Efficient Automata-based Planning and Control under Spatio-Temporal Logic Specifications

Lars Lindemann and Dimos V. Dimarogonas

Abstract—The use of spatio-temporal logics in control is motivated by the need to impose complex spatial and temporal behavior on dynamical systems, and to control these systems accordingly. Synthesizing correct-by-design control laws is a challenging task resulting in computationally demanding methods. We consider efficient automata-based planning for continuous-time systems under signal interval temporal logic specifications, an expressive fragment of signal temporal logic. The planning is based on recent results for automata-based verification of metric interval temporal logic. A timed signal transducer is obtained accepting all Boolean signals that satisfy a metric interval temporal logic specification, which is abstracted from the signal interval temporal logic specification at hand. This transducer is modified to account for the spatial properties of the signal interval temporal logic specification, characterizing all real-valued signals that satisfy this specification. Using logic-based feedback control laws, such as the ones we have presented in earlier works, we then provide an abstraction of the system that, in a suitable way, aligns with the modified timed signal transducer. This allows to avoid the state space explosion that is typically induced by forming a product automaton between an abstraction of the system and the specification.

I. INTRODUCTION

The control of dynamical systems under complex temporal logic specifications has lately received increasing attention. One can distinguish between temporal logics that allow to express qualitative, e.g., linear temporal logic (LTL) [1], and quantitative, e.g., metric interval temporal logic (MITL) [2], temporal properties. An MITL specification can be translated into a language equivalent timed automaton [2]. If the accepted language of this automaton is not empty, the MITL specification is satisfiable. Emptiness can be checked by abstracting the timed automaton into its untimed region automaton [3]. There exists no tool to algorithmically translate an MITL specification, interpreted over continuous-time semantics, into its language equivalent timed automaton. For point-wise semantics, such a tool has been presented in [4]. Point-wise semantics, however, do not guarantee the satisfaction of the MITL specification in continuous time. The procedure of [2], for continuous-time semantics, is complex and rather of theoretical nature. The results from [5], [6] are more intuitive and present a compositional way to construct a timed signal transducer for an MITL specification. More recently, spatio-temporal logics have been considered that further allow to reason about spatial properties. Such spatio-temporal logics are signal temporal logic (STL) [7] or a variant of MITL, where propositions are associated with observation maps [8]. The richness and complexity of the chosen temporal logic increases by going from qualitative to quantitative temporal properties as well as by going from non-spatial to spatial properties.

Classical control theoretical tools, which deal with invariance and stability of dynamical systems, are not rich enough to solely deal with the control problem at hand. Hence, automata-based tools have been used to divide a specification into subtasks that can be achieved sequentially by low-level feedback control laws. There exist numerous approaches for LTL [9]–[11] and for MITL [12]–[17]. The idea is to abstract the system into an automaton and to form a product automaton with an automaton representing the LTL/MITL specification. This procedure is subject to a computational blowup due to an exponential explosion in the resulting state space. Spatio-temporal logics have not leveraged automata-based results. Thus far, STL and the associated robust semantics have been used for the full STL fragment and only for discrete-time systems resulting in computationally demanding mixed integer linear programs [18]. Other approaches have maximized the robust semantics in optimization-based frameworks, resulting again in computationally expensive methods [19], [20], prone to get stuck in local minima. For continuous-time systems and fragments of STL, robust and computationally-efficient time-varying feedback control laws have been presented in [21], [22].

We consider continuous-time systems under spatio-temporal logic specifications expressed in signal interval temporal logic (SITL), an expressive STL fragment where temporal operators can not be constrained by singular intervals. We remark that SITL is a more expressive fragment than the fragments of STL that have been considered in [21], [22]. The SITL specification at hand is first abstracted into an MITL specification that is translated into its language equivalent timed signal transducer [6]. This transducer is modified to account for the error induced by considering propositions (MITL) instead of predicates (SITL). The modified timed signal transducer characterizes all real-valued signals that satisfy the SITL specification and it can hence be checked whether or not the specification is satisfiable. To the best of our knowledge, this is the first decidability result for SITL interpreted over continuous-time semantics. We then use logic-based feedback control laws that can achieve finite-time reachability and invariance, such as for instance presented.
in [21], [22], to define a timed abstraction of the system. This abstraction aligns, in a suitable way, with the modified timed signal transducer. In particular, this abstraction considers transitions between boolean combinations of predicates instead of transitions between cells of the continuous state space. In this way, an explosion of the state space in the product automaton between the abstraction and the timed signal transducer can be avoided. This product typically induces \( O(mn) \) states where \( m \) and \( n \) are the number of states in abstraction and specification automaton, respectively, while our approach works directly on an automaton with \( n \) or less states. The main contribution is hence an efficient planning and control framework for continuous-time systems under spatio-temporal logic specifications.

Sec. II presents preliminaries and problem formulation. Our proposed problem solution is stated in Sec. III. Simulations and conclusions are given in Sec. IV and Sec. V.

II. PRELIMINARIES AND PROBLEM FORMULATION

True and false are \( \top \) and \( \bot \) with \( \mathbb{B} := \{ \top, \bot \} \); \( \mathbb{R} \), and \( \mathbb{N} \) are the real, rational, and natural numbers, respectively, while \( \mathbb{R}_{\geq 0} \) and \( \mathbb{Q}_{\geq 0} \) denote their respective nonnegative subsets; \( \mathbb{R}_{> 0} \) denotes the positive real numbers.

A. Real-time Temporal Logics

Let \( P \) be a set of propositions. Metric interval temporal logic (MITL) [2] is based on propositions \( p \in P \) as well as Boolean and temporal operators. The syntax is given by

\[
\varphi := \top \mid p \mid \neg \varphi \mid \varphi' \land \varphi'' \mid \varphi' \cup \varphi''
\]

