An Axiomatization for Cylinder Computation Model*

Nan Zhang, Zhenhua Duan**, and Cong Tian

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

Abstract. To model and verify multi-core parallel programs, the paper proposes an axiom system for Propositional Projection Temporal Logic with Cylinder Computation Model (CCM-PPTL). To do so, the syntax and semantics of CCM-PPTL are presented. Further, based on the logical laws of PPTL, some algebraic laws of sequence expressions and logical laws regarding CCM operators are proved. Moreover, the axiom system of CCM-PPTL is established by extending that of PPTL with some axioms and inference rules of CCM operators. In addition, the soundness and completeness of the system are proved.

Keywords: Axiom System, Multi-core, Parallel, Formal Method.

1 Introduction

With the rapid development of integrated circuits technology and the demand for higher performance, on-chip multi-core processors (CMP) have been brought into being. The reality of multi-core processor has made parallel programs pervasive. Creating a correct parallel program is not a straightforward process even for a considerable small system, because programmers are forced to consider that the program will always yield to a correct result no matter what order the instructions are executed in. To improve the reliability of parallel programs, formal verification is an important viable approach. Modeling multi-core parallel programs is a crucial step for formal verification of correctness and reliability of many core parallel programs.

Model checking and theorem proving are two key verification methods. With model checking, the system is often 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 or not $M \models P$. 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, model checking suffers from the state explosion problem [10]. Further, most of web applications are data-intensive which are not suitable to be verified by means of model checking since the treatment of the data usually leads to a huge, even infinite state space. Some successful model checking tools are SPIN [9], SMV [10] and so on. By contrast, theorem proving can

* 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. 71–83, 2014.

handle many complex structures abstractly without state space explosion but requires more human intervention and are often time-consuming. With theorem proving, both the system behavior and the desired property are specified as formulas, say S and P, in some appropriate 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. Some famous theorem prover are PVS [11], ACL2 [2], Coq [1], Isabella [12], HOL [8] and so on. Verification of multi-core parallel programs raises a great challenge for theorem proving since it requires that the logic for modeling multi-core systems and specifying the expected properties has a powerful expressiveness. However, the widely used Propositional Linear Temporal Logic (PLTL) and Computational Tree Logic (CTL) are not powerful enough. In fact, they are not full regular. Further, Quantified Linear time Temporal Logic (QLTL) [13], Extended Temporal Logic (ETL) [16] and Linear mu-Calculus [15] have a more powerful expressiveness of full regular language. However, these logics are not practical since they are not intuitive or too complicated. Propositional Projection Temporal Logic (PPTL) [3] allows us to specify ω full regular properties [14]. Further, a decision procedure [4,7] and a complete proof system for PPTL [6] have been established. A model checker based on SPIN [5] and a theorem prover based on PVS have also been developed. Cylinder Computation Model (CCM) [17] is a concurrent semantic model which is defined based on PPTL and has been implemented in the interpreter of MSVL, which is an executable subset of Projection Temporal Logic. CCM can be employed to model multi-core parallel programs since the sequence expressions in it have the nature of regular expressions. With CCM, the autonomy and parallelism of the processes occupying different cores on one chip can be described neatly and concisely. In [6], we have proposed an axiom system for PPTL, and proved its soundness and completeness. To specify and verify multi-core parallel programs in a uniform framework, this paper proposes an axiom system for CCM-PPTL which extends that of PPTL by including transformation rules for sequence expressions and axioms and inference rules on CCM operators. Furthermore, the soundness and completeness of the extended axiom system are also proved.

The paper is organized as follows. In the next section, the underlying logic PPTL and the semantic model CCM are reviewed, including their syntax, semantics and PPTL axiom system. Based on PPTL, CCM-PPTL is proposed in Section 3, including its syntax and semantics. In Section 3, we further give an axiom system for CCM-PPTL and prove the soundness and completeness of the system. Finally, conclusions are drawn in Section 4.

2 Preliminaries

2.1 Propositional Projection Temporal Logic

Our underlying logic is Propositional Projection Temporal Logic. The formula P of PPTL is given by the following grammar.

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

where $p \in Prop$, $P_i (1 \leq i \leq m)$ and P are well-formed PPTL formulas, and $\bigcirc$, prj and $prj\oplus$ (projection-plus) are primitive temporal operators. A formula is a state formula if it contains no temporal operators, otherwise it is a temporal formula.

We define a state s over $Prop$ to be a mapping from $Prop$ to B. $s[p]$ denotes the valuation of p at state s. An interval σ is a non-empty finite or infinite sequence of states. The length, $|\sigma|$, of σ is ω if σ is infinite, and the number of states minus 1 if σ is finite. We consider the set N_0 of non-negative integers and ω, $N_\omega = N_0 \cup \{\omega\}$ and extend the comparison operators, $=$, $<$, $\leq$, to N_ω by considering $\omega = \omega$, and for all $i \in N_0$, $i < \omega$. Furthermore, we define $\preceq$ as $\leq -\{(\omega, \omega)\}$. σ is often denoted by $\langle s_0, \ldots, s_{|\sigma|}\rangle$, where $s_{|\sigma|}$ is undefined if σ is infinite. With such a notation, $\sigma_{(i \ldots j)}$ $(0 \leq i \preceq j \leq |\sigma|)$ denotes the sub-interval $\langle s_i, \ldots, s_j\rangle$ and σ^i $(0 \leq i \leq |\sigma|)$ denotes the prefix interval $\langle s_0, \ldots, s_i\rangle$. The concatenation of a finite σ with another interval (or empty string) σ' is denoted by $\sigma \cdot \sigma'$ (not sharing any states). Let $\sigma = \langle s_0, s_1, \ldots, s_{|\sigma|}\rangle$ be an interval and $r_1, \ldots, r_h$ be integers $(h \geq 1)$ such that $0 \leq r_1 \leq r_2 \leq \ldots \leq r_h \preceq |\sigma|$. The projection of σ onto $r_1, \ldots, r_h$ is the interval (called projected interval) $\sigma \downarrow (r_1, \ldots, r_h) = \langle s_{t_1}, s_{t_2}, \ldots, s_{t_l}\rangle$ where $t_1, \ldots, t_l$ are obtained from $r_1, \ldots, r_h$ by deleting all duplicates. That is, $t_1, \ldots, t_l$ is the longest strictly increasing subsequence of $r_1, \ldots, r_h$. An interpretation is a triple $\mathcal{I} = (\sigma, k, j)$, where σ is an interval, k an integer, and j an integer or ω such that $0 \leq k \preceq j \leq |\sigma|$. We use the notation $(\sigma, k, j) \models P$ to indicate that some 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 as follows.

