Transactions on Computational and Scientific Methods | Vo. 4, No. 11, 2024
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

Adaptive Resource Scheduling in Distributed Computing via Multi-Agent Reinforcement Learning and Graph Convolutional Modeling

Qingyuan Zhang
Boston University, Boston, USA
qyzhang0305@gmail.com

Abstract: This paper proposes an adaptive scheduling method based on multi-agent reinforcement learning to address the complexity and uncertainty of resource dynamic scheduling in distributed computing environments. The method centers on a multi-agent collaboration mechanism, modeling computing nodes as agents with local perception and autonomous decision-making capabilities. Through a centralized training and decentralized execution (CTDE) framework, it achieves a dynamic balance between global optimization and local autonomy. The model incorporates a graph convolutional network to capture topological connections and resource dependencies among nodes, while a joint value function enables global coordination of multi-agent policies, improving cooperative perception and policy convergence efficiency. A multi-level state representation and temporal modeling mechanism are designed, combined with a dynamic reward strategy to handle environmental non-stationarity, enabling adaptive optimization for task load fluctuations, network bandwidth changes, and node failures. Comparative experiments on real distributed task data demonstrate significant improvements in Average Completion Time, system throughput, and resource utilization. Further sensitivity analysis shows that factors such as learning rate, bandwidth limit, task length distribution skewness, and historical window size greatly influence model performance, indicating strong stability and generalization in complex distributed computing scenarios. The results confirm that the proposed framework effectively achieves task-resource matching optimization in highly dynamic environments, providing a practical learning-based scheduling solution for the efficient operation of intelligent distributed systems.

Keywords: Distributed computing; multi-agent reinforcement learning; resource scheduling; graph convolutional modeling

1. Introduction

In the era of deep integration between cloud and edge computing, efficient scheduling of distributed computing resources has become a key issue for ensuring system performance and service quality. With the rapid growth of Internet of Things devices and the increasing diversity of business scenarios, computing demands are now highly dynamic, uncertain, and heterogeneous in both spatial and temporal dimensions[1]. Traditional centralized or static scheduling strategies can no longer adapt to complex system state changes. Under the trend of multi-data center collaboration, edge node autonomy, and cross-domain resource sharing, achieving globally optimized resource scheduling in dynamic environments has become a critical direction in distributed system research. This challenge is not only related to efficient utilization of computing power and energy reduction but also directly affects the execution efficiency of latency-sensitive tasks, service stability, and the overall intelligence of the system[2].

Traditional distributed scheduling methods often rely on heuristic rules or deterministic models, and their performance usually depends on idealized assumptions about system states. However, in real distributed environments, task arrivals are highly stochastic, communication latency and computational capacity vary greatly across nodes, and resource states change rapidly over time. Such complex dynamic interactions make it difficult for traditional scheduling strategies to achieve global optimization under multi-dimensional constraints. Moreover, as computing shifts from cloud to edge and terminal devices, distributed systems have become increasingly hierarchical and autonomous. Centralized scheduling mechanisms show poor scalability, low fault tolerance, and slow responsiveness. Therefore, there is an urgent need for intelligent scheduling methods capable of adaptive decision-making in uncertain and dynamic environments, enabling efficient task-resource matching and global optimization[3].

The rise of reinforcement learning offers a new perspective for solving adaptive optimization problems in distributed resource scheduling. This paradigm allows agents to learn optimal decision strategies through interaction with the environment without relying on explicit system models. However, in complex distributed environments, single-agent reinforcement learning struggles to handle large-scale systems, high-dimensional state spaces, and conflicting objectives. It also fails to capture cooperative relationships among nodes. Multi-Agent Reinforcement Learning (MARL) introduces multiple agents that can perceive and collaborate with each other, enabling resource allocation decisions to balance local autonomy and global coordination. This results in stronger scalability and robustness. Each agent can represent a computing node or a task entity, and through game-like interactions and shared policy updates, the distributed system achieves dynamic adaptive scheduling in complex environments.

Introducing MARL into distributed computing scenarios holds significant theoretical and practical implications. From a theoretical perspective, MARL provides a data-driven optimal control framework for dynamic scheduling. Jointly modeling the state space, action space, and reward mechanism enables self-learning and self-evolution of strategies in high-dimensional nonlinear systems, overcoming the limitations of traditional algorithms that depend on precise modeling and complete global information. From an engineering perspective, MARL-based dynamic scheduling effectively addresses issues such as uneven task distribution, network fluctuations, and node failures, achieving elastic scaling and load balancing in distributed systems. Its distributed decision-making nature allows nodes to approximate global optimality based on local observations, greatly reducing communication overhead and central bottlenecks, thus offering a feasible path for large-scale intelligent computing infrastructures[4].

