Cooperative Task Planning of Multi-Agent Systems Under Timed Temporal Specifications

Alexandros Nikou, Jana Tumova and Dimos V. Dimarogonas

***Abstract*— In this paper the problem of cooperative task planning of multi-agent systems when timed constraints are imposed to the system is investigated. We consider timed constraints given by Metric Interval Temporal Logic (MITL). We propose a method for automatic control synthesis in a two-stage systematic procedure. With this method we guarantee that all the agents satisfy their own individual task specifications as well as that the team satisfies a team global task specification.**

I. Introduction

Cooperative control of multi-agent systems has traditionally focused on designing local control laws in order to achieve tasks such as consensus, formation, network connectivity, and collision avoidance ([1]–[7]). Over the last decade or so, the field of control of multi-agent systems with complicated behavior under complex high-level task specifications has been gaining significant research attention. Such high-level tasks may have the form of "Periodically survey regions A, B, C while avoiding region D", or "Visit regions A, B, C, in this order", and many others. Multiple robotic vehicles then may perform these types of tasks faster and more efficiently than a single robot. In this work, we aim to introduce specific time bounds into the complex tasks, such as "Periodically survey regions A, B, C, avoid region D and always keep the longest time between two consecutive visits to A below 5 time units", or "Visit regions A, B, C, in this order within 10 time units".

The team of agents is usually associated with a set of tasks that should be fulfilled by the group of agents as a whole at the discrete level. A three-step hierarchical procedure to address such a problem is described as follows ([8], [9]): First the robot dynamics is abstracted into a finite or countable, discrete transition system using sampling or cell decomposition methods based on triangulations, rectangular or other partitions. Second, invoking ideas from verification methods, a discrete plan that meets the high-level task is synthesized. Third, the discrete plan is translated into a sequence of continuous controllers for the original system.

The specification language that has extensively been used to express the tasks is the Linear Temporal Logic (LTL) (see, e.g., [10]). LTL has proven a valuable tool for controller synthesis, because it provides a compact mathematical formalism for specifying desired behaviors of a system. There is a rich body of literature containing algorithms for verification and synthesis of system obeying temporal logic specifications ([11], [12]). A common approach in multi-agent planning under LTL specifications is the consideration of a centralized, global task specification for the team of agents which is then decomposed into local tasks to be accomplished by the individual agents. For instance, the authors in [13] utilized the parallel composition (synchronous products) of multi-robot systems in order to decompose a global specification that is given to a team of robots into individual specifications. This method has been proven computationally expensive due to state space explosion problem and in order to relax the computational burden the authors in [14], [15] proposed a method that does not require the computation of the parallel composition. This method, however, is restrictive to specifications that can be expressed in certain subclasses of LTL. In [16], the specification formula was given in LTL in parallel with the problem of minimum inter-robot communication.

Explicit time constraints in the system modeling have been included e.g., in [17], where a method of automated planning of optimal paths of a group of agents satisfying a common high-level mission specification was proposed. The mission was given in LTL and the goal was the minimization of a cost function that captures the maximum time between successive satisfactions of the formula. Authors in [18], [19] used a different approach, representing the motion of each agent in the environment with a timed automaton. The composition of the team automaton was achieved through synchronization and the UPPAAL verification tool ([20]) was utilized for specifications given in Computational Tree Logic (CTL). In the same direction, authors in [21] modeled the multi-robot framework with timed automata and weighted transition systems considering LTL specifications and then, an optimal motion of the robots satisfying instances of the optimizing proposition was proposed.

Most of the previous works on multi-agent planning consider temporal properties which essentially treat time in a qualitative manner. For real applications, a multi-agent team might be required to perform a specific task within a certain time bound, rather than at some arbitrary time in the future (quantitative manner). Timed specifications have been considered in [22]–[25]. In [22], the authors addressed the problem of designing high-level planners to achieve tasks for switching dynamical systems under Metric Temporal Logic (MTL) specification and in [23], the authors utilized a counterexample-guided synthesis to cyber-physical systems subject to Signal Temporal Logic (STL) specifications. In [24], the MTL formula for a single agent was translated into linear constraints and a Mixed Integer Linear Programming (MILP) problem was solved. However, these works are restricted to single agent motion planning and are not expendable to multi-agent systems in a straightforward way. In [25], the vehicle routing problem was considered under the presence of MTL specifications. The approach does not rely

The authors are with the ACCESS Linnaeus Center, School of Electrical Engineering, KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden and with the KTH Centre for Autonomous Systems. Email: {anikou,tumova,dimos}@kth.se. This work was supported by the H2020 ERC Starting Grant BUCOPHSYS and the Swedish Research Council (VR).

on automata-based approach to verification, as it constructs a set of linear inequalities from MTL specification formula in order to solve an MILP problem.

In this work, we aim at designing automated planning procedure for a team of agents that are given an individual, independent timed temporal specification each and a single global team specification. This constitutes the first step towards including time constraints to temporal logic-based multi-agent control synthesis. We consider a quantitative logic called Metric Interval Temporal Logic (MITL) ([26]) in order to specify explicit time constraints. The proposed solution is fully automated and completely desynchronized in the sense that a faster agent is not required to stay in a region and wait for the slower one. It is decentralized in handling the individual specifications and centralized only in handling the global team specification. To the best of the authors' knowledge this is the first work that address the cooperative task planning for multi-agent systems under individual and global timed linear temporal logic specifications.

The remainder of the paper is structured as follows. In Sec. II a description of the necessary mathematical tools, the notations and the definitions are given. Sec. III provides the model of the multi-agent system, the task specification, several motivation examples as well as the formal problem statement. Sec. IV discusses the technical details of the solution. Sec. V is devoted to an illustrative example. Finally, the conclusions and the future work directions are discussed in Sec. VI.

II. Notation and Preliminaries

Given a set S, we denote by $|S|$ its cardinality and by 2^S the set of all its subsets. An infinite sequence of elements of S is called a infinite word over the set S and it is denoted by $w = w(0)w(1)\dots$ The i-th element of a sequence is denoted with $w(i)$. We denote by $\mathbb{Q}_+, \mathbb{N}$ the set of positive rational and natural numbers including 0, respectively. Let us also define $\mathbb{T}_\infty = \mathbb{T} \cup \{\infty\}$ for a set of numbers $\mathbb{T}$.

Definition 1. ([27]) A *time sequence* $\tau = \tau(0)\tau(1)\cdots$ is a infinite sequence of time values $\tau(j) \in \mathbb{T} = \mathbb{Q}_+$, satisfying the following constraints:

- Monotonicity: $\tau(j) < \tau(j+1)$ for all $j \geq 0$.
- Progress: For every $t \in \mathbb{T}$, $\exists\ j \geq 1$, such that $\tau(j) > t$.

An *atomic proposition* p is a statement over the problem variables and parameters that is either True $(\top)$ or False $(\bot)$ at a given time instance.

Definition 2. ([27]) Let AP be a finite set of atomic propositions. A *timed word* w over the set AP is an infinite sequence $w = (w(0), \tau(0))(w(1), \tau(1))\cdots$ where $w(0)w(1)\dots$ is an infinite word over the set 2^{AP} and $\tau(0)\tau(1)\dots$ is a time sequence with $\tau(j) \in \mathbb{T},\ j \geq 0$. A *timed language* $Lang_\mathcal{T}$ over AP is a set of timed words over AP.

A. Weighted Transition System

Definition 3. A *Weighted Transition System* (WTS) is a tuple $(S, S_0, \longrightarrow, d, AP, L)$ where S is a finite set of states; $S_0 \subseteq S$ is a set of initial states; $\longrightarrow \subseteq S \times S$ is a transition relation; $d : \longrightarrow \longrightarrow \mathbb{T}$ is a map that assigns a positive weight to each transition; AP is a finite set of atomic propositions; and $L : S \longrightarrow 2^{AP}$ is a labeling function.

For simplicity, we use $s \to s'$ to denote the fact that $(s, s') \in \to$.

