a}, \mathbf{a}^{-a}_t)\right]}_{\text{Counterfactual Baseline}}, \tag{13}$$

where Q^b is the centralized critic evaluating agent b's long-term utility. The first term represents the estimated value for agent b given agent a's actual action, while the second term estimates the expected value if agent a had acted according to its average policy, keeping other agents' actions $\mathbf{a}^{-a}_t$ fixed.

In our implementation, the counterfactual baseline $\mathbb{E}_{\bar{a}\sim\pi^a}[Q^b(s_t, \bar{a}, \mathbf{a}^{-a}_t)]$ is estimated using a single-sample approximation by directly feeding the mean action of the current policy $\bar{a}^a_t = \mathbb{E}[\pi^a(\cdot|s^a_t)]$ into the centralized critic. This serves as a powerful variance reduction technique by isolating the marginal contribution from the global joint action space. Notably, we do not maintain separate, standalone critic networks for Q^b. Instead, VAMAPPO reuses the successor agent b's existing critic network V_{ϕ^b}, approximating the value via the TD target $Q^b \approx r^b_t + \gamma V_{\phi^b}(s^b_{t+1})$. Consequently, CAF introduces zero additional network parameters or memory footprint, adding only one critic forward pass per successor.

By comparing the actual outcome against this counterfactual baseline, CAF effectively isolates agent a's specific impact on downstream performance, enabling precise credit assignment even under stochastic latency spikes. The computed A^a_{coord} is subsequently used to calculate the policy enhancement weight w^a_t in the value-adaptive update phase.

D. Value-Adaptive Policy Network Design

To capture the stochasticity of GenAI inference latency and enable robust decision-making, we redesign the traditional actor-critic architecture using a Bayesian [26] perspective.

1) Uncertainty-Aware Network Architecture: We employ a specialized neural architecture capable of processing the heterogeneous state information defined in the previous section.

Input embedding: The composite state s^a_t is split into service requirements $\mathbf{o}_{req}$ and network status $\mathbf{o}_{net}$. These are projected into high-dimensional embeddings.

Attention-based encoder: To capture the complex dependencies between the microservice chain and the topology graph, we utilize a multi-head attention mechanism. It computes the relevance between the current service request and the capabilities of distributed edge nodes, effectively filtering out irrelevant nodes, e.g., those lacking the required GenAI model.

Stochastic parameterization: To accommodate the stringent computational and memory constraints of edge environments, we implement approximate variational inference via Monte Carlo (MC) Dropout, which avoids the parameter doubling required by explicit Bayesian formulations. We impose an isotropic Gaussian prior $p(\theta) = \mathcal{N}(0, \lambda^{-1}I)$ on the network weights. Specifically, stochastic parameterization is restricted to the uncertainty estimation branch corresponding to the Critic's value output. We apply a hierarchical decaying dropout structure ($p = 0.4 \to 0.3 \to 0.2 \to 0.1$) to capture multi-scale epistemic uncertainty, while shared feature extractors utilize a standard dropout ($p = 0.1$) for regularization. The reparameterization trick is inherently realized as $W_l = M_l \cdot \text{diag}(\mathbf{z}_l)$, where the dropout mask $z_{l,j} \sim \text{Bernoulli}(1-p_l)$ acts as the injected noise ϵ, and M_l is the deterministic weight representing the variational mean μ. This formulation guarantees that the loss remains differentiable with respect to M_l.

2) Adaptive Update Mechanism: Standard PPO updates often fail in dynamic edge environments because they assume a stationary transition capability. To address this, we propose a value-adaptive update rule that dynamically calibrates the learning step based on three defined real-time metrics:

Decision Uncertainty (U^a_t): Quantified via $T = 3$ stochastic MC forward passes. It is computed as the sum of the mean

softplus outputs and half of the predictive variance, i.e.,

$$U_t^a = \frac{1}{T}\sum_{\tau=1}^{T} f(s_t^a;\hat{\theta}_\tau) + \frac{1}{2}\mathrm{Var}\left[\{f(s_t^a;\hat{\theta}_\tau)\}_{\tau=1}^{T}\right] + \eta_{\min}(k), \tag{14}$$

where $\eta_{\min}(k)$ is a progressively annealed lower bound to prevent premature convergence.

Environmental Volatility (Δ_t^a): Computed as the Euclidean distance $||s_t^a - s_{t-1}^a||_2$ in the state embedding space (the output of the shared feature layer), effectively capturing semantic shifts in network load rather than raw observation noise.

Inter-Agent Consistency (C_t^a): Defined as the cosine similarity between the compressed gradients of adjacent agents, i.e., $\cos(\tilde{g}^a, \tilde{g}^b)$. In our asynchronous parameter-server setup, C_t^a calculation piggybacks directly on synchronized gradients, introducing zero extra communication steps.

These metrics are fused into a dynamic adaptive factor $\alpha_t^a \in (0, 1]$:

$$\alpha_t^a = \sigma(\beta_1 U_t^a + \beta_2 \Delta_t^a + \beta_3 C_t^a), \tag{15}$$

where $\sigma(\cdot)$ is the sigmoid function and β are learnable coefficients.

Furthermore, to prioritize high-value experiences in the replay buffer, we introduce a enhancement weight w_t^a. It scales the gradient magnitude based on the coordinated advantage:

$$w_t^a = \exp\left(\frac{\gamma \cdot |A_{coord}^a|}{\tau}\right), \tag{16}$$

where τ is a temperature parameter controlling the sensitivity to advantage values.

3) Modified PPO Loss: Integrating the adaptive mechanisms, the final loss function for agent a becomes:

$$\mathcal{L}^a(\theta) = \mathbb{E}_t\Big[\alpha_t^a w_t^a \mathcal{L}_t^{CLIP}(\theta) - c_1\mathcal{L}_t^{VF}(\phi) + c_2 S[\pi_\theta] - \lambda_{KL}\mathcal{D}_{KL}\Big], \tag{17}$$

where $\mathcal{L}^{CLIP}$ is the standard PPO clipped objective, $\mathcal{L}^{VF}$ is the value function error, S is entropy, and $\mathcal{D}_{KL}$ is the KL divergence term required for variational inference regularization. The adaptive factor α_t^a acts as a gate, dampening updates during periods of high uncertainty or volatility to prevent catastrophic forgetting, while w_t^a accelerates learning from high-quality transitions.

