2021 American Control Conference (ACC)
New Orleans, USA, May 25-28, 2021

Decentralized Game-Theoretic Control for Dynamic Task Allocation Problems for Multi-Agent Systems

Efstathios Bakolas Yoonjae Lee

Abstract— **We propose a decentralized game-theoretic framework for dynamic task allocation problems for multi-agent systems. In our problem formulation, the agents' utilities depend on both the rewards and the costs associated with the successful completion of the tasks assigned to them. The rewards reflect how likely is for the agents to accomplish their assigned tasks whereas the costs reflect the effort needed to complete these tasks (this effort is determined by the solution of corresponding optimal control problems). The task allocation problem considered herein corresponds to a dynamic game whose solution depends on the states of the agents in contrast with classic static (or single-act) game formulations. We propose a greedy solution approach in which the agents negotiate with each other to find a mutually agreeable (or individually rational) task assignment profile based on evaluations of the task utilities that reflect their current states. We illustrate the main ideas of this work by means of extensive numerical simulations.**

I. Introduction

We consider a dynamic task allocation problem for a multi-agent system whose agents have continuous state and input spaces and have to complete a set of spatially distributed tasks (obtain in-situ measurements or pick up packages from different locations over a given spatial domain). We adopt a game-theoretic approach which seeks for task assignments that maximize the individual utility of each agent conditional on the assignments of their teammates (individual rationality principle) while also ensuring that the self-interests of the agents are aligned with those of the team. To this aim, we design the agents' utilities based on the concept of *wonderful life utility* [1] (WLU) which allows us to associate the dynamic task allocation problem with a sequence of potential games [2]. We propose a greedy decentralized algorithm that computes task assignment profiles which are mutually agreeable in the long run.

Literature review: Task allocation problems for multi-agent systems can be addressed by auction-based techniques, distributed and / or multi-objective optimization and game-theoretic methods. The auction-based techniques are centralized when the agents negotiate with each other under the guidance of an auctioneer [3] and decentralized when they negotiate directly with each other [4]–[6]. Centralized methods rely on a single point of failure whereas the communication cost in decentralized methods can be substantial if not prohibitive. Distributed optimization [7] and multi-objective optimization [8] for task allocation problems are typically quite complex and require some knowledge about the utilities of the other agents and, more importantly, do not necessarily yield solutions which are mutually agreeable. As suggested in [9], game-theoretic tools constitute one of the most natural approaches to task allocation problems for intelligent, autonomous agents. Reference [9], which is the main inspiration of this paper, utilizes the framework of potential games to define in a systematic way the task and agent utilities as well as several negotiation protocols (game-theoretic learning algorithms [10], [11]) for the computation of mutually agreeable task assignments in a decentralized or distributed way. The negotiation protocols utilized in [9] converge to mutually agreeable task assignment profiles without requiring that any agent should know the utility functions of her teammates (decentralized task allocation). However, their convergence is conditional on the game remaining the same (e.g., the functional description of the utilities does not change throughout the negotiation process). Thus, although the equilibrium of the game is found iteratively in [9], the task allocation problem itself is essentially modeled as a static game. Extensions of the game-theoretic framework for multi-agent control problems can be found in [12]–[14]. Ref. [15] proposes a myopic solution approach to a dynamic task allocation problem modeled as a sequence of static (single-act) potential games. The approach in [15] cannot handle state-dependent utilities in general. Finally, the framework of state-dependent potential games [16] is only applicable to problems with finite (discrete) state spaces.

Contributions: In this paper, we address a dynamic task allocation problem in which the task utilities depend on both the rewards earned by the agents for completing their assigned tasks as well as the costs they incur while doing so (cost-to-go functions of corresponding optimal control problems). Consequently, the utilities are in general state-dependent. We adopt a decentralized game-theoretic solution approach (each agent knows only her own utility function). The (individual) agent utilities are designed in accordance with the WLU framework which ensures that their self-interests are aligned with the team's interests under the framework of potential games. We propose a greedy solution approach in which the negotiations between the agents take place on-the-fly while the agents move in their state space towards their assigned tasks. Every time an agent changes her individual assignment (and thus her final state destination) she has to update the estimate of the cost-to-go and consequently her utility function as well (state-dependent utilities). We design the negotiation process such that the agents compute a mutually agreeable profile which is not likely to change during the last phase of the process.

This work was supported in part by ARL under W911NF2020085. E. Bakolas (Associate Professor) and Y. Lee (graduate student) are with the Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, Texas 78712-1221, USA, Emails: bakolas@austin.utexas.edu; yol033@utexas.edu

978-1-6654-4197-1/$31.00 ©2021 AACC

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.

Outline: The rest of the paper is organized as follows. In Section II, we discuss the problem preliminaries. The task, team and agent utilities are defined in Section III. The open-loop task allocation is addressed in Section IV and the dynamic problem in Section V. Numerical simulations are presented in Section VI. Finally, Section VII presents concluding remarks and directions for future work.

II. Preliminaries and Problem Setup

Notation: We denote by $\mathbb{R}^n$ the set of n-dimensional real vectors. We denote by $\mathbb{Z}$ the set of integers. Given $a, b \in \mathbb{Z}$ with $a \leq b$, we denote by $[a, b]_d$ the discrete time interval from a to b, that is, $[a, b]_d := [a, b] \cap \mathbb{Z}$. Given $k \in \mathbb{Z}$, we write $\mathbb{Z}_k$ to denote the (unbounded) discrete interval $[k, \infty) \cap \mathbb{Z}$. Given a finite set $\mathcal{A}$, we denote by $\mathrm{card}(\mathcal{A})$ its cardinality.

