Extremely Large Collective Coalition Formation: Scalability

Neha G. Pusalkar
Oregon State University
Corvallis, USA
pusalkan@oregonstate.edu

Julie A. Adams
Oregon State University
Corvallis, USA
julie.a.adams@oregonstate.edu

ABSTRACT

Efficient coordination algorithms facilitate the deployment of robotic collectives for complex missions. The coalition formation for task allocation problem organizes robots into task-specific teams capable of optimal task execution. Developing scalable coalition formation algorithms for extremely large scale collectives is challenging because the computational complexity grows with the collective's size and the number of mission tasks. This work presents LN-GRAPE-S, a hedonic game-based coalition formation algorithm that generates near-optimal solutions for heterogeneous robotic collectives comprised of up to 10,000 robots and at most 5,000 tasks. LN-GRAPE-S leverages the collective's limited heterogeneity, relative to its size, and formulates a leader-level hedonic game that yields high-quality solutions with low communication costs. The algorithm achieves an average 29× improvement in runtime efficiency and an average 825× reduction in total communication overhead as validated through a centralized evaluation with simulated heterogeneous collectives. LN-GRAPE-S is the first distributed hedonic coalition formation algorithm to achieve a minimal communication overhead (i.e., 6.5 MB average) for extremely large scale collectives.

KEYWORDS

Coalition Formation; Hedonic Games; Swarms; Collectives

ACM Reference Format:
Neha G. Pusalkar and Julie A. Adams. 2026. Extremely Large Collective Coalition Formation: Scalability. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 10 pages. https://doi.org/10.65109/EENQ4709

1 INTRODUCTION

Robotic collectives are large groups of relatively simple robots that demonstrate complex coordinated behaviors through local interactions [35]. Collectives' emergent behavior draws inspiration from biological swarms (e.g., colonies of ants, flocks of birds) [5, 44]. Chains of thousands of marine salps extending up to hundreds of meters in lengths motivated the development of the Floatsalp prototype (Figure 1), as components of a novel reconfigurable robotic system [8, 45, 55]. Real-world deployments of these systems are envisioned to mimic biological salps in number and complexity, requiring coordination techniques for extremely large collectives. Solving the combinatorially complex coalition formation problem at this scale necessitates algorithms that produce solutions within reasonable computational and communication limits. This manuscript proposes a distributed coordination algorithm that leverages the extremely large collectives' compositional properties to produce high quality solutions faster and with lower communication costs.

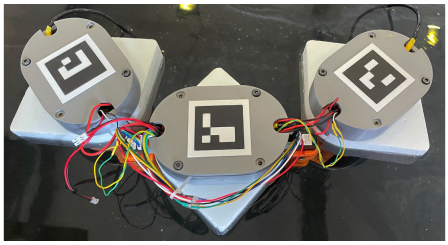

Figure 1: FloatSalp robot prototype [8, 55].

Multiple robot systems and collectives leverage individual members' diverse capabilities to accomplish complex tasks that cannot be performed by a single robot. Coalition formation partitions robots into task-oriented groups, such that the groups' aggregate capabilities satisfy the tasks' requirements. The coalition structure generation problem is NP-hard; thus, finding an optimal coalition structure scales exponentially with the number of robots [36, 41]. Traditional multiple robot solutions using exact and approximation algorithms are not scalable [2, 38, 57]. Auction-based approaches are applicable to collectives; however, they incur a significant communication overhead for large scale collectives [14]. Additionally, no existing algorithms' communication requirements have been evaluated for extremely large collectives containing up to 10,000 robots. Thus, developing scalable and communication efficient coalition formation algorithms for these collectives remains challenging.

Collective coalition formation methods have leveraged hedonic games to produce viable near real-time (e.g., < 5 minutes) solutions for heterogeneous collectives of up to 1,000 robots with centralized computation [13, 14]. The Group Agent Partitioning and Placing Event framework with the services extension (GRAPE-S) incorporated a services model [50] into an anonymous hedonic game for coalition formation with heterogeneous and multi-service robots [13, 23]. LeaderGRAPE-S incorporated a leader-follower hierarchy that reduced GRAPE-S's runtime for extremely large scale collectives, but did not significantly improve communication [32].

This manuscript presents Leader Negotiation GRAPE-S (LN-GRAPE-S), a hedonic coalition formation algorithm that leverages the leader-follower hierarchy and the collectives' heterogeneity to produce scalable and efficient coalition formation solutions for extremely large scale heterogeneous collectives. A simulation-based evaluation demonstrates that LN-GRAPE-S achieves minimal communication overhead (i.e., a 6.5 MB average) for 10,000 robot collectives, making it the first algorithm to demonstrate distributed

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/EENQ4709

hedonic game-based coalition formation with significantly lower communication costs. Furthermore, LN-GRAPE-S achieves near real-time performance for 5,000 robot collectives, produces optimal solutions for 10,000 robot multi-capability heterogeneous collectives in under 120 minutes, and demonstrates at least 9× runtime improvement over an existing hedonic game-based coalition formation algorithm for single capability collectives of 10,000 robots.

2 RELATED WORK

Coalition formation has been extensively studied for multiple agent [37, 42] and multiple robot systems [20, 49], with solutions including greedy heuristics [40, 41], market-based auctions [20, 22], optimization techniques [6, 33], and game-theoretic approaches [11, 26]. Greedy and optimization-based algorithms provide solution quality guarantees and support heterogeneity, but often rely on forming small fixed-size coalitions, failing to accommodate the collective's diverse task requirements, thereby limiting their applicability. Market-based algorithms require robots bid on tasks using ascending combinatorial auctions [48, 53], descending auctions [39], double round auctions [21, 54], etc. These methods incorporate heterogeneity, and eliminate the need for a central controller and restriction on coalition size; however, the communication overhead and runtime scale poorly with collective size [14].

Prior coalition formation in extremely large scale collectives research remains limited. A heuristic-based approach leveraged quantum-inspired multi-objective particle swarm optimization to generate Pareto-optimal solutions, which were ranked using a multi-criteria ranking method to find the best coalition [30]. Another approach modeled coalition formation as an iterative process and utilized a simulation-based framework to evaluate various agents' coalition selection strategies and obtained greedy solutions [24]. A similar iterative approach, C-Link, employed a hierarchical clustering algorithm to iteratively merge the most similar coalitions based on the characteristic function and produced suboptimal solutions for 2,372 agents in about four minutes [17, 18]. However, these methods either considered homogeneous agents [18, 24], or allocated a single task at a time [30]. Further, these approaches were centralized, limiting their applicability to collectives [18, 24, 30].

Recently, anonymous hedonic games were leveraged for collective coalition formation, where agents have preferences over coalition membership based on the coalition size [4, 15, 23]. Agents optimized their individual utilities by switching coalitions until a Nash stable coalition structure was obtained. Centralized hedonic game-based algorithms generated an initial coalition structure with singleton coalitions by identifying mutually best robot-task pairs using bipartite graph matching, with the remaining robots assigned to coalitions that maximized their individual utilities [10, 11]. A Nash stable solution was generated in 1.57s for 2,000 robots and ten tasks, but these approaches' centralized architecture necessitates frequent communication, which is impractical for extremely large scale collectives. The distributed Group Agent Partitioning and Placing Event (GRAPE) algorithm's anonymous hedonic game generated a Nash stable partition in polynomial time [23]. Distributed hedonic coalition formation algorithms for heterogeneous multiple robot systems were neither evaluated for 10,000 robots, nor assessed for their communication requirements [16, 52]. Hedonic games [23] demonstrated significant runtime improvement relative to auction-based methods [39, 48] for a 1,000 robot heterogeneous collective [13, 14]. Additionally, GRAPE-S produced near-optimal solutions (i.e., $\geq$ 95% utility) in near real-time for 1,000 robot collectives. LeaderGRAPE-S improved upon GRAPE-S's runtime performance by integrating a leader-follower model that required adding dedicated leader robots to the collective, providing evidence for using hedonic games for extremely large scale collectives [32].