Definition 4. A *timed run* of a WTS is an infinite sequence $r^t = (r(0), \tau(0))(r(1), \tau(1))\dots$, such that $r(0) \in S_0$, and for all $j \geq 1$, $r(j) \in S$ and $r(j) \to r(j+1)$. The *time stamps* $\tau_k(j), j \geq 0$ are inductively defined as

1) $\tau(0) = 0$.
2) $\tau(j+1) = \tau(j) + d(r(j), r(j+1)),\ \forall\ j \geq 1$.

Every timed run r^t generates a *timed word* $w(r^t) = (L(r(0)), \tau(0))\ (L(r(1)), \tau(1))\dots$ over the set 2^{AP} where $w(j) = L(r(j)), \forall\ j \geq 0$ is the subset of atomic propositions that are true at state $r(j)$ at time $\tau(j)$.

B. Metric Interval Temporal Logic and Timed Automata

The syntax of *Metric Interval Temporal Logic (MITL)* over a set of atomic propositions AP is defined by the grammar

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc_I\ \varphi \mid \Diamond_I \varphi \mid \Box_I \varphi \mid \varphi_1\ \mathcal{U}_I\ \varphi_2 \quad (1)$$

where $p \in AP$, and $\bigcirc$, $\Diamond$, $\Box$ and $\mathcal{U}$ is the next, future, always and until temporal operator, respectively. $I \subseteq \mathbb{T}$ is a non-empty time interval in one of the following forms: $[i_1, i_2], [i_1, i_2), (i_1, i_2], (i_1, i_2), [i_1, \infty], (i_1, \infty)$ where $i_1, i_2 \in \mathbb{T}$ with $i_1 < i_2$. MITL can be interpreted either in continuous or point-wise semantics. We utilize the latter one and interpret MITL formulas over timed runs such as the ones produced by a WTS (Def. 4).

Definition 5. ([28], [29]) Given a run $r^t = (r(0), \tau(0))(r(1), \tau(1))\dots$ of a WTS and an MITL formula φ, we define $(r^t, i) \models \varphi$, for $i \geq 0$ (read r^t satisfies φ at position i) as follows

$$\begin{aligned}
&(r^t, i) \models p \Leftrightarrow p \in L(r(i)) \\
&(r^t, i) \models \neg\varphi \Leftrightarrow (r^t, i) \not\models \varphi \\
&(r^t, i) \models \varphi_1 \wedge \varphi_2 \Leftrightarrow (r^t, i) \models \varphi_1 \text{ and } (r^t, i) \models \varphi_2 \\
&(r^t, i) \models \bigcirc_I\ \varphi \Leftrightarrow (r^t, i+1) \models \varphi \text{ and } \tau(i+1) - \tau(i) \in I \\
&(r^t, i) \models \Diamond_I \varphi \Leftrightarrow \exists j, i \leq j, \text{ s.t. } (r^t, j) \models \varphi, \tau(j) - \tau(i) \in I \\
&(r^t, i) \models \Box_I \varphi \Leftrightarrow \forall j, i \leq j,\ \tau(j) - \tau(i) \in I \Rightarrow (r^t, j) \models \varphi \\
&(r^t, i) \models \varphi_1\ \mathcal{U}_I\ \varphi_2 \Leftrightarrow \exists j, i \leq j, \text{ s.t. } (r^t, j) \models \varphi_2, \\
&\qquad \tau(j) - \tau(i) \in I \text{ and } (r^t, k) \models \varphi_1 \text{ for every } i \leq k < j
\end{aligned}$$

Timed Büchi Automata (TBA) were introduced in [27] and in this work, we also partially adopt the notation from [30], [31]. Let $X = \{x_1, x_2, \dots, x_M\}$ be a finite set of *clocks*. The set of *clock constraints* $\Phi(X)$ is defined by the grammar

$$\phi := \top \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid x \bowtie c \quad (2)$$

where $x \in X$ is a clock, $c \in \mathbb{T}$ is a clock constant and $\bowtie \in \{<, >, \geq, \leq, =\}$. A clock *valuation* is a function $\nu : X \to \mathbb{T}$ that assigns a real value to each clock. A clock x_i has valuation ν_i for $i \in \{1, \dots, M\}$, and $\nu = (\nu_1, \dots, \nu_M)$. We denote by $\nu \models \phi$ the fact that the valuation ν satisfies the clock constraint ϕ.

Definition 6. A *TBA* is a tuple $\mathcal{A} = (S, S^{\text{init}}, X, I, E, F, AP, \mathcal{L})$ where S is a finite set of locations; $S^{\text{init}} \subseteq S$ is the set of initial locations; X is a finite set of clocks; $I : S \to \Phi(X)$ is the invariant; $E \subseteq S \times \Phi(X) \times 2^X \times S$ gives the set of transitions; $F \subseteq S$ is a set of accepting locations; AP is a finite set of atomic propositions; and $\mathcal{L} : S \to 2^{AP}$ labels every state with a subset atomic propositions.

A state of $\mathcal{A}$ is a pair (s, ν) where $s \in S$ and ν satisfies the *invariant* $I(s)$, i.e., $\nu \models I(s)$. The initial state of $\mathcal{A}$ is $(s(0), (0, \dots, 0))$, where $s(0) \in S_0$. Given two states (s, ν) and (s', ν') and an edge $e = (s, \gamma, R, s')$, there exists a *discrete transition* $(s, \nu) \xrightarrow{e} (s', \nu')$ iff ν satisfies the *guard* of the transition γ, i.e., $\nu \models \gamma$, $\nu' \models I(s')$, and R is the *reset set*, i.e., $\nu'_i = 0$ for $x_i \in R$ and $\nu'_i = \nu_i$ for $x_i \notin R$. Given a $\delta \in \mathbb{T}$, there exists a *time transition* $(s, \nu) \xrightarrow{\delta} (s', \nu')$ iff $s = s', \nu' = \nu + \delta$ and $\nu' \models I(s)$. An infinite run of $\mathcal{A}$ starting at state $(s(0), \nu)$ is an infinite sequence of time and discrete transitions $(s(0), \nu(0)) \xrightarrow{\delta_0} (s(0)', \nu(0)') \xrightarrow{e_0} (s(1), \nu(1)) \xrightarrow{\delta_1} (s(1)', \nu(1)') \dots$, where $(s(0), \nu(0))$ is an initial state. This run produces the timed word $w = (\mathcal{L}(s(0)), \tau(0))(\mathcal{L}(s(1)), \tau(1)) \dots$ with $\tau(0) = 0$ and $\tau(i+1) = \tau(i) + \delta_i, \forall\, i \geq 1$. The run is called *accepting* if $s(i) \in F$ for infinitely many times. A timed word is *accepted* if there exists an accepting run that produces it. The problem of deciding the emptiness of the language of a given TBA $\mathcal{A}$ is *PSPACE*-complete [27]. In other words, we can synthesize an accepting run of a given a TBA $\mathcal{A}$, if one exists.

Remark 1. Traditionally, the clock constraints and the TBAs are defined with $\mathbb{T} = \mathbb{N}$, however, they can be extended to accommodate $\mathbb{T} = \mathbb{Q}_+ \cup \{0\}$. By multiplying all the rational numbers that are appearing in the state invariants and the edge constraints with their least common multiple, we have equivalently only natural numbers occurring to the TBA. For the sake of physical understanding of the timed properties of the under investigation framework, we will be working with $\mathbb{T} = \mathbb{Q}_+ \cup \{0\}$.

Any MITL formula φ over AP can be algorithmically translated to a TBA with the alphabet 2^{AP}, such that the language of timed words that satisfy φ is the language of timed words produced by the TBA [26], [32], [33].

