Proceedings of the 14th Asian Control Conference
July 5-8, 2024, Dalian, China

Distributed Task Allocation with Time Constraints under Limited Communication Resources in Multi-Agent System

Weilin Li, Jing Wu, Chengnian Long

***Abstract*—In this paper, a new distributed allocation algorithm based on performance impact with collective influence (PI-CI) is proposed. Considering inefficiency of creating time slots in PI-MaxAss, two completely independent phases with different communication frequencies are designed to ensure allocation both real-time and conflict-free under limited communication. To solve the misconvergence of the baseline algorithm PI-MaxAss in allocation results, an evaluation function is put forward in consensus phase to directly determines the final attribution of repeatedly assigned tasks, guaranteeing convergence of allocation. Comparison experiments validate the performance improvement of the PI-CI algorithm over the baseline algorithm.**

***Index Terms*—Distributed task allocation, multi-agent system, limited communication resources, collective influence.**

I. Introduction

Distributed task allocation methods are essentially the study of task coordination and allocation mechanisms among multiple agents to maximise the utility value of agent cluster systems. It has a wide range of applications in search and rescue missions [1]–[3], military operations such as attack or surveillance [4], manufacturing [5], [6] and other scenarios.

Currently, the main approaches to distributed task allocation in multi-agent systems include auction and market mechanism based approaches, optimisation based approaches [7]–[11], game theory based approaches [12]–[15] and reinforcement learning approaches [16]–[18]. Compared with other algorithms, auction and market mechanism based algorithms are characterised by good decentralisation, high solving efficiency and moderate communication resource consumption [19], which received high interests from researchers.

Algorithms based on auctions and market mechanisms play an important role in distributed task planning methods for complex systems. Such algorithms simulate markets and mechanisms in economics, where members of the system make bids based on their respective task costs or utility values, and their optimisation goal is often to achieve the highest utility for the assigned tasks. For example, consensus based Bundle Algorithm (CBBA) is one of the classical algorithms. The algorithm has been proved to provide a suboptimal solution to the single-task assignment problem [20]. [21] extends the CBBA algorithm to address scenarios involving heterogeneous unmanned system teams and complex mission constraints. In [22], the concept of importance is defined for each task and the value is measured by its contribution to the local costs incurred by the vehicle, and the performance is better than CBBA. On the basis of [22], [23] proposes a new concept of Performance Impact (PI) and shortens the waiting time of the task by integrating the architecture with additional action selection methods that increase the exploratory properties. But in real-world scenarios such as rescue, we actually want more tasks to be completed rather than the shortest wait time. So [24] proposes PI-MaxAss for task allocation in distributed multi-robot systems under strict time constraints, which has a higher success rate of allocation compared to the CBBA and PI algorithms.

However, the PI-MaxAss algorithm also has some shortcomings. One is the heavy reliance on communication. In PI-MaxAss, agents try to include new tasks by creating time slots while the communication link may be extremely long and inclusion fails once the signal is lost in the process. Even if inclusion is successfully carried out, frequent iteration also increases the power consumption of the agents. The other problem is that two or more vehicles occasionally get stuck in an infinite loop of exchanging the same tasks. As the number of tasks increases, the allocation results may not converge.

Therefore, in order to maximize task completion rate in multi-agent system under limited communication resources and guarantee the convergence of allocation, this paper proposes a new distributed allocation algorithm based on performance impact with collective influence (PI-CI). The main contributions are summarized as follows:

- An algorithm consisting of two completely independent phases with different communication frequencies is proposed. By iterating inclusion phase at high frequency and consensus phase at low frequency asynchronously, agents can quickly respond to emerging tasks while ensuring the results do not conflict.
- The concept of Displacement Performance Impact

Weilin Li, Jing Wu and Chengnian Long are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China.

*This work was supported by the National Natural Science Foundation of China(62273236, 62136006 and 62073215), and Key R&D projects in Hainan Province ZDYF2024GXJS009.

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.

