

Learning Altruistic Collaboration in Heterogeneous Multi-Team Systems

Riwa Karam¹, Ruoyu Lin¹, Brooks A. Butler¹, Magnus Egerstedt²

Abstract—This paper studies heterogeneous multi-team collaboration through dynamic robot allocation, where robots are treated as transferable resources. Leveraging Hamilton’s rule from ecology as an altruistic decision making mechanism, we propose a multi-team collaborative resource allocation framework with heterogeneous capabilities, transfer costs, and capability-dependent contributions. The resulting allocation problem is combinatorial and is shown to be NP-hard. To address scalability, we develop a graph neural network policy under centralized training and decentralized execution that approximates the altruistic allocations based on Hamilton’s rule. The model operates over the team interaction graph and predicts robot-level transfer decisions and next robot-to-team assignments. The proposed approach is validated in a fire-fighting scenario through simulations and experiments, demonstrating that the learned policy achieves near-optimal performance while scaling to larger systems.

Index Terms—Heterogeneous collaboration, graph neural networks, altruism, NP-hard problems.

I. INTRODUCTION

Multi-robot systems hold out the promise of providing a scalable and robust approach for complex tasks such as disaster response [1], distributed sensing [2], and environmental monitoring [3]. A fundamental challenge in these systems is *resource and task allocation*, where different resources and/or tasks must be assigned to different agents to optimize overall system performance [4]–[8]. In many applications, these decisions must be made dynamically and in a decentralized manner while accounting for limited resources, spatial constraints, and time-varying environments [9]–[11].

Typical multi-robot allocation formulations assume independent or additive contributions of agents, where the performance of a team depends primarily on the number of assigned robots [7], [12]. However, heterogeneity inherently increases the complexity of each agent’s contributions to its team, especially when mission success relies on collaborative interactions [13]. In heterogeneous systems, robots may possess complementary sensing, actuation, or computational capabilities, and the contribution of a robot depends on the composition of the team it joins. As a result, team performance may become set-dependent and generally non-separable, introducing combinatorial coupling across agents.

Recent work has explored learning-based approaches for combinatorial optimization in multi-agent systems [14], [15], with comprehensive overviews highlighting the potential of

machine learning for solving large-scale combinatorial problems [16]. In particular, graph neural networks (GNNs) provide a framework for learning decentralized policies over relational structures, enabling scalable decision-making in multi-robot systems [17]–[20]. These methods have demonstrated efficiency with applications in routing [21], scheduling [22], and multi-agent control [23], focusing on coordinated interactions. In this article, we are interested in how heterogeneous robots with complementary capabilities contribute to the team performance through joint effort.

Biological systems provide inspirations for interaction in multi-agent settings [24], with evolutionary perspectives offering fundamental explanations for cooperation and altruistic behavior [25]. In particular, *Hamilton’s rule* [26] characterizes altruistic behavior through a trade-off between cost and benefit weighted by genetic relatedness. This principle has been adopted in multi-robot systems to regulate inter-team resource exchanges via a local decision rule in [7]. In that framework, robots are treated as transferable resources, and exchanged based on Hamilton’s rule, which yields globally optimal allocations under homogeneous assumptions and diminishing returns. Similarly, it has been used to model inter-robot altruistic interactions and decision-making towards global objective contributions in [27].

In this paper, we extend the altruistic collaboration framework from [7] to heterogeneous multi-team systems. Heterogeneity is modeled through capability vectors and set-dependent mission evaluation functions, and introduces explicit transfer costs associated with robot migration. The resulting allocation problem is NP-hard [28], [29], making exact solutions intractable at scale. We consider an event-triggered hybrid system with continuous-time mission dynamics and discrete-time robot exchanges, and aim to maximize a global mission evaluation function in a decentralized setting. To enable scalable decision-making, the problem is formulated as a structured prediction task over graphs, and a GNN policy is developed under centralized training and decentralized execution (CTDE) that operates over the team interaction graph and predicts robot-level transfer decisions, enabling scalable approximation of the underlying combinatorial optimization problem.

The main contributions of this paper are:

- 1) An altruistic multi-team collaborative allocation framework with heterogeneous robots, capability-dependent mission evaluation functions and transfer costs;
- 2) NP-hardness proof of the resulting heterogeneous multi-robot allocation problem via reduction from the Partition problem [28], motivating the learning-based framework;

¹R. Karam, R. Lin, and B. A. Butler are with the Samueli School of Engineering, University of California, Irvine, Irvine, CA, 92697, USA. Email: {rwkaram, rlin10, bbutler2}@uci.edu.

²M. Egerstedt is with the University of North Carolina at Chapel Hill, Chapel Hill, NC, 27599, USA. Email: magnus@unc.edu.

- 3) A learning-based approach, formulated as a GNN under CTDE, that allows for a real-time solution to the otherwise intractable optimization problem; and
- 4) Instantiation of the framework in a fire-fighting scenario, capturing heterogeneous sensing and actuation capabilities of different robots, and corresponding experiments in simulations and on a hardware testbed, demonstrating the scalability and near-optimal performance of the framework in this application.

The remainder of this paper is organized as follows. Section II reviews the homogeneous altruistic collaboration framework and presents the heterogeneous extension of it. Section III analyzes and compares the complexity of the homogeneous and heterogeneous resource allocation problems. Section IV presents the learning-based approach. Section V instantiates the theoretical altruistic collaboration model to a fire-fighting application. Section VI provides experimental results, and Section VII concludes the paper.

II. PROBLEM FORMULATION

In this section, the multi-team robot allocation problem underlying altruistic collaboration is formalized. We begin by reviewing the homogeneous framework introduced in [7], where robots are treated as identical resources and Hamilton's rule provides a local mechanism for regulating inter-team transfers. This framework is then extended to the heterogeneous setting, where robots possess distinct capabilities and team performance depends on the composition of assigned robots rather than their count alone.

A. Homogeneous Altruistic Framework

This article builds upon the framework introduced in [7], where Hamilton's rule from ecology is reinterpreted as a team-level decision criterion to regulate inter-team robot transfers. In that formulation, Hamilton's rule serves as a local decision-maker that determines admissible exchanges, enabling distributed decision-making while implicitly improving a global objective. In this section, the homogeneous framework is summarized and its associated optimization problem is formulated.

In the homogeneous setting of [7], multiple teams of identical robots operate in distinct regions, and robots are treated as transferable resources whose allocation determines team performance. Interactions between teams are represented by an undirected graph $G = (V, E)$, where V is the set of teams with cardinality $M = |V|$, and $E \subseteq V \times V$ encodes the pairwise team connections. The objective is to reallocate robots across teams to maximize a global mission evaluation function.

Hamilton's rule, originally expressed as $rB \geq C$ in evolutionary biology [26], is adapted to define a decision-making rule for robot transfers between teams. The genetic relatedness term is modeled through a ratio of mission-importance weights, chosen to reflect differences in team priority, as

$$r_{ij} = \frac{w_j}{w_i}, \quad (1)$$

where $w_i, w_j > 0$ denote the relative importance assigned to teams i and j , respectively. A transfer of a robot from team

i to team j is admissible if $r_{ij}B_j > C_i$, where B_j is the marginal benefit of gaining a robot for team j , and C_i is the marginal cost incurred by team i . This condition ensures that the weighted gain of the receiving team exceeds the loss of the donor team.

Each team $v \in V$ is associated with a mission evaluation function $F_v(n_v) : \mathbb{N} \rightarrow \mathbb{R}$, where n_v denotes the number of robots assigned to team v . The functions F_v are assumed to satisfy the following assumptions:

- *Strict monotonicity*: $F_v(n_v + 1) > F_v(n_v)$,
- *Diminishing returns*:

$$F_v(n_v + 1) - F_v(n_v) \leq F_v(n_v) - F_v(n_v - 1).$$

These assumptions imply that the marginal contribution of an additional robot decreases as team size increases, a property closely related to submodularity in set functions [30]. The marginal benefit and cost are defined as

$$\begin{aligned} B_j(n_j) &:= F_j(n_j + 1) - F_j(n_j), \\ C_i(n_i) &:= F_i(n_i) - F_i(n_i - 1). \end{aligned} \quad (2)$$

A key structural property established in [7] is the *pairwise uni-directionality* of admissible transfers. Under the above assumptions, the conditions

$$r_{ij}B_j(n_j) > C_i(n_i) \quad \text{and} \quad r_{ji}B_i(n_i) > C_j(n_j) \quad (3)$$

cannot hold simultaneously. This property prevents cyclic exchanges between pairs of teams and ensures that transfers have a consistent direction.

When multiple teams are present, a team may have several admissible incoming and outgoing transfers. To resolve such conflicts, a local bidding mechanism is introduced in [7]. The pairwise uni-directionality of admissible transfers according to Hamilton's rule is referred to as *Hamilton-admissible transfer* in this paper. For each Hamilton-admissible transfer $i \rightarrow j$, team i computes the net gain $\Delta_{i \rightarrow j} = r_{ij}B_j(n_j) - C_i(n_i)$. Each team selects at most one outgoing and one incoming transfer based on the highest positive gain. These local decisions collectively aim to maximize the global objective

$$\mathcal{G}(n_1, \dots, n_M) := \sum_{v \in V} w_v F_v(n_v). \quad (4)$$

The resulting process alternates between determining Hamilton-admissible transfers, selecting transfers through local biddings, and executing only those that strictly improve the global objective. Under the assumptions of homogeneity, diminishing returns, and a fixed total number of robots, it is shown in [7] that through an iterative negotiation and reallocation algorithm:

- the global objective $\mathcal{G}$ strictly increases, with diminishing returns, with each executed transfer,
- the process terminates in a finite number of steps, and
- the final allocation is a global maximizer of $\mathcal{G}$.

For the rest of this section, we formulate the problem introduced in [7] as an optimization problem. Under homogeneity, the allocation is fully described by the team sizes as

$$\mathbf{n} := (n_1, \dots, n_M) \in (\mathbb{Z}^+)^M,$$

with $\sum_{v=1}^M n_v = N$, where N is the total number of robots. To express the problem at the robot level, let $\mathcal{R} = \{1, \dots, N\}$ and the assignment variable be

$$x_{r,v} = \begin{cases} 1, & \text{if robot } r \in \mathcal{R} \text{ is assigned to team } v \in V, \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

Since robots are homogeneous in [7], the mission evaluation function of team v depends only on its team size, given by $n_v = \sum_{r \in \mathcal{R}} x_{r,v}$, $\forall v \in V$. The global objective can then also be written as

$$\mathcal{G}(X) = \sum_{v \in V} w_v F_v \left(\sum_{r \in \mathcal{R}} x_{r,v} \right), \quad (6)$$

where $X = [x_{r,v}]_{r \in \mathcal{R}, v \in V} \in \{0, 1\}^{N \times M}$ denotes the assignment matrix. The generalized optimization problem whose feasible solutions correspond to the robot-to-teams assignment that is being solved in [7] can then be written as the following optimization problem where the global mission evaluation function from Equation (6) is maximized over the assignment matrix X :

$$\begin{aligned} \max_X \quad & \mathcal{G}(X) \\ \text{s.t.} \quad & \sum_{v \in V} x_{r,v} = 1, \quad \forall r \in \mathcal{R}, \\ & \sum_{r \in \mathcal{R}} x_{r,v} \geq 1, \quad \forall v \in V, \\ & x_{r,v} \in \{0, 1\}, \quad \forall r \in \mathcal{R}, \forall v \in V. \end{aligned} \quad (7)$$

The constraints in Equation (7) represent that each robot is assigned to one team only and that each team has to have at least one robot at all times, respectively. This formulation represents the global assignment problem that the distributed process in [7] seeks to solve through local decisions based on Hamilton's rule. The iterative approach to altruistic resource allocation is presented in [7].

