Weak Bounded Semantics and Bounded Verification of LTL Formulas *

Wenhui Zhang

Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences, Beijing, China
zwh@ios.ac.cn

Abstract. Bounded model checking based on SAT has been introduced as a complementary method to binary decision diagram based symbolic model checking in 1999 [2, 3]. Since then, there has been a lot of research on the improvement, extension and application along this direction. For general LTL formulas, bounded model checking has traditionally aimed at error detection, taking the advantage that error detection may only need to explore a small portion of the whole state space. The problem of verification is that it looks difficult to reason about all involved paths of a model, since the number of such paths is necessarily big. An approach to verification of LTL formulas based on bounded model checking principles has been reported in [15]. This paper presents a weak bounded semantics as a theoretical basis for the verification approach and as a theoretical consideration, we define a canonical representation for LTL formulas, and proves for a subset of LTL that the verification approach is complete.

1 Introduction

Model checking [7, 9, 8] has been successfully used in the last decades for the verification of finite state systems, and it is considered as one of the most practical applications of theoretical research in the verification of concurrent systems. The practical applicability of model checking is however limited by the state explosion problem which could be caused by for instance, the representation of currency of operations by their interleaving. Then much effort has been put into the combating of this problem and many techniques have been developed, e.g., [5, 6].

One of the techniques developed for combating the state explosion problem is bounded model checking (BMC) based on satisfiability testing (SAT) [2–4]. The basic idea is to search for a counter example of a particular length and to generate a propositional formula that is satisfied iff such a counter example exists. The efficiency of this method is based on the observation that if a system is faulty then only a fragment of its state space is sufficient for finding an error.

* Supported by the National Natural Science Foundation of China under Grant No. 60573012 and 60721061, and the National Grand Fundamental Research 973 Program of China under Grant No. 2002cb312200.

Given a finite transition system M, an LTL formula φ and a natural number k, a BMC procedure decides whether there exists a computation in M of length k or less that violates φ. If we have given M and φ such that M satisfies φ, then the practical value of this approach depends on the existence of a relatively small value of the completeness threshold. As stated in [11], knowing the completeness threshold is essential for making BMC complete for practical applications. Without it, there is no way of knowing whether the property holds or rather the bound is not sufficiently high. Even if we know the completeness threshold, for a reasonably large system, this threshold would possibly be large enough to make the verification become intractable due to the complexity of solving the corresponding SAT instance. For attacking this problem, many techniques have been developed, for instance, for properties such as simple safety and liveness properties [14, 12, 10, 1], in particular, a technique was developed to partly avoid this problem for general LTL properties such that the approach may certify that the property holds without knowing a completeness threshold [15].

In this paper, for the first, we present a weak bounded semantics as a theoretical basis for the verification approach presented in [15]. For the second, since the verification approach based on this weak bounded semantics is not a complete approach, we define a canonical representation for LTL formulas, and proves for a subset of LTL that the verification approach is complete.

2 Propositional Linear Temporal Logic

Propositional linear temporal logic (LTL) is a logic introduced by Pnueli as a specification language for concurrent programs [13]. Let AP be a set of proposition symbols. The set of LTL formulas is defined as follows:

- Every member of AP is an LTL formula.
- Logical connectives of LTL include: $\neg$, $\wedge$, $\vee$, and $\rightarrow$.
  If φ and ψ are LTL formulas, then so are $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, and $\varphi \rightarrow \psi$.
- Temporal operators include: X, F, G, U, and R.
  If φ and ψ are LTL formulas, then so are: $X\,\varphi$, $F\,\varphi$, $G\,\varphi$, $\varphi\,U\,\psi$, and $\varphi\,R\,\psi$.

2.1 Semantics of LTL

The formal semantics of LTL is defined with respect to paths of a Kripke structure. Let $M = \langle S, T, I, L\rangle$ be a Kripke structure where S is a set of states, $T \subseteq S \times S$ is a transition relation which is total, $I \subseteq S$ is a set of initial states and $L : S \rightarrow 2^{AP}$ is a labeling function. Let φ be an LTL formula. Let $\pi = \pi_0\pi_1 \cdots$ be a path of M and π^i be the subpath of π starting at π_i. We define the relation "φ holds on π", denoted $\pi \models \varphi$, as follows.

$\pi \models p$	iff $p \in L(\pi_0)$.
$\pi \models \neg\varphi$	iff $\pi \not\models \varphi$
$\pi \models \varphi \wedge \psi$	iff $\pi \models \varphi$ and $\pi \models \psi$
$\pi \models \varphi \vee \psi$	iff $\pi \models \varphi$ or $\pi \models \psi$
$\pi \models \varphi \rightarrow \psi$	iff $\pi \models \varphi$ implies $\pi \models \psi$
$\pi \models X\varphi$	iff $\pi^1 \models \varphi$

$$
\begin{array}{l}
\hline
\pi \models F\varphi \quad \text{iff } \exists k \geq 0.\pi^k \models \varphi \\
\pi \models G\varphi \quad \text{iff } \forall k \geq 0.\pi^k \models \varphi \\
\pi \models \varphi U \psi \text{ iff } \exists k \geq 0.\forall j < k.(\pi^k \models \psi \wedge \pi^j \models \varphi) \\
\pi \models \varphi R \psi \text{ iff } \forall j \geq 0.(\pi^j \models \psi) \vee \exists k \geq 0.((\pi^k \models \varphi) \wedge (\forall j \leq k.(\pi^j \models \psi)) \\
\hline
\end{array}
$$

For simplicity, we call a Kripke structure a model. An LTL formula φ is *true* in the model M, denoted $M \models \varphi$, iff φ is *true* on all paths starting from an arbitrary initial state of M.

2.2 Negation Normal Form

An LTL formula is in negation normal form (NNF), if the symbol $\rightarrow$ does not appear in the formula and $\neg$ is applied only to proposition symbols. We call proposition symbols and the negation of proposition symbols literals. In another words, NNF formulas are constructed from literals with $\vee$, $\wedge$, X, G, F, U, and R. Every formula can be transformed into an equivalent formula in NNF by using the following rules:

$$
\begin{array}{|ll|}
\hline
\varphi \rightarrow \psi & = \neg\varphi \vee \psi \\
\neg\neg\varphi & = \varphi \\
\neg(\varphi \vee \psi) & = (\neg\varphi \wedge \neg\psi) \\
\neg(\varphi \wedge \psi) & = (\neg\varphi \vee \neg\psi) \\
\hline
\neg X\varphi & = X\neg\varphi \\
\neg F\varphi & = G\neg\varphi \\
\neg(\varphi U \psi) & = \neg\varphi R \neg\psi \\
\neg G\varphi & = F\neg\varphi \\
\neg(\varphi R \psi) & = \neg\varphi U \neg\psi \\
\hline
\end{array}
$$

For simplicity, in the rest of this paper, we required every formula to be an NNF formula. A formula not in NNF is considered as an abbreviation of such a formula. By defining *true* to be $p \vee \neg p$ for a given proposition symbol p, we have the following equivalences

$$
\begin{array}{l}
F\varphi \quad = \mathit{true}\, U \varphi \\
\varphi R \psi = (\psi U (\varphi \wedge \psi)) \vee G\psi
\end{array}
$$

Without loss of generality, we only consider NNF formulas constructed from literals with $\vee$, $\wedge$, X, G, and U, since the operators F and R are definable.

