International Journal of Control, Automation, and Systems 22(4) (2024) 1212-1224
http://dx.doi.org/10.1007/s12555-022-1182-5 ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Hedonic Coalition Formation for Distributed Task Allocation in Heterogeneous Multi-agent System

Lexing Wang, Tenghai Qiu*, Zhiqiang Pu, Jianqiang Yi, Jinying Zhu, and Wanmai Yuan

Abstract: Due to the complexity of tasks in the real world, multiple agents with different capabilities tend to cooperate to handle diverse requirements of these tasks by forming coalitions. To solve the problem of finding optimal heterogeneous coalition compositions, this paper proposes a novel distributed hedonic coalition formation game method to solve the task allocation problem for multiple heterogeneous agents. Firstly, to quantify the intention of an agent joining each coalition, a utility function for each agent is designed based on the cost and the reward with regard to the given tasks, where the heterogeneous requirements of tasks are also considered. Then, a preference relation related to the utility function is designed for the self-interested agents autonomously choose to join or leave a coalition. Subsequently, a theorem is presented, and analyses have been conducted to show that the proposed method achieves a Nash-stable solution in the heterogeneous system. Further, to develop a Nash stable partition result, a distributed hedonic coalition formation algorithm containing prioritization and consensus stages is designed for each agent to make decisions. The algorithm is implemented based on local interactions with neighbor agents under a connected communication network. Finally, simulations are conducted to verify the performance of the proposed method. Results show that the proposed method has the feasibility in solving heterogeneous composition and the broader scalability in different scenarios.

Keywords: Coalition formation, hedonic games, heterogeneous agents, Nash stable, task allocation.

1. INTRODUCTION

Multi-agent cooperation has attracted considerable attention for its wide applications in multidisciplinary engineering problems [1-5], e.g., planetary exploration, microgrids' energy imbalance, hazardous waste cleanup, etc. Multi-agent cooperation can accomplish a task that a single agent may fail to handle, which presents a desirable performance in scalability, reconfigurability, and adaptivity [6-8]. In addition, because of the dynamic and unpredictable nature of tasks, it is difficult to design an agent that can efficiently adapt to all tasks, and it is not cost-effective to equip all sensors or actuators to an agent [9,10]. Therefore, heterogeneous agents cooperating to perform tasks has become one of the attractive issues in multi-agent cooperation due to agents' complementary capabilities, which can complete tasks more efficiently [11]. The term 'heterogeneous' refers to the agents equipped with different sets of sensors/actuators [12], or a heterogeneous group of agents (e.g., drones, autonomous ground vehicles, underwater vehicles, etc.) [13]. However, the challenge is how to find the optimal heterogeneous coalition compositions.

Task allocation of multiple agents is proved to be NP-hard [14]. The problem becomes more complicated for agents with cardinality increasing and heterogeneity in a large-scale heterogeneous system. Therefore, it is necessary to design a task allocation method for heterogeneous multiple agents, which satisfies specific task requirements while explicitly taking into account the specialized capabilities of the agents.

Several methods for task allocation with homogeneous agents have been proposed [15-17]. However, these methods assume that all agents in a coalition are interchangeable and thus are not suitable for multi-task scenarios that involve several heterogeneous agents. For multiple agents with heterogeneous capabilities, typical task allocation methods assign tasks to agents based on the suit-

Manuscript received December 16, 2022; revised April 6, 2023; accepted April 29, 2023. Recommended by Associate Editor Hongyi Li under the direction of Senior Editor Hyun Myung. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0102404, in part by the External Cooperation Key Project of Chinese Academy Sciences under Grant No.173211KYSB20200002, and in part by the Beijing Nova Program under Grant 20220484077.

Lexing Wang is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: wanglele2017@ia.ac.cn). Tenghai Qiu, Zhiqiang Pu, Jianqiang Yi, and Jinying Zhu are with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mails: {tenghai.qiu, zhiqiang.pu, jianqiang.yi, jinying.zhu}@ia.ac.cn). Wanmai Yuan is with the Information Science Academy, China Electronics Technology Group Corporation, China (e-mail: yuanwanmai7@163.com).
* Corresponding author.

ability of the agents to perform specific tasks as well as the requirements of the tasks [13].

Swarm intelligence algorithms such as particle swarm optimization or genetic algorithm were used to solve such problems [18-20]. However, such centralized methods are limited by communication and are susceptible to a single point of failure. To reduce the computation and communication overhead, decentralized multi-agent methods are utilized to solve the above problem [21-24], where the agents can autonomously make decisions through local information.

A greedy algorithm [25] was utilized to solve this problem in distributed multi-agent systems. However, the agents must compute all possible coalitions and agree on the best one. Hence, this algorithm is best applied when the space of possible coalitions is naturally limited. In [26], a decentralized approach for heterogeneous robot swarms is introduced. The approach computes optimal rates at which the robots must switch between different tasks. These optimal rates, in turn, are used to compute probabilities that determine the stochastic control policies of each robot. Zhang *et al.* [27] proposed a fuzzy collaborative intelligence-based algorithm to optimize the package loading/unloading task, where it improved coordination between heterogeneous agents using the agents' capabilities. However, the algorithm only considers the number of capabilities and ignores the meaning of these capabilities or whether there are beneficial with respect to the given tasks. Market-based approaches have also been used to handle task allocation problems [28]. However, a common feature of the auction or market-based methods is that extensive communication is required in the bidding mechanisms, and it is generally used for single-task agents. Similarly, consensus-algorithm [29] also requires large amounts of data transmission to reach consistent convergence. Moreover, in the market-based or consensus-algorithm method, each agent is required to notify its altered decision with negotiation, which increases inter-agent communication of the agents' system.

Among the different methods for distributed task allocation, game-theoretical methods [30-33] present some superior performances in efficiency, scalability, computational cost, etc. [34]. Moreover, the hedonic game methods [35] provide analytical and theoretical enlightenment for researchers to solve the distributed task allocation problem. A hedonic game (also known as a hedonic coalition formation game) refers that each agent holds a preference order over the coalitions and autonomously forms several disjoint coalitions, where Nash stability plays a key role since it yields a social agreement among the agents even without having any negotiation. Hedonic games have been a topic of high interest in cooperative game theory. In [36], a game-based method was utilized for self-organizing agents to work to wireless networks, i.e., modeling the data collection by agents from several arbitrarily located tasks. Jang *et al.* [37] proposed a game-theoretical autonomous decision-making framework to divide the agents into several disjoint coalitions toward the tasks. This algorithm achieved a convergence to Nash stability and presented a good performance in asynchronous and dynamic environments.

To the best of our knowledge, few studies have considered using hedonic games to solve the task allocation problem for heterogeneous agents. The authors in [12,38] firstly applied a hedonic coalition game to task assignment of heterogeneous UAVs and proposed a hedonic coalition game algorithm based on bipartite graph matching. However, the methods in [12,38] are centralized and susceptible to a single point of failure. Meanwhile, the methods have set no constraints on the coalition sizes. The number of agents in a coalition may exceed the requirements of task, resulting a redundant allocation.

Unlike the methods in [12,38], our goal is to propose a distributed and scalable method to solve the task allocation problem, allowing each agent to make decisions autonomously. Meanwhile, in the task allocation of heterogeneous multi-agent system, the redundant allocation problem should be considered, i.e., not to allocate all the agents in the system to the tasks, but to allocate the agents reasonably according to the task's requirements. Motivated by the above observations, this paper studies a task allocation problem for heterogeneous agents, i.e., how heterogeneous multiple agents autonomously and intelligently form disjoint (non-overlapping) coalitions with respect to given tasks. The heterogeneity of a multi-agent system in this paper is described by the assignment constraints of tasks. To find the optimal heterogeneous coalition compositions, a distributed hedonic coalition formation game method is proposed to solve the problem, where each agent prioritizes the given tasks while explicitly taking into account its utility in performing the tasks. According to the proposed method, the agents are distributed into a set of distinct and non-overlapping coalitions depending on the positions and requirements of the tasks exploiting a hedonic game theory.

