ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume 29, Number 3, 2026, 249–260
doi: 10.59277/ROMJIST.2026.3.04

Toward a Nash Equilibrium for Multi-agent Dynamic Task Assignment via Multi-stage Distributed Potential Game

Huawei HU[1,2], Yuxing XING[2], Jie WU[2], Jie CHEN[3,*], and Radu-Emil PRECUP[4,5]

[1]Chengdu NewPro Technology Co.,Ltd, 610000 Chengdu, China
[2]School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi, 214122 Jiangsu, China
[3]Department of Automation, University of Science and Technology of China (USTC), Hefei, 230026 Anhui, China
[4]Department of Automation and Applied Informatics, Politehnica University of Timisoara, 300223 Timisoara, Romania
[5]Center for Fundamental and Advanced Technical Research, Romanian Academy – Timisoara Branch, 300223 Timisoara, Romania

Email: huawei.hu@newprotech.com.cn, 6233115001@stu.jiangnan.edu.cn, jiewu20@jiangnan.edu.cn, chenjie2025@ustc.edu.cn*, radu.precup@upt.ro
* Corresponding author

Abstract. Dynamic task assignment in multi-agent systems aims to distribute limited agent resources among time-varying tasks to improve operational efficiency and reduce completion time. This paper develops a game-theoretic distributed learning framework for dynamic task assignment by modeling the problem as an exact potential game that aligns individual agent utilities with a global performance objective. The task dynamics capture time-varying task loads under fixed agent execution capacities, and an analysis of optimal static assignment characterizes completion-time minimization. Based on this potential game formulation, a synchronous distributed learning algorithm is proposed that utilizes the potential function as a Lyapunov measure, ensuring monotonic improvement and convergence to a pure-strategy Nash equilibrium without centralized coordination. The algorithm relies only on local information and supports parallel updates, enabling scalability and adaptability in dynamic environments. Numerical simulations demonstrate that the proposed method achieves improved convergence behavior and higher task completion efficiency compared with representative baseline algorithms.

Key-words: Distributed synchronous optimization; dynamic task assignment; Nash equilibrium; potential game.

1. Introduction

Task assignment in multi-agent systems distributes limited resources among dynamic tasks to improve efficiency and reduce completion time [1]. Effective strategies are essential for coordinated operation, since improper assignment may lead to underutilization and inefficient execution [2, 3]. The single-task agent and multi-agent task (ST-MR) problem, where each agent executes at most one task while tasks may require multiple agents, represents a fundamental formulation [4, 5]. Because task assignment is NP-hard and scales with system size, scalable and efficient methods are required for practical deployment [6, 7].

Centralized optimization achieves high-quality solutions but suffers from limited scalability and single points of failure in dynamic environments [8, 9]. Distributed optimization enables agents to make decisions using local information and limited communication, improving scalability and robustness. However, purely heuristic or locally optimal decisions may fail to achieve system-wide optimality when coordination is insufficients [10, 11]. Those challenges motivate game-theoretic and distributed learning approaches that combine decentralized decision-making with convergence guarantees [12, 13].

Game theory provides a framework for modeling interactive decision-making in multi-agent systems and assignment problems [14, 15]. By treating task assignment as a strategic game, agents optimize individual utilities while interacting with others [16]. Potential games offer a structure in which a global potential function reflects system performance, enabling distributed learning to converge toward equilibrium solutions aligned with collective objectives [17]. Although potential game theory has been effective for assignment, most studies focus on static environments, whereas dynamic task assignment with varying tasks and agents remains less explored [18, 19].

To address those challenges, this paper develops a game-theoretic distributed learning framework for dynamic task assignment. The problem is formulated as an exact potential game linking individual utilities with the global objective. A synchronous learning algorithm exploits the potential function as a Lyapunov measure, guaranteeing monotonic improvement and convergence to a pure-strategy Nash equilibrium without centralized coordination. The method operates in a distributed manner and adapts to dynamic environments, providing scalable and robust task assignment

The main contributions of this paper are summarized as follows: 1) The dynamic task assignment problem is formulated as an exact potential game and prove the existence of a pure-strategy Nash equilibrium. The potential function serves as a global performance metric, enabling distributed optimization of task assignments. 2) A synchronous distributed learning algorithm is proposed to solve the assignment problem. The algorithm exploits the monotonic properties of the potential function and guarantees convergence to equilibrium under finite iterations. 3) Numerical experiments demonstrate the effectiveness of the proposed method in dynamic task environments. Simulation results show superior convergence performance and task completion efficiency compared with representative baseline algorithms.

