One-Pass and Tree-Shaped Tableau Systems for TPTL and TPTL$_\mathsf{b}$+Past

Luca Geatti, Nicola Gigante, and Angelo Montanari
University of Udine, Italy
{geatti.luca,gigante.nicola}@spes.uniud.it
angelo.montanari@uniud.it

Mark Reynolds
The University of Western Australia
mark.reynolds@uwa.edu.au

In this paper, we propose a novel one-pass and tree-shaped tableau method for Timed Propositional Temporal Logic and for a bounded variant of its extension with past operators. Timed Propositional Temporal Logic (TPTL) is a *real-time* temporal logic, with an EXPSPACE-complete satisfiability problem, which has been successfully applied to the verification of real-time systems. In contrast to LTL, adding past operators to TPTL makes the satisfiability problem for the resulting logic (TPTL+P) *non-elementary*. In this paper, we devise a one-pass and tree-shaped tableau for both TPTL and *bounded* TPTL+P (TPTL$_\mathsf{b}$+P), a syntactic restriction introduced to encode timeline-based planning problems, which recovers the EXPSPACE-complete complexity. The tableau systems for TPTL and TPTL$_\mathsf{b}$+P are presented in a unified way, being very similar to each other, providing a common skeleton that is then specialised to each logic. In doing that, we characterise the semantics of TPTL$_\mathsf{b}$+P in terms of a purely syntactic fragment of TPTL+P, giving a translation that embeds the former into the latter. Soundness and completeness of the system are proved fully. In particular, we give a greatly simplified model-theoretic completeness proof, which sidesteps the complex combinatorial argument used by known proofs for the one-pass and tree-shaped tableau systems for LTL and LTL+P.

1 Introduction

Among the reasoning methods used to decide the satisfiability of logical formulae, *tableau methods* are among the earliest proposed and most studied solutions [4]. Classic tableau methods for logics of the linear time, such as, for instance, Linear Temporal Logic (LTL) [8, 9], build a graph structure which is then traversed to look for possible models of the formula. Despite being a useful theoretical tool, such graph-shaped tableau systems are not efficient in practice as they need to build and traverse a huge graph structure in multiple passes. Various ways of overcoming such a limitation have been proposed in the literature, including incremental [7] and single pass techniques [12]. Recently, a one-pass and tree-shaped tableau system for LTL has been devised [11], which does not build any huge preliminary structure and, thanks to its pure rule-based tree-search structure, proved to be amenable to efficient implementation and easy parallelisation [3, 10]. Recent work also suggested that its modular structure makes it possible to easily extend it to other linear time logics (the extension to LTL with *past operators* is described in [6]).

Timed Propositional Temporal Logic (TPTL) is a linear time logic, which extends LTL with the ability to express real-time properties of systems and computations [2]. The greater expressive power of TPTL is

These results were developed mainly while A. Montanari was on leave at the *Stockholm University*, and N. Gigante was on leave at the *University of Western Australia*, supported by the *AIxIA Outgoing mobility grant 2017*. The work was partially supported by the Italian GNCS project *Formal methods for verification and synthesis of discrete and hybrid systems* (N. Gigante and A. Montanari), the PRID project *ENCASE - Efforts in the uNderstanding of Complex interActing SystEms* (N. Gigante and A. Montanari), and the Australian Research Council funding–DP140103365 (M. Reynolds).

A. Orlandini, M. Zimmermann (Eds.): 9th Symposium on Games, Automata, Logics and Formal Verification (GandALF'18)
EPTCS 277, 2018, pp. 176–190, doi:10.4204/EPTCS.277.13

reflected in the computational complexity of its satisfiability problem, which is EXPSPACE-complete. Originally proposed as a formal tool for the verification of real-time systems, it recently found interesting applications in the area of artificial intelligence, to encode a meaningful class of *timeline-based planning* problems [5]. This and other application scenarios benefit from/require the use of *past operators*, which allow the logic to compactly predicate about events in the past of the current time point. However, in contrast to the case of LTL, where past operators can be supported without harm, adding them to TPTL greatly increases the complexity of its satisfiability problem, which becomes *non-elementary* [1]. For this reason, *bounded TPTL with Past* ($\mathsf{TPTL_b}$+P) has been introduced [5], which supports past operators, but suitably restricts their use in order to recover an EXPSPACE-complete satisfiability problem. While initially introduced as a specific tool to encode planning problems, $\mathsf{TPTL_b}$+P is interesting by itself, since it enables the use of past operators in a fairly natural way.

In this paper, we exploit the extensibility of the aforementioned tableau system to provide a one-pass and tree-shaped tableau method for TPTL and $\mathsf{TPTL_b}$+P. We present both tableau systems, which are very similar, in a unified way by first (i) factoring out the common structure, and then (ii) showing how to specialise it in the case of TPTL (future-only) and $\mathsf{TPTL_b}$+P (bounded) formulae, thus obtaining a one-pass and tree-shaped tableau system for both logics. To show how the tableau for $\mathsf{TPTL_b}$+P formulae works, (iii) we characterise the semantics of the logic in terms of a guarded fragment of the full TPTL+P logic, showing how to translate $\mathsf{TPTL_b}$+P into this fragment. Furthermore, (iv) the completeness of the two tableau systems is shown by a greatly simplified proof exploiting a new model-theoretic technique which sidesteps the complex combinatorial argument used by known proofs for LTL and LTL+P.

The tableau systems presented here for TPTL and $\mathsf{TPTL_b}$+P are truly extensions of the previously known ones for LTL and LTL+P, respectively, in the sense that their rules and behaviour are exactly the same as before when applied to pure LTL/LTL+P formulae, further confirming the modular and extensible nature of the one-pass tree-shaped system.

The paper is structured as follows. Syntax and semantics of TPTL, TPTL+P, and $\mathsf{TPTL_b}$+P are illustrated in Section 2. Then, Section 3 describes the tableau systems for TPTL and $\mathsf{TPTL_b}$+P. It first introduces the general skeleton common to both, and then it shows how to tailor it to TPTL and $\mathsf{TPTL_b}$+P. Finally, soundness and completeness of both systems are proved in Section 4. Section 5 concludes with some considerations on the obtained results and open problems.

2 Timed Propositional Temporal Logic

This section defines syntax and semantics of TPTL [2], TPTL+P [2], and $\mathsf{TPTL_b}$+P [5]. Let $\mathsf{AP} = \{p, q, r, \ldots\}$ be a set of *proposition letters* and $\mathsf{V} = \{x, y, z, \ldots\}$ be a set of *variables*. A TPTL+P formula ϕ over AP and V is recursively defined as follows:

$$\begin{aligned}\phi := p \mid \neg\phi_1 \mid \phi_1 \vee \phi_2 \mid x.\phi_1 \mid x \leq y + c \mid x \leq c \mid x \equiv_m y + c \\ \mid \mathsf{X}\,\phi_1 \mid \phi_1\,\mathsf{U}\,\phi_2 \mid \phi_1\,\mathsf{R}\,\phi_2 \mid \mathsf{Y}\,\phi_2 \mid \phi_1\,\mathsf{S}\,\phi_2 \mid \phi_1\,\mathsf{T}\,\phi_2,\end{aligned}$$

where $p \in \mathsf{AP}$, ϕ_1 and ϕ_2 are TPTL+P formulae, $x, y \in \mathsf{V}$, $c \in \mathbb{Z}$, $m \in \mathbb{N}$, and $\equiv_m$ is the congruence modulo the constant m. Formulae of the form $x.\phi$ are called *freeze quantifications*, while those of the forms $x \leq y + c$, $x \leq c$, and $x \equiv_m y$ are called *timing constraints*. Standard logical and temporal shortcuts, e.g., $\top$ for $p \vee \neg p$, for some $p \in \mathsf{AP}$, $\bot$ for $\neg\top$, $\phi_1 \wedge \phi_2$ for $\neg(\neg\phi_1 \vee \neg\phi_2)$, $\mathsf{F}\,\phi$ for $\top\,\mathsf{U}\,\phi$, $\mathsf{G}\,\phi$ for $\neg\mathsf{F}\neg\phi$, and $\mathsf{P}\,\phi$ for $\top\,\mathsf{S}\,\phi$, as well as constraint shortcuts, e.g., $x \leq y$ for $x \leq y + 0$, $x > y$ for $\neg(x \leq y)$, and $x = y$ for $\neg(x < y) \wedge \neg(y < x)$, are used. A formula ϕ is *closed* if each occurrence of a variable x is enclosed by a subformula of the form $x.\psi$. As for LTL+P, the temporal operators can be partitioned in *future*

(*tomorrow* X, *until* U, and *release* R) and *past* (*yesterday* Y, *since* S, and *triggered* T) ones. TPTL is the fragment of TPTL+P where only future operators are used.

