Simultaneous Intermittent Communication Control and Path Optimization in Networks of Mobile Robots

Yiannis Kantaros and Michael M. Zavlanos

Abstract—In this paper, we propose an intermittent communication framework for mobile robot networks. Specifically, we consider robots that move along the edges of a connected mobility graph and communicate only when they meet at the nodes of that graph giving rise to a dynamic communication network. Our proposed distributed controllers ensure intermittent connectivity of the network and path optimization, simultaneously. We show that the intermittent connectivity requirement can be encapsulated by a global Linear Temporal Logic (LTL) formula. Then we approximately decompose it into local LTL expressions which are then assigned to the robots. To avoid conflicting robot behaviors that can occur due to this approximate decomposition, we develop a distributed conflict resolution scheme that generates non-conflicting discrete motion plans for every robot, based on the assigned local LTL expressions, whose composition satisfies the global LTL formula. By appropriately introducing delays in the execution of the generated motion plans we also show that the proposed controllers can be executed asynchronously.

I. INTRODUCTION

Communication among robots has been typically modeled using proximity graphs and the communication problem is often treated as preservation of graph connectivity [1]–[4]. Common in the above works is that point-to-point or end-to-end network connectivity is required to be preserved for all time. However, this requirement is often very conservative, since limited resources may hinder robots from accomplishing their assigned goals. Motivated by this fact, in this paper we propose a distributed intermittent communication protocol for mobile networks. In particular, we consider that robots move along the edges of a connected mobility graph and communicate only when they meet at the nodes of this graph giving rise to a dynamic communication network. We design distributed controllers that ensure intermittent communication of the network while minimizing at the same time the distance traveled between meeting points. We show that intermittent communication can be captured by a global Linear Temporal Logic (LTL) formula that forces robots to meet infinitely often at the rendezvous points. Given such an LTL expression, existing model checking techniques [5], [6] can be employed in order to implement correct by construction controllers for all robots.

LTL-based control synthesis and task specification for mobile robots build upon either a bottom-up approach when independent LTL expressions are assigned to robots [7]–[9] or top-down approaches when a global LTL describing a collaborative task is assigned to a team of robots [10], [11], as in our work. Bottom-up approaches generate a discrete high-level motion plan for all robots based on a synchronous product automaton among all agents and, therefore, they are resource demanding and scale poorly with the number of robots. To mitigate these issues, we propose a novel technique that approximately decomposes the global LTL formula into local ones and assigns them to robots. Since the approximate decomposition of the global LTL formula can result in conflicting robot behaviors we develop a distributed conflict resolution scheme that generates discrete motion plans for every robot based on the assigned local LTL expressions. By appropriately introducing delays in the execution of the generated motion plans we show the proposed controllers can also be executed in an asynchronous fashion. In contrast, most relevant literature assumes that robot control is performed in a synchronous way [10], [11]. Asynchronous robot mobility is considered in [12] by introducing ‘traveling states’ in the discretized abstraction of the environment decreasing in this way the scalability of the proposed algorithm.

The most relevant works to the one proposed here are presented in [13]–[16]. A centralized intermittent communication control scheme is presented in [13] that ensures communication among robots infinitely often but it does not scale well with the number of robots. In [14] a distributed intermittent communication control scheme is proposed that requires synchronization among robots, unlike the approach developed here. Moreover, [14] considers a priori determined communication points while in this paper, rendezvous points are selected optimally, to minimize the distance traveled by the robots. [15] proposes a distributed synchronization scheme that allows robots that move along the edges of a bipartite mobility graph to meet periodically at the vertices of this graph. Instead, here we make no assumptions on the graph structure on which robots reside or on the communication pattern to be achieved. On the other hand, [16] proposes a receding horizon framework for periodic connectivity that ensures recovery of end-to-end connectivity within a given time horizon. As the number of robots or the size of the time horizon grows, this approach can become computationally expensive. To the contrary, our proposed method scales well to large numbers of robots and can handle situations where the whole network can not be connected at once, by ensuring connectivity over time, infinitely often.
II. Problem Formulation

Consider a team of $N$ robots that move in a workspace $W \subset \mathbb{R}^n$ according to $\mathbf{x}_i(t) = \mathbf{u}_i(t)$, where $\mathbf{x}_i(t) \in \mathbb{R}^n$ is the position of robot $i$, $i \in \{1, 2, \ldots, N\}$, at time $t$ and $\mathbf{u}_i(t) \in \mathbb{R}^n$ is a control input. Also, consider $L$ locations in $W$ denoted by $\ell_j$, $j \in \{1, 2, \ldots, L\}$ located at positions $\mu_j \in W$ and paths $\gamma_{ij} : [0, 1] \rightarrow \mathbb{R}^n$ that connect two locations $\ell_j$ and $\ell_j$ such that $\gamma_{ij}(0) = \mu_i$ and $\gamma_{ij}(1) = \mu_j$.

The union of locations $\mu_i$ and paths $\gamma_{ij}$ gives rise to an undirected graph $\mathcal{G} = (V, E)$, where the set of nodes $V = \{1, 2, \ldots, L\}$ is indexed by the set of locations $\ell_j$ and the set of edges $E \subseteq V \times V$ is determined by the paths $\gamma_{ij}$ such that an edge $(i, j) \in E$ exists if and only if there is a path $\gamma_{ij}$ exists. Hereafter, we call $\mathcal{G}$ a mobility graph and we assume that the robots move along its edges to possibly accomplish a high-level task; see Figure 1(a). For example, robots may travel along their assigned paths to monitor different parts of a region and then coordinate to transmit their measurements to a user. The presence of a network allows the robots to transmit data in a multi-hop fashion, so that they do not have to leave their assigned region. Applications of this framework involve distributed coverage, state estimation, or surveillance. In what follows, we assume that the mobility graph $\mathcal{G}$ is connected.

A. Intermittent Communication

We assume that the robotic team is divided into $M$ subgroups. The indices $i$ of robots that belong to the $m$-th subgroup are collected in a set denoted by $T_m$, $m \in \{1, 2, \ldots, M\}$ while every robot can belong to more than one subgroups. Robots in a subgroup $T_m$ can communicate only when all of them are present simultaneously at a common location $\ell_j$. The locations $\ell_j$ where communication can take place for the robotic team $T_m$ are collected in a set $C_m$. This way, a dynamic robot communication graph $\mathcal{G}_c = (V_c, E_c)$ is constructed where the set of nodes $V_c$ is indexed by robots, i.e., $V_c = \{1, 2, \ldots, N\}$ and $E_c \subseteq V_c \times V_c$ is the set of communication links that emerge when, e.g., all robots in a team $T_m$ meet at a common rendezvous point $\ell_j \in C_m$ simultaneously. Given the robot teams $T_m$, a graph $\mathcal{G}_T = (V_T, E_T)$ is constructed whose set of nodes $V_T = \{1, 2, \ldots, M\}$ is indexed by the teams $T_m$ and set of edges $E_T$ consists of links between nodes $m$ and $n$ if $T_m \cap T_n \neq \emptyset$; see Figure 1(b). We assume that the graph $\mathcal{G}_T$ is connected in order to ensure dissemination of information in the network.