Problem setup: We consider a multi-agent system (MAS) comprised of n agents. We denote by $x_i \in \mathcal{S}_i \in \Sigma$ and $u_i \in \mathcal{U}_i$, for $i \in [1, n]_d$, the state and input of the i-th agent of the MAS at time $t \geq 0$, where $\mathcal{S}_i$ and $\mathcal{U}_i$ denote her state space and input space, respectively, and $\Sigma \subseteq \mathbb{R}^m$. In addition, we denote by $\boldsymbol{x} \in \mathcal{S}$ the joint state of the MAS, where $\boldsymbol{x} := (x_1, \ldots, x_n)$ and $\mathcal{S} := \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ (joint state space), and by $\boldsymbol{u} \in \mathcal{U}$ the joint input of the MAS, where $\boldsymbol{u} := (u_1, \ldots, u_n)$ and $\mathcal{U} := \mathcal{U}_1 \times \cdots \times \mathcal{U}_n$ (joint input space). Furthermore, we denote by $\boldsymbol{x}_{-i} \in \mathcal{S}_{-i}$ and $\boldsymbol{u}_{-i} \in \mathcal{U}_{-i}$ the concatenations of the states and the inputs of all the agents except from the i-th agent (the sets $\mathcal{S}_{-i}$ and $\mathcal{U}_{-i}$ are defined accordingly). The motion of the i-th agent is described by

$$\dot{x}_i = f_i(\boldsymbol{x}, \boldsymbol{u}), \qquad x_i(0) = x_i^0, \qquad i \in [1, n]_d, \tag{1}$$

for $t \geq 0$, where $x_i^0 \in \mathcal{S}_i$ is the initial state of the i-th agent and $f_i : \mathcal{S} \times \mathcal{U} \to \mathcal{S}_i$ is her associated vector field. We assume that the vector field f_i satisfies regularity conditions that ensure the existence and uniqueness of solutions to the differential equations (1) for all piecewise continuous joint inputs $\boldsymbol{u}$ taking values in $\mathcal{U}$ and all joint states $\boldsymbol{x} \in \mathcal{S}$. Finally, we write

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \qquad \boldsymbol{x}(0) = \boldsymbol{x}^0, \tag{2}$$

where $\boldsymbol{x}^0 = (x_1^0, \ldots, x_n^0) \in \mathcal{S}$ is the joint initial state and $\boldsymbol{f} := (f_1, \ldots, f_n)$ is the joint vector field.

The task allocation problem seeks for individual assignments for a team of n agents and for a given set of p tasks, $\mathcal{T} := \{\mathcal{T}_1, \ldots, \mathcal{T}_p\}$. Each task is associated with a distinct state in Σ. We denote by $X_{\mathcal{T}}$ the set of states associated with the given tasks, where $X_{\mathcal{T}} := \{x_{\mathcal{T}_1}, \ldots, x_{\mathcal{T}_p}\}$. In principle, an agent can be assigned at most one task in $\mathcal{T}$ at each instant of time although more than one agents can be assigned to the same task simultaneously. We denote by $\mathcal{A}_i := \{a_i^k : \ k \in [1, \mathrm{card}(\mathcal{A}_i)]_d\}$ the set of possible task assignments for the i-th agent given a set of tasks $\mathcal{T}$. Later on, we will see that each assignment a_i^k induces a corresponding (admissible) control input $u_i(\cdot)$ via the solution of a corresponding optimal control problem. We assume that $a_\varnothing \in \mathcal{A}_i$, where $a_\varnothing$ denotes the null assignment (i.e., the i-th agent is not assigned to any task); when $a_i = a_\varnothing$, then $u_i(t) = 0$, for all $t \geq 0$. Each assignment $a_i^k \in \mathcal{A}_i$ is equal to either a task in $\mathcal{T}$, that is, $a_i^k = \mathcal{T}_\ell$ where $\mathcal{T}_\ell \in \mathcal{T}$, or the null assignment, that is, $a_i^k = a_\varnothing$. Thus, $\mathcal{A}_i \subseteq \overline{\mathcal{T}}$, where $\overline{\mathcal{T}} := \mathcal{T} \cup \{a_\varnothing\}$. The set $\mathcal{A}_i$ can be taken to be state-dependent by requiring that a task $\mathcal{T}_\ell \in \mathcal{T}$ can belong to $\mathcal{A}_i$ only if its corresponding state $x_{\mathcal{T}_\ell}$ is within a certain range $\varrho > 0$ from the state x_i of the i-th agent, that is, $|x_i - x_{\mathcal{T}_\ell}| \leq \varrho$; when ϱ is finite we say that $\mathcal{A}_i$ is range-constrained and is range-unconstrained, otherwise (i.e., $\varrho \to \infty$). All the subsequent analysis will be carried out for the range-unconstrained (nominal) case although the range-constrained case can be handled similarly after the necessary modifications have been carried out. Finally, we denote by $\mathcal{A}$, where $\mathcal{A} := \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$, the joint assignment space of the multi-agent team.

III. Task Utilities

The completion of a task $\mathcal{T}_j \in \mathcal{T}$ will accrue rewards to the agents assigned to it. As we will explain shortly later, these rewards do not depend on the states of the agents explicitly and for this reason, we will refer to them as the static task utilities. Furthermore, an agent will have to incur a cost to complete her assigned task (e.g., the transition cost to a certain location associated with this task), which we will refer to as the *dynamic task completion cost* (the latter cost is state-dependent).

