IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Collaborative Task Allocation for Heterogeneous Multi-Robot Systems Through Iterative Clustering

David R. Martin, *Graduate Student Member, IEEE*, Brooks A. Butler, *Member, IEEE*, Scott Nivison, *Member, IEEE*, Magnus Egerstedt, *Fellow, IEEE*, Mohammad Abdullah Al Faruque, *Senior Member, IEEE*, and Pramod Khargonekar, *Life Fellow, IEEE*

***Abstract*—Multi-robot systems face the challenge of efficiently allocating teams of heterogeneous robots to tasks. The task allocation problem is complicated by collaborative interactions between robots where teams of robots develop emergent capabilities that enable them to complete tasks that would be inefficient or impossible for individual robots. To address these challenges, we present an iterative clustering algorithm for collaborative task allocation in heterogeneous multi-robot systems. This approach partitions the computationally intractable global optimization problem into smaller, tractable subproblems by iteratively forming clusters of robots and tasks, then optimizing assignments within each cluster. By ensuring robots remain clustered with their currently assigned tasks, we guarantee monotonic improvement in overall utility with each iteration. We analyze the convergence of the algorithm and characterize how cluster size constraints determine which suboptimal assignments could trap the algorithm. In simulation, iterative clustering consistently outperforms simulated annealing, and a group-based auction in both computation time and solution quality, and outperforms a hedonic game approach in solution quality.**

***Index Terms*—Multi-robot systems, task planning, multi-robot task allocation (MRTA), cooperating robots.**

I. Introduction

MULTI-ROBOT systems have demonstrated significant potential in numerous domains, including disaster relief [1], sensor fusion [2], construction [3], warehousing [4], [5], and beyond [6], [7]. Of particular interest are robot teams where collaboration enables new capabilities, allowing them to complete tasks that would be impossible or highly inefficient for individual robots. Recent examples highlight this synergistic potential: in [2], multiple drones equipped with cameras collaboratively estimate 3D object depth with higher accuracy than lidar, in [8], aquatic robots transport terrestrial robots across water bodies to reach otherwise inaccessible goals, in [3] and [5], robot teams collaborate to carry oversized payloads.

A challenge in collaborative multi-robot systems is the multi-robot task allocation (MRTA) problem, which is the process of determining optimal robot-to-task assignments that maximize system performance [7]. Researchers have addressed MRTA using diverse approaches, including the Hungarian algorithm [9], mixed integer programming [10], auctions [5], [11], game-theoretic approaches [12], [13], metaheuristics (such as simulated annealing [14] and evolutionary algorithms [15], [16]), clustering techniques [17], [18], and machine learning methods [3], [19]. Distributed algorithms such as group-based auctions [11] and game theoretic approaches [13] have shown a remarkable ability to scale efficiently to large systems.

Despite recent advances, existing algorithms struggle to account for collaborative synergy in multi-robot teams. Most existing approaches assume predetermined team sizes and capability requirements for each task [3], [10], [15]. However, in many systems, the optimal team size may be unclear. When resources are abundant and tasks are few, it may be advantageous to assign more resources to each task. Conversely, when resources are constrained, it may be necessary to reduce team sizes or omit less important tasks completely. Collaborative dynamics complicate resource allocation. Existing works that allow variable team sizes generally rely on assigning individual robots to tasks where they provide the greatest marginal contribution [12], [20]. However, these approaches struggle in situations where robots' contributions are not easily separable from those of their teammates. For instance, a task requiring three robots might yield no utility until all three are assigned. In general, these methods fail in scenarios requiring simultaneous multi-robot reassignments, where a single reassignment provides no benefit.

To account for collaborative team effects, we model the MRTA problem as a coalition structure generation problem [21], partitioning robots into coalitions assigned to each task with the goal of optimizing total utility. While coalition structure generation is NP-hard [22], we propose a novel iterative clustering algorithm that scales effectively. The proposed method directly addresses the key limitations of existing methods by: (1) providing a computationally efficient solution to the coalition structure generation problem, (2) modeling team utility as a nonlinear function that can be greater than the sum of the

Received 7 July 2025; accepted 22 October 2025. Date of publication 12 November 2025; date of current version 18 November 2025. This article was recommended for publication by Associate Editor J.-H. Lee and Editor C.-B. Yan upon evaluation of the reviewers' comments. This work was supported in part by the Air Force Research Laboratory, Munitions Directorate (RWTA), Eglin AFB, FL. *(Corresponding author: David R. Martin.)*

David R. Martin, Brooks A. Butler, Magnus Egerstedt, Mohammad Abdullah Al Faruque, and Pramod Khargonekar are with the Department of Electrical Engineering and Computer Science, Henry Samueli School of Engineering, University of California, Irvine, CA 92697 USA (e-mail: davidrm3@uci.edu; alfaruqu@uci.edu; pramod.khargonekar@uci.edu).

Scott Nivison is with Air Force Research Laboratory, Munitions Directorate, Eglin AFB Okaloosa County, FL 32542 USA.

Digital Object Identifier 10.1109/LRA.2025.3632086

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

individual robot contributions, and (3) determining appropriate team compositions for each task while accounting for this collaborative synergy.

The proposed method iteratively forms clusters of robots and tasks, then optimizes assignments within each cluster. We develop two variants of this approach: one that forms clusters randomly and another that uses heuristics to guide cluster formation. We prove that assignment utility increases monotonically with each iteration and analyze how cluster size constraints affect the algorithm's ability to escape suboptimal solutions. In simulation, the iterative clustering approach achieves 99.9% of optimal utility for small problems, outperforms exisiting methods for large problems, and exhibits fast convergence that scales effectively to problems with up to 1000 robots and 500 tasks.

II. System Model and Problem Formulation

This work focuses on ST-MR-IA [23] problems where ST indicates single-task robots (each robot can perform only one task at a time), MR denotes multi-robot tasks (tasks may be performed by multiple robots), and IA refers to instantaneous assignment (planning without considering future allocations).

A. System Model

Consider a set of ν robots $\mathcal{R} = \{r_1, r_2, \ldots, r_\nu\}$ to be assigned to a set of μ tasks $\mathcal{T} = \{t_1, t_2, \ldots, t_\mu\}$.