where \( \varphi, \varphi', \) and \( \varphi'' \) are MITL formulas, \( \neg \) and \( \land \) denote negation and conjunction, respectively, and \( U_t \) is the until operator with \( I \subseteq \mathbb{Q}_{\geq 0} \) and \( I \) not being a singleton. We define \( \varphi' \lor \varphi'' := \neg (\neg \varphi' \land \neg \varphi'') \) (disjunction), \( F_t \varphi := \top \cup \varphi \) (eventually operator), and \( G_t \varphi := \neg F_t \neg \varphi \) (always operator). Let \( d : \mathbb{R}_{\geq 0} \rightarrow \mathbb{B}^{P} \) be a Boolean signal corresponding to truth values of the propositions in \( P \) over time. Define also the projection of \( d \) onto \( p \in P \) as \( \text{proj}_p(d) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{B} \). The expression \( (d, t) \models \varphi \) indicates that the signal \( d \) satisfies an MITL formula \( \varphi \) at time \( t \). The continuous-time semantics of an MITL formula \( \varphi \) in [4, Sec. 4] are then defined as \( (d, t) \models p \) iff \( \text{proj}_p(d(t)) = \top \), \( (d, t) \models \neg \varphi \) iff \( (d, t) \models \varphi' \land \varphi'' \) iff \( (d, t) \models \varphi' \cup \varphi'' \) iff \( \exists t'' \in t + I \), \( (d, t'') \models \varphi' \) and \( \forall t' \in (t, t'') \), \( (d, t') \models \varphi' \). We further define signal interval temporal logic (SITL), a fragment of signal temporal logic (STL) [7], as a simple yet expressive spatio-temporal logic by excluding, similar to MITL, singular time intervals in the temporal operators. SITL considers, instead of propositions, predicates \( \mu \in M \) where \( M \) denotes a set of predicates. The truth value of \( \mu \) is determined by a predicate function \( h : \mathbb{R}^n \rightarrow \mathbb{R} \) so that, for \( \zeta \in \mathbb{R}^n \), \( \zeta \models \mu \) iff \( h(\zeta) \geq 0 \). An SITL formula is thus a predicate formula over predicates. The SITL syntax is hence the same as for MITL formulas, but with predicates \( \mu \) instead of propositions \( p \). Let \( (x, t) \models \phi \) denote that the signal \( x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n \) satisfies \( \phi \) at time \( t \). Let \( (x, t) \models \mu \) iff \( h(x(t)) \geq 0 \), while the semantics for Boolean and temporal operators are the same as for MITL. An SITL formula \( \phi \) is satisfiable if there exists \( x \in \mathbb{R}^n \) such that \( (x, 0) \models \phi \).

The symbols \( \varphi \) and \( \phi \) are used to distinguish between MITL and SITL formulas, respectively. We will consider, in particular, an SITL formula \( \phi \) that consists of the predicates \( \mu_i \in M \) with \( i \in \{1, \ldots, |M|\} \) and abstract \( \phi \), in a first step, into an MITL formula \( \varphi \) as follows. Associate with each predicate \( \mu_i \in M \) a proposition \( p_i \) and let \( P := \{p_1, \ldots, p_{|M|}\} \). Let then \( \varphi := Pr(\phi) \) be an MITL formula that is obtained by replacing each predicate \( \mu_i \in M \) with a proposition \( p_i \in P \), e.g., \( \phi := F_t(\mu_1 \land \mu_2) \) becomes \( \varphi := Pr(\phi) = F_t(p_1 \land p_2) \). This way, spatial properties of \( \phi \) are neglected in \( \varphi \). Conversely, let \( Pr^{-1}(\varphi) = Pr^{-1}(Pr(\phi)) = \phi \) be obtained by replacing each proposition \( p_i \in P \) in \( \varphi \) with the corresponding predicate \( \mu_i \in M \).

B. MITL to Timed Signal Transducer

An MITL formula \( \varphi \) can be translated into a language equivalent timed signal transducer [6] by means of a simple compositional procedure, as summarized next. Let \( c := [c_1, \ldots, c_O] \in \mathbb{R}_{\geq 0}^{O} \) be a vector of \( O \) clock variables that obey the continuous dynamics \( \dot{c}_o(t) := 1 \) with \( c_o(0) := 0 \) for \( o \in \{1, \ldots, O\} \). Discrete dynamics occur at instantaneous times in form of clock resets. Let \( R : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}^{O} \) be a reset function such that \( R(c) = c' \) where either \( c_0' = c_0 \) or \( c_0' = 0 \). With a slight abuse of notation, we also use \( R(c_0) = c_0 \) and \( R(c_0) = 0 \). Clocks evolve with time when visiting a state of a timed signal transducer, while clocks may be reset during transitions between states. We further define clock constraints as Boolean combinations of conditions of the form \( c_o \leq k \) and \( c_o \geq k \) for some \( k \in \mathbb{Q}_{\geq 0} \).

Definition 1 (Timed Signal Transducer [6]):

A timed signal transducer is a tuple \( TST := (S, s_0, \Delta, \Gamma, c, t, \Delta, \lambda, \gamma, \mathcal{F}) \) where \( S \) is a finite set of locations, \( s_0 \) with \( s_0 \cap S = \emptyset \) is the initial state, \( \Delta \) and \( \Gamma \) are finite sets of input and output variables, respectively, \( t : S \rightarrow \Phi(c) \) assigns clock constraints over \( c \) to each location, \( \Delta \) is a transition relation so that \( \delta = (s, g, R, s') \in \Delta \) indicates a transition from \( s \in S \cup s_0 \) to \( s' \in S \) satisfying the guard constraint \( g \in \Phi(c) \) and resetting the clocks according to \( R : \mathcal{S} \cup \mathcal{S}_0 \rightarrow \mathcal{B}(\Delta) \) and \( \gamma : \mathcal{S} \cup \mathcal{S}_0 \rightarrow \mathcal{B}(\Gamma) \) are input and output labeling functions where \( \mathcal{B}(\Delta) \) and \( \mathcal{B}(\Gamma) \) denote the sets of all Boolean combinations over \( \Delta \) and \( \Gamma \), respectively, and \( \mathcal{F} \subseteq 2^\mathcal{S} \Delta = \) a generalized Büchi acceptance condition.

A run of a \( TST \) over an input signal \( d : \mathbb{R}_{\geq 0} \rightarrow \mathbb{B}^{|A|} \) is an alternation of time and discrete steps resulting in an output signal \( y : \mathbb{R}_{\geq 0} \rightarrow \mathbb{B}^{|A|} \). A time step of duration \( \tau \in \mathbb{R}_{\geq 0} \) is denoted by \((s, c(t)) \sim (s, c(t) + \tau) \) with \( d(t + \tau') \models x_1(\tau) \) and \( y(t + \tau') \models x_2(\tau) \) for each \( \tau' \in (0, \tau) \). A discrete step at time \( t \) is denoted by \((s, c(t)) \Delta (s', R(c(t))) \) for some transition \( \delta = (s, g, R, s') \in \Delta \) such that \( d(t) \models x_1(\delta) \) and \( y(t) \models x_2(\delta) \) and \( c(t) = g \). Each run starts with a discrete step from the initial configuration \((s_0, c(0))\). Formally, a run of a \( TST \) over \( d \) is a sequence
(s_0, c(0)) \xrightarrow{\delta_0} (s_1, R_0(c(0))) \xrightarrow{\tau_1} (s_1, R_0(c(0)) + \tau_1) \xrightarrow{\delta_1} \ldots.