(DPI) is introduced to make agents able to substitute worthier tasks for those with less reward without going through long communication links to create time slots. It reduces dependence on communication and increases the success rate of allocation.
- An evaluation function considering collective influence (CI) is provided in consensus phase to measure the receptivity of the agent to the task. Neighbour agents only need to compare the evaluation function once to determine attribution of tasks, thus ensuring convergence of allocation.

The rest of the paper is organized as follows. Section II describes the problem formally. Section III describes the proposed algorithm. Section IV shows the results of the experiment and gives corresponding analyses. Section V gives conclusion.

II. Problem Formulation

Consider a distributed multi-agent system with m agents for real-time assignment of n tasks with time window constraints. Tasks will appear randomly anywhere in the region at unpredictable times. Only when a task sends out a request, its relevant information can be received by the agents. Tasks are time-constrained, with corresponding rewards for completing them in valid time. The agents have communication constraints, which means the agents can only communicate with its neighbor agents and receive task requests within a certain range. Agents use a single-hop communication mechanism, that is, they can only store information and cannot forward it.

In addition to the above problem description, the following assumptions need to be satisfied:

- The speed of an agent is constant when travelling to the site of task.
- Time to complete the mission is negligible compared to the time spent on travelling to the task.
- Agents do not need to return to the starting position in the end.

To solve the above problem, a rigorous mathematical description was built. We first give some definitions to the properties of the agents and the tasks. The sets of agents and tasks are defined as $\mathcal{A} = \{a_1, a_2, \cdots, a_m\}$ and $\mathcal{T} = \{t_1, t_2, \cdots, t_n\}$, respectively. According to the problem requirement, an agent can communicate with other agents at distances less than D. Tasks assigned to the agent a_i are arranged in an order as $\mathbf{a}_i$. The time period between the appearance of a task request and the deadline for completion of the task is defined as valid time of the task, which is notated by $\mathcal{V}_k$.

Then optimisation objective function and constraints of the problem can be formulated as

$$J = \max \left\{ \frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{A}|} \sum_{k=1}^{|\mathcal{T}|} r(a_i, t_k) \mathbb{1}_{\mathbf{a}_i}(t_k) \mathbb{1}_{\mathcal{V}_k}\left(c_k(\mathbf{a}_i)\right) \right\} \quad (1)$$

TABLE I
Notation Definitions

Notation	Definition
$\mathcal{A} = \{a_1, a_2, \cdots, a_m\}$	Set of m agents
$\mathcal{R}_i$	Region of acceptable tasks of agent a_i
D	The maximum range within which an agent can exchange information
v_i	Velocity of agent a_i
$\mathbf{a}_i$	Ordered task allocation of the i^{th} agent a_i
$\mathcal{T} = \{t_1, t_2, \cdots, t_n\}$	Set of n tasks
$r(a_i, t_k)$	Inherent reward determined by task attributes for agent a_i to complete task t_k
$\mathbf{a}_i \oplus_l t_k$	$\mathbf{a}_i$ inserted by task t_k at position l
$u(\mathbf{a}_i, t_k)$	The utility function of task t_k in $\mathbf{a}_i$
$\omega^{\dagger}(\mathbf{a}_i, t_k)$	The Insertion Performance Impact (IPI) of a task t_k in $\mathbf{a}_i$
$c_k(\mathbf{a}_i)$	The time cost of task t_k in $\mathbf{a}_i$
$\mathcal{V}_k$	Valid time of the k^{th} task t_k
$\mathbf{a}_i \odot_l t_k$	$\mathbf{a}_i$ displaced by task t_k at position l
$\omega^{\natural}(\mathbf{a}_i, t_k)$	The Displacement Performance Impact (DPI) of a task t_k in $\mathbf{a}_i$
$\mathbf{a}_i^{\diamond}$	Ordered unfinished task allocation of the i^{th} agent a_i
$\mathrm{CI}_1(i)$	Collective influence of the i^{th} agent a_i
$\phi_i(t_k)$	Evaluation functions for task t_k by agent a_i in the consensus phase

subject to:

$$\sum_{i=1}^{|\mathcal{A}|} \mathbb{1}_{\mathbf{a}_i}(t_k) \leqslant 1, \forall t_k \in \mathcal{T}. \quad (2)$$

For the sake of convenience, all the important notations and their definitions in this paper are given in Tab. I. Our objective is to find the optimal real-time task allocation that maximizes the total reward. In particular, when the reward for each task is the same, the goal translates into maximising the completion rate of the tasks.

