Normal Form Expressions of Propositional Projection Temporal Logic*

Zhenhua Duan, Cong Tian**, and Nan Zhang

Institute of Computing Theory and Technology, and ISN Laboratory
Xidian University, Xi'an 710071, China
ctian@mail.xidian.edu.cn

Abstract. This paper presents normal form expressions of Propositional Projection Temporal Logic (PPTL). For doing so, a PPTL formula is represented as the disjunction of formulas in form of $e^k_\varepsilon = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon$ or $e^{(k,l)}_\omega = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j (\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l})$, $1 \le l \in N_0$. Here e^k_ε denotes a finite model with length being k while $e^{(k,l)}_\omega$ indicates an infinite model. We show that any PPTL formula can be expressed as a normal form expression. As a consequence, satisfiability of PPTL formulas can easily be achieved.

Keywords: Propositional projection temporal logic, normal form expression, normal form, specification, satisfiability.

1 Introduction

Temporal logics are popular formalisations that can express properties about the temporal order of events. They are widely used in model checking for specifying desired properties of a system to be verified. The family of temporal logics has grown over the years, containing linear [9] and branching time logics [4,2], and, more recently, game, alternating time, and coordination logics [1,10]. While linear time temporal logics are concerned with properties of paths, branching time logics describe properties that depend on the branching of computational tree structures.

Interval-based temporal logics such as Interval Temporal Logic (ITL) [11] and Projection Temporal Logic (PTL) [5,6,7] which extends ITL with infinite models and a new projection construct, $(P_1, \ldots, P_m)\,\mathsf{prj}\,Q$, are a more recent branch of temporal logics with their own niche of interesting applications. Propositional PTL (PPTL) is a propositional subset of PTL with a usual next construct and the projection construct that is able to express chop construct, often denoted by the symbol '$;$', by $P;Q \stackrel{\text{def}}{=} (P,Q)\,\mathsf{prj}\,\varepsilon$. Compared with classic temporal logics, interval-based temporal logics greatly simplify the formulation of certain correctness properties [12], which underlines the usefulness of these logics for specification and formal reasoning about concurrent systems. Interval-based temporal logics lend themselves particularly well to reasoning about

* The 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 No. 61133001, 61202038, 61272117, 61272118, 61322202 and 91218301.

** Corresponding author.

Z. Cai et al. (Eds.): COCOON 2014, LNCS 8591, pp. 84–93, 2014.

properties with a 'scope'; such properties are quite common in most programming languages. Further, with *chop* operators, sequential behaviours can be described elegantly and succinctly; and full regular expressiveness is achieved by projection construct.

In this paper, we present normal form expression that represents a PPTL formula as the disjunction of formulas in form of

$$e_{\varepsilon}^{k} = \bigwedge_{0 \le i \le k \in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon$$

or

$$e_{\omega}^{(k,l)} = \bigwedge_{0 \le i \le k \in N_0} \bigcirc^i S_i \wedge \bigwedge_{k \le j \in N_\omega} \bigcirc^j (\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l})$$

$1 \le l \in N_0$, that implicitly depicts a finite or an infinite model of the corresponding PPTL formula, respectively. We prove that any PPTL formula can be expressed in a normal form expression. As a consequence, satisfiability of PPTL formulas can easily be achieved.

The rest of the paper is organized as follows. The following section presents syntax and semantics of PPTL. Normal form expressions are defined in Section 3. We then show that any PPTL formula can be represented as a normal form expression in Section 4. As a consequence, a decision procedure for checking the satisfiability of PPTL formulas based on normal form expressions is presented in Section 5. Finally, conclusions are drawn in Section 6.

2 Propositional Projection Temporal Logic

Propositional Projection Temporal Logic (PPTL) [5,13] is an extension of Propositional ITL (PITL) [14] with infinite models and a new projection construct [6,15].

Let *Prop* be a countable set of atomic propositions and $B = \{true, false\}$ the boolean domain. We use small letters, possibly with subscripts, like *p,q,r* to denote atomic propositions, and capital letters, possibly with subscripts, for instance *P, Q, R* to indicate general PPTL formulas. Formulas of PPTL are defined by the following grammar:

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

where $p \in Prop$, $\bigcirc$ (next), and prj (projection) are temporal operations.