Static task utility: Given an action profile $\boldsymbol{a} = (a_1, \ldots, a_n)$, we denote by $\mathcal{T}_j^{-1}(\boldsymbol{a})$ the index-set corresponding to the agents assigned to task $\mathcal{T}_j \in \mathcal{T}$ under the particular profile, that is, $\mathcal{T}_j^{-1}(\boldsymbol{a}) = \{i \in [1, n]_d : a_i = \mathcal{T}_j\}$. The completion of task $\mathcal{T}_j$ will accrue a reward $r_{\mathcal{T}_j} \geq 0$ to the agent or agents assigned to this task. In general, $r_{\mathcal{T}_j}$ is a function of the task assignment profile, that is,

$$r_{\mathcal{T}_j(\boldsymbol{a})} := \bar{r}_{\mathcal{T}_j}[1 - \textstyle\prod_{i \in \mathcal{T}_j^{-1}(\boldsymbol{a})}(1 - p_{ij})], \tag{3}$$

where $\bar{r}_{\mathcal{T}_j}$ is the nominal reward of $\mathcal{T}_j$ and $p_{ij} \in [0, 1]$ is the probability that the task $\mathcal{T}_j$ will be completed successfully by the i-th agent. Finally, we set $r_{\mathcal{T}_j}(\boldsymbol{a}) := 0$ when $\mathcal{T}_j^{-1}(\boldsymbol{a}) = \varnothing$. The nominal reward $\bar{r}_{\mathcal{T}_j}$ does not depend on which agents will be assigned to task $\mathcal{T}_j$ or the effort required by them to complete this task; instead its value reflects the importance of $\mathcal{T}_j$ relative to the other tasks in $\mathcal{T}$. The probability p_{ij} reflects the suitability of the i-th agent (in terms of her preferences and capabilities) to carry out task $\mathcal{T}_j$ as well as the difficulty of the latter task to be completed by this agent. Because the probabilities p_{ij}, for all $i \in \mathcal{T}_j^{-1}(\boldsymbol{a})$, contribute to the value of $r_{\mathcal{T}_j}$, it follows that the latter will depend on which agents are assigned to task $\mathcal{T}_j$. However, $r_{\mathcal{T}_j(\boldsymbol{a})}$ still does not depend on the cost that the agents assigned to $\mathcal{T}_j$ will incur for its completion.

State-dependent task completion cost: Next, we define the cost for completing the task $\mathcal{T}_j$ associated with the state $x_{\mathcal{T}_j}$ at time $t = t_\mathsf{f}$ by the i-th agent. Essentially, the task completion cost is taken to be the cost incurred by the i-th agent, which starts from the state x_i^0 at time $t = 0$, to reach the state $x_{\mathcal{T}_j}$ at time $t = t_\mathsf{f}$. The latter state transition cost is defined as the optimal cost-to-go corresponding to the following optimal control problem:

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.

Problem 1: Let $a_i = \mathcal{T}_j$, where $\mathcal{T}_j \in \mathcal{T}$ and $i \in [1,n]_d$. Furthermore, let $x_{\mathcal{T}_j} \in \mathcal{S}_i$ be the state associated with the task $\mathcal{T}_j$ and let $t_\mathsf{f} > 0$ be the corresponding completion time (fixed and common for all the agents). Then, find an optimal piece-wise continuous input $u_i^\star(\cdot) : [0, t_\mathsf{f}] \to \mathcal{U}_i$ that minimizes the following performance index:

$$\mathcal{J}_i(u_i(\cdot); x_i^0, x_{\mathcal{T}_j}) := \int_0^{t_\mathsf{f}} \mathcal{L}_i(x_i(t), u_i(t))\mathrm{d}t, \qquad (4)$$

subject to the dynamic constraints (1) and the terminal condition: $x_i(t_\mathsf{f}) = x_{\mathcal{T}_j}$. Finally, the optimal cost-to-go is denoted by $\rho_i(x_i^0; x_{\mathcal{T}_j})$, where $\rho_i(x_i^0; x_{\mathcal{T}_j}) := \mathcal{J}_i(u_i^\star(\cdot); x_i^0, x_{\mathcal{T}_j})$.

Total Task Utility: The total cost of completion of task $\mathcal{T}_j$ under the action profile $\boldsymbol{a} = (a_1, \ldots, a_n)$, which is denoted as $\mathcal{R}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0, x_{\mathcal{T}_j})$, is defined as the sum of the individual task completion costs of all the agents assigned to that task. More precisely,

$$\mathcal{R}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0, x_{\mathcal{T}_j}) := \sum\nolimits_{i \in \mathcal{T}_j^{-1}(\boldsymbol{a})} \rho_i(x_i^0; x_{\mathcal{T}_j}). \qquad (5)$$

Note that $\mathcal{R}_{\mathcal{T}_j}$ depends on the initial state $\boldsymbol{x}^0$ (more precisely, the initial states of the agents assigned to the task $\mathcal{T}_j$). Furthermore, the total task utility associated with task $\mathcal{T}_j$ for a given $\boldsymbol{x}_0$ is denoted as $\mathcal{U}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0)$ and defined as follows:

$$\mathcal{U}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0) := \max\{0, r_{\mathcal{T}_j}(\boldsymbol{a}) - \mathcal{R}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0, x_{\mathcal{T}_j})\}. \qquad (6)$$

Note that $\mathcal{U}_{\mathcal{T}_j}$ is state-dependent because the task completion costs $\rho_i(x_i^0; x_{\mathcal{T}_j})$, for $i \in \mathcal{T}_j^{-1}(\boldsymbol{a})$, are state-dependent.

Individual and Team Utilities and Solution Concepts: First, we define the team's utility (the latter reflects the team's collective welfare), which is denoted by $\mathcal{U}(\boldsymbol{a}; \boldsymbol{x}^0)$, as follows:

$$\mathcal{U}(\boldsymbol{a}; \boldsymbol{x}^0) := \sum\nolimits_{\mathcal{T}_j \in \mathcal{T}} \mathcal{U}_{\mathcal{T}_j}(\boldsymbol{a}; \boldsymbol{x}^0). \qquad (7)$$

The individual utility of the i-th agent given a task profile $\boldsymbol{a} = (a_i, a_{-i})$, which is denoted as $\mathcal{U}_i(a_i, a_{-i})$ or $\mathcal{U}_i(\boldsymbol{a})$, is taken to be equal to her marginal contribution to the team's utility $\mathcal{U}(\boldsymbol{a}; \boldsymbol{x}^0)$, that is,

$$\mathcal{U}_i(\boldsymbol{a}; \boldsymbol{x}^0) := \mathcal{U}((a_i, a_{-i}); \boldsymbol{x}^0) - \mathcal{U}((a_\varnothing, a_{-i}); \boldsymbol{x}^0) \qquad (8)$$

from which it follows, in view of (7), that

$$\mathcal{U}_i(\boldsymbol{a}; \boldsymbol{x}^0) = \mathcal{U}_{\mathcal{T}_j}((a_i, a_{-i}); \boldsymbol{x}^0) - \mathcal{U}_{\mathcal{T}_j}((a_\varnothing, a_{-i}); \boldsymbol{x}^0), \quad (9)$$

where $(a_\varnothing, a_{-i})$ corresponds to the action profile when the i-th agent has a null assignment, that is, $a_i = a_\varnothing$. Next, we provide the definition of the basic solution concept that will be used in our task allocation problem.