1) Assignments: The individual assignment of each robot r_i is represented as an integer $a_i \in \{0, 1, \ldots, \mu\}$ such that $a_i = 0$ means r_i is unassigned and $a_i = j$ for $j \neq 0$ means r_i is assigned to task t_j. A global assignment $A = \{S_0, S_1, \ldots, S_\mu\}$ partitions the set of all robots $\mathcal{R}$ into $\mu + 1$ teams $S_0, S_1, \ldots, S_\mu \subseteq \mathcal{R}$ where team $S_j = \{r_i \in \mathcal{R} : a_i = j\}$ contains the robots assigned to task t_j for $j \neq 0$ and $S_0 = \{r_i \in \mathcal{R} : a_i = 0\}$ contains all unassigned robots. Because each robot is assigned to no more than one task, an assignment $A = \{S_0, S_1, \ldots, S_\mu\}$ forms a partition of $\mathcal{R}$ meaning $S_\alpha \cap S_\beta = \emptyset$ for $\alpha \neq \beta$ and $\bigcup_{j=0}^{\mu} S_j = \mathcal{R}$. Let $\mathcal{A}$ denote the set of all possible assignments.

2) Robot Types: Each robot $r_i \in \mathcal{R}$ is characterized by a single robot type that represents different hardware or software capabilities (e.g. UAV, rover... etc.). Robot type for robot r_i is denoted $z_i \in \{1, 2, \ldots \kappa\}$, where κ is the total number of distinct robot types. Robots of the same type are considered identical in terms of their capabilities.

3) Teams: Each robot team S_j has an associated team composition vector $C_j = [c_1, c_2, \ldots, c_\kappa]$ where c_λ denotes the number of robots of type λ in the team. While the team size is variable, we constrain the maximum team size to an integer limit L: $|S_j| \leq L$ for $j \neq 0$. The number of unassigned robots $|S_0|$ is unconstrained.

4) Team Utility Modeling: Each task $t_j \in \mathcal{T}$ is associated with a benefit function $B_j(C) : \mathbb{N}_0^\kappa \to \mathbb{R}$, which specifies the utility of assigning a team of robots with team composition vector $C = [c_1, \ldots, c_\kappa]$ to task t_j. Intuitively, it captures how good a particular team is for completing the task. For example, consider a search and rescue mission with two robot types ($\kappa = 2$): drones and ground vehicles. A single drone may be unable to complete the rescue and provide no benefit ($B_j([1, 0]) = 0$). A team with one drone for victim localization and one ground vehicle for extraction could provide high utility ($B_j([1, 1]) = 100$), while adding another drone may provide quicker localization ($B_j([2, 1]) = 120$). Benefit functions thus provide a quantitative model to directly represent complex team interactions, including collaborative synergies and marginally decreasing returns. We do not impose any particular structure or form on the benefit functions since they depend on the specifics of tasks, robots, and mission priorities. Within the task assignment problem, benefit functions are treated as known inputs, under the assumption that the system designer can specify or evaluate them using domain expertise or data-driven methods.

Assumption 1: We assume that the benefit functions of all tasks, robot capabilities, and locations of robots and tasks are fully known and deterministic.

5) Team Formation Cost: Each task t_j has a team formation cost function $J_j(S_j) : 2^{|\mathcal{R}|} \to \mathbb{R}$ which quantifies the cost of assembling the team S_j prior to task initiation. This cost may encompass factors such as travel time and energy, as well as any time spent waiting for team members to arrive. Like the benefit function, the team formation cost function depends on the nature of the task and the assigned robots. For the simulated problems, each $J_j(S_j)$ is the cumulative distance traveled by robots to the task location:

$$J_j(S_j) = \sum_{r_i \in S_j} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{1}$$

where (x_i, y_i) is the location of robot r_i and (x_j, y_j) is the location of task t_j.

6) Utility: The net utility of completing a task t_j with team S_j is the benefit subtracted by the team formation cost:

$$U(S_j) = B_j(C_j) - J_j(S_j). \tag{2}$$

The utility of the unassigned robots is defined to be zero: $U(S_0) = 0$. Define the function $\mathcal{U}(A)$ to be be the total utility produced by an assignment A:

$$\mathcal{U}(A) = \sum_{S_j \in A} U(S_j) \tag{3}$$

B. Problem Formulation

The objective of collaborative task assignment is to maximize the total utility generated across all teams. We include a user-defined team size limit L which represents the largest reasonable number of robots for any single task.

$$\max_{A \in \mathcal{A}} \sum_{S_j \in A} U(S_j)$$
$$\text{subject to} \quad |S_j| \leq L, \quad \forall j \in \{1, 2, \ldots, \mu\} \tag{4}$$

This is a special case of the coalition structure generation problem discussed in [22], where each team $S_j \in A$ has its own utility function associated with a specific task. As discussed in [22], coalition formation is an NP-hard problem, making it computationally challenging even under restrictive assumptions.

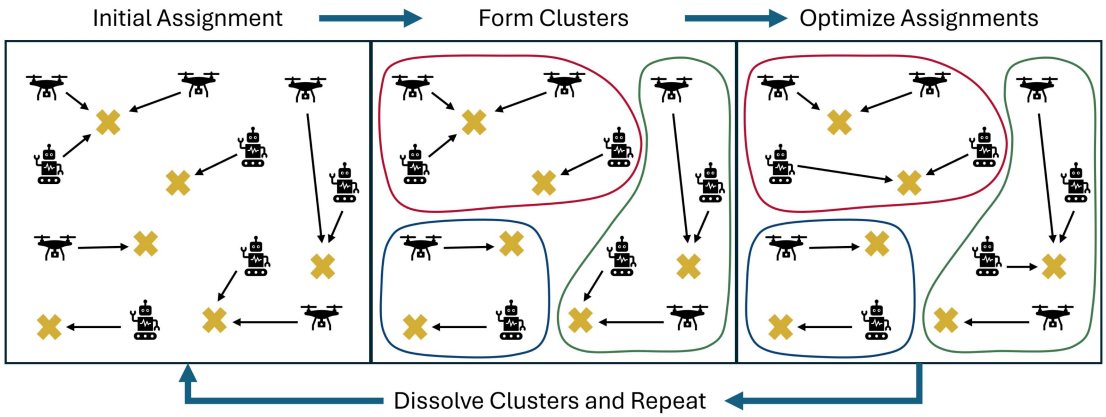

Fig. 1. Illustration of one iteration of Algorithm 1. First, robots and tasks are partitioned into clusters without separating robots from their currently assigned task. Then, an optimal assignment algorithm optimizes the assignments within each cluster. The clusters are then dissolved and the process is repeated.