III. Algorithm

This section describes the framework and the specific procedure of the proposed algorithm and analyse the significance of each improvement.

A. Architecture

The architecture of PI-CI is shown in Fig. 1. PI-CI is composed of inclusion phase and consensus phase, which are independent and can be carried out asynchronously. The inclusion phase is performed frequently to search for assignable tasks. The consensus phase in which agents communicate with others is operated at a low frequency to make sure the allocation is conflict-free and convergent.

B. Inclusion Phase

The first phase of the algorithm is the task inclusion phase, where the agents receive task requests within a specific range, and there is no need for extra communication between the agents in this phase.

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.

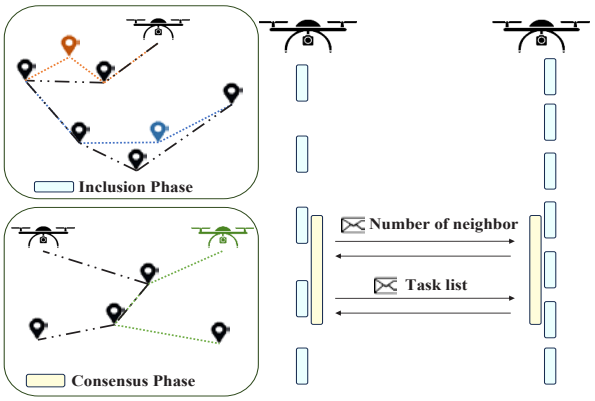

Fig. 1. Architecture of PI-CI.

As a basis for subsequent calculations, we define the function that describes the utility of task t_k in $\mathbf{a}_i$ as

$$u(\mathbf{a}_i, t_k) = \frac{r(a_i, t_k)\mathbb{1}_{\mathcal{V}_k}\left(c_k(\mathbf{a}_i)\right)}{[c_k(\mathbf{a}_i) - c_{k-1}(\mathbf{a}_i)]\left\{1 + e^{\mu[(c_k(\mathbf{a}_i) - c_{k-1}(\mathbf{a}_i))v_i - D]}\right\}}, \tag{3}$$

where μ is the coefficient to vary the distance preference.

Remark III.1. *The utility function is designed to decay additionally with distance, so that the agent will prefer to add proximal tasks to its allocation list.*

Tasks enter the allocation lists of agents in two ways: insertion or displacement, corresponding to the Insertion Performance Impact (IPI) and the Displacement Performance Impact (DPI). IPI is defined as

$$\omega^{\dagger}(\mathbf{a}_i, t_k) = \max_{l=1}^{|\mathbf{a}_i|+1} \left\{\omega_{\text{ins}}(\mathbf{a}_i, t_k, l)\right\}, \tag{4}$$

where

$$\omega_{\text{ins}}(\mathbf{a}_i, t_k, l) = \sum_{t_q \in \mathbf{a}_i \oplus_l t_k} u(\mathbf{a}_i \oplus_l t_k, t_q) - \sum_{t_q \in \mathbf{a}_i} u(\mathbf{a}_i, t_q). \tag{5}$$

Equation (5) computes the insertion utility of t_k at each position l in a_i, where $\mathbf{a}_i \oplus_l t_k$ is $\mathbf{a}_i$ inserted by task t_k at position l. DPI is defined as

$$\omega^{\natural}(\mathbf{a}_i, t_k) = \max_{l=1}^{|\mathbf{a}_i|} \left\{\omega_{\text{dis}}(\mathbf{a}_i, t_k, l)\right\}, \tag{6}$$

where

$$\omega_{\text{dis}}(\mathbf{a}_i, t_k, l) = \sum_{t_q \in \mathbf{a}_i \odot_l t_k} u(\mathbf{a}_i \odot_l t_k, t_q) - \sum_{t_q \in \mathbf{a}_i} u(\mathbf{a}_i, t_q). \tag{7}$$

Equation (7) computes the insertion utility of t_k at each position l in a_i, where $\mathbf{a}_i \odot_l t_k$ is $\mathbf{a}_i$ displaced by task t_k at position l.