Leader-based methods were used to control swarms of up to 1,000 robots [9, 25, 51]. Formation control problems allocated robots to specific formation positions using an auction-like negotiation process [29]. Real-time formation control combined the leader-follower strategy with mobile ad-hoc network-based communication [58]. A resource-constrained coalition formation solution designated agents that detected tasks as leaders who broadcast coalition proposals [1, 31]. Followers evaluated multiple leaders' requests, and bid to join the coalitions providing the highest individual utility. Coalition leaders formed optimal coalitions using merge-and-split [1] or quantum-inspired genetic algorithms [31]. Another resource-constrained coalition formation approach's leader robots provided key resources for tasks [7]. Each leader's preliminary coalitions added follower robots that satisfied the task requirements, and the coalitions with maximum utility were assigned the tasks with a $O(n^3)$ computational complexity for n robots. A similar approach generated initial coalitions by randomly assigning followers to tasks [56]. Leaders evaluated followers' utilities considering task completion and resource utilization rates, and followers selected coalitions based on altruistic preferences until a Nash stable partition was obtained in $O(nm^2 l_{max})$ time, where m and l_{max} denote the number of tasks and the number of iterations to converge, respectively.

LeaderGRAPE-S, a hedonic game-based coalition formation algorithm, employed the leader-follower hierarchy with one leader robot per capability, facilitating decomposing the collective coalition formation problem into smaller subproblems [32]. Each leader independently coordinated negotiations among their respective follower subgroups using GRAPE-S, rather than the entire collective, improving runtime ($O(n^2 m/|S|)$ for $|S|$ number of services). The proposed algorithm, LN-GRAPE-S, incorporates the leader-follower hierarchy in a novel way by formulating a hedonic game over the set of leaders, rather than between individual followers as in LeaderGRAPE-S. LN-GRAPE-S abstracts decision-making, which involves iterative belief updates and synchronization between hedonic game players, at the leader-level, rather than the individual robot level (i.e., followers). This abstraction substantially reduces the magnitude of negotiations, resulting in faster runtimes and lower communication overhead relative to LeaderGRAPE-S.

Computationally intractable problems with large numbers of agents were reduced to polynomially tractable problems consisting of a constant number of agent types [47]. The notion of agent types permitted the use of a concise characteristic function representation that reduced complexity [19, 46]. Agents of the same types with the same marginal contributions were considered to be strategically equivalent [43]. Inspired by this type-based approach, this manuscript treats all robots that provide the same capabilities as strategically equivalent with respect to the task requirements. Restricting the number of agent types to be significantly smaller than the collective size supports scalability.

3 PROBLEM FORMULATION

The coalition formation for task allocation problem is modeled with a set of N heterogeneous robots, $\mathcal{R} = \{R_1, \ldots, R_N\}$ and a set of M tasks, $\mathcal{T} = \{T_1, \ldots, T_M\}$. The solution is a coalition structure, $\Pi = \{\pi_1, \ldots, \pi_M \mid \pi_i = \langle C_i, T_i \rangle, \forall i \in [1, M]\}$, where C_i is the robot coalition assigned to task T_i, and each robot is assigned to a single task. The collective's heterogeneity is expressed using a capability model, where each robot provides diverse capabilities from a set of P capabilities, $\mathcal{Q} = \{q_1, \ldots, q_P\}$. The capabilities represent different functions or high-level behaviors (e.g., manipulation, perception, grasping) individual robots can perform. The choice to limit capabilities per robot is motivated by the limited functionality of individual zooids in a salp chain [12]. The robots can have at most two capabilities; thus, a robot R_i's capability vector is expressed as $Cap_{R_i} = [q_x, \ldots] \mid q_x \in \mathcal{Q}, |Cap_{R_i}| \leq 2$. Each task can require multiple units of multiple capability types; thus, a task T_j's requirements are expressed as $Cap_{T_j} = [q_x : s_{j,x}, \ldots]$ where $q_x \in \mathcal{Q}$ represents the required capability type, and $s_{j,x}$ represents the number of units required for task T_j. A coalition is awarded the task's full utility if the coalition collectively provides the task's required capabilities. LeaderGRAPE-S [32] used a services model [50], where services have the same underlying notion of robot capabilities. The key difference, however, is that the services model permits each robot to contribute only one service to the task, whereas LN-GRAPE-S allows a robot to provide multiple capabilities to a task.

4 LEADER NEGOTIATION WITH GRAPE-S

LN-GRAPE-S's leader-follower structure reduces the coalition formation problem's computational and communication complexity. Robots possessing an identical set of capabilities randomly select one leader, with the remainder being followers. Unlike traditional leader-follower approaches, where leaders are associated with individual tasks and form coalitions by requesting followers, LN-GRAPE-S's leaders are the principal decision makers in the hedonic coalition formation process, allocating their followers (including themselves) across different tasks. LN-GRAPE-S incorporates a type-based robot heterogeneity representation. Each leaders' followers are equivalent and have the same marginal contribution per capability, based on the types, which enables leaders to consider only those tasks that require their capabilities. Thus, the leader-follower hierarchy, type-based representation, and hedonic game properties enable LN-GRAPE-S to lower algorithmic runtime and communication overhead, while producing high quality solutions.

A hedonic coalition formation game is defined by a pair $(\mathcal{N}, \succsim)$, where $\mathcal{N}$ represents a finite set of players, and $\succsim = (\succsim_1, \ldots, \succsim_{|\mathcal{N}|})$ is the players' preference profile [3, 4, 27]. Hedonic game agents have preferences over the coalitions to which they belong, as captured by each agent's preference relation ($\succsim_i$). LN-GRAPE-S's hedonic game is formulated over a set of leaders ($\mathcal{L}$), who prefer coalitions in which their follower allocation maximally contributes towards satisfying the task requirements. The number of followers $f_{i,j}$, that each leader L_i allocates to task T_j is determined as the minimum of L_i's unallocated followers (v_i), and the largest unmet demand among the capabilities L_i can provide for the task and is given by:

$$f_{i,j} = \min(v_i, \max_{q \in Cap_{L_i}} (s_{j,q} - c_{j,q}^{-i})), \tag{1}$$

where $s_{j,q}$ is the number of units of capability q required by task T_j and $c_{j,q}^{-i}$ denotes the number of units of capability q assigned to task T_j without capability contributions from L_i's followers, and Cap_{L_i} represents L_i's capabilities. The task's capability allocation after including $f_{i,j}$ followers is given by:

$$c_{j,q} = c_{j,q}^{-i} + \alpha_{i,j,q}(f_{i,j}),$$

where $\alpha_{i,j,q}(f_{i,j})$ denotes capability q's effective contribution towards task T_j from $f_{i,j}$ followers depending on the task's capability deficit ($\alpha_{i,j,q}(.) = 0$ for $q \notin Cap_{L_i}$). The task's per task-capability utility with $f_{i,j}$ followers can be derived using Equation (2):

$$Util_{j,q}(c_{j,q}) = c_{j,q} \times e^{-\frac{c_{j,q}}{s_{j,q}}+1}, \tag{2}$$

where $c_{j,q}$ is the number of capability q units assigned to T_j (from all leaders). T_j's total utility is given by $Util_j = \sum_{q \in \mathcal{Q}} Util_{j,q}(c_{j,q})$. A leader L_i's preference, expressed as its utility $marg_i(j, f_{i,j})$, represents L_i's marginal utility for assigning $f_{i,j}$ followers in the current iteration to task T_j as given by Equation (3).

$$marg_i(T_j, f_{i,j}) = \sum_{q \in Cap_{L_i}} Util_{j,q}(c_{j,q}) - Util_{j,q}(c_{j,q}^{-i}), \tag{3}$$

where $Util_{j,k}(c_{j,q})$ represents T_j's per task-capability utility when considering contributions from all leaders as given by Equation (2), and $Util_{j,q}(c_{j,q}^{-i})$ denotes T_j's per task-capability utility without capability contributions from Leader L_i's followers.

