Journal of Computer Technology and Software
ISSN:2998-2383
Vol. 4, No. 9, 2025

Joint Scheduling Method for Resource Orchestration Optimization in Distributed Microservices via Hierarchical Reinforcement Learning

Cheng-Chuan Peng[1*]
[1]Carnegie Mellon University, Mountain View, USA
[*]Corresponding author: Cheng-Chuan Peng; pengc01245@gmail.com

Abstract: With the continuous development of cloud-native platforms and service-oriented software architectures, distributed microservice systems are increasingly widely used in complex business scenarios. However, the increasing number of service instances, intertwined call relationships, and rapid fluctuations in resource states pose greater challenges to resource orchestration and joint scheduling. Addressing the limitations of traditional methods, such as insufficient complex dependency modeling, limited global optimization capabilities, and single-granularity decision-making, this paper proposes a joint scheduling method for distributed microservice resource orchestration optimization, organically integrating call topology-aware representation learning and hierarchical reinforcement learning into a unified framework. First, a topology-aware state representation is constructed around the dynamic call relationships between microservices, jointly encoding service node attributes, dependency relationships, and runtime resource information to enhance the model's ability to identify key call chains, local congestion areas, and structural coupling features. Second, a representation learning mechanism is used to extract service embeddings with structural semantics, enabling the scheduling process to obtain more stable and discriminative state representations in complex dependency contexts. Subsequently, a hierarchical reinforcement learning joint scheduling framework is constructed, completing global scheduling intent planning in high-level strategies and executing fine-grained resource allocation and action control in low-level strategies, thereby achieving effective connection between system-level goals and local execution behaviors. Furthermore, this paper designs a unified optimization objective around latency control, resource utilization, and quality of service constraints, enabling the joint scheduling process to balance response efficiency, resource coordination, and operational stability. Related results show that the proposed method can more effectively adapt to the scheduling requirements of complex call dependencies and dynamic resource competition in a distributed microservice environment, demonstrating strong advantages in comprehensive orchestration capabilities, service stability maintenance, and resource collaborative optimization. This method provides a new modeling approach for intelligent microservice resource management and offers a targeted solution to the joint scheduling optimization problem in complex cloud platforms.

Keywords: Microservice resource orchestration; call topology modeling; hierarchical strategy learning; joint scheduling optimization

1. Introduction

With the continuous evolution of cloud computing infrastructure, distributed microservice architecture has become a crucial technical paradigm supporting complex business systems[1,2]. Compared to traditional monolithic systems, microservices significantly improve system maintainability and flexibility through service decomposition, independent deployment, and elastic scaling. However, this also makes resource orchestration and task scheduling more complex. In large-scale operating environments, service instances commonly exhibit high-frequency interactions, chained dependencies, and dynamic coupling relationships. Heterogeneous resources such as computing, storage, and networks also exhibit strong dynamic and uncertain characteristics. How to achieve efficient, stable, and globally collaborative resource orchestration optimization under complex call relationships and multi-dimensional resource constraints has become a key scientific problem for improving the performance, resource utilization, and service quality of microservice systems[3].

Existing microservice scheduling research mostly focuses on resource load balancing, latency control, or single-step decision-making. While these methods alleviate local resource conflicts and performance degradation to some extent, they still fall short in characterizing the deep structural dependencies inherent in the service call topology. Service nodes in a microservice system are not isolated; their call paths, dependency strength, critical link locations, and local topological context directly affect task execution efficiency and overall system stability. Ignoring the structural semantics of the call graph and making scheduling decisions solely based on instantaneous resource states often leads to hotspot node clustering, link congestion propagation,

and global performance fluctuations[4]. Therefore, incorporating call topology information into the resource orchestration modeling process and extracting high-order correlation features between services through topology-aware representation learning has significant theoretical and practical value in enhancing the scheduling model's understanding of complex dependencies and improving the foresight and robustness of scheduling decisions.

Meanwhile, microservice resource orchestration is essentially a continuous decision-making process with long temporal dependencies, a combined action space, and multi-objective constraints. Relying solely on a flat, single-layer optimization strategy often fails to balance global objectives with local execution efficiency. Hierarchical reinforcement learning, by decomposing complex scheduling tasks into sub-decision processes at different levels, can model system-level optimization objectives at the macro level and refine resource allocation and instance scheduling behavior at the micro level, thus better adapting to the multi-stage, multi-granularity joint optimization needs in a distributed microservice environment[5]. Combining topology-aware representation learning with hierarchical reinforcement learning helps to construct a joint scheduling method that combines structural cognition and sequential decision-making capabilities. This not only provides new research ideas for microservice resource orchestration optimization, but also provides more solid methodological support for cloud platform autonomous management, intelligent operation and maintenance, and high-reliability service assurance.

