Exhaustiveness Does Not Necessarily Mean Better: Selective Task Planning for Multi-robot Systems

Xinye Xiong*, Xingyao Han*, Zhe Liu and Hesheng Wang

Abstract— **The performance of multi-robot systems heavily relies on efficient task allocation and motion coordination. However, for a group of a large number of robots, finding the optimal solution is inevitably time-consuming and may become impossible. Recognizing that tasks vary in their impact on system performance, our main idea is to identify their critical subset that significantly influences the entire system, and enhance task allocation efficiency by optimally planning critical tasks while distributing the remaining tasks randomly or with simple strategies. We call this approach Selective Multi-robot Task Planning (SMTP), which contributes to significantly reducing the computational requirements and solution time, and in the meanwhile, maintaining the system performance. In addition, by implementing a filtering mechanism based on the conditional expectation to eliminate less essential tasks, SMTP shows high extendability, maximizes task allocation efficiency, and balances computational efficiency and solution quality. Massive simulation and real experiments demonstrate that our algorithm decreases the computation time and maintains the properties of the base system. Large-scale experiments show that our approach only takes 11% computation time to reach 80% optimization objective and 94.8% traffic balance performance.**

I. Introduction

Multi-robot systems (MRSs) coordinate a group of robots to execute complex missions such as surveillance and foraging [1], improving flexibility and fault tolerance compared with single-robot systems [2]. Task planning, which typically includes task assignment, information and resource sharing, path planning, motion coordination, and local management, is crucial for a MRS [3]. In this paper, we investigate *Multi-Robot Task Planning (MRTP)* which studies the task assignment and path planning for mobile robots, taking robot capabilities, cooperation performance, task dependencies, and other requirements into consideration [4].

The MRTP problem, as an instance of the generalized assignment problem (GAP), is an NP-hard combinatorial optimization problem [5]. Although a few approaches, exemplified by the Hungarian algorithm for solving classic assignment problems [6] and the branch-and-bound method for integer optimization [7], yield optimal solutions, they cannot obtain solutions within the polynomial time. Therefore, massive robots and tasks in large-scale environments greatly strain the computational resources of the central processor [8]. Addressing MRTP problems with numerous tasks and agents necessitates compromising optimality in favor of computational feasibility, such as heuristic approaches [9] and evolutionary algorithms [10].

To alleviate the computational stress and enhance the efficiency of task planning in MRSs, we propose the Selective Multi-robot Task Planning (SMTP) algorithm, which diminishes the number of tasks requiring optimal scheduling. Diverging from existing methods that partition the original problem into sub-problems of comparable size and formulation (such as planning in batches), our technique for task pruning identifies a critical subset of tasks. Tasks inside the critical subset are meticulously planned to minimize the overall cost function, while the remaining tasks are randomly planned.

Our contribution can be summarized as follows:

1) We propose the idea of identifying the critical subset from the entire tasks and concentrating limited resources only on the critical subset, which contributes to significantly reducing the computational requirements while maintaining the system performance. The proposed SMTP algorithm is well formulated from both mathematical and statistical perspectives.

2) We provide comprehensive evaluations of the SMTP algorithm, which validates our efficiency and demonstrates our scalability and generalizability to various scenarios. In large-scale experiments, we achieve about 80% of the optimization objective and 94.8% of the traffic balance performance with only 11% computing time compared with the base model. Additionally, physical experiments are provided to demonstrate our practical applicability.

II. Related Works

A. Multi-Robot Task Planning

In large-scale robot networks, MRTP is typically divided into task allocation and path planning [4]. In this process, the system assigns transportation tasks to the most suitable robots and formulates trajectories for these robots to reach their designated task locations [11]. Traditional MRTP approaches decouple task allocation from path planning, autonomously isolating and addressing the two stages [12]. However, neglecting robot pathways during task allocation gives rise to localized area congestion or even robot deadlocks [13], especially in congested environments such as logistics warehouses [14]. Optimization over the combined task and motion planning allows for improved solutions compared with separately optimized algorithms [4]. For instance,

*The first two authors contribute equally

X. Xiong is with the School of Mathematical Sciences, Shanghai Jiao Tong University, China. X. Han and Z. Liu are with the MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China. H. Wang is with the Department of Automation, Shanghai Jiao Tong University, China. Corresponding author: H. Wang (E-mail: wanghesheng@sjtu.edu.cn)

combined task and motion planning [15] can adaptively manage tasks in a dynamics warehouse. In our previous work [16], [17], we introduced a task planning approach based on conflict graphs which effectively mitigates the issue of local area congestion. Nonetheless, the approach confronts challenges pertaining to real-time responsiveness within large-scale robot networks.