$\mathcal{I} \models p$ 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 exist integers $k = r_0 \leq \cdots \leq r_{m-1} \preceq r_m \leq j$; for all $1 \leq l \leq m$, $(\sigma, r_{l-1}, r_l) \models P_l$; $(\sigma', 0, |\sigma'|) \models Q$ for one of the following σ':
- $r_m < j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_m) \cdot \sigma_{(r_m+1..j)}$, or
- $r_m = j$ and $\sigma' = \sigma \downarrow (r_0, \ldots, r_h)$ for some $0 \leq h \leq m$.

$\mathcal{I} \models (P_1, \ldots, (P_u, \ldots, P_l)^{\oplus}, \ldots, P_m)\ prj\ Q$ iff one of following cases holds:
- $1 \leq u \leq l \leq m$ and there exists an integer $n \geq 1$ and $\mathcal{I} \models (P_1, \ldots, (P_u, \ldots, P_l)^{(n)}, \ldots, P_m)\ prj\ Q$, or
- $1 \leq u \leq l = m$, $j = \omega$ and there exist infinitely many integers $k = r_0 \leq r_1 \leq \cdots \leq r_n \preceq \omega$ and $\lim\limits_{n\to\infty} r_n = \omega$ such that for all $1 \leq x \leq u-1$, $(\sigma, r_{x-1}, r_x) \models P_x$, and $(\sigma, r_{u+t(l-u+1)+n-1}, r_{u+t(l-u+1)+n}) \models P_{u+n}$, for all $t \geq 0$ and $0 \leq n \leq l-u$, and $\sigma \downarrow (r_0, r_1, \ldots, r_h, \omega) \models Q$ for some $h \in N_\omega$.

The axiom system for CCM-PPTL presented later is based on that of PPTL, which has been proved to be sound and complete. For more detail, please refer to [6].

2.2 Cylinder Computation Model

In this section, Cylinder Computation Model (CCM) is reviewed [17], including its syntax and semantics which are based on sequence expressions. Then the logical laws on CCM are drawn. First, sequence expressions are defined as follows.

$$l ::= \emptyset \mid \epsilon \mid n \mid l_1, l_2 \mid l_1 \otimes l_2 \mid l^*$$

From the syntax of the sequence expression given above, we see that it is an analogue of regular expressions where $\emptyset$ denotes empty set, ϵ empty sequence expression and n any non-negative integer. The concatenation ($\cdot$), sum ($\otimes$) or Kleene closure ($*$) of any two sequence expressions is also a sequence expression. The semantics is also defined by a satisfaction relation, $\Vdash$, by means of interpretation $\mathcal{I} = (\sigma, k, j)$.

1. $\mathcal{I} \not\Vdash \emptyset$ for all $\mathcal{I}$.
2. $\mathcal{I} \Vdash \epsilon$ iff $j = k$.
3. $\mathcal{I} \Vdash n$ iff $j - k = n$.
4. $\mathcal{I} \Vdash l_1 \otimes l_2$ iff $\mathcal{I} \Vdash l_1$ or $\mathcal{I} \Vdash l_2$.
5. $\mathcal{I} \Vdash l_1, l_2$ iff there exists r, $k \leq r \preceq j$, such that $\mathcal{I}_1 = (\sigma, k, r) \Vdash l_1$ and $\mathcal{I}_2 = (\sigma, r, j) \Vdash l_2$.
6. $\mathcal{I} \Vdash l^*$ iff $j = k$ or there exist finitely many integers $k = r_0 \leq r_1 \leq \ldots \preceq r_n = j$ such that for all h, $1 \leq h \leq n$, $(\sigma, r_{h-1}, r_h) \Vdash l$.

In the semantics of sequence expressions, $\emptyset$ cannot be satisfied by any interpretation. The empty sequence expression ϵ is equivalent to the sequence expression 0. In order to avoid an excessive number of parentheses, the precedence rules are given from high to low: (1) $*$ (iteration); (2) $\cdot$ (concatenation); (3) $\otimes$ (selection). Some algebraic laws of sequence expressions are summarized in [17]. Then, the syntax of CCM is defined as follows.

$$CCM ::= P \texttt{ ov } (l) \mid CCM_1 \parallel CCM_2$$

where P is a PPTL formula, l a sequence expression and the parallel ($\parallel$) composition of any two CCM formulas is also a CCM formula. CCM operators " ov " and " $\parallel$ " are temporal. So all the CCM formulas are temporal formulas. With CCM formulas, the interpretation of P is controlled by the sequence expression l. The beginning and ending points generated by the interpretation of l make up of the coarse-grained interval of P. Therefore, to give the semantics of CCM formulas, it is necessary to define the set of ending point lists denoted by $S_l^{\mathcal{I}}$. First, some notations are defined. For any two strings $X \stackrel{\text{def}}{=} (x_1, \ldots, x_m)$ and $Y \stackrel{\text{def}}{=} (y_1, \ldots, y_n)$, the concatenation of X and Y is defined as: $X, Y \stackrel{\text{def}}{=} (x_1, \ldots, x_m), (y_1, \ldots, y_n) \stackrel{\text{def}}{=} (x_1, \ldots, x_m, y_1, \ldots, y_n)$. For any two sets S_1 and S_2, of strings , the concatenation of S_1 and S_2 is defined as: $S_1, S_2 \stackrel{\text{def}}{=} \{X, Y \mid X \in S_1 \text{ and } Y \in S_2\}$.

