Transactions on Computational and Scientific Methods | Vo. 4, No. 3, 2024
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

# HeterSched: Structure-Aware Reinforcement Learning for Heterogeneous Distributed Task Scheduling

**Jiajing Liao**
University of Florida, Gainesville, USA
jjliao15@gmail.com

**Abstract:** To address the scheduling challenges posed by diverse task types, significant resource state fluctuations, and complex task dependencies in dynamic heterogeneous distributed computing environments, this paper proposes an intelligent task scheduling method oriented towards structure awareness and long-term decision optimization. This method models the distributed scheduling process as a unified sequential decision problem. By constructing a heterogeneous graph representation involving both task nodes and resource nodes, it incorporates pre-task and post-task constraints, node differences, and interactions between tasks and resources into a single modeling framework. Building upon this, a graph neural network is used to deeply represent the system state, enhancing the model's understanding of complex topological relationships and dynamic runtime contexts. Furthermore, deep reinforcement learning is integrated to learn scheduling strategies, enabling the model to formulate more rational task allocation schemes from a global benefit perspective. Simultaneously, to improve the adaptation quality between tasks and resources, a task-resource matching module is designed to characterize the correlation strength between executable tasks and candidate computing nodes, and a joint objective optimization approach coordinates the structure representation learning and strategy optimization processes. Related research shows that the proposed method effectively improves the state representation capability, decision consistency, and resource allocation rationality in the scheduling process, providing a unified solution with structural modeling and adaptive decision-making capabilities for intelligent scheduling in complex distributed scenarios.

**Keywords:** Distributed task scheduling; graph neural networks; deep reinforcement learning; heterogeneous resource allocation

## 1. Introduction

With the rapid development of cloud computing, edge computing, and high-performance computing, distributed computing environments have become the core infrastructure supporting large-scale data processing, intelligent service deployment, and complex task execution[1,2]. In real-world business scenarios, distributed systems often handle a large number of concurrent tasks simultaneously, exhibiting significant differences in node types, computing capabilities, communication overhead, and resource constraints, resulting in dynamic, heterogeneous, and coupled characteristics. Against this backdrop, how to achieve efficient, reasonable, and stable scheduling based on task characteristics and resource status has become a key issue affecting the overall system throughput, resource utilization efficiency, and service quality. Task scheduling is no longer just a simple resource allocation process but an important research direction related to optimizing computing performance, ensuring system elasticity, and improving intelligent operation capabilities[3].

Traditional task scheduling methods typically rely on manually designed rules, heuristic strategies, or static optimization mechanisms. While these methods can achieve certain results in specific scenarios, they often suffer from insufficient adaptability, limited generalization ability, and weak long-term optimization capabilities when facing frequently changing task arrival patterns, resource competition relationships, and topological dependencies in dynamic and heterogeneous distributed environments. Especially when complex tasks involve dependencies, priority differences, and resource coupling constraints, scheduling decisions must consider not only current local gains but also future execution paths and the overall operational state. This high-dimensional, dynamic, and nonlinear decision-making characteristic makes distributed task scheduling increasingly resemble a complex intelligent decision-making problem, placing higher demands on the state modeling and policy learning capabilities of scheduling methods[4].

The development of artificial intelligence technology provides new theoretical support and methodological pathways for solving these problems. Graph neural networks can effectively characterize the complex relationships between tasks, nodes, and resources, making them suitable for handling the widespread topological structures, dependency structures, and heterogeneous interaction information in distributed systems. Deep reinforcement learning, on the other hand, can learn long-term benefit-oriented decision-making strategies through continuous interaction with the environment, achieving adaptive optimization in a high-dimensional state space and possessing the online decision-making and policy iteration capabilities required for dynamic scheduling problems. Synergistically integrating the structural representation advantages of graph neural networks with the sequential decision-making advantages of deep reinforcement learning is expected to overcome the limitations of traditional methods in complex relationship modeling and long-term global optimization, providing a more intelligent, flexible, and efficient solution for task scheduling in dynamic heterogeneous distributed environments[5].