2. Background

In recent years, with the rapid development of cloud-native technologies, containerized deployment models, and service-oriented software engineering, distributed microservice systems have been widely applied in various high-concurrency business scenarios such as e-commerce, smart healthcare, financial computing, and the industrial internet. These systems improve system flexibility, scalability, and continuous delivery capabilities by breaking down complex business processes into multiple independently developable, deployable, and scalable service units[6]. However, the continuous refinement of service granularity has also led to a more complex service dependency network within the system. Frequent remote calls, asynchronous communication, and multi-stage collaborations exist between different services, resulting in highly coupled spatial and temporal characteristics in computing resource requirements, communication overhead, and load fluctuations. Against this backdrop, resource orchestration is no longer a simple problem of instance placement or static allocation, but has gradually evolved into a comprehensive optimization task involving service association modeling, dynamic resource awareness, and global performance coordination.

Meanwhile, the modern cloud platform operating environment exhibits significant dynamism and heterogeneity. Different nodes experience continuous changes in computing power, network status, resource contention levels, and failure risks, further increasing the complexity of microservice scheduling modeling. Traditional orchestration strategies based on rules, heuristic search, or single-layer optimization can achieve basic scheduling results in some scenarios, but they often struggle to effectively characterize hierarchical dependencies and global linkages between services when facing large-scale service chains, multi-objective constraints, and long-term performance evolution[7]. Especially under complex business flow-driven conditions, locally optimal scheduling behavior can lead to resource contention and delayed delivery across service chains, thereby weakening the overall system stability and resource utilization efficiency. Therefore, research focusing on call topology modeling, structural feature representation, and multi-level intelligent decision-making has become a crucial foundation for driving distributed microservice resource orchestration from passive response to proactive optimization.

3. Methodology

In a distributed microservice environment, resource orchestration must capture not only instantaneous resource availability but also the structural dependency induced by service invocations, since isolated placement decisions often amplify latency propagation and contention accumulation along critical call chains. This paper also presents the overall model architecture, as shown in Figure 1.

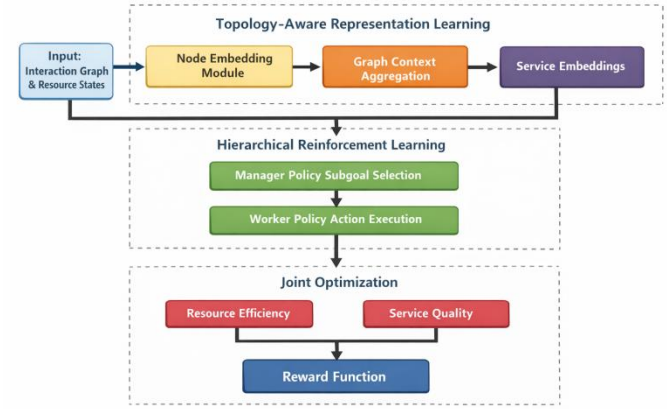

Figure 1. Overall model architecture

To preserve this system-level context, the microservice application is first modeled as a topology-aware interaction graph:

$$\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \boldsymbol{X}_t, \boldsymbol{A}_t)$$

in which $\mathcal{V}_t$ denotes the active service instances at time step t, $\mathcal{E}_t$ represents invocation relations, $\boldsymbol{X}_t$ aggregates node-level observations such as CPU utilization, memory pressure, queue length, and response delay, and $\boldsymbol{A}_t$ records directed dependency strengths between upstream and downstream services. Rather than treating invocation links as binary relations, the orchestration process assigns each edge a normalized structural importance so that dominant

paths receive stronger attention during representation propagation:

$$\alpha_{ij}^{(t)} = \frac{exp\left(\phi\left(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_j^{(t)}, a_{ij}^{(t)}\right)\right)}{\sum_{k\in\mathcal{N}(i)} exp\left(\phi\left(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_k^{(t)}, a_{ik}^{(t)}\right)\right)}$$

and this design allows the scheduler to distinguish latency-sensitive dependencies from weak background communications. By embedding topology intensity into the state construction stage, the method forms a richer description of coordinated service behavior, which is essential for preventing local decisions from disrupting globally important invocation routes.

