Time-constrained leader-follower multi-agent task scheduling and control synthesis

Pian Yu and Dimos V. Dimarogonas

***Abstract*—This paper addresses the problem of time-constrained leader-follower multi-agent task scheduling and control synthesis. The leader-follower multi-agent system is subject to a set of dynamically activated tasks, each of which is associated with a relative deadline and can be completed at different Quality-of-Satisfaction levels. By taking into account the reward and cost of satisfying these tasks, a novel scheduling problem is formulated and a dynamic scheduling strategy is proposed. Based on the dynamic plan, distributed control laws are designed accordingly for the leader and follower agents. Under the condition that the information of the target regions are available only to the leader agents, it is shown that the proposed control laws guarantee the satisfaction of each task at its desired Quality-of-Satisfaction level. A simulation example is given to verify the theoretical results.**

***Index Terms*—Multi-agent systems, Quality-of-Satisfaction, dynamic task scheduling, time constraints, control synthesis.**

I. Introduction

THE integration of multi-agent task scheduling and cooperative control is of great practical interests. Over the past decades, the problem of task scheduling is widely investigated for applications, such as manufacturing [1], traffic systems [2], and service robots [3]. In the area of multi-agent cooperative control, the research has been usually focused on achieving one single global task, such as reference-tracking [4], consensus [5] or formation [6]. Recently, multi-agent cooperative control under complex task specifications, such as temporal logic tasks, has been gaining attention [7]–[9]. However, most efforts have been devoted to static specifications, for which off-line motion planning can be conducted. In practice, a group of agents (robots) may encounter the request of a sequence of tasks which are activated dynamically. Under this circumstance, off-line motion planning is no longer feasible and on-line scheduling is necessary. Furthermore, deadline constraints on the completion of each task is a common requirement, e.g., "visit region A within 10 time units". Therefore, jointly scheduling of deadline-constrained dynamically activated task sequences and designing the distributed controllers for a group of agents is challenging.

In real-time systems, a task is usually characterized by a specific deadline. For such systems, many dynamic scheduling algorithms rely on Earliest Deadline First (EDF) policies [10], [11] and their extensions [12], [13]. The objective of these algorithms is to execute a set of tasks in order to meet the most possible number of deadlines. Reward-Based scheduling (RBS) [14] represents another class of dynamic scheduling algorithms, in which the reward of a task is associated with its execution time rather than the deadline. For example, Increasing Reward with Increasing Service (IRIS) models [15] fall within the scope of this framework. Different from EDF, RBS is capable of modeling tasks that are characterized not only by a deadline, but also by their 'importance'. The performance of the algorithm is then evaluated by computing the cumulative reward gained on a task set.

In this paper, we address the dynamic scheduling problem under the framework of RBS. In the IRIS models, the reward of a task increases with the residual time[1] [15]. In our task model, we consider that a task can be completed at several Quality-of-Satisfaction (QoSa) levels, and each QoSa level corresponds to a time interval within which the task should be completed. However, different from IRIS, the reward of a task does not necessarily increase with the residual time. Apart from reward, the cost of satisfying a set of tasks is further considered, which is defined as the (estimated) total distance travelled by a group of agents. Based on the above setting, a dynamic scheduling strategy is proposed to combine the ideas of EDF and RBS.

On the other hand, to guarantee that the desired performance (in terms of reward and cost) can be achieved based on the dynamic scheduling strategy, one has to ensure that each task is completed at the desired QoSa level. This further requires that each task is completed at the specific time interval associated with the QoSa level, e.g., "visit region A within $[6, 8]$ time units", and brings additional difficulties to the control synthesis. To the best of our knowledge, this type of control design problem under specific time interval constraints is rarely considered in the context of multi-agent systems (MAS). In our previous work [16], multi-agent control under specific deadline constraints was studied and a linear feedback controller was designed. Nevertheless, the control design is not applicable when specific time interval constraints are presented. In [17], [18], multi-agent control under signal temporal logic tasks was investigated and a barrier function based controller was proposed for each agent. However, the control policy requires the knowledge of each agent of the task plan and the target regions. In this paper, we consider a leader-follower MAS, in which only the leader agents know

This work was supported in part by the Swedish Research Council (VR), the Swedish Foundation for Strategic Research (SSF), the Knut and Alice Wallenberg Foundation (KAW), the ERC project LEAFHOUND and the EU project CANOPIES.

The authors are with School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 10044 Stockholm, Sweden. piany@kth.se, dimos@kth.se

[1]The amount of time between the completion time and the absolute deadline.

the information of the target regions. Therefore, it is necessary that the group of agents collaborate for the task completion. In this case, the fully distributed control synthesis is challenging, particular under an explicit time interval constraint.

This paper investigates the jointly design of task scheduling and distributed controller for leader-follower MAS subject to a sequence of dynamically activated tasks. More specifically, each task is associated with a region of interest, where the group of agents need to visit together within a deadline. The main contributions of the paper are twofold: i) the notion of QoSa levels is introduced for completing each task. By associating the reward of completing a set of tasks to the QoSa levels and the cost to the (estimated) total travelling distance, a new task scheduling problem is formulated and a dynamic scheduling strategy is proposed; ii) under the condition that the information of the target regions is available only to the leader agents, distributed control laws are designed respectively for the leader and follower agents. It is shown that the designed control laws can guarantee the satisfaction of each task at the desired QoSa level. We note that the control design is motivated by the funnel control (FC) [20] and the prescribed performance control (PPC) [19], which have been originally used to solve control problems with transient performance constraints [19]–[21] and recently applied to the multi-agent setting [22], [23]. In this work, we propose for the first time a way to transform time interval constraints into transient performance constraints under the leader-follower MAS framework, and hence, FC and PPC can be applied.

This paper generalizes the preliminary conference results of [24] in four main directions. First, we propose a dynamic scheduling strategy for MAS whose goal is to complete a set of dynamically activated tasks. Second, we introduce the hybrid system framework to model the discrete task evolution and the continuous state evolution. Third, the input constraint for each agent is taken into account. Finally, we provide comparisons between the proposed dynamic scheduling strategy and the EDF policy in the simulation. The rest of the paper is organized as follows. In Section II, notation and preliminaries are introduced, while Section III formalizes the considered problem. Section IV presents the proposed solution in detail, which is further verified by simulations in Section V. Conclusions are given in Section VI.

II. Preliminaries

A. Notation

Let $\mathbb{R} := (-\infty,\infty)$, $\mathbb{R}_{\geq 0} := [0,\infty)$. Let $\mathbb{N}$ be the set of natural numbers. Denote by $\mathbb{R}^n$ the n dimensional real vector space, $\mathbb{R}^{n\times m}$ as the $n\times m$ real matrix space. I_n is the identity matrix of order n. For $x_1 \in \mathbb{R}^{n_1},\ldots,x_m \in \mathbb{R}^{n_m}$, the notation $(x_1,x_2,\ldots,x_m) \in \mathbb{R}^{n_1+n_2+\cdots+n_m}$ stands for $[x_1^T,x_2^T,\ldots,x_m^T]^T$. Let $|\lambda|$ be the absolute value of a real number λ, $\|x\|$ and $\|A\|$ be the Euclidean norm of vector x and matrix A, respectively. For a set Ω, 2^Ω represents the power set of Ω and $|\Omega|$ represents the cardinality of Ω. A set-valued mapping from $\mathbb{R}^m$ associates with every point in $\mathbb{R}^m$ a subset of $\mathbb{R}^n$. The notation $M:\mathbb{R}^m \rightrightarrows \mathbb{R}^n$ indicates that M is a set-valued mapping with $M(x)\subset\mathbb{R}^n$ for all $x\in\mathbb{R}^m$. The operators $\cup$ and $\cap$ represent set union and set intersection, respectively. For two sets A and B, the set difference $A\backslash B$ is defined by $A\backslash B := \{x : x\in A \text{ and } x\notin B\}$. In addition, we use $\wedge$ to denote the logical operator AND and $\vee$ to denote the logical operator OR. $P\succ 0$ means that P is a positive definite matrix and P^T is the transpose of P. The Kronecker product is denoted by $\otimes$.

B. Graph Theory

Let $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ be a graph with the set of nodes $\mathcal{V}=1,2,\ldots,N$, and $\mathcal{E}\subseteq\{(i,j) : i,j\in\mathcal{V}, j\neq i\}$ the set of edges. If $(i,j)\in\mathcal{E}$, then node j is called a neighbor of node i and node j can receive information from node i. The neighboring set of node i is denoted by $\mathcal{N}_i=\{j\in\mathcal{V}|(i,j)\in\mathcal{E}\}$. A graph is called undirected if $(i,j)\in\mathcal{E}\Leftrightarrow(j,i)\in\mathcal{E}$. Given an edge $e_k := (i,j)\in\mathcal{E}$, i is called the head of e_k and j is called the tail of e_k. A graph is called connected if for every pair of nodes (i,j), there exists a path which connects i and j, where a path is an ordered list of edges such that the head of each edge is equal to the tail of the following edge.

C. Hybrid system

A hybrid system $\mathcal{H}$ is a tuple $(\mathcal{C},\mathcal{D},F,G)$, where $\mathcal{C}\in\mathbb{R}^n$ and $\mathcal{D}\in\mathbb{R}^n$ are the flow and jump sets, and $F:\mathbb{R}^n\rightrightarrows\mathbb{R}^n$ and $G:\mathbb{R}^n\rightrightarrows\mathbb{R}^n$ the possibly set-valued mappings, called flow and jump maps, respectively. Then $\mathcal{H}$ is represented by

$$\mathcal{H}:\begin{cases}\dot{x}\in F(x) & x\in\mathcal{C}\\ x^+\in G(x) & x\in\mathcal{D},\end{cases} \tag{1}$$

where $x\in\mathbb{R}^n$ is the hybrid state. Roughly speaking, the state can flow through $\mathcal{C}$ following the dynamics given by F, and it can jump in the jump set $\mathcal{D}$ according to G [28]–[30].

III. Problem Formulation

A. Leader-follower MAS

Consider a leader-follower MAS consists of N agents, each of which obeys the following second-order dynamics

$$\begin{aligned}\dot{x}_i(t) &= v_i(t),\\ \dot{v}_i(t) &= u_i(t), \quad i=1,2,\ldots,N,\end{aligned} \tag{2}$$

where $x_i\in\mathbb{R}^n, v_i\in\mathbb{R}^n$ and $u_i\in U_i\subseteq\mathbb{R}^n$ are the position, velocity and control input of agent i, respectively. Let $x=(x_1,\ldots,x_N), v=(v_1,\ldots,v_N), u=(u_1,\ldots,u_N)$ be the stack vector of positions, velocities, and inputs, respectively. The input of agent i is constrained to a set U_i. Without loss of generality, we suppose that the first $n_l, n_l\leq N$ agents are leaders. Then, one can define the leader set $\mathcal{V}_L := \{1,2,\ldots,n_l\}$ and the follower set $\mathcal{V}_F=\{n_l+1,\ldots,N\}$.

The input of each agent is subject to the following constraint $\|u_i\|\leq u_i^{\max}, i\in\mathcal{V}$, where $u_i^{\max}>0$. Then, the input set U_i is given by

$$U_i=\{u\in\mathbb{R}^n : \|u\|\leq u_i^{\max}\}, i\in\mathcal{V}. \tag{3}$$

It is assumed that the graph $\mathcal{G}$ formed by the leader-follower MAS is undirected and connected. Denote by $\bar{x}=(\bar{x}_1,\ldots,\bar{x}_p)$ the p-dimensional stack vector of relative positions of pairs of agents that form an edge in $\mathcal{G}$, where p is the number of edges. The kth element of vector $\bar{x}$, denoted by $\bar{x}_k := x_{ij} = x_i - x_j, k\in\{1,2,\ldots,p\}$, corresponds to an edge $e_k=(i,j)$ in the graph.