2.3 Bounded Semantics of LTL Formulas

Let $M = \langle S, T, I, L \rangle$ be a model and $k \in \mathbf{N}$ a natural number. We call a finite path $\pi = \pi_0 \cdots \pi_k$ a k-path. A k-path $\pi = \pi_0 \cdots \pi_k$ is a (k,l)-loop, if $\pi' = (\pi_0 \cdots \pi_k)(\pi_l \cdots \pi_k)^\omega$ is an infinite path of M. A k-path $\pi = \pi_0 \cdots \pi_k$ is a k-loop, if it is a (k,l)-loop for some $0 \leq l \leq k$. The following definition of bounded semantics is according to [2] with modifications such that the relation is defined on finite paths instead of infinite ones.

Definition 1 (Bounded Semantics for a Loop). *Let $k \geq 0$ and π be a k-loop. Then an LTL formula φ is true on π, written $\pi \models_k \varphi$, iff $\pi' \models \varphi$ with $\pi' = (\pi_0 \cdots \pi_k)(\pi_l \cdots \pi_k)^\omega$ for some $0 \leq l \leq k$.*

Definition 2 (Bounded Semantics without a Loop). *Let $k \geq 0$ and π be a k-path which is not a k-loop. Then an LTL formula φ is true on π, written $\pi \models_k \varphi$, iff $\pi \models_k^0 \varphi$ where:*

$\pi \models_k^i p$	iff $p \in L(\pi_i)$
$\pi \models_k^i \neg p$	iff $\pi \not\models_k^i p$
$\pi \models_k^i \varphi \wedge \psi$	iff $\pi \models_k^i \varphi$ *and* $\pi \models_k^i \psi$
$\pi \models_k^i \varphi \vee \psi$	iff $\pi \models_k^i \varphi$ *or* $\pi \models_k^i \psi$
$\pi \models_k^i X\varphi$	iff $i < k$ *and* $\pi \models_k^{i+1} \varphi$
$\pi \models_k^i G\varphi$	iff *false*.
$\pi \models_k^i \varphi U \psi$	iff $\exists j \in \{i, ..., k\}.\ \forall n \in \{i, ..., j-1\}.(\pi \models_k^j \psi \wedge \pi \models_k^n \varphi)$

Note that $\pi \models_k^i G\varphi$ is *false* by definition if the k-path is not a k-loop. This is explained by that a global property can only be witnessed by an infinite path (or a path with a loop).

Let φ be an LTL formula. Generally, if there is an infinite path π of M such that $\pi \models \varphi$, there is a $\pi' = (\pi_0 \cdots \pi_k)(\pi_l \cdots \pi_k)^\omega$ such that $\pi' \models \varphi$. Therefore if there is an infinite path π of M such that $\pi \models \varphi$, then there is a k-path π' such that $\pi' \models_k \varphi$. On the other hand, if there is a k-path π such that $\pi \models_k \varphi$, then there is an infinite path π' of M such that $\pi' \models \varphi$. Therefore, $M \not\models \varphi$ iff there is a path π and a $k \geq 0$ such that $\pi \models_k \neg\varphi$.

The principle of this bounded semantics is based on showing the existence of a witness (a path on which a formula is true) with respect to k-paths. If $\pi \models_k \varphi$, then we have a k-path π such that an infinite path π' can be constructed from the k-path such that $\pi' \models \varphi$. This is sufficient as a counter example for $M \models \neg\varphi$ [2].

2.4 Weak Bounded Semantics of LTL

The principle of the weak bounded semantics is to show the existence of a bounded model with certain property contradicts the existence of witness.

Definition 3 (Weak Bounded Semantics). *Let $k \geq 0$ and $\pi = \pi_0 \cdots \pi_k$ be a k-path. Then an LTL formula φ is true with respect to the weak bounded semantics on π, written $\pi \models_e \varphi$, iff $\pi \models_e^0 \varphi$ where:*

$\pi \models_e^i p$	iff $p \in L(\pi_i)$
$\pi \models_e^i \neg p$	iff $\pi \not\models_e^i p$
$\pi \models_e^i \varphi \wedge \psi$	iff $\pi \models_e^i \varphi$ *and* $\pi \models_e^i \psi$
$\pi \models_e^i \varphi \vee \psi$	iff $\pi \models_e^i \varphi$ *or* $\pi \models_e^i \psi$
$\pi \models_e^i X\varphi$	iff $i = k$ *or* $\pi \models_e^{i+1} \varphi$
$\pi \models_e^i G\varphi$	iff $\forall n \in \{i, ..., k\}.(\pi \models_e^n \varphi)$
$\pi \models_e^i \varphi U \psi$	iff $\exists j \in \{i, ..., k\}.$ $\forall n \in \{i, ..., j-1\}.(\pi \models_e^j \psi \wedge \pi \models_e^n \varphi) \vee \forall n \in \{i, ..., k\}.(\pi \models_e^n \varphi)$

The weak bounded semantics is weaker than the bounded semantics. This can be seen from their definitions.

Proposition 1. *Let π be a k-path and φ an LTL formula. If $\pi \models_k \varphi$, then $\pi \models_e \varphi$.*

Proof. We consider two cases: π is a k-loop and π is not a k-loop. For the first case, let π' be $\pi\pi''$ where π'' is the loop part of the k-loop, we prove that

$$\text{if } \pi'^i \models \varphi, \text{ then } \pi \models_e^i \varphi \text{ for all } 0 \leq i \leq k.$$

This can be seen using the structural induction. We have that the definitions of $\pi'^i \models \varphi$ and $\pi \models_e^i \varphi$ are the same for φ being a literal, and the defining term for $\pi'^i \models \varphi$ is at least equally strong as that of $\pi \models_e^i \varphi$ for composed formulas. The second case is similar, by comparing the definitions and proving that

$$\text{if } \pi \models_k^i \varphi, \text{ then } \pi \models_e^i \varphi \text{ for all } 0 \leq i \leq k.$$

For convenience, for an empty string ϵ, we define $\epsilon \models_e \varphi$ to be true for φ. According to the definition, we obtain that, for any φ, a k-path π can be extended to an infinite path π' such that $\pi' \models \varphi$ only if $\pi \models_e \varphi$. In another words, we have the following proposition.

Proposition 2. *Let π be an infinite path and φ an LTL formula. If there is some finite prefix π' of π such that $\pi' \not\models_e \varphi$, then $\pi \models \neg\varphi$.*

Proof. Let $k \geq 0$ be arbitrarily given and $\pi' = \pi_0\pi_1 \cdots \pi_k$. We reformulate the problem in the other direction and prove that if $\pi \models \varphi$, then $\pi' \models_e \varphi$. Then, similar to the proof of Proposition 1, we only need to prove

$$\text{if } \pi^i \models \varphi, \text{ then } \pi' \models_e^i \varphi.$$

We have that the definitions of $\pi^i \models \varphi$ and $\pi' \models_e^i \varphi$ are the same for φ being a literal, and the defining term for $\pi^i \models \varphi$ is at least equally strong as that of $\pi' \models_e^i \varphi$ for composed formulas. Therefore the statement holds by structural induction.