We define a *state s* over *Prop* to be a mapping from *Prop* to *B*, $s : Prop \to B$. We write $s[p]$ to denote the valuation of *p* at state *s*. An *interval* $\sigma = \langle s_0, s_1, ... \rangle$ is a non-empty sequence of states, which can be finite or infinite. The length of σ, $|\sigma|$, is the number of states in σ minus one if σ is finite; otherwise it is ω. Let N_0 denote the set of non-negative integers. To have a uniform notation for both finite and infinite intervals, we will use *extended integers* as indices, that is $N_\omega = N_0 \cup \{\omega\}$, and extend the comparison operators, $=, <, \le$, to N_ω by considering $\omega = \omega$ and for all $i \in N_0, i < \omega$. Moreover, we write $\le$ as $\le -\{(\omega, \omega)\}$.

To formalize the semantics of the projection construct, we need an auxiliary operator $\downarrow$. Let $\sigma = \langle s_0, s_1, \ldots \rangle$ be an interval and $r_1, \ldots, r_h$ be integers ($h \ge 1$) such that

$0 \le r_1 \le \ldots \le r_h \preceq |\sigma|$. The projection of σ onto $r_1, \ldots, r_h$ is the *projected interval*, $\sigma \downarrow (r_1, \ldots, r_h) \stackrel{\text{def}}{=} \langle s_{t_1}, s_{t_2}, \ldots, s_{t_l} \rangle$, where $t_1, \ldots, t_l$ are attained from $r_1, \ldots, r_h$ by deleting all duplicates. In other words, $t_1, \ldots, t_l$ is the longest strictly increasing subsequence of $r_1, \ldots, r_h$. For instance, $\langle s_0, s_1, s_2, s_3 \rangle \downarrow (0, 2, 2, 2, 3) = \langle s_0, s_2, s_3 \rangle$. The concatenation$(\cdot)$ of an interval $\sigma = \langle s_0, s_1, \ldots, s_{|\sigma|} \rangle$ with another interval $\sigma' = \langle s'_0, s'_1, \ldots, s'_{|\sigma|} \rangle$ is represented by $\sigma \cdot \sigma' = \langle s_0, s_1, ..., s_{|\sigma|}, s'_0, s'_1, \ldots, s'_{|\sigma|} \rangle$ (not sharing any states).

An *interpretation* is a tuple $\mathcal{I} = (\sigma, k, j)$, where $\sigma = \langle s_0, s_1, \ldots \rangle$ is an interval, k is a non-negative integer, and j is an integer or ω, such that $0 \le k \le j \le |\sigma|$. We write (σ, k, j) to mean that a formula is interpreted over a subinterval $\sigma_{k,\cdots,j}$ with the current state being s_k. We utilize I^k_{prop} to stand for the state interpretation at state s_k. The satisfaction relation $\models$ for formulas is given as follows:

$\mathcal{I} = (\sigma, k, j) \models p$ iff $s_k[p] = I^k_{prop}[p] = true$

$\mathcal{I} = (\sigma, k, j) \models \neg P$ iff $\mathcal{I} \not\models P$

$\mathcal{I} = (\sigma, k, j) \models P_1 \wedge P_2$ iff $\mathcal{I} \models P_1$ and $\mathcal{I} \models P_2$

$\mathcal{I} = (\sigma, k, j) \models \bigcirc P$ iff $k < j$ and $(\sigma, k+1, j) \models P$

$\mathcal{I} = (\sigma, k, j) \models (P_1, \ldots, P_m)\ \mathsf{prj}\ P$ iff there exist integers $r_0, \ldots, r_m$, and $k = r_0 \le \ldots \le r_{m-1} \le r_m \le j$ such that $(\sigma, r_{l-1}, r_l) \models P_l$ for all $1 \le l \le m$ and $(\sigma', 0, |\sigma'|) \models P$ for σ' given by :
(1) $r_m < j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_m) \cdot \sigma_{(r_m+1,\ldots,j)}$
(2) $r_m = j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_h)$ for some $0 \le h \le m$

For convenience, some derived formulas from elementary PPTL formulas are presented below. The abbreviations true, false, $\vee$, $\rightarrow$ and $\leftrightarrow$ are defined as usual.