4.1 LN-GRAPE-S Algorithm

The LN-GRAPE-S's hedonic game integrates the leader-follower hierarchy into the hedonic game framework and leverages the collective's limited heterogeneity to form coalitions for extremely large scale collectives. Unlike LeaderGRAPE-S [32] that incorporates additional leader robots, LN-GRAPE-S selects leaders at random from the collective's homogeneous capability groups to form the leaders set $\mathcal{L}$. A leader L_i considers only the subset of tasks $\mathcal{T}(i) \subseteq \mathcal{T}$ that require its capabilities. All robots are initially assigned to a void task T_0 with zero capability requirements that represents being unassigned (line 2). Each leader maintains a belief Π^i, containing the most recent information about each task's assigned capabilities. Given Π^i, the leader evaluates the marginal utility of all tasks with unmet capability requirements and allocates followers to the task yielding the highest marginal utility at each iteration (lines 4-9). The leader updates its belief Π^i to reflect the new capability allocations, increments the number of times its belief has been updated (w^i), and generates a new random timestamp $time^i$ (lines 10-12). A leader is satisfied when its marginal utility is 0, indicating no incentive to change its current allocation. All leaders locally update their beliefs during each iteration, and broadcast their beliefs to the neighboring leaders Nb_i at the end of the iteration (line 14). A valid belief is selected to synchronize the leaders' local beliefs. Each leader evaluates messages received from the neighboring leaders to determine if a belief Π^d has been updated more times ($w^d > w^k$), or more recently ($time^d > time^i$), and regards Π^d as the valid belief (lines 15-22). The leader re-evaluates its preferred allocation of followers to tasks using Π^d during the next iteration. The algorithm terminates when a Nash stable solution is obtained.

Algorithm 1: LN-GRAPE-S on leader L_i

```
/* Initialization                                          */
w^i ← 0; time^i ← 0;
π^i ← {C_0 = R, C_t = ∅ ∀t ∈ T};
while solution is not Nash stable do
    for j ∈ |T(i)| do
        Calculate f_{i,j} using Equation (1)
        Calculate marg_i(T_j, f_{i,j}) using Equations (3) and (2)
    end
    (T_j^*, f_{i,j}^*) ← arg max_j marg_i(T_j, f_{i,j})
    if (marg_i^*(T_j^*, f_{i,j}^*) > 0) then
        Allocate f_{i,j}^* followers to T_j^* and update Π^i;
        w^i ← w^i + 1;
        time^i ∈ unif[0,1];
    end
    /* Broadcast the local belief to neighbors             */
    Broadcast M^i = {w^i, time^i, Π^i} and receive M^k from
      neighbors ∀l_k ∈ Nb_i.
    Construct M_rcv^i = {M^i, ∀M^k};
    /* Distributed Mutex                                   */
    for each message M_d ∈ M_rcv^i do
        if (w^i < w^d) or (w^i = w^d & time^i < time^d) then
            Π^i ← Π^d;
            w^i ← w^d;
            time^i ← time^d;
        end
    end
end
```

LN-GRAPE-S's hedonic game is formulated over leaders. Each leader L_i iterates over a task subset $\mathcal{T}(i)$. Leaders negotiate their capability contributions (and corresponding follower allocations) to tasks until a Nash stable partition is achieved. The robots are assigned to task-specific coalitions according to each leader's allocations. $|\mathcal{L}| \ll$ N; thus, the reduced scale of negotiations results in a substantial reduction in runtime and communication overhead. LN-GRAPE-S assumes a fully connected communication network, consistent with GRAPE-S and LeaderGRAPE-S.

4.2 Analysis

Theorem 4.1. *LN-GRAPE-S converges to a Nash stable partition.*

Proof. LN-GRAPE-S's convergence is established by demonstrating that LN-GRAPE-S constitutes a finite potential game, which guarantees the existence of a Nash stable partition [28]. The system's global potential Φ is defined as the sum of per task-capability utilities for all tasks and all capabilities:

$$\Phi = \sum_{j \in \mathcal{T}} \sum_{q \in Q} Util_{j,q}(c_{j,q}).$$

A unilateral change in leader L_i's follower assignment to task T_j from $f'_{i,j}$ to $f''_{i,j}$ in the current iteration changes task T_j's capability allocation for capability q from $c'_{j,q}$ to $c''_{j,q}$:

$$c'_{j,q} = c^{-i}_{j,q} + \alpha_{i,j,q}(f'_{i,j})$$
$$c''_{j,q} = c^{-i}_{j,q} + \alpha_{i,j,q}(f''_{i,j})$$

The change in L_i's marginal utility can be derived using Equations (2) and (3) as follows:

$$\begin{aligned}
&\Delta marg_i \\
&= \sum_{q \in Cap_{L_i}} [Util_{j,q}(c''_{j,q}) - Util_{j,q}(c'_{j,q})] \\
&= \sum_{q \in Cap_{L_i}} [Util_{j,q}(c^{-i}_{j,q} + \alpha_{i,j,q}(f''_{i,j})) - Util_{j,q}(c^{-i}_{j,q} + \alpha_{i,j,q}(f'_{i,j}))].
\end{aligned}$$

The summation over leader's capabilities indicates that the change in follower assignment only affects the capabilities that a leader can provide to a task. The corresponding change in the global potential ($\Delta\Phi$) is the difference in the per task-capability utilities associated with task T_j and capabilities of L_i before and after the change in the follower assignment. The per task-capability utilities associated with tasks $j \notin \mathcal{T}$ and capabilities $q \notin Cap_{L_i}$ remain unchanged.

$$\Delta\Phi = \sum_{q \in Cap_{L_i}} Util_{j,q}(c''_{j,q}) - Util_{j,q}(c'_{j,q}).$$

Thus, the change in the global potential equals the change in L_i's marginal utility, showing that LN-GRAPE-S satisfies the exact potential game property. Each leader's unilateral follower reassignment increases the global potential by exactly its marginal utility gain, and leaders only make moves that result in strictly positive gains (i.e., $\Delta marg_i > 0$). The number of plausible improving allocations is finite, as leaders have a finite number of followers. Therefore, LN-GRAPE-S is a finite potential game and is guaranteed to converge to a Nash stable partition [28]. □

Lemma 4.2. *LN-GRAPE-S reaches a Nash stable partition in at most $|\mathcal{L}| \cdot M \cdot f_{max}$ iterations, where f_{max} represents the maximum number of followers allocated to any leader.*

Proof. Each leader iteratively assigns followers to a task that yields the highest positive marginal utility, until no further improvement is possible. The number of followers assigned per iteration (Equation (1)) is one in the worst case (e.g., smaller capability deficits for tasks). Thus, leader L_i requires a maximum of $M \cdot f_i$ iterations to allocate its f_i followers across M tasks. Considering all $|\mathcal{L}|$ leaders in the collective, and the maximum possible number of followers per leader f_{max}, LN-GRAPE-S's total number of iterations is bounded by $|\mathcal{L}| \cdot M \cdot f_{max}$. Note that this bound is conservative, as leaders assign multiple followers in a single iteration, resulting in a smaller number of iterations in practice. □

4.2.1 Time complexity. LN-GRAPE-S reaches Nash stability in $O(|\mathcal{L}|Mf_{max})$ iterations (Lemma 4.2). Every leader evaluates at most M tasks and S capabilities per iteration. Thus, LN-GRAPE-S's per leader runtime is bounded by $O(|\mathcal{L}|M^2 f_{max} S)$. The number of capabilities evaluated by each leader equals the number of capabilities it provides. The runtime represents a conservative upper bound, assuming the worst-case scenario where only one follower may be assigned per leader iteration.

