A Game-theoretic Utility Network for Cooperative Multi-Agent Decisions in Adversarial Environments

Qin Yang Ramviyas Parasuraman

***Abstract*—Underlying relationships among multi-agent systems (MAS) in hazardous scenarios can be represented as Game-theoretic models. This paper proposes a new network-based model called Game-theoretic Utility Tree (GUT), which decomposes high-level strategies into executable low-level actions for cooperative MAS decisions in adversarial environments. It combines a new payoff measure based on agent needs for real-time strategy games. We demonstrated the applicability of the GUT using the Robotarium platform, which is a simulator-hardware testbed for verifying multi-robot system algorithms. The performances verified the effectiveness of the GUT in the real robot application and validated that the GUT could effectively organize MAS cooperation strategies, helping the group with fewer advantages achieve higher performance.**

I. Introduction

Natural systems have been the key inspirations in the design, study, and analysis of Multi-Agent Systems (MAS) [1]. *Distributed Intelligence* refers to systems of entities working together to reason, plan, solve problems, think abstractly, comprehend ideas and language, and learn [2]. Especially for cooperative MAS, the individual is aware of other group members, and actively shares and integrates its needs, goals, actions, plans, and strategies to achieve a common goal and benefit the entire group [3]. It can maximize global system utility and guarantee sustainable development for each group member [4].

Systems with a wide variety of agent heterogeneity and communication abilities can be studied, and collaborative and adversarial issues also can be combined in a real-time situation [5]. Considering working in adversarial environments, opponents can prevent MAS from achieving global and local tasks, even impair individual or system necessary capabilities or normal functions [6]. Combining multi-agent cooperative decision-making and robotics disciplines, researchers developed the *Adversarial Robotics* focusing on autonomous agents operating in adversarial environments. [7], [8]. From the robot's[1] needs [9], [10] and motivations perspective, we can classify an **Adversary** into two general categories: **Intentional** (such as enemy or intelligent opponent agent, which consciously and actively impairs the MAS needs and capabilities) and **Unintentional** (like obstacles and weather, which unaware and passively threaten MAS abilities) adversary.

MAS research domains focus on solving path planning problems for avoiding static or dynamical obstacles [7] and formation control [11], [8] from the unintentional adversary perspective. For intentional adversaries, the "pursuit domain" [12], [13] primarily deals with how to guide one or a group of pursuers to catch one or a group of moving evaders [14], [15]. Foundations for normal-form team games and extensive-form adversarial team games are provided in [16] and [17], respectively. Nevertheless, it is more realistic and practical for MAS to organize more complex relationships and behaviors, achieving given tasks with higher success probability and lower costs in adversarial environments.

Fig. 1. Illustration of the *Explore Game* scenario where the Aliens (opponent agents - peer adversaries) block the paths to a target of the Explorers (protagonist agents).

This paper proposes a new hierarchical network model called *Game-theoretic Utility Tree (GUT)* to achieve MAS cooperative decision-making in adversarial environments. *GUT* consists of *Game-theoretic Utility Computation Units* (Fig. 2) distributed in multiple levels by decomposing strategies, thereby significantly lowering the game-theoretic operations in strategy space dimension. It combines the core principles of *Game Theory* [18] and *Utility Theory* [19], [20]. Further more, we present a game of Explorers vs. Aliens (referred as *"Explore Game"* - Fig. 1) to evaluate the MAS performance from the perspective of balancing the success probability of achieving tasks and system costs by organizing involved individuals' relationships and suitable groups' strategies in adversarial environments.

We demonstrate the effectiveness of the GUT against the *random* and *greedy* approaches in the *Explore Game* through the Robotarium [21] hardware-simulator multiagent testbed. The results indicated that *GUT* could organize more complex relationships among MAS cooperation. It helps the group achieving challenging tasks with lower costs and higher winning probability.

The proposed approach can be applied to other Real-Time

* 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*.

[1]Here, we use the terms agent and robot interchangeably.

Strategy (RTS) tasks, which involve agents decomposing the high-level strategies into primitive actions or group atomic operations [9] in the specific mission, such as robot soccer, multi-robot urban search and rescue (USAR) missions, etc.