ε	$\stackrel{\text{def}}{=}$	$\neg \bigcirc \mathsf{true}$	more	$\stackrel{\text{def}}{=}$	$\neg\, \varepsilon$
$\Diamond P$	$\stackrel{\text{def}}{=}$	$(\mathsf{true}, P)\ \mathsf{prj}\ \varepsilon$	$\Box P$	$\stackrel{\text{def}}{=}$	$\neg \Diamond \neg\, P$
$\mathsf{fin}(P)$	$\stackrel{\text{def}}{=}$	$\Box(\varepsilon \rightarrow P)$	$\mathsf{halt}(P)$	$\stackrel{\text{def}}{=}$	$\Box(\varepsilon \leftrightarrow P)$
$\mathsf{keep}(P)$	$\stackrel{\text{def}}{=}$	$\Box(\neg\varepsilon \rightarrow P)$	$\mathsf{rem}(P)$	$\stackrel{\text{def}}{=}$	$\Box(\mathsf{more} \rightarrow \bigcirc P)$
$P ; Q$	$\stackrel{\text{def}}{=}$	$(P, Q)\ \mathsf{prj}\ \varepsilon$	$P ;_{\mathsf{w}} Q$	$\stackrel{\text{def}}{=}$	$(P ; Q) \vee (P \wedge \Box \mathsf{more})$
fin	$\stackrel{\text{def}}{=}$	$\Diamond\varepsilon$	inf	$\stackrel{\text{def}}{=}$	$\Box \mathsf{more}$
$\mathsf{len}(0)$	$\stackrel{\text{def}}{=}$	ε	$len(n)$	$\stackrel{\text{def}}{=}$	$\bigcirc len(n-1), n \ge 1$
$\odot P$	$\stackrel{\text{def}}{=}$	$\varepsilon \vee \bigcirc P$	$skip$	$\stackrel{\text{def}}{=}$	$len(1)$

A PPTL formula containing no temporal operators is called a state formula.

Further, we have the following useful logic laws, whose proofs can be found in [13]:

(L1) $\Diamond P$	$\equiv P \vee \bigcirc\Diamond P$	(L2) $\Box P$	$\equiv P \wedge \varepsilon \vee P \wedge \bigcirc\Box P$
(L3) $\neg \bigcirc P$	$\equiv \odot \neg P$	(L4) $Q;(P_1 \vee P_2)$	$\equiv (Q;P_1) \vee (Q;P_2)$
(L5) $\Box\neg Q$	$\equiv \neg\Diamond Q$	(L6) keep(P)	$\equiv \varepsilon \vee P \wedge \bigcirc$keep($P$)
(L7) $\Box P \vee \Box Q$	$\supset \Box(P \vee Q)$	(L8) halt(P)	$\equiv P \wedge \varepsilon \vee \neg P \wedge \bigcirc$halt($P$)
(L9) $\Box(P \wedge Q)$	$\equiv \Box P \wedge \Box Q$	(L10) fin(P)	$\equiv P \wedge \varepsilon \vee \bigcirc$fin($P$)
(L11) true	$\equiv \Diamond\varepsilon \vee \Box$more	(L12) $P_1;(P_2;P_3)$	$\equiv (P_1;P_2);P_3$
(L13) $\bigcirc(P \vee Q)$	$\equiv \bigcirc P \vee \bigcirc Q$	(L14) $\Box(P \wedge$ more$)$	$\equiv P \wedge \bigcirc\Box(P \wedge$ more$)$
(L15) $\bigcirc(P \wedge Q)$	$\equiv \bigcirc P \wedge \bigcirc Q$	(L16) $\Box(P \to Q)$	$\supset (\Box P \to \Box Q)$
(L17) true	$\equiv \varepsilon \vee \bigcirc$true	(L18) more $\wedge \bigcirc\neg P$	$\equiv$ more $\wedge \neg \bigcirc P$

3 Normal Form Expressions

Now we define normal form expressions that implicitly express models of temporal logic formulas.