B. Scaling Down Problems Releasing Computation Load

Decomposition methods [18], [19] mitigate the peak computational burden of complex assignments by reducing the problem size considered each time. For instance, the authors in [20] construct a Lagrange relaxation problem and iteratively solve two easily solvable sub-problems. Dowdy et al. [21] assign files by tackling two sub-problems: determining optimal probabilities and finding a realization. Moreover, Randomized Decomposition [22] randomly partitions the decision variables of the nonlinear and non-convex problem into subsets, where variables in each subset are then optimized individually while keeping the remaining variables fixed. These decomposition methods reduce the complexity and improve computational efficiency by dividing an intricate problem into smaller sub-problems, which can be heuristically solved or yield optimal solutions [23].

C. Exhibiting Distinct Attention to Individual Tasks

In contrast to conventional approaches enduing all tasks indistinguishable values, task pruning mechanisms enhance task allocation efficiency [24], [25]. Researchers have agreed on the necessity to perform variable importance analysis [26]. Denninnart et al. [27] propose an algorithm with distinctive attention to individual tasks, bolstering the system's robustness. This method deliberately defers or drops tasks with low probabilities of meeting their respective deadlines, thereby increasing the likelihood of meeting deadlines for other tasks. By proactively optimizing the allocation of available resources, this mechanism prioritizes tasks with higher chances of timely completion, improving the overall reliability and effectiveness of the task allocation system.

III. Problem Formulation And System Structure

A. Problem Formulation

Given N_r available robots denoted as $\{r_n\}$, with $n \in [N_r] \triangleq \{1, \ldots, N_r\}$, and a bunch of assignments $\{a_i\}$, where $i \in [N_t]$, we investigate the planning of tasks to different robots. The task assignment vector

$$\boldsymbol{\mu} = (\mathbf{u}_1^T, \mathbf{u}_2^T, \ldots, \mathbf{u}_{N_t}^T)^T \in \{0,1\}^{N_t \times N_r}$$

is designed to represent a task allocation plan, where $(\mathbf{u}_i)_n \triangleq \boldsymbol{\mu}_{(i-1)*N_r+n} = 1$ if the task a_i is assigned to robot r_n, and $(\mathbf{u}_i)_n = 0$ otherwise ($\mathbf{u} \in \{0,1\}^{N_r}$). Denoting the objective function of the quadratic assignment problem by $f(\boldsymbol{\mu}) = \boldsymbol{\mu}^T \mathbf{M} \boldsymbol{\mu}$, without loss of generality, each element in matrix $\mathbf{M}$ can be assumed to be positive: $\mathbf{M} \subseteq \mathbb{R}_+^{N_t N_r \times N_t N_r}$.

The constraints of the assignment problem state that each task (*e.g.*, a_i) should be assigned to one and only one robot ($1^T \mathbf{u}_i = 1$), and each robot r_n can receive at most one task, *i.e.*, $\sum_{i=1}^{N_t} \boldsymbol{\mu}_{i,n} \leq 1$. The objective of the assignment problem is to find an optimal task assignment vector $\boldsymbol{\mu}^\star$ that minimizes the goal function $f(\boldsymbol{\mu})$, subject to the constraints above, *i.e.*,

$$\begin{aligned} \min_{\boldsymbol{\mu}} \quad & \boldsymbol{\mu}^T \mathbf{M} \boldsymbol{\mu} \\ \text{s.t.} \quad & \sum_{n=1}^{N_r} (\mathbf{u}_i)_n = \sum_{n=1}^{N_r} \boldsymbol{\mu}_{(i-1)*N_r+n} = 1, n \in [N_r] \\ & \sum_{i=1}^{N_t} (\mathbf{u}_i)_n = \sum_{i=1}^{N_t} \boldsymbol{\mu}_{(i-1)*N_r+n} \leq 1, \forall i \in [N_t] \\ & \boldsymbol{\mu} \in \{0,1\}^{N_t \times N_r}. \end{aligned} \tag{1}$$

B. Structure

The proposed approach will be evaluated on conflict-based allocation for solving pick-and-place problems. A unidirectional roadmap on a known physical environment is built [16]. The roadmap is represented as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node $v_i \in \mathcal{V}$ corresponds to a location in the 2D space, where a node can only be occupied by one robot at a time. The edge $e = (v_i, v_j) \in \mathcal{E}$ represents a straight path directly connecting nodes v_i and v_j, and the direction represents that robot can pass along. The roadmap is partitioned into sectors.