III. Problem Formulation

A. System Model

Consider a multi-agent team composed by N agents operating in a bounded workspace $\mathcal{W}_0 \subseteq \mathbb{R}^n$. Let $\mathcal{I} = \{1, \dots, N\}$ denote the index set of the agents. We assume that the workspace $\mathcal{W}_0$ is partitioned into a finite number (assume W) of regions of interest $\pi_1, \dots, \pi_W$ where

$$\mathcal{W}_0 = \bigcup_{i \in \mathcal{W}} \pi_i \ \text{ and } \ \pi_i \cap \pi_j \neq \emptyset, \forall\, i \neq j \ \text{ with } \ i, j \in \mathcal{W} \quad (3)$$

for the index set $\mathcal{W} = \{1, \dots, W\}$. We denote by π_i^k the agent k being at region π_i, where $k \in \mathcal{I}, i \in \mathcal{W}$. In this work, we focus on interaction and high-level control strategies rather than on nonlinear models, and we assume that the dynamics of each agent is given by a single integrator

$$\dot{x}_i = u_i, \ i \in \mathcal{I}. \quad (4)$$

The partitioned environment (3) is a discretization that allows us to control the agents with dynamics (4) using finite models such as finite transition systems (e.g., [9], [34]–[36]). We define a weighted transition system (see Def. 7) so that

- if there exists a controller $u_i, \ i \in \mathcal{I}$ such that the agent k can be driven from any point within the region π^i to a neighboring region π^j, then we allow for a transition $\pi_k^i \to_k \pi_k^j$ between the respective system states, and
- the weight of each transition estimates the time each agent needs in order to move from one region to another. In particular, the travel time is here determined as the worst-case shortest time needed to travel from an arbitrary point of the current region to the boundary of the following region. This estimate is indeed conservative, however, it is sufficient for specifications that we are generally interested in within multi-agent control. Namely, it is suitable for scenarios where tasks are given deadlines and upper rather than lower bound requirements are associated with events along the agents' runs.

Definition 7. The motion of each agent $k \in \mathcal{I}$ in the workspace is modeled by a WTS $\mathcal{T}_k = (\Pi_k, \Pi_k^{\text{init}}, \to_k, d_k, AP_k, L_k)$ where

- $\Pi_k = \{\pi_1^k, \pi_2^k, \dots, \pi_W^k\}$ is the set of states of agent k. Any state of an agent k can be denoted as $\pi_j^k \in \Pi_k$ for $k \in \mathcal{I}, j \in \mathcal{W}$. The number of states for each agent is $|\Pi_k| = W$.
- $\Pi_k^{\text{init}} \subseteq \Pi_k$ is the initial states of agent k, i.e. the set of regions where agent k may start.
- $\to_k \subseteq \Pi_k \times \Pi_k$ is the transition relation. For example, by $\pi_3^3 \to_3 \pi_5^3$ we mean that the agent 3 can move from region π_3 to region π_5.
- $d_k : \to_k \to \mathbb{T}$ is a map that assigns a positive weight (duration) to each transition. For example, $d_2(\pi_2^2, \pi_5^2) = 0.7$, where $\pi_2^2 \to_2 \pi_5^2$, means that agent 2 needs at most 0.7 time units to move from any point of region π_2 to the boundary of the neighboring region π_5.
- AP_k is a finite set of atomic propositions known to agent k. Without loss of generality, we assume that $AP_k \cap AP_{k'} = \emptyset$ for all $k \neq k' \in \mathcal{I}$.
- $L_k : \Pi_k \to 2^{AP_k}$ is a labeling function that assigns to each state $\pi_j^k \in \Pi_k$ a subset of atomic propositions AP_k that are satisfied when agent k is in region π_j.

1) Individual Timed Runs and Words: The behaviors of the individual agents can be captured through their timed runs and timed words. The timed run $r_k^t = (r_k(0), \tau_k(0))(r_k(1), \tau_k(1)) \cdots, \ k \in \mathcal{I}$ of each WTS $\mathcal{T}_k, \ k \in \mathcal{I}$ and the corresponding timed words $w(r_k^t) = (L_k(r_k(0)), \tau_k(0)) \ (L_k(r_k(1)), \tau_k(1)) \ \cdots$ are defined by using the terminology of Def. 4.

2) Collective Timed Run and Word: At the same time, the agents form a team and we are interested in their global, collective behaviors, which we formalize through the following definition.

Definition 8. Let $r_1^t, \dots, r_N^t$ be individual timed runs of the agents $1, \dots, N$, respectively, as defined above. Then, the *collective timed run* $r_G = (r_G(0), \tau_G(0))(r_G(1), \tau_G(1)) \dots$ of the team of agents is defined inductively as follows

1) $(r_G(0), \tau_G(0)) = ((r_1(0), \dots, r_N(0)), \tau_G(0))$.

2) Let $(r_G(i), \tau_G(i)) = ((r_1(i_1), \dots, r_N(i_N)), \tau_G(i))$, where $i \geq 0$ be the current state and time stamp of the collective timed run. Then the next state and time stamp $(r_G(i+1), \tau_G(i+1)) = ((r_1(j_1), \dots, r_N(j_N)), \tau_G(i+1))$ are given by the following

- $\ell = \underset{k \in \mathcal{I}}{\text{argmin}}\{\tau_k(i_k+1)\}$.
- $\tau_G(i+1) = \tau_\ell(i_\ell+1)$.
- $r_k(j_k) = \begin{cases} r_\ell(i_\ell+1) & \text{if } k = \ell \\ r_k(i_\ell) & \text{if } k \neq \ell. \end{cases}$

Intuitively, given the current states $r_1(i_1), \dots, r_N(i_N)$ and the next states $r_1(i_1+1), \dots, r_N(i_N+1)$ of the individual agents at time $\tau_G(i)$, ℓ is the index of the agent k who will finish its current transition from $r_\ell(i_\ell)$ to $r_\ell(i_\ell+1)$ the soonest amongst all. The time of agent ℓ's arrival to its next state $r_\ell(i_\ell+1)$ becomes the new time stamp $\tau_G(i+1)$ of the collective timed run. The next state of the collective timed run reflects that each agent k which cannot complete its transition from $r_k(i_k)$ to $r_k(i_k+1)$ before $\tau_G(i+1)$ remains in $r_k(i_k)$.

In what follows, $r_G^t = (r_G(0), \tau_G(0))(r_G(1), \tau_G(1))\dots$, where $r_G(i) = (r_1(i_1), \dots, r_N(i_N))$, $i, i_k \geq 0$ and $k \in \mathcal{I}$ denotes the collective timed run.

Definition 9. We define the global set of atomic propositions $AP_G = \bigcup_{k=1}^{N} AP_k$ and for every state $r_G(i) = (r_1(i_1), \dots, r_N(i_N))$ of a collective timed run, where $i, i_k \geq 0$ and $k \in \mathcal{I}$, we define the labeling function $L_G : \Pi_1 \dots \Pi_N \to AP_G$ as $L_G(r_G(i)) = \bigcup_{k=1}^{N} L_k(r_k(i_k))$.

A collective timed run r_G^t thus naturally produces a timed word $w_G^t = (L_G(r_G(0)), \tau_G(0))(L_G(r_G(1)), \tau_G(1))\dots$ over AP_G.

Example 1. Consider $N = 2$ robots operating in a workspace with $\mathcal{W} = \pi_1 \cup \pi_2 \cup \pi_3, W_0 = 3$ and $\mathcal{I} = \{1, 2\}$ modeled as the WTSs illustrated in Fig. 1. Let $AP_1 = \{green\}$, and $AP_2 = \{red\}$. The labeling functions are $L_1(\pi_1^1) = \{green\}, L_1(\pi_2^1) = L_1(\pi_3^1) = \emptyset$, and $L_2(\pi_1^2) = L_2(\pi_2^2) = \emptyset, L_2(\pi_3^2) = \{red\}$.

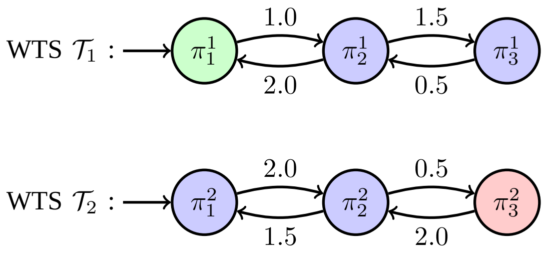

Fig. 1. WTSs T_1, T_2 representing two agents in $\mathcal{W}$. $\Pi_1 = \{\pi_1^1, \pi_2^1, \pi_3^1\}$, $\Pi_1^{\text{init}} = \{\pi_1^1\}$, $\Pi_2 = \{\pi_1^2, \pi_2^2, \pi_3^2\}, \Pi_2^{\text{init}} = \{\pi_1^2\}$, the transitions are depicted as arrows which are annotated with the corresponding weights.