4.2.2 Communication complexity. Each leader broadcasts its belief with a message of size $O(|\mathcal{L}|MP)$, where P represents the number of capabilities. Considering the total number of iterations to achieve convergence, as given by Lemma 4.2, LN-GRAPE-S's total communication complexity is bounded by $O(|\mathcal{L}|^2M^2f_{max}P)$.

Lemma 4.3. *LN-GRAPE-S achieves a Nash stable partition in at most $M \cdot N$ iterations for an N-robot homogeneous collective.*

Proof. Applying the leader-follower hierarchy, a homogeneous collective of size N will have one leader and at most N followers. Using Lemma 4.2, the number of iterations for a N-robot homogeneous collective to converge to a Nash stable partition is $M \cdot N$. □

5 ALGORITHMIC COMPARISON

Auctions and hedonic games were identified as the most suitable algorithms for collective coalition formation [14]; however, auction-based approaches' communication requirement and runtime scaled poorly with collective size [39, 48]. Existing scalable hedonic game-based coalition formation approaches [10, 11, 16, 52] have only been evaluated on collectives of up to 2,400 robots [10], and provide no analysis of total communication requirements, limiting their suitability as baselines. The hedonic game-based GRAPE-S generated near-optimal (i.e., ≥ 95% utility) Nash stable coalition structures in near real-time (i.e., < 5 minutes) for 1,000 robot collectives, but had high communication requirements [13], making GRAPE-S a suitable baseline. LeaderGRAPE-S achieved faster runtimes [32] and lower communication overheads compared to GRAPE-S for extremely large scale collectives containing up to 10,000 robots [32], making it an appropriate algorithm for comparison with LN-GRAPE-S.

GRAPE-S's hedonic game is modeled over all robots in the collective. GRAPE-S [13] uses a services model that allows the use of only one of the robots' capabilities for the task. Each robot individually maintains a belief state, and evaluates each task-capability pair to find its most preferred coalition. Robots broadcast their beliefs to the entire collective, synchronize their beliefs using a distributed mutex, and repeat the entire process until a Nash-stable solution is achieved. LeaderGRAPE-S [32] and LN-GRAPE-S employ a leader-follower hierarchy along with a concise type-based game representation. LeaderGRAPE-S divides the collective coalition formation problem into multiple leader-centric subproblems by including additional leader robots. Each leader represents a single capability type and all other robots providing that capability are designated as the leader's followers. Each leader independently runs GRAPE-S over its followers. This decomposition reduces the number of players participating in each leader's hedonic game to the number of followers per leader, which is still larger than LN-GRAPE-S's hedonic game. LN-GRAPE-S exploits the leader-follower hierarchy by implementing a leader-level synchronization mechanism and formulating the hedonic game over a set of leaders. LN-GRAPE-S's hedonic game operates over a substantially smaller set of players relative to GRAPE-S and LeaderGRAPE-S.

5.1 Experimental Design

A centralized simulation-based experiment was used to assess the performance of the three algorithms in terms of runtime, solution optimality, and communication overhead. The independent variables were based on a simulated salp-inspired robotic collective. The collective size represents the total number of robots in the collective (Collective size: 1,000, 5,000, 10,000). The number of tasks is a percentage of the collective size, or *percent tasks* (% tasks: 1, 10, 50). Biological salps' limited functional diversity motivated the design of collectives characterized by ten unique capabilities, with at most 2 capabilities per robot (Cap/R: 1,2).

Twenty-five achievable mission problems (i.e., the collective provided sufficient capabilities to execute all tasks simultaneously) were randomly generated for each independent variable combination, resulting in a total of 450 trials per algorithm. The number of tasks, task requirements, task utilities, and individual robot capabilities were identical across all algorithms' problems. LeaderGRAPE-S problems added leader robots equivalent to the number of capability types to the base collective size, but were not assigned to any coalition. Robots' and tasks' capabilities were randomly selected from the total number of ten capabilities. Task utilities were randomly assigned in the range [1, 50]. GRAPE-S and LeaderGRAPE-S experiments were performed on a HP Z640 Workstation (Intel Xeon processor, 62 GB RAM), and a Dell Precision 5820 Tower (Intel Xeon processor, 32 GB RAM), respectively, both running Ubuntu 18.04. LN-GRAPE-S experiments were performed on a Dell Precision 5820 Tower (Intel Xeon processor, 32 GB RAM), running Ubuntu 24.04. All experiments assumed a fully-connected communication topology and instantaneous communication. All algorithms are decentralized, but were evaluated in a centralized fashion.

The runtime dependent variable represents the total time required to produce a coalition formation solution, and is expressed as hours, minutes, and seconds (HH:MM:SS). Percent utility is the ratio of the solution's utility to the total possible mission utility. The total communication metric aggregates all message sizes, and is measured in megabytes (MB). Solutions with lower runtimes, higher percent utilities, and lower communication are preferred.

5.2 Results

All three algorithms produced solutions for all trials. The results were analyzed using Kruskal-Wallis tests. Mann-Whitney-U post-hoc tests were used for statistically significant results for collective size and % tasks. The Kruskal-Wallis effect size, η^2 is based on the H-statistic. The Mann-Whitney U post-hoc tests' effect size reported using Rank Biserial Correlation r_{rb} is calculated using the U-statistic. Mann-Whitney U tests analyzed performance with respect to Cap/R, with the r_{rb} effect size based on the U-statistic. Large effect sizes ($\eta^2 > 0.14$) signify stronger influence of a variable on the results. A large r_{rb} ($|r_{rb}| \geq 0.5$) indicates a substantial difference between the two groups being compared. The algorithms were compared using the Friedman test with Conover post-hoc analysis for all trials across all collective sizes, % tasks, and Cap/R. The Friedman effect size, Kendall's W was calculated using the Friedman χ^2 statistic. A large W (W > 0.5) suggests a strong difference among the algorithms. The Conover post-hoc effect size is given by r_{rb}.

5.2.1 Runtime Results. LN-GRAPE-S had faster runtimes and lower variances than GRAPE-S across all corresponding independent variable combinations, as shown in Table 1. LN-GRAPE-S performed better for all collectives with 1% tasks across both capability levels,

for all collectives with 10% tasks and 1 Cap/R, and for collectives with 5,000 and 10,000 robots for 10% tasks with 2 Cap/R. LN-GRAPE-S required slightly longer average runtimes (≈ 9 seconds) for 1,000 robot collectives with 2 Cap/R and 10% tasks. LN-GRAPE-S for the 1 Cap/R, 1,000 robot collectives was 88.23% faster for 1% tasks, and 16.81% faster for 10% tasks. LN-GRAPE-S required substantially longer runtimes for all collective sizes and Cap/R at 50% tasks. The extremely large collective with 10,000 robots, 50% tasks, and 2 Cap/R incurred the longest runtimes across all algorithms.

Table 1: The runtime (HH:MM:SS) descriptive statistics [mean (std)] for ten capability types by algorithm, collective size, number of tasks, and number of capabilities per robot.