B. Task Specifications

For the group of agents, we assume the existence of a set of $M \in \mathbb{N}$ tasks, denoted by $\phi := \{\phi_1, \phi_2, \ldots, \phi_M\}$. To be more specific, each task is associated with a target region that the MAS has to visit within a deadline. To model the dynamic task generation process, we need the notion of a task generation signal. Let the set

$$T_g := \{T_0, T_1, \ldots\} \tag{4}$$

be the sequence of *task generation times*, where $T_0 < T_1 < \cdots$. Without loss of generality, we assume $T_0 = 0$. Then, we define the function $\delta : T_g \to 2^{\phi}$ as the *task generation signal*, which maps each task generation time T_k to a subset of tasks that are activated at T_k. We note that the task generation information, i.e., the task generation times T_k and the subset of tasks being activated at each T_k, is unknown to the scheduler.

The task set ϕ is assumed to have the following features:

1. Tasks are preemptable[2] and activated dynamically;
2. Each task can be activated multiple times. However, a task can not be activated again if the previous activated one is not completed.

Each task $\phi_l, l \in \{1, \ldots, M\}$ is defined as a tuple $\phi_l := (\mathbb{X}_l, a_l, d_l, D_l, A_l, k_l, \mathrm{I}_l, R_l)$, where

- $\mathbb{X}_l \subset \mathbb{R}^n$: the target region associated with task ϕ_l;
- a_l: the set of activation times $a_l := \{a_{l,1}, a_{l,2}, \ldots\} \subseteq T_g$, *i.e.,* the subset of task generation times at which the task ϕ_l is activated and becomes ready to execute;
- d_l: the relative deadline, which is a constant value for each task;
- D_l: the absolute deadline $D_l := \{D_{l,1}, D_{l,2}, \ldots\}$. For task ϕ_l that arrives at $a_{l,k} \in a_l$, $D_{l,k}$ is computed as $D_{l,k} = a_{l,k} + d_l$;
- A_l: the (actual) completion times $A_l := \{A_{l,1}, A_{l,2}, \ldots\}$, where $A_{l,k}$ corresponds to $a_{l,k}$. We note that each $A_{l,k}$ is determined online;
- $k_l \in \mathbb{N}$: the number of QoSa levels for ϕ_l;
- I_l: the interval set $\mathrm{I}_l := \{\mathrm{I}_l[k_l - 1], \ldots, \mathrm{I}_l[1], \mathrm{I}_l[0]\}$, where $\mathrm{I}_l[\hat{k}], \hat{k} \in \{0, 1, \ldots, k_l - 1\}$ is the time interval corresponding to task ϕ_l and QoSa level $\hat{k}$. In addition, I_l satisfies the following properties: i) $\mathrm{I}_l[0] = (d_l, \infty)$; ii) $\cup_{m=1}^{k_l - 1} \mathrm{I}_l[m] = [0, d_l]$; iii) $\mathrm{I}_l[m_1] \cap \mathrm{I}_l[m_2] = \emptyset, \forall m_1 \neq m_2$; and iv) $\forall t_1 \in \mathrm{I}_l[m_1], t_2 \in \mathrm{I}_l[m_2]$, $m_1 > m_2$ implies $t_1 < t_2$;
- R_l : the reward set $R_l := \{R_l[k_l - 1], \ldots, R_l[1], R_l[0]\}$, where $R_l[\hat{k}], \hat{k} \in \{0, 1, \ldots, k_l - 1\}$ represents the reward that task ϕ_l contributes if it is completed at QoSa level $\hat{k}$.

For the sake of notation simplicity, in the following, we use a_l, D_l, A_l to represent any $a_{l,k} \in a_l, D_{l,k} \in D_l, A_{l,k} \in A_l$, respectively, when there is no ambiguity. Note that when a_l corresponds to $a_{l,k}$, D_l, A_l correspond to $D_{l,k}, A_{l,k}$, respectively. Then, we have the following definition.

Definition 1: Given the task ϕ_l and the activation time a_l, we say that ϕ_l *is completed at time* A_l if $x_i(A_l) \in \mathbb{X}_l, \forall i \in \mathscr{V}$. In addition, we say that ϕ_l *is completed at QoSa level* $\hat{k}$ if the (actual) completion time satisfies

$$A_l \in \{a_l + t | t \in \mathrm{I}_l[\hat{k}]\}.$$

[2]Preemptable means that the execution of a uncompleted task can be temporarily halted and later resumed.

Remark 1: Each task ϕ_l has at least two QoSa levels, i.e., $k_l \geq 2$. From the properties i)-iv) of I_l and Definition 1, one can see that a QoSa level of bigger than or equal to 1 means that the task is completed before the deadline, while QoSa level 0 means that the deadline of the task is violated. In addition, a higher QoSa level means that the task is completed faster. However, we note that the reward is not necessarily higher for a higher QoSa level. It is assumed here that for each task ϕ_l, $\mathrm{I}_l[\hat{k}], R_l[\hat{k}], \forall \hat{k} \in \{0, \ldots, k_l - 1\}$ are given a priori and are known to each agent.

In the following, each target set $\mathbb{X}_l$ will be over-approximated by its largest inscribed ball, denote by $\overline{\mathbb{X}}_l$. Let c_l be the Chebyshev center of $\mathbb{X}_l$. Then, $\overline{\mathbb{X}}_l$ can be represented by

$$\overline{\mathbb{X}}_l := \mathscr{B}(c_l, \bar{r}_l) = \{y \in \mathbb{R}^n : \|y - c_l\| \leq \bar{r}_l\}, \tag{5}$$

where $\bar{r}_l$ is the radius of $\overline{\mathbb{X}}_l$. It is assumed that all target regions are compact and have non-empty interiors, *i.e.,* there exist $r_{\min}, r_{\max} > 0$ such that $r_{\min} \leq \bar{r}_l \leq r_{\max}, \forall l$. In addition, the target regions are disjoint, *i.e.,* $\mathbb{X}_{k_1} \cap \mathbb{X}_{k_2} = \emptyset, \forall k_1, k_2 \in \{1, \ldots, M\}, k_1 \neq k_2$. Note that the problem of finding the maximum inscribed ball can be solved offline by several iterative algorithms, such as the one proposed in [25].

Remark 2: In this paper, we consider that the task ϕ_l is completed once all agents lie inside the target region $\mathbb{X}_l$. However, in practice, there might be a (set of) service(s) associated with ϕ_l, for which the group of agents can provide only when being present in the region of interest $\mathbb{X}_l$. Hence, upon the visit to $\mathbb{X}_l$, the group of agents may still need to accomplish services, such as loading or releasing a cargo, etc. Note that in this paper, we do not focus on how these services are being provided at the target region, but rather aim at guaranteeing the necessary preconditions for providing these services: being present at the target region.

C. Reward and cost of a given task set

In this paper, we are interested in the quantitative reward and cost of satisfying the dynamically activated tasks. Let $\eta \in 2^{\phi} \setminus \emptyset$ be any subset of ϕ except $\emptyset$. The index set associated with η is denoted by $\mathbb{I}_\eta$. Let Π_η be the set containing all permutations of $\mathbb{I}_\eta$, and $\pi \in \Pi_\eta$ be $\pi := (\pi[1], \ldots, \pi[|\mathbb{I}_\eta|])$.

Example 1: Let $\eta = \{\phi_1, \phi_5, \phi_7\}$, then $\mathbb{I}_\eta = \{1, 5, 7\}$ and $\Pi_\eta = \{(1,5,7), (1,7,5), (5,1,7), (5,7,1), (7,1,5), (7,5,1)\}$. If $\pi = (5,7,1) \in \Pi_\eta$, then $\pi[1] = 5, \pi[2] = 7, \pi[3] = 1$.

Denote by $\mathbf{P}_\eta(\pi[0]\pi) = \mathbf{P}_\eta(\pi[0]\pi[1]\ldots\pi[|\mathbb{I}_\eta|]) = \mathbb{X}_{\pi[0]} \ldots \mathbb{X}_{\pi[|\mathbb{I}_\eta|]}$ a *path* generated by the task set η with the order of completion given by π, where $\pi[0] := 0$, $\mathbb{X}_{\pi[0]}$ is a collection of the agents' initial states, which thus contains finite number of elements and will be formally defined later.

1) *Reward*: The reward of the path $\mathbf{P}_\eta(\pi[0]\pi)$ is given by

$$R(\mathbf{P}_\eta(\pi[0]\pi)) = \sum_{l=1}^{|\mathbb{I}_\eta|} R_{\pi[l]}[\hat{k}_{\pi[l]}], \tag{6}$$

where $\hat{k}_{\pi[l]} \in \{0, \ldots, k_{\pi[l]} - 1\}, l = 1, \ldots, |\mathbb{I}_\eta|$, represents the QoSa level of task $\phi_{\pi[l]}$. Note that the QoSa levels are defined in terms of the (actual) completion time and each task can be completed at different QoSa levels (and thus return different

rewards). Therefore, $R(\mathbf{P}_\eta(\pi[0]\pi))$ is dependent on the time needed to complete each task.

2) *Cost*: Motivated by [27], the cost of the path $\mathbf{P}_\eta(\pi[0]\pi)$ is defined as the (estimated) distance travelled by the group of agents. Let $W(\mathbb{X}_k,\mathbb{X}_j)\in\mathbb{R}_{\geq 0}$ be the transition cost from set $\mathbb{X}_k$ to $\mathbb{X}_j$, which is given by

$$W(\mathbb{X}_k,\mathbb{X}_j):=\begin{cases}\sum_{x\in\mathbb{X}_k}\|x-c_j\|, & \text{if } k=0, j\in\{1,\ldots,M\}\\ N(\|c_k-c_j\|), & \text{if } k,j\in\{1,\ldots,M\}, k\neq j.\end{cases}$$

Then, the cost of $\mathbf{P}_\eta(\pi[0]\pi)$ is defined as:

$$C(\mathbf{P}_\eta(\pi[0]\pi))=\sum_{k=0}^{|\mathbb{I}_\eta|-1}W(\mathbb{X}_{\pi[k]},\mathbb{X}_{\pi[k+1]}). \tag{7}$$

It is obvious that the completion of the tasks in η can be scheduled in different orders, which corresponds to different paths. Let $\mathbb{P}_\eta=\cup_{\pi\in\Pi_\eta}\{\mathbf{P}_\eta(\pi[0]\pi)\}$ be the set of all paths. In this paper, we want to maximize the reward as well as minimizing the cost. Therefore, we propose the following objective function

$$\mathbf{J}(\mathbb{P}_\eta)\triangleq \max_{\substack{\pi\in\Pi_\eta,\\ \hat{k}_{\pi[l]}\in\{0,\ldots,k_{\pi[l]}-1\},\\ l\in\{1,\ldots,|\mathbb{I}_\eta|\}}} \{\varsigma R(\mathbf{P}_\eta(\pi[0]\pi))-(1-\varsigma)C(\mathbf{P}_\eta(\pi[0]\pi))\}, \tag{8}$$

where $\varsigma\in[0,1]$ is a parameter used to balance between the reward and the cost.