Research on this issue has both significant theoretical value and substantial practical implications. From a theoretical perspective, this research helps to promote the deep integration of distributed system optimization problems and intelligent decision-making methods, expands the application boundaries of graph structure learning and reinforcement learning in the regulation of complex systems, and provides a new research paradigm for the design of dynamic resource management and intelligent scheduling mechanisms. From an application perspective, building efficient task scheduling methods for dynamic heterogeneous distributed computing environments helps to improve resource utilization, shorten task completion time, reduce system operating costs, and enhance the robustness and adaptability of the system under complex load conditions. This has a crucial practical driving force for supporting the high-quality operation of future cloud-edge-device collaborative computing, intelligent computing networks, the Industrial Internet, and large-scale intelligent service platforms.

## 2. Related work

Distributed computing is a computing model that breaks down computational tasks and distributes them collaboratively among multiple computing nodes. Its core objective is to improve system processing power and service efficiency through resource sharing and parallel execution. With the continuous growth in computing power demands, modern distributed environments have evolved from relatively simple cluster structures to complex computing systems comprised of cloud platforms, edge nodes, heterogeneous acceleration devices, and network infrastructure. In this system, different nodes exhibit significant differences in processor type, storage capacity, network bandwidth, energy consumption levels, and workloads[6]. Tasks themselves often have different scales, priorities, dependencies, and latency constraints. Therefore, the resource organization and task execution processes in distributed computing environments exhibit highly complex structural characteristics. Traditional management methods based on static assumptions are insufficient to accurately describe their operational patterns, necessitating in-depth research from two levels: system correlation modeling and dynamic decision-making mechanisms.

Task scheduling, as a crucial link in distributed systems, is essentially a process of jointly coordinating task execution order, resource mapping relationships, and system operating status. Its research not only involves computing resource management issues but is also closely related to system modeling, optimization theory, and intelligent decision-making. In dynamic and heterogeneous scenarios, schedulers need to continuously perceive changes in task queues, fluctuations in node states, and structural relationships between tasks to make reasonable decisions under complex constraints[7]. Since task dependencies and resource interactions naturally exhibit graph-like structures, and the scheduling process involves continuous decision-making and long-term benefit accumulation, recent research has gradually shifted from rule-driven and experience-driven approaches to data-driven and learning-driven approaches. Constructing scheduling models for complex environments based on structural representation learning and intelligent optimization methods has become an important development direction in the field of distributed computing, providing a new theoretical foundation for improving the system's autonomous perception, analysis, and optimization capabilities.

## 3. Method

### 3.1 Overall Framework

This study formulates task scheduling in a dynamic heterogeneous distributed environment as a structure-aware sequential decision problem, where the scheduler must jointly consider task dependencies, resource heterogeneity, and temporal state variation rather than making isolated local assignments. A unified representation is introduced to preserve the interaction among waiting tasks, running tasks, and computing nodes, because the quality of a scheduling policy depends not only on the current executable set but also on the latent coupling between resource contention and downstream execution paths. To capture this property, the system state at decision step $t$ is modeled as a heterogeneous graph with attributed vertices and typed relations, so that both topological constraints and operational context can be embedded into a shared latent space.

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

Here, $\mathcal{V}_t$ denotes the vertex set composed of task vertices and resource vertices, $\mathcal{E}_t$ denotes dependency edges and allocation-related edges, $\boldsymbol{X}_t$ contains the dynamic attributes of vertices, and $\boldsymbol{R}_t$ records relation types that distinguish different interaction patterns. Such a formulation is essential because the scheduler must infer whether assigning a task to a node is beneficial under both current load and future critical-path evolution. Instead of relying on hand-crafted priority rules, the proposed method learns a compact state embedding through graph propagation, which allows structurally correlated entities to exchange information during message passing.

$$\boldsymbol{H}_t^{(l+1)} = \sigma\left(\sum_{r\in\boldsymbol{R}_t} \widetilde{\boldsymbol{A}}_t^r \boldsymbol{H}_t^{(l)} \boldsymbol{W}_r^{(l)}\right)$$