Compared with existing works, the main novelties and contributions of this article are listed as the following:

1) Different from handling with the task allocation problem in a centralized way [12,38], this paper proposes a distributed hedonic coalition formation game method for task allocation with heterogeneous agents, which enables self-interested agents autonomously choose to join or leave a coalition according to their intention.
2) Different from ignoring the redundant allocation problem in existing researches, the paper considers the size of a coalition while aiming to find the optimal heterogeneous coalition compositions, i.e., the capabilities and number of members in the coalitions

are suitable to the requirements of tasks.

3) Unlike only finding a feasible solution in the other methods, a Nash stable result is considered in this paper. All agents are satisfied with their current utilities and are in the coalitions they most prefer. Moreover, the use of Nash stability can reduce the communication burden between agents required to reach a social agreement.

The remainder of this paper is organized as follows: Section 2 introduces a problem formulation and some preliminaries. In Section 3, we firstly define some notions for hedonic games. Then, a preference order is designed to form Nash stable coalitions for the heterogeneous multi-agent system. Moreover, we propose a distributed algorithm to achieve coalition formation for task allocation in the system. Section 4 presents simulation results. Section 5 concludes the paper and identifies further work.

2. PRELIMINARIES

2.1. Problem formulation

Consider a group of n agents and a group of m tasks, which are denoted as the sets $A = \{a_1, a_2, ..., a_n\}$ and $T = \{t_1, t_2, ..., t_m\}$, respectively. Multiple agents can work cooperatively on a task, and each agent can perform at most one task.

Each agent $a_i \in A$ has some different capabilities, i.e., equipped with different sets of sensors or actuators. The capabilities set of a_i is denoted as C_{a_i}, where $c_l \in C_{a_i}$ is a type of capability provided by a_i. It is supposed that $|C_{a_i}| \geqslant 1$, which means a_i has at least one capability. Notably, $|\cdot|$ represents the cardinality of a set throughout this paper, i.e., the number of elements in the set. For example, $C_{a_i} = \{c_1, c_2, c_3\}$, $|C_{a_i}| = 3$. The capabilities requirement set of a task t_j is denoted as C_{t_j}, where $c_k \in C_{t_j}$ is a type of capability required by t_j. Each task's requirements satisfy $\left|C_{t_j}\right| \geqslant 1$ as well. C is the set of all capabilities considered in the allocation system. Thus, $C_{a_i} \subseteq C$ and $C_{t_j} \subseteq C$. In this paper, it supposes that $\cup_n C_{a_i} = C$ and $\cup_m C_{t_j} = C$.

Some assumptions are made as following:

Assumption 1: It is assumed that each agent can only be assigned to perform at most a single task. Every task may require multiple agents. Since an individual agent is considered to have limited capabilities to execute a task, it is beneficial for the agent to form a coalition with others. Each agent and task can be assigned to no more than one coalition (i.e., task-coalition pairs).

Assumption 2: Each agent is self-interested, i.e., each agent autonomously makes a decision according to its own intention. Note that agents in this paper may have different preferences with respect to given tasks, e.g., for an agent, a spatially closer task is more preferred, whereas this may not be the case for another agent.

Assumption 3: It is assumed that the required capacities of agents to complete a given task is pre-known information. The agents can have some of the same capabilities when performing a task.

Remark 1: Assumption 1 describes a taxonomy of the task allocation problems in this paper. This case falls into a single-task robot and multi-robot task (ST-MR) category as described in [39,40]. Assumption 2 defines the attributes and characteristics of agents. This article considers a distributed decision-making problem of a multi-agent system without a centralized node sending instructions. To realize it, each agent can make independent decisions and utilize its private objective function to update its state. Assumption 3 describes prerequisite information and assumes that some capacities of agents are allowed to be the same when be assigned to tasks. Otherwise, the task allocation may not be completed.

Problem 1: Therefore, the problem of this paper is to consider how to partition a group of heterogeneous agents autonomously with respect to all given tasks while meeting the requirements of tasks. At the same time, the objective is to maximize the sum of utilities for all agents in task allocation, where the term 'utility' refers to an evaluation of the value or cost of a specific task.

Example 1: For a simple example of a rescue scenario, each rescue site is regarded as a task, and each rescue vehicle is regarded as an agent. Consider two tasks, three agents, and three capabilities. Let c_1 be medical facility, c_2 be hoisting equipment, and c_3 be fire fighting facility. Agent a_1 can provide capabilities $\{c_1, c_2\}$, agent a_2 can provide capabilities $\{c_1, c_2, c_3\}$, and agent a_3 can provide capabilities $\{c_2, c_3\}$. Both rescue sites need medical supplies and have the fire breaking out, and the first rescue site also suffers from collapse. Thus, task t_1 requires $\{c_1, c_2, c_3\}$, and task t_2 requires $\{c_1, c_3\}$.

A feasible solution to Example 1 is that a_2 executes t_1 while a_1 and a_3 cooperatively execute task t_2.

Committed to solving the above problem, this paper proposes a hedonic coalition formation method for heterogeneous multiple agents forming disjoint coalitions corresponding to given tasks in a distributed manner.

2.2. Communication network of multiple agents

Information interactions among the agents can be represented by a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ is the set of vertexes indexed by the agents set, and $\mathcal{E} = \{(v_i, v_j) \mid v_i, v_j \in \mathcal{V}\}$ is an edge set that denotes the set of interaction links.

The adjacency matrix $W = [w_{ij}]_{n\times n}$ of graph G is defined as

$$w_{ij} = \begin{cases} 1, & \text{if } (v_i, v_j) \in \mathcal{E}, \\ 0, & \text{else}. \end{cases} \tag{1}$$

In this paper, we consider an undirected graph G, i.e., $w_{ij} = w_{ji}$. Besides, the self-connected edge of every vertex is defined as $w_{ii} = 1, \forall i$.

The neighbor of agent a_i is defined as a set of agents within its communication range, denoted as

$$\mathcal{N}_i = \{a_j \in A : \|q_j - q_i\| \leqslant R_c\}, \tag{2}$$

where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^2$. q_i and q_j denote the positions of agents a_i and a_j, respectively. R_c is the communication radius of the agents. $|\mathcal{N}_i|$ represents the number of neighbors of agent a_i. It is obvious that $a_j \in \mathcal{N}_i \Leftrightarrow a_i \in \mathcal{N}_j$.

3. HEDONIC COALITION FORMATION

The hedonic coalition formation method is presented in this section. For self-interested agents, they always choose the most 'interested' coalition to join. This property is defined by a preference over all the given tasks. Hence, some definitions and notions are introduced first. Then, a preference order is designed for decision-making for agents. Meanwhile, a utility function is designed based on the position and requirement of tasks. It proves that Nash stability can always be achieved. Finally, a distributed algorithm is designed for each agent to make decisions.

3.1. Notions and definitions of hedonic games

Firstly, we transform Problem 1 into a hedonic (coalition formation) game where each agent selfishly tends to join a coalition according to its preference.

Hedonic games are a subclass of cooperative games which concentrate on the competition between groups of agents. In hedonic games, multiple agents tend to form coalitions in accordance with an enforceable agreement and then perform decision-making. Generally, hedonic games have the following two properties [12] :

1) Utility of an agent depends solely on the other agents in the coalition the agent currently belongs to.
2) The coalitions are formed based on the preferences that the agents have over the set of all possible coalitions.

Problem 1 is closely related to the hedonic game for coalition formation, e.g., the similarities between utilities and coalitions. The difference is that the utility in Problem 1 is not only dependent on the coalitions but also on the tasks. Therefore, we model Problem 1 as a hedonic coalition formation game where the utility function for each agent is designed based on the cost and the reward with regard to the given tasks (introduced in Subsection 3.2). In order to better understand hedonic games, some definitions and concepts are introduced as follows.