Due to the alternation of time and discrete steps, the signals d(t) and y(t) may be a concatenation of sequences consisting of points and open intervals. Zeno signals are excluded by assumption [6]. We associate a function q : \mathbb{R}_{\geq 0} \rightarrow S \cup \Delta with a run as q(0) := \delta_0, q(t) = s_1 for all t \in (0, \tau_1), \ldots ; F is a generalized Büchi acceptance condition so that a run over d(t) is accepting if, for each F \in \mathcal{F}, inf(q) \cap F \neq \emptyset where inf(q) contains the states in S that are visited, in q, for an unbounded time duration and transitions in \Delta that are taken, in q, infinitely many times.

We define the language of TST to be L(TST) := \{d : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^{\Delta} | TST has an accepting run over d(t)\}.

Definition 2 (Synchronous Product [6]): Given TST_i := (S_i, s_0, i, \Lambda_i, \Gamma_i, \varepsilon_i, \iota_i, \Delta_i, \lambda_i, \gamma_i, F_i) with i \in \{1, 2\}, their synchronous product is TST := (S, s_0, \Lambda, \Gamma, \varepsilon, \iota, \Delta, \lambda, \gamma, F) with S := S_1 \times S_2, s_0 := s_0_1 \times s_0_2, \Lambda := \Gamma_1 \cup \Gamma_2, \Gamma := \Gamma_1 \times \Gamma_2, \varepsilon := [c_1^T, c_2^T]^T, and (s_1, s_2) := (t_1(s_1) \land t_2(s_2), t_1(s_1), t_2(s_2)).

The transition relation \Delta is defined as

- (s_1, s_2, g, R, (s'_1, s'_2)) \in \Delta where (s_1, g_1, R_1, s'_1) \in \Delta_1, (s_2, g_2, R_2, s'_2) \in \Delta_2, g := g_1 \land g_2, and R := [R_1^T \ R_2^T]^T (simultaneous transitions),
- (s_1, s_2, g_1 \lor t_2(s_2), R_1, (s'_1, s'_2)) \in \Delta where (s_1, g_1, R_1, s'_1) \in \Delta_1 (left-sided transitions),
- (s_1, s_2, t_1(s_1) \land g_2, R_2, (s'_1, s'_2)) := (s_1, s_2, g_2, R_2, (s'_1, s'_2)) \in \Delta_2 (right-sided transitions),

while the output labeling function is constructed the same way as the input labeling function. The Büchi acceptance condition is F := \{F_1 \times (S_2 \cup \Delta_2), (S_1 \cup \Delta_1) \times F_2\}.

Definition 3 (Input-Output Composition [6]): Given TST_i := (S_i, s_0, i, \Lambda_i, \Gamma_i, \varepsilon_i, \iota_i, \Delta_i, \lambda_i, \gamma_i, F_i) with i \in \{1, 2\}, the input-output composition where the output of TST_1 is the input of TST_2 is TST := (S, s_0, \Lambda, \Gamma, \varepsilon, \iota, \Delta, \lambda, \gamma, F) with S := \{(s_1, s_2) : s_1 \in S_1 \times S_2 | \text{if } d \models \gamma_1(s_1) \text{ implies } d \models \gamma_2(s_2), \Lambda := \Delta_1 \cup \Delta_2, \Gamma := \Gamma_1 \cup \Gamma_2, s_0 := s_0_1 \times s_0_2, c := c_1 \cup c_2, and F := \text{as defined in the synchronous product}.

The transition relation \Delta is defined as

- (s_1, s_2, g, R, (s'_1, s'_2)) \in \Delta where \delta_1 := (s_1, g_1, R_1, s'_1) \in \Delta_1, \delta_2 := (s_2, g_2, R_2, s'_2) \in \Delta_2, g := g_1 \land g_2, and R := [R_1^T \ R_2^T]^T if d \models \gamma_1(\delta_1) \text{ implies } d \models \lambda_2(\delta_2) (simultaneous transitions),
- (s_1, s_2, g_1 \lor t_2(s_2), R_1, (s'_1, s'_2)) \in \Delta where \delta_1 := (s_1, g_1, R_1, s'_1) \in \Delta_1 (left-sided transitions),
- (s_1, s_2, t_1(s_1) \land g_2, R_2, (s'_1, s'_2)) \in \Delta where \delta_2 := (s_2, g_2, R_2, s'_2) \in \Delta_2 if d \models \gamma_1(s_1) \text{ implies } d \models \lambda_2(\delta_2) (right-sided transitions),

while the output labeling function is constructed the same way as the input labeling function. The Büchi acceptance condition is F := \{F_1 \times (S_2 \cup \Delta_2), (S_1 \cup \Delta_1) \times F_2\}.

C. Problem Formulation

Consider a dynamical system as given by

\[ x(t) = f(x(t)) + g(x(t))u(t), \quad x(0) := x_0 \]  \tag{1}

with u(t) \in \mathbb{R}^m and x(t) \in \mathbb{R}^n. The functions f : \mathbb{R}^n \rightarrow \mathbb{R}^n and g : \mathbb{R}^n \rightarrow \mathbb{R}^n \times \mathbb{R}^m are locally Lipschitz continuous.

Problem 1: Assume that (1) is subject to an SITL task \varphi. Derive a control law u(x, t) so that (x, 0) \models \varphi where x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n is the solution to (1) under u(x, t).

III. PLANNING AND CONTROL APPROACH

We first abstract the SITL formula \varphi into the MITL formula \varphi := Pr(\varphi). In Section III-A, we modify TST_{\varphi} to account for the error induced by neglecting predicates of \varphi in \varphi. Based on this modified TST_{\varphi}, denoted by TST_{\varphi}, and without considering the dynamics in (1), we find high-level plans d_{\mu} : \mathbb{R}_{\geq 0} \rightarrow BC(M) (formally defined below) that characterize all signals x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n such that (x, 0) \models \varphi. In Section III-B, we abstract (1) into a timed signal transducer TST_{\varphi} that can be used to check if d_{\mu}(t) can be executed by (1). This abstraction is based on the assumption of existing logic-based feedback control laws, such as presented in our works [21, 22]. In Section III-C,
we modify $TST_\varphi$ into $TST_{\varphi}^\text{m}$ to ensure that $d_\mu : \mathbb{R}_{\geq 0} \rightarrow BC(M)$, now found from $TST_{\varphi}^\text{m}$, can be executed by (1).