Collective Size	GRAPE-S	LeaderGRAPE-S	LN-GRAPE-S
1 Capability Per Robot			
1% tasks			
1000	:13 (:00)	:00 (:00)	:00 (:00)
5000	:27:08 (:20)	:01:38 (:01)	:01 (:00)
10000	03:36:36 (:01:32)	:14:01 (:05)	:09 (:00)
10% tasks			
1000	:59 (:00)	:04 (:00)	:04 (:01)
5000	02:06:00 (:54)	:10:37 (:05)	:03:45 (:02)
10000	16:54:00 (:05:55)	01:29:21 (:37)	:25:30 (:12)
50% tasks			
1000	:02:15 (:01)	:07 (:00)	:29 (:00)
5000	:04:52:12 (:02:20)	:15:43 (:06)	:34:31 (:14)
10000	39:23:03 (:33:26)	02:12:08 (:54)	04:18:57 (:45)
2 Capabilities Per Robot			
1% tasks			
1000	:23 (:05)	:02 (:00)	:00 (:00)
5000	:45:50 (:10:36)	:06:52 (:02)	:11 (:00)
10000	06:02:24 (01:03:36)	:56:44 (:20)	:49 (:00)
10% tasks			
1000	:02:40 (:37)	:19 (:00)	:28 (:00)
5000	05:28:12 (:53:43)	:42:45 (:19)	:19:10 (:10)
10000	42:59:34 (09:46:48)	05:53:24 (:02:55)	01:54:00 (:49)
50% tasks			
1000	:05:43 (:01:31)	:28 (:00)	:04:13 (:01)
5000	12:18:00 (02:37:48)	01:03:00 (:29)	03:37:31 (:02:27)
10000	107:13:18 (27:36:33)	08:44:24 (:02:29)	24:29:54 (:06:40)

Baseline: A comparison of all algorithms across all independent variable combinations revealed significant differences in runtime ($\chi^2(n = 450) = 679.68, p < 0.001, W = 0.75$). Post-hoc analysis indicated significant differences ($p < 0.001$) between LN-GRAPE-S's and GRAPE-S's runtimes with a large effect size ($|r_{rb}| = 0.523$). LN-GRAPE-S's and LeaderGRAPE-S's runtimes differed significantly ($p < 0.001$), but with a smaller effect size ($|r_{rb}| = 0.131$).

Collective Size and Percent Tasks: LN-GRAPE-S's runtimes increased significantly with collective size ($H(n = 450) = 152.74, p < 0.001, \eta^2 = 0.33$) and % tasks ($H(n = 450) = 261.06, p < 0.001, \eta^2 = 0.58$). Significant runtimes ($p < 0.001$) were also observed within GRAPE-S and LeaderGRAPE-S by collective size and % tasks [32]. The effect size η^2 represents the proportion of variance of a dependent variable associated with an independent variable [34]. LN-GRAPE-S's large effect sizes indicate both collective size and % tasks strongly affect its runtime scalability. Both GRAPE-S and LeaderGRAPE-S had large effect sizes by collective size (η^2>0.14) and medium to large effect sizes by % tasks ($\eta^2 > 0.06$) [32]. LN-GRAPE-S's smaller effect size (0.33) for collective size compared to the other two GRAPE-S: (0.79), and LeaderGRAPE-S: (0.80) [32], explained the comparatively lower runtime variances with increasing collective size. Post-hoc analysis indicated significant pairwise differences ($p < 0.05$) in runtime across all collective sizes and across all % tasks within each algorithm. LN-GRAPE-S had large post-hoc effect sizes between 1,000 and 5,000 robots, and between 1,000 and 10,000 robots. A medium effect size for 5,000 vs. 10,000 robots ($|r_{rb}| = 0.33$) indicated lower runtime variance with respect to collective size for larger collectives. Large post-hoc effect sizes were observed for all pairs of % tasks.

Capabilities Per Robot: LN-GRAPE-S exhibited significant runtime differences from 1 to 2 Cap/R with a smaller effect size ($U(n = 225) = 18,605.0, p < 0.001, |r_{rb}| = 0.26$). Significant results were observed for GRAPE-S ($U(n = 225) = 19,963.0, p < 0.001, |r_{rb}| = 0.21$) and LeaderGRAPE-S ($U(n = 225) = 19,375.0, p < 0.001, |r_{rb}| = 0.23$). LN-GRAPE-S's runtimes were shorter and had lower variance for 1 Cap/R than for 2 Cap/R across the corresponding values of collective sizes and % tasks. LN-GRAPE-S's runtimes for 1,000 robots with 1% tasks were 85% faster for 1 Cap/R than for 2 Cap/R.

5.2.2 Utility Results. LN-GRAPE-S generated optimal solutions (i.e., utility = 100%) for all collective sizes and % tasks combinations with 1 Cap/R. An 89.50% average utility (standard deviation 10.50%), with a 62.95% minimum, was observed for 1,000 robot collectives with 1% tasks and 2 Cap/R. All other 2 Cap/R combinations produced optimal solutions. LeaderGRAPE-S generated optimal solutions for all trials [32]. GRAPE-S had optimal and near-optimal (i.e., ≥ 95% utility) solutions for 1 and 2 Cap/R, respectively [32].

5.2.3 Communication Results. LN-GRAPE-S had the lowest communication with lowest variances of the three algorithms across all corresponding independent variable combinations, as shown in Table 2. LN-GRAPE-S's minimum communication, 0.02 MB, occurred for a 1,000 robot collective with 1% tasks and 1 Cap/R, while the maximum, 6.55 MB, occurred for a 10,000 robot collective with 50% tasks and 2 Cap/R. LN-GRAPE-S's maximum communication requirement (6.55 MB) was 99.93% and 99.31% lower than the maximum values for GRAPE-S (9,762.70 MB) and LeaderGRAPE-S (960.87 MB), respectively. LN-GRAPE-S's minimum communication requirement (0.02 MB) was 99.91% lower than GRAPE-S's minimum (24 MB) and 99.19% lower than LeaderGRAPE-S's minimum communication requirement (2.47 MB). LN-GRAPE-S's total communication increased with collective size, % tasks, and Cap/R; however, the increase was negligible relative to GRAPE-S and LeaderGRAPE-S. LN-GRAPE-S also exhibited negligible variances in total communication.

Baseline: A comparison of all algorithms across all collective sizes, % tasks, and Cap/R exhibited significant differences in the total communication ($\chi^2(n = 450) = 898.00, p < 0.001, W = 0.99$). LN-GRAPE-S and GRAPE-S differed significantly ($p < 0.001$) with a large post-hoc effect size ($|r_{rb}| = 0.99$). LN-GRAPE-S's and LeaderGRAPE-S's communication also differed significantly ($p < 0.001$) with a large post-hoc effect size ($|r_{rb}| = 0.94$).

Collective Size and Percent Tasks: LN-GRAPE-S's communication differed significantly with increasing collective size ($H(n =$

$450) = 185.79, p < 0.001, \eta^2 = 0.41$) and % tasks ($H(n = 450) = 200.58, p < 0.001, \eta^2 = 0.44$). LN-GRAPE-S's large effect size indicated strong effects of collective size and % tasks on its communication. Both GRAPE-S's ($H(n = 450) = 405.12, p < 0.001, \eta^2 = 0.90$) and LeaderGRAPE-S's ($H(n = 450) = 364.51, p < 0.001, \eta^2 = 0.81$) communication increased significantly ($\eta^2 > 0.14$) with collective size, but was not significantly affected by % tasks [32]. LN-GRAPE-S's smaller effect size (0.41) by collective size compared to GRAPE-S (0.90) and LeaderGRAPE-S (0.83) indicated a comparatively weaker influence of collective size on LN-GRAPE-S's communication requirements. LN-GRAPE-S's post-hoc analysis exhibited significant pairwise differences ($p < 0.001$) across all collective sizes, with large post-hoc effect sizes for 1,000 vs. 5,000 robot collectives, and 1,000 vs. 10,000 robot collectives. A medium post-hoc effect size ($|r_{rb}| = 0.33$) was observed for 5,000 vs. 10,000 robot collectives. LN-GRAPE-S demonstrated significant pairwise differences ($p < 0.001$) across all % tasks, with large post-hoc effect sizes for 1% vs. 10% tasks, and 1% vs. 50% tasks, and a medium post-hoc effect size ($|r_{rb}| = 0.22$) for 10% vs. 50% tasks. Both GRAPE-S and LeaderGRAPE-S revealed significant post-hoc analysis ($p < 0.001$) by collective size with large effect sizes ($|r_{rb}| > 0.5$) for all pairs of collective sizes.

Table 2: The communication (MB) descriptive statistics [mean (std)] for ten capabilities by algorithm, collective size, number of tasks, and number of capabilities per robot.