Furthermore, we can define the set of neighbors of node $m \in V_T$ by $\mathcal{N}_m = \{n \in V_T | \langle m, n \rangle \in E_T\}$. Also, for robot $i$ we define the set that collects the indices of all other robots that belong to common teams with robot $i$ as $\mathcal{S}_i = \{j | j \in T_m, \forall m \in \mathcal{S}_j \} \setminus \{i\}$, and the set that collects the indices of teams that robot $i$ belongs to as $\mathcal{S}_i = \{m | i \in T_m, \forall m \in \{1, 2, \ldots, M\}\}$.

At a rendezvous point $\ell_j \in C_m$ communication takes place when all robots in $T_m$ are present there, simultaneously. Then the communication graph $\mathcal{G}_c$ is defined to be connected over time if all robots in $T_m$ meet infinitely often at a region $\ell_j \in C_m$, for all $m \in V_T$. Such a requirement can be captured by the following global LTL expression:

$$\phi = \bigwedge_{m \in \{1, 2, \ldots, M\}} \left( \bigwedge_{i|m \in V_T} \left( \bigvee_{j \in \mathcal{N}_m} \mu_i \in \ell_j \wedge \bigwedge_{k \in \mathcal{S}_i} \ell_j \in C_k \wedge \bigwedge_{l \in \mathcal{S}_i} \ell_j \in C_l \right) \right),$$

(1)

where $\pi^{\ell_j}_i$ is an atomic proposition that is true when robot $i$ is sufficiently close to $\mu_i$. In (1), $\wedge$ and $\lor$ are the conjunction and disjunction operator, respectively, while $\Box$ and $\Diamond$ stand for the temporal operators 'always' and 'eventually', respectively. For more details on LTL, we refer the reader to [5], [6]. In words, the LTL expression in (1) requires all robots in a team $T_m$ to meet infinitely often at least one communication point $\ell_j \in C_m$, for all $m \in V_T$.

B. Discretized Abstraction of the Workspace

We can model the environment in which robot $i$ resides by a weighted transition system (wTS) denoted by wTS, that is defined as follows.

Definition 2.1 (weighted Transition System): A weighted Transition System $\text{wTS}$ is a tuple $(\mathcal{Q}_i, q_0^i, A_i, \rightarrow_i, w_i, AP, L_i)$ where: (a) $\mathcal{Q}_i = \{q_{ij}^i\}_{j \in \ell_j \in \mathcal{C}_m, m \in S_i}$ is the set of states, where a state $q_{ij}^i$ indicates that robot $i$ is at $\ell_j$; (b) $q_0^i \in \mathcal{Q}_i$ is the initial state of robot $i$; (c) $A_i$ is a set of actions. The available actions at state $q_{ij}^i$ are "wait" and "go to state $q_{ij}^i$" for every $k$ such that $(j, k) \in E_1; (d) \rightarrow_i \subseteq \mathcal{Q}_i \times A_i \times \mathcal{Q}_i$ is the transition relation; (e) $w_i : \mathcal{Q}_i \times \mathcal{Q}_i \rightarrow \mathbb{R}_+$ is a cost function that assigns weights/cost to each possible transition in wTS. These costs are associated with the distance between two states $q_{ij}^i$ and $q_{ij}^k$; (f) $AP$ is the set of atomic propositions; and (h) $L_i : \mathcal{Q}_i \rightarrow 2^{AP}$ is an observation/output relation giving the set of atomic propositions that are satisfied in a state.

In what follows we give definitions related to wTS, that we will use throughout the rest of the paper.


diagram

Fig. 1. A graphical illustration of the problem formulation. Figure 1(a) depicts the mobility graph $\mathcal{G}$ for a network of $N = 5$ robots (black dots) divided into $M = 5$ teams. The robot teams are selected to be: $T_1 = \{1, 2\}$, $T_2 = \{2, 3\}$, $T_3 = \{3, 4\}$, $T_4 = \{2, 4, 5\}$, and $T_5 = \{1, 5\}$. Red dots represent communication points and the black dashed lines between two sets $C_m$ and $C_n$ imply that there is at least one path $\gamma_{ij}$ that connects communication points $\ell_j \in C_m$ and $\ell_j \in C_n$. Figure 1(b) depicts the associated graph $\mathcal{G}_T$. 

1Throughout the rest of the paper, for the sake of simplicity we assume that the mobility graph $\mathcal{G}$ is constructed so that there is a path from any location $\ell_j$ to any $\ell_k$, where $q_{ij}^i \in \mathcal{Q}_i$ and $q_{ij}^k \in \mathcal{Q}_i$, that does not pass though a location $\ell_j \in C_m$, for some $m \in S_i$, for all robots $i$. This assumption can be relaxed if we incorporate all communication points $\ell_j$ in the set of states $\mathcal{Q}_i$. 

Definition 2.2 (Infinite Path): An infinite path \( \tau_i \) of \( \text{wTS}_i \) is an infinite sequence of states, \( \tau_i = \tau_i(1)\tau_i(2)\tau_i(3) \ldots \) such that \( \tau_i(1) = q_i^0 \), \( \tau_i(n) \in Q_i \), and \( (\tau_i(n), a_i, \tau_i(n + 1)) \in R_i \), for some \( a_i \in A_i \), \( \forall n^2 \).

Definition 2.3 (Composition): Composition of \( M \) infinite paths \( \tau_m = \tau_m(1)\tau_m(2)\tau_m(3) \ldots \) where \( m \in \{1, \ldots, M\} \), denoted by \( \tau = \otimes_{m=1}^M \tau_m \) is an infinite sequence of states defined as \( \tau = (\tau(1)\tau(2)\ldots = \lfloor \tau(n) \rfloor_{n=1}, \) where \( \tau(n) = (\tau(n), \tau_2(n), \ldots, \tau_M(n)) \).

Definition 2.4 (Cost): The cost incurred by the composition of \( N \) infinite paths \( \tau_i \) denoted by \( \tau = \otimes_{i=1}^N \tau_i \) is given by

\[
J(\tau) = \sum_{n=0}^{\infty} \sum_{i=1}^{N} w_i(\tau_i(n), \tau_i(n + 1)),
\]

Definition 2.5 (Projection): For an infinite path \( \tau = \tau(1)\tau(2)\tau(3) \ldots \), we denote by \( \Pi_{m \in M} \tau_m \) its projection onto \( \text{wTS}_i \), which is obtained by erasing all states in \( \tau \) that do not belong to \( Q_i \).

