Necessary and Sufficient Conditions for Satisfying Linear Temporal Logic Constraints using Control Barrier Certificates

Luyao Niu[1], Andrew Clark[2], and Radha Poovendran[1]

***Abstract*— Temporal logic specifications have been used to express complex tasks for control systems. Discretization-free approaches, which do not require discretizing the state and input spaces of the system, have been proposed for control synthesis under temporal logic specifications. Among these approaches, control barrier certificate (CBC)-based solutions have attracted increasing attention. The existing CBC-based approaches, however, have no guarantee on always finding control laws to satisfy the specification, and hence are sound but not complete. In this paper, we derive the necessary and sufficient conditions for a control law to satisfy a temporal logic specification over finite traces using CBCs. By leveraging the equivalence between satisfying the specification and violating the negated specification, we first negate the specification and construct the deterministic finite automaton (DFA) as a representation. We then decompose the DFA into a set of safety problems, where each decomposed problem corresponds to a transition in the DFA. We derive the necessary and sufficient conditions for a control law to solve each safety problem via CBC-based approach. We further develop the necessary and sufficient conditions to verify whether the control laws associated with different safety problems are composable or not. The composability captures whether a sequence of transitions in the DFA can be realized by the system or not. If the set of composable control laws does not render an accepting run on the DFA, then the system can satisfy the specification. We illustrate the proposed approach using a numerical case study on a multi-agent system.**

I. Introduction

Temporal logics [1] have been widely used to specify complex tasks across various application domains including robotics [2], [3] and traffic network control [4]. As a consequence, verification and control synthesis under temporal logic properties have gained increasing attention.

Verification and control synthesis under temporal logic constraints can be achieved using off-the-shelf model checking algorithms [1]. These algorithms construct a finite abstraction via discretization to model the original system [2], [5]–[8]. When the discretization granularity is sufficiently small, (approximately) equivalent abstractions can be generated, rendering discretization-based control synthesis to be sound and complete [9], at the expenses of intensive computational complexity. Recently, researchers have proposed compositional abstraction of large-scale systems to mitigate the curse of dimensionality, including small-gain type conditions [10], dissipativity approaches [11], and dynamic Bayesian networks [12].

An alternative way to mitigate the computational challenge incurred by the discretization-based approaches is through discretization-free approaches [13]–[16], which focus on the continuous state space without discretization. Recently, control barrier certificates (CBCs) and control barrier functions (CBFs) have been used to satisfy temporal logic properties [13]–[20]. CBC- and CBF-based approaches decompose the temporal logic specification into a sequence of safety and/or reachability problems, where each decomposed problem corresponds to a transition in the automaton representing the temporal logic formula. These approaches derive a control law to satisfy each decomposed specification. The aforementioned CBC- and CBF-based approaches, however, only provide sufficient conditions without any completeness guarantee. Consequently, there may exist control laws that allow the system to satisfy the given temporal logic specification, but cannot be found by the CBC- and CBF-based approaches. One reason of such incompleteness is that the existing CBC- and CBF-based approaches treat each transition in the automaton independently. However, such an independence assumption is not valid due to the underlying system dynamics [21], [22]. At present, necessary and sufficient conditions for discretization-free approaches to satisfy a temporal logic specification have been less studied.

In this paper, we consider a continuous-time control-affine system subject to a linear temporal logic (LTL) specification. We consider a fragment of LTL formula, namely LTL over finite traces. Our objective is to derive the necessary and sufficient conditions for a CBC-based approach to satisfy the given LTL specification. We observe that satisfying the specification is equivalent to violating its negation. We then negate the specification and construct an equivalent finite automaton to represent the negated specification. We construct a set of safety verification problems, with each problem corresponding to a transition in the automaton. We then solve each decomposed safety verification problem using CBCs. We finally develop a set of conditions to compose the result from each individual safety verification problem. We make the following contributions in this paper.

- We derive the necessary and sufficient conditions for a control law to guarantee the satisfaction of each decomposed safety verification problem using CBCs. We propose a labeling procedure to label each transition in the automaton to indicate the result of the corresponding

[1]Luyao Niu and Radha Poovendran are with the Network Security Lab, Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195-2500 {luyaoniu,rp3}@uw.edu

[2]Andrew Clark is with the Electrical and Systems Engineering Department, McKelvey School of Engineering, Washington University in St. Louis, St. Louis, MO 63130 andrewclark@wustl.edu

This work was supported by the Office of Naval Research grant N00014-20-1-2636, National Science Foundation grant CNS 1941670, and Air Force Office of Scientific Research grants FA9550-20-1-0074 and FA9550-22-1-0054

safety verification problem.

- We explicitly consider the dependencies among transition by investigating the composability of the control laws associated with the safety verification problems. We derive the necessary and sufficient conditions for the control laws to be composable. The composability result together with the safety verification result yield the necessary and sufficient conditions for falsifying the existence of an accepting run on the automaton, and hence the satisfaction of the given specification.
- We demonstrate the proposed approach using a multi-agent system. We show that the controller obtained using the derived conditions guarantees the agents to satisfy the given specification.

The remainder of this paper is organized as follows. Section II presents preliminary background on LTL over finite traces. We formulate the problem in Section III. The necessary and sufficient conditions to satisfy LTL specification are derived in Section IV. We present a numerical case study in Section V. We conclude the paper in Section VI.

II. Preliminary Background

In this section, we present preliminary background on linear temporal logic (LTL) over finite traces, denoted as LTL$_F$ [23]. We also introduce deterministic finite automaton which can be used to express the LTL$_F$ formula.