In this expression, $\boldsymbol{H}_t^{(l)}$ is the hidden representation at layer $l$, $\widetilde{\boldsymbol{A}}_t^r$ is the normalized adjacency matrix under relation type $r$, and $\boldsymbol{W}_r^{(l)}$ is the learnable transformation associated with that relation. By preserving relational semantics during feature aggregation, the encoder can distinguish precedence constraints from resource affinity and therefore improve the quality of state abstraction before policy optimization. Once graph propagation is completed, node-level features are integrated into a global decision representation that summarizes the scheduling context at the current step.

$$\boldsymbol{s}_t = READOUT\left(\boldsymbol{H}_t^{(L)}\right)$$

This global state vector $\mathbf{s}_t$ serves as the semantic interface between structured environment modeling and reinforcement learning, enabling the downstream policy to reason over a compressed yet topology-preserving description of the system. The significance of this design lies in reducing the mismatch between graph-structured workload dynamics and the flat state assumptions commonly used in conventional learning-based schedulers. This paper presents the overall model architecture, as shown in Figure 1.

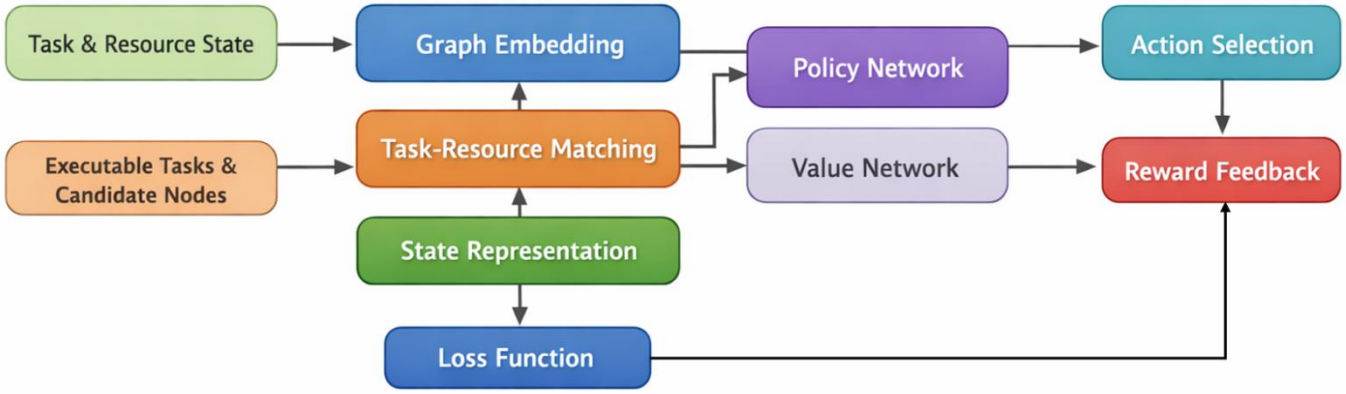


**Figure 1.** Overall model architecture

### 3.2 Task Resource Coupling Representation

Beyond global abstraction, the method further emphasizes fine-grained compatibility modeling between executable tasks and available resources, because a good scheduler should assign tasks to nodes whose computation profile, communication condition, and residual capacity jointly support efficient execution. Let $\mathcal{Q}_t$ denote the executable task set at step $t$, and let $\mathcal{M}_t$ denote the candidate resource set. For each task node $i \in \mathcal{Q}_t$ and resource node $j \in \mathcal{M}_t$, a pairwise matching score is constructed to quantify the suitability of allocating task $i$ to resource $j$ under the current system state.

$$e_{ij}^t = \boldsymbol{a}^\top tanh\left(\boldsymbol{W}_q \boldsymbol{h}_{i,t}^q + \boldsymbol{W}_m \boldsymbol{h}_{j,t}^m + \boldsymbol{W}_s \boldsymbol{s}_t\right)$$

The vectors $\boldsymbol{h}_{i,t}^q$ and $\boldsymbol{h}_{j,t}^m$ denote the encoded features of task $i$ and resource $j$, respectively, while $\boldsymbol{s}_t$ injects the global context so that local matching is not performed in isolation. Because direct score comparison may be unstable when the candidate pool changes over time, a normalized allocation preference is obtained through a soft selection mechanism.

$$\alpha_{ij}^t = \frac{\exp\left(e_{ij}^t\right)}{\sum_{k \in \mathcal{M}_t} \exp\left(e_{ik}^t\right)}$$

Such a normalization makes the decision space adaptive to dynamic resource availability and provides a differentiable estimate of the relative assignment quality for each executable task. In practical terms, this step allows the model to balance heterogeneous factors such as processing speed, queue occupancy, communication latency, and task urgency in a unified matching function, thereby reducing the risk of myopic scheduling behavior caused by single-factor prioritization.