Definition 2.6 (Trace of infinite path): The trace of an infinite path \( \tau_i = \tau_i(1)\tau_i(2)\tau_i(3) \ldots \) of a wTS wTSi, denoted by \( \text{trace}(\tau_i) \), is an infinite word that is determined by the sequence of atomic propositions that are true in the states along \( \tau_i \), i.e., \( \text{trace}(\tau_i) = L_i(\tau_i(1))L_i(\tau_i(2)) \ldots \).

Definition 2.7 (Motion Plan): Given an LTL formula \( \phi \), a wTS wTSi, both defined over the set of atomic propositions \( AP \), an infinite path \( \tau_i \) of \( \text{wTS}_i \) is called motion plan if and only if \( \text{trace}(\tau_i) \in \text{Words}(\phi) \), where \( \text{Words}(\phi) = \{ \sigma \in (2^{AP})^\omega | \sigma \models \phi \} \) is defined as the set of words \( \sigma \in (2^{AP})^\omega \) that satisfy the LTL \( \phi \) and \( \models (2^{AP}) \times \phi \) is the satisfaction relation. The relation \( \text{trace}(\tau_i) \in \text{Words}(\phi) \) is equivalently denoted by \( \tau_i \models \phi \).

The problem we address in this paper can be stated as:

**Problem 1:** Given any initial configuration of the robots in the mobility graph \( G \) determine motion plans \( \tau_i \) for all robots \( i \) that satisfy global LTL expression given in (1), i.e., communication graph \( G \) is connected over time and minimize the total distance traveled by robots captured by the objective function (2).

III. INTERMITTENT COMMUNICATION CONTROL

To solve Problem 1, known centralized model checking techniques can be employed, that typically rely on a discretized abstraction of the environment captured by a wTS and the construction of a synchronized product system among all robots of the network. As a result, such approaches are resource demanding and scale poorly with the size of the network. Therefore, a distributed solution is preferred whereby discrete high-level motion plans for every robot can be computed locally across the network. For this purpose, notice first that although the global LTL formula (1) is not decomposable with respect to robots, it can be decomposed in local LTL formulas \( \phi_{\tau_m} \) associated with a robot team \( T_m \), which are coupled with each other by the conjunction operator \( \wedge \). Specifically, we can write \( \phi = \wedge_{m \in V_T} \phi_{\tau_m} \), where \( \phi_{\tau_m} \) is defined as

\[
\phi_{\tau_m} = \Box_\phi (\lor \ell_{\in m}(\land \forall i \in \tau_m (\pi_i^{m}))),
\]

and forces all robots \( i \) in \( T_m \) to meet infinitely often at at least one rendezvous point \( \ell_j \in C_m \).

Given the decomposition of \( \phi \) into local LTL formulas \( \phi_{\tau_m} \), every robot \( i \in T_m \) needs to develop motion plans \( \tau_i \) so that the composition of plans \( \tau_i, \forall i \in T_m \) denoted by \( \tau_m = \otimes_{i \in T_m} \tau_i \) satisfies the local LTL expression \( \phi_{\tau_m} \), for all \( m \in S_i \). In this way, we can ensure that the composition of \( \tau_i, \forall i \in \{1, 2, \ldots, N\} \), satisfies the global LTL expression (1), since all local LTL expressions \( \phi_{\tau_m} \) are satisfied.

Motion plans \( \tau_m = \phi_{\tau_m}, \forall m \in V_T \), can be constructed using existing tools from model checking theory [5], [6]. However, notice that constructing plans \( \tau_m \), \( \forall m \in V_T \), independently cannot ensure that the robots’ behavior in the workspace will satisfy the global LTL formula (1). The reason is that the local LTL formulas \( \phi_{\tau_m} \) in (3) are not independent from the local LTL expressions \( \phi_{\tau_m} \) for which it holds \( n \in N_T \), since they are coupled by robots’ state in their respective transition systems. For instance, assume that \( n \in N_T \) and that robot \( i \) belongs to teams \( T_m \) and \( T_n \). Then robot \( i \) is responsible for communicating with the other robots that belong to teams \( T_n \) and \( T_m \) at a communication point in \( C_m \) and \( C_n \), respectively. This equivalently implies that the LTL expressions \( \phi_{\tau_m} \) and \( \phi_{\tau_n} \) are coupled due to robot \( i \) through the atomic propositions \( \pi_i^{m}, \forall \ell_j \in C_m \) and \( \pi_i^{n}, \forall \ell_k \in C_n \). Consequently, generating plans \( \tau_m = \phi_{\tau_m} \) that ignore the LTL expressions \( \phi_{\tau_m} \), \( \forall n \in N_T \) may result in conflicting robot behaviors, since the projection of motion plans \( \tau_m \) and \( \tau_n \) onto \( \text{wTS}_i \) may result in two different motion plans \( \tau_i \) for a robot \( i \in T_m \cap T_n \), \( n \in N_T \). This means that cases where a robot \( i \) needs to be in more than one states in \( \text{wTS}_i \) simultaneously may occur.

To circumvent these issues, we propose a distributed algorithm in Section III-B that implements free-of-conflict discrete motion plans \( \tau_i \), for all robots \( i \), so that the global LTL expression \( \phi \) is satisfied. These motion plans will be constructed based on the prefix parts of motion plans \( \tau_m = \phi_{\tau_m} \), constructed in Section III-A, for all \( m \in S_i \).

A. Optimal Automata-based Model Checking

Given an LTL formula \( \phi_{\tau_m} \) and the wTS \( \text{wTS}_i \) of all robots \( i \in T_m \) a motion plan \( \tau_m = \phi_{\tau_m} \), can be implemented using existing automata-based model checking methods [5], [6]. First the weighted Product Transition System (wPTS) \( \text{wPTS} \) is constructed, which essentially captures all the possible combinations of robots’ states in their respective \( \text{wTS}_i \), \( \forall i \in T_m \), and is defined as follows:

**Definition 3.1 (Product Transition System):** Given \( |T_m| \) weighted Transition Systems \( \text{wTS}_{ik} = (Q_{ik}, \phi_{ik}, A_{ik}, q_{ik}, w_{ik}, AP, L_{ik}) \), where \( k = 1, 2, \ldots, |T_m| \), the weighted Product Transition System \( \text{wPTS}_{T_m} = \text{wTS}_{i_1} \otimes \text{wTS}_{i_2} \otimes \cdots \otimes \text{wTS}_{i_{|T_m|}} \) is
a tuple \((Q_{T_m}, q^0_{T_m}, A_{T_m}, \rightarrow_{T_m}, w_{T_m}, AP, L_{T_m})\) where:

(a) \(Q_{T_m} = Q_{i1} \times Q_{i2} \times \cdots \times Q_{i|S_i|}\) is the set of states;
(b) \(q^0_{T_m} = (q^0_{i1}, q^0_{i2}, \ldots, q^0_{i|S_i|}) \in Q_{T_m}\) is the initial state;
(c) \(A_{T_m} = A_{i1} \times A_{i2} \times \cdots \times A_{i|S_i|}\) is a set of actions;
(d) \(\rightarrow_{T_m} \subseteq Q_{T_m} \times A_{T_m} \times Q_{T_m}\) is the transition relation

defined by the rule\(^3\)

\[
q^m_{T_m} \xrightarrow{a_{i1} \cdots a_{i|S_i|}} q'^m_{T_m}
\]

(e) \(w_{T_m}(q_{T_m}, q'_{T_m}) = \sum_{i=1}^{m} w_i(P_{\text{wpts}}q_{i|m}, \Pi_{\text{wts}}q_{i|m}, \Pi_{\text{wts}}q_{0|m})\);
(f) \(AP\) is the set of atomic propositions; and, (h) \(L_{T_m} = \bigcup_{i \in S_i} L_{i}\) is an output/observation giving the set of atomic propositions that are satisfied at a state.

Next the LTL formula \(\phi_{T_m}\) is translated into a Non deterministic Büchi Automaton (NBA) over \(2^{AP}\) denoted by \(B_{T_m}\) [17], which is defined as follows:

**Definition 3.2: A Non deterministic Büchi Automaton (NBA) \(B_{T_m}\) over \(2^{AP}\) is defined by the tuple \(B_{T_m} = \left(Q_{B_{T_m}}, Q^0_{B_{T_m}}, 2^{AP}, \rightarrow_{B_{T_m}}, \mathcal{F}_{B_{T_m}}\right)\) where:

(a) \(Q_{B_{T_m}}\) is the set of states; (b) \(Q^0_{B_{T_m}} \subseteq Q_{B_{T_m}}\) is a set of initial states;
(c) \(\Pi = 2^{AP}\) is an alphabet; (d) \(\rightarrow_{B_{T_m}} \subseteq Q_{B_{T_m}} \times \Sigma \times Q_{B_{T_m}}\) is the transition relation; and (e) \(\mathcal{F}_{B_{T_m}} \subseteq Q_{B_{T_m}}\) is a set of accepting/final states.

Once the \(wPTS\) \(wPTS_{T_m}\) and the NBA \(B_{T_m}\) that corresponds to the LTL \(\phi_{T_m}\) are constructed, a motion plan \(\tau_{T_m} \models \phi_{T_m}\) can be found by checking the non-emptiness of the language of the Product Büchi Automaton (PBA) \(P_{T_m} = wPTS_{T_m} \otimes B_{T_m}\) [5], which is defined as follows:

**Definition 3.3 (Product Büchi Automaton):**

Given the product transition system \(wPTS_{T_m} = (Q_{wPTS_{T_m}}, Q^0_{wPTS_{T_m}}, A_{wPTS_{T_m}}, \rightarrow_{wPTS_{T_m}}, w_{T_m}, AP, L_{T_m})\) and the NBA \(B_{T_m} = (Q_{B_{T_m}}, Q^0_{B_{T_m}}, 2^{AP}, \rightarrow_{B_{T_m}}, \mathcal{F}_{B_{T_m}})\), the Product Büchi Automaton (PBA) \(P_{T_m} = wPTS_{T_m} \otimes B_{T_m}\) is a tuple \((Q_{P_{T_m}}, Q^0_{P_{T_m}}, A_{P_{T_m}}, \rightarrow_{P_{T_m}}, \mathcal{F}_{P_{T_m}})\) where:

(a) \(Q_{P_{T_m}} = Q_{wPTS_{T_m}} \times Q_{B_{T_m}}\) is the set of states; (b) \(Q^0_{P_{T_m}} = Q^0_{wPTS_{T_m}} \times Q^0_{B_{T_m}}\) is a set of initial states; (c) \(\rightarrow_{P_{T_m}} \subseteq Q_{P_{T_m}} \times A_{P_{T_m}} \times 2^{AP} \times Q_{P_{T_m}}\) is the transition relation defined

\[
q_{P_{T_m}} \xrightarrow{a_{i1} \cdots a_{i|S_i|}, X_{i1} \cdots X_{i|S_i|}} q'_{P_{T_m}}
\]

by the rule:

\[
q_{P_{T_m}} = (q_{wPTS_{T_m}}, q_{B_{T_m}}) \xrightarrow{a_{i1} \cdots a_{i|S_i|}, X_{i1} \cdots X_{i|S_i|}} (q'_{wPTS_{T_m}}, q'_{B_{T_m}})
\]

(d) \(\mathcal{F}_{P_{T_m}} = Q_{P_{T_m}} \times \mathcal{F}_{B_{T_m}}\) is a set of accepting/final states.

To check the non-emptiness of the language of \(P_{T_m}\) denoted by \(L_{P_{T_m}} = \text{trace}(wPTS_{T_m}) \cap L_{B_{T_m}}\) and to find the motion plan that both satisfies \(\phi_{T_m}\) and at the same time minimizes \(J(\tau_{T_m})\), we can employ existing model checking methods that are based on graph search algorithms; see, e.g., [18], [19]. Such motion plans can be written in a prefix-suffix structure \(\tau_{T_m} = \tau_{pre,k} \tau_{suf}\) where the prefix part \(\tau_{pre,k}\) is executed only once and the suffix part \(\tau_{suf}\) is repeated infinitely. In principle, in these approaches the PBA

\(^3\)The notation of this rule is along the lines of the notation used in [5]. In particular, it means that if the proposition above the solid line is true, then so does the proposition below the solid line.

\(^4\)The state \(q_{T_m}\) stands for the state \((q_{i1}, \ldots, q_{i|S_i|}) \in Q_{T_m}\). The state \(q_{T_m}\) is defined accordingly.

is viewed as a weighted directed graph with weights assigned on each edge that are inherited by the function \(w_{T_m}\). Then finding the shortest path from an initial state to a final state and projecting this path onto \(wPTS_{T_m}\) results in the prefix part \(\tau_{pre,k}\). The suffix part \(\tau_{suf}\) is constructed similarly by computing the shortest cycle around that final state.