Let Π be a set of atomic propositions. An LTL$_F$ formula is constructed using the atomic proposition set Π, and is defined inductively as [23]

$$\varphi = \top \mid \pi \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathbf{U} \varphi_2 \mid \Diamond\varphi \mid \Box\varphi,$$

where $\mathbf{U}$, $\Diamond$, and $\Box$ are until, eventually, and globally operators, respectively, $\pi \in \Pi$. Since we will work in the continuous time domain, we omit the next operator of LTL$_F$.

Let β be a sequence of assignments of truth values to atomic propositions $\pi \in \Pi$. We let the length of β be $|\beta|$. Then the semantics of an LTL$_F$ formula is defined over β. Let the set of atomic propositions that are true at i-th position of β be $\beta(i)$. Then the satisfaction of LTL$_F$ formula φ at the position i, denoted as $\beta, i \models \varphi$, is recursively defined as

- $\beta, i \models \pi$ iff $\pi \in \beta(i)$;
- $\beta, i \models \neg\varphi$ iff $\beta, i \not\models \varphi$;
- $\beta, i \models \varphi_1 \wedge \varphi_2$ iff $\beta, i \models \varphi_1$ and $\beta, i \models \varphi_2$;
- $\beta, i \models \varphi_1 \vee \varphi_2$ iff $\beta, i \models \varphi_1$ or $\beta, i \models \varphi_2$;
- $\beta, i \models \varphi_1 \mathbf{U} \varphi_2$ iff there exists some position $i \leq j \leq |\beta|$ such that $\sigma, j \models \varphi_2$ and $\beta, k \models \varphi_1$ for all $i \leq k < j$;
- $\beta, i \models \Diamond\varphi$ iff there exists some position $i \leq j \leq |\beta|$ such that $\sigma, j \models \varphi$;
- $\beta, i \models \Box\varphi$ iff for all positions $i \leq j \leq |\beta|$ we have that $\sigma, j \models \varphi$ holds.

Given an LTL$_F$ formula, we can construct a deterministic finite automaton (DFA) as an equivalent representation [23]. The DFA will accept all and only words over Π that satisfy φ. A DFA is defined as follows.

Definition 1 (Deterministic Finite Automaton (DFA) [1])**.** *A DFA is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, $\Sigma = 2^{\Pi}$ is the finite set of alphabet, $\delta : Q \times \Sigma \mapsto Q$ is a finite set of transitions, $q_0 \in Q$ is the initial state, and F is a finite set of accepting states.*

Given a state $q \in Q$, we define the set of neighbor states of q as $\mathcal{N}(q) = \bigcup_{\sigma \in \Sigma} \delta(q, \sigma) \setminus \{q\}$, i.e., a state $q' \neq q$ is a neighbor state of q if there exists some transition δ such that the DFA can transition from q to q'. A finite word on DFA $\mathcal{A}$ is a finite sequence of symbols in Σ, defined as $\sigma = \sigma_0, \sigma_1, \ldots, \sigma_{n-1}$. Given a word σ, a run η on $\mathcal{A}$ is a finite sequence of states $\eta = q_0, q_1, \ldots, q_n$ such that $q_{j+1} = \delta(q_j, \sigma_j)$, where $j = 0, 1, \ldots, n-1$. A run η is an accepting run if it intersects with the accepting states F.

III. Problem Formulation

We consider a continuous-time control-affine system

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \tag{1}$$

where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ is the system state at time t, and $u(t) \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input. The initial state at time $t = 0$ is denoted as $x(0)$. The vector fields f and g are locally Lipschitz continuous. Given the current system state $x(t)$, a feedback control law is a function $\mu : \mathcal{X} \times \mathcal{M} \to \mathcal{U}$ specifying the control input at each time $t \geq 0$, where $\mathcal{M}$ is a finite set representing the memory. As we will detail in Section IV, set $\mathcal{M}$ can be chosen as $\mathcal{M} = Q$ to track the state evolution on automaton $\mathcal{A}$. Given the initial system state $x(0)$ and a control law μ, we define the trajectory of system (1) as $\mathbf{x} : \mathbb{R}_{\geq 0} \to \mathcal{X}$, which specifies the system states $\mathbf{x}(t; x(0), \mu)$ achieved by applying control law μ at each time $t \geq 0$ when the system starts from $x(0)$.

The system presented in Eqn. (1) is given a specification modeled by LTL$_F$. The objective of the system is to satisfy the given LTL$_F$ specification. We let Π be the finite set of atomic propositions. We define a labeling function $L : \mathcal{X} \to 2^{\Pi}$ that maps any state $x \in \mathcal{X}$ to a subset of atomic propositions that hold true at state x. For each atomic proposition $\pi \in \Pi$, we define define $[\![\pi]\!] = \{x : \pi \in L(x)\}$ to be the set of states that satisfies the atomic proposition $\pi \in \Pi$. With a slight abuse of the notation $[\![\cdot]\!]$, for a subset of atomic propositions $A \in 2^{\Pi}$, we define

$$[\![A]\!] = \begin{cases} \mathcal{X} \setminus \cup_{\pi \in \Pi} [\![\pi]\!], & \text{if } A = \emptyset, \\ \cap_{\pi \in A} [\![\pi]\!] \setminus \cup_{\pi \in \Pi \setminus A} [\![\pi]\!], & \text{otherwise.} \end{cases} \tag{2}$$

That is, $[\![A]\!]$ is the subset of system states $\mathcal{X}$ that satisfy all and only the propositions in A [24]. For an LTL$_F$ formula $\sigma = \pi \bowtie \sigma_1$ obtained using Boolean connectives, we define $[\![\sigma]\!]$ recursively as

$$[\![\sigma]\!] = \begin{cases} [\![\pi]\!] \cap [\![\sigma_1]\!], & \text{if } \bowtie = \wedge \\ [\![\pi]\!] \cup [\![\sigma_1]\!], & \text{if } \bowtie = \vee \end{cases}$$

where σ_1 is an LTL$_F$ formula involving only Boolean connectives. In the following, we define the satisfaction of a given LTL$_F$ specification by system (1) using the trace of a system trajectory $\mathbf{x}$ given as follows.