Examples of the agents' runs are:

$$\begin{aligned} r_1^t =& (r_1(0) = \pi_1^1, \tau_1(0) = 0.0)(r_1(1) = \pi_2^1, \tau_1(1) = 1.0) \\ & (r_1(2) = \pi_3^1, \tau_1(2) = 2.5)(r_1(3) = \pi_2^1, \tau_1(3) = 3.0) \\ & (r_1(4) = \pi_1^1, \tau_1(4) = 5.0)\dots \\ r_2^t =& (r_2(0) = \pi_1^2, \tau_2(0) = 0.0)(r_2(1) = \pi_2^2, \tau_2(1) = 2.0) \\ & (r_2(2) = \pi_3^2, \tau_2(2) = 2.5)(r_2(3) = \pi_2^2, \tau_2(3) = 4.5) \\ & (r_2(4) = \pi_3^2, \tau_2(4) = 5.0)\dots \end{aligned}$$

Given r_1^t and r_2^t the collective run r_G is given according to Def. 8 as follows:

$$\begin{aligned} r_G^t =& (\underbrace{(\pi_1^1, \pi_1^2)}_{r_G(0)}, \tau_G(0) = 0.0)(\underbrace{(\pi_2^1, \pi_1^2)}_{r_G(1)}, \tau_G(1) = 1.0) \\ & (\underbrace{(\pi_2^1, \pi_2^2)}_{r_G(2)}, \tau_G(2) = 2.0)(\underbrace{(\pi_3^1, \pi_3^2)}_{r_G(3)}, \tau_G(3) = 2.5) \\ & (\underbrace{(\pi_2^1, \pi_3^2)}_{r_G(4)}, \tau_G(4) = 3.0)(\underbrace{(\pi_2^1, \pi_2^2)}_{r_G(5)}, \tau_G(5) = 4.5) \\ & (\underbrace{(\pi_1^1, \pi_3^2)}_{r_G(6)}, \tau_G(6) = 5.0)\dots \end{aligned}$$

The produced collective timed word is

$$\begin{aligned} w_G^t =& (\{green\}, 0.0)(\emptyset, 1.0)(\emptyset, 2.0)(\{red\}, 2.5) \\ & (\{red\}, 3.0)(\emptyset, 4.5)(\{green, red\}, 5.0)\dots. \end{aligned}$$

B. Specification

Several different logics have been designed to express timed properties of real-time systems, such as MTL [37] that extends the until operator of LTL with a time interval. Here, we consider a fragment of MTL, called MITL (see Sec. II for definition) which has been proposed in [26]. Namely, we utilize its point-wise semantics and interpret its formulas over timed runs. Unlike MTL, MITL excludes punctual constraints on the until operator. For instance, the formula $\Box(a \Rightarrow \Diamond_{=1} b)$ saying that every a is followed by a b precisely 1 time unit later, is not allowed in MITL, whereas $\Box(a \Rightarrow \Diamond_{(0,1]} b)$, saying that every a is followed by a b at most after 1 time unit later, is. While MTL formulas cannot be generally translated into TBAs, MITL formulas can [26].

1) Local Agent's Specification: Each agent k, $k \in \mathcal{I}$ is given an individual, local, independent specification in the form of a MITL formula φ_k over the set of atomic propositions AP_k. The satisfaction of φ_k is decided from the agent's own perspective, i.e., on the timed run r_k^t.

2) Global Team Specification: In addition, the team of agents is given a global team specification, which is a MITL formula φ_G over the set of atomic propositions AP_G. The team specification satisfaction is decided on the collective timed run r_G^t.

Example 1 (Continued)**.** Recall the two agents from Example 1. Each of the agents is given a local, independent, specification and at the same time, the team is given an overall goal that may require collaboration or coordination. Examples of local specification formulas are $\varphi_1 = \Box\Diamond_{\leq 10}(green)$ and $\varphi_2 = \Box(red \Rightarrow \bigcirc\Box_{\leq 5}(\neg red))$ stating that "The green region is periodically visited with at most 10 time units between two consecutive visits" and "Whenever a red region is visited, it will not be visited for the following 5 time units again", respectively. While φ_1 is satisfied on r_1^t, φ_2 is not satisfied on r_2^t. An example of the global specification is $\varphi_G = \Box\Diamond_{\leq 5}(green \wedge red)$ that imposes requirement on the agents' collaboration; it states that agents 1 and 2 will periodically simultaneously visit the green and the red region, respectively, with at most 5 time units between two consecutive visits.

C. Problem Statement

Problem 1 (Run Synthesis)**.** Given N agents governed by dynamics as in (4), a task specification MITL formula φ_G for the team of robots, over a set of atomic propositions AP_G and N local task specifications φ_k over $AP_k,\ k \in \mathcal{I}$, synthesize a sequence of individual timed runs $r_1^t, \dots, r_N^t$ such that the following hold

$$\left(r_G^t \models \varphi_G\right) \wedge \left(r_1^t \models \varphi_1 \wedge \ldots \wedge r_N^t \models \varphi_N\right). \qquad (5)$$

Though it might seem that the satisfaction of the individual specifications $\varphi_1, \dots, \varphi_N$ can be treated as the satisfaction of the formula $\bigwedge_{k\in\mathcal{I}} \varphi_k$ on the collective timed run r_G^t, this is generally not the case, as demonstrated through the following example:

Example 1 (Continued)**.** Recall the two agents from Example 1 and a local specification $\varphi_2 = \Box(red \Rightarrow \bigcirc\Box_{\leq 2}(\neg red))$. While this specification is satisfied on r_2^t since $w(r_2^t) = (\emptyset, 0.0)(\emptyset, 2.0)(\{red\}, 2.5)(\emptyset, 4.5)(\{green, red\}, 5.0)\ldots$, it can be easily seen that it is not satisfied on r_G^t.

Formally, we have

$$r_G^t \models \bigwedge_{k\in\mathcal{I}} \varphi_k \not\Leftrightarrow r_1^t \models \varphi_1 \wedge \ldots \wedge r_N^t \models \varphi_N. \qquad (6)$$

Hence, Problem 1 may not be treated in a straightforward, fully centralized way. We propose a two-stage solution that first pre-computes all timed runs of the individual agents in a decentralized way and stores them efficiently in weighted transition systems enhanced with a Büchi acceptance condition. Second, these are combined and inspected with respect to guaranteeing the satisfaction of the team specification by the collective timed run.

IV. Proposed Solution

In this section, we introduce a systematic solution to Problem 1. Our overall approach builds on the following steps:

1) We construct TBAs $\mathcal{A}_k,\ k \in \mathcal{I}$ and $\mathcal{A}_G$ that accept all the timed words satisfying the specification formulas $\varphi_k,\ k \in \mathcal{I}$ and φ_G, respectively (Sec. IV-A).
2) We construct a *local Büchi WTS* $\widetilde{\mathcal{T}}_k = \mathcal{T}_k \otimes \mathcal{A}_k$, for all $k \in \mathcal{I}$. The accepting timed runs of $\widetilde{\mathcal{T}}_k$ are the timed runs of the $\mathcal{T}_k$ that satisfy the corresponding local specification formula $\varphi_k, k \in \mathcal{I}$ (Sec. IV-B).
3) We construct a *product Büchi WTS* $\mathcal{T}_G = \widetilde{\mathcal{T}}_1 \otimes \cdots \otimes \widetilde{\mathcal{T}}_N$ such that its timed runs are collective timed runs of the team and their projections onto the agents' individual timed runs are admissible by the local Büchi WTSs $\widetilde{\mathcal{T}}_1, \ldots \widetilde{\mathcal{T}}_N$ respectively (Sec. IV-C).
4) We construct a *global Büchi WTS* $\widetilde{\mathcal{T}}_G = \mathcal{T}_G \otimes \mathcal{A}_G$. The accepting timed runs of the $\widetilde{\mathcal{T}}_G$ are the timed runs of the $\mathcal{T}_G$ that satisfy the team formula φ_G (Sect. IV-D).
5) We find an accepting timed run $\widetilde{r}_G^t$ of the global Büchi WTS $\widetilde{\mathcal{T}}_G$ and project it onto timed runs of the product Büchi WTS $\mathcal{T}_G$, then onto timed runs of the local Büchi WTSs $\widetilde{\mathcal{T}}_1, \dots, \widetilde{\mathcal{T}}_N$, and finally onto individual timed runs $r_1^t, \dots, r_N^t$ of the original WTSs $\mathcal{T}_1, \dots, \mathcal{T}_N$. By construction, $r_1^t, \dots, r_N^t$ are guaranteed to satisfy $\varphi_1, \dots, \varphi_N$, respectively, and furthermore r_G^t satisfies φ_G (Sec. IV-E).

