CIT. Journal of Computing and Information Technology, Vol. 33, No. 4, December 2025, 249–266
doi: 10.20532/cit.2025.1006202

Multi-Agent Reinforcement Learning with Graph Convolutional Networks for Collaborative Task Scheduling in Distributed Virtual Reality Systems

JieChen Zhao

Huangshan Vocational Technical College, Huangshan, Anhui, China

Efficiently scheduling collaborative tasks in multi-user, distributed virtual reality systems is challenging due to the dynamic nature of user interactions and resource constraints. To address this problem, a novel task scheduling framework integrating multi-agent reinforcement learning and graph convolutional networks is proposed. The distributed VR scheduling problem is formulated as a Markov game, for which the multi-agent proximal policy optimisation algorithm serves as the foundational decision-making framework. Graph convolutional networks are used to capture the dynamic topological relationships between tasks and users. This enhances global perception capabilities and enables more refined collaborative decision-making through graph attention mechanisms. Experimental results demonstrated that the proposed model outperformed baseline approaches in terms of task completion rate, system throughput, and resource utilisation. In simulation environments, the framework maintained low latency and high success rates across a range of VR application scenarios. This research presents a structured methodology for intelligent scheduling in distributed collaborative systems, promoting the integration of graph-based learning and multi-agent coordination in complex virtual environments. This work makes a valuable contribution to scheduling theory and practical system design, offering valuable insights for the development of responsive and scalable VR platforms.

ACM CCS (2012) Classification: Computing methodologies → Artificial intelligence → Distributed artificial intelligence → Cooperation and coordination

Keywords: multi-agent reinforcement learning, distributed virtual reality, collaborative task scheduling, graph convolutional networks, markov games

1. Introduction

With the rapid advancement of artificial intelligence, edge computing, and virtual reality (VR) technologies, multi-user collaborative tasks have garnered significant attention in fields such as remote collaborative design, multiplayer online gaming, and virtual training [1]. These applications involve numerous users simultaneously executing complex tasks within virtual environments, imposing stringent demands on system real-time performance, collaborative efficiency, and resource scheduling capabilities [2]. However, traditional centralized task scheduling (TS) methods struggle to effectively handle dynamic, multi-user environments, suffering from high response delays, low resource utilisation, and insufficient coordination efficiency [3].

Addressing these challenges requires innovations that lie at the intersection of distributed computing and intelligent systems. Multi-agent reinforcement learning (MARL) has emerged as a promising paradigm for enabling decentralized decision-making in collaborative environments [4]. MARL is well suited for scalable and adaptive coordination in distributed VR systems due to its ability to operate based on local observations without global information [5].

For instance, Johnson *et al.* developed a decentralized MARL-based scheduling method uti-

lizing deep Q-networks with centralized training, which effectively reduced task completion times compared to heuristic approaches [6]. Similarly, Jayanetti *et al.* designed a multi-agent deep reinforcement learning (DRL) framework for dynamic wireless networks. They demonstrated that distributed decision-making could substantially improve the quality of the user experience [7]. Extending this, Betalo *et al.* proposed a collaborative multi-agent DRL scheduling framework, achieving performance gains of up to 70% over competing methods [8]. In large-scale, distributed systems, such as satellite-ground networks, Lin *et al.* decomposed the scheduling problem into long-term load balancing and short-term decisions. This approach achieved high throughput with minimal latency [9]. Applications in smart grids were advanced by Zhang *et al.*, who employed a multi-step MARL framework for energy scheduling, enhancing both economic efficiency and user satisfaction [10]. Despite these advances, a common limitation of existing MARL approaches is their inadequate modelling of structural dependencies among tasks and users. The lack of a global relational perspective often constrains the decision-making process, restricting the potential for fully optimized collaboration in complex, networked environments.

To address the relational complexity in multi-agent systems, graph convolutional networks (GCNs) have recently been integrated with reinforcement learning. GCNs are a powerful mechanism for representing and reasoning about graph-structured data. They enable agents to capture topological features and interdependencies [11]. For example, Jing *et al.* integrated GCNs with MARL for scheduling in flexible manufacturing workshops, yielding superior performance in complex scenarios [12]. Yang *et al.* used GCNs to reduce the computational complexity of DRL in time-sensitive networks and improve end-to-end delay performance [13]. Liu *et al.* combined MARL with graph attention networks to achieve efficient communication and lower overhead [14]. Furthermore, Yang *et al.* incorporated GCNs into a multi-policy DRL framework, thereby increasing resource utilisation by 18%. Additionally, Xiao *et al.* proposed a two-stage GCN-assisted DRL method for service chain embedding, which improved acceptance rates in multi-datacenter networks [15−16]. Although these studies emphasize the benefits of graph-based learning, the use of GCN-enhanced MARL in distributed VR settings is still underdeveloped. Additionally, effectively fusing dynamic topological reasoning with multi-agent policy optimisation under real-time system constraints has not been sufficiently addressed.

The reviewed literature reveals several interconnected gaps. First, MARL-based schedulers often lack explicit mechanisms for modelling dynamic task-user topologies. Second, although GCNs improve relational reasoning, they have not been adapted to meet the high-concurrency, low-latency requirements of distributed VR environments. Third, a comprehensive solution that unifies topological perception and collaborative scheduling under real-world VR conditions is still lacking.

To bridge these gaps, a multi-user collaborative task scheduling (MUCTS) optimisation model that integrates MARL with GCNs has been proposed. This model not only improves upon existing baselines in terms of performance but also represents a computing innovation that enhances coordination and scalability in distributed VR architectures. The innovation of this study lies in:

1. A novel computing framework has been developed that deeply integrates GCNs with the multi-agent proximal policy optimisation (MAPPO) algorithm. This integration allows topological dependencies between tasks and users to be modelled dynamically, thereby enhancing global perception in distributed VR environments.
2. The introduction of graph attention mechanisms and structural reconstruction loss enables adaptive learning of node importance and improves feature representation (FR), supporting more efficient and scalable collaborative decision-making.

2. Optimisation Method of MUCTS

This section constructs a VR TS model based on MARL. Subsequently, GCNs are introduced to propose a policy enhancement method based on heterogeneous GCNs, achieving scheduling optimisation for multi-user collaborative tasks.

2.1. VR TS modelling based on MARL

Distributed VR systems face challenges in terms of TS that demand high concurrency, low latency, and high reliability. Traditional centralised scheduling methods struggle to handle dynamic, partially observable, multi-user environments, resulting in poor resource utilisation, high response latency, and inefficient coordination [17]. MARL enables locally optimal scheduling in the absence of global information through distributed decision-making and collaborative optimisation [18]. Therefore, this study constructs a distributed VR TS model based on MARL, aiming to achieve dynamic and efficient scheduling of multi-user tasks. Equation (1) illustrates how the research initially models the MUCTS problem as a Markov game process under the MARL framework.

$$\begin{cases} J(\pi) = E_{\pi}\left[\sum_{t=0}^{T} \gamma^{t} R_{t}\right] \\ R_{t} = \sum_{i=1}^{N} r_{t}^{i} \end{cases} \tag{1}$$

In Equation (1), $J(\pi)$ denotes the expected cumulative discounted reward obtained by the intelligent population under policy $\pi \cdot \pi$ represents the joint policy of all agents. E_{π} is the expectation of policy π and state transition probabilities. γ represents the discount factor. R_t is the global reward at time step (TiS) t. N denotes the total number of agents. r_t^i is the individual reward obtained by agent i at TiS t. Figure 1 depicts a schematic diagram of the Markov game process.

In Figure 1, multiple agents simultaneously perceive their own states within a distributed environment. They take activities and make judgments based on local observations, each of which is rewarded with feedback. To evaluate strategy performance, each agent must maintain its own policy function and value function [19]. Agent i's policy function is $\pi_{\theta}(a_t^i|s_t^i)$, and its value function is $V_{\phi}(s_t)$, parameterized by actor network (AN) and critic networks (CN) respectively. The study employs an advantage function (AF) to measure an action's superiority relative to the mean, thereby more accurately estimating the advantage value, as shown in Equation (2) [20].