TPTL+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 $\sigma_i \in 2^{\mathsf{AP}}$, for $i \geq 0$, and $\tau = \langle \tau_0, \tau_1, \ldots \rangle$ is an infinite sequence of *timestamps* $\tau_i \in \mathbb{N}$, for $i \geq 0$, such that (i) $\tau_{i+1} \geq \tau_i$ (*monotonicity*), and (ii) for all $t \in \mathbb{N}$, there is some $i \geq 0$ such that $\tau_i \geq t$ (*progress*). Formally, the semantics of TPTL+P is defined as follows. Functions $\xi : \mathsf{V} \to \mathbb{N}$ mapping variables to timestamps are called *environments*. A timed state sequence ρ *satisfies* a formula ϕ at position $i \geq 0$, with environment ξ, written $\rho \models^i_\xi \phi$, if (and only if):

1. $\rho \models^i_\xi p$ iff $p \in \sigma_i$;
2. $\rho \models^i_\xi \phi_1 \vee \phi_2$ iff $\rho \models^i_\xi \phi_1$ or $\rho \models^i_\xi \phi_2$;
3. $\rho \models^i_\xi \neg\phi_1$ iff $\rho \not\models^i_\xi \phi_1$;
4. $\rho \models^i_\xi x \leq y + c$ iff $\xi(x) \leq \xi(y) + c$;
5. $\rho \models^i_\xi x \leq c$ iff $\xi(x) \leq c$;
6. $\rho \models^i_\xi x \equiv_m y + c$ iff $\xi(x) \equiv_m \xi(y) + c$;
7. $\rho \models^i_\xi x.\phi_1$ iff $\rho \models^i_{\xi'} \phi_1$ where $\xi' = \xi[x \leftarrow \tau_i]$;
8. $\rho \models^i_\xi \mathsf{X}\phi_1$ iff $\rho \models^{i+1}_\xi \phi_1$;
9. $\rho \models^i_\xi \phi_1 \mathsf{U} \phi_2$ iff there exists $j \geq i$ such that $\rho \models^j_\xi \phi_2$ and $\rho \models^k_\xi \phi_1$ for all $i \leq k < j$;
10. $\rho \models^i_\xi \phi_1 \mathsf{R} \phi_2$ iff either $\rho \models^j_\xi \phi_2$ for all $j \geq i$, or there exists a $k \geq i$ such that $\rho \models^k_\xi \phi_1$ and $\rho \models^j_\xi \phi_2$ for all $i \leq j \leq k$;
11. $\rho \models^i_\xi \mathsf{Y}\phi_1$ iff $i > 0$ and $\rho^{i-1} \models_\xi \phi_1$;
12. $\rho \models^i_\xi \phi_1 \mathsf{S} \phi_2$ iff there exists $j \leq i$ such that $\rho \models^j_\xi \phi_2$, and $\rho \models^k_\xi \phi_1$ for all $j < k \leq i$;
13. $\rho \models^i_\xi \phi_1 \mathsf{T} \phi_2$ iff either $\rho \models^j_\xi \phi_2$ for all $0 \leq j \leq i$, or there exists a $k \leq i$ such that $\rho \models^k_\xi \phi_1$ and $\rho \models^j_\xi \phi_2$ for all $i \geq j \geq k$,

where $\xi' = \xi[x \leftarrow \tau_i]$ is the environment equal to ξ possibly excepting $\xi'(x) = \tau_i$. A *closed* formula ϕ is satisfied by a timed state sequence ρ, written $\rho \models \phi$, if $\rho \models^0_\xi \phi$, for any ξ. TPTL and TPTL+P can thus be viewed as (*metric*) extensions of, respectively, LTL and LTL+P with the *freeze quantifier* $x.\phi$, that allows one to bind a variable to the *timestamp* of the current state, which can then be compared with other variables by the timing constraints. In contrast to LTL and LTL+P, which both have a PSPACE-complete satisfiability problem, adding past operators to TPTL causes a complexity blowup: the satisfiability problem is EXPSPACE-complete for TPTL, but *non-elementary* for TPTL+P [2].

The unconstrained use of past operators in these timed logics is thus impossible in practice. However, there are many scenarios where referring to the past may be needed, and thus it is useful to search for possible ways of adding past operators to TPTL while retaining a (relatively) practicable complexity. TPTL$_\mathsf{b}$+P has been introduced to encode a meaningful class of timeline-based planning problems, whose synchronisation rules can interchangeably refer to the future or the past [5]. The syntax of TPTL$_\mathsf{b}$+P is similar to that of TPTL+P, the only difference being that each temporal operator is subscripted with a bound which constrains the visibility of the operator. Formally, a TPTL$_\mathsf{b}$+P formula ϕ over AP and V is recursively defined as follows:

$$\begin{aligned}\phi := {} & p \mid \neg\phi_1 \mid \phi_1 \vee \phi_2 \mid x.\phi_1 \mid x \leq y + c \mid x \leq c \mid x \equiv_m y + c \\ & \mid \mathsf{X}_w \phi_1 \mid \hat{\mathsf{X}}_w \phi_1 \mid \phi_1 \mathsf{U}_w \phi_2 \mid \phi_1 \mathsf{R}_w \phi_2 \mid \mathsf{Y}_w \phi_2 \mid \hat{\mathsf{Y}}_w \phi_2 \mid \phi_1 \mathsf{S}_w \phi_2 \mid \phi_1 \mathsf{T}_w \phi_2,\end{aligned}$$

where $w \in \mathbb{N} \cup \{+\infty\}$, $p \in \mathsf{AP}$, ϕ_1, ϕ_2 are TPTL$_\mathsf{b}$+P formulae, $x, y \in \mathsf{V}$, $m \in \mathbb{N}$, and $c \in \mathbb{Z}$. The bound on

any temporal operator can be $w = +\infty$ (or omitted) only if applied to a *closed* formula. This restriction limits any temporal modality (including future ones) to look only as far as their bound. As it will be shown later, this implies that when interpreting any timing constraint, such as, e.g., $x \leq y + c$, the timestamps x and y can be distant, at most, an amount of time which is exponential in the size of the formula.

Formally, the semantics of $\mathsf{TPTL_b{+}P}$ is defined as follows. Let ρ be a timed state sequence and let ξ be an environment. We say that ρ satisfies a $\mathsf{TPTL_b{+}P}$ formula ϕ at position $i \geq 0$ with environment ξ, written $\rho \models^i_\xi \phi$, if (and only if):

1. $\rho \models^i_\xi \mathsf{X}_w \phi_1$ iff $\tau_{i+1} - \tau_i \leq w$ and $\rho \models^{i+1}_\xi \phi_1$;
2. $\rho \models^i_\xi \hat{\mathsf{X}}_w \phi_1$ iff $\tau_{i+1} - \tau_i \leq w$ implies $\rho \models^{i+1}_\xi \phi_1$;
3. $\rho \models^i_\xi \phi_1 \mathsf{U}_w \phi_2$ iff there exists $j \geq i$ such that: (i) $\tau_j - \tau_i \leq w$, (ii) $\rho \models^j_\xi \phi_2$, and (iii) $\rho \models^k_\xi \phi_1$ for all $i \leq k < j$;
4. $\rho \models^i_\xi \phi_1 \mathsf{R}_w \phi_2$ iff either (i) $\tau_j - \tau_i \leq w$ implies $\rho \models^j_\xi \phi_2$ for all $j \geq i$, or (ii) there exists $k \geq i$ such that $\tau_k - \tau_i \leq w$ and $\rho \models^k_\xi \phi_1$, and $\rho \models^j_\xi \phi_2$ for all $i \leq j \leq k$;
5. $\rho \models^i_\xi \mathsf{Y}_w \phi_1$ iff $i > 0$, $\tau_i - \tau_{i-1} \leq w$, and $\rho \models^{i-1}_\xi \phi_1$;
6. $\rho \models^i_\xi \hat{\mathsf{Y}}_w \phi_1$ iff $i > 0$ and $\tau_i - \tau_{i-1} \leq w$ imply $\rho \models^{i-1}_\xi \phi_1$;
7. $\rho \models^i_\xi \phi_1 \mathsf{S}_w \phi_2$ iff there exists $j \leq i$ such that: (i) $\tau_i - \tau_j \leq w$, (ii) $\rho \models^j_\xi \phi_2$, and (iii) $\rho \models^k_\xi \phi_1$ for all $j < k \leq i$;
8. $\rho \models^i_\xi \phi_1 \mathsf{T}_w \phi_2$ iff either (i) $\tau_i - \tau_j \leq w$ implies $\rho \models^j_\xi \phi_2$ for all $0 \leq j \leq i$, or (ii) there exists $k \leq i$ such that $\tau_i - \tau_k \leq w$ and $\rho \models^k_\xi \phi_1$, and $\rho \models^j_\xi \phi_2$ for all $k \leq j \leq i$;
9. same semantics as $\mathsf{TPTL{+}P}$ for the remaining operators.

In addition to the bounded versions of all the temporal operators of $\mathsf{TPTL{+}P}$, $\mathsf{TPTL_b{+}P}$ includes a *weak* version of both the *tomorrow* and *yesterday* ones. While the formula $\mathsf{X}_w \phi$ (resp., $\mathsf{Y}_w \phi$) require the next (resp., previous) state to be distant at most w time steps and to satisfy ϕ, the *weak tomorrow* (resp., *yesterday*) operator in a formula of the form $\hat{\mathsf{X}} \phi$ (resp., $\hat{\mathsf{Y}} \phi$), requires the next (resp., previous) state to satisfy ϕ *only if* such a state exists and its distance is at most w. The weak tomorrow and yesterday operators are introduced as *duals* of the standard ones, in such a way that $\neg \mathsf{X}_w \phi \equiv \hat{\mathsf{X}}_w \neg \phi$ and $\neg \hat{\mathsf{X}}_w \phi \equiv \mathsf{X}_w \neg \phi$ (and similarly for the yesterday ones). This ensures that each temporal modality has its own negated dual (such as the *until*/*release* and *since*/*triggered* pairs), so that any $\mathsf{TPTL_b{+}P}$ formula can be put into *negated normal form*, where negations are only applied to proposition letters and timing constraints. The existence of a negated normal form for $\mathsf{TPTL_b{+}P}$ formulae will play an important role in the definition of the tableau system (see Section 3.3).

3 The tableau systems for TPTL and $\mathsf{TPTL_b{+}P}$

This section describes the one-pass and tree-shaped tableau systems for TPTL and $\mathsf{TPTL_b{+}P}$, that respectively extend those for LTL and $\mathsf{LTL{+}P}$ presented in [5, 6]. Soundness and completeness of the systems are proved in Section 4. The two systems are very similar, differing only in specific parts and sharing the vast majority of their workings. Hence, a common skeleton is first described, making some assumptions that will then be fulfilled for the two specific logics.

3.1 The common skeleton

The parts in common between the two tableau systems will be presented as if they were supposed to handle TPTL+P formulae. TPTL is a proper fragment of TPTL+P, and TPTL$_b$+P, as it will be shown later, can be fully embedded in a proper guarded fragment of TPTL+P. Hence, both tableaux do indeed handle TPTL+P formulae, albeit of a specific kind. We will mention the specific logics when stating results that are not proved for the full TPTL+P logic.

W.l.o.g, we may assume formulae to be in *negated normal form*, which is guaranteed to exist for formulae of both logics. As shown in [2] for TPTL and in [5] for TPTL$_b$+P, w.l.o.g., we can also restrict ourselves to models with a bound on the maximum *temporal* distance between two subsequent states.