Definition 1: An assignment profile $\boldsymbol{a}^\star := (a_i^\star, a_{-i}^\star) \in \mathcal{A}$ is a pure strategy Nash equilibrium of the game $\mathcal{G}^0$, where $\mathcal{G}^0 := \langle \mathcal{U}_1(\boldsymbol{a}; \boldsymbol{x}^0), \ldots, \mathcal{U}_n(\boldsymbol{a}; \boldsymbol{x}^0); \mathcal{A} \rangle$, if $\forall i \in [1,n]_d$

$$\mathcal{U}_i(a_i^\star, a_{-i}^\star; \boldsymbol{x}^0) \geq \mathcal{U}_i(a_i, a_{-i}^\star; \boldsymbol{x}^0), \qquad \forall a_i \in \mathcal{A}_i. \qquad (10)$$

Remark 1 The solution concept of (pure strategy) Nash equilibrium is fundamental in non-cooperative game theory. When all agents play in accordance with the Nash equilibrium, they act selfishly and try to maximize their own utilities conditional on the decisions of others (*individual rationality*).

IV. The Open-Loop Task Allocation Problem and its Decentralized Solution

A. Problem formulation and analysis

Next, we formulate the task allocation problem as a non-cooperative game which is only based on the estimates of the task completion costs at the initial time (open-loop approach).

Problem 2 (OLTA: Open-Loop Task Allocation): Let $t_\mathsf{f} > 0$ and $\boldsymbol{x}^0 \in \mathcal{S}$ be given. Then, find a (time-invariant) task assignment profile $\boldsymbol{a}^\star \in \mathcal{A}$, where $\boldsymbol{a}^\star := (a_i^\star, a_{-i}^\star)$, such that for all $i \in [1,n]_d$ the inequality (10) is satisfied. In other words, the task assignment profile $\boldsymbol{a} := (a_i^\star, a_{-i}^\star)$ corresponds to a Nash equilibrium of the game $\mathcal{G}^0$.

Remark 2 In the formulation of Problem 2, the functional descriptions of the agents' utilities do not change with time, as the agents progress towards the states of their assigned tasks. This is because their estimated task completion costs are based on knowledge available at time $t = 0$ and these estimates are not updated afterwards. To the i-th individual task assignment $a_i^\star$ from the optimal profile $\boldsymbol{a}^\star$, where, say, $a_i^\star = \mathcal{T}_j$, we associate a corresponding state $x_{\mathcal{T}_j}$, which in turn determines the terminal constraint $x_i(t_\mathsf{f}) = x_{\mathcal{T}_j}$ in Problem 1. Because all the task assignments are time-invariant, the control input $u_i^\star(\cdot)$ that solves Problem 1 will not be updated along the i-th agent's ensuing trajectory.

It is well-known that potential games correspond to a special class of non-cooperative games that always admit pure strategy Nash equilibria. We claim that Problem 2 corresponds to an exact potential game [17].

Definition 2: The game, $\mathcal{G}^0$, corresponds to an exact potential game, if there exists an exact potential, that is, a function $\mathcal{P} : \mathcal{A} \to \mathbb{R}$ such that, for all $i \in [1,n]_d$ and $\boldsymbol{a} \in \mathcal{A}$

$$\begin{aligned}\mathcal{U}_i(a_i', a_{-i}; \boldsymbol{x}^0) - \mathcal{U}_i(a_i, a_{-i}; \boldsymbol{x}^0) \\ = \mathcal{P}(a_i', a_{-i}) - \mathcal{P}(a_i, a_{-i}),\end{aligned} \qquad (11)$$

for all $a_i, a_i' \in \mathcal{A}_i$.

Proposition 1: The OLTA problem (Problem 2) with task team utility and individual utilities defined by (7) and (10), respectively, is an exact potential game with potential $\mathcal{P} = \mathcal{U}(a_i, a_{-i}; \boldsymbol{x}^0)$, for all $a_i \in \mathcal{A}_i$.

Proof: We will directly verify (11). In particular,

$$\begin{aligned}
&\mathcal{U}(a_i', a_{-i}; \boldsymbol{x}^0) - \mathcal{U}(a_i, a_{-i}; \boldsymbol{x}^0) \\
&\quad = \sum_{\mathcal{T}_j \in \mathcal{T}} \big(\mathcal{U}_{\mathcal{T}_j}(a_i', a_{-i}; \boldsymbol{x}^0) - \mathcal{U}_{\mathcal{T}_j}(a_i, a_{-i}; \boldsymbol{x}^0)\big) \\
&\quad = \sum_{k \neq i} \mathcal{U}_{\mathcal{T}_j}(a_k, a_{-k}; \boldsymbol{x}^0) - \sum_k \mathcal{U}_{\mathcal{T}_j}(a_k, a_{-k}; \boldsymbol{x}^0) \\
&\qquad + \mathcal{U}_{\mathcal{T}_j}(a_i', a_{-i}; \boldsymbol{x}^0) \\
&\quad = \big(\mathcal{U}_{\mathcal{T}_j}(a_i', a_{-i}; \boldsymbol{x}^0) - \mathcal{U}_{\mathcal{T}_j}(a_\varnothing, a_{-i}; \boldsymbol{x}^0)\big) \\
&\qquad - \big(\mathcal{U}_{\mathcal{T}_j}(a_i, a_{-i}; \boldsymbol{x}^0) - \mathcal{U}_{\mathcal{T}_j}(a_\varnothing, a_{-i}; \boldsymbol{x}^0)\big) \\
&\quad = \mathcal{U}_i(a_i', a_{-i}; \boldsymbol{x}^0) - \mathcal{U}_i(a_i, a_{-i}; \boldsymbol{x}^0),
\end{aligned}$$