$$A_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \tag{2}$$

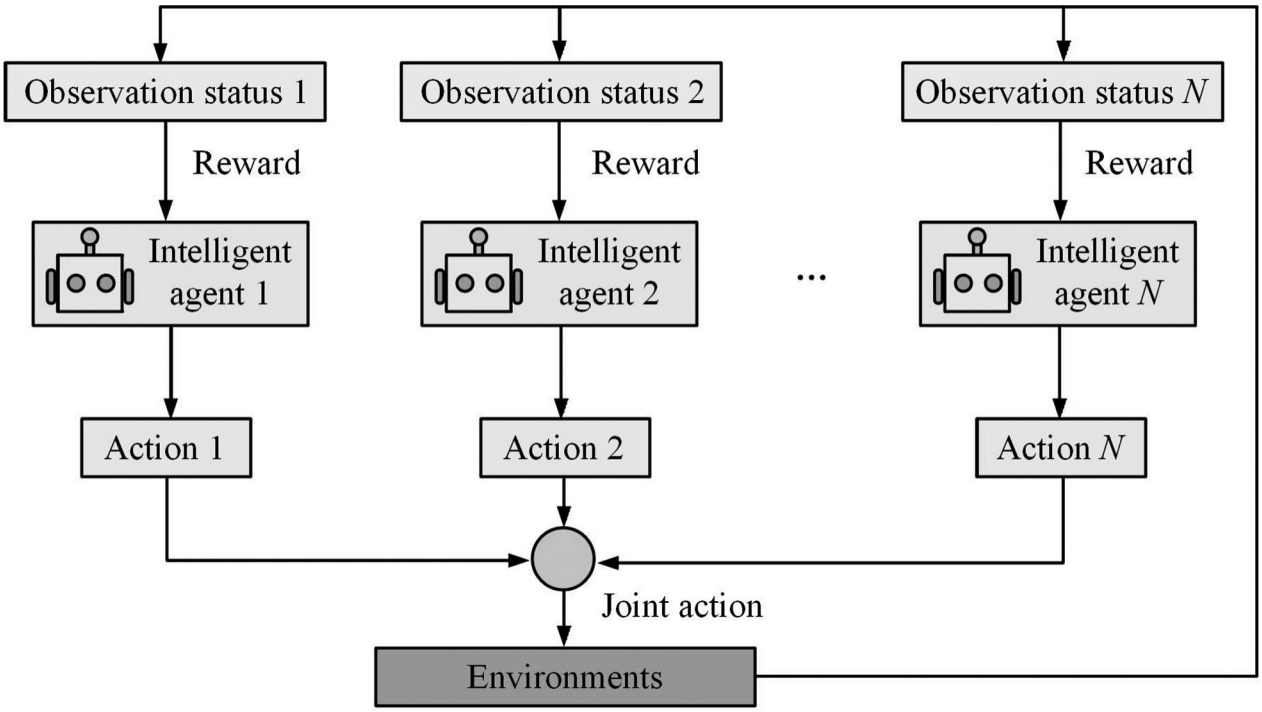


Figure 1. Schematic diagram of Markov game process.

In Equation (2), A_t denotes the estimated value of the AF at TiS t. δ_t is the temporal difference error at TiS t. λ denotes the AF parameter. T indicates the total quantity of TiSs in the trajectory. The calculation of the temporal difference error is shown in Equation (3).

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t) \tag{3}$$

In Equation (3), r_t displays the immediate reward at TiS t. $V_\phi(s_t)$ represents the value function's estimated value for state s_t under parameter ϕ. ϕ denotes the parameters of the value function. Figure 2 displays the actor-critic network's structural diagram.

Figure 2(a) depicts the AN, comprising an input layer (IL), hidden layer (HL), and output layer. Before the output layer creates actions, the HL processes the state information that is sent to the IL. Figure 2(b) shows the CN, whose output layer estimates the state value function to evaluate state quality and guide policy optimisation. To ensure training stability, the study uses the proximal policy optimisation (PPO) technique to limit the step size of policy updates [21]. The PPO objective function (OF) for each agent is shown in Equation (4).

$$L^{\mathrm{CLIP}}(\theta) = E_t\left[\min(p_t(\theta)A_t, \mathrm{clip}(pt(\theta), 1-\varepsilon, 1-\varepsilon)A_t)\right] \tag{4}$$

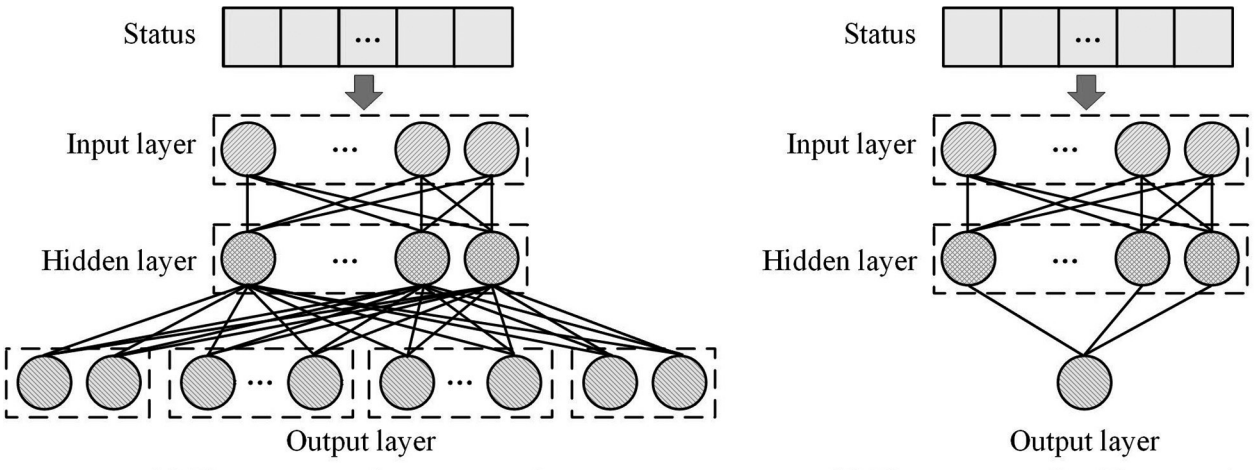
In Equation (4), $L^{\mathrm{CLIP}}(\theta)$ denotes the pruning OF of the PPO. θ are the parameters of the policy network (PN). $p_t(\theta)$ indicates the probability ratio. clip($pt(\theta)$, 1−ε, 1−ε) signifies the pruning function. ε denotes the pruning hyperparameter. The study further trains the CN by minimizing the mean squared error between the predicted and target values of the value function, with its loss function shown in Equation (5).

$$\begin{cases} L(\phi) = Et\left[(V\phi(s_t) - V_{\mathrm{target}}(s_t))^2\right] \\ V_{\mathrm{target}}(s_t) = \sum_{l=0}^{T-t} \gamma^l r_{t+1} \end{cases} \tag{5}$$

In Equation (5), $L(\phi)$ denotes the value loss function. $V\phi(s_t)$ and $V_{\mathrm{target}}(s_t)$ represent the predicted and target values of the value function, respectively. r_{t+1} denotes the immediate reward obtained at TiS t + 1. To encourage the agent to explore unknown state and action spaces, an entropy regularization term is incorporated into the OF, as shown in Equation (6).

$$L^{\mathrm{ENT}}(\theta) = Et\left[H(\pi_\theta(\cdot|s_t))\right] \tag{6}$$

In Equation (6), $L^{\mathrm{ENT}}(\theta)$ denotes the entropy regularization term. H represents the entropy function. $\pi_\theta(\cdot|s_t)$ indicates the action probability distribution generated by policy π under state s_t. By adjusting the weights of different loss terms to balance the relationship among policy per-



(a) The structure of actor network (b) The structure of critic network

Figure 2. Schematic diagram of the structure of actor network and critic network.

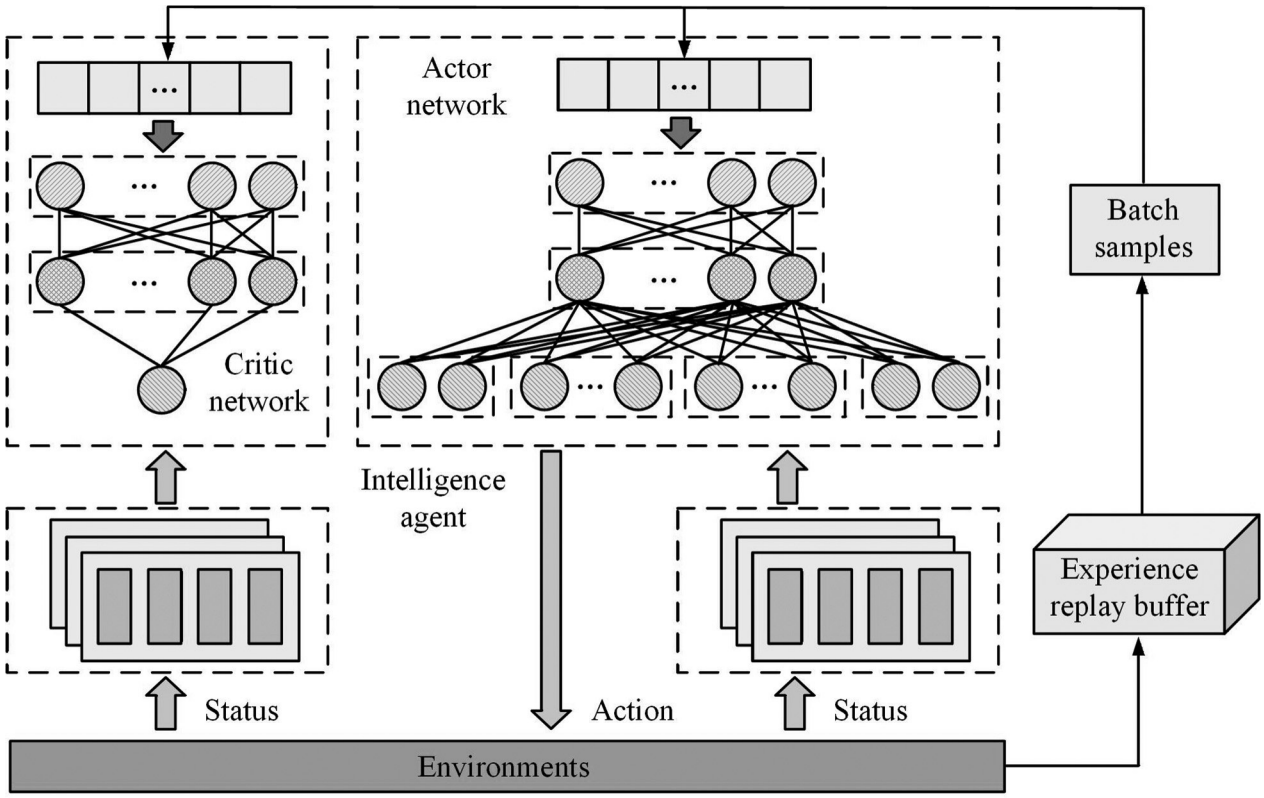


Figure 3. Schematic diagram of VR TS framework based on MARL and PPO algorithm.