Proposition 1 (δ-bounded models [2, 5])**.** *Let ϕ be a closed TPTL or TPTL$_b$+P formula. A model $\rho = \langle \sigma, \tau \rangle$ of ϕ is said to be* δ-bounded*, for some $\delta \geq 0$, if $\tau_{i+1} - \tau_i \leq \delta$ for all $i \geq 0$. Then, it holds that ϕ is satisfiable if and only if there exists some $\delta_\phi \geq 0$ such that ϕ has a δ_ϕ-bounded model.*

In [2, 5], it is shown how to compute δ_ϕ starting from the constants appearing in ϕ: roughly, δ_ϕ is the product of all the constants in ϕ. Similarly, we can assume that no *absolute* timing constraints (those of the form $x \leq c$) are used in the formulae (see Lemma 6 in [2]). W.l.o.g., we can also assume that any variable x is used only in one freeze quantifier in any formula, so that in a formula like $x.\psi$ any occurrence of x in ψ is free. Since freeze quantifiers can be pushed out of boolean connectives, when talking about closed formulae we will write them as $x.\psi$, with explicit reference to the outermost freeze quantifier.

We start by defining an important building block of the system.

Definition 2 (Temporal shift)**.** *Let us denote as* wwf *the set of all the well-formed TPTL+P formulae. The* temporal shift operator *is a function $\cdot^\delta : \mathsf{wwf} \times \mathbb{Z} \to \mathsf{wwf}$ such that:*

1. *for any closed TPTL+P formula $x.\psi$ and any $\delta \in \mathbb{Z}$, timed state sequence ρ, environment ξ, and position $i \geq 0$, it holds that $\rho \models^i_\xi x.\psi^\delta$ if and only if $\rho \models^i_{\xi'} \psi$, where $\xi' = \xi[x \leftarrow \tau_i - \delta]$;*
2. *there exists $\delta' \in \mathbb{Z}$ such that $x.\psi^{\delta'} = x.\psi^{\delta'+i}$ and $x.\psi^{-\delta'} = x.\psi^{-\delta'-i}$ for all $i \geq 0$.*

Item 1 of Definition 2 states that the truth value of $x.\psi^\delta$, interpreted at the current state, is the same as that of ψ in the case where x were bound to the timestamp of a previous state located exactly δ time units before. By Item 2, this transformation has to be defined in such a way that it converges to a fixed point after a large enough amount of shifting, so that for a given $x.\psi$, the number of different formulae of the form $x.\psi^\delta$ is finite. It is not known whether such an operator exists for full TPTL+P. Later, we will show how to define it in the cases of TPTL and TPTL$_b$+P.

The *closure* of a formula $z.\phi$ contains all the formulae that are relevant to the satisfaction of $z.\phi$.

Definition 3 (Closure of a formula)**.** *Let $z.\phi$ be a closed TPTL+P formula and let $\cdot^\delta$ be a temporal shift operator. Then, the* closure *of $z.\phi$ is the set $\mathcal{C}(z.\phi)$ recursively defined as follows:*

1. *$z.\phi \in \mathcal{C}(z.\phi)$;*
2. *if $x.(\psi_1 \wedge \psi_2) \in \mathcal{C}(z.\phi)$, then $\{x.\psi_1, x.\psi_2\} \subseteq \mathcal{C}(z.\phi)$;*
3. *if $x.(\psi_1 \vee \psi_2) \in \mathcal{C}(z.\phi)$, then $\{x.\psi_1, x.\psi_2\} \subseteq \mathcal{C}(z.\phi)$;*
4. *if $x.\mathsf{X}\psi \in \mathcal{C}(z.\phi)$, then $x.\psi^\delta \in \mathcal{C}(z.\phi)$, for all $\delta \geq 0$;*
5. *if $x.\mathsf{Y}\psi \in \mathcal{C}(z.\phi)$, then $x.\psi^{-\delta} \in \mathcal{C}(z.\phi)$, for all $\delta \geq 0$;*
6. *if $x.(\psi_1 \circ \psi_2) \in \mathcal{C}(z.\phi)$, where $\circ \in \{\mathsf{U}, \mathsf{R}, \mathsf{S}, \mathsf{T}\}$, then $\{x.\psi_1, x.\psi_2, x.\mathsf{X}(\psi_1 \circ \psi_2)\} \subseteq \mathcal{C}(z.\phi)$;*
7. *if $x.y.\psi \in \mathcal{C}(z.\phi)$, then $x.\psi[y/x] \in \mathcal{C}(z.\phi)$.*

Name	Rule		
CONJUNCTION	$x.(\psi_1 \wedge \psi_2)$	$\rightarrow$	$\{x.\psi_1, x.\psi_2\}$
FREEZE	$x.y.\psi_1$	$\rightarrow$	$\{x.\psi_1[y/x]\}$
DISJUNCTION	$x.(\psi_1 \vee \psi_2)$	$\rightarrow$	$\{x.\psi_1\} \mid \{x.\psi_2\}$
UNTIL	$x.(\psi_1 \,\mathsf{U}\, \psi_2)$	$\rightarrow$	$\{x.\psi_2\} \mid \{x.\psi_1, x.\mathsf{X}(\psi_1 \,\mathsf{U}\, \psi_2)\}$
SINCE	$x.(\psi_1 \,\mathsf{S}\, \psi_2)$	$\rightarrow$	$\{x.\psi_2\} \mid \{x.\psi_1, x.\mathsf{Y}(\psi_1 \,\mathsf{S}\, \psi_2)\}$
RELEASE	$x.(\psi_1 \,\mathsf{R}\, \psi_2)$	$\rightarrow$	$\{x.\psi_1, x.\psi_2\} \mid \{x.\psi_2, x.\mathsf{X}(\psi_1 \,\mathsf{R}\, \psi_2)\}$
TRIGGERED	$x.(\psi_1 \,\mathsf{T}\, \psi_2)$	$\rightarrow$	$\{x.\psi_1, x.\psi_2\} \mid \{x.\psi_2, x.\mathsf{Y}(\psi_1 \,\mathsf{T}\, \psi_2)\}$

Table 1: Expansion rules.

Note that, if $\cdot^\delta$ is a temporal shift operator, then $\mathcal{C}(z.\phi)$ is a finite set, thanks to Item 2 of Definition 2. Moreover, note that, by construction, every formula in $\mathcal{C}(z.\phi)$ is a *closed* formula.

Now we can effectively start describing the one-pass and tree-shaped tableau system for TPTL+P. The tableau for a closed formula $z.\phi$ is a tree where each node u of the tree is labelled with a *finite* set $\Gamma(u) \subseteq \mathcal{C}(z.\phi)$. Additionally, a non-negative integer $\mathsf{time}(u) \in \mathbb{N}$ is associated with each node u. Given two nodes u and v, we write $u \leq v$ ($u < v$) if u is a (proper) ancestor of v. The root note u_0 is labelled by the formula itself, i.e., $\Gamma(u_0) = \{z.\phi\}$, and is set at $\mathsf{time}(u_0) = 0$. The tableau is built top-down, from the root to the leaves, performing a state-by-state search for a model of the formula where each accepted branch of the complete tableau corresponds to a satisfying model. At each step, a set of *expansion rules* is applied to the leaf nodes of the tree, until no expansion rule can be applied anymore. Each application of an expansion rule results in the addition of one or more children to the selected node, making the tree grow and refining the choice of which formulae of the closure have to hold at the current state. Then, a set of *termination rules* decides if the current tableau branch has to be *accepted* (✓), *rejected* (✗), or if the branch can continue to be explored, making a step to the next state. Expansion rules are shown in Table 1. Each rule of the form $\psi \rightarrow \Delta'$ is applied to any node u such that $\psi \in \Gamma(u)$ and causes the addition of a child u' of u such that $\Gamma(u') = (\Gamma(u) \setminus \{\psi\}) \cup \Delta'$. Similarly, a rule of the form $\psi \rightarrow \Delta' \mid \Delta''$ causes the addition of two children u' and u'', where $\Gamma(u') = (\Gamma(u) \setminus \{\psi\}) \cup \Delta'$ and $\Gamma(u'') = (\Gamma(u) \setminus \{\psi\}) \cup \Delta''$.

By construction, repeatedly applying expansion rules will eventually result into leaves labelled only by proposition letters, timing constraints, or formulae of the forms $x.\mathsf{X}\psi$ or $x.\mathsf{Y}\psi$, which cannot be further expanded. Formulae of this kind are called *elementary formulae*, and a node (resp., a leaf) whose label contains only elementary formulae is a *poised node* (resp., poised leaf).

When a poised leaf is obtained, the search can proceed to the next temporal state. The formulae labelling the current state are used to determine the label of the next one. Moreover, an amount of time has to be guessed to choose the timestamp of the next state. This operation is performed by the STEP rule.

STEP Let u be a poised node, and let $\delta_{z.\phi} \geq 0$ be the bound as computed in Proposition 1. Then, $\delta_{z.\phi} + 1$ children nodes $u_0, \ldots, u_{\delta_{z.\phi}}$ are added to u, such that:

$$\begin{aligned} \Gamma(u_\delta) &= \{x.\psi^\delta \mid x.\mathsf{X}\psi \in \Gamma(u)\} \\ \mathsf{time}(u_\delta) &= \mathsf{time}(u) + \delta \end{aligned} \qquad \text{for all } 0 \leq \delta \leq \delta_{z.\phi}$$

The STEP rule is one of the most evident differences between the tableau system for TPTL and TPTL$_\mathsf{b}$+P, and those for LTL and LTL+P, since here we have to handle the advancement of the timestamp of the next state. The formulae in the subsequent state, which are taken from the *tomorrow* formulae of the current one, are shifted accordingly.

Besides the children added to by the STEP rule, others can be subsequently added to a poised node, as it will be shown later, if it does not fulfil some *past* request coming from the next state. Given a branch $\overline{u} = \langle u_0, \ldots, u_n \rangle$ and a poised node u_i, with $0 \leq i < n$, u_i is said to be a *step node* for the branch $\overline{u}$ if its child u_{i+1} has been added by the STEP rule. Moreover, if u_i is a step node for the branch $\overline{u}$, we define $\Delta(u_i) = \bigcup_{j<k\leq i} \Gamma(u_k)$, where u_j is the closest step node among the proper ancestors of u_i.