Definition 1 (Normal Form Expressions). Let

$$E_\varepsilon ::= \{e_\varepsilon^k \mid e_\varepsilon^k = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon\}$$
$$E_\omega ::= \{e_\omega^{(k,l)} \mid e_\omega^{(k,l)} = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j(\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l}), 1 \le l \in N_0\}$$

Here, each S (possibly with subscripts) is a state formula. The set of normal form expressions are defined by:

$$E ::= \{e \mid e = \bigvee_{1\le m\in N_0} e^m, e^m \in E_\varepsilon \cup E_\omega\}$$

Every $e \in E$ is a normal form expression. □

Intuitively, in a normal form expression, each

$$e_\varepsilon^k = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon$$

in E_ε denotes a finite interval with length being k, where for each $0 \le i \le k$, state formula S_i holds at state i as illustrated in Fig. 1 (1). Whereas each

$$e_\omega^{(k,l)} = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j(\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l})$$

in E_ω depicts an infinite model with a loop suffix where S_i holds at state i in case $0 \le i \le k$, and S_{k+j}, $1 \le j \le l$, holds at state $k + m \times j$ for all $m \ge 1$ as shown in Fig. 1 (2). Further, let E be the set of all normal form expressions, *true* (or T) and $false$ (or F) can be expressed by $true \stackrel{\text{def}}{=} E$ and $false \stackrel{\text{def}}{=} \emptyset$, respectively.

The merits of normal form expressions are twofold: (1) compared with temporal logic formulas, they are much more intuitive in acquiring the underlying meaning of the formula; (2) in contrast to graphical models of temporal logic formulas, they are more compact and convenient in logic operations. In the following, we show two simple examples of normal form expressions.

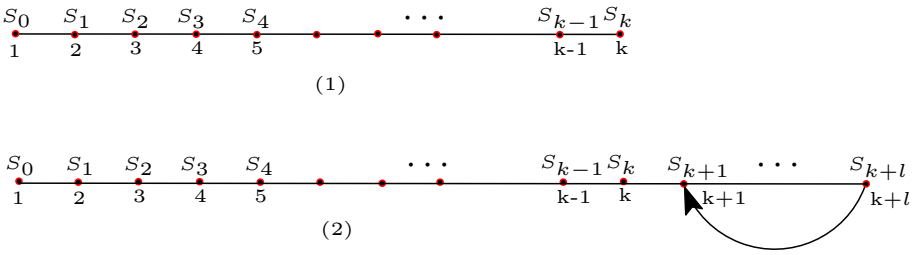

Fig. 1. Intervals expressed by e_{ε}^{k} and $e_{\omega}^{(k,l)}$

Example 1. Examples of normal form expressions:

(1) Normal form expression of a proposition p:

$$p \equiv \bigvee_{0 \le i \in N_0} p \wedge \bigcirc^i \varepsilon \vee p \wedge \bigcirc^\omega true$$

It hints that two kinds of models will satisfy p. The first kind of models contains the finite ones with an arbitrary length such that p holds at the first state as shown in Fig. 2 (1), while the second kind includes only one infinite model where p holds at the first states as illustrated in Fig. 2 (2), here T denotes true.

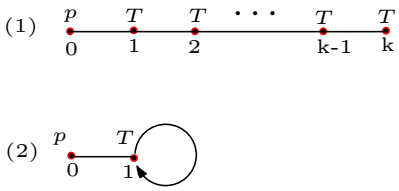

Fig. 2. Models of proposition p

(2) Normal form expression of formula $\Diamond p$:

$$\Diamond p \equiv \bigvee_{0 \le i < k \in N_0, 0 \le j} \bigcirc^i true \wedge \bigcirc^k p \wedge \bigcirc^{k+j} \varepsilon \vee \bigvee_{0 \le i < k \in N_0} \bigcirc^i true \wedge \bigcirc^k p \wedge \bigcirc^\omega true$$

It indicates that the models that satisfy $\Diamond p$ are finite or infinite ones where p holds at some state throughout the intervals as shown in Fig. 3 (1) and (2), respectively.

4 Normal Form Expressions of PPTL

In this section, we show that any PPTL formula can be equivalently transformed to a normal form expression. We first show some results useful in the transformation.

Lemma 1 shows that the negation of a normal form expression will still be a normal form expression.

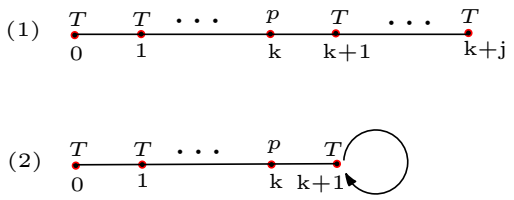

Fig. 3. Models of $\Diamond p$

Lemma 1. *For any $e \in E_\varepsilon \cup E_\omega$, $\neg e$ can be transformed to normal form expression.*