A coalition $S_j \in A$ is a task-specific subset of the agents for executing task t_j. S_ϕ is defined as an empty coalition, which is the set of agents who choose task t_ϕ, where t_ϕ represents the void task. Namely, when t_ϕ is allocated to some agents, they do not perform any actions. Hereinafter, this paper interchangeably uses S_0 to indicate S_ϕ.

Definition 1: A coalition structure or coalition partition CS is defined as a set of disjoint coalitions which partition the agents set A.

Let $CS = \{S_0, S_1, S_2, ..., S_m\}$ denote a set of coalitions and each coalition is matched with a single task. For $\forall S_u$, $S_v \in CS$, $S_u \cap S_v = \phi$ and $\cup_{u=0}^{m} S_u = A$.

A task-coalition pair (t_j, S_j) is defined as "to do task t_j with coalition S_j", i.e., each coalition is assigned a particular task and is responsible for its execution.

Definition 2: The preference relation or preference order specifies agent a_i a reflexive, complete, and transitive binary relation over the set of all coalitions that a_i can possibly forms.

The preference of agent a_i is represented by the symbol $\prec_i$ or $\sim_i$. For example, for two coalitions S_1 and S_2, $S_2 \prec_i S_1$ denotes that a_i has a strong preference on S_1 than S_2 while $S_1 \sim_i S_2$ denotes a_i has the same preference on S_1 and S_2. Likewise, $\preceq_i$ indicates the weak preference of agent a_i. It is worth noting that preference relation is a core concept that affects the final formation process of the coalition.

Definition 3 (Nash stable) [36]**:** A coalition structure is Nash stable if no agent can benefit from moving from its coalition S_u to another (possibly empty) coalition S_v. It can be formulated as

$$\forall a_i \in A,\ \forall S_v \in \{CS \backslash S_u\},\ \{S_v \cup \{a_i\}\} \preceq_i S_u, \tag{3}$$

where '\' represents the difference operation between two sets.

Nash stable is beneficial to reduce communication burden among multiple agents while yielding a social agreement among these agents even without having any negotiation. Furthermore, the significance of Nash stable is shown in [37,38]. Therefore, it is important to find a coalition partition with Nash stability.

Remark 2: In the hedonic games, the term 'hedonic' pertains that the utility of an agent only dependent on the members of the coalition to which the agent belongs to. However, in Problem 1, the utility is also related to the tasks. Therefore, to model Problem 1 as a hedonic coalition formation game, the preference relations of the agents must be clearly defined. The utility should indicate overall how well an agent fits within its coalition in terms of payoff and cost. In the next section, we will formally define how the preferences of the agents over the coalitions can be used for the coalition formation.

3.2. Preference relation

For a heterogeneous multi-agent system, the nature of various abilities of agents determines the differences in their preferences when prioritizing tasks. Meanwhile, in the task allocation for heterogeneous agents, the capabilities matching between agents in coalitions and tasks is an vital criterion for solutions.

In order to evaluate the capability matching between a coalition S_j and a task t_j, the size of overlapping capabilities subset between agents in S_j and t_j is defined as

$$K_{S_j} = \left|\cup_{a_i \in S_j} C_{a_i} \cap C_{t_j}\right|. \tag{4}$$

As (4) shows, the larger the value of K_{S_j}, the greater the capability matching between the coalition and the task.

It is supposed that if multiple agents execute a task together as a coalition, then they are given a certain level of reward for the task. The amount of the reward depends on the size of the capability matching between the agents in the coalition and the task. Therefore, the reward function of coalition S_j with respect to task t_j is defined as

$$V(t_j, K_{S_j}) = \frac{V_{\max} \cdot K_{S_j}}{\left|C_{t_j}\right|} \exp(-K_{S_j}/\left|C_{t_j}\right| + 1), \tag{5}$$

where $V_{\max}$ is the maximal reward when t_j's requirement is satisfied. As shown in (5), the reward of the coalition increases with an increasing K_{S_j}. When $K_{S_j} = \left|C_{t_j}\right|$, the coalition will receive a maximal reward, i.e., $V_{\max}$.

The reward function $V(t_j, K_{S_j})$ can be interpreted as follows: as the size of capability matching of coalition S_j increases, the reward obtained by the coalition gradually increases. When the size is equal to the requirement of the task, the maximum reward value is reached.

Accordingly, according to the definition of the reward function for the coalition, we define the individual reward for agent a_i.

$$V_{a_i}(t_j) = V(t_j, K_{S_j}) - V(t_j, K_{S'_j}), \tag{6}$$

where $S'_j = S_j \backslash \{a_i\}$. Equation (6) is called a marginal reward for agent a_i. Marginal reward describes the additional total reward generated by increasing overlapping capbilities by one agent, which serves as the income of the agent.

Remark 3: Equation (6) implies that, if agent a_i joins a coalition resulting in an increase in the capability matching value, the agent will acquire a positive reward. Otherwise, the reward that a_i acquires is zero.

Based on the above reward function (6), the individual's utility is defined as the marginal reward minus the individual's cost required to the given task. Notably, it is crucial for the utility to be non-negative in order to achieve a Nash-stable outcome [41]. Thus, the utility function is formulated as

$$u_{a_i}(t_j, K_{S_j}) = \begin{cases} V_{a_i}(t_j) - k_c \cdot \ cost_i(a_i, t_j), & u_{a_i} > 0, \\ 0, & \text{else}, \end{cases} \tag{7}$$

where k_c is a constant. $cost_i(a_i, t_j)$ is the cost that agent a_i needs to pay for task t_j. We simply set the cost as a function of the distance from agent a_i to task t_j. Of course, other indicators can also serve as the cost according to applications, such as fuel consumption.

As mentioned above, the preference relation of agents affects the final formation of coalitions. Therefore, according to the utility function designed in (7), a preference order is designed as follows:

Definition 4 (Preference relation)**:** Agents always prefer to join a coalition that makes their utility greater, i.e., $S_1 \preceq_i S_2 \Leftrightarrow u(t_1, K_{S_1}) \leqslant u(t_2, K_{S_2})$.

In this paper, $u_{a_i}(t_\phi) = \varepsilon$ (ε is a small positive number) is defined, which is beneficial to the convergence of the proposed algorithm. The reason is that if an agent is assigned to task t_j whose requirement is already satisfied, the agent's utility will be zero. Due to the utility that the agent chooses t_ϕ is more than t_j, thus, the agents will choose t_ϕ autonomously.

Theorem 1: For a heterogeneous multi-agent system composed of agents with preferences in Definition 4, Nash stability can always be achieved.

Proof: Suppose that each agent is in coalition S_ϕ at initialization. For a task-coalition pair (t_j, S_j), the difference between the number of capabilities required by t_j and the number of capabilities provided by S_j is defined as

$$\Delta_j := \left|C_{t_j}\right| - K_{S_j}, \tag{8}$$

where $\Delta_j > 0$ indicates that the requirement of task t_j has not been satisfied. While $\Delta_j = 0$, the system will be stable, that is, the requirements of each task have been satisfied. If an agent joins coalition S_j and causes a decrease in Δ_j, the agent will acquire a positive utility (according to (7)).

For an agent a_i who is not in S_j, if the utility of a_i in its current coalition (can be empty or non-empty) is less than the utility joining S_j, a_i will leave its current coalition and join S_j. Namely, when the requirement of a task is not satisfied, there must be agents who choose to join the task-coalition until $\Delta_j = 0$, because they can always obtain a greater utility.

When an agent deviates from its current coalition S_k, for the coalition S_k, it holds $\Delta_k > 0$. As mentioned above, other agents will join S_k until $\Delta_k = 0$. □

In this paper, the utility function is designed such that it tracks changes in its marginal improvement when one or more agents change their coalitions, which is a form of a potential function. Therefore, Theorem 1 also can be corroborated by the conclusion proposed by Monderer and Shapley [42], i.e., any such finite game would converge to a Nash-stable solution.

3.3. Hedonic coalition formation algorithm

In this section, a distributed algorithm is proposed for heterogeneous agents autonomously forming coalitions, where each agent makes decisions on its local information and shares information with its neighbor agents.