Remark III.2. *The DPI is designed to solve the task replacement challenge in the PI-MaxAss algorithm. Agents no longer need to communicate frequently with the rest of the agents to create time slots for task replacement.*

For each task request, the agent calculates the IPI and DPI of the task. If the values are both nonpositive, it means that this task is not worthy of being allocated to the agent. Otherwise, this task is assigned to the agent in the way according to the higher performance impact. A more specific procedure is shown in the Algorithm 1.

Algorithm 1 Inclusion Algorithm

for $t_k \in \mathcal{R}_i \backslash \bigcup_{j \neq i} \mathcal{R}_j$ **do**
  Calculate IPI of t_k according to (4)
  Calculate DPI of t_k according to (6)
  if $\omega^{\dagger}(\mathbf{a}_i, t_k) > \omega^{\natural}(\mathbf{a}_i, t_k)$ and $\omega^{\dagger}(\mathbf{a}_i, t_k) > 0$ **then**
    $l_m \leftarrow \arg\max_l \omega_{\text{ins}}(\mathbf{a}_i, t_k, l)$
    $\mathbf{a}_i \leftarrow \mathbf{a}_i \oplus_l t_k$
  else if $\omega^{\natural}(\mathbf{a}_i, t_k) > \omega^{\dagger}(\mathbf{a}_i, t_k)$ and $\omega^{\natural}(\mathbf{a}_i, t_k) > 0$ **then**
    $l_m \leftarrow \arg\max_l \omega_{\text{dis}}(\mathbf{a}_i, t_k, l)$
    $\mathbf{a}_i \leftarrow \mathbf{a}_i \odot_l t_k$
  end if
end for
for $t_k \in \mathcal{R}_i \cap \bigcup_{j \neq i} \mathcal{R}_j$ **do**
  Calculate IPI of t_k according to (4)
  Calculate DPI of t_k according to (6)
  if $\omega^{\dagger}(\mathbf{a}_i, t_k) > \omega^{\natural}(\mathbf{a}_i, t_k)$ and $\omega^{\dagger}(\mathbf{a}_i, t_k) > 0$ **then**
    $l_m \leftarrow \arg\max_l \omega_{\text{ins}}(\mathbf{a}_i, t_k, l)$
    $\mathbf{a}_i \leftarrow \mathbf{a}_i \oplus_l t_k$
  else if $\omega^{\natural}(\mathbf{a}_i, t_k) > \omega^{\dagger}(\mathbf{a}_i, t_k)$ and $\omega^{\natural}(\mathbf{a}_i, t_k) > 0$ **then**
    $l_m \leftarrow \arg\max_l \omega_{\text{dis}}(\mathbf{a}_i, t_k, l)$
    $\mathbf{a}_i \leftarrow \mathbf{a}_i \odot_l t_k$
  end if
end for

In the task inclusion phase, the computational process is the same for each task. But the order in which tasks are included is determined by where they are located. Agents will prioritise the calculation of performance impact for tasks in regions that do not overlap with other agents. This approach has a distinct advantage over the PI-MaxAss algorithm in that when a task is in the edge region and can only be completed in time by the only approaching agent, the agent will give priority to this task into its allocation list. By doing so, the need to go through long communication links to create time slots to include edge tasks is avoided, thus reducing the frequency of communications and guaranteeing the task completion rate.

C. Consensus Phase

After the task inclusion phase, the same task may be assigned to multiple agents, which can result in a waste of resources, and therefore a consensus on task allocation through inter-agent communication is also required.

Communication between agents is limited by the communication distance and different communication protocols on the one hand, and by their own energy constraints

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.

on the other. Therefore, this paper proposes a new consensus algorithm to make the system's task allocation achieve a better consensus result under the premise of reducing the data communication frequency as much as possible.

