Workflow-Aware Expert Routing for Distributed LLM Serving Over the Edge-Cloud Continuum

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

Abstract*—Deploying Large Language Models (LLMs) over the edge-cloud continuum faces severe stability challenges due to the conflict between stochastic network topology and complex workflow dependencies. Existing schedulers, relying either on computationally prohibitive Graph Neural Networks (GNNs) or topology-agnostic heuristics, fail to reconcile this tension. To bridge these gaps, we propose STEM, a service-level and topology-aware orchestration framework that formulates distributed LLM serving as a workflow-aware routing problem over a monitored service overlay, in which heterogeneous service instances act as specialized experts. At the core of STEM lies the STAR-PPO algorithm, utilizing a lightweight graph-free perception mechanism. By leveraging Squeeze-and-Excitation attention, it extracts critical bottleneck features from raw telemetry with linear complexity, bypassing the scalability limits of message-passing paradigms. To further achieve Pareto-efficient trade-offs, we develop a Dynamic Weight Adaptation (DWA) mechanism that autonomously recalibrates optimization preferences based on entropy-regularized metric drift. Extensive experiments on real-world datasets spanning 2,000 nodes demonstrate that our framework significantly outperforms state-of-the-art baselines. Specifically, STAR-PPO reduces network transmission costs by 96.8% and improves comprehensive inference efficiency by 24.4%, while sustaining robust zero-shot generalization across regions, with average latency within 1.09× of a target-domain-retrained reference under a strict cross-region protocol. Code and data are available at *https://github.com/gymorsiback/STARPPO

***Index Terms*—Large language models, edge-cloud continuum, distributed inference, topology-aware scheduling, reinforcement learning.**

I. Introduction

With the proliferation of generative artificial intelligence [1], Large Language Model (LLM) serving is shifting from single-endpoint inference to workflow-level orchestration over a geographically distributed and heterogeneous service continuum [2]. To navigate the trade-off between QoS and operational expenditure, LLM service platforms expose a multi-tier portfolio of model offerings. This portfolio spans from lightweight models [3] optimized for latency and cost-efficiency to heavyweight models delivering advanced reasoning capabilities, alongside domain-specific variants. Even within a single model family, the service pool hosts diverse configurations [4] varying in context window, quantization precision, and resource quotas. Consequently, these geographically dispersed and computationally heterogeneous inference endpoints constitute a global resource pool [5], [6], rather than isolated monolithic services.

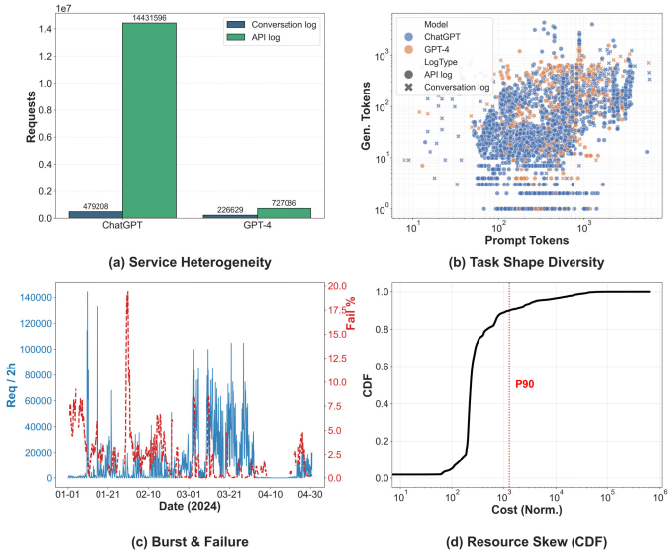


Fig. 1. Empirical analysis of large-scale LLM service traces from Microsoft Azure over 121 days.

As illustrated in Fig. 1, long-term empirical data from the Microsoft Azure [7] LLM platform underscores the high dimensionality and stochasticity of real-world service demands. Our analysis of a trace covering 5.29 million requests over 121 consecutive days reveals four critical characteristics: (1) Fig. 1a shows that although lightweight APIs dominate request volume, the long-tail of heavyweight-model calls consumes a substantial share of compute capacity, amplifying queue backlogs and tail-latency pressure under contention [8]; (2) Fig. 1b indicates that requests range from single-turn queries to extensive multi-turn interactions,

Received 22 June 2026; accepted 30 July 2026. Date of publication 12 August 2026; date of current version 20 August 2026. This work was supported in part by the Tianjin Natural Science Foundation Project (Grant No. 25JCYBJC01540). The associate editor coordinating the review of this article and approving it for publication was J. Wang. *(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).

Yonghui Ye is with the School of Cyber Security and Computer, Hebei University, Baoding 071002, China (e-mail: hui519975@gmail.com).

Jie Fu is with the School of New Media and Communication, Tianjin University, Tianjin 300072, China, and also with China National Software and Service Company Ltd., Beijing 100000, China (e-mail: m644784535@gmail.com).

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.2026.3722837

creating substantial variance in memory and compute requirements [9]; (3) Fig. 1c demonstrates that arrival patterns exhibit strong burstiness superimposed on diurnal cycles, frequently triggering node overload [7]; and (4) Fig. 1d reveals that a small portion of complex inference tasks consume the majority of computational resources, leading to severe resource asymmetry [10].

Based on these observations, we formalize large-scale LLM serving as a complex online decision-making environment constrained by multi-model [11], multi-specification, and multi-region factors [12]. Efficient service orchestration in this context faces four stringent challenges:

- **Model Heterogeneity:** [3], [13] Given the vast disparity in capabilities and costs across the service pool, static rules fail to achieve granular trade-offs within the "capability-cost-latency" triangle, often resulting in overprovisioning or performance degradation under load.
- **Spatio-Temporal Volatility:** [14] The low average RPS coupled with high burstiness renders distribution strategies based on stationary assumptions ineffective. Single-policy approaches struggle to adapt to rapid fluctuations in network congestion and computational availability.
- **Topology-Aware Constraint:** [15] LLM services operate across an *edge-cloud continuum*. Service instances are highly heterogeneous in computing scale, bandwidth, and link latency. Routing decisions must therefore jointly optimize for network topology, instance status, and data residency boundaries.
- **Workflow Coupling:** [16] Modern applications increasingly map tasks to Directed Acyclic Graph (DAG) workflows, e.g., intent recognition–difficulty assessment–inference. Routing decisions for individual sub-tasks cascade to affect subsequent steps, making local optimization insufficient for end-to-end performance.

To address these challenges, we propose STEM, a service-level and topology-aware orchestration framework for the edge-cloud continuum. STEM treats each heterogeneous service instance as a specialized expert and routes the dependent steps of a workflow across these instances. In contrast to conventional DAG workflow scheduling or service routing, which typically map independent tasks onto compute slots, STEM jointly handles three coupled decisions that are otherwise addressed in isolation: capability-aware expert selection, topology-aware placement under network and reliability constraints, and preference-conditioned control of the latency–cost–risk trade-off. The expert view is thus used only as a unifying service-level scheduling abstraction for these coupled decisions, distinct from intra-model token routing [17]; the novelty we claim lies in this coupling and its online preference adaptation rather than in the terminology itself.

Central to STEM is the STAR-PPO algorithm, which utilizes a lightweight graph-free perception mechanism. By leveraging Squeeze-and-Excitation attention, it extracts bottleneck features from raw telemetry with linear complexity, bypassing the scalability limits of message-passing paradigms. To navigate the Pareto frontier, we devise a Dynamic Weight Adaptation (DWA) mechanism that autonomously recalibrates optimization preferences based on entropy-regularized metric drift. Consequently, the agent captures the nonlinear correlations between global resources and local topology at a millisecond-level online decision scale, achieving Pareto-efficient optimization—balancing conflicting objectives, i.e., latency, cost, and risk, such that no single metric can be improved without compromising another—even within resource-constrained environments.

The main contributions are summarized as follows:

- We propose STEM, an orchestration framework for the edge-cloud continuum that abstracts distributed LLM service instances as heterogeneous experts and models requests as DAG workflows. Unlike per-task service routing, the formulation couples capability-aware expert selection with topology-aware placement and end-to-end workflow dependencies within a single routing problem.
- Distinct from computationally expensive GNNs, we develop STAR-PPO, a spatio-temporal augmented reinforcement learning approach. It employs a lightweight topology-aware state representation that captures critical graph telemetry with linear complexity. This design enables the agent to perceive and adapt to non-stationary network dynamics efficiently, ensuring scalability in large-scale infrastructure.
- To navigate the trade-off between QoS and operational costs, a DWA mechanism is devised. By leveraging an entropy-regularized feedback loop driven by epoch-level metric drift, DWA autonomously adjusts optimization preferences, ensuring Pareto-efficient decision-making that prevents the policy from collapsing into single-objective dominance.
- We conduct extensive evaluations over large-scale real-world traces and topology datasets, demonstrating that STEM achieves pareto-superior latency–cost trade-offs and improves robustness under bursty, non-stationary workloads compared with state-of-the-art baselines.

II. Related Work

With the ecosystem of LLMs evolving toward multimodality, heterogeneity, and globalization, minimizing operational costs while satisfying diverse service level agreements (SLAs) has emerged as a research imperative. Recent studies address this challenge through semantic-aware routing, distributed resource scheduling, and topology-aware joint optimization.

A. Semantic Routing and Sequence Modeling

To navigate the "capability-cost" trade-off within heterogeneous model pools, the research paradigm has shifted from predefined sequential cascading to semantic-aware routing. Early strategies, like FrugalGPT [18], employ such fixed cascading mechanisms, i.e., sequentially querying models until a confidence threshold is met, to reduce costs but suffer from latency accumulation. To address this, predictive frameworks, such as RouteLLM [19], leverage lightweight classifiers for one-shot difficulty prediction. More recently, the field has seen the emergence of Transformer-based solvers that treat routing as a sequence generation problem. For

instance, Equity-Transformer [20] introduced a self-attention mechanism to solve NP-hard Min-Max routing problems, enforcing fairness constraints globally via sequential generation. Similarly, Zhao et al. [21] employed autoregressive Transformers to encode query semantics for complex intent routing. However, while Transformer-based architectures like Equity-Transformer excel in capturing long-range dependencies and semantic contexts [22], their quadratic complexity ($\mathcal{O}(N^2)$) as a function of the number of service nodes makes them computationally prohibitive for low-latency online scheduling in large-scale clusters. Moreover, they typically focused on logical routing fairness while ignoring the physical network dynamics.

B. Distributed Scheduling in the Edge-Cloud Continuum

To address the computational bottlenecks of edge environments, research has been categorized into two dimensions: architectural execution paradigms, such as split computing, and algorithmic scheduling strategies, including resource-aware reinforcement learning. For the split computing, approaches, like FrankenSplit [23], optimized layer-wise partitioning via latent feature compression. In the domain of dynamic scheduling, Deep Reinforcement Learning (DRL) has become a dominant approach. Classic algorithms like A3C [24] have been widely applied to multi-cloud workflow scheduling, utilizing asynchronous gradient updates to stabilize training. To address resource heterogeneity in space-air-ground integrated networks (SAGINs), PF-PPO [25] proposed a proportional fairness-aware framework, which integrates resource capacity constraints directly into the reward function to prevent node starvation. Nevertheless, a critical limitation of standard DRL methods, e.g., A3C [24], is their sample inefficiency and instability in non-stationary environments. Even advanced variants like PF-PPO, while effective for resource fairness [29], rely on explicit resource state modeling and lack a specialized mechanism to capture the complex topological dependencies, inherent in edge-cloud workflows.