Definition 1 (set of ending point lists $S_l^{\mathcal{I}}$) Let $\mathcal{I}$ be an arbitrary interpretation (σ, i, k, j). $S_l^{\mathcal{I}}$ is inductively defined as follows:

1. $S_\emptyset^{\mathcal{I}} = \emptyset$.
2. If $\mathcal{I} \Vdash \epsilon$, then $S_\epsilon^{\mathcal{I}} = \{(k, j)\}$.
3. If $\mathcal{I} \Vdash n$, then $S_n^{\mathcal{I}} = \{(k, j)\}$.
4. If $\mathcal{I} \Vdash l_1, l_2$, then $S_{l_1, l_2}^{\mathcal{I}} = \left\{ \tau \middle| \begin{array}{l} \text{there exists } r, k \leq r \preceq j, \text{ such that } \mathcal{I}_1 = \\ (\sigma, k, r) \Vdash l_1 \text{ and } \mathcal{I}_2 = (\sigma, r, j) \Vdash l_2 \text{ and} \\ \tau \in S_{l_1}^{\mathcal{I}_1}, S_{l_2}^{\mathcal{I}_2} \end{array} \right\}$
5. If $\mathcal{I} \Vdash l_1 \otimes l_2$, then $S_{l_1 \otimes l_2}^{\mathcal{I}} = S_{l_1}^{\mathcal{I}} \bigcup S_{l_2}^{\mathcal{I}}$.

6. If $\mathcal{I} \Vdash l^*$, then $S^{\mathcal{I}}_{l^*} = S^{\mathcal{I}}_{\epsilon} \cup \left\{ \tau \middle| \begin{array}{l} \text{there exist finitely many integers } k = r_0 \leq r_1 \ldots \preceq \\ r_n = j \text{ such that for all } 1 \leq h \leq n, \mathcal{I}_h = \\ (\sigma, r_{h-1}, r_h) \Vdash l \text{ and } \tau \in S^{\mathcal{I}_1}_l, S^{\mathcal{I}_2}_l, \ldots, S^{\mathcal{I}_n}_l \end{array} \right\}$

Then, the semantics of Cylinder Computation Model is defined by a satisfaction relation $\models$ by means of the interpretation $\mathcal{I} = (\sigma, k, j)$.

$\mathcal{I} \models P$ ov (l) iff one of the following cases holds:
(a) $\mathcal{I} \Vdash l$ and there exists $(r_0, r_1, \ldots, r_n) \in S^{\mathcal{I}}_l$, $n \in N_0$ such that P is satisfied by $\sigma \downarrow (r_0, r_1, \ldots, r_h)$ for some $0 \leq h \leq n$;
(b) there exists r, $k \leq r \preceq j$ such that $\mathcal{I}_1 = (\sigma, k, r) \Vdash l$ and there exists $(r_0, r_1, \ldots, r_n) \in S^{\mathcal{I}_1}_l$, $n \in N_0$ and P is satisfied by $\sigma \downarrow (r_0, r_1, \ldots, r_n) \cdot \sigma_{(r_n+1..j)}$.
$\mathcal{I} \models CCM_1 \parallel CCM_2$ iff one of the following cases holds:
(a) $\mathcal{I} \models CCM_1$ and there exists r, $k \leq r \preceq j$, $(\sigma, k, r) \models CCM_2$
(b) $\mathcal{I} \models CCM_2$ and there exists r, $k \leq r \preceq j$, $(\sigma, k, r) \models CCM_1$

In fact, an element in $S^{\mathcal{I}}_l$ is a sequence of non-negative integers, and a sequence expression can be satisfied by an interpretation in more than one way. Each element in $S^{\mathcal{I}}_l$ records one particular way in which $\mathcal{I}$ satisfies l, containing all the beginning and ending points. The definition of $S^{\mathcal{I}}_l$ is necessary since the PPTL formula P in CCM is interpreted over a coarse-grained interval composed of the points from one of the elements in $S^{\mathcal{I}}_l$.

3 CCM-PPTL and Axiom System

To model and verify multi-core parallel programs, CCM is included in PPTL. Then an axiom system is proposed in this section. The syntax of CCM-PPTL is inductively defined as follows:

$$\begin{array}{rl} \beta ::= & p \mid \neg\beta \mid \bigcirc \beta \mid \beta_1 \vee \beta_2 \mid (\beta_1, \ldots, \beta_m)\ prj\ \beta \\ & \mid (\beta_1, \ldots, (\beta_u, \ldots, \beta_l)^{\oplus}, \ldots, \beta_m)\ prj\ \beta \mid CCM \end{array}$$

where p is an arbitrary atomic proposition; β, β_i are arbitrary CCM-PPTL formulas; CCM an arbitrary CCM formula defined in section 2. The semantics of CCM-PPTL is also defined as a satisfaction relation $\models$ by means of the interpretation $\mathcal{I} = (\sigma, k, j)$.

$\mathcal{I} \models p$ iff $s_k[p] = true$ for any atomic proposition p.
$\mathcal{I} \models \neg\beta$ iff $\mathcal{I} \not\models \beta$.
$\mathcal{I} \models \bigcirc\beta$ iff $(\sigma, k+1, j) \models \beta$.
$\mathcal{I} \models \beta_1 \vee \beta_2$ iff $\mathcal{I} \models \beta_1$ or $\mathcal{I} \models \beta_2$.
$\mathcal{I} \models (\beta_1, \ldots, \beta_m)\ prj\ \beta$ iff
there exist $k = r_0 \leq r_1 \leq \ldots \leq r_{m-1} \preceq r_m \leq j$ such that for all$1 \leq l \leq m$, $(\sigma, r_{l-1}, r_l) \models \beta_l$, and one of the following cases holds:
(a) $r_m < j$ and β is satisfied by $\sigma \downarrow (r_0, \ldots, r_m) \cdot \sigma_{(r_m+1..j)}$;
(b) $r_m = j$ and β is satisfied by $\sigma \downarrow (r_0, \ldots, r_h)$ for some $0 \leq h \leq m$.
$\mathcal{I} \models (\beta_1, \ldots, (\beta_u, \ldots, \beta_l)^{\oplus}, \ldots, \beta_m)\ prj\ \beta$ iff one of the following cases holds:
(a) $\mathcal{I} \models (\beta_1, \ldots, (\beta_u, \ldots, \beta_l)^{(n)}, \ldots, \beta_m)\ prj\ \beta$ for some $n \geq 1$ and $n \in N_0$;
(b) $l = m$ and $j = \omega$ and there exist infinitely many integers $k = r_0 \leq r_1 \leq \cdots$ and $\lim\limits_{x \to \infty} r_x = \omega$, such that

- $(\sigma, r_{x-1}, r_x) \models \beta_x$ for $1 \leq x \leq u-1$, and
- $(\sigma, r_{u+t(l-u+1)+n-1}, r_{u+t(l-u+1)+n}) \models \beta_{u+n}$, for $t \geq 0$ and $0 \leq n \leq l-u$, and
- β is satisfied by $\sigma \downarrow (r_0, r_1, \ldots, r_h)$ for some $h \in N_\omega$.