In the proposed algorithm, one iteration for an agent consists of two phases. The first phase of the algorithm is the process of selecting the preferred coalition, while the second phase facilitates the coalition partition converging to a consensus.

1) **Phase 1** (Prioritization process)**:** Agent a_i chooses the preferred coalition.
2) **Phase 2** (Consensus process)**:** Information exchanges between agent a_i and its neighbor agents, and update the agent's partition.

Initialization: Firstly, some local variables are defined for each agent. The symbol $\mathcal{O}_i$ is a binary value to specify whether an agent a_i is willing to choose the current task. CS_{a_i} is the up-to-date coalition partition known by agent a_i. $l_i \in Z_+$ is an integer variable to represent how many times CS_{a_i} has evolved (i.e., the number of iterations for updating CS_{a_i}).

Phase 1: Algorithm 1 shows the procedure of any agent a_i's (i.e., $a_i \in A$) Phase 1 at an iteration, where one iteration consists of a single run of Phase 1 and Phase 2. Given tasks, a_i calculates the size of the overlapping subset of the capabilities among the potential task-coalition pairs and individual rewards (Line 2, Algorithm 1). Then, a_i calculates the utility with respect to each task, and the task with maximal utility is marked as t_{j^*} (Line 3, Algorithm 1). Then, self-interested agent a_i compares the maximal utility with its own utility. The agent joins the newly found coalition if it is strongly preferred over its current coalition (Line 4, Algorithm 1). In this case, a_i updates its known partition and l_i increases. While a_i ascertains the most preferred coalition, the variable $\mathcal{O}_i$ becomes *true* (Lines 5-8, Algorithm 1).

Algorithm 1: Phase 1-Prioritization process.

```
   % Procedure: Choose the most preferred task
1: if $\mathcal{O}_i = false$ then
2:     For each task, calculate $K_{S_j}$ and $V(t_j, K_{S_j})$ using
       (4) and (5), respectively
3:     Calculate the utility of each task using (7) and the
       maximum is marked as $t_{j^*} = \arg\max_j u_{a_i}(t_j)$
4:     if $u_{a_i}(t_{j*}, K_{S_{j*}\cup\{a_i\}}) > u_{a_i}(t_j, K_{S_j})$ then
5:         join $S_{j*}$ and update $CS_{a_i}$
6:         $l_i = l_i + 1$
7:     end if
8:     $\mathcal{O}_i = true$
9: end if
   % end Procedure
```

Algorithm 2: Phase 2-Consensus process.

```
    % Procedure: Exchange local information with
    neighbor agents and update information
1:  Send $\Omega_i = \{l_i, CS_{a_i}\}$ to $a_k$ with $w_{ik} = 1$
2:  Receive $\Omega_k = \{l_k, CS_{a_k}\}$ from $a_k$ with $w_{ik} = 1$
3:  Construct $\Omega_{N_i} = \{\Omega_i, \cup_{w_{ik}=1}\Omega_k\}$
4:  for message $\Omega_k \in \Omega_{N_i}$ do
5:      if $VF_{a_k} > VF_{a_i}$ then
6:          $l_i = l_k$
7:          $CS_{a_i} = CS_{a_k}$
8:          $\mathcal{O}_i = false$
9:      end if
10: end for
    % end Procedure
```

Phase 2: The second phase is the consensus section of the algorithm. Here, a_i makes use of a consensus strategy to converge on final partition. The procedure of Phase 2 is shown in Algorithm 2.

Firstly, agent a_i sends its local information (including l_i and CS_{a_i}) to its neighbor agents, and correspondingly, it receives the information from its neighbors (Lines 1-2, Algorithm 2). Those received information composes the set Ω_{N_i}, including a_i' own information (Line 3, Algorithm 2). In order to compare these received task-coalition pairs (or existing partitions), one of the partitions is selected to exclusively update a_i's partition according to its individual preference. This partition is defined as a *valid partition* at a iteration [37]. If it is different with the partition of a_i, then a_i will update its known partition. In [37], the mechanism of valid partition is proved to be conducive to converging to a consensus partition.

Different from the method to choose a valid partition by a random sequence in [37], in this paper, we design a novel mechanism to choose it, where the evolution times and the distance to the task are considered. A valid factor is defined as

$$VF_{a_i} = l_i / \ cost_i(a_i, t_j). \tag{9}$$

If there exists any other partition CS_{a_k} such that $VF_{a_k} > VF_{a_i}$, then a_i considers CS_{a_k} more valid than CS_{a_i}, i.e., a *valid partition* (Line 5, Algorithm 2). Since CS_{a_k} is considered as more valid, agent a_i will need to re-examine if there is a preferred coalition. Thus, the agent sets $\mathcal{O}_i$ as false and updates the variable l_i (Lines 6-8, Algorithm 2). Algorithm 2 makes sure that there is only one valid partition that dominates any other partitions. In other words, multiple partitions locally evolve, but one of them eventually survives as long as a strongly connected network is given.

After completing Phase 2, agent a_i begins the next iteration until all agents' $\mathcal{O}_i$ are true.

Therefore, in the process of coalition formation for task allocation, each agent iterates asynchronously according to Algorithms 1 and 2. The coalition partition eventually tends to be stable.

3.4. Complexity analysis

As shown in Subsection 3.3, the hedonic coalition formation decision-making algorithm for each agent consists of Algorithms 1 and 2. The complexity of the algorithm primarily depend on lines 1-9 (Algorithm 1) and lines 4-10 (Algorithm 2). In Algorithm 1, each agent will check whether it can improve its utility by moving to any of the existing coalitions, the maximum count of which is m. The complexity of Algorithm 1 is $O(mn)$. In Algorithm 2, information exchange between an agent and its neighbor agents, the maximum count of which is n. Thus, the complexity of decision-making algorithm is $O(m\cdot n^2)$.

4. SIMULATIONS

4.1. Simulation settings

The locations of agents and tasks are sampled from a 100×100 environment. The initial positions of agents are randomly generated, and each agent is able to communicate with its nearby agents. The tasks are also randomly located away from the agents. The number of agents (n) varies between [6 200], and the number of tasks (m) varies between [2 80].

In order to verify the impact of heterogeneity on the algorithm (i.e., the number of capability types), in simulations, the type of capabilities is designed to vary from 3 to 5 for different scenarios. The total number of possible capabilities and the capabilities distribution are randomly generated for the agents and the tasks. Notably, the proposed method is implemented based on local interactions with neighbor agents under a connected communication network. Therefore, a strongly connected network is necessary for the agent system in simulations.

4.2. Simulation results

First, the proposed algorithm is verified in a simple scenario. Suppose the number of agents is 6 and the number of tasks is 2. Note that setting a small number of agents and tasks facilitates the analysis of the proposed algorithm. Simulations for large-scale multi-agent systems will be introduced later.

The spatial distribution of the agents and the tasks is shown in Fig. 1. Table 1 shows the capability set of the agents and the tasks. The allocation result in the scenario of Fig. 1 is that agent a_6 exclusively forms a coalition regarding to task t_1. Agents a_3 and a_5 form coalition corresponding to task t_2.

Remark 4: Setting only a few agents and tasks is beneficial to analyze the proposed algorithm. The scalability of the algorithm is verified below.

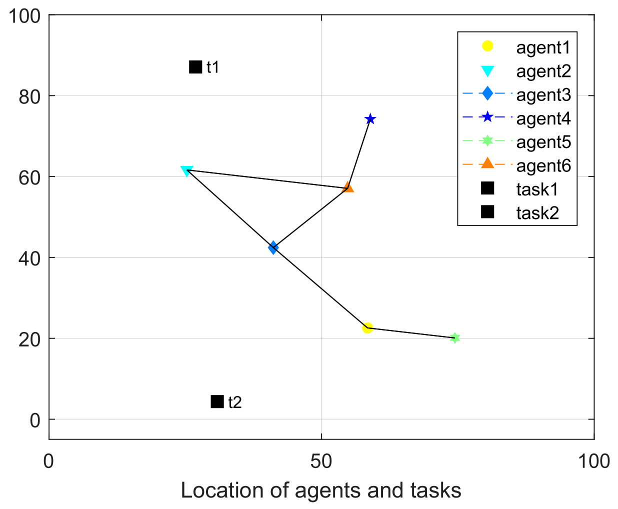


Fig. 1. Visualized locations of agents and tasks ($n=6$, $m=2$). The points of different shape and the lines between these points indicate the positions of heterogeneous agents and their communication network, respectively.