C. Topology-Awareness and Spatio-Temporal Optimization

At the macroscopic governance level, capturing the spatio-temporal correlations of network topology is critical. GNNs represent the state-of-the-art in this domain. GA-PPO [27] combines graph convolution with reinforcement learning to capture the structural information of dynamic workflows, facilitating topology-aware scheduling. Parallel to GNNs, spatio-temporal transformers, such as STARK [28], have demonstrated powerful capabilities in tracking temporal dynamics in visual tasks, a paradigm that can be adapted for grid-based network state encoding. To improve the exploration efficiency of RL agents in such high-dimensional spaces, PPO-CN [26] introduced colored noise into action sampling to enhance policy robustness against local optima. Existing topology-aware methods either incur prohibitive inference overhead or lack principled mechanisms for multi-objective optimization, motivating the need for a lightweight, graph-free framework for real-time LLM orchestration.

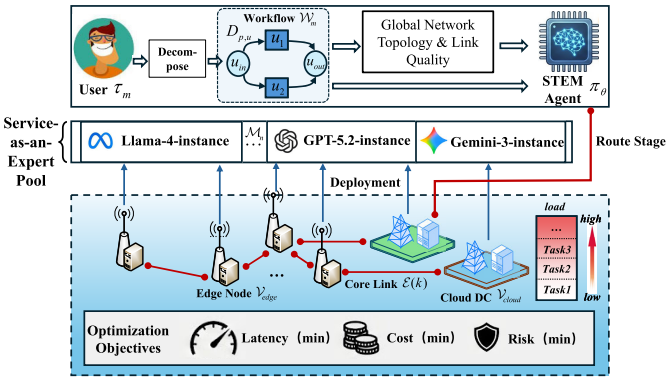


Fig. 2. System architecture of the proposed workflow-aware service-overlay routing framework over the edge–cloud continuum.

As qualitatively contrasted in Tab. I, prior arts generally either abstract away physical topology or incur prohibitive overhead via heavy graph encoders. STEM uniquely bridges this gap by reconciling semantic dependencies with network constraints. Unlike GNN-based approaches with $\mathcal{O}(N^2)$ complexity, our graph-free SE-Attention mechanism captures topological features with linear complexity, enabling real-time, Pareto-efficient orchestration in dynamic edge environments.

III. System Model and Problem Formulation

This section presents the service-level system model used for workflow-aware LLM serving over an edge-cloud continuum. In contrast to a physical-layer-only view, we model the infrastructure as a monitored service overlay: each vertex is a deployed inference service endpoint or an aggregated service pool, and each directed edge is a logical communication path whose effective performance is obtained from telemetry or from a deployment-specific link model. This abstraction keeps the formulation applicable to wireless edge deployments, wired WANs, cross-region cloud interconnects, and virtual service overlays, while still exposing the network bottlenecks that determine workflow-level latency and cost. The overall architecture is shown in Fig. 2.

A. Service-Overlay Infrastructure

We consider a slotted online orchestration process indexed by a decision window k. The serving infrastructure is represented by a time-varying directed service graph

$$\mathcal{G}(k) = \big(\mathcal{V}, \mathcal{E}(k)\big), \tag{1}$$

where $\mathcal{V} = \mathcal{V}_{\text{edge}} \cup \mathcal{V}_{\text{cloud}}$ denotes the set of edge and cloud service nodes, and $\mathcal{E}(k)$ denotes the set of logical service links observable during window k. A logical link may correspond to a wireless access/backhaul segment, a wired WAN path, a cloud interconnect, or a virtual overlay tunnel. We assume that the topology and telemetry are quasi-static within one decision window and may evolve across windows due to load variation, link degradation, routing changes, or node failures.

Each node $n \in \mathcal{V}$ is characterized by

$$\mathbf{a}_n(k) = \big\langle C_n^{\text{comp}}(k), M_n^{\text{mem}}, \epsilon_n^{\text{comp}}(k), p_n^{\text{rel}}(k), \text{Region}(n) \big\rangle, \tag{2}$$

TABLE I

QUALITATIVE COMPARISON OF STEM WITH STATE-OF-THE-ART BASELINES. STEM UNIQUELY INTEGRATES SEMANTIC ROUTING WITH TOPOLOGY-AWARE SCHEDULING UNDER A LIGHTWEIGHT GRAPH-FREE PARADIGM

Method	Core Technique	Key Capabilities for Edge LLM Orchestration			
		Topology Awareness	Low Overhead (Graph-Free)	Semantic Awareness	Multi-Obj Pareto Opt.
FrugalGPT [18]	Cascade Strategy	✗	✓	✓	✗
RouteLLM [19]	Router Training	✗	✓	✓	~
Equity-Trans [20]	Equity-Transformer	✗	✗	✓	✓
FrankenSplit [23]	Var. Compression	~	✓	✗	✗
Llumnix [8]	Live Migration	~	✓	✗	✗
A3C [24]	Asyn. Actor-Critic	✗	✓	✗	✗
PF-PPO [25]	Resource-Aware PPO	~	✓	✗	~
PPO-CN [26]	Colored Noise PPO	✗	✓	✗	✗
GA-PPO [27]	PPO + GNN	✓	✗	~	~
STARK [28]	ST-Transformer	✓	✗	✗	✗
STEM (Ours)	**PPO + SE-Attn**	✓	✓	✓	✓

Legend: ✓: Fully Supported; ~: Partially Supported; ✗: Not Supported.

where $C_n^{\text{comp}}(k)$ is the effective computing throughput, M_n^{mem} is the available model memory, $\epsilon_n^{\text{comp}}(k)$ is the unit computation cost, $p_n^{\text{rel}}(k) \in [0,1]$ is the estimated reliability in the current window, and $\text{Region}(n)$ records the administrative or geographic region of the node. The reliability term is not treated as a fixed hardware constant. Instead, it denotes the conditional availability estimate

$$p_n^{\text{rel}}(k) = \Pr\{A_n(k+1) = 1 \mid \mathcal{H}_n(k)\}, \tag{3}$$

where $A_n(k)$ is the availability state and $\mathcal{H}_n(k)$ is the recent telemetry history of node n. This conditional formulation captures temporally correlated failures arising from power fluctuation, network partition, or resource exhaustion, and can be instantiated by an empirical estimator, an exponential moving average, or a Markov failure process.

We assume a pre-deployed serving setting in which each node hosts a subset of service types, denoted by $\mathcal{M}_n$. A service type may represent a model family, a domain-specialized model, a quantization level, or a capability tier required by a workflow step. The memory feasibility of the pre-deployed service pool satisfies

$$\sum_{\psi \in \mathcal{M}_n} \text{Size}(\psi) \le M_n^{\text{mem}}, \quad \forall n \in \mathcal{V}. \tag{4}$$

The orchestration problem studied in this paper does not decide model installation or eviction; it routes workflow steps to already available service instances. Dynamic model lifecycle management is left as future work.

B. Effective Communication Model

For each logical service link $(i,j) \in \mathcal{E}(k)$, let $R_{ij}(k)$ denote the effective data rate and $T_{ij}^{\text{base}}(k)$ denote the baseline path delay including propagation, routing, and protocol overheads. When an intermediate data object of size D is transferred from node i to node j, its communication latency is modeled as

$$T_{\text{comm}}(i,j,D,k) = \frac{D}{R_{ij}(k)} + T_{ij}^{\text{base}}(k) + Q_{ij}^{\text{net}}(k), \tag{5}$$

where $Q_{ij}^{\text{net}}(k)$ is the congestion-induced network queuing delay observed or estimated in window k. The tuple $\big(R_{ij}(k), T_{ij}^{\text{base}}(k), Q_{ij}^{\text{net}}(k)\big)$ can be obtained from network telemetry, trace-driven measurements, or a deployment-specific simulator.

This formulation avoids assuming that every logical link owns a dedicated physical spectrum or link capacity. Let $\mathcal{B}$ be the set of shared communication resource groups, such as a common wireless spectrum band, an edge backhaul, a regional WAN egress, or a cloud interconnect. For each group $b \in \mathcal{B}$, let $\mathcal{E}_b(k) \subseteq \mathcal{E}(k)$ be the set of logical links using that shared resource. The effective allocated bandwidths must satisfy

$$\sum_{(i,j) \in \mathcal{E}_b(k)} W_{ij}(k) \le W_b^{\max}(k), \quad \forall b \in \mathcal{B}, \tag{6}$$

with $0 \le W_{ij}(k) \le \overline{W}_{ij}(k)$. Therefore, link contention is represented at the service-overlay level through shared resource groups and the resulting effective rates $R_{ij}(k)$.

For wireless-edge deployments, (5) can be instantiated by a physical-layer capacity model. In this case, the effective rate may be computed as

$$R_{ij}^{\text{wl}}(k) = W_{ij}(k) \log_2 \left(1 + \frac{P_{ij}^{\text{tx}} |h_{ij}(k)|^2 d_{ij}^{-\alpha}}{N_0 + I_{ij}(k)}\right), \tag{7}$$

where P_{ij}^{tx} is the transmit power, $h_{ij}(k)$ is the channel coefficient, d_{ij} is the physical distance, α is the path-loss exponent, N_0 is the noise power, and $I_{ij}(k)$ is the aggregate interference. Equation (7) is thus one optional instantiation of the effective rate rather than a universal communication assumption. For wired WAN or cloud-overlay settings, $R_{ij}(k)$ and $T_{ij}^{\text{base}}(k)$ are

instead obtained from service-oriented bandwidth/RTT traces or from a queueing-based network delay model.

C. Workflow Request Model

Let $\mathcal{T}(k)$ be the set of active requests arriving in decision window k. Each request $\tau_m \in \mathcal{T}(k)$ is decomposed into a directed acyclic graph (DAG)

$$\mathcal{W}_m = (\mathcal{U}_m, \mathcal{D}_m), \tag{8}$$

where $\mathcal{U}_m$ is the set of workflow steps and $\mathcal{D}_m$ is the set of dependency edges. A workflow step $u \in \mathcal{U}_m$ is described by

$$\mathbf{q}_{m,u} = \left\langle \psi_{m,u}, F_{m,u}, D^{\text{in}}_{m,u}, T^{\text{SLA}}_{m} \right\rangle, \tag{9}$$

where $\psi_{m,u}$ is the required service type, $F_{m,u}$ is the estimated computation demand, $D^{\text{in}}_{m,u}$ is the input size, and T^{SLA}_{m} is the request-level service deadline. For each dependency edge $(p,u) \in \mathcal{D}_m$, $D_{m,p,u}$ denotes the size of the intermediate output transferred from predecessor p to successor u.

The immediate predecessor set of step u is

$$\mathcal{P}_m(u) = \{p \in \mathcal{U}_m \mid (p,u) \in \mathcal{D}_m\}. \tag{10}$$

A workflow step becomes schedulable only after all its predecessors have completed and their intermediate outputs have been delivered to the selected hosting node of u. This dependency constraint couples computation placement with inter-node communication and makes local per-step routing insufficient for minimizing end-to-end latency.

When the policy observes a ready step, the locations of its immediate predecessors are summarized by a dependency embedding. Let $n_{m,p}$ be the node selected for predecessor p. We define a data-size-weighted predecessor embedding as

$$\mathbf{e}^{\text{dep}}_{m,u} = \begin{cases} \mathbf{0}, & \mathcal{P}_m(u) = \emptyset, \\ \displaystyle\sum_{p \in \mathcal{P}_m(u)} \frac{D_{m,p,u}}{\sum_{r \in \mathcal{P}_m(u)} D_{m,r,u} + \varepsilon} \mathbf{e}_{n_{m,p}}, & \mathcal{P}_m(u) \neq \emptyset, \end{cases} \tag{11}$$

where $\mathbf{e}_{n_{m,p}}$ denotes the location or region embedding of node $n_{m,p}$ and ε is a small positive constant. The pooling in (11) is an explicit, data-size-weighted aggregation that assigns greater influence to predecessors carrying larger intermediate transfers, so that placement reflects the dominant data flows of the workflow.

D. Computation and Queueing Model

If step u of request τ_m is assigned to node n, its nominal execution time is

$$T_{\text{exec}}(m,u,n,k) = \frac{F_{m,u}}{C^{\text{comp}}_n(k)}. \tag{12}$$

The term $C^{\text{comp}}_n(k)$ is an effective throughput and can aggregate multiple GPUs, replicas, batching behavior, or service workers behind the same logical service node. The node abstraction therefore does not assume that an LLM service is a physical single-server machine.

