Secure-by-Construction Optimal Path Planning for Linear Temporal Logic Tasks

Shuo Yang, Xiang Yin, Shaoyuan Li and Majid Zamani

***Abstract*— In this paper, we investigate the problem of planning an optimal infinite path for a single robot to achieve a linear temporal logic (LTL) task with security guarantee. We assume that the external behavior of the robot, specified by an output function, can be accessed by a passive intruder (eavesdropper). The security constraint requires that the intruder should never infer that the robot was started from a secret location. We provide a sound and complete algorithmic procedure to solve this problem. Our approach is based on the construction of the twin weighted transition systems (twin-WTS) that tracks a pair of paths having the same observation. We show that the security-aware path planning problem can be effectively solved based on graph search techniques in the product of the twin-WTS and the Büchi automaton representing the LTL formula. The complexity of the proposed planning algorithm is polynomial in the size of the system model. Finally, we illustrate our algorithm by a simple robot planning example.**

I. Introduction

A. Motivation

Path planning is a fundamental problem in robotics which asks to generate a planned trajectory from an initial location such that some desired requirements are fulfilled. Classical planning problems usually focus on low-level tasks such as obstacle avoidance or point-to-point navigation [1], [2]. In the past decades, temporal-logic-based high-level path planning for complex tasks has drawn considerable attention in the literatures; see, e.g., [3]–[6]. In this framework, the planning task is specified by linear temporal logic (LTL) or computation tree logic (CTL) formulae. In particular, LTL can be used to represent many important properties such as safety, liveness and priority [7]. By using automata-theoretic approach, algorithmic procedures are developed to automatically generate correct-by-construction plans to achieve the given temporal logic tasks.

While the temporal-logic-based planning has been extensively investigated for safety requirements, security and privacy requirments are left as an afterthought in many applications. For instance, in robot data collecting problem, a robot needs to visit different locations in order to gather data and then to transmit collected data to the cloud. However, the data transmission may not be secure in the sense that there may exist an eavesdropper "listening" to the communication. Such information leakage may reveal some crucial secret behavior of the robot, e.g., some information, the robot does not intend to transmit, may be *inferred* by the intruder. Therefore, one also needs to incorporate such a security constraint in the path planning algorithm. Due to its importance, security and privacy concerns have been attracting attentions in the robot path planning literature; see, e.g., [8], [9].

B. Our Contributions

Motivated by the security concerns in robotic systems, in this paper, we formulate and solve a security-aware optimal path planning problem with respect to LTL requirments. Specifically, we consider a single robot whose mobility is modeled as a weighted transition system (WTS). We consider an intruder modeled as an outside observer (or eavesdropper) who accesses the external behaviors of the system specified by an output function. We consider the planning problem of achieving a task specified by a general LTL formula, while hiding the secret initial location of the robot. To capture this security requirement, we adopt the notion of an information-flow security property called *initial-state opacity* [10]. Specifically, a planed path from a secret initial-state is said to be *secure* if there exists another path from a non-secret initial-state such that those two paths are observationally equivalent from the intruder's point of view.

Our approach is different from the standard initial-state opacity verification procedure [10], which requires to build the initial-state estimator whose size is exponential in the number of system states. Instead, we propose a computationally more efficient approach by constructing the twin-WTS structure which synchronizes the system with its copy based on the observation. Similar structures have been used in the literature for the purpose of property verification, e.g., diagnosability, observability and prognosability. Here, we show that the security-aware path planning problem can be effectively solved by a graph search in the product of the twin-WTS and the Büchi automaton that accepts the given LTL task. Also, we show that the constructed product system also preserves optimality. Furthermore, our algorithm fails to provide a solution only when no solution exists. Hence, we provide a sound and complete solution to the security-aware optimal LTL planning problem.

This work was supported by the National Natural Science Foundation of China (61803259, 61833012) and by Shanghai Jiao Tong University Scientific and Technological Innovation Funds.

Shuo Yang, Xiang Yin and Shaoyuan Li are with Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China. {xiang-yang,yinxiang,syli}@sjtu.edu.cn.

M. Zamani is with the Computer Science Department, University of Colorado Boulder, CO 80309, USA. M. Zamani is also with the Computer Science Department, Ludwig Maximilian University of Munich, 80539 Munich, Germany. Email: majid.zamani@colorado.edu.

C. Related Works

Optimal LTL path planning problem was originally formulated in [11], where the optimization objective is to minimize the worst cost between each satisfying instances. This framework has been extended to the case of multi-robot [12], [13], where each robot may have a local task or a team of robots need to collaborate to achieve a global task. Recently, sampling-based techniques have been applied to improve the scalability of optimal path planning algorithm [14], [15]. Optimal temporal logic path planning problems have also been studied for stochastic systems modeled as MDPs; see, e.g., [16]–[19]. However, none of the above mentioned works considers security constraints.

In the context of security-aware path planning, our work is mostly related to [20]. The differences between our work and [20] are as follows. First, the planning task considered in our work is expressed as a general LTL formula, while [20] considers a simple reachability task. Second, no optimality is consider in [20]. Finally, the security requirement considered in our work is different with that in [20]. In particular, [20] considers protecting the *current* secret location of the robot, while we consider protecting the *initial* secret location of the robot. We show that initial-type secret has a nice property and it suffices to track a pair of observational equivalent states in the system. Therefore, the complexity of our planning algorithm is *independent* from the number of secret states and is always quadratic in the number of system states. However, the complexity of the planning algorithm in [20] is based on the structure of K-detector, whose size grows exponentially in the number of secret states.

In the computer science literature, the concept of *hyper-properties* [21] has attracted many attentions in the past years, e.g., HyperLTL [22]. In particular, hyper-properties are closely related to security requirements as it allows to specify the relationships among multiple paths. Very recently, the authors of [23] show that initial-state opacity planning problem can be specified as an instant of the HyperLTL planning problem; symbolic algorithms for finite synthesis is also provided therein. This result is closely related to ours. However, initial-state opacity considered in [23] is based on the equivalence of atomic propositions. In our setting, atomic propositions are only used to specify the desired temporal logic task, while the observation equivalence is specified by a new output function. This setting is more general as the atomic propositions and the output sets can be different. Furthermore, our planning algorithm is tailored to initial-state security, which avoids the general large complexity in HyperLTL synthesis.