Table 1. Capabilities set of agents and tasks.

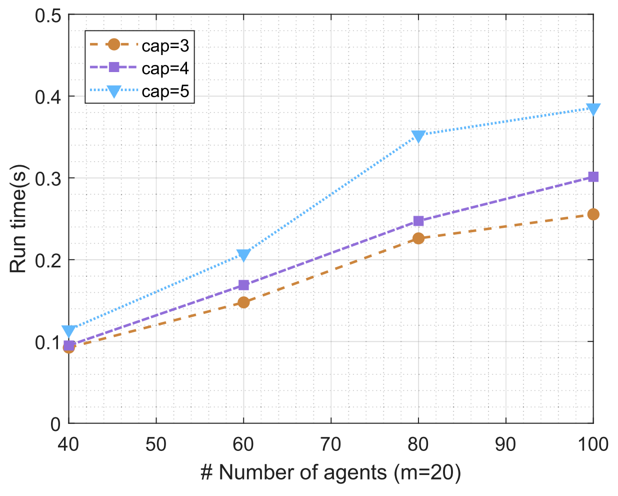
ID	Capabilities set	ID	Capabilities set
a_1	$\{c_1, c_3\}$	a_5	$\{c_1, c_2\}$
a_2	$\{c_2\}$	a_6	$\{c_2, c_3\}$
a_3	$\{c_1, c_3\}$	t_1	$\{c_2, c_3\}$
a_4	$\{c_2\}$	t_2	$\{c_1, c_2, c_3\}$

Take agent a_6 as an example to analyze the coalition formation in the proposed algorithm. Table 2 shows the iterations of agent a_6 in the scenario of Fig. 1.

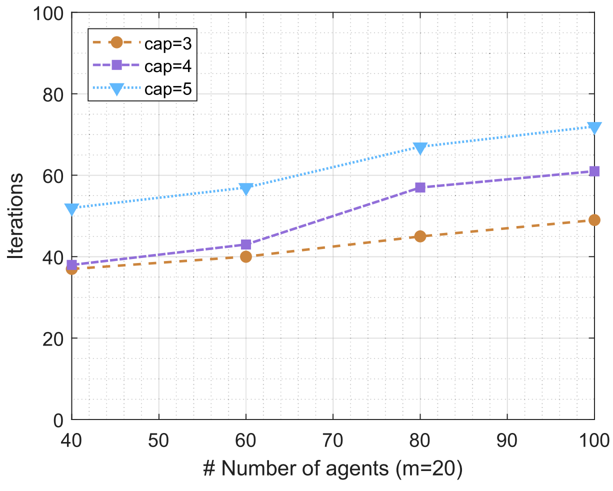
At iteration 1, due to the initialization (Initialization in Subsection 3.3), the known partition of a_6 is empty. a_6 plans to join S_1 according to its utility (Algorithm 1). Then, a_6 exchanges information with its neighbor agents and updates its known partition (Algorithm 2). Because the partition of neighbor agents is more valid (Lines 5-9, Algorithm 2), a valid partition updated by a_6 is $CS_{a_i}=\{S_1, S_2 \mid S_1=\{a_2\}, S_2=\{\phi\}\}$. At iteration 2, based on the partition in the last iteration, a_6 plans to join S_2 according to its utility (Algorithm 1). Then, due to the partition of neighbor agents is more valid, a_6 updates its valid partition as $CS_{a_i}=\{S_1, S_2 \mid S_1=\{a_2\}, S_2=\{a_3\}\}$ (Lines 5-9, Algorithm 2). At iteration 3, based on the partition in iteration 2, a_6 plans to join S_1 according to its utility (Algorithm 1). Notably, the valid partition is selected from a_6 itself, so valid partition is updated as $CS_{a_i}=\{S_1, S_2 \mid S_1=\{a_2, a_6\}, S_2=\{a_3\}\}$ (Lines 5-9, Algorithm 2). Likewise, in subsequent iterations, a_6 plans to join a coalition based on individual utility (Algorithm 1). Then, the agent exchanges information with its neighbor agents and chooses a valid partition (Algorithm 2). The iteration is conducted until

Table 2. Iterations of a_6 in coalition formation.

Iteration		1	2	3	4	5	6	7	8
Phase 1		S_1	S_2	S_1	S_1	S_1	S_1	S_1	S_1
Phase 2	S_1	$\{a_2\}$	$\{a_2\}$	$\{a_2, a_6\}$	$\{a_2, a_6\}$	$\{a_2, a_6\}$	$\{a_6\}$	$\{a_6\}$	$\{a_6\}$
	S_2	ϕ	$\{a_3\}$	$\{a_3\}$	$\{a_3, a_5\}$	$\{a_3, a_5\}$	$\{a_3, a_5\}$	$\{a_3, a_5\}$	$\{a_3, a_5\}$



(a) Run time.



(b) Iterations.

Fig. 2. Visualized task allocation results for different number of agents, where $n \in \{40, 60, 80, 100\}$, $m = 20$.

iteration 8, where a stable coalition structure is formed.

It is worth noting that it seems that multi-agent system has formed a final coalition structure at the iteration 6 of a_6 in Table 2. However, the fact is that other agents are still making decisions. This indicates that every agent locally updates its known partition and executes the algorithm asynchronously.

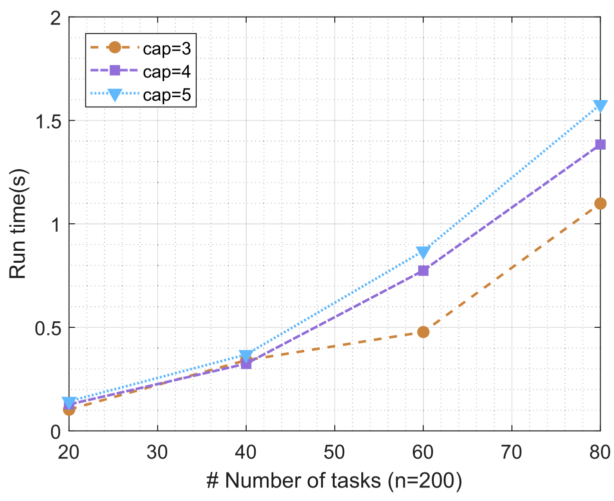
Remark 5: As the capabilities set is shown in Table 1, it seems that agent a_2 or a_4 is more suitable than a_5 to form a coalition with a_3 to perform task t_2. However, because the cost is also considered in the utility function (7), agent a_5 has a minimal cost to task t_2 than agents a_2 and a_4. Therefore, a_5 and a_3 will form the coalition S_2 to task t_2.

Then, the proposed algorithm is implemented in scenarios of different number of the agents. In these scenarios, the tasks number m is 20 while the agents count n varies from 40 to 100. Figs. 2(a) and 2(b) show the visualized task allocation results. Figs. 2(a) and 2(b) show the run time and iterations varies with the number of the agents, respectively. The blue, purple, and brown lines represent the simulation results when the type of required capabilities is 3, 4, and 5, respectively.