To expose resource contention to the scheduler, we use a telemetry-based queueing approximation. Let $\lambda_n(k)$ be the observed arrival rate of workflow steps assigned to node n in window k, and let $\mu_n(k)$ be the effective service rate estimated from recent completions. The utilization is

$$\rho_n(k) = \frac{\lambda_n(k)}{\mu_n(k) + \varepsilon}. \tag{13}$$

For online scheduling, the expected waiting time is approximated by a G/G/1-inspired Kingman estimator

$$\widehat{T}_{\text{queue}}(n,\psi,k) = \frac{\rho_n(k)}{1-\rho_n(k)+\varepsilon} \cdot \frac{c^2_{a,n}(k) + c^2_{s,n,\psi}(k)}{2} \cdot \overline{S}_{n,\psi}(k), \tag{14}$$

where $c_{a,n}(k)$ and $c_{s,n,\psi}(k)$ are the coefficients of variation of inter-arrival and service times, and $\overline{S}_{n,\psi}(k)$ is the mean service time of service type ψ on node n. In implementation, these quantities can be updated by sliding-window statistics or exponential moving averages.

Equation (14) is used as a lightweight congestion feature and latency estimator, not as an exact simulator of all LLM serving internals. It is most reliable when the node-level workload is sufficiently aggregated and the utilization is not extremely sparse. Under highly bursty or intermittent arrivals the approximation can lose accuracy, and its estimation error is therefore quantified empirically under Poisson, bursty, and On–Off arrivals in Sec. V.

E. Decision Variables and Feasibility Constraints

The orchestration decision is the placement of workflow steps onto feasible service nodes. Let

$$x_{m,u,n} = \begin{cases} 1, & \text{if step } u \text{ of request } \tau_m \text{ is assigned to node } n, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Each workflow step must be assigned to exactly one node:

$$\sum_{n \in \mathcal{V}} x_{m,u,n} = 1, \quad \forall \tau_m \in \mathcal{T}(k),\ \forall u \in \mathcal{U}_m. \tag{16}$$

The assignment must satisfy the service-type availability constraint:

$$x_{m,u,n} \le \mathbb{I}\big(\psi_{m,u} \in \mathcal{M}_n\big), \quad \forall m,u,n. \tag{17}$$

To couple placement and communication, we introduce an auxiliary indicator for each workflow dependency and service link:

$$y_{m,p,u,i,j} = x_{m,p,i} x_{m,u,j}, \quad \forall (p,u) \in \mathcal{D}_m,\ \forall (i,j) \in \mathcal{E}(k). \tag{18}$$

In an equivalent mixed-integer linear form, (18) can be enforced by

$$y_{m,p,u,i,j} \le x_{m,p,i}, \tag{19}$$

$$y_{m,p,u,i,j} \le x_{m,u,j}, \tag{20}$$

$$y_{m,p,u,i,j} \ge x_{m,p,i} + x_{m,u,j} - 1, \tag{21}$$

$$y_{m,p,u,i,j} \in \{0,1\}. \tag{22}$$

The shared resource constraint in (6) is imposed on the effective link states used to compute $R_{ij}(k)$. When bandwidth allocation is optimized jointly, $W_{ij}(k)$ enters as an explicit decision variable; otherwise it is treated as an exogenous telemetry state supplied by the underlying network controller.

F. Latency, Cost, Risk, and QoS Metrics

1) End-to-End Workflow Latency: Let $C_{m,u}$ be the completion time of step u in workflow $\mathcal{W}_m$ under placement $\mathbf{X}$. For a source step with no predecessor, its ready time is the arrival time A_m of request τ_m. For other steps, the ready time is determined by the latest predecessor completion plus the required intermediate transfer:

$$\begin{aligned} S_{m,u} &= A_m, \qquad \text{if } \mathcal{P}_m(u) = \emptyset; \\ S_{m,u} &= \max_{p \in \mathcal{P}_m(u)} \Bigg[C_{m,p} \\ &\quad + \sum_{(i,j) \in \mathcal{E}(k)} y_{m,p,u,i,j}\, T_{\text{comm}}(i,j,D_{m,p,u},k) \Bigg], \\ &\quad \text{if } \mathcal{P}_m(u) \neq \emptyset. \end{aligned} \tag{23}$$

The completion time of step u is then

$$\begin{aligned} C_{m,u} = S_{m,u} &+ \sum_{n \in \mathcal{V}} x_{m,u,n}\, T_{\text{exec}}(m,u,n,k) \\ &+ \sum_{n \in \mathcal{V}} x_{m,u,n}\, \widehat{T}_{\text{queue}}(n, \psi_{m,u}, k). \end{aligned} \tag{24}$$

Let $\mathcal{U}_m^{\text{exit}}$ be the set of exit steps with no outgoing dependency. The end-to-end latency of request τ_m is

$$L_m(\mathbf{X},k) = \max_{u \in \mathcal{U}_m^{\text{exit}}} C_{m,u} - A_m. \tag{25}$$

The average latency objective in window k is

$$J_L(\mathbf{X},k) = \frac{1}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} L_m(\mathbf{X},k). \tag{26}$$

2) Operational Cost: The operational cost includes computation cost and network transfer cost. The average cost per request is

$$\begin{aligned} J_C(\mathbf{X},k) = &\frac{1}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \sum_{u \in \mathcal{U}_m} \sum_{n \in \mathcal{V}} x_{m,u,n}\, \epsilon_n^{\text{comp}}(k)\, F_{m,u} \\ &+ \frac{1}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \sum_{(p,u) \in \mathcal{D}_m} \sum_{(i,j) \in \mathcal{E}(k)} \\ &y_{m,p,u,i,j}\, \gamma_{ij}^{\text{net}}(k)\, D_{m,p,u}, \end{aligned} \tag{27}$$

where $\gamma_{ij}^{\text{net}}(k)$ is the unit data-transfer price or accounting cost of link (i,j) in window k. The unit prices $\epsilon_n^{\text{comp}}(k)$ and $\gamma_{ij}^{\text{net}}(k)$ are fixed by the evaluation configuration, and their calibration is reported with the experimental setup.

3) Reliability and Cross-Domain Risk: We define risk as a service-level penalty associated with unreliable node usage and cross-region workflow coupling. The average risk is

$$\begin{aligned} J_R(\mathbf{X},k) = &\frac{\kappa_{\text{rel}}}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \sum_{u \in \mathcal{U}_m} \sum_{n \in \mathcal{V}} x_{m,u,n} \big(1 - p_n^{\text{rel}}(k)\big) \\ &+ \frac{\kappa_{\text{geo}}}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \sum_{(p,u) \in \mathcal{D}_m} \sum_{(i,j) \in \mathcal{E}(k)} y_{m,p,u,i,j} \\ &\times \mathbb{I}\big(\text{Region}(i) \neq \text{Region}(j)\big), \end{aligned} \tag{28}$$

where κ_{rel} and κ_{geo} balance node availability risk and cross-domain dependency risk. This term quantifies network- and service-level reliability rather than the semantic quality of LLM outputs. Additional deployment risks, such as data-residency constraints or link-failure probabilities, can be incorporated as separate additive terms.

4) SLA Violation and Composite QoS Score: The SLA violation rate is reported separately from latency:

$$V_{\text{SLA}}(\mathbf{X},k) = \frac{1}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \mathbb{I}\big(L_m(\mathbf{X},k) > T_m^{\text{SLA}}\big). \tag{29}$$

For optimization, SLA can be handled as either a hard constraint or a soft penalty. To avoid infeasibility under bursty traffic, we use a slack variable $\xi_m \geq 0$:

$$L_m(\mathbf{X},k) \leq T_m^{\text{SLA}} + \xi_m, \quad \forall \tau_m \in \mathcal{T}(k). \tag{30}$$

To avoid confusion with LLM semantic accuracy or human preference, we report a Composite QoS Score as an auxiliary scheduling metric. Let $\widetilde{J}_L$, $\widetilde{J}_C$, and $\widetilde{J}_R$ be normalized latency, cost, and risk values computed using fixed normalization constants from the evaluation protocol. A transparent composite score can be written as

$$\begin{aligned} S_{\text{QoS}} = 100\big[&\eta_L(1-\widetilde{J}_L) + \eta_C(1-\widetilde{J}_C) \\ &+ \eta_R(1-\widetilde{J}_R) + \eta_S(1-V_{\text{SLA}})\big], \end{aligned} \tag{31}$$

where $\eta_L, \eta_C, \eta_R, \eta_S \geq 0$ and $\eta_L + \eta_C + \eta_R + \eta_S = 1$. The score in (31) serves only as a secondary, aggregate indicator; the primary analysis relies on the separately reported latency, cost, risk, and SLA-violation metrics.

G. Preference-Conditioned Optimization Problem

The objectives above are conflicting: selecting a nearby or reliable node may increase computation cost, while selecting a low-cost node may increase communication delay or cross-region risk. We therefore formulate a preference-conditioned routing objective. Let $\boldsymbol{\omega}(k) = [\omega_L(k), \omega_C(k), \omega_R(k)]$ be a nonnegative preference vector satisfying $\omega_L(k) + \omega_C(k) + \omega_R(k) = 1$. The optimization problem in window k is

$$\begin{aligned} \mathcal{P}_1: \ \min_{\mathbf{X}, \boldsymbol{\xi}} \ &\omega_L(k) \frac{J_L(\mathbf{X},k)}{\overline{J}_L} + \omega_C(k) \frac{J_C(\mathbf{X},k)}{\overline{J}_C} \\ &+ \omega_R(k) \frac{J_R(\mathbf{X},k)}{\overline{J}_R} + \chi_{\text{SLA}} \frac{1}{|\mathcal{T}(k)|} \sum_{\tau_m \in \mathcal{T}(k)} \frac{\xi_m}{T_m^{\text{SLA}}} \\ \text{s.t. } &(16),\ (17),\ (18),\ (31), \\ &x_{m,u,n} \in \{0,1\}, \quad y_{m,p,u,i,j} \in \{0,1\}, \quad \xi_m \geq 0. \end{aligned} \tag{32}$$

Here $\overline{J}_L$, $\overline{J}_C$, and $\overline{J}_R$ are fixed or moving normalization constants used to prevent one physical unit from dominating the scalarized objective, and $\chi_{\text{SLA}} \geq 0$ is a fixed penalty coefficient for deadline violations. The preference vector can be set manually, swept to study trade-offs, or adapted by the learning algorithm. Weighted scalarization provides a tractable

Algorithm 1 STAR-PPO Training Framework

Require: Graph $\mathcal{G}(k)$, Task stream $\mathcal{D}_{stream}$, Epochs $N_{\max}$.
Initialize: Policy π_θ, Value V_ϕ, DWA weights $\boldsymbol{\omega}(1)$, and global environment state $\mathcal{S}_{global}$.
for epoch $\ell = 1$ **to** $N_{\max}$ **do**
  $\mathcal{B} \leftarrow \emptyset$; Update preference $\boldsymbol{\omega}(\ell)$ via Alg. 2 (if $\ell > 1$).
  for episode $e = 1$ **to** E_{batch} **do**
    Sample task $\tau_m \sim \mathcal{D}_{stream}$ with DAG $\mathcal{W}_m$; $t \leftarrow 0$.
    while task τ_m is not completed **do**
      Sub-task Selection: Identify ready set $\mathcal{R}_t$ and select $u_t \in \mathcal{R}_t$ via *Critical-Path-First* heuristic.
      Observation: specific $\mathbf{s}_t \leftarrow [\mathbf{s}_{task}(u_t), \mathbf{s}_{topo}(\mathcal{S}_{global}), \boldsymbol{\omega}(\ell)]$ and mask invalid nodes.
      Step: Sample $a_t \sim \pi_\theta(\cdot|\mathbf{s}_t)$; Update placement $\mathbf{X}_{t+1}$ and load state $\mathcal{S}_{global}$ (resource occupancy).
      Reward: Compute r_t via Eq. (35) with SLA penalty (if terminal); Store transition in $\mathcal{B}$.
      $t \leftarrow t + 1$.
    end while
  end for
  Update: Compute GAE advantages and optimize θ, ϕ via PPO clipped objective using batch $\mathcal{B}$.
  Feedback: Aggregate epoch metrics $\{O_L, O_C, O_R\}$ for next DWA step.
end for

preference-conditioned objective, but it does not guarantee that every point on a non-convex Pareto frontier is attainable. For this reason, latency, cost, and risk are reported separately in the experiments so that the underlying trade-off remains visible rather than being collapsed into a single scalar.