The remainder of this paper is organized as follows. Section 2 introduces the dynamic task model and problem formulation. Section 3 presents the optimal static task assignment framework. Section 4 presents dynamic task assignment and game model. Section 5 develops a distributed learning algorithm for dynamic task assignment. Section 6 provides numerical experiments and performance analysis. In the end, the entire paper is summarized.

2. Task Model and Problem Formulation

2.1. Dynamic task model

Denote by $s^0 > 0$ the initial task load and by s' the load increment per unit time Δt. Without agent execution, the task load evolves as [1]:

$$s^t = s^0 + s't, \tag{1}$$

where $s' \geq 0$ (the load remains constant if $s' = 0$). Let β denote the execution capacity of an agent, representing the load completed in unit time. Under agent execution, the task load becomes [1]:

$$s^t = s^0 + (s' - \beta)t. \tag{2}$$

If $s' \geq \beta$, then $\lim_{t\to\infty} s^t > 0$ and the task cannot be completed; otherwise, completion is achievable.

Consider a system with N_A agents and N_T tasks, whose sets are $V_A = \{1, \cdots, N_A\}$ and $V_T = \{1, \cdots, N_T\}$, respectively. Let Λ_j denote the set of agents assigned to task j. The total execution capacity on task j is

$$\beta_j^\Sigma = \sum\nolimits_{i\in\Lambda_j} \beta_i, \tag{3}$$

where β_i is the execution capacity of agent i. From (2), the dynamic task model is

$$s_j^t = s_j^0 + (s_j' - \beta_j^\Sigma)t, \quad j = 1, \cdots, N_T, \tag{4}$$

where s_j^t denotes the load of task j at time t. Each agent executes at most one task in a unit time.

Task dynamics are governed by agent assignment; if $\beta_j^\Sigma > s_j'$, task j completes in finite time with shorter completion time for larger β_j^Σ. The model therefore captures both single-task and multi-task dynamic scenarios.

2.2. Problem formulation

The multi-agent dynamic task assignment problem allocates limited agent capacities to multiple tasks with the objective of minimizing overall completion time. Let T_j denote the completion time of task j; the system completion time is determined by the slowest task [1]:

$$T_{final} = \max\{T_1, \cdots, T_{N_T}\}. \tag{5}$$

The assignment problem is formulated as [1]:

$$\min_{\beta_1^\Sigma,\cdots,\beta_{N_T}^\Sigma} T_{final}, \quad \text{s.t.} \sum\nolimits_{j=1}^{N_T} \beta_j^\Sigma = C,\ \beta_j^\Sigma > 0,\ \forall j \in V_T, \tag{6}$$

where C denotes the total execution capacity and β_j^Σ takes discrete values.

Table 1. Notations used in the dynamic multi-agent task assignment model

Notations	Definitions
N_A	the number of agents
N_T	the number of tasks
V_A	the set of agents
V_T	the set of tasks
s_j^t	the load of task j at time t
Δt	the unit time
β_i	the execution capacity of agent i
β_j^Σ	the total execution capacity on task j
Λ_j	the set of agents working on task j
s_j'	the load increment of task j in unit time
T_j	the completion time of task j
T_{final}	the final completion time of all tasks
C	the total execution capacity of all agents
$R_j(\beta_j^\Sigma)$	the reward of task j
$\lvert \cdot \rvert$	the cardinality of a set
$[t, \Delta t)$	the tth time stage
X^t	the strategy space
x_i^t	the strategy of player i at time t
x^t	the strategy profile at time t
F	the set of utility functions
f_i	the utility function of player i
f	the global objective function
φ	the potential function

3. Optimal Static Task Assignment

In this section, the following assumptions are first given.
i) The travel time of all agents is zero;
ii) The execution capacity of agents worked at each task is allowed to take continuous values;
iii) For each task $j \in V_T$, the execution capacity β_j^Σ satisfies $\beta_j^\Sigma > s_j'$.

