Theoretical Computer Science 497 (2013) 84–107

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

A complete proof system for propositional projection temporal logic☆

Zhenhua Duan [a], Nan Zhang [a,*], Maciej Koutny [b]

[a] Institute of Computing Theory and Technology and ISN Lab, Xidian University, China
[b] School of Computing Science, Newcastle University, NE1 7RU, United Kingdom

ARTICLE INFO

Keywords:
Temporal logic
Axiomatization
Soundness
Completeness
Inference rule

ABSTRACT

The paper presents a proof system for Propositional Projection Temporal Logic (PPTL) with projection-plus. The syntax, semantics, and logical laws of PPTL are introduced together with an axiom system consisting of axioms and inference rules. To facilitate proofs, some of the frequently used theorems are proved. A normal form of PPTL formulas is presented, and the soundness and completeness of the proof system are demonstrated. To show how the axiom system works, a full omega regular property for the mutual exclusion problem is specified by a PPTL formula and then a deductive proof of the property is performed.

1. Introduction

Temporal Logic (TL) has been proposed and used for the specification and verification of concurrent systems for over three decades [32]. Within this paradigm, a number of temporal logics – such as Linear Temporal Logic (LTL) [32,25], Branching Time Temporal Logic (CTL) [9], and Interval Temporal Logic (ITL) [27] as well as Temporal Logic of Actions (TLA) [24] and many others [23,2,35] – have been formalized and successfully used to deal with both hardware and software systems. Basically, two verification approaches, viz. model checking [8,30] and theorem proving [5], have become popular in practice.

Model checking is an automatic verification approach based on model theory. To verify whether a system meets a property, the system is modeled as a finite transition system or automaton M, and the property is specified using a temporal logic formula P. Then a model checking procedure is employed to check whether $M \models P$. If so, the property is verified otherwise a counterexample can be found. The advantage of model checking is that the verification can be done automatically. However, it suffers from the state explosion problem. It is also less suitable for data intensive applications since the treatment of the data usually leads to infinite state spaces [26]. Two examples of successful model checking tools are SPIN [18] and SMV [26].

In the theorem proving approach, to verify properties of concurrent reactive systems, both the system behavior and the desired property are specified as formulas, say S and P, in some appropriate (temporal) logic. To demonstrate that the system satisfies the property amounts to proving that $\vdash S \rightarrow P$ is a theorem within the proof system of the logic. Generally, this verification process involves human assistance. The advantage is that theorem proving is able to verify both finite and infinite state systems, and can also be done semi-automatically. It is therefore also suitable for data intensive applications. However, in the verification process, several assertions need to be inserted in the context of the program modeling the system, and the use of theorem prover requires considerable expertise to guide and assist the verification process. Three examples of successful theorem provers are PVS [34], ACL2 [7] and Coq [4,10].

☆ This research is supported by the National Program on Key Basic Research Project of China (973 Program) Grant No. 2010CB328102, National Natural Science Foundation of China under Grant Nos. 61133001, 60910004, 60873018, 91018010 and 61003078, and ISN Lab Grant No. ISN1102001.

* Correspondence to: P.O. Box 177, Xidian University, No.2 South Tai Bai Road, Xi'an 710071, China.
E-mail addresses: zhhduan@mail.xidian.edu.cn (Z. Duan), nanzhang@mail.xidian.edu.cn (N. Zhang), maciej.koutny@ncl.ac.uk (M. Koutny).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.01.026

There are several proof systems and supporting tools for LTL, CTL, and TLA [20,33,1]. However, in the propositional case, the expressive power of these logics is weaker than interval based temporal logics such as PITL (Propositional ITL) and PPTL (Propositional Projection TL) since the former is not full omega regular expressive while the latter is [39]. Therefore, they are practically useful for the specification and verification of complex reactive systems. Note that it is the compositional operator chop (;) and the iterative operator chop-star ($*$) or projection star ($\circledast$ *prj*) that make the logics more powerful.

Within the interval based TL community, several researchers have investigated axiom systems with different extensions. Rosner and Pnueli [36] presented an axiom system for a propositional choppy logic with chop, next and until operators, and based the completeness proof on a tableau-based decision procedure. Paech [31] formalized a complete Gentzen-style proof system over finite intervals with temporal operators chop, chop-star and until. Kono [21] gave an axiom system for PITL with projection over finite time. Moszkowski [28] presented axiom systems over finite intervals for PITL and first order ITL. The propositional part was claimed to be complete but only an outline of a proof was given. Later he extended this for chop and chop-star with infinite time [29]. Bowman and Thompson presented a tableau-based decision procedure for PITL over finite intervals with projection. Subsequently, they presented a completeness proof for an axiomatization of this logic [6].

One of the extensions of ITL is the Propositional Projection Temporal Logic (PPTL) [39,15,17,42] which contains temporal operators: next ($\bigcirc$), a new projection operator (*prj*) [11,14,13,16,12] and a projection-plus operator ($\oplus$ *prj*). The new projection operator (*prj*) can be treated as a combination of the parallel operator ($||$) and the original projection operator (*proj*). In the projection construct $(P_1, \ldots, P_m)$ *prj* Q, the sub-formulas $P_1, \ldots, P_m$ and Q are autonomous, and each can specify its own interval over which it is interpreted. Q is interpreted in a parallel manner with $P_1, \ldots, P_m$ but shares only some selected states for communication while the P_i's are interpreted sequentially. Intuitively, $P_1, \ldots, P_m$ are interpreted over a series of fine-grained intervals whereas Q is interpreted over a coarse-grained, projected discrete interval. These two types of intervals share finitely or infinitely many (rendezvous) states. The P_i's and Q communicate at the rendezvous states, and Q can terminate before, after or at the same time as P_m. Therefore, the projection construct can be used to specify computations with different time scales. As a matter of fact, PPTL allows one to model concurrent computations supporting both multi-tasking and multi-processors in a special abstract computational paradigm, termed cylinder parallelism (see Section 3 for details).

In this paper, we further extend PPTL to include the projection-plus operator ($\oplus$ *prj*), and in a formula $(P_1, \ldots, (P_i, \ldots, P_j)^{\oplus}, \ldots, P_m)$ *prj* Q the sequence $P_i, \ldots, P_j$ is referred to as the iterative part. The difference between the projection and projection-plus constructs is that in $(P_1, \ldots, P_m)$ *prj* Q the formulas $P_1, \ldots, P_m$ are interpreted sequentially (each formula is interpreted only once) while in $(P_1, \ldots, (P_i, \ldots, P_j)^{\oplus}, \ldots, P_m)$ *prj* Q the iterative part can be interpreted repeatedly (at least once). Note that, in the first order case, the projection-plus is merely a derived formula in PTL. Moreover, the new projection operators can subsume chop, chop-star and the original projection (*proj*) operators. For instance (see Section 2 for details),

$$\begin{aligned} P; Q &\equiv (P, Q)\ prj\ \varepsilon \\ P^* &\equiv \varepsilon \vee ((P)^{\oplus})\ prj\ \varepsilon \\ P\ proj\ Q &\equiv (((P)^{\oplus}, r)\ prj\ (Q; r)) \wedge halt(r) \\ halt(r) &\equiv \Box(\varepsilon \leftrightarrow r). \end{aligned}$$

As a result, the extended logic PPTL is more expressive and actually represents the full omega regular language without loss of decidability [39]. A decision procedure for checking the satisfiability of PPTL with both finite and infinite models was given in [15] and, using this decision procedure, a model checking approach based on SPIN for PPTL was proposed in [38]. Hence one can verify full omega regular properties specified by PPTL formulas for concurrent systems modeled by Promela in SPIN as finite state programs. However, such a verification suffers from the state explosion problem and so is not suitable for data intensive systems. Accordingly, theorem proving is necessary in some circumstances and works as a complementary approach to model checking. This provided a motivation for the formalization of a proof system for PPTL. In this paper, we will introduce a set of axioms and inference rules for PPTL and, in order to facilitate proofs, we will demonstrate a number of useful theorems. A normal form of PPTL formulas will be proved by induction on the syntax of PPTL, and then we will show the soundness and completeness of the axiom system. To examine how the axiom system works, a theorem prover based on PVS has recently been developed, and a number of successful case studies have been carried out.

The paper is organized as follows. In the following section, the syntax, semantics and some logical laws of PPTL are presented. In Section 3, examples are given to illustrate the usefulness of the underlying logic. In Section 4, the axiom system is formalized; in particular, axioms and inference rules are given, and some theorems proved. The normal form of formulas is introduced, and then the soundness and completeness of the axiom system is demonstrated. To show how the axiom system works, an illustrative example is discussed in Section 5. Section 6 discusses related work, and final conclusions are drawn in Section 7.

2. Propositional projection temporal logic

Our underlying logic is Propositional Projection Temporal Logic (PPTL) [11,12,14,13,16] which is an extension of Propositional Interval Temporal Logic (PITL) [27].

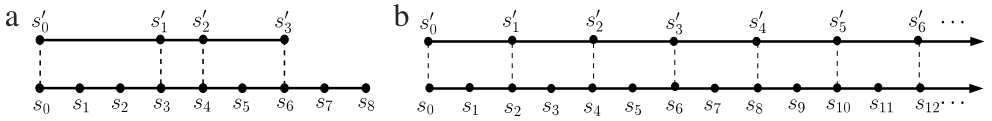

Fig. 1. Projected intervals.

2.1. Syntax

PPTL comprises propositional logic and three modal constructs used to reason about systems over time intervals. Let *Prop* be a countable set of atomic propositions, and $\mathbb{N}_0$ the set of natural numbers. The formulas of PPTL are defined as follows:

$$P ::= p \mid \bigcirc P \mid \neg P \mid P_1 \vee P_2 \mid (P_1, \ldots, P_m)\ prj\ P \mid (P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m)\ prj\ P$$

where $p \in Prop$ and $P, P_1, \ldots, P_i, \ldots, P_l, \ldots, P_m$ (for $1 \leq i \leq l \leq m$) are well-formed PPTL formulas. PPTL formulas consist of three basic temporal operators: next ($\bigcirc$), projection (*prj*) and projection-plus ($\oplus$ *prj*). A formula is called a *state formula* if it contains no temporal operators, otherwise it is called a *temporal formula*. Sometimes we use $P_{(i..j)}$ to denote $P_i, \ldots, P_j$.

2.2. Semantics

Following the definition of Kripke structures [22], we define a *state s* over *Prop* to be a mapping $s : Prop \to B$, where $B = \{true, false\}$. We use $s[p]$ to denote the truth value of p at state s.

An *interval* σ is a non-empty sequence of states, which can be finite or countably infinite. The length, $|\sigma|$, of σ is ω if σ is infinite, and the number of states minus 1 if σ is finite.

Let $\mathbb{N}_\omega = \mathbb{N}_0 \cup \{\omega\}$ be the set of natural numbers together with ω. We extend the usual comparison operators (i.e., $=$, $<$ and $\leq$) to $\mathbb{N}_\omega$ by setting $\omega = \omega$ and $i < \omega$, for all $i \in \mathbb{N}_0$. Furthermore, we define $\preceq$ as $\leq -\{(\omega, \omega)\}$. The concatenation of a finite interval σ with another interval σ' is denoted by $\sigma \cdot \sigma'$. To simplify notations, we denote σ as $\langle s_0, \ldots, s_{|\sigma|}\rangle$, where $s_{|\sigma|}$ is undefined if σ is infinite. Then:

- $\sigma_{(i..j)}$ $(0 \leq i \preceq j \leq |\sigma|)$ denotes the sub-interval $\langle s_i, \ldots, s_j\rangle$;
- σ^i $(0 \leq i \leq |\sigma|)$ denotes the prefix interval $\langle s_0, \ldots, s_i\rangle$; and
- $\sigma^{(i)}$ $(0 \leq i)$ denotes the suffix interval $\langle s_i, \ldots, s_{|\sigma|}\rangle$.

To define the semantics of projection and projection-plus, we need an auxiliary projection operation ($\downarrow$) over intervals. The idea is to construct a new coarse-grained interval, called projected interval, from a fine-grained one. Let $\sigma = \langle s_0, s_1, \ldots, s_{|\sigma|}\rangle$ be an interval and $r_1, \ldots, r_h$ ($h \in \mathbb{N}_\omega$) be integers such that $0 = r_0 \leq r_1 \leq r_2 \leq \cdots \leq r_{h-1} \leq r_h \leq |\sigma|$. Let $t = (t_1, \ldots, t_l)$ be the longest strictly increasing subsequence of $r_0, \ldots, r_h$.

- If t is finite and $t_l < \omega$, then $\sigma \downarrow (r_0, \ldots, r_h) = \sigma \downarrow t = \langle s_{t_1}, \ldots, s_{t_l}\rangle$. In this case, the projected interval is finite.
- If t is finite and $t_l = \omega$, then the projected interval is also finite and given by $\sigma \downarrow (r_0, r_1, \ldots, r_h) = \sigma \downarrow (t_1, \ldots, t_{l-1}) = \langle s_{t_1}, \ldots, s_{t_{l-1}}\rangle$.
- If t is infinite (i.e., for all $i \in \mathbb{N}_0$, $t_i < t_{i+1} < \omega$ and $\lim_{i\to\infty} t_i = \omega$), then the projected interval $\sigma \downarrow (r_0, r_1, \ldots, r_h) = \sigma \downarrow t = \langle s_{t_1}, s_{t_2}, \ldots\rangle$ is infinite.

For example, we have the following (see Fig. 1):

$$\langle s_0, \ldots, s_8\rangle \downarrow (0, 3, 3, 4, 4, 4, 6) = \langle s_0, \ldots, s_8\rangle \downarrow (0, 3, 4, 6) = \langle s_0, s_3, s_4, s_6\rangle$$
$$\langle s_0, s_1, \ldots\rangle \downarrow (0, 3, 21, 47, 47, \omega) = \langle s_0, s_1, \ldots\rangle \downarrow (0, 3, 21, 47) = \langle s_0, s_3, s_{21}, s_{47}\rangle$$
$$\langle s_0, s_1, \ldots\rangle \downarrow (0, 0, 2, 2, 4, 4, \ldots) = \langle s_0, s_1, \ldots\rangle \downarrow (0, 2, 4, \ldots) = \langle s_0, s_2, s_4, \ldots\rangle.$$

Intuitively, a projected interval is obtained by casting the original interval onto the longest strictly increasing subsequence of $r_0, r_1, \ldots, r_h$.

An *interpretation* is a triple $\mathcal{I} = (\sigma, k, j)$, where σ is an interval and $k, j \in \mathbb{N}_\omega$ $(0 \leq k \preceq j \leq |\sigma|)$. We use the notation $(\sigma, k, j) \models P$ to mean that a formula P is interpreted and satisfied over the subinterval $\langle s_k, \ldots, s_j\rangle$ of σ with the current state being s_k. The satisfaction relation ($\models$) is inductively defined in Table 1, where $(P_i, \ldots, P_j)^{(n)}$ stands for $\underbrace{P_i, \ldots, P_j, \ldots, P_i, \ldots, P_j}_{n\ times}$.

The meaning of $\neg$, $\vee$ and $\bigcirc$ is standard. However, the semantics of the projection and projection-plus constructs is somewhat involved and needs to be explained in detail. PPTL formulas can be interpreted over finite or infinite intervals. If $(P_1, \ldots, P_m)\ prj\ Q$ is interpreted over an infinite interval $(\sigma, 0, \omega)$, we know that either $r_m < |\sigma| = \omega$ or $r_m = |\sigma| = \omega$.

Table 1
Semantics of PPTL.