Definition 2 (Trace of Trajectory [24])**.** *Let $t_0, t_1, \ldots, t_N$ be a time sequence such that*

- $0 = t_0 < t_1 < \ldots < t_N$,
- $L(\mathbf{x}(t; x(0), \mu)) = L(\mathbf{x}(t_k; x(0), \mu))$ *for all* $t \in [t_k, t_{k+1})$ *where* $k = 0, \ldots, N$,
- $\lim_{\epsilon \to 0} L(\mathbf{x}(t_k - \epsilon; x(0), \mu)) \neq L(\mathbf{x}(t_k; x(0), \mu))$ *for all* $k = 0, \ldots, N$.

We then say the sequence $Trace(\mathbf{x}) = A_0, A_1, \ldots, A_N$ *is the trace of trajectory* $\mathbf{x}$*, where* $A_k = L(\mathbf{x}(t_k; x(0), \mu))$.

The trace given in Definition 2 describes the sequence of the atomic propositions that is satisfied by the system trajectory. Following the semantics of LTL$_F$, one can verify whether a given specification φ is satisfied or not. If $Trace(\mathbf{x}) \models \varphi$, we say system (1) satisfies the specification φ under control law μ, or control law μ satisfies φ. We summarize the problem studied in this paper as follows.

Problem 1. *Given the initial system state $x(0)$ for system* (1)*, derive the necessary and sufficient conditions for a control law μ so that system* (1) *satisfies the given LTL specification φ belonging to LTL$_F$.*

IV. Solution Approach

In this section, we present our proposed solution approach to Problem 1. Given the LTL$_F$ specification φ, we first take the negation of φ, denoted as $\neg\varphi$. Taking the negation allows us to convert the problem of reaching the accepting states of the DFA representing φ to an equivalent safety problem. We then construct the corresponding DFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ of $\neg\varphi$ as given in Definition 1. For each transition of $\mathcal{A}$, we verify whether system (1) can realize the transition or not. If the transition can be realized by the system under any control law, then we say the transition is feasible. The verification is achieved by searching for a control barrier certificate (CBC), whose existence is equivalent to the infeasibility of the transition. We label each transition of $\mathcal{A}$ based on the verification result to indicate if the transition is feasible or not. We finally verify if there exists an accepting run on DFA $\mathcal{A}$ starting from the initial state q_0 such that each transition of the accepting run is feasible. If no such accepting run exists, then system (1) violates $\neg\varphi$, and thus satisfies φ. If an accepting run exists after we label the automaton, then for each accepting run, we verify the composability of the control laws associated with the transitions along the accepting run. If the control laws are composable, then specification φ cannot be satisfied. A summary of our proposed approach is presented in Algorithm 1. We detail each step of our proposed solution approach in the remainder of this section.

A. Verifying the Feasibility of Transitions in the Automaton

In this subsection, we present how to verify the feasibility of each transition in the DFA $\mathcal{A}$ corresponding to the negated formula $\neg\varphi$, as needed in line 4 of Algorithm 1.

Given the DFA $\mathcal{A}$ of the negated formula $\neg\varphi$, we construct the following sets corresponding to the transition from state q to $q' \in \mathcal{N}(q)$ under input symbol σ:

$$\Omega_i^q = [\![\phi_q]\!],\ \Omega_r^{q,q'} = [\![\phi_{q'}]\!] \cap [\![\sigma]\!],\ \Omega^{q,q'} = \Omega_i^q \cup \Omega_r^{q,q'}, \quad (3)$$

where $\phi_q \in \Sigma$ is the input symbol leading to the transition $\delta(q, \phi_q) = q$. Here state set $\Omega_i^q \subseteq \mathcal{X}$ is the set of states such that $L(x)$ triggers DFA $\mathcal{A}$ to take the self-loop transition at state $q \in Q$ for all $x \in \Omega_i^q$. State set $\Omega_r^{q,q'} \subseteq \mathcal{X}$ is interpreted as the set of states whose labels cause the DFA to transition from state q to q' and remains at state q'. We remark that when ϕ_q is not explicitly given in the DFA, we can choose Ω_i^q in the following way. We define Σ' as $\Sigma^q = \cup_{q' \in \mathcal{N}(q)} \{\sigma : q' = \delta(q, \sigma)\}$. We then let $\Omega_i^q = [\![\Sigma \setminus \Sigma^q]\!]$. When $\Sigma^q = \Sigma$, then there must exist some state $q' \neq q$ such that the transition from state q to q' is automatically realized. Throughout this paper, we make the following assumption.

Assumption 1. *We assume that sets Ω_i^q and $\Omega_r^{q,q'}$ are compact for each transition from state $q \in Q$ to $q' \in Q$ in automaton $\mathcal{A}$.*

When Assumption 1 holds, we have that set $\Omega^{q,q'} \subseteq \mathcal{X}$ is also compact. In the following, we verify the feasibility of transition from state q to q' under input symbol σ using a safety property. The feasibility is defined as follows.

