Bounded Timed Propositional Temporal Logic with Past Captures Timeline-based Planning with Bounded Constraints

Dario Della Monica[*][†], **Nicola Gigante**[‡], **Angelo Montanari**[‡], **Pietro Sala**[§], **Guido Sciavicco**[¶]
[*] Universidad Complutense de Madrid, Spain
[†] Università di Napoli, Italy
[‡] Università di Udine, Italy
[§] Università di Verona, Italy
[¶] Università di Ferrara, Italy

Abstract

Within the timeline-based framework, planning problems are modeled as sets of independent, but interacting, components whose behavior over time is described by a set of temporal constraints. Timeline-based planning is being used successfully in a number of complex tasks, but its theoretical properties are not so well studied. In particular, while it is known that Linear Temporal Logic (LTL) can capture classical action-based planning, a similar logical characterization was not available for timeline-based planning formalisms. This paper shows that timeline-based planning with bounded temporal constraints can be captured by a bounded version of *Timed Propositional Temporal Logic*, augmented with past operators, which is an extension of LTL originally designed for the verification of real-time systems. As a byproduct, we get that the proposed logic is expressive enough to capture temporal action-based planning problems.

1 Introduction

Most of the languages used to describe *automated planning* problems, as, for instance, PDDL [Fox and Long, 2003; Gerevini *et al.*, 2009], follow an *action-based* paradigm, focusing on which *actions* can be performed by an executor in order to achieve its goal. On the other hand, in *timeline-based* planning, problems are described as a set of independent, but interacting, components, whose behavior over time (the timelines) is governed by a set of temporal constraints. This more declarative approach turns out to be useful in describing and reasoning about problems dealing with a high number of interacting components. Moreover, the concept of *flexible* timeline allows one to represent uncertainty in the duration of tasks, as well as *uncontrollable* components modeling the behavior of the surrounding environment. Since its introduction at NASA [Muscettola, 1994] in the context of planning of space operations, the timeline-based approach has been adopted by a number of planning systems, like EUROPA [Barreiro *et al.*, 2012], ASPEN [Chien *et al.*, 2000] and APSI-TRF [Cesta *et al.*, 2009], and successfully employed in a variety of complex tasks and missions [Jónsson *et al.*, 2000; Cesta *et al.*, 2007; Cesta *et al.*, 2010].

Action-based planning has been extensively studied from a theoretical point of view. *Classical* planning is known to be PSPACE-complete [Bylander, 1994], while *temporal planning*, where actions have explicit durations, is EXPSPACE-complete [Rintanen, 2007]. Logical characterizations of action-based planning have been provided in [Cialdea Mayer *et al.*, 2007; Cimatti *et al.*, 2017], encoding the problem into Linear Temporal Logic (LTL) [Pnueli, 1977] or its variations.

Little is known about the theoretical properties of timeline-based planning problems. In particular, a complete picture of computational complexity and expressiveness of timeline-based planning languages is missing. A formalization of the problem, including *flexible* timelines and *uncontrollable* components, can be found in [Cimatti *et al.*, 2013; Cialdea Mayer *et al.*, 2014; Cialdea Mayer *et al.*, 2016]. In [Gigante *et al.*, 2016], Gigante *et al.* isolated a fragment of *non-flexible* timeline-based planning, which forbids the use of *unbounded* interval relations and is expressive enough to capture *temporal* action-based planning, and proved that it is EXPSPACE-complete. A logical characterization of timeline-based planning is missing. Besides being a very natural question by itself, it would allow one to approach the problem of the *controllability* of problems with uncontrollable components in terms of logical *synthesis* [Schewe and Tian, 2011].

Timed Propositional Temporal Logic (TPTL) is an extension of LTL aimed at the verification of reactive systems with *real-time* constraints [Alur and Henzinger, 1994]. It extends LTL with a *freeze quantifier* that allows one to give a name to the timestamp of a given state, and then to employ it in *timing constraints*. The satisfiability checking problem for TPTL is EXPSPACE-complete, which makes it a good candidate for capturing timeline-based planning; unfortunately, it lacks the necessary *past* operators, and the extension of TPTL with past (TPTL+P) is non-elementary [Alur and Henzinger, 1993].

In this paper, we give a logical characterization of the fragment of *non-flexible* timeline-based planning studied in [Gigante *et al.*, 2016], by showing that it can be naturally captured by a guarded fragment of TPTL+P which imposes an

✉ gigante.nicola@spes.uniud.it
This work was partially supported by the Italian INdAM GNCS project *Logics and Automata for Interval Model Checking*. In addition, Dario Della Monica acknowledges the financial support from a Marie Curie INdAM-COFUND-2012 Outgoing Fellowship.

exponential upper bound to the scope of temporal operators when applied to formulae with free variables. We show that this restriction is enough to keep the satisfiability checking problem for the resulting logic EXPSPACE-complete. By exploiting the expressiveness result of [Gigante *et al.*, 2016], the result directly transfers to *temporal* action-based planning.

The rest of the paper is organized as follows. In Sect. 2, we introduce the timeline-based planning problem. Then, in Sect. 3 we show that the considered fragment of TPTL with past is sufficient to capture it. Next, in Sect. 4 we determine the complexity of the logic. We conclude the paper by discussing future developments.

2 Timeline-based Planning

This section introduces notation and terminology of timeline-based planning, mostly borrowed from [Gigante *et al.*, 2016] and extensively discussed in [Cialdea Mayer *et al.*, 2016].

Definition 1 (State variable). *A* state variable *x is a triple (V_x, T_x, D_x), where:*

- *V_x is the* finite domain *of the variable x;*
- *$T_x : V_x \to 2^{V_x}$ is the* value transition function*, which maps each value $v \in V_x$ to the set of values that x can take immediately after v;*
- *$D_x : V_x \to \mathbb{N} \times \mathbb{N}$ is a function that maps each $v \in V_x$ to a pair (d_{min}, d_{max}), with $d_{min} \leq d_{max}$, where d_{min} and d_{max} are respectively the minimum and maximum duration of an interval over which x takes value v.*

Which value is taken by a state variable over a specified time interval is described by means of *tokens*.

Definition 2 (Token). *A* token *for x is a tuple $\tau = (x, v, d)$, where $x = (V_x, T_x, D_x)$ is a state variable, $v \in V_x$, and $d \in \mathbb{N}$ is the* duration *of the token, with $d_{min} \leq d \leq d_{max}$, and $D_x(v) = (d_{min}, d_{max})$.*

The time-varying behavior of a state variable is represented by means of a finite sequence of tokens, called a *timeline*.