Some nodes in the network do not have a high degree, but may have a large collective influence on the network [25]. Therefore, We introduce the concept of CI in this phase and design a consensus evaluation function:

$$\phi_i(t_k) = \frac{\max\left\{\omega^{\dagger}(\mathbf{a}_i, t_k), \omega^{\natural}(\mathbf{a}_i, t_k)\right\}}{|\mathbf{a}_i^{\diamond}| \cdot \mathrm{CI}_1(i) + \epsilon}, \tag{8}$$

where ϵ is a tiny positive constant, and $\mathrm{CI}_\ell(i)$ is defined as

$$\mathrm{CI}_\ell(i) = (k_i - 1) \sum_{j \in \partial \mathrm{Ball}(i,\ell)} (k_j - 1), \tag{9}$$

where k_i is the number of neighbouring nodes of agent a_i.

In the consensus phase, all communicable agents exchange the number of neighbouring nodes and repeatedly assigned tasks with each other. After updating the number of neighbouring nodes, agents communicates again to exchange this value with each other and calculate $\mathrm{CI}_1(i)$ based on their own stored data. Finally agents exchange values of the evaluation function for the repeatedly assigned tasks. The agent with the smaller value removes operating repeatedly assigned task from its list of allocated tasks, thus achieving consistency in allocation. A more specific procedure is shown in the Algorithm 2.

Algorithm 2 Consensus Algorithm

```
for j ∈ ∂Ball(i, 1) do
  Communicate with neighbouring nodes a_j
  a_j^◇ ← task list of neighbourhood
  k_i ← number of neighbour nodes updated
  Communicate with neighbouring nodes a_j
  Calculate CI_1(i) according to (9)
  for t_k ∈ a_i^◇ ∩ a_j^◇ do
    Calculate φ_i(t_k) and φ_j(t_k) according to (8)
    if φ_i(t_k) ⩽ φ_j(t_k) then
      a_i^◇ ← a_i^◇ \ {t_k}
    end if
  end for
end for
```

Remark III.3. *The advantages of the proposed consensus algorithm over the PI-MaxAss process of reaching an allocation consensus are mainly in two aspects: (1) convergence can be guaranteed; (2) the load of agents is more balanced.*

First we elaborate on the convergence of PI-CI algorithm. In PI-MaxAss algorithm, two or more vehicles occasionally get stuck in an infinite loop of exchanging the same tasks. However, in PI-CI algorithm, any task t_k included by more than one agents must be assigned to the agent with the largest value of evaluation function in the end. And as long as there is no new task request, agent that finally obtains the task will not discard the task, thus ensuring the convergence of the allocation results.

In terms of load, the PI-CI algorithm assigns some of the tasks of busy agents to idle agents by reducing the weights of the agents that are more influential in the collective. At the same time, agents with more pending tasks are less competitive for repeatedly assigned tasks. Therefore, PI-CI can make the load of the agents relatively balanced, thus increasing the rate of completed tasks and total reward.

TABLE II
Parameter value of development experiment

Parameter	Value
Number of agents	12
Number of tasks	12, 24, 50, 120, 180, 240, 300, 360
Region of tasks	10 000m × 10 000m × 0m ground space
Radius for receiving tasks	4 000m
Velocity of agents	30m/s
Communication range	4 000m
Task valid duration	50s ∼ 200s
Time duration	300s

IV. Experiment Results

This section shows the results of several simulation experiments and analyses the results.

A. Development Experiment

Development experiments are mainly used to test the impact of various factors on the performance of the proposed algorithm. In order to facilitate comparisons with other algorithms in subsequent experiments, the evaluation metrics need to be harmonised. In view of the fact that some algorithms do not introduce the concept of reward, the reward for each task within the algorithm was set to 1 in the experiments, and thus the evaluation metric degenerated to the completion rate of the task. We focus on two factors, the task-to-agent ratio p and the starting position of agents.

To test the performance of the algorithm under different number ratios, a total of eight sets of experiments are set up, with the number of tasks in order of 12, 24, 50, 120, 180, 240, 300 and 360. Repeat the experiment 30 times for each group. Parameters about agents and tasks are shown in Tab.II.

