Hierarchical Needs Based Self-Adaptive Framework For Cooperative Multi-Robot System

Qin Yang Ramviyas Parasuraman

Abstract*—Research in multi-robot and swarm systems has seen significant interest in cooperation of agents in complex and dynamic environments. To effectively adapt to unknown environments and maximize the utility of the group, robots need to cooperate, share information, and make a suitable plan according to the specific scenario. Inspired by Maslow's hierarchy of human needs and systems theory, we introduce** *Robot's Need Hierarchy* **and propose a new solution called** *Self-Adaptive Swarm System* **(SASS). It combines multi-robot perception, communication, planning, and execution with the cooperative management of conflicts through a distributed** *Negotiation-Agreement Mechanism* **that prioritizes robot's needs. We also decompose the complex tasks into simple executable behaviors through several** *Atomic Operations, such as selection, formation, and routing. We evaluate SASS through simulating static and dynamic tasks and comparing them with the state-of-the-art collision-aware task assignment method integrated into our framework.**

I. Introduction

Natural systems (living beings) and artificial systems (robotic agents) are characterized by apparently complex behaviors that emerge as a result of often nonlinear spatiotemporal interactions among a large number of components at different levels of organization [1]. Simple principles acting at the agent level can result in complex behavior at the global level in a swarm system. Swarm intelligence is the collective behavior of distributed and self-organized systems [2].

Multi-robot systems (MRS) [3] potentially share the properties of swarm intelligence in practical applications such as search, rescue, mining, map construction, exploration. MRS that allows task-dependent dynamic reconfiguration into a team is among the grand challenges in Robotics [4], necessitating the research at the intersection of communication, control, and perception. Currently, planning-based approaches combined with star-shaped communication models can not generally scale or handle a large number of agents in a distributed or decentralized manner [5].

Rizk [6] group heterogeneous MRS into four levels based on the task complexity and the level of automation. The fourth level of automation combines task decomposition, coalition formation, task allocation, and task execution (planning) and control. However, researches addressing this fourth level is very thin. Therefore, to drive advanced automation systems, the MRS framework needs to be combined with auxiliary controllers to handle conflicts, decompose the complicated task, and adapt to dynamic changes in the task assignments.

* The authors are with the Heterogeneous Robotics (HeRo) Lab, Department of Computer Science, University of Georgia, Athens, GA 30602, USA. Email: {*qy03103,ramviyas*}*@uga.edu*.

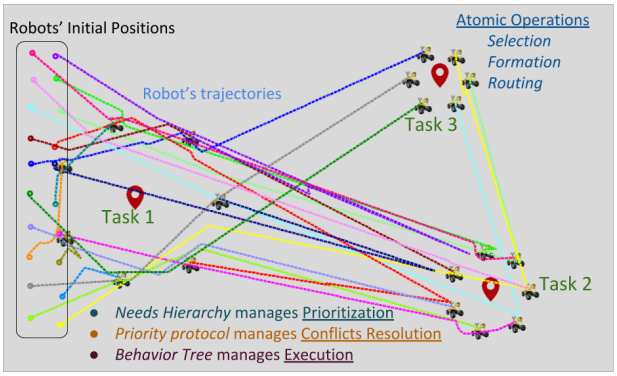


Fig. 1. Illustration of reactive multi-robot planning where robots move to Task 1 and Task 2 from their initial positions. During this execution, Task 3 is assigned and the robots react to this new task requirements.

In this paper, we propose a MRS cooperation concept, Self-Adaptive Swarm System (SASS)[1], that combines the parts of the perception, communication, planning, and execution to address this gap. See Fig. 1 for an illustration of the concept. Our preliminary work published as an extended abstract in [7] introduced the problem of multi-robots fulfilling dynamic tasks using state transitions represented through a Behavior Tree (BT) [8] and laid the foundations for the contributions made in this paper, which are outlined below.

- **Robot's Need Hierarchy:** To model an individual robot's $motivation$ and $needs$ in the negotiation process, we introduce the prioritization technique inspired by Maslow's hierarchy of human psychological needs [9] and solve the conflicts associated with the sub-tasks/elements in task planning. We define the *Robot's Need Hierarchy* at five different levels (see Fig. 2): safety needs; basic needs (energy [10], time constraints, etc.); capability (heterogeneity, hardware differences, communication, etc.); team cooperation (global utility, team performance, cooperation, and global behaviors); and self-upgrade (learning).
- **Negotiation-Agreement Mechanism**: We propose a *distributed Negotiation-Agreement* mechanism for selection (task assignment), formation (shape control), and routing (path planning) in MRS, represented through a BT [7] for automated planning of state-action sequences.
- **Atomic Operation:** We decompose the complex tasks into a series of simple sub-tasks through which we can recursively achieve those sub-tasks until we complete the high-level

[1]We use the term "Swarm" to denote the multi-agent context of the proposed multi-robot cooperation framework.

task. We provide several *Atomic Operations* for the swarm behavior: $Selection$, $Formation$, and $Routing$, which allows us to decompose a particular robot's action plans as flocking, pattern formation, and route planning under the same framework.

II. Related Work

Swarm robotics and swarm intelligence have been well studied in the literature [2]. Multi-robot modeling and planning algorithms are among those well-studied topics yet require task-specific or scenario-specific application limitations. Martinoli [11] presents the modeling technique based on rate equations, a promising method using temporal logic to specify and possibly prove emergent swarm behavior by Winfield et al. [12]. Soysal and Sahin [13] apply combinatorics, and linear algebra is deriving a model for an aggregation behavior of swarms. Some studies also applied control theory to model and analyzed multi-robot and swarm systems [14], [15]. Recently, Otte et al. [16] discussed various auction methods for multi-robot task allocation problem in communication-limited scenarios where the rate of message loss between the auctioneer and the bidders are uncertain.