The problem is NP-hard by reduction from precedence-constrained multiprocessor DAG scheduling. Consider a restricted case with a single request, one service type available on every node, constant queueing delay, zero communication cost, zero risk, and a latency-only objective. The problem reduces to assigning DAG tasks with heterogeneous processing times to machines so as to minimize the makespan, which is a classic NP-hard scheduling problem. Since this restricted case is contained in $\mathcal{P}_1$, the general service-overlay workflow routing problem is also NP-hard. This motivates the learning-based online policy developed in the following section.

IV. Topology-Aware and Preference-Adaptive Policy Optimization

Addressing the computational intractability and non-stationary dynamics inherent in the service-overlay workflow routing problem $\mathcal{P}_1$, we propose STAR-PPO, a specialized deep reinforcement learning framework designed to learn a robust placement policy π_θ. The overall architecture is illustrated in Fig. 3. Under the revised formulation, STAR-PPO observes workflow-step requirements, local service-overlay telemetry, and the current preference vector, and then selects a feasible service node for each ready workflow step. The training pipeline is formally summarized in **Alg.** 1.

A. Augmented MDP Formulation

We formulate the orchestration process as an augmented Markov Decision Process (MDP) [30], represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$.

1) Task-Centric Episode With Global Concurrency: To handle multi-task concurrency, we adopt a *task-centric* episode structure. Each episode corresponds to the sequential scheduling of a single DAG task τ_m from its entry to completion. While the planning process is serialized per task, the environment state $\mathcal{S}_{global}$ remains shared across all episodes. Specifically, the queue estimates $\widehat{T}_{\text{queue}}(\cdot,\cdot,k)$, effective link states $\{R_{ij}(k), T_{ij}^{\text{base}}(k), Q_{ij}^{\text{net}}(k)\}$, shared-resource indicators, and reliability estimates $p_n^{\text{rel}}(k)$ are updated from aggregate telemetry across concurrent tasks.

2) Augmented State Space ($\mathcal{S}$): At decision step t, the state $\mathbf{s}_t \in \mathcal{S}$ is defined as a concatenation of three feature vectors: $\mathbf{s}_t = [\mathbf{s}_{task}, \mathbf{s}_{topo}, \mathbf{s}_{pref}]$.

- *Task Semantics ($\mathbf{s}_{task}$):* Encodes the properties of the current workflow step u_t, including the required service type ψ_{m,u_t}, computation demand F_{m,u_t}, input/intermediate data size, and the predecessor-location embedding $\mathbf{e}_{m,u_t}^{\text{dep}}$ defined in Eq. (11).
- *Topological Telemetry ($\mathbf{s}_{topo}$):* We construct a locality feature vector to capture the service-overlay context. Let n_{curr} denote the node hosting the primary predecessor (the one with the largest data transfer D_{m,p,u_t}). The vector $\mathbf{s}_{topo}$ aggregates the real-time status of n_{curr} and its one-hop overlay neighbors:

$$\mathbf{s}_{topo} = \left[\widehat{T}_{\text{queue}}(n_{curr}, \psi_{m,u_t}, k), \Phi(\mathcal{N}(n_{curr})) \odot \mathbf{1}_{cong}\right], \tag{33}$$

  where $\mathcal{N}(\cdot)$ denotes the neighborhood set, and Φ is a permutation-invariant aggregation operator (e.g., mean/max pooling) applied to neighbor attributes. The neighbor attributes include effective link rate/delay, network queuing state, resource-group congestion indicators, and node-side queue/reliability telemetry when available. Here, $\mathbf{1}_{cong}$ is a binary indicator vector masking highly congested links, which explicitly encodes local bottleneck information into the state representation.
- *Preference Embedding ($\mathbf{s}_{pref}$):* Contains the current objective weight vector $\boldsymbol{\omega}(k)$ derived from the DWA module. This input conditions the policy on the specific optimization weights of the current epoch.

3) Action Space and Validity Masking ($\mathcal{A}$): To strictly enforce the service-type availability constraint (17), we apply a validity mask $M(\mathbf{s}_t, u_t)$ to the policy logits via

$$M_n(\mathbf{s}_t, u_t) = \begin{cases} 0, & \text{if } \psi_{m,u_t} \in \mathcal{M}_n, \\ -\infty, & \text{otherwise.} \end{cases} \tag{34}$$

By assigning $-\infty$ logits to infeasible nodes, the resulting probability distribution restricts the action support to the subset of nodes satisfying $\psi_{m,u_t} \in \mathcal{M}_n$.

4) Reward Design ($\mathcal{R}$): We design a step-wise reward function aligned with the scalarized global objective $\mathcal{P}_1$. At each step t, the immediate reward r_t is defined as the negative

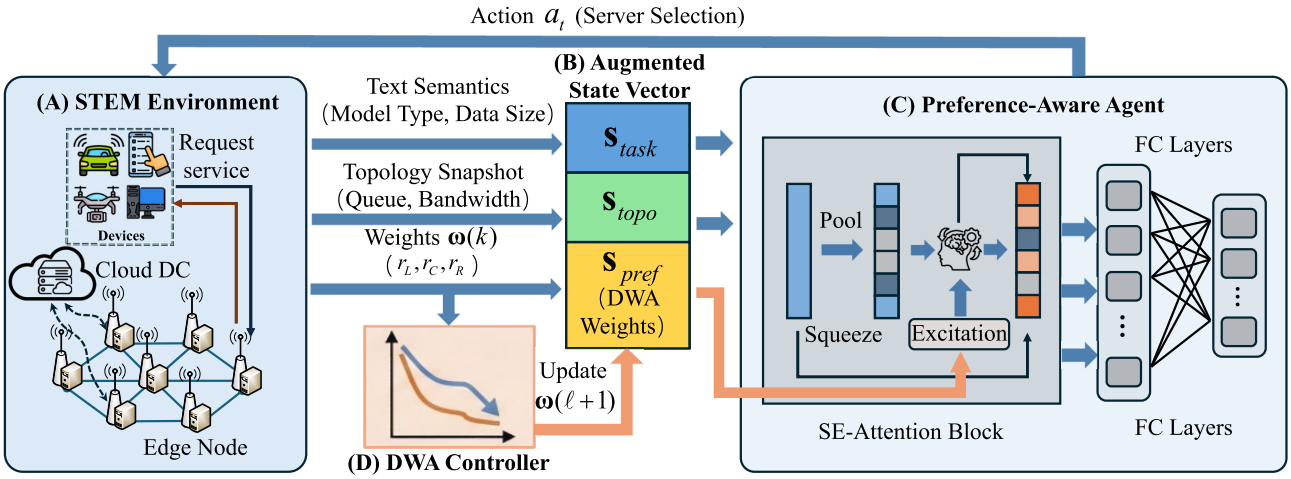


Fig. 3. The schematic architecture of the STAR-PPO framework.

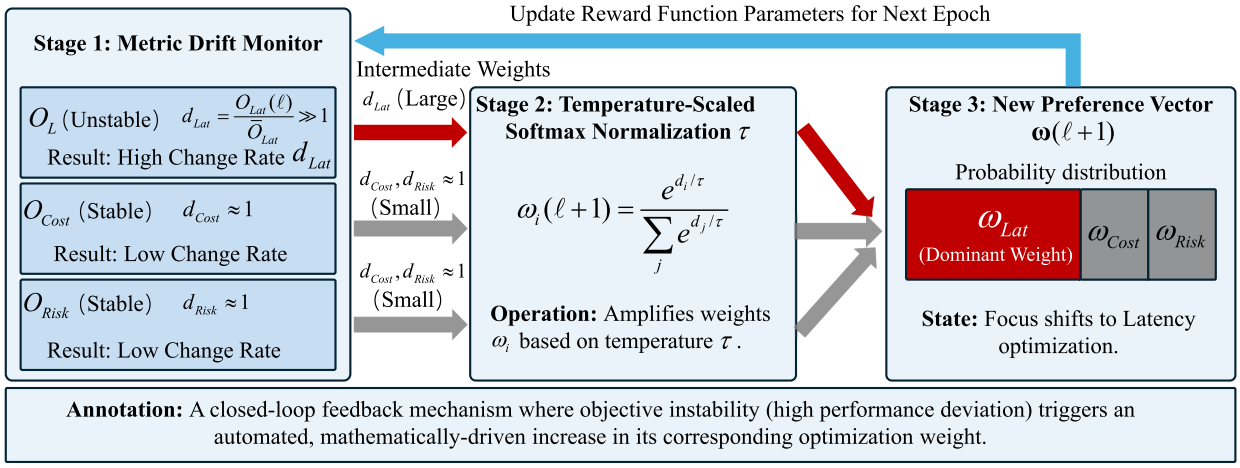


Fig. 4. The internal mechanism of Dynamic Weight Adaptation (DWA).

incremental contribution of the assignment $(u_t \mapsto a_t)$ to the global objectives as

$$r_t = -\sum_{obj\in\{L,C,R\}} \omega_{obj}(k)\frac{\Delta J_{obj}}{\bar{J}_{obj}}, \tag{35}$$

where ΔJ_{obj} represents the marginal increase in Latency (J_L), Cost (J_C), or Risk (J_R) resulting from the current action. For latency estimation ΔJ_L under partial placement, we employ a critical path method estimator with an optimistic lower bound for unplaced sub-tasks.

B. Preference-Aware Neural Architecture

To support fine-grained trade-offs, STAR-PPO employs a neural architecture explicitly formulated to perceive dynamic optimization priorities. As illustrated in Fig. 4, the network processes the composite state $\mathbf{s}_t$ through a feature recalibration module based on the Squeeze-and-Excitation (SE) mechanism.

1) Feature Extraction and Fusion: The task semantics $\mathbf{s}_{task}$ and topological telemetry $\mathbf{s}_{topo}$ are projected into latent embeddings via separate Multi-Layer Perceptrons (MLPs). These embeddings are subsequently concatenated to form a joint feature representation $\mathbf{h}_{joint}$. At this stage, $\mathbf{h}_{joint}$ aggregates the observable system status prior to the integration of preference weights.