Moreover, with the deep convergence of AI and computing infrastructure, intelligent scheduling is evolving into a core capability of autonomous systems. In cloud-edge-end collaborative computing, Internet of Vehicles, Industrial Internet, and smart city applications, real-time dynamic matching between tasks and resources determines system efficiency. The adoption of MARL not only drives the intelligent transformation of distributed scheduling but also promotes the development of autonomous decision-making and adaptive optimization in systems. It enables dynamic equilibrium between global coordination and local autonomy, achieving continuous optimization in complex environments. This learning-based distributed cooperative scheduling paradigm is becoming a new direction for intelligent computing systems toward autonomy and efficiency, with profound academic and practical significance.

2. Related work

The study of distributed computing resource scheduling has long been an important topic in the fields of high-performance computing and cloud computing. Early research mainly focused on centralized scheduling and heuristic algorithms, such as rule-based load balancing, priority-based task allocation, and scheduling strategies based on genetic or ant colony optimization. These methods achieved good resource utilization in static or semi-static environments, but their adaptability to dynamic changes was limited. As system scales expanded, task types diversified, and node resource states changed rapidly, centralized scheduling faced bottlenecks, and heuristic algorithms struggled to find optimal solutions in high-dimensional state spaces.

With the emergence of cloud and edge computing architectures, distributed systems now exhibit multi-level collaboration and heterogeneous integration, making it difficult for traditional static optimization models to balance real-time performance, scalability, and robustness[5].

In recent years, intelligent optimization algorithms have been increasingly applied to distributed resource scheduling to achieve adaptive decision-making and approximate global optimization. Some studies have integrated machine learning methods with resource scheduling, using historical data to predict task arrival rates or resource demands for proactive task planning. However, these methods often rely on offline training and fail to respond effectively to environmental changes in real time[6]. Other research has explored deep reinforcement learning frameworks for dynamic scheduling, where optimal strategies are learned through interactions with the environment, enabling self-learning and self-optimization without explicit system models. Such methods have shown promising results in small-scale systems, but as the number of nodes and task complexity grow, single-agent reinforcement learning encounters challenges such as state space explosion and unstable training. Furthermore, a single agent cannot effectively handle resource competition and cooperation among multiple tasks, which limits the generalization capability of the learned policy.

The emergence of multi-agent reinforcement learning (MARL) has provided a new solution for distributed resource scheduling. In this framework, different computing nodes or task entities are treated as independent agents. Each agent perceives its local state and makes autonomous decisions while achieving global optimization through cooperation or competition. Early studies mostly adopted independent learning approaches, where agents trained their own policies separately. However, this often led to non-stationary environments and poor convergence. To address these issues, research has shifted toward collaborative mechanisms with joint modeling and shared policies. The centralized training and decentralized execution paradigm (CTDE) allows information sharing and coordinated policy optimization, improving stability and scalability in complex environments. Recently, various MARL algorithms based on value decomposition, policy gradients, and graph modeling have been widely applied to task allocation, load balancing, and energy optimization, significantly enhancing the decision-making capability and robustness of distributed systems in dynamic settings[7].

Despite the great potential of MARL in distributed scheduling, several challenges remain. First, multi-agent systems with high-dimensional state and action spaces often suffer from heavy communication overhead and convergence difficulties. These issues become more severe in large-scale or task-intensive environments, where learning efficiency and stability are difficult to ensure. Second, the dynamic and uncertain nature of distributed computing environments causes frequent changes in the cooperation patterns among agents, and traditional fixed communication topologies or static reward designs cannot adapt effectively. Third, existing studies often lack mechanisms to balance global coordination and local autonomy. Some models achieve short-term performance gains but tend to experience policy degradation or cooperation imbalance during long-term operation. Therefore, developing a MARL framework with dynamic game characteristics, adaptive cooperation capabilities, and cross-layer resource optimization has become a major research trend in distributed computing resource scheduling. The continued exploration of this direction will provide new theoretical foundations and technical pathways for autonomous scheduling and efficient operation in future intelligent computing infrastructures.

3. Proposed Framework

This study addresses the dynamic and uncertain nature of resource scheduling in distributed computing environments by constructing an adaptive decision-making framework based on multi-agent reinforcement learning. The system comprises an environment perception layer, an agent decision-making layer, and a global coordination layer. Each computing node is treated as an independent agent, making resource allocation and task scheduling decisions based on its local observations. The system state can be represented as the set of global states t at time $S_t = \{s_t^1, s_t^2, \ldots, s_t^N\}$, where N is the number of nodes. Each agent

selects an action α_t^i based on its local observations o_t^i and receives an immediate reward r_t^i from the environment. The dynamic interactions of the system follow a Markov decision process (MDP), and its transition relationships can be represented as follows:

$$P(S_{t+1}/S_t, A_t) = \prod_{i=1}^{N} P(s_{t+1}^i/s_t^i, a_t^i)$$

Here, $A_t = \{\alpha_t^1, \alpha_t^2, \ldots, \alpha_t^N\}$ represents the set of joint actions. This model can characterize the parallel interactions and resource state changes among multiple nodes, providing a foundation for subsequent joint strategy optimization. Its overall model architecture is shown in Figure 1.