formance, value accuracy, and exploration level, the agent's total loss is expressed as shown in Equation (7).

$$L_i(\theta,\phi) = L^{\mathrm{CLIP}}(\theta) - c_1 L(\phi) + c_2 L^{\mathrm{ENT}}(\theta) \qquad (7)$$

In Equation (7), $L_i(\theta, \phi)$ denotes the total loss function of agent i. c_1 represents the value loss weighting hyperparameter. c_2 denotes the entropy regularization weighting hyperparameter. Under the CT and decentralized execution framework, all agents share network parameters but execute policies independently [22]. Global optimisation is achieved by synchronously updating the PN and value network (VN), as shown in Equation (8).

$$\begin{cases} \theta' \leftarrow \theta + \alpha \nabla_\theta \sum_{i=1}^{N} L_i(\theta,\phi) \\ \phi' \leftarrow \phi - \beta \nabla_\phi \sum_{i=1}^{N} L_i(\theta,\phi) \end{cases} \qquad (8)$$

In Equation (8), θ' and ϕ' denote the shared parameters of the PN and VN, respectively. α and β represent the learning rates (LRs) for the PN and VN, respectively. ∇_ϕ displays the gradient with respect to the policy parameters θ. ∇_ϕ displays the gradient with respect to the value parameters ϕ. A schematic diagram of the VR TS framework based on MARL and the PPO algorithm is shown in Figure 3.

In Figure 3, the VR TS framework comprises multiple agents, each of which is equipped with an AN and a CN. Agents gather experience data through interactions with the environment, which is then saved in a CT experience replay buffer. State information serves as input, while action outputs facilitate TS decisions, enabling distributed coordination and efficient resource allocation.

2.2. Policy Enhancement Method Based on GCNs

Although MARL exhibits robust local decision-making capabilities within distributed VR TS, it fails to consider the intricate topological relationships between tasks and users, leading

to inadequate global awareness [23]. To achieve more refined collaborative decision-making, the study incorporates GCNs further, proposing a policy enhancement method based on heterogeneous GCNs. This approach captures and leverages the dynamic topological relationships between tasks and users, thereby improving the agents' global awareness. Within the constructed heterogeneous GCN, the study enhances the expressive power of node representations through the topological structure defined by the adjacency matrix (AM) [24]. In the research graph, the node feature matrix (FM) is denoted as $\boldsymbol{H}^{(0)} \in R^{M \times F}$, and the AM $\boldsymbol{A} \in R^{M \times M}$ represents the connection relationships between nodes. Among them, M displays the quantity of nodes, and F is the feature dimension. To maintain numerical stability, the AM undergoes normalization processing as shown in Equation (9).

$$\boldsymbol{A'} = \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{A} \boldsymbol{D}^{-\frac{1}{2}} \tag{9}$$

In Equation (9), $\boldsymbol{A'}$ denotes the normalized AM. $\boldsymbol{D}$ denotes the degree matrix. $\boldsymbol{D}^{-\frac{1}{2}}$ denotes the diagonal matrix. The aggregation process of node features is expressed in Equation (10).

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\boldsymbol{A'} \boldsymbol{H}^{(l)} \boldsymbol{W}^{(l)}\right) \tag{10}$$

In Equation (10), $\boldsymbol{H}^{(l+1)}$ displays the FM of nodes in layer l+1. $\boldsymbol{W}^{(l)}$ displays the trainable weight matrix of layer l. σ displays the nonlinear activation function. A schematic diagram of the aggregation of node features within the region is shown in Figure 4.

Figure 4 illustrates the feature aggregation process in a GCN, where a value of 1 in the AM indicates a connection between nodes, and the FM represents the initial features of each node. Through the topology specified by the AM, the graph convolution operation (GCO) combines each node's features with those of its neighbours. This study uses the GCO to aggregate the features of surrounding nodes and update the representation of the current node to efficiently extract node features from the graph structure and facilitate information propagation [25]. Its output is shown in Equation (11).

$$h_u^{(l+1)} = \sigma\left(\sum_{v \in X(u)} \frac{1}{c_{vu}} W^{(l)} h_v^{(l)}\right) \tag{11}$$

In Equation (11), $h_u^{(l+1)}$ denotes the FR of node u at layer l+1. $X(u)$ represents the set of neighbouring nodes of node u. c_{vu} denotes the normalization constant. $h_v^{(l)}$ denotes the FR of neighbouring node v at layer l. Considering the heterogeneity of nodes and relationships in real-world systems, the study further introduces a heterogeneous GCN to distinguish different types of nodes and edges, as shown in Equation (12).

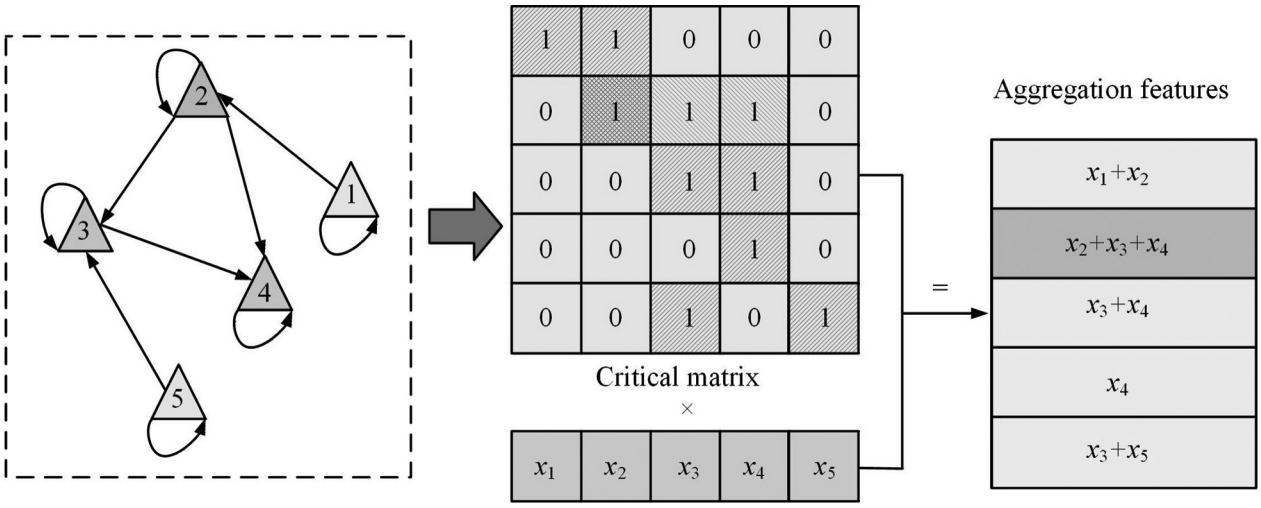


Figure 4. Schematic diagram of aggregation of node features within the region.

$$h_u^{(l+1)} = \sigma\left(\sum_{e\in E}\sum_{v\in X_e(u)}\frac{1}{c_{ue}}W_e^{(l)}h_v^{(l)}\right) \quad (12)$$

In Equation (12), E denotes the set of all relation types in the graph. $X_e(u)$ is the set of neighbour nodes connected to node u via relation r. To dynamically learn the importance of different neighbour nodes, the study introduces a graph attention mechanism and normalizes the attention coefficients (ACs) using the softmax function, as shown in Equation (13).

$$\begin{cases} b_{uv} = a(\boldsymbol{W}h_u, \boldsymbol{W}h_v) \\ a_{uv} = \exp(b_{uv}) / \left(\sum_{k\in X(u)}\exp(b_{uk})\right) \end{cases} \quad (13)$$