Subsequently, topology-aware representation learning is introduced to compress heterogeneous runtime signals and call-graph dependencies into discriminative service embeddings that remain suitable for sequential decision making under rapidly varying workloads. Each service node updates its hidden state by integrating its own resource profile with the weighted messages received from dependency neighbors:

$$\boldsymbol{h}_i^{(t)} = \sigma\left(\boldsymbol{W}_s \boldsymbol{x}_i^{(t)} + \sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(t)} \boldsymbol{W}_n \boldsymbol{x}_j^{(t)}\right)$$

so that the learned representation reflects both local execution stress and upstream or downstream influence. Beyond dimensionality reduction, this transformation improves the semantic consistency of scheduler inputs because structurally related services are mapped into nearby regions of the latent space when they participate in the same invocation pattern. A graph-level orchestration context is then obtained by pooling all service embeddings together with cluster-level resource summaries, enabling the later policy modules to reason over congestion diffusion, bottleneck concentration, and coordination opportunities without manually engineered dependency rules.

A hierarchical reinforcement learning framework is further constructed because microservice orchestration naturally contains coupled decisions at different temporal and spatial granularities, and a flat policy is often too myopic to balance cluster-wide objectives against service-level execution constraints. At the upper level, a manager policy selects a subgoal g_t from the global topology state $\boldsymbol{z}_t$, with $\boldsymbol{z}_t$ denoting the aggregated graph representation, so that the scheduling process can first determine a coarse orchestration intention such as load rebalancing, latency suppression, or hotspot relief:

$$g_t \sim \pi_H\left(g|\boldsymbol{z}_t\right)$$

while a lower-level worker policy executes concrete actions conditioned on the selected subgoal and the current service embeddings. Such an action may correspond to instance scaling, service migration, request redistribution, or affinity-aware placement, and its selection is defined as:

$$a_t \sim \pi_L\left(a|\boldsymbol{H}_t, g_t\right)$$

with $\boldsymbol{H}_t = \{\boldsymbol{h}_1^{(t)}, \boldsymbol{h}_2^{(t)}, \ldots, \boldsymbol{h}_{|\mathcal{V}_t|}^{(t)}\}$ collecting node representations for all active services. Since the manager captures long-horizon coordination intent and the worker focuses on executable orchestration primitives, the overall framework reduces the difficulty of searching over a large combinatorial action space and improves the interpretability of scheduling behavior.

Finally, the optimization objective couples service quality preservation with resource efficiency so that the learned policy does not overfit a single operational target at the expense of system robustness. A unified reward is formulated from latency, resource imbalance, scaling cost, and violation penalties:

$$r_t = -\lambda_1\mathcal{C}_t - \lambda_2\mathcal{B}_t - \lambda_3\mathcal{C}_t - \lambda_4\mathcal{P}_t$$

where $\mathcal{C}_t$ measures end-to-end service delay, $\mathcal{B}_t$ reflects cluster-level imbalance, $\mathcal{C}_t$ characterizes orchestration overhead, and $\mathcal{P}_t$ penalizes violations of resource or dependency constraints, while the nonnegative coefficients $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ regulate the relative priority of these objectives. From this formulation, the scheduler is encouraged to suppress cascading congestion and avoid unnecessary adjustments rather than greedily pursuing transient resource utilization gains. Through the joint interaction between topology-aware representation learning and hierarchical policy decomposition, the proposed method establishes a coherent optimization pathway for distributed microservice resource orchestration, allowing structural dependency perception and multi-level decision intelligence to reinforce each other throughout the scheduling process.

4. Experimental Results and Analysis

4.1 Dataset Introduction

This paper uses the open-source Train Ticket microservice tracing dataset as the foundation for experimental data. This dataset originates from the open-source Train Ticket benchmark system, whose core business is train ticket booking. It contains 41 microservices, capable of forming complex service call chains and hierarchical dependencies, and supports deployment in a Kubernetes environment. Therefore, it is highly compatible with distributed microservice resource orchestration, call topology modeling, and joint scheduling optimization tasks. Furthermore, the publicly released preprocessed tracing dataset extracts distributed tracing records for the reserve-tickets request type from actual operation. The data is organized in CSV format, with each sample containing a tracing identifier and the duration information of each microservice component. It also provides execution paths to characterize the call path, enabling it to simultaneously support service dependency graph construction, topology-aware representation learning, and time-series decision-oriented scheduling state modeling.