A. Checking Satisfiability of Signal Interval Temporal Logic

Let $TST_\varphi := (S, s_0, \Lambda, \Gamma, c, e, \Delta, \lambda, \gamma, F)$ be constructed for $\varphi$ according to Section II-B with $\Lambda := P$ and $P$ coming from the $Pr(\phi)$ transformation. Since we ultimately aim at satisfying $\phi$, we modify $TST_\varphi$ by the following operations.

[O1] Remove each state $s \in S$ for which there is no $x \in \mathbb{R}^n$ so that $x \models Pr^{-1}(\lambda(s))$. Remove the corresponding state from $\mathcal{F}$. Further remove the corresponding ingoing $((s', g, R, s) \in \Delta$ for some $s' \in S$) and outgoing $((s, g, R, s') \in \Delta$ for some $s' \in S$) transitions.

[O2] Remove each transition $\delta := (s, g, R, s') \in \Delta$ for which there is no $x \in \mathbb{R}^n$ so that $x \models Pr^{-1}(\lambda(\delta))$. Remove the corresponding state from $\mathcal{F}$. Note that, by employing techniques such as reported in [23, Ch. 2], a feasibility problem can be solved to check whether or not there exists $x \in \mathbb{R}^n$ such that $x \models Pr^{-1}(\lambda(s))$ and $x \models Pr^{-1}(\lambda(\delta))$ in [O1] and [O2], respectively. By these operations, we account for predicate dependencies although the planning is performed using propositions. The modified $TST_\varphi$ is denoted by $TST_\varphi^\text{m} := (S^\varphi, s_0, \Lambda, \Gamma, c, e, \Delta^\varphi, \lambda, \gamma, F^\varphi)$ for which naturally $S^\varphi \subseteq S$, $\Delta^\varphi \subseteq \Delta$, and $F^\varphi \subseteq \mathcal{F}$. The high-level plan synthesis is based on $TST_\varphi$ and the fact that we can translate $TST_\varphi^\text{m}$, which is in essence a timed automaton [3] when removing the output labels, to a region automaton $RA(TST_\varphi)$; $RA(TST_\varphi)$ can be used to check emptiness of $TST_\varphi$, i.e., to analyze reachability properties of $TST_\varphi$. Since $TST_\varphi$ has been found from $TST_\varphi^\text{m}$, now found from $TST_\varphi$, we modify $TST_\varphi$ by the following operations.

[O1] Remove each state $s \in S$ for which there is no $x \in \mathbb{R}^n$ so that $x \models Pr^{-1}(\lambda(s))$. Remove the corresponding state from $\mathcal{F}$. Further remove the corresponding ingoing $((s', g, R, s) \in \Delta$ for some $s' \in S$) and outgoing $((s, g, R, s') \in \Delta$ for some $s' \in S$) transitions.

[O2] Remove each transition $\delta := (s, g, R, s') \in \Delta$ for which there is no $x \in \mathbb{R}^n$ so that $x \models Pr^{-1}(\lambda(\delta))$. Remove the corresponding state from $\mathcal{F}$.

Fig. 1: Timed signal transducers for temporal and Boolean operators and an example formula tree.
A is the set of all clock regions so that \( Q := S^\phi \times A \).

- The initial states are \( q_0 := (s_0, \alpha_0) \in Q \) where \( \alpha_0 \) is the clock region corresponding to \( \epsilon(0) \).
- There is a transition \( (s, \alpha, \delta) \Rightarrow_R (s', \alpha') \) if and only if there is a transition \( (s, c, \delta) \Rightarrow (s', c') \) for \( c \in \alpha \) and \( c' \in \alpha' \).
- \( (s, \alpha) \in \mathcal{F}_R \) if \( s \in \mathcal{F}^\phi \).

Using standard graph search techniques such as the memory efficient variant of the nested depth first search [24], here adapted to deal with the generalized Büchi acceptance condition as in [25], we may obtain, if existent, sequences \( \bar{s} = ((s_0, \alpha_0), (s_1, \alpha_1), \ldots) \) with \( (s_j, \alpha_j, \delta_j) \Rightarrow_R (s_{j+1}, \alpha_{j+1}) \) for each \( j \in \mathbb{N} \) satisfying the generalized Büchi acceptance condition \( \mathcal{F}_R \). In particular, \( \bar{s} := (s_{p}, \bar{s}_p) \) consists of a prefix of length \( p + 1 \) and a suffix of length \( s \), here denoted by \( s_p := ((s_0, \alpha_0), \ldots, (s_p, \alpha_p)) \) and \( \bar{s}_p := (s_{p+1}, \alpha_{p+1}), \ldots, (s_{p+s}, \alpha_{p+s}) \).

Furthermore, we require that \( \gamma(\delta_0) = y \) to indicate that we want \( (d, 0) \models \phi \), opposed to \( \gamma(\delta_0) = -y \) indicating \( (d, 0) \not\models \phi \). What remains to be done is to add timings \( \bar{\tau} := (\bar{\tau}_p, \bar{\tau}_s) \) to \( \bar{s} \) with, similarly to \( s_p \) and \( \bar{s}_p \), \( \bar{\tau}_p := (\tau_0 := 0, \ldots, \tau_p) \) and \( \bar{\tau}_s := (\bar{\tau}_{p+1}, \ldots, \tau_{p+s}) \) where \( \tau_j \in \mathbb{R}_0^+ \) for \( j \geq 1 \) corresponds to the occurrence of \( \delta_j \), which happens \( \tau_j \) time units after the occurrence of \( \delta_{j-1} \). The proof of [3, Lemma 4.13] proposes a method to find timings for a simple acceptance condition, i.e., only requiring \( \tau_p \), while we deal with a generalized Büchi acceptance condition for which we present a solution in Section III-C and assume, for now, that \( \bar{\tau} \) has been obtained. Such \( \bar{s} \) with \( \bar{\tau} \) can be associated, by denoting \( T_{j} := \sum_{i=0}^{j} \tau_i \), with a high-level plan (later interpreted as \( d_{\mu}(t) \)) as

\[
d_{\mu}(t) := \begin{cases} 
\lambda(\delta_j) & \text{if } T_j = T_j, \\
\lambda(s_j) & \text{if } T_j < T_{j+1}
\end{cases} 
\]  

**Lemma 1:** Assume a signal \( d : \mathbb{R}_0^+ \to \mathbb{B}^{|P|} \). There is an accepting run of \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) over \( d(t) \) and \( (d, 0) \models \phi \) if only if there exists a plan \( d_{\mu}(t) \) so that \( d(t) \models d_{\mu}(t) \) for all \( t \in \mathbb{R}_0^+ \).

