*applied sciences*

MDPI

*Article*

# Multi-Robot Task Allocation with Spatiotemporal Constraints via Edge-Enhanced Attention Networks

**Yixiang Hu †, Daxue Liu *,†, Jinhong Li, Junxiang Li and Tao Wu**

College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China; huyixiang19@nudt.edu.cn (Y.H.); lijinhong19@nudt.edu.cn (J.L.); lijunxiang@nudt.edu.cn (J.L.); wutao@nudt.edu.cn (T.W.)

* Correspondence: daxueliu@nudt.edu.cn

† These authors contributed equally to this work.

**Abstract**

Multi-Robot Task Allocation (MRTA) with spatiotemporal constraints presents significant challenges in environmental adaptability. Existing learning-based methods often overlook environmental spatial constraints, leading to spatial information distortion. To address this, we formulate the problem as an asynchronous Markov Decision Process over a directed heterogeneous graph and propose a novel heterogeneous graph neural network named the Edge-Enhanced Attention Network (E2AN). This network integrates a specialized encoder, the Edge-Enhanced Heterogeneous Graph Attention Network (E2HGAT), with an attention-based decoder. By incorporating edge attributes to effectively characterize path costs under spatial constraints, E2HGAT corrects spatial distortion. Furthermore, our approach supports flexible extension to diverse payload scenarios via node attribute adaptation. Extensive experiments conducted in simulated environments with obstructed maps demonstrate that the proposed method outperforms baseline algorithms in task success rate. Remarkably, the model maintains its advantages in generalization tests on unseen maps as well as in scalability tests across varying problem sizes. Ablation studies further validate the critical role of the proposed encoder in capturing spatiotemporal dependencies. Additionally, real-time performance analysis confirms the method's feasibility for online deployment. Overall, this study offers an effective solution for MRTA problems with complex constraints.

**Keywords:** multi-robot task allocation; reinforcement learning; deep learning; graph neural network



Academic Editor: Yutaka Ishibashi

Received: 23 December 2025
Revised: 13 January 2026
Accepted: 14 January 2026
Published: 15 January 2026



## 1. Introduction

Multi-Robot Task Allocation (MRTA) stands as a critical challenge in the collaboration of multi-robot systems. Its core objective lies in determining the optimal "task–robot" matching through effective coordination mechanisms, aiming to maximize the overall system utility (e.g., task success rate) while ensuring conflict-free decision-making [1]. With rapid advancements in autonomous technologies, MRTA has demonstrated essential value in fields such as precision agriculture [2], logistics and warehousing [3], unmanned reconnaissance [4,5], hazardous environment monitoring [6,7], and cooperative mapping [8]. In these real-world applications, the complexity of the environment and the diversity of tasks pose severe challenges to the efficiency and robustness of allocation algorithms.

According to the taxonomy proposed by Gerkey [1], MRTA problems are categorized based on three axes: robot type (Single-Task vs. Multi-Task), task type (Single-Robot vs.

Multi-Robot) and assignment type (Instantaneous vs. Time-extended). This study focuses on the allocation problems with single-task robots, single-robot tasks, and time-extended assignments (ST-SR-TA), where a single robot executes tasks serially and independently. When tasks are subject to deadlines and robots are permitted to execute a sequence of tasks, the problem falls into the category of In-schedule Dependencies (ID). Nunes et al. [9] indicate that MRTA with temporal constraints represents the most intensively studied scenario, aligning with the rigorous deadline requirements of most practical applications.

MRTA is a typical NP-hard combinatorial optimization problem that has been addressed by a suite of classic approaches, including exact methods like Mixed Integer Linear Programming (MILP) [10,11] and approximate methods such as clustering algorithms [12,13], heuristics [14,15], meta-heuristics [16–18], and market-based mechanisms [19,20]. Exact methods centrally optimize and thus guarantee global optimality, whereas approximate methods trade optimality for real-time efficiency. Nevertheless, both non-learning paradigms struggle with large-scale or dynamic instances. On the one hand, exact solvers suffer from combinatorial explosion, consequently, their runtime grows prohibitively as problem size increases. On the other hand, heuristics depend on hand-crafted rules that are tightly coupled to specific scenarios, limiting their ability to generalize when the environment changes.

MRTA is a typical NP-hard combinatorial optimization problem that has been addressed by a suite of classic approaches, generally categorized into exact and approximate methods. Exact methods, such as Mixed Integer Linear Programming (MILP) formulated in [11] and the CLIP algorithm in [10], centrally optimize to guarantee global optimality. However, they suffer from combinatorial explosion, causing runtimes to grow prohibitively as problem size increases. To address real-time constraints, approximate methods trade optimality for efficiency. These include clustering algorithms [12,13], heuristics [14,15], meta-heuristics [16–18], market-based mechanisms [19,20], and graph-based approaches like the BiG-MRTA [10]. While effective in their respective domains, these non-learning paradigms encounter significant challenges when applied to the complex spatial environments with obstacles considered in this study. Specifically, graph-based and heuristic methods often rely on simplified metrics (e.g., Euclidean distance) to maintain responsiveness, which fails to capture the non-linear path costs imposed by obstacles. Conversely, incorporating precise path planning into their iterative evaluation processes (e.g., fitness calculation in meta-heuristics or edge weighting in graph matching) would introduce a prohibitive computational bottleneck. Furthermore, rule-based heuristics often depend on scenario-specific designs, limiting their generalization across varying map topologies.

In recent years, the rapid development of deep learning has offered a new paradigm for solving combinatorial optimization problems. Kool et al. [21] proposed modeling the Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP) as sequential decision-making frameworks, achieving near-optimal results using deep reinforcement learning. Inspired by this, Wang et al. [22] incorporated the MRTA problem into similar frameworks, achieving promising results in temporal-constrained scenarios via imitation learning; however, the reliance on expert data limits its generalization ability. To improve scalability, recent studies [23,24] represented the MRTA problem as a graph structure utilizing Capsule Networks and Covariant Compositional Networks for feature extraction.

However, despite retaining the effective sequential decision-making backbone, these methods often overlook the environmental spatial constraints that are inextricably linked to execution time. Existing approaches typically encode the coordinate attributes of robots and tasks directly, implicitly assuming that the distance between nodes is the Euclidean distance. However, in complex real-world environments characterized by high-density obstacles or structural partitions, the Euclidean distance fails to reflect actual path costs,

leading to spatial information distortion. A previous study [25] considered spatial constraints in MRTA, yet its focus remained on avoiding conflicts within robot workspaces rather than addressing the fundamental impact of obstructed maps on path planning and task reachability.

To capture these complex dependencies, Graph Neural Networks (GNNs) incorporating edge attributes, exemplified by architectures such as EGAT [26] and HGT [27], have emerged as a promising direction. In the domain of robotic task scheduling, recent works have adopted these concepts; notably, Gao et al. [28] utilized edge-featured GATs to model transmission delays, while Ma et al. [29] employed heterogeneous graphs to describe collaborative relationships. Nevertheless, a structural limitation persists regarding the integration of physical constraints within these architectures. Most existing approaches incorporate edge attributes primarily to modulate attention weights, effectively treating them as auxiliary scalar biases. This design implies that while edge attributes influence the aggregation mechanism, specific edge information becomes implicitly fused into node features rather than being propagated as a distinct component. In scenarios characterized by strict spatiotemporal coupling, reliance on such implicit fusion potentially compromises the fidelity of physical constraints within the final embeddings, thereby constraining the solver's ability to discern tasks that exhibit logical proximity but remain physically distant.

To address this limitation, this study formulates the MRTA problem as a Markov Decision Process over a directed heterogeneous graph and proposes the Edge-Enhanced Attention Network (E2AN). Central to this framework is the Edge-Enhanced Heterogeneous Graph Attention Network (E2HGAT) encoder. Distinct from generic edge-aware GNNs, E2HGAT is architecturally engineered to specifically preserve the fidelity of physical constraints against implicit fusion. It incorporates an MPNN-inspired [30] spatial pre-injection to ground node features with physical distances and employs a novel dual-path residual mechanism driven by edge attributes. This residual structure establishes a dedicated information channel for physical constraints, ensuring they are explicitly preserved alongside semantic features processed by edge-enhanced aggregation. Furthermore, a dynamic gating mechanism is introduced to adaptively fuse these spatial insights with high-level representations. Finally, we integrate an attention-based decoder to construct an asynchronous sequential decision-making framework. This integrated design not only corrects spatial information distortion in complex environments but also enhances the model's generalization across varying problem scales, with the flexibility to adapt node attributes for diverse payload requirements.

The main contributions of this paper can be summarized as:

1. To address environmental spatial constraints in temporal-constrained MRTA, we propose a Markov Decision Process (MDP) model founded on a directed heterogeneous graph. This model operates within an asynchronous sequential decision-making framework and accommodates extensions for diverse payload requirements. This formulation transforms the high-dimensional combinatorial optimization problem into sequential assignments for deep reinforcement learning. Ultimately, the integrated paradigm intrinsically embeds environmental spatial constraints within the MRTA architecture, ensuring that allocation decisions align with physical realities.
2. To adapt to the directed heterogeneous graph modeling, we propose the E2AN. Built upon an encoder–decoder architecture, the core E2HGAT encoder fuses spatial edge attributes with node features to correct spatial information distortion. Coupled with an attention-based decoder, the network enhances the policy model's capacity to capture spatiotemporal dependencies inherent in the coupled constraints.
3. We constructed a simulated environment considering temporal and environmental spatial constraints. Through multi-dimensional experimental evaluation—covering

in-distribution testing, cross-scenario generalization, and cross-scale scalability testing—we verified the effectiveness of the proposed method in both basic scenarios and extended diverse payload scenarios. Benchmarking against baselines and ablation variants shows higher task success rates across different map environments and problem sizes. Furthermore, the analysis of real-time performance confirms the method's computational efficiency and feasibility for online deployment.

## 2. Problem Formulation

This section focuses on the ST-SR-TA class of Multi-Robot Task Allocation problems with temporal and environmental spatial constraints. In this scenario, tasks far outnumber robots, so each robot must navigate through the environment to carry out multiple tasks in sequence. This study centers on ground robots whose motion plans are constrained by regional traversability. Building on this setting, the environmental spatial constraints are encoded directly into the optimization model by representing obstacle-partitioned regions on a grid map. The formulation is further generalized to accommodate diverse payload requirements, ensuring that the robot's working capacity matches the requirements of the assigned tasks. The subsequent subsections will first provide a detailed description and formal formulation of the MRTA problem, followed by its modeling as a Markov Decision Process over a heterogeneous graph.