Example 2: Following Example 1, we assume that the number of QoSa levels $k_1=3,k_5=4,k_7=2$. Then $\hat{k}_1\in\{0,1,2\},\hat{k}_5\in\{0,1,2,3\}$ and $\hat{k}_7\in\{0,1\}$. Given one schedule $\pi=(5,7,1)$, we have $R(\mathbf{P}_\eta(\pi[0]\pi))=R_5[\hat{k}_5]+R_7[\hat{k}_7]+R_1[\hat{k}_1]$ and $C(\mathbf{P}_\eta(\pi[0]\pi))=\sum_{x\in\mathbb{X}_0}\|x-c_5\|+N\|c_5-c_7\|+N\|c_7-c_1\|$, where N is the total number of agents, $\mathbb{X}_0:=\{x_1(0),\ldots,x_N(0)\}$. Note that the reward $R(\mathbf{P}_\eta(\pi[0]\pi))$ is further determined by the chosen QoSa level for each task.

D. Problem

In this work, we consider that only the leader agents know the information of the target regions. Therefore, it is necessary for the group of agents (both leader and follower agents) to coordinate to complete the tasks. Let $\eta(t)\in 2^\phi$ be a set of tasks that are active (being activated) at time t. The purpose of this paper is to maximize, in an online fashion, the objective function (8) for the set of active tasks $\eta(t)$. Formally, the problem is stated below.

Problem 1: Given the leader-follower MAS (2) and the task specifications in Section III-B, jointly schedule the dynamically activated tasks and design distributed (with only measurements of states of neighbors) control law u_i for each leader or follower agent i, such that $\mathbf{J}(\mathbb{P}_{\eta(t)})$ is maximized.

IV. Solution

The proposed solution consists of two layers: i) an online dynamic scheduling layer and ii) a (distributed) control synthesis layer which guarantees that the group of agents (both leader and follower) arrive at their progressive target regions at the corresponding QoSa levels at all times.

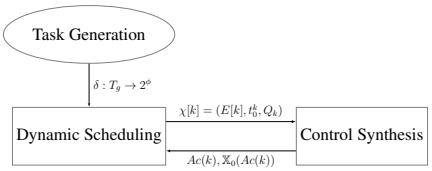

Fig. 1. The relation between dynamic scheduling and control synthesis.

The relation between the two layers is depicted in Fig. 1. The dynamic scheduling layer determines when and which task should be executed at which QoSa level. Let χ be the execution task sequence generated by the scheduling layer, which records the sequence of tasks being executed. The kth element of χ, denoted by $\chi[k]$, contains the information of the kth execution task. It is represented by a triple $\chi[k]=(E[k],t_0^k,Q_k)$, where $E[k]\in\phi$ represents the kth execution task, $t_0^k\in\mathbb{R}_{\geq 0}$ is the time that $E[k]$ starts execution, and Q_k is the desired QoSa level of $E[k]$, respectively. $\chi[k]$ will be used for the control synthesis. The (actual) completion time of the task $E[k]$, denoted by $Ac(k)$, is determined online by the synthesized controller. Define $\mathbb{X}_0(Ac(k)):=\{x_1(Ac(k)),\ldots,x_N(Ac(k))\}$ as the set which contains the position information of the MAS at time $Ac(k)$. As shown in Fig. 1, once $E[k]$ is completed, the information $Ac(k),\mathbb{X}_0(Ac(k))$ will be sent to the dynamic scheduling layer for generating the next execution task.

A. Hybrid structure

The task scheduling is conducted at a sequence of discrete times while the evolution of the states of each agent is continuous. Therefore, in this subsection, the hybrid system framework is introduced to model the whole process.

As stated previously, the information of the target regions is known only to the leader agents. Therefore, different control laws have to be synthesized for the leader and follower agents, respectively. In addition, when there is no task left to be executed, *i.e.,* $\chi[k]=\emptyset$, all agents will switch to idle mode. To cope with these variations, we propose three different control modes for each agent i:

i) the leader mode: $u_i=u_i^{\text{lead}}$;
ii) the follower mode: $u_i=u_i^{\text{fol}}$;
iii) the idle mode: $u_i=u_i^{\text{idle}}$,

where $u_i^{\text{lead}},u_i^{\text{fol}}$ and u_i^{idle} will be defined later.

The execution task sequence χ is updated either at the task generation time $t=T_l,T_l\in T_g$ or at the (actual) completion time $Ac(k)$ of the current execution task $\chi[k]$ (*i.e.,* $x_i(Ac(k))\in\mathbb{X}_{\mathbb{I}(E[k])},\forall i$). Here, $\mathbb{I}(E[k])$ returns the index of task $E[k]$. If at $Ac(k)$, there is no task to be executed next (*i.e.,* $E[k+1]=\emptyset$), all agents will switch to idle mode. We note that $\chi[k]=\emptyset$ if $E[k]=\emptyset$. Each $\chi[k]$ determines uniquely a control law u_i for each agent i, which is denoted by $u_i^{\chi[k]}$.

To model the discrete behavior of the scheduling and the control law switching, an integer variable $k\in\mathbb{N}$ is introduced, which is initialized to be 0. It remains constant during flows and it is incremented by 1 once a new execution task is generated. In other words,

$$F:\begin{cases}\dot{x}=v\\ \dot{v}=u^{\chi[k]}\\ \dot{\tau}=1\\ \dot{k}=0\end{cases}\qquad G:\begin{cases}x^+=x\\ v^+=v\\ \tau^+=\tau\\ k^+=k+1\end{cases} \tag{9}$$

where $u^{\chi[k]} = (u_1^{\chi[k]}, \ldots, u_N^{\chi[k]})$, and

$$u_i^{\chi[k]} = \begin{cases} u_i^{\text{idle}}, \text{ if } \chi[k] = \emptyset, \\ u_i^{\text{lead}}, \text{ if } \chi[k] \neq \emptyset \text{ and } i \in \mathscr{V}_L, \\ u_i^{\text{fol}}, \text{ if } \chi[k] \neq \emptyset \text{ and } i \in \mathscr{V}_F. \end{cases} \tag{10}$$

The jump set $\mathscr{D}$ is given by two kinds of events, that is

$$\mathscr{D} := \{\underbrace{\exists T_l \in T_g \text{ s.t. } \tau = T_l}_{\text{new tasks activated}} \vee \underbrace{x_i(t) \in \mathbb{X}_{\mathbb{I}(E[k])}, \forall i}_{\text{current task completed}}\}, \tag{11}$$

and the flow set $\mathscr{C}$ is given by

$$\mathscr{C} := \mathbb{R}^{2nN} \times \mathbb{R}_{\geq 0} \times \mathbb{N} \setminus \mathscr{D}. \tag{12}$$

At jumps, the control input for each agent will be updated according to the next execution task. The design strategy will be given in the later subsection.

Remark 3: The hybrid system (9) is introduced to model the discrete task evolution and the continuous state evolution of the MAS (2). It will become clear later in the control synthesis section that the control design for each agent and the analysis are conducted on a single task $\chi[k]$. We note that there is no jump during the execution of $\chi[k]$. Therefore, the hybrid time domain (t, j) and the solutions of hybrid systems on (t, j) are redundant and thus not introduced.

B. Dynamic scheduling algorithm

The dynamic scheduling process is shown in Fig. 2, which consists of two algorithms: *Arrival* and *Completion*.

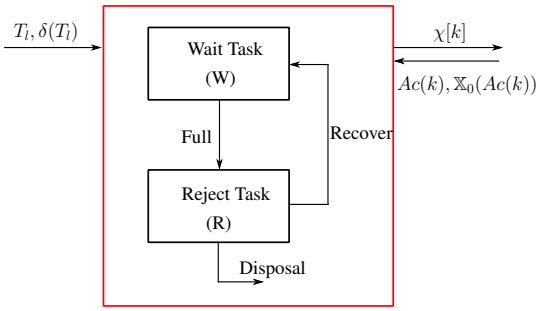

Fig. 2. Dynamic scheduling process.

Let W and R be the ordered wait and (temporarily) reject task set, respectively. Define $W[l] \in \phi$ and $R[l] \in \phi$ as the lth element of W and R, respectively. The index set associated with W is denoted by $\mathbb{I}_W$ and the set of permutations of $\mathbb{I}_W$ is denoted by Π_W. In this paper, it is assumed that the cardinality (length) of the wait task set W is limited (for computation efficiency), given by $\bar{m}$. In particular, there are at most $\bar{m}$ tasks maintained within it. However, we note that there are extra positions in W, which can be occupied instantaneously.

At the task generation time instant T_l, the set of tasks $\delta(T_l)$ is activated. Then, the algorithm *Arrival*, *i.e.,* Algorithm 1, will be activated to determine (among all active tasks) which task should be executed. Due to the fact that the cardinality of the wait task set W is limited, one needs to check whether there are enough spaces in W before scheduling. Otherwise, we first schedule all the active tasks (including the current executing task $E[k]$, the set of newly activated tasks $\delta(T_l)$ and all the tasks in W) according to the EDF policy. Then, the first $\bar{m}$ tasks are kept in W while the remaining tasks are put into the (temporarily) reject list R (lines 1-5). After this process, we (re)schedule the tasks in W using the optimization program $\mathscr{P}(W, \mathbb{X}_0(T_l), T_l)$. The first task in line, *i.e.,* $\pi^*[1]$, is chosen as the next executing task, and the corresponding desired QoSa level, is given by $\hat{k}_{\pi^*[1]}$ (lines 6-8).

Algorithm 1 *Arrival*

Input: $E[k], W, R, T_l, \delta(T_l)$ and $\mathbb{X}_0(T_l) \leftarrow \{x_1(T_l), \ldots, x_N(T_l)\}$.
Output: $E[k+1], t_0^{k+1}, Q_{k+1}$.

1: $W \leftarrow E[k] \cup W \cup \delta(T_l)$,
2: **if** $|W| \geq \bar{m} + 1$ **then**,
3: Schedule W according to the EDF policy,
4: $W \leftarrow W \backslash \{W[\bar{m}+1], \ldots W[|W|]\}, T \leftarrow T \cup \{W[\bar{m}+1], \ldots W[|W|]\}$,
5: **end if**
6: Solve the optimization program $\mathscr{P}(W, \mathbb{X}_0(T_l), T_l)$, which returns $\pi^*, \hat{k}_{\pi^*}$ (see (13)),
7: $E[k+1] \leftarrow \pi^*[1]$, $t_0^{k+1} \leftarrow T_l$, $Q_{k+1} \leftarrow \hat{k}_{\pi^*[1]}$,
8: $W[i] \leftarrow \pi^*[i+1]$, $i = 1, \ldots, |W| - 1$.