**Proof:** \( \Rightarrow \): Departing from \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \), the infinite state transition system \( (S \times \mathbb{R}_0^+, \Rightarrow) \) has, by construction, the same reachable set as \( \mathcal{T} \mathcal{S} \mathcal{F}_\phi \), i.e., the same reachable configurations \( (s, e(0)), (s, R(e(0))), (s, 1, R(e(0)) + \tau_1), \ldots \). Since is a bisimulation relation, reachability properties of \( \mathcal{T} \mathcal{S} \mathcal{F}_\phi \) can then equivalently be analyzed by considering the finite state transition system \( \mathcal{R} \mathcal{A}(\mathcal{T} \mathcal{S} \mathcal{F}_\phi) \) [3, Lemma 4.13]. If there hence exists an accepting run of \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) over \( d(t) \) and \( (d, 0) \models \phi \), i.e., \( \gamma(\delta_0) = y \), the plan \( d_{\mu}(t) \) can be constructed as described above by obtaining \( \bar{s} \) and \( \bar{\tau} \) directly from the accepting run of \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) over \( d(t) \). It will, by construction, hold that \( d(t) \models d_{\mu}(t) \) for all \( t \in \mathbb{R}_0^+ \).

\( \Leftarrow \): Finding accepting runs \( \bar{s} \) of \( \mathcal{R} \mathcal{A}(\mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi) \) using nested depth first search algorithms, and including suitable timings \( \bar{\tau} \), ensures that \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) has an accepting run for an input signal \( d(t) \models d_{\mu}(t) \) for all \( t \in \mathbb{R}_0^+ \). Removing states and transitions from \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) according to operations [O1] and [O2] resulting in \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) only removes behavior from \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) (not adding additional behavior), i.e., \( L(\mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi) \subseteq L(\mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi) \), so that, by [6, Thm. 6.7], an accepting run of \( \mathcal{T} \mathcal{S} \mathcal{T} \mathcal{F}_\phi \) over \( d(t) \)

inducing \( y(0) = \top \) results in \( (d, 0) \models \phi \).
$(\bar{s}, \bar{g}, 0, \bar{s}') \in \bar{Δ}$ where $\bar{g}$ is a guard that depends on (1).
In particular, we assume that $\bar{g}$ encodes intervals of the form $(C', C'')$, $(C', C'')$, $(C', C'')$, or conjunctions of them, where $C', C'' \in \mathbb{Q}_{\geq 0}$ with $C' \leq C''$. There exists a transition $\bar{δ} := (\bar{s}, \bar{g}, 0, \bar{s}') \in \bar{Δ}$ if, for all $\tau > 0$ with $\bar{g}$ and for all $x_0 \in \mathbb{R}^n$ with $x_0 = \bar{λ}(\bar{s})$, there exists a control law $u_\bar{s}(x, t)$ so that the solution $x(t)$ to (1) is such that:

- either, for all $t \in [0, \tau)$, $x(t) = \bar{λ}(\bar{s})$ and $x(\tau) = \bar{λ}(\bar{s}')$
- or, for all $t \in [0, \tau)$, $x(t) = \bar{λ}(\bar{s})$ and $x'(t) > \tau$ such that, for all $t \in (\tau, \tau']$, $x'(t) = \bar{λ}(\bar{s}')$.

We define $\bar{λ}(\bar{δ}) := \bar{λ}(\bar{s}')$ in the former and $\bar{λ}(\bar{δ}) := \bar{λ}(\bar{s})$ in the latter case. Note that $u_{\bar{s}}(x, t)$, achieving such a transition, has to ensure invariance and finite-time reachability properties. If these control laws are according to [21], [22], we emphasize that it is ensured that the solution $x(t)$ to (1) is defined for all $t \in \mathbb{R}_{\geq 0}$: $S_0$ is here a set and consists of all element $\bar{s}_0 \in S$ such that $x_0 = \bar{λ}(\bar{s}_0)$.

We now define a run of $TST_\phi$ slightly different compared to a run of $TST_\phi$. A run of $TST_\phi$ over the input signal $d_\phi : \mathbb{R}_{\geq 0} \to BC(M)$ again consists of an alternation of time and discrete steps $(\bar{s}_0, 0) \xrightarrow{\bar{g}} (\bar{s}_1, 0) \xrightarrow{\bar{g}} (\bar{s}_2, 1) \xrightarrow{\bar{g}} \ldots$. A time step duration $τ$ is denoted by $(\bar{s}, \bar{c}(t)) \xrightarrow{\bar{γ}} (\bar{s}', \bar{τ})$ with $d_\phi(t) + \bar{τ}' = \bar{λ}(\bar{s})$ for each $\bar{τ}' \in (0, \tau)$. A discrete step at time $t$ is denoted by $(\bar{s}, \bar{c}(t)) \xrightarrow{\bar{δ}} (\bar{s}', \bar{τ})$ for some transition $\bar{δ} := (\bar{s}, \bar{γ}, 0, \bar{s}') \in \bar{Δ}$ such that $\bar{c}(t) = \bar{γ}$ and for which $x = \bar{λ}(\bar{δ})$ implies that $x = d_\phi(t)$. If $d_\phi(t)$ does not result in a run of $TST_\phi$ over $d_\phi(t)$, then it can be concluded that (1) cannot execute $d_\phi(t)$.

Proposition: If $d_\phi(t)$ results in a run of $TST_\phi$ over $d_\phi(t)$, then applying $u(x, t)$ to (1) results in $(x, 0) = \phi$. According to Theorem 1, we can infer that $(x, 0) = \phi$.