Definition 3 (Feasibility of a transition)**.** *A transition from q to q' under input symbol σ of a DFA is said to be feasible if and only if there exists some state $x \in [\![\Omega_i^q]\!]$ and a time $T \geq 0$ such that every control law μ for system* (1) *satisfies $\mathbf{x}(T; x, \mu) \in [\![\Omega_r^{q,q'}]\!]$.*

If a transition is feasible, then we say the transition can be realized by system (1). Thus if there exists an accepting run on the DFA of $\neg\varphi$ whose transitions can be realized by system (1), then specification φ cannot be satisfied by the

Algorithm 1 Summary of the proposed solution approach

1: **Input:** Specification φ, system dynamics (1)
2: **Output:** Control law μ
3: Compute the automaton $\mathcal{A}$ corresponding to the negated formula $\neg\varphi$
4: Verify the feasibility of each transition of $\mathcal{A}$ via Theorem 1
5: Label each transition from state q to q' using a tuple $(y_{q,q'}, U_{q,q'})$
6: **if** the labels yield an accepting run **then**
7: **for** each labeled accepting run **do**
8: Verify the separation of each set $\Omega_i^{q,q'}$ along the accepting run via Proposition 1
9: **if** the conditions in Theorem 2 do not hold **then**
10: **return** failure
11: **break**
12: **end if**
13: **return** the control law μ
14: **end for**
15: **else**
16: **return** the control law μ
17: **end if**

system. We will verify the feasibility of each transition in $\mathcal{A}$ using a safety property defined as follows.

Definition 4 (Safety)**.** *Consider system* (1)*. Let $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{X}_i, \mathcal{X}_r \subseteq \mathcal{X}$ be given sets. System* (1) *is safe under a control law μ if there exists no system trajectory with $x(0) \in \mathcal{X}_i$, $\mathbf{x}(T; x(0), \mu) \in \mathcal{X}_r$ for some $T \geq 0$ and $\mathbf{x}(t; x(0), \mu) \in \mathcal{X}$ for all $t \in [0, T]$.*

As shown in [25], the safety property given in Definition 4 can be verified using a control barrier certificate (CBC). We have the following preliminary result.

Lemma 1 ([25])**.** *Consider system* (1)*. Let $\Omega^{q,q'} \subset \mathbb{R}^n$, $\Omega_i^q, \Omega_r^{q,q'} \subseteq \Omega^{q,q'}$ be compact sets. Suppose there exists a continuously differentiable function b such that $\frac{\partial b}{\partial x}(f(x) + g(x)\mu(x)) < 0$ holds for some control law μ and for all $x \in \Omega^{q,q'}$. Then there exists a continuously differentiable function B satisfying*

$$B(x) \leq 0, \ \forall x \in \Omega_i^q, \tag{4a}$$

$$B(x) > 0, \ \forall x \in \Omega_r^{q,q'}, \tag{4b}$$

$$\frac{\partial B}{\partial x}(x)[f(x) + g(x)\mu(x)] \leq 0, \ \forall x \in \Omega^{q,q'} \tag{4c}$$

if and only if the safety property in Definition 4 holds under some control law $\mu : \Omega^{q,q'} \to \mathcal{U}$, where $\Omega_i^q, \Omega_r^{q,q'}$, and $\Omega^{q,q'}$ correspond to $\mathcal{X}_i, \mathcal{X}_r$, and $\mathcal{X}$ in Definition 4, respectively.

Function B satisfying Eqn. (4) is a CBC. Leveraging Lemma 1, we verify the feasibility of the transition from state q to q' under input symbol σ as follows.

Theorem 1. *Consider system* (1) *and a transition from state q to q' under input symbol σ. Let Ω_i^q, $\Omega_r^{q,q'}$, and $\Omega^{q,q'}$ be defined as in Eqn.* (3) *satisfying Assumption 1. Suppose there exists a continuously differentiable function b such that $\frac{\partial b}{\partial x}(f(x) + g(x)u) < 0$ holds for all u and for all $x \in \Omega^{q,q'}$. Then the transition from state q to q' under input symbol σ cannot be realized by system* (1) *if and only if there exists a CBC $B(x)$ satisfying Eqn.* (4) *under a control policy μ.*

Proof. By Lemma 1, we have that the safety property holds if and only if there exists a CBC $B(x)$ satisfying Eqn. (4). Then the proof reduces to show the equivalence between the infeasibility of the transition on automaton $\mathcal{A}$ and the safety property as given in Definition 4.

We first show that if the safety property holds, then the transition from state q to q' under input symbol σ cannot be realized by system (1), i.e., the transition is infeasible. We prove by contradiction. Suppose that the safety property holds while transition from state q to q' under input symbol σ is feasible. Using Definition 4, we have that the system trajectory cannot reach $\Omega_r^{q,q'}$ for some $T \geq 0$ while ensuring $x_t \in \Omega^{q,q'}$ for all $t \in [0, T]$. To this end, the system trajectory can only reach $\Omega_r^{q,q'}$ by leaving set $\Omega^{q,q'}$, which can only be triggered via some other input symbol $\sigma' \neq \sigma$, leading to contradiction.