From (4), the completion time of task j is

$$T_j = \frac{s_j^0}{\beta_j^\Sigma - s_j'}, \tag{7}$$

which decreases with β_j^Σ. Since $\beta_j^\Sigma > s_j'$, it results that $T_j \in (0, +\infty)$. Lemma 1 in [1] shows that optimal assignment equalizes completion times.

The reward for task j is defined as

$$R_j(\beta_j^\Sigma) = -e^{(s_j' - \beta_j^\Sigma)/s_j^0} = -e^{-1/T_j(\beta_j^\Sigma)}, \tag{8}$$

with $R_j(\beta_j^\Sigma) \in (-1, 0)$. Increasing β_j^Σ reduces T_j and increases R_j, so minimizing completion time is equivalent to maximizing reward.

The total reward is

$$R(\beta_1^\Sigma, \cdots, \beta_{N_T}^\Sigma) = \sum_{j=1}^{N_T} R_j(\beta_j^\Sigma) = -\sum_{j=1}^{N_T} e^{(s_j' - \beta_j^\Sigma)/s_j^0} = -\sum_{j=1}^{N_T} e^{-1/T_j(\beta_j^\Sigma)}. \tag{9}$$

Theorem 1. If $\beta^{\Sigma^*} = (\beta_1^{\Sigma^*}, \cdots, \beta_{N_T}^{\Sigma^*})$ is the optimal solution to the following optimization problem

$$\begin{aligned}
&\max_{\beta_1^{\Sigma},\cdots,\beta_{N_T}^{\Sigma}} R(\beta_1^{\Sigma},\cdots,\beta_{N_T}^{\Sigma}) \\
&s.t. \beta_1^{\Sigma}+\cdots+\beta_{N_T}^{\Sigma}=\sum\nolimits_{j=1}^{N_T}\beta_j=C \\
&\beta_j^{\Sigma}>0, \forall j\in V_T.
\end{aligned} \tag{10}$$

then, β^{Σ^*} is also the optimal solution to the optimization problem (6).

Proof: Since $(1/e)^{(\beta_j^{\Sigma}-s'_j)/s_j^0}\in(0,1)$, by geometric mean inequality,

$$\sum\nolimits_{j=1}^{N_T} e^{(s'_j-\beta_j^{\Sigma})/s_j^0}\geq N_T \sqrt[N_T]{\prod\nolimits_{j=1}^{N_T} e^{(s'_j-\beta_j^{\Sigma})/s_j^0}},$$

with equality iff

$$\frac{s_1^0}{\beta_1^{\Sigma}-s'_1}=\cdots=\frac{s_{N_T}^0}{\beta_{N_T}^{\Sigma}-s'_{N_T}}. \tag{11}$$

Under (11), all terms in the sum are identical, so $R(\beta_1^{\Sigma},\cdots,\beta_{N_T}^{\Sigma})$ attains its maximum and

$$T_1(\beta_1^{\Sigma})=\cdots=T_{N_T}(\beta_{N_T}^{\Sigma}).$$

By Lemma 1 [1], this equal-completion solution is optimal for (6). Theorem 1 holds. ■

Since$\beta^{\Sigma^*}=(\beta_1^{\Sigma^*},\cdots,\beta_{N_T}^{\Sigma^*})$ solves (6) and (10), combining with (11) gives

$$\frac{\beta_1^{\Sigma^*}-s'_1}{s_1^0}=\cdots=\frac{\beta_{N_T}^{\Sigma^*}-s'_{N_T}}{s_{N_T}^0}=r, \tag{12}$$

where r is the constant ratio. Then, it results that

$$\beta_j^{\Sigma^*}=rs_j^0+s'_j, \forall j\in V_T. \tag{13}$$

Since

$$\beta_1^{\Sigma^*}+\cdots+\beta_{N_T}^{\Sigma^*}=\sum\nolimits_{j=1}^{N_T}\beta_j=C. \tag{14}$$

Thus

$$r(s_1^0+\cdots+s_{N_T}^0)+(s'_1+\cdots+s'_{N_T})=C.$$

It implies

$$r=\frac{C-\sum_{j=1}^{N_T}s'_j}{\sum_{j=1}^{N_T}s_j^0}, \tag{15}$$

where $C>\sum_{j=1}^{N_T}s'_j$.