From the multi-agent systems perspective, one of the earliest pioneering works, especially in the distributed artificial intelligence, includes [17], where the authors defined the Contract Net Protocol (CNP) for decentralized task allocation. Aknine [18] extended this idea to m manager agents and n contractor agents negotiation. A protocol for dynamic task assignment (DynCNET) has been developed by Weyns [19]. However, these methods rely on a central agent (such as an auctioneer or a contractor) to design the negotiation protocol and a supportive framework. They do not generally consider the changes in the agents' status, which restricts the direct applicability in real-world scenarios but only with adaptation.

Importantly, unlike distributed robotic systems, MRS emphasizes a large number of agents and promotes scalability, for instance, by using local communication [20], which plays a vital role in the whole system building various relationships between each robot to adapt to different environments and situations. In Tab. I, we present a comparison of SASS against common MRS frameworks that focus on task allocation and planning at different automation levels.

In multi-robot planning and control, several studies focus on navigating a robot from an initial state to a goal state [21], [22], [23]. So the individual robot's entire route planning is usually computed in high-dimensional joint configuration space. Since formations require robots to maintain stricter relative positions as they move through the environment, the flocking problem could be viewed as a sub-case of the formation control problem, requiring robots to move only minimal requirements for paths taken by specific robots [24]. The research question here is how to design suitable local control laws for each robot to complete the globally assigned tasks efficiently and cooperatively and how paths can be planned for permutation-invariant multi-robot formations [25]. Earlier, the solutions for such problems in flocking and formations are based on local interaction rules [26] or behavior-based approaches [27], [28]. More recent approaches focus on proving stability and convergence properties in multi-robot behaviors basing on control-theoretic principles [29], [30], [31].

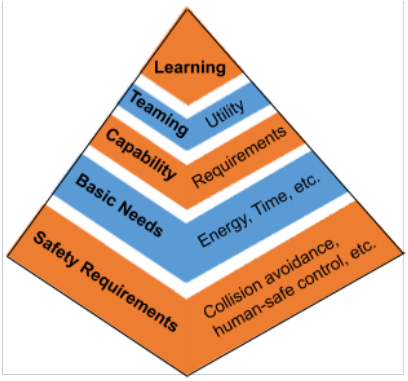


Fig. 2. Hierarchy of Robot Needs.

To summarize, we aim to develop more advanced MRS by seeking the fourth level of the autonomous system, which combines task decomposition, group formation, planning, and control [6]. More importantly, MRS should be able to adapt to the dynamic changes in the environment and task assignments.

III. Approach Overview

We design a simple scenario to implement SASS and distributed algorithms. In our scenarios, a group of swarm robots will cooperate to complete some tasks. Since the tasks are dynamically assigned, the robots need to change their plans and adapt to the new scenario to guarantee the group utility.

In our framework, we decompose the complex tasks into a series of sub-tasks and recursively achieve those sub-tasks until the entire task is completed. Accordingly, we can divide the task allocation and execution into three steps: selection, formation, and routing. This process can be illustrated as a Behavior Tree [8] that integrates the sense-think-act cycle [7]. The robots are assumed to have low-level motion control and sensor-based perception system for sensing and navigation.

First, the robots are partitioned into one or more groups to perform multiple tasks such as surveillance and patrolling. Then, they will compute the placement within a formation shape at each task (for simplicity, we assume a circular shape to circle the area of the task location, but other formation shapes can also be considered). Finally, the robots choose a suitable path to get to the goal point in that formation. When the new tasks are assigned, these robots need to split up to form new groups or merge into existing groups.

Each robot will first verify that there is a task assigned. If assigned, it will compute an appropriate plan according to its current state and needs. It will then communicate with other robots and perform the negotiation and agreement process until there are no conflicts. Finally, the robots will execute their plans. This process is continuously repeated as a loop in a behavior tree, which processes the flow from left to right.

The proposed framework is formalized using the tuple M=(S, A, δ, F). Here, S=(Pe, $Pl_1 \ldots_n$, $Ne_1 \ldots_n$,

TABLE I
COMPARISON OF TYPICAL MULTI-ROBOT SYSTEM FRAMEWORKS IN THE LITERATURE.

Approach	Ref.	HET	COOP	COM	NEGO	DEC	Learning	DIST	Scenario	Scale	Problems			
											Patrol	Coverage	Formation	Navi.
ABBA	[32]	√	√	√			√	√	dynamic	≤ 10				√
CHARON	[33]		√					√	static	≤ 10			√	
Token Passing	[34]		√	√				√	dynamic	≤ 10				√
Teamcore	[35]	√	√	√		√		√	dynamic	≤ 10				√
ALLIANCE	[36]	√	√	√	√			√	dynamic	≤ 20				√
ASyMTRe	[37]	√	√	√	√			√	dynamic	≤ 20				√
BITE	[38]	√	√	√		√		√	dynamic	≤ 10				√
DIST Layered	[39]		√	√	√			√	dynamic	≤ 10				√
STEAM	[40]		√	√		√		√	dynamic	≤ 100				√
Market-Based	[41]		√	√	√				dynamic	≤ 20				√
Hierarchy-Based	[42]		√	√		√			static	≤ 20		√	√	
SASS	Ours	√	√	√	√			√	dynamic	>100	√	√	√	√

HET: Heterogeneous, COOP: cooperation, COM: Communication, NEGO: Negotiation, DEC: Decision Making, DIST: Distributed Agents, Navi: Navigation.