Algorithm 2 Dynamic Weight Adaptation (DWA)

Require: Historical metric baseline $\bar{\mathbf{O}} = [\bar{O}_L, \bar{O}_C, \bar{O}_R]$; Current epoch metric averages $\mathbf{O}(\ell) = [O_L(\ell), O_C(\ell), O_R(\ell)]$; Temperature τ; Smoothing factor β; Stability constant ϵ.
Ensure: Updated preference weights $\boldsymbol{\omega}(\ell+1)$.
for each objective $i \in \{L, C, R\}$ **do**
  Calculate ratio: $d_i \leftarrow \frac{O_i(\ell)}{\bar{O}_i + \epsilon}$
  Clip d_i to interval $[0.5, 2.0]$ to prevent numerical instability.
end for
Compute normalizer: $Z \leftarrow \sum_{j\in\{L,C,R\}} \exp(d_j/\tau)$
for each objective $i \in \{L, C, R\}$ **do**
  $\omega_i(\ell+1) \leftarrow \frac{\exp(d_i/\tau)}{Z}$
end for
for each objective $i \in \{L, C, R\}$ **do**
  $\bar{O}_i \leftarrow \beta \cdot \bar{O}_i + (1-\beta) \cdot O_i(\ell)$
end for
return $\boldsymbol{\omega}(\ell+1)$

2) Preference-Driven Recalibration (SE Block): To incorporate the optimization objectives into the decision logic, we implement a modified SE block where the preference embedding $\mathbf{s}_{pref}$ governs the channel-wise excitation. Distinct from standard SE blocks that derive excitation weights solely from global pooling, this architecture utilizes the weight vector $\boldsymbol{\omega}(k)$ as a gating signal. Specifically, a gating network maps the concatenated vector $[\mathbf{h}_{joint}; \mathbf{s}_{pref}]$ to a channel-wise weight vector $\mathbf{g} \in (0,1)^d$. This vector modulates the joint features via element-wise multiplication using:

$$\tilde{\mathbf{h}} = \mathbf{g} \odot \mathbf{h}_{joint}. \tag{36}$$

3) Policy and Value Heads: The recalibrated representation $\tilde{\mathbf{h}}$ serves as input to the Actor and Critic heads. The Actor head outputs a probability distribution over the nodes, restricted by the validity mask $M(\mathbf{s}_t, u_t)$, while the Critic head estimates the value function $V(\mathbf{s}_t, \boldsymbol{\omega}(k))$ to guide the PPO updates.

C. Dynamic Weight Adaptation (DWA)

The global objective function $\mathcal{P}_1$ aggregates metrics characterized by physical dimensions, spanning millisecond-level latency, monetary operational costs, and discrete risk counts. Static scalarization often leads to optimization bias where gradients are dominated by the objective with the largest scale. To address this, we propose the DWA mechanism, which dynamically adjusts the preference vector $\boldsymbol{\omega}$ at the epoch boundary based on the relative performance drift. The operation of this feedback loop is illustrated in Fig. 4.

DWA operates on the epoch-level aggregate metrics $O_i(\ell)$, defined as the empirical mean of the objectives $\{J_L, J_C, J_R\}$ observed during epoch ℓ. As detailed in **Alg.** 2, the adaptation process consists of three sequential steps:

1) *Scale-Invariant Deviation:* For each objective i, we compute the ratio $d_i = O_i(\ell)/\bar{O}_i$ between the current observation and a historical moving average $\bar{O}_i$. This division eliminates unit dependence, yielding a dimensionless indicator of performance degradation.
2) *Boltzmann Weighting:* To prioritize lagging objectives, we map the deviations to the weight space using a Boltzmann distribution, e.g., Softmax, with temperature τ. An objective with a higher relative deviation d_i, signifying performance degradation, is assigned a larger weight $\omega_i(\ell+1)$ for the subsequent epoch, thereby directing the policy optimization to compensate for this specific metric.
3) *Baseline Update:* The historical baseline $\bar{O}_i$ is updated via an Exponential Moving Average (EMA) to track non-stationary shifts in the environment statistics.

D. Convergence and Complexity Analysis

1) Pareto Stationarity: First, we examine the relationship between the stationary points of the scalarized objective derived from DWA and the Pareto frontier of the original multi-objective problem $\mathcal{P}_1$.

Proposition 1: Let $\boldsymbol{\omega}^*$ be the weight vector derived from the DWA mechanism, and let $\mathcal{J}(\pi_\theta; \boldsymbol{\omega}^*)$ denote the scalarized objective function weighted by $\boldsymbol{\omega}^*$. If the policy π_{θ^*} is a stationary point of $\mathcal{J}(\pi_\theta; \boldsymbol{\omega}^*)$, then π_{θ^*} is a Pareto stationary point for the vector-valued optimization problem $\mathcal{P}_1$.

Proof: We proceed by contradiction. Suppose that π_{θ^*} is not a Pareto stationary point. By definition, there must exist a feasible descent direction $\mathbf{h}$ in the parameter space such that the gradient of every objective function J_i ($i \in \{L, C, R\}$) along $\mathbf{h}$ is strictly negative:

$$\nabla_\theta J_i(\pi_{\theta^*})^\top \mathbf{h} < 0, \quad \forall i \in \{L, C, R\}. \tag{37}$$

Consider the gradient of the scalarized objective $\mathcal{J}(\pi_\theta; \boldsymbol{\omega}^*)$ at θ^*. Since $\mathcal{J}$ is a linear combination of the objectives with non-negative weights $\omega_i^* \geq 0$ (assuming $\sum \omega_i^* > 0$), we have:

$$\nabla_\theta \mathcal{J}(\pi_{\theta^*}; \boldsymbol{\omega}^*)^\top \mathbf{h} = \sum_{i \in \{L,C,R\}} \omega_i^* \left(\nabla_\theta J_i(\pi_{\theta^*})^\top \mathbf{h} \right). \tag{38}$$

Given that $\omega_i^* \geq 0$ and at least one weight is strictly positive, the term on the right-hand side is strictly negative. This implies:

$$\nabla_\theta \mathcal{J}(\pi_{\theta^*}; \boldsymbol{\omega}^*)^\top \mathbf{h} < 0. \tag{39}$$

The existence of such a direction $\mathbf{h}$ indicates that $\mathcal{J}(\pi_{\theta^*}; \boldsymbol{\omega}^*)$ can be minimized, which contradicts the hypothesis that π_{θ^*} is a stationary point (where $\nabla_\theta \mathcal{J} = \mathbf{0}$ or satisfies the first-order optimality condition). Therefore, no such descent direction exists, and π_{θ^*} is Pareto stationary. ■

2): Topological Sufficiency of Graph-Free Perception: We next make explicit the conditions under which the graph-free state $\mathbf{s}_t$ carries enough topological information to support the per-step routing decision, so that message passing over the global graph $\mathcal{G}(k)$ is not required. For a ready step u with predecessor set $\mathcal{P}_m(u)$, let $\Delta_{m,u}(n)$ denote the marginal contribution of assigning u to a feasible candidate node n to the scalarized objective of $\mathcal{P}_1$, and let $\boldsymbol{\ell}_{i,j}(k) = \big(R_{ij}(k), T_{ij}^{\text{base}}(k), Q_{ij}^{\text{net}}(k)\big)$ be the link state. We introduce two structural conditions.

Proposition 2: Assume that: (C1) Additive locality: for every feasible candidate n, the marginal cost decomposes as

$$\Delta_{m,u}(n) = \phi_{\text{node}}\big(\mathbf{a}_n(k)\big) + c_0 + \sum_{p \in \mathcal{P}_m(u)} \phi_{\text{link}}\big(\boldsymbol{\ell}_{n_{m,p},n}(k), D_{m,p,u}\big), \tag{40}$$

where ϕ_{node} depends only on the attributes and queue telemetry of n, ϕ_{link} depends only on the state of the links joining each predecessor host $n_{m,p}$ to n, and c_0 is constant across candidates; and (C2) one-hop observability: the arguments of ϕ_{node} and ϕ_{link} are encoded in the local telemetry of n and its one-hop overlay neighbors captured by $\mathbf{s}_{topo}$, together with the predecessor-location embedding $\mathbf{e}_{m,u}^{\text{dep}}$ in $\mathbf{s}_{task}$. Then $\mathbf{s}_t$ is a sufficient statistic for ranking candidates by marginal cost: there exists a map g such that, for any feasible n, n',

$$\Delta_{m,u}(n) \leq \Delta_{m,u}(n') \iff g(\mathbf{s}_t, n) \leq g(\mathbf{s}_t, n'), \tag{41}$$

so the greedy per-step assignment $\arg\min_n \Delta_{m,u}(n)$ is recoverable from $\mathbf{s}_t$ without message passing over $\mathcal{G}(k)$.

Proof: By (C1) the only candidate-dependent terms are $\phi_{\text{node}}(\mathbf{a}_n(k))$ and the predecessor-link sum, since c_0 is shared by all candidates. By (C2) the arguments of these terms are measurable functions of $\mathbf{s}_{topo}$ (the local node and link telemetry of n and its one-hop neighbors) and of $\mathbf{s}_{task}$ (the predecessor locations through $\mathbf{e}^{\text{dep}}_{m,u}$ and the transfer sizes $D_{m,p,u}$). Defining $g(\mathbf{s}_t, n)$ as the right-hand side of (40) without c_0 yields a function of $\mathbf{s}_t$ and n only, which establishes (41) and hence preserves the minimizer. Any embedding produced by message passing over $\mathcal{G}(k)$ is a function of the same telemetry and therefore cannot alter an ordering already determined by g. Finally, the SE block rescales feature channels by strictly positive gates $\mathbf{g} \in (0,1)^d$, a monotone per-channel transformation that preserves this ordering. ■

Proposition 2 delimits when graph-free perception is information-sufficient: the marginal cost must be dominated by node-local and predecessor-link bottlenecks that are observable within one overlay hop. When (C1)–(C2) are violated, for instance under long-range coupling or non-additive interactions induced by shared-resource contention among distant links, $\mathbf{s}_t$ becomes an approximation of the bottleneck-sufficient statistic rather than an exact one. The residual cross-region zero-shot gap reported in Sec. V is consistent with such higher-order coupling, which message-passing encoders may capture at the cost of $\mathcal{O}(|\mathcal{V}|^2)$ or $\mathcal{O}(|\mathcal{E}|)$ complexity.

3): Computational Complexity: We analyze the asymptotic time and space complexity of the STAR-PPO inference phase. Let $|\mathcal{V}|$ denote the number of physical nodes, D_{in} the input feature dimension, and H the width of the hidden layers.

For time complexity, the generation of a placement decision involves three sequential operations: feature aggregation, neural inference, and action sampling. First, the construction of the topological vector $\mathbf{s}_{topo}$ requires querying the neighbors of the predecessor node; given that the overlay graph degree is bounded by a constant Δ_{deg}, this operation scales as $\mathcal{O}(\Delta_{deg})$. Second, the SE-augmented policy network processes the state through fixed-depth MLPs. The computational cost is dominated by matrix multiplications, scaling as $\mathcal{O}(L \cdot H^2 + |\mathcal{V}| \cdot H)$, where the term linear in $|\mathcal{V}|$ arises from the final projection to the action logits. Finally, applying the validity mask and sampling from the categorical distribution requires $\mathcal{O}(|\mathcal{V}|)$ operations. Consequently, the total per-step time complexity is $\mathcal{O}(|\mathcal{V}| \cdot H + H^2)$. This linear scaling with respect to the node set size $|\mathcal{V}|$ contrasts with the exponential complexity inherent in exact combinatorial solvers.

Regarding space complexity, the memory footprint is determined by the storage requirements for model parameters and the runtime state buffer. The parameter count scales as $\mathcal{O}(H^2 + D_{in}H + |\mathcal{V}|H)$, while the trajectory buffer for an episode of length T requires $\mathcal{O}(T \cdot D_{in})$. Thus, the space complexity remains linear, $\mathcal{O}(|\mathcal{V}| \cdot H)$, ensuring that the model fits within the memory constraints of edge controllers.

V. Experimental Analysis

A. Experimental Setup

1) Implementation Details: The STEM prototype is implemented with the Gymnasium interface and trained on a workstation with four NVIDIA GeForce RTX 3090 GPUs, across which rollout collection is parallelized. The policy is optimized by PPO with the scalarized reward of Eq. (35), and every learned result is averaged over five random seeds, with the shaded regions in the learning curves denoting the spread across seeds. For a fair comparison, all learned baselines share the same training budget and evaluation protocol. The remaining hyperparameters follow Tab. II.