**B. Conflict Resolution Coordination**

As discussed in the beginning of Section III, constructing motion plans \(\tau_{T_m}\) for all \(m \in S_i\), for all robots \(i\) can result in conflicting robot behaviors. To overcome this issue, we propose a distributed algorithm that resolves any conflicts in the robot behavior introduced by the motion plans \(\tau_{T_m}\) and constructs free-of-conflicts motion plans \(\tau_i\) for all robots \(i\) using the prefix parts \(\tau_{pre,i}\) constructed in Section III-A. The general form of these motion plans is

\[
\tau_i = \tau_i(1) \tau_i(2) \cdots \tau_i(n) = \left[ X_1 \cdots X_{\Pi_{wts}}^k \cdots X_{\Pi_{wts}}^k \right]_{i=1}^\infty = \left[ p_{i,k} \right]_{k=1}^\infty, (4)
\]

such that \(\tau_i(1) = q_0^i\). In (4), \(p_{i,k,m}^k\) is a finite path of \(wPTS_{T_m}\), where \(m_j \in S_i\) and \(j \in \{1, \ldots, |S_i|\}\). Also, \(X\) stands for a finite path in which robot \(i \) waits at its current state in \(wTS_i\). In (4), the concatenation of the paths \(X\) and \(p_{i,k,m}^k\) gives rise to the finite path \(p_{i,k}^k\). Hereafter, the \(e\)-th finite path in \(p_{i,k}^k\) is denoted by \(p_{i,k,e}^k\) where \(1 \leq e \leq \ell\), where \(\ell\) stands for the number of finite paths that appear in \(p_{i,k}^k\). The parameter \(\ell\) is a priori selected to be \(\ell = \max \{d_{T_m} M_{m=1,1} + 1\}\) for all robots, where \(d_{T_m}\) denotes the degree of vertex \(m\) in the graph \(G_T\). This particular choice for the parameter \(\ell\) ensures the construction of free-of-conflict motion plans, as it will be shown in Proposition 3.6. Moreover, we denote by \(p_i^k(n)\) the \(n\)-th state in the finite path \(p_i^k\), e.g., \(p_i^1(1) = p_i^{1,1}(1) = \tau_i(1) = q_0^i\), by construction of \(\tau_i\). The same notation extends to the infinite path \(\tau_i\).

In what follows we first describe the construction of the finite paths \(p_{i,k,m}^k\) and then we show how these finite paths are ordered in \(p_{i,k}^k\) giving rise to a free-of-conflict motion plan \(\tau_i\). First, for the finite path \(p_{i,k,m}^k\) it holds that \(p_{i,k,m}^k = \tau_{pre,k,m}^k\) where \(\tau_{pre,k,m}^k\) is the prefix part of the motion plan \(\tau_{pre,k,m}^k = \phi_{T_m}\) constructed as per Section III-A. The index \(k\) is introduced in \(\tau_{pre,k,m}^k\) to point out that the prefix part is recomputed as the index \(k\), introduced in (4) changes. The reason it needs to be recomputed is because the initial state of \(wPTS_{T_m}\) changes as \(k\) changes. Particularly, the initial state of \(wPTS_{T_m}\) is the state \(q_{0,k}^k = p_{i,k,m}^k(1)\). The state \(\Pi_{wts} p_{i,k,m}^k(1)\) is selected so that all sub-paths \(\Pi_{wts} p_{i,k,m}^k\) in \(p_{i,k}^k\) chain up consistently. Hence, we select the state \(\Pi_{wts} p_{i,k,m}^k(1)\) to be the final state of the previous sub-path that appeared in \(p_{i,k}^k\). Consequently, the state \(p_{i,k}^k(1)\) coincides with the the final state of the finite path \(p_{i,k}^k-1\).

$k = 1$, then $p^k_i(1)$ refers to the initial position of robot $i$ in the workspace.

The finite paths $p^k_i$ are constructed sequentially across the nodes $\ell_j \in V$, as follows. Let $S = \{\ell_1, \ldots, \ell_\ell\}$ be an ordered sequence of the nodes in the mobility graph $G$, so that consecutive nodes (communication points) $\ell_j, \ell_e$ in $S$ are associated with teams $T_n$ and $T_m$, respectively, that belong to neighboring nodes in the graph $G_T$, i.e., $\ell_e \in C_m, \ell_j \in C_n$ and $m \neq n$ and $m \in N_{T_n}$. We assume that $S$ is known by all robots and that every robot $i$ is initially located at the first communication point $\ell_c \in C_m$, that appears in $S$. Assume that paths have been constructed for all nodes in $S$ that precede $\ell_c \in C_m$ and that currently all robots $i \in T_m$ are located at node $\ell_c$ and coordinate to construct the paths $p^k_i$. Since the mobility graph $G$ is connected, consecutive nodes in $S$ are connected by a path in $G$, this means that there is at least one robot $j \in T_n \cap T_m$, $n \in N_{T_m}$, that previously constructed its path $p^k_j$ by placing at its $n^j_{T_m}$-th entry of the finite path $\Pi_{|wTS} p^k_j$, i.e., $p^k_j|_{n^j_{T_m}} = \Pi_{|wTS} p^k_m$. Then, robot $i$ constructs the path $p^k_i$ based on three rules. According to the first rule, the path $\Pi_{|wTS} p^k_{T_m}$ will be placed at the $n^i_{T_m}$-th entry, which is selected to be equal to $n^j_{T_m}$, which is common for all robots $j \in T_m$ [line 1, Alg. 1]. This ensures that robot $i$ and all other robots $j \in T_m$ will meet at a communication point that belongs to $C_m$, as it will be shown in Proposition 3.8. The next step is to place the paths $\Pi_{|wTS} p^k_f$, for all $m \in S \setminus \{m\}$, at the $n^m_{T_m}$-th entry of $p^k_i$. The index $n^m_{T_m}$ will be determined by one of the two following rules. Specifically, according to the second rule if there exist robots $j \in N_{T_m}$ that have already constructed the paths $p^k_j$, then the index $n^m_{T_m}$ is selected to be equal to $n^j_{T_m}$, which is common for all $j \in T_m$ [line 4, Alg. 1]. Otherwise, according to the third rule the path $\Pi_{|wTS} p^k_{T_m}$ can be placed at any free entry of $p^k_i$ indexed by $n^j_{T_m}$, provided that the $n^r_{T_m}$-th entry of all paths $p^k_j$ of robots $b \in N_j$ that have already been constructed does not contain states $\Pi_{|wTS} p^k_{T_m}$, with $h \in N_{T_m}$ [line 6, Alg. 1]. To highlight the role of this rule assume that $h \in N_{T_m}$. Then this means that there exists at least one robot $r \in T_h \cap T_g$. Then notice that without the third rule [line 5], at a subsequent iteration of this procedure, robot $r \in T_h \cap T_g$ would have to place the paths $\Pi_{|wTS} p^k_{T_m}$ and $\Pi_{|wTS} p^k_{T_g}$, at a common entry of $p^k_i$, i.e., $n^r_{T_m} = n^r_{T_g}$, due to the two previous rules and, therefore, a conflicting behavior for robot $r$ would occur. In all the remaining entries of $p^k_i$, $X$s are placed [line 8, Alg. 1]. This procedure is repeated until all robots $i \in T_m$ have constructed their respective paths $p^k_i$. Once this happens, all robots $i \in T_m$ depart from node $\ell_\ell \in C_m$ and travel to the next communication point $\ell_c \in C_v$, that appears more than once in $S$. In this case, robots $i \in T_m$ construct the finite paths $p^k_i$ only the first time that $\ell_i$ appears in $S$.