Definition 3 (Timeline). *A* timeline *for a state variable $x = (V_x, T_x, D_x)$ is a non-empty finite sequence $\mathsf{T} = \langle \tau_1, \ldots, \tau_k \rangle$ of tokens for x, where $v_{i+1} \in T_x(v_i)$ for $i \in \{1, \ldots, k-1\}$.*

Notice that the values of x in two consecutive tokens do not need to be different. A time interval can be associated with any token $\tau_i = (x, v_i, d_i)$ in a timeline $\mathsf{T} = \langle \tau_1, \ldots, \tau_k \rangle$ by means of the functions $\mathsf{start\_time}(\tau_i) = \sum_{j=1}^{i-1} d_j$ and $\mathsf{end\_time}(\tau_i) = \mathsf{start\_time}(\tau_i) + d_i$. The end time of the last token of a timeline is called the *horizon* of the timeline. In the following, when there is no ambiguity, we will interchangeably refer to a token and to the associated time interval.

The behavior of state variables is constrained by a set of *synchronization rules*, which relate tokens, possibly belonging to different timelines, through temporal relations among intervals or among intervals and time points. To express these temporal constraints, we adopt the compact notation proposed in [Gigante *et al.*, 2016]. Let $\Sigma = \{a, b, c, \ldots\}$ be a set of *token names* used to refer to tokens.

Definition 4 (Atoms). *An* atom *is either a clause of the form $a \leq^{e_1,e_2}_{[l,u]} b$ (interval), or of the form $a \leq^{e_1}_{[l,u]} t$ or $a \geq^{e_1}_{[l,u]} t$ (time-point), where $a, b \in \Sigma$, $l, u, t \in \mathbb{N}$, and e_1, e_2 are either* $\mathsf{start\_time}$ *or* $\mathsf{end\_time}$*, respectively* s *and* e *for short.*

As an example, the atom $a \leq^{\mathsf{s},\mathsf{e}}_{[l,u]} b$ constrains the token a to *start* before the *end* of b, and the distance between the two related endpoints to be between l and u.

Definition 5 (Synchronization rules). *Let SV be a set of state variables. An* existential statement *is a statement of the form:*

$$\exists a_1[x_1 = v_1] \ldots a_n[x_n = v_n] \; . \; \mathcal{C}$$

where $\mathcal{C} \equiv \rho_0 \wedge \ldots \wedge \rho_m$ is a conjunction of atoms, $x_i \in SV$, $a_i \in \Sigma$, and $v_i \in V_{x_i}$ for each $i \in \{1, \ldots, n\}$. The clauses $a_i[x_i = v_i]$ are called quantifiers*. A token name used in $\mathcal{C}$, but not occurring in any quantifier, is said to be* free*. A* synchronization rule *$\mathcal{R}$ is a clause in one of the following forms:*

$$a_0[x_0 = v_0] \longrightarrow \mathcal{E}_1 \vee \mathcal{E}_2 \vee \ldots \vee \mathcal{E}_k$$
$$\top \longrightarrow \mathcal{E}_1 \vee \mathcal{E}_2 \vee \ldots \vee \mathcal{E}_k$$

where $a_0 \in \Sigma$, $x_0 \in SV$, $v_0 \in V_{x_0}$, and $\mathcal{E}_1, \ldots, \mathcal{E}_k$ are existential statements *where only a_0 may appear free. In rules of the first form, the quantifier $a_0[x_0 = v_0]$ is called* trigger*. Rules of the second form are said to be* trigger-less.

Intuitively, the trigger is a universal quantifier, which says that *for all* the tokens a_0, where the variable x_0 takes the value v_0, at least one of the existential statements $\mathcal{E}_i$ must be true. The existential statements in turn assert the existence of tokens $a_1, \ldots, a_n$, where the respective state variables take the specified values, that satisfy the temporal constraints given by $\mathcal{C}$. Trigger-less rules simply assert the satisfaction of the existential statements.

Definition 6 (Semantics of atoms). *Given a set of tokens Γ and a function $\lambda : \Sigma \to \Gamma$ that assigns a token to each token name, an interval atom $a \leq^{e_1,e_2}_{l,u} b$ is* satisfied *by λ if $l \leq e_2(\lambda(b)) - e_1(\lambda(a)) \leq u$. A time-point atom of the form $a \leq^{e}_{[l,u]} t$ or $a \geq^{e}_{[l,u]} t$ is* satisfied *by λ if, respectively, $l \leq t - e(\lambda(a)) \leq u$ or $l \leq e(\lambda(a)) - t \leq u$.*

Definition 7 (Semantics of synchronization rules). *Given a set of tokens Γ and a function $\lambda : \Sigma \to \Gamma$, a quantifier $a[x = v]$ is* satisfied *by λ if $\lambda(a) = (x_a, v_a, d_a)$, with $x_a = x$ and $v_a = v$. An existential statement $\mathcal{E}$, with conjunct clause $\mathcal{C}$, is satisfied by λ if all the quantifiers of $\mathcal{E}$ and all the atoms in $\mathcal{C}$ are satisfied by λ. A synchronization rule $\mathcal{R}$ of the form $a_0[x_0 = v_0] \longrightarrow \mathcal{E}_1 \vee \mathcal{E}_2 \vee \ldots \vee \mathcal{E}_k$ is* satisfied *by Γ if, for every token $\tau = (x_\tau, v_\tau, d_\tau) \in \Gamma$ where $x_\tau = x_0$ and $v_\tau = v_0$, there is an existential statement $\mathcal{E}_i$ and a mapping $\lambda : \Sigma \to \Gamma$ such that $\lambda(a_0) = \tau$ and λ satisfies $\mathcal{E}_i$.*

A timeline-based planning domain is specified by a set of state variables and a set of synchronization rules representing their admissible behaviors. Trigger-less rules can be used to express initial conditions, domain invariants, and the goals of the problem.

Definition 8 (Planning problem). *A timeline-based planning* problem *is a pair $P = (SV, S)$, where SV is a set of state variables and S is a set of synchronization rules involving variables in SV. A* solution plan *for P is a set of timelines*

$\pi = \{\mathsf{T}_1, \ldots, \mathsf{T}_n\}$*, one for each* $x_i \in SV$*, such that all the synchronization rules in* S *are satisfied by the set* Γ *of all the tokens involved in* π*, that is,* $\Gamma = \{\tau \mid \tau \in \mathsf{T}_i, 1 \le i \le n\}$.

Some useful relations can be defined. As an example, a *bounded* version of the thirteen Allen's ordering relations between pairs of intervals [Allen, 1983] can be defined in terms of the basic interval atoms of Definition 4. For instance, the equality and the *contains* interval relations between a and b can be defined as, respectively, $a \le^{\mathsf{s},\mathsf{s}}_{[0,0]} b \wedge a \le^{\mathsf{e},\mathsf{e}}_{[0,0]} b$ and $a \le^{\mathsf{s},\mathsf{s}}_{[l_1,u_1]} b \wedge b \le^{\mathsf{e},\mathsf{e}}_{[l_2,u_2]} a$ for some bounds $l_1, l_2, u_1, u_2 \in \mathbb{N}$.