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.

where in the derivation of the last equality we have used Eq. (9). In view of Definition 2, we conclude that $\mathcal{G}^0$ corresponds to an exact potential game. ■

B. Negotiation protocols for decentralized task allocation

Problem 2 can be solved by utilizing standard tools for the computation of Nash equilibria of non-cooperative (static or single-act) games and in particular potential games [17]. An alternative approach is to employ game-theoretic learning algorithms which generate a sequence of task assignment profiles that converge to a Nash equilibrium. Some of these algorithms include the fictitious play (FP), spatial adaptive play (SAP), and generalized regret monitoring with fading memory and inertia (GRM) algorithms to name but a few; the reader may refer to [9] for more information on these and other similar algorithms. A key point is that for their realization, an agent does not have to know the utilities of her teammates (decentralized implementation).

During the negotiation (learning) process, the task assignment profile of the team is updated at different time instants that form a non-decreasing sequence $\{\tau_k\}_{k=0}^{\infty}$ in $[0, t_\mathsf{f}]$ such that $\tau_0 = 0$ and $\lim_{k\to\infty} \tau_k = t_\mathsf{f}$. The time instant τ_k corresponds to the k-th stage of the negotiation process. At that stage, the i-th agent picks her new task assignment, which we denote as $a_i(\tau_k)$; we also denote the corresponding profile of the whole team as $\boldsymbol{a}(\tau_k)$. The exact definition of the $a_i(\tau_k)$ will be determined by the particular learning algorithm that will be employed, which in turn will rely on a corresponding information set $\mathcal{I}_k^i$. The latter set may encode information about the past performance of the i-th agent (measured in terms of past values of her own utility) as well as information about the history (whole or truncated) of her teammates' actions (such information may correspond to, for instance, the empirical distribution of the agents' past task assignments). We write

$$\boldsymbol{a}(\tau_k) = \boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^0),\ a_i(\tau_k) = \phi_i(\mathcal{I}_k^i; \mathcal{G}^0),\ i \in [1, n]_d, \quad (12)$$

where $\phi_i : \mathcal{I}_k^i \to \mathcal{A}_i$, for $i \in [1, n]_d$, is the update law (or proposal) of the individual target assignment of the i-th agent whereas $\boldsymbol{\phi} : \boldsymbol{\mathcal{I}}^k \to \mathcal{A}$, with $\boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^0) := (\phi_1(\mathcal{I}_k^1; \mathcal{G}^0), \ldots, \phi_n(\mathcal{I}_k^n; \mathcal{G}^0))$, is the update law of the task assignment profile of the MAS given the joint information set $\boldsymbol{\mathcal{I}}_k := \mathcal{I}_k^1 \times \cdots \times \mathcal{I}_k^n$. It should be understood here that the function $\boldsymbol{\phi}$ does not admit in general an analytic characterization and for the computation of its value for a given joint information set $\boldsymbol{\mathcal{I}}_k$, one has to utilize a specialized algorithm like FP, SAP or GRM. The following claim is based on the analysis provided in [9] (refer to, for instance, Theorem 4.1) and references therein.

Claim 1: The update law $\boldsymbol{a}(\tau_k)$ which is defined as in (12) and corresponds to one of the decentralized negotiation protocols (game-theoretic learning algorithms) used in [9] will converge to a pure strategy Nash equilibrium of $\mathcal{G}^0$.

It is important to emphasize that the update law for the task assignment profile will solve Problem 2 under the assumption that throughout the interval $[0, t_\mathsf{f}]$, the functional description of the utilities will be based on their initial estimates at time $t = 0$. However, the agents' utilities are state-dependent and thus their functional description will change along the agents' ensuing trajectories. This variability in the agents' preferences and capabilities (as reflected on their utilities) cannot be captured in this update law as well as the OLTA problem itself. An alternative interpretation of the negotiation process is to assume that it does not take place over the time interval $[0, t_\mathsf{f}]$ but instantaneously, at time $t = 0$, as if the clock is paused until the negotiations have converged (within some acceptable tolerance) to a Nash equilibrium of the potential game $\mathcal{G}^0$. Subsequently, the agents can execute the corresponding inputs that will transfer them to the terminal states associated with their assigned tasks (these inputs are computed by solving Problem 1 for each agent). The input signal will remain the same function of time, for all $t \in [0, t_\mathsf{f}]$. It is worth mentioning that the game-theoretic learning algorithms can be implemented based on local information (distributed implementation) by taking the sets $\mathcal{A}_i$, for $i = [1, n]_d$, to be range-constrained for a given finite range $\varrho > 0$.

V. Dynamic Task Allocation and a Greedy Algorithm for its Solution

A. Problem formulation

Next, we formulate a dynamic task allocation (DTA) problem which accounts for the fact that the agents' utilities change along their ensuing trajectories in contrast with the OLTA problem. In this problem formulation, a new game $\mathcal{G}^t$, where $\mathcal{G}^t := \langle \mathcal{U}_1(\cdot; \boldsymbol{x}(t)), \ldots, \mathcal{U}_n(\cdot; \boldsymbol{x}(t)); \mathcal{A} \rangle$ is essentially obtained at each $t \in [0, t_\mathsf{f}]$ as the agents move in their state space.