Remark 3.5 (Optimality): To construct the motion plans $\tau_i$ in (4), we first decouple the global LTL formula (1) into local LTL expressions $\phi_{T_m}$. Then, we construct finite paths $\tau_{T_m}^i = \tau_{T_m}^{iX}$, where $\tau_{T_m}^{iX} \models \phi_{T_m}$, for some $m \in S_i$, and their concatenation (Algorithm 1) gives rise to the motion plans $\tau_i$. In principle, the paths $p^k_{T_m}$ connect the current configuration of robots $i \in T_m$, to the closest meeting point $\ell_c \in C_m$, where all robots in $T_m$ meet. Therefore, while these paths minimize the total distance traveled between the current and next meeting points, they do not optimize the infinite horizon cost function (2). Consequently, the proposed solution is suboptimal. To obtain an optimal solution that minimizes (2), standard model checking techniques can be applied to the global product system, that are known to be computationally expensive. Suboptimality here is a consequence of problem decomposition.

C. Correctness of the Proposed Algorithm

In this section, we show that the composition of the discrete motion plans $\tau_i$ generated by Algorithm 1 satisfies the global LTL expression (1), i.e., that the network is connected over time. To prove this result, we need first to show that Algorithm 1 can develop non-conflicting motion plans $\tau_i$, for which we have the following two results. Proposition 3.6 can be proved by following the steps of the proof of
Algorithm 1 Construction of motion plans $\tau_i = \{p^i_k\}_{k=1}^{\infty}$ at node $\ell_e \in C_m$

Require: Already constructed finite paths $p^j_k$ of robots $j \in N_i$.

Require: All robots in $T_m$ are located at node $\ell_e \in C_m$.

1. $p^i_{k,m} := \Pi|_{wTS}, p^i_k, n^i_e = n^i_{T_m}, \forall j \in T_m$.
2. for $g \in S_i \setminus \{m\}$ do
   3. if there exist constructed paths $p^j_k$, $j \in N_i \cap T_g$ then
      4. $p^i_{k,n^i_g} := \Pi|_{wTS}, p^i_k, n^i_g = n^i_j, \forall j \in T_g$;
   else
      5. $p^i_{k,n^i_0} := \Pi|_{wTS}, p^i_k$ provided either $p^j_k = X$, or $p^i_{k,n^i_0} = \Pi|_{wTS}, p^i_k$, with $h \notin N^i_0$, $\forall j \in N_i$;
   end if
5. Put X's in the remaining entries;
9. Transmit path $p^i_k$ to a robot in $T_m$ that has not constructed its motion plan. If there are not such robots, all robots $i \in T_m$ depart from node $\ell_e \in C_m$;
10. end for

Proposition 3.2 in [14] for the graph $G_T$. Proposition 3.7 holds by construction of the finite paths $p^i_k$ and its proof, which is omitted due to space limitations, is along the lines of the proof of Proposition 3.3 in [14].

Proposition 3.6: Algorithm 1 can always construct finite paths $p^i_k$ that consist of $\ell$ finite paths $p^i_{k,e}$ where $\ell \leq \max\{d_{T,m}^M \}_{m=1}^M + 1$.

Proposition 3.7: Algorithm 1 generates admissible discrete motion plans $\tau_i$, i.e., motion plans that are free of conflicts and satisfy the transition rule $\rightarrow_i$ defined in Definition 2.1.

Proposition 3.8: The composition of motion plans $\tau_i$ generated by Algorithm 1 satisfies the global LTL expression (1), i.e., connectivity of the robot network is ensured over time, infinitely often.

Proof: To prove this result, it suffices to show that $\tau_{T_m,*} \models \neg\phi_{T_m}$, for all teams $m \in V_T$, where $\phi_{T_m}$ is defined in (3) and $\tau_{T_m,*} = \otimes \forall i \in T_m \tau_i$, where the motion plans $\tau_i$ are generated by Algorithm 1. Equivalently, according to Definition 2.7 it suffices to show that

$$ \text{trace}(\tau_{T_m,*}) \in \text{Words}(\phi_{T_m}). \quad (5) $$

First, assume that the prefix part $\tau^\text{pre}_i$ of the motion plan $\tau^i_T = \tau^\text{pre}_i \otimes \tau^\text{suf}_i \models \neg\phi_{T_m}$ has been constructed, for all $k$, using a standard model checking method as described in Section III-A.\(^6\) By construction of the motion plans $\tau_i$, it holds that there is an index $n^i_{T_m} \leq \ell$, such that $p^i_{k,n^i_{T_m}} = \Pi|_{wTS}, p^i_k$ and an index $\epsilon^i_{T_m}$, which are common for all robots $i \in T_m$, such that $p^i_{k,(\epsilon^i_{T_m})} = \Pi|_{wTS}, p^i_k(1)$ for all $i \in T_m$. Consequently, in the finite path $\tau^i_{T_m,*} = \otimes \forall i \in T_m p^i_k$, there are indices $n^i_{T_m}$ and $\epsilon^i_{T_m}$ such that $\tau^i_{k,n^i_{T_m}} = p^i_k$ and $\tau^i_{k,(\epsilon^i_{T_m})} = p^i_k(1)$ for all $k$. Hence, we conclude that all robots $i \in T_m$ will start executing the finite path $p^i_k$, simultaneously.

Using the above observation, we examine the properties of the finite word that is generated by the team $T_m$ when the finite path $p^i_k$ is executed. Let $w^i_{T_m} \in (2^{AP})^*$ be a finite word defined as $w^i_{T_m} = \text{trace}(\tau^\text{pre}_m)$ or, equivalently, by construction of the finite paths $p^i_k$, $w^i_{T_m} = \text{trace}(p^i_k)$. Since $\tau^i_{T_m} \models \phi_{T_m}$ and by construction of the prefix part $\tau^\text{pre}_m$ for all $k$, we have that the infinite word $\sigma_{T_m} = w^i_{T_m} w^i_{T_m} w^i_{T_m} \cdots \in (2^{AP})^\omega$ satisfies