In Equation (13), b_{uv} is the AC between node u and v. a_{uv} represents the normalized attention weight. a denotes the shared attention function. b_{uk} denotes the AC between node u and its neighbour node k. Through multi-layer graph convolutions and attention mechanisms, each node obtains enhanced FRs [26]. These features are concatenated with the agent's original observations to form the augmented state input, as shown in Equation (14).

$$s_i' = \text{concat}(s_i, h_u^{(l)}) \quad (14)$$

In Equation (14), s_i' denotes the augmented state input. $h_u^{(l)}$ represents the augmented FR of a node u processed by the l layer graph attention network. To enhance the quality of FRs in GCNs, this study introduces a graph structure reconstruction loss during training, as shown in Equation (15).

$$L_{\text{GCN}} = \sum_{(u,v)\in X}\left\|h_u^{(l)} - h_v^{(l)}\right\|^2 \quad (15)$$

In Equation (15), L_{GCN} denotes the value of the graph structure reconstruction loss function. $h_u^{(l)}$ and $h_v^{(l)}$ represent the final FRs of nodes u and v after passing through the layer GCN. Integrating GCN outputs into MARL policy updates is a critical step in the proposed framework. GCN processes state information and captures dynamic topological relationships between tasks and users. The enhanced FRs obtained from the GCN are then concatenated with the original state information to create an augmented state input for each agent. This augmented state input is then used by the actor and CNs to make decisions and evaluate states, respectively. Specifically, the GCN outputs are incorporated into the policy updates as follows.

1. The state information for each agent is fed into the GCN to produce enhanced FRs.
2. These enhanced FRs are then concatenated with the original state information to form the augmented state input.
3. The augmented state input is used by the AN to select actions, and by the CN to estimate state values.
4. The CN's estimates are used to calculate the AF, and the AN is updated using the PPO OF, which incorporates the augmented state input and the calculated AF.

The computational complexity of the proposed MARL–GCN framework depends on the complexity of the MARL and GCN components. The MARL component involves training multiple actors and CNs, and its complexity is primarily determined by the number of agents, the size of the state and action spaces, and the number of iterations. The GCN component comprises GCOs and attention mechanisms. The complexity of this component depends on the number of nodes and edges in the graph, as well as the number of convolutional layers. Figure 5 illustrates the schematic diagram of MUCTS integrating MARL with a heterogeneous GCN.

In Figure 5, the MUCTS framework acquires state information through agent-environment interactions, while the experience replay buffer stores samples for CT. The GCN performs multi-layer aggregation and pooling on node features to enhance global perception capabilities. The actor-critic network achieves efficient, distributed TS optimisation through collaborative updates.

An algorithmic pseudocode the MARL-GCN training workflow is as follows.

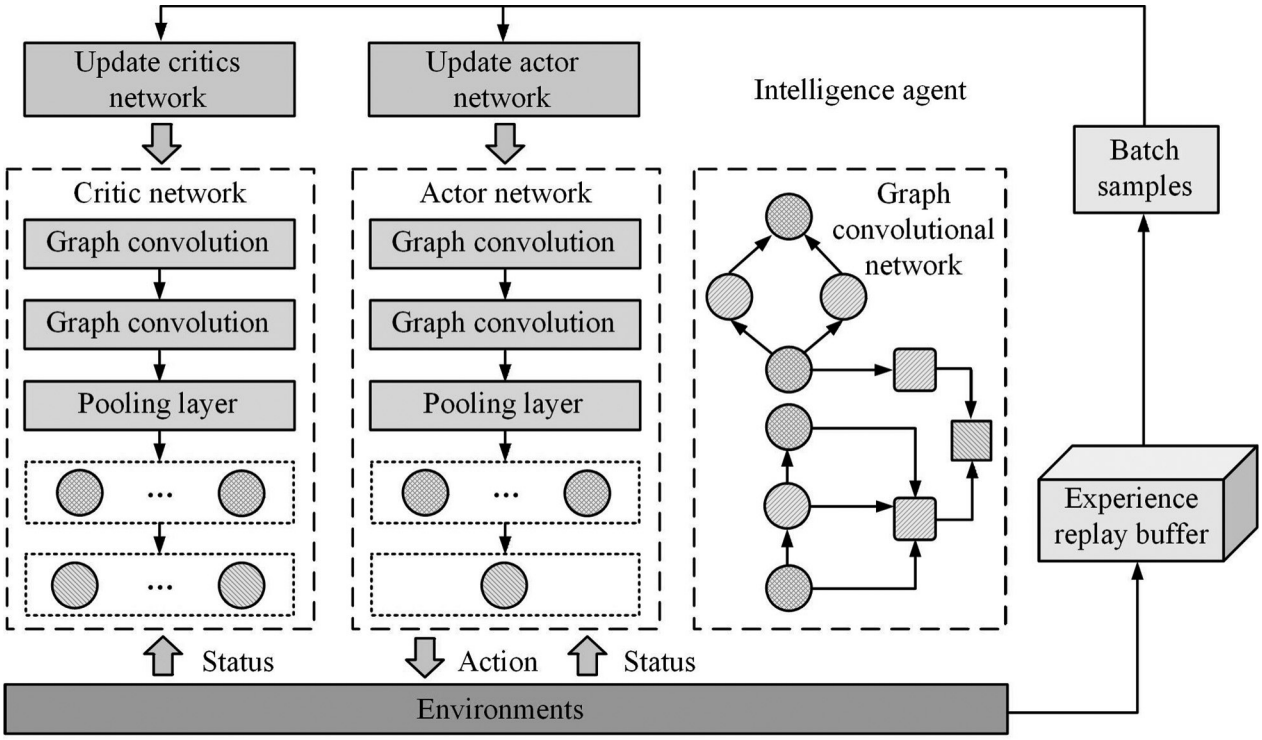


Figure 5. MUCTS integrating MARL and heterogeneous GCNs.

Algorithm 1. MARL-GCN Training Procedure.

Input: Task graph G = (V, E), Number of agents N, Training episodes E
Output: Trained actor and critic networks

1. Initialize actor networks π_i and critic networks V_i for all agents i ∈ [1, N]
2. Initialize GCN with parameters θ_G
3. Initialize experience replay buffer D
4. **for** episode = 1 to E do
5. Reset environment and observe initial state s_0
6. **for** t = 1 to T do
7. **for** each agent i do
8. Extract local observation o_i^t from s_t
9. Compute graph-enhanced state s_i^aug = [o_i^t, GCN(o_i^t; θ_G)]
10. Sample action a_i^t ~ π_i(·|s_i^aug)
11. **end for**
12. Execute joint action a^t = (a_1^t, ..., a_N^t)
13. Observe reward r^t and next state s_{t+1}
14. Store transition (s_t, a^t, r^t, s_{t+1}) in D
15. **end for**
16. **for** each training step do
17. Sample mini-batch from D
18. Update GCN using graph reconstruction loss
19. Update actor and critic networks using PPO loss and value loss
20. **end for**
21. **end for**

3. Comprehensive Analysis of a Novel MUCTS Optimisation Model

Through analysis, the suggested MUCTS optimisation model's performance is confirmed. To confirm the model's practical deployability, stability, and generalization ability in a setting that approximates distributed VR applications in the real world, simulation tests are conducted.

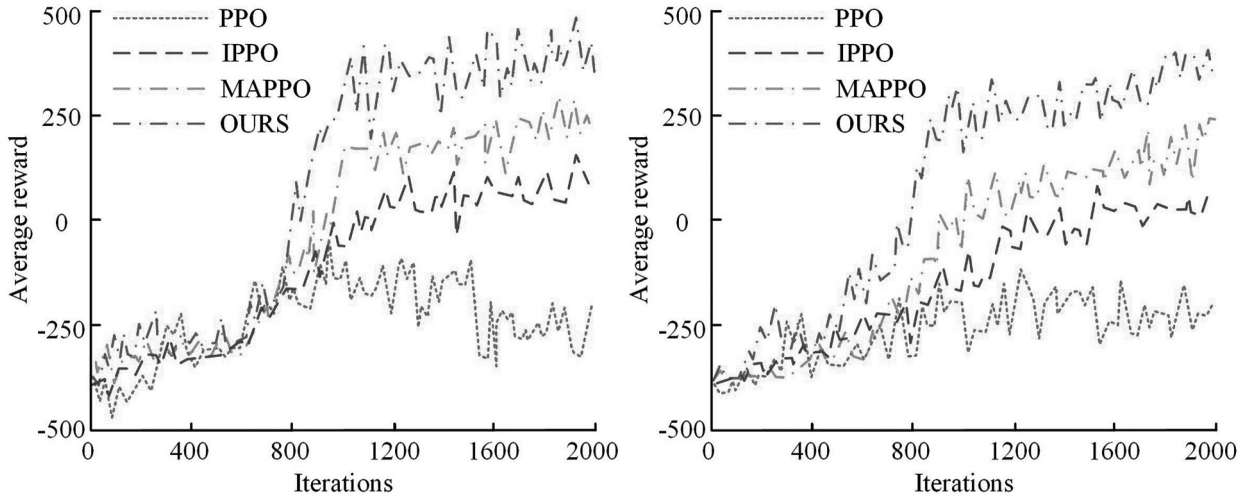
3.1. Performance Validation Analysis of a Novel MUCTS Optimisation Model

To validate the performance of the proposed model, experiments are conducted on two open-source datasets: the Alibaba Cluster Trace Program (ACTP) and VRCollaborate. The ACTP dataset comprises 12,548 task instances with diverse resource requirements. These include CPU-intensive rendering tasks (42%), memory-bound data synchronisation tasks (35%), and I/O-intensive interaction tasks (23%). User activity follows a diurnal pattern, reaching a peak of 280 simultaneous users. The VRCollaborate dataset comprises 8,932 collaborative tasks of varying complexity. Task durations range from 50 ms to 2.5 s, and resource demands span 2–16 CPU cores with memory requirements of 4–32 GB. The study compares the proposed model with existing mainstream MARL models based on PPO, including traditional PPO, independent proximal policy optimisation (IPPO), and MAPPO. The experiment is built using Python 3.9, PyTorch 1.13, and the Stable-Baselines3 framework. The runtime environment is Ubuntu 20.04 equipped with an NVIDIA RTX 3090 GPU. Table 1 lists the experimental parameters.