3 Bounded TPTL with Past Captures Timeline-based Planning

This section introduces $\mathsf{TPTL_B{+}P}$, a variant of *Timed Propositional Temporal Logic* (TPTL), and shows how to reduce the timeline-based planning problem to the satisfiability problem of $\mathsf{TPTL_B{+}P}$ formulae. TPTL is an extension of LTL originally introduced in the area of formal verification to model properties of real-time systems [Alur and Henzinger, 1994], whose satisfiability checking problem is $\mathsf{EXPSPACE}$-complete. In its original definition, the logic only supports *future* temporal operators, because the addition of *past* modalities makes the complexity of the problem for the resulting logic $\mathsf{TPTL{+}P}$ *non-elementary* [Alur and Henzinger, 1993].

As a matter of fact, the possibility of referring to the past is useful to compactly encode timeline-based planning problems. For this reason, in this paper we introduce $\mathsf{TPTL_B{+}P}$, a *guarded fragment* of $\mathsf{TPTL{+}P}$ that features past modalities, but restricts the scope of both future and past modalities. $\mathsf{TPTL_B{+}P}$ turns out to be expressive enough to capture timeline-based planning problems, without increasing the computational complexity of the satisfiability checking problem, which stays $\mathsf{EXPSPACE}$-complete.

Let $\mathcal{AP}$ be a set of *proposition letters* and $\mathcal{V}$ be a set of *variables*. A $\mathsf{TPTL_B{+}P}$ formula ϕ over $\mathcal{AP}$ and $\mathcal{V}$ is recursively defined as follows:

$$\begin{aligned}\phi := p \mid \neg\phi_1 \mid \phi_1 \vee \phi_2 \mid x.\phi_1 \mid x \le y + c \mid x \le c \\ \mid \mathsf{X}_w\phi_1 \mid \phi_1\, \mathcal{U}_w\, \phi_2 \mid \mathsf{Y}_w\phi_1 \mid \phi_1\, \mathcal{S}_w\, \phi_2\end{aligned}$$

where $p \in \mathcal{AP}$, ϕ_1, ϕ_2 are $\mathsf{TPTL_B{+}P}$ formulae, $x, y \in \mathcal{V}$, $c \in \mathbb{Z}$, $w \in \mathbb{N} \cup \{+\infty\}$, and in any formula of the forms $\mathsf{X}_w\phi_1$, $\mathsf{Y}_w\phi_1$, $\phi_1\,\mathcal{U}_w\,\phi_2$, and $\phi_1\,\mathcal{S}_w\,\phi_2$, if $w = +\infty$, then ϕ_1, ϕ_2 are closed formulae (a formula ψ is *closed* if any occurrence of a variable x is inside a subformula of the form $x.\phi$). Formulae of the form $x.\phi$ are called *freeze quantifications*, while those of the forms $x \le y{+}c$ and $x \le c$ are called *timing constraints*. A missing w subscript stands for $w = +\infty$. Standard logical and temporal shortcuts are used, such as $\top$ for $p \vee \neg p$, for some $p \in \mathcal{AP}$, $\bot$ for $\neg\top$, $\phi_1 \wedge \phi_2$ for $\neg(\neg\phi_1 \vee \neg\phi_2)$, $\mathsf{F}\phi$ for $\top\,\mathcal{U}\,\phi$, $\mathsf{G}\phi$ for $\neg\mathsf{F}\neg\phi$, and $\mathsf{P}\phi$ for $\top\,\mathcal{S}\,\phi$, as well as constraint shortcuts, such as, e.g., $x \le y$ for $x \le y + 0$, $x > y$ for $\neg(x \le y)$, and $x = y$ for $\neg(x < y) \wedge \neg(y < x)$.

$\mathsf{TPTL_B{+}P}$ formulae are interpreted over *timed state sequences*, *i.e.*, structures $\rho = (\sigma, \tau)$, where $\sigma = \langle\sigma_0, \sigma_1, \ldots\rangle$ is an infinite sequence of states $\sigma_i \in 2^{\mathcal{AP}}$, with $i \ge 0$, and $\tau = \langle\tau_0, \tau_1, \ldots\rangle$ is an infinite sequence of *timestamps* $\tau_i \in \mathbb{N}$, with $i \ge 0$, such that (i) $\tau_{i+1} \ge \tau_i$ (*monotonicity*) and (ii) for all $t \in \mathbb{N}$, there is some $i \ge 0$ such that $\tau_i \ge t$ (*progress*).

Formally, the semantics of $\mathsf{TPTL_B{+}P}$ is defined as follows. An *environment* is a function $\xi : \mathcal{V} \to \mathbb{N}$ that interprets a given variable as a timestamp. A timed state sequence $\rho = (\sigma, \tau)$ satisfies a $\mathsf{TPTL_B{+}P}$ formula ϕ at position $i \ge 0$, under the environment ξ, written $\rho^i \models_\xi \phi$, if and only if:

- $\rho^i \models_\xi p$ iff $p \in \sigma_i$;
- $\rho^i \models_\xi \phi_1 \vee \phi_2$ iff either $\rho^i \models_\xi \phi_1$ or $\rho^i \models_\xi \phi_2$;
- $\rho^i \models_\xi \neg\phi_1$ iff $\rho^i \not\models_\xi \phi_1$;
- $\rho^i \models_\xi x \le y + c$ iff $\xi(x) \le \xi(y) + c$;
- $\rho^i \models_\xi x \le c$ iff $\xi(x) \le c$;
- $\rho^i \models_\xi x.\phi_1$ iff $\rho^i \models_{\xi'} \phi_1$ where $\xi' = \xi[x \leftarrow \tau_i]$;
- $\rho^i \models_\xi \mathsf{X}_w\phi_1$ iff $\tau_{i+1} \le \tau_i + w$ and $\rho^{i+1} \models_\xi \phi_1$;
- $\rho^i \models_\xi \phi_1\,\mathcal{U}_w\,\phi_2$ iff there exists $j \ge i$ such that: (i) $\tau_j \le \tau_i + w$, (ii) $\rho^j \models_\xi \phi_2$, and (iii) $\rho^k \models_\xi \phi_1$ for all k such that $i \le k < j$;
- $\rho^i \models_\xi \mathsf{Y}_w\phi_1$ iff $i > 0$, $\tau_i \le \tau_{i-1} + w$, and $\rho^{i-1} \models_\xi \phi_1$;
- $\rho^i \models_\xi \phi_1\,\mathcal{S}_w\,\phi_2$ iff there exists $j \le i$ such that: (i) $\tau_i \le \tau_j + w$, (ii) $\rho^j \models_\xi \phi_2$, and (iii) $\rho^k \models_\xi \phi_1$ for all k such that $j < k \le i$;

where $\xi' = \xi[x \leftarrow \tau_i]$ is the environment that agrees with ξ on each variable but x, where $\xi'(x) = \tau_i$.