$$ \sigma_{T_m} \in \text{Words}(\phi_{T_m}). \quad (6) $$

Next, notice that between the execution of the finite paths $p^i_k$ and $p^i_{k+1}$ robots $i \in T_m$ will traverse through some states of their respective wTSs that are determined by $\tau_{T_m,*}$ until the state $p^i_{k+1}$ (1) is reached. In other words, the team of robots $T_m$ does not execute consecutively the finite paths $p^i_k$ and $p^i_{k+1}$. Now, we need to show that these intermediate transitions that robots make are admissible in wPTSs, and do not violate $\phi_{T_m}$. These transitions are admissible as implied by Proposition 3.7. Also, they cannot violate $\phi_{T_m}$, since the LTL expressions $\phi_{T_m}$ in (3), for all $m \in V_T$, do not include the negation operator $\neg$ in front of the atomic propositions $\pi^i_j$ defined in Section II-A. Therefore, this equivalently means that as the robots $i \in T_m$ execute the motion plan $\tau_{T_m,*}$, they will eventually pass through all states determined by the finite paths $p^i_k$, for all $k$, without violating $\phi_{T_m}$. Consequently, this means that as the robots $i \in T_m$ move according to the motion plan $\tau_{T_m,*}$, the generated trace $\text{trace}(\tau_{T_m,*})$ will certainly include the atomic propositions that are included in $\sigma_{T_m}$; see also Definition 2.6. In other words, the sequence of atomic propositions $\sigma_{T_m}$ is a subsequence of $\text{trace}(\tau_{T_m,*})$ while the additional atomic propositions that exist in $\text{trace}(\tau_{T_m,*})$ cannot violate $\phi_{T_m}$, as previously discussed. Therefore, due to (6), we conclude that (5) holds for all $m \in V_T$, which completes the proof.

In general, the motion plans $\tau_i = [\tau_i(n)]_{n=1}^\infty$ defined in (4) are infinite paths of wTSs, since the finite paths $p^i_k$ need to be updated for every $k \in \mathbb{N}$. Therefore, in practice they are hard to implement and manipulate. In the following proposition, we show that the motion plans $\tau_i$ constructed by Algorithm 1 have a finite representation and they can be expressed in a prefix-suffix structure, where the prefix part $\tau^\text{pre}_i$ is traversed only once and the suffix part $\tau^\text{suf}_i$ is repeated infinitely.

Proposition 3.9: Algorithm 1 generates discrete motion plans $\tau_i$ for all robots $i$ in a prefix-suffix structure, i.e., $\tau_i = \tau^\text{pre}_i \otimes \tau^\text{suf}_i \models \phi_{T_m}$, where $\tau^\text{pre}_i$ is repeated and, therefore, they do not need to be recomputed. First, notice that such an index $k_s$ is common for all motion

\(^6\)Note that, throughout this proof, the only difference between the plans $\tau_{T_m,*}$ and $\tau_{T_m}$ is that the first one has been derived through composing all non-conflicting motion plans $\tau_i, \forall i \in T_m$ generated by Algorithm 1, while the second one is the plan for all robots $i \in T_m$ computed as described in Section III-A.
plans \( \tau_i \). To illustrate this point assume that there is a robot \( i \in T_m \) such that \( \Pi_{wTS} P_{T_m}^k \neq \Pi_{wTS} P_{T_m}^{k+1} \) and, therefore, \( p_i^k \neq p_i^{k+1} \). Since the paths \( P_{T_m}^k \) are computed collectively by all robots \( i \in T_m \), the fact that \( \Pi_{wTS} P_{T_m}^k \neq \Pi_{wTS} P_{T_m}^{k+1} \) means that there may be another robot \( j \in T_m \) such that \( \Pi_{wTS} P_{T_m}^k \neq \Pi_{wTS} P_{T_m}^{k+1} \) and, consequently, \( p_j^k \neq p_j^{k+1} \), which may be propagated to all other robots in the network, since graph \( G_T \) is connected. Therefore, the index \( k_s \) is common for all robots.

In what follows, we show the existence of an index \( k_s \) defined above. By definition the finite paths \( p_i^k \) are a concatenation of \( \ell \) finite paths, where the \( \epsilon \)-th finite path in \( p_i^k \) is denoted by \( p_i^{k, \epsilon} \). Also, by construction we have that \( p_i^{k, \epsilon} \neq X \) is constructed by computing a shortest path in a graph associated with a PBA \( P_{T_m} \) from an initial to a final state in \( P_{T_m} \). Notice that the set of all states \( F_{P_{T_m}} \) in \( P_{T_m} \) remain the same for all \( k \), which is not the case for the set of initial states \( Q_{P_{T_m}} = \{ q_{P_{T_m}} \} \). The reason is that the initial state \( q_{P_{T_m}} \) of wPTS may change over \( k \) and, specifically, we have that \( q_{P_{T_m}}^k = p_i^{k,1}(1) = (p_i^{k,1}(1), \ldots, p_i^{k,1}(1)) \), where the construction of state \( p_i^{k,1}(1) \) was presented in Section III-A.

Since the number of possible combinations of robots’ states in their respective wTS is finite, we have that there is a finite number of possible initial states \( q_{P_{T_m}}^k \), for all \( m \in V_T \) and, consequently, a finite number of possible initial states \( q_{i_{P_{m}}}^0 = (p_i^{0,\epsilon}(1), \ldots, p_i^{0,\epsilon}(1)) \). Therefore, for any \( e \in \{1, \ldots, \ell \} \) there are always two indices \( k_e \) and \( k_2 \), \( k_e > k_2 \) such that \( p_i^{k_e,1}(1) = p_i^{k_2,1}(1) \), for all robots \( i \), since otherwise that would mean that there is an infinite number of possible \( q_{i_{P_{m}}}^0 = (p_i^{0,\epsilon}(1), \ldots, p_i^{0,\epsilon}(1)) \) for any \( e \in \{1, \ldots, \ell \} \). Then, the finite paths \( p_i^{k,\epsilon} \), for all \( e \in \{1, 2, \ldots, \ell \} \), for all robots \( i \), and for all \( k \geq k_e \) have already been constructed at \( k \) that satisfies \( k_e^2 \leq k \leq k_2^e \); for instance, for \( k = k_2^e + 1 \) it holds that \( p_i^{k_2^e+1,\epsilon} = p_i^{k_2^e+1,\epsilon} \). Let \( e^* = \arg\min\{k_e^2 \mid e \in \{1, \ldots, \ell \} \} \), then, similarly, we have that the finite paths \( p_i^{k,\epsilon} \), for all \( e \in \{1, 2, \ldots, \ell \} \), for all robots \( i \), and for all \( k \geq k_2^e \) have already been constructed at \( k \) that satisfies \( k_e^2 \leq k \leq k_2^e \). Consequently, this means that the finite paths \( p_i^k \) for all \( k \geq k_2^e \) have already been constructed as well at \( k \) that satisfies \( k_e^2 \leq k \leq k_2^e \); for instance \( p_i^{k,\epsilon} \) and \( p_i^{k,\epsilon+1} = p_i^{k,\epsilon} \). Hence, the motion plans \( \tau_i \) can be written in a prefix-suffix structure \( \tau_i = \tau_i^{\text{prep}} \tau_i^{\text{prep}} \), where \( \tau_i^{\text{prep}} = [p_i^1 \ldots p_i^{k_2^e-1}] \) and \( \tau_i^{\text{prep}} = [p_i^{k_2^e} \ldots p_i^{k_2^e+1}] \), i.e., \( k_e = k_2^e \) and \( k_s = k_2^e \), which completes the proof.