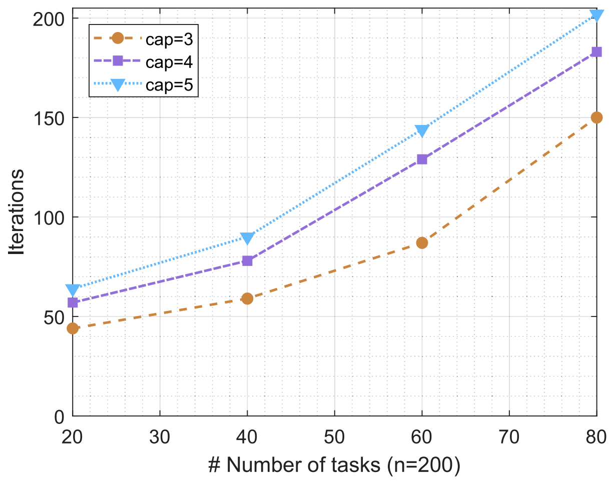
Figs. 2(a) and 2(b) show that, with the increase in the number of the agents, both the number of iterations and the running time are increasing. Moreover, the increase in the types of required capabilities will also lead to an increase in these two indicators. As shown in Fig. 2, when the number of required capabilities is fixed, the number of neighbors of an agent increases as the number of agents increases. Therefore, it will increase the convergence time of the consensus process (Algorithm 2), that is, the number of iterations and running time will increase. When the number of agents remains unchanged, due to the increase in capability number, it may be necessary to find more agents to form a coalition to satisfy the task's requirements. Therefore, iterations and running time have also increased. That is, the heterogeneity will increase the complexity of the task allocation.

Moreover, the proposed algorithm is implemented in scenarios of different number of the tasks. In these scenarios, the agents number n is 200 while the tasks number m varies from 20 to 80. Figs. 3(a) and 3(b) show the visualized task allocation results. As the number of tasks increases, the number of task-coalition pairs that can be selected by the agent will also increase. The agent systems require more time to achieve Nash stability. Thus, both the number of iterations and the running time increase as the number of tasks increases.

Therefore, the proposed algorithm presents good scalability when the number of the agents or tasks increases. Although the multiple capabilities required by the tasks make the problem more complex, the proposed algorithm can achieve a successful task allocation.



(a) Run time.



(b) Iterations.

Fig. 3. Visualized task allocation results for different number of tasks, where $m \in \{20, 40, 60, 80\}$, $n = 200$.

4.3. Performance comparisons

To further illustrate the effectiveness of the proposed algorithm, we compared it with the existing algorithms [12], namely the BGM-based (BGM: bipartite graph matching) algorithm.

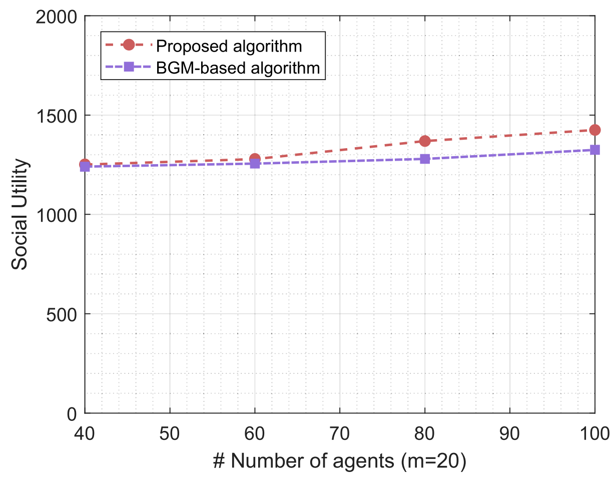
As illustrated in Problem 1, an objective is to maximize the utility in task allocation. Therefore, a social utility function is defined, i.e., the sum of utility functions of all agents.

$$U = \sum_{1}^{n} u_{a_i}(t_j, K_{S_j}). \tag{10}$$

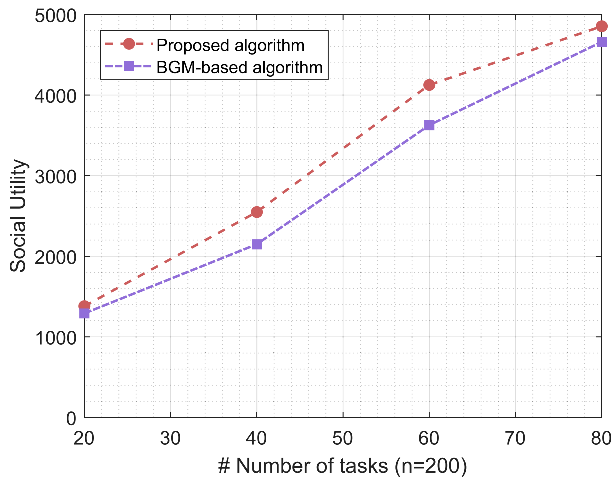
Figs. 4(a) and 4(b) show the social utility evolves in the scenarios of different agents and tasks, respectively. Red (circle) represents the proposed algorithm, purple (square) represents the BGM-based algorithm. In these scenarios, the proposed algorithm has acquired higher utility than the BGM-based algorithm.

In the allocation of heterogeneous agents, it should meet the heterogeneous requirements of tasks. Meanwhile, it is also necessary to avoid the redundancy of agent capabilities, i.e., the same capabilities of agents to perform a task. Here, a definition called redundancy capability matching is utilized to evaluate the redundant capacities in task allocation.

$$\rho = 1 - \frac{\sum\limits_{a_l \in CS \setminus S_0, t_j \in T} \left| C_{a_i} \cap C_{t_j} \right|}{\sum\limits_{a_l \in CS \setminus S_0} |C_{a_i}|}. \tag{11}$$



(a) Different number of agents.



(b) Different number of tasks.

Fig. 4. Comparison of social utility of two methods. (a) Different number of agents, where $n \in \{40, 60, 80, 100\}$, $m = 20$. (b) Different number of tasks, where $m \in \{20\ 40, 60, 80\}$, $n = 200$.

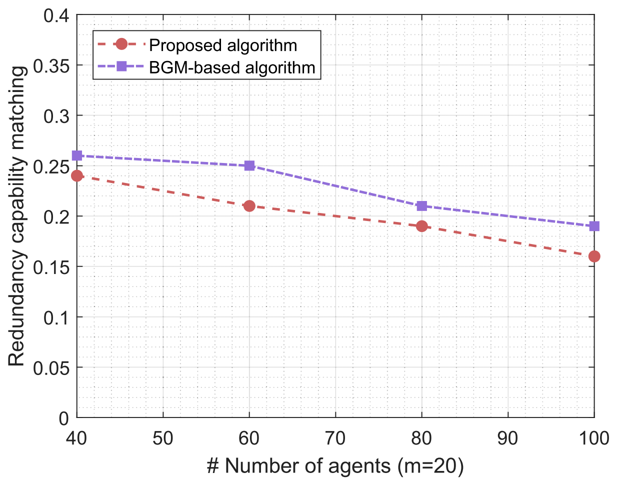


Fig. 5. Comparison of redundancy capability matching of two algorithms in different number of agents, where $n \in \{40, 60, 80, 100\}$, $m = 20$.

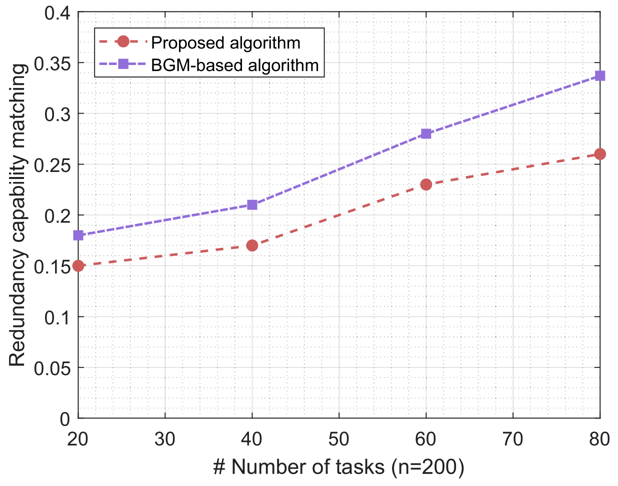


Fig. 6. Comparison of redundancy capability matching of two algorithms in different number of tasks, where $m \in \{20, 40, 60, 80\}$, $n = 200$.