II. Background and Preliminaries

This section provides a brief background to the Game theory principles used in this paper. Then, we present the agent needs hierarchy based on which the payoff utilities in our game-theoretic approach are designed. Further, we define an adversary agent based on the agent's needs expectations and discuss the *Explore Game* problem.

A. Game Theory Basics

Game Theory is the science of strategy, which provides a theoretical framework to conceive social situations among competing players and produce optimal decision-making of independent and competing actors in a strategic setting [18].

In a non-cooperative game, players compete individually and try to raise their profits alone. Especially in the zero-sum games, their total value is constant and will not decrease or increase, which means that one player's profit is associated with another loss. In contrast, if different players form several coalitions trying to take advantage of their coalition, that game will be cooperative [22]. In this paper, we focus on non-cooperative games, where the ally and enemy agent teams do not cooperate. But, the ally agents will cooperate within the team to decide on a common team strategy by sharing their perception data.

If a player chooses to take one action with probability 100%, then the player is playing a pure strategy. For the game's solutions, if players adopt a *Pure Strategy*, it will provide maximum profit or the best outcome. Therefore, it is regarded as the best strategy for every player of the game. On the other hand, in a *Mixed Strategy*, players execute different strategies with the possible outcome through a probability distribution over several actions. In the game theory, the *Normal Form* describes a game through a matrix, where each player has a set of (mixed) strategies. They select a strategy and play their selections simultaneously. Furthermore, the selection of strategies results in payoff or utility for each player, and its goal in a game is to maximize utility.

Furthermore, *Nash Existence Theorem* is a theoretical framework, which guarantees the existence of a set of mixed strategies for a finite, non-cooperative game of two or more players in which no player can improve its payoff by unilaterally changing strategy. It guarantees that every game has at least one Nash equilibrium [23], which means that every finite game has a *Pure Strategy Nash Equilibrium* or a *Mixed Strategy Nash Equilibrium*. Moreover, in any normal-form game with constant number of strategies per player, an *ϵ-approximate Nash Equilibrium* can be computed in time $O(n^{log\ n/\epsilon^2})$, where n is the description size of the game [24].

B. Agent Needs Hierarchy

In *Agent Needs Hierarchy* [9], the abstract needs of an agent for a given task are prioritized and distributed into multiple levels, each of them preconditioned on their lower levels. At each level, we express the needs as an expectation over its distribution of the factors/features corresponding to that level.

Here, we define five different levels of agent needs similar to Maslow's human needs pyramid. The lowest (first) level is the safety features of the agent (e.g., features such as collision detection, fault detection, etc., that assure safety to the agent, human, and other friendly agents in the environment). The safety needs (Eq. (1)) can be calculated through its safety feature's value and corresponding safety feature's probability based on the current state of the agent. After satisfying safety needs, the agent considers its basic needs (Eq. (2)), which includes features such as energy levels, data communication levels that help maintain the basic operations of that agent. Only after fitting the safety and basic needs, an agent can consider its capability needs (Eq. (3)), which are composed of features such as its health level, computing (e.g., storage, performance), physical functionalities (e.g., resources, manipulation), etc.

At the next higher level, the agent can identify its teaming needs (Eq. (4)) that accounts the contributions of this agent to its team through several factors (e.g., heterogeneity, trust [25], actions) that the team needs so that they can form a reliable and robust team to successfully perform a given mission.

Ultimately, at the highest level, the agent learns some skills/features to improve its capabilities and performance in achieving a given task. For instance, an agent may use RL to learn its policy features (e.g., Q table or reward function) using which it can execute appropriate actions based on the current state. Such learning features are accounted into its learning needs expectation (Eq. (5)). The expectation of agent needs at each level are given below:

$$Safety\ \ Needs: N_{s_j} = \sum_{i=1}^{s_j} S_i \cdot \mathbb{P}(S_i|X_j, T); \quad (1)$$

$$Basic\ \ Needs: N_{b_j} = \sum_{i=1}^{b_j} B_i \cdot \mathbb{P}(B_i|X_j, T, N_{s_j}); \quad (2)$$

$$Capability\ Needs: N_{c_j} = \sum_{i=1}^{c_j} C_i \cdot \mathbb{P}(C_i|X_j, T, N_{b_j}); \quad (3)$$

$$Teaming\ \ Needs: N_{t_j} = \sum_{i=1}^{t_j} T_i \cdot \mathbb{P}(T_i|X_j, T, N_{c_j}); \quad (4)$$

$$Learning\ \ Needs: N_{l_j} = \sum_{i=1}^{l_j} L_i \cdot \mathbb{P}(L_i|X_j, T, N_{t_j}); \quad (5)$$