4) On-Policy Rollout Buffer and Priority Sampling: We emphasize that VAMAPPO adheres to PPO's on-policy paradigm. The experience buffer, denoted as $\mathcal{D}^{(\pi_k)}$, functions exclusively as a temporary on-policy rollout buffer. During each iteration, agents collect a trajectory batch using the current policy $\pi_{\theta_{old}}$. Upon the completion of the gradient updates, $\mathcal{D}^{(\pi_k)}$ is entirely flushed; no historical data persists. Furthermore, the proposed priority sampling mechanism operates strictly within this current on-policy minibatch. Transitions are sampled non-uniformly with a probability proportional to the policy enhancement weight $w_t^a = \exp(\gamma \cdot |A_{coord}^a|/\tau)$. Because the behavioral policy and the target policy are identical during sample collection, no off-policy bias is introduced, rendering offline corrections unnecessary.

Algorithm 2 Value-Adaptive Bayesian Policy Update

Require: Batch $\mathcal{B}$ sampled from $\mathcal{D}$ via priority sampling
Require: Current VI parameters (μ^a, σ^a) and critic parameters ϕ^a
Require: Coordinated Advantage estimates A_{coord}^a (from Alg. 1)
Hyperparameters: Learning rates $\eta_\mu, \eta_\sigma, \eta_\phi$; Coefficients $\beta, \gamma, \tau, c_1, c_2, \lambda_{KL}$
for epoch $e = 1, \ldots, E_{ppo}$ **do**
  for minibatch $\mathcal{M} \subset \mathcal{B}$ **do**
    Calculate Decision Uncertainty U_t^a from variance σ^a
    Measure Environmental Volatility $\Delta_t^a = ||s_t - s_{t-1}||_2$
    Assess Inter-Agent Consistency C_t^a via gradient similarity
    Adaptive Factor: $\alpha_t^a \leftarrow \sigma(\beta_1 U_t^a + \beta_2 \Delta_t^a + \beta_3 C_t^a)$
    Enhancement Weight: $w_t^a \leftarrow \exp(\gamma \cdot |A_{coord}^a|/\tau)$
    Compute components: PPO objective $\mathcal{L}^{CLIP}$, Value loss $\mathcal{L}^{VF}$, Entropy S, KL divergence $\mathcal{D}_{KL}$
    Formulate composite Bayesian loss:
      $\mathcal{L}_{total}^a = \mathbb{E}_{\mathcal{M}}[\alpha_t^a w_t^a \mathcal{L}^{CLIP} - c_1\mathcal{L}^{VF} + c_2 S - \lambda_{KL}\mathcal{D}_{KL}]$
    Update parameters via gradient descent on $\mathcal{L}_{total}^a$:
      $(\mu^a, \sigma^a) \leftarrow \text{Optimizer}(\nabla_{\mu,\sigma}\mathcal{L}_{total}^a)$
      $\phi^a \leftarrow \text{Optimizer}(\nabla_\phi \mathcal{L}_{total}^a)$
  end for
end for
Return: Updated parameters $(\mu^a, \sigma^a), \phi^a$

5) Communication-Efficient Synchronization: We adopt an asynchronous parameter server [27] architecture. To minimize the data transfer of high-dimensional gradients, we implement Top-k gradient compression. Instead of transmitting the full gradient vector g, each agent only uploads the top $k\%$ elements with the largest absolute magnitudes. The sparse gradient $\tilde{g}$ is defined as:

$$\tilde{g} = \text{TopK}(g, \rho^* \cdot |g|), \tag{18}$$

where $\rho^* \in (0, 1]$ is the compression ratio. The parameter server aggregates these sparse updates and broadcasts the global gradient, ensuring synchronization efficiency without significant accuracy loss.

6) Adaptive PPO Clipping: In standard PPO, a fixed clipping parameter ϵ restricts the policy update step. However, in our stochastic GenAIaaS environment, fixed clipping can be either too conservative or too aggressive. We propose an adaptive clipping mechanism that modulates ϵ_t based on the variance of historical updates:

$$\epsilon_t = \epsilon_{min} + (\epsilon_{max} - \epsilon_{min}) \cdot \exp\left(-\eta \cdot \mathrm{Var}(\Delta\theta_{t-w:t})\right), \tag{19}$$

where $\mathrm{Var}(\Delta\theta_{t-w:t})$ measures the stability of policy updates over a sliding window w. When the policy oscillates (high variance), ϵ_t tightens to ensure safety; when stable, it relaxes to accelerate learning. The pseudo-code is provided in Alg. 2.

E. Online Execution Workflow and Deployment Assumptions

To bridge the gap between the theoretical MARL formulation and physical system deployment, VAMAPPO operates on a per-stage" logical binding rather than a per-node" basis. Each microservice stage $i \in \mathcal{I}$ is managed by a corresponding orchestration agent a_i. Physically, these agents are instantiated as lightweight daemons co-located with the edge network controllers. This design ensures that scalability is strictly bounded by the microservice chain depth $|\mathcal{I}|$ rather than the physical edge node count $|\mathcal{S}|$.

The online execution forms a closed-loop control cycle: (i) the agent observes the state and estimates Bayesian epistemic uncertainty; (ii) the CAF computes the coordinated advantage; (iii) the policy selects the target edge node and model; and (iv) the action is enforced on the microservice instance, followed by reward collection. During the online inference phase, agents execute a single forward pass per microservice request stage.

F. Convergence and Complexity Analysis

In this subsection, we provide a analysis of the convergence properties of VAMAPPO and evaluate its computational complexity in the context of edge service orchestration.

1) Theoretical Convergence: Standard PPO is known to maximize a surrogate objective function subject to a trust region constraint. We analyze how the introduction of the value-adaptive mechanism affects this property.

Theorem 1 (Convergence to Stationary Point): Under standard assumptions (Lipshitz continuity of the policy and bounded variance of advantage estimates) [28], the VAMAPPO algorithm with adaptive modulation factor α_t^a and coordinated advantage A_{coord}^a converges to a stationary point of the expected service utility function $J(\pi)$, provided that the adaptive factor is strictly bounded ($0 < \epsilon_\alpha \leq \alpha_t^a \leq 1$) and the coordination weight λ ensures bounded advantage variance.