Proof: In case $e = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon$. We have:

$$
\begin{aligned}
\neg e &\equiv \neg(\bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon) \\
&\equiv \bigvee_{0\le i\le k\in N_0} \neg \bigcirc^i S_i \vee \neg \bigcirc^k \varepsilon \\
&\equiv \bigvee_{0\le i\le k\in N_0} \odot^i \neg S_i \vee \odot^k \neg\varepsilon \\
&\equiv \bigvee_{0\le i\le k\in N_0} \odot^i \neg S_i \vee \odot^k \mathsf{more} \\
&\equiv \bigvee_{0\le i\le k\in N_0} \bigcirc^i \neg S_i \vee \bigcirc^k \mathsf{more} \vee \bigvee_{0\le i\le k-1\in N_0} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\le k\in N_0} \bigcirc^i (\bigvee_{0\le j\in N_0} \neg S_i \wedge \bigcirc^j \varepsilon \vee \neg S_i \wedge \bigcirc^\omega true) \vee \bigcirc^k \mathsf{more} \vee \bigvee_{0\le i\le k-1\in N_0} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\le k\in N_0} \bigvee_{0\le j\in N_0} \bigcirc^i \neg S_i \wedge \bigcirc^{i+j} \varepsilon \vee \bigcirc^i \neg S_i \wedge \bigcirc^\omega true \vee \bigcirc^k \mathsf{more} \vee \bigvee_{0\le i\le k-1\in N_0} \bigcirc^i \varepsilon
\end{aligned}
$$

So in this case $\neg e$ has been represented as a normal form expression. Further, if $e = \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j (\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l})$, $1 \le l \in N_0$. We have:

$$\neg e \equiv \neg(\bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j (\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l}))$$

$$
\begin{aligned}
&\equiv \bigvee_{0\le i\le k\in N_0} \bigcirc^i \neg S_i \vee \bigvee_{k\le j\in N_\omega} \bigcirc^j (\bigcirc \neg S_{k+1} \vee \bigcirc^2 \neg S_{k+2} \vee \cdots \vee \bigcirc^l \neg S_{k+l}) \vee \bigvee_{0\le i\in N_\omega} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\in N_\omega} \bigcirc^i \neg S_i \vee \bigvee_{0\le i\in N_\omega} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\in N_\omega} \bigcirc^i (\bigvee_{0\le j\in N_0} \neg S_i \wedge \bigcirc^j \varepsilon \vee \neg S_i \wedge \bigcirc^\omega true) \vee \bigvee_{0\le i\in N_\omega} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\le j\in N_\omega} \bigcirc^i \neg S_i \wedge \bigcirc^{i+j} \varepsilon \vee \bigvee_{0\le i\in N_\omega} \bigcirc^i \neg S_i \wedge \bigcirc^{i+\omega} true \vee \bigvee_{0\le i\in N_\omega} \bigcirc^i \varepsilon \\
&\equiv \bigvee_{0\le i\le j\in N_0} \bigcirc^i \neg S_i \wedge \bigcirc^{i+j} \varepsilon \vee \bigvee_{0\le i\in N_0} \bigcirc^i \neg S_i \wedge \bigcirc^\omega true \vee \bigvee_{0\le i\in N_0} \bigcirc^i \varepsilon
\end{aligned}
$$

Hence the lemma holds. □

Lemma 2 indicates that the conjunction of normal form expressions is still a normal form expression.

Lemma 2. *Let e_1 and e_2 be normal form expressions. $e_1 \wedge e_2$ can be expressed by a normal form expression.*

Proof: Suppose $e_1 \equiv \bigvee_{0\le k_1\in N_0} e_\varepsilon^{k_1} \vee \bigvee_{0\le k_1'\in N_0} e_\omega^{(k_1',l_1)}$, $1 \le l_1 \in N_0$, and $e_2 \equiv \bigvee_{0\le k_2\in N_0} e_\varepsilon^{k_2} \vee \bigvee_{0\le k_2'\in N_0} e_\omega^{(k_2',l_2)}$, $1 \le l_2 \in N_0$. We have,

$$\begin{aligned} e_1 \wedge e_2 &\equiv (\bigvee_{0\le k_1\in N_0} e_\varepsilon^{k_1} \vee \bigvee_{0\le k_1'\in N_0} e_\omega^{(k_1',l_1)}) \wedge (\bigvee_{0\le k_2\in N_0} e_\varepsilon^{k_2} \vee \bigvee_{0\le k_2'\in N_0} e_\omega^{(k_2',l_2)}) \\ &\equiv \bigvee_{0\le k_1\in N_0} \bigvee_{0\le k_2\in N_0} e_\varepsilon^{k_1} \wedge e_\varepsilon^{k_2} \vee e_\omega^{(k_1',l_1)} \wedge e_\omega^{(k_2',l_2)} \end{aligned}$$

It is ready that $e_1 \wedge e_2$ can be expressed by a normal form expression. □

Lemma 3 presents how normal form expression of a chop construct is obtained.