$A\&E_1 \ldots_n$), where Pe represents perception, Pl represents plan, Ne represents negotiation, and $A\&E$ represents agreement. The subscript represent the number of iterations for each process, which is finite. $A = (A_1, A_2, \ldots, A_n)$ is a set of individual robot's actions (behaviors). The transition function that maps states (conditions) to actions (behaviors) is δ, defined as $S * A \rightarrow S$. F represent the accept state.

After the perception phase, a robot could have n plans $Pl = \{Pl_1, Pl_2, \ldots, Pl_n\}$, where each plan depend on each other requiring sequential execution. Therefore, each plan has negotiation $Ne = \{Ne_1, Ne_2, \ldots, Ne_n\}$ and agreement phases $A\&E = \{A\&E_1, A\&E_2, \ldots, A\&E_n\}$ separately.

To model an individual robot's motivation and needs in the negotiation process, we introduce the priority queue technique inspired by Maslow's hierarchy of human psychological needs [9]. We define a robot's need hierarchy at several levels, as shown in Fig 2. The lowest level represents the robot's safety needs. In all scenarios, a robot should first consider the safety issues (including human-safe operation) like avoiding conflicts or adversarial attacks. When the situation satisfies the robot's safety needs, the robot will consider its basic but vital needs, such as energy and time availability. Then, it will review its capabilities against the task requirements are subsuming the task priority considerations. In the fourth level, the robot finds rewards and utility costs (such as distance and communication) for cooperation within a group. The fifth level is reserved for self-upgrade that could potentially allow multi-robot learning strategies. For instance, after finishing the tasks, robots can upgrade their capabilities based on their experiences through the learning and evolution process. It is worth noting that robots' basic and safety needs are flipped compared to the human needs in Maslow's hierarchy.

The needs at the low level are the precondition of entering higher levels. If a robot cannot satisfy its low-level needs, it will change its behaviors accordingly. We design the priority queue to abstract this model and implement it in the negotiation process of our multi-robots planning framework. Also, the individual robot's current needs can dynamically change according to different scenarios. For example, in the normal state, the robot's behaviors and plans could maximize the task's requirement and minimize its energy. However, in cases of conflicting plans with other robots (e.g., potential collision with a neighboring robot), it will ensure that the safety needs are guaranteed. Similarly, if the robot runs out of battery, it also needs to find its basic needs first (e.g., recharging battery) before completing it.

IV. ALGORITHMS FOR SELF-ADAPTIVE SWARM SYSTEM

SASS considers the following modules: Perception; Communication; Planning; Negotiation; Agreement, and Execution. We will discuss them separately below.

1. Perception: Each robot uses various on-board (local) sensors for localization, mapping, and recognizing objects/obstacles in the environment.

2. Communication: The process of communication between robots includes broadcasting and reception of the robot's messages (state) to/from other robots [43]. The distributed communication in MRS can be regarded as a connected graph. Each robot keeps on communicating with its adjacent/neighbor's robots and exchanging data until they reach *Information Equilibrium*, which means that every group member has the same information for the entire group (see Alg. 1).

3. Planning: In the planning stage, we divide this process into three steps Selection, Formation, and Routing. These **Atomic Operations** are illustrated in Fig. 3 with an example planning problem. In each step, we also introduce a priority queue technique, which can help individual robot negotiate with other robots and get an agreement efficiently.

3A. Selection Planning: In our scenario, we assume that each task has an ID representing its priority (level-4 in Fig. 2), the minimum number of robots required to perform the task, and a task duration (timeout). Also individual robot has an ID and Energy (battery level) (level-2 in Fig. 2). We assume all robots are homogeneous and do not implement the third level of needs (capability). However, for heterogeneous robotic systems, we should consider this priority as well.

For selection planning, we minimize the sum of the group's energy costs to get a reasonable grouping/partitioning. Then, according to the tasks' priority and requirement, we distribute the diverse tasks to different groups, abstracting it to *The Linear Partition Problem*.

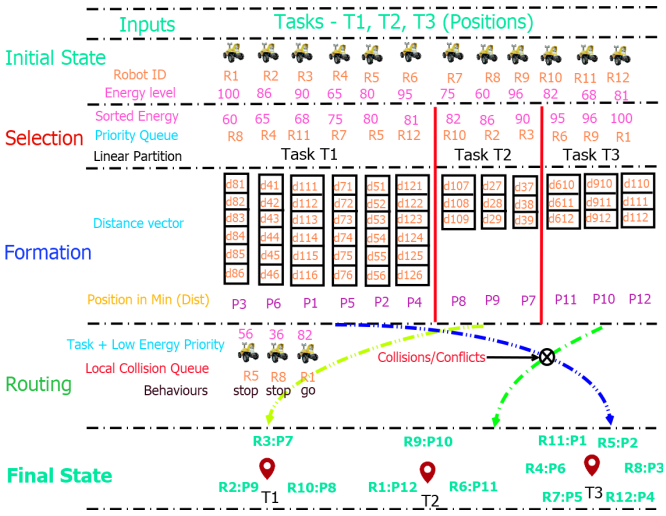


Fig. 3. Illustration of Task Decomposition through planning at Selection, Formation, and Routing phases with 12 robots and 3 tasks.