Proof: We rely on the monotonic improvement property of trust region methods. The standard PPO lower bound on policy improvement depends on the expected advantage and the KL divergence penalty. In VAMAPPO, the parameter update direction is scaled by the factor $\alpha_t^a \cdot w_t^a$. Since α_t^a is generated by a sigmoid function, it acts as a dynamic learning rate scheduler constrained within $(0, 1)$. When epistemic uncertainty is high, α_t^a decreases, implicitly tightening the trust region to prevent destructive updates that would violate the KL constraint. Furthermore, the coordinated advantage A_{coord}^a modifies the variance of the gradient estimator without introducing bias to the local expectation, provided the counterfactual baseline is unbiased. Consequently, the sequence of value-adaptive updates generates a monotonically non-decreasing sequence of lower bounds on the objective function $J(\pi)$, ensuring asymptotic convergence to a stationary point. □

2) Computational Complexity: We analyze the runtime efficiency of VAMAPPO compared to traditional optimization methods. The per-step decision complexity is primarily dominated by the actor network execution. Specifically, the multi-head attention mechanism over S edge nodes scales quadratically $O(S^2)$ (or linearly $O(S)$ with sparse implementations). Crucially, thanks to the proposed offline QoS registry, fetching quality scores Q_m is reduced to a constant-time $O(1)$ lookup operation, successfully avoiding the prohibitive cost of runtime model evaluation. Regarding inter-agent coordination, calculating the CAF requires querying the centralized critic for immediate successors. Since the microservice chain depth I is typically limited, this coordination overhead remains linear $O(|\mathcal{N}_{out}|)$. In stark contrast to global combinatorial solvers, e.g., MILP, which suffer from exponential complexity $O(S^I \cdot M^I)$, VAMAPPO maintains polynomial time complexity, thereby ensuring scalability for real-time orchestration in dense edge networks.

V. Experimental Analysis

A. Experimental Setup

1) Trace-Driven Evaluation Environment: We establish a high-fidelity trace-driven evaluation platform based on Gymnasium [29]. By integrating real-world network topologies with industrial workload traces, this platform captures the stochastic resource demands of GenAI microservices, with state spaces mapped directly to the MDP formulation. The experiments are executed on a computing cluster equipped with four NVIDIA RTX 3090 GPUs. The implementation is managed via Anaconda, utilizing PyTorch with CUDA acceleration for deep reinforcement learning. Standard libraries, including NumPy, Pandas, and Matplotlib, are employed for numerical computation, data processing, and visualization.

2) Dataset and Workload Description: To comprehensively evaluate VAMAPPO, we construct a realistic evaluation benchmark by integrating real-world network topologies with industrial workload traces.

On GPU-accelerated edge nodes, single-stage inference for these architectures typically ranges from 50 to 300 ms. For communication delay, intermediate tensors (0.1–10 MB) transmitted over edge links up to 1 Gbps yield delays in the tens of milliseconds.

Physical network topology: We utilize the *Spatio-Temporal Edge Computing Dataset* [30] to model the underlying infrastructure. We selected four representative topologies with distinct scales and geographical characteristics:

- *S1 (Switzerland):* A regional edge network comprising 25 heterogeneous service nodes (avg. capacity 34.6) with sparse connectivity (degree 3.2), representing a small-scale deployment.
- *S2 (UK):* A national-scale network with 51 nodes, characterized by medium density and moderate resource heterogeneity.
- *S3 (Germany):* A large-scale complex environment with 101 nodes, serving as the primary testbed for scalability and stability analysis.
- *S4 (Milan):* A dense urban edge network (31 nodes), reserved exclusively for zero-shot generalization testing to evaluate cross-environment adaptability.

Trace-driven workload generation: To capture the stochastic nature of GenAIaaS traffic, we move beyond synthetic poisson distributions. We derive service request patterns from the Alibaba cluster trace [31], preserving real-world characteristics

TABLE III
HYPERPARAMETER SETTINGS FOR VAMAPPO AGENTS

Parameter	Value
Optimizer	Adam
Learning Rate (η)	3×10^{-4} (Linear Decay)
Discount Factor (γ)	0.99
Batch Size	1024
Mini-batch Size	64
PPO Clip Range (ϵ)	0.2
GAE Parameter (λ_{GAE})	0.95
Entropy Coefficient	0.01
Value Loss Coefficient (c_1)	0.5

such as bursty arrival spikes and diurnal load variations. Specifically, we extracted trace segments to form training sets (1,000 requests) and testing sets (200 requests) for each topology. Each request is instantiated as a 5-stage microservice chain, e.g., Prompt $\rightarrow$ LLM $\rightarrow$ Refine $\rightarrow$ ImgGen $\rightarrow$ Output. Crucially, the inference latency for each stage is not random but determined by our offline profiling of real models, superimposed with a stochastic noise factor $\delta_{s,t}$ to simulate runtime resource contention.

3) Experimental Configuration and Hyperparameters: The training process is executed over 1,000,000 time steps. We adopt a strict Train-Test Split strategy: the first 80% of the trace duration is used for training, while the remaining 20% is reserved for evaluation. The VAMAPPO agents are implemented with the hyperparameters detailed in Tab. III. These values were rigorously determined via a coarse-to-fine grid search strategy to ensure optimal performance. Specifically, the learning rate was empirically set to 3×10^{-4} after evaluating the range $[10^{-5}, 10^{-3}]$; this value was found to offer the best trade-off between convergence speed and training stability in our stochastic edge environment. Similarly, the PPO clip range and entropy coefficients were tuned to prevent premature convergence while maintaining sufficient exploration.

4) Baselines: To validate the superiority of the proposed framework, we compare VAMAPPO against four representative algorithms, covering state-of-the-art DRL, optimization-based, and heuristic approaches:

- *MAPPO:* [14] A state-of-the-art multi-agent PPO algorithm originally designed for communication-dependent task orchestration. We adapt it to the GenAIaaS context as the primary DRL baseline. It shares the centralized training, decentralized execution (CTDE) architecture with our method but relies on standard advantage estimation, lacking the value-adaptive mechanism to handle inference latency stochasticity.
- *METAPPO:* [15] An advanced optimization-based method originally proposed for edge AR rendering. It features a task migration mechanism to dynamically release resources. We include it as a representative dynamic scheduling baseline to evaluate whether traditional resource-aware migration strategies can cope with the bursty computation patterns of GenAI microservices.
- *Greedy:* [32] A heuristic strategy that adopts a myopic optimization approach. For each microservice stage, it selects the edge node with the highest immediate resource availability, e.g., maximum available VRAM, to minimize local latency, disregarding global chain dependencies and future load spikes.
- *Random:* [33] A lower-bound baseline that assigns microservices to available edge nodes uniformly at random. This serves to benchmark the fundamental complexity of the orchestration problem and quantify the gain of intelligent decision-making.