When a task a_i is assigned to a robot r_n, the system performs sector-level trajectory planning with predicted time, represented as the time extended sector path (TESP):

$$\Phi_{i,n} = \left\{ (s_{i,n}^1, h_{i,n}^1), (s_{i,n}^1, h_{i,n}^1), \ldots \right\}$$

Here, the jth tuple $(s_{i,n}^j, h_{i,n}^j)$ indicates that the robot r_n travels through the j-th sector in the planned sector-path $(s_{i,n}^j)$ from the timestep $h_{i,n}^j$ to $h_{i,n}^{j+1}$. Then for two task-robot pairs, their potential conflict can be examined as the *accumulated coincidence time* between the two planned TESPs is represented as $|\Phi_{i,n} \oplus \Phi_{j,m}| \in \mathbb{R}$ (see in [16]). The following relationship holds based on the former definitions:

$$0 \leq |\Phi_{i,n} \oplus \Phi_{j,m}| \leq \min \left\{ |\Phi_{i,n} \oplus \Phi_{i,n}|, |\Phi_{j,m} \oplus \Phi_{j,m}| \right\} \tag{2}$$

C. Conflict Matrix

The computation of the generalized conflict matrix $\boldsymbol{\Psi} \in \mathbb{R}^{N_r N_t \times N_r N_t}$ is based on the TESPs. $\boldsymbol{\Psi}_{(i,n),(j,m)}$ is defined to quantify the created interactive conflict when task a_i is assigned to robot r_n and task a_j is assigned to robot r_m:

$$\boldsymbol{\Psi}_{(i,n),(j,m)} \triangleq \boldsymbol{\Psi}_{(i-1)*N_r+n,(j-1)*N_r+m} = |\Phi_{i,n} \oplus \Phi_{j,m}|,$$

indicating the symmetry of the matrix: $\boldsymbol{\Psi} = \boldsymbol{\Psi}^T$. Furthermore, the element $\boldsymbol{\Psi}_{(i,n),(i,n)}$ corresponds to the predicted travel time of the path when task a_i is allocated to robot r_n.

IV. Methods

To reduce the computational cost of task allocation, we partition tasks into critical and trivial subsets. The allocation system carefully distributes tasks among the critical subset while randomly allocating the remaining tasks. This approach involves two main steps: identifying the critical subset and solving the reduced assignment problem.

A. Regarding a Statistical Process

Denote the index of the critical task subset by $\Lambda_t \subseteq [N_t]$. The main challenge lies in ranking tasks appropriately by predetermining the size of the critical subset as $|\Lambda_t| = k$. This entails finding the index set Λ_t, such that a sequence $\{\hat{\mathbf{u}}_p\}_{p\in\Lambda_t} \subseteq \{0,1\}^{N_r}$ can be obtained, satisfying

$$f(\boldsymbol{\mu}) - f(\boldsymbol{\mu}^\star) < \epsilon, \text{only if } \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t. \quad (3)$$

Partition the matrix $\mathbf{M}$ into $N_t \times N_t$ submatrices of size $N_r \times N_r$. The submatrix $\mathbf{M}^{i,j}$ summarizes all information between task a_i and task a_j. The reduced assignment problem can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{u}_p, p\in\Lambda_t} \quad & \sum_{i,j\in\Lambda_t} \mathbf{u}_i^T \mathbf{M}^{i,j} \mathbf{u}_j \\ \text{s.t.} \quad & \sum_{n=1}^{N_r} (\mathbf{u}_p)_n = 1, p \in \Lambda_t \\ & \sum_{p\in\Lambda_t} (\mathbf{u}_p)_n \leq 1, \forall n \in [N_r] \\ \text{where } & \mathbf{u}_p \in \{0,1\}^{N_r} \end{aligned} \quad (4)$$

Denote one group of optimal sub-problem solutions as $\hat{\mathbf{u}}_p, p \in \Lambda_t$. When allocating the remaining task a_q ($q \notin \Lambda_t$) is randomized, $\mathbf{u}_q$ can be regarded as a random variable following the discrete uniform distribution. Since none of the tasks can be assigned to robots having been assigned by tasks in Λ_t, the expectation of $\mathbf{u}_q$ equals

$$E[\mathbf{u}_q] = \frac{\hat{\boldsymbol{\delta}}}{N_r - k}, \text{where } \hat{\boldsymbol{\delta}} = \mathbf{1}_{N_t} - \sum_{p\in\Lambda_t} \hat{\mathbf{u}}_p.$$

Therefore, the problem in Eq. 3 can be statistically formulated as finding $\hat{\boldsymbol{\mu}}$ and Λ_t such that

$$P(f(\boldsymbol{\mu}) - f(\boldsymbol{\mu}^\star) \leq \epsilon | \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t) \geq \kappa,$$

where κ represents the probability that the resulted objective function is smaller than $f(\boldsymbol{\mu}^\star) + \epsilon$. Alternatively,

$$E[f(\boldsymbol{\mu}) | \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t] \leq \xi(\epsilon, \kappa, \boldsymbol{\mu}^\star),$$

where the right-hand side represents a threshold. We concentrate on the formulation of the conditional probability

$$\begin{aligned} & E[f(\boldsymbol{\mu}) | \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t] \\ = & \sum_{p\in\Lambda_t, q\notin\Lambda_t} \frac{\hat{\mathbf{u}}_p^T \mathbf{M}^{p,q} \hat{\boldsymbol{\delta}}}{N_r - k} + \sum_{q\notin\Lambda_t, p\in\Lambda_t} \frac{\hat{\boldsymbol{\delta}}^T \mathbf{M}^{q,p} \hat{\mathbf{u}}_p}{N_r - k} \\ & + \sum_{q_1,q_2\notin\Lambda_t} E[\mathbf{u}_{q_1}^T \mathbf{M}^{q_1,q_2} \mathbf{u}_{q_2} | \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t] \\ & + \sum_{p_1,p_2\in\Lambda_t} \hat{\mathbf{u}}_{p_1}^T \mathbf{M}^{p_1,p_2} \hat{\mathbf{u}}_{p_2}. \end{aligned} \quad (5)$$