Collective Size	GRAPE-S	LeaderGRAPE-S	LN-GRAPE-S
1 Capability Per Robot			
1% tasks			
1000	24.00 (0.00)	2.42 (0.00)	0.02 (0.00)
5000	600.00 (0.00)	60.10 (0.03)	0.11 (0.00)
10000	2400.00 (0.00)	240.24 (0.09)	0.21 (0.00)
10% tasks			
1000	24.00 (0.00)	2.42 (0.00)	0.13 (0.00)
5000	600.00 (0.00)	60.09 (0.04)	0.69 (0.00)
10000	2400.00 (0.00)	240.23 (0.10)	1.38 (0.00)
50% tasks			
1000	24.00 (0.00)	2.42 (0.00)	0.20 (0.00)
5000	600.00 (0.00)	60.10 (0.05)	1.01 (0.00)
10000	2400.00 (0.00)	240.20 (0.08)	2.03 (0.00)
2 Capabilities Per Robot			
1% tasks			
1000	34.89 (7.64)	9.64 (0.01)	0.07 (0.00)
5000	843.39 (188.52)	240.19 (0.11)	0.30 (0.00)
10000	3340.83 (580.83)	960.39 (0.15)	0.58 (0.00)
10% tasks			
1000	50.57 (11.55)	9.63 (0.01)	0.38 (0.00)
5000	1244.95 (202.89)	240.21 (0.09)	1.83 (0.01)
10000	4887.62 (1100.06)	960.37 (0.17)	3.66 (0.01)
50% tasks			
1000	50.27 (16.76)	9.63 (0.02)	0.65 (0.00)
5000	1311.99 (279.98)	240.17 (0.07)	3.25 (0.01)
10000	5706.33 (1477.26)	960.34 (0.16)	6.50 (0.01)

Capabilities per robot: LN-GRAPE-S's total communication differed significantly from 1 to 2 Cap/R, with a medium effect size ($U(n = 225) = 15,625.0, p < 0.001, |r_{rb}| = 0.38$). Significant results with medium effect sizes were also observed for GRAPE-S ($U(n = 225) = 17,062.5, p < 0.001, |r_{rb}| = 0.32$) and LeaderGRAPE-S ($U(n = 225) = 14,717.5, p < 0.001, |r_{rb}| = 0.41$). LN-GRAPE-S's average communication for 1 Cap/R was at least 60% lower than that for 2 Cap/R, for the corresponding collective sizes and % tasks.

5.3 Discussion

The comparative evaluation demonstrates the strong influence of each algorithm's mechanism on scalability in terms of runtime and total communication, especially for extremely large collectives.

Runtime: LN-GRAPE-S was hypothesized to incur shorter runtimes than GRAPE-S, and comparable runtimes to LeaderGRAPE-S. LN-GRAPE-S achieved significantly shorter runtimes than GRAPE-S, but the hypothesis in relation to LeaderGRAPE-S was only supported with 1% and 10% tasks. LN-GRAPE-S's faster runtimes are due to its fundamentally different coordination mechanism (Section 5). LN-GRAPE-S's hedonic game is formulated over a set of leaders representing unique capability sets. The number of leaders depends on the number of available unique capability sets, which is typically significantly smaller than the collective size. Belief synchronization among leaders, rather than among the entire collective, accelerates convergence to a Nash stable solution. Furthermore, LN-GRAPE-S's leaders allocate followers in batches (i.e., more than one follower can be assigned to a task per iteration). Each follower contributes multiple capabilities to a task if required. Both, batch allocation and allocation of multiple capabilities per follower in LN-GRAPE-S result in larger marginal gains per leader per iteration because of faster task requirement fulfillment. Larger gains reduce the number of iterations required to reach a high-potential state (i.e., maximum utility state), resulting in faster runtimes.

GRAPE-S's hedonic game requires individual robots to maintain, update and synchronize beliefs. An increase in collective size substantially increases the magnitude of belief synchronization and the number of iterations to achieve a Nash stable solution. Larger collective sizes lead to higher stochasticity in selecting the deciding robot, resulting in a higher runtime variance. An increase in % tasks and Cap/R increases the number of task-capability pairs each robot must evaluate, further contributing to GRAPE-S's slower runtimes and higher variance. LN-GRAPE-S's belief synchronization occurs for a significantly smaller subset of robots (i.e., leaders), which leads to faster runtimes. Furthermore, synchronization at the leader level, not the individual robot level, substantially reduces the randomness in selecting the deciding leader, resulting in lower variances.

LeaderGRAPE-S's leaders coordinate allocations by independently running GRAPE-S over their respective follower sets (Section 5). The collective size determines the size of leaders' follower sets, resulting in increased runtimes with increasing collective size, that are slower than LN-GRAPE-S for lower percent tasks (i.e., 1% and 10%). LN-GRAPE-S's leader-level abstraction and batch allocation of followers contribute to faster runtimes for 1% and 10% tasks. An increase in the number of tasks reduces the per-task capability requirement and the average coalition size per task (assuming achievable missions, see Section 5.1). Thus, the 1% and 10% task problems have higher per-task capability requirements compared to the 50% task problems. LN-GRAPE-S's leaders' batch allocation of followers enables simultaneous fulfillment of multiple task requirements for 1% and 10% tasks, reducing the overall number of

iterations, and the runtime. LeaderGRAPE-S's individual followers evaluate task-capability pairs, resulting in increased runtime for lower % tasks, which have higher per-task capability requirements. LN-GRAPE-S loses the benefit of batch allocation at 50% tasks where per-task capability requirements are smaller, implying a smaller average coalition size. LeaderGRAPE-S's socially inhibitive utility function (same as GRAPE-S [13]) encourages followers to choose smaller coalitions, and thus, followers find higher utility coalitions faster, thereby reducing the runtime. Smaller average coalition size limits the number of followers LN-GRAPE-S allocates per iteration (the worst-case being allocating one follower per iterations). Allocating fewer followers per iteration produces smaller marginal gains, resulting in a higher number of iterations to reach convergence, and hence, the increased runtime. LN-GRAPE-S's 9s slower runtime relative to LeaderGRAPE-S for a 1,000 robot collective and 10% tasks also results from the smaller coalition size per task.

Utility: LN-GRAPE-S's hypothesis for generating near-optimal solutions was supported for most independent variable combinations, yielding optimal solutions. Suboptimal solutions were produced in 60% of the trials for 1,000 robots with 1% tasks and 2 Cap/R. LN-GRAPE-S followers with 2 Cap/R allocated both capabilities (even when one may not be required), unlike LeaderGRAPE-S, that allocated only 1 capability to the task. Additionally, fewer tasks (i.e., 10 tasks for $N = 1,000$) implied fewer options with perfectly matching capability deficits, forcing LN-GRAPE-S leaders to settle on suboptimal (i.e., only 1 capability contributing to the task) allocations. LN-GRAPE-S's minimum utility (62.95%) exceeds the 50% suboptimality lower bound of GRAPE's Nash stable partitions [23].

Communication: LN-GRAPE-S was hypothesized to incur significantly shorter communication requirements than both algorithms and the hypothesis was fully supported. LN-GRAPE-S demonstrates the least total communication requirement across all experiments, achieving average reductions of 825× compared to GRAPE-S and 121× compared to LeaderGRAPE-S. The leader-level abstraction in LN-GRAPE-S's hedonic game significantly reduces the belief synchronization required to achieve a Nash stable solution. LN-GRAPE-S's message size depends on $|\mathcal{L}| \ll N$, and on $M \leq N/2$. The smaller message size relative to the collective size, and the leader-level belief synchronization contributes to LN-GRAPE-S's minimal communication overhead. The batch allocation of followers reduces the number of iterations required to achieve Nash convergence, decreasing the total number of message broadcasts, and hence, the total communication. The absolute increase in LN-GRAPE-S's communication with an increase in collective size is orders of magnitude smaller than GRAPE-S and LeaderGRAPE-S, indicating that leader-level belief synchronization effectively achieves low bandwidth communication, even with 10,000 robots. LN-GRAPE-S's iterations vary negligibly across trials per independent variable combination (Lemma 4.2), explaining the minimal communication variance. LN-GRAPE-S's significantly lower communication overhead demonstrate its suitability for deployment in bandwidth-constrained real-world missions.