A. Construction of TBAs

As stated in Sec. II, every MITL formula φ can be translated into a language equivalent TBA. Several approaches are proposed for that purpose, for instance [26], [32], [33], [38]. Here, we translate each local specification φ_k, where $k \in \mathcal{I}$ into a TBA $\mathcal{A}_k = (S_k, S_k^{\text{init}}, X_k, I_k, E_k, \mathcal{F}_k, AP_k, \mathcal{L}_k)$, and the global specification φ_G into a TBA $\mathcal{A}_G = (S_G, S_G^{\text{init}}, X_G, I_G, E_G, \mathcal{F}_G, AP_G, \mathcal{L}_G)$.

B. Construction of the local Büchi WTSs $\widetilde{\mathcal{T}}_1, \dots, \widetilde{\mathcal{T}}_N$

Definition 10. Given a WTS $\mathcal{T}_k = (\Pi_k, \Pi_k^{\text{init}}, \rightarrow_k, AP_k, L_k, d_k)$, and a TBA $\mathcal{A}_k = (S_k, S_k^{\text{init}}, X_k, I_k, E_k, F_k, AP_k, \mathcal{L}_k)$ with $M_k = |X_k|$ and C_k^{max} being the largest constant appearing in $\mathcal{A}_k$, we define their *local Büchi WTS* $\widetilde{\mathcal{T}}_k = \mathcal{T}_k \otimes \mathcal{A}_k = (Q_k, Q_k^{init}, \rightsquigarrow_k, \widetilde{d}_k, \widetilde{F}_k, AP_k, \widetilde{L}_k)$ as follows:

- $Q_k \subseteq \{(r_k, s_k) \in \Pi_k \times S_k : L_k(r_k) = \mathcal{L}_k(s_k)\} \times \mathbb{T}_\infty^{M_k}$.
- $Q_k^{init} = \Pi_k^{init} \times S_k^{init} \times \underbrace{\{0\} \times \ldots \times \{0\}}_{M_k\ products}$.
- $q \rightsquigarrow_k q'$ iff
  - $q = (r, s, \nu_1, \dots, \nu_{M_k}) \in Q_k$, $q' = (r', s', \nu'_1, \dots, \nu'_{M_k}) \in Q_k$,
  - $r \rightarrow_k r'$, and
  - there exists γ, R, such that $(s, \gamma, R, s') \in E_k$, $\nu_1, \dots, \nu_{M_k} \models \gamma$, $\nu'_1, \dots, \nu'_{M_k} \models I_k(s')$, and for all $i \in \{1, \dots, M_k\}$

$$\nu'_i = \begin{cases} 0, & \text{if } x_i \in R \\ \nu_i + d_k(r, r'), & \text{if } x_i \notin R \text{ and} \\ & \nu_i + d_k(r,r') \leq C_k^{max} \\ \infty, & \text{otherwise.} \end{cases}$$

  Then $\widetilde{d}_k(q, q') = d_k(r, r')$.
- $\widetilde{F}_k = \{(r_k, s_k, \nu_1, \dots, \nu_{M_k}) \in Q_k : s_k \in F_k\}$.
- $\widetilde{L}_k(r_k, s_k, \nu_1, \dots, \nu_{M_k}) = L_k(r_k)$.

Each local Büchi WTS $\widetilde{\mathcal{T}}_k, k \in \mathcal{I}$ is in fact a WTS with a Büchi acceptance condition $\widetilde{F}_k$. A timed run of $\widetilde{\mathcal{T}}_k$ can be written as $\widetilde{r}_k^t = (q_k(0), \tau_k(0))(q_k(1), \tau_k(1))\ldots$ using the terminology of Def. 4. It is *accepting* if $q_k(i) \in \widetilde{F}_k$ for infinitely many $i \geq 0$. An accepting timed run of $\widetilde{\mathcal{T}}_k$ projects onto a timed run of $\mathcal{T}_k$ that satisfies the local specification formula φ_k by construction. Formally, the following lemma, whose proof follows directly from the construction and and the principles of automata-based LTL model checking (see, e.g., [39]), holds:

Lemma 1. *Consider an accepting timed run $\widetilde{r}_k^t = (q_k(0), \tau_k(0))(q_k(1), \tau_k(1))\ldots$ of the local Büchi WTS $\widetilde{\mathcal{T}}_k$ defined above, where $q_k(i) = (r_k(i), s_k(i), \nu_{k,1}, \ldots, \nu_{k,M_k})$ denotes a state of $\widetilde{\mathcal{T}}_k$, for all $i \geq 1$. The timed run $\widetilde{r}_k^t$ projects onto the timed run $r_k^t = (r_k(0), \tau_k(0))(r_k(1), \tau_k(1))\ldots$ of the WTS $\mathcal{T}_k$ that produces the timed word $w(r_k^t) = (L_k(r_k(0)), \tau_k(0))(L_k(r_k(1)), \tau_k(1))\ldots$ accepted by the TBA $\mathcal{A}_k$ via its run $\rho_k = s_k(0)s_k(1)\ldots$ Vice versa, if there exists a timed run $r_k^t = (r_k(0), \tau_k(0))(r_k(1), \tau_k(1))\ldots$ of the WTS $\mathcal{T}_k$ that produces a timed word $w(r_k^t) = (L_k(r_k(0)), \tau_k(0))(L_k(r_k(1)), \tau_k(1))\ldots$ accepted by the TBA $\mathcal{A}_k$ via its run $\rho_k = s_k(0)s_k(1)\ldots$ then there exist the accepting timed run $\widetilde{r}_k^t = (q_k(0), \tau_k(0))(q_k(1), \tau_k(1))\ldots$ of $\widetilde{\mathcal{T}}_k$, where $q_k(i)$ denotes $(r_k(i), s_k(i), \nu_{k,1}(i), \ldots, \nu_{k,M_k}(i))$ in $\widetilde{\mathcal{T}}_k$.*

C. Construction of the product Büchi WTS $\mathcal{T}_G$

Now we aim to construct a finite product WTS $\mathcal{T}_G$ whose timed runs represent the collective behaviors of the team and whose Büchi acceptance condition ensures that the accepting timed runs account for the local specifications. In other words, $\mathcal{T}_G$ is a product of all the local WTS $\widetilde{\mathcal{T}}_k$ built above. In the construction of $\mathcal{T}_G$, we need to specifically handle the cases when transitions of different agents are associated with different time durations, i.e, different transition weights. To this end, we introduce a vector $b = (b_1, \ldots, b_N) \in \mathbb{T}^N$. Each element of the vector is a rational number $b_k \in \mathbb{T}, k \in \mathcal{I}$ which can be either 0, when the agent k has just completed its transition, or the time elapsed from the beginning of the agent's current transition, if this transition is not completed, yet. The state of the team of agents is then in the form $q_G = (q_1, \ldots, q_N, b_1, \ldots, b_N, \ell)$ where q_k is a state of $\widetilde{\mathcal{T}}_k$, for all $k \in \mathcal{I}$, and $\ell \in \mathcal{I}$ has a special meaning in relation to the acceptance condition of $\mathcal{T}_G$ that will become clear shortly. Taking the above into consideration we define the global model $\mathcal{T}_G$ as follows:

Definition 11. Given N local Büchi WTSs $\widetilde{\mathcal{T}}_1, \ldots, \widetilde{\mathcal{T}}_N$ from Def. 10, their *product Büchi WTS* $\mathcal{T}_G = \widetilde{\mathcal{T}}_1 \otimes \ldots \otimes \widetilde{\mathcal{T}}_N = (Q_G, Q_G^{init}, \rightarrow_G, d_G, F_G, AP_G, L_G)$ is defined as follows:

- $Q_G \subseteq Q_1 \times \cdots \times Q_N \times \mathbb{T}^N \times \{1, \ldots, N\}$.
- $Q_G^{init} = Q_1^{init} \times \ldots \times Q_N^{init} \times \underbrace{\{0\} \times \ldots \times \{0\}}_{N\ products} \times \{1\}$.
- $q_G \rightarrow_G q'_G$ iff
  - $q_G = (q_1, \ldots, q_N, b_1, \ldots, b_N, \ell) \in Q_G$,
    $q'_G = (q'_1, \ldots, q'_N, b'_1, \ldots, b'_N, \ell') \in Q_G$,
  - $\exists\ q''_k \in Q_k : q_k \rightsquigarrow_k q''_k$, for some $k \in \mathcal{I}$,
  - $$b'_k = \begin{cases} 0, & \text{if } b_k + d_{min} = \widetilde{d}_k(q_k, q''_k) \\ & \text{and } q'_k = q''_k \\ b_k + d_{min}, & \text{if } b_k + d_{min} < \widetilde{d}_k(q_k, q''_k) \\ & \text{and } q'_k = q_k \end{cases}$$
    where $d_{min} = \min\limits_{k \in \{1, \ldots, N\}} (\widetilde{d}_k(q_k, q''_k) - b_k)$ is (loosely speaking) the smallest time step that can be applied, and
  - $$\ell' = \begin{cases} \ell, & \text{if } q_\ell \notin \widetilde{F}_\ell \\ ((\ell + 1) \mod N), & \text{otherwise} \end{cases}$$

  Then $d_G(q_G, q'_G) = d_{min}$.
- $F_G = \{(q_1, \ldots, q_N, b_1, \ldots, b_N, N) \in Q_G : q_N \in \widetilde{F}_N\}$.
- $AP_G = \bigcup\limits_{k=1}^{N} AP_k$.
- $L_G((q_1, \ldots, q_N, b_1, \ldots, b_N, \ell) = \bigcup\limits_{k=1}^{N} \widetilde{L}_k(q_k)$.

The product WTS $\mathcal{T}_G$ is again a WTS with a Büchi acceptance condition. Informally, the index ℓ in a state $q_G = (q_1, \ldots, q_N, b_1, \ldots, b_N, \ell) \in Q_G$ allows to project an accepting timed run of $\mathcal{T}_G$ onto an accepting run of every one of the local Büchi WTS. The construction is based on the standard definition of Büchi automata intersection (see, e.g., [39]).

The following lemma follows directly from the construction and and the principles of automata-based LTL model checking (see, e.g., [39]):

Lemma 2. *For all $k \in \mathcal{I}$, an accepting timed run r_G^t of the product Büchi WTS $\mathcal{T}_G$ projects onto an accepting timed run $r_k^t$of the local Büchi WTS $\widetilde{\mathcal{T}}_k$ that produces a timed word $w(r_k^t)$ accepted by the corresponding TBA $\mathcal{A}_k$. Vice versa, if there exists a timed run r_k^t of the local Büchi WTS $\widetilde{\mathcal{T}}_k$ that produces a timed word $w(r_k^t)$ accepted by the TBA $\mathcal{A}_k$ for each $k \in \mathcal{I}$, then there exist an accepting timed run r_G^t of $\mathcal{T}_G$.*

D. Construction of the global Büchi WTS $\widetilde{\mathcal{T}}_G$

Definition 12. Finally, given the product Büchi WTS $\mathcal{T}_G = (Q_G, Q_G^{\text{init}}, \rightarrow_G, d_G, F_G, AP_G, L_G)$, and a TBA $\mathcal{A}_G = (S_G, S_G^{\text{init}}, X_G, I_G, E_G, \mathcal{F}_G, AP_G, \mathcal{L}_G)$ that corresponds to the team specification formula φ_G with $M_G = |X_G|$ and C_G^{max} being the largest constant appearing in $\mathcal{A}_G$, we define their product WTS $\widetilde{\mathcal{T}}_G = \mathcal{T}_G \otimes \mathcal{A}_G = (\widetilde{Q}_G, \widetilde{Q}_G^{init}, \rightsquigarrow_G, \widetilde{d}_G, \widetilde{F}_G, AP_G, \widetilde{L}_G)$ as follows:

- $\widetilde{Q}_G \subseteq \{(q, s) \in Q_G \times S_G : L_G(q) = \mathcal{L}_G(s)\} \times \mathbb{T}_\infty^{M_G}$.
- $\widetilde{Q}_G^{init} = Q_G^{init} \times S_G^{init} \times \underbrace{\{0\} \times \ldots \times \{0\}}_{M_G-products} \times \{1, 2\}$.
- $q \rightsquigarrow_G q'$ iff
  - $q = (r, s, \nu_1, \ldots, \nu_{M_G}, \ell) \in Q_G$,
    $q' = (r', s', \nu'_1, \ldots, \nu'_{M_G}, \ell') \in Q_G$,
  - $r \rightarrow_G r'$, and
  - there exists γ, R, such that $(s, \gamma, R, s') \in E_G$, $\nu_1, \ldots, \nu_{M_G} \models \gamma$, $\nu'_1, \ldots, \nu'_{M_G} \models I_G(s')$, and for all $i \in \{1, \ldots, M_G\}$
    $$\nu'_i = \begin{cases} 0, & \text{if } x_i \in R \\ \nu_i + d_G(r, r'), & \text{if } x_i \notin R \text{ and} \\ & \nu_i + d_G(r, r') \leq C_G^{max} \\ \infty, & \text{otherwise} \end{cases}$$
  - $$\ell' = \begin{cases} 1 \text{ if } \ell = 1 \text{ and } r \notin F_G, \text{ or } \ell = 2 \text{ and } s \in \mathcal{F}_G \\ 2 \text{ otherwise} \end{cases}$$

  Then $\widetilde{d}_G(q, q') = d_G(r, r')$.
- $\widetilde{F}_G = \{(r, s, \nu_1, \ldots, \nu_{M_G}, 1) \in Q_G : r \in F_G\}$.
- $\widetilde{L}_G(r_G, s_G, \nu_1, \ldots, \nu_{M_G}) = L_G(r_G)$.