The start positions of the agents are set to random and specified positions, respectively. Starting from a specified location simulates the real situation where groups of agents collectively start from a specific workstation. In order to ensure coverage of the area, the experiment set four workstations located at one quarter and three quarters of the horizontal and vertical coordinates. That is, the 12 agents are divided into 4 groups of 3 each, starting from [2500, 2500], [2500, 7500], [7500, 2500] and [7500, 7500], respectively.

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.

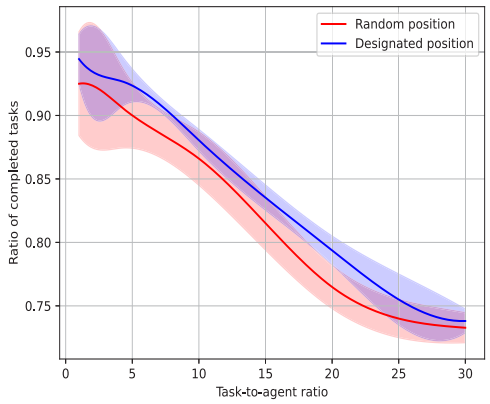

Fig. 2. Ratio of completed tasks with different task-to-agent ratio and from different starting positions.

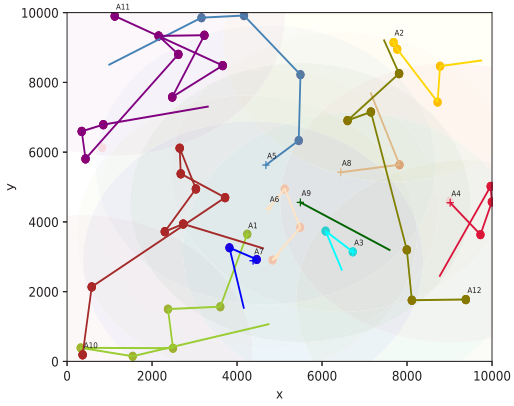

Fig. 3. Task allocation with task-to-agent $p = 4.2$ from random starting positions.

Means and standard deviations of task completion rates are obtained for each of the eight sets of experiments and fitted by Bessel curves. The results are presented in Fig. 2. The transparent part of the figure indicates the standard deviation.

It can be seen that the blue curve is consistently slightly higher than the red curve, indicating that the starting position has a small effect on the task completion rate. One of the possible reasons for this phenomenon is that the workstation locations are set to ensure that the whole area is covered that cannot be satisfied by random starting points. But the trend between the two is the same and the difference is not that big. So in order to verify the robustness of the algorithms, comparison experiments in next subsection all use random starting points.

As with the expected results, the task completion rate gradually decreases as the task-to-agent ratio p increases. It is worth noting that the curve begins to flatten when the ratio is greater than 25, which may be due to the fact that the tasks are too dense resulting in short distances between tasks, and the aforementioned assumption about ignoring the time required to complete a task is no longer satisfied. Since ratios greater than 25 are undoubtedly difficult to occur in practical applications and have little practical value, this case will not be analysed.

To provide a clearer explanation of how each experiment was performed, allocation results with 12 agents and 50 tasks are shown as an example in Fig. 3. The transparent regions are the communication distance limits for agents. The assigned task is displayed in the colour corresponding to the agent. Translucent dots represent tasks that are ultimately unfinished.

B. Comparison Experiment

In order to verify the effectiveness of the proposed algorithm, we need to compare the performance of the PI-CI algorithm with other algorithms.

On the one hand, we compare the PI-CI algorithm with the baseline algorithm PI-MaxAss to verify the effectiveness of the improvement. On the other hand, since the PI-CI algorithm is based on market and auction mechanism, we compare it with the DHBA which is an optimisation-based excellent task allocation algorithm so as to evaluate the algorithm performance horizontally.

In comparison experiment, the values of the parameters are consistent with Tab.II, except that the number of tasks was set to 50 ($p = 4.2$) and 180 ($p = 15$), corresponding to low task loads and high task loads, respectively. Starting points of agents are random.