We next prove that if the transition from state q to q' under input symbol σ is infeasible, then the safety property holds. Suppose that the transition is infeasible while the safety property does not hold. Therefore, there exists some system trajectory such that $x_0 \in \Omega_i^q$, $x_T \in \Omega_r^{q,q'}$ for some $T \geq 0$ and $x_t \in \Omega^{q,q'}$ for all $t \in [0, T]$. Using Eqn. (3), we have that this trajectory realizes the transition from from state q to q' under input symbol σ, leading to contradiction. □

Using Theorem 1, we can verify whether each transition in DFA can be realized by system (1) or not. In the following, we design a labeling procedure to label each transition from state q to q' in DFA using one or multiple pairs $(y_{q,q'}, U_{q,q'})$ based on its feasibility, where $y_{q,q'} \in \{0, 1\}$ and $U_{q,q'} \subseteq \mathcal{U}$. This corresponds to line 5 of Algorithm 1. For a transition from state q to q' in DFA $\mathcal{A}$, if we can find some CBC $B(x)$ and non-empty $U_{q,q'} \neq \emptyset$ such that any $u \in U_{q,q'} \subseteq \mathcal{U}$ renders Eqn. (4) to be satisfied under control law $\mu : \Omega^{q,q'} \to U_{q,q'}$, then we label transition from state q to q' as $(y_{q,q'}, U_{q,q'})$, where $y_{q,q'} = 0$. In addition, if there exists a self-loop transition at state q, we label the self-loop transition as $(y_{q,q}, U_{q,q})$, where $y_{q,q} = 1$ and $U_{q,q} = U_{q,q'}$. If no CBC $B(x)$ can be found to satisfy Eqn. (4), we label $y_{q,q'} = 1$ and $y_{q,q} = 0$ when the self-loop transition at state q exists.

B. Composable Control Laws and Realizability of $\neg\varphi$

Using Theorem 1, we have that the feasibility of each transition in automaton $\mathcal{A}$ can be verified by computing a CBC $B(x)$. Given the feasibility of each transition, we then label each transition of $\mathcal{A}$ using $(y_{q,q'}, U_{q,q'})$. Using these labels, we finally verify whether there exists an accepting run on $\mathcal{A}$ starting from q_0 such that each transition along the run is labeled as $y_{q,q'} = 1$ (line 6 of Algorithm 1). If no such run exists (line 15-17 of Algorithm 1), then system (1) satisfies the specification φ under some control law μ. We formally state this result as follows.

Lemma 2. *There exists a control law μ for system* (1) *to satisfy specification φ if there does not exist an accepting run on $\mathcal{A}$ with all transitions being labeled as $y_{q,q'} = 1$.*

Proof. The lemma holds by the equivalence between violating $\neg\varphi$ and the absence of an accepting run. □

We next discuss the scenario where an accepting run on $\mathcal{A}$, denoted as η, can be found with each transition along η being labeled as $y_{q,q'} = 1$. This scenario corresponds to line 6-14 in Algorithm 1. We observe that even if an accepting run on $\mathcal{A}$ is found and labeled in Algorithm 1, system (1) may not always be capable of realizing it. The reason is that there exist dependencies between the transitions along η [21], [22]. The dependencies, raised due to dynamics in Eqn. (1), capture whether the control laws associated with the transitions along η can be composed or not. In the following, we derive the conditions for control laws to be composable and thus system (1) can realize the accepting run η.

We first discuss the dependencies between the transitions. Consider an accepting run η that is labeled as $y_{q,q'} = 1$ for each transition from q to q' along η. For each state q along the accepting run η, we define the forward reachable

set when starting from Ω_i^q as

$$\mathcal{R}_{fwd}(\Omega_i^q) = \bigcup_{T\geq 0, x_0\in\Omega_i^q} \{x_T : x_T = \mathbf{x}(T; x_0, \mu) \text{ for some control law } \mu\}. \quad (5)$$

We additionally define the backward reachable set as

$$\mathcal{R}_{bck}(\Omega_i^q) = \{x : \exists T \geq 0 \text{ s.t. } \mathbf{x}(T; x, \mu) \in \Omega_i^q \text{ for some control law } \mu\}. \quad (6)$$

We observe that accepting run η cannot be realized by system (1) when there exists at least one state q on η such that

$$\mathcal{R}_{fwd}(\Omega_i^{pre(q;\eta)}) \cap \mathcal{R}_{bck}(\Omega_i^{suc(q;\eta)}) = \emptyset,$$

where $pre(q;\eta)$ and $suc(q;\eta)$ are the predecessor and successor of state q along run η, respectively. In the remainder of this subsection, we verify the emptiness of $\mathcal{R}_{fwd}(\Omega_i^{pre(q;\eta)}) \cap \mathcal{R}_{bck}(\Omega_i^{suc(q;\eta)})$ via set separation which is defined below.

Definition 5 (Set Separation)**.** *Let sets $\mathcal{R}_1, \mathcal{R}_2 \subset \Omega_r^{q,q'}$ be closed. We say $\mathcal{R}_1$ and $\mathcal{R}_2$ are separable by system* (1) *if there exists no time $T \geq 0$ such that $\mathbf{x}(T; x(0), \mu) \in \mathcal{R}_2$ and $\mathbf{x}(t; x(0), \mu) \in \Omega_r^{q,q'}$ for all $t \in [0, T]$ when starting from $\mathcal{R}_1$ by implementing some control law μ.*

As shown in [26], set separation can be verified via a certificate $D(x)$.