C. Plan Synthesis for Signal Interval Temporal Logic

Sections III-A and III-B present a way to synthesize $d_\phi(t)$ that can be checked against $TST_\phi$ as in Theorem 2. It may, however, occur that $d_\phi(t)$ does not result in a run of $TST_\phi$ due the system in (1) being unable to follow $d_\phi(t)$. We propose a complete algorithm that avoids a state space explosion that is typically the outcome of forming automata products. This follows since the input label of each state or transition in $TST_\phi$ corresponds to one state in $TST_\phi$, i.e., $TST_\phi$ and $TST_\phi$ align in a way, so that $TST_\phi$ (defined below and corresponding to the product of $TST_\phi$ and $TST_\phi$) has no more states than $TST_\phi$.

Remark 1: The usefulness of avoiding such state explosion is illustrated as follows. If $\varphi$ is built from three elementary signal transducers, e.g., one until and two eventually operators as in Figs. 1a and 1b, $TST_\varphi$ will have $4^3 = 64$ states in the worst case (depending on the operations [O1] and [O2]). Assuming a discrete abstraction DA of (1), such as a weighted transition system [13], with 100 states, e.g., corresponding to a discretization of $\mathbb{R}^n$, the product of $TST_\varphi$ and DA may contain up to 6400 states. The situation gets even worse when forming the region automaton which induces $O(|S| \cdot 2^{C_{\text{max}}})$ states where $S$ are the states of the product automaton and $C_{\text{max}}$ is the maximum clock constant contained in $S$ [3, Thm. 4.16].

Our approach relies on two facts: 1) the removal of states and edges, as presented in [O1] and [O2] and continued below, resulting in $TST_\varphi$ and 2) constraining guards $g$ of transitions in $TST_\varphi$ so that it is possible to determine timings $\bar{τ}$, if possible, for $\bar{s}$ that result in $d_\phi(t)$ being a run of $TST_\phi$. We modify $TST_\varphi$ to account for $TST_\varphi$ as follows.

O3 Remove each transition $\bar{δ} := (\bar{s}, \bar{g}, R, \bar{s}') \in \bar{Δ}$ for which there exists no transition $\bar{δ} := (\bar{s}, \bar{γ}, 0, \bar{s}') \in \bar{Δ}$ with $\bar{λ}(\bar{s}) = P \bar{λ}(\bar{s})$, $\bar{λ}(\bar{s}') = P \bar{λ}(\bar{s}')$, and for which $x = \bar{λ}(\bar{δ})$ implies $x = P \bar{λ}(\bar{δ})$. Remove the corresponding $\bar{δ}$ from $\mathcal{F}^\phi$.

O4 Remove each $d_\phi := (\bar{s}_0, \bar{g}, R, \bar{s}') \in \bar{Δ}$ with $x_0 \neq P \bar{λ}(\bar{s}')$. Remove the corresponding $d_\phi$ from $\mathcal{F}^\phi$.

Denote the obtained states by $S^m, \mathcal{Δ}^m$, and $\mathcal{F}^m$ for which $S^m \subseteq S^\phi, \mathcal{Δ}^m \subseteq \mathcal{Δ}^\phi$, and $\mathcal{F}^m \subseteq \mathcal{F}^\phi$. Operation [O3] removes transitions in $TST_\varphi$ for which there exists no corresponding transition in $TST_\phi$, while operation [O4] takes care of the initial position $x_0$. We further take care of the timings including an additional clock into $TST_\varphi$. Therefore, let $\mathcal{G}^m := [c^T \bar{c}]^T$ and perform the final operation.

O5 For each transition $\mathcal{G}^m := (\bar{s}, \bar{g}, R, \bar{s}') \in \mathcal{Δ}^m$ let $g^m := g \wedge \bar{γ}$ where $\bar{δ} := (\bar{s}, \bar{γ}, 0, \bar{s}') \in \bar{Δ}$ with $\bar{λ}(\bar{s}) = P \bar{λ}(\bar{s})$, $\bar{λ}(\bar{s}') = P \bar{λ}(\bar{s}')$, and for which $x = \bar{λ}(\bar{δ})$ implies $x = P \bar{λ}(\bar{δ})$. Replace $g$ and $R$ in $\mathcal{G}^m$ with $g^m$ and $R^m$, respectively, where $R^m$ is obtained in an obvious manner.

We emphasize that adding $\bar{c}$ and $\bar{g}$ is crucial to ensure correctness. Let the modified timed signal transducer be denoted by $TST^m := (S^m, s_0, \Lambda, \Gamma, \mathcal{E}^m, t, \mathcal{Δ}^m, \lambda, \gamma, \mathcal{F}^m)$ and note that $L(TST^m) \subseteq L(TST_\varphi) \subseteq L(TST_\phi)$. Remark 2: The operations [O3]-[O5] result in the timed signal transducer $TST^m$ that restricts the behavior of $TST_\varphi$ exactly to the behavior allowed by $TST_\phi$ and corresponds hence to a product automaton without exhibiting an exponential state space explosion.

We next explain how to find an accepting plan $d_\phi(t)$ from $TST_\phi$. Similar to Section III-A, we find $\bar{s}$ by a nested depth first search now performed on $RA(TST^m)$. Recall that the nested depth first search provides $\bar{s}$ that consist of a prefix $\bar{s}_p$ and a suffix $\bar{s}_s$. For such a sequence $\bar{s}$, we now determine if suitable timings $\bar{τ}_p$ and $\bar{τ}_s$ can be found so that the resulting $d_\phi(t)$ is accepted by $TST_\phi$. For ease of reading, we use $TST_\phi$ and $TST_\phi$ (and not $TST^m$) in the remainder. Note also that the guards $g$ and invariants $\varphi(s)$ in $TST_\varphi$ are always conjunctions of the form $c_o < C_o$ or $c_o = C_o$ for clocks $o \in \mathbb{Q}$. Section III-A and III-B present a way to synthesize $d_\phi(t)$ that can be checked against $TST_\phi$ as in Theorem 2. It may, however, occur that $d_\phi(t)$ does not result in a run of $TST_\phi$ due to the system in (1) being unable to follow $d_\phi(t)$. We propose a complete algorithm that avoids a state space explosion that is typically the outcome of forming automata products. This follows since the input label of each state or transition in $TST_\phi$ corresponds to one state in $TST_\phi$, i.e., $TST_\phi$ and $TST_\phi$ align in a way, so that $TST^m$ (defined below and corresponding to the product of $TST_\phi$ and $TST_\phi$) has no more states than $TST_\phi$.
where $C_o \in \mathbb{Q}_{\geq 0}$, while the guards $\tilde{g}_j$ in $TSTS$ are, by assumption, always of the form $(C', C''_j)$, $(C', C''_j)$, and $(C', C''_j)$, or conjunctions of them.