5) Ablation Study Design: To rigorously quantify the contribution of each algorithmic component to the orchestration robustness, we designed an ablation study with five configurations. All variants were trained and evaluated in the standard S2 environment:

- *Full VAMAPPO:* The complete framework integrating the Bayesian policy network, multi-Head attention encoder, and CAF. It utilizes both the adaptive modulation factor α_t^a and enhancement weight w_t^a.
- *w/o Attention:* Replaces the multi-head attention mechanism with standard fully connected layers for state encoding. This variant tests the importance of capturing topological dependencies between service nodes and microservice requirements.
- *w/o Enhancement:* Disables the policy enhancement weight mechanism (setting $w_t^a = 1$). This variant evaluates the impact of prioritizing high-advantage transitions derived from the CAF.
- *w/o Uncertainty:* Replaces the Bayesian policy network with a standard deterministic actor network. This eliminates the variational inference mechanism and the adaptive factor (α_t^a), testing the system's resilience to stochastic latency spikes without uncertainty quantification.
- *Minimal Model:* A vanilla Multi-Agent PPO implementation that simultaneously removes the attention encoder, uncertainty-aware adaptation, and coordination mechanisms, serving as a baseline for fundamental complexity.

Each configuration was evaluated over 100 independent test episodes using identical Alibaba trace segments to ensure statistical rigor and comparability.

6) Performance Metrics: To rigorously evaluate the GenAIaaS orchestration performance, we employ the following key metrics. Note: In our experimental figures, we utilize standard RL terminology to denote these service-oriented metrics:

- *Average service utility:* The cumulative composite utility $\mathcal{U}_{total}$ obtained by the agent per episode. Since this utility constitutes the immediate optimization objective. Higher utility indicates a better trade-off between inference fidelity and SLA compliance.
- *SLA satisfaction rate:* The percentage of microservice requests successfully completed within the soft deadline T_{max}. This metric reflects the reliability of the orchestration framework in meeting strict QoS guarantees.

Fig. 3. Convergence of average service utility across three network topologies (S1, S2, S3) over 1M training steps.

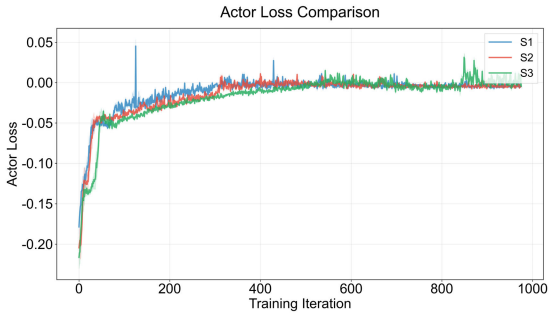


Fig. 4. Comparison of actor network loss across topologies.

Fig. 5. Comparison of critic network loss across topologies.

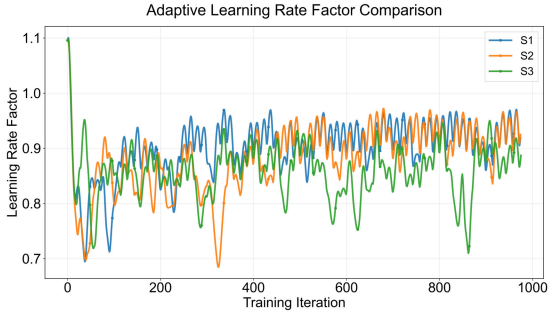


Fig. 6. Evolution of adaptive learning rate schedules during the training process.

- *Response latency:* The end-to-end delay encompassing both computation and communication across the microservice chain.
- *Communication overhead:* The total time consumed by intermediate tensor transmission between distributed edge nodes. This captures the efficiency of the topological routing strategy.
- *Node resource usage diversity:* Quantified by the coefficient of variation of edge node usage frequency. This metric evaluates the load balancing characteristics: a lower value implies a more balanced distribution, while a higher value suggests resource concentration.
- *Loss & learning rate:* We monitor the actor and critic losses to verify policy convergence. Additionally, the adaptive learning rate trajectory reflects how the VAMAPPO agent modulates step sizes (α_t^a) in response to environmental volatility and epistemic uncertainty.

B. Basic Performance Evaluation

1) Training Phase Convergence Analysis: We evaluated the training dynamics of the VAMAPPO framework across three representative network topologies. Fig. 3 illustrates the convergence of the average service utility over 1 million timesteps. The results exhibit characteristic reinforcement learning behavior, featuring rapid initial improvement followed by gradual stabilization. Specifically, in the S1 topology (25 nodes), the framework achieved a 163% performance gain, rising from an initial utility of 2.32 to a final stable value of 6.10. The S2 topology (51 nodes) attained the highest convergence value of 6.34, representing a 3.9% improvement over S1, which highlights the orchestration advantages inherent in its network structure. Although the S3 topology (101 nodes) converged to a slightly lower utility of 5.76, it demonstrated minimal fluctuation during the late training phase, indicating superior stability in learning complex orchestration policies under stochastic trace-driven workloads.

Fig. 4 presents a comparative analysis of the Actor network loss, revealing effective policy optimization. The S2 network achieved the most efficient policy learning, with the Actor loss converging to 0.022, while S1 and S3 stabilized at 0.023 and 0.024, respectively. Similarly, Fig. 5 illustrates the Critic loss comparison, where all three topologies converged to approximately 0.020, indicating consistent value function approximation. Notably, the S3 network exhibited the fastest convergence rate and smoothest descent curves, reflecting the scalability of our VAMAPPO algorithm in large-scale heterogeneous resource environments.

Fig. 6 depicts the adaptive learning rate adjustment process. The results indicate that all networks employed similar decay strategies: maintaining higher learning rates during early training to accelerate exploration, then gradually reducing them to ensure stability. The S3 network demonstrated the smoothest adjustment curve, avoiding drastic fluctuations. This validates that the adaptive factor α_t^a effectively modulates learning steps in response to the environmental volatility introduced by the trace data, preventing policy collapse in complex environments. While S1 and S2 exhibited similar patterns, S2 showed more responsive adjustments during the intermediate phase, adapting to its specific topological complexity.