Problem 3 (DTA: Dynamic Task Allocation): Let $t_\mathsf{f} > 0$ and $\boldsymbol{x}^0 \in \mathcal{S}$ be given. Then, find a time-varying task assignment profile $\boldsymbol{a}^\star(\cdot) : [0, t_\mathsf{f}] \to \mathcal{A}$, where $\boldsymbol{a}^\star(t) := (a_i^\star(t), a_{-i}^\star(t))$, which is such that for all $i \in [1, n]_d$:

$$\mathcal{U}_i(a_i^\star(t), a_{-i}^\star(t); \boldsymbol{x}(t)) \geq \mathcal{U}_i(a_i(t), a_{-i}^\star(t); \boldsymbol{x}(t)),$$

for all $a_i(t) \in \mathcal{A}_i$, as $t \to t_\mathsf{f}$. In other words, the task assignment profile $\boldsymbol{a}^\star(t)$ converges to a Nash equilibrium of the game $\mathcal{G}^t$ as $t \to t_\mathsf{f}$.

Remark 3 Problem 3 seeks for a task assignment profile that will become mutually agreeable among the agents as $t \to t_\mathsf{f}$. Note that if t_f is taken to be sufficiently large, then the solution to the Problem 3 will essentially converge to a *steady-state* task assignment profile.

B. A greedy algorithm for dynamic task allocation

Next we propose a greedy solution approach to address Problem 3. To ensure that the game-theoretic learning algorithms discussed in Section IV-B will converge to a Nash equilibrium as $t \to t_\mathsf{f}$, we propose to stop updating the agents' utilities at time $t = t_\mathsf{f} - \epsilon$ for some $0 < \epsilon < t_\mathsf{f}$ so that the utilized learning algorithm (whose convergence is guaranteed only for a static game) are given the chance to converge during the sub-interval $[t_\mathsf{f} - \epsilon, t_\mathsf{f}]$. A key difference between the DTA and OLTA problems is that the static

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.

game that determines the task assignment profile in the former is not $\mathcal{G}^0$ (game corresponding to time $t = 0$), as in the latter problem, but a game corresponding to time $t = t_\mathsf{f} - \epsilon$, assuming that the (dynamic) game has evolved for $t \in [0, t_\mathsf{f} - \epsilon]$.

Next, we present the main steps of the proposed greedy algorithm. To this aim, let us consider a sequence $\{\tau_k\}_{k=0}^{\infty}$ as in Section IV-B and let $K = K(\epsilon)$ be the first positive integer at which $\tau_K > t_\mathsf{f} - \epsilon$ for the given ϵ (the existence of such K is guaranteed by the fact that $\tau_k \to t_\mathsf{f}$ as $k \to \infty$). Now, let $\boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^{\tau_k})$ denote the update law of a negotiation protocol as in Section IV-B. Let us consider the following update law:

$$\boldsymbol{a}_d(\tau_k) := \begin{cases} \boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^{\tau_k}), & \text{for } k \in [0, K-1]_d, \\ \boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^{\tau_K}), & \text{for } k \in \mathbb{Z}_K. \end{cases} \tag{13}$$

We claim that the update law (13) will find an approximate (in the sense that we will explain shortly later) solution to Problem 3.

Proposition 2: The piecewise constant dynamic task assignment profile $\boldsymbol{a}(t) = \boldsymbol{a}_d(\tau_k), \quad \forall t \in [\tau_k, \tau_{k+1})$ for all $k \in \mathbb{Z}_{\geq 0}$, where $\boldsymbol{a}_d$ is defined as in (13), will converge, as $t \to t_\mathsf{f}$, to a Nash equilibrium of the game $\mathcal{G}^T$, where $T \in [t_\mathsf{f} - \epsilon, t_\mathsf{f}]$.

Proof: Given that $\{\tau_k\}_{k=0}^{\infty}$ is a non-decreasing sequence in $[0, t_\mathsf{f}]$ which converges to t_f as $k \to \infty$, we conclude that $\tau_k \in [t_\mathsf{f} - \epsilon, t_\mathsf{f}]$, for all $k \in \mathbb{Z}_K$. After truncating the first $K-1$ elements of $\{\tau_k\}_{k=0}^{\infty}$, we obtain a new non-decreasing sequence $\{\sigma_n\}_{n=0}^{\infty}$, where $\sigma_n = \tau_{n+K}$, for $n \in \mathbb{Z}_0$, which implies that $\sigma_0 = \tau_K$ and $\lim_{n\to\infty} \sigma_n = t_\mathsf{f}$. In view of Claim 1, the update law $\boldsymbol{a}(\tau_k) := \boldsymbol{\phi}(\boldsymbol{\mathcal{I}}_k; \mathcal{G}^0)$ defined in (12) will converge to a Nash equilibrium of the game $\mathcal{G}^0$ as $k \to \infty$. From real analysis, we know that after truncating the first $K-1$ elements of the convergent sequence $\{\boldsymbol{a}(\tau_k)\}_{k=0}^{\infty}$, we obtain a new sequence $\{\boldsymbol{a}(\sigma_n)\}_{n=0}^{\infty}$ that will remain convergent with the same limit. The previous claim on convergence holds true for any game $\mathcal{G}^{\tau_k}$ for a fixed k when the latter is treated as a static game (with a possibly different limit for each τ_k). We conclude that the sequence $\{\boldsymbol{a}_d(\sigma_n)\}_{n=0}^{\infty}$, where $\boldsymbol{a}_d$ is defined in (13), will also converge to a Nash equilibrium of the game $\mathcal{G}^{\tau_K}$, where by definition $\tau_K \in [t_\mathsf{f} - \epsilon, t_\mathsf{f}]$. This completes the proof. ■

Remark 4 If the individual assignment of the i-agent at time $t = \tau_k$ is different from that at $t = \tau_{k-1}$, then the i-th agent will have to solve Problem 1 with the updated terminal constraint and with her initial state be set equal to $x_i(\tau_k)$ and the final time to $t_\mathsf{f} - \tau_k$.