The relation between finite paths and their prefix can be established as follows: for any φ, a k-path π can be extended to an $(k+1)$-path π' such that $\pi' \models \varphi$ only if $\pi \models_e \varphi$.

Proposition 3. *Let π be a k-path and φ an LTL formula. If there is some prefix π' of π such that $\pi' \not\models_e \varphi$, then $\pi \not\models_e \varphi$.*

Proof. Let $\pi = \pi_0\pi_1 \cdots \pi_k$ and $\pi' = \pi_0\pi_1 \cdots \pi_l$ for $l \leq k$. We prove that

$$\text{if } \pi \models_e^i \varphi, \text{ then } \pi' \models_e^i \varphi \text{ for all } 0 \leq i \leq l.$$

The reasoning is similar to the that of Proposition 2 and the statement holds by structural induction.

Proposition 3 guarantees that if $\pi \not\models_e \varphi$, then there is a π' such that for all π' which is a proper prefix of π, $\pi' \models_e \varphi$, and for all π'' of which π is a prefix, $\pi'' \not\models_e \varphi$.

Let Π_k be a set of k-paths.

Let $\Pi_k \models_e \varphi$ denote that for all $\pi \in \Pi_k$, $\pi \models_e \varphi$ and $\Pi_k \#_e \varphi$ denote that for all $\pi \in \Pi_k$, $\pi \not\models_e \varphi$.

Let M_k denote the set of k-path of M where each such k-path starts from some initial state of M.

Theorem 1. *Let M be a model, φ an LTL formula. $M \models \neg\varphi$ if there is a k such that $M_k \#_e \varphi$.*

Proof. By Proposition 2, if there is a k and for each $\pi \in M_k$, $\pi \not\models_e \varphi$, then no infinite extension of a path of M_k satisfies φ, therefore every path starting from some initial state of M satisfies $\neg\varphi$.

3 Encoding of the Problem in SAT-Formulas

A SAT-based encoding of LTL model checking problem has been given in [15]. In this section, we relate this encoding to the weak bounded semantics. Given a model M, an LTL formula φ and a bound k. The problem considered here is $M \models_k \varphi$. we will construct encodings for the pair (M, φ) for each given k. Let $u_0, ..., u_k$ be a finite sequence of state variables. We first define $[[M]]_k$ to be a formula representing that $u_0 \cdots u_k$ is a finite prefix of a valid path of M starting from an arbitrary initial state.

Definition 4 (Transition Relation). *Let $M = \langle S, T, I, L\rangle$ be a model and $k \geq 0$.*

$$[[M]]_k := I(u_0) \wedge \bigwedge_{i=0}^{k-1} T(u_i, u_{i+1})$$

This translation of transition relation corresponds to that in [2]. In the following, we fixed the model under consideration to be $M = \langle S, T, I, L\rangle$ unless otherwise is stated.

3.1 Encoding of LTL formulas

Let $p \in AP$ be a proposition symbol and $p(u)$ represent the propositional formula representing the states in which p is *true* according to L. For a state and a formula, we present the encoding for (formula,state) pair according to the weak bounded semantics of LTL.

Definition 5 (Translation of LTL formulas). *Let $u_0, ..., u_k$ be state variables and φ be a formula. The encoding $[[\varphi, u_i]]_k$ is defined as follows.*

$$
\begin{array}{ll}
[[p, u_i]]_k & = p(u_i) \\
[[\neg p, u_i]]_k & = \neg p(u_i) \\
[[\varphi \vee \psi, u_i]]_k & = [[\varphi, u_i]]_k \vee [[\psi, u_i]]_k \\
[[\varphi \wedge \psi, u_i]]_k & = [[\varphi, u_i]]_k \wedge [[\psi, u_i]]_k \\
\hline
[[X\varphi, u_i]]_k & = [[\varphi, u_{i+1}]]_k \\
[[G\varphi, u_i]]_k & = \bigwedge_{j=i}^{k} [[\varphi, u_j]]_k \\
[[\varphi U \psi, u_i]]_k & = \bigvee_{j=i}^{k} ([[\psi, u_j]]_k \wedge \bigwedge_{t=i}^{j-1} [[\varphi, u_t]]_k) \vee \bigwedge_{t=i}^{k} [[\varphi, u_t]]_k
\end{array}
$$

where $[[\varphi, u_{k+1}]]_k = \mathit{true}$.

The term $[[\varphi, u_{k+1}]]_k$ is only a representation of *true* for convenience of writing the definition. u_{k+1} needs not be interpreted to be a corresponding state of M. Given that an interpretation α of $u_0, ..., u_k$ corresponds to a path $\alpha(u_0) \cdots \alpha(u_k)$ of M, the definition of $[[\varphi, u_0]]_k$ captures the meaning of $\alpha(u_0) \cdots \alpha(u_k) \models_e \varphi$.

Lemma 1. *Let α be an assignment of $\{u_0, ..., u_k\}$. $\alpha(u_0) \cdots \alpha(u_k) \models_e \varphi$ iff $[[\varphi, u_0]]_k$.*

Proof. Let $\pi = \alpha(u_0) \cdots \alpha(u_k)$. We prove that

$$\pi \models_e^i \varphi \text{ iff } [[\varphi, u_0]]_k \text{ is true under the assignment } \alpha.$$

By comparing the definitions, it is easily seen that this holds by structural induction.

Definition 6. $[[M, \varphi]]_k := [[M]]_k \wedge [[\varphi, u_0]]_k$

This definition combines the definition of $[[\varphi, u_0]]_k$ with path information. Every assignment α satisfying $[[M, \varphi]]_k$ makes $\alpha(u_0) \cdots \alpha(u_k)$ a valid path of M starting from some initial state of M, in addition to that $\alpha(u_0) \cdots \alpha(u_k) \models_e \varphi$.

Theorem 2. *Let M be a model, φ an LTL formula. $M_k \not\models_e \varphi$ iff $[[M, \varphi]]_k$ is unsatisfiable.*

Proof. Let α be an assignment of $\{u_0, ..., u_k\}$ that maps them to $k+1$ states (not necessary all different) of M that satisfies $[[M, \varphi]]_k$. Then $\alpha(u_0) \cdots \alpha(u_k)$ is a k-path of M_k (i.e. a valid path of length $k+1$ of M starting from some initial state of M), since α satisfies $[[M]]_k$. Then $\alpha(u_0) \cdots \alpha(u_k) \models_e \varphi$ according to Lemma 1, since α also satisfies $[[\varphi, u_0]]_k$. On the other hand, if α is an assignment of $\{u_0, ..., u_k\}$ that does not satisfy $[[M, \varphi]]_k$, then either $\alpha(u_0) \cdots \alpha(u_k)$ is not a path of M_k or $\alpha(u_0) \cdots \alpha(u_k) \not\models_e \varphi$ when $\alpha(u_0) \cdots \alpha(u_k)$ is a path of M_k.

3.2 Bounded Model Verification

Theorem 1 and Theorem 2 provide a theoretical basis for verification of LTL formulas. As a corollary, we have

Corollary 1. *$M \models \varphi$ if there is a k such that $[[M, \neg\varphi]]_k$ is unsatisfiable.*

This has also been proved in [15] in a different way, and yields a iterative bounded model checking procedure as follows. (1) Start with $k = 0$; (2) Compute $[[M, \neg\varphi]]_k$ and if $[[M, \neg\varphi]]_k$ is unsatisfiable, report that $M \models \varphi$ is valid; (3) Increase k and repeat the process until a resource bound is reached.