2) Inference Phase Performance Evaluation: Fig. 7 illustrates the distribution of inference service utilities across 200 test episodes using trace-driven workloads. The results clearly demonstrate that the S3 network achieved the highest average utility of 32.64 with the smallest standard deviation, exhibiting a highly concentrated normal distribution that reflects exceptional QoS consistency. The S2 network attained an average

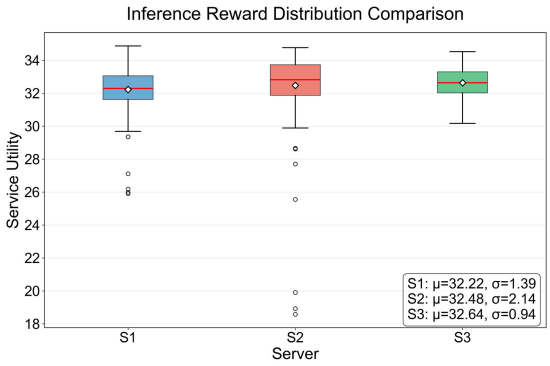


Fig. 7. Statistical distribution of inference service utilities across 200 test episodes for three network topologies.

TABLE IV
PERFORMANCE COMPARISON UNDER REAL-WORLD TOPOLOGIES

Topology (Region)	**SLA Sat.** (%)	**Utility** ($\mathcal{U}_{total}$)	**Stab.** (Std.)	**Comm.** (s)	**Usage** (Nodes/%)
S1 (Switzerland)	98.0	32.22	1.39	1.41	9 (36%)
S2 (UK)	97.5	32.48	1.42	1.31	18 (35%)
S3 (Germany)	100.0	32.64	1.21	1.35	31 (31%)

utility of 32.48, marginally higher than S1's 32.22. Significantly, the S3 network exhibited the smallest interquartile range and fewest outliers, further confirming its robustness in handling runtime latency spikes.

Fig. 8 presents the communication efficiency characteristics by analyzing the relationship between communication overhead and achieved utilities. Fig. 8a shows that the S1 network exhibits a strong negative correlation between overhead and utility. Fig. 8b demonstrates that the S2 network shows relative advantages in communication efficiency with a 7.1% improvement, reflecting structural benefits in distributed data transmission. Fig. 8c illustrates the characteristics of the S3 network, positioning it between S1 and S2.

3) Comprehensive Performance Summary: Tab. IV summarizes the comparison results. Regarding reliability, the S3 network (101 nodes) achieved a perfect 100% SLA satisfaction rate, significantly outperforming the S1 network (98%) and S2 network (97.5%). For average service utility, performance improved with network scale: S3 led with 32.64, followed by S2 at 32.48 and S1 at 32.22. Standard deviation analysis reveals that S3 achieved the minimum value of 1.21, indicating optimal performance consistency.

Communication overhead analysis reveals the impact of network topology on distributed communication efficiency. The S2 network demonstrated the best efficiency with the lowest overhead of 1.31 seconds. Notably, considering the S3 network's 100% SLA satisfaction, its communication overhead represents significant cost-effectiveness advantages. Regarding resource utilization, as network scale increased, the relative proportion of active nodes decreased (S3: 31%), indicating efficient resource consolidation.

C. Value-Adaptive Mechanism Ablation Experiments

To systematically evaluate the effectiveness of each core component in the proposed value-adaptive mechanism, we designed a rigorous ablation study. We compared the Full Model against variants removing Attention, Enhancement weights, Uncertainty quantification, and the Minimal Model.

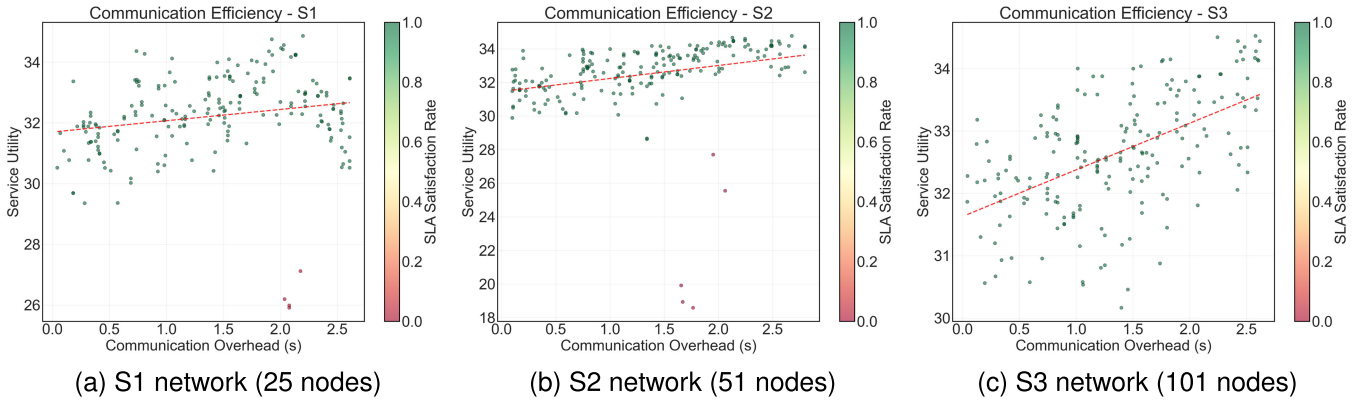
1) Average Utility Analysis: Fig. 9 illustrates the comprehensive impact of each algorithmic component on system performance. Specifically, regarding service utility, Fig. 9a shows that significant performance disparities exist across configurations. The Full Model achieved optimal performance with an average utility of 31.63 ± 5.63, significantly outperforming all simplified configurations. Particularly noteworthy is the dramatic performance degradation in the w/o Attention configuration and the Minimal Model, where average utility plummeted to ≈ 2.30 ($\approx 92.7\%$ decrease). This strongly indicates the critical role of the attention mechanism in encoding topological dependencies.

In contrast, the w/o Enhancement and w/o Uncertainty configurations maintained relatively high performance levels. However, the drop in the w/o Uncertainty model (30.71 ± 6.40) highlights that lack of uncertainty quantification impairs the agent's ability to handle the stochastic latency spikes inherent in trace-driven workloads, leading to suboptimal decisions.

2) SLA Satisfaction Rate Analysis: As depicted in Fig. 9b, the SLA satisfaction rate follows a similar trend to the utility metric. The Full Model achieved an excellent rate of 94.6%, demonstrating the effectiveness of the value-adaptive mechanism in ensuring QoS. Consistent with utility metrics, removing the attention mechanism had catastrophic effects. The w/o Uncertainty configuration (92.6%) showed a degradation, confirming that modeling epistemic uncertainty contributes to robustness against SLA violations in dynamic environments.