### 2.1. MRTA Problem Description and Formulation

We formulate the problem as a MILP model. Building on temporal-constrained approaches [10,31] and incorporating environmental spatial constraints from the map, we state the model as follows. Let $\mathcal{R} = \{1, \ldots, n_R\}$ denote the set of robots, and $\mathcal{T} = \{1, \ldots, n_T\}$ denote the set of tasks. Each task $i$ is characterized by a workload $w_i$ and a deadline $D_i$. The working capacity of robot $r$ is denoted by $c_r$, and the service time of task $i$ is given by $s_{r,i} = w_i / c_r$. To satisfy spatial constraints, we precompute exact robot-to-task travel distances offline with path planning algorithms like A* instead of using Euclidean estimates on the 2D map. Assuming a uniform and consistent travel speed for the robots, we derive the exact travel time $t_{i,j}$ between locations and the arrival time $t_{r,i}^{\text{start}}$. The decision variables of this problem are defined as follows: the binary variable $y_{r,s,i}$ equals 1 if task $i$ is assigned as the $s$-th task in robot $r$'s sequence, where $S_{\max}$ is the maximum number of tasks per robot. Variable $x_i$ equals 1 if task $i$ is completed successfully. Additionally, the task start time $T_i$ and completion time $C_i$ are defined as continuous variables. The domains of the relevant variables are defined as follows:

$$y_{r,s,i} \in \{0,1\}, \quad x_i \in \{0,1\}, \quad T_i \geq 0, \quad C_i \geq 0. \tag{1}$$

The optimization objective is to maximize the number of completed tasks that satisfy deadline constraints. The objective function is formulated as:

$$\max \; n_{\text{success}} = \sum_{i \in T} x_i. \tag{2}$$

The constraints for the MILP model are described in Equations (3)–(8). Equation (3) ensures that a task is assigned to a single robot and occupies a unique position in the sequence. To linearize conditional constraints and ensure they apply only to active robot–task assignments, a sufficiently large constant $M$ is introduced. The value of $M$ is set to twice the maximum deadline, i.e., $M = 2 \max_i D_i$. Equation (4) enforces the validity of all assignments, ensuring that completion times do not exceed deadlines. Due to temporal constraints, if a task can be successfully assigned to a robot (satisfying the deadline), it is considered successfully completed, as implied by Equation (5). Equation (6) guarantees

that the completion time of a task is greater than its start time plus the execution duration, where $\mathcal{S} = \{0, \ldots, S_{\max}\}$ denotes the range of the task sequence. Equation (7) ensures that the sequential arrangement of tasks adheres to spatial constraints, where $\hat{\mathcal{S}} = \mathcal{S} \setminus \{S_{\max}\}$. Equation (8) guarantees that every robot can reach its first assigned task from its start point.

$$\sum_{r \in \mathcal{R}} \sum_{s=0}^{S_{\max}} y_{r,s,i} = 1 \quad \forall i \in \mathcal{T}, \tag{3}$$

$$C_i \leq D_i + M \cdot (1 - x_i) \quad \forall i \in \mathcal{T}, \tag{4}$$

$$x_i = \sum_{r \in \mathcal{R}} \sum_{s=0}^{S_{\max}} y_{r,s,i} \quad \forall i \in \mathcal{T}, \tag{5}$$

$$C_i \geq T_i + s_{r,i} - M \cdot (1 - y_{r,s,i}) \quad \forall r \in \mathcal{R},\ \forall s \in S,\ \forall i \in \mathcal{T}, \tag{6}$$

$$T_j \geq C_i + t_{i,j} - M \cdot (2 - y_{r,s,i} - y_{r,s+1,j}) \quad \forall r \in \mathcal{R},\ \forall s \in \hat{\mathcal{S}},\ \forall i, j \in \mathcal{T},\ i \neq j, \tag{7}$$

$$T_i \geq t_{r,i}^{\text{start}} - M \cdot (1 - y_{r,0,i}) \quad \forall r \in \mathcal{R},\ \forall i \in \mathcal{T}. \tag{8}$$

To address diverse payload requirements, we extend the robot capacity $\mathbf{c}_r$ and the task workload $\mathbf{w}_i$ to $K$-dimensional vectors in $\mathbb{R}_+^K$. Assuming each task $i$ requires a specific payload type $k_i$, $\mathbf{w}_i$ is defined as a sparse vector with a non-zero value exclusively in the $k_i$-th dimension.

This study defines a binary variable $p_{r,k} \in \{0, 1\}$ to indicate if robot $r$ is equipped with payload $k$. Robots can carry multiple payloads, resulting in multi-dimensional capacity vectors $\mathbf{c}_r$. Payloads are divided into basic (carried by all robots) and specific. We assume each robot carries exactly one specific payload, which determines its category. Accordingly, task requirements are subdivided into basic requirements and specific requirements: a basic requirement corresponds to a basic payload and can be satisfied by any robot; a specific requirement corresponds to a specific payload and can only be satisfied by the corresponding category of robots. Since robots can execute multiple types of tasks, different types of tasks must compete for the same robotic resources, which increases the coupling and complexity of the problem.

To ensure the matching between capabilities and requirements, task $i$ can only be assigned to a robot equipped with the corresponding payload $k_i$. This constraint is formalized as:

$$\sum_{s=0}^{S_{\max}} y_{r,s,i} \leq p_{r,k_i} \quad \forall r \in \mathcal{R},\ i \in \mathcal{T}. \tag{9}$$

This constraint guarantees that a robot lacking the necessary payload cannot execute the task. Accordingly, the service duration $s_{r,i}$ is determined jointly by the workload in the corresponding dimension and the robot's efficiency in that dimension, calculated as:

$$s_{r,i} = \frac{w_{i,k_i}}{c_{r,k_i}}. \tag{10}$$

### 2.2. Markov Decision Process over a Heterogeneous Graph

Since the essence of the MRTA problem lies in seeking an optimal matching between two sets of entities, we draw upon the classic bipartite graph matching model to formulate the problem as an MDP over a complete directed heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The node set $\mathcal{V} = \mathcal{V}_R \cup \mathcal{V}_T$ comprises two categories: robot nodes and task nodes. Robot node attributes consist of real-time position, current time, working capability, and operational status (moving, working, idle). Task node attributes include deadline, coordinates, workload, and task status (pending, completed). In this study, both workload and working capability

can be extended to $K$-dimensional vectors to match the diverse payload requirements of the scenario. The edge set $\mathcal{E}$ describes the spatial correlation between robots and tasks, and its weights $\omega_{ij}$ are designed to quantify the connection strength between nodes. In scenarios with diverse payload requirements, edges can only be established between robot and task nodes when the payload matching relationship is satisfied. For these valid connections, to eliminate dimensional discrepancies, a normalized weight function based on reachability is defined:

$$\omega_{ij} = \alpha\left(1 - \frac{d_{ij}}{d_{\max}}\right), \tag{11}$$

where $d_{ij}$ is the precise distance solved based on spatial constraints, $d_{\max}$ is the distance calculated between diagonal endpoints of the environmental map, and $\alpha$ is a tuning coefficient. This formula ensures that spatially proximal nodes possess higher correlation weights.

Based on the aforementioned heterogeneous graph structure, we model the task allocation process as a sequential decision-making problem, formalized as an MDP tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$.

State Space $\mathcal{S}$: This describes the global state of the system at decision moment $t$. Specifically, state $S_t$ is composed of the current dynamic heterogeneous graph $\mathcal{G}_t$ and the static environmental map. The heterogeneous graph stores the current physical states of robots and attribute features of tasks, serving as the primary input for the policy. The environmental map provides spatial constraint information describing traversable areas and obstacles, serving as the basis for accurate state updates. In this research scenario, static obstacles are known a priori, and communication bandwidth among agents is sufficient; thus, we assume the environment is fully observable, meaning the central scheduler can access complete global state information in real-time, avoiding information loss caused by partial observability.

Action Space $\mathcal{A}$: To avoid conflicts arising from concurrent decisions by multiple agents, the decision step does not equal the time step; instead, an asynchronous event-triggered mechanism is adopted. At each decision step $t$, the system selects the robot $r^*$ that is currently the first idle as the decision agent. The action set is defined as the index set of optional tasks $\{0, \ldots, N_T\}$. The selected action must be valid: candidate tasks must be in the pending state and must not be reassigned. If the scenario involves diverse payload requirements, the robot is required to be equipped with the specific payload required by the candidate task.

State Transition $\mathcal{P}$: We formulate the MRTA problem as a deterministic transition model, where a valid task assignment means the task is deterministically completed, without guaranteeing it finishes before its deadline. The model is described in Algorithm 1. When action $a_t$ is executed—i.e., task $i$ is assigned to robot $r^*$—the graph structure updates accordingly: task $i$ is marked as completed, its completion time is logged, and it is removed from the heterogeneous graph. Meanwhile, the state of robot $r^*$ is updated based on task $i$, with its position transitioning to the task's endpoint and its time state set to the task completion moment.

Reward $\mathcal{R}$: As current decisions have long-term impacts on the global schedule, quantifying their immediate effect is difficult. Therefore, we adopt a sparse reward mechanism to evaluate the overall solution only at the end episode. The reward function is defined as follows:

$$R = \begin{cases} -\sum_{i=1}^{n_T} R_i, & \text{if } n_{\text{success}} < n_T \\ e^{-T_\Psi}, & \text{if } n_{\text{success}} = n_T \end{cases}, \tag{12}$$

where $n_{\text{success}}$ is the number of tasks completed on time, $R_i$ includes the overdue penalty per task, and $T_\Psi$ denotes the global time cost. Their specific calculation methods are as follows:

$$R_i = \begin{cases} \frac{C_i}{D_i}, & \text{if } C_i > D_i \\ 0, & \text{otherwise} \end{cases}, \tag{13}$$

$$T_\Psi = \frac{1}{D_{\max}} \sum_{r \in \mathcal{R}} C_r^{\text{end}}. \tag{14}$$

In Equation (14), $C_r^{\text{end}}$ is the completion time of robot $r$, and $D_{\max}$ represents the maximum deadline. The reward function is designed to balance feasibility and optimality:

- Penalty based on delay length: During the early training, when the policy fails to complete all tasks, we use a penalty $R_i$ based on delay duration instead of the task failure rate. This metric quantifies the degree of failure. It encourages the agent to reduce the overdue duration even when a deadline violation is inevitable, preventing passive task abandonment.
- Reward based on time efficiency: When the policy completes all tasks before their deadlines, the focus shifts to minimizing execution costs to further improve the policy. At this stage, the reward function becomes positive, adopting an exponential decay form $e^{-T_\Psi}$. We utilize the total working duration of the robots as the cost metric and use $D_{\max}$ as a scaling factor for normalization.

Finally, to integrate the defined state, action, transition, and reward components into a coherent operational framework, the complete asynchronous decision-making process is detailed in Algorithm 1. This pseudo-code illustrates the complete interaction cycle, demonstrating how the scheduler samples actions for idle agents, updates the global state according to the transition model, and derives the sparse reward, thereby linking the mathematical definitions to the procedural execution.

**Algorithm 1** MDP Allocation Algorithm

**Inputs:** Robots $\mathcal{R}$, Tasks $\mathcal{T}$, Map $\mathcal{M}$, Policy $\pi_\theta$
**Initialize:** $\mathcal{R}_{active} \leftarrow \mathcal{R}$, $\mathcal{T}_{pending} \leftarrow \mathcal{T}$, $t \leftarrow 0$, Construct $\mathcal{G}_0$
**while** $\mathcal{T}_{pending} \neq \emptyset \wedge \mathcal{R}_{active} \neq \emptyset$ **do**
  $r^* \leftarrow \arg\min_{r \in \mathcal{R}_{active}}(r.\text{time})$ ▷ Select first idle robot
  $\mathbf{m}_t \leftarrow \text{GetMask}(r^*, \mathcal{T}_{pending})$
  **if** $\mathbf{m}_t = \mathbf{0}$ **then**
    $\mathcal{R}_{active} \leftarrow \mathcal{R}_{active} \setminus \{r^*\}$ ▷ No valid tasks
  **else**
    $a_t \sim \pi_\theta(\mathcal{G}_t) \odot \mathbf{m}_t$ ▷ Sample valid task $i^*$
    $S_{t+1} \leftarrow \text{StateUpdate}(S_t, a_t)$
    $\mathcal{G}_{t+1} \leftarrow \text{UpdateGraph}(\mathcal{G}_t, S_{t+1})$
    $\mathcal{T}_{pending} \leftarrow \mathcal{T}_{pending} \setminus \{a_t\}$ ▷ Update pending task set
    $t \leftarrow t + 1$
  **end if**
**end while**
**if** $n_{\text{success}} = n_T$ **then**
  $R \leftarrow e^{-T_\Psi}$ ▷ Reward for global time efficiency
**else**
  $R \leftarrow -\sum R_i$ ▷ Penalty for failed tasks
**end if**
**return** $R$

## 3. Methods

Based on the Heterogeneous Graph Markov Decision Process constructed in Section 2.2, the present section implements a deep reinforcement learning method within a heterogeneous graph framework. Specifically, this study proposes the Edge-Enhanced Attention Network (E2AN). This method employs an encoder–decoder architecture as the policy model to generate optimal task allocation schemes through asynchronous sequential decision-making, as illustrated in Figure 1.