Algorithm 1: Distributed Communication Mechanism (DCM)

Input : Robot i data d_i, The number of robots in group n and Adjacent Robots Set r_a
Output : The Information Equilibrium data set D_i

initialization;
memory data set $D_i = \{\varnothing\}$
D_i.add(d_i)
while *length*(D_i) $\neq n$ **do**
  for *each* $j \in r_a$ **do**
    $D_i = D_i \cup D_j$
  end
end

Algorithm 2: Selection/Formation/Routing Negotiation

Input : Unsorted priority queue q_i, potential collision queue q_c,
Output : Sorted priority queue q_{si}

if *State* == *Selection/Formation/Routing* **then**
  q_n = hierarchical needs order queue;
end
if *State* == *Routing* **then**
  $tmp1$ = DCM(q_c);
  for *each* $item$ *in Union-Find*(tmp) **do**
    if $i \in item$ **then**
      $q_i = item$
    end
  end
end
if $q_n != NULL$ **then**
  q_{si} = Sort q_i with $q_n.first$;
  $tmp2$ = DCM(q_{si});
  while $Agreement(tmp2) == "conflict"$ **do**
    q_{si} = Sort q_i with $q_n.next$;
    $tmp2$ = DCM(q_{si});
  end
end

Algorithm 3: Selection/Formation/Routing Agreement

Input : Sorted priority queue list Q
Output : Execute the plan or negotiation again

initialization;
count = 1;
for *each* $item$ *in* Q **do**
  if $Q.first$!= $item$ **then**
    count++
  end
end
if *count* == *1* **then**
  return "end" and execute the corresponding "Atomic Operation";
else
  return "conflict".
end

Through this process, we can ensure that every group can achieve that specific task. Since each robot computing this process in a distributed manner, the local result at the individual robot could potentially have conflicts with the results of other robots depending on the uncertainty in the data it possesses and receives. For instance, one robot might have been assigned to different groups by different robots in the Selection phase. To prevent such a situation, we initiate a negotiation mechanism. We use the priority queue before this process and sort this priority queue with different priority levels, until we get a unique priority queue for all the robots, which can then be used to perform non-conflicting selection planning.

3B. Formation Planning: Each robot will make a formation plan according to the selected plan. Here, each robot knows which group (task) it belongs to. To simplify the formation models, we assume the robots need to create a regular polygon surrounding the task assignment location (group's center). The initial point will be located at the North position and follow the clockwise order to arrange the other point assignments within the formation by minimizing the system utility as mentioned before (we use distance as utility cost and energy level for prioritizing the robot's needs).

3C. Route Planning: Each robot computes the routes (path plan) using the local environment map of the sensor data and selects the shortest path getting to the goal point resulting from the Formation plan. Suppose each robot has two kinds of actions/behaviors that can be selected: one is moving at a constant speed(v), the other is stop. In case some robots have a conflict in the process of making the routing plan, they will create a priority queue with all conflicting robots $ID_{i...j}$ and share with the neighbors through local communication. Then, according to the Task and Energy priority of the robots, each robot in this queue will negotiate to decide on the corresponding actions on the robots that have conflicts. Until an agreement is reached, the priority queue is updated based on the needs hierarchy and solve the conflict.

4. Negotiation: Robots will compare the plans received from other group members with their own. For the Selection and Formation plan, the negotiation will be performed until the robots are assigned to only one group (in case of selection) or one position in the formation. Route planning involved creating a unique priority queue that avoids conflicts. Subsequently, each robot will reach an agreement on the priority queue and the corresponding plans. For example, in the selection and formation section, we use Low Energy (Low_E), which means the robots with lower battery levels get higher priority and similarly the High Energy $High_E$ law. We combine them with using Task-based priority queue (each task will have a specific priority in assigning the tasks), resulting in $T + High_E$ and $T + Low_E$. We also consider whether or not to consider the

priority needs of conflicts for route planning (conflict case and No conflict case). We present the algorithmic representation of the negotiation process in Alg. 2.

For instance, in the Formation plan negotiation, we use distances between its local position and the task's polygon points in the boundary of the formation shape. Each robot communicates this distance vector with other robots in the same group assigned to this specific task. Each robot will compute a matrix (Level 4 Team needs - Utility cost) representing each robot's distances to all the polygon points in the specific task. Then it will use the unique queue order to select the corresponding distance until all the group member gets the specific task goal point as long as the priority queues of Task, Energy, and Safety are satisfied. For example, the low energy robot will have a high priority choosing the point closest to it, thereby reducing energy consumption.

5. Agreement and Execution: If all the robots' plans do not have conflict after the negotiation phase, they will have a final agreement per process (Selection/Formation/Routing). The algorithm for implementing the agreement process is presented in Alg. 3. After the agreement, each process is executed as and when necessary.

V. Evaluation through Simulation Studies

To simulate our framework, we chose to use the "Common Open Research Emulator (CORE)" network simulator [44] since we are interested in implementing our algorithm in a network-based tool as CORE allows dynamic changes in the node/agent mobility and communication. We consider 20 robots in our simulations due to limitations in the CORE framework for an illustration of a single task assignment with 20 fully connected robots). In the evaluations, we consider only the lower three levels (Safety + Basic + Capability) in the robot needs hierarchy (see Fig. 2) for aiding rigorous analysis and validation of our cooperation framework. Composition of higher level needs will be investigated in our future work.