GRAPE-S has the highest total communication because each robot maintains beliefs about every other robots' task-capability assignments and broadcasts messages to the entire collective each iteration. Increasing % tasks and Cap/R increases the number of task-capability pairs evaluated by each robot, increasing the number of iterations. GRAPE-S's global synchronization involves broadcasting larger messages at the collective's scale, resulting in communication cubically increasing with collective size ($O(N^3)$). LN-GRAPE-S broadcasts messages only among the leaders, and as $|\mathcal{L}| \ll N$, LN-GRAPE-S's communication increases negligibly compared to GRAPE-S, even for extremely large collectives.

LeaderGRAPE-S runs GRAPE-S within each leader's follower subgroups. Message size depends on the number of followers per subgroup, which is smaller than the collective size, but significantly larger than the number of LN-GRAPE-S's leaders. The absence of leader-level negotiation restricts LeaderGRAPE-S's belief synchronization to individual leader-centric problems. Consequently, LeaderGRAPE-S's overall communication requirement is determined by the number and size of follower-level messages exchanged within each leader subgroup ($O(f^3)$, consistent with GRAPE-S), explaining its higher communication than LN-GRAPE-S.

LN-GRAPE-S's significant improvements in runtime efficiency and communication overhead are attributed to the novel hedonic game formulation over a set of leaders that enables belief synchronization at the leader-level, rather than the individual robot level. Manufacturing constraints in real-world extremely large collectives limit the capability diversity in the collective ($P < 10$) and per robot ($Cap/R < 5$), preserving $|\mathcal{L}| \ll N$ and ensuring scalability despite the combinatorial growth in $|\mathcal{L}|$ with P and Cap/R.

6 CONCLUSION

This manuscript introduced LN-GRAPE-S, a hedonic game-based coalition formation algorithm that leverages collectives' limited heterogeneity and a leader-follower hierarchy to generate near-optimal solutions with a minimal communication overhead of 6.5 MB in the most complex case involving a 10,000 robot collective with 2 capabilities per robot and 5000 tasks. LN-GRAPE-S demonstrated 825× and 121× average communication reductions compared to GRAPE-S and LeaderGRAPE-S, respectively. The algorithm achieved a 29× average runtime reduction compared to GRAPE-S, and consistently outperformed LeaderGRAPE-S in most settings, with a modest runtime increase over LeaderGRAPE-S for 50% tasks, representing an acceptable tradeoff given the substantial communication reduction. These results demonstrate LN-GRAPE-S's scalability and potential for real-world coalition formation in extremely large heterogeneous robotic collectives, reflecting the coordination observed in salps.

Future work will focus on reducing LN-GRAPE-S's runtimes for 50% tasks without compromising its minimal communication overhead or solution optimality, and on evaluating all three algorithms in a distributed setting for a more realistic performance assessment. Extending LN-GRAPE-S to robots with more than two capabilities can broaden its applicability to other heterogeneous collectives. Finally, incorporating techniques for robot reallocations to address leader failures can improve LN-GRAPE-S's applicability to dynamic uncertain environments for long duration autonomous missions.

ACKNOWLEDGMENTS

This effort was supported by ONR grant N00014-23-1-2171. The views, opinions, and findings are those of the authors and do not represent the official views or policies of ONR.

REFERENCES

[1] Fatemeh Afghah, Mohammad Zaeri-Amirani, Abolfazl Razi, Jacob Chakareski, and Elizabeth Bentley. 2018. A coalition formation approach to coordinated task allocation in heterogeneous UAV networks. In *Annual American Control Conference*. IEEE, 5968–5975.

[2] Ashay Aswale and Carlo Pinciroli. 2023. Heterogeneous coalition formation and scheduling with multi-skilled robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5402–5409.

[3] Haris Aziz, Florian Brandl, Felix Brandt, Paul Harrenstein, Martin Olsen, and Dominik Peters. 2019. Fractional hedonic games. *ACM Transactions on Economics and Computation* 7, 2 (2019), 1–29.

[4] Haris Aziz and Rahul Savani. 2016. *Hedonic Games*. Cambridge University Press, Cambridge. 356–376 pages.

[5] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. 2013. Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence* 7 (2013), 1–41.

[6] Lei Cao, He shun Tan, Hui Peng, and Ming cong Pan. 2014. Multiple UAVs hierarchical dynamic task allocation based on PSO-FSA and decentralized auction. In *IEEE International Conference on Robotics and Biomimetics*. IEEE, 2368–2373.

[7] Jian Chen, Xiao Yan, Haoyao Chen, and Dong Sun. 2010. Resource constrained multirobot task allocation with a leader-follower coalition method. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5093–5098.

[8] Jinwoo Choi. 2025. *Gait-family-based hierarchical control of kinematic locomoting systems*. Ph.D. Dissertation. Oregon State University.

[9] Iain D Couzin, Jens Krause, Nigel R Franks, and Simon A Levin. 2005. Effective leadership and decision-making in animal groups on the move. *Nature* 433, 7025 (2005), 513–516.

[10] Emily Czarnecki and Ayan Dutta. 2021. Scalable hedonic coalition formation for task allocation with heterogeneous robots. *Intelligent Service Robotics* 14, 3 (2021), 501–517.

[11] Emily Czatnecki and Ayan Dutta. 2019. Hedonic coalition formation for task allocation with heterogeneous robots. In *IEEE International Conference on Systems, Man and Cybernetics*. 1024–1029.

[12] A Damian-Serrano, M Hughes, and KR Sutherland. 2023. A new molecular phylogeny of salps (Tunicata: Thalicea: Salpida) and the evolutionary history of their colonial architecture. *Integrative Organismal Biology* 5, 1 (2023), obad037.

[13] Grace Diehl. 2023. *Collective coalition formation*. Ph.D. Dissertation. Oregon State University.

[14] Grace Diehl and Julie A Adams. 2024. The viability of domain constrained coalition formation for robotic collectives. *Swarm Intelligence* (2024), 1–42.

[15] Jacques H Dreze and Joseph Greenberg. 1980. Hedonic coalitions: Optimality and stability. *Econometrica: Journal of the Econometric Society* (1980), 987–1003.

[16] Ayan Dutta, Vladimir Ufimtsev, Tuffa Said, Inmo Jang, and Roger Eggen. 2021. Distributed hedonic coalition formation for multi-robot task allocation. In *IEEE International Conference on Automation Science and Engineering*. IEEE, 639–644.

[17] Alessandro Farinelli, Manuele Bicego, Filippo Bistaffa, and Sarvapali D Ramchurn. 2017. A hierarchical clustering approach to large-scale near-optimal coalition formation with quality guarantees. *Engineering Applications of Artificial Intelligence* 59 (2017), 170–185.

[18] Alessandro Farinelli, Manuele Bicego, Sarvapali Ramchurn, and Marco Zuchelli. 2013. C-Link: A hierarchical clustering approach to large-scale near-optimal coalition formation. In *International Joint Conference on Artificial Intelligence*.

[19] Gianluigi Greco, Enrico Malizia, Francesco Scarcello, and Luigi Palopoli. 2012. Hard and easy k-typed compact coalitional games: the knowledge of player types marks the boundary. In *European Conference on Artificial Intelligence* (Montpellier, France). 372–377.

[20] José Guerrero and Gabriel Oliver. 2012. Multi-robot coalition formation in real-time scenarios. *Robotics and Autonomous Systems* 60, 10 (2012), 1295–1307.

[21] Jose Guerrero, Gabriel Oliver, and Oscar Valero. 2017. Multi-robot coalitions formation with deadlines: Complexity analysis and solutions. *PLOS One* 12, 1 (2017), e0170659.

[22] Muhammad Irfan and Adil Farooq. 2016. Auction-based task allocation scheme for dynamic coalition formations in limited robotic swarms with heterogeneous capabilities. In *International Conference on Intelligent Systems Engineering*. IEEE, 210–215.