In any case, *before* applying the STEP rule to advance to the next state, the branch has to be checked for contradictions and any other condition that can cause it to be rejected or accepted. To this end, the following *termination rules* are applied to poised leaves. In what follows, any formula $x.\psi \in \mathcal{C}(\phi)$ of the form $x.\mathsf{X}(\psi_1 \mathsf{U} \psi_2)$ is called an X-eventuality. Let $\overline{u} = \langle u_0, \ldots, u_n \rangle$ be a branch of the tableau. An X-eventuality $x.\psi = x.\mathsf{X}(\psi_1 \mathsf{U} \psi_2)$ is said to be *requested* at position i if $x.\psi \in \Gamma(u_i)$, and *fulfilled* at position $j > i$ if $x.\psi_2^{\delta_j} \in \Gamma(u_j)$ and $x.\psi_1^{\delta_k} \in \Gamma(u_k)$, for all $i < k < j$, where $\delta_l = \mathsf{time}(u_l) - \mathsf{time}(u_i)$, for $i < l \leq j$.

CONTRADICTION Let u be a poised leaf. If $\{x.p, x.\neg p\} \subseteq \Gamma(u)$, for some $p \in \mathsf{AP}$, then u is crossed and the branch is rejected.

EMPTY Let u be a poised leaf such that $\Gamma(u) = \emptyset$. Then, u is ticked and the branch is accepted.

SYNC Let u be a poised node. If either $x.(x \leq x+c) \in \Gamma(u)$, $x.\neg(x \leq x+c) \in \Gamma(u)$, $x.(x \equiv_m x+c) \in \Gamma(u)$, or $x.\neg(x \equiv_m x+c) \in \Gamma(u)$, but, respectively, $c < 0$, $c \geq 0$, $c \not\equiv_m 0$, or $c \equiv_m 0$, then u is crossed and the branch is rejected.

YESTERDAY Let v be a poised leaf such that $x.\mathsf{Y}\,\psi \in \Gamma(v)$ for some $x.\psi \in \mathcal{C}(z.\phi)$. If v is the first *step node* of its branch, then it is crossed and the branch is rejected. Otherwise, let $u < v$ be the closest *step node* among the proper ancestors of v, $\delta_{u,v} = \mathsf{time}(v) - \mathsf{time}(u)$, and $\Omega = \{x.\psi^{-\delta_{u,v}} \mid x.\mathsf{Y}\,\psi \in \Gamma(v)\}$. If $\Omega \not\subseteq \Delta(u)$, then v is crossed, the branch is rejected, and a child u' is added to u such that $\Gamma(u') = \Gamma(u) \cup \Omega$.

LOOP Let v be a poised leaf, and $u < v$ a *step node*, proper ancestor of v, such that $\Gamma(u) = \Gamma(v)$ and all the X-eventualities requested in u are fulfilled between u and v (included). Then,

LOOP_1 if $\mathsf{time}(u) = \mathsf{time}(v)$, then v is crossed and the branch rejected;

LOOP_2 if $\mathsf{time}(u) < \mathsf{time}(v)$, then v is ticked and the branch accepted.

PRUNE Let w be a poised leaf. If there exist three *step nodes* $u < v < w$ such that $\Gamma(u) = \Gamma(v) = \Gamma(w)$, and each X-eventuality requested in u and fulfilled between v and w is also fulfilled between u and v, then, w is crossed and the branch rejected.

The above rules resemble the structure of the one-pass and tree-shaped tableau for LTL+P presented in [6], but adapted to the new logic. The SYNC rule has been added to the termination rules to detect contradictory timing constraints. The STEP rule, thanks to the temporal shift operator, can push freeze quantifiers to the next state, without explicitly keeping track of variable bindings. In such a way, it ensures that nodes are labelled only by closed formulae of the form $x.\psi$, the base case of timing constraints consisting only in formulae of the form $x.(x \sim x+c)$, which involve a single variable. Judging the validity of the constraints is then trivial. This mechanism was originally exploited in the graph-shaped tableau for TPTL given in [2]. The LOOP rule handles the case where the branch is cycling through a segment which fulfils all the requests, and thus a satisfying model of the formulae has been found. However, since timed state sequences must satisfy the *progress* property, the rule has to reject those branches where the loop has not advanced in time (LOOP_1) and to accept a branch only if some progress has been made (LOOP_2). In Fig. 1, we give a brief example of tableau for the TPTL formula $x.\mathsf{G}\,y.(p \rightarrow y \leq x+2)$, which expresses the property that p holds only on states with timestamp less than 2. Firstly we focus on node u_2: it is

crossed by the LOOP_1 rule because there is another node (*i.e.*, u_1) such that all the conditions of the LOOP rule are satisfied but time does *not* increase between these two nodes. Nevertheless, if we choose to increment by one time unit the candidate model by means of the STEP rule, we eventually reach node u_3, which does not contain any timed constraint, since they all have been simplified by the temporal shift $\cdot^{\delta}$. Now the LOOP_2 can be applied on node u_4, since nodes u_3 and u_4 have the same label, all the X-eventualities (there are none) are fulfilled in between, and the time between u_3 and u_4 *does* increase: thus, we tick u_4 and accept the corresponding branch. This, in turn, corresponds to a correct model of the input formula which starts from the root of the tableau, goes down to u_4 and then cycles between u_3 and u_4.

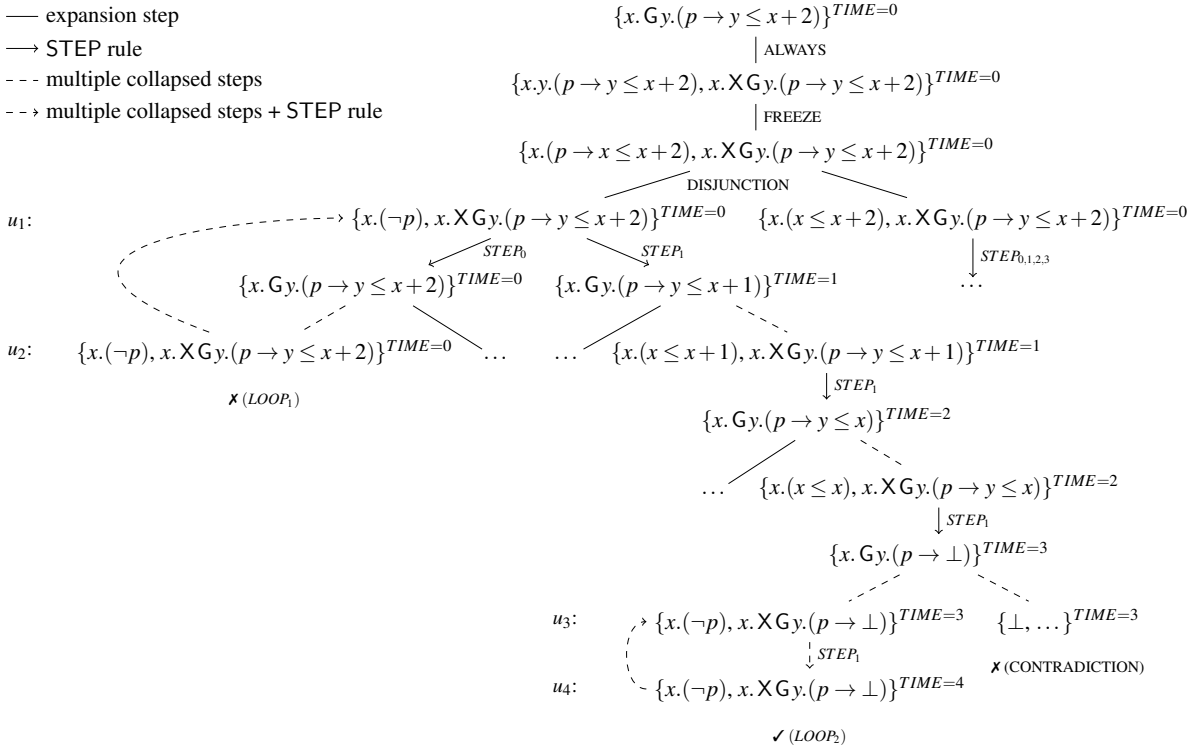

Figure 1: The tableau for the formula $x.\mathsf{G}y.(p \to y \leq x+2)$

The PRUNE rule handles the case where the branch is cycling without being able to fulfil all the requests, possibly because some of them are unsatisfiable. This rule was the main novelty of the one-pass and tree-shaped tableau system for LTL in [11], and, notably, it does not need to be changed at all to work for TPTL+P as well. An interesting example showing an application of the PRUNE rule in the context of a tableau for LTL is shown in [3].

Note that, supposing to employ a proper temporal shift operator, the set of Definition 3 is finite. This fact allows us to prove the termination of the construction of the tableau with a simple argument.

Theorem 4 (Termination of tableau construction)**.** *Let $z.\phi$ be a closed TPTL+P formula and let $\cdot^{\delta}$ be a proper temporal shift operator. Then, the construction of a complete tableau for $z.\phi$, built with $\cdot^{\delta}$, always terminates in a finite number of steps.*

Proof. First, observe that the tableau for a TPTL+P formula $z.\phi$ has a finite branching factor, since all the expansion rules create at most two children for any node, and the number of children created by the

STEP rule is bounded by δ_ϕ. Other children may be added to a poised node by failed instances of the YESTERDAY rule, but since $\mathcal{C}(z.\phi)$ is finite, the number of possible different labels is finite, and since the rule never creates two nodes with the same label, then the number of children added in this way is finite as well. Thus, by König's lemma, for the construction to proceed forever the tree should contain at least one infinite branch. However, since the number of possible labels is finite, two nodes with the same label are guaranteed to appear, and if they do not trigger the LOOP rule, then, after a finite number of repetitions of the same label, the PRUNE rule is guaranteed to be eventually triggered because the different combinations of X-eventualities satisfied between either two of those nodes is finite as well. □

3.2 The tableau system for TPTL

Let us now specialise the above general rules to TPTL formulae. Basically, we need to define a proper temporal shift operator. Consider a formula $x.\psi \in \mathcal{C}(z.\phi)$, and any other variable y appearing in ψ. Since $x.\psi$ is a closed formula, y must be quantified inside ψ, and, being ψ a future-only formula, it can only be bound to a timestamp greater than or equal to x. Hence, any timing constraint of the form $x \leq y+c$, with $c \geq 0$, always holds regardless of the specific evaluation of the variables. A similar consideration can be made for timing constraints of the form $y \leq x+c$, with $c < 0$, which are always false. This fact, originally observed in [2], leads to the following definition of the temporal shifting operator for TPTL formulae.