The proposed iterative clustering approach provides a computationally efficient method for obtaining quick, anytime solutions to this otherwise intractable problem.

III. Methodology

The task allocation problem is addressed using an iterative approach where each iteration has two phases. In the first phase, robots and tasks are split into a set of clusters $G = \{G_1, \ldots, G_\ell\}$, where the number of clusters ℓ may vary. Each cluster $G_k = (G_k^r, G_k^t)$ is defined by a subset of robots $G_k^r \subseteq \mathcal{R}$ and a subset of tasks $G_k^t \subseteq \mathcal{T}$. Clustering in this way partitions the large, computationally intractable problem into a set of smaller, tractable subproblems. In the second phase, an optimal assignment algorithm is used to compute the optimal assignments within the clusters i.e., $A_k^* = \{S_0^k\} \cup \{S_j : t_j \in G_k^t\}$ for all $k = 1, 2, \ldots, \ell$ where $S_j \subseteq G_k^r$ is the set of robots assigned to task $t_j \in G_k^t$ and $S_0^k \subseteq G_k^r$ includes any robots not assigned to a task. Recall our problem formulation does not require all robots to be assigned to tasks. We construct the global assignment by combining cluster assignments, with all unassigned robots merged into S_0:

$$A = \overline{\bigcup_{k=1}^{\ell}} A_k^* := \left\{ S_0 = \bigcup_{k=1}^{\ell} S_0^k \} \cup \bigcup_{k=1}^{\ell} \{S_j \in A_k^* : j \neq 0 \right\}. \tag{5}$$

While a single iteration produces optimal assignments within each cluster, it does not ensure global optimality. However, by repeatedly re-clustering the problem while ensuring robots remain clustered with their currently assigned tasks, we show that the global assignment progressively improves. Algorithm 1 outlines our overall solution framework. Each choice of cluster formation algorithm and optimal assignment algorithm leads to a concrete and distinct algorithm for solving the main problem.

A. Cluster Formation

The cluster formation algorithm builds clusters of robots and tasks, limiting each cluster to user-defined maximums of L_r robots and L_t tasks while ensuring robots are clustered with their currently assigned tasks. Cluster size constraints present a trade-off: increasing L_t and L_r reduces the number of suboptimal assignments that could trap Algorithm 1, but also increases the computational complexity of the task assignment problem within each cluster.

Algorithm 1: Task Assignment Through Iterative Clustering.

Input: Robots $\mathcal{R}$, Tasks $\mathcal{T}$, Number of iterations.
Output: Assignment $A = \{S_0, S_1, \ldots, S_\mu\}$
Initialize $A = \{\mathcal{R}, \emptyset, \ldots, \emptyset\}$ with all robots unassigned;
for each iteration **do**
  $G \leftarrow$ ClusterFormation$(A, \mathcal{R}, \mathcal{T})$; *(Algorithm 2)*
  for each cluster $G_k \in G$ **do**
    $A_k^* \leftarrow$ OptimalAssignment(G_k); *(Section III-C)*
  end for
  $A \leftarrow \overline{\bigcup}_{k=1}^{\ell} A_k^*$; *(Form global assignment per Eq. 5)*
end for
Return A

Definition 1: A cluster $G_k = (G_k^r, G_k^t)$ is feasible if it satisfies the following constraints:

$$|G_k^r| \leq L_r \tag{6a}$$

$$|G_k^t| \leq L_t \tag{6b}$$

$$S_j \subseteq G_k^r \quad \forall j : t_j \in G_k^t \tag{6c}$$

where constraints 6a and 6b limit the number of robots and tasks in the cluster and 6c guarantees robots are clustered with their currently assigned task.

Any clustering algorithm can be employed that ensures all clusters are feasible. Ensuring robots remain clustered with their currently assigned tasks guarantees that the utility of the assignment is monotonically increasing with each iteration.

Lemma 1: Let A^i and A^{i+1} denote the assignments before and after the i-th iteration of Algorithm 1, respectively. If clusters are feasible and assignments within each cluster are optimal, then $\mathcal{U}(A^i) \leq \mathcal{U}(A^{i+1})$.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Algorithm 2: Clustering Algorithm.

Input: Assignment A, Robots $\mathcal{R}$, Tasks $\mathcal{T}$, Limits L_r, L_t.
Output: Clustering $G = \{G_1 \ldots G_\ell\}$
Create initial incomplete clusters G^i by grouping tasks with their assigned robots. Any unassigned robot or task will form its own cluster;
Initialize empty list of complete clusters: $G^c \leftarrow \emptyset$;
while $G^i \neq \emptyset$ **do**
  Select an incomplete cluster: $G_\alpha \sim \text{Uniform}(G^i)$;
  Find merge candidates: $M \leftarrow \{G_\gamma \in G^i : |G^r_\alpha| + |G^r_\gamma| \leq L_r, |G^t_\alpha| + |G^t_\gamma| \leq L_t, \alpha \neq \gamma\}$;
  if $M = \emptyset$ **then**
    Add G_α to complete clusters list: $G^c \leftarrow G^c \cup G_\alpha$;
    Remove from incomplete clusters list: $G^i \leftarrow G^i \setminus G_\alpha$;
  else
    Choose a cluster to merge with: $G_\beta \sim \text{MergePolicy}(M, G_\alpha)$; *(Section III-B)*
    Merge clusters: $G^i \leftarrow G^i \setminus \{G_\alpha, G_\beta\} \cup (G_\alpha \cup G_\beta)$;
  end if
end while
Return G^c

Proof: Let $G = \{G_1 \ldots G_\ell\}$ be a feasible clustering. Since robots are clustered with their currently assigned tasks, the assignment A^i can be written as the combination of the assignments within each cluster: $A^i = \bigcup_{k=1}^{\ell} A^i_k$ ((5)). The value of the assignment A^i is the sum of the utilities generated within each cluster.

$$\mathcal{U}(A^i) = \sum_{k=1}^{\ell} \mathcal{U}(A^i_k) = \sum_{k=1}^{\ell} \sum_{S_j \in A^i_k} U(S_j) \tag{7}$$