$\mathcal{I} \models p$	*iff*	$s_k[p] = true$, for any atomic proposition p
$\mathcal{I} \models \neg P$	*iff*	$\mathcal{I} \not\models P$
$\mathcal{I} \models \bigcirc P$	*iff*	$k < j$ and $(\sigma, k+1, j) \models P$
$\mathcal{I} \models P \vee Q$	*iff*	$\mathcal{I} \models P$ or $\mathcal{I} \models Q$
$\mathcal{I} \models (P_1, \ldots, P_m)\ prj\ Q$		
	iff	there are $k = r_0 \leq \cdots \leq r_{m-1} \leq r_m \leq j$ such that for all $1 \leq l \leq m$, $(\sigma, r_{l-1}, r_l) \models P_l$, and one of the following holds: (a) $r_m < j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_m) \cdot \sigma_{(r_m+1..j)} \models Q$ (b) $r_m = j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_h) \models Q$ for some $0 \leq h \leq m$
$\mathcal{I} \models (P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m)\ prj\ Q$		
	iff	one of the following holds: (a) $\mathcal{I} \models (P_1, \ldots, (P_i, \ldots, P_l)^{(n)}, \ldots, P_m)\ prj\ Q$ for some $n \geq 1$ (b) $l = m$ and $j = \omega$ and there exist infinitely many integers $k = r_0 \leq r_1 \leq \cdots$ and $\lim_{x\to\infty} r_x = \omega$ such that: - $(\sigma, r_{x-1}, r_x) \models P_x$ for $1 \leq x \leq i-1$ - $(\sigma, r_{i+t(l-i+1)+n-1}, r_{i+t(l-i+1)+n}) \models P_{i+n}$ for $t \geq 0$ and $0 \leq n \leq l-i$ - $\sigma \downarrow (r_0, r_1, \ldots, r_h) \models Q$ for some $h \in \mathbb{N}_\omega$

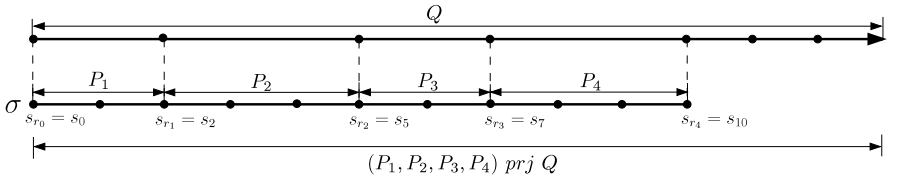

Fig. 2. Semantics of projection (1).

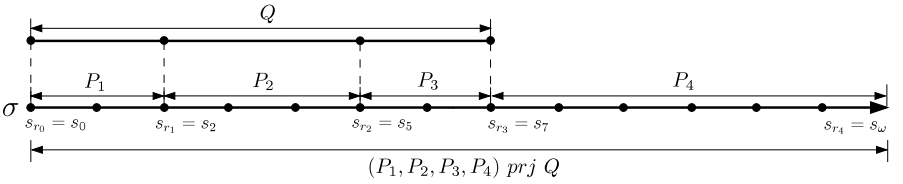

Fig. 3. Semantics of projection (2).

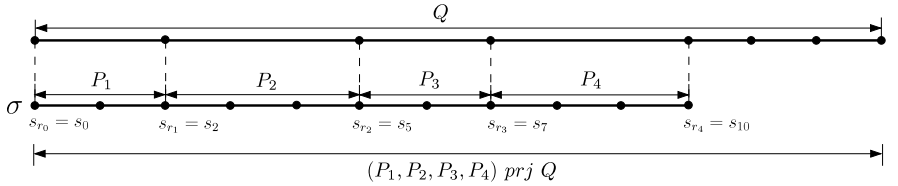

Fig. 4. Semantics of projection (3).

- If $r_m < \omega$, i.e., P_m is interpreted over a finite subinterval of σ, then Q must be interpreted over the projected interval which is the concatenation of the endpoints of the subintervals on which the P_i's are interpreted, and the infinite suffix subinterval $\langle s_{r_m+1}, \ldots \rangle$ (see Fig. 2).
- If $r_m = \omega$, i.e., P_m is interpreted over an infinite suffix subinterval of σ, then there exists some h $(0 \leq h \leq m)$ such that $\sigma \downarrow (r_0, r_1, \ldots, r_h) \models Q$. When $h = m$, from the definition of operation $\downarrow$, we have $\sigma \downarrow (r_0, r_1, \ldots, r_{m-1}, \omega) = \sigma \downarrow (r_0, r_1, \ldots, r_{m-1}) \models Q$. Since P_m is satisfied by an infinite interval, it is not possible to carry out this projection. Hence h ranges from 0 to $m-1$ (see Fig. 3).

If $(P_1, \ldots, P_m)\ prj\ Q$ is interpreted over a finite interval σ, we know that either $r_m < |\sigma| < \omega$ or $r_m = |\sigma| < \omega$.

- If $r_m < |\sigma|$, then Q must be interpreted over the projected interval which is the concatenation of the endpoints of subintervals on which the P_i's are interpreted, and the finite suffix subinterval $\sigma_{(r_m+1..|\sigma|)}$. In this case, Q must terminate after P_m (see Fig. 4).
- If $r_m = |\sigma|$, then there exists h $(0 \leq h \leq m)$ such that $\sigma \downarrow (r_0, r_1, \ldots, r_h) \models Q$. In this case, Q may terminate before P_m (see Fig. 5(a)) or together with P_m (see Fig. 5(b)).

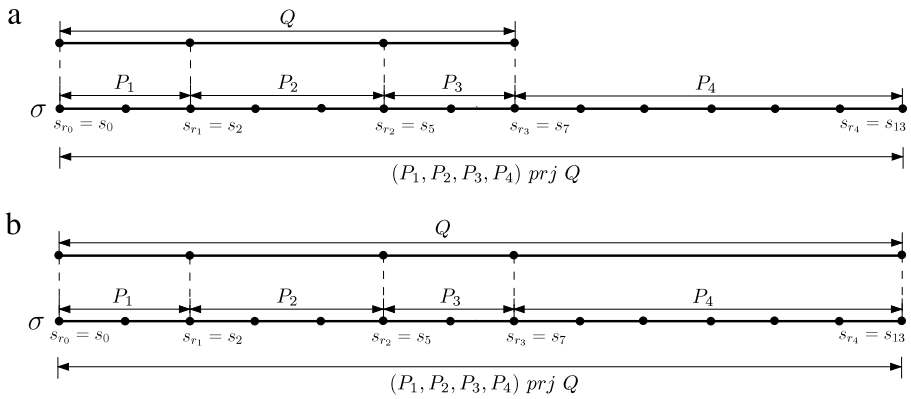

Fig. 5. Semantics of projection (4).

The projection-plus construct, $(P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m)$ *prj* Q, contains a looping part $P_i, \ldots, P_l$ which has to be interpreted at least once. If $l < m$, then the looping part is interpreted finitely many times to ensure that $P_{l+1}, \ldots, P_m$ is also interpreted. But in the construct $(P_1, \ldots, (P_i, \ldots, P_m)^{\oplus})$ *prj* Q (i.e., if $l = m$), the looping part may be interpreted infinitely many times. In this case, Q can either terminate at some point or be interpreted along with the looping part forever. The formation of the projected intervals of projection-plus is similar to that of the projection construct.

2.3. Discussion

Compared with the original projection construct P *proj* Q (i) in ITL [27] and the improved version given in [6] in which P is not allowed to be a terminal formula, the projection construct $(P_1, \ldots, P_m)$ *prj* Q (ii) enjoys some similar properties but it also has its own specific characteristics.

Similarities

- *Two scales of intervals:* The $P_1, \ldots, P_m$ in (ii) and the P in (i) are interpreted over a sequence of fine-grained subintervals while both Q's are interpreted over a coarse-grained interval consisting of the endpoints of the subintervals over which the P_i's or P is interpreted.
- *Control and monitor:* In both (i) and (ii), the process (formula) Q can be viewed as a controller or monitor which manages processes $P_1, \ldots, P_m$ or P.
- *Parallelism:* The P_i's or P and Q are interpreted in a parallel manner.

Specific characteristics

- *Variety:* In practice, various kinds of processes $P_1, \ldots, P_m$ need to be involved in a projection construct. It is natural to deal with them using (ii) while it may be more difficult to render them in (i).
- *Autonomy:* In (ii), the P_i's and Q are all autonomous since each formula has the right to specify its own interval over which it is interpreted. In particular, P_m and Q can be interpreted over infinite intervals whereas $P_1, \ldots, P_{m-1}$ over finite intervals.
- *Non-simultaneous termination:* P_m and Q can terminate at different time points in (ii), while P and Q are required to terminate at the same time in (i). However, interpretations are not always so regular in practice.
- *Emptiness:* (ii) can handle terminal formulas which must be interpreted over intervals of zero length while (i) cannot easily do this. Moreover, it is unreasonable that P is restricted to a non-terminal formula in [6] since the length of an interval over which a formula is interpreted is unforeseen, and a singleton interval over which a formula is interpreted is necessary in some circumstances. In (i), P is interpreted repeatedly over a series of consecutive subintervals whose endpoints form the projected interval of Q. P is not allowed to be interpreted over subintervals of zero length as this would duplicate a state in the interpretation of Q. On the other hand, if some of the P_i's in (ii) are interpreted over subintervals of zero length, we can eliminate all the duplicate states through the projection operation ($\downarrow$) on intervals.
- *Infinity:* The new projection construct can be interpreted over both finite and infinite intervals while the original projection construct is only suitable for finite intervals. However, when we use projection to describe some non-terminating systems, the semantics of infinity is indispensable.
- *Subsumption:* As mentioned earlier, the new projection operators can subsume chop, chop-star and the original projection operators. Thus, the basic temporal operators for PPTL are next, projection and projection-plus.
- *Cylinder parallel computation:* With (ii), special parallel computation model, called cylinder computation, can be formalized. Intuitively, a main time interval is the sequence of fine-grained unit subintervals with length one, while several coarse-grained projected intervals over which processes are interpreted are in parallel with the main time interval (see Section 3).

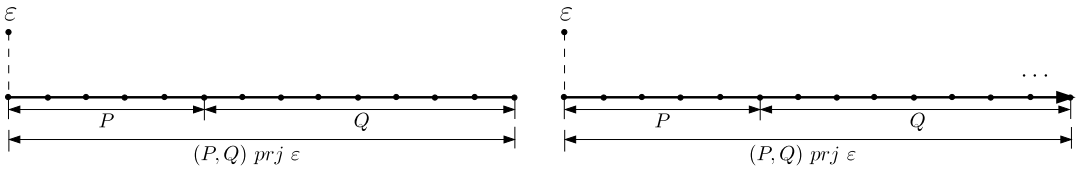

Fig. 6. Chop derived from projection.

2.4. Derived formulas

The abbreviations *true*, *false*, $\wedge$, $\rightarrow$ and $\leftrightarrow$ are defined as usual. In particular, $true \stackrel{\text{def}}{=} P \vee \neg P$ and $false \stackrel{\text{def}}{=} P \wedge \neg P$, for any formula P. Furthermore, we use the following abbreviations, where $n \in \mathbb{N}_0$:

A1	ε	$\stackrel{\text{def}}{=}$	$\neg \bigcirc true$	A2	$more$	$\stackrel{\text{def}}{=}$	$\bigcirc true$
A3	$\bigcirc^0 P$	$\stackrel{\text{def}}{=}$	P	A4	$\bigcirc^n P$	$\stackrel{\text{def}}{=}$	$\bigcirc(\bigcirc^{n-1} P)(n > 0)$
A5	$\odot P$	$\stackrel{\text{def}}{=}$	$\varepsilon \vee \bigcirc P$	A6	$P; Q$	$\stackrel{\text{def}}{=}$	$(P, Q)\ prj\ \varepsilon$
A7	$\Diamond P$	$\stackrel{\text{def}}{=}$	$true; P$	A8	$\Box P$	$\stackrel{\text{def}}{=}$	$\neg\Diamond\neg P$
A9	$len(n)$	$\stackrel{\text{def}}{=}$	$\bigcirc^n \varepsilon$	A10	$skip$	$\stackrel{\text{def}}{=}$	$len(1)$
A11	P^+	$\stackrel{\text{def}}{=}$	$((P)^{\oplus})\ prj\ \varepsilon$	A12	P^*	$\stackrel{\text{def}}{=}$	$((P)^{\circledast})\ prj\ \varepsilon$
A13	fin	$\stackrel{\text{def}}{=}$	$\Diamond\varepsilon$	A14	inf	$\stackrel{\text{def}}{=}$	$\neg fin$
A15	$P \mid\mid Q$	$\stackrel{\text{def}}{=}$	$(P; true) \wedge Q \vee P \wedge (Q; true)$				
A16	$(P_1, \ldots, (P_i, \ldots, P_l)^{\circledast}, \ldots, P_m)\ prj\ Q$						
		$\stackrel{\text{def}}{=}$	$(P_1, \ldots, \varepsilon, \ldots, P_m)\ prj\ Q \ \vee$				
			$(P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m)\ prj\ Q.$				

A6 tells us that the strong version chop construct [27] can be derived from the projection in PPTL. The interpretation of $(P, Q)\ prj\ \varepsilon$ is just to interpret P and Q sequentially, since ε can be interpreted over any singleton interval which consists of only one state (see Fig. 6). Notice that, with A12, the chop-star operator $*$ [27] derived from $\oplus\ prj$ is different from Kleene Closure for strings of languages [19]. In fact, if V is a set of characters, the definition of Kleene closure on V is the collection of all finite length strings generated from the characters in V. However, for the PPTL formula P^*, P may be interpreted repeatedly for finite or infinite number of times.

2.5. Precedence rules and properties of formulas

To avoid excessive use of parentheses, the following precedence rules are used, where 1 = highest and 9 = lowest.

1. $\neg$
2. $+,\ *$
3. $\bigcirc,\ \odot,\ \Diamond,\ \Box$
4. $\wedge$
5. $;$
6. $\vee$
7. $prj,\ \oplus\ prj,\ \circledast\ prj$
8. $\mid\mid$
9. $\rightarrow,\ \leftrightarrow$.

A formula P is satisfied by an interval σ iff $(\sigma, 0, |\sigma|) \models P$ (we denote this by $\sigma \models P$). Moreover, P is *satisfiable* iff $\sigma \models P$ for some σ, and P is *valid* iff $\sigma \models P$ for all σ (we denote this by $\models P$). We denote $\models \Box(P \leftrightarrow Q)$ by $P \equiv Q$, and $\models \Box(P \rightarrow Q)$ by $P \supset Q$.

A formula P is *terminable* iff $P \wedge \Diamond\varepsilon \not\equiv false$; otherwise it is *non-terminable*.

3. Examples

In this section, we show how projection can be used to model hardware systems and parallel computations.

3.1. Example 1: pulse generator

In the pulse generator [11,27], a variable x can assume two values: $x = 0$ or $x = 1$. There are two types of processes. The first process is $hold(i)$ $(i \geq 1)$ which is interpreted over an interval of length i and ensures that the value of x remains constant in each but the final state:

$$hold(i) \stackrel{\text{def}}{=} len(i) \wedge \Box(\bigcirc more \rightarrow (\bigcirc x = x)).$$

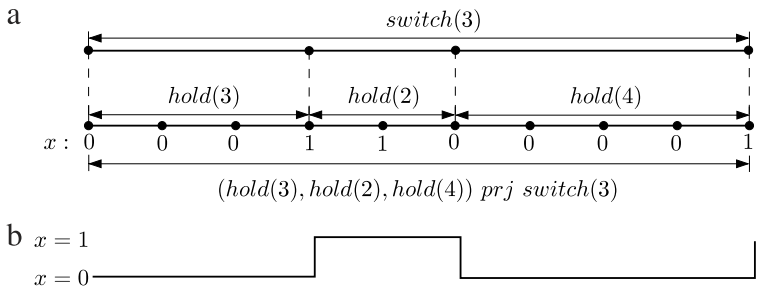

Fig. 7. Pulse generator *pulse*(3, 2, 4, 3).