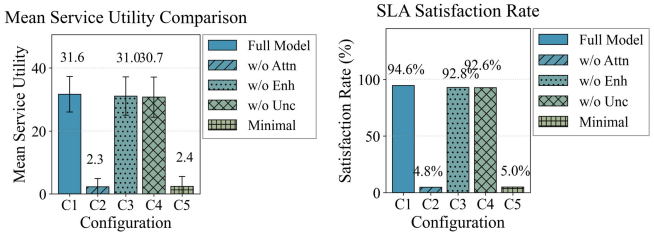
Based on the ablation results, we can explicitly map the unique composition of VAMAPPO to the specific physical edge challenges it resolves: (1) The attention mechanism actively filters out unreachable nodes or those lacking required models. As demonstrated by the 92.7% utility drop in the w/o Attention variant, without it, agents are overwhelmed by noise from irrelevant candidates in highly heterogeneous environments like the S1 topology. (2) When sudden GPU contention occurs, the epistemic uncertainty U_t^a surges, triggering the adaptive factor to dampen the policy update. This prevents the agent from overfitting to a transient noise spike, avoiding the severe oscillation observed in the w/o Uncertainty variant. (3) CAF specifically resolves cascading credit assignment failures. If an upstream stage suffers a transient delay, the downstream stage will likely miss its SLA. While standard MARL would falsely penalize the downstream agent, CAF utilizes the counterfactual influence term $A_{inf}^{a \to b}$ to isolate the upstream agent's structural impact, effectively protecting the downstream agent's advantage from unjust penalization.

As a concrete example from the S3 trace-driven evaluation, around timestep $t \approx 450$ a transient compute-contention event consistent with thermal throttling inflated the runtime noise factor $\delta_{s,t}$ in Eq. (3). The decision uncertainty U_t^a spiked accordingly, driving down the adaptive factor α_t^a and dampening the policy update so that the agent reassigned the affected stage on the next decision rather than overcommitting to the congested node. This trace-level event empirically grounds the



(a) S1 network (25 nodes) (b) S2 network (51 nodes) (c) S3 network (101 nodes)

Fig. 8. Communication efficiency analysis showing the relationship between communication overhead and achieved service utilities during the inference phase.



(a) Average service utilities. (b) SLA satisfaction rates.

Fig. 9. Ablation study results comparing different model configurations in terms of (a) average service utility and (b) SLA satisfaction rate.

theoretical role of $\delta_{s,t}$ and the uncertainty-modulated update introduced in Section IV.

D. Comparative Experiments

To validate the effectiveness of the VAMAPPO algorithm, we conducted comprehensive performance comparisons in the unseen S4 environment against four baselines: MAPPO, METAPPO, Greedy, and Random, as illustrated in Fig. 10.

1) SLA Satisfaction Rate Analysis: Fig. 10a presents the SLA satisfaction rates. Our VAMAPPO algorithm achieved the highest rate at 97.0%, representing a 3.2% improvement over MAPPO (94.0%) and an 8.4% improvement over METAPPO (89.5%). Notably, VAMAPPO achieved improvements of over 10% compared to Greedy and Random strategies. These results demonstrate VAMAPPO's superior adaptability in orchestrating complex microservice chains under strict QoS constraints.

2) Response Latency Performance: Regarding end-to-end response latency (Fig. 10b), VAMAPPO achieved the shortest average time of 0.257 seconds. This represents a 6.5% reduction compared to MAPPO (0.275s) and a 4.8% reduction compared to METAPPO (0.270s). Against traditional methods, the gains are substantial: 18.9% reduction vs. Greedy and 24.2% vs. Random. This advantage is primarily attributed to VAMAPPO's uncertainty-aware scheduling, which proactively avoids edge nodes exhibiting high latency variance.

3) Communication Overhead Optimization: As depicted in Fig. 10c, VAMAPPO achieved the lowest communication overhead of 1.287 seconds, representing a 6.6% reduction compared to MAPPO and a 4.3% reduction compared to METAPPO. More significantly, compared to Greedy and Random strategies, overhead was reduced by 16.1% and 21.2%, respectively. This demonstrates that VAMAPPO optimizes network resource utilization while maintaining high performance.

4) Resource Usage Diversity Analysis: In terms of node resource usage diversity (Fig. 10d), METAPPO exhibited the highest diversity index (1.440), reflecting its advantage in load balancing. VAMAPPO achieved a diversity index of 1.071, which, while lower than METAPPO, is significantly higher than MAPPO (0.801), Greedy (0.771), and Random (0.709). This indicates that VAMAPPO strikes a balance: it maintains relatively balanced resource utilization while prioritizing the primary objective of maximizing service utility.

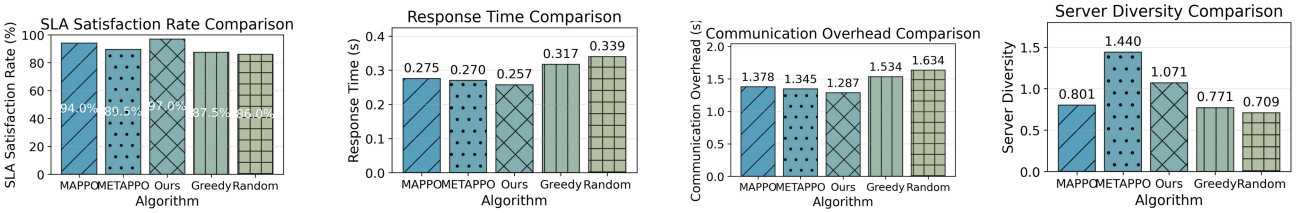
E. Zero-Shot Generalization Testing

To evaluate the generalization capability of the VAMAPPO value-adaptive model in heterogeneous network environments, we designed cross-server generalization experiments. Models trained on S1 (M1), S2 (M2), and S3 (M3) were directly evaluated on the unseen S4 topology without fine-tuning.

1) Key Performance Metrics Comparison: To rigorously validate the robustness of our zero-shot generalization and ensure statistical significance, we evaluated the models across 5 independent random seeds $\{42, 123, 256, 512, 1024\}$, totaling 1,000 evaluation episodes per configuration. Tab. V summarizes the zero-shot generalization performance of models trained on different topologies (M1, M2, M3) when deployed directly to the unseen S4 environment. The results now report the mean performance alongside the standard deviation, supplemented by 95% confidence intervals computed via Bootstrap resampling ($B = 10{,}000$). Furthermore, paired Welch's t-tests confirmed that the performance improvements of VAMAPPO over the baselines remain statistically significant at the $p < 0.05$ level even in unseen environments.