**Prefix ($\tau_0 - \tau_p$):** Note first that $\tau_0 := 0$. For $\tau_j$ with $j \in \{1, \ldots, p\}$, the transitions $(s_j, \alpha_j, \delta_j) \Rightarrow R (s_j, \alpha_j + 1)$ have to be considered where $\delta_j := (s_j, \delta_j, 1, \delta_j + 1) \in \Delta$. For each such transition $\delta_j$, let $\tilde{g}_j$ be the corresponding guard in $TSTS$ in accordance with operation [O3], i.e., $\delta_j := (s_j, \delta_j, 1, \delta_j + 1) \in \Delta$ of $TSTS$ with $\lambda(s_j) = Pr(\lambda(\delta_j))$, $\lambda(\delta_j + 1) = Pr(\lambda(\delta_j + 1))$, and for which $x \models \lambda(\delta_j)$ implies $x \models Pr(\lambda(\delta_j))$. Let $N_{j,o}$ with $0 \leq N_{j,o} < 2$ denote the number of preceding transitions in which the clock $c_o$ was not reset, i.e., $R_k(c_o) = c_o$ for all $k \in \{j - N_{j,o}, \ldots, j - 1\}$ and $R_{j-N_{j,o}-1}(c_o) = 0$ if $j - N_{j,o} > 0$. With this definition, let us further define $T_{j,o} := \sum_{k=j-N_{j,o}}^{j-1} \tau_k$ if $N_{j,o} > 0$ and $T_{j,o} := 0$ if $N_{j,o} = 0$. We next consider four cases for determining the timings $\tilde{\tau}_p$.

Case 1) If there exists $o \in \{1, \ldots, O\}$ so that $c_o(\tau_j) \models \tilde{g}_j$ only if $c_o(\tau_j) = C_o$, then it has to hold that

$$\tau_j \in \{C_o - T_{j,o}\} \cap \tilde{g}_j.$$  

(3)

Otherwise, i.e., $c_o(\tau_j) \models \tilde{g}_j$ does not imply that $c_o(\tau_j) = C_o$, partition $\{1, \ldots, O\}$ as $c_o(\tau_j) = C_o$, partition $\{1, \ldots, O\}$ so that $R_k(c_o) = c_o$ for all $o \in O_{NR,j}$, and $R_{\lambda}(c_o) = c_o$ for all $o \in O_{NR,j}$. Let $\tilde{c}_{j,o}$ and $\tilde{c}_{j,o}$ be the oth elements of $\tilde{c}_j := \text{argsup}_{e \in \alpha_j+1} \|e\|$ and $\tilde{c}_j := \text{arginf}_{e \in \alpha_j+1} \|e\|$, respectively.

Case 2) If not Case 1 and, for some $o \in O_{NR,j}$, we have $\tilde{c}_{j,o} = \tilde{c}_{j,o}$, then, for $o \in O_{NR,j}$, it has to hold that

$$\tau_j \in \{\tilde{c}_{j,o} - T_{j,o}\} \cap \tilde{g}_j.$$  

(4)

Case 3) If not Cases 1 and 2, then, for $o \in O_{NR,j}$, let $\delta_{\lambda,o} := \tilde{c}_{j,o} - T_{j,o}$, and $\delta''_{\lambda,o} := \tilde{c}_{j,o} - T_{j,o}$.

Case 4) If not Cases 1 and 2, then, for $o \in O_{NR,j}$, let $\delta_{\lambda,o} := 0$ and $\delta''_{\lambda,o} := C_o - T_{j,o}$.

For Cases 3 and 4, we then require that

$$\tau_j \in \left( \max_{o \in \{1, \ldots, O\}} \delta_{\lambda,o} + \epsilon, \min_{o \in \{1, \ldots, O\}} \delta''_{\lambda,o} \right) \cap \tilde{g}_j$$

(5)

where $\epsilon > 0$ avoids Zeno signals and guarantees progressive runs [3]. We see that (3)-(5) are constrained, in a similar way, by $\tilde{g}_j$ which exactly corresponds to operation [O5].

**Suffix ($\tau_{p+1} - \tau_p$):** The suffix can be found in a similar way as the prefix, i.e., considering Cases 1-4. We only need to add a lasso shape condition. In other words, we find $\tau_{p+1}$ until $\tau_{p+s}$ as described in Steps 1-4, but now additionally requiring that, for $o \in O_{NR,p+1}$,

$$T_{p,o} = T_{p+s,o}.$$  

(6)

To obtain $\tilde{\tau}$ and check if there exists a $\tilde{\tau}$ corresponding to $\tilde{s}$, consider the following optimization problem.

$$\arg \min_{\tau \in \mathbb{R}^{p+1}, \tau \geq e} \left( \begin{array}{c}
\tau_0 := 0, e > 0 \\
\tau_j \text{ for } j \in \{1, \ldots, p - 1\} \text{ according to (3)-(5)} \\
T_{p,o} = T_{p+s,o} \text{ for } o \in O_{NR,p+1} \text{ according to (6).}
\end{array} \right.$$  

(7d)

**Corollary 1:** The optimization problem in (7) is a linear and hence convex optimization problem.

**Proof:** Note that Cases 1-4 can not happen simultaneously. The constraint in (3) can be written as $\tau_j = C_o - T_{j,o}$ and $\tau_j = \tilde{g}_j$. The latter constraint can be written into separate linear constraints using the constants $C'_j$ and $C''_j$ associated with $\tilde{g}_j$. Note that $T_{j,o}$ is a linear combination of $\tau_j$. Hence (3) is a linear constraint in $\tilde{\tau}$. It is straightforward to show the same for (4) and (6). The constraint (5) can be written into constraints $\tau_j > \delta_{\lambda,o} + \epsilon$ and $\tau_j < \delta''_{\lambda,o}$ for each $o \in \{1, \ldots, O\}$ where $\delta_{\lambda,o}$ and $\delta''_{\lambda,o}$ are again linear in $\tilde{\tau}$. It can also be seen that (5) is linear in $\epsilon$. The optimization problem in (7) is hence linear and thus convex.