For tasks q_1, q_2 not part of the critical subset, the conditional expectation of their cost can be expressed as

$$\begin{aligned} & E[\mathbf{u}_{q_1}^T \mathbf{M}^{q_1,q_2} \mathbf{u}_{q_2} | \mathbf{u}_p = \hat{\mathbf{u}}_p, \forall p \in \Lambda_t] \\ = & \frac{1}{A_{N_r-k}^2} (\hat{\boldsymbol{\delta}}^T \mathbf{M}^{q_1,q_2} \hat{\boldsymbol{\delta}} - \sum_{\mathbf{e}_n \notin \{\hat{\mathbf{u}}_p, p\in\Lambda_t\}} \mathbf{M}_{n,n}^{i,j}) \end{aligned}$$

When $(\mathbf{M}^{q,p})^T = \mathbf{M}^{p,q}$, the expression in Eq. 5 can be further simplified as

$$\begin{aligned} & \sum_{p_1,p_2\in\Lambda_t} \hat{\mathbf{u}}_{p_1}^T \mathbf{M}^{p_1,p_2} \hat{\mathbf{u}}_{p_2} + 2 \sum_{p\in\Lambda_t, q\notin\Lambda_t} \frac{\hat{\mathbf{u}}_p^T \mathbf{M}^{p,q} \hat{\boldsymbol{\delta}}}{N_r - k} \\ & + \sum_{q_1,q_2\notin\Lambda_t} \frac{1}{A_{N_r-k}^2} [\hat{\boldsymbol{\delta}}^T \mathbf{M}^{q_1,q_2} \hat{\boldsymbol{\delta}} - tr(\mathbf{M}^{q_1,q_2}) \\ & + \sum_{p\in\Lambda_t} \hat{\mathbf{u}}_p^T \mathbf{M}^{q_1,q_2} \hat{\mathbf{u}}_p] \end{aligned} \quad (6)$$

This expression resembles iterative methods, where the first term represents the reduced optimal function Eq. 4.

B. Intuitive Algorithm

When only pruning one task, we can examine the impact of randomizing a single task on the overall cost. Denote the task to be randomized as a_i and the robot assigned to perform this task as r_n, then one instance of Eq. 6 is:

$$\begin{aligned} & \sum_{p_1,p_2\in\Lambda_t} \hat{\mathbf{u}}_{p_1}^T \mathbf{M}^{p_1,p_2} \hat{\mathbf{u}}_{p_2} + 2 \sum_{p\neq i} \frac{\hat{\boldsymbol{\delta}}^T}{N_r - k} \mathbf{M}^{i,p} \hat{\mathbf{u}}_p \\ & + \frac{1}{A_{N_r-k}^2} [\hat{\boldsymbol{\delta}}^T \mathbf{M}^{i,i} \hat{\boldsymbol{\delta}} - tr(\mathbf{M}^{i,i}) + \sum_{p\in\Lambda_t} \hat{\mathbf{u}}_p^T \mathbf{M}^{i,i} \hat{\mathbf{u}}_p]. \end{aligned}$$

Inspired by this formulation:

$$\sum_{p\neq i} \hat{\boldsymbol{\delta}}^T \mathbf{M}^{i,p} \hat{\mathbf{u}}_p = \sum_{p\neq i} (\sum_{n\neq m} \mathbf{e}_n^T) \mathbf{M}^{i,p} \mathbf{e}_m = \sum_{p\neq i} \sum_{n\neq m} \mathbf{M}_{n,m}^{i,p},$$

we designed the following selective algorithm.