Optimal assignments A^*_k are made within each cluster G_k. Since $\mathcal{U}(A^i_k) \leq \mathcal{U}(A^*_k)$ it follows that after one iteration:
$\mathcal{U}(A^i) = \sum_{k=1}^{\ell} \mathcal{U}(A^i_k) \leq \sum_{k=1}^{\ell} \mathcal{U}(A^*_k) = \mathcal{U}(A^{i+1})$. ■

The cluster formation algorithm (Algorithm 2) begins by creating initial clusters by grouping each task with the robots assigned to it, while any unassigned task or robot starts as its own cluster. The algorithm then keeps track of two lists: incomplete clusters (G^i), which can still be enlarged, and complete clusters (G^c), which cannot grow further without violating the robot limit L_r or the task limit L_t. At each step, the algorithm randomly picks an incomplete cluster G_α and searches for other clusters that can be merged with it without exceeding the size limits. If such candidates exist, one is chosen and merged with G_α; if none exist, G_α is moved to the complete clusters list. This process repeats until every cluster is complete, meaning no further merges are possible within the given limits.

This work considers two merge policies for selecting which clusters to merge. The first is a uniform random selection of all feasible merge candidates. The second is a heuristic-based selection that forms higher-quality clusters by accounting for robot capabilities and spatial locations.

B. Merge Candidate Selection Heuristics

This section develops heuristics that guide cluster formation by aligning robot capabilities with task requirements while promoting spatial proximity.

1) Capability Matching: Robots should be clustered with tasks for which their capabilities are well suited. Therefore, clusters G_α and G_β are only merged if at least one robot $r_i \in G_\alpha$ can contribute to a task $t_j \in G_\beta$ or vice versa. A robot r_i is considered to be capable of contributing to a task t_j if there exists a team composition vector C with at least one robot of type z_i such that $B_j(C) > 0$. Define a symmetric capability matching function $m(G_\alpha, G_\beta)$:

$$m(G_\alpha, G_\beta) = \begin{cases} 1 & \text{if } \exists r_i \in G^r_\alpha, t_j \in G^t_\beta, C \in \mathbb{N}_0^\kappa : \\ & \quad B_j(C) > 0 \text{ and } C_{[z_i]} > 0 \\ 1 & \text{if } \exists r_i \in G^r_\beta, t_j \in G^t_\alpha, C \in \mathbb{N}_0^\kappa : \\ & \quad B_j(C) > 0 \text{ and } C_{[z_i]} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

2) Distance-Based Cluster Merging Probability: To minimize travel distance, we prioritize merging clusters that are closer together. Define a cluster G_k's center of mass $\text{CoM}(G_k) = (x_k, y_k)$ to be the average of all robot locations (x_i, y_i) and task locations (x_j, y_j) for all $r_i, \in G^r_k$ and $t_j \in G^t_k$.

$$\text{CoM}(G_k) = \frac{\sum_{r_i \in G^r_k} (x_i, y_i) + \sum_{t_j \in G^t_k} (x_j, y_j)}{|G^r_k| + |G^t_k|} \tag{9}$$

Distances between $\text{CoM}(G_\alpha)$ and $\text{CoM}(G_\gamma)$ for all merge candidates $G_\gamma \in M$ are converted into a probability mass function $P(G_\beta | G_\alpha)$ using a temperature-controlled softmax function [24]:

$$P_{G_\beta | G_\alpha}(G_\beta = G_\gamma | G_\alpha) = \frac{\exp(-d(G_\alpha, G_\gamma)/\tau)}{\sum_{G' \in M} \exp(-d(G_\alpha, G')/\tau)} \tag{10}$$

where $d(G_\alpha, G_\gamma)$ represents the Euclidean distance between $CoM(G_\alpha)$ and $CoM(G_\gamma)$, and τ is a tunable temperature parameter that controls the selectivity of the distribution. This probabilistic merging strategy favors combining nearby clusters while still exploring less obvious combinations.

3) Heuristic Selection Approach: The capability matching and spatial heuristics are combined into a single merge policy. First, the policy removes any merge candidates that do not satisfy the capability matching function (8):

$$M' = \{G_\gamma \in M : m(G_\alpha, G_\gamma) = 1\} \tag{11}$$

Then merge candidate $G_\beta \in M'$ is chosen according to the spatial probability distribution defined in (10).

$$G_\beta \sim P_{G_\beta | G_\alpha}(G_\beta = G_\gamma | G_\alpha) \, \forall G_\gamma \in M' \tag{12}$$

C. Optimal Assignment Within Clusters

Given a cluster G_k of n robots and m tasks, we seek an optimal assignment A^*_k that maximizes the cluster's utility by assigning the cluster's robots G^r_k to its tasks G^t_k. The optimization problem for each cluster closely mirrors the global problem in Equation

(4), but with much smaller sets of robots and tasks:

$$\max_{A_k \in \mathcal{A}_k} \sum_{S_j \in A_k} U(S_j)$$
$$\text{subject to} \quad |S_j| \leq L, \quad \forall j : j \neq 0 \tag{13}$$

where $\mathcal{A}_k$ is the set of all possible assignments within the cluster.

Any method that finds the optimal assignment within the cluster can be used. We employ a branch and bound approach using integer partitions following the examples of [25] and [26] where the search space is divided into partitions based on how many robots are assigned to each task, with upper bounds calculated for each partition. By exploring subspaces in descending order of these bounds and pruning those below our current best solution, the algorithm efficiently converges to the optimal assignment within each cluster.

IV. Analysis

This section analyzes the complexity and convergence of Algorithm 1 and how cluster size constraints determine which suboptimal assignments can trap the algorithm.

A. Computational Complexity

There are $(\mu+1)^\nu$ possible assignments of ν robots to μ tasks plus an unassigned option. For a single cluster G_k, containing at most L_r robots and L_t tasks, the worst-case branch-and-bound complexity is $(L_t+1)^{L_r}$. The number of clusters is bounded above by $\nu+\mu$, as it cannot exceed the total number of robots and tasks, and bounded below by $\max(\nu/L_r, \mu/L_t)$ due to cluster size constraints. Both bounds scale linearly with ν and μ, so the total number of clusters grows linearly with problem size. Consequently, the per-iteration time complexity of Algorithm 1 is

$$\mathcal{O}\left((L_t+1)^{L_r} \cdot (\nu+\mu)\right)$$

which is linear in ν and μ. However, Algorithm 1 may require multiple iterations for convergence as discussed in the next section.