We first present an illustrative example to show the potential advantage of this approach and discuss the combining of this with the original bounded model checking approach, and then in Section 4, we show that for a subset of LTL formulas a stronger result can be obtained to improve the bounded model checking procedure.

Illustrative Example and Discussion This approach also has advantage over BDD based approaches, when a small k is sufficient for the verification. We use an example to illustrate this advantage. Let $p_0, ..., p_{n-2}, q, r$ be variables of the domain $\{0, 1\}$ and $\oplus$ be the function: addition modulo 2. Let the system be consist of n processes. A, B and C_i for $i = 0, ..., n-3$ (each is a sequential process which executed in parallel to each other with the interleaving semantics) with the following specification:

$A: \ r = r \oplus 1; p_0 = p_0 \oplus 1$
$B: \ p_{n-2} = p_{n-2} \oplus 1; q = q \oplus 1$
$C_i : p_i = p_i \oplus 1; p_{i+1} = p_{i+1} \oplus 1$ for $i = 0, ..., n-3$

Let the initial state be $p_i = 0$ and $q = r = 1$.

Let $\varphi = ((p_1 \vee p_3 \cdots \vee p_{n-2})Rq)$ for an odd number n.

For verifying $\varphi(n)$, we first transform the problem to CNF formula according to the proposed transformation scheme, then we use zChaff [1] for verification. For $n = 7, 9, 11, 13$, the property is verified when k reaches respectively $6, 7, 8, 9$. The verification times by zChaff for $n = 7, 9, 11, 13$ are as follows.

Property	k	Time (s)	Variables	Clauses	SAT-Time (s)
$\varphi(7)$	6	0.12	252	4206	0.07
$\varphi(9)$	7	0.42	366	7825	0.15
$\varphi(11)$	8	1.62	500	13053	0.31
$\varphi(13)$	9	13.01	654	20181	0.66

The second column shows the value of k which is sufficient for verifying the property. The third column is the time used by zChaff for the round where k is sufficient for verification, or for the k-th round. The fourth and fifth columns are the numbers of variables and clauses generated in the same round. The last column show the total time used in the previous rounds (i.e. from 0-th round to $(k-1)$-th round of the unsuccessful verification attempts) in which the inputs to zChaff are all satisfiable.

The formula can also be written as the CTL formula $A((p_1 \vee p_3 \cdots \vee p_{n-2})Rq)$. For comparison, we have carried out the same verification task using SMV (release 2.5.4.3) [2]. An example of SMV code for $n = 3$ is shown as follows.

[1] Available from http://www.princeton.edu/∼chaff/zchaff.html
[2] Available from http://www.cs.cmu.edu/∼modelcheck/smv.html

```
MODULE main
VAR
    q: boolean; r: boolean;
    p0: boolean; p1: boolean;
    b0: process cc(p1,q);
    c0: process cc(p0,p1,q);
ASSIGN
    init(r):=1; init(q):=1;
    init(p0):=0; init(p1):=0;
VAR  a: {a0,a1,a2};
ASSIGN
    init(a) := a0;
    next(a) := case a=a0: a1; a=a1: a2; 1: a; esac;
    next(r) := case a=a0: !r; 1: r; esac;
    next(p0) := case a=a1: !p0; 1: p0; esac;
SPEC  (!E[!(p1)U!(q)])
MODULE cc(p,s)
VAR  c: {c0,c1,c2};
ASSIGN
    init(c) := c0;
    next(c) := case c=c0: c1; c=c1: c2; 1: c; esac;
    next(p) := case c=c0: !p; 1: p; esac;
    next(s) := case c=c1: !s; 1: s; esac;
```

The verification data for $n = 7, 9, 11, 13$ are shown as follows.

Property	Time (s)	BDD nodes	Memory (KB)
$\varphi(7)$	1.14	16481	1441.792
$\varphi(9)$	9.07	102518	2818.048
$\varphi(11)$	101.22	638219	11337.728
$\varphi(13)$	2248.33	3789672	61800.448

For $n = 7$, the difference of time between the two approaches is around 6 times (counting the total time used by zChaff in all $k+1$ rounds), and for $n = 13$, the difference is more than 150 times. Therefore the tables show clear advantage of using the bounded verification approach over the BDD based verification approach for this example.

3.3 Combining Approaches

As mentioned, the approach presented previously is effective for verification of certain model checking problem instances, but there is no guarantee that we finally obtain a conclusion, especially for unsatisfied properties. This can be combined with the bounded model checking approach [2, 3] such that there are possibility of quickly coming to a conclusion for either satisfiable or unsatisfiable properties, and there is a guarantee that a conclusion is reached when k

is sufficiently large (assuming that we have sufficient computation resources). We first present the encoding of the bounded model checking approach for $(formula, state)$ pairs based on the bounded semantics.

The encoding of $(formula, state)$ pairs in the bounded model checking approach is based on the concept of (k, l)-loop. Let min() be the minimum operation and $s(i, k, l)$ denote

$$\text{if } (k = i) \text{ then } l \text{ else } i + 1.$$

Definition 7 (Translation of LTL formulas). *Let $u_0, ..., u_k$ be state variables and φ be an LTL formula. The encoding of (φ, u_i), denoted by $[[\varphi, u_i]]_{k,l}$, is defined as follows.*

$$\begin{array}{ll}
[[p, u_i]]_{k,l} & = p(u_i) \\
[[\neg p, u_i]]_{k,l} & = \neg p(u_i) \\
[[\varphi \vee \psi, u_i]]_{k,l} & = [[\varphi, u_i]]_{k,l} \vee [[\psi, u_i]]_{k,l} \\
[[\varphi \wedge \psi, u_i]]_{k,l} & = [[\varphi, u_i]]_{k,l} \wedge [[\psi, u_i]]_{k,l} \\
\hline
[[X\varphi, u_i]]_{k,l} & = [[\varphi, u_{s(i,k,l)}]]_{k,l} \\
[[G\varphi, u_i]]_{k,l} & = \bigwedge_{j=\min(i,l)}^{k} [[\varphi, u_j]]_{k,l} \\
[[\varphi U \psi, u_i]]_{k,l} & = \bigvee_{j=i}^{k} ([[\psi, u_j]]_{k,l} \wedge \bigwedge_{t=i}^{j-1} [[\varphi, u_t]]_{k,l}) \vee \\
 & \bigwedge_{t=i}^{k} [[\varphi, u_t]]_{k,l} \wedge \\
 & \bigvee_{j=l}^{i-1} ([[\psi, u_j]]_{k,l} \wedge \bigwedge_{t=l}^{j-1} [[\varphi, u_t]]_{k,l})
\end{array}$$

where $[[\varphi, u_{-1}]]_{k,l} = false$.

In the above definition, u_{-1} is a special symbol used only for the purpose of uniform formula representation (avoiding specification of different cases explicitly), for instance, $[[p \vee q, u_{-1}]]_{k,l}$ must be replaced by *false* and not by $[[p, u_{-1}]]_{k,l} \vee [[q, u_{-1}]]_{k,l}$. In addition, we define $T(u_k, u_{-1}) = true$. The subscript (k, l) in the definition indicates that the path is a (k, l)-loop for $l \geq 0$, otherwise the path is considered loop free (when $l = -1$).