VI. Numerical Simulations

In this section, we present numerical simulations to illustrate the main ideas of the methods proposed so far. We consider a team of agents with double integrator dynamics, that is, $\ddot{p}_i = u_i$, with $p_i(0) = p_i^0$ and $\dot{p}_i(0) = v_i^0$, where $p_i \in \mathbb{R}^2$ ($p_i^0 \in \mathbb{R}^2$) and $\dot{p}_i \in \mathbb{R}^2$ ($v_i^0 \in \mathbb{R}^2$) denote, respectively, the position and velocity of the i-th agent at time t ($t = 0$), $i \in [1, n]_d$. The performance index is given by $\mathcal{J}(u_i(\cdot)) := (1/2)\int_0^{t_\mathsf{f}} |u_i(t)|^2 \mathrm{d}t$ subject to the terminal constraint $x_i(t_\mathsf{f}) = x_{\mathcal{T}_j}$, where $x_i := (p_i,\ \dot{p}_i) \in \mathbb{R}^4$ and $x_{\mathcal{T}_j} := (p_{\mathcal{T}_j}, 0) \in \mathbb{R}^4$, which means that the i-th agents tries to reach the position $p_{\mathcal{T}_j}$ associated with her assigned task $\mathcal{T}_j$ at time $t = t_\mathsf{f}$ with zero terminal velocity (soft landing). It turns out (see, for instance, [18]) that the optimal control input is given by $u_i^\star(t) = \alpha + t\beta$, $\alpha := (6/t_\mathsf{f}^2)(p_{\mathcal{T}_j} - p_i^0 - t_\mathsf{f} v_i^0) + (2/t_\mathsf{f})v_i^0$, $\beta := -(12/t_\mathsf{f}^3)(p_{\mathcal{T}_j} - p_i^0 - t_\mathsf{f} v_i^0) - (6/t_\mathsf{f}^2)v_i^0$, and the optimal cost-to-go by $\rho_i(x_i^0; x_{\mathcal{T}_j}) := (1/2)(t_\mathsf{f}|\alpha|^2 + t_\mathsf{f}^2 \alpha^\mathrm{T}\beta + (1/3)t_\mathsf{f}^3|\beta|^2)$.

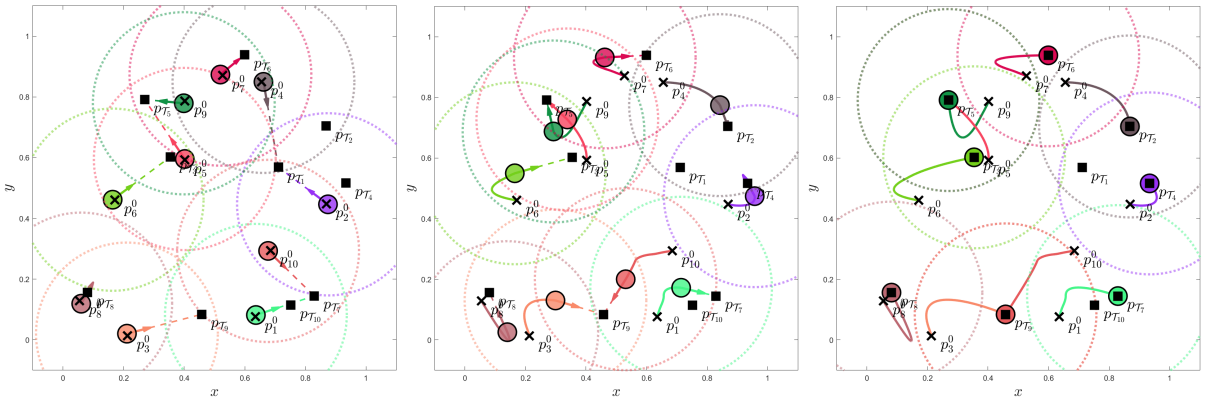
TABLE I
Team Utilities ($n = p = 100$) for range constrained ($\varrho = 0.3$) and unconstrained ($\varrho \to \infty$) cases

	GRM				SAP			
	OLTA		DTA		OLTA		DTA	
$t_\mathsf{f} \setminus \varrho$	∞	0.3	∞	0.3	∞	0.3	∞	0.3
2	33.5	36.7	22.9	32.5	33.1	36.4	31.4	34.7
5	44.3	42.9	43.2	43.0	43.6	41.6	43.2	41.7
10	45.8	43.5	45.8	44.1	45.4	42.6	45.3	43.2

We will present numerical simulations for both Problem 2 (OLTA) and Problem 3 (DTA) based on the SAP and GRM algorithms from [9] for both the range constrained and unconstrained cases. We will use a constant time step δt (although in Proposition 2, we proposed a decreasing time step, it turns out that a sufficiently small constant step is adequate for our simulations). The negotiation process for the OLTA ran for $k = t_\mathsf{f}/\delta t$ rounds (all these rounds took place at time $t = 0$ per the discussion in Section IV-B) in order to converge to a Nash equilibrium before the agents start moving toward the states corresponding to their assigned tasks. The negotiation process for the DTA starts with a random task assignment profile at $t = 0$ and subsequently, the agents continue to update their utilities and individual assignments at every time step ($k = t_\mathsf{f}/\delta t$). We have noticed that δt must be smaller for the SAP algorithm than the GRM algorithm to achieve convergence. For this reason, we select $\delta t = 0.1$ for GRM and $\delta t = 0.01$ for SAP when solving the DTA problem. Per the discussion in Section V-B, the agents' utilities are not updated after time $t = t_\mathsf{f} - \epsilon$ whereas the negotiation process continues until $t = t_\mathsf{f}$. In our simulations, we have used the following parameter values: $\epsilon = t_\mathsf{f}/5$, $p_i^0 \in [0,1]^2$, $\dot{p}_i^0 \in [-0.1, 0.1]^2$, $p_{\mathcal{T}_j} \in [0,1]^2$, $\bar{r}_{\mathcal{T}_j} \in [0,1]$, and $p_{i,j} \in [0,1]$ where $i \in [1,n]_d$ and $j \in [1,p]_d$. For the implementation of the GRM and SAP algorithms, we have used $\rho = 0.1$ (discount factor), $\alpha = 0.5$ (parameter for the agents' willingness to optimize at each time step) and randomization level $\tau = 10/k^2$. Finally, $\varrho = 0.3$ (parameter for range constrained implementations of SAP and GRM). All the graphs and numerical outcomes presented herein are averaged data from 10^2 simulation runs.