Lemma 3 ([26])**.** *Let sets $\mathcal{R}_1$ and $\mathcal{R}_2$ be closed subsets of $\mathcal{X}$. Then $\mathcal{R}_1$ and $\mathcal{R}_2$ are separable if and only if there exists some function $D(x)$ satisfying*

$$D(x_1) \leq D(x_2),\ \forall x_1 \in \mathcal{R}_1, x_2 \in \mathcal{R}_2 \quad (7a)$$

$$\inf_{u\in\mathcal{U}} \{\frac{\partial D}{\partial x}(f(x) + g(x)u)\} < 0,\ \forall x \in \mathcal{X}. \quad (7b)$$

Leveraging Lemma 3, we verify the the emptiness of $\mathcal{R}_{fwd}(\Omega_i^{pre(q;\eta)}) \cap \mathcal{R}_{bck}(\Omega_i^{suc(q;\eta)})$ as follows.

Theorem 2. *Suppose that Assumption 1 holds. System* (1) *cannot satisfy specification φ if and only if there exists some accepting run η on DFA $\mathcal{A}$ such that each transition from state q to q' along η is labeled with $y_{q,q'} = 1$ by line 5 of Algorithm 1, and for any state q along accepting run η, sets $\mathcal{R}_1(q), \mathcal{R}_2(q) \subseteq \Omega_i^q$ are not separable by system* (1) *under any control law μ, where*

$$\mathcal{R}_1(q) = \mathcal{R}_{fwd}(\Omega_i^{pre(q;\eta)}) \cap \Omega_i^q,$$
$$\mathcal{R}_2(q) = \mathcal{R}_{bck}(\Omega_i^{suc(q;\eta)}) \cap \Omega_i^q.$$

Proof. We first prove the 'if' direction. If line 5 of Algorithm 1 yields an accepting run with $y_{q,q'} = 1$ for each transition from q to q' along η, and $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ are not separable by system (1) under any control law μ, we have that system (1) joins sets $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ without leaving set Ω_i^q. As a consequence, the accepting run η is realized by system (1) regardless of the choice of control law μ. Since η is accepted by the automaton corresponding to $\neg\varphi$, we have that system (1) cannot satisfy φ.

We next prove the 'only if' direction. When the system cannot satisfy specification φ, it implies that run η corresponding to $\neg\varphi$ can be realized by the system under any control law. Therefore, Algorithm 1 must label some accepting run, i.e., each transitions from state q to q' along η is feasible. The self-loop transition at each state q along η is feasible for all q, indicating that $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ are not separable under any control law. □

Theorem 1 and Theorem 2 together give the necessary and sufficient conditions for a control law to satisfy specification φ. That is, any control law that satisfies Theorem 1 and Theorem 2 guarantees the satisfaction of φ. If there exists no control law μ that can satisfy Theorem 1 and Theorem 2, then specification φ cannot be satisfied by system (1).

Computing the reachable sets for nonlinear system (1) is difficult [27]. In what follows, we show that the conditions given in Theorem 2 can be verified by synthesizing two separable sets $\mathcal{W}_1, \mathcal{W}_2 \subset \Omega_i^q$ and checking the feasibility of a set of inequalities.

Proposition 1. *Suppose that Assumption 1 holds. Let q be some state along an accepting run η given by line 5 of Algorithm 1. Let σ be defined as $q = \delta(pre(q;\eta), \sigma)$. Sets $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ are not separable if and only if there exist sets $\mathcal{W}_1, \mathcal{W}_2 \subset \Omega_i^q$ and no continuously differentiable functions $B_1(x), B_2(x)$, and $D(x)$ satisfying*

$$B_1(x) \leq 0,\ \forall x \in \Omega_i^{pre(q;\eta)}, \quad (8a)$$

$$B_1(x) > 0,\ \forall x \in \mathcal{W}_1, \quad (8b)$$

$$\inf_{u\in\mathcal{U}} \{\frac{\partial B_1}{\partial x}(x)[f(x) + g(x)u]\} \leq 0, \quad \forall x \in \left(\Omega_i^{pre(q;\eta),q} \cup \mathcal{W}_1 \cap [\![\sigma]\!]\right), \quad (8c)$$

$$B_2(x) \leq 0,\ \forall x \in \mathcal{W}_2, \quad (8d)$$

$$B_2(x) > 0,\ \forall x \in \Omega_r^{q,suc(q;\eta)}, \quad (8e)$$

$$\inf_{u\in\mathcal{U}} \{\frac{\partial B_2}{\partial x}(x)[f(x) + g(x)u]\} \leq 0, \quad \forall x \in \left(\mathcal{W}_2 \cup \Omega_r^{q,suc(q;\eta)}\right), \quad (8f)$$

$$D(x_1) \leq D(x_2),\ \forall x_1 \in \mathcal{W}_1, x_2 \in \mathcal{W}_2, \quad (8g)$$

$$\inf_{u\in\mathcal{U}} \{\frac{\partial D}{\partial x}(f(x) + g(x)u)\} < 0,\ \forall x \in \Omega_i^q. \quad (8h)$$

Proof. We first prove the 'if' direction. If line 5 of Algorithm 1 yields an accepting run with $y_{q,q'} = 1$ for each transition from q to q' along η, and $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ are not separable by system (1) under any control law μ, we have that system (1) joins sets $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ without leaving set Ω_i^q. As a consequence, the accepting run η is realized by system (1) regardless of the choice of control law μ. Since η is accepted by the automaton corresponding to $\neg\varphi$, we have that system (1) cannot satisfy φ.