B. Heterogeneous Altruistic Collaboration

To extend the framework in [7], a heterogeneous multi-team setting is considered in which robots possess different capabilities. Unlike the homogeneous case, where team performance depends only on the number of assigned robots, the contribution of each robot now depends on its capabilities and the composition of the team it joins. As a result, the mission evaluation functions become set-dependent, fundamentally altering the structure of the assignment problem. Robot heterogeneity is encoded through capability vectors and introduce transfer costs associated with robot migration between teams.

Let each robot r be characterized by a capability vector $\kappa_r \in \{0, 1\}^Q$, where each entry indicates the presence (1) or absence (0) of a specific capability; Q denotes the number of capability types. For each team $v \in V$, let the set of robots assigned to team v at iteration k be defined as

$$\mathcal{S}_v^k := \{r \in \mathcal{R} \mid x_{r,v}^k = 1\}, \quad \mathcal{S}_v^k \subset \mathcal{R}.$$

An example of the heterogeneous multi-team system is illustrated in Figure 1, where teams, heterogeneous robots, and

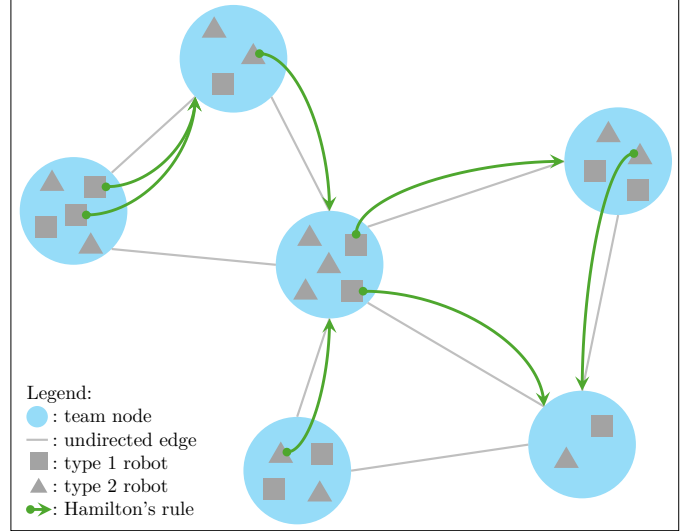


Fig. 1. Example of a heterogeneous multi-team system where teams, depicted as blue nodes, are connected through an interaction graph with undirected edges and composed of robots with different capabilities. Robots of different types (type 1 and type 2, shown as squares and triangles, respectively) correspond to distinct capability classes as defined by their capability vectors. Directed edges (green arrows) represent candidate inter-team robot transfers that satisfy Hamilton-admissibility.

Hamilton-admissible transfer directions are shown. Note that Hamilton's rule in the figure are robot-specific; by contrast, in the homogeneous setting in Section II-A, Hamilton's rule would point from one team to the other, with no regard to which robot is chosen.

Since the assignment matrix X^k satisfies the constraint in Equation (5), for each robot $r \in \mathcal{R}$ there exists a unique team $v \in V$ such that $x_{r,v}^k = 1$. The current team of robot r (i.e., at iteration k) is defined as

$$\text{cur}(r) = v \quad \text{such that} \quad x_{r,v}^k = 1,$$

and its chosen next team (i.e., at iteration $k+1$) is defined as

$$\text{next}(r) = v \quad \text{such that} \quad x_{r,v}^{k+1} = 1.$$

Let the binary transfer variables $y_{r,i \rightarrow j}^{k+1}$, indicating whether robot r moves from team i to team j , be

$$y_{r,i \rightarrow j}^{k+1} = \begin{cases} 1, & \text{if robot } r \in \mathcal{R} \text{ moves from team } i \\ & \text{to team } j \text{ at iteration } k+1, \\ 0, & \text{otherwise,} \end{cases} \quad (8)$$

for all $r \in \mathcal{R}$ and $(i, j) \in E$. Note that $\text{next}(r) \neq \text{cur}(r)$ happens when $y_{r, \text{cur}(r) \rightarrow \text{next}(r)}^{k+1} = 1$, otherwise, $\text{next}(r) = \text{cur}(r)$, meaning robot r was not transferred at iteration k . In particular, we get $r \in \mathcal{S}_{\text{cur}(r)}^k$ and $r \in \mathcal{S}_{\text{next}(r)}^{k+1}$. Let $Y^{k+1} = \{y_{r,i \rightarrow j}^{k+1}\}$ then denote the set of binary transfer decision variables.

The mission evaluation function of each team will depend not only on the number of robots in team v as defined in [7] and explained in Section II-A, but also on the set of robots forming team v , $\mathcal{S}_v^k$, since robots are heterogeneous and have

a certain set of capabilities. Hence, for each team $v \in V$, the mission evaluation function is defined as

$$\mathcal{F}_v : \{\mathcal{S} \mid \mathcal{S} \subseteq \mathcal{R}\} \rightarrow \mathbb{R}, \quad (9)$$

evaluated at the assigned robot set $\mathcal{S}_v^k$. In particular, $n_v^k = |\mathcal{S}_v^k|$, where $|\mathcal{S}_v^k|$ denotes the cardinality of $\mathcal{S}_v^k$. This formulation captures the dependence of team performance on the joint capabilities of its members, introducing collaboration as defined in [13], thereby leading to combinatorial interactions between robots.

Since a transfer cost is introduced, we define the distance needed for robot r to travel from its current team $\text{cur}(r)$ to any team $v \in V$ to be $d_{r, \text{cur}(r) \rightarrow v} \in \mathbb{R}^+$. Hence, the transfer cost becomes

$$\mathcal{C}(X^k, X^{k+1}) := \sum_{r \in \mathcal{R}} \sum_{v \in V} x_{r,v}^{k+1} c_{r,v}(x_{r,v}^k), \quad (10)$$

where $c_{r,v} : \{0, 1\} \rightarrow \mathbb{R}^+$ can depend on the distance $d_{r, \text{cur}(r) \rightarrow v}$ and the speed of robot r denoted as $s_r \in \mathbb{R}^+$.

It is worth mentioning that both the mission evaluation function and the transfer cost, as defined in Equations (9) and (10) can be then formulated in a mission-specific manner, where multiple formulation choices may be modeled even for the same mission. However, the definitions and notations in this section, summarized in Table I, present a general formulation of what each represents.

Since the goal of the multi-team system is to optimally and successfully achieve the assigned missions, we wish to optimize the system efficiency in a way that the global mission evaluation function is maximized and the global transfer cost is minimized. The following optimization problem reflects those needs:

$$\begin{aligned} \max_X \quad & \mathcal{G}(X) - \lambda \mathcal{C}(X^0, X) \\ \text{s.t.} \quad & \sum_{v \in V} x_{r,v} = 1, \quad \forall r \in \mathcal{R}, \\ & \sum_{r \in \mathcal{R}} x_{r,v} \geq 1, \quad \forall v \in V, \\ & x_{r,v} \in \{0, 1\}, \quad \forall r \in \mathcal{R}, \forall v \in V, \end{aligned} \quad (11)$$

with $\mathcal{G}(X^k) := \sum_{v \in V} w_v \mathcal{F}_v(\mathcal{S}_v^k)$ being the heterogeneous global mission evaluation function, the regularization parameter $\lambda > 0$ representing the balance of the trade-off between minimizing the costs and maximizing mission performance, and X^0 being a random initial assignment matrix. Note that, similar to Equation (7), the use of k is omitted since the optimization problem in Equation (11) presents the general assignment problem to be solved in the heterogeneous formulation, with same constraints but different objective functions than Equation (7).

Next, the iterative, altruistic, and heterogeneous optimization problem that we wish to solve in this article is presented. Let the Hamilton-admissible indicator be

$$\begin{aligned} h_{r,i \rightarrow j}^{k+1} &:= \mathbb{I} \{ r_{ij} B_{r,j}(\mathcal{S}_j^k) > C_{r,i}(\mathcal{S}_i^k) \}, \\ h_{r,i \rightarrow j}^{k+1} &\in \{0, 1\}, \quad \forall (i, j) \in E, \end{aligned} \quad (12)$$

with

$$\begin{aligned} B_{r,j}(\mathcal{S}_j^k) &:= \mathcal{F}_j(\mathcal{S}_j^k \cup \{r\}) - \mathcal{F}_j(\mathcal{S}_j^k), \\ C_{r,i}(\mathcal{S}_i^k) &:= \mathcal{F}_i(\mathcal{S}_i^k) - \mathcal{F}_i(\mathcal{S}_i^k \setminus \{r\}). \end{aligned} \quad (13)$$

TABLE I
PROBLEM FORMULATION NOTATION

Symbol	Description
V	Set of teams
E	Set of team edges
$G = (V, E)$	Team interaction graph
$\mathcal{R}$	Set of robots
κ_r	Capability vector of robot r
s_r	Speed of robot r
$\mathcal{S}_v^k$	Set of robots assigned to team v at iteration k
$\mathcal{F}_v(\mathcal{S}_v^k)$	Mission evaluation function
X^k	Assignment matrix at iteration k
$\mathcal{G}(X^k)$	Global mission evaluation function
$y_{r,i \rightarrow j}^{k+1}$	Transfer decision variable
$h_{r,i \rightarrow j}^{k+1}$	Hamilton-admissible indicator
$\mathcal{C}(X^k, X^{k+1})$	Global transfer cost
λ	Regularization parameter

Note that, in the homogeneous setting, the benefit and cost factors consider the addition or subtraction of one robot (Equation (2)), whereas in Equation (13), they become set-dependent. The same relatedness factor is considered as in Equation (1). Hence, we get the following optimization problem that seeks to solve Equation (7) iteratively:

$$\begin{aligned} \max_{X^{k+1}, Y^{k+1}} \quad & \mathcal{G}(X^{k+1}) - \lambda \mathcal{C}(X^k, X^{k+1}) \\ \text{s.t.} \quad & \sum_{v \in V} x_{r,v}^{k+1} = 1, \quad \forall r \in \mathcal{R}, \\ & \sum_{r \in \mathcal{R}} x_{r,v}^{k+1} \geq 1, \quad \forall v \in V, \\ & x_{r,v}^{k+1} \in \{0, 1\}, \quad \forall r \in \mathcal{R}, \forall v \in V, \\ & x_{r,j}^{k+1} = x_{r,j}^k - \sum_{l: (j,l) \in E} y_{r,j \rightarrow l}^{k+1} + \sum_{i: (i,j) \in E} y_{r,i \rightarrow j}^{k+1}, \\ & \quad \forall r \in \mathcal{R}, \forall j \in V, \\ & y_{r,i \rightarrow j}^{k+1} \in \{0, 1\}, \quad \forall r \in \mathcal{R}, \forall (i, j) \in E, \\ & y_{r,i \rightarrow j}^{k+1} \leq x_{r,i}^k, \quad \forall r \in \mathcal{R}, \forall (i, j) \in E, \\ & y_{r,i \rightarrow j}^{k+1} \leq h_{r,i \rightarrow j}^{k+1}, \quad \forall r \in \mathcal{R}, \forall (i, j) \in E. \end{aligned} \quad (14)$$

Unlike the homogeneous case, teams are not restricted to a single incoming or outgoing transfer per iteration. Instead, multiple robots may be transferred simultaneously, reflecting the heterogeneous contributions of individual robots. Given the heterogeneity of robots and the set-dependent evolution of the team missions, the uni-directionality property in Equation (3) no longer applies. Now, the Hamilton-admissible indicator (12) is dependent on the robot itself given the heterogeneity and individual contribution of each. Equation (14) contains the constraints present in Equations (11) and (7), with the addition of constraints reflecting those additions. Those additional constraints ensure that each robot, if transferred, moves from its current team to exactly one destination team at each iteration, while enforcing Hamilton's rule that restricts transfers to those

with positive weighted net benefit.