Thus, the optimal execution capacity deployed on task j is expressed as

$$\beta_j^{\Sigma^*}=\frac{C-\sum_{j=1}^{N_T}s'_j}{\sum_{j=1}^{N_T}s_j^0}\cdot s_j^0+s'_j, \forall j\in V_T. \tag{16}$$

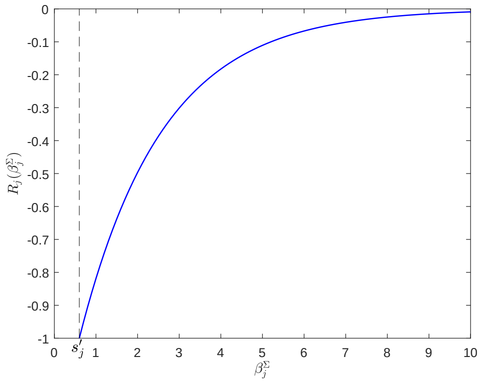

Fig. 1. A schematic figure of the function $R_j(\beta_j^\Sigma)$, where $s_j' = 0.6$, $s_j^0 = 2$.

If all agents have identical execution capacity $\tilde{\beta}$, then $\beta_j^\Sigma = n_j\tilde{\beta}$, where n_j is the number of agents assigned to task j. From (16), the optimal assignment is

$$n_j^* = \beta_j^{\Sigma^*}/\tilde{\beta}, \forall j \in V_T. \tag{17}$$

In general, n_j^* may be non-integer, but it provides guidance for finding the nearest feasible integer solution $(n_1^*, \cdots, n_{N_T}^*)$.

Theorem 2. The function $R(\beta_1^\Sigma, \cdots, \beta_{N_T}^\Sigma) = \sum_{j=1}^{N_T} R_j(\beta_j^\Sigma)$ is strictly concave over $(s_1', +\infty)\times \cdots \times (s_{N_T}', +\infty)$.

Proof: From (8),

$$\frac{dR_j}{d\beta_j^\Sigma} = \frac{e^{(s_j'-\beta_j^\Sigma)/s_j^0}}{s_j^0} > 0, \tag{18}$$

$$\frac{d^2R_j}{d(\beta_j^\Sigma)^2} = -\frac{e^{(s_j'-\beta_j^\Sigma)/s_j^0}}{(s_j^0)^2} < 0, \tag{19}$$

so R_j is strictly increasing and strictly concave on $(s_j', +\infty)$. Since R is a sum of concave functions (9), it is also strictly concave.

For any $\lambda \in (0,1)$, $R(\lambda\beta^\Sigma + (1-\lambda)\bar{\beta}^\Sigma) > \lambda R(\beta^\Sigma) + (1-\lambda)R(\bar{\beta}^\Sigma)$, it is the strict concavity of each R_j. Thus, $R(\cdot)$ is strictly concave over $(s_1', +\infty) \times \cdots \times (s_{N_T}', +\infty)$. ∎

A schematic figure of the function $R_j(\beta_j^\Sigma)$ is given, and it is shown in Fig. 1. A corollary is presented as follows.

Corollary 1. The function $T(\beta_1^\Sigma, \cdots, \beta_{N_T}^\Sigma) = \max\{T_1(\beta_1^\Sigma), \cdots, T_{N_T}(\beta_{N_T}^\Sigma)\}$ is strictly convex over $(s_1', +\infty) \times \cdots \times (s_{N_T}', +\infty)$, where $T_j(\beta_j^\Sigma)$ is shown in (7) [1].

The proof can follow along the lines of the proof of Theorem 2 , thus, the proof of Corollary 1 is omitted.

Remark 1. The analysis shows that optimal assignment equalizes completion times and reduces the search space for discrete solutions. Designing (8) also models diminishing marginal contribution of agents.