Finally, our work is also related to opacity-enforcing supervisory control in the context of discrete-event systems [24]–[27]. However, the opacity-enforcing control problem is essentially a reactive synthesis problem under security constraint whose complexity is exponential in the size of the system. Here, we consider a security-aware path planning problem that can be solved more efficiently. Furthermore, no LTL specification and optimality were considered in the opacity control problem in [24]–[27].

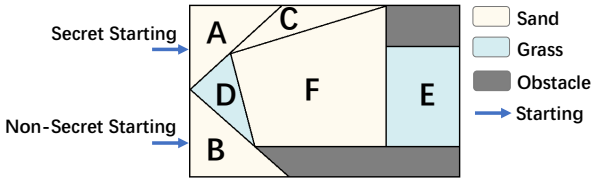

Fig. 1. Work space of the single robot.

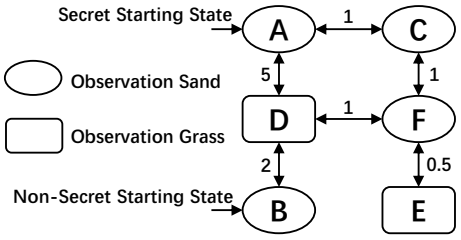

Fig. 2. The specification automaton of the motivating example. The intruder has two observations on the robot: the robot is at sand or grass land. The robot can start from A (secret) or B (non-secret). Bidirectional transition means that robot can move in both directions; numbers beside the transition represent the cost of this transition.

D. Organization

The rest of the paper is organized as follows. In Sections II and III, a motivating example and some necessary preliminaries are presented, respectively. The security-aware LTL planning problem is formally formulated in Section IV. In Section V, we discuss how to solve this problem based on the twin-WTS. A case study is presented in Section VI to illustrate the proposed algorithm. Finally, we conclude the paper by Section VII.

II. Motivating Example

Before we formally formulate the main problem in this paper, we first consider a motivating example. Suppose that a single mobile robot moves in a workspace with grass and sand lands as shown in Figure 1. The workspace is partitioned to six regions of interest and black regions denote obstacles. At each instant, the robot can only move to regions that are adjacent to its current region (sharing at least an edge). We assume that the robot always knows exactly its current location. On the other hand, we assume that there is an *outside observer* that knows whether the robot is currently at a grass or a sand land. The mobility model of the robot can be represented by the transition system shown in Figure 2. Furthermore, we assume that there is a cost moving from one region to another, which is specified by the number associated to each bidirectional transition in Figure 2.

The task of the robot is to deliver goods between regions F (representing, e.g., a factory) and E (representing, e.g., a warehouse), i.e, visit F and E infinitely often. The robot may initially start from regions A or B. However, it does not want the outside observer to know that it started from region A (if so). This may because, for example, starting

from different locations implies that different robot-types are used, which may further reveal which kind of goods the factory is delivering.

Now, suppose that the robot is starting from region A. Clearly, the optimal plan to achieve the temporal logic task is

$$A \to C \to (F \to E)^{\omega},$$

where notation ω over parentheses means the infinite repetition of the finite execution inside them. However, this plan is not secure in the sense that the observer will know for sure that the robot started from region A after observing two consecutive $Grass$. This is because there is no feasible path from region B that can generate the same observation. On the other hand, the robot may take the plan below

$$A \to D \to (F \to E)^{\omega}.$$

This plan is more expensive as the robot will incur higher cost when moving from A to D. However, this plan is secure in the sense that there exists another path

$$B \to D \to (F \to E)^{\omega},$$

starting from region B that generates the same observation. Therefore, although more cost is paid, the robot is able to hide the secret about its initial location.

III. Temporal Logic Task Planning

In this section, we define basic notations that we use in the rest of the paper and introduce some necessary preliminaries. For a set A, we denote by $|A|$ and 2^A its cardinality and its power set, respectively. A finite sequence over A is a sequence in the form of $a_1 \cdots a_n$, where $a_i \in A$; we denote by A^* the set of all finite sequences over A. Similarly, we denote by A^{ω} the set of all infinite sequences over A.

A. Weighted Transition Systems

We consider a scenario where single mobile robot works in a workspace $\mathcal{W} \subseteq \mathbb{R}^2$. The workspace is partitioned as n disjoint regions of interest denoted by $r_1, \ldots, r_n$ and we denote by $\mathcal{I} := \{1, \cdots, n\}$ the index set. In general, workspace regions can be of any arbitrary shape partitioned based on the task properties and the dynamic of the robot; see, e.g., [28], [29] for details on region partition. In this work, we focus on the task planning problem; hence, we model the mobility of the robot in the workspace as a *Weighted Transition System* (WTS) defined as follows.

Definition 1: (Weighted Transition System) A weighted transition system is a 6-tuple

$$T = (Q, Q_0, \to, w, \mathcal{AP}, L),$$

where

- $Q = \{q_i : i \in \mathcal{I}\}$ is the set of states and each state q_i indicates that the robot is at location r_i;
- $Q_0 \subseteq Q$ is the set of initial states representing all possible starting locations of the robot;
- $\to \subseteq Q \times Q$ is the transition relation such that $(q_i, q_j) \in \to$ means that there exists a controller that can drive robot from region r_i to r_j without going through any other regions;
- $w : Q \times Q \to \mathbb{R}_+$ is a cost function that assigns to each transition $(q_i, q_j) \in \to$ a positive weight $w(q_i, q_j)$ representing the cost incurred for driving the robot from region r_i to r_j, e.g., the distance between r_i and r_j;
- $\mathcal{AP}$ is the set of atomic propositions used for representing some basic properties of interest;
- $L : Q \to 2^{\mathcal{AP}}$ is the labeling function that assigns each state to a set of atomic propositions.