The comparison results are shown as box plots in Fig. 4 and Fig. 5. The dots in the graph represent outliers in the box plot. FC and WC represent the connectivity of network topology. FC means agents are fully connected, which means distance is the only constraint on communication between agents. WC means agents are weakly connected, which means even if two agents are close, they may not be able to communicate due to different communication protocols, etc. From a graph-theoretic point of view, the difference between FC and WC can be characterised by the second smallest eigenvalue of the Laplace matrix, λ_2. FC corresponds to a large λ_2 while WC corresponds to a λ_2 close to 0.

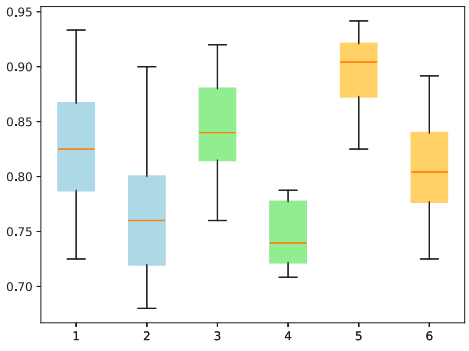

Fig. 4. Comparison of DHBA, PI-MaxAss and PI-CI under fully connected and weakly connected networks with the task-to-agent ratio $p = 4.2$.

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.

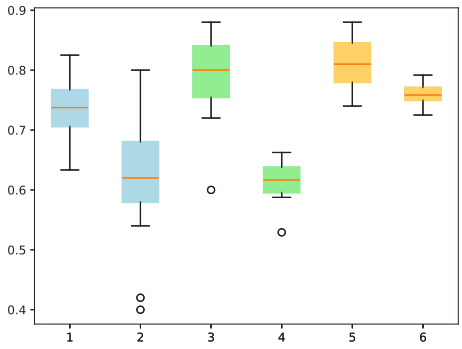

Fig. 5. Comparison of DHBA, PI-MaxAss and PI-CI under fully connected and weakly connected networks with the task-to-agent ratio $p = 15$.

Comparison of Fig. 4 and Fig. 5 shows that the performance of the DHBA algorithm decreases significantly at high task-to-agent ratio, reflecting the fact that the optimisation-based algorithms are computationally demanding, and that the computation time extends considerably when the number of tasks increases. In contrast, algorithms based on market and auction mechanisms are less affected by an increase in the number of tasks.

As can be seen from Figs. 4 and 5 separately, the PI-CI algorithm has the highest task completion rate regardless of the network topology. Moreover, reducing the communication connectivity has a significantly lower impact on PI-CI compared with PI-MaxAss.

V. Conclusion

In this paper, we design a new distributed task allocation algorithm based on the PI-MaxAss for multi-agents systems with limited communication resources. The algorithm is refined into two completely separate phases which can be performed asynchronously to increase efficiency. The concepts of IPI and DPI are introduced to reduce the dependence on communication. By introducing CI into the consensus evaluation function, the results of tasking are bound to converge and the load of agents is relatively balanced. Comparison experiments of DHBA, PI-MaxAss and PI-CI under different communication connectivity task-to-agent ratios demonstrate the effectiveness of our proposed PI-CI algorithm.

References

[1] Pilloni V, Ning H, Atzori L. Task allocation among connected devices: Requirements, approaches, and challenges. *IEEE Internet of Things Journal*, 2021, 9(2): 1009-1023.

[2] Khani M, Ahmadi A, Hajary H. Distributed Task Allocation in Multi-agent Environments using Cellular Learning Automata. *Soft Computing*, 2019, 23(4): 1199-1218.

[3] Hooshangi N, Alesheikh A A. Developing An Agent-based Simulation System for Post-earthquake Operations in Uncertainty Conditions: A Proposed Method for Collaboration Among Agents. *ISPRS International Journal of Geo-Information*, 2018, 7(1): 27.

[4] Kurdi H A, Aloboud E, Alalwan M, et al. Autonomous Task Allocation for Multi-UAV Systems Based on the Locust Elastic Behavior. *Applied Soft Computing*, 2018, 71: 110-126.