Definition 8. *Let M be a model and φ be an LTL formula.* $[[M, \varphi]]_k^o := [[M]]_k \wedge \bigvee_{l=-1}^{k} (T(u_k, u_l) \wedge [[\varphi, u_0]]_{k,l})$.

The encoding of $[[M, \varphi]]_k^o$ corresponds to that in [2] with a simplification where a condition $\bigwedge_{l=0}^{k} \neg T(u_k, u_l)$ representing loop-free-ness is removed (or more precisely, replaced by *true*). This change does not affect the satisfiability of the formula, since if a formula is satisfied by a k-path which is not interpreted as a k-loop, then it is also satisfied by the same k-path interpreted as a k-loop (if it indeed is one). We formulate this fact as a lemma as follows.

Lemma 2. $[[\varphi, u_0]]_{k,-1} \rightarrow [[\varphi, u_0]]_{k,l}$ for $l \in \{0, ..., k\}$.

Proof. We prove a more general property for $[[\varphi, u_i]]_{k,l}$ and consider the lemma as a special case where $i = 0$ of this property. The property is as follows.

$$[[\varphi, u_i]]_{k,-1} \rightarrow [[\varphi, u_i]]_{k,l} \text{ for } i \in \{0, ..., k\} \text{ and } l \in \{0, ..., k\}$$

This property is to be proved by structural induction. The case is trivial for φ being a proposition or negation of a proposition. Assume the induction hypothesis.

- The case is also trivial for φ being a conjunctive or disjunctive formula, according to the induction hypothesis.
- If $\varphi = X\varphi_0$, then $[[\varphi, u_i]]_{k,-1}$ is either *false* $(i = k)$ or the same as $[[\varphi_0, u_{i+1}]]_{k,-1}$ $(i < k)$.
  In the latter case, $[[\varphi, u_i]]_{k,l} = [[\varphi_0, u_{i+1}]]_{k,l}$. Then, according to the induction hypothesis, $[[\varphi, u_i]]_{k,-1} \rightarrow [[\varphi, u_i]]_{k,l}$.
- If $\varphi = G\varphi_0$, then $[[\varphi, u_i]]_{k,-1}$ is *false*. Therefore $[[\varphi, u_i]]_{k,-1} \rightarrow [[\varphi, u_i]]_{k,l}$.
- If $\varphi = \varphi_0 U \varphi_1$,
  then $\bigvee_{j=-1}^{i-1}([[\varphi_1, u_j]]_{k,-1} \wedge \bigwedge_{t=-1}^{j-1}[[\varphi_0, u_t]]_{k,-1}) = \mathit{false}$.
  Therefore
  $[[\varphi, u_i]]_{k,-1} = \bigvee_{j=i}^{k}([[\varphi_1, u_j]]_{k,-1} \wedge \bigwedge_{t=i}^{j-1}[[\varphi_0, u_t]]_{k,-1})$.
  Then, according to the induction hypothesis,
  $[[\varphi, u_i]]_{k,-1} \rightarrow \bigvee_{j=i}^{k}([[\varphi_1, u_j]]_{k,l} \wedge \bigwedge_{t=i}^{j-1}[[\varphi_0, u_t]]_{k,l})$.
  Since the right side of the implication is a disjunctive part of $[[\varphi, u_i]]_{k,l}$, we obtain $[[\varphi, u_i]]_{k,-1} \rightarrow [[\varphi, u_i]]_{k,l}$.

The following theorem corresponds to the soundness theorem of the bounded LTL model checking approach as presented in [2].

Theorem 3. *Let M be a model, φ be an LTL formula. Let $k \geq 0$. There is a path π of M such that $\pi \models_k \varphi$ iff $[[M, \varphi]]_k$ is satisfiable.*

Proof. Since the only difference in the encoding $[[M, \varphi]]_k$ and that in [2] is that a condition representing loop-free-ness is removed, the fact that this change does not affect the satisfiability of the formula is easily seen based on Lemma 2.

Corollary 2. *Let M be a model, φ an LTL formula. $M \not\models \varphi$ iff there is a k such that $[[M, \neg\varphi]]_k^o$ is satisfiable.*

A combination of Corollary 1 and Corollary 2 suggests the following combination of verification and error detection approach. Let M be a model and φ be an LTL formula to be verified.

- Start with $k = 0$;
- If $[[M, \neg\varphi]]_k$ is unsatisfiable, report that $M \models \varphi$ is valid;
- If $[[M, \neg\varphi]]_k^o$ is satisfiable, report that $M \models \varphi$ does not hold;
- If a given completeness threshold is reached, report that $M \models \varphi$ is valid;
- Increase k and repeat the process.

As implied by the example, the procedure may terminate before reaching a completeness threshold. However, in the general case, it may be necessary to repeat the process until a completeness threshold is reached. For instance, if we have the trivial property $\varphi = G\,\mathit{true}$, which is *true* for all systems, then we have $[[M, \neg\varphi]]_k^o = \mathit{false}$ and $[[M, \neg\varphi]]_k = I(u_0) \wedge \bigwedge_{i=0}^{k-1} T(u_i, u_{i+1})$. Then the first one is unsatisfiable and the second is always satisfiable. The above approach can only terminate when a completeness threshold is reached. In the next section, we improve the approach for a subset of LTL formulas, such that the completeness threshold is avoided for this subset of formulas.

4 LTL(X,G)

The subset of LTL considered here is LTL formulas in NNF not containing temporal operators not in {X,G}. This subset is denoted LTL(X,G). Define X-rank of a formula to be the number of nested levels of X and G-rank to be the number of nested levels of G. Formally:

Definition 9. *Let $gr(\varphi)$ be the G-rank of φ and $xr(\varphi)$ be the X-rank of φ. Then*

$gr(p) = 0$ *for a literal* p
$gr(X\varphi) = gr(\varphi)$
$gr(G\varphi) = gr(\varphi) + 1$
$gr(\varphi U \psi) = gr(\varphi \vee \psi) = gr(\varphi \wedge \psi) = max(gr(\varphi), gr(\psi))$
$xr(p) = 0$ *for a literal* p
$xr(X\varphi) = xr(\varphi) + 1$
$xr(G\varphi) = xr(\varphi)$
$xr(\varphi U \psi) = xr(\varphi \vee \psi) = xr(\varphi \wedge \psi) = max(xr(\varphi), xr(\psi))$

For convenience, we write $\pi \models \varphi$ for $\pi \models_k \varphi$ when π is a finite path with $|\pi| = k - 1$. Then π in $\pi \models \varphi$ may be a finite path or an infinite path according to the context.

For an LTL(X,G) formula with G-rank = 0, the prefix of length $xr + 1$ of a path (with xr being the X-rank of the formula) is sufficient for proving or disproving whether the path satisfies the formula.