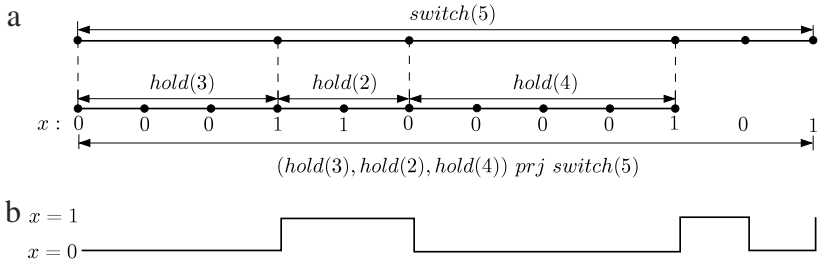

Fig. 8. Pulse generator *pulse*(3, 2, 4, 5).

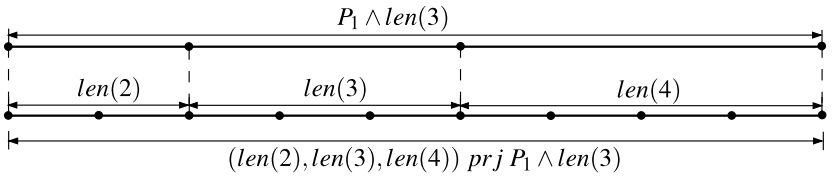

Fig. 9. P_1's projected interval.

The second process is *switch*(*j*) which ensures that the value of *x* is first set to 0 and then changed at every subsequent state:

$$switch(j) \stackrel{\text{def}}{=} (x = 0) \wedge len(j) \wedge \Box(more \rightarrow (\bigcirc x = 1 - x)).$$

Then one can define the pulse generator processes with varying number and length of low and high intervals of *x*:

$$pulse(i_1, \ldots, i_k, k+n)\ (n \geq 0) \stackrel{\text{def}}{=} \begin{cases} switch(n) & k = 0 \\ (hold(i_1), \ldots, hold(i_k)) & \\ \qquad prj\ switch(k+n) & k \geq 1. \end{cases}$$

The above formalization allows one to model, simulate and verify the design of hardware systems or embedded systems specified by Verilog codes.

The interval satisfying *pulse*(3, 2, 4, 3) is shown in Fig. 7(a), and the corresponding pulse waveform is shown in Fig. 7(b). When $n > 0$ as, for example, in *pulse*(3, 2, 4, 5), we can see that *switch*(5) terminates after *hold*(4). The interval and pulse waveform are shown in Fig. 8.

3.2. Example 2: cylindric computations

Consider the formula (*len*(2), *len*(3), *len*(4)) *prj* ($P_1 \wedge len(3)$) which allows process P_1 to be executed over a projected interval $\sigma \downarrow (0, 2, 5, 9)$, as shown in Fig. 9. Thus, the formula

(*len*(2), *len*(3), *len*(4)) *prj* ($P_1 \wedge len(3)$)
|| (*len*(2), *len*(5), *len*(6), *len*(4)) *prj* ($P_2 \wedge len(4)$)
|| (*len*(4), *len*(3), *len*(2), *len*(5)) *prj* ($P_3 \wedge len(4)$)

allows processes P_1, P_2 and P_3 to be executed in a special parallel manner in which the three processes share a common interval as their time axis, but each process runs over its own interval. The communication only takes place in the shared states including s_0, s_2, s_7 and s_9, as shown in Fig. 10. This computation model can be viewed as either three tasks that share a single processor or three processors cooperating to solve a task in a parallel way. We term this kind of parallelism a *cylinder computation*.

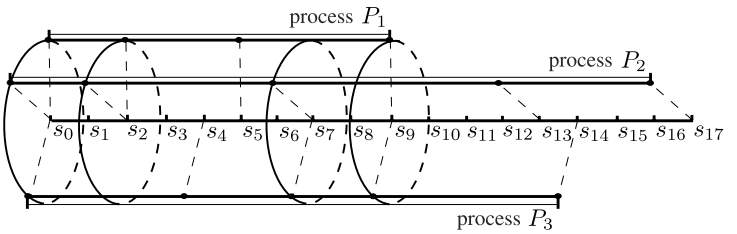

Fig. 10. Parallelism with projection.

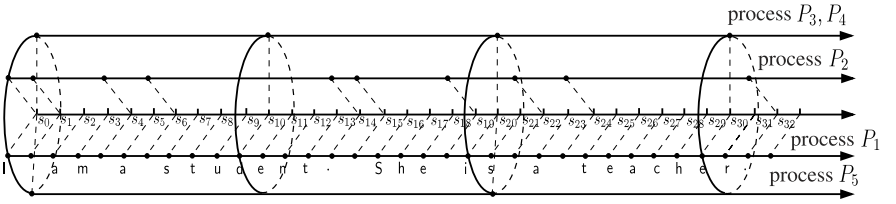

Fig. 11. A parallel model of a word processor.

Consider the outline of a word-processor given in [3]:

- Process P_1: Read characters and collect as words.
- Process P_2: Collect words to fill a line.
- Process P_3: Hyphenate, if necessary.
- Process P_4: Introduce spaces to justify the line with the right margin.
- process P_5: Collect enough lines to make a page and print the page.

The word processor can be designed as an ordinary program which is executed sequentially. However, it is not easy to program in this way since hyphenation may require a portion of a word be 'returned' to the stream of words to await for the next line. The program would be much easier to understand if it was written as a set of concurrent processes so that they can be executed in a cylinder parallel computation model, as shown in Fig. 11.

4. Proof system

In this section, a proof system Π_{pptl} of PPTL is formalized. For the convenience of deduction, we denote $\vdash P \leftrightarrow Q$ by $P \cong Q$. Moreover, for the ease of presentation, we use:

- $\Omega(0)$ to denote any sequence expression $P_1, \ldots, P_m$;
- $\Omega(1)$ to denote $P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m$ $(1 \leq i \leq l \leq m)$;
- Ω to denote $\Omega(0)$ or $\Omega(1)$; and
- τ to denote the empty formula sequence.

Sometimes, we use $\Omega(0)_\tau$, $\Omega(1)_\tau$ and Ω_τ to respectively denote sequence expressions $\Omega(0)$, $\Omega(1)$ and Ω, or the empty sequence τ. In addition, we define $(\tau)\ prj\ P$ to be P. Any sequence expression Ω joined with τ by ',' is still Ω. That is, we have: $\Omega, \tau = \tau, \Omega = \Omega$.

4.1. Axioms and inference rules

The set of axioms of Π_{pptl} is as follows, where w is a state formula and $m \geq 1$:

TAU	$\vdash \psi$ where ψ is an instance of a propositional tautology
NXN	$\vdash \bigcirc P \rightarrow \neg \bigcirc \neg P$
NXC	$\bigcirc P; Q \cong \bigcirc(P; Q)$
ECI	$\vdash P \wedge \Diamond \neg P \rightarrow \Diamond(P \wedge \bigcirc \neg P)$
AIS	$\vdash P \wedge \Box(P \rightarrow Q \vee w \wedge \bigcirc^n P)$
	$\rightarrow Q \vee w \wedge ((\bigcirc^n(w \wedge \varepsilon))^+ \wedge inf \vee w \wedge (\bigcirc^n(w \wedge \varepsilon))^*; \bigcirc^n Q$
AES	$\Box w \cong w \wedge (\bigcirc(w \wedge \varepsilon))^*$
CEL	$(P_1; P_3 \wedge \bigcirc^n \varepsilon) \wedge (P_2; P_4 \wedge \bigcirc^n \varepsilon) \cong (P_1 \wedge P_2); (P_3 \wedge P_4 \wedge \bigcirc^n \varepsilon)$
CDP	$\vdash P^+; P^+ \rightarrow P^+$
PEB	$P\ prj\ \varepsilon \cong P$
PEF	$\varepsilon\ prj\ P \cong P$

PNX $(\bigcirc P, \Omega_\tau)\ prj\ \bigcirc Q \cong \bigcirc(P; (\Omega_\tau)\ prj\ Q)$

PDF $(\Omega(i)_{1\tau}, (P \vee P'), \Omega(j)_{2\tau})\ prj\ Q$
$\cong (\Omega(i)_{1\tau}, P, \Omega(j)_{2\tau})\ prj\ Q \vee (\Omega(i)_{1\tau}, P', \Omega(j)_{2\tau})\ prj\ Q\ (i+j \in \{0,1\})$

PDB $(\Omega)\ prj\ (Q \vee Q') \cong ((\Omega)\ prj\ Q) \vee ((\Omega)\ prj\ Q')$

PSM $(\Omega(i)_{1\tau}, w \wedge \varepsilon, P, \Omega(j)_{2\tau})\ prj\ Q$
$\cong (\Omega(i)_{1\tau}, w \wedge P, \Omega(j)_{2\tau})\ prj\ Q\ (i+j \in \{0,1\})$

PSB $(\Omega)\ prj\ (w \wedge Q) \cong w \wedge (\Omega)\ prj\ Q$

PSF $(w \wedge P, \Omega_\tau)\ prj\ Q \cong w \wedge (P, \Omega_\tau)\ prj\ Q$

PEE $(\Omega(i)_{1\tau}, P \wedge \Diamond\varepsilon, \Omega(j)_{2\tau})\ prj\ Q$
$\cong (\Omega(i)_{1\tau}, P, \varepsilon, \Omega(j)_{2\tau})\ prj\ Q\ (i+j \in \{0,1\})$

PEC $(P, \Omega_\tau, P')\ prj\ \varepsilon \cong (P, (\Omega_\tau, P')\ prj\ \varepsilon)\ prj\ \varepsilon \cong ((P, \Omega_\tau)\ prj\ \varepsilon, P')\ prj\ \varepsilon$

PIF $P \wedge \neg\Diamond\varepsilon\ prj\ Q \cong P \wedge \neg\Diamond\varepsilon\ prj\ Q \wedge \varepsilon$

IEC $(\Omega(0)_{1\tau}, (P_1, \ldots, P_m)^{\oplus}, \Omega(0)_{2\tau})\ prj\ \varepsilon$
$\cong (\Omega(0)_{1\tau}, (P_1; \ldots; P_m)^{+}, \Omega(0)_{2\tau})\ prj\ \varepsilon$

IUP $(\Omega(0)_{1\tau}, (\Omega(0)_2)^{\oplus}, \Omega(0)_{3\tau})\ prj\ Q$
$\cong (\Omega(0)_{1\tau}, \Omega(0)_2, \Omega(0)_{3\tau})\ prj\ Q \vee$
$(\Omega(0)_{1\tau}, \Omega(0)_2, (\Omega(0)_2)^{\oplus}, \Omega(0)_{3\tau})\ prj\ Q$

IUM $(\Omega(0)_{1\tau}, (P_1, \ldots, P_m)^{\oplus}, \Omega(0)_{2\tau})\ prj\ Q$
$\cong (\Omega(0)_{1\tau}, P_1, \ldots, P_m, \Omega(0)_{2\tau})\ prj\ Q$
$\vee \bigvee_{t=1}^{m-1} (\Omega(0)_{1\tau}, \bigwedge_{h=0}^{t-1} P_h \wedge \varepsilon, P_t \wedge \neg\varepsilon, P_{t+1}, \ldots, P_m, \Omega(0)_{2\tau})\ prj\ Q$
$\vee \left(\Omega(0)_{1\tau}, \bigwedge_{h=0}^{m-1} P_h \wedge \varepsilon, P_m \wedge \neg\varepsilon, (P_1, \ldots, P_m)^{\oplus}, \Omega(0)_{2\tau}\right)\ prj\ Q$

IEU $(\Omega(0)_{1\tau}, (\Omega(0)_2)^{\oplus}, \Omega(0)_3)\ prj\ Q$
$\cong (\Omega(0)_{1\tau}, (\Omega(0)_2)^{\circledast}, \Omega(0)_2, \Omega(0)_3)\ prj\ Q$

IFI $(\Omega(0)_{1\tau}, (\Omega(0)_{2\tau}, P)^{\oplus})\ prj\ Q$
$\cong (\Omega(0)_{1\tau}, (\Omega(0)_{2\tau}, P)^{\circledast}, \Omega(0)_{2\tau}, P)\ prj\ Q$
$\vee((\Omega(0)_{1\tau}, (\Omega(0)_{2\tau}, P \wedge \Diamond\varepsilon)^{\oplus})\ prj\ Q) \wedge \neg\Diamond\varepsilon$

IOC $\vdash ((P)^{\circledast})\ prj\ (Q; Q') \rightarrow (((P)^{\circledast})\ prj\ Q); (((P)^{\circledast})\ prj\ Q')$.

Explanations for some of projection axioms are needed. PDF and PDB describe that projection (*prj*) is distributive over disjunction ($\vee$). PSM shows that $\varepsilon \wedge w$ appearing in the middle of a sequence expression can be eliminated while the state formula w can be incorporated into the formula following $\varepsilon \wedge w$. PEE tells us that if ε appears in a sequence expression, the formula followed by ε must be interpreted over a finite interval. PSB and PSF indicates that if a state formula w conjuncted with a formula P as the first formula of a sequence expression or a projected formula, then w can be separated from P. PIF describes that if a sequence expression consists of only one formula and the formula is a non-terminable formula, the projected formula must be a terminal formula. IUP represents the semantics of the projection plus operator. IUM describes that if the iteration part in a sequence expression is interpreted over a non-empty interval, there must exist a formula in the part interpreted over a non-empty part firstly. The projection plus construct can also be represented by the projection star operator in IEU. IFI means that if the iteration part is infinitely interpreted, each interpretation must be over a finite interval.

The inference rules are as follows:

MP $\vdash P \rightarrow Q,\ \vdash P \Longrightarrow\ \vdash Q$

IMP1 $\vdash P_i \rightarrow P'_i\ (1 \le i \le m),\ \vdash Q \rightarrow Q' \Longrightarrow$
$\vdash (P_1, \ldots, P_i, \ldots, P_m)\ prj\ Q \rightarrow (P'_1, \ldots, P'_i, \ldots, P'_m)\ prj\ Q'$

IMP2 $\vdash P_i \rightarrow P'_i\ (1 \le i \le j \le m),\ \vdash Q \rightarrow Q' \Longrightarrow$
$\vdash (P_1, \ldots, (P_i, \ldots, P_j)^{\oplus}, \ldots, P_m)\ prj\ Q$
$\rightarrow (P'_1, \ldots, (P'_i, \ldots, P'_j)^{\oplus}, \ldots, P'_m)\ prj\ Q'$

ALW $\vdash P \Longrightarrow\ \vdash \Box P$

NXT1 $\vdash P_1 \wedge \cdots \wedge P_m \rightarrow Q \Longrightarrow\ \vdash \bigcirc P_1 \wedge \cdots \wedge \bigcirc P_m \rightarrow \bigcirc Q$

NXT2 $\vdash P \rightarrow (Q \vee \bigcirc P) \Longrightarrow\ \vdash P \rightarrow (\Diamond Q \vee \Box \bigcirc P)$.

Theorem proving can be applied in order to formally verify properties of systems. In the case of PPTL, both a system and desired property are specified by PPTL formulas, say S and P. The system satisfies the property if and only if we can find a formal proof of $\vdash S \rightarrow P$ in the axiom system.

A *formal proof* is a finite sequence of well-formed PPTL formulas each of which is an axiom or conclusion derived from the preceding formulas in the sequence by an inference rule. A formula is then called a *theorem* if and only if it is the last formula of a formal proof.