In line 6, the optimization program $\mathscr{P}(W, \mathbb{X}_0, t)$ is given by:

$$\max_{\pi \in \Pi_W, \hat{k}_\pi} \varsigma \sum_{l=1}^{|\mathbb{I}_W|} R_{\pi[l]} - (1-\varsigma) \sum_{l'=0}^{|\mathbb{I}_W|-1} W(\mathbb{X}_{\pi[l']}, \mathbb{X}_{\pi[l'+1]}) \tag{13a}$$

subject to

$$\hat{k}_{\pi[l]} \in \{0, 1, \ldots, k_{\pi[l]} - 1\}, \tag{13b}$$
$$R_{\pi[l]} := R_{\pi(l)}[\hat{k}_{\pi[l]}], \tag{13c}$$
$$E_{\pi[l]}[0] = a_{\pi[l]} + d_{\pi[l]} + \varepsilon, \varepsilon > 0, \tag{13d}$$
$$E_{\pi[l]}[\hat{k}_{\pi[l]}] = a_{\pi[l]} + \sup\{\mathrm{I}_{\pi[l]}[\hat{k}_{\pi[l]}]\}, \quad \hat{k}_{\pi[l]} \geq 1, \tag{13e}$$
$$E_{\pi[1]}[\hat{k}_{\pi[1]}] > t + t_{\min}(\mathbb{X}_0, \mathbb{X}_{\pi[1]}), \tag{13f}$$
$$E_{\pi[l+1]}[\hat{k}_{\pi[l+1]}] > E_{\pi[l]}[\hat{k}_{\pi[l]}] + t_{\min}(\mathbb{X}_{\pi[l]}, \mathbb{X}_{\pi[l+1]}), \quad l = 1, \ldots, |\mathbb{I}_W| - 1, \tag{13g}$$

where $\hat{k}_\pi := \{\hat{k}_{\pi[1]}, \hat{k}_{\pi[2]}, \ldots, \hat{k}_{\pi[|\mathbb{I}_W|]}\}$. In (13f) and (13g),

$$t_{\min}(\mathbb{X}_{\pi[l]}, \mathbb{X}_{\pi[l+1]}) := \max_{i \in \mathscr{V}} \min T_i^f \tag{14a}$$

subject to

$$x_i(0) = \mathbb{X}_{\pi[l]}[i], v_i(0) = 0, \quad l = 0, \tag{14b}$$
$$x_i(0) = c_{\pi[l]}, v_i(0) = 0, \quad l \geq 1, \tag{14c}$$
$$\dot{x}_i = v_i, \dot{v}_i = u_i, \tag{14d}$$
$$u_i \in U_i, \tag{14e}$$
$$x_i(T_i^f) \in \mathbb{X}_{\pi[l+1]}, \tag{14f}$$

representing the (estimated) minimal time required to drive the MAS from set $\mathbb{X}_{\pi[l]}$ to set $\mathbb{X}_{\pi[l+1]}$ under the input constraints (14e). Note that in (14b), $\pi[0] = 0$ and $\mathbb{X}_{\pi[0]}[i] = \mathbb{X}_0[i]$ is the ith element of set $\mathbb{X}_0$, where $\mathbb{X}_0$ contains the initial state of the group of agents. The optimal solution of $\mathscr{P}(W, \mathbb{X}_0, t)$ is given by $\pi^* = \{\pi^*[1], \ldots, \pi^*[|\mathbb{I}_W|]\}$ and $\hat{k}_{\pi^*} = \{\hat{k}_{\pi^*[1]}, \ldots, \hat{k}_{\pi^*[|\mathbb{I}_W|]}\}$. In (13d)-(13g), the notation $E_{\pi[l]}[\hat{k}_{\pi[l]}]$ represents the estimated completion time of task $\phi_{\pi[l]}$ at QoSa level $\hat{k}_{\pi[l]}$. It is defined as $a_{\pi[l]} + d_{\pi[l]} + \varepsilon$ for QoSa level 0 and $a_{\pi[l]} + \sup\{\mathrm{I}_l[\hat{k}_{\pi[l]}]\}$ for QoSa level $\hat{k}_{\pi[l]} \geq 1$, as seen in (13d) and (13e) respectively.

The constant ε is used to distinguish between $E_{\pi[l]}[1]$ and $E_{\pi[l]}[0]$. Conditions (13f) and (13g) mean that the estimated completed time of the $l+1$th task in line must be bigger than the estimated completed time of the lth task plus the minimal time required to move from $\mathbb{X}_{\pi[l]}$ to $\mathbb{X}_{\pi[l+1]}$.

Algorithm 2 *Completion*

Input: W, R, $Ac(k)$ and $\mathbb{X}_0(Ac(k)) \leftarrow \{x_1(Ac(k)),\ldots,x_N(Ac(k))\}$.
Output: $E[k+1], t_0^{k+1}, Q_{k+1}$.
1: Remove all infeasible tasks in W and R,
2: **if** $R \neq \emptyset$, **then**
3: Schedule R according to the EDF policy,
4: Move tasks from R to W, until either W is full or R is empty,
5: **end if**
6: **if** $W = \emptyset$, **then**
7: $E[k+1] \leftarrow \emptyset, t_0^{k+1} \leftarrow \emptyset, Q_{k+1} \leftarrow \emptyset$,
8: All agents switch to idle mode,
9: **else**
10: Solve algorithm $\mathscr{P}(W, \mathbb{X}_0(Ac(k)), Ac(k))$, which returns $\pi^*, \hat{k}_{\pi^*}$,
11: $E[k+1] \leftarrow \pi^*[1]$, $t_0^{k+1} \leftarrow Ac(k)$, $Q_{k+1} \leftarrow \hat{k}_{\pi^*[1]}$,
12: $W[l] \leftarrow \pi^*[l+1]$, $l = 1,\ldots,|W|-1$.
13: **end if**

When the kth execution task is completed at time $Ac(k)$, algorithm *Completion* (Algorithm 2), is activated to determine the next execution task. We note that if the activating time and the completion time coincide, algorithm *Arrival* will run first and then algorithm *Completion* is executed. As one can see in (13), the optimization program $\mathscr{P}(W, \mathbb{X}_0, t)$ admits tasks with QoSa level 0 as feasible solutions. Therefore, before rescheduling the wait list, one needs to further inspect to ensure that tasks in the wait list W and the temporarily reject list T are feasible at the current time $Ac(k)$. All infeasible tasks, *i.e.,* $\{\phi_l : \phi_l \in W \cup R, D_l \leq Ac(k)\}$ will be removed forever (line 1). After the inspection, if there are empty positions in W, then tasks in R will be recovered (lines 2-5). At this stage, if $W = \emptyset$, it means that there is no feasible task to be executed next, and then all agents will switch to idle mode (lines 6-9). The idle mode of an agent will be defined later. Otherwise, the wait list W will be rescheduled according to (13), and the first task in W will be chosen as the next execution task and removed from W (lines 10-13).

Remark 4: In the optimization program (13), tasks with QoSa level 0 are considered as feasible and kept in W or R for robustness considerations. The reason is that in real-time execution, a task may be completed before its estimated completion time, and in this case, the previous infeasible tasks (task with QoSa level 0) may become feasible again.

Remark 5: One can see from Algorithm 2 that rescheduling will be conducted (whenever a task is completed) even though there are no new tasks being generated. This is also due to the fact that a task may be completed before its estimated completion time, allowing for other possibilities of schedule. In addition, we note that if the current execution task $E[k]$ is uncompleted when new tasks are activated, then it is possible that $E[k]$ is preempted by a newly activated task, and resumed later.

Remark 6: The optimization program $\mathscr{P}(W, \mathbb{X}_0, t)$ can be viewed as a two-layer optimization problem. The first (outer) layer is a combinatorial optimization problem, where the set of permutations of $\mathbb{I}_W$, *i.e.,* Π_W needs to be computed. The second (inner) layer is a mixed integer linear program. This is because for each $\pi \in \Pi_W$, the cost function (only nonlinear term) $\sum_{l'=0}^{|\mathbb{I}_W|-1} W(\mathbb{X}_{\pi[l']}, \mathbb{X}_{\pi[l'+1]})$ is a constant. In general, finding a solution can be difficult when the cardinality of the wait task set W is large. To cope with this issue, we assume that the cardinality of W is upper bounded by $\bar{m}$. Moreover, we note that the only integer constraint is (13b). Nevertheless, since the number of QoSa levels, given by $k_{\pi[l]}$, is finite for each task, each integer parameter $\hat{k}_{\pi[l]}$ has only a limited number of choices. Therefore, we conclude that the optimization program $\mathscr{P}(W, \mathbb{X}_0, t)$ can be solved efficiently since it is equivalent to solving a finite number ($\leq \bar{m}! \sum_{l \in \mathbb{I}_W} k_l$) of linear programs.

Remark 7: The purpose of this paper is to maximize the objective function $\mathbf{J}(\mathbb{P}_{\eta(t)})$. Therefore, fulfillment of all tasks is a soft constraint in view of the aforementioned maximization. However, we note that by setting the absolute value of the rejection penalty $|R_l[0]|, \forall l$, to be high enough (e.g., higher than the sum of the highest rewards of all tasks being activated), one can verify that the proposed dynamic scheduling algorithm is optimal in the sense that the *miss ratio*[3] is minimized.

C. Control law synthesis

Given the information $\chi[k] = (E[k], t_0^k, Q_k)$, the next step is to do control synthesis. If $\chi[k] = \emptyset$, all agents will switch to the idle mode, and the control law is given by

$$u_i^{\text{idle}} = -\sum_{j \in \mathscr{N}_i} (x_i - x_j) - v_i, \forall i \in \mathscr{V}.$$

Otherwise, we assume without loss of generality that

$$E[k] = \phi_l,\ \mathrm{I}_l[Q_k] = (\underline{t}, \bar{t}], \tag{15}$$

which means that the kth task being executed is ϕ_l and the time interval corresponding to QoSa level Q_k of task ϕ_l is $(\underline{t}, \bar{t}]$. Then, we have the following proposition.

Proposition 1: Given the task ϕ_l and the starting time t_0^k, ϕ_l is completed at QoSa level Q_k if the conditions

C1. $\forall t \in [t_0^k, a_l + \underline{t}]$, $\exists i \in \mathscr{V}_L$ such that $x_i(t) \notin \mathbb{X}_l$; and
C2. $\exists t \in (a_l + \underline{t}, a_l + \bar{t}], \forall i \in \mathscr{V}$ such that $x_i(t) \in \mathbb{X}_l$

hold simultaneously, where a_l is the activation time of ϕ_l.

Before presenting the control synthesis, the following assumption is needed.

Assumption 1: Given the leader-follower MAS (2) with the leader set $\mathscr{V}_L$ and the target region $\mathbb{X}_l, l \in \{1,\ldots,M\}$, with its Chebyshev center c_l. There exist constants $\sigma, d_s > 0$ such that

- $\mathscr{B}(c_l, \sigma) \subseteq \mathbb{X}_l$ and
- $x_i \in \mathscr{B}(c_l, \sigma), \forall i \in \mathscr{V}_L$ and $\|x_i - x_j\| \leq d_s, \forall (i,j) \in \mathscr{E}$, imply $x_i \in \mathbb{X}_l, \forall i \in \mathscr{V}$.

[3]For a sequence of dynamically activated tasks, the *miss ratio* is defined as the number of reject tasks divided by the total number of activated tasks [26].

Remark 8: Assumption 1 is not conservative because i) the constants σ, d_s can be different for different target regions and ii) one feasible choice of σ, d_s for $\mathbb{X}_l$ is $\sigma = \underline{r}_l/2, d_s = \underline{r}_l/(2(N-1))$, where $\underline{r}_l$ is the radius of the minimal enclosing ball of $\mathbb{X}_l$ centered at c_l.

In addition, the following definitions are introduced.

Definition 2: [19] A function $\rho : \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$ will be called a *performance function* if ρ is bounded, nonnegative and non-increasing.

Definition 3: [19] A function $S : \mathbb{R} \to \mathbb{R}$ will be called a *transformation function* if S is strictly increasing, injective and admitting an inverse.

In particular, we define two transformation functions S_1 and S_2 as $S_1(z) = \ln(\frac{1}{1-z})$ and $S_2(z) = \ln(\frac{1+z}{1-z})$.

For leader agent $i \in \mathscr{V}_L$, we prescribe the norm of the tracking error $x_i(t) - c_l$ within the following bounds,

$$\alpha_i(t) < \|x_i(t) - c_l\| < \beta_i(t), \quad i \in \mathscr{V}_L. \tag{16}$$

However, since the follower agents do not know where the target region is, we prescribe the norm of the relative distance between neighboring agents within the following bounds,

$$\|x_{ij}(t)\| < \gamma_i(t), \quad i \in \mathscr{V}, (i,j) \in \mathscr{E}, \tag{17}$$

where $\alpha_i(t), \beta_i(t), \gamma_i(t)$ are performance functions to be defined.