Here, $X_j = \{P_j, C_j\} \in \Psi$ is the combined state of the agent j with P_j being the perceived information by that agent and C_j representing the communicated data from other agents. T is the assigned task (goal or objective). S_i, B_i, C_i, T_i, and L_i are the utility values of corresponding feature/factor i in the corresponding levels - Safety, Basic, Capability, Teaming, and Learning, respectively. s_j, b_j, c_j, t_j, and l_j are the sizes of agent j's feature space on the respective levels of needs.

The collective need of an agent j is expressed as the union of needs at all the levels in the needs hierarchy as in Eq. (6).

$$N_j = N_{s_j} \cup N_{b_j} \cup N_{c_j} \cup N_{t_j} \cup N_{l_j} \tag{6}$$

The set of agent needs in a multiagent system can be regarded as a kind of motivation or requirements for cooperation between agents to achieve a specific group-level task.

C. Adversarial Agent Definition

A friendly (ally) agent can contribute to the team, decreasing the individual needs of the team members, while an adversary can harm the team, increasing the overall needs of every team member. Based on this concept, we define an agent R_1 as adversary or friendly with respect to an agent R_2 as follows. For a certain state $\psi \in \Psi$, the agent R_1 is fulfilling a task T. Supposing the current needs of R_1 is $N_{R_1}(\psi, T)$. Considering another agent R_2 entering the R_1's observation space, the needs of R_1 can be represented as $N_{R_1}(\psi \cup R_2, T)$ under the presence of the agent R_2. The following equations define the relationship between R_1 and R_2:

$$N_{R_1}(\psi \cup R_2, T) - N_{R_1}(\psi, T) > 0; \quad (Adversary) \tag{7}$$
$$N_{R_1}(\psi \cup R_2, T) - N_{R_1}(\psi, T) < 0; \quad (Friendly) \tag{8}$$
$$N_{R_1}(\psi \cup R_2, T) - N_{R_1}(\psi, T) = 0. \quad (Neutral) \tag{9}$$

Definition 1 (Adversary): If the needs of R_1 increase when R_2 is present, then R_1 regards R_2 as an Adversary (Eq. (7)).

Definition 2 (Friendly): If the needs of R_1 decrease when R_2 is present, then R_1 sees R_2 as a friendly agent (Eq. (8)).

Definition 3 (Neutral): If the needs of R_1 do not change because of R_2's presence, then R_2 is neutral to R_1 (Eq. (9)).

Note, an obstacle is still an (unintentional) adversary as per this definition, since obstacles will increase the needs of an agent in terms of using more energy to avoid collision risk.

D. Explore Game

To simplify theoretical analysis and numerical calculations, we consider an exemplar problem domain called *Explore Game*, which is described below. In *Explore Game*, α number of agents (called **Explorers** hereafter) are performing a task T, which is to explore an environment and collect rewards by reaching treasure locations. Supposing there are β number of (intentional) adversaries (called **Aliens** hereafter). Explorer can choose a strategy s_e from its strategy space S_e and aliens can choose a strategy s_a from its strategy space S_a. We assume both these strategy spaces are known to the Explorer agents. We also assume that the Aliens do not have a cooperation strategy, and each Alien acts independently on its own.

Let C_i represent the system costs of explorer i to perform this task and W denote the success probability (win rate) of the explorer team under the presence of alien(s). We model this as a bi-objective optimization problem (Eq. (10)) of finding an optimal collective team strategy for the explorers $s_e^* \in S_e$ under the premise of maximizing success W (against aliens) while minimizing costs C using the collective needs of the explorers $N_e = \sum_{i=1}^{\alpha} N_i$.

$$s_e^* = \arg\max_{s_e \in S_e} [W(s_e|T, S_a, N_e) - \sum_{i=1}^{\alpha} C_i(s_i|T, s_e, N_i)] \tag{10}$$

Without an adversary, the problem shrinks to a typical exploration problem (optimizing C alone) [26], and without a task T, the problem shrinks to a typical non-cooperative zero-sum game problem (optimizing W alone) [18].