Given a WTS T, an *infinite internal path* is an infinite sequence of states $\tau = \tau(1)\tau(2)\tau(3)\cdots \in Q^{\omega}$ such that $\tau(1) \in Q_0$ and $(\tau(i), \tau(i+1)) \in \to, \forall i \in \mathbb{N}_+$. A *finite internal path* of a WTS is defined analogously. Hereafter, an internal path will just be referred to as path for the sake of simplicity. We denote by $\texttt{Path}^{\omega}(T)$ and $\texttt{Path}^{*}(T)$ the set of all infinite and finite paths in T, respectively. The cost function w is considered to be additive; therefore, the cost of a finite path $\tau \in \texttt{Path}^{*}(T)$, denoted by $J(\tau)$, is defined by

$$J(\tau) = \sum_{i=1}^{|\tau|-1} w(\tau(i), \tau(i+1)), \tag{1}$$

where $|\tau|$ is the length of the path. In words, the cost $J(\tau)$ captures the total cost incurred during the execution of finite path τ. The *trace* of an infinite path $\tau \in Q^{\omega}$ denoted by $\texttt{trace}(\tau)$ is an infinite sequence over $2^{\mathcal{AP}}$ such that $\texttt{trace}(\tau) = L(\tau(1))L(\tau(2))L(\tau(3))\cdots$. Given a set of states $Q' \subseteq Q$, we denote by $\texttt{Reach}(Q')$ the set of states reachable from Q'. We say a state $q \in Q$ is in a cycle of T if there exists a sequence $q_1 q_2 \ldots q_k \in Q^*$ such that $q_1 = q_k = q$ and $(q_i, q_{i+1}) \in \to, \forall i \in \mathbb{N}_+$. We denote by $\texttt{cycle}(T)$ the set of all states that are in some cycles of T.

B. Linear Temporal Logic and Büchi Automata

Let $\mathcal{AP}$ be the set of atomic propositions. A Linear Temporal Logic (LTL) formula is constructed based on atomic propositions, Boolean, and temporal operators. Specifically, an LTL formula ϕ is recursively defined by

$$\phi ::= true \mid p \mid \phi_1 \wedge \phi_2 \mid \neg\varphi \mid \bigcirc\phi \mid \phi_1 U \phi_2,$$

where $p \in \mathcal{AP}$ is an atomic proposition; $\bigcirc$ and U denote, respectively, "next" and "until". The above syntax also induces temporal operators $\Diamond$ ("eventually") and $\Box$ ("always"), where $\Diamond\phi := true U \phi$ and $\Box\phi := \neg\Diamond\neg\phi$.

LTL formulas are used to evaluate whether or not *infinite words* satisfy some properties. Formally, an infinite word $\sigma \in (2^{\mathcal{AP}})^{\omega}$ is an infinite sequence over alphabet $2^{\mathcal{AP}}$. We denote by $\sigma \models \phi$ if σ satisfies the LTL formula ϕ. For example, $\Box\Diamond\phi$ means that property ϕ should be satisfied infinitely often. The reader is referred to [7] for more details about the syntax and the semantics of LTL, which are omitted here for the sake of brevity. We define $\texttt{Words}(\phi) = \{\sigma \in (2^{\mathcal{AP}})^{\omega} : \sigma \models \phi\}$ as the set of all words satisfying LTL formula ϕ.

Definition 2: (Nondeterministic Büchi Automaton) A Nondeterministic Büchi Automaton (NBA) is a 5-tuple $B = (Q_B, Q_{0,B}, \Sigma, \to_B, F_B)$, where Q_B is the set of states,

$Q_{0,B} \subseteq Q_B$ is the set of initial states, Σ is an alphabet, $\rightarrow_B \subseteq Q_B \times \Sigma \times Q_B$ is the transition relation and $F_B \subseteq Q_B$ is the set of accepting states.

Given an infinite word $\sigma = \pi_0\pi_1\pi_2 \cdots \in \Sigma^\omega$, an infinite *run* of B over σ is an infinite sequence $\rho = q_0q_1q_2 \cdots \in Q_B^\omega$ such that $q_0 \in Q_{0,B}$ and $(q_i, \pi_i, q_{i+i}) \in \rightarrow_B$ for any $i \in \mathbb{N}$. An infinite run $\rho \in Q_B^\omega$ is said to be *accepted* by B if $\texttt{Inf}(\rho) \cap F_B \neq \emptyset$, where $\texttt{Inf}(\rho)$ denotes the set of states that appears infinite number of times in ρ. Then, an infinite word σ is said to be accepted by B if it induces an infinite run accepted by B. We denote by $\mathcal{L}_B \subseteq \Sigma^\omega$ the set of all accepted words by NBA B.

For any LTL formula ϕ, it is well-known [30] that there always exists an NBA over $\Sigma = 2^{\mathcal{AP}}$ that accepts exactly all infinite words satisfying ϕ, i.e., $\mathcal{L}_B = \texttt{Words}(\phi)$. Throughout this paper, $B = (Q_B, Q_{0,B}, 2^{\mathcal{AP}}, \rightarrow_B, F_B)$ is used to denote the NBA corresponding to the LTL formula ϕ of interest.

C. Temporal Logic Path Planning

The standard LTL path planning problem asks to find an infinite path $\tau \in \texttt{Path}^\omega(T)$ of system T such that $\texttt{trace}(\tau) \models \phi$. Due to the structure of the accepting condition in Büchi automata, it suffices to find an infinite path with the following *prefix-suffix structure*

$$\tau = q_1 \cdots q_k[q_{k+1} \cdots q_{k+m}]^\omega \in \texttt{Path}^\omega(T)$$

such that $\texttt{trace}(\tau) \in \mathcal{L}_B$. Intuitively, $q_{k+1} \cdots q_{k+m}$ is the suffix that forms a cycle such that the robot should execute infinitely often, while $q_1 \cdots q_k$ is the prefix representing the transient path that leads to the cyclic path. Such a prefix-suffix structure is also referred to a *plan*. In this work, we consider the cost of a plan, which is an infinite path, as the cost of its prefix and suffix, i.e.,

$$\hat{J}(\tau) = J(q_1 \cdots q_k q_{k+1} \cdots q_{k+m} q_{k+1}). \tag{2}$$

In order to find an optimal plan with the least cost, one can perform modified shortest path search in the *product system* composed by T and B; see, e.g., [13].

Remark 1: The cost function defined in (2) essentially treats the transient cost $J_{pre} = J(q_1 \cdots q_k q_{k+1})$ and the steady-state cost $J_{suf} = J(q_{k+1} \ldots q_{k+m} q_{k+1})$ equivalently. In general, we can define the cost function as $\hat{J}(\tau) = \alpha J_{pre} + (1-\alpha) J_{suf}$, where $\alpha \in [0, 1]$ is a parameter adjusting the weight of each part. Our work considers the case of $\alpha = 0.5$ for the sake of simplicity; all results can be easily extended to the general case.

Remark 2: Given an infinite path (plan), depending on how we decompose prefix and suffix, the plan may have different costs. For example, $q_1(q_2q_3)^\omega$ and $q_1q_2(q_3q_2)^\omega$ are the same path but have different costs. Hereafter, for a plan τ, $\hat{J}(\tau)$ is always considered as the cost for the prefix-suffix structure of τ having the minimum cost.