A *closed* formula ϕ is said to be satisfied by a timed state sequence ρ, written $\rho \models \phi$, if $\rho^0 \models_\xi \phi$ for any ξ.

The logic $\mathsf{TPTL_B{+}P}$, as well as the original TPTL, is an extension of standard LTL with the addition of the freeze quantifier $x.\phi$, which allows one to give a name to the *timestamp* of the current state. Then, timing constraints can refer back to two states, *e.g.*, named x and y, comparing their timestamps. Thus, TPTL is a sort of *metric* extension of LTL, combined with some features typical of hybrid logics, but in a way that retains decidability and (relatively) low complexity. Here, in addition, we allow the use of past modalities *yesterday* and *since*. Adding them naïvely to TPTL would cause the complexity of the satisfiability problem to become non-elementary [Alur and Henzinger, 1993]. Here, we restrict all temporal modalities (including future ones) to look only as far as w time steps from the current state. An infinite bound $w = +\infty$ is allowed only if the argument formula is closed. This implies that in any timing constraint $x \le y + c$, the timestamps x and y can be distant, at most, an amount of time that is exponential in the size of the formula. Note that $\mathsf{TPTL_B{+}P}$ is still an extension of LTL (standard LTL operators are obtained from bounded ones by always using $w = +\infty$), and the bounded operators are expressible in the full $\mathsf{TPTL{+}P}$ by replacing, for instance, a formula $\mathsf{X}_w\phi$, with $w \ne +\infty$, by $x.\mathsf{X}y.(y \le x + w \wedge \phi)$. $\mathsf{TPTL_B{+}P}$ is thus a *guarded fragment* of full $\mathsf{TPTL{+}P}$.

Intuitively, the syntactic restriction imposed in $\mathsf{TPTL_B{+}P}$ occurs naturally in the specification of timeline-based problems as defined in Sect. 2. As an example, consider the rule:

$$\begin{aligned}a[x_0 = v_0] \longrightarrow \exists b[x_1 = v_1] c[\cdots] d[\cdots] e[\cdots]\ . \\ a \le^{\mathsf{s},\mathsf{s}}_{l,u} b \wedge b \le^{\mathsf{s},\mathsf{s}}_{l',u'} c \wedge d \le^{\mathsf{s},\mathsf{s}}_{l'',u''} e.\end{aligned}$$

The involved token names can be partitioned into two *components*, namely, $\{a, b, c\}$ and $\{d, e\}$, such that in both of them each element is related to the others. Since each atom provides an upper bound on the distance of the corresponding token endpoints, they can span an amount of time that is bounded by the sum of all the coefficients involved in the atoms of the component.

Definition 9. *Let $P = (SV, S)$ be a timeline-based planning problem, $\mathcal{R} \in S$ be a synchronization rule, and $\mathcal{E}$ be an existential statement of $\mathcal{R}$. Let $\Sigma_{\mathcal{E}}$ be the subset of token names used in $\mathcal{E}$ and $G_{\mathcal{E}} = (\Sigma_{\mathcal{E}}, E)$ be an undirected graph such that $(a, b) \in E$ iff $\mathcal{E}$ contains an atom $a \leq^{e_1,e_2}_{l,u} b$. A* component *Γ of $\mathcal{E}$ is a* maximal *subset of $\Sigma_{\mathcal{E}}$ whose elements are vertices of a connected component of $G_{\mathcal{E}}$.*

Definition 10. *Let $P = (SV, S)$ be a timeline-based planning problem. The* window *of the problem W_P is the product of all the non-zero coefficients appearing in the problem as duration bounds of tokens (D_x function of any variable x) or as bounds of atoms of any synchronization rules.*

Intuitively, W_P is a very weak, but correct, upper bound on how far away a component of an existential statement can look from any of its elements. In [Gigante *et al.*, 2016], this observation led to a doubly exponential upper bound to the size of the solution, which was exploited to devise an exponential-space decision procedure. Here, it allows us to provide a $\mathsf{TPTL_B{+}P}$ encoding of timeline-based planning problems: each token required by an existential statement is identified by a combination of *bounded* temporal operators and suitable timing constraints. While a single component can be confined into its exponential-size window, different components can be located arbitrarily far away from each other. For this reason, in [Gigante *et al.*, 2016] we restricted the result to rules with only a single component. Here, we can relax this restriction thanks to the *unbounded* temporal operators of $\mathsf{TPTL_B{+}P}$: since components are unrelated to each other, they can be described by closed formulae. In what follows, w.l.o.g., we will assume that the names used in each rule of a timeline-based planning problem are unique.

Thus, if $P = (SV, S)$ is a timeline-based planning problem, the equivalent $\mathsf{TPTL_B{+}P}$ formula ϕ_P is made as follows:

$$\phi_P \equiv \phi_0 \wedge \bigwedge_{x \in SV} \phi_x \wedge \bigwedge_{\mathcal{R} \in S} \phi_{\mathcal{R}}$$

where $\phi_0 \equiv t.(t = 0)$ states that time starts at timestamp zero, ϕ_x encodes the basic facts about each state variable $x \in SV$, and $\phi_{\mathcal{R}}$ encodes the semantics of synchronization rules $\mathcal{R} \in S$. For each state variable $x \in SV$, the formula uses two sets of proposition letters: for each value $v \in V_x$, a proposition letter s^x_v, to mark the *start* of a token for the variable x where $x = v$, and a proposition letter $\bot_x$ to mark the end of the timeline for x. A shortcut formula $s^x \equiv \bigvee_{v \in V_x} s^x_v$ will be used to assert the start of a token for x of any value.