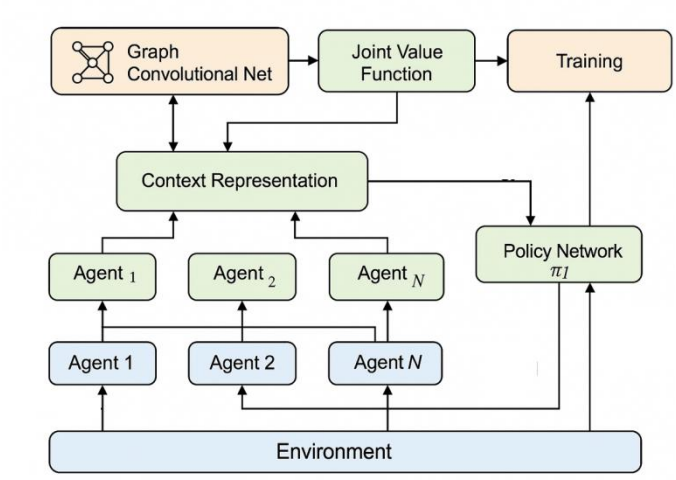

Figure 1. Overall model architecture diagram

To achieve optimal decision-making in dynamic environments, each agent continuously updates its policy function through reinforcement learning. Let $\pi_i(\alpha_t^i/o_t^1;\theta_t)$ be the policy function of each agent i, and its objective is to maximize the long-term cumulative reward, i.e.:

$$J_i(\theta_i) = E_{\pi_i}[\sum_{t=0}^{T} \gamma^t r_t^i]$$

Where $\gamma \in (0,1)$ is the discount factor, reflecting the importance of future rewards. The strategy optimization process is achieved through gradient ascent, and its update rule is as follows:

$$\nabla_{\theta_i} J_i(\theta_i) = E_{\pi_i}[\nabla_{\theta_i} log \pi_i(a_t^i/o_t^i;\theta_i) Q_i(o_t^i, a_t^i)]$$

Here, $Q_i(\bullet)$ is the agent's action value function, used to measure the expected reward obtained by acting in a specific state. This mechanism enables the agent to continuously adjust its strategy during interactions to adapt to the time-varying characteristics of resources and tasks.

In multi-agent scenarios, a centralized training and distributed execution (CTDE) paradigm is adopted to balance local autonomy and global coordination. During the training phase, each agent shares global state information S_t and achieves collaborative optimization through a joint value function; during the

execution phase, they make independent decisions based solely on local observations. The joint value function can be defined as:

$$Q_{tot}(S_t, A_t) = f(Q_1(o_t^1, a_t^1), Q_2(o_t^2, a_t^2), \ldots, Q_N(o_t^N, a_t^N))$$

Here, $f(\bullet)$ is an additive or decomposable function that satisfies monotonicity constraints to ensure that local optima lead to global optima. Through this mechanism, the system maintains distributed scalability while enhancing the collaborative capabilities among multiple agents, avoiding efficiency degradation caused by resource contention.

To further enhance the dynamic adaptability of scheduling, this study introduces a graph-based relational modeling mechanism at the global layer. The resource dependencies of a distributed system can be abstracted as an undirected graph $G = (V, E)$, where the node set V represents computing nodes and the edge set E represents resource communication relationships between nodes. By using a Graph Convolutional Network (GCN) to propagate and aggregate system state features, the context representation of each node can be obtained.

$$h_t^{(l+1)} = \sigma\left(\sum_{j \in N(i)} \frac{1}{c_{ij}} W^{(l)} h_j^{(l)}\right)$$

Where $h_j^{(l)}$ represents the node features of the l-th layer, $W^{(l)}$ represents the learnable weights, $\sigma(\bullet)$ represents the nonlinear activation function, and c_{ij} represents the adjacency normalization coefficient. This representation enables joint modeling of task dependencies, node load, and network topology, allowing multiple agents to make collaborative decisions under structure-aware conditions. Ultimately, the global scheduling objective can be expressed as minimizing the weighted sum of overall system latency and energy consumption:

$$\min_{\Pi} E_{\Pi}\left[\alpha \sum_{i=1}^{N} T_i + \beta \sum_{i=1}^{N} E_i\right]$$

Where T_i represents task execution latency, E_i represents node energy consumption, and α and β are trade-off parameters. This optimization objective achieves a balance between performance and energy efficiency, enabling the system to maintain efficient and stable resource scheduling performance even under dynamically changing load conditions.