In summary, the problem that we seek to solve in this article is stated in Problem 1.

Problem 1. *Given a multi-team robot system with an interaction graph G and a set of heterogeneous robots $\mathcal{R}$, where each robot is characterized by a capability vector κ_r and each team v 's performance is quantified by a set-dependent mission evaluation function $\mathcal{F}_v$, the objective is to determine a sequence of local robot-to-team assignment decisions that maximizes the global mission performance $\mathcal{G}$ while minimizing the global transfer cost $\mathcal{C}$. These decisions are made in a decentralized manner through local interactions, leading to an optimal assignment X .*

As detailed in this section, we approach this problem through an altruistic framework, where local robot transfer decisions are regulated using Hamilton's rule to balance individual team costs with global performance improvements.

III. COMPLEXITY ANALYSIS

In this section, we analyze and compare the computational complexity of the optimization problems in Equation (7), the iterative process in [7], Equation (11), and Equation (14).

A. Homogeneous Resource Allocation

In the homogeneous case, the feasible set is the discrete simplex

$$\left\{ \mathbf{n} \in (\mathbb{Z}^+)^M \mid \sum_{v=1}^M n_v = N \right\},$$

whose cardinality is $\binom{N-1}{M-1}$. A brute-force solution to Equation (7) enumerates all feasible allocations and evaluates the global objective $\mathcal{G}(\mathbf{n}) = \sum_{v \in V} w_v F_v(n_v)$ for each. Assuming evaluating $\mathcal{G}$ takes $O(M)$ time, the brute-force complexity is

$$O\left(M \binom{N-1}{M-1}\right),$$

which grows combinatorially in N and M .

The altruistic iterative process in [7] restricts explicit allocation by using Hamilton's rule and a bidding mechanism over $G = (V, E)$. Per iteration, it performs:

- Hamilton's-rule checks and bid computations on each edge: $O(|E|)$;
- selecting at most one outgoing and one incoming collaboration per team (implemented by scanning neighbors): $O(|E|)$ total;
- updating team sizes and marginal values: $O(M)$.

Hence the per-iteration complexity is $O(|E| + M) = O(|E|)$. Moreover, under the assumptions in [7], the global objective $\mathcal{G}$ strictly increases whenever a transfer is executed, so the same allocation $\mathbf{n}$ cannot be revisited. Because there are only $\binom{N-1}{M-1}$ feasible allocations, the number of executed transfers is at most $\binom{N-1}{M-1} - 1$. Therefore, a loose worst-case runtime bound is

$$O\left(|E| \binom{N-1}{M-1}\right).$$

In practice, far fewer transfers occur since the algorithm follows a monotone path to an optimum rather than exploring the full feasible set; the average runtime scales approximately as $O(|V||E|)$.

B. Heterogeneous Resource Allocation

In the heterogeneous formulation (Equation (11)), the decision variable is the binary assignment matrix $X \in \{0, 1\}^{N \times M}$. The number of feasible assignments is at most M^N (each robot chooses one of M teams), so brute force requires $O(M^N)$ objective evaluations, which is exponential in N .

Proposition 1. *The heterogeneous allocation problem in Equation (11) is NP-hard.*

Proof. We reduce from the *Partition* problem [28], which is NP-complete. Given a set of positive integers $\{a_1, \dots, a_N\}$ with total sum $A = \sum_{r=1}^N a_r$, the Partition problem asks whether there exists a subset $\mathcal{I} \subseteq \mathcal{R}$ such that $\sum_{r \in \mathcal{I}} a_r = \sum_{r \notin \mathcal{I}} a_r = A/2$. A special instance of the heterogeneous allocation problem is constructed as follows:

- There are $M = 2$ teams, denoted as 1 and 2.
- Each robot $r \in \mathcal{R}$ has a scalar capability equal to a_r .
- The mission evaluation functions are defined as

$$\mathcal{F}_v(\mathcal{S}_v) = - \left| \sum_{r \in \mathcal{S}_v} a_r - \frac{A}{2} \right|, \quad v \in \{1, 2\}.$$

- The team weights are equal: $w_1 = w_2 = 1$.
- The transfer cost is set to zero: $\lambda = 0$.
- All transfers are admissible, i.e., $h_{r,i \rightarrow j}^{k+1} = 1$ for all feasible (r, i, j) .

Under these reductions, the objective in Equation (11) reduces to $\mathcal{G}(X) = \mathcal{F}_1(\mathcal{S}_1) + \mathcal{F}_2(\mathcal{S}_2)$. Since $\mathcal{S}_1$ and $\mathcal{S}_2$ form a partition of $\mathcal{R}$, we have $\sum_{r \in \mathcal{S}_1} a_r + \sum_{r \in \mathcal{S}_2} a_r = A$. Let $S = \sum_{r \in \mathcal{S}_1} a_r$. Then $\sum_{r \in \mathcal{S}_2} a_r = A - S$, and the objective can be written as $\mathcal{G}(X) = -|S - \frac{A}{2}| - |(A - S) - \frac{A}{2}|$. Noting that $|(A - S) - \frac{A}{2}| = |\frac{A}{2} - S| = |S - \frac{A}{2}|$, we get $\mathcal{G}(X) = -2|S - \frac{A}{2}|$. Thus, maximizing $\mathcal{G}(X)$ is equivalent to minimizing $|S - \frac{A}{2}|$. The maximum value of $\mathcal{G}(X)$ is achieved if and only if $S = A/2$, which corresponds exactly to a valid partition. Therefore, solving the heterogeneous allocation problem allows us to determine whether a solution to the Partition problem exists. Since Partition is NP-complete, the heterogeneous allocation problem is NP-hard. Finally, Equation (11) is at least as general as this restricted instance, as it includes additional constraints such as transfer structure and costs. Hence, NP-hardness carries over to the full problem. $\square$

Note that the hardness result does not rely on monotonicity or diminishing returns (assumptions made in the homogeneous formulation), and therefore applies to the general heterogeneous formulation where mission evaluation functions are arbitrary set-dependent functions.

Corollary 1. *The heterogeneous, iterative, and altruistic allocation problem in Equation (14) is NP-hard.*

Proof. The iterative formulation in Equation (14) restricts moves to edge-admissible transfers via variables $\{y_{r,i \rightarrow j}^{k+1}\}$,

as per Equation (8). A naive evaluation of all candidate robot-edge transfers involves up to $N|E|$ binary variables, and computing all Hamilton-admissible indicators $h_{r,i \rightarrow j}^{k+1}$ from Equation (12) may require evaluating marginal changes in $\mathcal{F}_i$ and $\mathcal{F}_j$ for each candidate transfer. Consequently, without additional structure in $\mathcal{F}_v(\cdot)$, the per-iteration cost can scale on the order of $O(N|E|)$ evaluations in the worst case.

Furthermore, Proposition 1 establishes that the general heterogeneous allocation problem is NP-hard. The iterative formulation corresponds to solving this problem over a restricted set of feasible transitions between successive allocations. Since the centralized problem can be recovered as a special case of the iterative formulation (e.g., by allowing all transfers and considering a single-step transition), the iterative problem is at least as hard. Hence, Equation (14) remains NP-hard in general. $\square$

In summary, while the homogeneous problem already exhibits combinatorial growth, its separable structure allows the algorithm in [7] to reliably find a global optimum for small scale problems. The heterogeneous assignment problem, however, strictly generalizes this setting, has a feasible set that grows exponentially with N (up to M^N), and is NP-hard as per Proposition 1, motivating the learning-based approach developed in the next section.

IV. LEARNING ALTRUISTIC COLLABORATION

Given the NP-hardness of the heterogeneous allocation problem in Equations (11) and (14), we seek a scalable approximation that can be executed in real time for solving Problem 1. To this end, the problem is formulated as a structured prediction task over a graph and learn a policy using a GNN under CTDE. The learned policy maps the current multi-team state to robot-level transfer decisions that approximate the solution of the one-step optimization problem. In this section, we go over the process of generating synthetic data as well as presenting the GNN architecture to be used to train a model predicting the output of Equation (14).

A. Synthetic Data Generation

To train the GNN policy, we require labeled data corresponding to one-step collaboration decisions under the heterogeneous altruistic optimization problem in Equation (14). Since exact optimization is intractable at large scale, *small-scale* synthetic problem instances are generated for which the exact one-step solution can be computed via exhaustive search. These exact solutions serve as supervisory labels during training.

Each generated instance defines a supervised learning sample of the form (Γ, Y) , where Γ encodes the current system state as a graph and Y specifies the optimal next assignment for all robots. The learning objective is to approximate the mapping from the current state to the optimal reassignment under the heterogeneous altruistic objective. Each synthetic instance is defined by a team interaction graph, team-specific mission evaluation functions, a heterogeneous robot population, and an initial assignment. The resulting state is encoded

as a graph, and the corresponding label is the optimal next-team decision for each robot.

Given M , an undirected and connected graph $G = (V, E)$ is sampled. Connectivity is ensured by first constructing a spanning tree, after which additional edges are inserted independently with a certain determined probability. Each team $v \in V$ is assigned a position in a shared workspace, with a minimum separation constraint to ensure spatial distinction. A region $\mathcal{D}_v$, representing the region of interest, is assigned to each team to represent its operational area. Pairwise distances between team positions are precomputed and stored in a lookup table, which is later used to evaluate inter-team travel distances, i.e., $d_{r, \text{cur}(r) \rightarrow v}$.

Next, a heterogeneous population of N robots is sampled. Each robot is assigned its capability vector with a certain probability distribution of the different capabilities. An initial assignment X^0 is then sampled subject to the feasibility constraints.