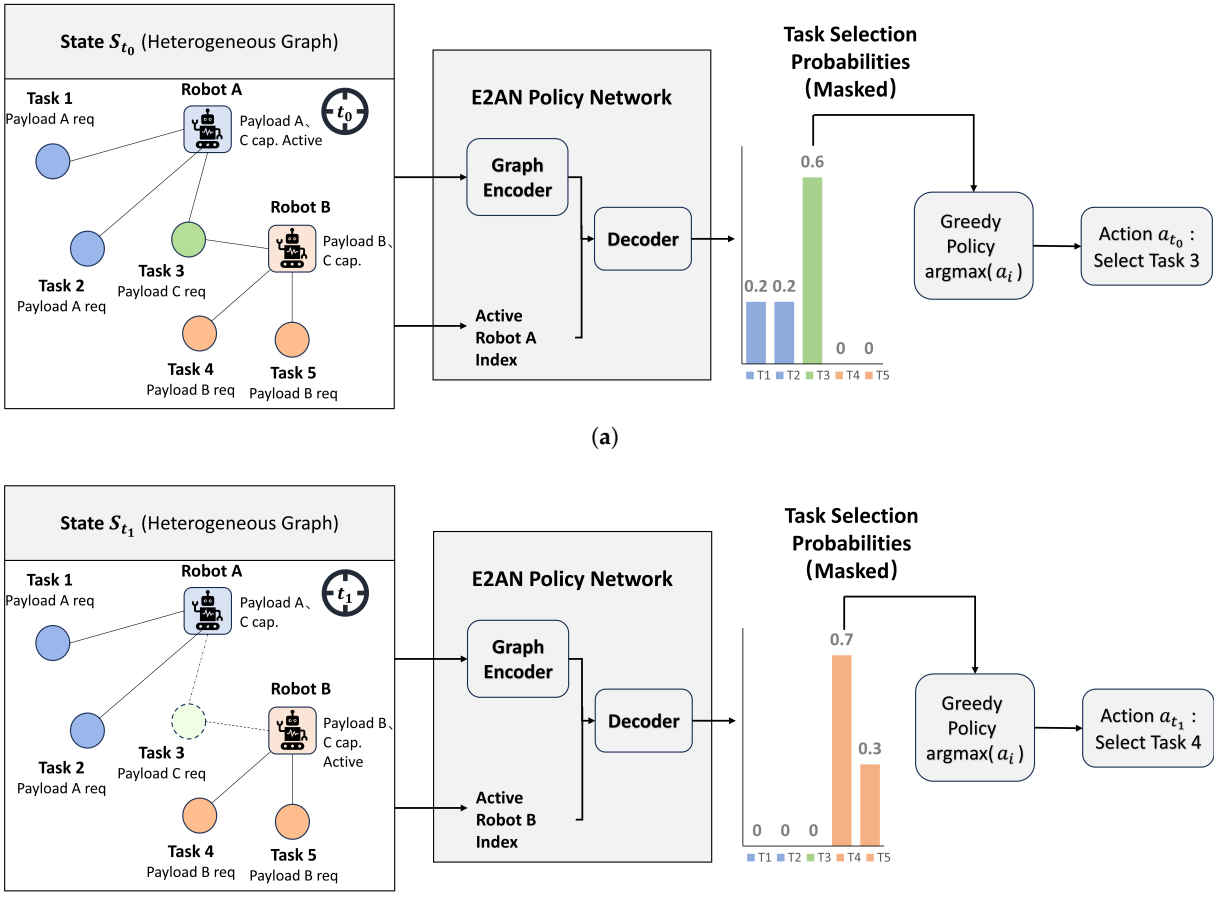


**Figure 1.** Illustration of an MRTA policy deployment via E2AN. (**a**) At time $t_0$, task 3 is assigned to robot A; (**b**) at time $t_1$, task 4 is assigned to robot B. The E2AN output is reset to zero for the previously assigned task (task 3 in panel (**b**)) and for tasks that do not satisfy the payload matching relationship due to the lack of corresponding payloads.

Initially, the Edge-Enhanced Heterogeneous Graph Attention Network (E2HGAT) functions as the core encoder and takes the constructed heterogeneous graph as input. It extracts the attributes of robot and task nodes along with their topological connections. The encoder maps all nodes into high-dimensional embeddings containing global spatiotemporal information. These node embeddings are input to the attention decoder. It also receives the index of the robot making decisions, which is the first idle one. The decoder then calculates the selection probability distribution for the robot across all valid tasks. The policy greedily selects actions based on this distribution to interact with the environment, and this process iterates until a complete task sequence is formed. For optimization, the proposed method employs the REINFORCE algorithm with a baseline to reduce variance in gradient estimation and enhance policy convergence.

### *3.1. Edge-Enhanced Heterogeneous Graph Attention Network Encoder*

We propose an Edge-Enhanced Heterogeneous Graph Attention Network encoder as the core encoder module. As shown in Figure 2, taking the raw heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ containing robot and task nodes as input, the encoder maps all nodes into a unified high-dimensional embedding space $\mathbf{H}_{final}$ containing both local attributes and global topological structures through stacked multilayer message passing. Functioning as an "Edge-to-Edge aware" architecture, it distinguishes itself from generic GNN architectures by structurally integrating spatial constraints across all processing stages: initialization, attention modeling, propagation, and fusion.

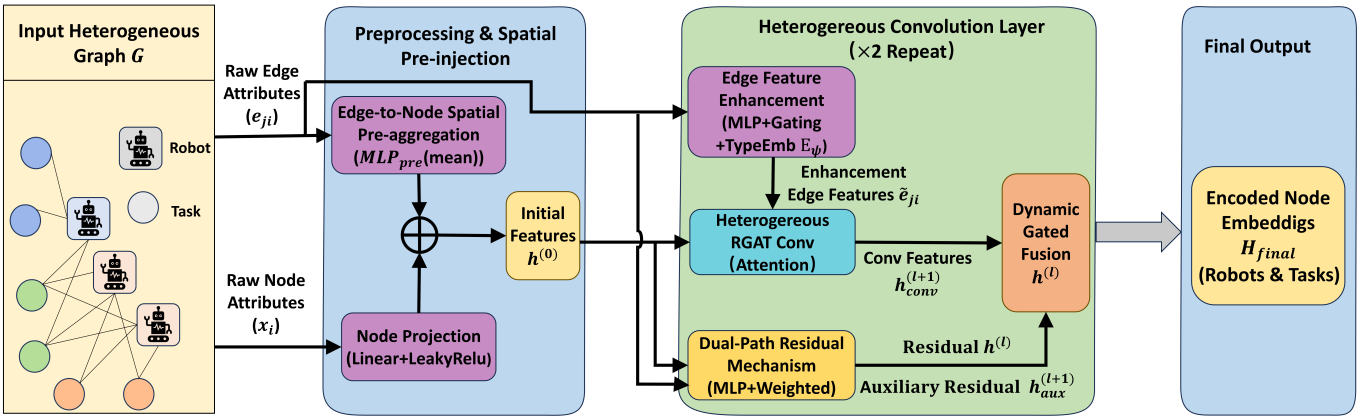


**Figure 2.** Overview of the E2HGAT Encoder. After initializing $\mathbf{h}^{(0)}$ from raw node attributes $\mathbf{x}_i$ and spatial edge attributes $\mathbf{e}_{ji}$, the model utilizes stacked convolution layers. Within each layer, a Dynamic Gated Fusion module integrates features from the RGAT Convolution (driven by enhanced edges $\tilde{\mathbf{e}}_{ji}$) and the Dual-Path Residual Mechanism. This process iteratively updates embeddings $\mathbf{h}^{(l+1)}$ to generate the final representation $H_{final}$.

Addressing the MRTA with diverse payloads, the framework projects node attributes of varying lengths, such as $k$-dimensional capability requirements, into a unified feature space. This mechanism enables the model to adapt to arbitrary input dimensions without altering the core architecture. Unlike homogeneous graph networks, this framework models robots and tasks through type-specific projection layers. This design integrates edge attributes throughout. Therefore, the model captures both the environmental spatial constraints and the highly coupled temporal constraints in MRTA. By leveraging the intrinsic relation-aware characteristics of RGAT, the encoder achieves permutation invariance, so node embeddings are robust to variations in node indexing order. The architecture includes five modules for enhancing spatial awareness: Spatial Information Pre-injection, Edge Feature Enhancement, Heterogeneous RGAT Convolution, Dual-Path Residual Mechanism, and Dynamic Gated Fusion.

Drawing inspiration from the aggregation philosophy of Message Passing Neural Networks (MPNN) [30], we implement a Spatial Pre-injection mechanism. Distinct from standard MPNNs that use edges conditionally during message construction, we position this as a preprocessing step to explicitly embed neighbor physical distances into the initial node representation $\mathbf{h}^{(0)}$ before node attribute interaction. Initially, node features $\mathbf{x}_i$ are converted to the same hidden dimension D using type-specific linear layers.

$$\mathbf{h}_i^{(0)} = \text{LeakyReLU}\left(\mathbf{W}_{\text{proj}}^{\phi(i)}\mathbf{x}_i\right), \tag{15}$$

where $\phi(i)$ represents the node type. The edge attribute $\mathbf{e}_{ji}$ is aggregated through summation operations and then projected into vectors by the multi-layer perceptron ($\mathrm{MLP}_{pre}$), which will be injected into the initial features:

$$\mathbf{h}_i^{(0)} \leftarrow \mathbf{h}_i^{(0)} + \mathrm{MLP}_{pre}\left(\frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)}\mathbf{e}_{ji}\right), \tag{16}$$

where $\mathcal{N}(i)$ is the set of neighbors for node $i$.

The encoder consists of two stacked heterogeneous graph attention convolutional layers. This design enables interaction information between indirectly connected nodes, such as tasks competing for the same robot, to propagate through multi-hop message passing.

In each convolutional layer, Edge Feature Enhancement is applied as an initial step to fuse $\mathbf{e}_{ji}$ with its corresponding edge type embedding $\mathbf{E}_{\psi(j,i)}$. Although the topology contains only robot–task connections, directionality is modeled by learning distinct edge type embeddings $\mathbf{E}_{\psi(j,i)}$ conditioned on the start node type. Furthermore, a gating function $\sigma(\cdot)$ is incorporated to suppress long-range or weakly relevant connections, acting as a physical filter before attention calculation. The formulation is defined as

$$\mathbf{e}'_{ji} = \mathrm{LayerNorm}\big(\mathrm{MLP}_{enc}(\mathbf{e}_{ji})\big) + \mathbf{E}_{\psi(j,i)}, \tag{17}$$

$$\tilde{\mathbf{e}}_{ji} = \sigma\big(\mathbf{W}_{gate}[\mathbf{e}'_{ji} \parallel \mathbf{e}_{ji}]\big) \odot \mathbf{e}'_{ji}, \tag{18}$$

where $\parallel$ is concatenation, $\odot$ denotes element-wise product, and $\tilde{\mathbf{e}}_{ji}$ represents the enhanced edge features.

The RGAT mechanism is used to model graph heterogeneity, allowing it to learn independent parameters and attention patterns for different edge types. Unlike standard RGAT [32] which relies solely on discrete relation types, or EGAT [26] which focuses on continuous attributes, our design implements a hybrid attention. This study introduces enhanced edge features $\tilde{\mathbf{e}}_{ji}$ in the attention coefficient calculation. By concatenating edge and node attributes, we incorporate spatial information into the input, as shown in Equation (19).

$$\alpha_{ji} = \mathrm{softmax}\Big(\mathrm{LeakyReLU}\Big(\mathbf{a}^T[\mathbf{W}_q\mathbf{h}_i^{(l)} \parallel \mathbf{W}_k\mathbf{h}_j^{(l)} \parallel \mathbf{W}_e\tilde{\mathbf{e}}_{ji}]\Big)\Big), \tag{19}$$

$$\mathbf{h}_{i,\mathrm{conv}}^{(l+1)} = \sum_{j\in\mathcal{N}(i)}\alpha_{ji}\mathbf{W}_v\mathbf{h}_j^{(l)}, \tag{20}$$

where $\mathbf{h}_i^{(l)}$ and $\mathbf{h}_{i,\mathrm{conv}}^{(l+1)}$ are the $l$-th layer input features and current layer convolutional output features; $\mathbf{W}_{\{q,k,v,e\}}$ are learnable projection matrices; $\mathbf{a}$ is the attention vector corresponding to the directed edge type; and $\mathcal{N}(i)$ is the neighbor set of node $i$.