The dynamic scheduling algorithm in Section IV-A ensures that $a_l + \bar{t} > t_0^k$. However, it is possible that $a_l + \underline{t} \leq t_0^k$. In this case, C1 of Proposition 1 is meaningless, and one only needs to design performance functions such that C2 of Proposition 1 is satisfied. Nevertheless, in the other case, *i.e.,* $a_l + \underline{t} > t_0^k$, it is more complicated to design the performance functions since one needs to ensure that both C1 and C2 are satisfied. Therefore, in the following, we will present the design of the performance functions and the control synthesis according to the two different cases, respectively.

Remark 9: Note that if at time t_0^k, one has $x_i(t_0^k) \in \mathbb{X}_l, \forall i \in \mathscr{V}$, *i.e.,* all agents are already in the target region $\mathbb{X}_l$ at t_0^k, then according to the proposed dynamic scheduling strategy, the algorithm *Completion* will be activated, and the group of agents will proceed to the next task (no control action is needed).

1) Case I: $a_l + \underline{t} \leq t_0^k$: Define

$$\alpha_i(t) = 0, \tag{18a}$$
$$\beta_i(t) = \beta_{i0} e^{-\kappa_{i,1}(t - t_0^k)}, \tag{18b}$$
$$\gamma_i(t) = \gamma_{i0} e^{-\mu_{i,1}(t - t_0^k)}, \tag{18c}$$

for $t \geq t_0^k$, where $\beta_{i0} > \max\{\|x_i(t_0^k) - c_l\|, \sigma\}$, $\gamma_{i0} > \max\{\max_{j \in \mathscr{N}_i}\{\|x_{ij}(t_0^k)\|\}, d_s\}$, and σ, d_s satisfy Assumption 1. In addition,

$$\kappa_{i,1} = \frac{1}{(a_l + \bar{t} - t_0^k)} \ln \frac{\beta_{i0}}{\sigma}, \tag{19a}$$
$$\mu_{i,1} = \frac{1}{(a_l + \bar{t} - t_0^k)} \ln \frac{\gamma_{i0}}{d_s}. \tag{19b}$$

Remark 10: The definitions of β_{i0}, γ_{i0} guarantee that the performance bounds (16) and (17) are satisfied at the starting time t_0^k. In addition, from (18b) and (19a) one can get

$$\|x_i(a_l + \bar{t}) - c_l\| < \beta_i(a_l + \bar{t}) = \beta_{i0} e^{-\kappa_{i,1}(a_l + \bar{t} - t_0^k)} = \sigma, \forall i \in \mathscr{V}_L. \tag{20}$$

Moreover, from (18c) and (19b) one can get

$$\|x_{ij}(a_l + \bar{t})\| < \gamma_i(a_l + \bar{t}) = \gamma_{i0} e^{-\mu_{i,1}(a_l + \bar{t} - t_0^k)} = d_s, \forall (i,j) \in \mathscr{E}. \tag{21}$$

According to Assumption 1, (20) and (21) together imply $x_i(a_l + \bar{t}) \in \mathbb{X}_l, \forall i \in \mathscr{V}$. Therefore, C2 is satisfied.

Based on Remark 10, one can conclude that if for task ϕ_l,

i). the tracking error $\|x_i - c_l\|, \forall i \in \mathscr{V}_L$ is evolving within the prescribed performance bound (16),

ii). the relative distance $\|x_{ij}\|, \forall (i,j) \in \mathscr{E}$ is evolving within the prescribed performance bound (17)

for $t \geq t_0^k$, then the task ϕ_l will be completed at the desired QoSa level Q_k.

Define the normalized errors as

$$\xi_i(t) := \frac{\|x_i(t) - c_l\|}{\beta_i(t)}, \quad \xi_{ij}(t) := \frac{\|x_{ij}(t)\|}{\gamma_i(t)}, \tag{22}$$

respectively. Then, the corresponding sets

$$D_{\xi_i} := \{\xi_i(t) : \xi_i(t) \in [0,1)\} \tag{23}$$
$$D_{\xi_{ij}} := \{\xi_{ij}(t) : \xi_{ij}(t) \in [0,1)\} \tag{24}$$

are equivalent to (16) and (17), respectively. The normalized errors ξ_i and ξ_{ij} are transformed through the transformation function S_1. We denote the transformed error $\zeta_i(\xi_i)$ and $\varepsilon_{ij}(\xi_{ij})$ by

$$\zeta_i(\xi_i) := S_1(\xi_i), \quad \varepsilon_{ij}(\xi_{ij}) := S_1(\xi_{ij}), \tag{25}$$

respectively, where we dropped the time argument t for notation convenience.

Let $\nabla S_1(\xi_i) = \partial S_1(\xi_i)/\partial \xi_i, \nabla S_1(\xi_{ij}) = \partial S_1(\xi_{ij})/\partial \xi_{ij}$. The control laws for the leader and follower agents are proposed respectively as:

$$\begin{aligned} u_i^{\text{lead}} = &- \sum_{j \in \mathscr{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} \\ &- \frac{1}{\beta_i} \nabla S_1(\xi_i) \zeta_i(\xi_i) \mathbf{n}_i - K_i v_i, \quad i \in \mathscr{V}_L, \end{aligned} \tag{26}$$

and

$$u_i^{\text{fol}} = - \sum_{j \in \mathscr{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} - K_i v_i, , \quad i \in \mathscr{V}_F, \tag{27}$$

where $\mathbf{n}_i = (x_i - c_l)/\|x_i - c_l\|, \mathbf{n}_{ij} = (x_i - x_j)/\|x_{ij}\|$, and K_i is a positive control gain to be determined later.

Let $\bar{\xi}_k = \xi_{ij}, \bar{\varepsilon}_k = \varepsilon_{ij}, y_i = x_i - c_l$. Let also $\bar{\varepsilon} = (\bar{\varepsilon}_1(\bar{\xi}_1), \ldots, \bar{\varepsilon}_p(\bar{\xi}_p)), \bar{\zeta} = (\zeta_1(\xi_1), \ldots, \zeta_N(\xi_N))$ be the stack vector of the transformed errors and $y = (y_1, \ldots, y_N)$. Define the following function

$$\begin{aligned} V_1(y, v, \bar{\varepsilon}, \bar{\zeta}) = &\frac{1}{2} \begin{bmatrix} y & v \end{bmatrix} \left\{ \begin{bmatrix} K\theta & \theta \\ \theta & I_N \end{bmatrix} \otimes I_n \right\} \begin{bmatrix} y \\ v \end{bmatrix} \\ &+ \frac{1}{2} \bar{\varepsilon}^T \bar{\varepsilon} + \frac{1}{2} \bar{\zeta}^T H \bar{\zeta}, \end{aligned} \tag{28}$$

where $H \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries h_i ($h_i = 1$ if $i \in \mathscr{V}_L$ and $h_i = 0$ otherwise), $K \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries K_i, and θ is a diagonal matrix with entries $\theta_i = \max\{\mu_{i,1}, \kappa_{i,1}\}$.

Let $V_1^0 := V_1(y(t_0^k), v(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k)$ and $\Theta := S_1^{-1}(\sqrt{2V_1^0})$. Then, the following holds.

Theorem 1: Consider the leader-follower MAS (2) and the kth execution task $\chi[k]$ given in (15). The control inputs for the leader and follower agents are given by (26) and (27), respectively. Suppose Assumption 1 holds and $a_l + \underline{t} \le t_0^k$. Assume that the control gain $K_i > \max\{\mu_{i,1}, \kappa_{i,1}\}, \forall i$, and the input constraints for leader and follower agents satisfy

$$u_i^{\max} \ge \left(\frac{|\mathscr{N}_i|}{d_s(1-\Theta)} + \frac{1}{\sigma(1-\Theta)}\right)\sqrt{2V_1^0} + K_i(\sqrt{2V_1^0} + \theta_i\beta_{i0}), \quad \forall i \in \mathscr{V}_L,$$

and

$$u_i^{\max} \ge \frac{|\mathscr{N}_i|}{d_s(1-\Theta)} \ln\frac{1}{(1-\Theta)} + K_i(\sqrt{2V_1^0} + \theta_i\beta_{i0}), \forall i \in \mathscr{V}_F,$$

respectively. Then, i) the tracking error $\|x_i - c_l\|, \forall i \in \mathscr{V}_L$ and the relative distance $\|x_{ij}\|, (i,j) \in \mathscr{E}$ will evolve respectively within the prescribed performance bounds (16) and (17) for all $t \ge t_0^k$ and ii) the input constraint for each agent i, *i.e.,* $u_i(t) \in U_i, \forall i$, will be satisfied for all $t \in [t_0^k, a_l + \bar{t}]$.

Proof: The proof is provided in Appendix. □

2) Case II: $a_l + \underline{t} > t_0^k$: For a given set $\mathbb{X}$, the indicator function is defined as

$$\mathbb{1}_{\mathbb{X}}(x) = \begin{cases} 1, & \text{if} \quad x \in \mathbb{X} \\ 0, & \text{if} \quad x \notin \mathbb{X}. \end{cases}$$

Then, the performance functions $\alpha_i, \beta_i, \gamma_i$ are defined as

$$\alpha_i(t) = \begin{cases} 0, & t \ge t_0^k, \quad \text{if } \mathbb{1}_{\bar{\mathbb{X}}_l}(x_i(t_0^k)) = 1, \\ 0, & t > a_l + \underline{t}, \quad \text{if } \mathbb{1}_{\mathbb{X}_l}(x_i(t_0^k)) = 0, \\ \alpha_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, t \in [t_0^k, a_l + \underline{t}], \text{ if } \mathbb{1}_{\bar{\mathbb{X}}_l}(x_i(t_0^k)) = 0 \end{cases} \tag{29a}$$

$$\beta_i(t) = \begin{cases} \beta_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, & t \in [t_0^k, a_l + \underline{t}] \\ \beta_i(\underline{t})e^{-\kappa_{i,3}(t-a_l-\underline{t})}, & t > a_l + \underline{t}, \end{cases} \tag{29b}$$

$$\gamma_i(t) = \begin{cases} \gamma_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, & t \in [t_0^k, a_l + \underline{t}] \\ \gamma_i(\underline{t})e^{-\mu_{i,2}(t-a_l-\underline{t})}, & t > a_l + \underline{t}, \end{cases} \tag{29c}$$

where

$$\alpha_{i0} = \|x_i(t_0^k) - c_l\| - \Delta_i, \tag{30a}$$

$$\beta_{i0} = \|x_i(t_0^k) - c_l\| + \Delta_i, \tag{30b}$$

$$\gamma_{i0} > \max\left\{\max_{j\in\mathscr{N}_i}\{\|x_{ij}(t_0^k)\|\}, d_s\right\}, \tag{30c}$$

and $0 < \Delta_i < \|x_i(t_0^k) - c_l\| - \bar{r}_l, \forall i \in \mathscr{V}_L$. In addition,

$$\kappa_{i,2} = \frac{1}{(a_l + \underline{t} - t_0^k)} \ln\frac{\alpha_{i0}}{\bar{r}_l}, \tag{31a}$$

$$\kappa_{i,3} = \frac{1}{(\bar{t} - \underline{t})} \ln\frac{\beta_{i0}\bar{r}_l}{\sigma\alpha_{i0}}, \tag{31b}$$

$$\mu_{i,2} = \frac{1}{(\bar{t} - \underline{t})} \ln\frac{\gamma_{i0}\bar{r}_l}{d_s\alpha_{i0}}, \tag{31c}$$

where $\bar{r}_l$ defined in (5) is the radius of $\overline{\mathbb{X}}_l$.