Given the current state, we compute the Hamilton-admissible mask $h_{r,i \rightarrow j}^{k+1}$, which defines the feasible destination set for each robot. We then enumerate all feasible

Algorithm 1 Synthetic One-Step Data Generation

- 1: **Input:** number of samples N_{data} , team and robot ranges, model parameter λ
 - 2: **Output:** supervised dataset D
 - 3: Initialize $D \leftarrow \emptyset$
 - 4: **for** $n = 1, \dots, N_{\text{data}}$ **do**
 - 5: Sample an instance size (M, N)
 - 6: Sample a connected team interaction graph $G = (V, E)$
 - 7: Sample team positions and corresponding team regions $\{\mathcal{D}_v\}_{v \in V}$
 - 8: Compute the pairwise team-distance lookup table
 - 9: Sample team weights $\{w_v\}_{v \in V}$
 - 10: **for** each team $v \in V$ **do**
 - 11: Generate and sample mission-specific team parameters
 - 12: **end for**
 - 13: Sample N heterogeneous robots with corresponding sampled capability vectors
 - 14: **for** each robot $r \in \mathcal{R}$ **do**
 - 15: Generate and sample mission-specific robot parameters
 - 16: **end for**
 - 17: Sample an initial assignment X^k
 - 18: Compute the Hamilton-admissible mask $h_{r,i \rightarrow j}^{k+1}$
 - 19: Enumerate all feasible next assignments X^{k+1} consistent with $h_{r,i \rightarrow j}^{k+1}$
 - 20: Evaluate each feasible assignment using

$$\mathcal{G}(X^{k+1}) - \lambda \mathcal{C}(X^k, X^{k+1})$$
 - 21: Let X^* be the maximizing assignment
 - 22: Construct graph features from X^k and label the sample with X^*
 - 23: Append the resulting graph-label pair to D
 - 24: **end for**
 - 25: **return** D
-

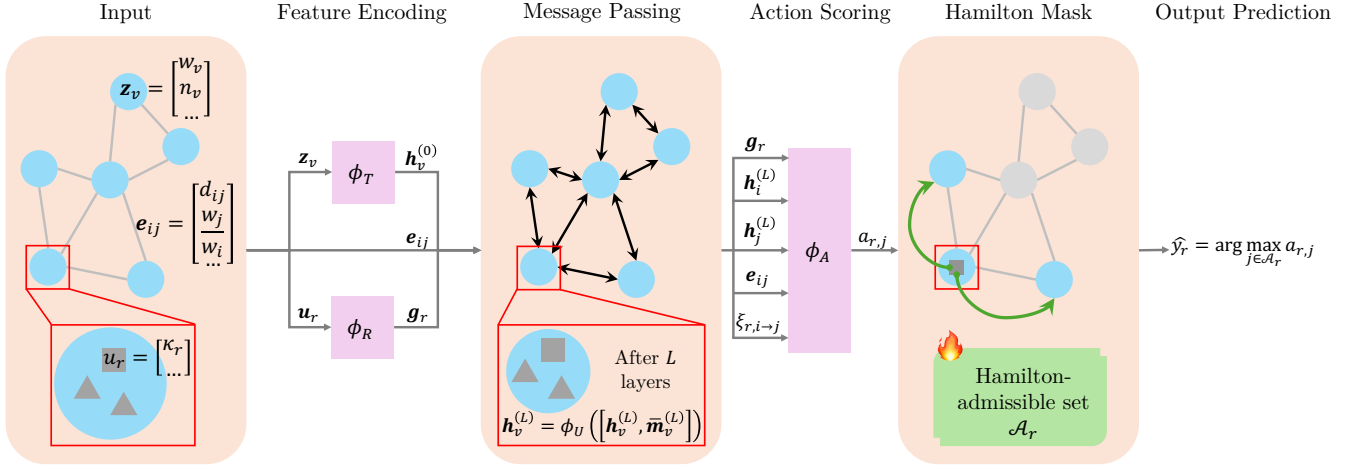


Fig. 2. Overview of the proposed GNN-based policy for heterogeneous altruistic collaboration with an example graph input $G = (V, E)$. The current system state is encoded with node features $\mathbf{z}_v$, edge features $\mathbf{e}_{ij}$, and robot features $\mathbf{u}_r$. Feature encoders map these inputs to latent embeddings, after which message passing over the team interaction graph produces context-aware team representations. For each robot r , candidate transfers $(r, i \rightarrow j)$ are evaluated using an action-scoring function conditioned on the robot embedding, source and destination team embeddings, edge features, and a robot-dependent transfer descriptor $\xi_{r,i \rightarrow j}$. A Hamilton-admissible mask restricts the action set to feasible transfers, and the final decision is obtained via a masked maximization over the admissible set. The MLPs are represented with the pink squares and rectangle, and the fire emoji highlights the altruistic decision-maker.

next assignments that are consistent with this mask and the feasibility constraints. Among these, the assignment X^* that maximizes $\mathcal{G}(X^{k+1}) - \lambda \mathcal{C}(X^k, X^{k+1})$ is selected, as defined in Equation (14). The supervisory label is represented as a categorical assignment for each robot, indicating its destination team at the next iteration.

Each instance is then encoded as a graph for training. Team features include the team weight, number of robots, number of robots under each capability type, mission-specific parameters, and mission value. Robot features include capability indicators and mission-specific parameters. Edge features include relative team position, pairwise distance, the mission-weight ratio w_j/w_i , and the adjacency indicator. In addition, each robot is associated with a Hamilton mask, a candidate destination mask, and a travel-distance matrix. The target label specifies the optimal destination team for each robot.

Algorithm 1 summarizes the overall data-generation procedure. In general, N_{data} is generated with such graph-label pairs, which are used to train the GNN policy presented in the next section. This dataset enables the model to learn a mapping from local graph-structured information to globally consistent transfer decisions, thereby approximating the solution of the NP-hard allocation problem of Equations (11) and (14) in a manner suitable for CTDE.

B. GNN Architecture

In this section, we introduce a policy that approximates the one-step optimal reassignment while remaining computationally efficient at execution time. The problem is formulated as a masked multi-class prediction problem over a graph and train a GNN to map the current heterogeneous multi-team state to robot-level transfer decisions leading to the next robot-to-team assignment.

At iteration k , the system state is represented by a graph

$$\mathcal{H}^k = (G, \mathcal{Z}^k, \mathcal{U}^k, \mathcal{E}^k),$$

where G is the team interaction graph, $\mathcal{Z}^k = \{\mathbf{z}_v^k\}_{v \in V}$ denotes team features, $\mathcal{U}^k = \{\mathbf{u}_r^k\}_{r \in \mathcal{R}}$ denotes robot features, and $\mathcal{E}^k = \{\mathbf{e}_{ij}^k\}_{(i,j) \in V \times V}$ denotes directed edge features. The team features $\mathbf{z}_v^k$ summarize the current state of team v , including its weight w_v , current team size n_v^k , mission-specific parameters, and any additional quantities required to evaluate $\mathcal{F}_v(\mathcal{S}_v^k)$. The robot features $\mathbf{u}_r^k$ encode the capability vector κ_r , robot-specific parameters, and any state variables relevant to transfer decisions. The edge features $\mathbf{e}_{ij}^k$ encode pairwise information between teams, such as adjacency, relative Euclidean distance, pairwise transfer cost descriptors, and the weight ratio w_j/w_i .

For each robot r , the action space is restricted by the Hamilton-admissible indicator. Specifically, given the current team $\text{cur}(r)$, the feasible destination set is

$$\mathcal{A}_r^k = \{\text{cur}(r)\} \cup \left\{ j \in V \mid h_{r, \text{cur}(r) \rightarrow j}^{k+1} = 1 \right\},$$

which includes the option of staying in the current team together with all Hamilton-admissible destination teams. The learning problem is therefore to predict, for each robot r , its next team $\text{next}(r) \in \mathcal{A}_r^k$.

The proposed GNN consists of three stages:

- 1) feature encoding,
- 2) team-level message passing, and
- 3) robot-level action scoring.

First, team and robot features are embedded into a common latent space using learnable encoders,

$$\mathbf{h}_v^{(0)} = \phi_T(\mathbf{z}_v^k), \quad \mathbf{g}_r = \phi_R(\mathbf{u}_r^k),$$

where ϕ_T and ϕ_R are multilayer perceptrons (MLPs). Next, information is propagated over the team interaction graph through L rounds of message passing. At layer ℓ , the message from team i to team j is

$$\mathbf{m}_{i \rightarrow j}^{(\ell)} = \phi_m \left(\left[\mathbf{h}_i^{(\ell)}, \mathbf{h}_j^{(\ell)}, \mathbf{e}_{ij}^k \right] \right),$$

where ϕ_m is a learnable message function. Messages are then aggregated at each team j over its neighbors $\mathcal{N}_j$,

$$\bar{\mathbf{m}}_j^{(\ell)} = \frac{1}{|\mathcal{N}_j|} \sum_{i \in \mathcal{N}_j} \mathbf{m}_{i \rightarrow j}^{(\ell)},$$

and the team embedding is updated according to

$$\mathbf{h}_j^{(\ell+1)} = \phi_U \left(\left[\mathbf{h}_j^{(\ell)}, \bar{\mathbf{m}}_j^{(\ell)} \right] \right),$$

where ϕ_U is a learnable update map. After L layers, each team embedding $\mathbf{h}_v^{(L)}$ summarizes both local team information and contextual information from neighboring teams.

To evaluate candidate transfers, the policy scores each robot-destination pair (r, j) with $j \in \mathcal{A}_r^k$. Let $i = \text{cur}(r)$; the action score is computed as follows

$$a_{r,j} = \phi_A \left(\left[\mathbf{g}_r, \mathbf{h}_i^{(L)}, \mathbf{h}_j^{(L)}, \mathbf{e}_{ij}^k, \xi_{r,i \rightarrow j}^k \right] \right),$$

where ϕ_A is an action-scoring MLP and $\xi_{r,i \rightarrow j}^k$ denotes any robot-dependent transfer descriptor, such as a cost or distance feature.

The predicted next team of robot r is then given by

$$\hat{y}_r^k = \arg \max_{j \in \mathcal{A}_r^k} a_{r,j}.$$

Equivalently, infeasible actions are masked before the maximization, ensuring that the policy only outputs either Hamilton-admissible decisions or the “no-move” decision. Training is performed using the exact labels produced by Algorithm 1 in Section IV-A. The overall architecture of the proposed GNN-based policy is illustrated in Figure 2, which summarizes the encoding of the system state, message passing over the team interaction graph, and the robot-level action scoring with Hamilton-admissible masking. If y_r^* denotes the optimal destination team of robot r in the exact one-step solution X^* , then the policy parameters are learned by minimizing a masked cross-entropy loss,

$$\mathcal{L} = -\frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \log \frac{\exp(a_{r,y_r^*})}{\sum_{j \in \mathcal{A}_r^k} \exp(a_{r,j})}. \quad (15)$$

This loss penalizes the model only over feasible actions, thereby aligning the learning objective with the constrained decision structure of Equation (14).

The resulting policy follows a CTDE paradigm, where the model is trained offline using globally generated supervisory labels, while at execution time each team exchanges messages only with its one-hop neighbors, as shown in the next section, and computes robot transfer scores using robot features, source-team and destination-team embeddings, and local edge descriptors. Thus, the learned policy provides a fast approximation of the one-step optimizer without solving the full combinatorial optimization problem online, which will then be able to solve Problem 1. Note that the abstract feature definitions above are intentionally mission-agnostic; in any specific application, they are instantiated by choosing team, robot, and edge descriptors consistent with the mission evaluation function $\mathcal{F}_v(\mathcal{S}_v)$ and transfer cost $\mathcal{C}(\cdot)$.