IV. Security-Aware Path Planning Problem

As we discussed in the motivating example, the solution to the standard LTL path planning problem does not necessarily provide security guarantees. In this section, we present the considered information-flow security model and formulate the security-aware path planning problem.

Given WTS $T = (Q, Q_0, \rightarrow, w, \mathcal{AP}, L)$, we assume that the internal state of the system is not available to the intruder (malicious observer) directly. Instead, the intruder can only infer the behavior of the system via its outputs. Formally, we model the intruder's observation of the system as an output function

$$H : Q \rightarrow Y,$$

where Y is the set of outputs. The execution of any infinite internal path $\tau = \tau(1)\tau(2)\tau(3) \cdots \in \texttt{Path}^\omega(T)$ will generate an infinite *external path* $H(\tau(1))H(\tau(2))H(\tau(3)) \cdots \in Y^\omega$; we also denote this external path by $H(\tau)$ with a slight abuse of notation. A finite external path is defined analogously.

In this work, we consider the problem of protecting *secret initial location* of the robot. To this end, we assume that $Q_S \subset Q_0$ is the set of *secret initial states*. Hereafter, a WTS T equipped with output function H and secret initial states Q_S is also written as $T = (Q, Q_0, \rightarrow, w, \mathcal{AP}, L, H, Y, Q_S)$ for simplicity. To guarantee security, we want to make sure that the intruder is not able to infer confidentially that the robot started from a secret location. This requirement is formalized as follows.

Definition 3: (Security) Let $T = (Q, Q_0, \rightarrow, w, \mathcal{AP}, L, H, Y, Q_S)$ be a WTS. An infinite path $\tau \in \texttt{Path}^\omega(T)$ is said to be *secure* if there exists an infinite path $\tau' \in \texttt{Path}^\omega(T)$ such that $\tau'(1) \notin Q_S$ and $H(\tau) = H(\tau')$.

Remark 3: The above definition of security is related to the notion of initial-state opacity proposed in [10]. Essentially, initial-state opacity is a system property such that all paths generated by the system are secure in our sense. However, as we are considering path planning problem, security is defined only for a specific path rather than the entire system.

Problem 1: (Security-Aware Optimal LTL Path Planning Problem) Given a WTS T, secret states $Q_S \subset Q$, output function $H : Q \rightarrow Y$ and LTL formula ϕ, for each possible initial-state $q_0 \in Q_0$, determine a plan $\tau \in \texttt{Path}^\omega(T)$ with $\tau(1) = q_0$ such that the following conditions hold:

1) $\texttt{trace}(\tau) \models \phi$;
2) τ is secure;
3) For any other plan $\tilde{\tau} \in \texttt{Path}^\omega(T)$ satisfying the above requirements, we have $\hat{J}(\tau) \leq \hat{J}(\tilde{\tau})$.

Remark 4: (Intruder Model) In the above problem formulation, it essentially assumes that the intruder knows the followings:

1) the mobility model of the robot, i.e., WTS T; and
2) the external path generated by the robot, i.e., $H(\tau)$.

However, it does not know the exact internal state of the robot, which has to be inferred by observing outputs. On the other hand, the robot is assumed to know exactly its initial and current state; therefore, this is still a planning problem under perfect information from the robot's point of view. This setting is reasonable in many applications because: (i) the system usually has more ability to acquire information

about itself than the intruder; and (ii) the intruder's information sometimes comes from eavesdropping the information transmission which is a partial information of the robot's knowledge.

Remark 5: According to Definition 3, if a path τ is started from a non-secret initial state $q_0 \in Q_0 \setminus Q_S$, then it is always secure as we can choose $\tau' = \tau$. Therefore, for non-secret initial states, we just need to solve the standard optimal LTL path planning problem; see, e.g., [11]. However, for those secret initial states, the security constraint has to be taken into account. This issue will be addressed in the next section.

V. Planning Algorithm

In this section, we present the security-aware path planning algorithm. Our approach is based on constructing a new transition system that effectively captures the security constraint.

A. Twin-WTS

In order to handle the security constraint, one needs to track the information of the outside observer based on the external path. Such an information-tracking task can be achieved by constructing the *initial-state estimator* [10]. However, the size of an initial-state estimator grows exponentially as the number of states in the system increases due to the subset construction.

Here, we present a computationally more efficient approach that does not rely on the construction of the initial-state estimator. Instead, we propose a new structure called the *twin-WTS*, which is used to track all current states pairs of two paths that have the same external path from the intruder's point view. This structure is formally defined as follows.

Definition 4: (Twin-WTS) Given a WTS $T = (Q, Q_0, \rightarrow, w, \mathcal{AP}, L,\ H, Y, Q_S)$, its twin-WTS is a new WTS

$$V = (X, X_0, \rightarrow_V, w_V, \mathcal{AP}, L_V),$$

where

- $X \subseteq Q \times Q$ is the set of states;
- $X_0 = \{(q_1, q_2) \in Q_0 \times Q_0 : H(q_1) = H(q_2)\}$ is the set of initial-states;
- $\rightarrow_V \subseteq X \times X$ is the transition relation defined by: for any $x = (q_1, q_2) \in X$ and $x' = (q_1', q_2') \in X$, we have $(x, x') \in \rightarrow_V$ if the followings hold:
  - $(q_1, q_1') \in \rightarrow$;
  - $(q_2, q_2') \in \rightarrow$;
  - $H(q_1') = H(q_2')$.
- $w_V : X \times X \rightarrow \mathbb{R}_+$ is the cost function defined by: for any $x = (q_1, q_2) \in X$ and $x' = (q_1', q_2') \in X$, we have $w_V(x, x') = w(q_1, q_1')$;
- $L_V : X \rightarrow 2^{\mathcal{AP}}$ is the labeling function defined by: for any $x = (q_1, q_2) \in X$, we have $L_V(x) = L(q_1)$.

Remark 6: Intuitively, the twin-WTS tracks two internal paths that generate the same external path. Specifically, the first component is used to represent the trajectory in the real system, while the second component is used to represent a copy that mimics the real system in the sense of output equivalence. Therefore, for any path $(\tau_1(1), \tau_2(1))(\tau_1(2), \tau_2(2)) \cdots$ in V, we have $H(\tau_1(1))H(\tau_1(2)) \cdots = H(\tau_2(1))H(\tau_2(2)) \cdots$. On the other hand, for any two paths τ_1, τ_2 in T such that $H(\tau_1) = H(\tau_2)$, we can find a path τ in V such that its first component is τ_1 and the second component is τ_2. Also, we note that the cost function w_V and the labeling function L_V are all defined based on the states in the first component, which is the part for the real system. Finally, the size of V is polynomial in the size of T as it contains at most $|Q|^2$ states.