4.2 Experimental Results and Analysis

To ensure that the comparative analysis is consistent with the distributed microservice resource orchestration optimization problem studied in this paper, representative

studies focusing on microservice scheduling, resource allocation, elastic scaling, call relationship modeling and reinforcement learning optimization were selected. The experimental results are summarized in Table 1, focusing on latency control, tail service quality, resource utilization and service-level target breach.

Table 1. Experimental results compared with other models

Method	AL	P99	RU	SVR
Park et al.[8]	128.46	231.88	72.14	4.93
Jian et al.[9]	121.37	219.54	74.02	4.41
Li et al.[10]	117.82	210.63	75.26	4.08
Cao et al.[11]	114.59	203.17	76.84	3.87
Hu et al.[12]	109.74	192.46	78.35	3.42
Cai et al.[13]	112.68	198.35	77.19	3.65
Liang et al.[14]	106.93	186.71	79.42	3.18
Ours	98.27	171.35	82.64	2.41

Table 1 uses four metrics to comprehensively characterize the distributed microservice resource orchestration method. AL measures the average latency level of the system during the overall request processing, reflecting the impact of the scheduling strategy on service response efficiency. P99 characterizes the latency performance of tail requests under high load or complex call scenarios, which is crucial for evaluating the stability of critical links and service quality under extreme conditions. RU represents resource utilization level, mainly reflecting the sufficiency and coordination of computing resources during orchestration. SVR describes the service-level target breach situation; the lower this metric, the better the scheduling strategy can maintain service quality constraints under complex operating conditions. These metrics evaluate the method's performance from aspects such as latency control, tail quality assurance, resource collaborative utilization, and service stability maintenance, comprehensively reflecting the actual effectiveness of the microservice resource orchestration algorithm.

Overall, the proposed method demonstrates strong advantages across all evaluation dimensions, indicating that it can more effectively adapt to the scheduling needs of complex dependencies and dynamic resource states in a distributed microservice environment. On the one hand, the constructed call topology-aware representation learning mechanism enhances the model's ability to identify structural relationships between services, the impact of critical paths, and the characteristics of local congestion propagation, thereby helping to improve the targeting of resource allocation and task orchestration. On the other hand, the hierarchical reinforcement learning framework organically connects global scheduling objectives with local execution decisions, enabling the resource regulation process to maintain overall synergy while taking into account fine-grained execution efficiency. Therefore, this method not only better improves resource utilization but also maintains superior scheduling stability under quality-of-service constraints, demonstrating high comprehensive orchestration capabilities and practical application value.

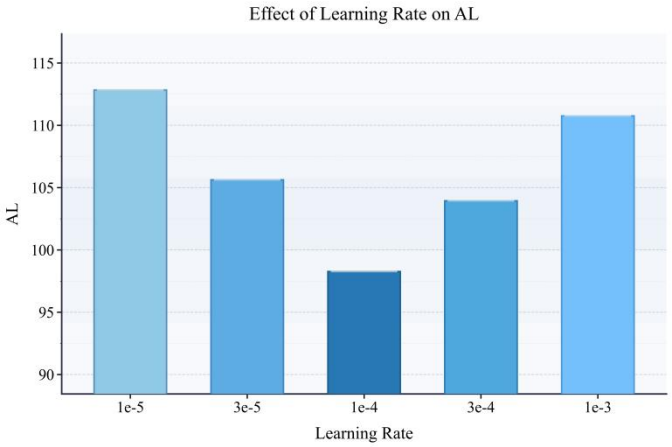

Figure 2. The impact of learning rate on AL

The learning rate directly affects the optimization quality of the proposed method, as shown in Figure 2. A reasonable parameter update step size can more fully utilize the advantages of joint modeling using topology-aware representation learning and hierarchical reinforcement learning, enabling the model to maintain excellent scheduling capabilities under complex microservice dependencies and dynamic resource states. By setting an appropriate learning rate, the proposed algorithm can more effectively coordinate the relationship between global orchestration goals and local execution decisions, thereby achieving more ideal performance in latency control. This shows that the proposed method not only has good structural modeling capabilities but also strong training adaptability and optimization stability, providing reliable methodological support for distributed microservice resource orchestration tasks.