4. Experimental Analysis

4.1 Dataset

This study uses the Google Cluster Workload Trace dataset as the primary source of experimental data. The dataset consists of real operation logs collected from large-scale distributed computing clusters. It records task scheduling, resource allocation, and load variation across thousands of servers over an extended period. These data accurately reflect the dynamic characteristics of cloud scheduling systems under complex and changing environments. The dataset includes multi-dimensional resource usage information such as CPU, memory, and disk I/O, as well as scheduling-related attributes such as task submission time, execution duration, priority, and failure status. The time span exceeds one month, and the total record count reaches billions, providing strong support for performance modeling and intelligent scheduling in distributed systems.

In the data preprocessing stage, several steps were taken to ensure model generalization and stability. First, the raw logs were cleaned, and invalid tasks and abnormal node records were removed. Then, task-level and node-level features were normalized and aligned over time to construct a temporal resource state matrix. This matrix represents the overall resource utilization and inter-node relationships of the cluster at any given moment, forming the foundation for multi-agent state modeling. Meanwhile, to improve training efficiency and dynamic adaptability, the dataset was divided into training, validation, and testing subsets with a ratio of 7:2:1, while maintaining consistency in task categories and load distribution.

The dataset is characterized by high dimensionality, strong dynamics, and significant temporal correlations, making it well-suited for evaluating the performance of multi-agent reinforcement learning in complex distributed environments. Its multi-source heterogeneous structure allows for the assessment of model adaptability under varying load fluctuations, resource constraints, and task priorities. Furthermore, the real-cluster characteristics of the Google Cluster Workload Trace make the experimental results more reliable and meaningful, providing valuable data support and verification for deploying intelligent scheduling systems in practical cloud and edge computing platforms.

4.2 Experimental Results

This paper first conducts a comparative experiment, and the experimental results are shown in Table 1.

Table 1: Comparative experimental results

Method	ACT	Makespan	Throughput	CPU Utilization
DQN[8]	12.84	41.52	134.7	68.3%
DDQN[9]	11.37	39.86	142.5	71.2%
SARSA[10]	13.25	42.90	129.8	66.5%
DDPG[11]	10.74	38.64	148.9	73.1%
A2C[12]	10.58	37.42	152.3	74.8%
PPO[13]	9.96	36.81	156.7	76.5%
Ours	8.73	33.45	169.4	81.2%

As shown in Table 1, different reinforcement learning methods exhibit significant differences in task completion efficiency, resource utilization, and system throughput in distributed computing resource scheduling. Traditional value-based algorithms such as DQN and SARSA show limitations in handling complex and dynamic resource allocation scenarios, as reflected in higher Average Completion Time (ACT) and Makespan values. These methods struggle to capture inter-node interactions effectively in high-dimensional state spaces, leading to slow policy convergence and insufficient global coordination, which results in relatively lower overall system performance.

In contrast, policy gradient-based methods such as DDPG, A2C, and PPO perform better in continuous decision spaces. They can better balance global and local optimization in dynamic environments. As shown in the table, these methods achieve significantly lower ACT and Makespan compared with value-based algorithms, indicating stronger adaptability under fluctuating workloads. Their improvements in throughput and CPU utilization also demonstrate enhanced capability in handling concurrent tasks and achieving balanced resource allocation, leading to higher system stability and faster response.

Furthermore, the proposed multi-agent reinforcement learning model (Ours) achieves the best performance across all four evaluation metrics. Through multi-agent collaboration and graph-based state modeling, the

model effectively captures resource dependencies and dynamic interactions among computing nodes, which greatly enhances global decision coordination. The average task completion time decreases to 8.73, and the throughput increases to 169.4, showing that the model can adaptively optimize its policy under high concurrency and resource competition, achieving dual improvements in scheduling efficiency and resource utilization.

Overall, the experimental results confirm the effectiveness and advantages of multi-agent reinforcement learning in distributed computing resource scheduling. This approach overcomes the performance bottlenecks of traditional centralized scheduling in large-scale dynamic environments and establishes adaptive collaboration among heterogeneous computing nodes. By integrating a joint value function with a graph convolutional structure, the model achieves coordinated optimization between global perception and local decision-making, providing a new technical path and theoretical foundation for building intelligent, adaptive, and efficient distributed resource scheduling systems.

This paper also presents an experiment on the sensitivity of the learning rate to ACT, and the experimental results are shown in Figure 2.