Definition 5 (Temporal shift operator for TPTL formulae [2])**.** *Let $x.\psi$ be a* closed *TPTL formula and $\delta \in \mathbb{N}$. Then, $x.\psi^\delta$ is the formula obtained by applying the following steps:*

1. *replace any timing constraint of the forms $x \leq y+c$, $y \leq x+c$, and $x \equiv_m y+c$, for any other variable $y \in \mathsf{V}$, by, respectively, $x \leq y+c'$, $y \leq x+c''$, and $x \equiv_m y+(c' \mod m)$, where $c' = c+\delta$ and $c'' = c-\delta$; and then*
2. *replace any timing constraint of the forms $x \leq y+c'$ and $y \leq x+c''$, with $c' \geq 0$ and $c'' < 0$, by, respectively, $\top$ and $\bot$.*

The one-pass and tree-shaped tableau system for TPTL is obtained from the set of rules of Section 3.1 by considering the temporal shift operator of Definition 5. It can be easily checked that Definition 5 satisfies the requirements of Definition 2 for any non-negative $\delta \geq 0$. Since the YESTERDAY rule never comes into play with TPTL formulae, this is sufficient, as the proofs in Section 4 will confirm.

3.3 The tableau system for TPTL$_b$+P

Let us now specialise the above set of tableau rules to TPTL$_b$+P. TPTL$_b$+P is not a proper fragment of TPTL+P *as-is*, and thus it may seem that those rules cannot be directly applied to TPTL$_b$+P formulae. However, TPTL$_b$+P can be embedded into a *guarded* fragment of TPTL+P, that is, a syntactic fragment of the logic, that we call G(TPTL+P), where each occurrence of any temporal operator is guarded by an additional formula which implements the bounded semantics of TPTL$_b$+P operators. G(TPTL+P) syntax is defined as follows:

$$\begin{aligned}\phi := {} & p \mid \neg\phi_1 \mid \phi_1 \vee \phi_2 \mid x \leq y+c \mid x \leq c \mid x \equiv_m y+c \\ & \mid x.\mathsf{X}y.(\gamma_w^{x,y} \wedge \phi_1) \mid x.\mathsf{X}y.(\gamma_w^{x,y} \rightarrow \phi_1) \mid x.\mathsf{Y}y.(\gamma_w^{x,y} \wedge \phi_1) \mid x.\mathsf{Y}y.(\gamma_w^{x,y} \rightarrow \phi_1) \\ & \mid x.\big(z.(\gamma_w^{x,z} \rightarrow \phi_1)\,\mathsf{U}\,y.(\gamma_w^{x,y} \wedge \phi_2)\big) \mid x.\big(z.(\gamma_w^{x,z} \wedge \phi_1)\,\mathsf{R}\,y.(\gamma_w^{x,y} \rightarrow \phi_2)\big) \\ & \mid x.\big(z.(\gamma_w^{x,z} \rightarrow \phi_1)\,\mathsf{S}\,y.(\gamma_w^{x,y} \wedge \phi_2)\big) \mid x.\big(z.(\gamma_w^{x,z} \wedge \phi_1)\,\mathsf{T}\,y.(\gamma_w^{x,y} \rightarrow \phi_2)\big),\end{aligned}$$

where $\gamma_w^{x,y} = y \leq x + w$, if $w \neq +\infty$, and $\gamma_w = \top$ otherwise, with $w \in \mathbb{N} \cup \{+\infty\}$ and x and y fresh in ϕ_1 and ϕ_2. Moreover, as in $\mathsf{TPTL_b{+}P}$, each temporal operator can appear with $w = +\infty$ only if the corresponding formula is closed. All the temporal operators where $w \neq +\infty$ are called *guarded*.

One can check that (i) the *negated normal form* of a $\mathsf{G(TPTL{+}P)}$ formula is still a $\mathsf{G(TPTL{+}P)}$ formula, and (ii) each $\mathsf{TPTL_b{+}P}$ formula can be translated into an equivalent $\mathsf{G(TPTL{+}P)}$ one. A notable example is the translation of the X and $\hat{\mathsf{X}}$ operators (and, symmetrically, Y and $\hat{\mathsf{Y}}$), that both get translated into a formula using a guarded X operator, but with the guard that, respectively, is conjuncted to and implies the target formula, *i.e.*, $\mathsf{X}_w\,\psi \equiv x.\mathsf{X}y.(y \leq x + w \wedge \psi)$ and $\hat{\mathsf{X}}_w\,\psi \equiv x.\mathsf{X}y.(y \leq x + w \rightarrow \psi)$, if $w \neq +\infty$, and simply $\mathsf{X}_{+\infty}\,\psi \equiv \hat{\mathsf{X}}_{+\infty}\,\psi \equiv \mathsf{X}\,\psi$ otherwise. The translation provides a sound and complete embedding of $\mathsf{TPTL_b{+}P}$ into (the $\mathsf{G(TPTL{+}P)}$ syntactic fragment of) $\mathsf{TPTL{+}P}$.

Lemma 6. *Let ϕ be a $\mathsf{TPTL_b{+}P}$ formula over the proposition letters AP and the variables V. Then, there exists a $\mathsf{G(TPTL{+}P)}$ formula ϕ' such that for any timed state sequence ρ, any environment ξ, and any $i \geq 0$, it holds that $\rho \models^i_\xi \phi$ if and only if $\rho \models^i_\xi \phi'$.*

Hence, we can apply the general tableau rules to the $\mathsf{G(TPTL{+}P)}$ translation of any $\mathsf{TPTL_b{+}P}$ formula, provided that, similar to the TPTL case, a proper temporal shift operator can be defined. This can actually be done by exploiting the following observation: thanks to the bounds applied to the $\mathsf{TPTL_b{+}P}$ temporal operators, whose semantics is implemented in $\mathsf{G(TPTL{+}P)}$ formulae by means of the guards, when interpreting a timing constraint like $x \leq y + c$, the distance between variables x and y cannot be greater than an upper bound W that depends on the bounds applied to the temporal operators of the formula. This observation was exploited in [5] to prove decidability and $\mathsf{EXPSPACE}$-completeness of $\mathsf{TPTL_b{+}P}$. Now, given a $\mathsf{G(TPTL{+}P)}$ formula $z.\phi$, let m be the number of *guarded* temporal operators used in $z.\phi$, let $w_0 = \max\{w_1, \ldots, w_m, \delta_{z.\phi}\}$, where $w_1, \ldots, w_m$ are the bounds applied to the respective guarded temporal operators and $\delta_{z.\phi}$ is computed as per Proposition 1, and let $W_{z.\phi} = w_0 \cdot (m+1)$.

Definition 7 (Temporal shift operator for $\mathsf{G(TPTL{+}P)}$ [5])**.** *Let $z.\phi$ be a* closed *TPTL formula, $\delta \in \mathbb{N}$, and $x.\psi \in \mathcal{C}(z.\phi)$. Then, $x.\psi^\delta$ is the formula obtained by applying the following steps:*

1. *replace any timing constraint of the forms $x \leq y + c$, $y \leq x + c$, and $x \equiv_m y + c$, for any other variable $y \in V$, by, respectively, $x \leq y + c'$, $y \leq x + c''$, and $x \equiv_m y + (c' \mod m)$, where $c' = c + \delta$ and $c'' = c - \delta$; and then*
2. *replace any timing constraint of the forms $x \leq y + c$ and $y \leq x + c$ either by $\top$, if $c \geq W_{z.\phi}$, or by $\bot$, if $c < -W_{z.\phi}$.*

It can be easily shown that Definition 7 defines a temporal shift operator as per Definition 2 [5] .

4 Soundness and Completeness

We now prove soundness and completeness of the tableau systems for TPTL and $\mathsf{TPTL_b{+}P}$. Given that the two systems are nearly identical, excepting for the definition of the proper temporal shift operator, both proofs will be given at once, differentiating between the two logics only when necessary.

4.1 Soundness

Here we prove that the tableau system is *sound*, that is, if a complete tableau for a formula has a successful branch, then the formula is satisfiable (and a model for the formula can be effectively extracted from the successful branch). As a preliminary step, we introduce the notion of *pre-model*: an abstract, easy to manipulate representation of a model of a formula.

Definition 8 (Atom). *An* atom *for a TPTL / TPTL$_b$+P formula $z.\phi$ is a set $\Delta \subseteq \mathcal{C}(z.\phi)$ such that:*

1. $x.p \in \Delta$ *iff* $x.\neg p \notin \Delta$, *for any proposition* $x.p \in \mathcal{C}(z.\phi)$;
2. $x.y.\psi_1 \in \Delta$ *iff* $x.\psi_1[y/x] \in \Delta$;
3. $x.(\psi_1 \wedge \psi_2) \in \Delta$ *iff* $\{x.\psi_1, x.\psi_2\} \subseteq \Delta$;
4. $x.(\psi_1 \vee \psi_2) \in \Delta$ *iff either* $x.\psi_1 \in \Delta$ *or* $x.\psi_2 \in \Delta$;
5. $x.(\psi_1 \mathsf{U} \psi_2) \in \Delta$ *iff either* $x.\psi_2 \in \Delta$ *or* $\{x.\psi_1, x.\mathsf{X}(\psi_1 \mathsf{U} \psi_2)\} \subseteq \Delta$;
6. $x.(\psi_1 \mathsf{R} \psi_2) \in \Delta$ *iff either* $\{x.\psi_1, x.\psi_2\} \subseteq \Delta$ *or* $\{x.\psi_2, x.\mathsf{X}(\psi_1 \mathsf{R} \psi_2)\} \subseteq \Delta$;
7. $x.(\psi_1 \mathsf{S} \psi_2) \in \Delta$ *iff either* $x.\psi_2 \in \Delta$ *or* $\{x.\psi_1, x.\mathsf{Y}(\psi_1 \mathsf{S} \psi_2)\} \subseteq \Delta$;
8. $x.(\psi_1 \mathsf{T} \psi_2) \in \Delta$ *iff either* $\{x.\psi_1, x.\psi_2\} \subseteq \Delta$ *or* $\{x.\psi_2, x.\mathsf{Y}(\psi_1 \mathsf{T} \psi_2)\} \subseteq \Delta$.