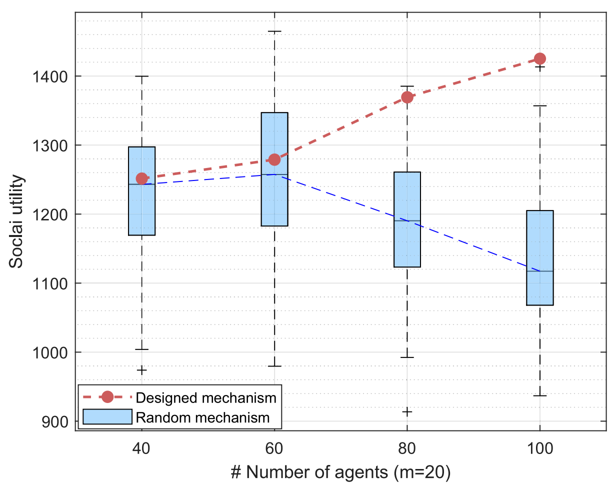


Fig. 7. Comparison of social utility of algorithms using proposed mechanism and using random mechanism, where $n \in \{40, 60, 80, 100\}$, $m = 20$.

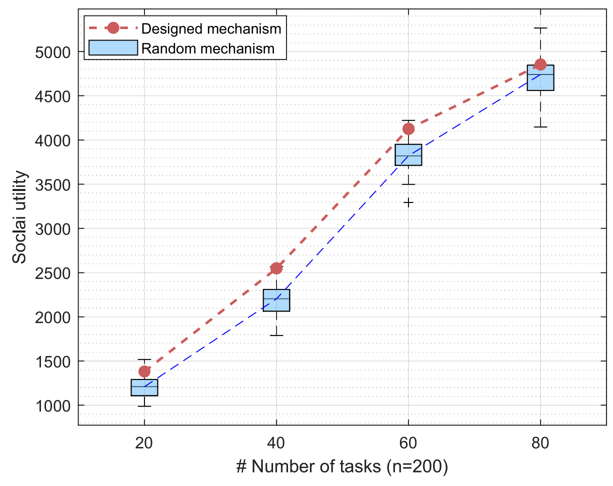


Fig. 8. Comparison of social utility of algorithms using proposed mechanism and using random mechanism, where $m \in \{20, 40, 60, 80\}$, $n = 200$.

As (11) shows, a lower ρ means fewer redundant capacities for the agent system. Figs. 5 and 6 show the ρ of the two algorithms in the scenarios with different agents and tasks. Likewise, red (circle) represents the proposed algorithm, and purple (square) represents the BGM-based algorithm. It can be seen that the proposed algorithm achieves a low redundancy capability matching than the BGM-based algorithm. When fixed $m = 20$, with the number of agents increasing, the number of neighbors agents of each agent may increase. Therefore, each agent can form a coalition with the neighbor agents with fewer redundant capabilities, which causes ρ decrease in Fig. 5. Correspondingly, with a fixed number of agents, the increase of tasks will increase the ρ in Fig. 6.

In Subsection 3.3, we propose a different mechanism to choose a valid partition for each agent, where the evolution times and the distance to the task are considered. The motivation or advantage of our proposed mechanism is to improve the utility of the task allocation. Here, The simulations are designed to compare the performance of the designed mechanism with the mechanism (denoted as random mechanism) in [37].

Likewise, we compare the social utility of the proposed algorithm when using the two different mechanisms for different agent scenarios and task scenarios, respectively. In simulations, we have conducted 50 times algorithms using the random mechanism. Figs. 7 and 8 show the statistical results using box-and-whisker plots, where the blue boxes indicate the results from the random mechanism. Since the designed mechanism is without randomness, there is no variance on the red curve, as Figs. 7 and 8 show.

As Figs. 7 and 8 show, the results using the designed mechanism have achieved good performance in both scenarios, that is, higher social utility. Compared with the random mechanism, the designed mechanism considers the distance to tasks when updating partitions (i.e., choosing a valid partition). From (7), the closer the distance is, the greater the utility is. Each agent chooses to form a coalition with neighbor agents with higher utility while forming a coalition, so the global utility is greater.

5. CONCLUSION

In this paper, we proposed a distributed hedonic coalition formation game method to solve the heterogeneous multi-agent task allocation problem. This problem has numerous real-world applications. However, finding the optimal solution for the studied problem is shown to be NP-hard in the literature. We designed a preference order for each agent and proved that the system could converge to a Nash-stable solution under the preference. Simulations were conducted to verify the performance of the proposed method. The proposed method had the feasibility in solving heterogeneous composition and the broader scalability in different scenarios.

In this paper, we only study the heterogeneous requirements of tasks in task allocation. However, the number of agents required by a task is also significant. Therefore, in future work, we will consider the heterogeneity of agents and the impact of the number on the allocation results.

CONFLICT OF INTEREST

The authors declare that there is no competing financial interest or personal relationship that could have appeared to influence the work reported in this paper.

REFERENCES

[1] F. Luo, Y. Chen, Z. Xu, G. Liang, Y. Zheng, and J. Qiu, "Multiagent-based cooperative control framework for microgrids' energy imbalance," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1046-1056, 2016.

[2] L. Cao, Y. Pan, H. Liang, and T. Huang, "Observer-based dynamic event-triggered control for multiagent systems with time-varying delay," *IEEE Transactions on Cybernetics*, 2022.

[3] H. Liang, L. Chen, Y. Pan, and H.-K. Lam, "Fuzzy-based robust precision consensus tracking for uncertain networked systems with cooperative-antagonistic interactions," *IEEE Transactions on Fuzzy Systems*, 2022.

[4] B. Wang, W. Chen, B. Zhang, Y. Zhao, and P. Shi, "Cooperative control-based task assignments for multiagent systems with intermittent communication," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 6697-6708, 2020.

[5] J. Lee and J. Back, "Robust distributed cooperative controller for DC microgrids with heterogeneous sources," *International Journal of Control, Automation, and Systems*, vol. 19, no. 2, pp. 736-744, 2021.

[6] G. Notomista, S. Mayya, S. Hutchinson, and M. Egerstedt, "An optimal task allocation strategy for heterogeneous multi-robot systems," *Proc. of IEEE European Control Conference*, pp. 2071-2076, 2019.

[7] H. Wu, A. Ghadami, A. E. Bayrak, J. M. Smereka, and B. I. Epureanu, "Impact of heterogeneity and risk aversion on task allocation in multi-agent teams," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7065-7072, 2021.

[8] S. Soumya and K. Guruprasad, "Model-based, distributed, and cooperative control of planar serial-link manipulators," *International Journal of Control, Automation, and Systems*, vol. 19, no. 2, pp. 850-863, 2021.

[9] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795-799, 2014.

[10] M. Aggravi, G. Sirignano, P. R. Giordano, and C. Pacchierotti, "Decentralized control of a heterogeneous human-robot team for exploration and patrolling," *IEEE Transactions on Automation Science and Engineering*, 2021.

[11] H. Ravichandar, K. Shaw, and S. Chernova, "Strata: Unified framework for task assignments in large teams of heterogeneous agents," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 2, pp. 1-25, 2020.

[12] E. Czatnecki and A. Dutta, "Hedonic coalition formation for task allocation with heterogeneous robots," *Proc. of IEEE International Conference on Systems, Man and Cybernetics*, pp. 1024-1029, 2019.

[13] Y. Emam, S. Mayya, G. Notomista, A. Bohannon, and M. Egerstedt, "Adaptive task allocation for heterogeneous multi-robot teams with evolving and unknown robot capabilities," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 7719-7725, 2020.

[14] A. Prorok, M. A. Hsieh, and V. Kumar, "The impact of diversity on optimal control policies for heterogeneous robot swarms," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 346-358, 2017.