One can verify that the functions $\alpha_i, \beta_i, \gamma_i$ satisfy Definition 2. The function α_i is designed to ensure condition C1. The functions β_i, γ_i are designed to ensure condition C2. However, for the special case $x_i(t_0^k) \in \overline{\mathbb{X}}_l, \forall i \in \mathscr{V}_L$, *i.e.,* all leader agents are already in the region $\overline{\mathbb{X}}_l$ at the starting time t_0^k. Thus, it is not always possible to satisfy C1.

Corollary 1: Suppose Assumption 1 holds and $\exists i \in \mathscr{V}_L, x_i(t_0^k) \notin \overline{\mathbb{X}}_l$. The performance functions α_i, β_i and γ_i defined in (29a), (29b) and (29c) guarantee that the conditions C1 and C2 are satisfied simultaneously.

Let $\rho_i(t) := (\beta_i(t) + \alpha_i(t))/2$ and $\delta_i(t) := (\beta_i(t) - \alpha_i(t))/2$. Then, (16) can be rewritten as

$$-\delta_i(t) + \rho_i(t) < \|x_i(t) - c_l\| < \rho_i(t) + \delta_i(t). \tag{32}$$

Define in this case the normalized error $\xi_i(t)$ as

$$\xi_i(t) := \frac{\|x_i(t) - c_l\| - \rho_i(t)}{\delta_i(t)},$$

and ξ_{ij} is defined the same as in (22). Then, the corresponding set

$$\hat{D}_{\xi_i} := \{\xi_i(t) : \xi_i(t) \in (-1, 1)\} \tag{33}$$

is equivalent to (16). The normalized errors ξ_i is transformed through transformation functions $S2$. We denote the transformed errors $\zeta_i(\xi_i)$ and $\varepsilon_{ij}(\xi_{ij})$ by

$$\zeta_i(\xi_i) := S_2(\xi_i), \quad \varepsilon_{ij}(\xi_{jk}) := S_1(\xi_{ij}), \tag{34}$$

respectively. Here, ξ_i is transformed through $S2$ since the definition of domain for ξ_i is $(-1, 1)$.

Let $\nabla S_2(\xi_i) = \partial S_2(\xi_i)/\partial\xi_i, \nabla S_1(\xi_{ij}) = \partial S_1(\xi_{ij})/\partial\xi_{ij}$. The control laws for the leader and follower agents are proposed respectively as:

$$\begin{aligned} u_i^{\text{lead}} = &- \sum_{j\in\mathscr{N}_i} \frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} \\ &- \frac{1}{\delta_i}\nabla S_2(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i - K_i v_i, \end{aligned} \tag{35}$$

and

$$u_i^{\text{fol}} = - \sum_{j\in\mathscr{N}_i} \frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} - K_i v_i, \tag{36}$$

where K_i is a positive control gain to be determined.

Let $z = (y, v)^T$. Define the following function

$$V_2(z, \bar{\varepsilon}, \bar{\zeta}) = \begin{cases} \frac{1}{2}(z^T G_1 z + + \bar{\varepsilon}^T\bar{\varepsilon} + \bar{\zeta}^T H\bar{\zeta}), \, t \in [t_0^k, a_l + \underline{t}) \\ \frac{1}{2}(z^T G_2 z + + \bar{\varepsilon}^T\bar{\varepsilon} + \bar{\zeta}^T H\bar{\zeta}), \, t \ge a_l + \underline{t} \end{cases} \tag{37}$$

where

$$G_1 = \begin{pmatrix} K\kappa_2 & \kappa_2 \\ \kappa_2 & I_N \end{pmatrix} \otimes I_n, G_2 = \begin{pmatrix} K\kappa_3 & \kappa_3 \\ \kappa_3 & I_N \end{pmatrix} \otimes I_n, \tag{38}$$

and $\kappa_2, \kappa_3 \in \mathbb{R}^{N\times N}$ are diagonal matrices with entries $\kappa_{i,2}$ and $\kappa_{i,3}$, respectively. The matrices H, K are defined in (28).

Let $V_2^0 := V_2(z(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k))$. Let $V_2^* = V_2^0 + \sum_{i=1}^N \left(K_i|\kappa_{i,2} - \kappa_{i,3}|\beta_{i0}^2 + 2|\kappa_{i,2} - \kappa_{i,3}|\beta_{i0}(\sqrt{2V_2^0} + \kappa_{i,2}\beta_{i0})\right)$ and $\Theta_1 := S_1^{-1}(\sqrt{2V_2^*}), \Theta_2 := S_2^{-1}(\sqrt{2V_2^*})$. Then, the following holds.

Theorem 2: Consider the leader-follower MAS (2) and the kth execution task $\chi[k]$ given in (15). The control inputs for the leader and follower agents are given by (35) and (36), respectively. Suppose Assumption 1 holds, $\underline{t} > a_l + t_0^k$, and $\exists i \in \mathscr{V}_L, x_i(t_0^k) \notin \overline{\mathbb{X}}_l$. Assume that the control gain $K_i > \max\{\kappa_{i,2}, \kappa_{i,3}\}, \forall i$, the constant γ_{i0} in (30c) satisfies $\gamma_{i0} < d_s \alpha_{i0} e^{\kappa_{i,3}(\underline{t}-\bar{t})}/\bar{r}_l, \forall i$, and the input constraints for leader and follower agents satisfy

$$u_i^{\max} \geq \left(\frac{|\mathscr{N}_i|}{d_s(1-\Theta_1)} + \frac{4}{\sigma(1-\Theta_2^2)}\right)\sqrt{2V_2^*} + K_i(\sqrt{2V_2^*} + K_i\beta_{i0}), \quad \forall i \in \mathscr{V}_L$$

and

$$u_i^{\max} \geq \frac{|\mathscr{N}_i|}{d_s(1-\Theta_1)}\sqrt{2V_2^*} + K_i(\sqrt{2V_2^*} + K_i\beta_{i0}), \forall i \in \mathscr{V}_F,$$

respectively. Then, i) the tracking error $\|x_i - c_l\|, \forall i \in \mathscr{V}_L$ and the relative distance $\|x_{ij}\|, (i,j) \in \mathscr{E}$ will evolve respectively within the prescribed performance bounds (16) and (17) for all $t \geq t_0^k$ and ii) the input constraint for each agent i, *i.e.,* $u_i(t) \in U_i, \forall i$, will be satisfied for all $t \in [t_0^k, a_l + \bar{t}]$.

Proof: The proof is provided in appendix. □

Remark 11: In the optimization program (13), we choose the estimated completion time $E_l[\hat{k}_l]$ for each task ϕ_l and QoSa level $\hat{k}_l \geq 1$ as $E_l[\hat{k}_l] = a_l + \sup\{\mathrm{I}_l[\hat{k}_l]\}$ (see constraint (13e)). However, as stated in Remark 5, a task may be completed before its estimated completion time in real-time execution, allowing for other possibilities of execution. Due to this reason, the obtained plan from the optimization program (13) may not be optimal. We note that this is unavoidable since the (actual) completion time of each task can not be foreseen at the time of scheduling. Another reason that affects the optimality of the plan is the length of the wait task set W. It is pointed out in Remark 10 that the computational complexity of (13) grows exponentially with respect to the length of W, therefore, we propose to limit the size of wait task set W for computational efficiency.

V. Simulation

In this section, a numerical example illustrates the theoretical results. Consider a group of 4 agents with $n = 2$, the communication graph $\mathscr{G}$ is characterized by the adjacency matrix $A = [0,1,1,0;1,0,1,1;1,1,0,1;0,1,0,0]$. The initial position $x_i(0)$ of each agent i is chosen randomly from the box $[0,2] \times [0,2]$, and the initial velocity $v_i(0) = [0,0]^T, \forall i$.

The task set ϕ consists of 6 tasks. For the sake of simplicity, it is assumed that each target region $\mathbb{X}_l, l \in \{1,\ldots,6\}$ has the shape of a ball, denoted by $\mathscr{B}(c_l, r_l)$, where c_l, r_l are the center and radius of the ball, respectively. For each task, the corresponding parameters (QoSa level and the corresponding time interval, reward, target set, activation times) are summarized in TABLE I. In addition, we further consider the general case that the group of leader and follower agents can be different for different tasks. In this example, the leader set $\mathscr{V}_L$ is given by $\{1,4\}$ for tasks $1,2,5$, $\{1,3\}$ for task 3, $\{1,2,3,4\}$ for task 4, and $\{1,2,4\}$ for task 6. One can see that 5 tasks are activated initially. In total, 11 tasks are activated.

TABLE I
Task Specifications.

Task	Time interval / Reward				Target set	Activation time
QoSa	3	2	1	0		
ϕ_1	(0,6]	(6,10]	(10,20]	(20,∞)	$\mathscr{B}((10,8),1)$	{0,10}
	8	20	5	-20		
ϕ_2	-	-	(0,12]	(12,∞)	$\mathscr{B}((8,2),1)$	{0,18}
	-	-	10	-20		
ϕ_3	-	(0,8]	(8,14]	(14,∞)	$\mathscr{B}((5,5),0.5)$	{0}
	-	10	20	-20		
ϕ_4	(0,6]	(6,10]	(10,23]	(23,∞)	$\mathscr{B}((15,9),0.8)$	{0,30}
	25	5	20	-20		
ϕ_5	-	-	(0,10]	(10,∞)	$\mathscr{B}((5,10),1)$	{6,18}
	-	-	10	-20		
ϕ_6	-	-	(0,15]	(15,∞)	$\mathscr{B}((20,15),1)$	{0,25}
	-	-	5	-20		

In the optimization program (13), we chose $\varsigma = 1$ (the reason for this choice is to make a fair comparison with the EDF policy). Firstly, the dynamic scheduling strategy (DSS) proposed in this paper is considered, where the cardinality of the wait task set W is 4. The simulation results are given in Figs. 3-5. Fig. 3 shows the evolution of positions for each agent with respect to time t, where x_{i1} and x_{i2} are position components. The 11 red circles (from left to right) represent the 11 target regions that being reached, respectively. In Fig. 4, the evolution of the tracking error $\|x_i - c_l\|$ for each agent and the corresponding performance bounds $\alpha_i, \beta_i, i \in \mathscr{V}_L$ are depicted. In addition, the evolution of the relative distances $\|x_{ij}\|$ between neighboring agents and the corresponding performance bounds γ_i are plotted in Fig. 5. One can see that the performance bounds are satisfied at all times.

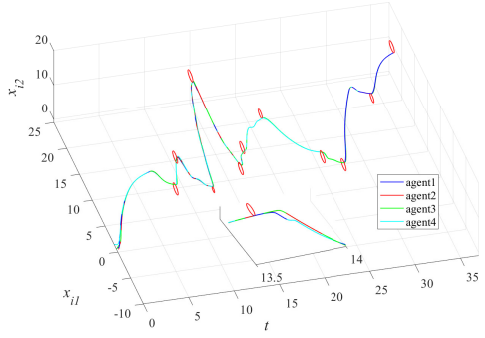

Fig. 3. The evolution of positions for each agent under DSS strategy.

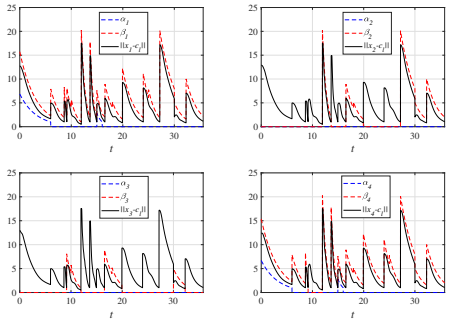

Fig. 4. The evolution of the tracking error and performance bounds α_i, β_i for each agent. Note that the performance bounds are presented only for the leader agents in each task.