Convert the problem of reducing the number of tasks to a task insertion problem. We consider each task as an intruder and assess its impact on the "original system", including all tasks except the intruder. We aim to determine the extent to which the cost measure of the original system will increase upon introducing the new task a_i. The function $f_{i,j}$ is employed to quantify the task-level cost between task a_i and task a_j. Specifically, $f_{i,j}(n,m) = \mathbf{M}_{n,m}^{i,j}$ represents the cost that arises when the intruder task a_i is assigned to robot r_n and affects the event of "robot r_m being assigned with task a_j". Consequently, the average incremental cost for task a_j when task a_i is allocated to robot r_n can be expressed as the average of the cost:

$$g_{i,j}(n) = E_m[f_{i,j}(n,m) | i, n] = \frac{\sum_{m\neq n} \mathbf{M}_{n,m}^{i,j}}{N_r - 1}.$$

Furthermore, the average increased cost of "task a_i to robot r_n" for all other tasks can be expressed as

$$h_i(n) \triangleq E_j(g_{i,j}(n)) = \frac{\sum_{j\neq i} (\sum_{m\neq n} \mathbf{M}_{n,m}^{i,j})}{(N_t - 1)(N_r - 1)}.$$

Two alternatives emerge for task omission. First, we can exclude the task with the lowest expectation of incremental cost on the original system, namely $h(i) \triangleq E_n[h_i(n)]$. This task is disregarded due to its minimal impact on the overall system. Secondly, it is reasonable to eliminate the task characterized by the minimum cost variation, *i.e.*, $h(i) \triangleq var_n(h_i(n))$. Ignore this task because different allocations have a negligible effect on the overall system. The subsequent step involves iteratively choosing and removing the chosen task until the predetermined iteration limit is reached, or a substantial increase in the value becomes evident.

Algorithm 1 Critical Task Selection

Input: the cost matrix $\mathbf{M}$; the number of robots, tasks and critical tasks: N_r, N_t and k; the select function $c(M, i)$ *e.g.*, $c(\mathbf{M}, i) = \frac{\sum_n \sum_{j \neq i} (\sum_{m \neq n} \mathbf{M}_{n,m}^{i,j})}{N_r(N_r-1)(N_t-1)}$.
Output: Index set Λ_t;
$\Lambda \leftarrow \{1, 2, \ldots, N_t\}$
$\Lambda_t \leftarrow \emptyset$
$\mathbf{M} \leftarrow \mathbf{M}$
for $p \leftarrow 1$ to k **do**
  $q = argmin_i \, c(\mathbf{M}, i)$
  $\Lambda_t \leftarrow \Lambda_t \cup q$
end for
return Λ_t

In the following experiments, we will take the conflict-based model as an example. Concerning both the system conflict and individual traveling periods of the robots, a combination of the system and individual based on conflict measure will be applied as the applied rank function of task:

$$M \triangleq VAR[E_j(g_{i,j}(n)) + \alpha \mathbf{\Psi}_{n,n}^{i,i}]. \quad (7)$$

C. Solving Quadratic Assignment Problems (QAPs)

QAPs pose a significant computational challenge, prompting researchers to explore alternative approaches to obtain solutions within a reasonable time [28]. In this context, a fusion of quadratic optimization techniques and a greedy algorithm is employed on the reduced QAP. Initially, a quadratic optimization method derives an integer-constraint-free solution $\boldsymbol{\mu}$, capturing the inherent problem structure. Subsequently, several iterations are performed to legalize the solution. During each iteration, a ceiling operation is applied to set the maximum value $(\mathbf{u}_i)_n$ to one, indicating the assignment of task a_i to robot r_n. Concurrently, the related elements $\mathbf{u}_i$ and $\mathbf{u}_{j,n}, \forall j$ are set to zero, excluding the task a_i and robot r_n from consideration. This combined approach, integrating quadratic optimization and the greedy algorithm, offers a promising solution for efficiently resolving large-scale QAP instances with a balance between computational efficiency and solution quality.

V. Experiments

The proposed algorithm is evaluated in a conflict-based MRS for the pick-and-place problem [16]. It effectively alleviates the strain on the allocation system while preserving the overall performance, thus striking a balance between efficiency and resource allocation.

A. Pipeline

The overarching workflow for task allocation involves multiple steps: rough path plan, critical tasks selection, task assignment, and accomplishing tasks. First, all sector-paths Φ are generated to determine efficient routes within designated sectors for all robot-task combinations. The conflict matrix is computed using these sector-paths to quantify the interaction conflicts between robot-task assignments. Subsequently, the conflict matrix can be transformed into a more comprehensive cost function, incorporating optional constraints to capture complexities. A subset of critical tasks is selected based on the cost function to mitigate the computational burden. This task reduction technique focuses on comparable essential tasks to streamline the computational process. The reduced QAP is then solved, and the tasks are systematically allocated to appropriate robots based on the obtained solution. This allocation process guarantees the average performance of the allocation. Finally, the remaining unassigned tasks are randomly distributed among the leftover robots, completing the task planning process.

TABLE I: Map Setup

	Small	Middle	Large
Map Size	68×56	116×80	164×104
# Pick-up Station	432	1800	4032
# Workstation	216	432	648
# Sector	62	194	398
# Robot	216	360	504
# Task	500	750	1000

#: The Number of

B. Experiments Setup

Assuming that the task posting system randomly generates transportation tasks at an average rate [29], the task planning system collects these published tasks into a task pool and allocates them once the number of tasks in the pool reaches a predetermined batch size N_t. Three distinct simulation maps with identical structures but different sizes are considered (Table I). The small map has the size of 68×56 and is partitioned into 62 sectors with 216 workstations. In this scenario, each robot collects one cargo from a pick-up and transports it to a workstation. All experiments use 0.01 for α in Eq. 7 for consistency, and the batch size in Table II and Table III is set as 5. The simulations use a PYTHON 3.7 simulator on a GTX 3070 GPU with 32GB RAM.