Lemma 3. *Let φ be an LTL(X,G) formula with G-rank = 0. Let xr be the X-rank of φ. If $\pi \models_e \varphi$ and $|\pi| \geq xr + 1$, then $\pi_0 \cdots \pi_{xr} \models \varphi$.*

Proof. Proof by induction on the X-rank and the structure of φ. The lemma is trivial for $xr = 0$. Let the X-rank of φ be $xr + 1$. If φ is a conjunction or a disjunction, then the conclusion follows from the induction hypothesis. Let $\varphi = X\varphi_0$. According to the induction hypothesis, if $\pi \models_e \varphi_0$ and $|\pi| \geq xr + 1$, then $\pi_0 \cdots \pi_{xr} \models \varphi_0$. Assume $\pi' \models_e X\varphi_0$ and $|\pi'| \geq xr + 2$. Let $\pi' = \pi'_0\pi''$. Then $\pi'' \models_e \varphi_0$ and $|\pi''| \geq xr + 1$. Therefore $\pi''_0 \cdots \pi''_{xr} \models \varphi_0$. By looking at the relation between π' and π'', we obtain that $\pi'_0 \cdots \pi'_{xr+1} \models X\varphi_0$. This concludes the proof.

For an LTL(X,G) formula with G-rank = 1, then a finite prefix of a path is sufficient for showing whether there is a path of M satisfying the formula.

Lemma 4. *Let φ be an LTL(X,G) formula with G-rank = 0. Let xr be the X-rank of φ. If $\pi \models_e G\varphi$ with $|\pi| \geq (2 \cdot xr + 1) \cdot k_M^{2 \cdot xr+1} + 2 \cdot xr \cdot k_M + 1$, then there is an $xr \leq i < \pi$ such that $\pi_0 \cdots \pi_i \models G\varphi$.*

Proof. Since $|\pi| \geq (2 \cdot xr + 1) \cdot k_M^{2 \cdot xr+1} + 2 \cdot xr \cdot k_M + 1$, there is an s such that s appears at least $m = (2 \cdot xr + 1) \cdot (k_M^{2 \cdot xr} + 1)$ times in π. Let the positions of the occurrences of s in π be $j_1, j_2, ..., j_m$. Let $a_j = xr + j \cdot (2xr + 1)$. Among the positions, choose $j_{a_0}, j_{a_1}, ..., j_{a_{k_M^{2 \cdot xr}}}$. The distance between any two

of these positions in π is at least $(2xr+1)$, the distances from $\pi_{j_{a_0}}$ to the first position of π and from $j_{a_{k_M^{2\cdot xr}}}$ to the last position of π are at least xr. For each such position j, we have a path $\pi_{j-xr}\cdots,\pi_j\cdots\pi_{j+xr}$ of length $2xr+1$ (in which $\pi_j = s$ for all such j) with no overlap with any other such path. Since there are $k_M^{2\cdot xr}+1$ such paths, at least two of them are identical. Let $a < b$ such that $\pi_{a-xr}\cdots\pi_a\cdots\pi_{a+xr} = \pi_{b-xr}\cdots\pi_b\cdots\pi_{b+xr}$. Then we have a path $\pi_0\cdots\pi_a\cdots\pi_b$ with $T(\pi_b,\pi_{a+1})$ holds such that $\pi_0\cdots\pi_a\cdots\pi_b \models G\varphi$ and $b < |\pi|$. Since φ is an LTL(X,G) formula with G-rank $= 0$ and $xr \leq b$, we have $\pi_0\cdots\pi_a\cdots\pi_b \models \varphi$. Therefore there is an $xr \leq b < |\pi|$ such that $\pi_0\cdots\pi_a\cdots\pi_b \models G\varphi$.

Corollary 3. *Let φ and ψ be LTL(X,G) formulas with G-rank $= 0$. Let xr be the X-rank of $\psi\wedge G\varphi$. If $\pi \models_e \psi\wedge G\varphi$ with $|\pi| \geq (2\cdot xr+1)\cdot k_M^{2\cdot xr+1}+2\cdot xr\cdot k_M+1$, then there is an $xr \leq i < \pi$ such that $\pi_0\cdots\pi_i \models \psi\wedge G\varphi$.*

For a general LTL(X,G) formula, then a finite prefix of a path is also sufficient for showing whether there is a path of M satisfying the formula. Before considering general LTL(X,G) formulas, we define a normal form for LTL by allowing generalized disjunction (a disjunction of a set of formulas).

4.1 LTL Disjunctive Normal Form

Definition 10. *LTL_{dnf} formulas are formulas constructed by the following rules:*

- *If $p_1,...,p_n$ are literals and $m_1,...,m_n \geq 0$, then $\bigvee_{i=1}^n X^{m_i}p_i$ is a base LTL_{dnf} formula. A base LTL_{dnf} formula is an LTL_{dnf} formula.*
- *If $\varphi_{i,0}$ is a base LTL_{dnf} formula, and $\varphi_{i,1},\varphi_{i,2},\varphi_{i,3}$ are LTL_{dnf} formulas for $i \in \{1,...,n\}$, then $\bigvee_{i=1}^n \psi_i$ is an LTL_{dnf} formula, where ψ_i is either $\varphi_{i,0}$, $G\varphi_{i,1}$, $\varphi_{i,2}U\varphi_{i,3}$, or a conjunction of two or all the three of these formulas.*

Every LTL formula can be transformed into an equivalent LTL_{dnf} formula. For the first we require an LTL formula to be in NNF. Then the operator X can be moved next to literals by applying the following equivalences:

$$\begin{aligned} X(\varphi_0\wedge\varphi_1) &= X\varphi_0\wedge X\varphi_1 \\ X(\varphi_0\vee\varphi_1) &= X\varphi_0\vee X\varphi_1 \\ XG\varphi_0 &= GX\varphi_0 \\ X(\varphi_0 U\varphi_1) &= X\varphi_0 U X\varphi_1 \end{aligned}$$

LTL formulas not involving U and G can be transformed into base LTL_{dnf} formulas by applying distributivity, associativity and commutativity of the operators $\wedge$ and $\vee$.

For more complicated formulas, this can be done inductively. If the outermost operation is $\vee$, let the formula be $\varphi\vee\psi$. Then the induction hypothesis is applicable to φ and ψ. If the outermost operation is G, let the formula be $G\varphi$. Then the induction hypothesis is applicable to φ. Let the result of the transformation of φ be φ'. Then $G\varphi'$ is an LTL_{dnf} formula.

The case is similar for the operator U. If the outermost operation is $\wedge$, then the formula can be treated as a conjunction of a set of formulas. If some of the formulas is a disjunction, we can apply the distributivity of $\wedge$, such that we have a disjunction of two simpler formulas and the induction hypothesis can be applied. Assume the formulas in the disjunction are of the form $X^n p$, $G\varphi$ and $\varphi U \psi$. We first collect the formulas of the form $X^n p$ to be a base $\mathrm{LTL}_{\mathrm{dnf}}$ formula. Then we collect and transform the formulas of the form $G\varphi$ (if there is any) to be a formula of the form $G\varphi'$ where φ' is an $\mathrm{LTL}_{\mathrm{dnf}}$ formula. This can be done as follows. Let the formulas of the form $G\varphi$ be $G\varphi_1$,...,$G\varphi_n$. We have

$$G\varphi_1 \wedge \cdots \wedge G\varphi_n = G(\varphi_1 \wedge \cdots \wedge \varphi_n).$$