Now, the formula ϕ_x that encodes the behavior of a variable $x = (V_x, T_x, D_x) \in SV$ is the conjunction of a few axioms, including the following: (1) at most one token starts at a given state, (2) the timeline eventually ends, and (3) either a token starts at $t = 0$ or it is preceded by another one. These three statements are encoded as follows:

$$\bigwedge\nolimits_{v \in V_x} \mathsf{G}\big(s^x_v \longrightarrow \neg \bigvee\nolimits^{v' \in V_x}_{v' \neq v} s^x_{v'}\big) \tag{1}$$

$$\mathsf{F}\bot_x \wedge \mathsf{G}\big(\bot_x \longrightarrow \mathsf{G}\neg s^x\big) \tag{2}$$

$$\mathsf{G}(s^x \wedge t.(t \neq 0) \longrightarrow \mathsf{Y}_{W_P}\mathsf{P}_{W_P} s^x). \tag{3}$$

Then, ϕ_x has to state that after the start of each token, the end must happen within the bounds required by the D_x function and the following token (if any) must respect the T_x function:

$$\mathsf{G}t_s.\Big(s^x_v \longrightarrow \mathsf{F}_{W_P} t_e.(\varepsilon^x_v(t_s) \wedge \\ t_e \geq t_s + d^v_{min} \wedge t_e \leq t_s + d^v_{max}))\Big)$$

where $D_x(v) = (d^v_{min}, d^v_{max})$ and, for all v, x, and t, $\varepsilon^x_v(t)$ is the following formula:

$$\varepsilon^x_v(t) \equiv \Big(\bot_x \vee \Big(\bigvee_{v' \in T_x(v)} s^x_{v'}\Big)\Big) \wedge \mathsf{Y}_{W_P}\big(\neg s^x \, \mathcal{S}_{W_P} \, t'.(t = t')\big)$$

which is true if the state at timestamp t was the start of a token for $x = v$, and the current state is the end of that token.

Note that the end of a token is represented by the start of the following one (or by $\bot_x$), and an empty token is described by two symbols s^x labeling states with the same timestamp.

The last part of ϕ_P describes the synchronization rules. Consider the synchronization rule $\mathcal{R} \equiv a_0[x = v_0] \longrightarrow \mathcal{E}_1 \vee \ldots \vee \mathcal{E}_k$. The formula $\phi_{\mathcal{R}}$ that encodes $\mathcal{R}$ has the form:

$$\phi_{\mathcal{R}} \equiv \mathsf{G}\, \overbrace{t^s_{a_0}.\Big(s^x_{v_0}}^{\text{start of } a_0} \longrightarrow \overbrace{\mathsf{F}_{W_P} t^e_{a_0}.(\varepsilon^x_{v_0}(t^s_{a_0})}^{\text{end of } a_0} \wedge \bigvee_{i=1}^{k} \phi_{\mathcal{E}_i})\Big)$$

where the formula $\phi_{\mathcal{E}_i}$ encodes the existential statement $\mathcal{E}_i$, as shown in the following.

Let Γ be a component of a given existential statement $\mathcal{E}$ and let $a_1[x_1 = v_1], \ldots, a_k[x_k = v_k]$ be the quantifiers used in $\mathcal{E}$ to introduce the names in Γ. Then, the formula encoding Γ is the following one:

$$\phi_\Gamma \equiv \underbrace{\overbrace{\exists_{W_P} t^s_{a_1}.\Big(s^{x_1}_{v_1}}^{\text{start of } a_1} \wedge \overbrace{\mathsf{F}_{W_P} t^e_{a_1}.(\varepsilon^{x_1}_{v_1}(t^s_{a_1})}^{\text{end of } a_1} \wedge (\cdots}_{\text{for all } a_i \in \Gamma} .\psi_\Gamma) \cdots)\Big)$$

where $\exists_w \phi \equiv \mathsf{F}_w \mathsf{P}_w \phi$ and ψ_Γ encodes the atoms of the component through the conjunction of suitable timing constraints. In ϕ_Γ, all the starting and ending endpoints of each token involved in the component are quantified, and their timestamps assigned to variables, respectively $t^s_{a_i}$ and $t^e_{a_i}$ for the start and end of the token a_i. Thus, in ψ_Γ, the atoms can be encoded directly. As an example, an atom $a_1 \leq^{\mathsf{s,e}}_{[3,7]} a_2$ can be encoded as $(t^e_{a_2} \geq t^s_{a_1} + 3) \wedge (t^e_{a_2} \leq t^s_{a_1} + 7)$.

Thus, let $\{\Gamma_1, \ldots, \Gamma_m\}$ be the components of the existential statement $\mathcal{E}$. At most one of these components, say Γ_i, contains the name a_0 used in the trigger, while the others are independent from it. Thus, the formula $\phi_{\mathcal{E}}$ that encodes $\mathcal{E}$ has the following form:

$$\phi_{\mathcal{E}} \equiv \phi_{\Gamma_i} \wedge \bigwedge\nolimits_{j \neq i} \mathsf{FP}\phi_{\Gamma_j}$$

Note that since all the components other than Γ_i do not mention the trigger, formulae Γ_j (for $j \neq i$) do not have free variables, as all the needed time points are quantified inside, and thus they are allowed to occur inside the unbounded *eventually* and *past* operators. Also note that the only free variables in any $\phi_{\mathcal{E}}$ are $t^s_{a_0}$ and $t^e_{a_0}$, *i.e.*, those referring to the trigger. A trigger-less rule is only determined by the satisfaction of its existential statements. Thus, the formula $\phi_{\mathcal{R}'}$ for a rule $\mathcal{R}' \equiv \top \longrightarrow \mathcal{E}'_1 \vee \ldots \vee \mathcal{E}'_k$ is simply $\phi_{\mathcal{R}'} \equiv \mathsf{F}\bigvee_{i=1}^{k}(\phi_{\mathcal{E}'_i})$; since $\mathcal{R}'$ has no free variables, it is syntactically well-formed. It can be verified that ϕ_P correctly encodes the problem P, and that a solution plan can be effectively extracted from a model of ϕ_P.

Theorem 1. *Given a timeline-based planning problem P with bounded constraints, there exists a solution plan for P if and only if the* $\mathsf{TPTL_B{+}P}$ *formula ϕ_P is satisfiable.*

Moreover, it is easy to see that the size of ϕ_P is polynomial in the size of P. Note that enumerating all the possible values of a variable, *e.g.*, in s^x, is permitted, since $|V_x| \in \mathcal{O}(|P|)$, as the values are already enumerated extensionally in the input representation of P to specify the T_x and D_x functions.

4 Complexity of $\mathsf{TPTL_B{+}P}$

We now show that, given a closed $\mathsf{TPTL_B{+}P}$ formula ϕ, deciding whether ϕ has a model is $\mathsf{EXPSPACE}$-complete. Hardness comes from the encoding shown in Sect. 3, and the complexity result from [Gigante *et al.*, 2016], thus here we only provide an exponential-space decision procedure. We start with some notation. Let $n = |\phi|$ be the *length* of ϕ, and m be the number of temporal modalities of ϕ with a *finite* bound, with $\{w_1, \ldots, w_m\}$ the set of all such finite bounds. Among those, let $w_0 = \max\{w_i \mid 1 \leq i \leq m\}$ be the maximum one, and let $W = w_0 \cdot (m+1)$. Similarly, let $\Delta = \prod_i |c_i|$ for all the *finite* and *non-zero* coefficients c_i appearing in timing constraints of the formula, *e.g.*, in $x \leq y + c$. Note that, since coefficients are succinctly encoded, the size of both Δ and W is exponential in n.