In addition to retaining the standard feature residual connection ($\mathbf{h}^{(l)} + \mathbf{h}_{\mathrm{conv}}^{(l+1)}$), Dual-Path Residual Mechanism introduces an auxiliary Edge-Attribute-Driven residual path. Distinct from standard residual connections that solely preserve node semantic continuity, this mechanism establishes a dedicated channel independent of node semantic similarity. It explicitly transmits edge-conditioned statistical information across layers to prevent spatial constraints from being obscured via implicit fusion with high-dimensional node features. Specifically, raw edge attributes are first mapped to scalar weights $w_{ji} = \sigma(\mathrm{MLP}_{dist}(\mathbf{e}_{ji}))$ via a projector. Subsequently, neighbor features are aggregated based on these weights to generate the auxiliary residual vector $\mathbf{h}_{i,\mathrm{aux}}^{(l+1)}$. As shown in Equation (21), this mechanism ensures that features of proximal nodes are highlighted and preserved.

$$\mathbf{h}_{i,\text{aux}}^{(l+1)} = \text{MLP}_{proj}\left(\frac{\sum_j w_{ji}\mathbf{h}_j^{(l)}}{\sum_j w_{ji} + \epsilon}\right), \tag{21}$$

where $\epsilon$ is a smoothing term added to prevent the denominator from becoming zero.

At the end of each graph convolutional layer, to adaptively integrate newly extracted topological features with retained residual information, we use Dynamic Gated Fusion. Unlike standard Gated GCNs that rely on node states, E2HGAT introduces an edge-conditioned gating mechanism, enabling the model to adaptively regulate message propagation strength according to edge semantics. Initially, a learnable update gate $\mathbf{z}_i$ is calculated to determine the optimal ratio for feature fusion. Then, the fused features are processed through Layer Normalization and an ELU activation function to generate the final node embedding $\mathbf{h}_i^{(l+1)}$ for the current layer:

$$\mathbf{z}_i = \sigma\left(\mathbf{W}_{update}\mathbf{h}_{i,\text{conv}}^{(l+1)}\right), \tag{22}$$

$$\mathbf{h}_i^{(l+1)} = \text{LayerNorm}\left(\text{ELU}\left(\mathbf{z}_i \odot \mathbf{h}_{i,\text{conv}}^{(l+1)} + (1-\mathbf{z}_i) \odot (\mathbf{h}_i^{(l)} + \mathbf{h}_{i,\text{aux}}^{(l+1)})\right)\right). \tag{23}$$

### 3.2. Multi-Head Attention Mechanism Based Decoder

This study implements a stepwise allocation decoder based on Multi-Head Attention (MHA), which transforms the decoding process into a probability generation task for a specific robot. In each decision-making step, the model selects the first idle robot, calculates the selection probability of the robot for all valid tasks, and then greedily selects the optimal task. This process is repeated until all tasks have been assigned. This point-wise construction mechanism can decompose combinatorial optimization problems into a series of sub-problems, which makes the strategy more robust under different problem scales and agent configurations.

The attention mechanism constructs the Query vector ($\mathbf{q}$) and Key–Value pairs ($\mathbf{K}, \mathbf{V}$). In this model, $\mathbf{K}$ and $\mathbf{V}$ are projections of the task node embeddings $\mathcal{H}_{task}$ and remain invariant across decision steps. Conversely, the Query vector $\mathbf{q}_t$ is dynamically constructed via a linear projection of the context vector. This context vector integrates the current robot's embedding, the mean embedding of peer robots, and the global mean of task embeddings. This design enables $\mathbf{q}_t$ to act as a probe, selecting the most compatible target from the task feature space.

The decoder performs multi-head scaled dot product attention calculation, using 8 parallel attention heads. In each head $h$, the dynamic robot Query interacts with the global task Keys to compute compatibility scores. The calculation process is as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{24}$$

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)\mathbf{W}^O, \tag{25}$$

where $\mathbf{W}^O$ is the output projection matrix. The outputs from the multiple heads are concatenated and linearly transformed to generate a latent vector rich in context information for the current decision step.

In the final output stage, to map the latent vector back to the discrete action space, the decoder utilizes a single-head attention mechanism to multiply the MHA output with a Logit $\mathbf{K}$ pointer, obtaining the final pointing score for each task. The Logit $\mathbf{K}$ pointer is derived from the global task node embeddings $\mathbf{H}_{task}$ via a linear transformation. Since

the decoding process must ensure the generated task sequence satisfies all constraints and is nonrepetitive, we introduce a hard masking strategy prior to softmax normalization. The system maintains a binary mask based on the current state, forcing the Logit values corresponding to already assigned tasks and tasks incompatible with the current robot's payload capacity to $-\infty$. This operation ensures the probability distribution is defined exclusively over the set of valid tasks. Subsequently, the policy outputs action $a_t$ via greedy decoding based on this distribution, marks the task as completed, and proceeds to the next decision step until $N$ assignments are completed.

### 3.3. Training Algorithm

Although the MILP formulation constructed in Section 2.1 theoretically provides a mathematical form for optimal solutions, the NP-hard nature of the MRTA problem renders the acquisition of large-scale, high-quality ground truth labels computationally prohibitive as the problem size increases. This computational bottleneck severely restricts the feasibility of supervised learning approaches. Consequently, building on the MDP framework defined in Section 2.2, the proposed method employs the REINFORCE policy gradient algorithm to train the proposed neural network.

To effectively mitigate the variance in gradient estimation, we incorporate a greedy rollout baseline mechanism. The baseline update condition is that the paired $t$-test shows that the current strategy reward is better than the baseline and $p$-value is less than the significance level $\alpha = 0.05$. The gradient approximation of the objective function $J(\theta)$ with respect to the model parameters $\theta$ is formulated as shown in Equation (26):

$$\nabla_\theta J(\theta) \approx \frac{1}{B} \sum_{i=1}^{B} (R(s_i, a_i) - b(s_i)) \nabla_\theta \log p_\theta(a_i|s_i), \tag{26}$$

where $B$ denotes the batch size; $s_i$ represents the state of the $i$-th input instance; $a_i$ is the action generated by the current policy $p_\theta$; and $R(s_i, a_i)$ is the cumulative reward. The term $b(s_i)$ serves as the baseline value, that is, the reward obtained by the baseline policy for the same state $s_i$.

Independent training and validation sets were generated for each epoch. During training, the decoder employs stochastic sampling instead of greedy strategy to enhance exploration and prevent premature convergence. The validation set evaluates the current policy $\pi_\theta$ against the baseline $\pi_{\theta_{BL}}$.

## 4. Results

To systematically evaluate the generalization and scalability of the proposed method, this study designed the experiments in two progressive stages. In the initial stage, we focused on a basic MRTA scenario involving temporal constraints and environmental spatial constraints. By constructing simulated environments and datasets, we established the performance advantage of our model on basic problems through baseline comparisons, generalization tests, scalability tests and ablation studies. Subsequently, the experimental scenario was extended to include diverse payload requirements, comprehensively verifying the model's adaptability under complex constraints. Finally, we measured the model's inference latency to evaluate the real-time responsiveness of the proposed approach, thereby completing the evaluation pipeline from basic generalization and scalability to online deployability.

### 4.1. Design of Experiments

This section specifies the simulation setup for the basic MRTA scenario. This study introduced environmental maps to impose spatial constraints, utilizing a benchmark dataset

from the field of Multi-Agent Path Finding (MAPF) [33] as the map resource library. The dataset includes mazes, rooms, city, and random obstacles, as shown in Figure 3. To increase the complexity of spatial constraints, we selected structurally benchmark maze map to construct the training environment and used other map types for generalization tests.

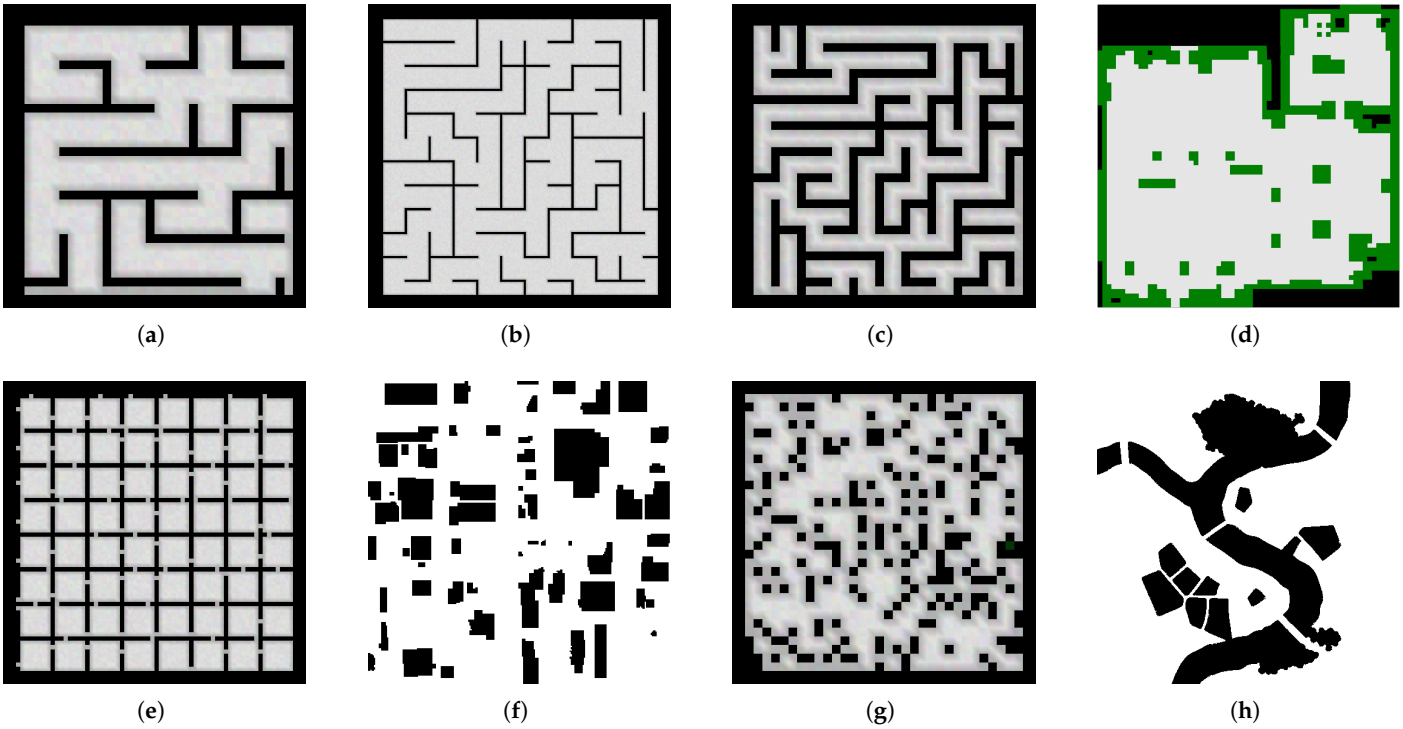


**Figure 3.** Environment maps selected for this study. (**a**) Benchmark maze. (**b**) Normal maze. (**c**) Complex maze. (**d**) Simple room. (**e**) Complex room. (**f**) City scenario. (**g**) Random obstacles. (**h**) Wild scenario. Black and green regions denote obstacles; white and gray regions denote traversable areas.

In the basic scenario configuration, the simulation parameters were adapted from the standard Task Allocation Problem with Time and Capacity (TAPTC) benchmark established by Mitiche et al. [34] and utilized by recent baselines [23]. The map was set as a 2D $100 \times 100$ grid to simulate a medium-sized environment characterized by complex spatial obstructions. While the original benchmark considers obstacle-free settings, we incorporated static obstacles to test spatial adaptability. Regarding system scale, we set $n_T = 100$ tasks and $n_R = 7$ robots (the upper bound of the standard dataset). This robot count was selected to provide sufficient total service capacity relative to the task volume in obstructed environments. Temporal and physical parameters were calibrated to maintain consistency with the benchmark while reflecting the dynamics of ground robots: the deadline $D_i$ was uniformly sampled from the interval $[36, 540]$, covering varying urgency levels. A speed of 4 grid units per time step was selected to match the ground robot kinematics with the map scale and environmental navigation constraints. The service capabilities of robots and the workload requirements of tasks were uniformly sampled from the intervals $[1, 2]$ and $[10, 15]$, respectively, modeling realistic variations in operational capacity and task difficulty.