Lemma 3. *$e_\varepsilon^{k_1}$; $e_\varepsilon^{k_2}$ and $e_\varepsilon^{k_1}$; $e_\omega^{k,l}$ can be expressed by normal form expressions.*

Proof: Suppose,

$$\begin{aligned} e_\varepsilon^{k_1} &\equiv \bigwedge_{0\le i\le k_1\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_1}\varepsilon \\ e_\varepsilon^{k_2} &\equiv \bigwedge_{0\le i\le k_2\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_2}\varepsilon \\ e_\omega^{(k,l)} &\equiv \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j(\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l}) \end{aligned}$$

We have,

$$\begin{aligned} e_\varepsilon^{k_1}\,;\, e_\varepsilon^{k_2} &\equiv (\bigwedge_{0\le i\le k_1\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_1}\varepsilon)\,;\,(\bigwedge_{0\le i\le k_2\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_2}\varepsilon) \\ &\equiv \bigwedge_{0\le i\le k_1+k_2\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_1+k_2}\varepsilon \end{aligned}$$

$$\begin{aligned} e_\varepsilon^{k_1}\,;\, e_\omega^{k,l} &\equiv (\bigwedge_{0\le i\le k_1\in N_0} \bigcirc^i S_i \wedge \bigcirc^{k_1}\varepsilon)\,;\,(\bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k\le j\in N_\omega} \bigcirc^j(\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l})) \\ &\equiv \bigwedge_{0\le i\le k_1+k\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k_1+k\le j\in N_\omega} \bigcirc^j(\bigcirc S_{k+1} \wedge \bigcirc^2 S_{k+2} \wedge \cdots \wedge \bigcirc^l S_{k+l}) \end{aligned}$$

Thus, the lemma holds. □

Lemma 4 shows that projection construct can be expressed by a normal form expression.

Lemma 4. *If $P_0, \cdots, P_m$, and Q can be expressed by normal form expressions, so does $(P_0, \cdots, P_m)$ prj Q.*

Proof: Without loss of generality, suppose

$$\begin{aligned} E_Q &\equiv e_\varepsilon^{k} \vee e_\omega^{(k',l)} \\ E_{P_0} &\equiv e_\varepsilon^{k_0} \vee e_\omega^{(k_0',l_0)} \\ &\cdots \\ E_{P_m} &\equiv e_\varepsilon^{k_m} \vee e_\omega^{(k_m',l_m)} \end{aligned}$$

Here for convenience, we represent E_Q as $\bigwedge_{0\le i\in N_\omega} \bigcirc^i S_i$. It has:

$$(P_0, \cdots, P_m)\, prj\ Q \equiv (E_{P_0}, \cdots, E_{P_m})\, prj\ E_Q$$

By the semantics of projection construction, P_1, $\cdots$, and P_{m-1} are confined to finite models. In case $i \le m$, we have:

$$(E_{P_0}, \cdots, E_{P_m})\, prj\ E_Q \equiv e_\varepsilon^{k_0} \wedge \mathsf{fin}(S_0)\,;\, \cdots\,;\, e_\varepsilon^{k_i} \wedge \mathsf{fin}(S_i)\,;\, e_\varepsilon^{k_{i+1}} \cdots\,;\, e_\varepsilon^{k_{m-1}}\,;\, (e_\varepsilon^{k_m} \vee e_\omega^{(k'_m, l_m)})$$

In case $i > m$, we have:

$$(E_{P_0}, \cdots, E_{P_m})\, prj\ E_Q \equiv e_\varepsilon^{k_0} \wedge \mathsf{fin}(S_0)\,;\, \cdots\,;\, e_\varepsilon^{k_m} \wedge \mathsf{fin}(S_m)\,;\, \bigwedge_{m<i} \bigcirc^i S_i$$

Note that each $e_\varepsilon^{k_i} \wedge \mathsf{fin}(S_i)$ means that S_i is conjuncted with the state formula holding at the last state of the interval specified by $e_\varepsilon^{k_i}$. Thus, by Lemma 3, the lemma is already proved. □

Now the main theorem is presented.