[5] Hanke J, Kosak O, Schiendorfer A, et al. Self-organized Resource Allocation for Reconfigurable Robot Ensembles. In *2018 IEEE 12th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, 2018: 110-119.

[6] Yussupova N, Rizvanov D. Decision-making Support in Resource Management in Manufacturing Scheduling. *IFAC-Papers On Line*, 2018, 51(30): 544-547.

[7] Farinelli A, Rogers A, Jennings N R. Agent-based Decentralised Coordination for Sensor Networks using the Max-Sum Algorithm. *Autonomous Agents and Multi-agent Systems*, 2014, 28(3): 337-380.

[8] Qin B, Zhang D, Tang S, et al. Distributed grouping cooperative dynamic task assignment method of UAV swarm. *Applied Sciences*, 2022, 12(6): 2865.

[9] Ramchurn S, Farinelli A, Macarthur K, et al. Decentralised Coordination in Robocup Rescue. *The Computer Journal*, 2009.

[10] Macarthur K, Stranders R, Ramchurn S, et al. A Distributed Anytime Algorithm for Dynamic Task Allocation in Multi-agent Systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011.

[11] Ismail S, Sun L. Decentralized Hungarian-based Approach for Fast and Scalable Task Allocation. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017: 23-28.

[12] Bayram H, Bozma H I. Coalition Formation Games for Dynamic Multirobot Tasks. *The International Journal of Robotics Research*, 2016, 35(5): 514-527.

[13] Afghah F, Zaeri-Amirani M, Razi A, et al. A Coalition Formation Approach to Coordinated Task Allocation in Heterogeneous UAV Networks. In *2018 Annual American Control Conference (ACC)*, 2018: 5968-5975.

[14] Muhuri P K, Rauniyar A. Immigrants Based Adaptive Genetic Algorithms for Task Allocation in Multi-robot Systems. *International Journal of Computational Intelligence and Applications*, 2017, 16(04): 1750025.

[15] Agarwal Martin J G, Muros F J, Maestre J M, et al. Multi-robot task allocation clustering based on game theory. *Robotics and Autonomous Systems*, 2023, 161: 104314.

[16] Arulkumaran K, Deisenroth M P, Brundage M, et al. Deep Reinforcement Learning: A Brief Survey. *IEEE Signal Processing Magazine*, 2017, 34(6): 26-38.

[17] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level Control through Deep Reinforcement Learning. *Nature*, 2015, 518(7540): 529-533.

[18] Bae H, Kim G, Kim J, et al. Multi-robot Path Planning Method using Reinforcement Learning. *Applied Sciences*, 2019, 9(15): 3057.

[19] Mosteo A R, Montano L. A Survey of Multi-robot Task Allocation. *Instituto de Investigacin en Ingenierla de Aragn (I3A)*, Tech. Rep, 2010.

[20] Choi H-L, Brunet L, How J P. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 2009, 25(4): 912–926.

[21] Choi H-L, Whitten A K, How J P. Decentralized Task Allocation for Heterogeneous Teams with Cooperation Constraints. In *Proceedings of the 2010 American Control Conference*, 2010: 3057-3062.

[22] Zhao W, Meng Q, Chung P W H. A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario. *IEEE transactions on cybernetics*, 2015, 46(4): 902-915.

[23] Whitbrook A, Meng Q, Chung P W H. A novel distributed scheduling algorithm for time-critical multi-agent systems. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015: 6451-6458.

[24] Turner J, Meng Q, Schaefer G, et al. Distributed Task Rescheduling with Time Constraints for the Optimization of Total Task Allocations in a Multirobot System. *IEEE Transactions on Cybernetics*, 2017, 48(9): 2583-2597.

[25] Morone F, Makse H A. Influence maximization in complex networks through optimal percolation. *Nature*, 2015, 524(7563): 65-68.

Authorized licensed use limited to: BEIHANG UNIVERSITY. Downloaded on June 05,2025 at 03:15:56 UTC from IEEE Xplore. Restrictions apply.