C. GNN Inference

Once trained, the GNN is used at inference time as a decentralized decision policy inside an iterative collaboration loop. The goal is to produce a feasible sequence of robot transfers without exhaustively solving Equation (14) online. At each decision step, the current system state is encoded as a graph, processed by the learned policy, and then converted into executable transfers through a local feasibility mask.

Given the current assignment X^k , the graph representation $\mathcal{H}^k$ described in Section IV-B is first constructed. The GNN then performs one round of message passing over the team interaction graph, so that each team exchanges latent information only with its one-hop neighbors. For each robot $r \in \mathcal{R}$, the policy computes action scores $a_{r,j}$ over the feasible destination set $\mathcal{A}_r^k$ and predicts the destination $\hat{y}_r^k$. The corresponding score $a_{r,\hat{y}_r^k}$ is used to rank the proposed transfer.

Because robot decisions are predicted in parallel, the joint proposal may contain simultaneous transfers that cannot all be executed together while preserving local team feasibility. We therefore interpret the GNN output as a set of candidate transfers and apply a local selection rule. Specifically, for each team $i \in V$, all proposed outgoing transfers are collected

$$\mathcal{P}_i^k = \left\{ (r, i \rightarrow \hat{y}_r^k, a_{r,\hat{y}_r^k}) \mid \text{cur}(r) = i, \hat{y}_r^k \neq i \right\},$$

sorted in descending order of score, and processed sequentially. A proposed transfer is accepted if executing it preserves the hard local constraints of the source team. Otherwise, it is discarded. In the general formulation, these feasibility checks

Algorithm 2 GNN-Guided Inference

- 1: **Input:** current assignment X^k , trained GNN policy, maximum number of iterations
 - 2: **Output:** updated assignment X^{k+1}
 - 3: Initialize $X^{k+1} \leftarrow X^k$
 - 4: **for** each step **do**
 - 5: Construct graph state $\mathcal{H}$ from X^{k+1}
 - 6: Compute Hamilton-feasible destination sets $\{\mathcal{A}_r\}_{r \in \mathcal{R}}$
 - 7: Run the GNN with one-hop message passing on $\mathcal{H}$
 - 8: **for** each robot $r \in \mathcal{R}$ **do**
 - 9: Predict $\hat{y}_r = \arg \max_{j \in \mathcal{A}_r} a_{r,j}$
 - 10: **end for**
 - 11: **for** each team $i \in V$ **do**
 - 12: Construct the proposed outgoing transfer set $\mathcal{P}_i^k$
 - 13: Sort $\mathcal{P}_i^k$ by decreasing score
 - 14: **for** each transfer in $\mathcal{P}_i^k$ **do**
 - 15: **if** executing the transfer preserves local feasibility **then**
 - 16: Accept the transfer and update X
 - 17: **end if**
 - 18: **end for**
 - 19: **end for**
 - 20: **if** no transfer was accepted **then**
 - 21: **break**
 - 22: **end if**
 - 23: **end for**
 - 24: **return** X^{k+1}
-

include the assignment constraint and the requirement that each team remain nonempty; additional mission-specific local constraints may also be enforced.

After all teams have processed their proposed outgoing transfers, the accepted moves define the next assignment X^{k+1} . The procedure is then repeated using the updated graph state. The iterative altruistic collaboration terminates when no team accepts any additional transfer, indicating that the learned policy does not propose any further locally feasible reallocation under the current Hamilton-admissible structure. Algorithm 2 summarizes the inference-time procedure.

V. FIRE-FIGHTING SCENARIO

In order to instantiate Problem 1 and train a GNN model, we need to apply the collaboration framework from Section II-B into a mission-specific setting. We choose to apply it towards fire-fighting missions in different geographical locations where team-to-team migration of robots can still happen. In this section, we define how the different components of the framework will be tailored towards this application.

For this application, two types of robots are considered: *sensing* robots, and *fire-fighting* robots (i.e., water-carrying robots); hence, $Q = 2$. Therefore, we get the following capability vector:

$$\kappa_r = [S_r \quad W_r] \in \{0, 1\}^2, \quad (16)$$

where

$$S_r = \begin{cases} 1, & \text{if robot } r \in \mathcal{R} \text{ is a sensing robot} \\ 0, & \text{otherwise} \end{cases}$$

and

$$W_r = \begin{cases} 1, & \text{if robot } r \in \mathcal{R} \text{ is a fire-fighting robot} \\ 0, & \text{otherwise.} \end{cases}$$

Each robot $r \in \mathcal{R}$ is assigned exactly one of the two capabilities. Team heterogeneity can then be shown through the number of robots under each capability given $N_{W,v}^k = \sum_{r \in \mathcal{S}_v^k} W_r$ and $N_{S,v}^k = \sum_{r \in \mathcal{S}_v^k} S_r$.

In the general iterative heterogeneous collaboration model (Equation (14)), we only enforce that each team maintains at least one robot. For this application, we require that this robot be a *sensing* robot; i.e., each team must retain at least one sensing robot at every discrete decision iteration. This application-specific feasibility constraint is enforced in the allocation decision variables (and is not assumed in the mission-agnostic formulation). Thus, the constraint $\sum_{r \in \mathcal{R}} S_r x_{r,v} \geq 1, \forall v \in V$ is added to the optimization problem.

Each robot type contributes differently to the success of the team mission. We define the role of the sensing robots through optimal area coverage, for which a common approach is the Voronoi-based coverage control framework introduced in [2]. Specifically, we consider the deployment of $N_{S,v}^k \in \mathbb{Z}^+$ sensing robots in team v over a domain of interest $\mathcal{D}_v \subset \mathbb{R}^d$, associated with a density function $\phi_v(q) : \mathcal{D}_v \rightarrow \mathbb{R}^+$ that captures the relative importance of points in $\mathcal{D}_v$. Assuming that coverage effectiveness decreases as a robot becomes farther

from a point $q \in \mathcal{D}_v$, the locational cost function for team $v \in V$ can be written as [2]

$$L_v(p) = \sum_{i \in \mathcal{N}} \int_{\mathcal{V}_i} \|p_i - q\|^2 \phi_v(q) dq, \quad (17)$$

where $p_i \in \mathbb{R}^d$ denotes the position of robot i , and the vector $p = [p_1^\top, p_2^\top, \dots, p_{N_{S,v}^k}^\top]^\top$ contains the positions of all sensing robots in team v . The index set $\mathcal{N} = \{1, \dots, N_{S,v}^k\}$ corresponds to the sensing robots after local relabeling within the team, and

$$\mathcal{V}_i = \{q \in \mathcal{D} \mid \|q - p_i\| \leq \|q - p_j\|, \forall j \neq i \in \mathcal{N}\} \quad (18)$$

defines the Voronoi cell associated with robot $i \in \mathcal{N}$. From Equation (18), it follows that $\cup_{i \in \mathcal{N}} \mathcal{V}_i = \mathcal{D}$, while the Lebesgue measure satisfies $\lambda(\cap_{i \in \mathcal{N}} \mathcal{V}_i) = 0$. A standard approach for obtaining an optimal coverage configuration, e.g., [2], is to let each robot i move according to the negative gradient flow of $L_v(p)$ with respect to $p_i, \forall i \in \mathcal{N}$. This corresponds to a continuous-time version of Lloyd's algorithm and asymptotically converges to a centroidal Voronoi tessellation (CVT), given by

$$p_i = c_i = \frac{\int_{\mathcal{V}_i} q \phi_v(q) dq}{\int_{\mathcal{V}_i} \phi_v(q) dq}, \quad \forall i \in \mathcal{N},$$

where $c_i \in \mathbb{R}^d$ denotes the centroid of the Voronoi cell of robot i , and the resulting CVT configuration is represented as $c = [c_1^\top, c_2^\top, \dots, c_{N_{S,v}^k}^\top]^\top$. Finally, we explicitly include the number of sensing robots $N_{S,v}^k$ in team v as an argument of the locational cost function in Equation (17), yielding

$$L_v(N_{S,v}^k, p) = \sum_{i=1}^{N_{S,v}^k} \int_{\mathcal{V}_i} \|p_i - q\|^2 \phi_v^k(q) dq. \quad (19)$$

In addition, the fire density, encoded by $\phi_v(q)$, is considered to be piecewise constant; so, we also introduce and include $\phi_v^k(q)$, being the fire density at iteration k of team v , to be an argument of the locational cost function (19), i.e.,

$$L_v(N_{S,v}^k, p, \phi_v^k(q)) = \sum_{i=1}^{N_{S,v}^k} \int_{\mathcal{V}_i} \|p_i - q\|^2 \phi_v^k(q) dq. \quad (20)$$

However, since L_v is in function of $N_{S,v}^k$, it is modified to be

$$L_v = \begin{cases} 0, & \text{if } N_{S,v}^k = 0, \\ \sum_{i=1}^{N_{S,v}^k} \int_{\mathcal{V}_i} \|p_i - q\|^2 \phi_v^k(q) dq, & \text{otherwise,} \end{cases} \quad (21)$$

since the locational cost in Equation (20) is not defined at $N_{S,v}^k = 0$.

We wish to encode the sensing contributions to be in $[0, 1]$; hence, we are motivated to wrap the locational cost in Equation (21) in a sigmoid function. The general form of the sigmoid function is defined as

$$\sigma(x) = \frac{1}{1 + e^{-a(x-b)}} \quad (22)$$

TABLE II
FIRE-FIGHTING MODEL COMPONENTS

Component	Description
S_r	Sensing capability indicator of robot r
W_r	Fire-fighting capability indicator of robot r
$N_{S,v}^k$	Number of sensing robots in team v at iteration k
$N_{W,v}^k$	Number of fire-fighting robots in team v at iteration k
$\phi_v^k(q)$	Fire density at location q for team v at iteration k
$\mathcal{D}_v$	Spatial domain associated with team v
L_v	Locational cost for sensing robots (coverage quality)
$\psi_v(N_{S,v}^k)$	Sensing effectiveness (sigmoid of inverse locational cost)
K_r	Fire-fighting capacity of robot r
$P_v(N_{W,v}^k)$	Aggregate fire-fighting power of team v
$\tau_{r,\text{cur}(r) \rightarrow v}$	Travel time from current team to team v
α	Cost scaling parameter
η	Fire decay rate parameter
Δt	Discrete time step

where b is the value of x at which $\sigma(x) = 0.5$. Finally, we say that the contributions of the sensing robots for each team v is given by

$$\psi_v(N_{S,v}^k) = \begin{cases} 0, & \text{if } N_{S,v}^k = 0, \\ \frac{1}{1+e^{-a(1/L_v - b)}}, & \text{otherwise} \end{cases} \quad (23)$$

with $a = 1$ and $b = 0$. We take x from (22) to be $x = 1/L_v(N_{S,v}^k, p, \phi_v^k(q))$ in (23); this choice maps lower locational cost (better coverage) to higher sensing effectiveness, and empirically exhibits diminishing returns as $N_{S,v}^k$ increases.