It is straightforward to prove within Π_{pptl} the following useful theorems needed for completeness proof:

$$\begin{array}{llll} \mathsf{T1} & \bigcirc P \wedge \bigcirc Q \cong \bigcirc(P \wedge Q) & \mathsf{T2} & \mathit{false} \cong \bigcirc \mathit{false} \\ \mathsf{T3} & \neg \odot P \cong \bigcirc \neg P & \mathsf{T4} & \Box P \cong P \wedge \odot \Box P \\ \mathsf{T5} & \Box P \wedge \Box Q \cong \Box(P \wedge Q) & \mathsf{T6} & \Box\Box P \cong \Box P \\ \mathsf{T7} & \vdash \Box P \rightarrow P. & & \end{array}$$

For example, the following is a formal proof of T1:

$$\begin{array}{lll} (1) & \vdash P \wedge Q \rightarrow P \wedge Q & \mathsf{TAU} \\ (2) & \vdash \bigcirc P \wedge \bigcirc Q \rightarrow \bigcirc(P \wedge Q) & \mathsf{NXT1}\ (1) \\ (3) & \vdash P \wedge Q \rightarrow P & \mathsf{TAU} \\ (4) & \vdash \bigcirc(P \wedge Q) \rightarrow \bigcirc P & \mathsf{NXT1}\ (3) \\ (5) & \vdash P \wedge Q \rightarrow Q & \mathsf{TAU} \\ (6) & \vdash \bigcirc(P \wedge Q) \rightarrow \bigcirc Q & \mathsf{NXT1}\ (5) \\ (7) & \vdash \bigcirc(P \wedge Q) \rightarrow \bigcirc P \wedge \bigcirc Q & (4)\ (6) \\ (8) & \bigcirc P \wedge \bigcirc Q \cong \bigcirc(P \wedge Q) & (2)\ (7). \end{array}$$

4.2. Soundness and completeness

We now introduce a normal form for PPTL formulas upon which the completeness proof is based.

Definition 1 (*Normal Form*)**.** A PPTL formula Q is in normal form if it adheres to the following syntax

$$Q_e \wedge \varepsilon \vee \bigvee_{i=1}^{r} (Q_i \wedge \bigcirc Q_i')$$

where $r \geq 1$, Q_e and the Q_i's are state formulas, whereas the Q_i''s are general PPTL formulas. Moreover, Q is in complete normal form if $\bigvee_{i=1}^{r} Q_i \equiv \mathit{true}$ and $\bigvee_{i \neq j}(Q_i \wedge Q_j) \equiv \mathit{false}$.

In the normal form, $Q_e \wedge \varepsilon$ is the *terminal product* while each $Q_i \wedge \bigcirc Q_i'$ is a *future product*. In addition, Q_e and the Q_i's are *present components* and each $\bigcirc Q_i'$ is a *next component* $(1 \leq i \leq r)$. Note that the commutativity law results in different (complete) normal forms for a given PPTL formula.

Complete normal form plays an important role in obtaining a normal form of $\neg Q$:

$$\neg Q_e \wedge \varepsilon \vee \bigvee_{i=1}^{r} (Q_i \wedge \bigcirc \neg Q_i').$$

Note that in a complete normal form, *false* may appear in some terminal product or future product. For instance, $\bigcirc P$ is already in normal form, and its complete normal form is $\mathit{false} \wedge \varepsilon \vee \mathit{true} \wedge \bigcirc P$ rather than $\mathit{true} \wedge \bigcirc P$. Thus we can get a normal form of $\neg \bigcirc P$ as $\mathit{true} \wedge \varepsilon \vee \mathit{true} \wedge \bigcirc \neg P$.

We now show that the proposed axiom system is sound.

Theorem 1 (*Soundness*)**.** *For any PPTL formula P, if* $\vdash P$*, then* $\models P$.

Proof. The proof proceeds in two steps. First, we prove that all the axioms are valid in the model theory. Then we prove that the inference rules are also true.

(TAU) The proof can be found in chapter 2 of [11].

(NXN) Let σ be an interval and $0 \leq k \preceq j \leq |\sigma|$.

$$\begin{array}{rl} & (\sigma, k, j) \models \bigcirc P \\ \Longleftrightarrow & (\sigma, k+1, j) \models P \\ \Longleftrightarrow & (\sigma, k+1, j) \not\models \neg P \\ \Longrightarrow & (\sigma, k, j) \not\models \bigcirc \neg P \\ \Longleftrightarrow & (\sigma, k, j) \models \neg \bigcirc \neg P. \end{array}$$

(ECI) Let σ be an interval and $0 \leq k \preceq j \leq |\sigma|$.

$$\begin{array}{rl} & (\sigma, k, j) \models P \wedge \Diamond \neg P \\ \Longleftrightarrow & (\sigma, k, j) \models P \text{ and there exists a minimal integer } l \text{ such that} \\ & k < l \preceq j \text{ and } (\sigma, l, j) \models \neg P \\ \Longrightarrow & (\sigma, l-1, j) \models P \text{ and } (\sigma, l, j) \models \neg P \text{ for some } l, k < l \preceq j \\ \Longleftrightarrow & (\sigma, l-1, j) \models P \wedge \bigcirc \neg P \text{ for some } l, k < l \preceq j \\ \Longleftrightarrow & (\sigma, k, j) \models \Diamond(P \wedge \bigcirc \neg P). \end{array}$$

(TRU) Since $(\bigcirc\varepsilon)^*$ can represent an arbitrary length, finite length and infinite length, $(\bigcirc\varepsilon)^*$ is satisfied, for any interval (σ, k, j).

(AES) Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$\begin{aligned}
& (\sigma, k, j) \models \Box w \\
\Longleftrightarrow\ & (\sigma, l, j) \models w \text{ for all } l,\ k \le l \preceq j \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \text{ and } k = j, \text{ or} \\
& (\sigma, k, j) \models w \text{ and } (\sigma, l, j) \models w \text{ for all } l,\ k < l \preceq j \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \wedge \varepsilon, \text{ or} \\
& (\sigma, k, j) \models w \text{ and } (\sigma, l, l) \models w \wedge \varepsilon \text{ for all } l,\ k < l \preceq j \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \wedge \varepsilon, \text{ or} \\
& (\sigma, k, j) \models w \text{ and } (\sigma, l-1, l) \models \bigcirc(w \wedge \varepsilon) \text{ for all } l,\ k < l \preceq j \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \wedge \varepsilon, \text{ or} \\
& (\sigma, k, j) \models w \text{ and } (\sigma, k, j) \models (\bigcirc(w \wedge \varepsilon))^+ \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \wedge (\bigcirc(w \wedge \varepsilon))^*.
\end{aligned}$$

(AIS) Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$\begin{aligned}
& (\sigma, k, j) \models w \wedge \Box(w \to \varepsilon \vee \bigcirc^n w) \\
\Longleftrightarrow\ & (\sigma, k, j) \models w \text{ and } (\sigma, l, j) \models w \to \varepsilon \vee \bigcirc^n w \text{ for all } l,\ k \le l \preceq j \\
\Longrightarrow\ & (\sigma, k, j) \models w \text{ and} \\
& (\sigma, k + n * i, j) \models w \text{ implies for all } i, k \le k + n * i \preceq j,\ k + n * i = j \text{ or} \\
& (\sigma, k + n * (i+1), j) \models w \text{ and } (\sigma, k + n * i, k + n * (i+1)) \models \bigcirc^n(w \wedge \varepsilon) \\
\Longrightarrow\ & (\sigma, k, j) \models w \wedge (\bigcirc^n(w \wedge \varepsilon))^*.
\end{aligned}$$

(CEL) Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$\begin{aligned}
& (\sigma, k, j) \models (P_1; P_3 \wedge \bigcirc^n\varepsilon) \wedge (P_2; P_4 \wedge \bigcirc^n\varepsilon) \\
\Longleftrightarrow\ & (\sigma, k, j) \models P_1; P_3 \wedge \bigcirc^n\varepsilon \text{ and } (\sigma, k, j) \models P_2; P_4 \wedge \bigcirc^n\varepsilon \\
\Longleftrightarrow\ & \text{for some } r_1, k \le r_1 \preceq j,\ (\sigma, k, r_1) \models P_1 \text{ and } (\sigma, r_1, j) \models P_3 \wedge \bigcirc^n\varepsilon \text{ and} \\
& \text{for some } r_2, k \le r_2 \preceq j,\ (\sigma, k, r_2) \models P_2 \text{ and } (\sigma, r_2, j) \models P_4 \wedge \bigcirc^n\varepsilon \\
\Longleftrightarrow\ & \text{for some } r = r_1 = r_2 = j - n, \\
& (\sigma, k, r) \models P_1 \wedge P_2 \text{ and } (\sigma, r, j) \models P_3 \wedge P_4 \wedge \bigcirc^n\varepsilon \\
\Longleftrightarrow\ & (\sigma, k, j) \models P_1 \wedge P_2; P_3 \wedge P_4 \wedge \bigcirc^n\varepsilon.
\end{aligned}$$

(CDP) Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$\begin{aligned}
& (\sigma, k, j) \models P^+; P^+ \\
\Longleftrightarrow\ & \text{there are } r, k \le r \preceq j, (\sigma, k, r) \models P^+ \text{ and } (\sigma, r, j) \models P^+ \\
\Longleftrightarrow\ & \text{there exist finitely many integers } k = l_0 \le l_1 \le \cdots \le l_n = r \\
& \text{such that for all } 0 < h \le n, (\sigma, l_{h-1}, l_h) \models P \text{ and} \\
& \text{there exist finitely many } r = l_n \le l_{n+1} \le \cdots \le l_{m-1} \preceq l_m = j \\
& \text{such that for all } n < h \le m, (\sigma, l_{h-1}, l_h) \models P \text{ or} \\
& \text{there are infinitely many } r = l_n \le l_{n+1} \le \cdots,\ \lim_{i\to\infty} l_i = \omega \\
& \text{such that for all } h \ge n, (\sigma, l_{h-1}, l_h) \models P \\
\Longrightarrow\ & \text{there are finitely many } k = l_0 \le l_1 \le \cdots \le l_{m-1} \preceq l_m = j \\
& \text{such that for all } 0 < h \le m, (\sigma, l_{h-1}, l_h) \models P \text{ or} \\
& \text{there are infinitely many } k = l_0 \le l_1 \le \cdots,\ \lim_{i\to\infty} l_i = \omega \\
& \text{such that for all } h > 0, (\sigma, l_{h-1}, l_h) \models P \\
\Longleftrightarrow\ & (\sigma, k, j) \models P^+.
\end{aligned}$$

Proofs of logical laws concerning the projection construct *prj* presented in [11] can easily be extended to the projection-plus construct, and so the latter are omitted.

In the following, we give the proof of characteristic logical laws of the projection-plus. For clarity of presentation, we only prove simplified form of logical laws and the conclusion can be extended to the general one.

(IEC) We prove a simplified form $((P_1, P_2)^{\oplus})\ prj\ \varepsilon \equiv ((P_1; P_2)^+)\ prj\ \varepsilon$. Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$(\sigma, k, j) \models ((P_1, P_2)^{\oplus})\ prj\ \varepsilon$$

$$
\begin{aligned}
&\Longleftrightarrow \text{there are finitely many } k = r_0 \le t_1 \le r_1 \le t_2 \le r_2 \le \cdots \le t_m \preceq r_m = j\\
&\quad \text{such that for all } 0 < h \le m,\ (\sigma, r_{h-1}, t_h) \models P_1 \text{ and } (\sigma, t_h, r_h) \models P_2, \text{ or}\\
&\quad \text{there are infinitely many } k = r_0 \le t_1 \le r_1 \le t_2 \le r_2 \le \cdots, \lim_{i\to\infty} r_i = \omega\\
&\quad \text{such that for all } h > 0,\ (\sigma, r_{h-1}, t_h) \models P_1 \text{ and } (\sigma, t_h, r_h) \models P_2\\
&\Longleftrightarrow \text{there are finitely many } k = r_0 \le t_1 \le r_1 \le t_2 \le r_2 \le \cdots \le t_m \preceq r_m = j\\
&\quad \text{such that for all } 0 < h \le m,\ (\sigma, r_{h-1}, r_h) \models P_1; P_2, \text{ or}\\
&\quad \text{there are infinitely many } k = r_0 \le t_1 \le r_1 \le t_2 \le r_2 \le \cdots, \lim_{i\to\infty} r_i = \omega\\
&\quad \text{such that for all } h > 0,\ (\sigma, r_{h-1}, r_h) \models P_1; P_2\\
&\Longleftrightarrow (\sigma, k, j) \models (P_1; P_2)^+\\
&\Longleftrightarrow (\sigma, k, j) \models ((P_1; P_2)^+)\ prj\ \varepsilon.
\end{aligned}
$$

(IUM) We prove the case of $m = 4$.

$$
\begin{aligned}
&((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q \vee (P_1, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge more, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge more, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge more, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge more, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge more, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4 \wedge more, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4 \wedge \varepsilon, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\equiv\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge more, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge more, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4 \wedge more, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee P_{1e} \wedge P_{2e} \wedge P_{3e} \wedge P_{4e} \wedge ((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q \qquad (*)
\end{aligned}
$$

where P_{1e}, P_{2e}, P_{3e} and P_{4e} are state formulas in the terminal products in normal forms of P_1, P_2, P_3 and P_4. Now, for ease of presentation, we define the following:

$$
\begin{aligned}
A \stackrel{\text{def}}{=}\ &((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
B \stackrel{\text{def}}{=}\ &(P_1, P_2, P_3, P_4)\ prj\ Q\\
&\vee(P_1 \wedge more, P_2, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge more, P_3, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge more, P_4, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
&\vee(P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4 \wedge more, (P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
C \stackrel{\text{def}}{=}\ &P_{1e} \wedge P_{2e} \wedge P_{3e} \wedge P_{4e} \wedge ((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q.
\end{aligned}
$$

According to the proof $(*)$ given above, we have $A \equiv B \vee C$. Now we will prove $C \supset B$. Since if $C \supset B$, then $B \vee C \equiv (B \vee C) \wedge (C \supset B) \equiv B \wedge (C \supset B) \vee C \wedge (C \supset B) \equiv B \wedge (C \supset B) \vee C \wedge B \equiv B \wedge true \vee C \wedge B \equiv B$, so $A \equiv B$. Let σ be an interval and $0 \le k \preceq j \le |\sigma|$.

$$
\begin{aligned}
&(\sigma, k, j) \models P_{1e} \wedge P_{2e} \wedge P_{3e} \wedge P_{4e} \wedge ((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q\\
\Longrightarrow\ &(\sigma, k, j) \models ((P_1, P_2, P_3, P_4)^{\oplus})\ prj\ Q.
\end{aligned}
$$

From the semantics of $prj\ \oplus$, we have the following two cases:
Case 1: There are finitely many $k = r_0 \le r_1 \le \cdots \le r_{h-1} \preceq r_h \le j$ such that $(\sigma, r_0, r_1) \models P_1,\ (\sigma, r_1, r_2) \models P_2, \ldots$.