Furthermore, the optimizer determines the direction of parameter updates and the gradient scaling mechanism, thus directly affecting the convergence behavior of the joint scheduling model in complex state spaces. For resource orchestration methods that integrate topology-aware representation learning and hierarchical reinforcement learning, different optimizers alter the model's adaptation to service dependency structures, resource contention relationships, and long-term scheduling goals. Therefore, it is necessary to analyze the optimizer settings independently to characterize the training stability and optimization adaptability of the proposed method under different parameter update mechanisms.

As shown in Figure 3, the optimizer settings directly affect the training state and scheduling performance of the proposed method. A suitable parameter update mechanism can more fully leverage the synergistic effect of topology-aware representation learning and hierarchical reinforcement learning, enabling the model to maintain stronger optimization capabilities in complex service dependencies and dynamic resource environments. With a reasonable optimizer configuration, the proposed algorithm can more effectively coordinate the relationship between global resource orchestration objectives and local action

decisions, thereby achieving more ideal latency control results. This further demonstrates that the proposed method not only possesses good structural modeling and decision-making adaptation capabilities but also exhibits strong training stability and practical deployment potential.

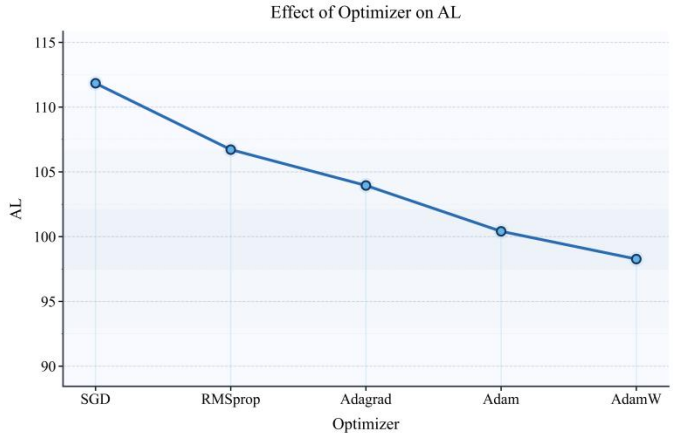

Figure 3. The impact of optimizers on AL results

5. Conclusion

This paper addresses the challenges of complex resource orchestration, tight call dependencies, and difficult scheduling decisions in distributed microservice environments. It proposes a joint scheduling method combining call topology-aware representation learning and hierarchical reinforcement learning. To overcome the limitations of traditional resource orchestration methods in fully characterizing inter-service structural relationships and balancing global goals with local execution behavior, the proposed method starts from the microservice call topology and unifies the modeling of service dependencies, resource state information, and the scheduling decision process. This allows resource orchestration to move beyond isolated node perspectives and achieve synergistic advancement of structural awareness and strategy optimization at a more complete system level. By incorporating key link relationships, dynamic resource constraints, and long-term scheduling goals into a unified framework, the proposed method enhances the model's understanding of potential bottleneck propagation, service coupling effects, and multi-stage scheduling requirements in complex microservice systems. This provides a new approach to improving resource allocation efficiency and service quality assurance in distributed environments.

From a methodological perspective, call topology-aware representation learning strengthens the model's ability to represent high-order dependencies within microservice systems, enabling the resource orchestration process to more accurately identify key service nodes, important call paths, and the impact of local congestion on the overall system. Meanwhile, hierarchical reinforcement learning, by decomposing complex scheduling tasks into multiple levels, effectively connects system-level optimization objectives with fine-grained execution actions. This enables the joint scheduling framework to maintain strong adaptability and decision-making coordination capabilities even in complex scenarios such as dynamic load changes, intensified resource competition, and enhanced service quality constraints. This research not only provides a unified modeling approach for distributed microservice resource orchestration problems that combines structural cognition and intelligent decision-making features, but also offers a theoretical foundation and methodological support with practical reference value for related application areas such as cloud platform autonomous management, intelligent operation and maintenance, service-level target assurance, edge cloud collaborative scheduling, and highly reliable online business support. For large-scale service systems that require long-term stable operation and have highly complex call relationships, the proposed method has significant application value in improving system resilience, reducing resource waste, and enhancing service continuity.