$\mathcal{I} \models CCM$ see the semantics of CCM in Section 2.

It should be noted that the formula P appearing in P ov (l) is a PPTL formula and doesn't contain the ov operator. That is why the syntax and semantics of PPTL and CCM-PPTL have to be separated to define. CCM is of a typical form of P_1 ov $(l_1) \parallel \cdots \parallel P_m$ ov (l_m) where each P_i is a PPTL formula and each l_i is a sequence expression. With this parallelism, 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. This computation model can be viewed as m processes that share one processor and each occupies an execution core cooperating to complete their tasks in a parallel way. Each process progresses in its own speed and communicates with each other at some global states which indicates the coordination among these processes. Sequence expression l_i is used to control and determine the execution points (states) of P_i. $\parallel$ is the main operator in CCM. Thus P_1 ov $(l_1) \parallel \cdots \parallel P_m$ ov (l_m) is endowned with the semantics of many-core parallel computing. For example, the interval satisfying CCM formula P_1 ov $(2,3,3,4) \parallel P_2$ ov $(3,5,3,6) \parallel P_3$ ov $(2,1,2,3,3,1,5)$ is given in Fig.1.

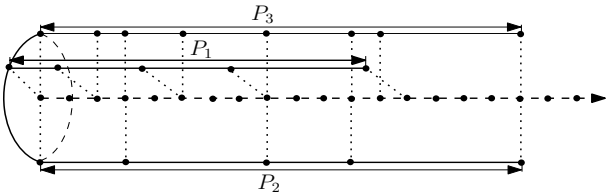

Fig. 1. P_1 ov $(2,3,3,4) \parallel P_2$ ov $(3,5,3,6) \parallel P_3$ ov $(2,1,2,3,3,1,5)$

All the logical laws in PPTL also hold in CCM-PPTL. In addition, we also prove some logical laws on CCM, for more details, refer to [3]. Some of these laws are chosen to be axioms later and are used to transform any CCM formula into its normal form. Now we introduce a normal form for CCM-PPTL formulas upon which the completeness proof of the axiom system is based.

Definition 2 (normal form of CCM-PPTL). A CCM-PPTL formula β is in normal form if it conforms to the following syntax: $\alpha_e \wedge \varepsilon \vee \bigvee_{i=1}^{r}(\alpha_i \wedge \bigcirc\beta_i)$, where $r \geq 1$, α_e and the α_i's are state formulas, whereas the β_i's are general CCM-PPTL formulas. Moreover, β is in complete normal form if $\vee_{i=1}^{r}\alpha_i \equiv true$ and $\vee_{i \neq j}(\alpha_i \wedge \alpha_j) \equiv false$.

Now, an axiom system for CCM-PPTL is formalized based on that of PPTL. Axioms and inference rules on CCM operators are included. Since the deduction of CCM formulas depends on the nature of sequence expressions, some essential transformation rules on sequence expressions need to be included in the proof system.

S1	$\epsilon \simeq \epsilon^* \simeq 0 \simeq 0^*$	S7	$l^*, l^* \simeq l^*$
S2	$0, l \simeq l, 0 \simeq l$	S8	$(l^*)^* \simeq l^*$
S3	$l_1, (l_2, l_3) \simeq (l_1, l_2), l_3 \simeq l_1, l_2, l_3$	S9	$l_1, (l_2, l_1)^* \simeq (l_1, l_2)^*, l_1$
S4	$l_1, (l_2 \otimes l_3), l_4 \simeq (l_1, l_2, l_4) \otimes (l_1, l_3, l_4)$	S10	$(0 \otimes l)^* \simeq l^*$
S5	$l^* \simeq \epsilon \otimes (l, l^*) \simeq (\epsilon \otimes l)^*$	S11	$l_1 \simeq l_2 \Longrightarrow l \simeq l[l_2/l_1]$
S6	$l, l^* \simeq l^*, l$	S12	$l \simeq (l_1, l) \otimes l_2 \Longrightarrow l \simeq l_1^*, l_2$

Then, the axioms on CCM formulas are given as follows based on the transformation rules of sequence expressions.

A1	$P \texttt{ ov } (l_1, \emptyset, l_2) \cong false$
A2	$P \texttt{ ov } (0) \cong P$
A3	$\varepsilon \texttt{ ov } (m, l) \cong \bigcirc(\varepsilon \texttt{ ov } (m-1, l))\ \ (m > 0)$
A4	$\bigcirc P \texttt{ ov } (m, l) \cong \bigcirc^m \varepsilon; (P \texttt{ ov } (l))\ \ (m > 0)$
A5	$(w \wedge P) \texttt{ ov } (l) \cong w \wedge (P \texttt{ ov } (l))$
A6	$P \texttt{ ov } (l_1 \otimes l_2) \cong (P \texttt{ ov } (l_1)) \vee (P \texttt{ ov } (l_2))$
A7	$(P_1 \vee P_2) \texttt{ ov } (l) \supset (P_1 \texttt{ ov } (l)) \vee (P_2 \texttt{ ov } (l))$
A8	$CCM_1 \parallel CCM_2 \cong (CCM_1; true) \wedge CCM_2 \vee (CCM_2; true) \wedge CCM_1$

The inference rules are presented in the following:

I1	$P \supset P' \Longrightarrow P \texttt{ ov } (l) \supset P' \texttt{ ov } (l)$
I2	$l_1 \simeq l_2 \Longrightarrow P \texttt{ ov } (l_1) \cong P \texttt{ ov } (l_2)$