TABLE II
Hyperparameter Configuration for STAR-PPO

Module	Hyperparameter	Value
PPO Backbone	Batch Size (Horizon)	1,024
	Update Epochs (K)	10
	Learning Rate	1×10^{-4}
	Discount Factor (γ)	0.99
	Clip Range (ϵ)	0.2
	Entropy Coeff.	$0.03 \rightarrow 0.002$
	GAE Parameter (λ)	0.95
	Max Gradient Norm	0.5
	Latent State Dim.	$10 \sim 20$
STAR & DWA Mechanism	Resource Guidance (α)	$0.0 \rightarrow 1.0$
	DWA Temperature (T)	3.0
	SE-Block Reduction Ratio	16

2) Dataset and Workload: We construct three service-overlay topologies from OpenCelliD geospatial records [31]: Server1 (S1) contains 500 service nodes mapped to Switzerland, Server2 (S2) contains 1000 nodes mapped to the United Kingdom, and Server3 (S3) contains 2000 nodes mapped to Germany. Unless otherwise stated, the main comparison is conducted on the adversarial S1_Trap setting, where high-compute "bait" nodes are assigned degraded inbound links to test whether a scheduler can distinguish compute-rich but network-poor placement choices.

The workload consists of DAG-based inference workflows with 2–5 service steps. Each step specifies a service type, computation demand, input data volume, and request-level SLA. This setting is intended to test workflow-level orchestration rather than single-hop request routing. Baselines include PPO-Std [32], PF-PPO [25], PPO-CN [26], GA-PPO [27], Equity-Trans [20], STARK [28], A3C [24], Greedy, and Random.

B. Overall Performance and Multi-Objective Trade-off

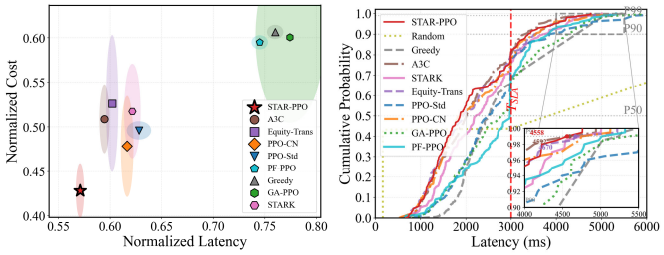
Tab. III reports the main S1_Trap inference results. STAR-PPO achieves the lowest average latency, 2112.22 ms, and the lowest P99 latency, 4558.04 ms. Relative to Greedy, this corresponds to a 24.4% reduction in average latency. The method also obtains the highest Composite QoS Score, 65.93, while maintaining the lowest reported service-level risk $J_R = 0$. The closest latency competitors, A3C and Equity-Trans, are within a small latency margin but incur higher cost and nonzero risk, which lowers their Composite QoS Scores to 56.95 and 57.07, respectively.

Beyond the metrics, Fig. 5 examines the comparison from two complementary angles, namely the latency–cost trade-off

TABLE III

PERFORMANCE COMPARISON ON S1_TRAP. COST AND COMPOSITE QOS VALUES ARE FROM THE METRIC-AUDIT DATA; P99 LATENCY AND DECISION TIME ARE COMPUTED FROM THE SAVED REQUEST-LEVEL NPZ FILES

Method	Latency (ms) ↓		Cost ↓	SLA Viol. ↓	QoS ↑	Decision ↓
	Avg	P99	(USD/req.)	(%)	(Score)	(ms)
STAR-PPO (Ours)	**2112.22**	**4558.04**	**0.0755**	**18.5**	**65.93**	22.22
A3C [24]	2135.40	4592.15	0.0916	19.5	56.95	23.07
Equity-Trans [20]	2158.75	4670.30	0.0896	20.0	57.07	26.24
PPO-CN [26]	2185.22	4751.65	0.0812	22.0	58.17	24.31
STARK [28]	2358.90	4748.20	0.1485	24.5	44.13	197.28
PPO-Std [32]	2634.55	5722.80	0.0978	22.0	50.99	23.64
PF-PPO [25]	2726.15	5018.45	0.2522	32.5	31.13	24.89
GA-PPO [27]	2744.80	5362.30	0.2545	34.0	30.71	25.37
Greedy	2793.58	5237.85	0.2709	31.5	30.47	0.80
Random	3868.79	10797.49	0.2165	56.5	18.30	0.49



(a) Latency–cost Pareto distribution. (b) CDF of end-to-end latency.

Fig. 5. Overall trade-off on S1_Trap: (a) latency–cost Pareto distribution and (b) CDF of end-to-end latency.

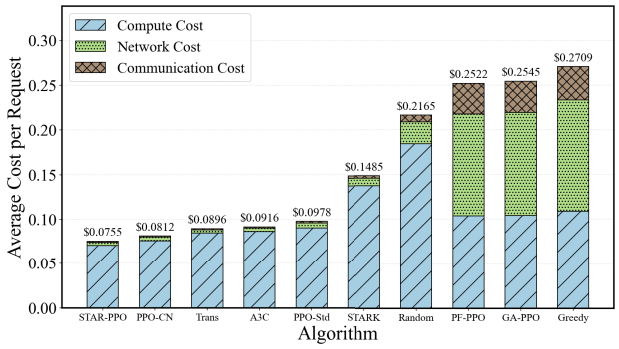
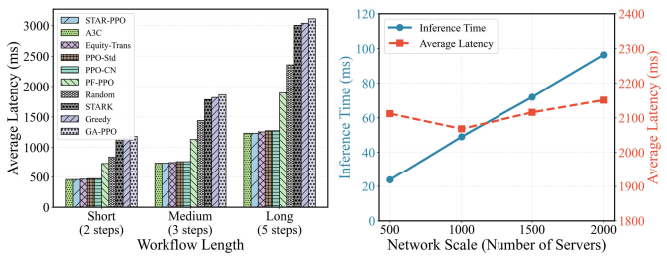


Fig. 6. Cost decomposition into computation, network, and communication components.



(a) Latency under different workflow lengths. (b) Single-policy scaling from 500 to 2000 nodes.

Fig. 7. Workflow awareness and scalability: (a) latency versus workflow length and (b) single-policy scaling from 500 to 2000 service nodes.

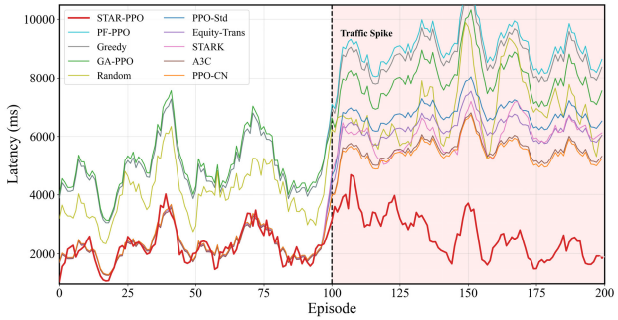


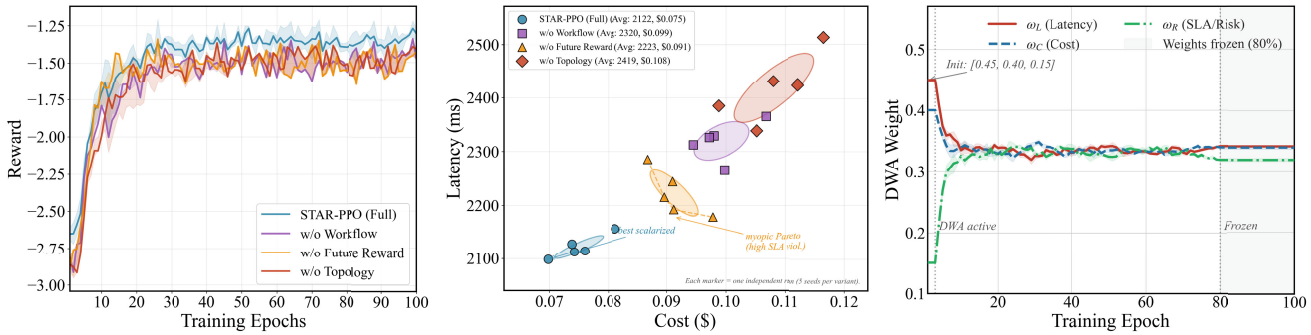
Fig. 8. Latency response after an injected traffic spike/failure event.

and the latency tail. In the latency–cost plane of Fig. 5(a), STAR-PPO occupies the lower-left region, which reflects a joint reduction of latency and cost rather than a single-metric gain. The CDF in Fig. 5(b) shows that the policy controls the latency tail: its P99 latency of 4558.04 ms is below A3C (4592.15 ms), Equity-Trans (4670.30 ms), and the remaining baselines. The two views are mutually consistent and follow from the service-overlay state design, in which the policy observes node-side load together with link-side bottlenecks before committing a workflow step to a service node.

The cost decomposition in Fig. 6 explains why STAR-PPO can keep total cost low without becoming a purely cheapest-node heuristic. Its average network cost is 0.0040 USD/request, compared with 0.1248 USD/request for Greedy, a 96.8% reduction. Greedy reduces the placement decision to local compute price and therefore frequently induces expensive cross-region transfers. STAR-PPO, in contrast, uses dependency-aware state features and local service-overlay telemetry, so it can select nodes that preserve data locality while avoiding congested paths.

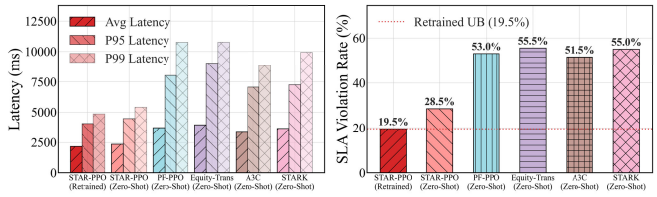
C. Workflow Awareness, Adaptability, and Scalability

Fig. 7 reports two complementary views of the policy under varying problem structure: latency as the workflow grows



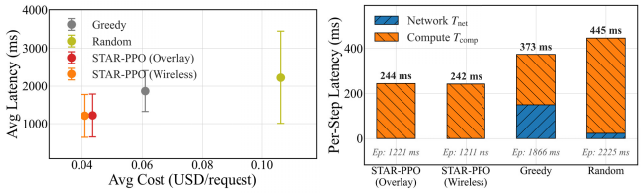
(a) Reward learning curves for ablated variants. (b) Ablation-induced shifts in the latency–cost plane. (c) Evolution of DWA weights and scalarized reward during training.

Fig. 9. Component and preference ablation on S1_Trap: (a) reward learning curves, (b) latency–cost shifts after ablation, and (c) evolution of DWA weights and scalarized reward.



(a) Average, P95, and P99 latency. (b) SLA violation rate.

Fig. 10. Strict cross-region zero-shot transfer (train on S1/Switzerland, test on S3/Germany): (a) latency and (b) SLA violation rate.



(a) Latency–cost comparison across link models. (b) Per-step communication/computation breakdown.

Fig. 11. Service-overlay link model and wireless-edge instantiation.

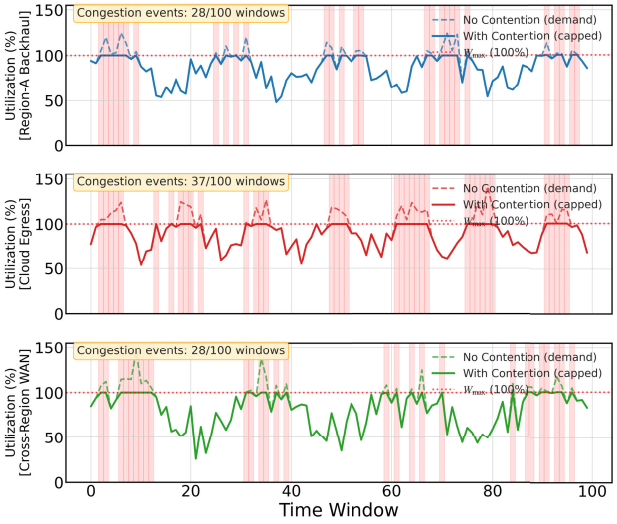


Fig. 12. Shared-resource utilization.

longer in panel (a), and single-policy behavior as the network scales in panel (b). Fig. 7(a) tests whether the policy merely solves local placement or can preserve good decisions across a dependency chain. STAR-PPO keeps the lowest average latency across 2-step, 3-step, and 5-step workflows; for 5-step workflows its average latency is 1233.29 ms, while PF-PPO and STARK reach 1900.84 ms and 3003.74 ms, respectively. The gap widens with workflow length, which is consistent with the predecessor-location embedding and future-return optimization that let the policy account for downstream steps.