In terms of average utility, Model M3 achieved the best performance with 32.29 points, representing slight but consistent improvements over M1 and M2. This indicates that exposure to complex, large-scale network patterns during training enhances the agent's feature extraction and decision optimization capabilities.

Regarding reliability, Model M3 similarly excelled with an SLA satisfaction rate of 98.00%, significantly higher than M1



(a) SLA satisfaction rates. (b) Average response latency. (c) Communication overhead. (d) Node resource usage diversity.

Fig. 10. Comprehensive performance comparison in the S4 environment.

TABLE V
ZERO-SHOT GENERALIZATION PERFORMANCE ON UNSEEN TOPOLOGY (MEAN ± STD. DEV

Metric (Trained on)	**Model M1** (S1)	**Model M2** (S2)	**Model M3** (S3)
Avg. Service Utility ($\mathcal{U}_{total}$)	32.20 ± 1.35	32.06 ± 1.42	32.29 ± 1.21
SLA Satisfaction Rate (%)	97.00 ± 1.50	95.50 ± 2.10	98.00 ± 1.00
Response Latency (s)	0.020 ± 0.005	0.030 ± 0.008	0.030 ± 0.006

Note: Results are averaged across 5 random seeds (1,000 evaluation episodes per configuration).

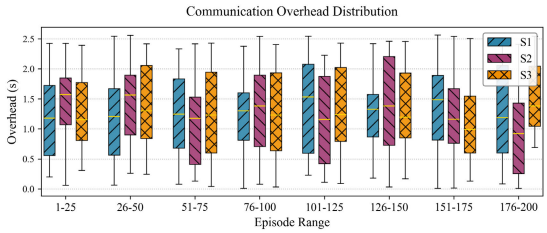


Fig. 11. Communication overhead evolution across test episodes.

(97.00%) and M2 (95.50%). Notably, Model M2's relatively lower success rate may be attributed to the specific topological mismatch between its training environment (UK) and the test environment (Milan). Interestingly, regarding response latency, Model M1 demonstrated the fastest raw speed (0.02s), potentially because it learned more aggressive, short-path strategies in the dense S1 network, albeit at the cost of slightly lower overall utility.

2) Communication Overhead Evolution Analysis: As shown in Fig. 11, the three models exhibited different evolutionary trends in communication overhead. During the initial phase (episodes 1-50), Model M1 showed high variability ($\approx 1.2s$), potentially related to its exploratory learning process. Model M2 stabilized around 1.0s in the middle phase. Model M3 exhibited the lowest communication overhead in the later phase (episodes 151-200), with the median dropping below 0.8s, demonstrating its ability to progressively optimize communication strategies during extended operation.

VI. CONCLUSION

In this paper, we addressed the critical challenge of orchestrating GenAIaaS in resource-constrained edge networks, where service demands manifest as interdependent microservice chains with highly stochastic latency. Moving beyond static scheduling approaches, we proposed VAMAPPO, a novel value-adaptive multi-agent reinforcement learning framework. By integrating variational inference, the framework explicitly quantifies epistemic uncertainty, allowing agents to dynamically modulate learning steps in response to runtime environmental spikes. Furthermore, the proposed CAF leverages counterfactual reasoning to disentangle complex inter-service dependencies, effectively solving the credit assignment problem in sequential workflows. Extensive experiments using trace-driven workloads demonstrate that our approach significantly outperforms state-of-the-art baselines, reducing response latency by 24.2% while maintaining strict SLA compliance. This work lays the foundation for uncertainty-aware cognitive edge computing. Future research will extend this architecture to privacy-preserving federated learning and energy-efficient green computing scenarios.

REFERENCES

[1] D. H. Hagos, R. Battle, and D. B. Rawat, "Recent advances in generative AI and large language models: Current status, challenges, and perspectives," *IEEE Trans. Artif. Intell.*, vol. 5, no. 12, pp. 5873–5893, Dec. 2024.

[2] D. Li, W. Lu, Y. Shen, K. Song, X. Tan, and Y. Zhuang, "HuggingGPT: Solving AI tasks with ChatGPT and its friends in hugging face," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 38154–38180.

[3] C. Du, Y. Li, Z. Qiu, and C. Xu, "Stable diffusion is unstable," in *Proc. Adv. Neural Inf. Process. Syst. 36*, 2023, pp. 58648–58669.

[4] L. Manduchi et al., "On the challenges and opportunities in generative AI," *ACM Comput. Surv.*, vol. 57, no. 1, pp. 14:1–14:42, 2024.

[5] X. He, Q. Peng, M. Shao, and H. Liu, "HPDM: A hierarchical popularity-aware debiased modeling approach for personalized news recommender," in *Proc. 33rd Int. Joint Conf. Artif. Intell.*, Aug. 2024, pp. 2883–2891.

[6] K. Li, X. Chen, T. Song, H. Zhang, W. Zhang, and Q. Shan, "GPTDrawer: Enhancing visual synthesis through ChatGPT," in *Proc. 5th Int. Conf. Neural Netw., Inf. Commun. Eng. (NNICE)*, Jan. 2025, pp. 368–372.

[7] H. Cai et al., "Enable deep learning on mobile devices: Methods, systems, and applications," *ACM Trans. Design Autom. Electron. Syst.*, vol. 27, no. 3, pp. 1–50, May 2022.

[8] Y. Yao et al., "MiniCPM-V: A GPT-4 V level MLLM on your phone," 2024, *arXiv:2408.01800*.