### 3.3 Reinforcement Learning Based Scheduling Policy

After structural encoding and task resource matching are established, scheduling is optimized through a deep reinforcement learning policy that selects the action with the highest long-term utility rather than the action with the largest immediate gain. This choice is important because early assignments often affect later waiting time, communication overhead, and critical-path delay, meaning that the consequence of a current decision may only become visible several steps afterward. At each step, the action $a_t$ corresponds to assigning one executable task to one feasible resource, and the policy distribution is generated from the shared state embedding and the candidate-specific matching features.

$$\pi_\theta\left(a_t|\boldsymbol{s}_t\right) = softmax\left(f_\theta\left(\boldsymbol{s}_t, \alpha_{ij}^t\right)\right)$$

Since policy learning should prefer decisions that shorten completion time, alleviate congestion, and preserve efficient execution order, the reward is defined as a composite signal reflecting both instantaneous scheduling quality and downstream execution benefit. A compact reward form is written as follows.

$$r_t = -\lambda_1 T_t^{wait} - \lambda_2 T_t^{comm} - \lambda_3 T_t^{finish} + \lambda_4 U_t$$

In this reward, $T_t^{\text{wait}}$ denotes queueing delay, $T_t^{\text{comm}}$ denotes communication overhead, $T_t^{\text{finish}}$ denotes task completion-related cost, and $U_t$ measures resource utilization quality, while $\lambda_1$ to $\lambda_4$ control the relative influence of these objectives. The value function is introduced to estimate the expected cumulative return from the current state, which stabilizes training and improves policy evaluation in nonstationary environments.

$$V_\phi\left(\boldsymbol{s}_t\right) = \mathbb{E}_{\pi_\theta}\left[\sum_{k=t}^{T} \gamma^{k-t} r_k|\boldsymbol{s}_t\right]$$

By coupling policy optimization with value estimation, the scheduler can gradually learn decisions that remain effective under fluctuating workload intensity and heterogeneous resource conditions, rather than overfitting to short-term allocation patterns.

### 3.4 Optimization Objective and Scheduling Significance

To make the learned scheduler simultaneously structure-aware and decision-efficient, the final objective integrates policy optimization with graph-based state representation learning, so that the encoder is updated toward features that are most useful for long-horizon scheduling. The policy parameters are optimized by maximizing the expected discounted return over scheduling trajectories.

$$\mathcal{L}_{RL}(\theta) = -\mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} \gamma^{t-1} r_t\right]$$

Meanwhile, representation learning contributes to scheduling quality by preserving discriminative relational patterns among tasks and resources, which helps the policy distinguish bottleneck-sensitive assignments from less critical ones. The overall training objective, therefore, combines the reinforcement learning term with a representation regularization term, yielding a joint optimization target that aligns structural modeling with scheduling performance.

$$\mathcal{L} = \mathcal{L}_{\text{RL}} + \beta \mathcal{L}_{\text{reg}}$$

The coefficient $\beta$ balances decision-oriented optimization and representation regularity, allowing the model to maintain stable graph embeddings while still prioritizing scheduling effectiveness. From a methodological perspective, this joint design gives the framework two important advantages: first, it turns the raw distributed environment into a learnable relational system rather than a collection of disconnected metrics; second, it enables the scheduler to move beyond handcrafted heuristics and acquire adaptive decision rules directly from state transition feedback. Consequently, the proposed method provides a principled way to address task scheduling in dynamic heterogeneous distributed computing environments, where structural dependency, temporal variation, and resource diversity must be handled in a unified manner.