W.l.o.g., hereafter we restrict ourselves to formulae without absolute timing constraints of the type $x \leq c$. To show that the problem can be solved in exponential space, we outline a tableau-based decision procedure based on the tableau for TPTL outlined in Alur and Henzinger [1994], which in turn exploits ideas very similar to those of classic tableau systems for LTL [Manna and Pnueli, 1995; Lichtenstein and Pnueli, 2000]: a graph is built, where each node represents a possible state of a model, and then a model is searched among the infinite paths of this graph. In contrast to the usual LTL tableaux, however, a tableau for TPTL and $\mathsf{TPTL_B{+}P}$ has also to keep track, in addition to the truth assignment of any given node, of how much time has to pass between two different states, and handle the freeze quantifications accordingly. The key ingredient is the following.

Definition 11. *Consider a* $\mathsf{TPTL_B{+}P}$ *formula $x.\psi$ and $\delta \in \mathbb{Z}$. Then, the formula $x.\psi^\delta$ is obtained from $x.\psi$ in two steps:*

1. *one first replaces all the subformulae of the forms $x \leq y + c$ and $y \leq x + c$ by, respectively, $x \leq y + (c + \delta)$ and $y \leq x + (c - \delta)$;*
2. *then, all the subformulae of the forms $x \leq y + c$ and $y \leq x + c$ are replaced by $\top$ (resp., $\bot$) if $c > W$ (resp., $c < -W$).*

Intuitively, $x.\psi^\delta$ is a *time-shifted* version of $x.\psi$ where x is mapped to $x - \delta$, in such a way that $x.\psi^\delta$ holds at time τ if and only if ψ holds at $\tau - \delta$.

As in classic tableaux, the nodes are sets of formulae drawn from the *closure set* of ϕ, which contains all the formulae needed to assess the truth of ϕ over a model.

Definition 12. *Given a* closed $\mathsf{TPTL_B{+}P}$ *formula ϕ of the form $x.\phi'$, the* closure *of ϕ is the smallest set of formulae $\mathcal{C}(\phi)$ containing $x.\phi'$ such that:*

- *if $x.(\psi_1 \vee \psi_2) \in \mathcal{C}(\phi)$, then $\{x.\psi_1, x.\psi_2\} \subseteq \mathcal{C}(\phi)$,*
- *if $x.\mathsf{X}_w\psi \in \mathcal{C}(\phi)$, then $\{x.\psi^\delta \mid \delta \in \mathbb{N}\} \subseteq \mathcal{C}(\phi)$,*
- *if $x.\mathsf{Y}_w\psi \in \mathcal{C}(\phi)$, then $\{x.\psi^{-\delta} \mid \delta \in \mathbb{N}\} \subseteq \mathcal{C}(\phi)$,*
- *if $x.(\psi_1 \,\mathcal{U}_w\, \psi_2) \in \mathcal{C}(\phi)$, then $x.\psi_1$, $x.\psi_2$, and $x.\mathsf{X}_w(\psi_1 \,\mathcal{U}_{w-\delta}\, \psi_2)$, for all $\delta \leq w$, belong to $\mathcal{C}(\phi)$,*
- *if $x.(\psi_1 \,\mathcal{S}_w\, \psi_2) \in \mathcal{C}(\phi)$, then $x.\psi_1$, $x.\psi_2$, and $x.\mathsf{Y}_w(\psi_1 \,\mathcal{S}_{w-\delta}\, \psi_2)$, for all $\delta \leq w$, belong to $\mathcal{C}(\phi)$,*
- *if $x.z.\psi \in \mathcal{C}(\phi)$, then $x.\psi[z/x] \in \mathcal{C}(\phi)$,*

where $w - \delta = w$ if $w = +\infty$ and $\psi[z/x]$ is obtained from ψ by replacing every free occurrence of z with x.

Note that, w.l.o.g., we can restrict our attention to formulae ϕ of the form $x.\phi'$ and observe that $\mathcal{C}(\phi)$ only contains closed formulae of the form $z.\psi$, for some variable z. As one may see, the definition of the closure of ϕ is very similar to that given for TPTL by Alur and Henzinger [1994], but it differs in the handling of the bounds in the temporal operators $\mathcal{U}_w$ and $\mathcal{S}_w$, which are decremented by the suitable amount at each expansion. Another crucial difference is the definition of $x.\psi^\delta$ (Definition 11), which has to deal with shifts in both directions. Note that the closure of $x.\mathsf{X}_w\psi$ (resp., $x.\mathsf{Y}_w\psi$) seems to generate an infinite number of formulae $x.\psi^\delta$, for all $\delta \in \mathbb{N}$, but this is not the case. In the case of TPTL, one may assume that in a closed formula of the form $x.\psi$, any other variable y that occurs in ψ is instantiated only at a timestamp later than x. This fact ensures that the closure set remains finite, since, for instance, any formula of the form $x \leq y+c$ can be replaced by $\top$ for any c. Here, however, since ψ may use past operators, this argument does not work, and the closure set is at risk of becoming infinite. Nevertheless, we ensure its finiteness by replacing with $\top$ or $\bot$ any formula $x \leq y + c$ where c goes respectively above W or below $-W$. This replacement is sound thanks to the restriction that unbounded temporal operators may only be used with closed formulae, which guarantees that the value of any x and y used in a timing constraint can only differ as much as W. In addition, we can bound the size of the closure set which, as in the TPTL case, is exponential in the size of the formula. Thus, the following two results can be proved.

Lemma 1. *Let $\rho = (\sigma, \tau)$ be a timed state sequence and ξ be an environment. Consider a position $i \geq 0$ and $\delta \in \mathbb{Z}$ such that $\delta \leq \tau_i$. Then, for any formula $x.\psi \in \mathcal{C}(\phi)$:*

$$\rho^i \models_\xi x.\psi^\delta \text{ iff } \rho^i \models_{\xi'} \psi,$$

where $\xi' = \xi[x \leftarrow \tau_i - \delta]$.

Lemma 2. *Let ϕ be a* $\mathsf{TPTL_B{+}P}$ *formula. Then,*

$$|\mathcal{C}(\phi)| \in \mathcal{O}(n \cdot W \cdot \Delta).$$

We now show how the tableau for ϕ is built. To this end, let $\mathcal{C}^*(\phi) = \mathcal{C}(\phi) \cup \{Prev_\delta \mid 0 \leq \delta \leq \Delta\} \cup \{Succ_\gamma \mid 0 \leq \gamma \leq \Delta\}$ be an extension of the closure set of ϕ with fresh proposition letters $Prev_\delta$ and $Succ_\gamma$, which keep track of the time between the current state and, respectively, the previous and the next one. Note that we consider only time steps up to Δ, since, as the TPTL case, we can restrict ourselves to timed state sequences where $\tau_i - \tau_{i-1} < \Delta$ for all $i > 0$.