Having defined the sensing robots' contributions to the team's mission, we can now also present the choice of defining the fire-fighting robots' contributions. Let K_r be robot r 's fire-fighting capacity, i.e., encoding the water-carrying and water-dumping capacities of robot r . The contribution of the fire-fighting robots will be as follows

$$P_v(N_{W,v}^k) = \sum_{r \in S_v^k: W_r = 1} K_r. \quad (24)$$

Since the fire density is piecewise constant as mentioned previously, we define the mission evaluation function for each team $v \in V$ as:

$$\mathcal{F}_v(\mathcal{S}_v^{k+1}) = - \int_{\mathcal{D}_v} \phi_v^{k+1}(q) dq, \quad (25)$$

with

$$\phi_v^{k+1}(q) = \phi_v^k(q) e^{-\frac{1}{\eta} P_v^{k+1}(N_{W,v}^{k+1}) \psi_v^{k+1}(N_{S,v}^{k+1}) \Delta t}, \quad (26)$$

where $\eta \in \mathbb{R}^+$ represents the speed of the exponential decay and $\Delta t > 0$ is the time step between iterations. This formulation captures collaborative capability complementarity (i.e., [13]) as sensing robots improve situational awareness through coverage, while fire-fighting robots reduce fire intensity. Neither capability alone is sufficient to suppress fires, and effective performance emerges from their joint contribution.

Now, the transfer cost is defined to be as follows:

$$\mathcal{C}(X^k, X^{k+1}) = \sum_{r \in \mathcal{R}} \sum_{v \in V} \alpha x_{r,v}^{k+1} \tau_{r,\text{cur}(r) \rightarrow v}, \quad (27)$$

while instantiating the generic cost function in Equation (10), for this application, as

$$c_{r,v}(x_{r,v}^k) = \alpha \tau_{r,\text{cur}(r) \rightarrow v},$$

and with

$$\tau_{r,\text{cur}(r) \rightarrow v} = \frac{d_{r,\text{cur}(r) \rightarrow v}}{s_r} \quad (28)$$

representing the time needed for robot r being considered for transfer, to travel the distance between its current team and team v , given its speed s_r , and α being a regularization constant with $\alpha > 0$.

Table II summarizes the key components of the fire-fighting application of the theoretical heterogeneous altruistic collaborative model presented in Section II-B.

VI. RESULTS

In this section, the synthetic data generation process in Algorithm 1 is instantiated for the fire-fighting application introduced in Section V, and the performance of the GNN model described in Section IV-B is trained using the generated data and then evaluated. The supervised learning framework is first assessed in terms of its ability to approximate the optimal one-step allocation decisions under this setting and then the learned policy is evaluated in larger-scale simulations and on a physical robot testbed.

A. Machine Learning Framework

The supervised learning framework used to train the proposed GNN policy is evaluated on a dataset of 18,000 synthetically generated heterogeneous collaboration instances. Each instance corresponds to a graph-structured multi-team system with 3 to 7 teams and a robot population that scales proportionally with the number of teams. Ground-truth labels are obtained using the exact one-step optimizer described in Algorithm 1, providing optimal robot-to-team assignment decisions under the heterogeneous altruistic objective.

In the generated dataset, the robot-level decision labels are imbalanced, as across the full dataset, 192,628 decisions correspond to staying with the current team, while 41,381 correspond to moving to another team. Thus, only 17.68% of robot-level decisions are transfer actions. This imbalance makes exact target destination of the "move" prediction substantially harder than the "stay" classification. However, given the nature of the problem being approximated, the fraction of "stay" actions is in general going to be larger than the one for the "move" actions.

The learning problem is formulated as a masked multi-class classification task. For each robot, the model predicts its next team from a feasible set defined by the Hamilton-admissible mask, including the current team ("stay") and admissible transfer destinations. This set also includes all admissible transfer destinations ("move" action(s)). Since the number of teams varies across instances, the number of candidate classes

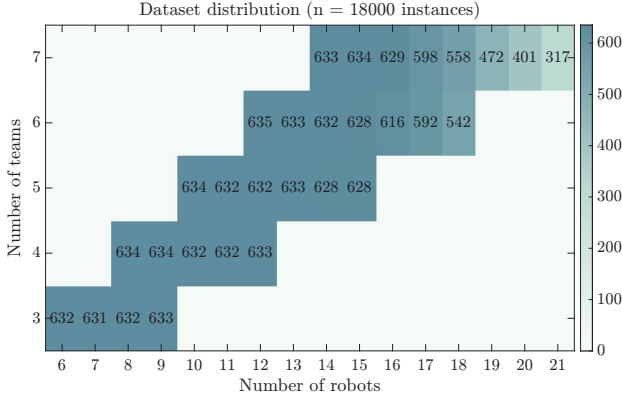


Fig. 3. Distribution of the 18,000-instance dataset over the number of teams and robots. The dataset covers heterogeneous collaboration problems with 3 to 7 teams and robot populations scaled with the number of teams.

is variable, with a maximum of seven classes in the generated dataset. The Hamilton mask is applied prior to the softmax operation, ensuring that probability mass is assigned only to feasible actions.

The GNN operates using one-hop message passing over the team interaction graph, consistent with the decentralized execution setting. Team, robot, and edge features are embedded into a latent space of dimension 128. The model is trained for 50 epochs using the AdamW optimizer [31] with a learning rate of 10^{-3} , weight decay of 10^{-4} , dropout rate of 0.1, and batch size of 128. All input features are normalized using statistics computed from the training set. The training objective is the masked cross-entropy loss defined in Equation (15), augmented with a binary auxiliary loss that predicts whether a robot should move or stay. This auxiliary loss is weighted by 0.15, and move-target decisions are further emphasized in the cross-entropy objective by a factor of 1.25, reflecting the increased difficulty of predicting destination teams among transfer actions.

The dataset is partitioned into 80% training, 10% validation, and 10% testing splits, corresponding to 14,400, 1,800, and 1,800 instances, respectively. The split preserves the distribution over problem sizes, ensuring that validation and test performance are evaluated on instances drawn from the same family of heterogeneous collaboration problems. Figure 3 illustrates the distribution of the dataset across the number of teams and robots. Due to the NP-hardness of the underlying optimization, as per Corollary 1, a timeout was imposed when computing exact labels; instances exceeding this limit were skipped and replaced, resulting in relatively fewer samples for larger team sizes.

Across all splits, the model achieves consistent predictive performance, indicating strong generalization beyond the training data. Four metrics are considered: exact destination accuracy, move/stay accuracy, top-3 accuracy, and move target accuracy. Exact accuracy measures whether the predicted destination matches the optimal assignment, while move/stay accuracy evaluates whether the model correctly predicts whether a robot remains in its current team or transfers. Top-3 accuracy captures whether the optimal destination is among the three

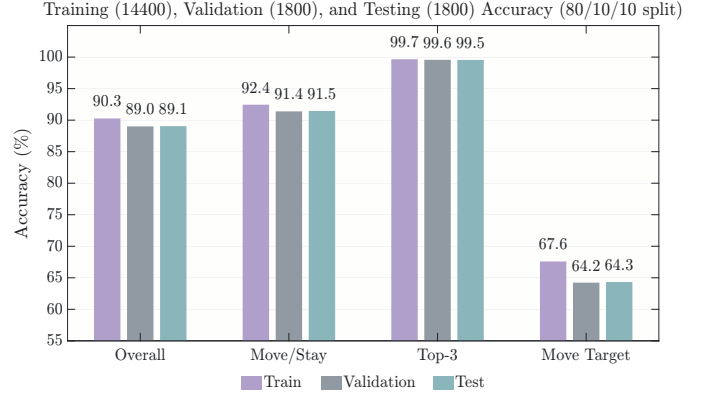


Fig. 4. Training, validation, and testing performance of the GNN policy. The model achieves similar validation and test accuracy, indicating that the learned policy generalizes beyond the training split.

highest-scoring feasible actions, and move target accuracy evaluates destination prediction conditioned on transfer actions. The quantitative results for these metrics are shown in Figure 4. The top-3 accuracy of 99.55% on the test split indicates that, even when the highest-scoring destination is not exactly correct, the optimal destination is almost always assigned high probability by the model. This suggests that many errors occur among closely ranked feasible destination teams rather than from a complete failure to identify the relevant action set.

We additionally evaluate move precision and move recall by treating the move-versus-stay decision as a binary classification problem, where the positive class corresponds to transferring a robot to another team. Move precision measures the fraction of predicted transfers that agree with the optimizer, while move recall measures the fraction of optimizer-labeled transfers that are identified by the learned policy. On the test split, the GNN achieves a move precision of 74.83% and a move recall of 77.94%. These values indicate that the model detects most beneficial transfers while maintaining a moderate false-transfer rate. Because transfer decisions are less frequent than “stay” decisions, move precision and move recall are

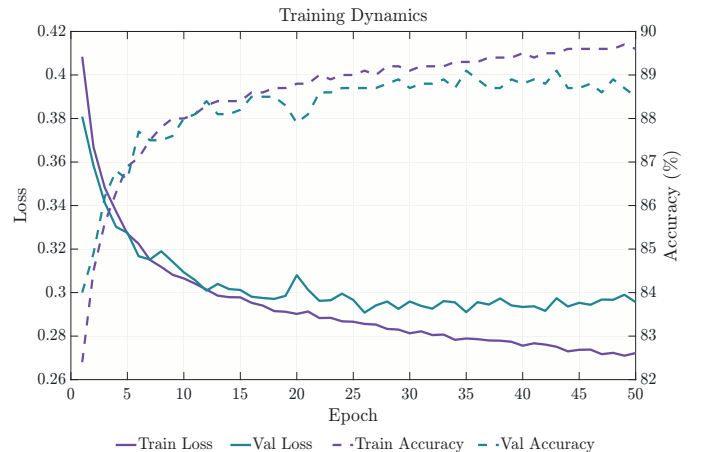


Fig. 5. Training dynamics of the GNN policy, showing the model loss and exact accuracy for both the training and validation splits.

TABLE III
MULTI-SEED ROBUSTNESS

Metric	Mean	Std.	Min	Max
Overall accuracy	88.94%	0.08%	88.79%	89.05%
Move/stay accuracy	91.33%	0.08%	91.22%	91.46%
Top-3 accuracy	99.51%	0.03%	99.46%	99.55%
Move target accuracy	63.29%	1.30%	60.58%	64.33%
Mean loss	0.2361	0.0013	0.2341	0.2376

reported in this section since they are standard classification measures derived from the binary confusion matrix [32] and are especially informative under class imbalance [33].

The training process exhibits stable convergence, with both loss and accuracy improving steadily over time. Moreover, the validation performance closely tracks the training performance, suggesting that the model does not overfit and instead learns a generalizable mapping from graph-structured system states to optimal transfer decisions. The evolution of the training and validation curves is presented in Figure 5.

To assess the robustness of the learned GNN policy, we repeated training and evaluation across eight random seeds using the same 18,000 dataset and the same 80/10/10 train-validation-test split protocol. The results, reported in Table III, show low variability across seeds for the main classification metrics: overall accuracy varied by less than 0.3%, move-stay accuracy remained tightly concentrated around 91.3%, and top-3 accuracy stayed above 99.4% for all runs. This indicates that the proposed model is not sensitive to a particular random initialization and that the reported performance in Figure 4 is representative of the learning framework. The move target metric exhibits larger variability, which is expected because it is evaluated only on the harder subset of decisions where a robot must transfer to another team and the model must select the exact destination. Overall, the multi-seed results demonstrate that the GNN policy learns a stable approximation of the Hamilton-admissible allocation rule across independent training runs.