B. Planning Algorithm

The twin-WTS can be used to capture the security constraint based on the following observation. For any secure path starting from a secret initial state $q_{s,0}$, there must exist an observation-equivalent path from a non-secret initial state $q_{ns,0}$. Furthermore, such a path-pair should exist in the twin-WTS V from state $(q_{s,0}, q_{ns,0})$. Therefore, to perform security-aware path planing, it suffices to perform planning from an initial-state in V in which the first component is the real (secret) initial-state and the second component is a non-secret state. Furthermore, in order to incorporate the temporal task, we need to synchronize the twin-WTS with the NBA B that accepts ϕ; this is defined as the product system.

Definition 5: (Product System) Given twin-WTS $V = (X, X_0, \rightarrow_V, w_V, \mathcal{AP}, L)$ and NBA $B = (Q_B, Q_{0,B}, \Sigma, \rightarrow_B, F_B)$, the product of V and B is a new (unlabeled) WTS

$$T_\otimes = (Q_\otimes, Q_{0,\otimes}, \rightarrow_\otimes, w_\otimes),$$

where

- $Q_\otimes \subseteq X \times Q_B$ is the set of states;
- $Q_{0,\otimes} = X_0 \times Q_{0,B}$ is the set of initial states;
- $\rightarrow_\otimes \subseteq Q_\otimes \times Q_\otimes$ is the transition relation defined by: for any $q_\otimes = (x, q_B) \in Q_\otimes$ and $q_\otimes' = (x', q_B') \in Q_\otimes$, we have $(q_\otimes, q_\otimes') \in \rightarrow_\otimes$ if the followings hold:
  - $(x, x') \in \rightarrow_V$; and
  - $(q_B, L_V(x), q_B') \in \rightarrow_B$.
- $w_\otimes : Q_\otimes \times Q_\otimes \rightarrow \mathbb{R}_+$ is the cost function defined by: for any $q_\otimes = (x, q_B) \in Q_\otimes$ and $q_\otimes' = (x', q_B') \in Q_\otimes$, we have $w_\otimes(q_\otimes, q_\otimes') = w(x, x')$.

Essentially, the product system further restricts the dynamic of V such that each movement should satisfy the LTL task ϕ, i.e., $(q_B, L_V(x), q_B') \in \rightarrow_B$. Note that the original WTS T is not synchronized with B as the dynamic of T has already been encoded in the first component of V. For each state $((q, q'), q_B) \in Q_\otimes$, we denote by $\Pi[((q, q'), q_B)] = q$ the projection to the state space of T; we also write $\Pi[((q_0, q_0'), q_{0,B}) \cdots ((q_n, q_n'), q_{n,B})] = q_0 \cdots q_n$.

For each initial-state $q_0 \in Q_0$ in T, we denote by $\textsc{Int}_{q_0}(T_\otimes) \subseteq Q_{0,\otimes}$ the set of initial-states in $T_\otimes$ whose first components are q_0 while the second component are non-secret states in T, i.e.,

$$\textsc{Int}_{q_0}(T_\otimes) = \{((q_0, q_0'), q_B) \in Q_{0,\otimes} : q_0' \notin Q_S\}.$$

Also, we define $\textsc{Goal}(T_\otimes) \subseteq Q_\otimes$ as the set of states in $T_\otimes$ whose last components are in F_B and they are in some

cycles of $T_\otimes$, i.e.,

$$\textsc{Goal}(T_\otimes) = \{((q,q'),q_B) \in Q_\otimes : q_B \in F_B \wedge ((q,q'),q_B) \in \texttt{cycle}(T_\otimes)\}.$$

In order to find an optimal path from initial state q_0 in T, it suffices to find an optimal path in the form of

$$\textsc{Int}_{q_0}(T_\otimes) \rightarrow (\textsc{Goal}(T_\otimes) \rightarrow \textsc{Goal}(T_\otimes))^\omega$$

in $T_\otimes$. Note that both sets $\textsc{Int}_{q_0}(T_\otimes)$ and $\textsc{Goal}(T_\otimes)$ are non-singleton in general. Therefore, we need to consider all possible combinations in order to determine an optimal path. This idea is formalized by Algorithm 1.

Specifically, lines 1-3 construct the NBA B, the twin-WTS V and the product system $T_\otimes$. Line 4 aims to determine if there is a feasible path from q_0 satisfying both the LTL constraint and the security constraint. In particular, if $\textsc{Int}_{q_0}(T_\otimes)$ cannot reach any goal state in cycle, then this means that there does not exist an infinite path accepted by ϕ that has an observation equivalent path from a non-secret initial state, i.e., there exists no feasible path starting from q_0. Otherwise, we consider, in lines 7 and 8, each combination of state q_I in $\textsc{Int}_{q_0}(T_\otimes)$ and state q_G in $\texttt{Reach}(\{q_I\}) \cap \textsc{Goal}(T_\otimes)$, which is a goal state reachable from q_1 and in some cycles. In lines 9-10, we determine the shortest path from q_I to q_G and the shortest path from q_G back to itself; the projection onto T by Π then gives us an infinite path satisfying both the LTL and the security constraint. Then among all such feasible combinations, we determine the optimal pair (q_I^*, q_G^*) that minimizes the path cost function defined in (2) and the optimal plan $\tau = \tau^{q_I^*,q_G^*}[\tau^{q_G^*,q_G^*}]^\omega$ is returned.