Definition 13 (Atoms for ϕ)**.** *An* atom *for ϕ is a maximal subset Φ of $\mathcal{C}^*(\phi)$ such that:*

- *$Prev_\delta \in \Phi$ and $Succ_\gamma \in \Phi$ for exactly one δ and exactly one γ between 0 and Δ. Such δ and γ will be denoted respectively as δ_Φ and γ_Φ, and, for $i \leq j$, we let $\delta_{\Phi_i,\Phi_j} = \sum_{i<k\leq j} \delta_{\Phi_k}$.*
- *$x.(x \leq x + c) \in \Phi$ iff $c \geq 0$,*
- *$x.z.\psi \in \Phi$ iff $x.\psi[z/x] \in \Phi$,*
- *$x.\neg\phi \in \Phi$ iff $x.\phi \notin \Phi$,*
- *$x.(\phi_1 \vee \phi_2) \in \Phi$ iff either $x.\phi_1 \in \Phi$ or $x.\phi_2 \in \Phi$,*
- *$x.(\phi_1 \,\mathcal{U}_w\, \phi_2) \in \Phi$ iff either $x.\phi_2 \in \Phi$ or both $x.\phi_1 \in \Phi$ and $x.\mathsf{X}_w(\phi_1 \,\mathcal{U}_{w-\gamma_\Phi}\, \phi_2) \in \Phi$,*
- *$x.(\phi_1 \,\mathcal{S}_w\, \phi_2) \in \Phi$ iff either $x.\phi_2 \in \Phi$ or both $x.\phi_1 \in \Phi$ and $x.\mathsf{Y}_w(\phi_1 \,\mathcal{S}_{w-\delta_\Phi}\, \phi_2) \in \Phi$.*

Definition 14 (Tableau for ϕ)**.** *The* tableau *for ϕ is a graph where the nodes are all the possible atoms for ϕ and there is an edge between Φ and Ψ iff the following conditions hold:*

1. *$\gamma_\Phi = \delta_\Psi$;*
2. *$z.\mathsf{X}_w\psi \in \Phi$ iff $\gamma_\Phi \leq w$ and $z.\psi^{\gamma_\Phi} \in \Psi$;*
3. *$z.\mathsf{Y}_w\psi \in \Psi$ iff $\delta_\Psi \leq w$ and $z.\psi^{-\delta_\Psi} \in \Phi$.*

Conditions 2 and 3 above handle the temporal operators *tomorrow* and *yesterday*, by ensuring that whenever there is an edge between two atoms, the formulae requested by temporal operators in the two atoms are present. Intuitively, this is also the point where the tableau handles the binding of variables without explicitly keeping track of any environment, by pushing the freeze quantification to the next state while shifting the formula to preserve the semantics, thanks to Lemma 1. Then, as in the tableaux for LTL and TPTL, the search for a model for ϕ is reduced to the search for a particular infinite path.

Definition 15. *Given the tableau for ϕ, a ϕ-path is an* infinite path *$\Phi = \langle \Phi_0, \Phi_1, \ldots \rangle$ of atoms from the tableau such that:*

1. *$\phi \in \Phi_0$,*
2. *No formulae of the form $x.\mathsf{Y}_w\psi$ belong to Φ_0,*
3. *$\delta_{\Phi_i} > 0$ for infinitely many $i \geq 0$,*
4. *for all $i \geq 0$ and all $x.(\phi_1 \,\mathcal{U}_w\, \phi_2) \in \Phi_i$, there is $k \geq i$ such that $\delta_{\Phi_i,\Phi_k} \leq w$, and both $x.\phi_2^{\delta_{\Phi_i,\Phi_k}} \in \Phi_k$ and $x.\phi_1^{\delta_{\Phi_i,\Phi_j}} \in \Phi_j$ for all j with $i \leq j < k$, and*
5. *for all $i \geq 0$ and all $x.(\phi_1 \,\mathcal{S}_w\, \phi_2) \in \Phi_i$, there is $k \leq i$ such that $\delta_{\Phi_k,\Phi_i} \leq w$, and both $x.\phi_2^{\delta_{\Phi_k,\Phi_i}} \in \Phi_k$ and $x.\phi_1^{\delta_{\Phi_j,\Phi_i}} \in \Phi_j$ for all j with $k < j \leq i$.*

Intuitively, a ϕ-path contains all the necessary pieces of information to build a model for the formula. Each atom corresponds to a state of the model, and its elements are the formulae (in particular, the proposition letters) that will be true in that state. The timestamps of each state can be derived from $\delta_{\Phi_0} + \delta_{\Phi_0,\Phi_i}$ for each i. Vice versa, given a model we can build a ϕ-path by simply listing in each atom all the formulae from the closure set that are true in each state. In verifying these claims, both directions of Lemma 1 play a crucial role, and conditions 4 and 5 of Definition 15 ensure its applicability. The decision of the satisfiability of ϕ can thus be reduced to the search for a ϕ-path into the tableau (see Lemma 3).

Lemma 3 (Correctness and Completeness)**.** *A* $\mathsf{TPTL_B{+}P}$ *formula ϕ is satisfiable if and only if its tableau has a ϕ-path.*

The search for a ϕ-path is non-trivial since there might be an infinite number of them. However, as in the TPTL case, we can restrict the search to *periodic* ϕ-paths of *bounded* length.

Lemma 4. *Let ϕ be a* $\mathsf{TPTL_B{+}P}$ *formula and let m be the number of nodes of the tableau for ϕ. If the tableau contains a ϕ-path, then it contains a periodic ϕ-path of the form $\Phi_0\Phi_1 \ldots \Phi_i(\Phi_{i+1} \ldots \Phi_l)^\omega$, for $i, l \in \mathcal{O}(m \cdot n \cdot W \cdot \Delta)$.*

Observe that the number m of nodes in the tableau is at most $2^{|\mathcal{C}(\phi)|}$, that is, doubly exponential in the size of the formula, by Lemma 2. Looking for a doubly exponentially long ϕ-path can be done without constructing the whole graph, in singly exponential space as in other LTL and TPTL tableau systems [Manna and Pnueli, 1995; Alur and Henzinger, 1994].