Theorem 5. *Any PPTL formula can be equivalently expressed by a normal form expression.*

Proof: The proof proceeds by induction on the structure of PPTL formulas. As the base case, we have shown that a proposition p can be expressed as a normal form expression. Suppose PPTL formulas P (or P with subscripts) and Q have been expressed as normal form expressions E_P and E_Q, respectively.

1. Formula $\neg P$ can be expressed as normal form expression by:

$$\begin{aligned} \neg P &\equiv \neg E_p \\ &\equiv \neg(\bigvee_{1\le i\in N_0} e^i), e^i \in E_\varepsilon \cup E_\omega \\ &\equiv \bigwedge_{1\le i\in N_0} \neg e^i \end{aligned}$$

   By Lemma 1 and 2, $\neg P \equiv \bigwedge_{1\le i\in N_0} \neg e^i$ can be further expressed in a normal form expression.
2. Formula $P_1 \vee P_2$ can be expressed as a normal form expression by:

$$P_1 \vee P_2 \equiv E_{P_1} \vee E_{P_2}$$

3. Formula $\bigcirc P$ can be expressed as a normal form expression by:

$$\bigcirc P \equiv \bigcirc E_P$$

   $\bigcirc E_P$ is already in normal form expression.
4. Formula $P\,;\, Q$ can be expressed as a normal form expression by:

$$\begin{aligned} P_1 ; P_2 &\equiv E_{P_1} ; E_{P_2} \\ &\equiv (\bigvee_{0\le k_1\in N_0} e_\varepsilon^{k_1} \vee \bigvee_{0\le k_1'\in N_0} e_\omega^{(k_1',l_1)}) ; (\bigvee_{0\le k_2\in N_0} e_\varepsilon^{k_2} \vee \bigvee_{0\le k_2'\in N_0} e_\omega^{(k_2',l_2)}) \\ &\equiv (\bigvee_{0\le k_1\in N_0} e_\varepsilon^{k_1}) ; (\bigvee_{0\le k_2\in N_0} e_\varepsilon^{k_2} \vee \bigvee_{0\le k_2'\in N_0} e_\omega^{(k_2',l_2)}) \\ &\equiv \bigvee_{0\le k_1\in N_0} \bigvee_{0\le k_2\in N_0} (e_\varepsilon^{k_1} ; e_\varepsilon^{k_2} \vee e_\varepsilon^{k_1} ; e_\omega^{(k_2',l_2)}) \end{aligned}$$

By Lemma 3, $P_1 ; P_2 \equiv \bigvee_{0\le k_1\in N_0} \bigvee_{0\le k_2\in N_0} (e_\varepsilon^{k_1} ; e_\varepsilon^{k_2} \vee e_\varepsilon^{k_1} ; e_\omega^{(k_2',l_2)})$ can be further expressed in a normal form expression.

5. By Lemma 4, formula $(P_1, \cdots, P_m)\, prj\ Q$ can be expressed in a normal form expression.

Accordingly, any PPTL formula can be equivalently expressed by a normal form expression. □

The above proofs also provide an approach for transforming a PPTL formula to a normal form expression.

5 Decision Procedure of PPTL

Based on normal form expressions, how to check the satisfiability of PPTL formulas becomes simple. Give a PPTL formula P, we first transform P to its normal form expression:

$$P \equiv \bigvee_{0\le k\in N_0} e_\varepsilon^k \vee \bigvee_{0\le k'\in N_0} e_\omega^{(k',l)}$$

where

$$\begin{aligned} e_\varepsilon^k &= \bigwedge_{0\le i\le k\in N_0} \bigcirc^i S_i \wedge \bigcirc^k \varepsilon \\ e_\omega^{(k',l)} &= \bigwedge_{0\le i\le k'\in N_0} \bigcirc^i S_i \wedge \bigwedge_{k'\le j\in N_\omega} \bigcirc^j (\bigcirc S_{k'+1} \wedge \bigcirc^2 S_{k'+2} \wedge \cdots \wedge \bigcirc^l S_{k'+l}) \end{aligned}$$

$1 \le l \in N_0$. Then for all k and k', if there exists an S (or with subscript) such that S is unsatisfiable, P is unsatisfiable; otherwise, P is satisfiable. As a matter of fact, each S (or with subscript) is a state formula without any temporal operators, i.e. a typical propositional logic formula, a SAT solver can be employed to check its satisfiability automatically.