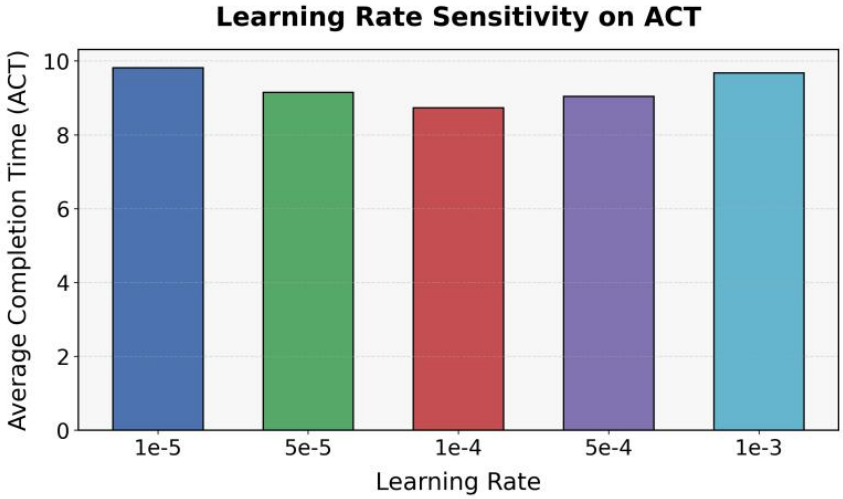

Figure 2. Sensitivity experiment of learning rate to ACT

As shown in Figure 2, the learning rate has a clear impact on the Average Completion Time (ACT). When the learning rate is low (1e-5 or 5e-5), the policy update speed of the model is slow. This makes it difficult for agents to quickly converge to high-quality scheduling strategies in the early training stage, resulting in higher ACT values. This indicates that in distributed resource scheduling scenarios, an excessively low learning rate weakens the model's ability to respond to dynamic environmental changes, leading to conservative task allocation and limited overall scheduling efficiency.

When the learning rate increases to a moderate level (1e-4), the ACT reaches its optimal value. At this point, the model achieves a good balance between stability and exploration. The multi-agent system performs more accurate task allocation and resource utilization through efficient policy iteration, which significantly reduces task completion time. This result suggests that a moderate learning rate promotes better policy coordination among agents, accelerates the convergence of the global value function, and enables the system to form a stable and optimal scheduling pattern in complex and dynamic environments.

However, when the learning rate continues to increase (5e-4 or 1e-3), the ACT begins to rise again. This shows that an excessively high learning rate causes overly aggressive policy updates, leading to instability during training. Agents oscillate frequently between local optima and fail to reach a stable global policy. Therefore, in distributed computing resource scheduling tasks, the learning rate should be controlled within a

moderate range to maintain both convergence speed and model stability. This balance ensures better generalization and enables more efficient and robust resource scheduling performance.

This paper also presents an experiment on the sensitivity of bandwidth limit to throughput, and the experimental results are shown in Figure 3.

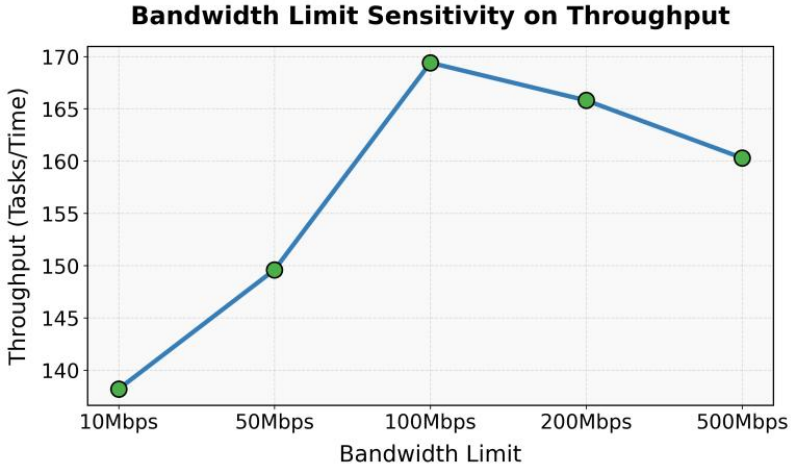

Figure 3. Experiment on the sensitivity of the bandwidth limit to throughput

As shown in Figure 3, the bandwidth limit has a significant impact on system throughput. Under low-bandwidth conditions (such as 10 Mbps and 50 Mbps), the system throughput is low. This indicates that communication bottlenecks restrict the efficiency of state synchronization and decision collaboration among agents. Since distributed computing resource scheduling requires frequent exchanges of node states, resource utilization, and task allocation information, limited bandwidth increases communication delay. This reduces the agents' ability to perceive and respond to global environmental changes, ultimately decreasing the overall task processing capacity.

When the bandwidth limit increases to a moderate level (around 100 Mbps), the system throughput reaches its peak. This suggests that within this range, agent collaboration and policy sharing achieve an optimal balance. Sufficient communication bandwidth ensures real-time transmission of state information, allowing the joint value function and global policy to be efficiently updated. At this point, agents can more accurately assess system load conditions and perform task reallocation, leading to a significant improvement in task completion rate and resource utilization. This performance enhancement highlights the critical role of communication resources in supporting reinforcement learning-based scheduling models.