Some explanations are needed. A1 means that any CCM formula P ov (l) with an unsatisfiable sequence expression l is also unsatisfiable. A2 means that CCM formula P ov (0) is deduced to the PPTL formula P. A3 means that if a sequence expression begins with a positive integer m and the PPTL formula is ε, we can extract one *next* operator directly with m decreasing by one. A4 means that if a sequence expression begins with a positive integer m and the major operator of the PPTL formula is the *next* operator ($\bigcirc$), we can extract m *next* operators directly with deleting m from the sequence expression and the next operator from $\bigcirc P$. A5 means that if the PPTL formula contains a conjunction being a state formula w, then w can be extracted from the PPTL formula and treated as a conjunction of the whole formula. A6 indicates the distributivity of the sum operator $\otimes$ over the ov operator. A7 describes the distributivity of disjunction over the ov operator. A8 presents the semantics of the parallel operator, that is, CCM_1 and CCM_2 are interpreted in parallel and may specify their own lengths. I1 means that the implication between any two PPTL formulas is preserved by the ov operator. I2 means that if l_1 is deduced to be the equivalent of l_2, the two CCM formulas with l_1 and l_2 being sequence expressions respectively are also deduced to be equivalent. Then the soundness and completeness of the axiom system of CCM-PPTL are demonstrated.

Theorem 1 (Soundness). *For any CCM-PPTL formula β, if $\vdash \beta$, then $\models \beta$.*

Proof. We need to prove that each axiom in the proof system of CCM-PPTL is valid in the model theory of CCM-PPTL and each inference rule preserves the validity of premises. Since the proof system of PPTL is sound, we only need to consider the axioms and inference rules on CCM operators. We can prove that each transformation rule of

sequence expressions is an algebraic law, each axiom on CCM is also a logical law. Two inference rules I1 and I2 are also easy to understand, which formalize the idea of substitution. All the above ensures the soundness of the axiom system of CCM-PPTL. Since the proof is not difficult, we omit it here.

To prove the completeness of the axiom system given in Theorem 3, ten lemmas are proved in advance. In general, the set of CCM-PPTL formulas are 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 β, if $\nvdash \beta \rightarrow false$, then β is satisfiable (Lemma 10). Lemma 10 is based on a fact that any CCM-PPTL formula can be deduced into a normal form which is proved in Theorem 2. The proof of Theorem 2 depends on Lemma 1-6. Because of space limitations, most of the details of proofs is omitted here.

Lemma 1. *For any CCM-PPTL formula β, if $\beta \cong \beta'$ where β' is in normal form, there exists a CCM-PPTL formula β_c in complete normal form satisfying $\beta \cong \beta_c$.*

Lemma 2. *Let $\alpha_1, \ldots, \alpha_n$ be state formulas, and β_i a general CCM-PPTL formula. If $\bigvee_{i=1}^{n} \alpha_i \cong true$ and $\bigvee_{i \neq j} \alpha_i \wedge \alpha_j \cong false$, then $\neg(\bigvee_{i=1}^{n} \alpha_i \wedge \beta_i) \cong \bigvee_{i=1}^{n}(\alpha_i \wedge \neg\beta_i)$.*

Lemma 1 indicates that any normal form can be deduced into a complete normal form. In the deduction of $\neg\beta$ into a normal form as we will see later on in Theorem 2, β is deduced into its normal form first, then further deduced into its complete normal form by means of Lemma 1. Finally, we deduce $\neg\beta$ into its normal form using Lemma 2 based on β's complete normal form.

Lemma 3. *If $\beta_i \cong \beta_i'$ $(0 \leq i \leq m)$, where β_i''s are CCM-PPTL formulas in normal form, then there exists a formula β in normal form such that $(\beta_1, \ldots, \beta_m)\ prj\ \beta_0 \cong \beta$.*

Lemma 4. *If $\beta_i \cong \beta_i'$ $(0 \leq i \leq m)$, where β_i''s are CCM-PPTL formulas in normal form, then (1) there exists a formula β in normal form such that*

$$((\beta_1, \ldots, \beta_l)^{\oplus}, \ldots, \beta_m)\ prj\ \beta_0 \cong \beta\ (1 \leq l \leq m);$$

(2) there exists a formula β in normal form such that

$$(\beta_1, \ldots, (\beta_i, \ldots, \beta_l)^{\oplus}, \ldots, \beta_m)\ prj\ \beta_0 \cong \beta\ (1 < i \leq l \leq m).$$

Lemma 3 and 4 show that the projection and projection plus operators can be deduced into normal forms. They are integral parts of the proof of Theorem 2. To deduce CCM formulas into their normal forms, first we need to deduce the sequence expression into one of the three following forms using transformation rules, which are formalized in the following lemma.

Lemma 5. *For any sequence expression l, l can be deduced using the transformation rules into one of the following forms:*
(Form 1) $\emptyset$
(Form 2) $0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i)$ where $m \geq 0$ and $n_i \in N$ for all $1 \leq i \leq m$.
(Form 3) $\bigotimes_{i=1}^{m}(n_i, l_i)$ where $m \geq 1$ and $n_i \in N$ for all $1 \leq i \leq m$.

Proof. The proof proceeds by induction on the structure of sequence expressions.
Base:
(1) l is $\emptyset$, then it is already in Form 1.
(2) l is ϵ, then $l \simeq 0$ according to the algebraic laws of sequence expressions, which is in Form 2 under the condition $m = 0$.
(3) l is n, if n is zero, it is the same as case (2); if n is a positive integer, according to the algebraic laws we have $n \simeq (n, 0)$, which is in Form 3 under the condition $m = 1$, $n_1 = n$ and $l_1 = 0$.
Induction:
(4) l is (l_1, l_2), with the hypothesis that both of l_1 and l_2 can be transformed into one of the three forms, then there are 3×3 possible combinations.
If l_1 or l_2 is transformed into $\emptyset$, which covers 5 possible combinations, l can be equivalently transformed into $\emptyset$ which is in Form 1.
If both of the transformations of l_1 and l_2 are in Form 2, l is transformed into Form 2.