Example 2. Satisfiability of PPTL formula $(p \wedge \Box \bigcirc p) ; q$.

We first present $p \wedge \Box \bigcirc p$ and q in normal form expressions:

$$\begin{aligned} p \wedge \Box \bigcirc p &\equiv p \wedge \bigwedge_{0\le j\in N_\omega} \bigcirc^j (\bigcirc p) \\ q &\equiv \bigvee_{0\le i\in N_0} q \wedge \bigcirc^i \varepsilon \vee q \wedge \bigcirc^\omega true \end{aligned}$$

There are no $e \in E_\varepsilon$ occuring in the normal form expression of $p \wedge \Box \bigcirc p$. Thus,

$$\begin{aligned} p \wedge \Box \bigcirc p ; q &\equiv p \wedge \bigwedge_{0 \le j \in N_\omega} \bigcirc^j (\bigcirc p) ; (\bigvee_{0 \le i \in N_0} q \wedge \bigcirc^i \varepsilon \vee q \wedge \bigcirc^\omega true) \\ &\equiv false \end{aligned}$$

So, PPTL formula $(p \wedge \Box \bigcirc p) ; q$ is unsatisfiable.

6 Conclusion

In this paper, we present normal form expressions and show that any PPTL formula can be represented as a normal form expression. When presented as a normal form expression, the underlying models of a PPTL formula is easy to be acquired that leads to a simple decision procedure for checking the satisfiability of PPTL formulas.

References

1. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-Time Temporal Logic. Journal of the ACM 49(5), 672–713 (2002)
2. Ben-Ari, M., Manna, Z., Pnueli, A.: The temporal logic of branching time. Acta Informatica 20, 207–226 (1983)
3. Chandra, A., Halpern, J., Meyer, A., Parikh, R.: Equations between regular terms and an application to process logic. In: Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing (STOC 1981), pp. 384–390 (1981)
4. Clarke, E.M., Emerson, E.A.: Design and synthesis of synchronization skeletons using branching time temporal logic. In: Kozen, D. (ed.) Logic of Programs 1981. LNCS, vol. 131, pp. 52–71. Springer, Heidelberg (1982)
5. Duan, Z.: An Extended Interval Temporal Logic and A Framing Technique for Temporal Logic Programming. PhD thesis. University of Newcastle Upon Tyne (May 1996)
6. Duan, Z., Koutny, M., Holt, C.: Projection in Temporal Logic Programming. In: Pfenning, F. (ed.) LPAR 1994. LNCS, vol. 822, pp. 333–344. Springer, Heidelberg (1994)
7. Duan, Z., Tian, C., Zhang, L.: A Decision Procedure for Propositional Projection Temporal Logic with Infinite Models. Acta Informatica 45(1), 43–78 (2008)
8. Clark, M., Gremberg, O., Peled, A.: Model Checking. The MIT Press (2000)
9. Pnueli, A.: The temporal logic of programs. In: Proceedings of the 18th IEEE Symposium on Foundations of Computer Science (FOCS 1977), pp. 46–57 (1977)
10. Finkbeiner, B., Schewe, S.: Coordination Logic. In: Dawar, A., Veith, H. (eds.) CSL 2010. LNCS, vol. 6247, pp. 305–319. Springer, Heidelberg (2010)
11. Halpern, J., Manna, Z., Moszkowski, B.: A hardware semantics based on temporal intervals. In: Díaz, J. (ed.) ICALP 1983. LNCS, vol. 154, pp. 278–291. Springer, Heidelberg (1983)
12. Emerson, E.A.: Temporal and Modal Logic, Computer Science Department. University of Texas at Austin, USA (1995)
13. Duan, Z.: Temporal Logic and Temporal Logic Programming. Science Press, Beijing (2006)
14. Moszkowski, B.: Executing temporal logic programs. Cambridge University Press (1986)
15. Duan, Z., Tian, C.: A practical decision procedure for propositional projection temporal logic with infinite models. Theoretical Computer Science (2014), doi:10.1016/j.tcs.2014.02.011