[23] Inmo Jang, Hyo-Sang Shin, and Antonios Tsourdos. 2018. Anonymous hedonic game for task allocation in a large-scale multiple agent system. *IEEE Transactions on Robotics* 34, 6 (2018), 1534–1548.

[24] Pavel Janovsky and Scott A DeLoach. 2016. Multi-agent simulation framework for large-scale coalition formation. In *IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE, 343–350.

[25] Michal A Lewkowicz, Rohil Agarwal, and Nilanjan Chakraborty. 2019. Distributed algorithm for selecting leaders for supervisory robotic swarm control. In *IEEE International Symposium on Multi-Robot and Multi-Agent Systems*. IEEE, 112–118.

[26] Qinyuan Li, Minyi Li, Bao Quoc Vo, and Ryszard Kowalczyk. 2021. An anytime algorithm for dynamic multi-agent task allocation problems. In *IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion*. IEEE, 249–256.

[27] Gianpiero Monaco, Luca Moscardelli, and Yllka Velaj. 2019. On the performance of stable outcomes in modified fractional hedonic games with egalitarian social welfare. In *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*. 873–881.

[28] Dov Monderer and Lloyd S Shapley. 1996. Potential games. *Games and economic behavior* 14, 1 (1996), 124–143.

[29] Sergio Monteiro and Estela Bicho. 2008. Robot formations: Robots allocation and leader-follower pairs. In *IEEE International Conference on Robotics and Automation*. 3769–3775.

[30] Carla Mouradian, Jagruti Sahoo, Roch H Glitho, Monique J Morrow, and Paul A Polakos. 2017. A coalition formation algorithm for multi-robot task allocation in large-scale natural disasters. In *IEEE International Wireless Communications and Mobile Computing Conference*. IEEE, 1909–1914.

[31] Sajad Mousavi, Fatemeh Afghah, Jonathan D Ashdown, and Kurt Turck. 2018. Leader-follower based coalition formation in large-scale UAV networks, a quantum evolutionary approach. In *IEEE Conference on Computer Communications Workshops*. IEEE, 882–887.

[32] Neha G. Pusalkar and Julie A. Adams. 2025. Leader-based coalition formation for extremely large scale collectives. In *IEEE Conference on Artificial Intelligence*. 701–706.

[33] Sarvapali Ramchurn, Maria Polukarov, Alessando Farinelli, Ngoc Cuong Truong, and Nicholas Jennings. 2010. Coalition formation with spatial and temporal constraints. *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems* 3, 1181–1188. https://doi.org/10.1145/1838186.1838191

[34] John TE Richardson. 2011. Eta squared and partial eta squared as measures of effect size in educational research. *Educational research review* 6, 2 (2011), 135–147.

[35] Erol Şahin. 2005. Swarm robotics: From sources of inspiration to domains of application. In *Swarm Robotics*, Erol Şahin and William M. Spears (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 10–20.

[36] Tuomas Sandholm, Kate Larson, Martin Andersson, Onn Shehory, and Fernando Tohmé. 1999. Coalition structure generation with worst case guarantees. *Artificial Intelligence* 111, 1-2 (1999), 209–238.

[37] Samriddhi Sarkar, Mariana Curado Malta, and Animesh Dutta. 2022. A survey on applications of coalition formation in multi-agent systems. *Concurrency and Computation: Practice and Experience* 34, 11 (2022), e6876.

[38] Travis C. Service and Julie A. Adams. 2011. Coalition formation for task allocation: theory and algorithms. *Autonomous Agents and Multi-Agent Systems* 22, 2 (2011), 225–248.

[39] Travis C Service, Sayan D Sen, and Julie A Adams. 2014. A simultaneous descending auction for task allocation. In *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 379–384.

[40] Onn Shehory and Sarit Kraus. 1995. Task allocation via coalition formation among autonomous agents. In *International Joint Conference on Artificial Intelligence - Volume 1* (Montreal, Quebec, Canada). 655–661.

[41] Onn Shehory and Sarit Kraus. 1998. Methods for task allocation via agent coalition formation. *Artificial intelligence* 101, 1-2 (1998), 165–200.

[42] Onn M Shehory, Katia Sycara, and Somesh Jha. 1998. Multi-agent coordination through coalition formation. In *Intelligent Agents IV Agent Theories, Architectures, and Languages: International Workshop*. Springer, 143–154.

[43] Tammar Shrot, Yonatan Aumann, and Sarit Kraus. 2010. On agent types in coalition formation problems. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. Citeseer, 757–764.

[44] David JT Sumpter. 2010. Collective animal behavior. In *Collective animal behavior*. Princeton University Press, Princeton.

[45] Kelly R Sutherland and Laurence P Madin. 2010. Comparative jet wake structure and swimming performance of salps. *Journal of Experimental Biology* 213, 17 (2010), 2967–2975.

[46] Suguru Ueda, Atsushi Iwasaki, Vincent Conitzer, Naoki Ohta, Yuko Sakurai, and Makoto Yokoo. 2018. Coalition structure generation in cooperative games with compact representations. *Autonomous Agents and Multi-Agent Systems* 32 (2018), 503–533.

[47] Suguru Ueda, Makoto Kitaki, Atsushi Iwasaki, and Makoto Yokoo. 2011. Concise characteristic function representations in coalitional games based on agent types. In *International Joint Conference on Artificial Intelligence*, Vol. 11. 393–399.

[48] Lovekesh Vig and Julie A Adams. 2006. Market-based multi-robot coalition formation. In *Distributed Autonomous Robotic Systems 7*. Springer, 227–236.

[49] Lovekesh Vig and Julie A Adams. 2006. Multi-robot coalition formation. *IEEE Transactions on Robotics* 22, 4 (2006), 637–649.

[50] Lovekesh Vig and Julie A Adams. 2007. Coalition formation: From software agents to robots. *Journal of Intelligent and Robotic Systems* 50 (2007), 85–118.

[51] Phillip Walker, Saman Amirpour Amraii, Michael Lewis, Nilanjan Chakraborty, and Katia Sycara. 2014. Control of swarms with multiple leader agents. In *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 3567–3572.

[52] Lexing Wang, Tenghai Qiu, Zhiqiang Pu, Jianqiang Yi, Jinying Zhu, and Wanmai Yuan. 2024. Hedonic coalition formation for distributed task allocation in heterogeneous multi-agent system. *International Journal of Control, Automation and*

Systems 22, 4 (2024), 1212–1224.

[53] Xiao Wen and Zhen-Gang Zhao. 2021. Multi-robot task allocation based on combinatorial auction. In *IEEE International Conference on Control, Mechatronics and Automation*. IEEE, 27–32.

[54] Bing Xie, Shaofei Chen, Jing Chen, and LinCheng Shen. 2018. A mutual-selecting market-based mechanism for dynamic coalition formation. *International Journal of Advanced Robotic Systems* 15, 1 (2018), 1729881418755840.

[55] Yanhao Yang, Nina L. Hecht, Yousef Salaman-Maclara, Nathan Justus, Zachary A. Thomas, Farhan Rozaidi, and Ross L. Hatton. 2025. Geometric data-driven multi-jet locomotion inspired by salps. arXiv:2503.08817 [cs.RO]

[56] Liwang Zhang, Dong Liang, Minglong Li, Wenjing Yang, and Shaowu Yang. 2024. Coalition formation game approach for task allocation in heterogeneous multi-robot systems under resource constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 3439–3446.

[57] Yu Zhang and Lynne E Parker. 2013. Considering inter-task resource constraints in task allocation. *Autonomous Agents and Multi-Agent Systems* 26, 3 (2013), 389–419.

[58] Yi Zhang, Li Zeng, Yanhua Li, and Quanjie Liu. 2009. Multi-robot formation control using leader-follower for MANET. In *IEEE International Conference on Robotics and Biomimetics*. IEEE, 337–342.