B. Convergence

We model the progression of Algorithm 1 as a finite Markov chain $\mathcal{M} = (\mathcal{A}, P)$, where $\mathcal{A}$ is the state space comprising all possible assignments, and $P(A, A')$ gives the probability of transitioning from assignment A to A' after one iteration of Algorithm 1. Lemma 1 shows that the utility of the assignment cannot decrease so $P(A, A') = 0$ when $\mathcal{U}(A') < \mathcal{U}(A)$. However, it could change between assignments of equal utility ($\mathcal{U}(A') = \mathcal{U}(A)$).

Definition 2: A trap set $\Omega(u) \subseteq \mathcal{A}$ is a set of assignments that all produce the same utility value u and cannot transition to any assignment outside the trap set.

$$\Omega(u) = \{A \in \mathcal{A} : \mathcal{U}(A) = u \text{ and}$$
$$P(A, A') = 0 \text{ for all } A' \notin \Omega(u)\} \tag{14}$$

Note that there is always at least one nonempty trap set associated with the set of globally optimal assignments $\mathcal{A}^*$ since any globally optimal assignment $A^* \in \mathcal{A}^*$ will produce the same optimal utility u^* and by Lemma 1 cannot transition to an assignment that is suboptimal.

Definition 3: The global trap set $\Omega = \bigcup_{u \in \mathbb{R}} \Omega(u)$ includes all assignments that trap the algorithm at any utility level $u \in \mathbb{R}$.

Definition 4: An assignment A' is reachable from assignment A (or equivalently A can reach A') if there exists a finite sequence of assignments $A = A^0, A^1, \ldots A^h = A'$ such that the probability of transitioning from A^i to A^{i+1} is non-zero for all $0 \leq i < h$.

Theorem 1: Let $A^0, A^1, A^2, \ldots$ denote the sequence of assignments produced by Algorithm 1. Let A^N be the assignment obtained after N iterations of Algorithm 1. Then

$$\lim_{N \to \infty} P(A^N \in \Omega) = 1.$$

Proof: First we show that every assignment $A^i \in \mathcal{A}$ can reach an assignment $\hat{A} \in \Omega$ in the global trap set. Let $\mathcal{A}^r$ be the set of assignments reachable from A^i, and let $u_{max} = \max_{A \in \mathcal{A}^r} \mathcal{U}(A)$ be the maximum utility produced by any assignment reachable from A^i. Define $\hat{\mathcal{A}} = \{A \in \mathcal{A}^r : \mathcal{U}(A) = u_{max}\}$ to be the set of assignments reachable from A that produce utility u_{max}. Consider any assignment $\hat{A} \in \hat{\mathcal{A}}$ and $A' \in \mathcal{A}$. We show $P(\hat{A}, A') > 0 \Rightarrow A' \in \hat{\mathcal{A}}$, meaning any assignment in $\hat{\mathcal{A}}$ only transitions to other assignments in $\hat{\mathcal{A}}$.

First note that if $P(\hat{A}, A') > 0$, then by Definition 4, A' must be reachable from A^i, meaning $A' \in \mathcal{A}^r$. By definition of $u_{\max}$, $\mathcal{U}(A') \leq u_{\max}$. By Lemma 1, the utility cannot decrease, so $\mathcal{U}(A') = u_{max}$. It follows that:

$$P(\hat{A}, A') > 0 \Rightarrow A' \in A^r \text{ and } \mathcal{U}(A') = u_{max}$$
$$\Rightarrow A' \in \hat{\mathcal{A}}.$$

Therefore, $\hat{\mathcal{A}}$ satisfies Definition 2, meaning $\hat{\mathcal{A}} \subseteq \Omega(u_{max})$.

It follows that for any assignment $A^i \in \mathcal{A}$, there exists an assignment $\hat{A} \in \Omega$ in the global trap set that is reachable from A^i. By Definition 4, this means there exists a finite sequence of assignments $A^i, A^1, \ldots, A^h = \hat{A}$ where each transition $A^i \to A^{i+1}$ has non-zero probability. Since $\mathcal{A}$ is finite, $h \leq |\mathcal{A}|$ for any such sequence. This implies that starting from any assignment $A^i \in \mathcal{A}$, there is a non-zero probability of entering the global trap set Ω within $|\mathcal{A}|$ iterations. Let ϵ denote the minimum probability of entering Ω within $|\mathcal{A}|$ steps, taken over all $A^i \in \mathcal{A}$.

Now consider the algorithm after $N = k|\mathcal{A}|$ iterations for any positive integer k:

$$P(A^{k|\mathcal{A}|} \notin \Omega) \leq (1-\epsilon)^k$$
$$P(A^{k|\mathcal{A}|} \in \Omega) \geq 1 - (1-\epsilon)^k$$

$\lim_{N \to \infty} P(A^N \in \Omega) = \lim_{k \to \infty} P(A^{k|\mathcal{A}|} \in \Omega) = 1.$ ■

Because the global trap set Ω may contain local optima, the iterative clustering algorithm is not guaranteed to converge to the globally optimal solution. This is only to be expected since the underlying combinatorial optimization problem is NP hard. That said, techniques such as restarting the algorithm or introducing random perturbations could be employed to modify

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

our proposed algorithm to escape local optima. However, a detailed investigation of these strategies is left for future work.

C. Characterizing Algorithm 1 Optimal Assignments

We call any assignment $\hat{A} \in \Omega$ in the global trap set an Algorithm 1 optimal assignment, since by Definition 2, it will never transition to an assignment with greater utility.

The cluster size constraints L_r and L_t determine which assignments are Algorithm 1 optimal. In the extreme case where $L_r = \nu$ and $L_t = \mu$, a single cluster would contain all robots and tasks. When optimized, this cluster would yield a globally optimal assignment. Tightening these constraints reduces computational complexity, but introduces more Algorithm 1 optimal assignments by limiting the algorithm's ability to explore larger reassignments.

Definition 5: A cluster formation algorithm is called universally reachable if for each feasible cluster (Definition 1), the probability of forming that cluster is non-zero.