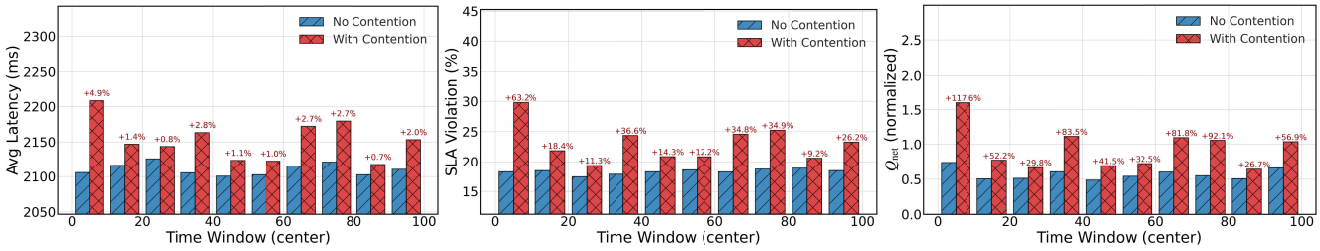
Fig. 7(b) then evaluates the cost of enlarging the action space as the network grows. The average decision time of STAR-PPO increases from 23.72 ms at 500 service nodes to 96.44 ms at 2000 nodes, while its average latency stays within 2067.85–2151.50 ms. This matches the complexity analysis in Sec. IV: the final action projection scales with $|\mathcal{V}|$, whereas the graph-free representation avoids full message passing over the service graph. The decision time therefore corresponds to millisecond-level online responsiveness rather than sub-millisecond scheduling.

Fig. 8 examines non-stationary adaptation by injecting a disturbance at the middle of the episode sequence. STAR-PPO shows a transient latency increase after the event but returns to a low-latency regime faster than the topology-agnostic and myopic baselines. This behavior follows from the window-level state design: node availability and queue/link telemetry are updated each decision window, so the policy reacts when the effective service overlay changes instead of relying on static node attributes.

D. Component and Preference Ablation

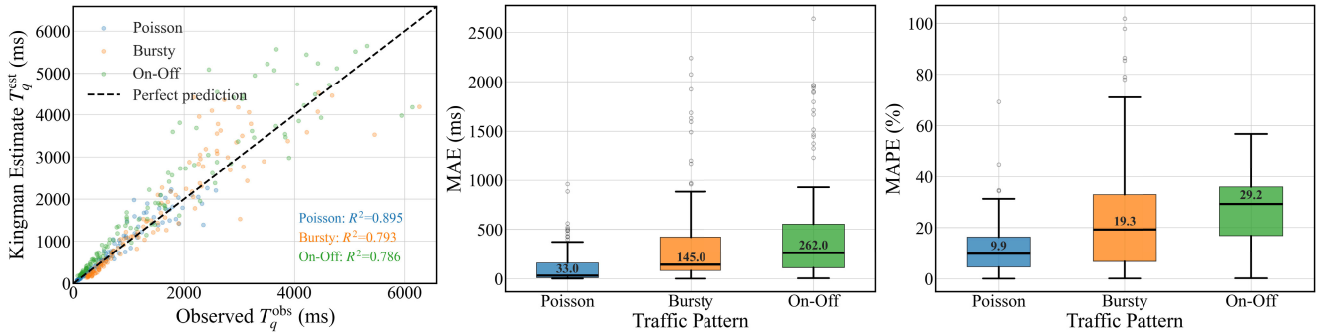
We next isolate the contributions of workflow awareness, topology telemetry, future reward propagation, and DWA. The S1_Trap ablation removes one component at a time while keeping the training and evaluation protocol unchanged.

Fig. 9 and Tab. IV together isolate the contribution of each design choice. The learning curves in Fig. 9(a) show that the full model attains the highest and most stable return, whereas removing any single component lowers the converged reward



(a) Latency impact. (b) SLA violation impact. (c) Network queueing impact.

Fig. 13. Validation of shared-resource contention constraints.



(a) Estimated versus observed queue delay. (b) MAE distribution. (c) MAPE distribution.

Fig. 14. Queueing-approximation validation under Poisson, bursty, and On–Off arrivals.

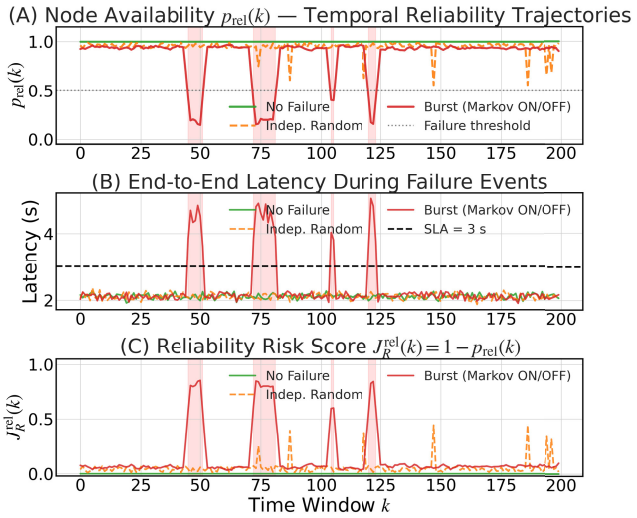


Fig. 15. Temporal reliability trajectories and failure-induced latency/risk responses.

TABLE IV
MAIN ABLATION IMPACT ON AVERAGE LATENCY J_L AND COST J_C

Variant	WF	FR	Topo	Avg Lat.↓	Avg Cost↓	ΔLat.	ΔCost
Full	✓	✓	✓	**2122.00**	**0.0750**	–	–
w/o Workflow	✗	✓	✓	2319.74	0.0993	+9.3%	+32.3%
w/o Future Reward	✓	✗	✓	2223.18	0.0913	+4.8%	+21.7%
w/o Topology	✓	✓	✗	2418.54	0.1081	+14.0%	+44.2%

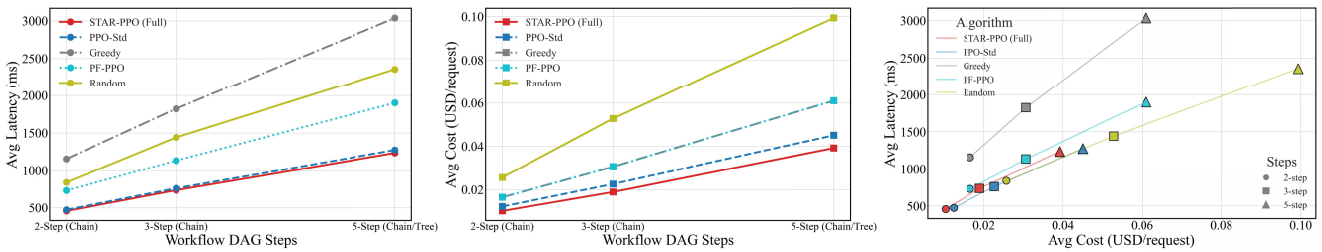
and slows training. The latency–cost plane in Fig. 9(b) shows the corresponding operating points: the full model stays in the lower-left region, while the ablated variants drift toward higher latency and cost. These trends are quantified in Tab. IV. Topology telemetry has the largest effect, as removing $\mathbf{s}_{topo}$ raises latency by 14.0% and cost by 44.2%; this is consistent with the trap-node design, in which a link- and queue-blind agent is more likely to route steps through attractive but congested nodes. Removing workflow awareness raises latency by 9.3% and cost by 32.3%, indicating that predecessor placement and intermediate-transfer locality are also central to the proposed workflow-aware routing formulation.

Finally, Fig. 9(c) gives a view of the adaptive scalarization process. Starting from $(\omega_L, \omega_C, \omega_R) = (0.45, 0.40, 0.15)$, DWA shifts the final mean weights to approximately $(0.3415, 0.3395, 0.3190)$ across five seeds, so that the risk/SLA-related weight increases by 112.7% while its normalized loss term decreases by 76.5% between the first and final epoch. This does not remove the limitation of linear scalarization on non-convex Pareto fronts; rather, it shows that the feedback controller prevents any single objective from dominating the learned policy under the evaluated settings.

E. Reviewer-Driven Validation

We next stress-test the modeling assumptions along several axes: strict zero-shot transfer, network modeling, shared-resource contention, queueing approximation, temporal reliability, metric transparency, traffic robustness, and decision-time consistency.

Fig. 10 follows a strict cross-region protocol, training on S1/Switzerland and testing directly on S3/Germany, with latency reported in panel (a) and SLA violation in panel (b). In Fig. 10(a), STAR-PPO zero-shot reaches 2349.6 ms average



(a) Workflow latency. (b) Workflow cost. (c) Latency–cost shift.

Fig. 16. Workflow/topology-awareness validation under different DAG lengths.

latency against 2151.5 ms for the retrained S3 reference, i.e., a 1.092× gap; the closest transferable baseline, A3C, reaches 3343.1 ms, so STAR-PPO lowers zero-shot latency by 29.7% relative to it. Fig. 10(b) reports the corresponding SLA violation rate, which rises from 19.5% under retraining to 28.5% under zero-shot transfer. Reporting both panels makes the residual cost of operating across an unseen region explicit instead of hiding it in the average latency alone.

Fig. 11 compares the service-overlay link model with its wireless-edge instantiation. The latency–cost scatter in Fig. 11(a) shows that STAR-PPO attains 1221.48 ms average latency and 0.0435 USD/request under the service-overlay model, against 1210.89 ms and 0.0409 USD/request under the wireless-edge instantiation, while Greedy and Random remain at higher latency and cost on the same data. The per-step decomposition in Fig. 11(b) explains this proximity: the per-step latency of STAR-PPO is 244 ms under the overlay model and 242 ms under the wireless model, in both cases dominated by computation with a negligible network component, whereas Greedy and Random reach 373 ms and 445 ms. The agreement between the two link models indicates that the formulation can be driven by either telemetry-like service links or a wireless-edge capacity model without altering the conclusions.

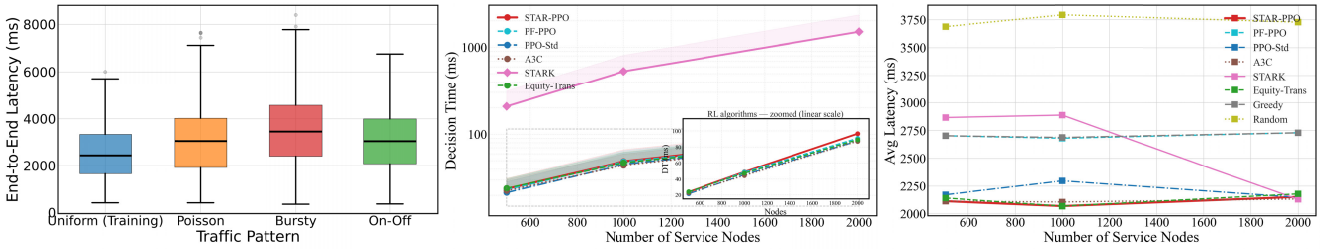
The shared-resource experiment constructs Region-A backhaul, cloud-egress, and cross-region WAN groups and enforces $\sum_{(i,j)\in\mathcal{E}_b} W_{ij} \le W_b^{\max}$ on each group. Fig. 12 traces the per-group utilization over time: when the uncapped demand exceeds the budget, the effective allocation is clipped at $W_b^{\max}$, producing 28, 37, and 28 congestion events in the three groups and 64 congested windows in total. Fig. 13 then reports the downstream effect of this contention. The latency panel in Fig. 13(a) shows that average latency rises from 2110.6 ms to 2153.0 ms, a 42.4 ms (2.0%) increase; the SLA-violation panel in Fig. 13(b) shows a rise from 18.29% to 23.07%; and the network-queueing panel in Fig. 13(c) attributes both effects to higher Q_{ij}^{net} on the contended links. Together, the two figures confirm that link contention is represented explicitly in the service-overlay model rather than left implicit.