Algorithm 1: Security-Aware Optimal LTL Plan

input : LTL formula ϕ, WTS T with H and Q_S, initial state q_0
output: Optimal plan τ from $q_0 \in Q_0$

1 Convert ϕ to NBA $B = (Q_B, Q_{0,B}, \Sigma, \rightarrow_B, F_B)$;
2 Construct twin-WTS $V = (X, X_0, \rightarrow_V, w_V, \mathcal{AP}, L_V)$;
3 Construct the product of V and B $T_\otimes = (Q_\otimes, Q_{0,\otimes}, \rightarrow_\otimes, w_\otimes)$;
4 **if** $\texttt{Reach}(\textsc{Int}_{q_0}(T_\otimes)) \cap \textsc{Goal}(T_\otimes) = \emptyset$ **then**
5 **return** "no feasible plan from q_0";
6 **else**
7 **for** $q_I \in \textsc{Int}_{q_0}(T_\otimes)$ **do**
8 **for** $q_G \in \texttt{Reach}(\{q_I\}) \cap \textsc{Goal}(T_\otimes)$ **do**
9 $\tau^{q_I,q_G} = \Pi[\texttt{Shortpath}(q_I, q_G)]$;
10 $\tau^{q_G,q_G} = \Pi[\texttt{Shortpath}(q_G, q_G)]$;
11 **end**
12 **end**
13 $(q_I^*, q_G^*) = \arg\min_{(q_I,q_G)} \hat{J}(\tau^{q_I,q_G}[\tau^{q_G,q_G}]^\omega)$;
14 **return** optimal plan $\tau = \tau^{q_I^*,q_G^*}[\tau^{q_G^*,q_G^*}]^\omega$ for q_0;
15 **end**

Remark 7: Let us discuss the complexity of Algorithm 1. First, we note that the product system $T_\otimes$ contains at most $|Q|^2|Q_B|$ states, where $|Q|$ is the number of states in the WTS model and $|Q_B|$ is the number of states in the Büchi automaton. Algorithm 1 involves at most $|Q|^4|Q_B|^2$ (very roughly estimated) shortest path problems which can be solved in polynomial-times in the number of states in $T_\otimes$. Therefore, the overall planning complexity is polynomial in both the number of states in the plant and the number of states in the Büchi automaton. Note that, in general, $|Q_B|$ is the length of ϕ. However, in practice, the size of the LTL formula ϕ is usually very small and Q, which represents the state-space, is usually the main factor for scalability.

C. Correctness of the Planning Algorithm

Now, we prove the correctness of the proposed planning algorithm. Hereafter, we assume that the robot is starting from initial state q_0 and τ is the optimal plan from q_0 returned by Algorithm 1. First, we show that the resulting plan satisfies the LTL task ϕ.

Proposition 1: $\texttt{trace}(\tau) \models \phi$.

Proof: We assume that optimal path is obtained from the following projection $\tau = \Pi[p^{pre}(p^{suf})^\omega]$, where

$$p^{pre} = ((q_0, q_0'), q_{0,B}) \cdots ((q_n, q_n'), q_{n,B})$$
$$p^{suf} = ((q_{n+1}, q_{n+1}'), q_{n+1,B}) \cdots ((q_{n+m}, q_{n+m}'), q_{n+m,B})$$

and $q_{n+1,B} \in F_B$. That is, $\tau = q_0 \cdots q_n (q_{n+1} \cdots q_{n+m})^\omega$. According to the transition rule of $T_\otimes$, $\rho = q_{0,B} \cdots q_{n,B}(q_{n+1,B} \cdots q_{n+m,B})^\omega$ is an infinite run induced by infinite word $\texttt{trace}(\tau) = L(q_0) \cdots L(q_n)(L(q_{n+1}) \cdots L(q_{n+m}))^\omega$. Since $q_{n+1,B} \in F_B$, we know that $\texttt{Inf}(\rho) \cap F_B \neq \emptyset$, which means $\texttt{trace}(\tau) \in \mathcal{L}_B = \texttt{Word}(\phi)$, i.e., $\texttt{trace}(\tau) \models \phi$. ■

Second, we show that the planned path is secure.

Proposition 2: τ is secure.

Proof: Without loss of generality, we assume that $q_0 \in Q_S$; otherwise, τ is secure trivially. Still, we assume that the optimal path is obtained by $\tau = \Pi[p^{pre}(p^{suf})^\omega]$, where

$$p^{pre} = ((q_0, q_0'), q_{0,B}) \cdots ((q_n, q_n'), q_{n,B})$$
$$p^{suf} = ((q_{n+1}, q_{n+1}'), q_{n+1,B}) \cdots ((q_{n+m}, q_{n+m}'), q_{n+m,B}).$$

Then we know that $(q_0, q_0') \cdots (q_n, q_n')((q_{n+1}, q_{n+1}') \cdots (q_{n+m}, q_{n+m}'))^\omega \in \texttt{Path}^\omega(V)$. According to the transition rule of V, we have $H(\tau) = H(q_0' \cdots q_n'(q_{n+1}' \cdots q_{n+m}')^\omega)$. Finally, since $((q_0, q_0'), q_{0,B}) \in \textsc{Int}_{q_0}(T_\otimes)$, we know that $q_0' \notin Q_S$. Therefore, $\tau' = q_0' \cdots q_n'(q_{n+1}' \cdots q_{n+m}')^\omega$ is an internal path from a non-secret initial state having the same observation with τ, i.e., τ is secure. ■

Finally, we show that the planned path is optimal.

Proposition 3: For any other secure path $\tilde{\tau} = \tilde{\tau}^{pre}[\tilde{\tau}^{suf}]^\omega$ such that $\texttt{trace}(\tilde{\tau}) \models \phi$, we have $\hat{J}(\tau) \leq \hat{J}(\tilde{\tau})$.

Proof: We prove by contradiction. Suppose that there exists a secure path $\tilde{\tau} \in \texttt{Path}^\omega(T)$ such that $\texttt{trace}(\tilde{\tau}) \models \phi$ and $\hat{J}(\tilde{\tau}) < \hat{J}(\tau)$. Since $\tilde{\tau}$ is secure, we know that there exists another path $\tilde{\tau}' \in \texttt{Path}^\omega(T)$ such that $H(\tilde{\tau}) = H(\tilde{\tau}')$ and $\tilde{\tau}'(1) \notin Q_S$. According to the definition of V, we know that there exists a path $\tau_V \in \texttt{Path}^\omega(V)$ in which the first component is $\tilde{\tau}$ and the second component is $\tilde{\tau}'$.

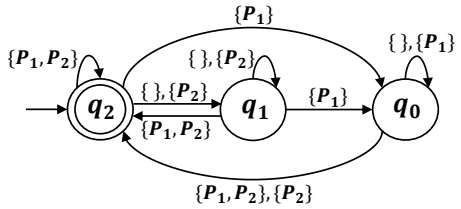

Fig. 3. An NBA translated from $\phi = \square\lozenge P_1 \wedge \square\lozenge P_2$.