Intuitively, *atoms* are sets of formulae such that the presence of each non-elementary formula is justified (*i.e.*, implied) by the elementary formulae in the set, and each non-elementary formula that can be justified by the set is present.

Definition 9 (Pre-model). *Let $z.\phi$ be a closed TPTL / TPTL$_b$+P formula. A* pre-model *of $z.\phi$ is a pair $\Pi = \langle \overline{\Delta}, \overline{\iota} \rangle$, where $\overline{\iota} = \langle \iota_0, \iota_1, \ldots \rangle$ is an infinite sequence of timestamps satisfying the* progress *and* monotonicity *conditions, and $\overline{\Delta} = \langle \Delta_0, \Delta_1, \ldots \rangle$ is an infinite sequence of atoms for $z.\phi$ such that, for all $i \geq 0$,*

1. $z.\phi \in \Delta_0$;
2. *if* $x.\mathsf{X}\psi \in \Delta_i$, *then* $x.\psi^{\delta_{i+1}} \in \Delta_{i+1}$;
3. *if* $x.(\psi_1 \mathsf{U} \psi_2) \in \Delta_i$, *then there exists a* $j \geq i$ *such that* $x.\psi_2^{\delta_{i,j}} \in \Delta_j$ *and* $x.\psi_1^{\delta_{i,k}} \in \Delta_k$ *for all* $i \leq k < j$;
4. *if* $x.\mathsf{Y}\psi \in \Delta_i$, *then* $i > 0$ *and* $x.\psi^{-\delta_i} \in \Delta_{i-1}$;
5. *if* $x.(\psi_1 \mathsf{S} \psi_2) \in \Delta_i$, *then there exists a* $j \leq i$ *such that* $x.\psi_2^{-\delta_{j,i}} \in \Delta_j$ *and* $x.\psi_1^{-\delta_{k,i}} \in \Delta_k$ *for all* $j < k \leq i$,

where $\delta_0 = \iota_0$, $\delta_{i+1} = \iota_{i+1} - \iota_i$ for $i \geq 0$, and $\delta_{n,m} = \sum_{n<p\leq m} \delta_p$ for all $n, m \in \mathbb{N}$.

Pre-models take their name from the fact that they abstractly represent a model for their formula, and thus the existence of a pre-model witnesses the satisfiability of the formula.

Lemma 10. *Let $z.\phi$ be a closed TPTL / TPTL$_b$+P formula. If $z.\phi$ has a pre-model, then $z.\phi$ is satisfiable.*

Proof. Let $\Pi = (\overline{\Delta}, \overline{\iota})$ be a pre-model of $z.\phi$ and let $\rho = (\sigma, \tau)$ be a timed state sequence such that $\iota_i = \tau_i$ and $x.p \in \Delta_i$ if and only if $\rho \models^i p$. Note that each τ satisfies the monotonicity and progress conditions because $\overline{\iota}$ does by definition of pre-model. Then, we show that $\rho \models z.\phi$ and thus the formula is satisfiable.

For any $x.\psi \in \mathcal{C}(z.\phi)$, let the *nesting degree* $\deg(x.\psi)$ of $x.\psi$ be defined inductively as follows: $\deg(x.p) = \deg(x.\neg p) = 0$ for $p \in \mathsf{AP}$, $\deg(x.y.\psi) = \deg(y.\psi) + 1$, and $\deg(x(\phi_1 \circ \phi_2)) = \max(\deg(x.\psi_1), \deg(x.\psi_2)) + 1$, with $\circ \in \{\wedge, \vee, \mathsf{U}, \mathsf{S}, \mathsf{R}, \mathsf{T}\}$. We prove by induction on $\deg(x.\psi)$ that if $x.\psi \in \Delta_i$, then $\rho \models^i \psi$ for any $x.\psi \in \mathcal{C}(z.\phi)$ and any $i \geq 0$ (since all $x.\psi \in \mathcal{C}(z.\phi)$ are closed, we do not need to take care of environments). The thesis then follows from Item 1 of Definition 9, since $z.\phi \in \Delta_0$.

As for the base case, if $x.p \in \Delta_i$ or $x.\neg p \in \Delta_i$, then the thesis follows by the definition of ρ.

As for the inductive step, we go by cases:

1. if $x.y.\psi \in \Delta_i$, then $x.\psi[y/x] \in \Delta_i$ and by the inductive hypothesis $\rho \models^i x.\psi[y/x]$, thus $\rho \models^i x.y.\psi$;
2. if $x.(\psi_1 \vee \psi_2) \in \Delta_i$ (resp., $x.(\psi_1 \wedge \psi_2)$), then by definition of atom and the inductive hypothesis, either $\rho \models^i x.\psi_1$ or $\rho \models^i x.\psi_2$ (resp., both), and thus $\rho \models^i x.(\psi_1 \vee \psi_2)$ (resp., $\rho \models^i x.(\psi_1 \wedge \psi_2)$);
3. if $x.\mathsf{X}\psi \in \Delta_i$, then, by Item 2 of Definition 9, it holds that $x.\psi^{\delta_{i+1}} \in \Delta_{i+1}$. Since $\deg(x.\psi) < \deg(x.\mathsf{X}\psi)$, by the inductive hypothesis it follows that $\rho \models^{i+1}_{\xi} x.\psi^{\delta_{i+1}}$, for any ξ. By Definition 2, this implies that $\rho \models^{i+1}_{\xi[x \leftarrow \tau_{i+1} - \delta_{i+1}]} \psi$, that is, $\rho \models^{i+1}_{\xi[x \leftarrow \tau_i]} \psi$. Then, by the semantics of the *tomorrow* operator and of the freeze quantifier, we have $\rho \models^{i}_{\xi[x \leftarrow \tau_i]} \mathsf{X}\psi$ and thus $\rho \models^{i}_{\xi} x.\mathsf{X}\psi$;

4. if $x.(\psi_1 \mathsf{U} \psi_2) \in \Delta_i$, then, by definition of atom, there exists $j \geq i$ such that $x.\psi_2^{\delta_{i,j}} \in \Delta_j$ and $x.\psi_1^{\delta_{i,k}} \in \Delta_k$, for all $i \leq k < j$. Then, by the inductive hypothesis, $\rho \models^j_\xi x.\psi_2^{\delta_{i,j}}$ and $\rho \models^k_\xi x.\psi_1^{\delta_{i,k}}$, for any ξ and all $i \leq k < j$. By Definition 2, we have that $\rho \models^j_{\xi[x \leftarrow \tau_j - \delta_{i,j}]} \psi_2$ and $\rho \models^k_{\xi[x \leftarrow \tau_k - \delta_{i,k}]} \psi_1$ for all $i \leq k < j$, that is, $\rho \models^j_{\xi[x \leftarrow \tau_i]} \psi_2$ and $\rho \models^k_{\xi[x \leftarrow \tau_i]} \psi_1$ for all $i \leq k < j$. Finally, by the semantics of the *until* operator and of the freeze quantifier, we have $\rho \models^i_{\xi[x \leftarrow \tau_i]} \psi_1 \mathsf{U} \psi_2$ and thus $\rho \models^i_\xi x.(\psi_1 \mathsf{U} \psi_2)$;
5. the case when $x.(\psi_1 \mathsf{R} \psi_2) \in \Delta_i$ is similar to Item 4, and the cases when $x.\mathsf{Y}\psi \in \Delta_i$, $x.(\psi_1 \mathsf{S} \psi_2) \in \Delta_i$ or $x.(\psi_1 \mathsf{T} \psi_2) \in \Delta_i$ are similar and specular to Items 3 and 4, respectively. □

To complete the proof, it suffices to show that a pre-model for a formula can be obtained from a successful branch of the tableau.

Lemma 11. *Let $z.\phi$ be a closed TPTL or TPTL$_b$+P formula and T a complete tableau for $z.\phi$. If T has a successful branch, then there exists a pre-model for $z.\phi$.*

Proof. Let $\overline{u} = \langle u_0, \ldots, u_n \rangle$ be a successful branch of T and let $\overline{\pi} = \langle \pi_0, \ldots, \pi_m \rangle$ be the subsequence of step nodes of $\overline{u}$. Intuitively, a pre-model for $z.\phi$ can be obtained from $\overline{u}$ by building the atoms from the labels of the step nodes, and extending them to an infinite sequence. Let $\Delta(\pi_i)$ be the atom obtained from $\Gamma(\pi_i)$ by arbitrarily completing it with missing literals and closing it over the requirements of Definition 8. The sequence of $\Delta(\pi_i)$, with $0 \leq i \leq m$, forms the basic skeleton of the pre-model $\Pi = (\overline{\Delta}, \overline{\iota})$ defined as follows. As for the atoms, $\Delta_i = \Delta(\pi_{\mathsf{K}(i)})$, where $\mathsf{K} : \mathbb{N} \to \{0, \ldots, m\}$, is defined differently depending on which rule caused the branch to be accepted:

1. if π_m was ticked by the LOOP_2 rule, then there exists $k < m$ such that $\Gamma(\pi_k) = \Gamma(\pi_m)$ and all the X-eventualities requested in π_k are fulfilled between π_k and π_m. Then, the pre-model repeats forever the atoms between $\Delta(\pi_{k+1})$ and $\Delta(\pi_m)$, and thus $\mathsf{K}(i) = i$, for $0 \leq i < k$, and $\mathsf{K}(i) = k + ((i-k) \bmod T)$, with $T = m - k$, for $i \geq k$;
2. if π_m was ticked by the EMPTY rule, then $\Gamma(\pi_m) = \emptyset$ and the pre-model repeats forever the atom $\Delta(\pi_m)$, hence $\mathsf{K}(i) = i$ if $i < m$, and $\mathsf{K}(i) = m$ if $i \geq m$.