Since $\varphi_1 \wedge \cdots \wedge \varphi_n$ is simpler than the formula we started with, the induction hypothesis is applicable. Therefore $\varphi_1 \wedge \cdots \wedge \varphi_n$ can be transformed to an $\mathrm{LTL}_{\mathrm{dnf}}$ formula. Finally, we collect and transform the formulas of the form $\varphi U \psi$ (if there is any) to be a formula of the form $\varphi' U \psi'$ where φ' and ψ' are $\mathrm{LTL}_{\mathrm{dnf}}$ Formulas. Let the formulas of the form $\varphi U \psi$ be $\varphi_1 U \psi_1$,...,$\varphi_m U \psi_m$. Let $[m]$ denote the set $\{1, ..., m\}$. We have

$$\begin{array}{l}(\varphi_1 U \psi_1) \wedge \cdots \wedge (\varphi_m U \psi_m) \\ = (\bigwedge_{i\in[m]} \varphi_i) U (\bigvee_{i\in[m]} (\psi_i \wedge \bigwedge_{j\in[m]\setminus\{i\}} (\varphi_j U \psi_j)))\end{array}$$

Since $\bigwedge_{i\in[m]} \varphi_i$ and $\psi_i \wedge \bigwedge_{j\in[m]\setminus\{i\}} (\varphi_j U \psi_j)$ are simpler than the formula we started with, the induction hypothesis is applicable and these formulas can be transformed to $\mathrm{LTL}_{\mathrm{dnf}}$ formulas. Therefore a formula with the outermost operation $\wedge$ can also be transformed into a formula in the required format.

As a summary, every LTL formula (in NNF) can be transformed into an equivalent $\mathrm{LTL}_{\mathrm{dnf}}$ formula by the following transformation rules (in addition to the associativity and commutativity of conjunction and disjunction).

$X(\varphi_0 \wedge \varphi_1)$	$= X\varphi_0 \wedge X\varphi_1$
$X(\varphi_0 \vee \varphi_1)$	$= X\varphi_0 \vee X\varphi_1$
$XG\varphi_0$	$= GX\varphi_0$
$X(\varphi_0 U \varphi_1)$	$= X\varphi_0 U X\varphi_1$
$\varphi \wedge (\psi_0 \vee \psi_1)$	$= (\varphi \wedge \psi_0) \vee (\varphi \wedge \psi_1)$
$G\varphi \wedge G\psi$	$= G(\varphi \wedge \psi)$
$(\varphi_0 U \varphi_1) \wedge (\psi_0 U \psi_1)$	$= (\varphi_0 \wedge \psi_0) U ((\varphi_1 \wedge (\psi_0 U \psi_1)) \vee$ $(\psi_1 \wedge (\varphi_0 U \varphi_1)))$

By examining these transformation rules, we have that the formulas on both sides are equivalent also with respect to the weak bounded semantics.

Theorem 4. *Let φ be an LTL formula, and φ' be an $\mathrm{LTL}_{\mathrm{dnf}}$ formula constructed from φ by applying the given rules. Then $\pi \models_e \varphi$ iff $\pi \models_e \varphi'$.*

Proof. Since each of the rules is sound with respect to the weak bounded semantics, the theorem follows from induction on the number of applications of the rules.

A property of the transformation rules is that if φ and ψ are LTL_{dnf} formulas with G-rank $\leq n$, then the G-rank of $\varphi \wedge \psi$ (viewed as an abbreviation of the equivalent LTL_{dnf} formula after the transformation) is also $\leq n$.

4.2 LTL_{dnf} (X,G)

Following the definition of LTL_{dnf} , every LTL(X,G) formula can be transformed into an equivalent LTL_{dnf} formula (with respect to both the normal semantics and the weak bounded semantics) restricted to temporal operators X and G. We denote this fragment of LTL_{dnf} by LTL_{dnf} (X,G). Instead of considering LTL(X,G) formulas, we now consider LTL_{dnf} (X,G) formulas. Without of loss of generality, We only consider LTL_{dnf} (X,G) formula of the following form

$$\psi_0 \vee \bigvee_{i=1}^{n} \psi_i \wedge G\varphi_i$$

where ψ_i for $i = 0, ..., n$ are base LTL_{dnf} formulas (ψ_i may be *true*) and φ_i for $i = 1, ..., n$ are LTL_{dnf} formulas of the above form. A formula not in this form is considered as an abbreviation of the equivalent formula of this form.

Lemma 5. *Let $\zeta = \zeta'_0 \vee (\zeta'_1 \wedge G\zeta_1) \vee \cdots \vee (\zeta'_n \wedge G\zeta_n)$ be an LTL_{dnf} (X,G) formula. Let gr be the G-rank of ζ, and xr be the X-rank of ζ. There is an f such that if $\pi \models_e \zeta$ with $|\pi| \geq f(gr, \zeta)$ then there is an $xr \leq i < \pi$ such that $\pi_0 \cdots \pi_i \models \zeta$.*

Proof. The case is obvious, if $\pi \models_e \zeta'_0$. Suppose that one of $(\zeta'_i \wedge G\zeta_i)$ for $i = 1, ..., n$ holds. The case where $gr = 1$ is covered by Corollary 3. Let $m = ((2 \cdot xr + 1) \cdot k_M^{2 \cdot xr+1} + 2 \cdot xr \cdot k_M + 1)$. In this case, we may just set $f(1, \zeta) = m$. Suppose that the lemma holds for formulas with G-rank $\leq t$ and $gr = t + 1$. Assume (1) $\pi \models_e (\zeta'_1 \wedge G\zeta_1)$ (the other cases are similar). Let ζ_1 be $\psi_0 \vee (\psi_1 \wedge G\varphi_1) \vee \cdots \vee (\psi_{n_1} \wedge G\varphi_{n_1})$. Let $\pi = \pi_0 \cdots \pi_k$. Since $\pi \models_e G\zeta_1$, without loss of generality, we assume that $G\varphi_i$ holds at some point on π for all i (otherwise, we only need to consider a simpler formula), and assume (2) there is $0 \leq a_0 \leq a_1 \leq a_2 \leq \cdots \leq a_{n_1} \leq k + 1$ (the other cases are similar) such that

- $\pi^j \models_e \psi_0$ for $j = 0, ..., a_0 - 1$;
- $\pi^{a_0} \models_e G\varphi_1$ and $\pi^j \models_e (\psi_0 \vee \psi_1)$ for $j = a_0, ..., a_1 - 1$;
- $\pi^{a_1} \models_e G(\varphi_1 \wedge \varphi_2)$ and $\pi^j \models_e (\psi_0 \vee \psi_1 \vee \psi_2)$ for $j = a_1, ..., a_2 - 1$;
- $\cdots$;
- $\pi^{a_{n_1-1}} \models_e G(\varphi_1 \wedge \cdots \wedge \varphi_{n_1})$ and $\pi^j \models_e (\psi_0 \vee \psi_1 \vee \cdots \vee \psi_{n_1})$ for $j = a_{n_1-1}, ..., a_{n_1} - 1$;

Let $\zeta_1(i)$ be

$$\begin{array}{l} \bigwedge_{j=0}^{a_0-1} X^j\psi_0 \wedge \\ \bigwedge_{j=a_0}^{a_1-1} X^j((\psi_0 \vee \psi_1) \wedge \varphi_1) \wedge \cdots \wedge \\ \bigwedge_{j=a_{i-1}}^{a_i-1} X^j((\psi_0 \vee \psi_1 \vee \cdots \vee \psi_{i-1}) \wedge \varphi_1 \wedge \cdots \wedge \varphi_{i-1}) \wedge \\ GX^{a_i}((\psi_0 \vee \psi_1 \vee \cdots \vee \psi_i) \wedge \varphi_1 \wedge \cdots \wedge \varphi_i) \end{array}$$

Considering $\zeta_1(i)$ as an abbreviation of the equivalent LTL_{dnf} (X,G) formula, the G-rank of $\zeta_1(i)$ is $\leq t$. This fact is needed for the use of induction.