Future research can further expand the applicability of this framework in microservice environments with larger scale, higher dynamism, and greater heterogeneity. On the one hand, by combining more granular service behavior modeling and online state awareness mechanisms, the method's adaptability to complex operating conditions such as sudden loads, link anomalies, and resource jitter can be further enhanced, enabling scheduling strategies to have stronger real-time response capabilities and environmental generalization capabilities. On the other hand, in-depth research can be conducted on cross-cluster collaboration, edge-cloud integrated deployment, multi-tenant resource competition, and joint optimization under multi-objective constraints, to promote resource orchestration methods from single-scenario optimization to general intelligent scheduling for complex business ecosystems. With the continuous development of cloud-native technology systems, the scale of distributed applications, and the demand for autonomous operation and maintenance, research on resource orchestration oriented towards call topology modeling and hierarchical decision-making collaborative optimization is expected to play a more profound role in fields such as smart industry, online trading platforms, real-time recommendation systems, vehicle networking services, and the digital operation of critical infrastructure, and provide continuous support for building a highly efficient, highly reliable, and highly elastic next-generation microservice system.

References

[1] S. Luo, C. Lin, K. Ye, et al., "Optimizing resource management for shared microservices: A scalable system design," ACM Transactions on Computer Systems, vol. 42, no. 1-2, pp. 1-28, 2024.

[2] L. Chen, S. Luo, C. Lin, et al., "Derm: Sla-aware resource management for highly dynamic microservices," in ACM/IEEE Annual International Symposium on Computer Architecture, 2024, pp. 424-436.

[3] L. Chen, Y. Bai, P. Zhou, et al., "On adaptive edge microservice placement: A reinforcement learning approach endowed with graph comprehension," IEEE Transactions on Mobile Computing, vol. 23, no. 12, pp. 11144-11158, 2024.

[4] S. Khan and S. Khan, "Latency aware graph-based microservice placement in the edge-cloud continuum," Cluster Computing, vol. 28, no. 2, p. 88, 2025.

[5] Q. Si, J. Shi, W. Li, et al., "DeepMRA: An Efficient Microservices Resource Allocation Framework with Deep Reinforcement Learning in the

Cloud," in International Conference on Intelligent Computing, Singapore: Springer Nature Singapore, 2024, pp. 455-466.

[6] S. Long, J. Yang, C. Rao, et al., "D3QN-based secure scheduling of microservice workflows in cloud environments," Computer Networks, vol. 263, p. 111227, 2025.

[7] A. Kallel, M. Rekik, and M. Khemakhem, "A deep reinforcement learning-based optimization approach for containerized microservice scheduling in Hybrid Fog/Cloud environments," Engineering Applications of Artificial Intelligence, vol. 141, p. 109745, 2025.

[8] J. Park, B. Choi, C. Lee, et al., "GRAF: A graph neural network based proactive resource allocation framework for SLO-oriented microservices," in Proceedings of the 17th International Conference on Emerging Networking Experiments and Technologies, 2021, pp. 154-167.

[9] Z. Jian, X. Xie, Y. Fang, et al., "DRS: A deep reinforcement learning enhanced Kubernetes scheduler for microservice-based system," Software: Practice and Experience, vol. 54, no. 10, pp. 2102-2126, 2024.

[10] X. Li, J. Zhou, X. Wei, et al., "Topology-aware scheduling framework for microservice applications in cloud," IEEE Transactions on Parallel and Distributed Systems, vol. 34, no. 5, pp. 1635-1649, 2023.

[11] H. Cao, X. Liu, H. Guo, et al., "QueueFlower: Orchestrating Microservice Workflows via Dynamic Queue Balancing," in IEEE International Conference on Web Services, 2024, pp. 1293-1299.

[12] Y. Hu, L. Hou, J. Hu, et al., "Time-varying microservice orchestration with routing for dynamic call graphs via multi-scale deep reinforcement learning," IEEE Transactions on Services Computing, vol. 18, no. 5, pp. 3276-3291, 2025.

[13] B. Cai, X. Wang, B. Wang, et al., "A self-stabilizing and auto-provisioning orchestration for microservices in edge-cloud continuum," Computer Networks, vol. 242, p. 110279, 2024.

[14]G. Carpio, A. Jukan, and R. Pries, "Gwydion: Efficient auto-scaling for complex containerized applications in Kubernetes through Reinforcement Learning," Journal of Network and Computer Applications, vol. 234, p. 104067, 2025, DOI: 10.1016/j.jnca.2024.104067.