- If $r_h > k$, then we can find the first integer i such that $r_i > r_{i-1}$. In other words, $k = r_0 = r_1 = \cdots = r_{i-1} < r_i \leq r_{i+1} \leq \cdots \leq r_h$. Suppose that $(\sigma, r_{i-1}, r_i) \models P_j$. Then, if r_i comes from the last iteration:

$$\begin{array}{l} \quad (\sigma, k, j) \models (P_1 \wedge \varepsilon, \ldots, P_{j-1} \wedge \varepsilon, P_j \wedge more, \ldots, P_4)\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models (P_1, P_2, P_3, P_4)\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models B \end{array}$$

  and if r_i does not come from the last iteration:

$$\begin{array}{l} \quad (\sigma, k, j) \models (P_1 \wedge \varepsilon, \ldots, P_{j-1} \wedge \varepsilon, P_j \wedge more, \ldots, P_4, (P_1, \ldots, P_4)^{\oplus})\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models B. \end{array}$$

- If $r_h = k$, then we have:

$$\begin{array}{l} \quad (\sigma, k, j) \models (P_1 \wedge \varepsilon, P_2 \wedge \varepsilon, P_3 \wedge \varepsilon, P_4 \wedge \varepsilon)\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models (P_1, P_2, P_3, P_4)\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models B. \end{array}$$

Case 2: There are infinitely many $k = r_0 \leq r_1 \leq r_2 \leq \cdots$, $\lim_{i\to\infty} r_i = \omega$. Then we can find the minimal integer i such that $r_i > r_{i-1}$. Suppose that $(\sigma, r_{i-1}, r_i) \models P_j$. Then we have:

$$\begin{array}{l} \quad (\sigma, k, j) \models (P_1 \wedge \varepsilon, \ldots, P_{j-1} \wedge \varepsilon, P_j \wedge more, \ldots, P_4, (P_1, \ldots, P_4)^{\oplus})\ prj\ Q \\ \Longrightarrow (\sigma, k, j) \models B. \end{array}$$

Since in both case we can derive $(\sigma, k, j) \models B$, it follows that $C \supset B$. In other words, the set of models satisfying C is a subset of models of B. So $B \vee C \equiv B$ and the axiom IUM is valid.

To complete the proof one needs to demonstrate that each inference rule preserves validity. We do this for ALW, NXT1 and NXT2.

(ALW) $\models P \Longrightarrow \models \Box P$.
Let σ be an interval. Since $\models P$, for all r with $0 \leq r \preceq |\sigma|$, we have $(\sigma, r, |\sigma|) \models P$. Thus, for any interval σ, $\sigma \models \Box P$ and so $\models \Box P$.

(NXT1) $\models P_1 \wedge \cdots \wedge P_m \rightarrow Q \Longrightarrow \models \bigcirc P_1 \wedge \cdots \wedge \bigcirc P_m \rightarrow \bigcirc Q$.
Let σ be an interval. Since $\sigma \models P_1 \wedge \cdots \wedge P_m \rightarrow Q$, we have

$$\begin{array}{rl} \sigma \models \bigcirc P_1 \wedge \cdots \wedge \bigcirc P_m & \Longleftrightarrow (\sigma, 0, |\sigma|) \models \bigcirc P_1 \wedge \cdots \wedge \bigcirc P_m \\ & \Longleftrightarrow (\sigma, 1, |\sigma|) \models P_1 \wedge \cdots \wedge P_m \\ & \Longrightarrow (\sigma, 1, |\sigma|) \models Q \\ & \Longleftrightarrow (\sigma, 0, |\sigma|) \models \bigcirc Q \\ & \Longleftrightarrow \sigma \models \bigcirc Q. \end{array}$$

(NXT2) $\models P \rightarrow (Q \vee \bigcirc P) \Longrightarrow \models P \rightarrow (\Diamond Q \vee \Box \bigcirc P)$.
Let σ be an interval. We first show that if k is such that $0 \leq k \preceq |\sigma|$, then we have:

$$\begin{array}{l} \quad (\sigma, k, |\sigma|) \models P \wedge \neg\Diamond Q \\ \Longrightarrow (\sigma, k, |\sigma|) \models (Q \vee \bigcirc P) \wedge \neg\Diamond Q \\ \Longleftrightarrow (\sigma, k, |\sigma|) \models \bigcirc P \wedge \neg\Diamond Q \\ \Longleftrightarrow (\sigma, k, |\sigma|) \models \bigcirc P \wedge (\neg Q \wedge \bigodot \neg\Diamond Q) \\ \Longleftrightarrow (\sigma, k, |\sigma|) \models \neg Q \wedge \bigcirc P \wedge \bigcirc\neg\Diamond Q \\ \Longrightarrow (\sigma, k, |\sigma|) \models \bigcirc P \wedge \bigcirc\neg\Diamond Q \\ \Longleftrightarrow k < |\sigma| \text{ and } (\sigma, k, |\sigma|) \models \bigcirc P \text{ and } (\sigma, k+1, |\sigma|) \models P \wedge \neg\Diamond Q. \end{array}$$

Suppose now that $(\sigma, 0, |\sigma|) \models P \wedge \neg\Diamond Q$. From what we have just shown it follows that $(\sigma, k, |\sigma|) \models \bigcirc P$, for all $0 \leq k \preceq |\sigma|$. This means that $(\sigma, 0, |\sigma|) \models \Box \bigcirc P$. Thus we obtained $\models P \wedge \neg\Diamond Q \rightarrow \Box \bigcirc P$ leading to $\models P \rightarrow (\Diamond Q \vee \Box \bigcirc P)$. □

To demonstrate that the axiom system is complete, we will show that if $\not\vdash \neg P$, then P is satisfiable, namely, $\not\models \neg P$.

The set of PPTL formulas can be partitioned into terminable and non-terminable formulas. We will prove that any terminable formula is satisfiable (Lemma 7), and that for any non-terminable formula P, if $\not\vdash \neg P$, then P is satisfiable (Lemma 12). The proof of Lemma 12 is based on the normal form of PPTL formulas. We will demonstrate that each PPTL formula can be transformed into normal form using axioms and inference rules (Theorem 2). To prove this, a number of lemmas are required (Lemmas 1–6).

Lemma 1. *For any PPTL formula P, if $P \cong Q$ where Q is a normal form formula, then there exists a PPTL formula R in complete normal form satisfying $P \cong R$.*

Proof. Suppose that $P \cong p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n}(p_i \wedge \bigcirc P_i)$, where p_e and the p_i's are state formulas. We need to construct a PPTL formula R in complete normal form such that $P \cong R$.

Following the standard definitions of min-term, max-term and basic sum in classical propositional logic, we treat each state formula p_i as an atomic proposition and construct 2^n min-terms, $m_0, \ldots, m_{2^n-1}$. Moreover, we can treat P_i as an atomic proposition and construct 2^n max-terms, $M_0, \ldots, M_{2^n-1}$. From these max-terms, we can obtain 2^n basic sums, namely, $M'_0, \ldots, M'_{2^n-1}$ (here M'_j denotes M_{2^n-1-j} after deleting negative disjunctions, and M'_0 denotes *false*). It is easy to prove the following theorem in our axiom system:

$$P \cong p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n}(p_i \wedge \bigcirc P_i) \cong p_e \wedge \varepsilon \vee \bigvee_{i=0}^{2^n-1}(m_i \wedge \bigcirc M'_i).$$

We only prove the case of $n = 2$.

$$\begin{aligned}
& p_e \wedge \varepsilon \vee \bigvee_{i=1}^{2}(p_i \wedge \bigcirc P_i) \\
\cong\ & p_e \wedge \varepsilon \vee p_1 \wedge (p_2 \vee \neg p_2) \wedge \bigcirc P_1 \vee p_2 \wedge (p_1 \vee \neg p_1) \wedge \bigcirc P_2 && \text{TAU} \\
\cong\ & p_e \wedge \varepsilon \vee p_1 \wedge p_2 \wedge \bigcirc(P_1 \vee P_2) \vee p_1 \wedge \neg p_2 \wedge \bigcirc P_1 \vee \neg p_1 \wedge p_2 \wedge \bigcirc P_2 && \text{TAU} \\
\cong\ & p_e \wedge \varepsilon \vee p_1 \wedge p_2 \wedge \bigcirc(P_1 \vee P_2) \vee p_1 \wedge \neg p_2 \wedge \bigcirc P_1 \vee \neg p_1 \wedge p_2 \wedge \bigcirc P_2 \\
& \vee \neg p_1 \wedge \neg p_2 \wedge \bigcirc \mathit{false}
\end{aligned}$$

Hence the lemma holds. □

Lemma 2. *If $\bigvee_{i=1}^{n} p_i \cong true$ and $\bigvee_{i\neq j} p_i \wedge p_j \cong false$, for state formulas $p_1, \ldots, p_n$, then $\neg p_k \cong \bigvee_{i\neq k} p_i$.*

Proof. Straightforward. □

Lemma 3. *Let $p_1, \ldots, p_n$ be state formulas, and Q_i be a general PPTL formula. If $\bigvee_{i=1}^{n} p_i \cong true$ and $\bigvee_{i\neq j} p_i \wedge p_j \cong false$, then $\neg(\bigvee_{i=1}^{n} p_i \wedge Q_i) \cong \bigvee_{i=1}^{n}(p_i \wedge \neg Q_i)$.*

Proof. We have the following:

$$\begin{aligned}
&(1) && \bigvee_{i=1}^{n} p_i \cong true \\
&(2) && \vdash \bigvee_{i=1}^{n} p_i && \text{MP (1)} \\
&(3) && \vdash \bigvee_{i=1}^{n}(p_i \wedge Q_i \vee p_i \wedge \neg Q_i) && \text{TAU (2)} \\
&(4) && \vdash \left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right) \vee \left(\bigvee_{i=1}^{n} p_i \wedge \neg Q_i\right) && \text{TAU (3)} \\
&(5) && \vdash \neg\left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right) \rightarrow \bigvee_{i=1}^{n}(p_i \wedge \neg Q_i) && \text{TAU (4)} \\
&(6) && \left(\bigvee_{i=1}^{n} p_i \wedge \neg Q_i\right) \wedge \left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right) \\
& && \cong \bigvee_{i=1}^{n}(p_i \wedge Q_i \wedge \neg Q_i) \vee \bigvee_{i\neq j}(p_i \wedge p_j \wedge \neg Q_i \wedge Q_j) && \text{TAU} \\
&(7) && \cong \mathit{false} && (1) \\
&(8) && \vdash \neg\left(\left(\bigvee_{i=1}^{n} p_i \wedge \neg Q_i\right) \wedge \left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right)\right) && \text{TAU (7)} \\
&(9) && \vdash \bigvee_{i=1}^{n}(p_i \wedge \neg Q_i) \rightarrow \neg\left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right) && \text{TAU (8)} \\
&(10) && \neg\left(\bigvee_{i=1}^{n} p_i \wedge Q_i\right) \cong \bigvee_{i=1}^{n}(p_i \wedge \neg Q_i) && (5)\ (9).
\end{aligned}$$

Hence the lemma holds. □

Lemma 4. *If $P_t \cong P'_t$ (for $t = 1, \ldots, m$) and $Q \cong Q'$, where the P'_t's and Q' are formulas in normal form, then there exists a formula P in normal form such that $(P_1, \ldots, P_m)$ prj $Q \cong P$.*

Proof. For $t = 1, \ldots, m$, let $P'_t = p_{te} \wedge \varepsilon \vee \bigvee_{i=1}^{n_t} p_{ti} \wedge \bigcirc P_{ti}$ and $Q' = q_e \wedge \varepsilon \vee \bigvee_{k=1}^{n} q_k \wedge \bigcirc Q_k$. The proof proceeds by induction on the number of P_i's.

Base case: If $m = 1$, then we have the following:

$$
\begin{aligned}
& P_1 \ prj \ Q \\
\cong\ & (p_{1e} \wedge \varepsilon) \ prj \ Q \vee \left(\bigvee_{i=1}^{n_1} p_{1i} \wedge \bigcirc P_{1i} \right) \ prj \ Q && \text{Premise PDF} \\
\cong\ & p_{1e} \wedge Q \vee \bigvee_{i=1}^{n_1} p_{1i} \wedge (\bigcirc P_{1i} \ prj \ Q) && \text{PSF PDF PEF} \\
\cong\ & p_{1e} \wedge q_e \wedge \varepsilon \vee \bigvee_{k=1}^{n} p_{1e} \wedge q_k \wedge \bigcirc Q_k && \text{Premise TAU} \\
& \vee \bigvee_{i=1}^{n_1} p_{1i} \wedge q_e \wedge \bigcirc P_{1i} && \text{PSB PEB} \\
& \vee \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_1} p_{1i} \wedge q_k \wedge \bigcirc (P_{1i}; Q_k) && \text{PDB PNX.}
\end{aligned}
$$

Inductive case: Suppose that $(P_{(2..m)}) \ prj \ Q \cong r_e \wedge \varepsilon \vee \bigvee_{j=1}^{s} r_j \wedge \bigcirc R_j$ $(*)$, and the formulas P_1 and Q are in normal form. Then we have the following:

$$
\begin{aligned}
& (P_1, \ldots, P_m) \ prj \ Q \\
\cong\ & (p_{1e} \wedge \varepsilon, P_{(2..m)}) \ prj \ Q && \text{Premise} \\
& \vee \bigvee_{i=1}^{n_1} (p_{1i} \wedge \bigcirc P_{1i}, P_{(2..m)}) \ prj \ Q && \text{PDF} \\
\cong\ & p_{1e} \wedge (P_{(2..m)}) \ prj \ Q && \text{PSF PSM} \\
& \vee \bigvee_{i=1}^{n_1} p_{1i} \wedge q_e \wedge (\bigcirc P_{1i}, P_{(2..m)}) \ prj \ \varepsilon && \text{Premise} \\
& \vee \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_1} p_{1i} \wedge q_k \wedge ((\bigcirc P_{1i}, P_{(2..m)}) \ prj \ \bigcirc Q_k) && \text{PSF PSB PDB} \\
\cong\ & p_{1e} \wedge r_e \wedge \varepsilon \vee \bigvee_{j=1}^{s} p_{1e} \wedge r_j \wedge \bigcirc R_j && (*) \\
& \vee \bigvee_{i=1}^{n_1} p_{1i} \wedge q_e \wedge \bigcirc (P_{1i}; P_2; \ldots; P_m) && \text{Def of ; PNX PEC} \\
& \vee \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_1} p_{1i} \wedge q_k \wedge \bigcirc (P_{1i}; (P_{(2..m)}) \ prj \ Q_k) && \text{PNX.}
\end{aligned}
$$

Therefore, there exists a normal form formula that can be derived in the axiom system equivalent to $(P_1, \ldots, P_m) \ prj \ Q$. □

Lemma 5. *If $P_t \cong P'_t$ (for $t = 1, \ldots, m$) and $Q \cong Q'$ and $(P_1, \ldots, P_m) \ prj \ Q \cong R$ where the P'_t's, Q' and R are formulas in normal form, then there exists a formula P in normal form such that $((P_1, \ldots, P_m)^{\oplus}) \ prj \ Q \cong P$.*

Proof. For $t = 1, \ldots, m$, let $P'_t = p_{te} \wedge \varepsilon \vee \bigvee_{i=1}^{n_t} p_{ti} \wedge \bigcirc P_{ti}$ and $Q' = q_e \wedge \varepsilon \vee \bigvee_{k=1}^{n} q_k \wedge \bigcirc Q_k$ and $R = r_e \wedge \varepsilon \vee \bigvee_{j=1}^{s} r_j \wedge \bigcirc R_j$. By IUM and $P_0 \stackrel{\text{def}}{=} \varepsilon$, we have $p_{0e} \cong \mathit{true}$, and

$$
\begin{aligned}
& ((P_1, \ldots, P_m)^{\oplus}) \ prj \ Q \\
\cong\ & (P_1, \ldots, P_m) \ prj \ Q \\
& \vee \bigvee_{t=1}^{m-1} \left(\bigwedge_{h=0}^{t-1} P_h \wedge \varepsilon, P_t \wedge \mathit{more}, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus} \right) \ prj \ Q && (*) \\
& \vee \left(\bigwedge_{h=0}^{m-1} P_h \wedge \varepsilon, P_m \wedge \mathit{more}, (P_1, \ldots, P_m)^{\oplus} \right) \ prj \ Q && (**)
\end{aligned}
$$