The study first conducts a comparative analysis of the average reward across different models using the ACTP and VRCollaborate datasets. In Figure 6(a), for the ACTP dataset, at 1200 iterations, the average rewards for PPO, IPPO, and MAPPO are −227.30, 17.62, and 208.57. The average reward for the proposed model is 335.24. At 2000 iterations, the average rewards for the four models are −212.15, 48.21, 242.36, and 382.17, respectively. In Figure 6(b), for the VRCollaborate dataset, the average rewards at iteration 1200 are −276.52 for PPO, −108.54 for IPPO, 62.36 for MAPPO, and 252.76 for the proposed model. The findings reveal the efficacy of the suggested model by displaying that it produces better rewards on both datasets. The superior convergence observed in Figure 6 is attributed to the GCN-enhanced state representations, which give the agents structural awareness of the task-user interaction patterns. This topological encoding allows for more efficient policy learning by identifying latent dependencies that are inaccessible to traditional MARL methods.

Table 1. Experimental parameter settings.

Parameter names	Parameter values	Parameter names	Parameter values
LR (Actor)	0.0003	Batch size	512
LR (Critic)	0.001	Experience replay buffer size	106
Discount factor	0.99	Number of convolutional layers in the graph	2
GAE parameters	0.95	HL dimension	128
PPO cutting coefficient	0.20	Iterations	2000

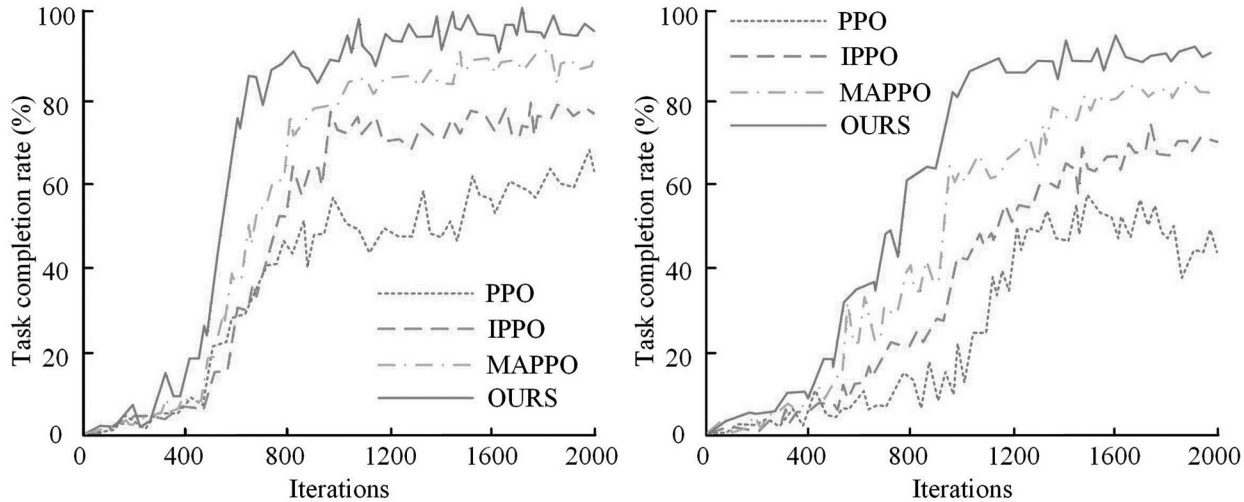


(a) The average reward of different models on the ACTP dataset

(b) The average reward of different models on the VRCollaborate dataset

Figure 6. Average rewards of different models on ACTP and VRCollaborate datasets.

The study further conducts a comparative analysis of task completion rates across different models using two open-source datasets. In Figure 7(a), for the ACTP dataset, when the iteration count reaches 1200, the task completion rates for PPO, IPPO, MAPPO, and the proposed model are 50.05%, 69.24%, 85.36%, and 93.87%, respectively. When the iteration count reached 2000, the task completion rates increased to 59.69%, 77.62%, 88.21%, and 94.52%, respectively. In Figure 7(b), for the VRCollaborate dataset, the task completion rates of the four models at 1200 iterations are 41.35%, 55.64%, 62.38%, and 89.69%, respectively. When the iteration count increases to 2000, the completion rates are 45.84%, 68.27%, 80.06%, and 92.39%, respectively. The results demonstrate that the proposed model achieves a higher task completion rate and greater stability in MUCTS. In Figure 7, the higher task completion rates demonstrate how the integration of graph structural priors facilitates better resource coordination, particularly in handling interdependent tasks that require synchronized execution across multiple users.



(a) Task completion rates of different models on the ACTP dataset

(b) Task completion rates of different models on the VRCollaborate dataset

Figure 7. Task completion rates of different models on ACTP and VRCollaborate datasets.

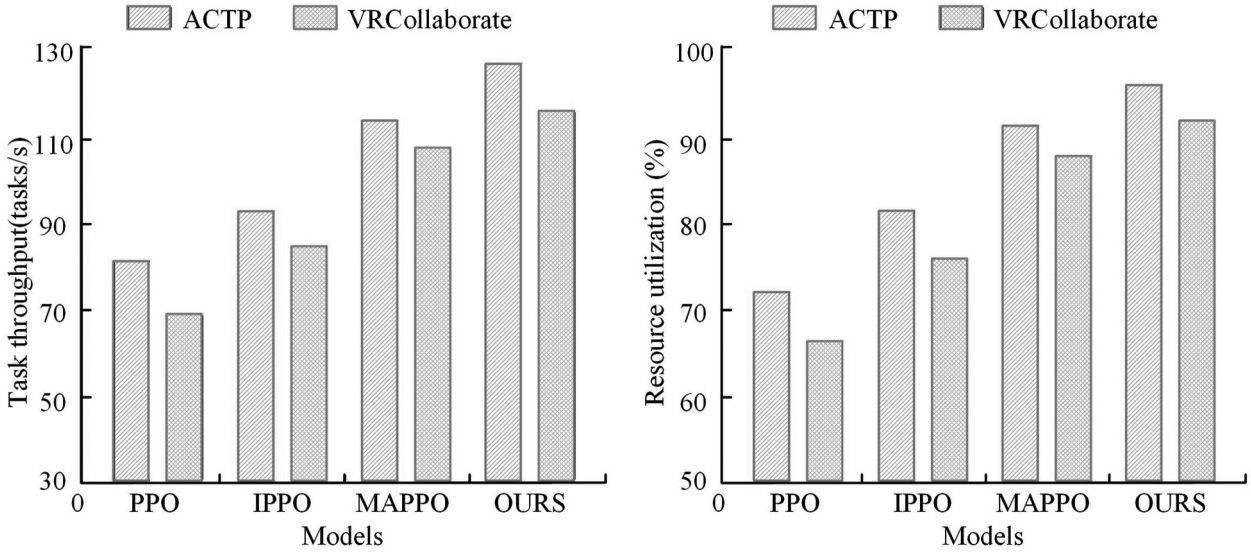
Ablation experiments are conducted to systematically evaluate the contributions of each component in the proposed MARL-GCN framework. The results are shown in Table 2. In Table 2, the complete model performs best on all evaluation metrics, achieving a task completion rate of 94.52%, an average reward of 382.17, a throughput of 125.86 tasks per second and a resource utilisation rate of 96.53%±1.08%. Removing the GCN component significantly decreases performance, with the task completion rate dropping from 91.7% to 85.36%. This indicates that GCN plays a critical role in improving model performance. After removing the attention mechanism, all indicators showed a moderate decrease, with an average reward decrease of 66.75%. This suggests that the attention mechanism can effectively enhance the quality of decisions. Although the impact of removing the structural loss function is relatively small, it still reduces the task completion rate by 2.69 percentage points, which verifies the stabilising effect of this component on model performance. GCN components contribute most to performance improvement overall, followed by attention mechanisms. The structural loss function mainly serves as an additional optimisation tool.