Based on these parameters, we constructed the dataset and trained the model for 100 epochs using the training algorithm described in Section 3.3. Detailed training hyperparameters are listed in Table 1. To ensure reproducibility and training stability, specific architectural and optimization parameters were explicitly defined. The E2HGAT encoder utilizes a 2-layer structure to capture sufficient topological depth without over-smoothing; a detailed sensitivity analysis justifying this choice is provided in Appendix A. Meanwhile,

while the decoder employs an 8-head attention mechanism to diversify the focus on different task features. Furthermore, a gradient clipping threshold of 1.0 and a tanh clipping of 10.0 were applied to prevent gradient explosion and constrain the logits range during the reinforcement learning process. The test dataset was constructed with same scenario configurations, each of which includes 100 fixed cases.

**Table 1.** The detailed hyperparameters used for model architecture and training.

| Hyperparameter | Value |
|---|---|
| Epoch | 100 |
| Training Dataset | 10,000 |
| Validation Dataset | 1000 |
| Hidden Embedding Dimension | 128 |
| Encoder GNN Layers | 2 |
| Attention Heads | 8 |
| Optimizer | Adam (PyTorch 2.0.1) |
| Learning Rate | $1 \times 10^{-4}$ |
| Gradient Clipping Norm | 1.0 |
| Tanh Clipping | 10.0 |
| Baseline Significance ($\alpha$) | 0.05 |
| Training Frequency | 100 |

All simulations and algorithm implementations were executed using Python 3.8. The proposed network architecture and training pipeline were developed using PyTorch 2.0.1, while the graph neural network modules within the encoder utilized the PyTorch Geometric (PyG) library (version 2.6.1). Additionally, the SciPy library was employed to accelerate path planning calculations in the simulation. All experiments were conducted on a workstation equipped with an Intel Core i9-10900KF CPU (@ 3.70 GHz) (Intel, Santa Clara, CA, USA), 64GB of RAM, and a single NVIDIA GeForce RTX 3090 GPU with 24 GB VRAM (NVIDIA, Santa Clara, CA, USA).

### *4.2. Baselines*

To comprehensively evaluate the performance of the proposed method, five baselines were selected for comparison, including random selection method, optimization-based method, graph-based method and two state-of-the-art learning-based methods:

1. Random Selection (RD): Based on the asynchronous framework, this method selects actions randomly from valid candidates. To ensure a fair comparison, we applied the masking strategy from Section 3.2 to filter out completed or payload-mismatched tasks, ensuring physical feasibility. This strategy serves to evaluate the difficulty of scenario cases and establishes a performance lower bound. It verifies that the proposed learning algorithm captures actual decision patterns rather than merely relying on constraints.
2. CLIP-MRTA: Adapted from prior work [10], this method formulates the problem as a MILP model using A* distances and environmental spatial constraints. We employed the Gurobi solver (version 11.0.3) [35] with a 10-min runtime cutoff. Due to computational complexity, it yields some feasible rather than optimal solutions within the limited time.
3. Capsule Attention Mechanism (CapAM): This approach [23] models MRTA problem as a homogeneous task graph processed by a graph capsule network. In its decoding stage, the query vector is constructed by concatenating the robot state with temporal and peer information to handle task allocation.

4. Covariant Attention Mechanism (CAM): Distinct from CapAM, this approach [24] employs a covariant compositional network encoder. It enhances representation by extracting local $k$-nearest neighbor structural features. To ensure a fair comparison, both CapAM and CAM were trained using the identical REINFORCE algorithm and hyperparameter settings (Table 1) as our proposed method.
5. Bipartite Graph-based MRTA (BiG-MRTA): Representing a recent graph-based approach [10], this method maps the allocation problem to a weighted bipartite graph structure. It employs a heuristic incentive model to quantify robot–task suitability as edge weights and subsequently utilizes maximum weighted matching to derive optimal assignments. Designed as an efficient online solver, this approach supports decentralized deployment, serving as a robust baseline for evaluating performance against traditional graph-theoretic algorithms in dynamic environments.

### *4.3. Generalization and Scalability Analysis*

This section evaluates the performance of the E2AN method in basic MRTA scenarios. Initially, this study conducted in-distribution testing to comprehensively compare E2AN with five baseline algorithms, verifying the fundamental advantages of the proposed method. Subsequently, we selected the two learning-based baselines to conduct further generalization and scalability experiments on the trained models.

To clarify the experimental scope, we defined generalization as the model's adaptability to unseen maps. Although the model was trained only on benchmark maze maps, it was evaluated on unseen maps including mazes and other categories. The probability distributions of instance configuration parameters in the test set, such as task positions, deadlines, and robot attributes, remain consistent with the training set, and the scenario scale (number of tasks and robots) remains constant. Scalability was defined as the model's ability to handle problems of different scales. Since the model was trained with a fixed scale, we generated evaluation cases of varying scales by changing the number of tasks (increasing or decreasing) while maintaining a constant robot-to-task ratio.

Regarding evaluation metrics, based on the problem definition in Section 2.1, this study adopted the task success rate as the core metric for allocation effectiveness. We visualized the distribution of task success rates across 100 cases in the test set using box plots. This study employed the median to quantify overall performance and the interquartile range (IQR) to characterize stability. Performance differences were validated via paired $t$-tests ($\alpha = 0.05$).

In-Distribution Testing: We compared the E2AN algorithm with five baseline algorithms on a test set that is equally distributed as the training set. As shown in Figure 4, the median task success rate of E2AN was significantly higher than that of the baseline algorithm. Meanwhile, the IQR of this algorithm was the narrowest, demonstrating the best stability.

After analyzing the performance of each algorithm, it was found that all methods except RD could achieve a relatively high task success rate, which confirms their ability to solve MRTA problems. Constrained by the computational complexity of the problem and the preset runtime, the CLIP method could not converge to the global optimal solution in all test scenarios. Overall, the ranking of each algorithm is as follows: E2AN > CAM > CapAM > BiG $\approx$ CLIP > RD. The results of the paired sample $t$-test further statistically verified the performance advantages of E2AN.

Generalization Tests: To assess the model's adaptability in unseen environments, we designed the generalization tests using six map types with completely different structures (corresponding to Figure 3b–g), which include new mazes, rooms, cities and random obstacles. The experimental results in Figure 5 indicate that, in all test scenarios, E2AN maintains a performance advantage over the baseline algorithms. Notably, the BiG-MRTA method exhibits a significant performance gap compared to other approaches. Further analysis reveals that

performance differentiation becomes subtler in simpler maps due to saturation effects. In simple scenarios (such as simple rooms and random obstacles), although the median of E2AN and CAM both reached 100%, the lower quartile (Q1) and minimum value of E2AN are better. This demonstrates that while most methods can handle general cases in simple environments, E2AN exhibits superior stability in handling edge cases.

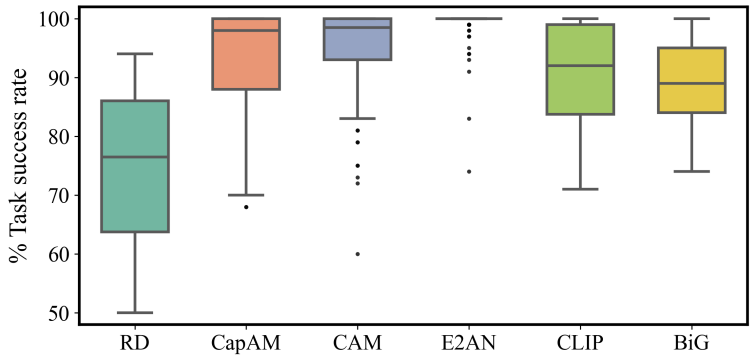


**Figure 4.** Boxplot of the task success rate for in-distribution testing in basic MRTA scenarios. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

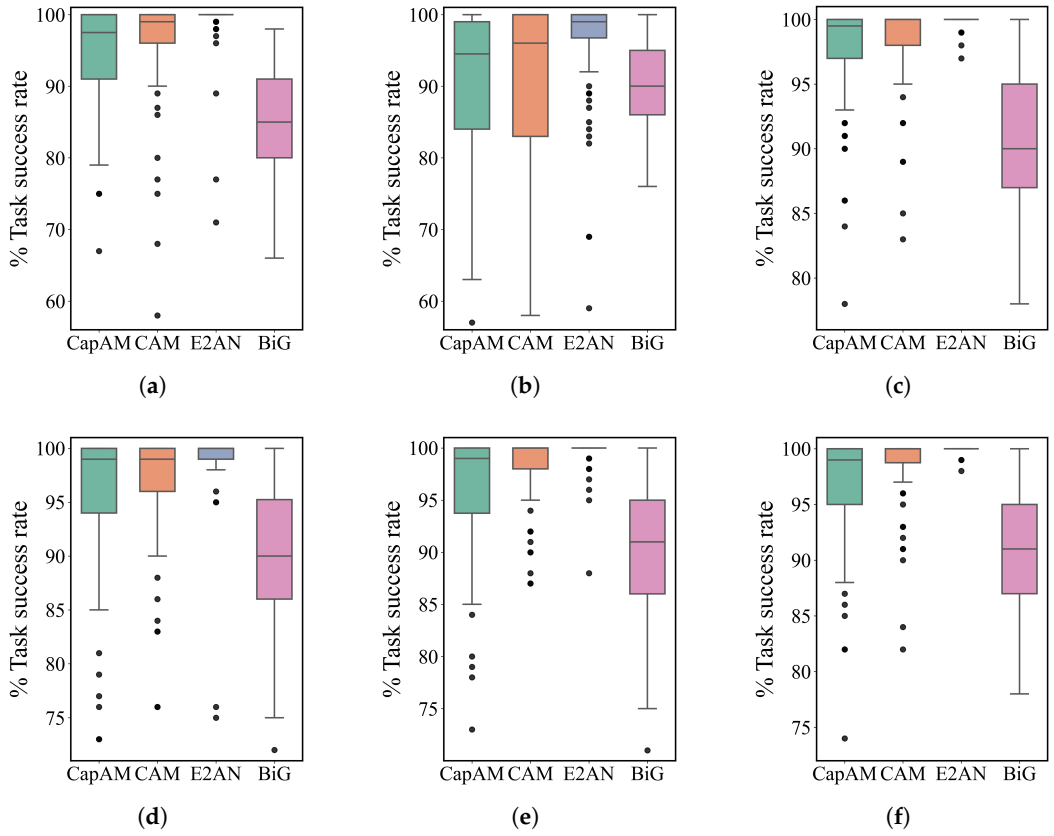


**Figure 5.** Boxplot of the task success rate for generalization tests in basic MRTA scenarios. (**a**) Scenarios in normal maze. (**b**) Scenarios in complex maze. (**c**) Scenarios in simple room. (**d**) Scenarios in complex room. (**e**) Scenarios in city. (**f**) Scenarios in random obstacles. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

Scalability Tests: We varied the problem scales by adjusting the number of tasks and robots, with configurations set as $(n_T, n_R) = (50, 4), (150, 10), (200, 14)$. As presented in Figure 6, E2AN is in a leading position at every scale.The comparison of experimental results in different scenarios shows that the task success rates of all algorithms increase as

the problem scale expands. The underlying reason lies in the fact that when the map size is fixed, increasing the number of robots will enhance the distribution density of service nodes, thereby reducing the difficulty of achieving full coverage of tasks.

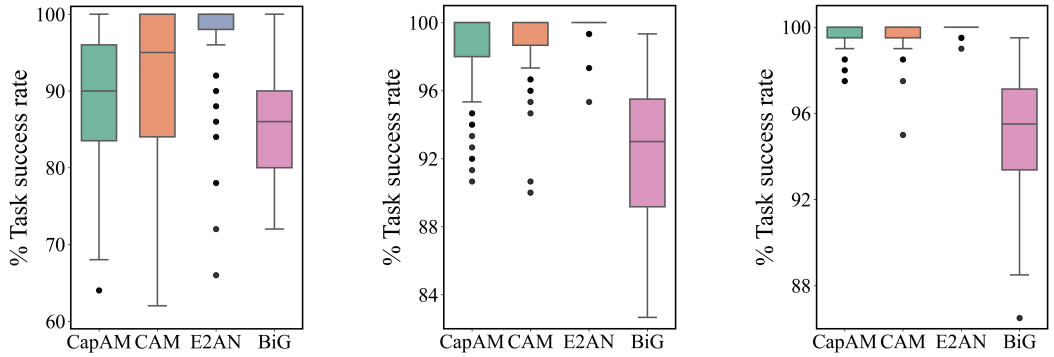


**Figure 6.** Boxplot of the task success rate for scalability tests in basic MRTA scenarios. (**a**) Scenarios with 50 tasks and 4 robots. (**b**) Scenarios with 150 tasks and 10 robots. (**c**) Scenarios with 200 tasks and 14 robots. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

### *4.4. MRTA with Diverse Payloads*

To test the algorithm under complex constraints, we extended the experiments to scenarios with diverse payload requirements. Following the definitions in Section 2.1 exactly, the payload dimension was set to $K = 4$. The scenario was configured with 15 robots partitioned into 4 types, where the quantity of each type was randomly distributed in the interval $[2, 6]$. The 100 tasks were categorized based on supply-demand scarcity into general demand (25 tasks across 2 types) and specific demand (75 tasks across 4 types). Except for the payload attributes, the generation logic for other spatio-temporal parameters remains consistent with the basic scenario. For training, we selected the wild scenario shown in Figure 3h, as it aligns better with diverse payload requirements. Since the introduction of multi-dimensional payload vectors changes the node feature dimensions, the input layer of the encoder was adapted. The model was then retrained with all other hyperparameters kept unchanged to ensure a fair comparison.