The disjunction $(*)$ can be shown equivalent to a formula in normal form:

$$\begin{aligned}
&\left(\bigwedge_{h=0}^{t-1} P_h \wedge \varepsilon, P_t \wedge more, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}\right) \ prj\ Q \\
\cong &\left(\bigwedge_{h=0}^{t-1} p_{he} \wedge \varepsilon, \bigvee_{i=1}^{n_t} p_{ti} \wedge \bigcirc P_{ti}, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}\right) \ prj\ Q \\
& \hfill \text{PREMISE} \\
\cong &\bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge (\bigcirc P_{ti}, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ Q \\
& \hfill \text{PDF PSM PSF} \\
\cong &\bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_e \wedge (\bigcirc P_{ti}, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ \varepsilon \vee \\
&\bigvee_{k=1}^{n} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_k \wedge (\bigcirc P_{ti}, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ \bigcirc Q_k \\
& \hfill \text{PREMISE PDB PSB} \\
\cong &\bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_e \wedge \bigcirc (P_{ti}; P_{t+1}; \ldots; P_m; (P_1; \ldots; P_m)^{+}) \vee \\
&\bigvee_{k=1}^{n} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_k \wedge \bigcirc (P_{ti}; (P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ Q_k) \\
& \hfill \text{IEC PEC PNX PEC DEF OF ;}
\end{aligned}$$

Then, by TAU and the conclusion proved above, we have the following:

$$\begin{aligned}
&\bigvee_{t=1}^{m-1} \left(\bigwedge_{h=0}^{t-1} P_h \wedge \varepsilon, P_t \wedge more, P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}\right) \ prj\ Q \\
\cong &\bigvee_{t=1}^{m-1} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_e \wedge \bigcirc (P_{ti}; P_{t+1}; \ldots; P_m; (P_1; \ldots; P_m)^{+}) \vee \\
&\bigvee_{t=1}^{m-1} \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_k \wedge \\
&\qquad \bigcirc\ (P_{ti}; (P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ Q_k).
\end{aligned}$$

Similar conclusion holds for the disjunction $(**)$:

$$\begin{aligned}
&\left(\bigwedge_{h=0}^{m-1} P_h \wedge \varepsilon, P_m \wedge more, (P_1, \ldots, P_m)^{\oplus}\right) \ prj\ Q \\
\cong &\bigvee_{i=1}^{n_m} \bigwedge_{h=0}^{m-1} p_{he} \wedge p_{mi} \wedge q_e \wedge \bigcirc (P_{mi}; (P_1; \ldots; P_m)^{+}) \\
&\vee \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_m} \bigwedge_{h=0}^{m-1} p_{he} \wedge p_{mi} \wedge q_k \wedge \bigcirc (P_{mi}; ((P_1, \ldots, P_m)^{\oplus}) \ prj\ Q_k).
\end{aligned}$$

Therefore, by TAU, the premise and what we have demonstrated, a projection-plus formula can be transformed into a formula in normal form:

$$\begin{aligned}
&((P_1, \ldots, P_m)^{\oplus}) \ prj\ Q \\
\cong &r_e \wedge \varepsilon \vee \bigvee_{j=1}^{s} r_j \wedge \bigcirc R_j \\
&\vee \bigvee_{t=1}^{m-1} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_e \wedge \bigcirc (P_{ti}; P_{t+1}; \ldots; P_m; (P_1; \ldots; P_m)^{+}) \\
&\vee \bigvee_{t=1}^{m-1} \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_t} \bigwedge_{h=0}^{t-1} p_{he} \wedge p_{ti} \wedge q_k \wedge \\
&\qquad \bigcirc\ (P_{ti}; (P_{t+1}, \ldots, P_m, (P_1, \ldots, P_m)^{\oplus}) \ prj\ Q_k)
\end{aligned}$$

$$\vee \bigvee_{i=1}^{n_m} \bigwedge_{h=0}^{m-1} p_{he} \wedge p_{mi} \wedge q_e \wedge \bigcirc(P_{mi}; (P_1; \ldots; P_m)^+)$$
$$\vee \bigvee_{k=1}^{n} \bigvee_{i=1}^{n_m} \bigwedge_{h=0}^{m-1} p_{he} \wedge p_{mi} \wedge q_k \wedge \bigcirc(P_{mi}; ((P_1, \ldots, P_m)^{\oplus}) \; prj \; Q_k).$$

Hence the lemma holds. □

Lemma 6. *If $P_t \cong P'_t$ (for $t = 1, \ldots, m$), $Q \cong Q'$ and $(P_1, \ldots, P_m)$ prj $Q \cong R$, where the P'_t's, Q' and R are formulas in normal form, then there exists a formula P in normal form such that $(P_1, \ldots, (P_i, \ldots, P_j)^{\oplus}, \ldots, P_m)$ prj $Q \cong P$.*

Proof. Similar to that of Lemma 5. □

We now turn to prove the theorem of normal form.

Theorem 2. *For any PPTL formula R, there exists a formula P' in normal form such that $R \cong P'$.*

Proof. Let R be a PPTL formula. The proof proceeds by induction on the structure of PPTL.
Base case: If R is an atomic proposition p, then:

$$\begin{aligned} p &\cong p \wedge true && \text{TAU} \\ &\cong p \wedge (\varepsilon \vee more) && \text{TAU DEF OF } \{\varepsilon, more\} \\ &\cong p \wedge \varepsilon \vee p \wedge \bigcirc true && \text{TAU.} \end{aligned}$$

Moreover, if R is $\bigcirc P$, then $R \cong false \wedge \varepsilon \vee true \wedge \bigcirc P$.
Inductive case: Suppose that $P_t \cong p_{te} \wedge \varepsilon \vee \bigvee_{i=1}^{n_t} p_{ti} \wedge \bigcirc P_{ti}$ $(t \in \mathbb{N}_0)$ and $Q \cong q_e \wedge \varepsilon \vee \bigvee_{k=1}^{n} q_k \wedge \bigcirc Q_k$. We then have the following:

- R is $\neg P_1$. With the hypothesis and Lemma 1, we can transform P_1 into its complete normal form as follows:

$$P_1 \cong p_{1e} \wedge \varepsilon \vee \bigvee_{j=1}^{r} p_{1j} \wedge \bigcirc P_{1j} \qquad (*)$$

where $\bigvee_{j=1}^{r} p_{1j} \cong true$ and $\bigvee_{i \neq j} p_{1i} \wedge p_{1j} \cong false$. Therefore,

$$\begin{aligned} \neg P_1 &\cong \neg(p_{1e} \wedge \varepsilon \vee \bigvee_{j=1}^{r} p_{1j} \wedge \bigcirc P_{1j}) && \text{FORMULA } (*) \\ &\cong (\neg p_{1e} \vee \neg\varepsilon) \wedge \left(\bigvee_{j=1}^{r} p_{1j} \wedge \neg \bigcirc P_{1j} \right) && \text{TAU Lemma 3} \\ &\cong (\neg p_{1e} \vee \neg\varepsilon) \wedge \left(\bigvee_{j=1}^{r} p_{1j} \wedge (\varepsilon \vee \bigcirc \neg P_{1j}) \right) && \text{T3} \\ &\cong (\neg p_{1e} \vee \neg\varepsilon) \wedge \left(\varepsilon \vee \bigvee_{j=1}^{r} p_{1j} \wedge \bigcirc \neg P_{1j} \right) && \bigvee_{j=1}^{r} p_{1j} \cong true \\ &\cong \neg p_{1e} \wedge \varepsilon \vee \bigvee_{j=1}^{r} p_{1j} \wedge \bigcirc \neg P_{1j} && \text{TAU.} \end{aligned}$$

- R is $P_1 \vee P_2$. With the hypothesis and TAU, we have,

$$P_1 \vee P_2 \cong (p_{1e} \vee p_{2e}) \wedge \varepsilon \vee \bigvee_{i=1}^{n_1} p_{1i} \wedge \bigcirc P_{1i} \vee \bigvee_{i=1}^{n_2} p_{2i} \wedge \bigcirc P_{2i}$$

- R is $(P_1, \ldots, P_m)$ *prj* Q. By Lemma 4, the conclusion holds.
- R is $(P_1, \ldots, (P_i, \ldots, P_l)^{\oplus}, \ldots, P_m)$ *prj* Q. By Lemma 6, the conclusion holds.

Hence the theorem holds. □

Theorem 2 tells us that any PPTL formula can be transformed into normal form by means of axioms and inference rules. This conclusion plays an important role in the proof of completeness since we only need to consider formulas in normal form rather than arbitrary formulas.

Lemma 7. *For any terminable PPTL formula P, P is satisfiable.*

Proof. Suppose P is terminable, but unsatisfiable, then $P \equiv false$. It follows that $P \wedge \Diamond\varepsilon \equiv false \wedge \Diamond\varepsilon$. Hence, $P \wedge \Diamond\varepsilon \equiv false$ and a contradiction is encountered. □

Lemma 8. *For any PPTL formulas P and P', if $P \equiv P'$ where P is non-terminable and P' in normal form, then P' is of the form $\bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ with each P_i being non-terminable.*

Proof. We first observe that P' must be a disjunction of future products $\bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$. That is, in P' there is disjunctive term $p_e \wedge \varepsilon$ where $p_e \not\equiv \mathit{false}$. To the contrary, suppose that P' is in the form of $p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ where $p_e \not\equiv \mathit{false}$. Then we have:

$$\begin{aligned} P \wedge \Diamond\varepsilon &\equiv \left(p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i\right) \wedge \Diamond\varepsilon \\ &\equiv p_e \wedge \varepsilon \wedge \Diamond\varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i \wedge \Diamond\varepsilon \\ &\equiv p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i \wedge \Diamond\varepsilon. \end{aligned}$$

Since $p_e \not\equiv \mathit{false}$ and p_e is a state formula, we have $p_e \wedge \varepsilon \not\equiv \mathit{false}$ and $P \wedge \Diamond\varepsilon \not\equiv \mathit{false}$. Therefore, P is terminable and we produced a contradiction.

We then observe that each P_i is non-terminable. From what we have just proved, we have $P \equiv \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$. Suppose that some P_i, say P_1, is terminable. We then obtain the following:

$$\begin{aligned} P \wedge \Diamond\varepsilon &\equiv \left(\bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i\right) \wedge \Diamond\varepsilon \\ &\equiv p_1 \wedge \bigcirc P_1 \wedge \Diamond\varepsilon \vee \bigvee_{i=2}^{n} p_i \wedge \bigcirc P_i \wedge \Diamond\varepsilon \\ &\equiv p_1 \wedge \bigcirc(P_1 \wedge \Diamond\varepsilon) \vee \bigvee_{i=2}^{n} p_i \wedge \bigcirc P_i \wedge \Diamond\varepsilon. \end{aligned}$$

Since $P_1 \wedge \Diamond\varepsilon \not\equiv \mathit{false}$, $p_1 \wedge \bigcirc(P_1 \wedge \Diamond\varepsilon) \not\equiv \mathit{false}$, from which it follows that $P \wedge \Diamond\varepsilon \not\equiv \mathit{false}$. Hence, P is terminable yielding a contradiction. □

From Lemma 8, we can derive the following:

Corollary 1. *For any PPTL formula P, if $P \equiv P'$ where P' is in normal form of $p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$, then:*

1. *If $p_e \not\equiv \mathit{false}$, then P is terminable.*
2. *If there exists some P_i being terminable, then P is terminable.*

The above corollary is used in the proof of Lemma 9. Notice that Lemma 9 is similar to Lemma 8 but it is formalized in the axiom system.

Lemma 9. *For any PPTL formulas P and P', if $P \cong P'$ where P is non-terminable and P' in normal form, then P' must be of the form $\bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ with each P_i being non-terminable.*

Proof. We first show that P' must be of the form $\bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ where there is no disjunctive term being a terminal product like $p_e \wedge \varepsilon$ where $\nvdash p_e \rightarrow \mathit{false}$.
To the contrary, suppose that P' is in normal form of $p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ and $\nvdash p_e \rightarrow \mathit{false}$. Then by Theorem 1, we have $P \equiv p_e \wedge \varepsilon \vee \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$. Since p_e is a state formula, according to the completeness of classical propositional logic, it follows that $\not\models p_e \rightarrow \mathit{false}$, namely, $p_e \not\equiv \mathit{false}$. By Corollary 1, we have P is terminable, yielding a contradiction.

We then observe that each P_i is non-terminable. From what we have just shown it follows that $P \cong \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$. From Theorem 1, it follows that $P \equiv \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$. If there exists some terminable P_i, using Corollary 1 it is easy to derive that P is terminable, producing a contradiction. □

Lemma 10. *If w is a state formula and $\nvdash \neg w$, then there exists a model $\sigma = \langle s \rangle$, such that $\sigma \models w$.*

Proof. By completeness of classical propositional logic, we have $\models \neg w \Rightarrow \vdash \neg w$. Hence, $\nvdash \neg w$ implies $\not\models \neg w$. Thus, by $\nvdash \neg w$, we have that $\neg w$ is not valid. In other words, w is satisfiable. Furthermore, since w is a state formula, w can be satisfied by a singleton interval. □

Lemma 11. $\nvdash \bigcirc P \rightarrow \mathit{false} \implies \nvdash P \rightarrow \mathit{false}$.

Proof. If $\vdash P \rightarrow \mathit{false}$, then we have:

(1)	$\vdash P \rightarrow \mathit{false}$	HYPOTHESIS
(2)	$\vdash \bigcirc P \rightarrow \bigcirc \mathit{false}$	NXT1 (1)
(3)	$\vdash \bigcirc P \rightarrow \mathit{false}$	T2

contradicting the assumption from the statement of the lemma. □

k	i	Formula $P^k_{m_k}$	set S_k
-1	$--$	$P \cong \bigvee_{i=1}^{n_0} p_i^0 \wedge \bigcirc P_i^0$	$S_0 = \{P_i^0 \mid 1 \le i \le n_0\}$
0	m_0	$P^0_{m_0} \cong \bigvee_{i=1}^{n_1} p_i^1 \wedge \bigcirc P_i^1$	$S_1 = \{P_i^1 \mid 1 \le i \le n_1\}$
1	m_1	$P^1_{m_1} \cong \bigvee_{i=1}^{n_2} p_i^2 \wedge \bigcirc P_i^2$	$S_2 = \{P_i^2 \mid 1 \le i \le n_2\}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	m_k	$P^k_{m_k} \cong \bigvee_{i=1}^{n_{k+1}} p_i^{k+1} \wedge \bigcirc P_i^{k+1}$	$S_{k+1} = \{P_i^{k+1} \mid 1 \le i \le n_{k+1}\}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

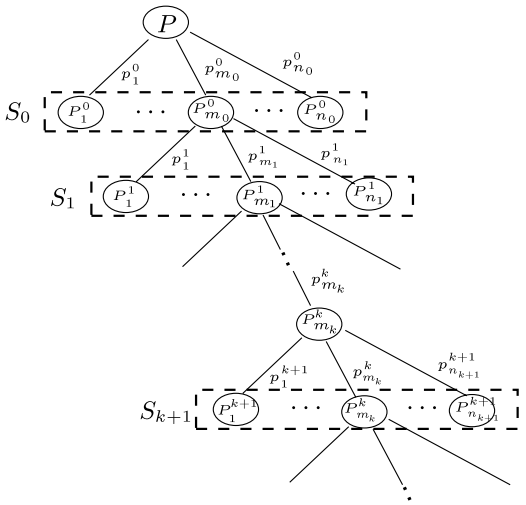

Fig. 12. Decomposition of non-terminable formula P.

Lemma 12. *If P is a non-terminable PPTL formula and $\nvdash P \rightarrow false$, then P is satisfiable.*

Proof. To prove the result, we need to generate a state sequence and show that the interval determined by the state sequence satisfies P.

(1) Generating a state sequence.