4. Dynamic Task Assignment and Game Model

In practical multi-agent systems, task execution capacities are discrete and the initial assignment is often suboptimal. To address time-varying task loads and distributed assignment, the dynamic task assignment problem is formulated, and a potential game model is constructed.

4.1. Dynamic task assignment

In Section 3, the reward function used the initial load s_j^0 for static analysis. For dynamic task assignment it is generalized to depend on the instantaneous load s_j^t, defined as

$$R_j(x^t) = -e^{(s'_j - \beta_j^{\Sigma_t})/s_j^t}. \tag{20}$$

The reward remains strictly concave in $\beta_j^{\Sigma_t}$ for $s_j^t > 0$, and maximizing reward is equivalent to minimizing completion time.

Let us consider a system with N_A agents and N_T tasks. Each agent selects one task at stage t, and the total capacity allocated to task j is

$$\beta_j^{\Sigma_t} = \sum\nolimits_{i \in V_A, x_i^t = j} \beta_i. \tag{21}$$

The task load evolves as

$$s_j^{t+\Delta t} = s_j^t + (s'_j - \beta_j^{\Sigma_t})\Delta t, \tag{22}$$

so task j completes if $\beta_j^{\Sigma_t} > s'_j$. The remaining completion time is

$$T_j(\beta_j^{\Sigma_t}) = \frac{s_j^t}{\beta_j^{\Sigma_t} - s'_j}. \tag{23}$$

Hence

$$T_{final} = t + \max_{j \in V_T} T_j(\beta_j^{\Sigma_t}). \tag{24}$$

Define the global reward

$$R(\beta_1^{\Sigma_t}, \cdots, \beta_{N_T}^{\Sigma_t}) = -\sum\nolimits_{j=1}^{N_T} e^{(s'_j - \beta_j^{\Sigma_t})/s_j^t} = -\sum\nolimits_{j=1}^{N_T} e^{-1/T_j(\beta_j^{\Sigma_t})}. \tag{25}$$

Therefore, maximizing $R(\cdot)$ is equivalent to minimizing T_{final}.

4.2. Potential game

Based on the above formulation, the dynamic task assignment problem at each time stage t can be modeled as a strategic game.

Definition 1. The dynamic task assignment game is defined as $G = (V_A, X^t, F)$, where

i) $V_A = \{1, \ldots, N_A\}$ is the set of agents (players);

ii) $X^t = \prod_{i=1}^{N_A} X_i^t$ is the strategy space at time t, where $X_i^t = V_T$, and $x_i^t \in X_i^t$ denotes the task selected by agent i;

iii) $F = \{f_1, \ldots, f_{N_A}\}$ is the set of utility functions.

The global objective function is defined as

$$f(x^t) = -\sum_{j=1}^{N_T} e^{\frac{s_j' - \beta_j^{\Sigma t}}{s_j^t}}. \tag{26}$$

The utility of agent i is designed as its marginal contribution to the global objective, i.e.,

$$f_i(x_i^t, x_{-i}^t) = f(x_i^t, x_{-i}^t) - f(\bar{x}_i, x_{-i}^t), \tag{27}$$

where $\bar{x}_i$ denotes a reference strategy in which agent i provides no execution capability.

Theorem 3. The game $G = (V_A, X^t, F)$ is an exact potential game with potential function

$$\varphi(x^t) = f(x^t). \tag{28}$$

Proof: For any agent i and any two strategies x_i' and x_i'', while fixing x_{-i}^t, by the definition of marginal utility it results that

$$f_i(x_i', x_{-i}^t) = f(x_i', x_{-i}^t) - f(\bar{x}_i, x_{-i}^t), \tag{29}$$

$$f_i(x_i'', x_{-i}^t) = f(x_i'', x_{-i}^t) - f(\bar{x}_i, x_{-i}^t). \tag{30}$$

Taking the difference yields

$$f_i(x_i', x_{-i}^t) - f_i(x_i'', x_{-i}^t) = f(x_i', x_{-i}^t) - f(x_i'', x_{-i}^t). \tag{31}$$

Thus, the change in the utility of any agent caused by unilateral deviation exactly equals the change in the global objective function. According to the definition of exact potential games, $f(x^t)$ is an exact potential function of the game. ■

5. Distributed Optimization Algorithm

Based on the exact potential game constructed in Section 4, a synchronous distributed learning algorithm is designed to solve the dynamic task assignment problem.