However, when the bandwidth is further increased to 200 Mbps or 500 Mbps, the system throughput begins to decline. Although higher bandwidth reduces communication latency, it introduces additional scheduling overhead and synchronization redundancy. This can cause some nodes to update their decisions too frequently, resulting in instability within the system. Moreover, excessive state exchanges among agents may lead to resource contention and scheduling conflicts, reducing the overall optimization efficiency. This phenomenon reflects a typical "communication redundancy effect."

In summary, the experiment demonstrates that the performance of distributed resource scheduling systems does not simply improve with larger bandwidth but instead depends on an optimal communication range. Moderate bandwidth achieves a balance between timely information transmission and system overhead, enabling the multi-agent collaboration mechanism to operate effectively. Therefore, the proposed multi-agent reinforcement learning framework exhibits strong adaptability under dynamic network conditions and maintains stable and efficient scheduling performance across different communication environments.

This paper also presents the impact of task length distribution skewness on the experimental results, and the experimental results are shown in Figure 4. To ensure the quantifiable comparability of task length skewness,

this study categorizes it into four typical levels based on the magnitude of the skewness coefficient. Specifically, a skewness coefficient between 0 and 0.5 indicates a relatively balanced task length distribution, defined as Low; a skewness coefficient between 0.5 and 1.0 indicates slight skewness, defined as Moderate; a skewness coefficient between 1.0 and 1.5 indicates a significantly increased proportion of long tasks, defined as High; and a skewness coefficient of 1.5 or higher signifies a highly unbalanced task distribution, defined as Extreme. This categorization effectively describes the changes in task length imbalance under different scenarios, providing clear quantitative boundaries and consistency for experimental comparisons.

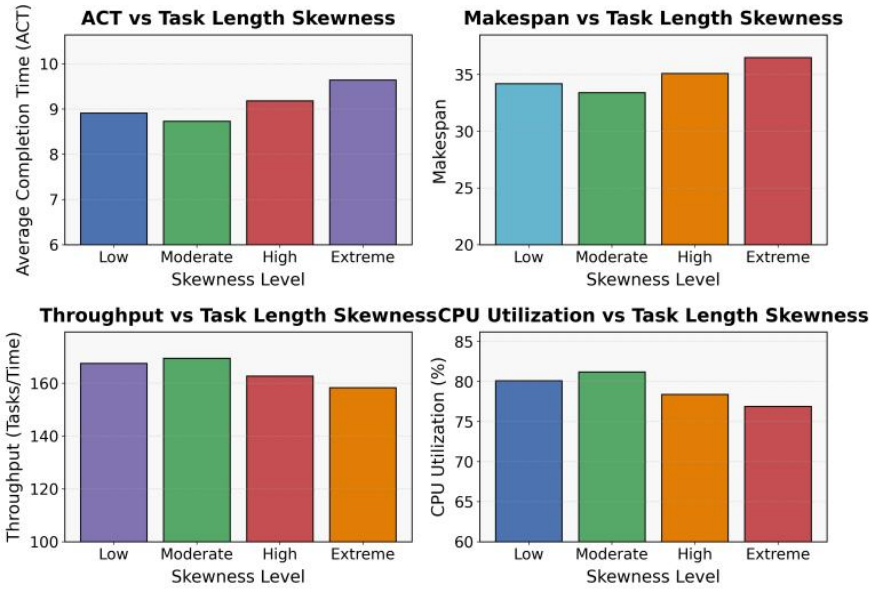

Figure 4. The effect of task length distribution skewness on experimental results

As shown in Figure 4, the skewness of the task length distribution has a significant impact on the performance of distributed computing resource scheduling. When the task length distribution is relatively uniform (Low or Moderate), both the Average Completion Time (ACT) and Makespan remain at low levels. This indicates that the model can achieve high scheduling efficiency under balanced task loads. It shows that the proposed multi-agent reinforcement learning model can quickly learn resource matching strategies under regular task distributions and realize dynamic global scheduling optimization.

As the skewness of the task length distribution increases (High or Extreme), both ACT and Makespan show an upward trend. This is mainly because a highly skewed distribution causes some tasks to be much longer than the average, which increases overall waiting time and creates uneven resource utilization. The concentration of long tasks leads to larger load differences among nodes. Agents must adjust their strategies more frequently to avoid resource congestion. This imbalance results in accumulated scheduling delays and reduced task parallelism, showing that complex task distributions place higher demands on multi-agent coordination and decision-making.

From the perspective of Throughput and CPU Utilization, the system maintains high resource usage efficiency and task processing speed when the skewness is low to moderate. A moderate degree of task length variation helps the model explore more efficient resource allocation patterns under diverse workloads. Agents achieve dynamic load balancing through joint policy optimization. However, as skewness continues to increase, both throughput and CPU utilization decline, indicating the emergence of local congestion effects in resource scheduling and limited overall execution efficiency.