As for the sequence of timestamps, it is taken directly from the step nodes accordingly:

1. if π_m was ticked by the LOOP_2 rule, then $\iota_i = \mathsf{time}_{i-1} + (\mathsf{time}(\pi_{\mathsf{K}(i)}) - \mathsf{time}(\pi_{\mathsf{K}(i)-1}))$ for all $i \geq 0$;
2. if π_m was ticked by the EMPTY rule, then $\iota_i = \mathsf{time}(\pi_i)$ for $i \leq m$, and $\iota_{i+1} = \iota_i + 1$ for all $i > m$.

We now show that Π is indeed a pre-model for $z.\phi$. First, note that, by construction, $\overline{\iota}$ satisfies the *progress* and *monotonicity* conditions (in particular, LOOP_2 rule ensures that $\mathsf{time}(\pi_m) > \mathsf{time}(\pi_k)$. Then, observe that $z.\phi \in \Delta_0$ because $z.\phi \in \Gamma(\pi_0)$ by construction, and thus Item 1 of Definition 9 is satisfied.

Consider now any formula $x.\mathsf{X}\psi \in \Delta_i$. Being an elementary formula, we know that $x.\mathsf{X}\psi \in \Gamma(\pi_{\mathsf{K}(i)})$. Two cases have to be considered. If $\pi_{\mathsf{K}(i+1)} = \pi_{\mathsf{K}(i)+1}$, *i.e.*, the next atom comes from the actual successor of the current one in the tableau branch, then, by the STEP rule, $x.\psi^{\delta_{i+1}} \in \Delta_{i+1}$. Otherwise, $\Delta_i = \Delta(\pi_m)$ and π_m was ticked by the LOOP_2 (because Δ_i is not empty), and thus $\Delta_{i+1} = \Delta(\pi_{k+1})$ for some $k < m$ such that $\Gamma(\pi_k) = \Gamma(\pi_m)$. Hence, $x.\mathsf{X}\psi \in \Gamma(\pi_k)$ as well, and, by the STEP rule applied to π_k, $x.\psi^{\delta_{i+1}} \in \Delta(\pi_{k+1}) = \Delta_{i+1}$, and thus Item 2 of Definition 9 is satisfied.

Finally, consider any formula $x.\mathsf{Y}\psi \in \Delta_i$ and thus $x.\mathsf{Y}\psi \in \Gamma(\pi_{\mathsf{K}(i)})$. By the $\mathsf{YESTERDAY}$ rule, $i > 0$. As in the previous case, either Δ_{i-1} is the atom coming from the previous step node, and thus $x.\psi^{-\delta_i} \in \Delta_{i-1}$ by the $\mathsf{YESTERDAY}$ rule, or $\Delta_i = \Delta(\pi_{k+1})$ for some k that triggered the LOOP_2 rule because $\Gamma(\pi_k) = \Gamma(\pi_m)$. By the $\mathsf{YESTERDAY}$ rule, $x.\mathsf{Y}\psi^{-\delta_{k+1}} \in \Delta_k$, and, since $\delta_{k+1} = \delta_i$, $x.\mathsf{Y}\psi^{-\delta_i} \in \Delta_m = \Delta_{i-1}$.

The other cases are straightforward in view of how expansion rules are defined. □

Theorem 12 (Soundness)**.** *Let $z.\phi$ be a closed TPTL / TPTL$_b$+P formula, and let T be a complete tableau for $z.\phi$. If T has a successful branch, then $z.\phi$ is satisfiable.*

Proof. Extract a pre-model for $z.\phi$ from the successful branch of T as in Lemma 11, and then obtain from it an actual model for the formula as in Lemma 10. □

4.2 Completeness

We now prove the completeness of the tableau system, i.e., if a formula $z.\phi$ is satisfiable, then any complete tableau T for it has an accepting branch. We make use of a new model-theoretic argument providing a much simpler and shorter proof, which sidesteps the complex combinatorial argument used in completeness proofs for the one-pass tree-shaped tableaux for LTL [11] and LTL+P [6].

To start with, we introduce the key concept of *greedy pre-model*. Given a pre-model $\Pi = \langle \overline{\Delta}, \overline{\iota} \rangle$, an X-eventuality $x\psi = x.\mathsf{X}(\psi_1 \,\mathsf{U}\, \psi_2)$ is *requested* at position $i \geq 0$ if $x.\psi \in \Delta_i$, and *fulfilled* at $j > i$ if j is the first position where $x.\psi_2^{\delta_{i,j}} \in \Delta_j$ and $x.\psi_1^{\delta_{i,k}} \in \Delta_k$, for all $i < k < j$. Let $\mathcal{E}(z.\phi) = \{x.\psi \in \mathcal{C}(z.\phi) \mid x.\psi \text{ is an X-eventuality}\}$. For each position $i \geq 0$, we define the *delay vector at position i* as a function $\mathsf{d}_i : \mathcal{E}(z.\phi) \to \mathbb{N}$ providing a natural number for each eventuality in $\mathcal{E}(z.\phi)$, as follows:

$$\mathsf{d}_i(x.\psi) = \begin{cases} 0 & \text{if } x.\psi \text{ is not requested at position } i \\ n & \text{if } x.\psi \text{ is requested at } i \text{ and fulfilled at } j \text{ such that } n = j - i \end{cases}$$

Intuitively, $\mathsf{d}_i(x.\psi)$ is the number of states elapsed between the request and the fulfilment of $x.\psi$. We denote as $\overline{\mathsf{d}} = \langle \mathsf{d}_0, \mathsf{d}_1, \ldots \rangle$ the sequence of delay vectors of the atoms of $\overline{\Delta}$, and define $\mathsf{d}_i \preceq \mathsf{d}'_i$ if and only if $\mathsf{d}_i(\psi) \leq \mathsf{d}'_i(\psi)$, for all $\psi \in \mathcal{E}(\phi)$. A pre-order relation on pre-models of a given formula can be defined by comparing the d_i lexicographically: $\Pi \preceq \Pi'$ if $\mathsf{d}_0 < \mathsf{d}'_0$ or $\mathsf{d}_0 = \mathsf{d}'_0$ and $\Pi_{\geq 1} \preceq \Pi'_{\geq 1}$, where $\Pi_{\geq 1} = \langle \overline{\Delta}_{\geq 1}, \overline{\iota}_{\geq 1} \rangle$ with $\overline{\Delta}_{\geq 1} = \langle \Delta_1, \Delta_2, \ldots \rangle$ and $\overline{\iota}_{\geq 1} = \langle \iota_1, \iota_2, \ldots \rangle$. Greedy pre-models are minimal elements of this pre-order. We show that if a formula admits a pre-model, then it admits a greedy pre-model. The completeness result can then be proved directly.

Definition 13 (Greedy pre-models)**.** *Let Π be a pre-model for a formula $z.\phi$. Π is* greedy *if there is no pre-model $\Pi' \neq \Pi$ such that $\Pi' \preceq \Pi$.*

Lemma 14. *Let Π be a pre-model for a formula $z.\phi$. Then, there is a* greedy *pre-model $\Pi' \preceq \Pi$.*

Proof. We distinguish two cases. If there is a finite sequence $\Pi_1\ (= \Pi) \succ \Pi_2 \succ \ldots \succ \Pi_n$, with $n \geq 1$, which is maximal with respect to $\succ$, i.e., it cannot be further extended, then $\Pi' = \Pi_n$ is a greedy model with $\Pi' \preceq \Pi$. Otherwise, let $\Pi_1\ (= \Pi) \succ \Pi_2 \succ \ldots$ be an infinite sequence of pre-models. We prove that its limit is a greedy model Π'. To this end, it suffices to show that for every $n \in \mathbb{N}$ (prefix length), there is $m \in \mathbb{N}$ (pre-model index) such that the prefix up to position n of pre-models $\Pi_m, \Pi_{m+1}, \ldots$ is the same.

For $i \geq 1$, let $\mathsf{d}^i = \langle \mathsf{d}^i_0, \mathsf{d}^i_1, \ldots \rangle$ be the sequence of delay vectors of Π_i. Let us consider the j-th pre-model Π_j, for some $j \geq 1$. By definition of $\succ$, there is a position $n_j \geq 0$ such that $\mathsf{d}^{j+1}_{n_j} < \mathsf{d}^j_{n_j}$, and $\mathsf{d}^{j+1}_m = \mathsf{d}^j_m$, for all $0 \leq m < n_j$. We show that there are finitely many indexes $l > j$ (let $\overline{l}$ be the largest one) for which there exists a position n_k, with $n_k \leq n_j$, such that $\mathsf{d}^{l+1}_{n_k} < \mathsf{d}^l_{n_k}$, and $\mathsf{d}^{l+1}_m = \mathsf{d}^j_m$, for all $0 \leq m < n_k$. We prove it by contradiction. Assume that there are infinitely many. Let n_h be the leftmost position that comes into play infinitely many times. If $n_h = 0$, then there is an infinite strictly decreasing sequence of delay vectors $\mathsf{d}^{h_1}_0 > \mathsf{d}^{h_2}_0 > \mathsf{d}^{h_3}_0 > \ldots$, with $j < h_1 < h_2 < h_3 < \ldots$, which cannot be the case since the ordered set $(\mathbb{N}^{|\mathcal{E}(z.\phi)|}, \leq)$ is well-founded (the definition of temporal shift operators ensures that the closure set of $z.\phi$ is finite, and thus $\mathcal{E}(z.\phi)$ is finite as well). Let $0 < n_h \leq n_j$. Since the positions to the left of n_h

are chosen only finitely many times, there exists a tuple $(\mathsf{d}_0, \ldots, \mathsf{d}_{n_h-1})$ which is paired with an infinite strictly decreasing sequence of delay vectors $\mathsf{d}^{h_1}_{n_h} > \mathsf{d}^{h_2}_{n_h} > \mathsf{d}^{h_3}_{n_h} > \ldots$, with $j < h_1 < h_2 < h_3 < \ldots$, which again cannot be the case since the ordered set $(\mathbb{N}^{|\mathcal{E}(z.\phi)|}, \leq)$ is well-founded. This allows us to conclude that the prefix up to position n_j of all pre-models of index greater than or equal to $\bar{l}$ is the same. □