Secondly, we consider the dynamic scheduling strategy, but for the case that there is no limit on $|W|$ (and which will be referred to as DSS* in the following). In this way, the true optimal schedule (i.e. considering all possible permutations of the wait task set) can be obtained. Finally, the EDF policy is used to schedule the same set of tasks. The simulation results for the three different policies are summarized in TABLE II. One can see that the best total reward is obtained when the DSS* policy is implemented. However, we note that the DSS* policy will result in a computational complexity problem when the size of the wait task set W is large. The DSS policy can be seen as a compromise between the reward and the computational efficiency. When the EDF policy is implemented, the total reward is 120, which is the worst among the three policies. This makes sense since the EDF policy involves only the deadline of each task.

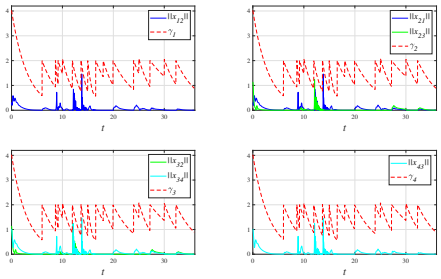

Fig. 5. The evolution of the relative distances between neighboring agents and performance bounds γ_i for each agent.

TABLE II
RESULTS FOR DSS, DSS* AND EDF POLICIES.

Strategy	Task completion order	Total reward
DSS	2-1-3-6-5-1-4-5-2-4-6	155
DSS*	4-1-2-3-6-5-1-2-5-6-4	160
EDF	2-3-6-5-1-4-5-1-2-6-4	120

VI. CONCLUSIONS

We proposed a dynamic task scheduling and distributed control synthesis strategy for the coordination of leader-follower MAS whose goal is to complete a set of dynamically activated tasks with deadline constraints. By utilizing ideas from PPC, we developed a distributed control law that guarantees the satisfaction of tasks under specific time interval constraints. A next step is to consider more general system dynamics and perform physical experiments. Moreover, practical issues, such as collision avoidance, will be considered in the future.

REFERENCES

[1] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems", *J. scheduling*, vol. 12, no. 4, pp. 417-431, 2009.

[2] S. A. Fayazi, A. Vahidi, and A. Luckow, "Optimal scheduling of autonomous vehicle arrivals at intelligent intersections via MILP", *in Proc. American Control Conference (ACC)* (pp. 4920-4925), 2017.

[3] H. Surmann and A. Morales, "Scheduling tasks to a team of autonomous mobile service robots in indoor enviroments", *J. Universal Comput. Sci.*, vol. 8, no.8, pp. 809-833, 2002.

[4] M. Ji, and M. B. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness", *IEEE Trans. Robot.*, vol. 23, no. 4, pp. 693-703, 2007.

[5] W. Ren, and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies", *IEEE Trans. Autom. Control*, vol. 50, no. 5, pp. 655-661, 2005.

[6] S. Mou, M. Cao, and A. S. Morse, "Target-point formation control", *Automatica*, vol. 61, pp. 113-118, 2015.

[7] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion", *IEEE Rob. Autom. Mag.*, vol. 14, no. 1, pp. 6171, 2007.

[8] G. E. Fainekos, A. Girard, H. Kress-Gazit and G. J. Pappas, "Temporal logic motion planning for dynamic mobile robots", *Automatica*, vol. 45, no. 2, pp. 343352, 2009.

[9] M. Guo and D. V. Dimarogonas, "Multi-agent plan reconfiguration under local LTL specifications", *Int. J. Rob. Res.*, vol. 34, no. 2, pp. 218-235, 2015.

[10] K. Ramamritham, and J. A. Stankovic, "Dynamic task scheduling in hard real-time distributed systems", *IEEE software*, vol. 1, no. 3, pp. 65-75, 1984.

[11] C. L. Liu, and James W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment", *Journal of the ACM*, vol. 20, no. 1, pp. 46-61, 1973.

[12] G. C. Buttazzo, and J. A. Stankovic, Red: Robust earliest deadline scheduling, University of Massachusetts, Department of Computer Science, Sep. 1993.

[13] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. C. Buttazzo, Deadline scheduling for real-time systems: EDF and related algorithms, Springer Science & Business Media, 2012.

[14] H. Aydin, R. Melhem, D. Mosse, and P. Mejía-Alvarez, "Optimal reward-based scheduling for periodic real-time tasks", *IEEE Trans. Comput.*, vol. 50, no. 2, pp. 111-130, 2001.

[15] J. K. Dey, J. Kurose, and D. Towsley, "On-line scheduling policies for a class of IRIS (increasing reward with increasing service) real-time tasks", *IEEE Trans. Comput.*, vol. 45, no. 7, pp. 802-813, 1996.

[16] M. Guinaldo, and D. V. Dimarogonas, "A hybrid systems framework for multi agent task planning and control", *in Proc. American Control Conference (ACC)*, 2017, pp. 1181-1186.

[17] G. Yang, R. Tron, and C. Belta, "Continuous-time Signal Temporal Logic Planning with Control Barrier Function", *arXiv preprint arXiv:1903.03860*, 2019.

[18] L. Lindemann and D. V. Dimarogonas, "Decentralized robust control of coupled multi-agent systems under local signal temporal logic tasks", *in Proc. American Control Conference (ACC)*, 2018, pp. 1567-1573.

[19] C. P. Bechlioulis and G. A. Rovithakis, "Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance", *IEEE Trans. Autom. Control*, vol. 53, no. 9, pp. 2090-2099, 2008.

[20] A. Ilchmann and H. Schuster, "PI-funnel control for two mass systems", *IEEE Trans. Autom. Control*, vol. 54, no. 4, pp. 918-923, 2009.

[21] C. M. Hackl, N. Hopfe, A. Ilchmann, M. Mueller, and S. Trenn, "Funnel control for systems with relative degree two", *SIAM Journal on Control and Optimization*, vol. 51, no. 2, pp. 965-995, 2013.

[22] C. P. Bechlioulis, M. A. Demetriou, and K. J. Kyriakopoulos, "A distributed control and parameter estimation protocol with prescribed performance for homogeneous lagrangian multi-agent systems", *Autonomous Robots*, pp. 1-17, 2018.

[23] J. G. Lee, S. Trenn, and H. Shim, "Synchronization with prescribed transient behavior: Heterogeneous multi-agent systems under funnel coupling", arXiv preprint arXiv:2012.14580, 2020.

[24] P. Yu, and D. V. Dimarogonas, "Time-constrained multi-agent task scheduling based on prescribed performance control", *in Proc. Decision and Control (CDC)*, 2018, pp. 2593-2598.

[25] Z. Allen-Zhu, Z. Liao, L. Orecchia, and M. I. T. Math, "Using optimization to find maximum inscribed balls and minimum enclosing balls", *arXiv preprint arXiv:1412.1001*, 2014.

[26] C. Lu, J. A. Stankovic, S. H. Son, and G. Tao, "Feedback control real-time scheduling: Framework, modeling, and algorithms", *Real-Time Systems*, vol. 23, no. 1-2, pp. 85-126, 2002.

[27] M. Guo, J. Tumova, and D. V. Dimarogonas, "Communication-free multi-agent control under local temporal tasks and relative-distance constraints", *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 3948-3962, 2016.

[28] R. G. Sanfelice, "Robust asymptotic stabilization of hybrid systems using control lyapunov functions", *in Proc. International Conference on Hybrid Systems: Computation and Control*, Apr. 2016, pp. 235-244.

[29] R. Goebel, R. G. Sanfelice, and A. R. Teel, "Hybrid dynamical systems", *IEEE Control Systems Magazine*, vol. 29, no.2, pp. 28-93, 2009.

[30] R. Goebel, R. G. Sanfelice, and A. R. Teel, Hybrid dynamical systems: modeling stability, and robustness. Princeton University Press, 2012.

APPENDIX

Proof of Theorem 1: Since $K_i > \max\{\mu_{i,1}, \kappa_{i,1}\} = \theta_i, \forall i$, one can derive that $V_1(y,v,\bar{\varepsilon},\bar{\zeta})$ is nonnegative for all $y,v,\bar{\varepsilon},\bar{\zeta}$.

Differentiating (28) along the trajectories of (2), one has

$$\begin{aligned}\dot{V}_1(y,v,\bar{\varepsilon},\bar{\zeta}) =& y^T K\theta v + y^T\theta u^{\chi[k]} + v^T\theta v + v^T u^{\chi[k]} \\ &+ \bar{\varepsilon}^T\dot{\bar{\varepsilon}} + \bar{\zeta}^T H\dot{\bar{\zeta}},\end{aligned} \tag{39}$$

where $u_i^{\chi[k]} = u_i^{\text{lead}}, i \in \mathscr{V}_L$ and $u_i^{\chi[k]} = u_i^{\text{fol}}, i \in \mathscr{V}_F$.

Substituting (26) and (27) into (39), we obtain

$$\begin{aligned}&\dot{V}_1(y,v,\bar{\varepsilon},\bar{\zeta}) \\ =&-\sum_{i=1}^N \theta_i y_i^T\Big\{\sum_{j\in\mathscr{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} - \frac{h_i}{\beta_i}\nabla S_1(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i\Big\} \\ &-\sum_{i=1}^N v_i^T\Big\{\sum_{j\in\mathscr{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} - \frac{h_i}{\beta_i}\nabla S_1(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i\Big\} \\ &-\sum_{i=1}^N (K_i-\theta_i)v_i^T v_i + \sum_{i=1}^p \varepsilon_{ij}(\xi_{ij})\nabla S_1(\xi_{ij})\dot{\xi}_{ij} \\ &+\sum_{i=1}^N h_i\zeta_i(\xi_i)\nabla S_1(\xi_i)\dot{\xi}_i.\end{aligned} \tag{40}$$

According to (22), one can get

$$\begin{aligned}\dot{\xi}_{ij} =& \frac{1}{\gamma_i}\frac{\|x_{ij}\|'\gamma_i - \|x_{ij}\|\dot{\gamma}_i}{\gamma_i} = \frac{1}{\gamma_i}\Big(\|x_{ij}\|' + \mu_{i,1}\|x_{ij}\|\Big), \\ \dot{\xi}_i =& \frac{1}{\beta_i}\frac{\|x_i - c_l\|'\beta_i - \|x_i - c_l\|\dot{\beta}_i}{\beta_i} = \frac{1}{\beta_i}\Big(\|y_i\|' + \kappa_{i,1}\|y_i\|\Big),\end{aligned}$$

where $-\dot{\beta}_i/\beta_i \equiv \kappa_{i,1}$ and $-\dot{\gamma}_i/\gamma_i \equiv \mu_{i,1}$. Due to symmetry, one has $\partial\left\|x_{ij}\right\|/\partial x_{ij} = \partial\left\|x_{ij}\right\|/\partial x_i = -\partial\left\|x_{ij}\right\|/\partial x_j$. From (2), one has that $\sum_{(i,j)\in\mathscr{E}}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\left\|x_{ij}\right\|' = 1/2\sum_{i=1}^N\sum_{j\in\mathscr{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\partial\left\|x_{ij}\right\|/\partial x_{ij}\dot{x}_{ij} = \sum_{i=1}^N v_i^T\sum_{j\in\mathscr{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij}$. In addition, $\sum_{i=1}^N y_i^T\sum_{j\in\mathscr{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} = \sum_{i=1}^N y_i^T\sum_{j\in\mathscr{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})y_i - y_j/\|y_i - y_j\| = 1/2\sum_{i=1}^N\sum_{j\in\mathscr{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\left\|x_{ij}\right\|$.