In summary, the experimental results show that the skewness of task length distribution is an important factor affecting distributed scheduling performance. The proposed multi-agent reinforcement learning scheduling model performs best under moderate skewness, demonstrating strong adaptability and generalization

capability. It can achieve efficient resource scheduling in heterogeneous task environments. However, when the task distribution becomes extremely uneven, the decline in system performance highlights the need for introducing task stratification and dynamic priority mechanisms in future research to further enhance the model's stability and scheduling robustness under extreme load conditions.

This paper presents the impact of the historical window step size on the experimental results, and the experimental results are shown in Figure 5.

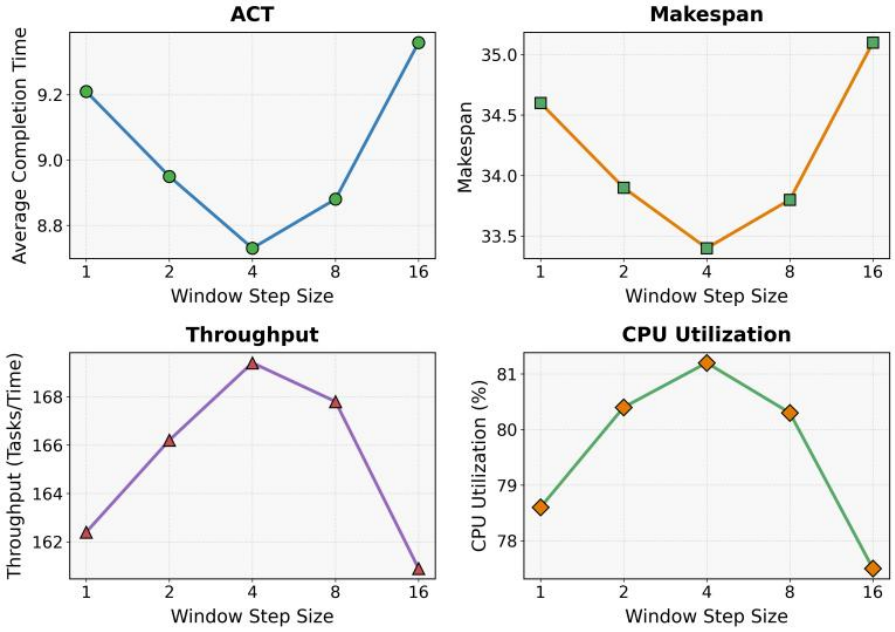

Figure 5. The impact of historical window step size on experimental results

As shown in Figure 5, the length of the historical window has a significant impact on the performance of distributed computing resource scheduling. When the window size is small (such as 1 or 2), both the Average Completion Time (ACT) and Makespan are relatively high. This indicates that when historical information is insufficient, agents fail to capture the dynamic changes in system states effectively. As a result, short-term policy fluctuations and accumulated scheduling delays occur. A short time window makes agents overly dependent on immediate feedback, causing them to ignore the temporal dependencies of task loads, which reduces policy stability and global coordination.

When the window size is moderate (around 4), both ACT and Makespan reach optimal values, while throughput and CPU utilization also achieve their highest levels. This shows that the model achieves a good balance between retaining historical information and maintaining real-time responsiveness. An appropriate window length helps agents capture trends in task load changes, allowing the model to consider both local timeliness and global consistency in decision-making. Consequently, the system achieves higher resource allocation efficiency and operational stability. These results demonstrate the adaptive capability of the proposed multi-agent reinforcement learning framework in temporal sequence modeling.

As the historical window continues to increase (8 or 16), both ACT and Makespan rise again, while throughput and CPU utilization decrease. This indicates that an excessively long window introduces redundant historical information, causing the model to suffer from "information inertia" in state estimation. In this case, agents tend to make decisions based on long-term average states, leading to delayed responses and slower scheduling decisions. This weakens the model's sensitivity to real-time environmental changes. The resulting performance degradation shows that in dynamic distributed environments, overly long temporal memory can also hinder the system's fast adaptability.

Overall, the experimental results show that the choice of historical window length plays a crucial role in multi-agent scheduling systems. A moderate window size can effectively enhance the model's temporal awareness and prediction accuracy, thereby improving resource scheduling efficiency and system throughput. The proposed method dynamically adjusts the historical window length within the reinforcement learning framework, enabling agents to maintain both flexibility and stability under varying workload fluctuations. This provides a practical optimization direction for achieving adaptive scheduling in complex distributed computing environments.

5. Conclusion

This study focuses on the problem of dynamic scheduling of distributed computing resources and proposes an adaptive optimization framework based on multi-agent reinforcement learning. The proposed method achieves a dynamic balance between global resource utilization and local task allocation through collaborative decision-making among multiple agents. It effectively improves system scheduling efficiency and robustness in complex environments. The model employs a joint value function and graph convolutional structure to capture the relationships among computing nodes, enabling real-time policy updates and global optimization in heterogeneous, dynamic, and non-stationary distributed systems. Experimental results show that the proposed approach achieves significant advantages in key metrics such as Average Completion Time, system throughput, and resource utilization, demonstrating its capability for adaptive learning and optimization under multi-dimensional constraints.