## 4. Experimental Results and Analysis

### 4.1 Dataset

This paper selects Alibaba Cluster Trace v2018 as the dataset. This dataset consists of cluster operation trajectories from a real production environment, publicly released for academic research, and can effectively reflect the operational status of modern data centers under dynamic load, heterogeneous resources, and multi-task concurrency conditions. According to the official description, cluster-trace-v2018 covers approximately 4000 machines, lasts for about 8 days, and consists of 6 core tables. It also includes DAG dependency information for production batch processing workloads, enabling it to not only describe task arrival, execution, and resource consumption processes, but also support modeling of task predecessor constraints and scheduling order. Therefore, it is highly compatible with the research topic of task scheduling in dynamic heterogeneous distributed computing environments.

Compared to data that only contains resource usage statistics or single job logs, Alibaba Cluster Trace v2018 records multi-level operational information, including machines, containers, online services, and offline tasks, providing a more complete state foundation for graph structure modeling and reinforcement learning decision-making. On the one hand, the dependencies between tasks, the mapping between tasks and nodes, and the changes in node load can all be used to construct heterogeneous graph representations. On the other hand, task queuing, resource contention, and execution evolution processes can also provide continuous decision-making cues for learning scheduling strategies. Therefore, this dataset is both open-source and highly representative of real-world scenarios, making it suitable as a data foundation for research on task scheduling optimization in dynamic heterogeneous distributed computing environments.

### 4.2 Experimental setup

The experiments were conducted on the Linux operating system. The overall development environment adopted Python and PyTorch deep learning frameworks to support the unified implementation of graph neural network encoding, policy network training, and task scheduling decision-making processes. Considering that the proposed method needs to simultaneously handle graph structure state representation, task resource matching, and reinforcement learning policy optimization, a GPU with strong parallel computing capabilities was used during the training phase to improve model iteration efficiency, while data preprocessing and scheduling environment simulation were mainly performed by the CPU. To ensure the stability and reproducibility of the experimental process, key parameters such as learning rate, discount factor, batch size, optimizer, number of graph encoding layers, and hidden layer dimension were all fixed. The specific hardware and software environment and core hyperparameter settings are shown in Table 1.

**Table 1:** Experimental Environment and Core Hyperparameter Settings

| Category | Item | Setting |
|---|---|---|
| Hardware Environment | CPU | Intel Xeon Silver 4314 |
| Hardware Environment | GPU | NVIDIA RTX 4090 24GB |

| Category | Item | Setting |
|---|---|---|
| Hardware Environment | Memory | 128 GB |
| Software Environment | Operating System | Ubuntu 22.04 |
| Software Environment | Python Version | Python 3.10 |
| Software Environment | Deep Learning Framework | PyTorch 2.1 |
| Software Environment | CUDA Version | CUDA 12.1 |
| Software Environment | Graph Learning Library | PyTorch Geometric 2.4 |
| Training Parameters | Optimizer | Adam |
| Training Parameters | Initial Learning Rate | 0.0001 |
| Training Parameters | Weight Decay | 0.00001 |
| Training Parameters | Batch Size | 32 |
| Training Parameters | Number of Epochs | 100 |
| Reinforcement Learning Parameters | Discount Factor | 0.99 |
| Reinforcement Learning Parameters | Replay Buffer Capacity | 100000 |
| Reinforcement Learning Parameters | Target Network Update Interval | 500 steps |
| Model Parameters | Graph Encoder Layers | 3 |
| Model Parameters | Hidden Dimension | 128 |
| Model Parameters | Activation Function | ReLU |
| Model Parameters | Dropout | 0.2 |

### 4.3 Experimental Results and Analysis

To further demonstrate the differences between our proposed method and existing research in dynamic heterogeneous distributed task scheduling, we selected representative methods closely related to cloud computing task scheduling, workflow scheduling, and deep reinforcement learning optimization as comparison objects, which are highly comparable in terms of task completion efficiency, resource utilization, and overall scheduling quality. Based on a unified evaluation dimension, Table 2 presents a comparison of different methods on key indicators, and our proposed method is comprehensively analyzed within the same framework.