We next prove the 'only if' direction. When the system cannot satisfy specification φ, it implies that run η corresponding to $\neg\varphi$ can be realized by the system under any control law. Therefore, Algorithm 1 must label some accepting run, i.e., each transitions from state q to q' along

η is feasible. The self-loop transition at each state q along η is feasible for all q, indicating that $\mathcal{R}_1(q)$ and $\mathcal{R}_2(q)$ are not separable under any control law. □

We finally discuss how we extract the control law $\mu : \mathcal{X} \times \mathcal{M} \to \mathcal{U}$. Note that here memory $\mathcal{M}$ is set as $\mathcal{M} = Q$, which is used to track the current state of automaton $\mathcal{A}$. Given the current system state x and the current state $q \in Q$ of automaton $\mathcal{A}$, the control law is defined as $\mu(x, q) = u$ such that $u \in U_{q,q'}$, where there exists no accepting run η starting from q' whose transitions are labeled as $y_{q'',q'''} = 1$.

V. Case Study

In this section, we evaluate the proposed approach using an example on multi-agent motion planning. There are two agents $i \in \{1, 2\}$ navigating in a bounded 2-dimensional domain. The dynamics of the system are given as

$$\left[\dot{x}_1, \dot{y}_1, \dot{x}_2, \dot{y}_2\right]^\top = \left[u_{x,1}, u_{y,1}, u_{x,2}, u_{y,2}\right]^\top$$

where x_i represents the position along X-coordinate and y_i represents the position along Y-coordinate of agent $i \in \{1, 2\}$. We use $u_{x,i}$ and $u_{y,i}$ to denote the horizontal and vertical velocity given to the agent. We denote $x = [x_1, y_1, x_2, y_2]^\top$. The initial positions of agents 1 and 2 are set as $x = [0, 0, 1, 0]^\top$. We assume that $x_i \in [-10, 10]$ and $y_i \in [-10, 10]$ for all $i \in \{1, 2\}$.

We let the atomic proposition set be defined as $\Pi = \{Dest\ 1, Dest\ 2, Dest\ 3, Obs\}$. Here Obs represents the set of states belonging to a static obstacle, whose geometric information is given as

$$[\![Obs]\!] = \prod_{i \in \{1,2\}} \{x : (x_i - 1)^2 + (y_i - 2)^2 \leq 1\}.$$

Atomic propositions $Dest$ 1, $Dest$ 2, and $Dest$ 3 label the set of states that belong to static destinations. The geometric information for each destination is given as

$$[\![Dest\ 1]\!] = \{x : (x_1 + 4)^2 + y_1^2 \leq 1\},$$
$$[\![Dest\ 2]\!] = \{x : (x_2 - 6)^2 + (y_2 - 4)^2 \leq 1\}.$$
$$[\![Dest\ 3]\!] = \{x : (x_1 - 2)^2 + y_1^2 \leq 1\}.$$

The specification φ given to the agents is formulated as

$$\varphi = \Diamond(Dest\ 1 \wedge \Diamond Dest\ 2) \wedge \Box\neg Obs \wedge (\Diamond Dest\ 2 \implies \Diamond Dest\ 3).$$

Specification φ requires that (i) destinations $[\![Dest\ 1]\!]$ and $[\![Dest\ 2]\!]$ are reached in order, (ii) both agents always avoid the obstacle Obs, and (iii) once $[\![Dest\ 2]\!]$ is reached, then $[\![Dest\ 3]\!]$ needs to be reached. Specification φ introduces coupling between agents 1 and 2.

Using our proposed approach, we first generate the DFA $\mathcal{A}$ corresponding to $\neg\varphi$. There are 7 states, denoted as $Q = \{0, 1, \ldots, 6\}$, and 25 transitions in the DFA. We adopt the sum-of-squares optimization-based approach [15], [16] to search for CBCs. Using Theorem 1, we can label the transitions ending at state 0 with $y_{q,0} = 1$, where $q \in \{0, 2, 3, 4, 5\}$, which also renders the self-loop transitions at states $\{2, 3, 4, 5, 6\}$ to be infeasible. In addition, we label the transitions ending at state 1 as $y_{q,1} = 0$, where $q \in \{0, 2, 3, 4, 5, 6\}$. In this case, there is always a feasible control input at state $q = 6$ since there exists no accepting run labeled to be feasible as long as there exists some q' such that $y_{6,q'} = 1$, where $q' \in \{2, 3, 4, 5\}$.

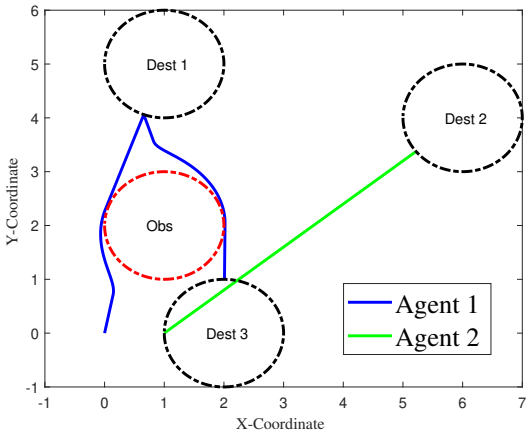

Fig. 1: This figure presents the trajectories of agents 1 and 2 for specification φ. The obstacle and destinations are plotted using dotted red and black lines, respectively. The trajectories of agent 1 and 2 obtained using our proposed approach are plotted using solid blue and green lines, respectively.