Through sensitivity analysis of different hyperparameters, communication environments, and data characteristics, this study further reveals the key factors influencing the performance of multi-agent reinforcement learning in resource scheduling tasks. The experiments show that parameters such as learning rate, bandwidth limit, task length distribution skewness, and historical window size significantly affect system performance. These findings provide valuable insights into the dynamic coordination mechanisms among agents in complex systems and offer theoretical guidance for designing distributed scheduling algorithms. In particular, the proposed framework maintains high performance stability even under uneven task distributions and limited network resources, showing strong application potential in cloud computing, edge computing, vehicular networks, and large-scale data center management.

The main contribution of this work lies in constructing a scheduling model with structural awareness and temporal adaptability. This enables multiple agents to achieve intelligent collaboration and dynamic optimization of distributed resources through reinforcement learning. The model overcomes the limitations of traditional centralized scheduling in scalability and real-time responsiveness while providing interpretable decision-making for autonomous scheduling in complex systems. The proposed framework has strong transferability and extensibility and can be applied to load balancing, task offloading, energy optimization, and multi-cloud collaborative management. It lays a technical foundation for the autonomous operation of next-generation intelligent computing infrastructures.

Future research can be extended in several directions. First, adaptive hierarchical control mechanisms and attention-based constraints can be introduced to achieve cross-scale task perception and multi-level decision coordination, thereby enhancing global consistency in highly dynamic environments. Second, integrating federated reinforcement learning with privacy-preserving techniques can enable collaborative and secure scheduling across multiple distributed domains. In addition, future work may explore multimodal input and generative policy modeling to deeply integrate environment perception with policy generation. This will promote distributed scheduling systems toward autonomous learning, continuous optimization, and high interpretability, providing forward-looking solutions for intelligent cloud-edge collaboration and large-scale computing resource management.

References

[1] Chen M, Guo A, Song C. Multi-agent deep reinforcement learning for collaborative task offloading in mobile edge computing networks[J]. Digital Signal Processing, 2023, 140: 104127.

[2] Yao Z, Ding Z, Clausen T. Multi-agent reinforcement learning for network load balancing in data center[C]//Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 2022: 3594-3603.

[3] Zhou G, Tian W, Buyya R, et al. Deep reinforcement learning-based methods for resource scheduling in cloud computing: A review and future directions[J]. Artificial Intelligence Review, 2024, 57(5): 124.

[4] Di Y, Deng L, Zhang L. A collaborative-learning multi-agent reinforcement learning method for distributed hybrid flow shop scheduling problem[J]. Swarm and Evolutionary Computation, 2024, 91: 101764.

[5] Li M, Gao J, Zhao L, et al. Deep reinforcement learning for collaborative edge computing in vehicular networks[J]. IEEE Transactions on Cognitive Communications and Networking, 2020, 6(4): 1122-1135.

[6] Zhang Y, Xia G, Yu C, et al. Fault-Tolerant Scheduling Mechanism for Dynamic Edge Computing Scenarios Based on Graph Reinforcement Learning[J]. Sensors, 2024, 24(21): 6984.

[7] Ding S. Multi-agent reinforcement learning for task allocation in cooperative edge cloud computing[C]//International Conference on Service-Oriented Computing. Cham: Springer International Publishing, 2021: 283-297.

[8] ZHANG Y, Yuanhao Z O U, Xiaodong Z. Manufacturing resource scheduling based on deep q-network[J]. Wuhan University Journal of Natural Sciences, 2022, 27(6): 531-538.

[9] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning[C]//Proceedings of the AAAI conference on artificial intelligence. 2016, 30(1).

[10] G. Tesauro, N. K. Jong, R. Das and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," Proceedings of the 2006 IEEE International Conference on Autonomic Computing, pp. 65-73, 2006.

[11] Shang D, Zhang C, Wang J. A NEW M-DDPG ALGORITHM FOR COMPUTING TASK SCHEDULING AND RESOURCE ALLOCATION IN EDGE ENVIRONMENTS[C]//Proceedings of the 2024 7th Artificial Intelligence and Cloud Computing Conference. 2024: 632-638.

[12] Sun Y, Zhang X. A2C learning for tasks segmentation with cooperative computing in edge computing networks[C]//GLOBECOM 2022-2022 IEEE Global Communications Conference. IEEE, 2022: 2236-2241.

[13] De La Fuente N, Guerra D A V. A comparative study of deep reinforcement learning models: Dqn vs ppo vs a2c[J]. arXiv preprint arXiv:2407.14151, 2024.