B. Software Simulations

To evaluate the learned GNN policy beyond the small-scale supervised training regime, software simulations of the heterogeneous fire-fighting scenario introduced in Section V are performed. In these experiments, the learned policy is executed iteratively using the inference procedure in Algorithm 2, allowing the multi-team system to evolve through repeated robot reallocations and continuous fire suppression dynamics.

The simulations are performed to evaluate:

- 1) the ability of the learned policy to generate feasible heterogeneous collaborations online,
- 2) the scalability of the framework beyond the problem sizes used during supervised training,
- 3) the evolution of team compositions under repeated altruistic reallocations, and
- 4) the resulting reduction in global fire intensity over time.

Unless otherwise specified, each simulation begins from a randomly generated heterogeneous allocation satisfying the

sensing feasibility constraint introduced in Section V. Team interaction graphs are generated as connected random graphs, and initial fire-density fields are sampled independently for each team region. At every collaboration iteration, the GNN evaluates Hamilton-admissible robot transfers using one-hop message passing over the current interaction graph. Accepted transfers are executed according to Algorithm 2, after which robots reposition locally within their assigned team regions and the fire density evolves according to Equation (26). The process continues until no additional locally feasible collaborations are proposed by the learned policy and until all fires are extinguished.

Figure 6 illustrates execution sequences for systems of different scales. The first row (Figures 6a and 6b) shows a collaboration instance with 7 teams and 32 heterogeneous robots, while the second row (Figures 6c and 6d) illustrates a larger-scale, out-of-training-distribution deployment with 15 teams and 55 robots. In both cases, the initial allocation exhibits imbalance in capability distribution and fire intensity across teams. During execution, the learned policy reallocates sensing and fire-fighting robots across the interaction graph according to the real-time prediction of the Hamilton-admissible transfers. The resulting allocations demonstrate important behaviors: the learned policy naturally produces heterogeneous capability balancing across teams, redistributing sensing and fire-fighting resources according to the evolving mission state and the collaboration process terminates despite the absence of explicit centralized coordination during inference.

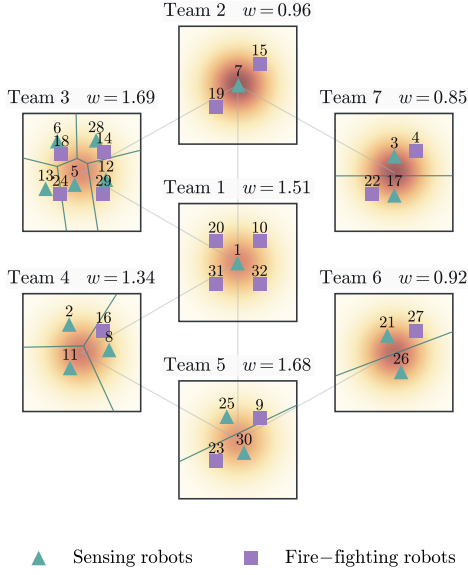
In the larger-scale simulation (Figure 6c and 6d), the learned policy generalizes beyond the problem sizes used during training. Although the GNN is trained only on instances containing up to 7 teams, the learned decentralized message-passing structure enables execution on substantially larger interaction graphs without architectural modification. This scalability arises because the policy operates locally over graph neighborhoods rather than relying on a fixed-size centralized state representation.

The runtime comparison between the exact optimizer and the learned GNN policy is summarized in Table IV. The results demonstrate a substantial difference in scalability between the two approaches. While the exact optimization procedure can solve small problem instances, its runtime grows rapidly with the number of teams and robots due to the combinatorial nature of the heterogeneous assignment problem. In contrast, the learned GNN policy maintains low inference times even for substantially larger systems.

For small-scale problems containing 3 to 5 teams, both methods execute within sub-second runtimes. However, beginning at 6 teams and 18 robots, the exact optimization runtime increases rapidly, requiring 35.4 seconds for a single simulation. At 7 teams and 21 robots, the exhaustive optimization becomes computationally intractable within the imposed timeout budget, consistent with the NP-hardness result established in Proposition 1. By comparison, the learned GNN policy continues to operate in real time, requiring only 0.201 seconds for the same system size.

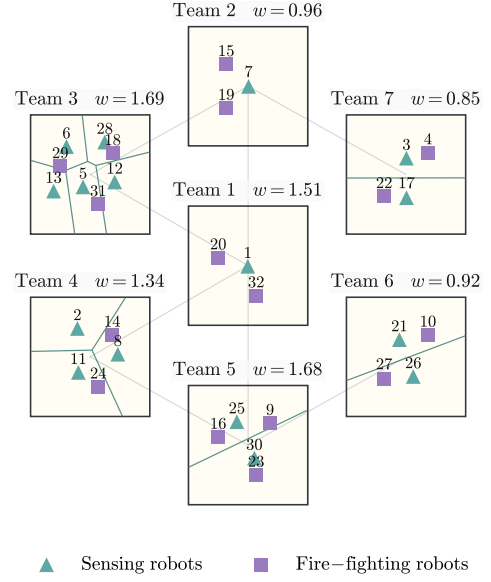
As the system size increases further, the runtime advantage of the learned policy becomes increasingly pronounced. Even

GNN Collaboration Execution | Step 0 | Initial State | Accepted Migrations: 0



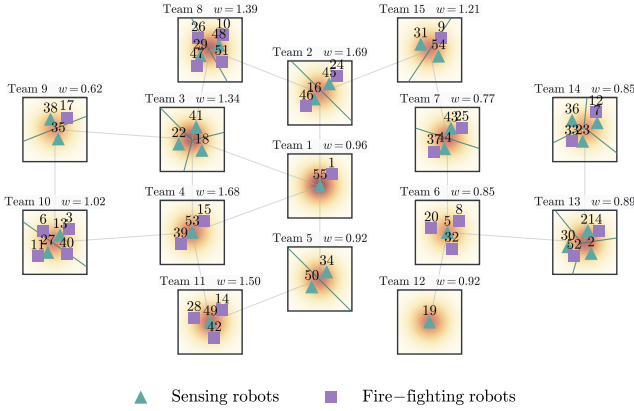
(a) Initial Allocation

GNN Collaboration Execution | Step 7 | Converged | Accepted Migrations: 0



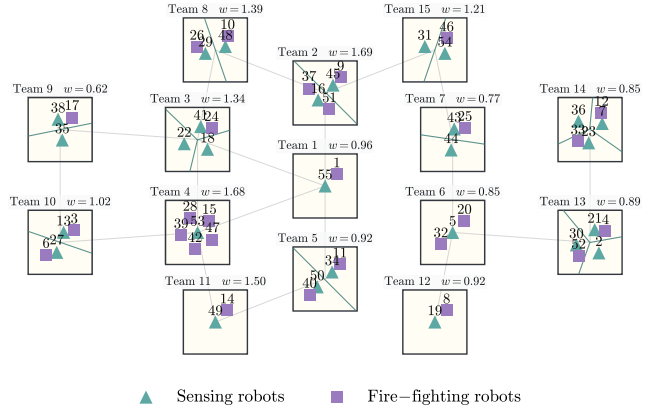
(b) Final Allocation

GNN Collaboration Execution | Step 0 | Initial State | Accepted Migrations: 0



(c) Initial Allocation

GNN Collaboration Execution | Step 8 | Converged | Accepted Migrations: 0



(d) Final Allocation

Fig. 6. Example executions of the proposed GNN-based heterogeneous altruistic collaboration framework in software simulations. The top row shows a system with 7 teams and 32 robots, while the bottom row shows a larger out-of-training-distribution system with 15 teams and 55 robots. Sensing robots and fire-fighting robots are represented by triangles and squares, respectively, and the numbers on top of each robot represents its respective identification number. Each team is assigned a weight w signifying its importance. Starting from the initial allocations, the learned GNN policy evaluates Hamilton-admissible transfers and iteratively executes decentralized robot migrations to improve collaborative fire suppression performance.

for large-scale systems containing 50 teams and 150 robots, the GNN policy completes the entire iterative collaboration process in approximately 55.3 seconds, with an average per-step runtime of 2.77 seconds. Importantly, these runtimes correspond to the complete execution of the iterative collaboration process, including graph construction, Hamilton-feasible transfer evaluation, message passing, and repeated allocation updates over multiple collaboration rounds. The learned policy therefore remains computationally tractable in regimes where exact optimization is no longer practically executable.

Overall, Table IV highlights the central motivation for the proposed learning-based framework. Although exact heterogeneous allocation rapidly becomes computationally infeasible,

the learned GNN policy provides a scalable approximation capable of generating real-time heterogeneous collaboration decisions on substantially larger multi-team systems.

C. Robotic Testbed Experiments

To validate the proposed framework on physical hardware, we deploy the learned GNN policy on a multi-robot testbed. The experiments evaluate whether the learned heterogeneous collaboration policy can successfully execute decentralized robot migrations and improve mission performance in a real robotic environment.

The experimental setup consists of 4 teams operating in distinct spatial regions connected through a team interaction

TABLE IV
 RUNTIME SCALING OF THE EXACT OPTIMIZATION AND THE GNN POLICY. EXACT AND GNN TOTAL RUNTIMES ARE FULL ITERATIVE EXECUTION RUNTIMES. MEAN STEP RUNTIMES ARE COMPUTED FROM THE PER-STEP RUNTIME EXECUTION.

Teams	Robots	Opt. Total (s)	Opt. Mean Step (s)	Opt. Steps	GNN Total (s)	GNN Mean Step (s)	GNN Steps
3	9	0.051	0.025	2	0.048	0.012	4
4	12	0.090	0.030	3	0.012	0.006	2
5	15	0.781	0.390	2	0.009	0.005	2
6	18	35.4	11.8	3	0.170	0.034	5
7	21	∞	—	1	0.201	0.050	4
8	24	∞	—	—	0.298	0.075	4
9	27	∞	—	—	0.816	0.204	4
10	30	∞	—	—	0.658	0.219	3
15	45	∞	—	—	1.957	0.326	6
25	75	∞	—	—	2.787	0.697	4
35	105	∞	—	—	16.3	2.335	7
40	120	∞	—	—	36.0	2.001	18
45	135	∞	—	—	48.5	2.425	20
50	150	∞	—	—	55.3	2.767	20

graph. A total of 15 robots are used, partitioned into sensing robots and fire-fighting robots according to the capability model introduced in Section V. The robots emulate the heterogeneous capability classes through software-defined roles. Robots marked with triangle overlays correspond to sensing robots, while robots marked with square overlays correspond to fire-fighting robots. The red heatmaps represent the fire-density fields $\phi_v(q)$ associated with each team region. The learned GNN policy runs externally in real time and communicates robot transfer decisions to the robotic execution layer.