Due to the computational complexity constraints of CLIP in solving high-dimensional problems, only RD, CapAM, and CAM were selected as baselines in this section. As illustrated in Figure 7, a comprehensive comparison shows that E2AN maintains a leading position in key metrics, including median task success rate and distribution variance. The results of the paired sample $t$-test further confirm that this performance difference is statistically significant.

We selected two unseen map environments (Figure 3a,b) for generalization tests, and adjusted the problem scales (number of tasks and robots) proportionally to $(50, 8)$ and $(200, 30)$ for scalability tests. The results in Figure 8, demonstrate that E2AN consistently achieves higher task success rates and narrower variance across all settings.

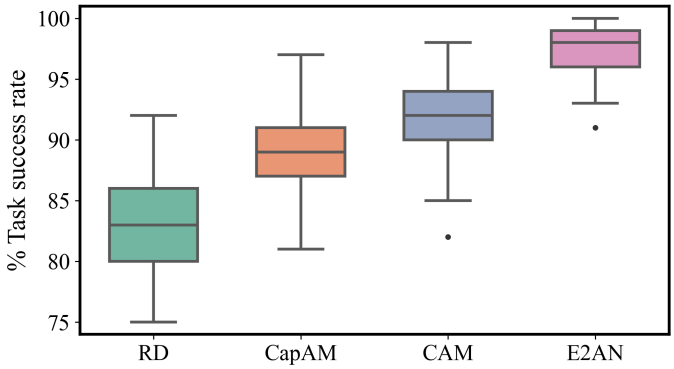


**Figure 7.** Boxplot of the task success rate for in-distribution testing in diverse payloads MRTA scenarios. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

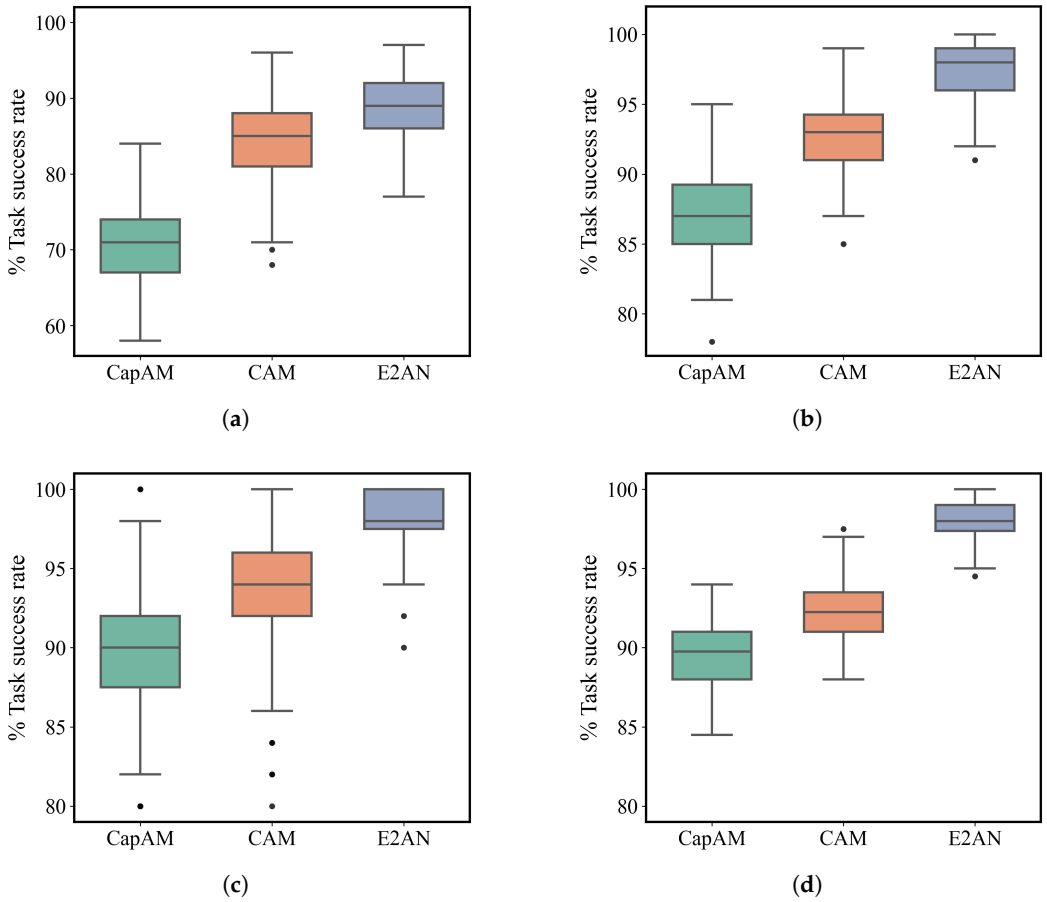


**Figure 8.** Boxplot of the task success rate for generalization tests and scalability tests in diverse payloads MRTA scenarios. (**a**) Scenarios in benchmark maze. (**b**) Scenarios in complex room. (**c**) Scenarios with 50 tasks and 8 robots. (**d**) Scenarios with 200 tasks and 30 robots. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

### *4.5. Ablation Studies*

To evaluate the E2HGAT encoder and attention decoder, we designed ablation studies by replacing these components with Feed-Forward Networks (FFN) to construct two ablation variants.

Encoder Ablation (FFE): For both robot and task nodes, independent MLP channels (containing two linear layers) were employed to map raw attributes to the embedding dimension, without performing any message passing between nodes.

Decoder Ablation (FFD): The Multi-Head Attention-based decoder was replaced with a feed-forward decoder. As shown in Equation (27), this module retains the original query vector construction logic. It concatenates the query with the embedding of each task, then reduces the feature dimension from $2D$ to 1 via a linear layer. Finally, it generates the unnormalized score (logit) for the task using a LeakyReLU activation. The subsequent masking strategy and decision-making logic remain strictly consistent with Section 3.2.

$$\text{logit}_i^{\text{task}} = \text{LeakyReLU}\Big(\text{FFN}\Big(\text{Concat}\Big(H_i^{\text{task}}, \mathcal{Q}\Big)\Big)\Big), \quad \text{where } i \in V_T. \tag{27}$$

We comprehensively evaluated the performance of E2AN and its ablation models under two scenario settings: basic MRTA and diverse payload MRTA. The evaluation covers in-distribution tests, generalization tests, and scalability tests. The experimental results are presented in Figure 9 (basic scenarios) and Figure 10 (diverse payload scenarios).

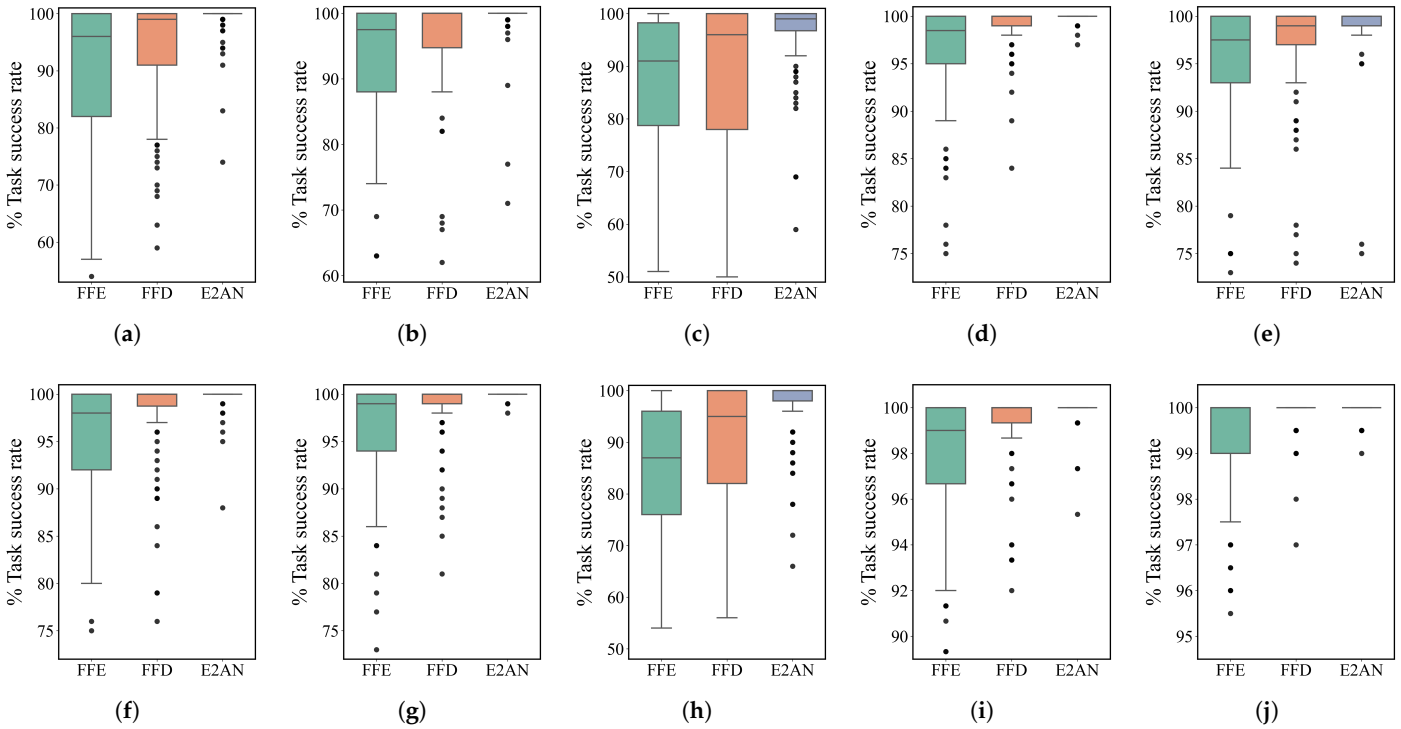


**Figure 9.** Boxplot of the task success rate for ablation tests in basic MRTA scenarios. (**a**) Scenarios like training. (**b**) Scenarios in normal maze. (**c**) Scenarios in complex maze. (**d**) Scenarios in simple room. (**e**) Scenarios in complex room. (**f**) Scenarios in city. (**g**) Scenarios in random obstacles. (**h**) Scenarios with 50 tasks and 4 robots. (**i**) Scenarios with 150 tasks and 10 robots. (**j**) Scenarios with 200 tasks and 14 robots. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

In basic scenarios, E2AN achieves optimal results across all evaluation metrics. Although the gap in median success rates narrows in some low-difficulty cases, E2AN still demonstrates robustness in handling the worst cases. Comparing the two ablation variants, FFD outperformed FFE. This observation suggests that the graph encoding mechanism contributes more to model performance than the attention decoding mechanism.

With the increased complexity of diverse payloads, the performance gap widens significantly, highlighting the limitations of the single module. This confirms that the complete framework is essential for capturing intricate matching relationships.