A comparison analysis of throughput and resource consumption across various models is carried out on two datasets to assess the effectiveness of the suggested model in terms of system efficiency and resource economy. In Figure 8(a), the task throughputs of PPO, IPPO, and MAPPO on the ACTP dataset are 82.27 tasks/s, 92.16 tasks/s, and 114.62 tasks/s, respectively. On the VRCollaborate dataset, their throughputs are 68.21 tasks/s, 83.57 tasks/s, and 108.10 tasks/s, respectively. In comparison, the proposed model achieves throughputs of 125.86 tasks/s and 116.73 tasks/s on the two datasets. In Figure 8(b), the resource utilisation rates of the four models in the ACTP dataset are 72.17%, 81.54%, 91.02%, and 96.53%, respectively. In the VRCollaborate dataset, the resource utilisation rates are 65.84%, 75.24%, 87.65%, and 91.87%. The results indicate that the suggested paradigm can improve system throughput while making better use of computational resources. The throughput improvements shown in Figure 8 are directly linked to the model's ability to prioritise critical paths in the task graph dynamically using attention mechanisms. This capability enables more intelligent load balancing and resource allocation, thereby reducing contention in high-concurrency scenarios.

Table 2. Ablation study results (Mean±95% CI).

Model variant	Task Completion rate (%)	Average reward	Throughput (tasks/s)	Resource utilisation (%)
Full Model (MARL+GCN)	94.52 ± 1.23	382.17 ± 8.45	125.86 ± 2.67	96.53 ± 1.08
w/o GCN (MARL only)	85.36 ± 2.14	242.36 ± 12.37	114.62 ± 3.82	91.02 ± 2.15
w/o attention mechanism	89.27 ± 1.87	315.42 ± 9.83	119.35 ± 3.15	93.76 ± 1.64
w/o structural loss	91.83 ± 1.65	348.26 ± 8.92	122.14 ± 2.89	95.12 ± 1.37

Note: Results are reported on ACTP dataset after 2000 training iterations.



(a) Task throughput of different models on two datasets

(b) Resource utilization of different models on two dataset

Figure 8. Throughput and resource utilisation of different models on ACTP and VRCollaborate datasets.

The study compares the running times of different models in the ACTP dataset, conducting a statistical analysis in the process. The results are shown in Table 3. On the ACTP dataset, the task completion rate of OURS is 94.52%±1.23%. This is 6.31%, 16.90%, and 34.83% higher than the rates for MAPPO, IPPO and PPO, respectively. Statistical tests shows significant differences ($p < 0.001$). Similar performance advantages are observed on the VRCollaborate dataset. However, this performance improvement comes at the cost of increased computational requirements: the training time of the proposed model is approximately 36% longer than that of MAPPO, and it also uses more memory. This reflects the trade-off between model complexity and performance, showing that the proposed method delivers a more significant performance improvement by incurring reasonable additional computational cost.

A sensitivity analysis of parameters is conducted to evaluate the impact of model hyperparameter settings on performance, and the results are shown in Table 4. The best performance is achieved on both the ACTP and VRCollaborate datasets when the LR is set to 0.0003: average rewards are 382.17 and 252.76, respectively, and task completion rates are 94.52% and 92.39%, respectively. Increasing the LR to 0.001 resulted in a decrease in the average rewards of the two datasets to 365.21 and 238.45, respectively. This indicates that excessive LRs can affect training stability. The effect is optimal when the number of GCN layers is 2. Reducing or increasing this number will result in a slight decrease in performance. The optimal balance is achieved with 2 attention heads, and increasing to 4 heads offers limited improvement. Setting the HL dimension to 128 yields optimal performance. At 64 dimensions, the average rewards for the two datasets decrease to 375.63 and 242.17, respectively. Increasing to 256 dimensions only brings marginal improvement. These results indicate that the model's optimal performance configuration is achieved with a LR of 0.0003, 2 GCN layers, 2 attention heads, and HL dimensions of 128. The model also exhibits good robustness to parameter changes.

3.2. Analysis of Simulation Application Results for a Novel MUCTS Optimisation Model

To further validate the model's performance in practical application scenarios, a distributed VR simulation platform is constructed to simulate two application scenarios: multi-user collaborative design and large-scale multiplayer online games. The VR collaborative design platform

Table 3. Runtime Performance and Statistical Analysis.

Models	Datasets	Training Time (h)	Inference Latency (ms)	Memory Usage (GB)	Task Completion Rate (%)	p-value
OURS	ACTP	8.52 ± 0.32	4.23 ± 0.18	3.82 ± 0.21	94.52 ± 1.23	/
MAPPO		6.24 ± 0.25	2.14 ± 0.12	2.42 ± 0.13	88.21 ± 1.87	< 0.001
IPPO		5.83 ± 0.27	1.82 ± 0.10	2.15 ± 0.15	77.62 ± 2.35	< 0.001
PPO		4.12 ± 0.21	1.23 ± 0.08	1.63 ± 0.11	59.69 ± 2.87	< 0.001
OURS	VRCollaborate	7.86 ± 0.28	3.91 ± 0.15	3.45 ± 0.18	92.39 ± 1.41	/
MAPPO		5.93 ± 0.23	1.96 ± 0.11	2.28 ± 0.12	80.06 ± 2.13	< 0.001
IPPO		5.47 ± 0.24	1.73 ± 0.09	2.03 ± 0.14	68.27 ± 2.64	< 0.001
PPO		3.89 ± 0.19	1.15 ± 0.07	1.52 ± 0.10	45.84 ± 3.12	< 0.001

Table 4. Runtime Performance and Statistical Analysis.

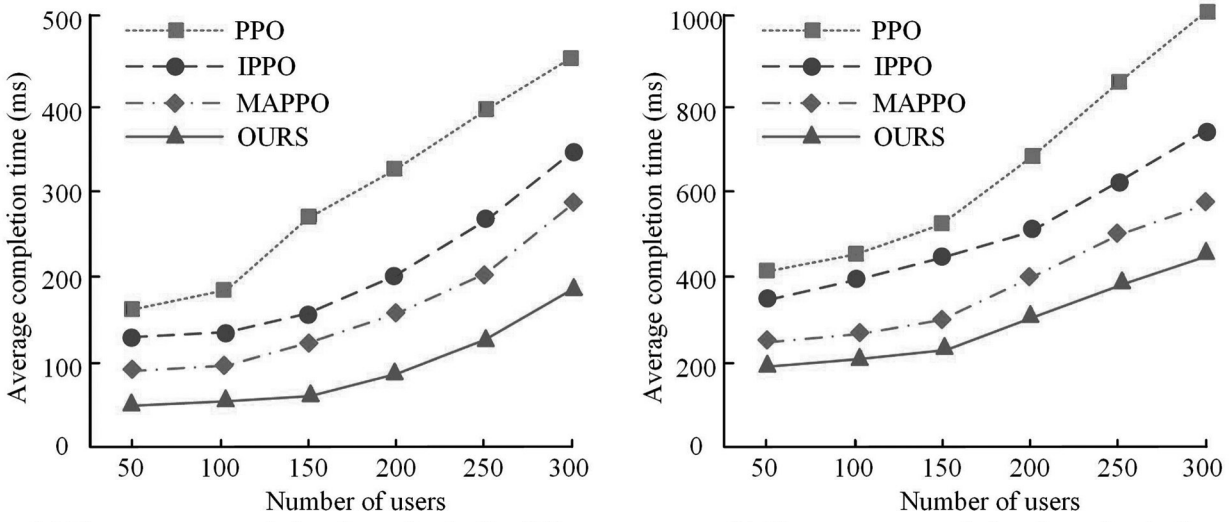
Parameters	Values	Average reward (ACTP)	Average reward (VRCollaborate)	Task completion rate (ACTP)	Task completion rate (VRCollaborate)
Learning rate	0.0001	375.63	245.18	93.85	90.12
	0.0003	382.17	252.76	94.52	92.39
	0.001	365.21	238.45	92.71	89.67
GCN layers	1	372.45	240.89	93.50	90.05
	2	382.17	252.76	94.52	92.39
	3	378.92	248.33	94.18	91.74
Attention heads	1	378.92	247.15	94.10	91.55
	2	382.17	252.76	94.52	92.39
	4	381.05	251.42	94.43	92.21
Hidden layer Dim	64	375.63	242.17	93.90	90.38
	128	382.17	252.76	94.52	92.39
	256	380.49	250.91	94.35	92.08

simulates a collaborative design environment in VR. This large-scale, multiplayer online VR game incorporates dynamic task generation, real-time path planning, and resource contention into a gaming scenario. These features test the system's real-time capabilities and collaborative scheduling abilities.