Theorem 2. *Given a* $\mathsf{TPTL_B{+}P}$ *formula ϕ, the problem of deciding whether ϕ is satisfiable is* $\mathsf{EXPSPACE}$*-complete.*

5 Conclusions

The present paper provides a logical characterization of the fragment of non-flexible timeline-based planning studied in [Gigante *et al.*, 2016], showing that it can be captured by $\mathsf{TPTL_B{+}P}$, a particular guarded fragment of TPTL augmented with past modalities. Thanks to the encoding shown in [Gigante *et al.*, 2016], the result directly transfers to action-based temporal planning, which thus can also be captured by $\mathsf{TPTL_B{+}P}$ (a different logical characterization of temporal planning has been recently given in terms of $\mathsf{LTL_{RA}}$ in [Cimatti *et al.*, 2017]). The proposed logic has been shown to be $\mathsf{EXPSPACE}$-complete by adapting the original tableau-based decision procedure for TPTL. The encoding of timeline-based planning problems into $\mathsf{TPTL_B{+}P}$ provided in Sect. 3 turned out to be very natural, and shows how *unbounded* interval relations affect the expressive power of the formalisms. In usual settings (see *e.g.*, [Cialdea Mayer *et al.*, 2016]), where timeline-based planning problems are given with a bound on the solution horizon, allowing for unbounded interval relations do not increase the expressive power (action-based temporal planning cannot be captured in those settings). On the contrary, enriching our setting with unbounded interval relations would result in increased expressive power, as we do not impose bounds on the horizon. Unfortunately, a naïve adaptation of our encoding into $\mathsf{TPTL_B{+}P}$ is feasible but results in exponentially-sized formulae. The problem of finding a polynomial encoding is under investigation.

References

[Allen, 1983] J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[Alur and Henzinger, 1993] R. Alur and T. A. Henzinger. Real-Time Logics: Complexity and Expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[Alur and Henzinger, 1994] R. Alur and T. A. Henzinger. A Really Temporal Logic. *Journal of the ACM*, 41(1):181–204, 1994.

[Barreiro *et al.*, 2012] J. Barreiro, M. Boyce, M. Do, J. Frank, M. Iatauro, T. Kichkaylo, P. Morris, J. Ong, E. Remolina, T. Smith, and D. Smith. EUROPA: A Platform for AI Planning, Scheduling, Constraint Programming, and Optimization. In *Proc. of the 4th International Competition on Knowledge Engineering for Planning and Scheduling*, 2012.

[Bylander, 1994] T. Bylander. The Computational Complexity of Propositional STRIPS Planning. *Artificial Intelligence*, 69(1-2):165–204, 1994.

[Cesta *et al.*, 2007] A. Cesta, G. Cortellessa, S. Fratini, A. Oddi, and N. Policella. An Innovative Product for Space Mission Planning: An A Posteriori Evaluation. In *Proc. of the 17th International Conference on Automated Planning and Scheduling*, pages 57–64, 2007.

[Cesta *et al.*, 2009] A. Cesta, G. Cortellessa, S. Fratini, and A. Oddi. Developing an End-to-End Planning Application from a Timeline Representation Framework. In *Proc. of the 21st Conference on Innovative Applications of Artificial Intelligence (IAAI-09)*, pages 66–71, 2009.

[Cesta *et al.*, 2010] Amedeo Cesta, Alberto Finzi, Simone Fratini, Andrea Orlandini, and Enrico Tronci. Validation and Verification Issues in a Timeline-based Planning System. *Knowledge Engineering Review*, 25(3):299–318, 2010.

[Chien *et al.*, 2000] S. Chien, G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, and D. Tran. ASPEN - Automated Planning and Scheduling for Space Mission Operations. In *Proc. of the 8th International Conference on Space Operations*, 2000.

[Cialdea Mayer *et al.*, 2007] M. Cialdea Mayer, C. Limongelli, A. Orlandini, and V. Poggioni. Linear Temporal Logic as an Executable Semantics for Planning Languages. *Journal of Logic, Language and Information*, 16(1):63–89, 2007.

[Cialdea Mayer *et al.*, 2014] M. Cialdea Mayer, A. Orlandini, and A. Ubrico. A Formal Account of Planning with Flexible Timelines. In *Proc. of the 21st International Symposium on Temporal Representation and Reasoning*, pages 37–46, 2014.

[Cialdea Mayer *et al.*, 2016] M. Cialdea Mayer, A. Orlandini, and A. Umbrico. Planning and Execution with Flexible Timelines: a Formal Account. *Acta Informatica*, 53(6-8):649–680, 2016.

[Cimatti *et al.*, 2013] A. Cimatti, A. Micheli, and M. Roveri. Timelines with Temporal Uncertainty. In *Proc. the 27th AAAI Conference on Artificial Intelligence*, 2013.

[Cimatti *et al.*, 2017] A. Cimatti, A. Micheli, and M. Roveri. Validating Domains and Plans for Temporal Planning via Encoding into Infinite-State Linear Temporal Logic. In *Proc. of the 31st AAAI Conference on Artificial Intelligence*, pages 3547–3554, 2017.

[Fox and Long, 2003] M. Fox and D. Long. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research*, 20:61–124, 2003.

[Gerevini *et al.*, 2009] A. Gerevini, P. Haslum, D. Long, A. Saetti, and Y. Dimopoulos. Deterministic Planning in the Fifth International Planning Competition: PDDL3 and Experimental Evaluation of the Planners. *Artificial Intelligence*, 173(5-6):619–668, 2009.

[Gigante *et al.*, 2016] N. Gigante, A. Montanari, M. Cialdea Mayer, and A. Orlandini. Timelines are Expressive Enough to Capture Action-based Temporal Planning. In *Proc. of the 23rd International Symposium on Temporal Representation and Reasoning*, pages 100–109, 2016.

[Jónsson *et al.*, 2000] A. K. Jónsson, P. H. Morris, N. Muscettola, K. Rajan, and B. D. Smith. Planning in Interplanetary Space: Theory and Practice. In *Proc. of 5th International Conference on Artificial Intelligence Planning and Scheduling*, pages 177–186, 2000.

[Lichtenstein and Pnueli, 2000] O. Lichtenstein and A. Pnueli. Propositional Temporal Logics: Decidability and Completeness. *Logic Journal of the IGPL*, 8(1):55–85, 2000.

[Manna and Pnueli, 1995] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems - Safety*. Springer, 1995.

[Muscettola, 1994] N. Muscettola. HSTS: Integrating Planning and Scheduling. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, chapter 6, pages 169–212. Morgan Kaufmann, 1994.

[Pnueli, 1977] A. Pnueli. The Temporal Logic of Programs. In *Proc. of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE Computer Society, 1977.

[Rintanen, 2007] J. Rintanen. Complexity of Concurrent Temporal Planning. In *Proc. of the 17th International Conference on Automated Planning and Scheduling*, pages 280–287, 2007.

[Schewe and Tian, 2011] S. Schewe and C. Tian. Synthesising Classic and Interval Temporal Logic. In *18th International Symposium on Temporal Representation and Reasoning, TIME 2011*, pages 64–71. IEEE, 2011.