$$
\begin{array}{rl}
l \simeq & (l_1, l_2) \\
\simeq & (0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i), 0 \otimes \bigotimes_{j=1}^{n}(n'_j, l'_j)) \qquad \text{Hypothesis} \\
\simeq & (0, 0 \otimes \bigotimes_{j=1}^{n}(n'_j, l'_j)) \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), 0 \otimes \bigotimes_{j=1}^{n}(n'_j, l'_j)) \qquad \text{S4} \\
\simeq & (0, 0) \otimes (0, \bigotimes_{j=1}^{n}(n'_j, l'_j)) \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), 0) \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), \bigotimes_{j=1}^{n}(n'_j, l'_j)) \qquad \text{S4} \\
\simeq & 0 \otimes \bigotimes_{j=1}^{n}(n'_j, l'_j) \otimes \bigotimes_{i=1}^{m}(n_i, l_i) \otimes \bigotimes_{i=1}^{m}\bigotimes_{j=1}^{n}(n_i, l_i, n'_j, l'_j) \qquad \text{S2, S4}
\end{array}
$$

If l_1 is in Form 2 and l_2 Form 3, then l is in Form 3.

$$
\begin{array}{rl}
l \simeq (l_1, l_2) \simeq & (0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i), \bigotimes_{j=1}^{n}(n'_j, l'_j)) \qquad \text{Hypothesis} \\
\simeq & (0, \bigotimes_{j=1}^{n}(n'_j, l'_j)) \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), \bigotimes_{j=1}^{n}(n'_j, l'_j)) \qquad \text{S4} \\
\simeq & \bigotimes_{j=1}^{n}(n'_j, l'_j) \otimes \bigotimes_{i=1}^{m}\bigotimes_{j=1}^{n}(n_i, l_i, n'_j, l'_j) \qquad \text{S2, S4}
\end{array}
$$

If l_1 is in Form 3 and l_2 Form 2, then l will be transformed into Form 3, and if both l_1 and l_2 are in Form 3, then l will be in Form 3. The proofs of these two cases are similar as the proof given above, so they are omitted here.
(5) l is $l_1 \otimes l_2$, with the hypothesis that l_1 and l_2 are transformed into l'_1 and l'_2, then $l'_1 \otimes l'_2$ is already in one of the three forms.
(6) l is $(l')^*$, using the transformation rule S5, we have $l \simeq \epsilon \otimes (l', (l')^*)$; then using S1, we have $l \simeq 0 \otimes (l', (l')^*)$. Suppose that l' has been transformed into l''. If $l'' \simeq \emptyset$, then $l = 0$ which is in Form 2. If $l'' \simeq 0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i)$, then l is in Form 2.

$$
\begin{array}{rl}
l \simeq & (l')^* \\
\simeq & (0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i))^* \qquad \text{Hypothesis} \\
\simeq & (\bigotimes_{i=1}^{m}(n_i, l_i))^* \qquad \text{S10} \\
\simeq & 0 \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), (\bigotimes_{i=1}^{m}(n_i, l_i))^*) \qquad \text{S5} \\
\simeq & 0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i, (\bigotimes_{j=1}^{m}(n_j, l_j))^*) \qquad \text{S4}
\end{array}
$$

If l'' is in Form 3, that is, $l'' \simeq \bigotimes_{i=1}^{m}(n_i, l_i)$, then l is in Form 2.

$$
\begin{array}{rl}
l \simeq & (l')^* \\
\simeq & 0 \otimes (l', (l')^*) \qquad \text{S5, S1} \\
\simeq & 0 \otimes (\bigotimes_{i=1}^{m}(n_i, l_i), (l')^*) \qquad \text{Hypothesis} \\
\simeq & 0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i, (l')^*) \qquad \text{S4}
\end{array}
$$

Lemma 6. *For any CCM formula β, there exists a formula β' in normal form such that $\beta \cong \beta'$.*

Proof. The proof proceeds by induction on the syntax of CCM.
Base: For $P \ \mathtt{ov}\ (l)$, by Lemma 5, if $l \simeq \emptyset$, then $P \ \mathtt{ov}\ (l) \cong false$ (A1).
If $l \simeq 0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i)$, by the theorem that any PPTL formula P can be deduced into its normal form, we have

$$
\begin{array}{ll}
& P \ \mathtt{ov}\ (l) \\
\cong & P \ \mathtt{ov}\ (0 \otimes \bigotimes_{i=1}^{m}(n_i, l_i)) & \text{I2} \\
\cong & P \ \mathtt{ov}\ (0) \vee \bigvee_{i=1}^{m} P \ \mathtt{ov}\ (n_i, l_i) & \text{A6} \\
\cong & P \vee \bigvee_{i=1}^{m} P \ \mathtt{ov}\ (n_i, l_i) & \text{A2} \\
\cong & P_e \wedge \varepsilon \vee \bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj}) \vee \bigvee_{i=1}^{m}(P_e \wedge \varepsilon \vee \bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj}))\ \mathtt{ov}\ (n_i, l_i) & \text{Hypothesis} \\
\cong & P_e \wedge \varepsilon \vee \bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj}) \vee \bigvee_{i=1}^{m}\bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj})\ \mathtt{ov}\ (n_i, l_i) & \text{A7} \\
\cong & P_e \wedge \varepsilon \vee \bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj}) \\
& \vee \bigvee_{i=1}^{m} P_e \wedge (\varepsilon \ \mathtt{ov}\ (n_i, l_i)) \vee \bigvee_{i=1}^{m}\bigvee_{j=1}^{n} P_{cj} \wedge (\bigcirc P'_{cj}\ \mathtt{ov}\ (n_i, l_i)) & \text{A5} \\
\cong & P_e \wedge \varepsilon \vee \bigvee_{j=1}^{n}(P_{cj} \wedge \bigcirc P'_{cj}) \\
& \vee \bigvee_{i=1}^{m} P_e \wedge \bigcirc(\varepsilon \ \mathtt{ov}\ (n_i - 1, l_i)) \vee \bigvee_{i=1}^{m}\bigvee_{j=1}^{n} P_{cj} \wedge \bigcirc^{n_i}(P'_{cj}\ \mathtt{ov}\ (l_i)) & \text{A3, A4}
\end{array}
$$