The study conducts a comparative analysis of the average task completion time (ATCT) across different models under varying user numbers in two simulation scenarios. In Figure 9(a), within the VR collaborative design platform simulation, when the number of users is 50, the ATCTs for PPO, IPPO, and MAPPO are 171.25 ms, 136.75 ms, and 98.24 ms, respectively. The average completion time (ACT) for the proposed model is 50.66ms. When the number of users reached 300, the ACTs for the four models were 452.71 ms, 347.58 ms, 294.37 ms, and 186.35 ms, respectively. In Figure 9(b), during the multiplayer online VR game simulation with 300 users, the ATCTs for PPO, IPPO, and MAPPO are 996.25 ms, 923.58 ms, and 571.59 ms, respectively, while the ACT for the proposed model is 423.62 ms. The results display that the proposed model maintains low latency and high stability even under high-concurrency, dynamic task environments. Figure 9 shows that the GCN's ability to anticipate bottlenecks and schedule tasks proactively reduces latency. This demonstrates how topological foresight can enhance the real-time performance of distributed VR environments.

The study further conducts a comparative analysis of the average user latency across different models under varying numbers of subtasks in two simulation scenarios. In Figure 10(a), during the simulation on the VR collaborative design platform, when the number of subtasks is 5, the average user latency for PPO, IPPO, MAPPO, and the proposed model is 17.25 ms, 13.86 ms, 9.82 ms, and 4.17 ms, respectively. When the number of subtasks reaches 30, the latency times are 39.87 ms, 32.35 ms, 23.62 ms, and 11.82 ms, respectively. In Figure 10(b), during the multiplayer online VR game simulation with 30 tasks, the average user latency for the four models is 70.03 ms, 63.16 ms, 43.37 ms, and 26.21 ms, respectively. The findings demonstrate that when processing jobs in multi-user scenarios, the suggested approach can more successfully lower user latency and provide a more seamless user experience. The consistent reduction in user latency across varying subtask loads demonstrates the robustness of the framework in maintaining quality of service under dynamic workloads. This is made possible by the heterogeneous GCN architecture, which can adapt to different node types and relationship patterns.

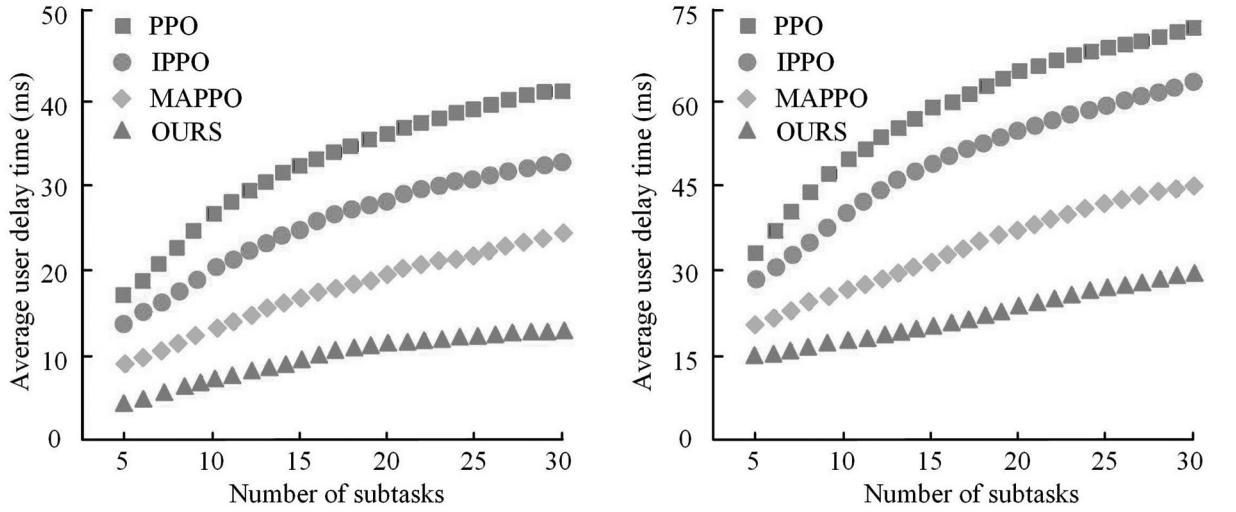
Additionally, the study conducts a comparative analysis of the task processing success rate across different models under varying numbers



(a) The average completion time of tasks for different models on VR collaborative design platform

(b) The average completion time of tasks for different models on multiplayer online VR games

Figure 9. The ACT of tasks for different models on two simulation scenarios.

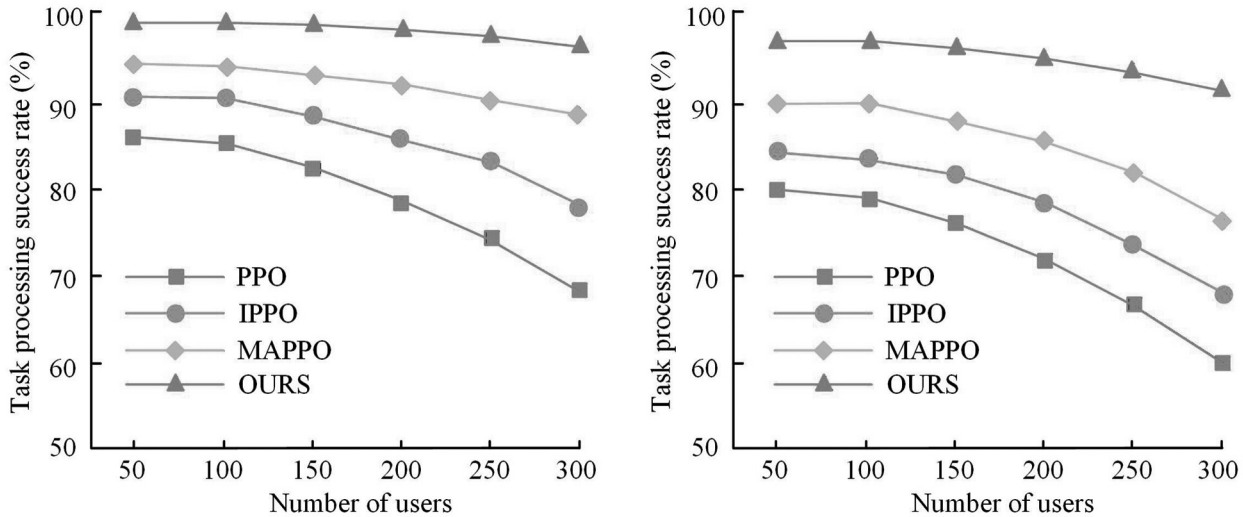


(a) The average user delay time of different models on the VR collaborative design platform

(b) The average user delay time of different models in multiplayer online VR games

Figure 10. The average user delay time of different models on two simulation scenarios.

of users in two simulation scenarios. In Figure 11(a), during the simulation of the VR collaborative design platform, when the user's quantity reaches 50, the task processing success rates for PPO, IPPO, MAPPO, and the proposed model are 86.12%, 91.35%, 94.27%, and 98.26%, respectively. When the user's quantity reaches 300, the task processing success rates for the four models are 67.84%, 76.98%, 89.27%, and 96.89%, respectively. In Figure 11(b), during the multiplayer online VR game simulation, when the number of users , the task processing success rates for PPO, IPPO, and MAPPO are 58.36%, 67.24%, and 76.87%, respectively. The suggested model has a 93.35% task processing success rate. The results demonstrate that the proposed model achieves a higher task processing success rate in multi-user environments and can more effectively handle collaborative tasks. Figure 11 shows that the model has high task processing success rates, especially under heavy user loads. This demonstrates the model's reliability in preserving system stability through effective overload prevention and recovery mechanisms.



(a) Task processing success rate of different models on the VR collaborative design platform

(b) Task processing success rate of different models in multiplayer online VR games

Figure 11. Task processing success rate of different models on two simulation scenarios.

4. Conclusion

This study established a novel, multi-user, collaborative TS framework by integrating MARL with GCNs. The primary theoretical contribution was the advancement of conventional scheduling theory through the incorporation of relational inductive biases into multi-agent policy learning. This enabled the formalisation of complex topological dependencies between tasks and users within distributed VR systems. Technically, the MARL-GCN integration was a significant innovation. Graph attention mechanisms facilitate dynamic relationship weighting, and structural reconstruction loss enhances global representation learning. These two features worked together to overcome the coordination limitations in partially observable environments.

The proposed framework was particularly valuable for organisations operating under high concurrency and resource constraints. Specifically, VR collaborative design platforms could use this approach to keep multiple users manipulating complex 3D models coordinated, and large-scale multiplayer online games could use it to allocate resources dynamically during peak user activity. The system's ability to dynamically prioritise critical interactions enabled efficient task allocation, even under severe computational constraints. This made it equally suitable for distributed training systems and industrial digital twins, where resource awareness was crucial.

There are several limitations that require acknowledgement. First, the current validation is conducted primarily in simulation environments, which may not fully capture real-world network volatility and device heterogeneity. Therefore, future work will focus on implementing physical testbed deployments and extending the framework's capabilities to support adaptive scheduling across heterogeneous edge devices. Further research will explore the use of transfer learning for cross-domain applications, as well as incorporating human-in-the-loop interactions to create more intuitive collaborative VR experiences.