We suppose each robot has different battery levels in the initial state, and every moving step will cost 0.1% energy. Also, every communication round and non-moving status will cost 0.01% and 0.04% energy, respectively. To simplify the visualization of the utility of the framework, we do not consider any obstacles. We design two scenarios – one to simulate static task assignments (all tasks are added at the initial stage) and another to simulate a dynamic task assignment (a task can be added anytime during the process).

We consider four combinations of priority: $High_E$ (High Energy), Low_E^{Num} (Low Energy + Task Priority Order), $T + High_E$ (Task Priority + High Energy), and $T + Low_E$ (Task Priority + Low Energy). For example, if we adopt a priority queue with task + low energy combination, the scenario would be to first address the emergency task and maintain robots in the field as long as possible. We intend to compare the utility and behaviors of the individual robot and the system with different priority combinations.

To compare our approach with a state of the art method, we implemented the algorithm called Collision-Aware Task Allocation (CATA) in [23] in which the authors proposed a new method for addressing collision-aware task assignment problem using collision cone and auction-based bidding algorithms to negotiate the conflicts. Since our framework is distributed and does not have a central agent to manage the bidding process, we implemented the algorithm in [23], which provides the rewards for each robot to each task location. The rewards are converted to a Utility matrix and fed to the negotiation mechanism in our framework. Therefore, we term this method as $CATA_U$ (Collision-aware Task ASsignment + Utility Matrix). Here, a robot first calculates the task's utility based on [23], and it chooses the maximum utility task based on the low energy priority law. Therefore, we compare this method only with our Low_E priority law. The experiment demonstrations are available online[2].

A. Static Multi-Task Assignments

We conduct ten simulation trials for each priority case in a static task assignment scenario. In every priority case, we use the same ten different initial battery levels sampled from a Gaussian distribution with a mean of 90% and a standard deviation of 10%. Fig. 4(a) and 4(b) shows the distance matrix results of four priority laws of conflict frequency with and without conflict negotiation costs comparing the CATA approach. In this experiment, the $T + Low_E$ priority combination had the best performance compared with other cases. At the same time, Fig. 4(b) shows that every group almost cost one-third of the entire energy in the negotiation and agreement part of solving the conflicts. This means that more conflicts will lead to more negotiation rounds and corresponding energy consumption in communication. In Fig. 4(c), the effect of each priority law in the total system distance is also demonstrated for our finding that different needs and priorities at the individual agent level will lead to various global performances.

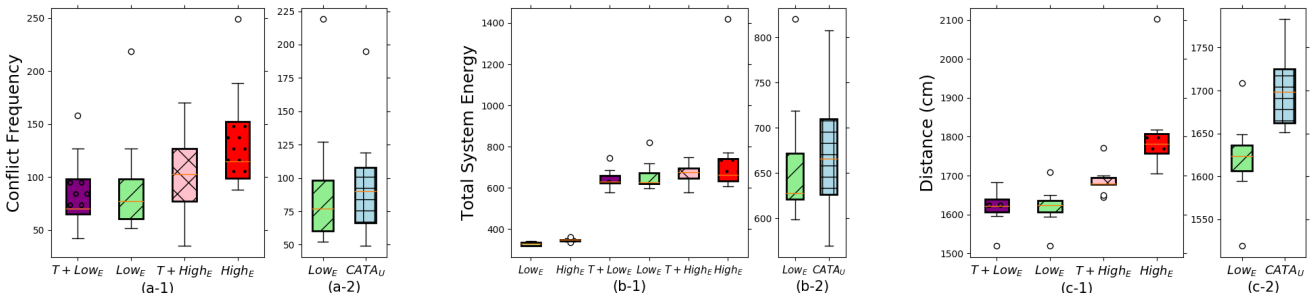
In the priority law of Low_E, our method had fewer conflicts than the $CATA_U$ method. We believe this is because prioritization of basic needs in our approach aims to avoid conflicts at the Formation planning stage. On the other hand, $CATA_U$ considers the conflicts only at the Route planning stage, which would leave more room for conflicts if the task polygon points are not efficiently assigned in the formation stage itself.

B. Scalability and Complexity

To verify SASS's scalability and the complexity of the MRS cooperation, we design four scales of robots' team implementing in different numbers of tasks: $R5 + T1$, $R10 + T2$, $R15 + T3$, $R20 + T4$. For example, $R10$ means ten robots in the system, and $T3$ means three different tasks are assigned.

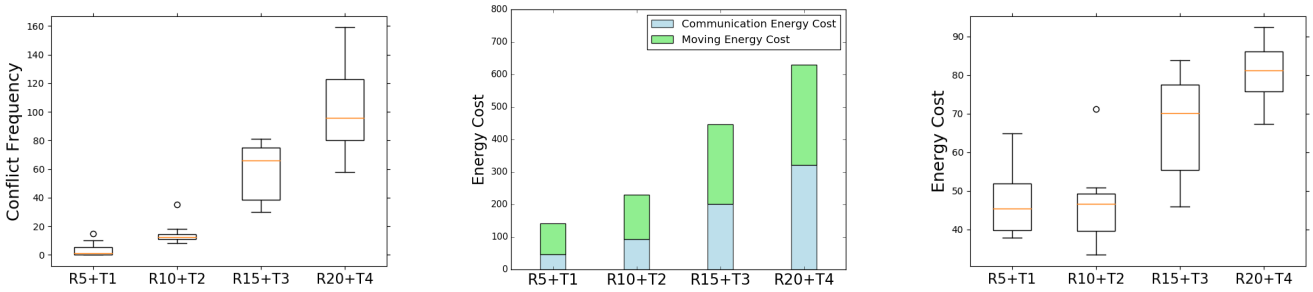
Through Fig. 5(a), we can notice that as the number of robots and the task complexity increase, the entire system's conflicts (conflict frequency) rise rapidly. The proportion of communication energy cost compared with the moving energy cost also increase with the increase in scale (see Fig. 5(b)).