Furthermore, since $\texttt{trace}(\tilde{\tau}) \models \phi$, by the definition of $T_\otimes$, there exists a path $\tau_\otimes \in \texttt{Path}^\omega(T_\otimes)$ in which the first component is τ_V and the second component is τ_B such that $\texttt{Inf}(\tau_B) \cap F_B \neq \emptyset$. Without loss of generality, we write $\tau_\otimes = p^{pre}(p^{suf})^\omega$ in the prefix-suffix structure, where

$$p^{pre} = ((q_0, q_0'), q_{0,B}) \cdots ((q_n, q_n'), q_{n,B})$$
$$p^{suf} = ((q_{n+1}, q_{n+1}'), q_{n+1,B}) \cdots ((q_{n+m}, q_{n+m}'), q_{n+m,B})$$

and $q_{n+1,B} \in F_B$. Since $q_0' = \tilde{\tau}'(1) \notin Q_S$, we know that $\tilde{q}_I := ((q_0, q_0'), q_{0,B}) \in \textsc{Int}_{q_0}(T_\otimes)$. Furthermore, we have $\tilde{q}_G := ((q_{n+1}, q_{n+1}'), q_{n+1,B}) \in \texttt{Reach}(\{\tilde{q}_I\}) \cap \textsc{Goal}(T_\otimes)$. However, $\hat{J}(\tau^{\tilde{q},\tilde{q}_G}[\tau^{\tilde{q}_G,\tilde{q}_G}]^\omega) = \hat{J}(\tilde{\tau}) < \hat{J}(\tau)$. This means that Algorithm 1 should at least output $\tau^{\tilde{q},\tilde{q}_G}[\tau^{\tilde{q}_G,\tilde{q}_G}]^\omega$ rather than τ, which is a contradiction. ■

The above three propositions show that the proposed algorithm is sound in the sense that the solution is correct if it finds one. Note that Algorithm 1 may return "no feasible plan from q_0". Next we show that the proposed algorithm is also complete.

Proposition 4: If Algorithm 1 returns "no feasible plan from q_0", then no solution to Problem 1 exists.

Proof: The proof is similar to the proof of Proposition 3. Suppose, for the sake of contraposition, that there exists a secure path $\tau \in \texttt{Path}^\omega(T)$ such that $\texttt{trace}(\tau) \models \phi$. Following the same argument in the proof of Proposition 3, there exists a path $\tau_\otimes \in \texttt{Path}^\omega(T_\otimes)$, which is in the form of $\tau_\otimes = ((\tau, \tau'), \tau_B)$ such that $\tau'(1) \notin Q_S$ and $\texttt{Inf}(\tau_B) \cap F_B \neq \emptyset$. Therefore, we have $\tilde{q}_I := ((\tau(1), \tau'(1)), \tau_B(1)) \in \textsc{Int}_{q_0}(T_\otimes)$. Furthermore, $\texttt{Reach}(\{\tilde{q}_I\}) \cap \textsc{Goal}(T_\otimes) \neq \emptyset$ since $\texttt{Inf}(\tau_B) \cap F_B \neq \emptyset$. Therefore, Algorithm 1 will not return "no feasible plan from q_0". ■

Finally, we summarize Propositions 1, 2, 3 and 4 by the following theorem.

Theorem 1: For any WTS $T = (Q, Q_0, \to, w, \mathcal{AP}, L)$ with output function $H : Q \to Y$, secret states Q_S and LTL formula ϕ, Algorithm 1 correctly solves the optimal security-aware LTL planning problem defined in Problem 1.

Proof: The soundness of the algorithm is established by Propositions 1, 2 and 3, and Proposition 4 shows the completeness of the algorithm. ■

VI. Case Study

We go back to the motivating example in Section II to illustrate the proposed planning algorithm. Consider again the WTS in Figure 2. To formalize the LTL task, we consider two atomic propositions $\mathcal{AP} = \{P_1, P_2\}$ with labeling function $L : Q \to 2^{\mathcal{AP}}$ defined by $L(F) = \{P_1\}$, $L(E) = \{P_2\}$ and $L(q) = \emptyset$ for other states. Then the task of the robot is expressed by the LTL formula

$$\phi = \square\lozenge P_1 \wedge \square\lozenge P_2.$$

The observation mapping is $H : Q \to \{Grass, Sand\}$ as specified in Figure 2. We define $Q_S = \{A\} \subseteq Q$, i.e., state A is the unique secret initial state.

To achieve the planning task, first we convert ϕ to NBA $B = (Q_B, Q_{0,B}, \Sigma, \to_B, F_B)$, which is shown in Figure 3; such a conversion can be done by, e.g., the tool developed in [31]. Then we construct the corresponding twin-WTS V, which is shown in Figure 4. Specifically, V contains four initial states $(A, A), (B, B), (A, B)$ and (B, A) since $H(A) = H(B) = Sand$ and four combination are all valid initial states. Then, for example, starting from (A, B), only state (D, D) can be reached as $A \to D, B \to D$ and $H(D) = H(D) = Grass$. Also, from state (C, C), we can reach (A, F) as $C \to A, C \to F$ and $H(A) = H(F) = Sand$. Finally, we need to construct the product system $T_\otimes$; for the sake of simplicity, we just show part of $T_\otimes$ in Figure 5, which is sufficient for the purpose of planning.

Now, we assume that the robot is starting from secret initial state A. Then we have $\textsc{Int}_A(T_\otimes) = \{((A, B), q_2)\}$, which is a singleton. Also, we have $((F, F), q_2) \in \texttt{Reach}(\{((A, B), q_2)\}) \cap \textsc{Goal}(T_\otimes)$. One can check that such a state pair is indeed the one that

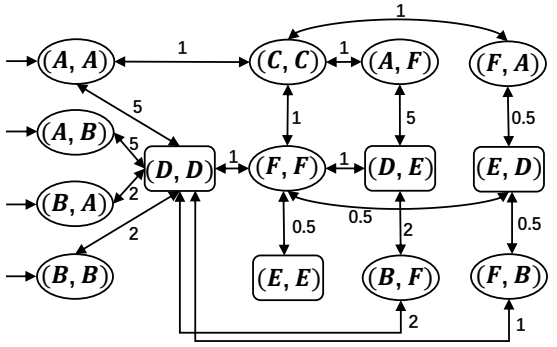

Fig. 4. Twin-WTS V of T in Figure 2.