In conclusion, this research advances the field of intelligent computing and distributed system coordination by establishing a new paradigm for graph-enhanced multi-agent learning. Integrating structural reasoning with decentralised decision-making provides theoretical insights and practical solutions for creating responsive, scalable, and user-aware scheduling systems for next-generation collaborative platforms.

Declaration of Competing Interests

The authors declare no conflict of interest.

Funding

This research was supported by the Natural Science Research Project of Anhui Educational Committee (China) [Grant No. 2023AH053102] for the project "Investigating the Use of Distributed Virtual Reality in Practical Instruction." No other specific funding or financial support was received from any institutions, foundations, or third parties.

Data Availability

Data used in this study are proprietary.

References

[1] A. Barrett *et al.*, "Technology Acceptance Model and Multi-user Virtual Reality Learning Environments for Chinese Language Education", *Interactive Learning Environments*, vol. 31, no. 3, pp. 1665–1682, 2023. http://dx.doi.org/10.1080/10494820.2020.1855209

[2] A. Schafer *et al.*, "A Survey on Synchronous Augmented, Virtual, and Mixed Reality Remote Collaboration Systems", *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–27, 2022. http://dx.doi.org/10.1145/3533376

[3] Y. Yang *et al.*, "Decentralized Cooperative Caching and Offloading for Virtual Reality Task Based on GAN-Powered Multi-Agent Reinforcement Learning", *IEEE Transactions on Services Computing*, vol. 17, no. 1, pp. 291–305, 2023. http://dx.doi.org/10.1109/TSC.2023.3347741

[4] K. Liu *et al.*, "Self-Attention-Based Multi-Agent Continuous Control Method in Cooperative Environments", *Information Sciences*, vol. 585, pp. 454–470, 2022. http://dx.doi.org/10.1016/J.INS.2021.11.054

[5] S. Wang *et al.*, "Consensus-Based Decentralized Task Allocation for Multi-Agent Systems and Simultaneous Multi-Agent Tasks", *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12593–12600, 2022. http://dx.doi.org/10.1109/LRA.2022.3220155

[6] D. Johnson *et al.*, "Multi-Agent Reinforcement Learning for Real-Time Dynamic Production Scheduling in a Robot Assembly Cell", *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7684–7691, 2022. http://dx.doi.org/10.1109/LRA.2022.3184795

[7] A. Jayanetti *et al.*, "Multi-Agent Deep Reinforcement Learning Framework for Renewable Energy-Aware Workflow Scheduling on Distributed Cloud Data Centers", *IEEE Transactions on Parallel and Distributed Systems*, vol. 35, no. 4, pp. 604–615, 2024. http://dx.doi.org/10.1109/TPDS.2023.3300962

[8] M. L. Betalo *et al.*, "Multi-Agent Deep Reinforcement Learning-Based Task Scheduling and Resource Sharing for O-RAN-Empowered Multi-UAV-Assisted Wireless Sensor Networks", *IEEE Transactions on Vehicular Technology*, vol. 73, no. 7, pp. 9247–9261, 2023. http://dx.doi.org/10.1109/TVT.2023.3330661

[9] Z. Lin *et al.*, "Satellite-Terrestrial Coordinated Multi-Satellite Beam Hopping Scheduling Based on Multi-Agent Deep Reinforcement Learning", *IEEE Transactions on Wireless Communications*, vol. 23, no. 8, pp. 10091–10103, 2024. http://dx.doi.org/10.1109/TWC.2024.3368689

[10] Y. Zhang *et al.*, "Multistep Multi-Agent Reinforcement Learning for Optimal Energy Schedule Strategy of Charging Stations in Smart Grid", *IEEE Transactions on Cybernetics*, vol. 53, no. 7, pp. 4292–4305, 2022. http://dx.doi.org/10.1109/TCYB.2022.3165074

[11] B. Qin *et al.*, "DGCQN: A RL and GCN Combined Method for DAG Scheduling in Edge Computing", *The Journal of Supercomputing*, vol. 80, no. 13, pp. 18464–18491, 2024. http://dx.doi.org/10.1007/s11227-024-06140-7

[12] X. Jing *et al.*, "Multi-Agent Reinforcement Learning Based on Graph Convolutional Network for Flexible Job Shop Scheduling", *Journal of Intelligent Manufacturing*, vol. 35, no. 1, pp. 75–93, 2024. http://dx.doi.org/10.1007/s10845-022-02037-5

[13] L. Yang *et al.*, "Joint Routing and Scheduling Optimization in Time-Sensitive Networks Using Graph-Convolutional-Network-Based Deep Reinforcement Learning", *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23981–23994, 2022. http://dx.doi.org/10.1109/JIOT.2022.3188826

[14] Z. Liu *et al.*, "Graph Neural Network Meets Multi-Agent Reinforcement Learning: Fundamentals, Applications, and Future Directions", *IEEE Wireless Communications*, vol. 31, no. 6, pp. 39–47, 2024. http://dx.doi.org/10.1109/MWC.015.2300595

[15] S. Yang *et al.*, "A Multi-Policy Deep Reinforcement Learning Approach for Multi-Objective Joint Routing and Scheduling in Deterministic Networks", *IEEE Internet of Things Journal*, vol. 11, no. 10, pp. 17402–17418, 2024. http://dx.doi.org/10.1109/JIOT.2024.3358403

[16] D. Xiao *et al.*, "A Two-Stage GCN-Based Deep Reinforcement Learning Framework for SFC Embedding in Multi-Datacenter Networks", *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4297–4312, 2023. http://dx.doi.org/10.1109/TNSM.2023.3284293

[17] W. Luo *et al.*, "Distributed Computing Optimization in Unmanned Aerial Vehicle Swarm Cooperative Networks", *Journal of Computing and Information Technology*, vol. 31, no. 4, pp. 203–218, 2023. http://dx.doi.org/10.20532/cit.2023.1005769

[18] C. Xu *et al.*, "Digital Twin-Driven Collaborative Scheduling for Heterogeneous Task and Edge-End Resource via Multi-Agent Deep Reinforcement Learning", *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3056–3069, 2023. http://dx.doi.org/10.1109/JSAC.2023.3310066

[19] Y. Wan *et al.*, "Multi-Agent DRL-Based Data-Driven Approach for PEVs Charging/Discharging Scheduling in Smart Grid", *Journal of the Franklin Institute*, vol. 359, no. 4, pp. 1747–1767, 2022. http://dx.doi.org/10.1016/j.jfranklin.2022.01.016

[20] V. N. D. Almeida *et al.*, "Knowledge Transfer in Multi-Objective Multi-Agent Reinforcement Learning via Generalized Policy Improvement", *Computer Science and Information Systems*, vol. 21, no. 1, pp. 335–362, 2024. http://dx.doi.org/10.2298/CSIS221210071A

[21] Z. Jiang *et al.*, "A Graph-Based PPO Approach in Multi-UAV Navigation for Communication Coverage", *International Journal of Computers Communications & Control*, vol. 18, no. 6, p. 5505, 2023. http://dx.doi.org/10.15837/ijccc.2023.6.5505

[22] Y. He *et al.*, "High-Frequency Quantitative Trading of Digital Currencies Based on Fusion of Deep Reinforcement Learning Models with Evolutionary Strategies", *Journal of Computing and Information Technology*, vol. 32, no. 1, pp. 33–45, 2024. http://dx.doi.org/10.20532/cit.2024.1005825

[23] P. C. Luo *et al.*, "Multi-Resource Constrained Dynamic Workshop Scheduling Based on Proximal Policy Optimisation", *International Journal of Production Research*, vol. 60, no. 19, pp. 5937–5955, 2022. http://dx.doi.org/10.1080/00207543.2021.1975057

[24] C. L. Liu *et al.*, "Dynamic Job-Shop Scheduling Problems Using Graph Neural Network and Deep Reinforcement Learning", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 11, pp. 6836–6848, 2023. http://dx.doi.org/10.1109/TSMC.2023.3287655

[25] P. Li *et al.*, "Graph Neural Network-Based Scheduling for Multi-UAV-Enabled Communications in D2D Networks", *Digital Communications and Networks*, vol. 10, no. 1, pp. 45–52, 2024. http://dx.doi.org/10.1016/j.dcan.2022.05.014

[26] Z. Zhao *et al.*, "Link Scheduling Using Graph Neural Networks", *IEEE Transactions on Wireless Communications*, vol. 22, no. 6, pp. 3997–4012, 2022. http://dx.doi.org/10.1109/TWC.2022.3222781

Received: September 2025
Revised: November 2025
Accepted: November 2025

Contact address:
JieChen Zhao
Huangshan Vocational Technical College
Huangshan
Anhui
China
e-mail: zhaojiechen2025@163.com

JieChen Zhao received his master's degree in computer application technology in 2008, from Xi'an University of Science and Technology. Since 2018, he has held the position of associate professor. His primary research interest lies in virtual reality technology.