[2]Experiment demonstration video is available at the website: https://hero.uga.edu/research/sass



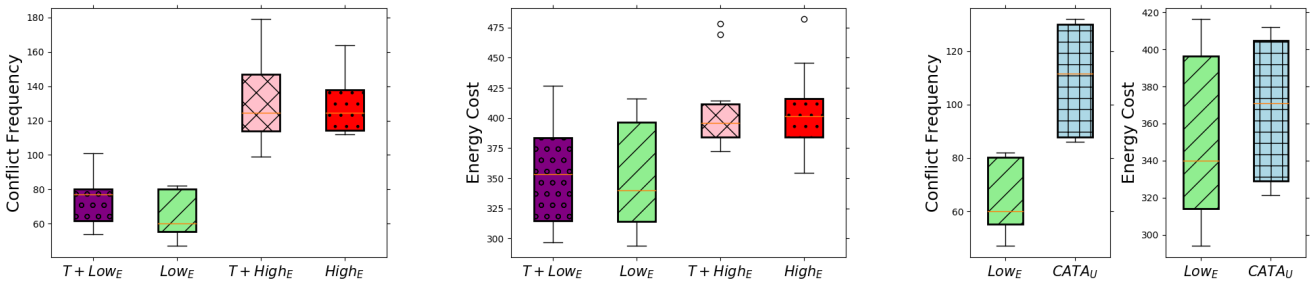
(a) Comparison of conflict frequency. (b) Comparison of total energy costs. (c) Comparison of total travel distance.

Fig. 4. Experiments on static task assignments with 20 robots and 3 tasks.



(a) Comparison of conflict frequency. (b) Comparison of Comm. & moving energy costs. (c) Comparison of per task Comm. energy costs.

Fig. 5. Experiments on Scalability and Complexity in Static Task Assignments. Here, R15 means 15 Robots and T4 means 4 Tasks for example.



(a) Comparison of conflict frequency. (b) Comparison of Comm. energy costs. (c) Comparison of Low_E and CATA_U.

Fig. 6. Experiments on the Impact of Different Priority Laws in Static Task Assignments.

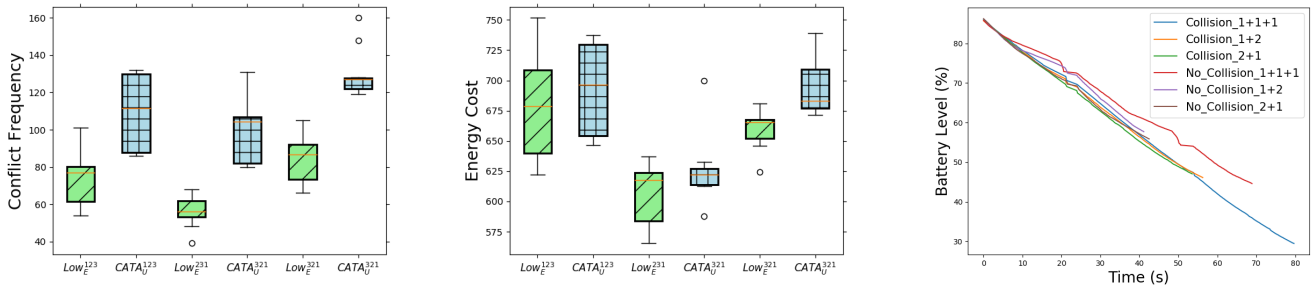
This points to the fact that the whole system spends more energy and time negotiating and cooperatively converging to a specific agreement. So considering the average communication energy cost per task (see Fig. 5(c)), if the task complexity is higher in some specific scenarios or the environment is more unstructured and unpredictable, an individual agent will spend more energy and time in communication to fulfill the tasks.

From another perspective, the fully distributed communication graph is inefficient in a large scale system of coordinating robots, especially in the swarm robots. Therefore, designing a proper communication architecture to adapt to a particular scale of agents' group and complexity scenarios is also an important and challenging problem that should be investigated further, which is an avenue for future work.

C. Impact of Different Priority Laws at Different Levels

If an individual agent has different needs or motivations, it might present various conduct, leading to the entire system displaying different performance. To verify this hypothesis, we use 20 robots having the same initial battery levels based on four different priority laws comparing with the distance matrix and CATA as we discussed above. We conduct ten trials with different initial battery levels sampled from a Gaussian distribution with a mean 90% and standard deviation 30% to represent heterogeneity in the energy capacities of the robots.

Fig. 6(a) shows that implementing different priority laws causes the whole system to exhibit different energy use. As



(a) Comparison of conflict frequency (b) Comparison of total energy cost. (c) Comparison of total battery level.

Fig. 7. Experiments on the impact of different priority laws and dynamic task assignments with 20 robots and 3 tasks. (Left and Center) Comparison of conflict frequency and energy cost based on the different task's priority in statics task assignments. E.g., In Low_E^{123}, the priority of T1 $>$ T2 $>$ T3. (Right) Comparison of the total battery level of the system measured in points combining all robot energy percentage levels in dynamic task assignments.