Our control law realizes the infinite non-accepting run $6, 5, 2, 0, 0, \ldots$, indicating that $[\![Dest\ 1]\!]$, $[\![Dest\ 2]\!]$, and $[\![Dest\ 3]\!]$ need to be reached in this order. We illustrate the corresponding trajectories for agents 1 and 2 in Fig. 1. We depict the trajectories for agent 1 and 2 using blue and green lines, respectively. We observe that agent 1 reaches $[\![Dest\ 1]\!]$ and agent 2 reaches $[\![Dest\ 2]\!]$. After that, agent 1 reaches $[\![Dest\ 3]\!]$. In the meantime, both agents avoid the obstacle region $[\![Obs]\!]$ to guarantee the safety property. Therefore, the trajectories of the agents satisfy the given specification φ.

VI. Conclusion

In this paper, we considered continuous-time control-affine systems under linear temporal logic constraints defined over finite traces. We developed the necessary and sufficient conditions for a control law of the system to satisfy the given specification. We first negated the given specification and generated the deterministic finite automaton to represent the negated specification. We then constructed a safety verification problem for each transition in the automaton. We derived the necessary and sufficient conditions for a control law to solve the decomposed safety verification problem via CBC. We formulated the dependencies among the transitions by considering the composability of control laws. We derived the necessary and sufficient conditions for the composability, and thus realizability of the specification. We illustrated the proposed approach using a numerical case study.

REFERENCES

[1] C. Baier, J.-P. Katoen, and K. G. Larsen, *Principles of Model Checking*. MIT Press, 2008.

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

[3] J. Fu and U. Topcu, "Synthesis of shared autonomy policies with temporal logic specifications," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 1, pp. 7–17, 2016.

[4] S. Coogan, E. A. Gol, M. Arcak, and C. Belta, "Traffic network control from temporal logic specifications," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 2, pp. 162–172, 2015.

[5] X. Ding, S. L. 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.

[6] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning for dynamical systems," in *48th IEEE Conference on Decision and Control (CDC)*. IEEE, 2009, pp. 5997–6004.

[7] J. Fu and U. Topcu, "Computational methods for stochastic control with metric interval temporal logic specifications," in *IEEE Conference on Decision and Control (CDC)*, 2015, pp. 7440–7447.

[8] L. Niu and A. Clark, "Optimal secure control with linear temporal logic constraints," *IEEE Transactions on Automatic Control*, 2019.

[9] J. Liu, "Robust abstractions for control synthesis: Completeness via robustness for linear-time properties," in *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, 2017, pp. 101–110.

[10] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional synthesis of finite abstractions for continuous-space stochastic control systems: A small-gain approach," *IFAC-PapersOnLine*, vol. 51, no. 16, pp. 265–270, 2018.

[11] ——, "Compositional abstraction of large-scale stochastic systems: A relaxed dissipativity approach," *Nonlinear Analysis: Hybrid Systems*, vol. 36, p. 100880, 2020.

[12] S. Esmaeil Zadeh Soudjani, A. Abate, and R. Majumdar, "Dynamic Bayesian networks for formal verification of structured stochastic processes," *Acta Informatica*, vol. 54, no. 2, pp. 217–242, 2017.

[13] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 96–101, 2018.

[14] M. Srinivasan and S. Coogan, "Control of mobile robots using barrier functions under temporal logic specifications," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 363–374, 2021.

[15] P. Jagtap, S. Soudjani, and M. Zamani, "Formal synthesis of stochastic systems via control barrier certificates," *IEEE Transactions on Automatic Control*, 2020.

[16] M. Anand, A. Lavaei, and M. Zamani, "Compositional synthesis of control barrier certificates for networks of stochastic systems against ω-regular specifications," *arXiv preprint arXiv:2103.02226*, 2021.

[17] L. Niu and A. Clark, "Control barrier functions for abstraction-free control synthesis under temporal logic constraints," *59th IEEE Conference on Decision and Control (CDC)*, 2020.

[18] M. Srinivasan, S. Coogan, and M. Egerstedt, "Control of multi-agent systems with finite time control barrier certificates and temporal logic," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 1991–1996.

[19] A. Bisoffi and D. V. Dimarogonas, "Satisfaction of linear temporal logic specifications through recurrence tools for hybrid systems," *IEEE Transactions on Automatic Control*, vol. 66, no. 2, pp. 818–825, 2020.

[20] A. Nejati, S. Soudjani, and M. Zamani, "Compositional construction of control barrier functions for continuous-time stochastic hybrid systems," *arXiv preprint arXiv:2012.07296*, 2020.

[21] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *The International Journal of Robotics Research*, vol. 18, no. 6, pp. 534–555, 1999.

[22] J. A. DeCastro and H. Kress-Gazit, "Synthesis of nonlinear continuous controllers for verifiably correct high-level, reactive behaviors," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 378–394, 2015.

[23] G. De Giacomo and M. Y. Vardi, "Linear temporal logic and linear dynamic logic on finite traces," in *International Joint Conference on Artificial Intelligence*. Association for Computing Machinery, 2013, pp. 854–860.

[24] T. Wongpiromsarn, U. Topcu, and A. Lamperski, "Automata theory meets barrier certificates: Temporal logic verification of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3344–3355, 2015.

[25] S. Prajna and A. Rantzer, "On the necessity of barrier certificates," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 526–531, 2005.

[26] R. Wisniewski and C. Sloth, "Converse barrier certificate theorems," *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1356–1361, 2015.

[27] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-Jacobi reachability: A brief overview and recent advances," in *IEEE Conference on Decision and Control (CDC)*, 2017, pp. 2242–2253.