Then, (40) can be rewritten as

$$\begin{aligned}\dot{V}_1(y,v,\bar{\varepsilon},\bar{\zeta}) =& -\frac{1}{2}\sum_{i=1}^N\sum_{j\in\mathscr{N}_i}\frac{\theta_i-\mu_{i,1}}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \\ &-\sum_{i=1}^N\frac{h_i(\theta_i-\kappa_{i,1})}{\beta_i}\zeta_i(\xi_i)\nabla S_1(\xi_i)\|y_i\| - (K-\theta)v^T v.\end{aligned}$$

According to the definition of S_1, one can derive that $\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \geq 0$ and $\zeta_i(\xi_i)\nabla S_1(\xi_i)\|y_i\| \geq 0$. In addition, $\theta_i - \mu_{i,1} \geq 0, \theta_i - \kappa_{i,1} \geq 0, \forall i$. Therefore, one derives that $\dot{V}_1(y,v,\bar{\varepsilon},\bar{\zeta}) \leq 0$, which in turn implies $V_1(y,v,\bar{\varepsilon},\bar{\zeta}) \leq V_1(y(t_0^k), v(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k)) = V_1^0$ and thus $|S_1(\xi_i)| = |\zeta_i(\xi_i)| \leq |\bar{\zeta}| \leq \sqrt{2V_1^0}, \forall i \in \mathscr{V}_L, |S_1(\xi_{ij})| = |\varepsilon_{ij}(\xi_{ij})| \leq |\bar{\varepsilon}| \leq \sqrt{2V_1^0}, \forall (i,j) \in \mathscr{E}$ for all $t \geq t_0^k$. Moreover, $\xi_i(t_0^k), \forall i \in \mathscr{V}_L$ and $\xi_{ij}(t_0^k), \forall (i,j) \in \mathscr{E}$ are within the regions (23) and (24), respectively. By using the inverse of S_1, we can bound $0 \leq \xi_i(t) \leq \Theta < 1$ and $0 \leq \xi_{ij}(t) \leq \Theta < 1$ for all $t > t_0^k$, where $\Theta = S_1^{-1}\left(\sqrt{2V_1^0}\right)$. That is to say, $\xi_i(t), \forall i \in \mathscr{V}_L$ and $\xi_{ij}(t), \forall (i,j) \in \mathscr{E}$ will evolve within the regions (23) and (24) for all $t \geq t_0^k$. Since (23) and (24) are equivalent to the prescribed performance bounds (16) and (17), respectively, item i) of Theorem 1 holds.

Since $\xi_i(t), \xi_{ij}(t) \in [0,\Theta)$ and the functions β_i, γ_i are monotonically decreasing, one has from (26) that

$$\begin{aligned}\|u_i^{\text{lead}}\| \leq& \left|\sum_{j\in\mathscr{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\right| + \left|\frac{1}{\beta_i}\nabla S_1(\xi_i)\zeta_i(\xi_i)\right| + \|K_i v_i\| \\ \leq& \left(\frac{|\mathscr{N}_i|}{\gamma_i(t_0^k+\bar{t})(1-\Theta)} + \frac{1}{\beta_i(t_0^k+\bar{t})(1-\Theta)}\right)\sqrt{2V_1^0} + \|K_i v_i\|\end{aligned} \tag{41}$$

for $t \in [t_0^k, a_l + \bar{t}]$. In addition, since $K_i > \theta_i, \forall i$, one has

$$\begin{aligned}2V_1^0 \geq& \begin{bmatrix} y & v\end{bmatrix}\left\{\begin{bmatrix} K\theta & \theta \\ \theta & I_N\end{bmatrix}\otimes I_n\right\}\begin{bmatrix} y \\ v\end{bmatrix} \\ \geq& \sum_{i=1}^N(\|v_i\| - \theta_i\|y_i\|)^2 \\ \geq& \sum_{i=1}^N(\|v_i\| - \theta_i\beta_{i0})^2.\end{aligned}$$

Then, one has

$$\|v_i\| \leq \sqrt{2V_1^0} + \theta_i\beta_{i0}. \tag{42}$$

Combining (41), (42) and $\gamma_i(a_l+\bar{t}) = d_s, \beta_i(a_l+\bar{t}) = \sigma$, one can further get

$$\begin{aligned}\|u_i^{\text{lead}}\| \leq& \left(\frac{|\mathscr{N}_i|}{d_s(1-\Theta)} + \frac{1}{\sigma(1-\Theta)}\right)\sqrt{2V_1^0} \\ &+ K_i(\sqrt{2V_1^0} + \theta_i\beta_{i0}) \leq u_i^{\max}, \forall i \in \mathscr{V}_L, \forall t \in [t_0^k, a_l+\bar{t}].\end{aligned}$$

Similarly, one can also derive that $\|u_i^{\text{fol}}\| \leq u_i^{\max}, \forall i \in \mathscr{V}_F, \forall t \in [t_0^k, a_l+\bar{t}]$. Item ii) of Theorem 1 holds. □

Proof of Theorem 2: Since $K_i > \max\{\underline{\kappa}_{i,2}, \kappa_{i,3}\}, \forall i$, one can derive $G_1 \succ 0, G_2 \succ 0$. Therefore, $V_2(z,\bar{\varepsilon},\bar{\zeta})$ is nonnegative for all $z,\bar{\varepsilon},\bar{\zeta}$.

For $t \in [t_0^k, a_l + \underline{t}]$, differentiating (37) along the trajectories of (2) and substituting (35) and (36), one has

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) =& -\sum_{i=1}^N\kappa_{i,2}y_i^T\sum_{j\in\mathscr{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} \\ &-\sum_{i=1}^N\kappa_{i,2}y_i^T\frac{h_i}{\delta_i}\nabla S_2(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i \\ &-\sum_{i=1}^N v_i^T\sum_{j\in\mathscr{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} \\ &-\sum_{i=1}^N v_i^T\frac{h_i}{\delta_i}\nabla S_2(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i \\ &-\sum_{i=1}^N(K_i-\kappa_{i,2})v_i^T v_i + \sum_{i=1}^p\varepsilon_{ij}(\xi_{ij})\nabla S_1(\xi_{ij})\dot{\xi}_{ij} \\ &+\sum_{i=1}^n h_i\zeta_i(\xi_i)\nabla S_2(\xi_i)\dot{\xi}_i,\end{aligned} \tag{43}$$

where

$$\begin{aligned}\dot{\xi}_{ij} =& \frac{1}{\gamma_i}\frac{\|x_{ij}\|'\gamma_i - \|x_{ij}\|\dot{\gamma}_i}{\gamma_i},\\ \dot{\xi}_i =& \frac{1}{\delta_i}\frac{(\|x_i-c_l\|'-\dot{\rho}_i)\delta_i-(\|x_i-c_l\|-\rho_i)\dot{\delta}_i}{\delta_i}\\ =& \frac{1}{\delta_i}\frac{(\|y_i\|'-\dot{\rho}_i)\delta_i-(\|y_i\|-\rho_i)\dot{\delta}_i}{\delta_i}.\end{aligned}$$

Similar to the proof of Theorem 1, one can further get

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) =& -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathscr{N}_i}\frac{\kappa_{i,2}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|\\ &-\sum_{i=1}^{N}\frac{h_i(\kappa_{i,2}-\hat{\delta}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)(\|y_i\|-\rho_i)\\ &-\sum_{i=1}^{N}\frac{h_i\rho_i(\kappa_{i,2}-\hat{\rho}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)\\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,2})v_i^T v_i,\end{aligned} \tag{44}$$

where $\hat{\gamma}_i = -\dot{\gamma}_i/\gamma_i$, $\hat{\delta}_i = -\dot{\delta}_i/\delta_i$ and $\hat{\rho}_i = -\dot{\rho}_i/\rho_i$. In addition, according to the definition of $\gamma_i(t), \delta_i(t)$ and $\rho_i(t)$, one can derive that $\hat{\gamma}_i(t) = \hat{\delta}_i(t) = \hat{\rho}_i(t) \equiv \kappa_{i,2}$ for all $t \in [t_0^k, \underline{t}]$. Therefore,

$$\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) \leq -\sum_{i=1}^{N}(K_i-\kappa_{i,2})v_i^T v_i \leq 0, \quad \forall [t_0^k,\underline{t}]. \tag{45}$$

For $t > a_l + \underline{t}$, differentiating (37) along the trajectories of (2) and substituting (35) and (36), one has

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) =& -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathscr{N}_i}\frac{\kappa_{i,3}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|\\ &-\sum_{i=1}^{N}\frac{h_i(\kappa_{i,3}-\hat{\delta}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)(\|y_i\|-\rho_i)\\ &-\sum_{i=1}^{N}\frac{h_i\rho_i(\kappa_{i,3}-\hat{\rho}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)\\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,3})v_i^T v_i,\end{aligned} \tag{46}$$

where $\hat{\delta}_i(t) = \hat{\rho}_i(t) \equiv \kappa_{i,3}, \forall t > a_l + \underline{t}$. Then, one can further have

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) =& -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathscr{N}_i}\frac{\kappa_{i,3}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|\\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,3})v_i^T v_i.\end{aligned} \tag{47}$$

If one chooses $\gamma_{i0} < d_s\alpha_{i0}e^{\kappa_{i,3}(\underline{t}-\bar{t})}/\bar{r}_l, \forall i$, one can verify that $\hat{\gamma}_i < \kappa_{i,3}, \forall i$. In addition, one has $\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \geq 0, \forall (i,j) \in \mathscr{E}$. Therefore,

$$\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) \leq 0, \quad \forall t > a_l + \underline{t}. \tag{48}$$

Combining (45) and (48), one can get that $V_2(z,\bar{\varepsilon},\bar{\zeta}) \leq \max\{V_2(z(t_0^k),\bar{\varepsilon}(t_0^k),\bar{\zeta}(t_0^k)), V_2(z(a_l+\underline{t}),\bar{\varepsilon}(a_l+\underline{t}),\bar{\zeta}(a_l+\underline{t}))\} \leq V_2^*$ and thus

$$|\zeta_i(\xi_i)| \leq |\bar{\zeta}| \leq \sqrt{2V_2^*}, \forall i \in \mathscr{V}_L$$

and

$$|\varepsilon_{ij}(\xi_{ij})| \leq |\bar{\varepsilon}| \leq \sqrt{2V_2^*}, \forall (i,j) \in \mathscr{E},$$

for all $t \geq t_0^k$.

The remainder of the proof is similar to that of Theorem 1 and hence omitted. □

Pian Yu (S'19) received the M.Sc. in Control Science and Control Engineering in 2016 from Wuhan University, China and the Ph.D. in Electrical Engineering in 2021 from KTH Royal Institute of Technology, Sweden. From April 2021, she is a Postdoctoral Researcher at the Division of Decision and Control Systems, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology. Her main research interest includes multi-agent systems, motion planning and coordination, event-triggered control, and formal methods.

Dimos V. Dimarogonas (M'10-SM'17) was born in Athens, Greece, in 1978. He received the Diploma in Electrical and Computer Engineering in 2001 and the Ph.D. in Mechanical Engineering in 2007, both from the National Technical University of Athens (NTUA), Greece. Between May 2007 and March 2010, he held postdoctoral positions at KTH Royal Institute of Technology, Stockholm, Sweden and at LIDS, MIT, Boston, USA. He is currently a Professor at the Division of Decision and Control Systems, School of EECS, at the KTH Royal Institute of Technology. His current research interests include Multi-Agent Systems, Hybrid Systems and Control, Robot Navigation and Networked Control. He serves in the Editorial Board of Automatica and the IEEE Transactions on Control of Network Systems and is a Senior Member of the IEEE. He received an ERC Starting Grant in 2014, an ERC Consolidator Grant in 2019, and was awarded a Wallenberg Academy Fellow grant in 2015.