The global objective function defined in Eq. (26) is the exact potential function of the game. Therefore, maximizing individual marginal utilities is equivalent to maximizing the global objective, and a synchronous best-response rule induces a distributed optimization process.

5.1. Synchronous game-based learning algorithm

Since the game is an exact potential game, the potential function $\varphi(x) = f(x)$ in Eq. (26) serves as a Lyapunov function.

At iteration t, all agents update in parallel. Each agent computes its best response

$$\tilde{x}_i^{t+1} = \arg\max_{k \in V_T} f(k, x_{-i}^t) \tag{32}$$

The candidate joint profile is

$$\tilde{x}^{t+1} = (\tilde{x}_1^{t+1}, \ldots, \tilde{x}_{N_A}^{t+1}) \tag{33}$$

The update is accepted only if

$$\varphi(\tilde{x}^{t+1}) \geq \varphi(x^t) \tag{34}$$

Otherwise, set $x^{t+1} = x^t$.

Algorithm 1 Synchronous Game-based Learning Algorithm

1. Initialize strategy profile x^0
2. Set $t \leftarrow 0$
3. Compute $\varphi(x^t)$ using Eq. (26)
4. In parallel, each agent computes best response (Eq. (32))
5. Form candidate profile (Eq. (33))
6. Compute $\varphi(\tilde{x}^{t+1})$
7. If Eq. (34) holds, set $x^{t+1} \leftarrow \tilde{x}^{t+1}$; otherwise set $x^{t+1} \leftarrow x^t$
8. If $x^{t+1} = x^t$, terminate and return x^*
9. Set $t \leftarrow t + 1$ and go to Step 3

5.2. Convergence analysis

Theorem 4. The potential sequence $\{\varphi(x^t)\}$ is non-decreasing. By the finite improvement property of exact potential games, and the finiteness of the strategy space, the algorithm converges in finite iterations to a pure-strategy Nash equilibrium.

Proof: Monotonicity follows from the update rule $\varphi(x^{t+1}) \geq \varphi(x^t)$. Because the strategy space is finite, the non-decreasing sequence must terminate. At convergence, no unilateral deviation improves the potential, hence the limit point is a pure Nash equilibrium. ■

6. Experimental Results and Analysis

To evaluate the effectiveness of the proposed synchronous game-based learning algorithm, simulations were conducted in a dynamic multi-agent task assignment environment following the theoretical model. The experiment uses $N_A = 100$ agents and $N_T = 20$ tasks, where each agent has execution capacity $\beta_i = 4$, yielding total system capacity $C = 400$. The initial task load is $S(0) = 1000$ and increases at rate $s' = 200$ per unit time. Agent-to-task assignments are initialized randomly to avoid bias. Performance is compared with three baselines: Synchronous Best Response (S-BR) [20], Synchronous Best Response with Delay (S-BRD) [21], and Synchronous Log-Linear Learning (S-LLL) [22].

Performance is measured by (i) System Completion Time T_{final}, defined as the first iteration when all tasks are completed, and (ii) Convergence Iterations, measuring steps to equilibrium.

According to the dynamic task model, the load evolution is $s^{t+1} = s^t + (s' - \beta^{\Sigma})$, where $\beta^{\Sigma} = 400$. Substituting parameters yields $s^{t+1} = s^t - 200$. Thus, the total load decreases by 200 units per iteration in the idealized model. In practice, discrete assignment effects, efficiency limitations, and synchronization overhead influence the empirical completion time.