Since P is non-terminable, by Theorem 2 and Lemma 9, we have $P \cong \bigvee_{i=1}^{n} p_i \wedge \bigcirc P_i$ with each P_i being non-terminable. We can decompose P repeatedly by means of axioms and inference rules of Π_{pptl} to obtain some of its particular sub-formulas in normal form. For convenience, let P_i^{-1} denote P, and k refer to the k^{th} decomposition of P. In general, we have the following:

$$P_i^k \cong \bigvee_{i=1}^{n_{k+1}} p_i^{k+1} \wedge \bigcirc P_i^{k+1} \quad (k = -1, 0, 1, 2, \ldots). \tag{1}$$

In the following table, S_k denotes the set of formulas P_i^k, obtained by the k^{th} decomposition of P ($k \ge 0$). In particular, we choose only one formula in S_k, $P^k_{m_k}$, to generate a new set S_{k+1}, where $1 \le m_k \le n_k$. This idea will be used in the generation of the state sequence. The idea behind the decomposition of P is illustrated in Fig. 12.

The following deduction further illustrates how to generate the state sequence. It is a loop where each iteration with a value of k can generate a new state s_{k+1} ($k = -1, 0, 1, 2, \ldots$). As is P non-terminable, the generating process is also non-terminable, and we get an infinite sequence $\sigma = \langle s_0, s_1, \ldots \rangle$.

(1) $\not\models P_i^k \to \mathit{false}$ PREMISE
(2) $\not\models \bigvee_{i=1}^{n_{k+1}} p_i^{k+1} \wedge \bigcirc P_i^{k+1} \to \mathit{false}$ Lemma 9
(3) $\exists\, m_{k+1}\ 1 \le m_{k+1} \le n_{k+1}$ and $\not\models p_{m_{k+1}}^{k+1} \wedge \bigcirc P_{m_{k+1}}^{k+1} \to \mathit{false}$
(4) $\not\models p_{m_{k+1}}^{k+1} \to \mathit{false}$ and $\not\models \bigcirc P_{m_{k+1}}^{0} \to \mathit{false}$
(5) $\not\models p_{m_{k+1}}^{k+1} \to \mathit{false}$ and $\not\models P_{m_{k+1}}^{k+1} \to \mathit{false}$ Lemma 11
(6) $\exists\, s_{k+1}\ \langle s_{k+1}\rangle \models p_{m_{k+1}}^{k+1}$ and $\not\models P_{m_{k+1}}^{k+1} \to \mathit{false}$ Lemma 10.

(2) We now show that $P \cong (\bigwedge_{i=0}^{k} \bigcirc^i p_{m_i}^i) \wedge \bigcirc^{k+1} P_{m_k}^k \vee P$ by induction on k.
Base case: If $k = 0$ then we have the following:

$$\begin{aligned} P &\cong \bigvee_{i=1}^{n_0} p_i^0 \wedge \bigcirc P_i^0 && \text{FORMULA (1)} \\ &\cong p_{m_0}^0 \wedge \bigcirc P_{m_0}^0 \vee P && \text{TAU.} \end{aligned}$$

Inductive case: Suppose the conclusion holds for $k \ge 0$. Then, for $k+1$, we have:

$$\begin{aligned} P &\cong \left(\bigwedge_{i=0}^{k} \bigcirc^i p_{m_i}^i\right) \wedge \bigcirc^{k+1} P_{m_k}^k \vee P && \text{HYPOTHESIS} \\ P_{m_k}^k &\cong \bigvee_{i=1}^{n_{k+1}} p_i^{k+1} \wedge \bigcirc P_i^{k+1} && \text{FORMULA (1)} \\ &\cong p_{m_{k+1}}^{k+1} \wedge \bigcirc P_{m_{k+1}}^{k+1} \vee P_{m_k}^k && \text{TAU} \\ P &\cong \left(\bigwedge_{i=0}^{k+1} \bigcirc^i p_{m_i}^i\right) \wedge \bigcirc^{k+2} P_{m_{k+1}}^{k+1} \vee P && \text{T1.} \end{aligned}$$

Thus, by Theorem 1, we have $P \equiv (\bigwedge_{i=0}^{k} \bigcirc^i p_{m_i}^i) \wedge \bigcirc^{k+1} P_{m_k}^k \vee P$. So any interval satisfying $\bigwedge_{i=0}^{k} \bigcirc^i p_{m_i}^i$ with some k is a prefix of the model of P.
(3) Each finite prefix of the infinite state sequence is a prefix of the final model. The proof proceeds by induction on the length of the prefix of the state sequence.
Base case: If $k = 0$, then $\sigma^0 = \langle s_0\rangle \models p_{m_0}^0 = \bigwedge_{i=0}^{0} \bigcirc^i p_{m_i}^i$. Then each prefix of interval $\langle s_0\rangle$ is a prefix of the final model.
Induction: Suppose that σ^k is a prefix of the final model. Then, for $k+1$, we have that $\langle s_{k+1}\rangle \models p_{m_{k+1}}^{k+1}$ and $\sigma^k \models \bigwedge_{i=0}^{k} \bigcirc^i p_{m_i}^i$ imply $\sigma^{k+1} \models \bigwedge_{i=0}^{k+1} \bigcirc^i p_{m_i}^i$. Hence σ^{k+1} is also a prefix of the final model.
(4) The infinite state sequence $\sigma = \sigma^\omega = \langle s_0, s_1, \ldots\rangle$ is a model of P. In the proof below, we make use of the fix-point theorem [37] and the fix-point induction given below [40].

Theorem 3 (*Tarski's Fix-Point Theorem*). *Every monotonic function F over a complete lattice $\langle B, \sqsubseteq\rangle$ has a unique least fix point $\sqcup_{n\in\mathbb{N}_0} F^n(\bot)$ and a unique greatest fix point $\sqcap_{n\in\mathbb{N}_0} F^n(\top)$.*

Theorem 4 (*Scott's Fix-Point Induction*). *Let B be a complete partial order with a bottom ($\bot$), $F : B \to B$ a continuous function, and D an inclusion subset of B. If $\bot \in D$ and $\forall x \in B.\ x \in D \to F(x) \in D$, then $fix(F) \in D$.*

The inclusion subset is defined as follows:

Definition 2 (*Inclusion Subset*). Let P be a complete partial order. A subset D of P is inclusive iff for all ω-chains $d_0 \sqsubseteq d_1 \ldots \sqsubseteq d_n \sqsubseteq \ldots$ in P if $d_n \in D$ for all $n \in \mathbb{N}_0$ then $\bigsqcup_{n\in\mathbb{N}_0} d_n \in D$.

First, we introduce auxiliary notations. Let $\sigma_s^{-1} = \emptyset$, $\sigma_s^i = \{(0, s_0), \ldots, (i, s_i)\}$ (for $i \in \mathbb{N}_0$), and $\sigma_s^\omega = \lim_{i\to\infty} \sigma_s^i$, where σ_s^i denotes a coded set corresponding to the prefix interval σ^i. Then we define $B = \{\sigma_s^i | i \in \{-1\} \cup \mathbb{N}_\omega\}$ with the inclusion relation $\subseteq$. It is easy to prove that $\sigma_s^i \subseteq \sigma_s^j$ iff $i \le j$. Furthermore, let $F : B \to B$ be a function given by $F(\sigma_s^i) = \sigma_s^{i+1}$, for $i \in \{-1\} \cup \mathbb{N}_\omega$. Then, $F(\sigma^\omega) = \sigma^\omega$. Next, we prove the two facts.
(4.1) $(B, \subseteq)$ is a complete lattice.
We first observe that $(B, \subseteq)$ is a partial order:

- reflexivity: for all $\sigma_s^i \in B$, we clearly have $\sigma_s^i \subseteq \sigma_s^i$.
- anti-symmetry: If $\sigma_s^i \subseteq \sigma_s^j$ and $\sigma_s^j \subseteq \sigma_s^i$, then have $i \le j$ and $j \le i$, leading to $i = j$. Hence $\sigma_s^i = \sigma_s^j$.
- transitivity: If $\sigma_s^i \subseteq \sigma_s^j$ and $\sigma_s^j \subseteq \sigma_s^k$, then $i \le j \le k$. Hence $\sigma_s^i \subseteq \sigma_s^k$.

We then observe that there exists a least upper bound (*lub*) and a greatest lower bound (*glb*) in every non-empty subset $A = \{\sigma_s^{i_1}, \ldots, \sigma_s^{i_n}\}$ of B, where $-1 \leq i_1 \leq \cdots \leq i_{n-1} \preceq i_n \leq \omega$. Indeed, we have $glb(A) = \sqcap_{m=1}^{n} \sigma_s^{i_m} = \cap_{m=1}^{n} \sigma_s^{i_m} = \sigma_s^{i_1}$. If A is a finite set ($n < \omega$), then we have $lub(A) = \sqcup_{m=1}^{n} \sigma_s^{i_m} = \cup_{m=1}^{n} \sigma_s^{i_m} = \sigma_s^{i_n}$, and if A is infinite, then $lub(A) = \sqcup_{n\in\mathbb{N}_0} \sigma_s^{i_n} = \cup_{n\in\mathbb{N}_0} \sigma_s^{i_n} = \sigma_s^{\omega}$.

Hence we can conclude that $(B, \subseteq)$ is a complete lattice.

(4.2) F is continuous.

Suppose that $\sigma_s^i \subseteq \sigma_s^j$. Then $i \leq j$, and so $i + 1 \leq j + 1$. As a result,

$$F(\sigma_s^i) = \sigma_s^{i+1} \subseteq \sigma_s^{j+1} = F(\sigma_s^j)$$

Hence F is monotonic. Furthermore, let $\sigma_s^{i_0} \subseteq \sigma_s^{i_1} \subseteq \sigma_s^{i_2} \subseteq \ldots$ be an arbitrary ω-chain in B. It is clear that

$$\sqcup_{n\in\mathbb{N}_0} F(\sigma_s^{i_n}) = \sqcup_{n\in\mathbb{N}_0} \sigma_s^{i_n+1} = \cup_{n\in\mathbb{N}_0} \sigma_s^{i_n+1} = \sigma_s^{\omega} = F(\sigma_s^{\omega}) = F(\sqcup_{n\in\mathbb{N}_0} \sigma_s^{i_n})$$

and so F is continuous.

By (4.1), (4.2) and Theorem 3, we can obtain the least fix-point of F:

$$fix(F) = \sqcup_{n\in\mathbb{N}_0} F^n(\sigma_s^{-1}) = \cup_{n\in\mathbb{N}_0} F^n(\sigma_s^{-1}) = \sigma_s^{\omega}$$

where σ_s^{ω} denotes the coded set corresponding to the whole state sequence σ^{ω}. Now we can construct a subset D of B, as follows:

$$D = \{\sigma_s^i | \sigma_s^i \in B \text{ and } \sigma^i \text{ is a prefix of a model of } P\}$$

In the previous step, we have shown that each prefix σ^i determined by the set σ_s^i is a prefix of a model of P. Hence, for any ω-chain $\sigma_s^0 \sqsubseteq \sigma_s^1 \sqsubseteq \ldots \sqsubseteq \sigma_s^i \sqsubseteq \ldots$ in B, this ω-chain is in fact in D since $\sigma_s^i \in D$. Moreover, $\sigma_s^i \cup \sigma_s^{i+1} = \sigma_s^{i+1}$ for all $i \in \mathbb{N}_0$, we have $\bigcup_{i\in\mathbb{N}_0} \sigma_s^i \in D$, namely, $\bigsqcup_{i\in\mathbb{N}_0} \sigma_s^i \in D$. Therefore, D is an inclusion subset of B. In addition, $\emptyset \in D$ and, by Theorem 4, $fix(F) = \sigma_s^{\omega} \in D$. Hence σ^{ω} is a model of P. □

Theorem 5 (*Completeness*). *The proof system Π_{pptl} is complete, i.e., for all PPTL formula P, $\models P \Longrightarrow \vdash P$.*

Proof. We have the following:

	$\models P$	PREMISE
$\Longleftrightarrow$	$\neg P$ is unsatisfiable	validity and satisfiability
$\Longrightarrow$	$\vdash \neg(\neg P)$	Lemmas 7 and 12
$\Longleftrightarrow$	$\vdash P$	TAU.

Hence the theorem holds. □

5. Verification example

In this section, we show how to use Π_{pptl} in order to verify properties of concurrent systems. We consider a mutual exclusion problem in which two processes, A and B shown below, compete for access to a shared resource R.

process A	process B
forever	forever
$A.flag = 0$	$B.flag = 0$
wait for $(A.flag == 1)$	wait for $(B.flag == 1)$
access R	access R

Accessing the shared resource is managed by a scheduler process S whose strategy is to ensure that the difference between the numbers of accesses granted to A and B is never greater than two:

```
process S
 r = 0
 forever
   wait for (A.flag == 0 and B.flag == 0)
   if(|r| < 2)
   then   P = random(A, B)
          if(P == A)   then(r = r + 1 and A.flag = 1)
                       else(r = r − 1 and B.flag = 1)
   else   if(r < 2)    then(r = r − 1 and B.flag = 1)
                       else(r = r + 1 and A.flag = 1)
```

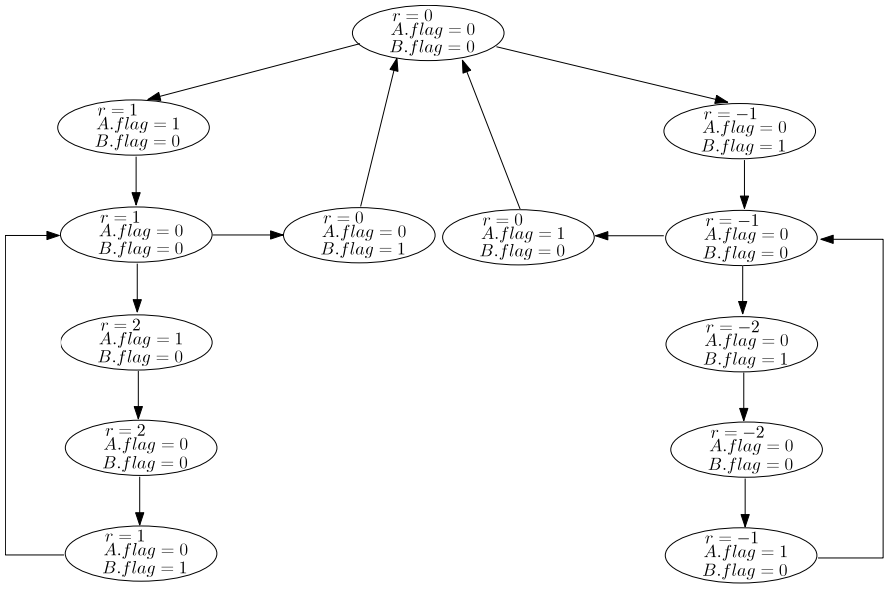

Fig. 13. Mutual exclusion system.