If $l \simeq \bigotimes_{i=1}^{m}(n_i, l_i)$, $P \ \mathtt{ov}\ (l)$ also can be deduced into its normal form in a similar way.
Induction: For $CCM_1 \parallel CCM_2$, suppose that CCM_1 and CCM_2 have been deduced into their normals, then

$$
\begin{array}{ll}
& CCM_1 \parallel CCM_2 \\
\cong & (CCM_1; true) \wedge CCM_2 \vee (CCM_2; true) \wedge CCM_1 & \text{A8} \\
\cong & ((\alpha_{1e} \wedge \varepsilon \vee \bigvee_{i=1}^{m} \alpha_{1i} \wedge \bigcirc\beta_{1i}); true) \wedge (\alpha_{2e} \wedge \varepsilon \vee \bigvee_{j=1}^{n} \alpha_{2j} \wedge \bigcirc\beta_{2j}) \\
& \vee((\alpha_{2e} \wedge \varepsilon \vee \bigvee_{j=1}^{n} \alpha_{2j} \wedge \bigcirc\beta_{2j}); true) \wedge (\alpha_{1e} \wedge \varepsilon \vee \bigvee_{i=1}^{m} \alpha_{1i} \wedge \bigcirc\beta_{1i}) & \text{Hypothesis} \\
\cong & ((\alpha_{1e} \wedge \varepsilon; true) \vee \bigvee_{i=1}^{m}(\alpha_{1i} \wedge \bigcirc\beta_{1i}; true)) \wedge (\alpha_{2e} \wedge \varepsilon \vee \bigvee_{j=1}^{n} \alpha_{2j} \wedge \bigcirc\beta_{2j}) \\
& \vee((\alpha_{2e} \wedge \varepsilon; true) \vee \bigvee_{j=1}^{n}(\alpha_{2j} \wedge \bigcirc\beta_{2j}; true)) \wedge (\alpha_{1e} \wedge \varepsilon \vee \bigvee_{i=1}^{m} \alpha_{1i} \wedge \bigcirc\beta_{1i}) & \text{PDF, PEB} \\
\cong & (\alpha_{1e} \wedge \varepsilon; true) \wedge (\alpha_{2e} \wedge \varepsilon) \\
& \vee \bigvee_{j=1}^{n}(\alpha_{1e} \wedge \varepsilon; true) \wedge (\alpha_{2j} \wedge \bigcirc\beta_{2j}) \vee \bigvee_{i=1}^{m}\bigvee_{j=1}^{n}(\alpha_{1i} \wedge \bigcirc\beta_{1i}; true) \wedge (\alpha_{2j} \wedge \bigcirc\beta_{2j}) \\
& \vee \bigvee_{i=1}(\alpha_{2e} \wedge \varepsilon; true) \wedge (\alpha_{1i} \wedge \bigcirc\beta_{1i}) \vee \bigvee_{j=1}^{n}\bigvee_{i=1}^{m}(\alpha_{2j} \wedge \bigcirc\beta_{2j}; true) \wedge (\alpha_{1i} \wedge \bigcirc\beta_{1i}) & \text{TAU} \\
\cong & \alpha_{1e} \wedge \alpha_{2e} \wedge \varepsilon \\
& \vee \bigvee_{j=1}^{n} \alpha_{1e} \wedge \alpha_{2j} \wedge \bigcirc\beta_{2j} \vee \bigvee_{i=1}^{m} \alpha_{2e} \wedge \alpha_{1i} \wedge \bigcirc\beta_{1i} \\
& \vee \bigvee_{i=1}^{m}\bigvee_{j=1}^{n} \alpha_{1i} \wedge \alpha_{2j} \wedge \bigcirc((\beta_{1i}; true) \wedge \beta_{2j} \vee (\beta_{2j}; true) \wedge \beta_{1i}) & \text{PSM, PEB}
\end{array}
$$

Lemma 6 tells us that any CCM formula can be deduced into a normal form after its sequence expression having been deduced into one of three forms, which is also an integral part of the proof of Theorem 2.

Theorem 2. *For any CCM-PPTL formula β, there exists a formula β' in normal form such that $\beta \cong \beta'$.*

Proof. The proof proceeds by induction on the syntax of CCM-PPTL.
Base:
(1) For any atomic proposition p, $p \cong p \wedge \varepsilon \vee p \wedge \bigcirc true$.
(2) For $\bigcirc\beta$, $\bigcirc\beta \cong true \wedge \bigcirc\beta$.
Induction:
(3) For $\neg\beta$, suppose that β can be deduced into its normal form, from Lemma 1, β also can be deduced into its complete normal form $\alpha_e \wedge \varepsilon \vee \vee_{i=1}^{r}(\alpha_i \wedge \bigcirc\beta_i)$ where $\vee_{i=1}^{r}\alpha_i \cong true$ and $\vee_{i\neq j}\alpha_i \wedge \alpha_j \cong false$. Then we have

$$\neg\beta \cong \neg(\alpha_e \wedge \varepsilon \vee \vee_{i=1}^{r}(\alpha_i \wedge \bigcirc\beta_i)) \cong \neg\alpha_e \wedge \varepsilon \vee \vee_{i=1}^{r}(\alpha_i \wedge \bigcirc\neg\beta_i)$$

(4) For $\beta_1 \vee \beta_2$, suppose β_1 and β_2 have been transformed into their normal form, then

$$\begin{aligned}\beta_1 \vee \beta_2 &\cong \alpha_{1e} \wedge \varepsilon \vee \vee_{i=1}^{m}\alpha_{1i} \wedge \bigcirc\beta_{1i} \vee \alpha_{2e} \wedge \varepsilon \vee \vee_{j=1}^{n}\alpha_{2j} \wedge \bigcirc\beta_{2j} \\ &\cong (\alpha_{1e} \vee \alpha_{2e}) \wedge \varepsilon \vee \vee_{i=1}^{m}\alpha_{1i} \wedge \bigcirc\beta_{1i} \vee \vee_{j=1}^{n}\alpha_{2j} \wedge \bigcirc\beta_{2j}\end{aligned}$$

(5) For $(\beta_1, \ldots, \beta_m)\ prj\ \beta_0$, refer to the proof of Lemma 3.
(6) For $(\beta_1, \ldots, (\beta_i, \ldots, \beta_l)^{\oplus}, \ldots, \beta_m)\ prj\ \beta_0$, refer to the proof of Lemma 4.
(7) For CCM, refer to the proof of Lemma 6.