Theorem 15 (Completeness)**.** *Let $z.\phi$ be a closed TPTL / TPTL$_b$+P formula and let T be a complete tableau for $z.\phi$. If $z.\phi$ is satisfiable, then T contains a successful branch.*

Proof. Let $\rho = \langle \sigma, \tau \rangle$ be a model for $z.\phi$. It is straightforward to build a pre-model for $z.\phi$ from ρ. Then, given a pre-model for $z.\phi$, Lemma 14 ensures that a *greedy* pre-model for it exists. We can thus restrict our attention to greedy pre-models. Let $\Pi = \langle \overline{\Delta}, \bar{\imath} \rangle$ be a greedy pre-model for $z.\phi$. We look for a successful branch in T by using Π as a guide to descend down the tree until a leaf is found, showing that any leaf found in this way must be ticked. The descent proceeds as follows. At each step $i \geq 0$, we maintain a sequence of nodes (which will be the prefix of some branch of the tree) $\overline{u}_i = \langle u_0, u_1, \ldots, u_i \rangle$ that is extended to $\overline{u}_{i+1} = \langle u_0, u_1, \ldots, u_i, u_{i+1} \rangle$ by choosing u_{i+1} among the children of u_i. A map $J : \mathbb{N} \to \mathbb{N}$ is built during the descent, where initially $J(0) = 0$, which links each nodes in $\overline{u}_i$ to a position in the pre-model by maintaining the invariant that if $x.\psi \in \Gamma(u_k)$, then $x.\psi \in \Delta_{J(k)}$, for each $0 \leq k \leq i$ and each $x.\psi \in \mathcal{C}(z.\phi)$. At each step $i \geq 0$, u_{i+1} is chosen among the children of u_i in the following way: if u_i is not a poised node, u_{i+1} is chosen as any of its children u'_i such that $\Gamma(u'_i)$ satisfies the invariant. It is easy to check that at least one such child exists by construction because of how expansion rules are defined and the fact that Π is a pre-model. If, otherwise, u_i is a poised node, then it has $\delta_{z.\phi}$ children $\langle u^S_0, \ldots, u^S_{\delta_{z.\phi}} \rangle$ created by the STEP rule, and potentially other children $\langle u^Y_0, \ldots, u^Y_n \rangle$ added by failed instances of the YESTERDAY rule. If there is any u^Y_i whose label satisfies the invariant, then one of those is selected as u_{i+1}. If no such child exists, u_{i+1} is chosen according to the timestamp of the next atom in the pre-model, *i.e.*, $u_{i+1} = u^S_{\delta_{J(i+1)}}$. The invariant in this case is satisfied by construction because of the definition of the STEP rule.

Since each step always descends down the tree, we will eventually reach a leaf u_n. We now show that u_n has to be a ticked leaf. If instead u_n was crossed, it could not have been crossed by contradiction, because there would be some p and $\neg p$ in $\Delta(u_n)$ that would imply that $p \in \Delta_{J(n)}$ and $\neg p \in \Delta_{J(n)}$, which cannot be the case. Similarly, it could not have been crossed by the SYNCH rule. Furthermore, the LOOP$_1$ rule could not have crossed u_n, because the timestamps were chosen following $\bar{\imath}$, which by definition satisfies the *progress* and *monotonicity* conditions. Then, u_n has to have been crossed by the PRUNE rule, hence there exist other two nodes u_m and u_r such that $\Gamma(u_m) = \Gamma(u_r) = \Gamma(u_n)$ and all the eventualities requested in u_m and fulfilled between u_r and u_n are also fulfilled between u_m and u_r, and $\Delta_{J(m)} = \Delta_{J(r)} = \Delta_{J(n)}$. Now, it can be checked that the pre-model Π' obtained by removing all the atoms between $\Delta_{J(r)+1}$ and $\Delta_{J(n)}$ is still a pre-model for $z.\phi$. Then, we show that $\Pi' \preceq \Pi$, leading to a contradiction, since we supposed that Π was greedy.

We proceed by showing that $\mathsf{d}'_i \prec \mathsf{d}_i$, while $\mathsf{d}'_n \preceq \mathsf{d}_n$ for all $n < i$, thus implying that $\Pi' \prec \Pi$. To this end, we need to show that there is at least one X-eventuality $x.\psi$ for which $\mathsf{d}'_i(x.\psi) < \mathsf{d}_i(x.\psi)$ while the other values of the delay vector for the other eventualities remains constant. First, consider an eventuality $x.\psi$ which is requested in Δ_i, but not in Δ_j. Then, it holds that its first fulfilment happens before Δ_j and the cut between Δ_j and Δ_k cannot change its delay. Now, suppose $x.\psi$ is requested in Δ_i and Δ_j and is fulfilled between Δ_j and Δ_k. Hence, by definition of PRUNE rule, it is also fulfilled between Δ_i and Δ_j, thus again its first fulfilment after Δ_i is before Δ_j, and the cut does not change its delay. The remaining case is that of $x.\psi$ being requested in Δ_i and Δ_j but not fulfilled between them, and thus neither between Δ_j and Δ_k. At least one eventuality of this kind is required to exist by the definition of PRUNE rule. Then, since $x.\psi$ is not fulfilled before Δ_k, it must be requested there, and fulfilled later, and the cut between Δ_j and Δ_k will decrease the value of $\mathsf{d}'_i(x.\psi)$. Thus $\mathsf{d}'_i \prec \mathsf{d}_i$. Now, consider any position $n < i$. In any of those positions,

for any eventuality $x.\psi$, $\mathsf{d}_n(x.\psi)$ cannot increase because of the cut, otherwise the first fulfilment of $x.\psi$ would have been between Δ_j and Δ_k, which cannot be the case because all the eventualities fulfilled there are fulfilled also before, between Δ_i and Δ_k. Hence $d'_n = d_n$ for all $n < i$, and thus $\Pi' \prec \Pi$. □

5 Conclusions

In this paper, we developed one-pass and tree-shaped tableau systems for TPTL and TPTL$_\mathsf{b}$+P. They extend those for LTL [11] and LTL+P [6] with the ability of dealing with freeze quantifiers and timing constraints. Notably, the PRUNE rule, which was the main novelty of the one-pass and tree-shaped tableau system for LTL, did not need to be changed at all to work in the new systems. This confirms the great extensibility of this tableau system. The completeness of the PRUNE rule has been proved here with a new model-theoretic argument, much simpler than those used in the proofs for LTL and LTL+P. Whether or not such a tableau system can be extended to support full TPTL+P is still an open problem.

References

[1] R. Alur & T. A. Henzinger (1993): *Real-Time Logics: Complexity and Expressiveness*. Information and Computation 104(1), pp. 35–77, doi:10.1006/inco.1993.1025.

[2] R. Alur & T. A. Henzinger (1994): *A Really Temporal Logic*. Journal of the ACM 41(1), pp. 181–204, doi:10.1145/174644.174651.

[3] M. Bertello, N. Gigante, A. Montanari & M. Reynolds (2016): *Leviathan: A New LTL Satisfiability Checking Tool Based on a One-Pass Tree-Shaped Tableau*. In: Proc. of the 25th International Joint Conference on Artificial Intelligence, IJCAI/AAAI Press, pp. 950–956.

[4] M. D'Agostino, D.M. Gabbay, R. Hähnle & J. Posegga, editors (1999): *Handbook of Tableau Methods*. Springer, doi:10.1023/A:1017520120752.

[5] D. Della Monica, N. Gigante, A. Montanari, P. Sala & G. Sciavicco (2017): *Bounded Timed Propositional Temporal Logic with Past Captures Timeline-based Planning with Bounded Constraints*. In: Proc. of the 26th International Joint Conference on Artificial Intelligence, pp. 1008–1014, doi:10.24963/ijcai.2017/140.

[6] N. Gigante, A. Montanari & M. Reynolds (2017): *A One-Pass Tree-Shaped Tableau for LTL+Past*. In: Proc. of 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning, EPiC Series in Computing 46, pp. 456–473, doi:10.29007/3hb9.

[7] Y. Kesten, Z. Manna, H. McGuire & A. Pnueli (1993): *A Decision Algorithm for Full Propositional Temporal Logic*. In: Proc. of the 5th International Conference on Computer Aided Verification, LNCS 697, Springer, pp. 97–109, doi:10.1007/3-540-56922-7_9.

[8] O. Lichtenstein & A. Pnueli (2000): *Propositional Temporal Logics: Decidability and Completeness*. Logic Journal of the IGPL 8(1), pp. 55–85, doi:10.1093/jigpal/8.1.55.

[9] Z. Manna & A. Pnueli (1995): *Temporal Verification of Reactive Systems - Safety*. Springer, doi:10.1007/978-1-4612-4222-2.

[10] J. Christopher McCabe-Dansted & M. Reynolds (2017): *A Parallel Linear Temporal Logic Tableau*. In P. Bouyer, A. Orlandini & P. San Pietro, editors: Proceedings 8th International Symposium on Games, Automata, Logics and Formal Verification, EPTCS 256, pp. 166–179, doi:10.4204/EPTCS.256.12.

[11] M. Reynolds (2016): *A New Rule for LTL Tableaux*. In: Proc. of the 7th International Symposium on Games, Automata, Logics and Formal Verification, EPTCS 226, pp. 287–301, doi:10.4204/EPTCS.226.20.

[12] S. Schwendimann (1998): *A New One-Pass Tableau Calculus for PLTL*. In: Proc. of the 7th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods, LNCS 1397, Springer, pp. 277–292, doi:10.1007/3-540-69778-0_28.