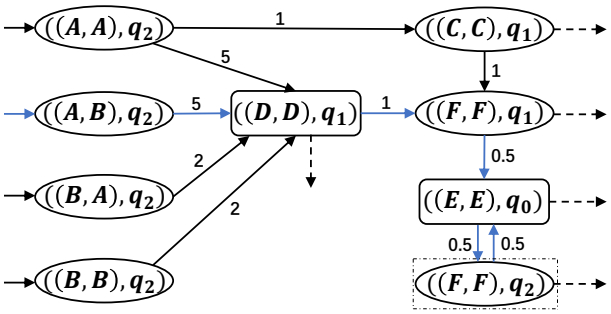

Fig. 5. Example of the construction of the $T_\otimes$. Red transitions represent the optimal feasible path. Due to limited space, some states and transitions are omitted and part of the product system is shown.

minimizes the cost function if we draw the complete product system. Therefore, we obtain an optimal plan $\tau = \Pi[((A,B),q_2)((D,D),q_1)((F,F),q_1)(((E,E),q_0)((F,F),q_2))^\omega] = AD(FE)^\omega$, which is highlighted by red transitions in Figure 5.

VII. Conclusion

In this paper, we solved a security-aware optimal path planning problem for linear temporal logic tasks. A polynomial-time algorithm was proposed based on the product of the twin-system and the Büchi automaton. The synthesized solution is *secure-by-construction* in the sense that it provides provably security guarantees for the designed systems against temporal logic tasks. Note that, in this work, we consider security requirement for protecting the initial secret location of the system. In the future, we would like to extend the proposed algorithm to other types of security, e.g., infinite-step opacity. Also, we are interested in investigating optimal LTL path planning for multi-robot systems with security guarantees.

References

[1] S. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[2] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[3] G. Fainekos, A. Girard, H. Kress-Gazit, and G. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.

[4] T. Wongpiromsarn, U. Topcu, and R. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2817–2830, 2012.

[5] H. Kress-Gazit, M. Lahijanian, and V. Raman, "Synthesis for robots: Guarantees and feedback for robot behavior," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 211–236, 2018.

[6] M. Kloetzer and C. Mahulea, "Path planning for robotic teams based on ltl specifications and petri net models," *Discrete Event Dynamic Systems*, vol. 30, no. 1, pp. 55–79, 2020.

[7] C. Baier and J. Katoen, *Principles of model checking*. MIT press, 2008.

[8] L. Li, A. Bayuelo, L. Bobadilla, T. Alam, and D. Shell, "Coordinated multi-robot planning while preserving individual privacy," in *International Conference on Robotics and Automation (ICRA)*, pp. 2188–2194, 2019.

[9] H. Zheng, J. Panerati, G. Beltrame, and A. Prorok, "An adversarial approach to private flocking in mobile robot teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1009–1016, 2020.

[10] A. Saboori and C. Hadjicostis, "Verification of initial-state opacity in security applications of discrete event systems," *Information Sciences*, vol. 246, pp. 115–132, 2013.

[11] S. Smith, J. Tumova, C. Belta, and D. Rus, "Optimal path planning for surveillance with temporal-logic constraints," *The International Journal of Robotics Research*, vol. 30, no. 14, pp. 1695–1708, 2011.

[12] 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.

[13] A. Ulusoy, S. Smith, X. Ding, C. Belta, and D. Rus, "Optimality and robustness in multi-robot path planning with temporal logic constraints," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 889–911, 2013.

[14] L. Li and J. Fu, "Sampling-based approximate optimal temporal logic planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1328–1335, 2017.

[15] Y. Kantaros and M. Zavlanos, "Sampling-based optimal control synthesis for multirobot systems under global temporal tasks," *IEEE Transactions on Automatic Control*, vol. 64, no. 5, pp. 1916–1931, 2018.

[16] E. Wolff, U. Topcu, and R. Murray, "Robust control of uncertain markov decision processes with temporal logic specifications," in *51st IEEE Conference on Decision and Control (CDC)*, pp. 3372–3379, 2012.

[17] X. Ding, S. Smith, C. Belta, and D. Rus, "Optimal control of Markov decision processes with linear temporal logic constraints," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1244–1257, 2014.

[18] K. Deng, Y. Chen, and C. Belta, "An approximate dynamic programming approach to multiagent persistent monitoring in stochastic environments with temporal logic constraints," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4549–4563, 2017.

[19] M. Guo and M. Zavlanos, "Probabilistic motion planning under temporal tasks and soft constraints," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4051–4066, 2018.

[20] C. Hadjicostis, "Trajectory planning under current-state opacity constraints," in *14th IFAC Workshop on Discrete Event Systems (WODES)*, pp. 337–342, 2018.

[21] M. Clarkson and F. Schneider, "Hyperproperties," *Journal of Computer Security*, vol. 18, no. 6, pp. 1157–1210, 2010.

[22] M. Clarkson, B. Finkbeiner, M. Koleini, K. Micinski, M. Rabe, and C. Sánchez, "Temporal logics for hyperproperties," in *International Conference on Principles of Security and Trust*, pp. 265–284, 2014.

[23] Y. Wang, S. Nalluri, and M. Pajic, "Hyperproperties for robotics: Motion planning via HyperLTL," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. arXiv:1911.11870.

[24] J. Dubreil, P. Darondeau, and H. Marchand, "Supervisory control for opacity," *IEEE Trans. Automatic Control*, vol. 55, no. 5, pp. 1089–1100, 2010.

[25] A. Saboori and C. Hadjicostis, "Opacity-enforcing supervisory strategies via state estimator constructions," *IEEE Trans. Automatic Control*, vol. 57, no. 5, pp. 1155–1165, 2011.

[26] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Trans. Automatic Control*, vol. 61, no. 8, pp. 2140–2154, 2016.

[27] Y. Tong, Z. Li, C. Seatzu, and A. Giua, "Current-state opacity enforcement in discrete event systems under incomparable observations," *Discrete Event Dynamic Systems: Theory & Appllications*, vol. 28, no. 2, pp. 161–182, 2018.

[28] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 320–330, 2007.

[29] C. Belta, B. Yordanov, and E. Gol, *Formal methods for discrete-time dynamical systems*, vol. 89. Springer, 2017.

[30] M. Vardi and P. Wolper, "An automata-theoretic approach to automatic program verification," in *Proceedings of the First Symposium on Logic in Computer Science*, pp. 322–331, 1986.

[31] P. Gastin and D. Oddoux, "Fast LTL to Büchi automata translation," in *International Conference on Computer Aided Verification (CAV)*, pp. 53–65, Springer, 2001.