**Table 2:** Experimental results compared with other models

| Method | MS | WT | RU | TSR |
|---|---|---|---|---|
| Dong et al.[8] | 145.82 | 23.47 | 82.31 | 91.28 |
| Sun et al.[9] | 139.64 | 21.95 | 84.06 | 92.14 |

| | | | | |
|---|---|---|---|---|
| Wang et al.[10] | 136.27 | 20.88 | 84.91 | 92.76 |
| Cheng et al.[11] | 132.58 | 19.46 | 86.37 | 93.88 |
| Gu et al.[12] | 129.94 | 18.72 | 87.45 | 94.21 |
| Pan et al.[13] | 127.63 | 17.98 | 88.16 | 94.85 |
| Jayanetti et al.[14] | 126.48 | 17.35 | 88.92 | 95.13 |
| Ours | 121.37 | 15.84 | 91.46 | 96.72 |

The evaluation metrics in the table characterize the comprehensive performance of the distributed task scheduling method from different perspectives. MS primarily reflects the overall scheduling overhead and execution efficiency during task completion, demonstrating the scheduling strategy's control over the overall completion quality. WT measures the time cost incurred by a task during the waiting phase; a better metric indicates a faster response to task demands and reduced resource idleness. RU characterizes the sufficiency of system resources, reflecting the matching level between task allocation and node states. TSR primarily reflects the scheduling's ability to complete tasks, comprehensively reflecting the algorithm's execution stability and scheduling reliability in complex environments. Therefore, these four metrics together constitute a relatively comprehensive evaluation basis for the scheduling method's performance, considering both efficiency and overhead, as well as resource organization quality and task execution effectiveness.

The overall results show that the proposed algorithm exhibits strong advantages across all evaluation metrics, demonstrating that this method can achieve more reasonable task allocation and execution control in dynamic heterogeneous distributed environments. On one hand, graph structure modeling enhances the algorithm's ability to characterize task dependencies and resource interactions, enabling scheduling decisions to move beyond local states and optimize by incorporating global relational information. On the other hand, deep reinforcement learning mechanisms enable the model to make decisions oriented towards long-term gains, continuously adjusting scheduling strategies under complex state changes and improving the coordination and stability of system operation. The above results demonstrate that the proposed method has significant application value in improving scheduling efficiency, alleviating waiting overhead, enhancing resource utilization, and ensuring task execution reliability. This also illustrates the clear research significance of synergistically introducing graph neural networks and deep reinforcement learning into distributed task scheduling problems.

To further verify the contribution of key components in our proposed method to scheduling performance, we conduct an ablation analysis focusing on graph structure representation, task resource matching mechanism, and joint optimization objective. The relevant results are shown in Table 3. The changes in various indicators under different settings can intuitively reflect the roles of each module in state modeling, decision quality improvement, and overall scheduling stability enhancement, thus illustrating the necessity of each component within the complete framework.

**Table 3:** Ablation Study Results of the Proposed Method

| Ablation Setting | MS | WT | RU | TSR |
|---|---|---|---|---|
| w/o Graph Embedding | 130.42 | 18.91 | 87.63 | 94.38 |
| w/o Task-Resource Matching | 128.76 | 17.84 | 88.24 | 95.02 |
| w/o Joint Objective | 125.94 | 16.73 | 89.57 | 95.61 |

| | | | | |
|---|---|---|---|---|
| Ours | 121.37 | 15.84 | 91.46 | 96.72 |

Ablation analysis results demonstrate that the proposed complete method outperforms the overall system, indicating that each key module plays an irreplaceable role in the task scheduling process. The graph embedding module enhances the model's ability to express task dependencies and resource association structures, enabling scheduling decisions to be based on more comprehensive global information. The task resource matching module improves the fit between task requirements and node states, thus contributing to more reasonable allocation results. Joint objective optimization further coordinates the relationship between structural representation learning and policy learning, enabling the model to achieve more stable and effective decision-making capabilities in complex environments. Therefore, the proposed method not only possesses strong overall design integrity but also exhibits good synergy among its components. This further illustrates the high application value and research significance of the constructed framework in task scheduling problems within dynamic heterogeneous distributed computing environments.