Analogously to above, the global Büchi WTS $\widetilde{\mathcal{T}}_G$ is a WTS with a Büchi acceptance condition. An accepting timed run of $\widetilde{\mathcal{T}}_G$ guarantees the satisfaction of the team specification formula φ_G by construction. Furthermore, the projected individual timed runs of the original $\mathcal{T}_1, \ldots, \mathcal{T}_N$ satisfy their respective local specifications. The following lemma follows directly from the construction and and the principles of automata-based LTL model checking (see, e.g., [39]):

Lemma 3. *An accepting timed run $\widetilde{r}_G^t$ of the global Büchi WTS $\widetilde{\mathcal{T}}_G$ projects onto an accepting timed run r_G^t of the product Büchi WTS $\mathcal{T}_G$ that produces a timed word $w(r_G^t)$ accepted by the TBA $\mathcal{A}_G$. Vice versa, if there exists a timed run r_G^t of the product Büchi WTS $\mathcal{T}_G$ that produces a timed word $w(r_G^t)$ accepted by the TBA $\mathcal{A}_G$ then there exist an accepting timed run $\widetilde{r}_G^t$ of $\widetilde{\mathcal{T}}_G$.*

E. Projection to the desired timed runs of $\mathcal{T}_1, \ldots, \mathcal{T}_N$

An accepting run $\widetilde{r}_G^t$ of the global Büchi WTS $\widetilde{\mathcal{T}}_G$ can be found efficiently leveraging ideas from automata-based LTL model checking [39]. Namely, $\widetilde{\mathcal{T}}_G$ is viewed as a graph that is searched for a so-called accepting lasso; a cycle containing an accepting state that is reachable from the initial state. Once $\widetilde{r}_G^t$ is obtained, Lemmas 3, 2, and 1 directly provide guidelines for projection of $\widetilde{r}_G^t$ onto the individual timed runs of $\mathcal{T}_1, \ldots, \mathcal{T}_N$. In particular, $\widetilde{r}_G^t$ is projected onto a timed run r_G^t of $\mathcal{T}_G$, which is projected onto timed runs $\widetilde{r}_1^t, \ldots, \widetilde{r}_N^t$ of $\widetilde{\mathcal{T}}_1, \ldots, \mathcal{T}_N$, which are finally projected onto timed runs $r_1^t, \ldots, r_N^t$ of $\mathcal{T}_1, \ldots, \mathcal{T}_N$, respectively. Such a projection guarantees that $r_1^t, \ldots, r_N^t$ are a solution to Problem 1.

V. Illustrative Example

For an illustrative example, consider 2 robots in the shared workspace of Fig. 2. The workspace is partitioned into $W = 21$ cells and a robot's state is defined by the cell it is currently present at. Agent 1 (R1) is depicted in green and it is two times faster than Agent 2 (R2) which is depicted in red. We assume that the environment imposes such moving constraints that the traveling right and up is faster than left and down. Let Agent 1 need 1 time unit for up and right moves and 2 time units for down and left moves. Let also Agent 2 need 2 time units for up and right moves and 4 time units for down and left moves.

We consider a scenario where the robots have to eventually meet at yellow regions (global team task), and at the same time, they have to recharge within a certain time interval in recharge locations (blue squares with the circles in the respective color). The individual specifications are $\varphi_1 = \Diamond_{\leq 6}(\mathit{recharge1})$ and $\varphi_2 = \Diamond_{\leq 12}(\mathit{recharge2})$ stating that agent 1 has to recharge within 5 time units and agent 2 within 10 units, respectively, and the team task is $\varphi_G = \Diamond_{\leq 30}\{(\mathit{meet}_1^A \wedge \mathit{meet}_2^A) \vee (\mathit{meet}_1^B \wedge \mathit{meet}_2^B)\}$ stating that the agents have to meet either in yellow region A or B within 30 time units.

By following the process that was described in Section IV step by step we have that an accepting timed run is $\widetilde{r}_G^t = ((\pi_4^1, \pi_{18}^2), 0)((\pi_{11}^1, \pi_{18}^2), 2) \ldots ((\pi_9^1, \pi_{10}^2), 6)((\pi_{16}^1, \pi_3^2), 8) \ldots ((\pi_{13}^1, \pi_5^2), 13)((\pi_6^1, \pi_6^2), 14) \ldots$ with corresponding timed word $w(\widetilde{r}_G^t) = (\emptyset, 0)(\emptyset, \pi_{18}^2), 2) \ldots (\{\mathit{recharge1}\}, 6)(\{\mathit{recharge2}\}, 8) \ldots (\emptyset, \pi_5^2), 13)((\{\mathit{meet}_1^A, \mathit{meet}_2^A\}, 14) \ldots$ which satisfies formula ϕ_G. The run $\widetilde{r}_G^t$ can be projected

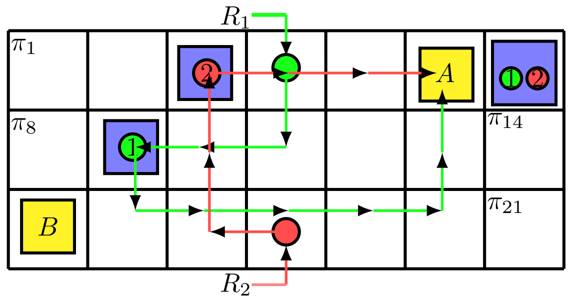

Fig. 2. An illustrative example with 2 robots evolving in a common workspace. Let $\mathcal{W}_0 = \pi_1 \cup \ldots \cup \pi_{21}$. We enumerate the regions starting from the left region in every row and ending in the right. The initial positions of robots R_1, R_2 are depicted by a green and a red circle, respectively, the desired meeting points in yellow and the recharging spots by the agents' respective colors inside a blue box. The accepting runs for task specifications ϕ_1, ϕ_2, ϕ_G are depicted with green and red arrows for agent 1 and agent 2 respectively.

onto individual the timed runs $\widetilde{r_1}^t = (\pi_4^1, 0)(\pi_{11}^1, 2)(\pi_{10}^1, 4)(\pi_9^1, 6)(\pi_{16}^1, 8)(\pi_{17}^1, 9)(\pi_{18}^1, 10)(\pi_{19}^1, 11)(\pi_{20}^1, 12)(\pi_{13}^1, 13)(\pi_6^1, 14) \ldots$ and $\widetilde{r_2}^t = (\pi_{18}^2, 0)(\pi_{17}^2, 4)(\pi_{10}^2, 6)(\pi_3^2, 8)(\pi_4^2, 10)(\pi_5^2, 12)(\pi_6^2, 14) \ldots$ (they are depicted in Fig. 2 with green and red arrows respectively) with corresponding timed words $w(\widetilde{r}_1^t) = (\emptyset, 0)(\emptyset, 2)(\emptyset, 4)(\{\mathit{recharge1}\}, 6)(\emptyset, 8)(\emptyset, 9)(\emptyset, 10)(\emptyset, 11)(\emptyset, 12)(\emptyset, 13)(\{\mathit{meet}_1^A\}, 14) \ldots$ and $w(r_2^t) = (\emptyset, 0)(\emptyset, 4)(\emptyset, 6)(\{\mathit{recharge2}\}, 8)(\emptyset, 10)(\emptyset, 12)(\{\mathit{meet}_2^A\}, 14) \ldots$ which satisfy formulas ϕ_1 and ϕ_2 respectively. All conditions from (5) are satisfied. The runs and the words of the illustrative example are depicted in Fig. 3.

Consider now the alternative runs of the agents, where they first meet in the meeting point A (after 8 time units) and then recharge in the region π_7 (after 9 and 10 time units, respectively). Regardless of how the agents continue, they have accomplished the untimed formulas $\varphi_1' = \Diamond(\mathit{recharge1})$, $\varphi_2' = \Diamond(\mathit{recharge2})$, and $\varphi_G' = \Diamond\{(\mathit{meet}_1^A \wedge \mathit{meet}_2^A) \vee (\mathit{meet}_1^B \wedge \mathit{meet}_2^B)\}$. Although this is in fact a more efficient way to satisfy the untimed formulas φ_1', φ_2', and φ_G' than the one described above, the formula φ_1 is violated due to its time constraint.

VI. Conclusions and Future Work

We have proposed a systematic method for multi-agent controller synthesis aiming cooperative planning under high-level specifications given in MITL formulas. The solution involves a sequence of algorithmic automata constructions such that not only team specifications but also individual specifications should be fulfilled. Future research directions include the consideration of more complicated dynamics than the fully actuated ones in (4), the decentralized solution such that every agent has information only from his neighbors as well as the modeling of the system with Markov Decision Processes (MDPs) and probabilistic verification.

References

[1] W. Ren and R. Beard, "Consensus Seeking in Multi-agent Systems under Dynamically Changing Interaction Topologies," *IEEE Transactions on automatic control*, vol. 50, no. 5, pp. 655–661, 2005.