C. Experimental Results

Table II records the simulations conducted by randomly assigning tasks outside the critical subset, considering the number of critical tasks ranging from 0 to 5. Experiments are repeated five times on different task sets to ensure reliability, recording means and variances. The variance exhibits a decreasing trend as the number of critical subsets increases, as Section IV-B indicates. The traffic density quantifies the congestion levels, affirming that our algorithm preserves the base model characteristics well. The execution time measures the average duration from task posting to completion [30], while the computation time records the average time needed for planning a batch of tasks.

To validate the adaptability of our approach, Table III compares the random strategy and nearest assign strategy on the large map. In each column block, the left column shows the random assigning of the remaining tasks, and the right column indicates the assignment to the nearest robot.

TABLE II: The Influence of the Number of Key Tasks on the Metrics of the Task Allocation System

	# Critical Tasks	**0**	**1**	**2**	**3**	**4**	**5**
S	**Computation Time**	0.09 (4.45e-6)	0.15 (1.56e-6)	0.21 (8.67e-6)	0.31 (8.89e-6)	0.46 (4.85e-5)	0.66 (6.93e-5)
	Execution Time	97.99 (2.26e+0)	90.97 (8.38e-1)	84.13 (1.48e+0)	78.26 (3.93e-1)	72.0 (7.93e-1)	67.9 (7.66e-1)
	Traffic Density	6.2 (4.98e-2)	6.13 (2.86e-2)	5.85 (9.88e-2)	5.45 (4.33e-3)	5.09 (1.23e-2)	4.74 (4.98e-3)
M	**Computation Time**	0.38 (3.50e-5)	0.47 (7.92e-6)	0.62 (2.80e-5)	0.9 (1.29e-4)	1.39 (7.42e-4)	2.34 (9.44e-4)
	Execution Time	158.91 (1.21e+0)	146.13 (7.14e-1)	134.57 (1.17e+0)	123.48 (8.56e-1)	113.79 (1.70e+0)	107.22 (1.45e+0)
	Traffic Density	4.82 (1.58e-2)	4.55 (1.80e-2)	4.19 (1.08e-2)	3.9 (6.05e-3)	3.64 (8.52e-3)	3.39 (7.79e-3)
L	**Computation Time**	0.97 (1.89e-4)	1.16 (9.66e-5)	1.46 (5.24e-5)	2.12 (3.79e-4)	3.77 (4.38e-3)	6.73 (3.87e-2)
	Execution Time	216.37 (4.12e+0)	195.94 (2.92e+0)	179.76 (3.14e+0)	163.1 (4.91e+0)	151.11 (2.28e+0)	142.7 (8.51e-1)
	Traffic Density	5.71 (1.79e-2)	5.42 (5.96e-3)	5.11 (7.82e-3)	4.71 (6.39e-3)	4.46 (2.74e-3)	4.17 (6.49e-3)

TABLE III: The Improvement Compared to Optimal

# CT†	**Computation Time**		**Total Cost**		**Traffic Density**	
	Random	Nearest	Random	Nearest	Random	Nearest
5 (optimal planning)	**6.73** (100%)		**164.88** (100%)		**4.17** (100%)	
4	3.77 (56.02%)	4.07 (60.48%)	189.13 (114.7%)	171.63 (104.1%)	4.46 (107.0%)	4.14 (99.28%)
3	2.12 (31.50%)	2.44 (36.26%)	236.63 (143.5%)	176.38 (107.0%)	4.71 (113.0%)	4.19 (100.5%)
2	1.46 (21.69%)	1.82 (27.04%)	284.63 (172.6%)	174.38 (105.8%)	5.11 (122.5%)	4.20 (100.7%)
1	1.16 (17.24%)	1.47 (21.84%)	289.00 (175.3%)	201.00 (121.9%)	5.42 (123.0%)	4.21 (101.0%)

† The Number of Critical Tasks

The total costs are the values of the objective function in Eq. 1. The performance on 5 and 4 critical task scenarios with random assignment demonstrates SMTP's effectiveness: a sacrifice of only 14.7% total cost and 7% traffic density saves 43.98% computation time. The random assignment has guaranteed the lower bound of the algorithm's performance while incorporating strategies, *e.g.*, the nearest robot assignment, can further improve its performance. In the experiments with the nearest strategy, traffic density, and total cost remain consistent, while the computation time decreases considerably, highlighting the adaptability of our algorithm. All in all, SMTP can highly reduce computation time and achieve similar performance.