[9] M. Zhang, X. Shen, J. Cao, Z. Cui, and S. Jiang, "EdgeShard: Efficient LLM inference via collaborative edge computing," *IEEE Internet Things J.*, vol. 11, no. 13, pp. 23528–23541, Jun. 2024.
[10] B. Ali, M. Golec, S. S. Gill, F. Cuadrado, and S. Uhlig, "EdgeAIBus: AI-driven joint container management and model selection framework for heterogeneous edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 36, no. 1, pp. 137–154, Jan. 2025.
[11] Y. Guan, S. Zou, H. Peng, W. Ni, Y. Sun, and H. Gao, "Cooperative UAV trajectory design for disaster area emergency communications: A multiagent PPO method," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 8848–8859, Mar. 2024.
[12] Q. Wang, S. Zou, Y. Sun, M. Liwang, X. Wang, and W. Ni, "Toward intelligent and adaptive task scheduling for 6G: An intent-driven framework," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 5, pp. 1975–1988, Oct. 2024.
[13] Z. Yao, Z. Tang, W. Yang, and W. Jia, "Enhancing LLM QoS through cloud-edge collaboration: A diffusion-based multi-agent reinforcement learning approach," *IEEE Trans. Serv. Comput.*, vol. 17, no. 5, pp. 3226–3240, May 2024.
[14] Q. Han, X. Wang, and W. Shen, "Communication-dependent computing resource management for concurrent task orchestration in IoT systems," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 6271–6287, Jun. 2024.
[15] W. Qian and R. W. L. Coutinho, "A reinforcement learning-based orchestrator for edge computing resource allocation in mobile augmented reality systems," in *Proc. IEEE 34th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2023, pp. 1–6.
[16] B. Lai et al., "Resource-efficient generative mobile edge networks in 6G era: Fundamentals, framework and case study," *IEEE Wireless Commun.*, vol. 31, no. 4, pp. 66–74, Aug. 2024.
[17] M. Navardi, S. Gao, M. Walczak, F. Camacho, and T. Mohsenin, "Metareasoning for edge-cloud collaborative LLM planning for efficient autonomous navigation," *ACM Trans. Embed. Comput. Syst.*, vol. 23, no. 5, pp. 72:1–72:24, 2024.
[18] J. Wen et al., "Diffusion-model-based incentive mechanism with prospect theory for edge AIGC services in 6G IoT," *IEEE Internet Things J.*, vol. 11, no. 21, pp. 34187–34201, Nov. 2024.
[19] Y. Wen, S. Zou, Y. Sun, and H. Gao, "Adaptive network function chain orchestration strategy for 6G inclusive intelligent services," in *Proc. Workshop Mobility Evolving Internet Archit.*, Nov. 2024, pp. 7–12.
[20] J. Zhang, S. Zou, M. Liwang, Y. Sun, W. Ni, and X. Wang, "H^3DRA: Achieving resilient task scheduling for zero-interruption vehicular edge computing in sixth-generation networks," *IEEE Trans. Veh. Technol.*, vol. 73, no. 9, pp. 13809–13824, Sep. 2024.
[21] D. A. Menascé and S. Bardhan, "TDQN: Trace-driven analytic queuing network modeling of computer systems," *J. Syst. Softw.*, vol. 147, pp. 162–171, Jan. 2019.
[22] B. Hao, M. Shao, Z. Wo, Y. Chu, Y. Liu, and R. Wang, "LLMTM: Benchmarking and optimizing LLMs for temporal motif analysis in dynamic graphs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 40, 2026, pp. 14802–14810.
[23] W.-L. Chiang et al., "Chatbot arena: An open platform for evaluating LLMs by human preference," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2024, pp. 8359–8388.
[24] Y. Chen, L. Liu, and C. Ding, "X-IQE: Explainable image quality evaluation for text-to-image generation with visual large language models," 2023, *arXiv:2305.10843*.
[25] L. Gong et al., "AtomoVideo: High fidelity image-to-video generation," 2024, *arXiv:2403.01800*.
[26] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1613–1622.
[27] M. Li et al., "Scaling distributed machine learning with the parameter server," in *Proc. 11th USENIX Symp. Oper. Syst. Design Implement.*, 2014, pp. 583–598.
[28] B. Liu, Q. Cai, Z. Yang, and Z. Wang, "Neural trust region/proximal policy optimization attains globally optimal policy," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.
[29] M. Towers et al., "Gymnasium: A standard interface for reinforcement learning environments," 2024, *arXiv:2407.17032*.
[30] B. Xiang, J. Elias, F. Martignon, and E. Di Nitto, "A dataset for mobile edge computing network topologies," *Data Brief*, vol. 39, Dec. 2021, Art. no. 107557.
[31] S. Luo et al., "Characterizing microservice dependency and performance: Alibaba trace analysis," in *Proc. ACM Symp. Cloud Comput.*, Nov. 2021, pp. 412–426.
[32] D. Jungnickel, "The greedy algorithm," in *Graphs, Networks and Algorithms*. Cham, Switzerland: Springer, 1999, pp. 129–153.
[33] R. Motwani and P. Raghavan, "Randomized algorithms," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 33–37, 1996.

Yan Gao is currently pursuing the Ph.D. degree with the School of New Media and Communication, Tianjin University, China. His research focuses on distributed LLM serving, resource allocation, and reinforcement learning. He is particularly interested in the design and optimization of intelligent orchestration mechanisms for large-scale AI services in edge-cloud computing environments.

Shaoyuan Huang received the B.S. degree from Tianjin University, Tianjin, China, in 2020, where he is currently pursuing the Ph.D. degree with the College of Intelligence and Computing. He has published technical papers in CIKM, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MOBILE COMPUTING, INFOCOM, and SIG KDD. His current research interests include spatial-temporal prediction, workload forecasting, and system performance modeling.

Minglai Shao received the Ph.D. degree from the School of Computer Science and Engineering, Beihang University, China. He is currently an Elite Associate Professor and a Distinguished Research Fellow with the School of New Media and Communication, Tianjin University, China. His research interests include deep learning, machine learning, trustworthy AI, anomaly detection, graph mining, intelligent communication, recommender systems, large language models, data mining, and data science for real-world applications.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from Harbin Institute of Technology, China, in 2009 and 2011, respectively, and the Ph.D. degree (Hons.)in computer science from Freie Universität Berlin, Germany, in 2015.

He is currently a Full Professor with the Center for Applied Mathematics, Tianjin University, China. His research interests include mobile cloud computing, edge computing, the Internet of Things, deep learning, complex networks, and DNA storage.

Yansha Deng (Senior Member, IEEE) is currently a Full Professor of intelligent communication systems with the Department of Engineering, King's College London, London, U.K. She has secured more than (£4.5 million in research funding as a Principal Investigator and £14.5 million as a Co-Investigator. She has received the ERC Starting Grant and the EPSRC NIA Award. She has published more than 140 journal articles and more than 70 IEEE/ACM conference papers. Her research interests include molecular communication and machine learning for 5G/6G wireless networks. She was a recipient of the Best Paper Awards from ICC 2016 and GLOBECOM 2017 as the first author, and the IEEE Communications Society Best Young Researcher Award for the Europe, Middle East, and Africa Region, in 2021.