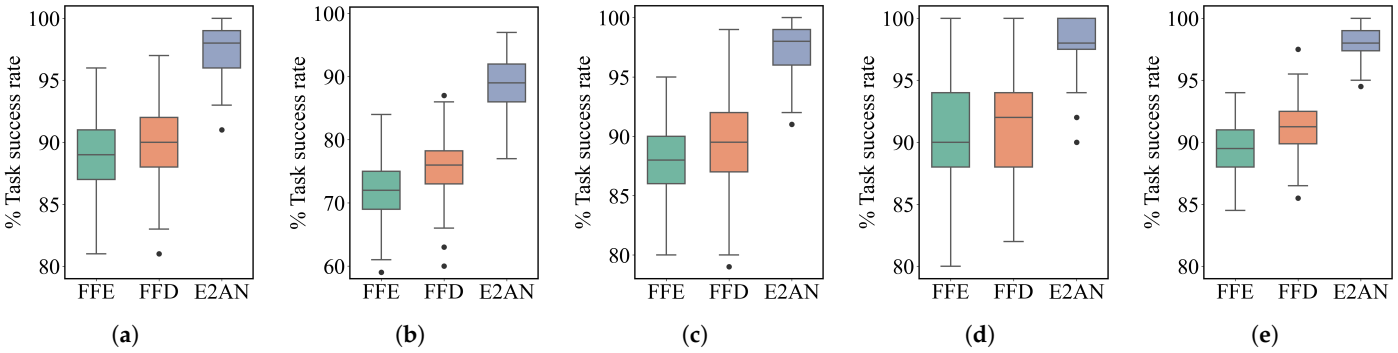


**Figure 10.** Boxplot of the task success rate for ablation tests in diverse payloads MRTA scenarios. (**a**) Scenarios like training. (**b**) Scenarios in benchmark maze. (**c**) Scenarios in complex room. (**d**) Scenarios with 50 tasks and 8 robots. (**e**) Scenarios with 200 tasks and 30 robots. Different colors represent different algorithms as labeled on the x-axis, and the dots denote outliers.

### *4.6. Analysis of Computational Efficiency and Scalability*

Tables 2 and 3 provide a detailed analysis of inference latency under varying problem scales and scenario types. The results from the six distinct data groups indicate that although inference latency trends upward with problem scale, this growth is notably nonlinear. For instance, quadrupling the number of tasks from 50 to 200 results in a marginal increase of only approximately 2 ms, confirming that computational complexity does not escalate linearly with input size. Moreover, the transition from basic to diverse payload scenarios incurs minimal computational overhead, demonstrating that E2AN maintained stable performance regardless of problem size or constraint complexity.

Optimization-based methods, such as CLIP, operate as global planners that must solve the entire problem instance before generating a final executable plan. This batch-processing paradigm prevents them from making immediate decisions during execution. In contrast, E2AN employs an asynchronous sequential decision-making framework, modeling task allocation as an event-driven online process where decisions are triggered instantaneously only when a robot becomes idle. Given this operational paradigm, inference latency, which measures the computation time required for a single decision step, was adopted as the primary metric for evaluating real-time capability in dynamic environments.

We evaluated real-time performance by calculating the average inference latency per decision step across the test sets for each scenario configuration. The performance metrics reported in this section were obtained using the workstation configuration detailed in Section 4.1. The results are summarized in Tables 2 and 3.

**Table 2.** Comparison of average inference latency across different algorithms and ablation variants.

| Method | CapAm | CAM | E2AN | FFE | FFD |
|---|---|---|---|---|---|
| Inference Latency | 2.43 ms | 2.23 ms | 11.09 ms | 0.74 ms | 10.64 ms |

Table 2 compares the inference latency of E2AN against baselines and ablation variants. Although E2AN (11.09 ms) entails a higher computational cost than simpler homogeneous methods like CapAM and CAM, or the ablation variant FFE, this difference is primarily driven by the E2HGAT encoder, as evidenced by the performance gap between FFE and FFD. However, this latency remains at the millisecond level, which is negligible compared to robot travel time. This indicates that E2AN effectively exploits potential idle time during system operation. Thus, trading a slight increase in computation for significant gains in decision quality is a justifiable strategy.

**Table 3.** Comprehensive analysis of computational efficiency under varying scenario types and problem scales.

| Scenario Configuration | Scale ($n_R/n_T$) | Latency (ms) | Peak VRAM (MB) |
|---|---|---|---|
| Basic Scenario | 4/50 | 10.57 | 1105 |
| | 7/100 | 11.09 | 1141 |
| | 14/200 | 12.34 | 1249 |
| Diverse Payload | 8/50 | 11.00 | 1349 |
| | 15/100 | 11.74 | 1367 |
| | 30/200 | 12.76 | 1503 |
| Large Scale Extension | 35/500 | 27.51 | 2079 |
| | 53/750 | 49.95 | 3255 |
| | 70/1000 | 85.12 | 4985 |
| | 100/1000 | 119.90 | 6551 |

To further investigate the hardware feasibility under extreme conditions, we extended the experimental scope beyond the generalization tests in Section 4.3. While previous results established that the success rate remains stable given the consistent map density, this section focuses on the trends in computational cost as the problem scale of the basic scenario expands up to 100 robots and 1000 tasks. The comprehensive results covering basic scenarios (at standard and large scales) and diverse payload scenarios are summarized in Table 3.

The results indicate that the system exhibits robust scalability with a predictable cost growth pattern. By comparing the Basic and Diverse Payload scenarios, it is evident that increasing constraint complexity incurs minimal computational overhead regarding latency, although it introduces a slight constant increase in memory usage due to the additional feature dimensions. Regarding memory consumption, the data reveals a quadratic growth trend governed by the $O(n_R \times n_T)$ space complexity of the interaction matrix in the graph attention mechanism. It is important to note a fixed base overhead of approximately 1100 MB required for the framework context and model weights; this base cost dominates at smaller scales but becomes less significant as the dynamic portion increases with problem size. Despite this growth, the peak memory usage in the largest tested scenario (100 robots, 1000 tasks) reaches 6.55 GB. Since the algorithm is designed to be deployed on a centralized workstation acting as a global scheduler, this consumption is well within the capacity of standard high-performance GPUs specified in Section 4.1.

Furthermore, while the inference latency rises to approximately 120 ms at the maximum scale, this delay remains operationally viable. Given the asynchronous, event-triggered nature of the proposed framework, decisions are required only at discrete moments when a robot completes a task. Consequently, this millisecond-level computation time is negligible compared to the physical execution time of tasks ranging from seconds to minutes, ensuring seamless operational continuity even in large-scale deployments.

## 5. Discussion

This study focused on the MRTA problem with coupled constraints of obstructed maps and deadlines while further handling diverse payload requirements. To address this problem, we formulated the task allocation as a Markov Decision Process over a heterogeneous graph and proposed the E2AN method. Tests indicate that E2AN outperforms baselines across basic and diverse-payload scenarios and maintains robustness even as problem complexity increases.

### 5.1. Mechanism Analysis

Further mechanism analysis reveals how the design of the framework contributes to its performance. The generalization ability of the model in unseen map scenes is related to the input form corresponding to the spatial constraints. This method does not use the raw map image but takes precise path costs as edge attributes to avoid overfitting to the topological structure of a specific map. Heterogeneous graph structures and asynchronous sequential decision-making frameworks are key factors for the scalability of the system under different problem scales. Among them, the heterogeneous graph structure allows nodes to flexibly achieve quantity changes by altering some connections while the asynchronous decision-making mechanism enables the decoder to dynamically adjust the query vector according to the real-time system state. The adaptability of E2AN to diverse payload requirements stems from its heterogeneous-graph encoder. Distinct from homogeneous methods relying on decoders, E2AN utilizes a heterogeneous graph encoder to capture robot–task dependencies and produces more discriminative features for downstream decisions.

### 5.2. Practical Deployment Framework

To clarify the applicability of the proposed algorithm in physical environments, we delineate its operational positioning within a hierarchical control architecture. In this framework, E2AN functions as a centralized global scheduler deployed on a high-performance workstation. Crucially, this high-level decision-making module is structurally decoupled from the low-level motion control and execution of individual robots. The specific operational workflow is detailed as follows:

- Input Stream: The system receives real-time state updates from the robot fleet via a communication network. These updates include robot poses obtained from localization algorithms, current operational status, and dynamic payload attributes. This mechanism allows the system to support real-time updates for scenarios involving payload switching or payload degradation. Simultaneously, the environmental map is maintained globally. If onboard sensors detect new static obstacles, the global map is updated and the edge weights in the heterogeneous graph are recalculated using global path planning algorithms based on the latest map snapshot.
- Decision Process: The E2AN policy is triggered asynchronously. Whenever a robot completes a task or a new agent joins the network, the scheduler constructs the current state graph $S_t$ and executes inference. As analyzed in Section 4.6, this inference occurs on the workstation with millisecond-level latency to ensure immediate responsiveness.
- Output Stream: The scheduler outputs a target assignment ID to the specific robot. The onboard local planner of the robot then executes the specific navigation. This layer handles sensor noise, localization correction, and immediate avoidance of dynamic obstacles such as pedestrians during execution.

This hierarchical separation ensures robustness. Although the global scheduler assumes a deterministic transition model, the asynchronous event-triggered mechanism prevents error accumulation. Even if sensor noise or dynamic obstacles cause execution delays at the robot level, these deviations simply delay the next decision trigger without invalidating the global logic because the scheduler always acts on the confirmed real-time state.

### 5.3. Limitations and Future Directions

Despite the promising simulation results and the feasible deployment framework, this study has several limitations that define the scope for future research:

- Dependence on Global Environmental Models: The framework relies on a consistent global map and stable communication to compute precise path costs. Consequently, its adaptability may be compromised in partially observable scenarios such as autonomous exploration or in highly dynamic environments where topological structures fluctuate rapidly. Future research could integrate predictive traffic models into the edge attribute construction to enhance robustness against environmental uncertainty.
- Computational Scalability Limits: The interaction mechanism within the graph attention network entails a memory complexity that grows quadratically with the system scale. This characteristic imposes an upper bound on the maximum supportable problem size and necessitates the deployment on centralized high-performance workstations. While suitable for standard logistics scenarios, this computational cost restricts infinite scalability for massive-scale swarms.
- Complex Collaborative Constraints: While the current model supports diverse payload requirements via node attributes, it assumes a one-to-one mapping between robots and payloads. Complex capacity trade-offs or multi-robot coalition tasks are not explicitly modeled. Future research could extend the action space and graph schema to address these combinatorial optimization challenges.
- Real-World Validation: The absence of physical experiments limits the evaluation of the model regarding resilience to non-Gaussian noise and communication instability found in the real world. Future studies are encouraged to deploy E2AN on a physical multi-robot testbed to validate the proposed architecture and explore Sim-to-Real transfer techniques to bridge the gap between simulation training and physical deployment.

## 6. Conclusions

This study proposes E2AN to address the MRTA problem characterized by temporal and environmental spatial constraints, with the flexibility to accommodate diverse payload requirements. By constructing a heterogeneous graph MDP model that supports asynchronous sequential decision-making, we transform the high-dimensional combinatorial optimization problem into a learnable sequential decision process. Experimental results show that the proposed method consistently outperforms existing baselines across various map topologies and problem scales. In particular, ablation and extended experiments verify that the core E2HGAT encoder mitigates spatial information distortion in complex environments by incorporating spatial edge attributes within a heterogeneous graph framework. Moreover, the analysis of inference latency demonstrates that the method maintains computational efficiency, validating its feasibility for online deployment.

However, specific costs and limitations exist. The graph attention mechanism entails a quadratic growth in memory consumption relative to the system scale, which imposes a theoretical limit on infinite scalability and necessitates high-performance centralized hardware. Additionally, the reliance on global map availability may constrain robustness in partially observable or highly dynamic settings.

This study provides an effective solution method with strong generalization, scalability, and real-time capability for MRTA in complex dynamic environments.