Note that the uniform random clustering method is universally reachable, while the heuristic approach in Section III-B is not, as it only merges clusters that fulfill specific capability matching criteria. Any algorithm that lacks universal reachability can be modified to achieve it by incorporating a small probability of random clustering.

Theorem 2: Let $L_n = \min(L_r/L - 1, L_t - 1)$. If Algorithm 1 employs a universally reachable cluster formation algorithm, then for any assignment $\hat{A} \in \Omega$ in the global trap set, any utility improvement requires reassigning more than L_n robots.

Proof: We show that any assignment $A = \{S_0, S_1, \ldots, S_\mu\}$ that can be improved by reassigning $x \leq L_n$ robots to reach a new assignment $A' = \{S'_0, S'_1, \ldots, S'_\mu\}$ with $\mathcal{U}(A) < \mathcal{U}(A')$ is not in the trap set. We demonstrate this by constructing a feasible cluster containing all robots and tasks necessary to perform a reassignment that increases utility.

Model the reassignment from A to A' as a graph (N, E) where nodes $N = \{S_j \in A : S_j \neq S'_j\}$ represent teams modified by the reassignment, and each edge $e \in E$ represents a transfer of robots between two teams. We show that if graph (N, E) is connected, then a feasible cluster can produce assignment A', and if the graph is disconnected, then a smaller reassignment exists that increases utility.

Case 1: (N, E) is a connected graph. Then to reassign x robots, there are at most $|E| \leq x$ edges and at most $|N| \leq x + 1$ modified teams.

We construct a cluster G_1 containing all robots and tasks required to perform the reassignment from A to A'. If S_0 (unassigned robots) is one of the modified teams ($S_0 \in N$), the cluster only needs to include the specific robots in S_0 being reassigned. However, for any assigned team $S_j \neq S_0$, constraint (6c) requires including the entire team S_j and its task t_j to ensure robots are clustered with their currently assigned task. To determine maximum cluster size, we assume the worst case scenario where all robots are assigned to tasks: $S_0 \notin N$.

Let $G_1 = (G_1^r, G_1^t)$ where $G_1^r = \bigcup_{S_j \in N} S_j$ contains all robots in the modified teams and $G_1^t = \{t_j : S_j \in N, j \neq 0\}$ contains their associated tasks. This cluster enables the reassignment from A to A'.

Since $|N| \leq x + 1$, then $|G_1^r| \leq L(x + 1)$ and $|G_1^t| \leq (x + 1)$ where L is the maximum team size. Therefore, cluster G_1 is feasible when:

$$L(x + 1) \leq L_r \Rightarrow x \leq \frac{L_r}{L} - 1 \tag{15}$$

$$x + 1 \leq L_t \Rightarrow x \leq L_t - 1 \tag{16}$$

This is equivalent to $x \leq \min(L_r/L - 1, L_t - 1) = L_n$ which is the condition given in the theorem.

Case 2: $\{N, E\}$ is not a connected graph. Then it can be decomposed into a set of at least two connected subgraphs $\{(N_1, E_1), \ldots, (N_\ell, E_\ell)\}$ such that no robots are exchanged between teams in different subgraphs.

Let $N' = \{S'_j \in A' : S_j \in N\}$ and $\mathcal{U}(N) = \sum_{S_j \in N} \mathcal{U}(S_j)$. Since $\mathcal{U}(A) < \mathcal{U}(A')$ and only teams in N are modified, it follows that:

$$\mathcal{U}(N_1) + \cdots + \mathcal{U}(N_\ell) < \mathcal{U}(N'_1) + \cdots + \mathcal{U}(N'_\ell) \tag{17}$$

Therefore, for some $N_a \in \{N_1, \ldots, N_\ell\}$, $\mathcal{U}(N_a) < \mathcal{U}(N'_a)$. It follows that the utility of assignment A can be increased by constructing a smaller cluster G_2 that includes only the teams $S_j \in N_a$. Since (N_a, E_a) is a connected graph with fewer than x edges, it follows from Case 1 that G_2 is feasible.

In both cases, we have demonstrated the existence of a feasible cluster that, when optimized, will transition the assignment A to one with higher utility. Given Definition 5, with each iteration, there is a nonzero probability of forming this cluster and making this transition. Therefore, A is not in the global trap set. ■

V. Experimental Results

The iterative clustering algorithm is evaluated within a simulated 100×100 unit workspace, where ν robots and μ tasks are independently initialized at uniformly random positions. Robots are evenly partitioned into $\kappa = 3$ types. Tasks are evenly divided into 10 task types, each associated with a benefit function $B_j(C)$ (Section II-A4) that maps the team composition C to the utility for task t_j. The utility for each task depends on both the number and type of assigned robots and can exhibit linear returns, marginally decreasing returns, or marginally increasing returns (synergistic collaboration) as more robots are assigned. Each task's team size is capped at $L = 3$ robots, and the team-formation cost $J_j(S_j)$ is the cumulative distance traveled by assigned robots to the task location (Eq. 1). Unless otherwise noted, the iterative clustering procedure uses cluster-size limits $L_r = 6$ (robots) and $L_t = 3$ (tasks). Complete documentation, source code, and scripts to replicate all experiments and figures are available at https://github.com/AICPS/IterativeClusteringMRTA.

A. Effect of Cluster Size

The random clustering approach was used to evaluate the effects of different cluster sizes. As shown in Fig. 2, while smaller clusters require more iterations, they converge faster in time due to their lower computational cost per iteration. Cluster

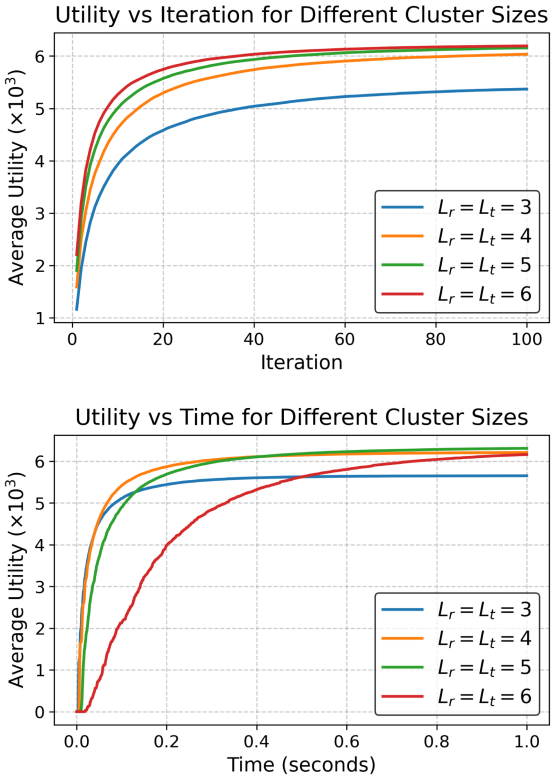

Fig. 2. Utility produced by Algorithm 1 when employing the random clustering strategy, tested with various cluster size limits L_r and L_t. Results are averaged over 100 tests with $\nu = 100$ robots, $\mu = 50$ tasks, and $\kappa = 3$ robot types.