In particular that (7) is always feasible. With $\tilde{s}$ and $\tilde{\tau}$ as obtained above, we can then synthesize $d_p(t)$ as in (2).

**Theorem 3:** The proposed method is sound. Given a timed abstraction $TSTS$, the proposed method is also complete.

**Proof:** Regarding soundness. If the nested depth first search finds $\tilde{s}$ and $\tilde{\tau}$ is obtained from (7), then $d_p(t)$ results in a run of $TSTS$ over $d_p(t)$ due to operations [O3][O5] performed on $TSTS$ resulting in $TSTS^m$. Applying $u(x,t)$ to (1) then results in $\phi$ according to Theorem 2. Note that Theorems 1 and 2 hold even when $d_p(t)$ is obtained from $TSTS^m$ instead of $TSTS$ since $L(TSTS^m) \subseteq L(TSTS)$.

Regarding completeness. The proof of Lemma 2 shows completeness for plans found from $TSTS$, but without considering $TSTS$; $TSTS^m$ restricts the language of $TSTS$ and only removing behavior that $TSTS$ can not execute. Hence we can find a plan $d_p(t)$ from $TSTS^m$ if there exists a plan $d_p(t)$ that is accepted by $TSTS$.

Note that Theorem 3 guarantees completeness on the planning level, i.e., when given an abstraction $TSTS$.

**IV. SIMULATIONS**

We consider an academic example that is easy to follow, yet rich enough to illustrate the theoretical findings of this paper. Consider a system consisting of $x := [x_1^T \ x_2^T]^T \in \mathbb{R}^4$, e.g., a system consisting of two robots. The STIL formula is $\phi := (\mu_1 U_{\infty} \mu_2) \wedge F_{\infty}(x_3) \wedge H_{\infty}(x_3) \mu_4$ with predicate functions $h_1(x) := -\|x_1 - x_2 - f_A\|$, $h_2(x) := -\|x_1 - x_2 - f_A\|$, $h_3(x) := -\|x_1 - x_2 - f_B\|$, and $h_4(x) := -\|x_1 - x_2 - f_B\|$ and it holds that $x_0 = \mu_1$, while $x_0 \neq \mu_2$, $x_0 \neq \mu_3$, and $x_0 \neq \mu_4$ (important for operation [O4]). Let $\epsilon := 0.25$ and $f_A := [-0.5 \ 0.5]^T$ and $f_B := [-0.5 \ 0.2]^T$ so that $\mu_1$ and $\mu_4$ encode formations between the robots. Let further $p_A := [1 \ 1]^T$ and $p_B := [-1 \ 1]^T$ so that $\mu_2$ and $\mu_3$ encode reachability specifications of robots 1 and 2, respectively. Note that there is no $x \in \mathbb{R}^4$ so that $x = (\mu_2 \wedge \mu_3) \wedge (\mu_1 \wedge \mu_4)$ (important for operations [O1] and [O2]). The corresponding MITL formula $\phi := PR^{-1}(\phi) := (p_1 U_{\infty} \mu_2) \wedge F_{\infty}(x_3) \mu_3 \wedge F_{\infty}(x_1) \mu_4$ was translated to $TSTS$ resulting in 65 states. We assume the dynamics $x = f(x) + u$ where $f(x)$ may be unknown and consider, for instance, control laws as derived in [21]. These control laws can achieve invariance and finite time reachability specifications. In other words, there exists control laws $u^\phi(x,t)$ that can satisfy STIL formulas such as $G_{(0,\epsilon)} \mu_{\text{inv}} \wedge F_{(0,\epsilon)} \mu_{\text{reach}}$
in case that the predicate function associated with $\mu_{\text{env}}$ and $\mu_{\text{each}}$ are concave and satisfiable, which is the case for conjunctions of $\mu_1 \land \mu_2$. We also assume, for simplicity, that each possible transition $\delta$ can be made within $1$-$4$ time units, i.e., $C' := 1$ and $C'' := 4$ so that $b \in \{C', C''\}$ (important for operation $[05]$). $TST_\varphi$ was then transformed into $TST_\varphi^*$ which was again transformed into $TST_\varphi^*$ by performing operations $[01]$-$[05]$. Based on this, $RA(TST_\varphi^*)$ was obtained with, in total, $2723$ states. To illustrate that our method deals with spatiotemporal specifications, we compare the nested depth first search of $RA(TST_\varphi^*)$ with a nested depth first search performed on $RA(TST_\varphi^*)$. For $RA(TST_\varphi^*)$, a sequence $s^\varphi := ((s^\varphi_0, \alpha^\varphi_0), (s^\varphi_1, \alpha^\varphi_1), \ldots)$ is obtained with $\lambda(s^\varphi_0) = p_1 \land \neg p_2 \land p_3 \land \neg p_4$ and $\lambda(s^\varphi_1) = p_1 \land p_2 \land p_3 \land p_4$ for which there exist no $x \in \mathbb{R}^n$ so that $x = Pr^{-1}(\lambda(s^\varphi_0))$. Our method, however, finds a sequence $s := ((s_0, 0), (s_1, 0), \ldots)$ from $RA(TST_\varphi^*)$ with $\lambda(s_0) = p_1 \land \neg p_2 \land p_3 \land \neg p_4$, $\lambda(s_1) = p_1 \land p_2 \land p_3 \land \neg p_4$, and $\lambda(s_2) = p_1 \land p_3 \land p_4$ accounting for the spatial properties induced by the predicates, i.e., for each $\lambda(s_0), \lambda(s_1), \lambda(s_2)$, there exists $x \in \mathbb{R}^n$ so that $x = Pr^{-1}(\lambda(s_0))$, $x = Pr^{-1}(\lambda(s_1))$, and $x = Pr^{-1}(\lambda(s_2))$, respectively. A timing sequence $\tau$ is obtained with $\tau := (0, 1, 1, 1, \ldots)$ defining the plan $d_\mu(t)$ that can be implemented as stated in Theorem 3, resulting in $(x, 0) = \phi$ as shown in Fig. 2.

V. Conclusion

This paper presents an efficient automata-based planning and control framework for spatio-temporal logics, here in particular signal interval temporal logic. Results from automata-based verification for metric interval temporal logic have been leveraged to account for the spatial properties induced by the signal interval temporal logic specification at hand. Furthermore, the state explosion, typically induced by forming a product automaton between the specification automaton and an abstraction of the system, is avoided. For future work, we will consider the robust semantics as well as uncontrollable events within the planning framework.

REFERENCES