[2] R. Olfati-Saber and R. Murray, "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[3] A. Jadbabaie, J. Kin, and S. Morse, "Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

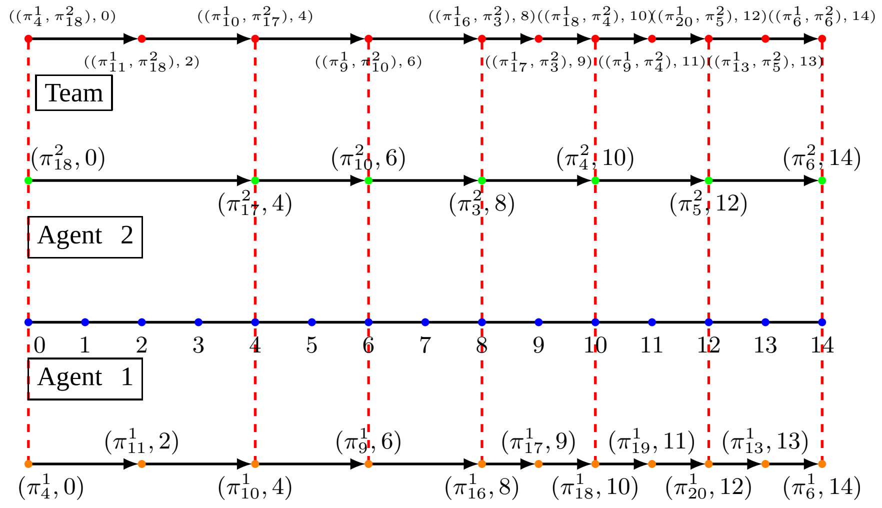

Fig. 3. The accepting runs $\tilde{r}^t_1, \tilde{r}^t_2$, the collective run $\tilde{r}^t_G$ and the corresponding timed stamps. We denote with red dashed lines the times that both agents have the same time stamps

[4] M. Zavlanos and G. Pappas, "Distributed Connectivity Control of Mobile Networks," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1416–1428, 2008.

[5] M. Egerstedt and X. Hu, "Formation Constrained Multi-Agent Control," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 947–951, 2001.

[6] H. Tanner, A. Jadbabaie, and G. Pappas, "Flocking in Fixed and Switching Networks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 863–868, 2007.

[7] D. Dimarogonas and K. J. Kyriakopoulos, "On the Rendezvous Problem for Multiple Nonholonomic Agents," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 916–922, 2007.

[8] A. Bhatia, M. Maly, L. Kavraki, and M. Vardi, "Motion Planning with Complex Goals," *IEEE Robotics and Automation Magazine,*, vol. 18, no. 3, pp. 55–64, 2011.

[9] H. Kress-Gazit, G. Fainekos, and G. Pappas, "Temporal-Logic-Based Reactive Mission and Motion Planning," *Robotics, IEEE Transactions on*, vol. 25, no. 6, pp. 1370–1381, 2009.

[10] S. Loizou and K. Kyriakopoulos, "Automatic Synthesis of Multi-Agent Motion Tasks Based on LTL Specifications," *IEEE Conference on Decision and Control (CDC)*, vol. 1, pp. 153–158, 2004.

[11] M. Guo and D. Dimarogonas, "Multi-Agent Plan Reconfiguration Under Local LTL Specifications," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 218–235, 2015.

[12] S. Karaman and E. Frazzoli, "Linear Temporal Logic Vehicle Routing with Applications to Multi-UAV Mission Planning," *International Journal of Robust and Nonlinear Control*, vol. 21, no. 12, pp. 1372–1395, 2011.

[13] M. Kloetzer and C. Belta, "Automatic Deployment of Distributed Teams of Robots From Temporal Logic Motion Specifications," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 48–61, 2010.

[14] Y. Chen, X. Ding, A. Stefanescu, and C. Belta, "A Formal Approach to Deployment of Robotic Teams in an Urban-Like Environment," *Distributed Autonomous Robotic Systems*, pp. 313–327, 2013.

[15] Y. Chen, Ding, X. Chu, A. Stefanescu, and C. Belta, "Formal Approach to the Deployment of Distributed Robotic Teams," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 158–171, 2012.

[16] M. Kloetzer, X. C. Ding, and C. Belta, "Multi-Robot Deployment from LTL Specifications with Reduced Communication," *IEEE Conference o Decision and Control (CDC)*, pp. 4867–4872, 2011.

[17] A. Ulusoy, S. Smith, X. Ding, C. Belta, and D. Rus, "Optimality and Robustness in Multi-Robot Path Planning with Temporal Logic Constraints," *The Int. Jour. of Rob. Res.*, vol. 32, no. 8, pp. 889–911.

[18] M. Quottrup, T. Bak, and R. Zamanabadi, "Multi-Robot Planning: A Timed Automata Approach," *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 5, pp. 4417–4422, 2004.

[19] M. Andersen, R. Jensen, T. Bakand, and M. Quottrup, "Motion Planning in Multi-Robot Systems Using Timed Automata," *5th IFAC/EURON*, 2004.

[20] K. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a Nutshell," *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 1, no. 1, pp. 134–152, 1997.

[21] A. Ulusoy, S. Smith, X. Ding, C. Belta, and D. Rus, "Optimal Multi-Robot Path Planning with Temporal Logic Constraints," *2011 IROS*, pp. 3087–3092, 2011.

[22] J. Liu and P. Prabhakar, "Switching Control of Dynamical Systems from Metric Temporal Logic Specifications," *2014 ICRA*, pp. 5333–5338, 2014.

[23] V. Raman, A. Donzé, D. Sadigh, R. Murray, and S. Seshia, "Reactive Synthesis from Signal Temporal Logic Specifications," pp. 239–248.

[24] Y. Zhou, D. Maity, and J. S. Baras, "Optimal Mission Planner with Timed Temporal Logic Constraints," *European Control Conference (ECC)*, 2015.

[25] S. Karaman and E. Frazzoli, "Vehicle Routing Problem with Metric Temporal Logic Specifications," *IEEE Conference on Decision and Control (CDC)*, pp. 3953–3958, Dec 2008.

[26] R. Alur, T. Feder, and T. A. Henzinger, "The Benefits of Relaxing Punctuality," *Journal of the ACM (JACM)*, vol. 43, no. 1, pp. 116–146, 1996.

[27] R. Alur and D. Dill, "A Theory of Timed Automata," *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, 1994.

[28] D. D. Souza and P. Prabhakar, "On the Expressiveness of MTL in the Pointwise and Continuous Semantics," *International Journal on Software Tools for Technology Transfer*, vol. 9, no. 1, pp. 1–4, 2007.

[29] J. Ouaknine and J. Worrell, "On the Decidability of Metric Temporal Logic," *IEEE Symposium on Logic in CS*, pp. 188–197, 2005.

[30] P. Bouyer, "From Qualitative to Quantitative Analysis of Timed Systems," *Mémoire D ' habilitation, Université Paris*, vol. 7, pp. 135–175, 2009.

[31] S. Tripakis, "Checking Timed Buchi Automata Emptiness on Simulation Graphs," *ACM TCL*, vol. 10, no. 3, 2009.

[32] O. Maler, D. Nickovic, and A. Pnueli, "From MITL to Timed Automata," *Formal Modeling and Analysis of Timed Systems*, pp. 274–289, 2006.

[33] T. Brihaye, M. Estivenart, and G. Geeraerts, "On MITL and Alternating Timed Automata," *Formal Modeling and Analysis of Timed Systems*, vol. 8053, pp. 47–61, 2013.

[34] P. Tabuada, *Verification and Control of Hybrid Systems: a Symbolic Approach*. Springer Science and Business Media, 2009.

[35] A. Girard and G. J. Pappas, "Approximation Metrics for Discrete and Continuous Systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 782–798, 2007.

[36] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, "Discrete Abstractions of Hybrid Systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, 2000.

[37] R. Koymans, "Specifying Real-Time Properties with Metric Temporal Logic," *Real-time systems*, vol. 2, no. 4, pp. 255–299, 1990.

[38] D. Ničković and N. Piterman, "From MTL to Deterministic Timed Automata," *Formal Modeling and Analysis of Timed Systems*, pp. 152–167, 2010.

[39] C. Baier and J. Katoen, *Principles of Model Checking*. MIT Press, 2008.