TABLE I
CLUSTERING METHODS COMPARED WITH OPTIMAL

Method	Exact Optimal Matches (%)	Average Utility Ratio (%)
Random	99.6	99.9
Heuristic	99.0	99.9

size limits L_r and L_t present a clear trade-off: larger clusters reduce convergence to suboptimal utilities and require fewer iterations to converge, but increase per-cluster computational complexity.

B. Comparison to Optimal for Small Problems

For small-scale problems, the clustering methods can be directly compared against optimal solutions. Table I presents the performance comparison on 500 problem instances with $\nu = 10$ robots and $\mu = 5$ tasks. Both random and heuristic clustering methods achieve optimal or near-optimal utility, with no statistically significant difference observed between their results.

C. Task Allocation Methods Comparison

For a larger problem with $\nu = 50$ robots and $\mu = 25$ tasks, heuristic and random iterative clustering methods were compared against a group-based auction, hedonic game, and simulated annealing. Overall, heuristic clustering outperformed the other methods, closely followed by random clustering. Fig. 3

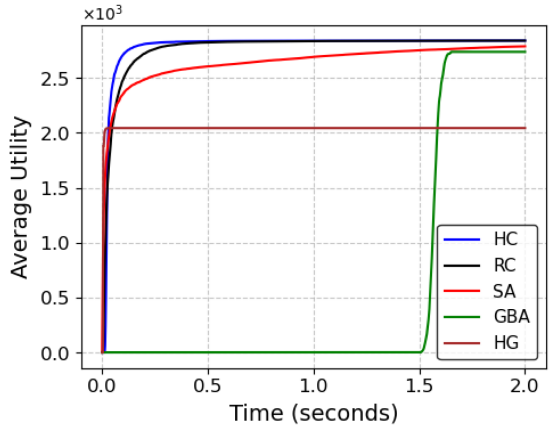

Fig. 3. Average algorithm performance over 100 tests for Heuristic Clustering (HC), Random Clustering (RC), Simulated Annealing (SA), Group-Based Auction (GBA), and Hedonic Game (HG) approaches.

TABLE II
ALGORITHM PERFORMANCE: AVERAGE UTILITY, STANDARD DEVIATION, AND MAXIMUM UTILITY AS PERCENTAGES OF THE BEST SOLUTION FOUND

Method	0.1 seconds		0.5 seconds		2 seconds		
	Avg	σ	Avg	σ	Avg	σ	Max
HC	**95.2**	2.6	**99.4**	0.5	**99.6**	0.4	100.0
RC	87.4	3.4	99.1	0.6	**99.6**	0.4	100.0
SA	82.7	2.7	91.4	1.5	97.8	0.9	99.5
GBA	0.0	0.0	0.0	0.0	96.0	0.0	96.0
HG	71.7	2.8	71.7	2.8	71.7	2.8	78.1

shows the averaged results, and Table II reports mean utilities and standard deviations at 0.1, 0.5, and 2 seconds.

The group-based auction algorithm presented in [11] was adapted to assign teams of robots to tasks. In this version, task representatives evaluate all feasible teams and place bids based on each team's average reward per robot. While it achieves similar utility to iterative clustering, it is far less computationally efficient since all feasible teams must be evaluated before the auction begins, whereas clustering considers only candidate teams within each cluster.

Hedonic games are a common heuristic for task allocation [12], [27]. We compare against a hedonic game where each robot iteratively joins the team that maximizes their marginal contribution, converging to a Nash equilibrium. Although computationally efficient, this method struggles on collaborative tasks requiring simultaneous multi-robot assignments, where individual reassignments provide no benefit.

The simulated annealing algorithm perturbs solutions by randomly replacing a single team while preserving other task assignments. Improvements are always accepted and worse solutions are accepted with decreasing probability over time. Simulated annealing converged more slowly than iterative clustering due to its randomized search approach, whereas iterative clustering employs a more directed search through optimal cluster assignments.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

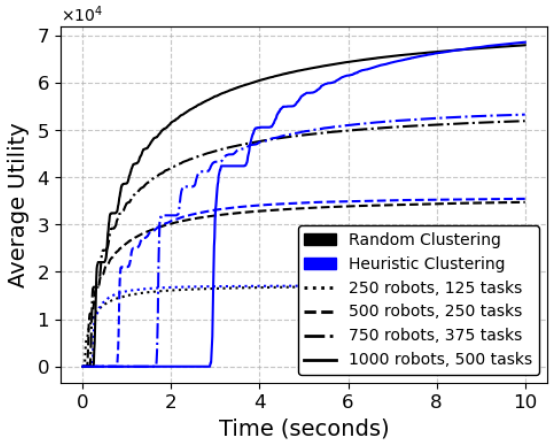

Fig. 4. Average algorithm performance on four different problem sizes.

D. Scalability

Algorithm 1's scalability was tested on four problem sizes, with results in Figure 4 averaged over 100 trials. Random clustering scaled efficiently, while heuristic clustering degraded due to the cost of computing merge probabilities across many clusters. For the largest case (1000 robots, 500 tasks), the heuristic required about 3 seconds for the first iteration. Overall, the framework scales well, though larger problems demand alternative heuristics.