If $a_0 \geq m$, then $\pi_0 \cdots \pi_{m-1} \models_e \psi_0$. Then according to Lemma 4, there is an $xr \leq j < m$ such that $\pi_0 \cdots \pi_j \models G\psi_0$, and therefore $\pi_0 \cdots \pi_j \models G\zeta_1$, $\pi_0 \cdots \pi_j \models \zeta_1' \wedge G\zeta_1$, and $\pi_0 \cdots \pi_j \models \zeta$. For the remaining cases, the induction hypothesis is to be used.

Assume $a_0 < m$. Let $m_0 = m$ and $m_i = max\{f(t, \zeta_1(i-1)) \mid a_j < m_j, j = 0, ..., i-1\}$. Since we have $\pi_0 \cdots \pi_{a_i - 1} \models_e \zeta_1(i)$, if $a_i \geq m_i$, let xr' be the X-rank of $\zeta_1(i)$, then according to the induction hypothesis, there is an $xr \leq xr' \leq j < a_i$ such that $\pi_0 \cdots \pi_j \models \zeta_1(i)$ and therefore $\pi_0 \cdots \pi_j \models G\zeta_1$, $\pi_0 \cdots \pi_j \models \zeta_1' \wedge G\zeta_1$, and $\pi_0 \cdots \pi_j \models \zeta$. Let $f_{1,1}(t+1, \zeta) = max\{m_0, m_1, ..., m_{n_1}\}$. Then, in the case with assumption (1) and (2), we have that if $|\pi| \geq f_{1,1}(t+1, \zeta)$, there is an $xr \leq i < |\pi|$ such that $\pi_0 \cdots \pi_i \models \zeta$.

Similar for the other cases, we can define $f_{1,j}(t+1, \zeta)$ for $j = 2, ..., n_1!$ similar to $f_{1,1}(t+1, \zeta)$ for each permutation of the order of a_i, and define $f_{i,j}(t+1, \zeta)$ for $i = 2, ..., n$ and $j = 1, ..., n_i!$ for the case when $\pi \models_e \zeta_i' \wedge G\zeta_i$ holds. Finally, we define $f(t+1, \zeta)$ to be the maximum of these numbers. Then we have that if $|\pi| \geq f(t+1, \zeta)$, there is an $xr \leq i < |\pi|$ such that $\pi_0 \cdots \pi_i \models \zeta$. □

Since every LTL(X,G) formula can be transformed into an equivalent LTL_{dnf} (X,G) formula, the result on LTL_{dnf} (X,G) formulas applies also to LTL(X,G) formulas.

Corollary 4. *Let M be a model, ψ an LTL(X,G) formula. Let π be an infinite path of M. If for all k, $\pi_0 \cdots \pi_k \models_e \psi$, then there is an infinite path π' of M such that $\pi' \models \psi$.*

4.3 Verification Approach for the Negation of LTL(X,G) Formulas

Theorem 5. *Let M be a model, φ an LTL(X,G) formula. $M \models \neg\varphi$ iff there is a k such that $M_k \not\models_e \varphi$.*

Proof. This theorem follows from Theorem 1 and Corollary 4.

Corollary 5. *Let M be a model, φ an LTL(X,G) formula. $M \models \neg\varphi$ iff there is a k such that $[[M, \varphi]]_k$ is unsatisfiable.*

The bounded model verification approach in the Section 3 can therefore be improved to be as follows for verification of the negation of LTL(X,G) formulas. Let M be a model and φ be a temporal formula such that $\neg\varphi$ is an LTL(X,G) formula. For verification of $M \models \varphi$, we have the following steps.

- Start with $k = 0$;
- If $[[M, \neg\varphi]]_k$ is unsatisfiable, report that $M \models \varphi$ is valid;
- If $[[M, \neg\varphi]]_k^o$ is satisfiable, report that $M \models \varphi$ does not hold;
- Increase k and repeat the process.

In this procedure, the use of the completeness threshold has been avoided. The termination of this procedure is guaranteed by Theorem 5 and Theorem 3.

5 Concluding Remarks

For the first, a weak bounded semantics for LTL has been presented as a theoretical basis for the SAT-based bounded verification of LTL formulas. Then as known that this is not a complete approach, we have proved for a subset of LTL that this verification approach is complete. We have also provided an example to show the potential advantage of the verification approach and discussed the advantage of a combination of this approach with the traditional bounded model checking approach.

References

1. A. Biere, A. Cimmatti, E. Clarke, O. Strichman, and Y. Zhu. Bounded Model Checking. Advances in Computers 58, Academic Press, 2003.
2. A. Biere, A. Cimmatti, E. Clarke, and Y. Zhu. Symbolic Model Checking without BDDs. LNCS 1579:193-207. TACAS 99.
3. Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Masahiro Fujita, Yunshan Zhu. Symbolic Model Checking Using SAT Procedures instead of BDDs. DAC 1999: 317-320
4. Edmund M. Clarke, Armin Biere, Richard Raimi, Yunshan Zhu. Bounded Model Checking Using Satisfiability Solving. Formal Methods in System Design 19(1): 7-34 (2001).
5. J. R. Burch, E. M. Clarke, K. L. McMillan, and D. L. Dill. Sequential circuit verification using symbolic model checking. DAC 1990:46-51.
6. J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and J. Hwang. Symbolic model checking: 10^{20} states and beyond. IEEE Symposium on Logic in Computer Science 5: 428-439, 1990.
7. E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. Lecture Notes in Computer Science 131:52-71. Springer-Verlag, 1981.
8. E. M. Clarke and E. A. Emerson and A. P. Sistla. Automatic Verification of Finite state Concurrent Systems Using Temporal Logic Specifications. POPL 1983:117-126.
9. E. Allen Emerson and E. M. Clarke. Using Branching-time Temporal Logics to Synthesize Synchronization Skeletons. Science of Computer Programming 2(3):241-266. 1982.
10. Ranjit Jhala and Kenneth L. McMillan. Interpolation and SAT-based Model Checking. CAV 2003: 1-13.
11. D. Kroening, O. Strichman. Efficient Computation of Recurrence Diameters. VMCAI 2003: 298-309.
12. Leonardo de Moura, Harald Ruess, Maria Sorea. Bounded Model Checking and Induction: From Refutation to Verification. CAV 2003: 14-26.
13. A. Pnueli. A temporal logic of concurrent programs. Theoretical Computer Science 13:45-60. 1981.
14. Mary Sheeran, Satnam Singh and Gunnar Stlmarck. Checking Safety Properties Using Induction and a SAT-Solver. FMCAD 2000: 108-125.
15. Wenhui Zhang: SAT-Based Verification of LTL Formulas. Lecture Notes in Computer Science 4346 (FMICS/PDMC 2006):277-292.