The proposed approach can be applied to other Real-Time Strategy (RTS) games, such as air combat and StarCarft, and cooperative multiagent/robot mission, like urban search and rescue (USAR) [10], pursuit-evasion game [27] and robot soccer [28]. These domains involve both ally agents and opponent agents (intentional adversaries), and the nature of the strategies the agents or the team can take can allow dissolving the high-level group strategies into primitive or atomic actions.

For instance, in Star Craft [29], the strategies a player can make can be composed of the following primitives: What to do? (e.g., Build, Move, Attack, etc.); Who to perform? (which player to execute this action or which opponent to attach, etc.); Where? (physical location or point of interest); When? (immediately or delayed or how long); etc.

Similarly, in robot soccer competitions [28], a player can choose a composition of low-level sub-strategies such as an action (what to do? - kicking, passing, or shooting a ball, etc.) combined with where or who (ball location or player destination, for examples). As we can see, a final strategy the team can take is a composition of these primitives. Using GUT, we can solve for atomic actions at the level of primitives and hierarchically merge them to find the best high-level strategies for the ally team against enemy agents.

III. Approach

Fig. 2 outlines the structure of the *Game-theoretic Utility Tree (GUT)* and its computation units distributed in each level. First, the *game-theoretic module* (Fig. 2 (a)) calculates the nash equilibrium based on the utility values $(u_{11}, ..., u_{nm})$ of corresponding situations, $(p_1, ..., p_{nm})$ presenting the probability of each situation. Then, through the *conditional probability(CP) module* (Fig. 2 (b)), the CP of each situation can be described as $(p_{i1}, ..., p_{inm})$, where $p_{inm} = (p_{nm}|p_{i-1}),\ i, n, m \in Z^+$. Here, p_{i-1} and S_i present the probability of previous situation and current Game-theoretic state; s_a, s_b and n, m represent their strategy space and size on both sides, respectively. In this section, we explain the decision-making process in GUT and describe the specific implementation in "Explore Game".

A. GUT-based Decision-Making

For intentional adversaries, agents first decompose the specific goal into several independent subtasks based on the same category of individual low-level behaviors or atomic operations (basic group strategies). Then, through calculating various Nash equilibrium based on different situation utility values in each level's *Game-theoretic Utility Computation*

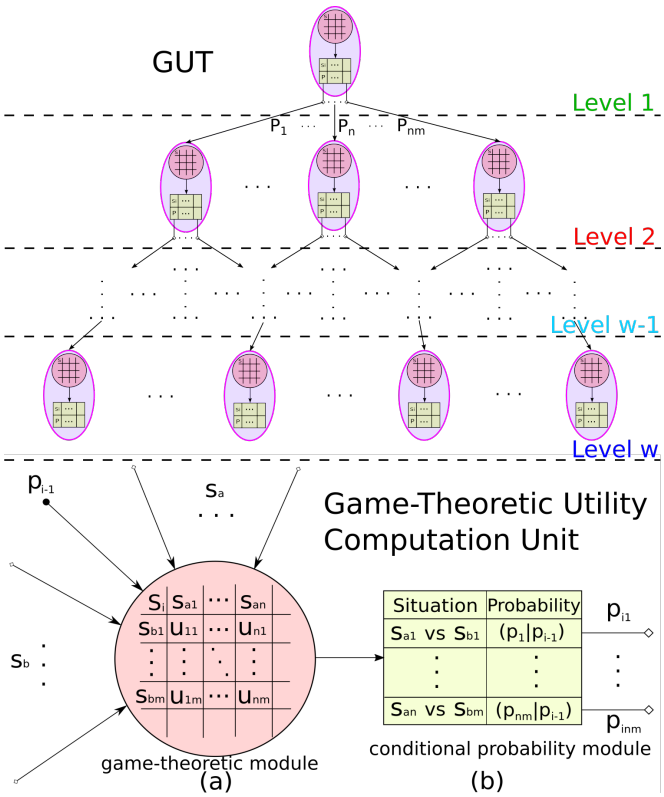


Fig. 2. General Individual Agent's *GUT*

Units, agents can get optimal or sub-optimal strategy sets tackling the current status according to *Nash Existence Theorem*. So *GUT* also can be regarded as a *Task-Oriented Decision Tree*. We formalize it as Theorem 1.