**Author Contributions:** Conceptualization, Y.H. and D.L.; Data curation, Y.H.; Formal analysis, Y.H. and D.L.; Funding acquisition, D.L.; Investigation, Y.H. and J.L. (Jinhong Li); Methodology, Y.H.; Project administration, D.L.; Resources, Y.H., J.L. (Junxiang Li) and T.W.; Software, Y.H.; Supervision, D.L.; Validation, Y.H. and J.L. (Jinhong Li); Writing—original draft, Y.H.; Writing—review & editing, D.L., J.L. (Junxiang Li) and T.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author. Part of the data are not publicly available due to our laboratory's confidentiality agreement and policies.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| MRTA | Multi-Robot Task Allocation |
| ST-SR-TA | single-task robots, single-robot tasks, and time-extended assignments |
| ST-MR-TA | single-task robots, multi-robot tasks, and time-extended assignments |
| TSP | Traveling Salesman Problem |
| VRP | Vehicle Routing Problem |
| RGAT | Relational Graph Attention Network |
| E2AN | Edge-Enhanced Attention Network |
| E2HGAT | Edge-Enhanced Heterogeneous Graph Attention Network |
| MDP | Markov Decision Process |
| MILP | Mixed Integer Linear Programming |
| MHA | Multi-Head Attention |
| IQR | interquartile range |
| FFN | Feed-Forward Network |

## Appendix A. Sensitivity Analysis of Encoder Layers

This appendix provides a detailed sensitivity analysis of the number of convolutional layers ($L$) in the E2HGAT encoder. This hyperparameter is critical as it determines the receptive field of the Graph Neural Network, influencing both the model's ability to capture topological dependencies and its computational efficiency. We evaluated models configured with $L = 1$, $L = 2$, and $L = 3$ on the standard test set.

The impact of layer depth on task success rate is illustrated in Figure A1. The model configured with a single layer ($L = 1$) consistently underperforms compared to deeper architectures, confirming that sufficient depth is essential for robust generalization. This limitation is attributed to the restricted receptive field, which prevents the effective aggregation of information from indirect neighbors (e.g., resolving conflicting relationships between multiple robots targeting the same task). Increasing the depth to $L = 2$ yields a substantial improvement in success rates, indicating that a two-hop propagation is sufficient to capture critical local topological constraints. However, further increasing the depth to $L = 3$ results in negligible performance gains, suggesting a performance saturation where additional layers offer no further benefit and may potentially introduce over-smoothing risks.

In addition to performance, we analyzed the computational cost, as shown in Table A1. The data indicates that peak GPU memory consumption remains consistent across all configurations (≈1140 MB), demonstrating that increasing the number of encoder layers has a negligible impact on memory overhead. However, the inference latency increases linearly with the number of layers. The 3-layer model increases the inference time by approximately 35% compared to the 2-layer model (15.00 ms vs. 11.09 ms) without providing a performance gain. Therefore, to balance decision accuracy with real-time responsiveness, we determined $L = 2$ to be the optimal parameter for the E2HGAT encoder.

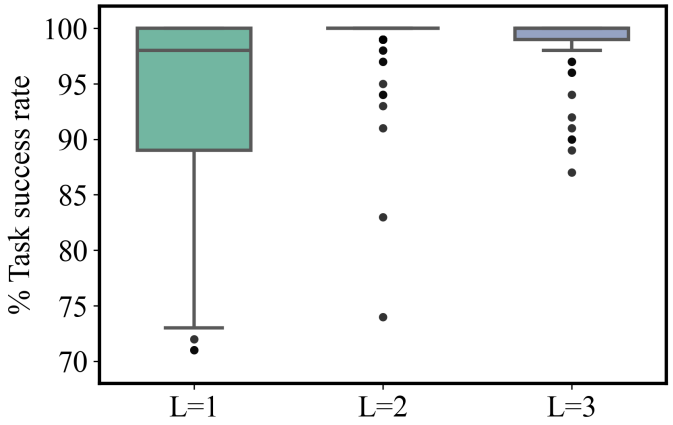


**Figure A1.** Boxplot of task success rates on the standard test set across different encoder layer counts ($L = 1, 2, 3$). The median success rate saturates at $L = 2$, while $L = 1$ shows significantly lower performance and higher variance. Different colors represent algorithms with different layer counts as labeled on the x-axis, and the dots denote outliers.

**Table A1.** Computational efficiency analysis with varying encoder layers.

| Encoder Layers ($L$) | Inference Latency (ms) | Peak GPU Memory (MB) |
|---|---|---|
| 1 | 6.54 | 1140 |
| 2 | 11.09 | 1141 |
| 3 | 15.00 | 1143 |

## References


1. Gerkey, B.P.; Matarić, M.J. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *Int. J. Robot. Res.* **2004**, *23*, 939–954. [CrossRef]
2. Kim, J.; Son, H.I. A Voronoi Diagram-Based Workspace Partition for Weak Cooperation of Multi-Robot System in Orchard. *IEEE Access* **2020**, *8*, 20676–20686. [CrossRef]
3. Li, K.; Liu, T.; Ram Kumar, P.N.; Han, X. A Reinforcement Learning-Based Hyper-Heuristic for AGV Task Assignment and Route Planning in Parts-to-Picker Warehouses. *Transp. Res. E Logist. Transp. Rev.* **2024**, *185*, 103518. [CrossRef]
4. Gao, S.; Wu, J.; Ai, J. Multi-UAV Reconnaissance Task Allocation for Heterogeneous Targets Using Grouping Ant Colony Optimization Algorithm. *Soft Comput.* **2021**, *25*, 7155–7167. [CrossRef]
5. Tian, X.; Xu, T.; Luo, X.; Jia, Y.; Yin, J. Multi-UAV Reconnaissance Task Allocation in 3D Urban Environments. *IEEE Access* **2024**, *12*, 30989–30999. [CrossRef]
6. Wang, R.; Zhao, D.; Min, B.-C. Initial Task Allocation for Multi-Human Multi-Robot Teams with Attention-Based Deep Reinforcement Learning. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 7915–7922.
7. Wang, R.; Zhao, D.; Gupte, A.; Min, B.-C. Initial Task Allocation in Multi-Human Multi-Robot Teams: An Attention-Enhanced Hierarchical Reinforcement Learning Approach. *IEEE Robot. Autom. Lett.* **2024**, *9*, 3451–3458. [CrossRef]
8. Zhou, C.; Li, J.; Shi, M.; Wu, T. Multi-Robot Path Planning Algorithm for Collaborative Mapping under Communication Constraints. *Drones* **2024**, *8*, 493. [CrossRef]
9. Nunes, E.; Manner, M.; Mitiche, H.; Gini, M. A Taxonomy for Task Allocation Problems with Temporal and Ordering Constraints. *Robot. Auton. Syst.* **2017**, *90*, 55–70. [CrossRef]
10. Ghassemi, P.; Chowdhury, S. Multi-Robot Task Allocation in Disaster Response: Addressing Dynamic Tasks with Deadlines and Robots with Range and Payload Constraints. *Robot. Auton. Syst.* **2022**, *147*, 103905. [CrossRef]
11. Wang, H.; Li, Y.; Li, S.; Xu, G. Research on Multiple AUVs Task Allocation with Energy Constraints in Underwater Search Environment. *Electronics* **2024**, *13*, 3852. [CrossRef]
12. Martin, J.G.; Muros, F.J.; Maestre, J.M.; Camacho, E.F. Multi-Robot Task Allocation Clustering Based on Game Theory. *Robot. Auton. Syst.* **2023**, *161*, 104314. [CrossRef]
13. Hu, J.; Bhowmick, P.; Jang, I.; Arvin, F.; Lanzon, A. A Decentralized Cluster Formation Containment Framework for Multi-robot Systems. *IEEE Trans. Robot.* **2021**, *37*, 1936–1955. [CrossRef]

14. Cui, W.; Li, R.; Feng, Y.; Yang, Y. Distributed Task Allocation for a Multi-UAV System with Time Window Constraints. *Drones* **2022**, *6*, 226. [CrossRef]
15. David, J.; Valencia, R. Simultaneous Path Planning and Task Allocation in Dynamic Environments. *Robotics* **2025**, *14*, 17. [CrossRef]
16. Hu, Y.; Wu, X.; Zhai, J.; Lou, P.; Qian, X.; Xiao, H. Hybrid Task Allocation of an AGV System for Task Groups of an Assembly Line. *Appl. Sci.* **2022**, *12*, 10956. [CrossRef]
17. Liu, F.; Xu, W.; Feng, Z.; Yu, C.; Liang, X.; Su, Q.; Gao, J. Task Allocation and Path Planning Method for Unmanned Underwater Vehicles. *Drones* **2025**, *9*, 411. [CrossRef]
18. Guo, H.; Miao, Z.; Ji, J.; Pan, Q. An Effective Collaboration Evolutionary Algorithm for Multi-Robot Task Allocation and Scheduling in a Smart Farm. *Knowl.-Based Syst.* **2024**, *289*, 111474. [CrossRef]
19. Hwang, N.E.; Kim, H.J.; Kim, J.G. Centralized Task Allocation and Alignment Based on Constraint Table and Alignment Rules. *Appl. Sci.* **2022**, *12*, 6780. [CrossRef]
20. Lin, J.; Li, P.; Dai, J.; Liu, S.; Tian, W.; Li, B.; Wang, C. Multi-Robot Cooperative Assembly Task Planning for Large-Scale Aerospace Structures. In Proceedings of the 2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC), Beijing, China, 19–20 November 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 795–800.
21. Kool, W.; van Hoof, H.; Welling, M. Attention, Learn to Solve Routing Problems! In Proceedings of the 7th International Conference on Learning Representations (ICLR 2019), New Orleans, LA, USA, 6–9 May 2019.
22. Wang, Z.; Gombolay, M. Learning Scheduling Policies for Multi-Robot Coordination with Graph Attention Networks. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4509–4516. [CrossRef]
23. Paul, S.; Ghassemi, P.; Chowdhury, S. Learning Scalable Policies over Graphs for Multi-Robot Task Allocation Using Capsule Attention Networks. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 8815–8822.
24. Paul, S.; Chowdhury, S. Learning to Allocate Time-Bound and Dynamic Tasks to Multiple Robots Using Covariant Attention Neural Networks. *J. Comput. Inf. Sci. Eng.* **2024**, *24*, 091005. [CrossRef]
25. Wang, Z.; Liu, C.; Gombolay, M. Heterogeneous Graph Attention Networks for Scalable Multi-Robot Scheduling with Tempospatial Constraints. *Auton. Robot.* **2022**, *46*, 249–268. [CrossRef]
26. Wang, Z.; Chen, J.; Chen, H. EGAT: Edge-Featured Graph Attention Network. In Proceedings of the 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, 14–17 September 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 253–264.
27. Hu, Z.; Dong, Y.; Wang, K.; Sun, Y. Heterogeneous Graph Transformer. In Proceedings of the Web Conference 2020 (WWW '20), Taipei, Taiwan, 20–24 April 2020; ACM: New York, NY, USA, 2020; pp. 2704–2710.
28. Gao, W.; Yu, Z.; Wang, T.; Wang, L.; Cui, H.; Guo, B.; Xiong, H. GNN-Based Deep Reinforcement Learning for Computation Task Scheduling in Autonomous Multi-Robot Systems. *J. Syst. Archit.* **2025**, *168*, 103534. [CrossRef]
29. Ma, Z.; Gong, H.; Xiong, J.; Wang, X. Heterogeneous Multiagent Task Allocation Based on Graph-Based Convolutional Assignment Neural Network. *IEEE Internet Things J.* **2025**, *12*, 17281–17299. [CrossRef]
30. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural Message Passing for Quantum Chemistry. In Proceedings of the 34th International Conference on Machine Learning (ICML 2017), Sydney, Australia, 6–11 August 2017; PMLR: Cambridge, MA, USA, 2017; pp. 1263–1272.
31. Park, B.; Kang, C.; Choi, J. Cooperative Multi-Robot Task Allocation with Reinforcement Learning. *Appl. Sci.* **2021**, *12*, 272. [CrossRef]
32. Wang, K.; Shen, W.; Yang, Y.; Quan, X.; Wang, R. Relational Graph Attention Network for Aspect-based Sentiment Analysis. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 3229–3238.
33. Stern, R.; Sturtevant, N.; Felner, A.; Koenig, S.; Ma, H.; Walker, T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T.K.; et al. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. *Proc. Int. Symp. Comb. Search* **2021**, *10*, 151–158. [CrossRef]
34. Mitiche, H.; Boughaci, D.; Gini, M. Iterated Local Search for Time-Extended Multi-Robot Task Allocation with Spatio-Temporal and Capacity Constraints. *J. Intell. Syst.* **2019**, *28*, 347–360. [CrossRef]
35. Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual;* Gurobi: Beaverton, OR, USA, 2024.