The processes A, B and S are executed concurrently, in the following way (the state diagram of the mutual exclusion system is given in Fig. 13). The system is controlled by three variables: $A.flag$, $B.flag$ and r. Processes A and B are in the critical section if $A.flag = 1$ and $B.flag = 1$, respectively. Variable r indicates the difference between the numbers of times that A and B accessed resource R. If $|r| \leq 1$ then A or B is selected by calling $random(A, B)$; if $r \geq 2$ then B is selected; and if $r \leq -2$ then A is selected. Processes A and B communicate with S by means of variables $A.flag$ and $B.flag$, respectively. The system can be modeled by the following PPTL formula:

$$System \stackrel{\text{def}}{=} r = 0 \wedge ALWAYS \wedge PRJ$$

where the sub-formulas $ALWAYS$ and PRJ are given by:

$$\begin{aligned}
ALWAYS &\stackrel{\text{def}}{=} \Box(A.flag = 0 \wedge B.flag = 0 \rightarrow Opt_1 \wedge Opt_2 \wedge Opt_3)\\
PRJ &\stackrel{\text{def}}{=} (len(2))^{\oplus}\ prj\ \Box(A.flag = 0 \wedge B.flag = 0 \wedge more)\\
Opt_1 &\stackrel{\text{def}}{=} |r| < 2 \rightarrow \bigcirc(A.flag = 1 \wedge B.flag = 0) \wedge \bigcirc^2 r = r + 1 \vee\\
&\qquad\qquad \bigcirc(A.flag = 0 \wedge B.flag = 1) \wedge \bigcirc^2 r = r - 1\\
Opt_2 &\stackrel{\text{def}}{=} r \geq 2 \rightarrow \bigcirc(A.flag = 0 \wedge B.flag = 1) \wedge \bigcirc^2 r = r - 1\\
Opt_3 &\stackrel{\text{def}}{=} r \leq -2 \rightarrow \bigcirc(A.flag = 1 \wedge B.flag = 0) \wedge \bigcirc^2 r = r + 1.
\end{aligned}$$

Note that $ALWAYS$ means that, for any state, if processes A and B are both out of the critical section, S will definitely choose one process to access R at the next state and update the value of r at the following state. PRJ means that at each even state, the two flags are reset to zero and a new process will be chosen to access R. Note also that $System$ never terminates. According to these observations, now want to verify the following property:

$$\begin{aligned}
Property \stackrel{\text{def}}{=}\ & |r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0\\
& \wedge\ (\bigcirc^2(|r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \varepsilon))^*.
\end{aligned}$$

It is well known that $Property$ is a full omega regular property which cannot be expressed by a PLTL or CTL formula [41]. To show that the system satisfies the property we need to prove that $\vdash System \rightarrow Property$ which can be done in two ways, either by a manual proof, or semi-automatically using a theorem prover. In the former case, we first prove the following auxiliary result:

$$\begin{aligned}
\mathsf{TH1}\quad \vdash\quad & r = 0 \wedge ALWAYS \wedge PRJ \rightarrow r = 0 \wedge A.flag = 0 \wedge B.flag = 0 \wedge\\
& \bigcirc^2((r = 1 \vee r = -1) \wedge ALWAYS \wedge PRJ).
\end{aligned}$$

Note that TH1 means that if the current value of r is 0, then it will change to 1 (choosing A) or -1 (choosing B) at the state after the next one. The details of the proof are as follows:

$$
\begin{array}{ll}
 & r = 0 \wedge ALWAYS \wedge PRJ \\
\supset & r = 0 \wedge A.flag = 0 \wedge B.flag = 0 \\
 & \wedge (A.flag = 0 \wedge B.flag = 0 \rightarrow Opt_1 \wedge Opt_2 \wedge Opt_3) \wedge \bigcirc ALWAYS \\
 & \wedge (len(2), (len(2))^{\oplus})\ prj\ \bigcirc \Box (A.flag = 0 \wedge B.flag = 0 \wedge more) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{T4 IUP PSB} \\
\supset & r = 0 \wedge A.flag = 0 \wedge B.flag = 0 \\
 & \wedge (\bigcirc (A.flag = 1 \wedge B.flag = 0) \wedge \bigcirc^2 r = 1 \vee \bigcirc (A.flag = 0 \wedge B.flag = 1) \wedge \\
 & \bigcirc^2 r = -1) \wedge \bigcirc ALWAYS \wedge \bigcirc^2 PRJ \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{TAU PNX} \\
\supset & r = 0 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \bigcirc^2 ((r = 1 \vee r = -1) \wedge ALWAYS \wedge PRJ) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{T1 T4 TAU.}
\end{array}
$$

In a similar way, we can derive the following theorems:

$$
\begin{array}{ll}
\text{TH2} \vdash & r = 1 \wedge ALWAYS \wedge PRJ \\
 & \rightarrow r = 1 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \\
 & \qquad \bigcirc^2 ((r = 0 \vee r = 2) \wedge ALWAYS \wedge PRJ) \\
\text{TH3} \vdash & r = -1 \wedge ALWAYS \wedge PRJ \\
 & \rightarrow r = -1 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \\
 & \qquad \bigcirc^2 ((r = 0 \vee r = -2) \wedge ALWAYS \wedge PRJ) \\
\text{TH4} \vdash & r = 2 \wedge ALWAYS \wedge PRJ \\
 & \rightarrow r = 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \bigcirc^2 (r = 1 \wedge ALWAYS \wedge PRJ) \\
\text{TH5} \vdash & r = -2 \wedge ALWAYS \wedge PRJ \\
 & \rightarrow r = -2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \bigcirc^2 (r = -1 \wedge ALWAYS \wedge PRJ).
\end{array}
$$

From TH1–TH5, theorem T1 and axiom TAU, we can easily derive:

$$
\begin{array}{ll}
\text{TH6} \vdash & |r| \leq 2 \wedge ALWAYS \wedge PRJ \\
 & \rightarrow |r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \bigcirc^2 (|r| \leq 2 \wedge ALWAYS \wedge PRJ).
\end{array}
$$

Now we can prove that $\vdash System \rightarrow Property$, that is that the mutual exclusion system satisfies the desired property:

$$
\begin{array}{ll}
 & r = 0 \wedge ALWAYS \wedge PRJ \\
\supset & |r| \leq 2 \wedge ALWAYS \wedge PRJ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{TAU} \\
\cong & |r| \leq 2 \wedge ALWAYS \wedge PRJ \\
 & \wedge \Box (|r| \leq 2 \wedge ALWAYS \wedge PRJ \\
 & \quad \rightarrow |r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \bigcirc^2 (|r| \leq 2 \wedge ALWAYS \wedge PRJ)) \\
 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{TH6 ALW} \\
\supset & |r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \\
 & \wedge (\bigcirc^2 (|r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \varepsilon))^+ \wedge inf \qquad\qquad\qquad\qquad \text{AIS} \\
\supset & |r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \\
 & \wedge (\bigcirc^2 (|r| \leq 2 \wedge A.flag = 0 \wedge B.flag = 0 \wedge \varepsilon))^* \qquad\qquad\qquad\qquad \text{Def of } *
\end{array}
$$

We have also successfully implemented a theorem prover for Π_{pptl} based on PVS [34]. With this prover, we use PVS specification language to encode the syntax and semantics of PPTL formulas as well as the axioms and inference rules of the proof system. Proof proceeds step by step when we use instructions of PVS to choose axioms or inference rules. Thus, a proof tree can also be generated. We omit full implementation details in this paper.

6. Related work

Among proof systems developed for different extensions of PITL given in the literature [36], three systems are close to that proposed in this paper: (i) The axiom system for a propositional choppy logic with chop, next and until operators was proposed by Rosner and Pnueli [36]. This proof system is complete, and the completeness proof is based on tableau-based decision procedure. In this logic, the model can be both finite and infinite, and it supports both a decision procedure and a complete proof system. However, the logic does not contain chop-star and projection operators. (ii) The axiom system for a version of PITL with next, chop and projection operators was given by Bowman and Thompson [6]. This logic also supports both a decision procedure and a complete proof system. However, the logic is confined to finite models. (iii) The axiom system of PITL with next, chop and projection was given by Moszkowski in [28] for finite models and later extended to infinite models apart from the projection construct. This system is claimed to be complete but no detailed proof is given. Also, there is no decision procedure for this logic.

In contrast, our logic contains next, new projection, and projection-star operators; in particular, the new operators subsume chop, chop-star, and the original projection operator. Furthermore, the logic works with both finite and infinite models, and supports a decision procedure as well as a complete proof system. In addition, we have developed supporting tools for both the decision procedure and the proof system. A model checker for PPTL based on SPIN [38], and a theorem prover based on PVS have also been developed.

7. Conclusion

We presented an axiom system for PPTL which supports both finite and infinite models. We also proved its soundness and completeness. Further, an example was given to illustrate how the system works. This enables one to verify properties of systems by means of deductive approach. We have also developed a prototype theorem prover to support automatic verification. In future, we plan to further develop the theorem prover and use it to tackle large case studies including composite web-services.

References

[1] M. Abadi, An axiomatization of Lamport's temporal logic of actions, in: J. Baeten, J. Klop (Eds.), Proceedings of CONCUR'90, Theories of Concurrency: Unification and Extension, in: LNCS, vol. 458, Springer, Heidelberg, 1990, pp. 57–69.
[2] B. Banieqbal, H. Barringer, Temporal logic with fixed points, in: B. Banieqbal, H. Barringer, A. Pnueli (Eds.), Temporal Logic in Specification, in: LNCS, vol. 389, Springer, Heidelberg, 1987, pp. 62–74.
[3] M. Ben-Ari, Principles of Concurrent and Distributed Programming, first edition, Reading, Massachusetts, 1990.
[4] Y. Bertot, P. Castéran, Interactive Theorem Proving and Program Development, Heidelberg, 2004.
[5] W. Bledsoe, D. Loveland, Automating Theorem Proving: After 25 Years, Providence, 1984.
[6] H. Bowman, S. Thompson, A decision procedure and complete axiomatization of finite interval logic with projection, J. Logic Comput. 13 (2003) 195–239.
[7] B. Brock, M. Kaufmann, J. Moore, ACL2 theorems about commercial microprocessors, in: M. Srivas, A. Camilleri (Eds.), Proceedings of the First International Conference on Formal Methods in Computer-Aided Design, in: LNCS, vol. 1166, Springer, London, 1996, pp. 275–293.
[8] E.M. Clarke, E.A. Emerson, Design and synthesis of synchronisation skeletons using branching time temporal logic, in: D. Kozen (Ed.), Proceedings of the Workshop on Logic of Programs, in: LNCS, vol. 131, Springer, Heidelberg, 1981, pp. 52–71.
[9] E.M. Clarke, E.A. Emerson, A.P. Sistla, Automatic verification of finite state concurrent system using temporal logic specification, ACM Trans. Program. Lang. Syst. 8 (2) (1986) 244–263.
[10] T. Coquand, G. Huet, Constructions: a higher order proof system for mechanizing mathematics, in: B. Buchberger (Ed.), EUROCAL'85: European Conference on Computer Algebra, Volume 1: Invited lectures, in: LNCS, vol. 203, Springer, Heidelberg, 1985, pp. 151–184.
[11] Z. Duan, Temporal Logic and Temporal Logic Programming, Science Press, Beijing, 2005.
[12] Z. Duan, M. Holcombe, A. Bell, A logic for biosystems, Biosyst. 55 (1–3) (2000) 93–105.
[13] Z. Duan, M. Koutny, A framed temporal logic programming language, J. Comput. Sci. Technol. 19 (2004) 333–344.
[14] Z. Duan, M. Koutny, C. Holt, Projection in temporal logic programming, in: F. Pfenning (Ed.), Proceedings of Logic Programming and Automated Reasoning, in: LNAI, vol. 822, Springer, Heidelberg, 1994, pp. 333–334.
[15] Z. Duan, C. Tian, L. Zhang, A decision procedure for propositional projection temporal logic, Acta Inform. 45 (2008) 43–78.
[16] Z. Duan, X. Yang, M. Koutny, Semantics of framed temporal logic programs, in: M. Gabbrielli, G. Gupta (Eds.), Proceedings of ICLP 2005, in: LNCS, vol. 3668, Springer, Heidelberg, 2005, pp. 256–270.
[17] Z. Duan, X. Yang, M. Koutny, Framed temporal logic programming, Sci. Comput. Program. 70 (2008) 31–61.
[18] G.J. Holzmann, The model checker SPIN, IEEE Trans. Softw. Eng. 23 (5) (1997) 279–295.
[19] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory Languages and Computation, New Jersey, 1979.
[20] Y. Kesten, A. Pnueli, A complete proof system for QPTL, in: Proceedings of the 10th IEEE Symposium on Logic in Computer Science, IEEE Computer Society, Washington, 1995, pp. 2–12.
[21] S. Kono, A combination of clausal and non-clausal temporal logic programs, in: G. Goos, J. Hartmanis, J. Leeuwen (Eds.), Proceedings of the Workshop on Executable Modal and Temporal Logics, in: LNCS, vol. 897, Springer, Heidelberg, 1995, pp. 40–57.
[22] S.A. Kripke, Semantical analysis of modal logic I: normal propositional calculi, Z. Math. Log. Grund. Math. 9 (1963) 67–96.
[23] F. Kröger, Temporal Logic of Programs, New York, 1987.
[24] L. Lamport, The temporal logic of actions, ACM Trans. Program. Lang. Syst. 16 (1994) 872–923.
[25] Z. Manna, A. Pnueli, The Temporal Logic of Reactive and Concurrent System, New York, 1992.
[26] K.L. McMillan, Symbolic Model Checking: An Approach to the State Explosion Problem, Dordrecht, 1993.
[27] B.C. Moszkowski, Executing Temporal Logic Programs, Cambridge, 1986.
[28] B.C. Moszkowski, Some very compositional temporal properties, in: E.-R. Olderog (Ed.), Proceedings of Programming Concepts, Methods and Calculi, IFIP Trans. A-56, IFIP, Elsevier Science B.V, North-Holland, 1994, pp. 307–326.
[29] B.C. Moszkowski, Compositional reasoning about projected and infinite time, in: Proceedings of the 1st IEEE International Conference on Engineering of Complex Computer Systems, ICECCS'95, IEEE Computer Society, Washington, 1995, pp. 238–245.
[30] J. Queille, J. Sifakis, Specification and verification of concurrent systems in CESAR, in: M. Dezani-Ciancaglini, U. Montanari (Eds.), Proceedings of the 5th Colloquium on International Symposium in Programming, in: LNCS, vol. 137, Springer, London, 1982, pp. 337–351.
[31] B. Paech, Gentzen-Systems for propositional temporal logics, in: E. Börger (Ed.), Proceedings of the 2nd Workshop on Computer Science Logic, in: LNCS, vol. 385, Springer, London, 1988, pp. 240–253.
[32] A. Pnueli, The temporal logic of programs, in: Proceedings of the 18th Symposium on the Foundations of Computer Science, IEEE Computer Society, Washington, 1977, pp. 46–57.
[33] A. Pnueli, Y. Kesten, A deductive proof system for CTL, in: L. Brim, P. Jancar, M. Kretínský, A. Kucera (Eds.), Proceedings of the 13th International Conference on Concurrency Theory, in: LNCS, vol. 2421, Springer, London, 2002, pp. 24–40.
[34] S. Owre, J. Rushby, N. Shankar, PVS: a prototype verification system, in: Proceedings of the 11th International Conference on Automated Deduction, in: LNAI, vol. 607, Springer, Heidelberg, 1992, pp. 748–752.
[35] N. Rescher, A. Urquhart, Temporal Logic, New York, 1978.
[36] R. Rosner, A. Pnueli, A choppy logic, in: Proceedings of the 1st IEEE Symposium on Logic in Computer Science, IEEE Computer Society, Washington, 1986, pp. 306–314.
[37] A. Tarski, A lattice-theoretical fixpoint theorem and its applications, Pacific J. Math. 5 (1955) 285–309.
[38] C. Tian, Z. Duan, Model checking propositional projection temporal logic based on SPIN, in: M. Butler, M.G. Hinchey, M.M. Larrondo-Petrie (Eds.), Proceedings of the 9th International Annual Conference on Formal Engineering Methods, in: LNCS, vol. 4789, Springer, Heidelberg, 2007, pp. 246–265.
[39] C. Tian, Z. Duan, Complexity of propositional projection temporal logic with star, Math. Struct. Comput. Sci. 19 (2009) 1–28.
[40] G. Winskel, The Formal Semantics of Programming Languages: An Introduction, Cambrige, 1993.
[41] P.L. Wolper, Temporal logic can be more expressive, Inf. Control 56 (1983) 72–99.
[42] X. Yang, Z. Duan, Operational semantics of framed tempura, J. Log. Algebr. Program. 78 (2008) 22–51.