Theorem 1 (GUT Decision): Let A and B represent the groups of Explorers and Aliens. The simultaneous normal-form game representing the non-cooperative game between explorers and aliens is a structure G=$\langle\{A, B\}, \{S_e, S_a\}, \{N_{t_A}, N_{c_B}\}\rangle$. Supposing the GUT at the explorer group has w levels. $G_i\langle\{A,B\},\{S_e, S_a\},N_{t_{A_i}}\rangle$, i $\in$ w (Fig. 2.GUT) describes corresponding zero-sum game in each level. Then, A has at least one dominant strategy series (s_1, s_2, ... , s_w) in GUT.

B. Complexity Analysis

Like *the master theorem* [30], supposing each sub-game has the same size (strategies space), the GUT can be described as the running time of an approach that recursively divides a game $G(\xi)$ of size ξ into a sub-games, each of size ξ/b, $a, b \in Z^+$. If $G(\xi)$ is the one-level game, the complexity obeys [24]. Then $G(\xi)$ has the following asymptotic bounds:

$$\xi^{log_b^a} \leq G(\xi) \leq \xi^{log\ \xi/\epsilon^2},\ \epsilon \in (0,1). \tag{11}$$

It runs in $O(log_b^{\xi})$ time on searching the specific strategy set, showing the scalability in the strategies space (game size). The scalability in terms of the number of agents depends on the particular communication graph in information sharing.

The *Utility Function* design is critical to determine whether or not an individual can calculate reasonable tactics. However, to simplify the computation process, we can adopt the winning probability (W), basic needs (energy cost - $E(e)$), and safety needs (HP (health power) cost - $E(hp)$) representing the expected utility values for corresponding levels.

In the whole process, explorers present a kind of global behaviors performing *Collective Rationality* and caring about *Group interest*. In contrast, aliens show *Self-interest* and do not cooperate for relative definitions). For explorers, their *Teaming Needs* (expected utilities) is under the premise of maximizing the chance of finding the treasure to minimize HP cost based on fitting their low-level needs, such as safety and basic needs.

IV. Experiments

To demonstrate the GUT on the multi-robot applications, we implement our method in the Robotarium [21] platform, a remote-accessible multi-robot experiment testbed that supports controlling up to 20 robots simultaneously on a 3.2m $\times$ 2.0m large rectangular area. Each robot has the dimensions 0.11 m $\times$ 0.1 m $\times$ 0.07 m in the testbed.

In the Robotarium experiments, we consider three different propositions between the number of explorers and aliens in the *Explore Game* domain. They are one *explorer vs. one alien* (Fig. 3), *one explorer vs. two aliens* (Fig. 4), and *four explorers vs. three aliens* (Fig. 5). To highlight the difference between each experiment, we do not consider any obstacles in the first two scenarios. The third scenario involves (simulated) obstacle regions and two different adversarial regions (Encounters).

Our Robotarium experiments consider four different strategies (Set S_e) for the explorer team: *attacking and changing direction*, *attacking and changing speed*, *defending and changing direction*, and *defending and changing speed*. We decompose this strategy set into two levels for GUT implementation in Robotarium: Level 1 considers deciding attack or defend (Table I); and Level 2 considers changing direction or speed (Table II) for a single explorer game while it considers triangle or diamond formation shape (Table III) for a multiple explorer game. Two different tactics payoff matrices are designed in Level 2 to differentiate the strategies between single-agent and multiagent cooperation.

We consider the *Winning Utility* following *Bernoulli Distribution* to represent individual high-level expected utility (teaming & cooperation needs) in the first level (Eq. (12)). And we assume that the second level's utility is described as the relative *Expected Energy Cost* Eq. (13) following *Normal Distribution*. Here, n and m represent the number of Explorers and Aliens respectively; t_e and t_a represent corresponding average energy levels of both sides; d represents the group average distance between two opponents; a and b are corresponding coefficients.

$$W(t_e, t_a, n, m) = a(\frac{t_e}{t_a})^{\frac{m}{n}}; \tag{12}$$

$$E(n, m, d) = b_0 + b_1 \int_{-\infty}^{+\infty} (n-m)x \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-d)^2}{2}} \mathrm{d}x \tag{13}$$