In the large-scale experiment with a task batch size of 20, the algorithm is performed with the nearest strategy on the small map with more than 200 robots and 500 tasks (Table IV). Comparing the optimal baseline, by only considering 5 critical tasks, we can obtain that a sacrifice of only 5.5% traffic density saves 88.9% computation time (i.e., it only takes about 11% computation time to reach near 80% optimization objective and 94.8% traffic balance performance). Moreover, the listed total costs reveal a slightly decreasing optimality if treating the mathematical function as the evaluation metric. This demonstrates that only carefully assigning a small portion (5 out of 20) of tasks can achieve remarkable performance in a large-scale environment. The maintained traffic density highlights our advantage in preserving base model characteristics.

TABLE IV: The Improvement Compared to Optimal with Task Batch 20 and Nearest Strategy on Small Map

# CT	5	10	15	20
Computation Time	5.37 (11.1%)	9.73 (20.0%)	21.95 (45.2%)	48.56
Total Cost	363 (126.0%)	337.2 (117.1%)	319.3 (110.9%)	288.0
Traffic Density	2.11 (105.5%)	2.11 (105.5%)	2.07 (103.5%)	2.00

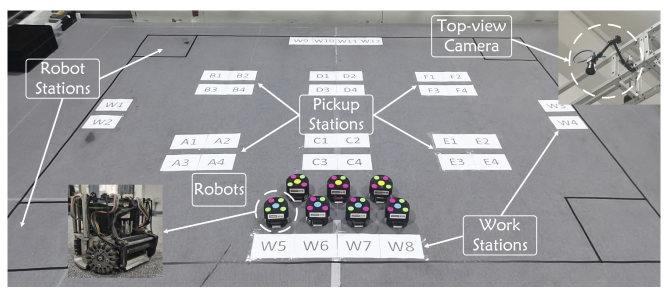

Fig. 1: Experiment System.

D. Real Experiments

To validate the proposed method's effectiveness, we conducted physical experiments using four robots positioned by four overhead cameras at different angles, as shown in Fig. 1. The experimental environment was divided into 12 zones, with each zone's movable area discretized into a connected grid of four cells. Each grid cell has a size of $0.3m \times 0.3m$, while the size of the robots is $0.2m \times 0.2m$. To ensure safety, we imposed a constraint that each grid cell could only be occupied by a single robot at any given time step, preventing collisions. In the experiments, four tasks were simultaneously assigned, two designated primary tasks while the remaining tasks were randomly allocated to the robots. The trajectory of our robots (Fig. 2) illustrates their coordinated movement without collisions, confirming SMTP's feasibility and practical applicability.

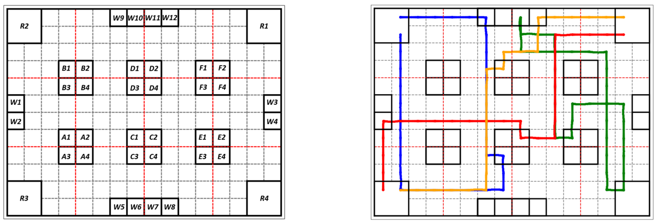

Fig. 2: Real robot trajectories recorded by top-view cameras.

VI. Conclusion

The proposed SMTP algorithm employs a two-pronged approach: it selectively assigns tasks within a critical task subset while randomly allocating the remaining tasks, thereby optimizing overall resource utilization. Extensive experiments have demonstrated its superior performance in reducing the computational requirements of the allocation system and preserving the fundamental characteristics of the base model. Furthermore, the identification of a pivotal task subset facilitates seamless integration into various contexts, endowing it with adaptability as a versatile module in any assignment framework. By incorporating the selection strategy with additional models, further enhancements would be achieved, leading to practical implementation and advancements in MRSs. Prospective researchers can explore alternative task elimination algorithms based on our formulations or investigate methodologies for determining the size of the critical subset in complex and dynamic environments.

In the future, we plan to extend our SMTP approach to heterogeneous problems.

References

[1] Rachael N. Darmanin and Marvin K. Bugeja. A review on multi-robot systems categorised by application domain. In *2017 25th Mediterranean Conference on Control and Automation (MED)*, pages 701–706, 2017.

[2] Kuppan Chetty RM, Ramya MM, and Mukund Nilakantan Janardhanan. Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems. *Industrial Robot: the international journal of robotics research and application*, 47(6):929–942, 2020.

[3] Eliahu Khalastchi and Meir Kalech. Fault detection and diagnosis in multi-robot systems: A survey. *Sensors*, 19(18), 2019.

[4] Luke Antonyshyn, Jefferson Silveira, Sidney Givigi, and Joshua Marshall. Multiple mobile robot task and motion planning: A survey. *ACM Computing Surveys*, 55(10):1–35, 2023.

[5] Dirk G. Cattrysse and Luk N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260–272, 1992.