[15] P. MacAlpine, E. Price, and P. Stone, "Scram: Scalable collision-avoiding role assignment with minimal-makespan for formational positioning," *Proc. of Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[16] R. Patel, E. Rudnick-Cohen, S. Azarm, M. Otte, H. Xu, and J. W. Herrmann, "Decentralized task allocation in multi-agent systems using a decentralized genetic algorithm," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 3770-3776, 2020.

[17] S. Choudhury, J. K. Gupta, M. J. Kochenderfer, D. Sadigh, and J. Bohg, "Dynamic multi-robot task allocation under uncertainty and temporal constraints," *Autonomous Robots*, vol. 46, no. 1, pp. 231-247, 2022.

[18] C. Wei, Z. Ji, and B. Cai, "Particle swarm optimization for cooperative multi-robot task allocation: a multi-objective approach," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2530-2537, 2020.

[19] H. ElGibreen and K. Youcef-Toumi, "Dynamic task allocation in an uncertain environment with heterogeneous multi-agents," *Autonomous Robots*, vol. 43, no. 7, pp. 1639-1664, 2019.

[20] Z. Jia, J. Yu, X. Ai, X. Xu, and D. Yang, "Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm," *Aerospace Science and Technology*, vol. 76, pp. 112-125, 2018.

[21] J. A. Adams *et al.*, "Coalition formation for task allocation: theory and algorithms," *Autonomous Agents and Multi-Agent Systems*, vol. 22, no. 2, pp. 225-248, 2011.

[22] J. K. Verma and V. Ranga, "Multi-robot coordination analysis, taxonomy, challenges and future scope," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, pp. 1-36, 2021.

[23] M. Braquet and E. Bakolas, "Greedy decentralized auction-based task allocation for multi-agent systems," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 675-680, 2021.

[24] L. An and G.-H. Yang, "Distributed optimal coordination for heterogeneous linear multiagent systems," *IEEE Transactions on Automatic Control*, vol. 67, no. 12, pp. 6850-6857, 2021.

[25] L. Vig and J. A. Adams, "Multi-robot coalition formation," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 637-649, 2006.

[26] L. Matthey, S. Berman, and V. Kumar, "Stochastic strategies for a swarm robotic assembly system," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 1953-1958, 2009.

[27] L. Zhang, H. Zhong, and S. Y. Nof, "Adaptive fuzzy collaborative task assignment for heterogeneous multirobot systems," *International Journal of Intelligent Systems*, vol. 30, no. 6, pp. 731-762, 2015.

[28] Q. Li, M. Li, B. Q. Vo, and R. Kowalczyk, "Distributed near-optimal multi-robots coordination in heterogeneous task allocation," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4309-4314, 2020.

[29] S. Wang, Y. Liu, Y. Qiu, and J. Zhou, "Consensus-based decentralized task allocation for multi-agent systems and simultaneous multi-agent tasks," *IEEE Robotics and Automation Letters*, 2022.

[30] L. Han, T. Morstyn, and M. McCulloch, "Estimation of the shapley value of a peer-to-peer energy sharing game using multi-step coalitional stratified sampling," *International Journal of Control, Automation, and Systems*, vol. 19, no. 5, pp. 1863-1872, 2021.

[31] J. J. Roldán, J. Del Cerro, and A. Barrientos, "Should we compete or should we cooperate? applying game theory to task allocation in drone swarms," *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5366-5371, 2018.

[32] R. Rădulescu, P. Mannion, D. M. Roijers, and A. Nowé, "Multi-objective multi-agent decision making: a utility-based analysis and survey," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 1, pp. 1-52, 2020.

[33] Y. Zheng, H. Zhao, and C. He, "Robust control design with optimization for uncertain mechanical systems: Fuzzy set theory and cooperative game theory," *International Journal of Control, Automation, and Systems*, vol. 20, no. 4, pp. 1377-1392, 2022.

[34] G. M. Skaltsis, H.-S. Shin, and A. Tsourdos, "A survey of task allocation techniques in MAS," *Proc. of IEEE International Conference on Unmanned Aircraft Systems*, pp. 488-497, 2021.

[35] K. Taywade, "Multi-agent reinforcement learning for decentralized coalition formation games," *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 18, pp. 15738-15739, 2021.

[36] W. Saad, Z. Han, T. Basar, M. Debbah, and A. Hjorungnes, "Hedonic coalition formation for distributed task allocation among wireless agents," *IEEE Transactions on Mobile Computing*, vol. 10, no. 9, pp. 1327-1344, 2010.

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

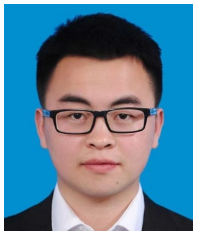
[38] E. Czarnecki and A. Dutta, "Scalable hedonic coalition formation for task allocation with heterogeneous robots," *Intelligent Service Robotics*, pp. 1-17, 2021.

[39] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495-1512, 2013.

[40] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939-954, 2004.

[41] V. Bilò, A. Fanelli, M. Flammini, G. Monaco, and L. Moscardelli, "Nash stable outcomes in fractional hedonic games: Existence, efficiency and computation," *Journal of Artificial Intelligence Research*, vol. 62, pp. 315-371, 2018.

[42] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124-143, 1996.



Lexing Wang received his B.Eng. degree in automation from Jiangsu University, Zhenjiang, China, in 2017. He is currently pursuing a Ph.D. degree in control theory and control engineering with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current research interests include path planning, collective intelligence, and multi-agent systems.

Tenghai Qiu received his B.Eng. degree in automation from Xiamen University, Xiamen, China, in 2013, and an M.Eng. degree in control theory and control engineering from Beihang University, Beijing, China, in 2016. He is currently a research assistant with the Integrated Information System Research Center, Institute of Automation, Chinese Academy of Sciences. He has been supported by the Innovation Guidance Found, Chinese Academy of Sciences, since 2020. His current research interests mainly include intelligence decision making, collective intelligence, swarm intelligence, multi-agent, and applications of unmanned autonomous systems.

Zhiqiang Pu received his B.Eng. degree in automation from Wuhan University, Wuhan, China, in 2009, and a Ph.D. degree in control theory and control engineering from Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2014. He is currently a Professor in the Integrated Information System Research Center, Institute of Automation, Chinese Academy of Sciences. He has been supported by the Talent Program of Youth Innovation Promotion Association CAS since 2017. His research interests include decision intelligence, collective intelligence, and applications of unmanned autonomous systems.

Jianqiang Yi received his B.Eng. degree in mechanical engineering from Beijing Institute of Technology, Beijing, China, in 1985, and his M.Eng. and Ph.D. degrees in automation from the Kyushu Institute of Technology, Kitakyushu, Japan, in 1989 and 1992, respectively. From 1992 to 1994, he worked as a Research Fellow with the Computer Software Development Company, Tokyo, Japan. From 1994 to 2001, He was with MYCOM, Inc., Kyoto, Japan. Since 2001, he has been a Full Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. He has authored or co-authored over 100 international journal papers and 270 international conference papers. His research interests mainly include intelligent control, adaptive control, robot system, and collective intelligence.

Jinying Zhu received his B.Eng. degree from the School of Engineering and Technology, China University of Geosciences (Beijing) in 2008, and a Ph.D. degree in dynamics and control from Peking University in 2015. He was a postdoctor in the School of Mechatronical Engineering, Beijing Institute of Technology, from April 2015 to May 2018. He is currently an Assistant Professor in the Institute of Automation, Chinese Academy of Sciences. His research interests include prosthetics, robotics, and artificial intelligence.

Wanmai Yuan received his B.Eng. degree in communication engineering from Xidian University, Xian, China, in 2014, and a Ph.D. degree in electronics and information engineering from the Harbin Institute of Technology, Harbin, China, in July 2019. He also received a Ph.D. degree in electronic and information engineering from the Hong Kong Polytechnic University, Kowloon, Hong Kong, in September 2019. From 2018 to 2019, he was a Ph.D. Visiting Student with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada. Since July 2019, he has been an Engineer with the China Academy of Electronics and Information Technology, Beijing, China. His main research interests include flocking control and formation control for UAVs.