Similar to the software simulations in Section VI-B, the experimental procedure follows the iterative inference pipeline described in Algorithm 2. At each collaboration round, the current multi-team state is encoded as a graph and processed by the learned GNN policy. The model evaluates Hamilton-admissible transfers through one-hop message passing over the interaction graph and predicts robot migration decisions. Accepted transfers are then executed physically by commanding robots to navigate from their current team region to the destination region. Once migrations terminate, robots locally reposition within their assigned regions and continue the fire-fighting task according to the dynamics in Equation (26).

Figure 7 illustrates the execution sequence of one hardware experiment. Figure 7a shows the initial allocation, where teams exhibit different sensing and fire-fighting compositions together with different fire intensities. The learned GNN policy identifies beneficial collaborations and initiates robot migrations across the interaction graph. Figure 7b shows the first collaboration phase, during which robots begin migrating toward teams predicted to benefit from additional capabilities. Figure 7c illustrates a subsequent collaboration round after the system state has been updated following the first migrations. Finally, Figure 7d shows the final allocation after the collaboration process converges and no additional beneficial Hamilton-admissible transfers are predicted by the learned policy.

The physical experiments demonstrate the proposed framework under realistic robotic constraints; the learned policy continues to generate feasible collaborations and successfully

supports robot migrations across teams. While the learned policy achieves strong performance and scalability, it does not guarantee global optimality and may occasionally produce suboptimal transfer sequences due to approximation errors in the learned action scores. Furthermore, because decisions are generated locally through one-hop message passing, performance may degrade in environments where optimal reallocations depend strongly on long-range (or multi-hop) global reassignment.

VII. CONCLUSION

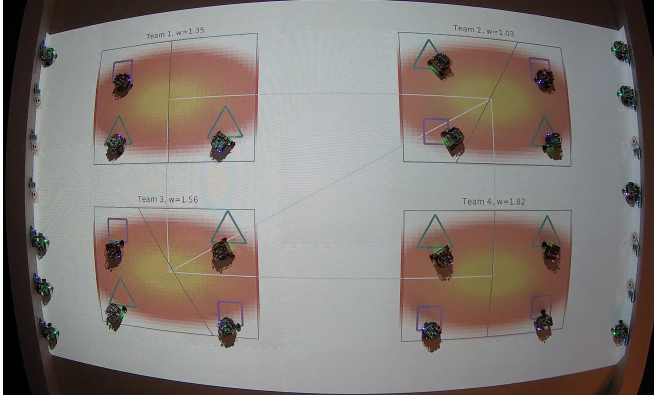
In conclusion, this paper presented a learning-based framework for heterogeneous altruistic collaboration in multi-team robotic systems. Building upon prior work that adapts Hamilton’s rule as a decentralized mechanism for regulating inter-team robot transfers, we extended the formulation to heterogeneous settings with capability-dependent mission evaluation functions and transfer costs. We showed that the resulting allocation problem is NP-hard and proposed a GNN policy under a CTDE framework to approximate Hamilton-admissible allocation decisions through decentralized message passing over the team interaction graph. The framework was instantiated in a heterogeneous fire-fighting scenario involving robots with complementary capabilities. Experimental results demonstrated that the learned policy achieves strong predictive performance, scales beyond the tractable regime by exact optimization, and maintains low inference runtime in large-scale simulations. Furthermore, deployment on a multi-robot testbed validated the ability of the proposed framework to execute decentralized robot migrations and support heterogeneous collaboration on physical robotic hardware.

ACKNOWLEDGMENT

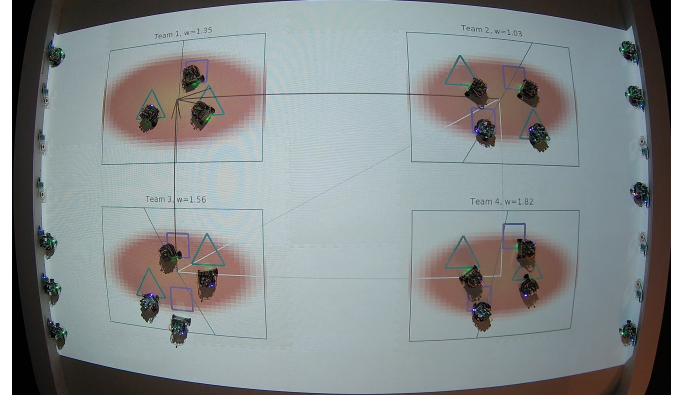
The authors want to thank Professor Yanning Shen and Alex Nguyen for helpful discussions.

REFERENCES

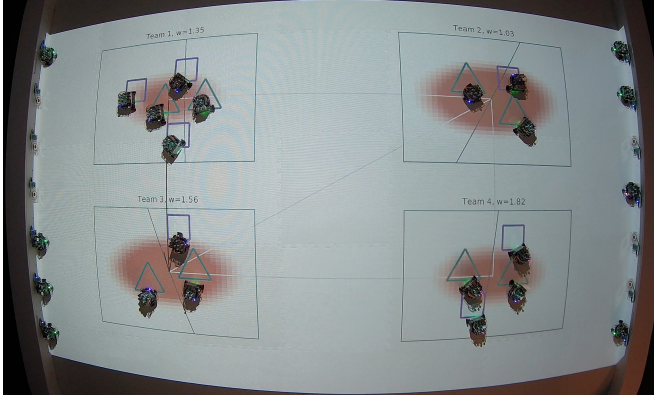
- [1] R. R. Murphy, *Disaster Robotics*. MIT Press, 2017.



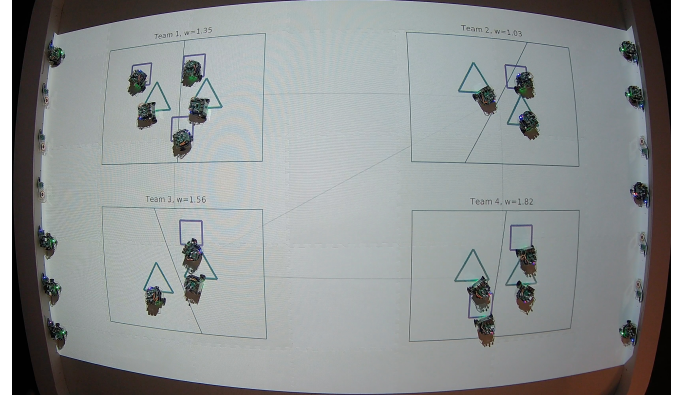
(a) Initial Allocation



(b) First Robot Migrations



(c) Second Robot Migrations



(d) Final Allocation

Fig. 7. Execution sequence of the proposed GNN-based heterogeneous altruistic collaboration framework on a multi-robot testbed. Teams operate in distinct spatial regions with heterogeneous sensing and fire-fighting robots, shown as triangles and squares, respectively. The learned policy evaluates Hamilton-admissible transfers online and coordinates decentralized robot migrations across teams to improve collaborative fire-fighting performance. The sequence illustrates the initial allocation, intermediate migration phases, and the final allocation after the collaboration process converges. The team weights, from top-left to bottom-right, are 1.35, 1.03, 1.56, and 1.82, respectively.

- [2] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.
- [3] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.
- [4] M. M. Zavlanos, L. Spesivtsev, and G. J. Pappas, "A distributed auction algorithm for the assignment problem," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*. IEEE, 2008, pp. 1212–1217.
- [5] S. Park, Y. D. Zhong, and N. E. Leonard, "Multi-robot task allocation games in dynamically changing environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8678–8684.
- [6] G. Notomista, S. Mayya, Y. Emam, C. Kroninger, A. Bohannon, S. Hutchinson, and M. Egerstedt, "A resilient and energy-aware task allocation framework for heterogeneous multirobot systems," *IEEE Transactions on Robotics (T-RO)*, vol. 38, no. 1, pp. 159–179, 2021.
- [7] R. Karam, R. Lin, B. A. Butler, and M. Egerstedt, "Resource allocation with multi-team collaboration based on hamilton's rule," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*. IEEE, 2025, pp. 6891–6898.
- [8] K. Lu, H. Xu, and Y. Zheng, "Distributed resource allocation via multi-agent systems under time-varying networks," *Automatica*, vol. 136, p. 110059, 2022.
- [9] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.
- [10] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.
- [11] P. Anussornnitarn, S. Y. Nof, and O. Etzion, "Decentralized control of cooperative and autonomous agents for solving the distributed resource allocation problem," *International Journal of Production Economics*, vol. 98, no. 2, pp. 114–128, 2005.
- [12] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics (T-RO)*, vol. 25, no. 4, pp. 912–926, 2009.
- [13] R. Karam, A. A. Nguyen, R. Lin, D. R. Martin, D. Morales, B. A. Butler, and M. Egerstedt, "Collaboration in multi-robot systems: Taxonomy and survey over frameworks for collaboration," 2026. [Online]. Available: <https://arxiv.org/abs/2603.23898>
- [14] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *Journal of Machine Learning Research*, vol. 21, no. 178, pp. 1–51, 2020.
- [15] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
- [16] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'hORIZON," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.
- [17] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.
- [18] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu,

- “Relational inductive biases, deep learning, and graph networks,” 2018. [Online]. Available: <https://arxiv.org/abs/1806.01261>
- [19] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, “Graph neural networks for decentralized multi-robot path planning,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 785–11 792.
 - [20] E. Tolstaya, F. Gama, J. Paulos, G. Pappas, V. Kumar, and A. Ribeiro, “Learning decentralized controllers for robot swarms with graph neural networks,” in *Proceedings of the Conference on Robot Learning*. PMLR, 2020, pp. 671–682.
 - [21] W. Kool, H. van Hoof, and M. Welling, “Attention, learn to solve routing problems!” 2019. [Online]. Available: <https://arxiv.org/abs/1803.08475>
 - [22] C. Zhang, W. Song, Z. Cao, J. Zhang, P. S. Tan, and X. Chi, “Learning to dispatch for job shop scheduling via deep reinforcement learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1621–1632.
 - [23] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, “Graph networks as learnable physics engines for inference and control,” in *International Conference on Machine Learning (ICML)*. PMLR, 2018, pp. 4470–4479.
 - [24] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. Oxford university press, 1999, no. 1.
 - [25] S. A. West, A. S. Griffin, and A. Gardner, “Evolutionary explanations for cooperation,” *Current biology*, vol. 17, no. 16, pp. R661–R672, 2007.
 - [26] W. D. Hamilton, “The genetical evolution of social behaviour,” *Journal of Theoretical Biology*, vol. 7, no. 1, pp. 1–16, 1964.
 - [27] B. A. Butler and M. Egerstedt, “Hamilton’s rule for enabling altruism in multi-agent systems,” in *IEEE Conference on Decision and Control (CDC)*, 2025, pp. 6776–6783.
 - [28] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.
 - [29] R. M. Karp, “Reducibility among combinatorial problems,” in *50 Years of Integer Programming 1958-2008: from the Early Years to the State-of-the-Art*. Springer, 2009, pp. 219–241.
 - [30] A. Krause and D. Golovin, “Submodular function maximization,” *Tractability*, vol. 3, no. 71-104, p. 3, 2014.
 - [31] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv preprint arXiv:1711.05101*, 2017.
 - [32] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.
 - [33] J. Davis and M. Goadrich, “The relationship between precision-recall and roc curves,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2006, pp. 233–240.