We compare our GUT approach with two different baseline approaches: 1) Random value selection approach (Eq. (14)),

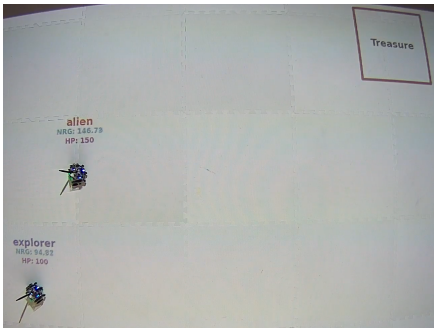


Fig. 3. 1 explorer vs 1 alien

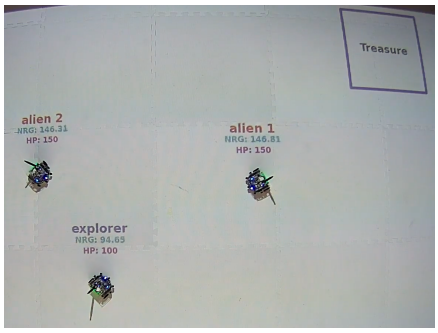


Fig. 4. 1 explorer vs 2 aliens

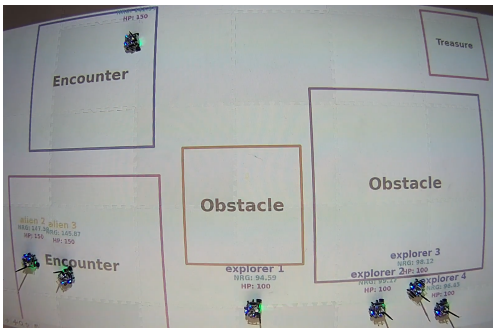
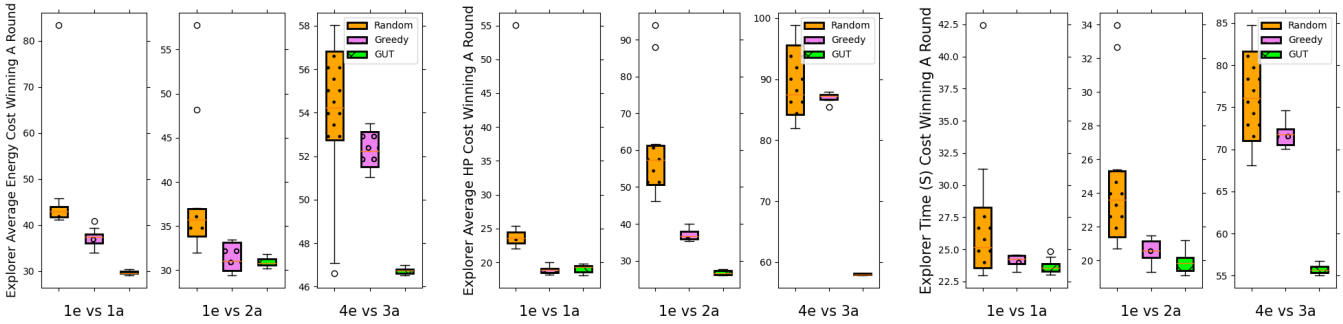


Fig. 5. 4 explorers vs 3 aliens



(a) Explorer Average Energy Cost (b) Explorer Average HP Cost (c) Explorer Time Cost

Fig. 6. The Performance Results in the Robotarium Experiments with Different Proportion of the Explore Game.

which chooses a strategy from a set S_e by maximizing a reward function with two random variables; 2) Greedy algorithm [31] (Eq. (15)), which maximizes the combined utilities of win rate and HP metric together (equivalent to a one-level GUT).

$$s_e^* = \arg\max_{i \in S_e} [100 - c_1 \cdot s_{i_1} \cdot d - c_2 \cdot s_{i_2} \cdot n_a \cdot u_{hp}] \quad (14)$$

$$s_e^* = \arg\max_{i \in S_e} [W_i \cdot HP_i] \quad (15)$$

Here, s_{i_1} and s_{i_2} represent the energy-dependent and health-dependent random reward values of strategy i, respectively. They follow the Gaussian distributions with different expectations. d is the distance between explorer and goal point. n_a and u_{hp} are the number of active aliens and their average unit attacking damage cost. c_1 and c_2 are the corresponding coefficients. W_i and HP_i are the winning and HP utility values of strategy i.