TABLE II
ENERGY LEVEL COMPARISON IN DYNAMIC TASK ASSIGNMENTS

Tasks Style	Priority	conflict	Max	Min	Mean
1+1+1	High$_E$	$\surd$	69.42	55.52	64.18
1+1+1	Low$_E$	$\surd$	63.09	45.70	56.00
1+1+1	T+High$_E$	$\surd$	70.04	56.75	63.24
1+1+1	T+Low$_E$	$\surd$	63.25	49.18	56.62
1+2	T+Low$_E$	$\surd$	45.97	31.78	39.60
2+1	T+Low$_E$	$\surd$	44.69	31.66	39.07
1+1+1	T+Low$_E$	-	49.88	29.48	41.20
1+2	T+Low$_E$	-	32.70	23.14	28.50
2+1	T+Low$_E$	-	38.53	24.72	30.36

we can see, the system has better performance in $T + Low_E$ and Low_E priority laws than $T + High_E$, $High_E$ priority laws. This is because the former two lead to fewer conflicts than the latter two. We can confirm these observations through Fig. 6(b) and 6(c).

We also shuffle the task priorities (e.g., the priority of T1 $>$ T2 $>$ T3 in the superscript 123) to simulate the variation in the third level of the needs hierarchy (Capability/Requirements) and the results are shown in Fig. 7(a) and Fig. 7(b). We can see that our approach's distance matrix always has better performance than the CATA under the same conditions in terms of conflict resolution and system energy.

In practical applications, an intelligent agent might dynamically change its needs or motivation according to the situations, especially in the adversarial or unpredictable environment. This may lead to chaos in the system resulting in higher energy costs and loss of system utility. In the MRS design, letting the individual agents select suitable laws for coordination while guaranteeing the optimal system utility is also an exciting and challenging avenue for future work.

D. Dynamic Multi-Task Assignments

In the dynamic multi-tasks scenario, we design three kinds of dynamic tasks. One is three tasks added sequentially after every task is completed $(1+1+1)$. The rest of the two cases are two tasks appearing at the start $(2+1)$ and the end $(1+2)$ in the entire process. Also, we combine this with a conflict or no conflict cases.

The first experiment we consider using four different priority laws and 20 robots implementing the $1+1+1$ scenario (see Table II). Here, the initial energy level and position for each priority laws were set the same. We can observe that the $T + Low_E$ and Low_E combinations achieved the best system utility as in the static assignment cases.

In the second experiment, we evaluate the impact of the conflict negotiation process by comparing the energy costs of three different tasks with and without considering conflicts. In Fig. 7(c), we notice that the negotiation cost occupies a large part in the entire system costs (hence, the higher battery level consumed when collisions are considered). We can also observe that the difference between each combination's negotiation energy cost reflects the environment's unstructured level, which means the difference will increase in the more chaotic and dynamic considerations.

VI. CONCLUSION

Our work introduces a novel SASS framework for cooperation heterogeneous multi-robot systems for dynamic task assignments and automated planning. It combines robot perception, communication, planning, and execution in MRS, which considers individual robot's needs and action plans and emphasizes the complex relationships created through communication between the robots. Specifically, we proposed *Robot's Needs Hierarchy* to model the robot's motivation and offer a priority queue in a distributed *Negotiation-Agreement Mechanism* avoiding plan conflicts effectively. Then, we provide several *Atomic Operations* to decompose the complex tasks into a series of simple sub-tasks. The proposed solution is evaluated through extensive simulations under different static and dynamic task scenarios. The experimental analysis showed that the needs-based cooperation mechanism outperformed state-of-the-art methods in maximizing global team utility and reducing conflicts in planning and negotiation.

SASS leaves room for many future improvements. For instance, we plan to optimize the communication architecture and add decision learning levels into individual robot's hierarchical needs completing our robot's needs model. This

improvement will cause the individual robots to upgrade by itself based on the learned experiences and lead to the self-evolution of the whole system.

REFERENCES

[1] S. A. Levin, "Ecosystems and the biosphere as complex adaptive systems," *Ecosystems*, vol. 1, no. 5, pp. 431–436, 1998.

[2] H. Hamann and H. Wörn, "A framework of space–time continuous models for algorithm design in swarm robotics," *Swarm Intelligence*, vol. 2, no. 2-4, pp. 209–239, 2008.

[3] L. E. Parker, "Multiple mobile robot systems," *Springer Handbook of Robotics*, pp. 921–941, 2008.

[4] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield *et al.*, "The grand challenges of science robotics," *Science Robotics*, vol. 3, no. 14, 2018.

[5] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams with complex constraints," *Autonomous Robots*, vol. 32, no. 4, pp. 385–403, 2012.

[6] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–31, 2019.

[7] Q. Yang, Z. Luo, W. Song, and R. Parasuraman, "Self-Reactive Planning of Multi-Robots with Dynamic Task Assignments," in *IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2019.

[8] M. Colledanchise and P. Ögren, *Behavior Trees in Robotics and Al: An Introduction*. CRC Press, 2018.

[9] A. H. Maslow, "A theory of human motivation." *Psychological review*, vol. 50, no. 4, p. 370, 1943.

[10] R. Parasuraman, P. Pagala, K. Kershaw, and M. Ferre, "Energy management module for mobile robots in hostile environments," in *Conference Towards Autonomous Robotic Systems*. Springer, 2012, pp. 430–431.

[11] A. Martinoli, "Swarm intelligence in autonomous collective robotics: From tools to the analysis and synthesis of distributed control strategies," Ph.D. dissertation, Citeseer, 1999.