Definition 3 (terminable formula and non-terminable formula) For any CCM-PPTL formula β, if $\beta \wedge \diamond\varepsilon \not\equiv false$, then β is a terminable formula. Otherwise, it is a non-terminable formula.

We can easily prove that any terminable CCM-PPTL formula β, β is satisfiable. Since β is terminable, by Definition 3, $\beta \wedge \diamond\varepsilon \not\equiv false$, which means that there exists a model σ satisfies $\beta \wedge \diamond\varepsilon$. Then σ is also a model of β and so β is satisfiable. Then we derive the following conclusion.

Lemma 7. *For any terminable CCM-PPTL formula β, if $\nvdash \beta \rightarrow false$, then β is satisfiable.*

We can prove by contradiction that for any CCM-PPTL formulas β and β', if $\beta \equiv \beta'$ where β is non-terminable and β' in normal form, then β' is of the form $\bigvee_{i=1}^{n}\alpha_i \wedge \bigcirc\beta_i$ with each α_i being a state formula and each β_i being non-terminable. From hypothesis, we can derive a contradiction to the premise that β is non-terminable. Then we have the following conclusion.

Lemma 8. *For any CCM-PPTL formula β, if $\beta \equiv \beta'$ where β' is in normal form of $\alpha_e \wedge \varepsilon \vee \bigvee_{i=1}^{n}\alpha_i \wedge \bigcirc\beta_i$, then*
(1) If $\alpha_e \not\equiv false$, then β is terminable.
(2) If there exists some β_i being terminable, then β is terminable.

With Lemma 8 which is a conclusion on model theory of CCM-PPTL, we can derive a similar conclusion on the axiom system by using contradiction proof.

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

Lemma 9 means that if a non-terminable formula β has been deduced into its normal form β' using the axiom system, then we can infer that there is no terminal product $\alpha_e \wedge \varepsilon$ in β', and each future product in normal form be also non-terminable. Otherwise, there will be a contradiction to the premise that β is non-terminable.

Lemma 10. *For any non-terminable CCM-PPTL formula β, if $\nvdash \beta \rightarrow false$, then β is satisfiable.*

The proof of Lemma 10 is with intricacy. It involves constructing an interval for β and then prove the interval is indeed a model of β. Two famous theorems on fix-point are

used in the proof, one is Taski's fix-point theorem and the other is Scott's fix-point induction. The proof is omitted here. From Lemma 7 and Lemma 10, we can derive that any CCM-PPTL formula β, no matter whether it is terminable or non-terminable, if $\nvdash \beta \rightarrow false$, then β is satisfiable. Then we have the following corollary.

Corollary 1. *For any CCM-PPTL formula β, if β is unsatisfiable, then $\vdash \beta \rightarrow false$.*

Theorem 3 (Completeness). *For any CCM-PPTL formula β, if $\models \beta$, then $\vdash \beta$.*

Proof. From the premise of β is valid, we can derive the duality that $\neg\beta$ is unsatisfiable. By Corollary 1, we have $\neg\beta \rightarrow false$ is a theorem in the proof system of CCM-PPTL, which means that β is a theorem in the proof system.

4 Conclusion

We introduce a Cylinder Computation Model into Propositional Projection Temporal Logic and propose an axiom system for CCM-PPTL, which can be employed to model and verify many-core computation systems. In the future, we need to do some further case studies for more complex many-core computation. Further, to provide a highly automatical verification approach, the existing tool for PPTL theorem proving will be extended to support CCM operators. Moreover, we will explore the verification methodology which combines model checking and theorem proving.

References

1. Bertot, Y., Castéran, P.: Interactive Theorem Proving and Program Development, Heidelberg (2004)
2. Brock, B., Kaufmann, M., Moore, J.: ACL2 theorems about commercial micro-processors. In: Srivas, M., Camilleri, A. (eds.) FMCAD 1996. LNCS, vol. 1166, pp. 275–293. Springer, Heidelberg (1996)
3. Duan, Z.: Temporal Logic and Temporal Logic Programming. Science Press, Beijing (2006)
4. Duan, Z., Tian, C., Zhang, L.: A decision procedure for propositional projection temporal logic with infinite models. Acta Informatica 45, 43–78 (2008)
5. Duan, Z., Tian, C.: A unified model checking approach with projection temporal logic. In: Liu, S., Araki, K., Maibaum, T. (eds.) ICFEM 2008. LNCS, vol. 5256, pp. 167–186. Springer, Heidelberg (2008)
6. Duan, Z., Zhang, N., Koutny, M.: A complete proof system for propositional projection temporal logic. Theoretical Computer Science 497, 84–107 (2013)
7. 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
8. Gordon, M., Melham, T.: Introduction to HOL: A Theorem Proving Environment for Higher Order Logic. Cambridge University Press (1993)
9. Holzmann, G.: The model checker SPIN. IEEE Trans. Softw. Eng. 23(5), 279–295 (1997)
10. McMillan, K.: Symbolic Model Checking: An Approach to the State Explosion Problem, Dordrecht (1993)
11. Owre, S., Rushby, J., Shankar, N.: PVS: A prototype verification system. In: Kapur, D. (ed.) CADE 1992. LNCS (LNAI), vol. 607, pp. 748–752. Springer, Heidelberg (1992)
12. Paulson, L.C.: Isabelle. LNCS, vol. 828. Springer, Heidelberg (1994)

13. Sistla, A.: Theoretical issues in the design and verification of distributed systems, Ph.D. Thesis. Harvard University (1983)
14. Tian, C., Duan, Z.: Expressiveness of propositional projection temporal logic with star. Theoretical Computer Science 412(18), 1729–1744 (2011)
15. Vardi, M.: A temporal fixpoint calculus. In: POPL 1988, pp. 250–259 (1988)
16. Wolper, P.: Temporal logic can be more expressive. Information and Control 56, 72–99 (1983)
17. Zhang, N., Duan, Z., Tian, C.: A cylinder computation model for many-core parallel computing. Theoretical Computer Science 497, 68–83 (2013)