Figure 2 shows that the proposed algorithm achieves good scheduling performance when the graph encoding depth is set appropriately, indicating that graph structure representation plays a crucial role in task scheduling. A suitable graph encoding depth can more fully capture the complex relationships between task dependencies, resource associations, and system states, thus providing more effective structural information support for subsequent policy learning. This demonstrates that the proposed method not only possesses strong state modeling capabilities in dynamic heterogeneous distributed computing environments but also enhances the accuracy and stability of scheduling decisions through graph representation learning, further showcasing the application value of this framework in improving task scheduling quality.

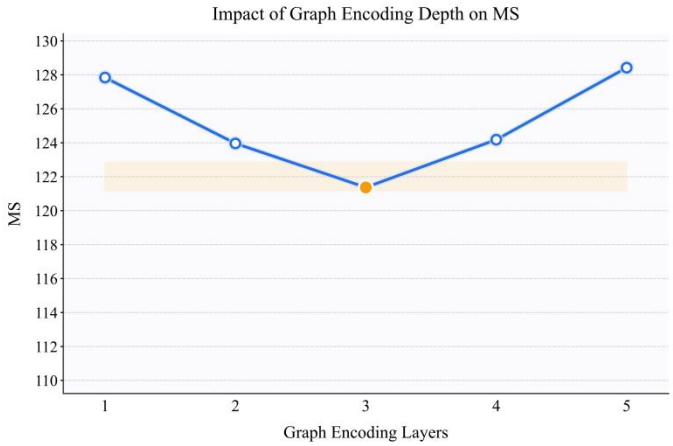


**Figure 2.** Experimental results on the impact of graph coding layer number on scheduling performance on MS.

## 5. Conclusion

This paper focuses on the task scheduling optimization problem in dynamic heterogeneous distributed computing environments. Addressing the shortcomings of traditional methods, such as insufficient modeling of complex dependencies, limited adaptability to heterogeneous resources, and weak long-term decision optimization effects, a task scheduling method based on collaborative modeling using graph neural networks and deep reinforcement learning is proposed. Starting from the actual operational characteristics of distributed systems, this research incorporates task dependencies, resource state changes, and scheduling decision processes into a unified framework. This allows task scheduling to move beyond local rule-driven approaches and enable structured representation and intelligent optimization of complex system states. In this

way, this paper constructs a unified modeling approach for dynamic environments at the methodological level, providing a new research path for intelligent solutions to distributed task scheduling problems.

In terms of methodological design, this paper constructs a relatively complete scheduling model from multiple levels, including the overall framework, structural representation, task-resource matching, and joint objective optimization. Graph structure modeling enables the algorithm to better characterize the predecessor-successor relationships between tasks, the differences between resources, and the interactions between tasks and resources, thereby enhancing its ability to represent the state of complex distributed environments. The deep reinforcement learning mechanism further endows the model with the ability to make decisions based on long-term benefits, enabling the scheduling strategy to continuously adjust and maintain strong adaptability under dynamically changing system states. Overall, the proposed method establishes an effective link between structural information extraction and sequence decision optimization, enhancing the model's understanding of complex scheduling scenarios and improving decision quality. It also demonstrates the feasibility and advancement of combining graph learning and reinforcement learning for distributed system optimization.

From a research perspective, this paper not only provides a new technical solution for task scheduling in dynamic heterogeneous distributed computing environments but also has a strong practical driving effect on related application fields. With the continuous development of cloud computing platforms, edge computing systems, intelligent computing networks, and large-scale online service architectures, task scheduling efficiency, resource utilization quality, and system operational stability have become crucial factors affecting overall service capabilities. The framework proposed in this paper can provide more intelligent and efficient decision support for resource orchestration, service deployment, workflow execution, and task collaboration in complex business scenarios, helping to reduce computing resource waste, improve system responsiveness, and enhance platform robustness. For future intelligent scheduling needs in industrial internet, smart cities, intelligent manufacturing, unmanned system collaboration, and high-performance computing service platforms, this research has strong promotional value and provides a useful reference for the shift in distributed system optimization from rule-driven to learning-driven approaches.