[6] Ventseslav Kirilov Shopov and Vanya Dimitrova Markova. Application of hungarian algorithm for assignment problem. In *2021 International Conference on Information Technologies (InfoTech)*, pages 1–4, 2021.

[7] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations Research*, 14(4):699–719, 1966.

[8] Reza Javanmard Alitappeh and Kossar Jeddisaravi. Multi-robot exploration in task allocation problem. *Applied Intelligence*, 52(2):2189–2211, 2022.

[9] Omar Cheikhrouhou and Ines Khoufi. A comprehensive survey on the multiple traveling salesman problem: Applications, approaches and taxonomy. *Computer Science Review*, 40:100369, 2021.

[10] Muhammad Usman Arif and Sajjad Haider. A flexible evolutionary algorithm for task allocation in multi-robot team. In *Computational Collective Intelligence: 10th International Conference, ICCCI 2018, Bristol, UK, September 5-7, 2018, Proceedings, Part II 10*, pages 89–99. Springer, 2018.

[11] Zhe Liu, Shunbo Zhou, Hesheng Wang, Yi Shen, Haoang Li, and Yun-Hui Liu. A hierarchical framework for coordinating large-scale robot networks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6672–6677. IEEE, 2019.

[12] Ketan Savla, Emilio Frazzoli, and Francesco Bullo. On the point-to-point and traveling salesperson problems for dubins' vehicle. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 786–791. IEEE, 2005.

[13] Janardan Kumar Verma and Virender Ranga. Multi-robot coordination analysis, taxonomy, challenges and future scope. *Journal of intelligent & robotic systems*, 102:1–36, 2021.

[14] Xin Chen, Yanjie Li, and Lintao Liu. A coordinated path planning algorithm for multi-robot in intelligent warehouse. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2945–2950. IEEE, 2019.

[15] Ali Bolu and Ömer Korçak. Adaptive task planning for multi-robot smart warehouse. *IEEE Access*, 9:27346–27358, 2021.

[16] Zhe Liu, Huanshu Wei, Hongye Wang, Haoang Li, and Hesheng Wang. Integrated task allocation and path coordination for large-scale robot networks with uncertainties. *IEEE Transactions on Automation Science and Engineering*, 19(4):2750–2761, 2022.

[17] Zhe Liu, Hesheng Wang, Huanshu Wei, Ming Liu, and Yun-Hui Liu. Prediction, planning, and coordination of thousand-warehousing-robot networks with motion and communication uncertainties. *IEEE Transactions on Automation Science and Engineering*, 18(4):1705–1717, 2020.

[18] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[19] Vittorio Maniezzo, Marco Antonio Boschetti, and Thomas Stützle. The generalized assignment problem. In *Matheuristics: Algorithms and Implementations*, pages 3–33. Springer, 2021.

[20] Mingfang Ni, Zhanke Yu, Feng Ma, and Xinrong Wu. A lagrange relaxation method for solving weapon-target assignment problem. *Mathematical Problems in Engineering*, 2011.

[21] Lawrence W. Dowdy and Derrell V. Foster. Comparative models of the file assignment problem. *ACM Comput. Surv.*, 14(2):287–313, jun 1982.

[22] Krešimir Mihić, Kevin Ryan, and Alan Wood. Randomized decomposition solver with the quadratic assignment problem as a case study. *INFORMS Journal on Computing*, 30(2):295–308, 2018.

[23] Gary Froyland, Stephen J. Maher, and Cheng-Lung Wu. The recoverable robust tail assignment problem. *Transportation Science*, 48(3):351–372, 2014.

[24] Yung-Cheng Ma, Tien-Fu Chen, and Chung-Ping Chung. Branch-and-bound task allocation with task clustering-based pruning. *Journal of Parallel and Distributed Computing*, 64(11):1223–1240, 2004.

[25] Oliver Sinnen. Reducing the solution space of optimal task scheduling. *Computers Operations Research*, 43:201–214, 2014.

[26] Pengfei Wei, Zhenzhou Lu, and Jingwen Song. Variable importance analysis: A comprehensive review. *Reliability Engineering System Safety*, 142:399–432, 2015.

[27] Chavit Denninnart, James Gentry, Ali Mokhtari, and Mohsen Amini Salehi. Efficient task pruning mechanism to improve robustness of heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, 142:46–61, 2020.

[28] Mohamed Abdel-Basset, Gunasekaran Manogaran, Heba Rashad, and Abdel Nasser H Zaied. A comprehensive review of quadratic assignment problem: variants, hybrids and applications. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–24, 2018.

[29] Robert M. Nauss. Solving the generalized assignment problem: An optimizing and heuristic approach. *INFORMS Journal on Computing*, 15(3):249–266, 2003.

[30] Ernesto Nunes and Maria Gini. Multi-robot auctions for allocation of tasks with temporal constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), Feb. 2015.