TABLE I
LEVEL 1 (ATTACK/DEFEND) TACTICS PAYOFF MATRIX.

Utility: ET / AT	Attack	Defend
Attack	W_{AA}	W_{DA}
Defend	W_{AD}	W_{DD}

ET - Explorer Tactics
AT - Alien Tactics

We implement each case with real robots in the Robotarium and conduct ten simulation trials (rounds) for each scenario in the Robotarium simulator. We initialize each robot in the same group (explorer or adversary) with equal energy (battery) levels and health points (HP); for example, explorer with 100 points and alien with 150 points in Energy and HP correspondingly. Also, we assumed that every moving step and attack damage cost 0.1% of energy and 0.3% of HP, respectively. Other settings are similar to the previous experiments. The video demonstrates the simulation and hardware application of corresponding experiments in Robotarium is available at https://youtu.be/eBJayhZQ6X4.

TABLE II
LEVEL 2 PAYOFF MATRIX FOR SINGLE EXPLORER.

Utility: ET / AT	ΔSpeed	ΔDirection
Follow	HP_{SF}	HP_{DF}
Retreat	HP_{SR}	HP_{DR}

TABLE III
LEVEL 2 PAYOFF MATRIX FOR MULTIPLE EXPLORERS.

Utility: ET / AT	Triangle	Diamond
Follow	HP_{TF}	HP_{DF}
Retreat	HP_{TR}	HP_{DR}

Results and Discussion: Fig. 6 presents the results of the Robotarium experiments. The average costs of energy, HP, and the time taken to complete the mission of explorer are shown in Figs. 6(a), 6(b), and 6(c), respectively. The data shows that in the single explorer cases (scenarios *1e vs. 1a* and *1e vs. 2a*), both GUT and greedy stand out compared to the random approach, but the GUT is not significantly better than the greedy approach consistently in all the performance metrics. We attribute this to the fact that GUT uses only two levels of action decomposition and therefore does not offer significant advantages compared to the one-level GUT/greedy approach in this simple scenario. However, for the multiple explorer case (scenario *4e vs. 3a*), the GUT shows superior performance over other methods in all metrics, demonstrating that the GUT can help MAS organize their behaviors and select

TABLE IV
PERFORMANCE RESULTS IN ROBOTARIUM EXPERIMENTS.

APP / PRD	Winning Rate			Lost Explorers Per Round/win		
	Random	Greedy	GUT	Random	Greedy	GUT
1e vs 1a	100%	100%	100%	-	-	-
1e vs 2a	90%	100%	100%	0.1	-	-
4e vs 3a	50%	90%	100%	2.8	2.0	-

a suitable strategy adapting to complex situations. We can get a similar observation from the perspective of the winning rate and the average number of explorers lost from the Table. IV.

Generally speaking, by demonstrating the *Explore* domain in the Robotarium, we demonstrated that the GUT could help the intelligent agent (robot) rationally analyze different situations in real-time, effectively decompose the high-level strategy into low-level tactics, and reasonably organize groups' behaviors to adapt to the current scenario. From the system perspective, through applying the GUT, the group presents more complex strategies or behaviors to solve the dynamic changing issues and optimize or sub-optimize the group utilities in MAS cooperation. From the individual agent perspective, GUT reduces the agent's costs and guarantees sustainable development for each group member, much like human society.

V. CONCLUSION AND FUTURE WORK

We introduce a new network model called *Game-theoretic Utility Tree* (GUT), mimicking the multi-agent decision-making process for cooperation working in adversarial environments. We verified the effectiveness of the GUT in the real robot application through the implementation of the *Explore Game* on the Robotarium hardware-simulator multi-agent testbed and compared it with the *random* and *greedy* approaches in three scenarios.

For future work, it will be essential to improve GUT from different perspectives, such as optimizing GUT structure through learning from different scenarios, designing appropriate utility functions, building suitable predictive models, and estimating reasonable parameters fitting the specific scenario.