Future research can further expand upon this work in more complex application constraints and broader system scenarios. On one hand, factors such as energy consumption control, service fairness, task security, and multi-objective resource management can be incorporated into a unified scheduling framework, enabling the model to meet efficiency requirements while also considering green computing and reliable operation needs. On the other hand, generalized scheduling mechanisms for cloud-edge-device collaborative environments, multi-tenant scenarios, and cross-regional computing networks can be further explored to enhance the model's migration and deployment capabilities under complex business conditions. Simultaneously, with the increasing demand for large-model-driven intelligent systems and autonomous computing scheduling, combining structured learning methods with higher-level intelligent decision-making mechanisms is expected to drive the development of distributed computing environments towards higher levels of autonomous operation. Overall, this research provides a solid theoretical foundation and methodological support for intelligent distributed task scheduling and will demonstrate a sustained and profound impact in future related application areas.

## References

[1] G. Chen, J. Qi, Y. Sun, et al., "A collaborative scheduling method for cloud computing heterogeneous workflows based on deep reinforcement learning," Future Generation Computer Systems, vol. 141, pp. 284-297, 2023.

[2] F. Cheng, Y. Huang, B. Tanpure, et al., "Cost-aware job scheduling for cloud instances using deep reinforcement learning," Cluster Computing, vol. 25, no. 1, pp. 619-631, 2022.

[3] T. Dong, F. Xue, H. Tang, et al., "Deep reinforcement learning for fault-tolerant workflow scheduling in cloud environment," Applied Intelligence, vol. 53, no. 9, pp. 9916-9932, 2023.

[4] S. Mangalampalli, S. S. Hashmi, A. Gupta, et al., "Multi objective prioritized workflow scheduling using deep reinforcement based learning in cloud computing," IEEE Access, vol. 12, pp. 5373-5392, 2024.

[5] Z. Chen, L. Zhang, X. Wang, et al., "Cloud-edge collaboration task scheduling in cloud manufacturing: An attention-based deep reinforcement learning approach," Computers & Industrial Engineering, vol. 177, p. 109053, 2023.

[6] Z. Chen, J. Hu, G. Min, et al., "Adaptive and efficient resource allocation in cloud datacenters using actor-critic deep reinforcement learning," IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 8, pp. 1911-1923, 2021.

[7] X. Yan, J. Du, L. Wang, Y. Liang, J. Hu and B. Wang, "The Synergistic Role of Deep Learning and Neural Architecture Search in Advancing Artificial Intelligence", Proceedings of the 2024 International Conference on Electronics and Devices, Computational Science (ICEDCS), pp. 452-456, Sep. 2024.

[8] T. Dong, F. Xue, C. Xiao, et al., "Task scheduling based on deep reinforcement learning in a cloud manufacturing environment," Concurrency and Computation: Practice and Experience, vol. 32, no. 11, p. e5654, 2020.

[9] C. Sun, T. Yang, and Y. Lei, "DDDQN – TS: A task scheduling and load balancing method based on optimized deep reinforcement learning in heterogeneous computing environment," International Journal of Intelligent Systems, vol. 37, no. 11, pp. 9138-9172, 2022.

[10] Y. Wang, S. Dong, and W. Fan, "Task scheduling mechanism based on reinforcement learning in cloud computing," Mathematics, vol. 11, no. 15, p. 3364, 2023.

[11] Y. Cheng, Z. Cao, X. Zhang, et al., "Multi objective dynamic task scheduling optimization algorithm based on deep reinforcement learning" The Journal of Supercomputing, vol. 80, no. 5, pp. 6917-6945, 2024.

[12] Y. Gu, F. Cheng, L. Yang, et al., "Cost-aware cloud workflow scheduling using DRL and simulated annealing," Digital Communications and Networks, vol. 10, no. 6, pp. 1590-1599, 2024.

[13] J. Pan and Y. Wei, "A deep reinforcement learning-based scheduling framework for real-time workflows in the cloud environment," Expert Systems with Applications, vol. 255, p. 124845, 2024.

[14] A. Jayanetti, S. Halgamuge, and R. Buyya, "A deep reinforcement learning approach for cost optimized workflow scheduling in cloud computing environments," in Proceedings of the 2024 Asia Pacific Conference on Computing Technologies, Communications and Networking, 2024, pp. 74-82.