IV. ASYNCHRONOUS INTERMITTENT COMMUNICATION

In section III, we showed that if all robots \( i \) pick synchronously their next states in wTS, according to the motion plans \( \tau_i \), then the LTL expression (1) is satisfied. In this section, we show that the generated motion plans can be executed asynchronously, as well, by appropriately introducing delays in the continuous-time execution of \( \tau_i \). We omit a formal proof of this result due to space limitations, and instead validate the proposed asynchronous scheme through numerical simulations in Section V.

Due to the asynchronous execution of the controllers, the motion plans \( \tau_i \) can be written as in (4) replacing the indices \( n \) and \( k \) with \( n_i \) and \( k_i \), respectively, which allows us to model the situation where the robots pick asynchronously their next states in wTS. In the asynchronous execution of the infinite paths \( \tau_i \) robot \( i \) moves from state \( \tau_i(n_i-1) \in Q_i \) to \( \tau_i(n_i) \in Q_i \) according to a continuous-time motion controller \( \mathbf{u}_i(t) \in \mathbb{R}^n \) that belongs to the tangent space of \( \gamma_{ij} \). Without loss of generality, assume that \( i \in T_m \) and \( \tau_i(n_i) = q_i^{e^*} \), for some \( e^* \in C_m \). When robot \( i \) arrives at state \( \tau_i(n_i) \) it checks if \( \tau_i(n_i) = \Pi_{wTS} P_{T_m}(f_i) \) for all \( j \in T_m \). If this is the case, then robot \( i \) waits at node \( e^* \) until all other robots \( j \in T_m \) are present there. When this happens, or if there is at least one robot \( j \in T_m \) such that \( \tau_j(n_j) \neq \Pi_{wTS} P_{T_m}(f_j) \), then robot \( i \) moves towards the next state \( \tau_i(n_i+1) \).

V. SIMULATION STUDIES

In this section, a simulation study is provided that illustrates our approach for a network of \( N = 5 \) robots that move along the edges of the mobility graph with \( L = 20 \) communication points as shown in Figure 2. The network is divided in \( M = 5 \) teams which are \( T_1 = \{1, 2\}, T_2 = \{2, 3\}, T_3 = \{3, 4\}, T_4 = \{2, 4, 5\}, \) and \( T_5 = \{1, 5\} \) and, therefore, the graph \( G_T \) is as shown in Figure 1(b). Also, the mobility graph is constructed so that there is a path \( \gamma_{ij} \) from any node \( e_i \) to any other node \( e_j \). Therefore, the finite paths \( p_{T_m}^k \) constructed as per Section III-A have the form \( p_{T_m}^k = q_{T_m}^{0,k} q_{T_m}^k \), where \( q_{T_m}^{0,k} \) is the initial state of wPTS constructed as defined in Section III-B and \( q_{T_m}^k = (q_{T_m}^1, \ldots, q_{T_m}^k) \) is a state where all robots of team \( T_m \) are located at a common meeting point \( e_j \in C_m \). The motion plans \( \tau_i \) generated by Algorithm 1 have the following structure:

\[
\begin{align*}
\tau_1 &= [p_{k_1}^{1}]^{k_1} = [\Pi_{wTS} T_1^{k_1} \Pi_{wTS} T_2^{k_1}, X]^{k_1} = 1, \\
\tau_2 &= [p_{k_2}^{1}]^{k_2} = [\Pi_{wTS} T_2^{k_2} \Pi_{wTS} T_3^{k_2} \Pi_{wTS} T_5^{k_2}], X]^{k_2} = 1, \\
\tau_3 &= [p_{k_3}^{1}]^{k_3} = [\Pi_{wTS} T_3^{k_3} \Pi_{wTS} T_2^{k_3} \Pi_{wTS} T_5^{k_3}], X]^{k_3} = 1, \\
\tau_4 &= [p_{k_4}^{1}]^{k_4} = [\Pi_{wTS} T_4^{k_4} \Pi_{wTS} T_3^{k_4} \Pi_{wTS} T_5^{k_4}], X]^{k_4} = 1, \\
\tau_5 &= [p_{k_5}^{1}]^{k_5} = [\Pi_{wTS} T_5^{k_5} \Pi_{wTS} T_2^{k_5} \Pi_{wTS} T_3^{k_5}], X]^{k_5} = 1,
\end{align*}
\]

which can be written in a prefix-suffix structure with \( k_s = 2 \) and \( k_p = 1 \), where the indices \( k_s \) and \( k_p \) are defined in Proposition 3.9. The motion plans \( \tau_i \) defined above are depicted in Figure 2. Notice that the distances between any two meeting points vary across \( G \) and so do the robots’ velocities and, therefore, robots pick asynchronously their next states in wTS. Consequently, this results in waiting times for every robot \( i \in T_m \) at the meeting points \( e_j \in C_m \), which are non-integer multiples of each other, for all \( m \in
In this paper we considered the problem of controlling intermittent communication in mobile robot networks. We assumed that robots move along the edges of a mobility graph and they can communicate only when they meet at its nodes, which gave rise to a dynamic communication network. The network was defined to be connected over time if communication takes place at the rendezvous points infinitely often which was captured by an LTL formula. Then this LTL expression was approximately decomposed into local LTL expressions which were assigned to robots. To avoid conflicting robot behaviors that could occur due to this approximate decomposition, we developed a distributed conflict resolution scheme that generated free-of-conflicts discrete motion plans for every robot that ensured connectivity over time, infinitely often and minimized the distance traveled by the robots. We also showed that the generated motion plans can be executed asynchronously by introducing delays in their continuous-time execution.

VI. CONCLUSION

In this paper we considered the problem of controlling intermittent communication in mobile robot networks. We assumed that robots move along the edges of a mobility graph and they can communicate only when they meet at its nodes, which gave rise to a dynamic communication network. The network was defined to be connected over time if communication takes place at the rendezvous points infinitely often which was captured by an LTL formula. Then this LTL expression was approximately decomposed into local LTL expressions which were assigned to robots. To avoid conflicting robot behaviors that could occur due to this approximate decomposition, we developed a distributed conflict resolution scheme that generated free-of-conflicts discrete motion plans for every robot that ensured connectivity over time, infinitely often and minimized the distance traveled by the robots. We also showed that the generated motion plans can be executed asynchronously by introducing delays in their continuous-time execution.

REFERENCES