VI. Conclusion

This letter presented an iterative clustering algorithm for solving collaborative task allocation in multi-robot systems. The approach iteratively forms small clusters of robots and tasks, optimizing assignments within each cluster while ensuring robots remain clustered with their currently assigned tasks. This guarantees the utility is monotonically increasing with each iteration. Larger clusters avoid becoming trapped at suboptimal assignments, while smaller clusters offer faster convergence time. Experimental results validated the scalability, accuracy, and scalability of our approach compared to state of the art heuristics.

Future work will explore learning-based approaches to cluster generation, decentralized implementations of iterative clustering, and how clustering techniques can adapt to dynamic and uncertain environments.

Acknowledgment

Opinions, findings and conclusions, or recommendations are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

References

[1] P. Ghassemi and S. Chowdhury, "Multi-robot task allocation in disaster response: Addressing dynamic tasks with deadlines and robots with range and payload constraints," *Robot. Auton. Syst.*, vol. 147, 2022, Art. no. 103905.

[2] Y. Hu, Y. Lu, R. Xu, W. Xie, S. Chen, and Y. Wang, "Collaboration helps camera overtake LiDAR in 3D detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 9243–9252.

[3] W. Dai, A. Bidwai, and G. Sartoretti, "Dynamic coalition formation and routing for multirobot task allocation via reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 16567–16 573.

[4] Y. Liu, F. Liu, L. Tang, C. Bai, and L. Liu, "Multirobot adaptive task allocation of intelligent warehouse based on evolutionary strategy," *J. Sensors*, vol. 2022, no. 1, 2022, Art. no. 2056617.

[5] X. Shan, Y. Jin, M. Jurt, and P. Li, "A distributed multi-robot task allocation method for time-constrained dynamic collective transport," *Robot. Auton. Syst.*, vol. 178, 2024, Art. no. 104722.

[6] K. Athira and U. Subramaniam, "A systematic literature review on multi-robot task allocation," *ACM Comput. Surv.*, vol. 57, no. 3, pp. 1–28, 2024.

[7] H. Chakraa, F. Guérin, E. Leclercq, and D. Lefebvre, "Optimization techniques for multi-robot task allocation problems: Review on the state-of-the-art," *Robot. Auton. Syst.*, vol. 168, 2023, Art. no. 104492.

[8] A. A. Nguyen, F. Jabbari, and M. Egerstedt, "Mutualistic interactions in heterogeneous multi-agent systems," in *Proc. 62nd IEEE Conf. Decis. Control*, 2023, pp. 411–418.

[9] S. Chopra, G. Notarstefano, M. Rice, and M. Egerstedt, "A distributed version of the Hungarian method for multirobot assignment," *IEEE Trans. Robot.*, vol. 33, no. 4, pp. 932–947, Aug. 2017.

[10] G. Notomista et al., "A resilient and energy-aware task allocation framework for heterogeneous multirobot systems," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 159–179, Feb. 2022.

[11] X. Bai, A. Fielbaum, M. Kronmüller, L. Knoedler, and J. Alonso-Mora, "Group-based distributed auction algorithms for multi-robot task assignment," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 2, pp. 1292–1303, Apr. 2023.

[12] L. Zhang, D. Liang, M. Li, W. Yang, and S. Yang, "Coalition formation game approach for task allocation in heterogeneous multi-robot systems under resource constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 3439–3446.

[13] Z. Zhang et al., "Distributed dynamic task allocation for unmanned aerial vehicle swarm systems: A networked evolutionary game-theoretic approach," *Chin. J. Aeronaut.*, vol. 37, no. 6, pp. 182–204, 2024.

[14] Z. Junwei and Z. Jianjun, "Study on multi-UAV task clustering and task planning in cooperative reconnaissance," in *Proc. 6th Int. Conf. Intell. Hum.- Mach. Syst. Cybern.*, 2014, pp. 392–395.

[15] I. Younas, F. Kamrani, M. Bashir, and J. Schubert, "Efficient genetic algorithms for optimal assignment of tasks to teams of agents," *Neurocomputing*, vol. 314, pp. 409–428, 2018.

[16] H. Guo, Z. Miao, J. Ji, and Q. Pan, "An effective collaboration evolutionary algorithm for multi-robot task allocation and scheduling in a smart farm," *Knowl.-Based Syst.*, vol. 289, 2024, Art. no. 111474.

[17] J. You, J. Jia, X. Pang, J. Wen, Y. Shi, and J. Zeng, "A novel multi-robot task assignment scheme based on a multi-angle k-means clustering algorithm and a two-stage load-balancing strategy," *Electronics*, vol. 12, no. 18, 2023, Art. no. 3842.

[18] J. G. Martin, F. J. Muros, J. M. Maestre, and E. F. Camacho, "Multi-robot task allocation clustering based on game theory," *Robot. Auton. Syst.*, vol. 161, 2023, Art. no. 104314.

[19] M. Goarin and G. Loianno, "Graph neural network for decentralized multi-robot goal assignment," *IEEE Robot. Autom. Lett.*, vol. 9, no. 5, pp. 4051–4058, May 2024.

[20] R. Karam, R. Lin, B. A. Butler, and M. Egerstedt, "Resource allocation with multi-team collaboration based on Hamilton's rule," 2025, *arXiv:2505.09016*.

[21] O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," *Artif. Intell.*, vol. 101, no. 1-2, pp. 165–200, 1998.

[22] T. Rahwan, T. P. Michalak, M. Wooldridge, and N. R. Jennings, "Coalition structure generation: A survey," *Artif. Intell.*, vol. 229, pp. 139–174, 2015.

[23] B. Gerkey and M. J. Mataric, "A formal framework for the study of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, 2004.

[24] R. S. Sutton et al., *Reinforcement Learning: An Introduction*, vol. 1. Cambridge, MA, USA: MIT Press, 1998.

[25] F. Präntare, and F. Heintz, "An anytime algorithm for optimal simultaneous coalition structure generation and assignment," *Auton. Agents Multi-Agent Syst*, vol. 34, no. 1, pp. 1–31, 2020

[26] T. Rahwan, S. D. Ramchurn, N. R. Jennings, and A. Giovannucci, "An anytime algorithm for optimal coalition structure generation," *J. Artif. Intell. Res.*, vol. 34, pp. 521–567, 2009.

[27] I. Jang, H.-S. Shin, and A. Tsourdos, "Anonymous hedonic game for task allocation in a large-scale multiple agent system," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1534–1548, Dec. 2018.