[12] A. F. Winfield, J. Sa, M.-C. Fernández-Gago, C. Dixon, and M. Fisher, "On formal specification of emergent behaviours in swarm robotic systems," *International journal of advanced robotic systems*, vol. 2, no. 4, p. 39, 2005.

[13] O. Soysal and E. Şahin, "A macroscopic model for self-organized aggregation in swarm robotic systems," in *International Workshop on Swarm Robotics*. Springer, 2006, pp. 27–42.

[14] V. Gazi and K. M. Passino, "Stability analysis of swarms," *IEEE transactions on automatic control*, vol. 48, no. 4, pp. 692–697, 2003.

[15] J. T. Feddema, C. Lewis, and D. A. Schoenwald, "Decentralized control of cooperative robotic vehicles: theory and application," *IEEE Transactions on robotics and automation*, vol. 18, no. 5, pp. 852–864, 2002.

[16] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Autonomous Robots*, pp. 1–38, 2019.

[17] R. G. Smith and R. Davis, "Frameworks for cooperation in distributed problem solving," *IEEE Transactions on systems, man, and cybernetics*, vol. 11, no. 1, pp. 61–70, 1981.

[18] S. Aknine, S. Pinson, and M. F. Shakun, "An extended multi-agent negotiation protocol," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 1, pp. 5–45, 2004.

[19] D. Weyns, N. Boucké, T. Holvoet, and B. Demarsin, "Dyncnet: A protocol for dynamic task assignment in multiagent systems," in *First International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2007)*. IEEE, 2007, pp. 281–284.

[20] H. Hamann, *Swarm robotics: A formal approach*. Springer, 2018.

[21] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artificial Intelligence*, vol. 219, pp. 1–24, 2015.

[22] N. Ayanian and V. Kumar, "Abstractions and controllers for groups of robots in environments with obstacles," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 3537–3542.

[23] F. Wu, V. S. Varadharajan, and G. Beltrame, "Collision-aware task assignment for multi-robot systems," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 30–36.

[24] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on automatic control*, vol. 51, no. 3, pp. 401–420, 2006.

[25] S. Kloder and S. Hutchinson, "Path planning for permutation-invariant multirobot formations," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 650–665, 2006.

[26] C. W. Reynolds, *Flocks, herds and schools: A distributed behavioral model*. ACM, 1987, vol. 21, no. 4.

[27] M. J. Mataric, "Designing emergent behaviors: From local interactions to collective intelligence," in *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, 1993, pp. 432–441.

[28] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE transactions on robotics and automation*, vol. 14, no. 6, pp. 926–939, 1998.

[29] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2012.

[30] R. M. Murray, "Recent research in cooperative control of multivehicle systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 571–583, 2007.

[31] R. Parasuraman, J. Kim, S. Luo, and B.-C. Min, "Multipoint rendezvous in multirobot systems," *IEEE transactions on cybernetics*, 2018.

[32] D. Jung and A. Zelinsky, "An architecture for distributed cooperative planning in a behaviour-based multi-robot system," *Robotics and Autonomous Systems*, vol. 26, no. 2-3, pp. 149–174, 1999.

[33] R. Fierro, A. Das, J. Spletzer, J. Esposito, V. Kumar, J. P. Ostrowski, G. Pappas, C. J. Taylor, Y. Hur, R. Alur *et al.*, "A framework and architecture for multi-robot coordination," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 977–995, 2002.

[34] A. Farinelli, L. Iocchi, D. Nardi, and V. A. Ziparo, "Assignment of dynamically perceived tasks by token passing in multirobot systems," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1271–1288, 2006.

[35] M. Tambe, D. V. Pynadath, N. Chauvat, A. Das, and G. A. Kaminka, "Adaptive agent integration architectures for heterogeneous team members," in *Proceedings Fourth International Conference on MultiAgent Systems*. IEEE, 2000, pp. 301–308.

[36] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE transactions on robotics and automation*, vol. 14, no. 2, pp. 220–240, 1998.

[37] L. E. Parker and F. Tang, "Building multirobot coalitions through automated task solution synthesis," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1289–1305, 2006.

[38] G. A. Kaminka and I. Frenkel, "Flexible teamwork in behavior-based robots," in *Proceedings Of The National Conference On Artificial Intelligence*, vol. 20, no. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 108.

[39] D. Goldberg, V. Cicirello, M. B. Dias, R. Simmons, S. Smith, T. Smith, and A. Stentz, "A distributed layered architecture for mobile robot coordination: Application to space exploration," in *The 3rd International NASA Workshop on Planning and Scheduling for Space*, 2002.

[40] M. Tambe, "Towards flexible teamwork," *Journal of artificial intelligence research*, vol. 7, pp. 83–124, 1997.

[41] M. B. Dias and A. Stentz, "A free market architecture for distributed control of a multirobot system," 2000.

[42] H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. S. Kumar, N. Ayanian, and S. Koenig, "Overview: A hierarchical framework for plan generation and execution in multirobot systems," *IEEE Intelligent Systems*, vol. 32, no. 6, pp. 6–12, 2017.

[43] D. Tardioli, R. Parasuraman, and P. Ögren, "Pound: A multi-master ros node for reducing delay and jitter in wireless multi-robot networks," *Robotics and Autonomous Systems*, vol. 111, pp. 73–87, 2019.

[44] J. Ahrenholz, "Comparison of core network emulation platforms," in *2010-Milcom 2010 Military Communications Conference*. IEEE, 2010, pp. 166–171.