The experiment validates the convergence properties of the proposed game-theoretic learning approach. As shown in Fig. 2, the system converges under game dynamics with monotonic

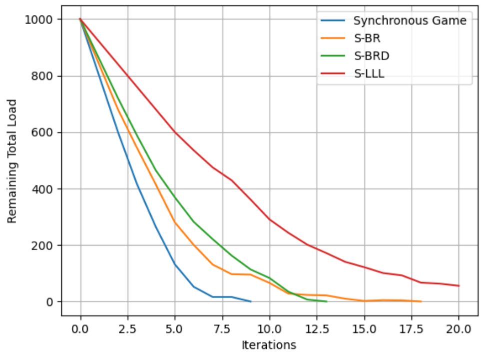

Fig. 2. Remaining task load comparison with representative existing learning algorithms

improvement of the potential function, achieving coordinated task assignment and reaching zero remaining load at 8 iterations. S-BR and S-BRD converge more slowly due to delayed or sequential updates, while S-LLL requires additional iterations because of probabilistic exploration but enhances robustness. Overall, the results demonstrate that the game-theoretic mechanism enables efficient task assignment, strong adaptability in dynamic environments, and superior convergence performance compared with baseline methods.

The synchronous algorithm converges in finite iterations and outperforms baseline methods while maintaining stability, confirming the effectiveness of game-theoretic learning for dynamic multi-agent task assignment.

7. Conclusions and Future Research

In this paper, a dynamic multi-agent task assignment framework based on potential game theory and distributed learning was proposed. By modeling task assignment as an exact potential game and proving convergence to a pure-strategy Nash equilibrium, the framework provides theoretical guarantees for decentralized optimization and scalable coordination. A synchronous distributed learning algorithm is proposed, which takes the potential function as a Lyapunov measure to ensure monotonic improvement and finite-iteration convergence to a pure-strategy Nash equilibrium without centralized coordination. Leveraging only local information and supporting parallel updates, this algorithms enables monotonic improvement and finite-time convergence without centralized control. Those results demonstrate that the proposed method provides an effective game-theoretic solution to the complexity of dynamic task environments and distributed resource assignment.

This paper is limited by the assumptions of zero agent travel time and fixed execution capacities, which deviate from real-world multi-agent system dynamics such as mobility constraints and variable task execution efficiency. Future research will extend the model to incorporate agent travel costs and dynamic execution capacities, and explore asynchronous distributed learning algorithms to further reduce communication overhead and improve real-time performance in large-scale multi-agent task assignment scenarios.

Acknowledgements. This work was partly supported in part by Youth Fund of the National Natural Science Foundation of China under Grant 62103166, the Jiangsu Shuangchuang (Innovation and Entrepreneurship) Talent Program [No.JSSCBS20210840], in part by National Natural Science Foundation of China under Grant 62073345, and in part by a grant of the Romanian Ministry of Research, Innovation and Digitization, CNCS/CCCDI - UEFISCDI, project number ERANET-ENUAC-e-MATS, within PNCDI IV.

References

[1] J. CHEN, Y. GUO, Z. QIU, B. XIN, Q.-S. JIA and W. GUI, *Multiagent dynamic task assignment based on forest fire point model*, IEEE Transactions on Automation Science and Engineering **19**(2), 2022, pp. 833–849.

[2] G. GAO, Y. MEI, Y.-H. JIA, W. N. BROWNE and B. XIN, *Adaptive coordination ant colony optimization for multipoint dynamic aggregation*, IEEE Transactions on Cybernetics **52**(8), 2022, pp. 7362–7376.

[3] R.-C. ROMAN, R.-E. PRECUP and E. M. PETRIU, Active disturbance rejection control and model-free control tuned via fictitious reference iterative tuning, Facta Universitatis, Series: Mechanical Engineering **23**(2), 2025, pp. 183–196.

[4] S. SUN, K. XU, D. FENG and B. DING, *Bidirectional influence and interaction for multiagent reinforcement learning*, IEEE Transactions on Artificial Intelligence **5**(10), 2024, pp. 4984–4995.

[5] P. MILIĆ, D. MARINKOVIĆ and Ž. ĆOJBAŠIĆ, Geometrically nonlinear analysis of piezoelectric active laminated shells by means of isogeometric FE formulation, Facta Universitatis, Series: Mechanical Engineering **23**(2), 2025, pp. 387–405.