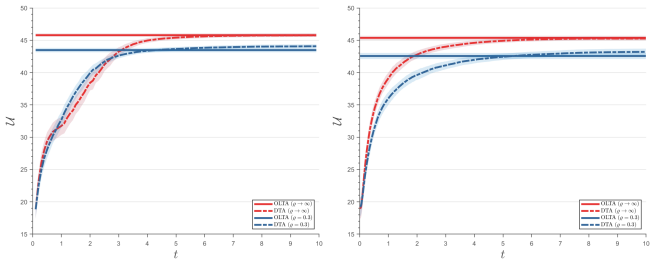
Figure 1 illustrates the evolution of the agents' trajectories computed for the DTA problem at different time instants for the range constrained case. In particular, the dash lines indicate the current task assignments whereas the solid curves correspond to the past segments of the agents' trajectories. Fig. 2 shows that both the team utility obtained by the GRM and SAP negotiation protocols for the DTA problem reach

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.



(a) $t = 0^+$ (b) $t = 5$ (c) $t = 10$

Fig. 1. Dynamic task allocation for the range constrained case (GRM, $n = p = 10$, $\varrho = 0.3$, $t_\mathsf{f} = 10$, $\mathcal{U} = 2.7538$)



(a) GRM (b) SAP

Fig. 2. Team utilities versus time ($n = p = 100$, $t_\mathsf{f} = 10$)

the team utility attained by the solution to the OLTA problem. In addition, the negotiations converge to a pure strategy Nash equilibrium as $t \to t_\mathsf{f}$ in agreement with Proposition 2. Table I shows the values of the total team utility $\mathcal{U}$ for different scenarios for both the range constrained and range unconstrained cases with a significant number of agents and tasks and for different values of the terminal time t_f. We observe that for the DTA problem the team's performance improves as t_f increases. As we have discussed in Remark 3, when t_f is large, then the equilibrium assignment profile corresponds to a "steady-state" profile in which case the performance achieved by the solutions to both the OLTA and DTA problems are expected to be similar. The obtained results confirm the latter claim.

VII. Concluding Remarks

In this paper, we have presented a framework to address dynamic task allocation problems for multi-agent systems with state-dependent utilities. Our approach, which leverages game-theoretic learning algorithms for the solution of static potential games, offers a practical solution to a class of more realistic and challenging dynamic task allocation problems for autonomous mobile agents. In our future work, we plan to extend the results presented herein to even more realistic task allocation problems including scenarios with deadlines attached to tasks, pop-up tasks and agents with varying capabilities and preferences.

References

[1] D. H. Wolpert, K. R. Wheeler, and K. Tumer, "General principles of learning-based multi-agent systems," in *Proceedings of the third annual conference on Autonomous Agents*, pp. 77–83, 1999.

[2] D. Monderer and L. S. Shapley, "Potential games," *Games and economic behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[3] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multi-robot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.

[4] H. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[5] M. Nanjanath and M. Gini, "Repeated auctions for robust task execution by a robot team," *Robotics and Autonomous Systems*, vol. 58, no. 7, pp. 900 – 909, 2010.

[6] J. Capitan, M. T. Spaan, L. Merino, and A. Ollero, "Decentralized multi-robot cooperation with auctioned POMDPs," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 650–671, 2013.

[7] K. S. Macarthur, R. Stranders, S. Ramchurn, and N. Jennings, "A distributed anytime algorithm for dynamic task allocation in multi-agent systems," in *25th AAAI Conf. on Artificial Intelligence*, 2011.

[8] A. T. Tolmidis and L. Petrou, "Multi-objective optimization for dynamic task allocation in a multi-robot system," *Eng. Appl. Artif. Intell.*, vol. 26, no. 5-6, pp. 1458–1468, 2013.

[9] G. Arslan, J. R. Marden, and J. S. Shamma, "Autonomous Vehicle-Target Assignment: A Game-Theoretical Formulation," *J. Dyn. Syst. Meas. Control*, vol. 129, pp. 584–596, 04 2007.

[10] D. Fudenberg and D. Levine, "Learning in games," *European economic review*, vol. 42, no. 3-5, pp. 631–639, 1998.

[11] H. P. Young, *Individual strategy and social structure: An evolutionary theory of institutions*. Princeton University Press, 2020.

[12] J. R. Marden, G. Arslan, and J. S. Shamma, "Joint strategy fictitious play with inertia for potential games," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 208–220, 2009.

[13] G. C. Chasparis, J. S. Shamma, and A. Rantzer, "Perturbed learning automata in potential games," in *CDC (2011)*, pp. 2453–2458, 2011.

[14] Y. Wang and L. Pavel, "A modified Q-learning algorithm for potential games," in *19th IFAC World Congress*, pp. 8710 – 8718, 2014.

[15] A. C. Chapman, R. A. Micillo, R. Kota, and N. R. Jennings, "Decentralized dynamic task allocation using overlapping potential games," *The Computer Journal*, vol. 53, no. 9, pp. 1462–1477, 2010.

[16] J. R. Marden, "State based potential games," *Automatica*, vol. 48, no. 12, pp. 3075–3088, 2012.

[17] D. González-Sánchez and O. Hernández-Lerma, "A survey of static and dynamic potential games," *Science China Mathematics*, vol. 59, no. 11, pp. 2075–2102, 2016.

[18] E. Bakolas, "A decentralized spatial partitioning algorithm based on the minimum control effort metric," in *Proceedings of American Control Conference 2014*, (Portland, OR), pp. 5264–5269, June 2014.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on September 09,2026 at 17:31:39 UTC from IEEE Xplore. Restrictions apply.