Fig. 14 evaluates the Kingman-inspired estimator against the observed queue delay under three arrival processes. The scatter in Fig. 14(a) clusters around the perfect-prediction diagonal, with a coefficient of determination of $R^2 = 0.895$ for Poisson, 0.793 for bursty, and 0.786 for On–Off arrivals, so the estimate is most accurate under near-Poisson load and disperses as burstiness grows. The error distributions in Fig. 14(b) and (c) quantify the same trend: weighted over utilization bins, the mean absolute error is 117.63 ms, 330.67 ms, and 460.54 ms, and the weighted MAPE is 11.76%, 25.08%, and 26.77% for Poisson, bursty, and On–Off arrivals, respectively. This behavior matches the interpretation in Sec. III-D: the G/G/1-inspired term is a lightweight telemetry feature and estimator rather than an exact simulator of every LLM serving queue.

Fig. 15 validates the dynamic reliability term $p_n^{\text{rel}}(k)$. In the bursty Markov ON/OFF regime, the mean reliability drops to 0.8502, the average reliability-risk term increases to 0.1498, and 23 failure windows across four burst events produce an 11.5% SLA violation rate. By contrast, the no-failure and independent-random regimes do not produce SLA violations in this setting. This contrast shows that the dynamic term $p_n^{\text{rel}}(k)$ captures temporally correlated failures that a static reliability constant cannot represent.

Fig. 16 repeats the workflow/topology analysis in a controlled setting with explicit DAG lengths, separating latency, cost, and their joint shift. The latency panel in Fig. 16(a) shows that, for 3-step workflows, the full model attains 707.83 ms average latency, while removing workflow awareness or topology telemetry raises it to 763.1 ms and 763.0 ms, respectively, and the full model stays the lowest-latency option across the 2-, 3-, and 5-step settings. The cost panel in Fig. 16(b) preserves the same ordering, with the full model at 0.0184 USD/request for 3-step workflows. The latency–cost plane in Fig. 16(c) combines both effects and places the full model in the lower-left region. These results indicate that jointly modeling workflow dependencies and topology telemetry yields measurable gains over per-step service routing that ignores these structures.

Finally, Fig. 17 groups three consistency checks that probe whether the latency, cost, and decision-time behavior of STAR-PPO is preserved under workload shift and network growth. Fig. 17(a) reports latency robustness under unseen traffic patterns: relative to the uniform training distribution, average latency increases by 17.15% under Poisson traffic, 41.13% under bursty traffic, and 26.46% under On–Off traffic, with finite medians in all cases (2511.20 ms, 3228.76 ms, and 2879.91 ms, respectively), and a Kruskal–Wallis test over the four traffic groups gives $p = 2.21 \times 10^{-23}$, so the shift is statistically visible and the behavior is best read as robustness with measurable degradation rather than invariance. Fig. 17(b) reports decision time versus network size: for STAR-PPO the mean decision time grows from 23.7 ms at 500 nodes to



(a) Latency under unseen traffic patterns. (b) Decision time versus service-node count. (c) Latency versus service-node count.

Fig. 17. Robustness and scalability consistency: (a) latency distributions under unseen traffic patterns ($n = 600$ per group, whiskers following the 1.5IQR rule); (b) decision time versus service-node count; and (c) average latency versus service-node count.

96.4 ms at 2000 nodes, whereas STARK grows from 207.5 ms to 1497.9 ms over the same range. Fig. 17(c) reports the corresponding average latency, which stays within a narrow band as the network scales, while the heuristic and sequence-model baselines incur larger penalties. Taken together, the three panels indicate that the policy degrades gracefully and predictably under both a traffic-distribution shift and a fourfold increase in service-node count, with decision time remaining at the millisecond level rather than sub-millisecond.

VI. Conclusion

This paper studied stability-aware orchestration for workflow-level LLM serving over the edge-cloud continuum, where stochastic topology conditions and DAG-coupled inference pipelines jointly challenge low-latency, low-cost, and robust execution. We proposed STEM, a service-level expert-routing framework that formulates distributed inference as workflow-aware and topology-aware routing under reliability and cost constraints, and developed STAR-PPO as its decision engine. STAR-PPO replaces costly message-passing encoders with a lightweight graph-free perception design based on Squeeze-and-Excitation feature recalibration, and employs DWA to adapt scalarization preferences under non-stationary metric drift, enabling practical Pareto-efficient optimization. Extensive evaluations on real-world traces and OpenCelliD-derived large-scale topologies demonstrated that STEM achieves superior latency–cost trade-offs with linear-time inference and strong generalization. Future work will extend STEM toward richer failure and compliance models, online expert lifecycle management, and decentralized execution across multiple controllers while preserving end-to-end workflow guarantees.

References

[1] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "HuggingGPT: Solving AI tasks with ChatGPT and its friends in hugging face," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2023, pp. 38154–38180.

[2] Z. Shen et al., "EdgeLoRA: An efficient multi-tenant LLM serving system on edge devices," in *Proc. 23rd Annu. Int. Conf. Mobile Syst., Appl. Services*, Jun. 2025, pp. 138–153.

[3] P. Li, H. Dong, L. Qian, S. Zhou, and Y. Wu, "FlexGen: Efficient on-demand generative AI service with flexible diffusion model in mobile edge networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 11, no. 2, pp. 961–973, Apr. 2025.

[4] K. C. Cheekuri, "Unified gateway architecture for multi-tenant large language model serving," *J. Int. Crisis Risk Commun. Res.*, vol. 18, pp. 188–201, Nov. 2025.

[5] W. Feng et al., "Exploring collaborative diffusion model inferring for AIGC-enabled edge services," *IEEE Trans. Cognit. Commun. Netw.*, vol. 11, no. 2, pp. 946–960, Apr. 2025.

[6] Y. Yang et al., "Leveraging generative AI for security defense of mobile edge-enabled digital twins: A survey," *IEEE Trans. Cognit. Commun. Netw.*, vol. 12, pp. 7490–7521, 2026.

[7] Y. Wang et al., "BurstGPT: A real-world workload dataset to optimize LLM serving systems," in *Proc. 31st ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2025, pp. 5831–5841.

[8] B. Sun et al., "Llumnix: Dynamic scheduling for large language model serving," in *Proc. 18th USENIX Symp. Oper. Syst. Des. Implement. (OSDI)*, 2024, pp. 173–191.

[9] B. Gao et al., "Cost-efficient large language model serving for multi-turn conversations with cachedattention," in *Proc. USENIX Annu. Tech. Conf. (ATC)*, 2024, pp. 111–126.

[10] Y. Sheng et al., "Fairness in serving large language models," in *Proc. 18th USENIX Symp. Oper. Syst. Des. Implement. (OSDI)*, 2024, pp. 965–988.

[11] Z.-L. Chang, S. Hosseinalipour, M. Chiang, and C. G. Brinton, "Asynchronous multi-model dynamic federated learning over wireless networks: Theory, modeling, and optimization," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 5, pp. 1989–2004, Oct. 2024.

[12] M. Yang, D. Gao, C. Heng Foh, W. Quan, and V. C. M. Leung, "Multi-agent reinforcement learning-based joint caching and routing in heterogeneous networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 5, pp. 1959–1974, Oct. 2024.

[13] Y. Mei, Y. Zhuang, X. Miao, J. Yang, Z. Jia, and R. Vinayak, "Helix: Serving large language models over heterogeneous GPUs and network via max-flow," in *Proc. 30th ACM Int. Conf. Architectural Support Program. Lang. Operating Syst.*, vol. 1, Mar. 2025, pp. 586–602.

[14] A. A. Ismail, N. E. Khalifa, and R. A. El-Khoribi, "A survey on resource scheduling approaches in multi-access edge computing environment: A deep reinforcement learning study," *Cluster Comput.*, vol. 28, no. 3, p. 184, Jun. 2025.

[15] B. Li, Y. Jiang, V. Gadepally, and D. Tiwari, "LLM inference serving: Survey of recent advances and opportunities," in *Proc. IEEE High Perform. Extreme Comput. Conf. (HPEC)*, Sep. 2024, pp. 1–8.

[16] D. Wei, J. Wen, P. Xu, H. Shi, and C. Pan, "Dependent tasks joint scheduling and offloading for edge computing based on deep reinforcement learning," *Ad Hoc Netw.*, vol. 183, Mar. 2026, Art. no. 104111.

[17] Z. Zeng, Y. Miao, H. Gao, H. Zhang, and Z. Deng, "AdaMoE: Token-adaptive routing with null experts for mixture-of-experts language models," 2024, *arXiv:2406.13233*.

[18] L. Chen, M. Zaharia, and J. Zou, "FrugalGPT: How to use large language models while reducing cost and improving performance," 2023, *arXiv:2305.05176*.

[19] I. Ong et al., "RouteLLM: Learning to route LLMs with preference data," 2024, *arXiv:2406.18665*.

[20] J. Son, M. Kim, S. Choi, H. Kim, and J. Park, "Equity-transformer: Solving NP-hard min-max routing problems as sequential generation with equity context," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, 2024, pp. 20265–20273.

[21] E. Zhao, P. Awasthi, Z. Chen, S. Gollapudi, and D. Delling, "Semantic routing via autoregressive modeling," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 37, 2024, pp. 10060–10087.

[22] Y. Yang, H. Du, Z. Xiong, D. Niyato, A. Jamalipour, and Z. Han, "Enhancing wireless networks with attention mechanisms: Insights from mobile crowdsensing," *IEEE Wireless Commun.*, vol. 32, no. 5, pp. 152–161, Oct. 2025.

[23] A. Furutanpey, P. Raith, and S. Dustdar, "FrankenSplit: Efficient neural feature compression with shallow variational bottleneck injection for mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 10770–10786, Dec. 2024.
[24] X. Tang et al., "Workflow scheduling based on asynchronous advantage actor–critic algorithm in multi-cloud environment," *Expert Syst. Appl.*, vol. 258, Dec. 2024, Art. no. 125245.
[25] G. Sun, Y. Wang, H. Yu, and M. Guizani, "Proportional fairness-aware task scheduling in space-air-ground integrated networks," *IEEE Trans. Services Comput.*, vol. 17, no. 6, pp. 4125–4137, Nov. 2024.
[26] J. Hollenstein, G. Martius, and J. Piater, "Colored noise in PPO: Improved exploration and performance through correlated action sampling," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, 2024, pp. 12466–12472.
[27] Y. Yang, G. Chen, H. Ma, C. Zhang, Z. Cao, and M. Zhang, "Graph assisted offline-online deep reinforcement learning for dynamic workflow scheduling," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025, pp. 1–29.
[28] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10428–10437.
[29] Y. Yang, Y. Chen, J. Wang, G. Sun, and D. Niyato, "Embodied AI-empowered low altitude economy: Integrated sensing, communications, computation, and control (ISC3)," 2024, *arXiv:2412.19996*.
[30] F. Garcia and E. Rachelson, "*Markov decision processes*," in *Markov Decision Processes in Artificial Intelligence*. Hoboken, NJ, USA: Wiley, 2013, pp. 1–38.
[31] 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.
[32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

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.

Yonghui Ye received the B.Eng. degree in computer science and technology from Minnan Normal University, China. He is currently pursuing the M.S. degree with the School of Cyber Security and Computer, Hebei University, China. His research interests include edge intelligence, generative AI, large model serving and orchestration, and reinforcement learning.

Jie Fu is currently pursuing the joint Ph.D. degree with the School of New Media and Communication, Tianjin University, Tianjin, China, and China National Software and Service Company Ltd., Beijing, China. His research interests include distributed artificial intelligence, edge-cloud computing, large-scale model deployment, intelligent resource orchestration, and the development and practical application of enterprise-level intelligent software systems.

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 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 has received the ERC Starting Grant and the EPSRC NIA Award. 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.