[6] B.-L. NGUYEN, D.-D. NGUYEN, H.-X. NGUYEN, D.-T. NGO and M. WAGNER, *Multi-agent task assignment in vehicular edge computing: A regret-matching learning-based approach*, IEEE Transactions on Emerging Topics in Computational Intelligence **8**(2), 2024, pp. 1527–1539.

[7] X. YUAN, S. HU, W. NI, X. WANG and A. JAMALIPOUR, *Deep reinforcement learning-driven reconfigurable intelligent surface-assisted radio surveillance with a fixed-wing UAV*, IEEE Transactions on Information Forensics and Security **18**, 2023, pp. 4546–4560.

[8] R.-E. PRECUP, C.-A. BOJAN-DRAGOS, E.-L. HEDREA, R.-C. ROMAN and E. M. PETRIU, Evolving fuzzy models of shape memory alloy wire actuators, Romanian Journal of Science and Technology **24**(4), 2021, pp. 353–365.

[9] S. AL-HUSSAINI, J.-M. GREGORY and S.-K. GUPTA, *Generating task reallocation suggestions to handle contingencies in human-supervised multi-robot missions*, IEEE Transactions on Automation Science and Engineering **21**(1), 2024, pp. 367–381.

[10] Z. DAI, Y. ZHANG, W. ZHANG, X. LUO and Z. HE, *A multi-agent collaborative environment learning method for UAV deployment and resource allocation*, IEEE Transactions on Signal and Information Processing over Networks **8**, 2022, pp. 120–130.

[11] H. CHENG, B. TIAN, X. ZHANG and H. SHEN, *A distributed algorithm for multi-robot task allocation via weighted buffered Voronoi partition*, IEEE Transactions on Industrial Electronics **71**(12), 2024, pp. 16098–16107.

[12] X. SONG, Y. HUA, Y. YANG, G. XING, F. LIU, L. XU and T. SONG, *Distributed resource allocation with federated learning for delay-sensitive IoV services*, IEEE Transactions on Vehicular Technology **73**(3), 2024, pp. 4326–4336.

[13] E. KHALIL, H. DAI, Y. ZHANG, B. DILKINA and L. SONG, *Learning combinatorial optimization algorithms over graphs*, Advances in Neural Information Processing Systems **30**, 2017, pp. 6347–6357.

[14] H. XU, Z. WANG and H. YANG, *Learning simple thresholded features with sparse support recovery*, IEEE Transactions on Circuits and Systems for Video Technology **30**(4), 2019, pp. 970–982.

[15] M. S. ALI, P. COUCHENEY and M. COUPECHOUX, *Distributed learning in noisy-potential games for resource allocation in D2D networks*, IEEE Transactions on Mobile Computing **19**(12), 2019, pp. 2761–2773.

[16] H. WU and H. SHANG, *Potential game for dynamic task allocation in multi-agent system*, ISA Transactions **102**, 2020, pp. 208–220.

[17] H. BOROWSKI and J. R. MARDEN, *Fast convergence in semianonymous potential games*, IEEE Transactions on Control of Network Systems **4**(2), 2015, pp. 246–258.

[18] X. ZHAO, Q. ZONG, B. TIAN and M. YOU, *Finite-time dynamic allocation and control in multiagent coordination for target tracking*, IEEE Transactions on Cybernetics **52**(3), 2022, pp. 1872–1880.

[19] X. F. LIU, Y. FANG, Z. H. ZHAN, J. ZHANG, *Strength learning particle swarm optimization for multiobjective multirobot task scheduling*, IEEE Transactions on Systems, Man, and Cybernetics: Systems **53**(7), 2023, pp. 4052–4063.

[20] B. SWENSON, R. MURRAY and S. KAR, *On best-response dynamics in potential games*, SIAM Journal on Control and Optimization **56**(4), 2018, pp. 2734–2767.

[21] J. LEI, U. V. SHANBHAG, J. S. PANG, S. SEN, *On synchronous, asynchronous, and randomized best-response schemes for stochastic Nash games*, Mathematics of Operations Research **45**(1), 2020, pp. 157–190.

[22] J. R. MARDEN and J. S. SHAMMA, *Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation*, Games and Economic Behavior **75**(2), 2012, pp. 788–808.