REFERENCES

[1] M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[2] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems." in *AAAI Fall Symposium: Regarding the Intelligence in Distributed Intelligent Systems*, 2007, pp. 1–6.

[3] 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*, 2019, extended Abstract.

[4] J. Shen, X. Zhang, and V. Lesser, "Degree of local cooperation and its implication on global utility," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*. IEEE Computer Society, 2004, pp. 546–553.

[5] P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[6] M. Jun and R. D'Andrea, "Path planning for unmanned aerial vehicles in uncertain and adversarial environments," in *Cooperative control: models, applications and algorithms*. Springer, 2003, pp. 95–110.

[7] N. Agmon, G. A. Kaminka, and S. Kraus, "Multi-robot adversarial patrolling: facing a full-knowledge opponent," *Journal of Artificial Intelligence Research*, vol. 42, pp. 887–916, 2011.

[8] R. Yehoshua and N. Agmon, "Adversarial modeling in the robotic coverage problem," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 891–899.

[9] Q. Yang and R. Parasuraman, "Hierarchical needs based self-adaptive framework for cooperative multi-robot system," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 2991–2998.

[10] ——, "Needs-driven heterogeneous multi-robot cooperation in rescue missions," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 252–259.

[11] Y. Shapira and N. Agmon, "Path planning for optimizing survivability of multi-robot formation in adversarial environments," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4544–4549.

[12] M. Benda, V. Jagannathan, and R. Dodhiawala, "On optimal cooperation of knowledge sources - an empirical investigation," Boeing Advanced Technology Center, Boeing Computing Services, Seattle, WA, USA, Tech. Rep. BCS–G2010–28, July 1986.

[13] P. Cheng, "A short survey on pursuit-evasion games," *Department of Computer Science, University of Illinois at Urbana-Champaign*, 2003.

[14] W. L. Scott III, "Optimal evasive strategies for groups of interacting agents with motion constraints," Ph.D. dissertation, PhD thesis, Princeton University, 2017.

[15] V. R. Makkapati and P. Tsiotras, "Optimal evading strategies and task allocation in multi-player pursuit–evasion problems," *Dynamic Games and Applications*, pp. 1–20, 2019.

[16] B. von Stengel and D. Koller, "Team-maxmin equilibria," *Games and Economic Behavior*, vol. 21, no. 1-2, pp. 309–321, 1997.

[17] A. Celli and N. Gatti, "Computational results for extensive-form adversarial team games," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[18] R. B. Myerson, *Game theory*. Harvard university press, 2013.

[19] P. C. Fishburn, "Utility theory for decision making," Research analysis corp McLean VA, Tech. Rep., 1970.

[20] M. J. Kochenderfer, *Decision making under uncertainty: theory and application*. MIT press, 2015.

[21] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1699–1706.

[22] M. K. Sohrabi and H. Azgomi, "A survey on the combined use of optimization methods and game theory," *Archives of Computational Methods in Engineering*, vol. 27, no. 1, pp. 59–80, 2020.

[23] A. X. Jiang and K. Leyton-Brown, "A tutorial on the proof of the existence of nash equilibria," *University of British Columbia Technical Report TR-2007-25. pdf*, vol. 14, 2009.

[24] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The complexity of computing a nash equilibrium," *SIAM Journal on Computing*, vol. 39, no. 1, pp. 195–259, 2009.

[25] Q. Yang and R. Parasuraman, "How can robots trust each other? a relative needs entropy based trust assessment models," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2021.

[26] J. Kim, "Multirobot exploration while building power-efficient sensor networks in three dimensions," *IEEE transactions on cybernetics*, vol. 49, no. 7, pp. 2771–2778, 2018.

[27] T. H. Chung, G. A. Hollinger, and V. Isler, "Search and pursuit-evasion in mobile robotics," *Autonomous robots*, vol. 31, no. 4, p. 299, 2011.

[28] S. Nadarajah and K. Sundaraj, "A survey on team strategies in robot soccer: team strategies and role description," *Artificial Intelligence Review*, vol. 40, no. 3, pp. 271–304, 2013.

[29] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[30] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[31] A. Vince, "A framework for the greedy algorithm," *Discrete Applied Mathematics*, vol. 121, no. 1-3, pp. 247–260, 2002.