Bounded Semantics of CTL and SAT-based Verification

Wenhui Zhang
State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
20 OCT 2009
Problem Area

Automatic verification of concurrent systems

• One of the main concerns is efficient methods for model checking
• Existing methods include symbolic model checking and bounded model checking
Our Work

Two concerns:

• Bounded semantics - considered as the basis for developing bounded model checking techniques and tools

• Tools - for doing bounded model checking, based on such bounded semantics
Basic Terminologies

• Model Checking - check whether a model satisfies a property

• Bounded Model Checking - make restrictions on a model in order to have possibility to verify some properties quickly
Model: Kripke Structure
Model: Execution Paths

\{p\} \quad \{p,q\}
Specification Languages

CTL, CTL*, mu-Calculus

Universal fragments:
ACTL, ACTL*, LTL

Existential fragments:
ECTL, ECTL*, LTL(E)
Model Checking

Given a model: $M$

Given a property (typically a universal one): $\varphi$

Check: $M \models \varphi$

Has been successful

Main limitation: the state explosion problem
State Explosion Problem

• Cause - there are many components in a system description such that the number of states is exponential in the number of components

• Consequence - the number of states in a model may be too large to be handled by a computing device.
Techniques

Our focus: bounded model checking

• Symbolic representation of states and paths
• Considering bounded paths
Model: Execution Paths
Boundel Models
(Boundary) Loops
Internal Loops
Bounded Model Checking

\[ M \models \varphi \]

If there is a bounded model \( M_k \) such that
\[ M_k \models \varphi \]

This is much simpler, when a small \( k \) is sufficient
Problems

How to define bounded semantics for different logics (fragments of such logics)

How to develop efficient implementations of verification approaches based on such semantics
Main Contents

• Define a framework for the discussion of bounded semantics
• Provide a bounded semantics for CTL
• Show that there is no sound and complete bounded semantics for CTL* within the framework

• Apply the bounded semantics of CTL to derive a SAT-based characterization of ACTL properties
• Compare such a characterization with BDD based verification approaches
Framework for Bounded Semantics

- Model: Kripke Structure
- Specification Language: CTL*
- Good Properties of Semantics of Temporal Logics
- Soundness andCompleteness of Bounded Semantics
Model: Kripke Structure

Given a set of propositions $AP$. A Kripke structure over $AP$ is a quadruple $M = \langle S, T, I, L \rangle$ 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
- $L : S \to 2^{AP}$ is a labeling function that maps each state to a subset of propositions of $AP$.

An infinite path $\pi = \pi_0 \pi_1 \cdots$ of $M$ is an infinite sequence of states such that $(\pi_i, \pi_{i+1}) \in T$ for all $i \geq 0$. A finite path $\pi$ of $M$ is a finite prefix of an infinite path of $M$. 
Let $AP$ be a set of propositional symbols. The set of CTL* formulas over $AP$ is defined as follows:

- If $p \in AP$, then $p$ is a state formula.
- If $\varphi_0$ and $\varphi_1$ are state formulas, then $\neg \varphi_0$, $\varphi_0 \land \varphi_1$ and $\varphi_0 \lor \varphi_1$ are state formulas.
- If $\psi$ is a path formula, then $E\psi$ and $A\psi$ are state formulas.

- If $\varphi$ is a state formula, then $\varphi$ is a path formula.
- If $\psi_0$ and $\psi_1$ are path formulas, then $\neg \psi_0$, $\psi_0 \land \psi_1$, $\psi_0 \lor \psi_1$, $X\psi_0$, $F\psi_0$, $G\psi_0$, $\psi_0 U \psi_1$ and $\psi_0 R \psi_1$ are path formulas.
Good Properties of Semantics of Temporal Logics

- Compositionality w.r.t. Path Quantifiers
- Compositionality w.r.t. Propositional Connectives
- Consistency w.r.t. Labeling
Compositionality w.r.t. Path Quantifiers

Let $M$ be a model and $s$ be a state.

Let $\models$ be a relation defined for path formulas and state formulas.

The relation $\models$ is compositional with respect to path quantifiers, if the following hold:

- $M, s \models A\varphi$ iff $M, \pi(s) \models \varphi$ for all $\pi(s)$ of $M$.
- $M, s \models E\varphi$ iff $M, \pi(s) \models \varphi$ for some $\pi(s)$ of $M$. 
Compositionality w.r.t. Propositional connectives

Let $M$ be a model, $s$ be a state, and $\pi$ be a path.

Let $\models$ be a relation for path formulas and state formulas.

The relation $\models$ is compositional with respect to propositional connectives, if the following hold:

- $M, s \models \varphi \lor \psi$ iff $M, s \models \varphi$ or $M, s \models \psi$
- $M, s \models \varphi \land \psi$ iff $M, s \models \varphi$ and $M, s \models \psi$
- $M, \pi \models \varphi \lor \psi$ iff $M, \pi \models \varphi$ or $S \models \psi$
- $M, \pi \models \varphi \land \psi$ iff $M, \pi \models \varphi$ and $M, \pi \models \psi$
Consistency w.r.t. Labeling

Let \( X \) be the next-time operator, and \( p \) be a proposition.

Let \( \models \) be a relation defined for path formulas and state formulas.

The relation \( \models \) satisfies the consistency property, if the following hold:

- \( M, s \models p \) iff \( p \in L(s) \).
- \( M, \pi \models X^n p \) iff \( p \in L(\pi_n) \) when \( \pi_n \) is the \((n + 1)\)th state of \( \pi \).
Semantic Relation

A simple semantic relation:
• compositional with respect to path quantifiers
• compositional with respect to propositional connectives
• consistent with respect to labeling

A (composed) semantic relation:
• a simple semantic relation or
• propositional combination of such relations
A bounded semantics is then represented by a family of semantic relations $\models_k$ each defined on a bounded model $M_k$ with the parameter $k$ indicating the bound.
The bounded semantics defined by $\models_k$ is sound and complete with respect to a given relation $\models$, iff the following hold:

- (Soundness) If $M_k \models_k \varphi$ for some $k \geq 0$, then $M \models \varphi$.
- (Completeness) If $M \models \varphi$, then there is a $k \geq 0$ such that $M_k \models_k \varphi$. 

Soundness and Completeness of a Bounded Semantics
## Properties of Existing Bounded Semantics

<table>
<thead>
<tr>
<th>Language</th>
<th>Publication</th>
<th>Sound and Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL(E)</td>
<td>TACAS 1999</td>
<td>yes</td>
</tr>
<tr>
<td>ECTL</td>
<td>Fundam. Inform. 2002</td>
<td>yes</td>
</tr>
<tr>
<td>ECTL*</td>
<td>Fundam. Inform. 2004</td>
<td>yes</td>
</tr>
<tr>
<td>CTL*</td>
<td>JCST 2006</td>
<td>no</td>
</tr>
<tr>
<td>ACTL</td>
<td>ICFEM 2007</td>
<td>yes</td>
</tr>
</tbody>
</table>
## Properties of Existing Bounded Semantics

<table>
<thead>
<tr>
<th>Language</th>
<th>Publication</th>
<th>Sound and Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL(E)</td>
<td>TACAS 1999</td>
<td>yes</td>
</tr>
<tr>
<td>ECTL</td>
<td>Fundam. Inform. 2002</td>
<td>yes</td>
</tr>
<tr>
<td>ECTL*</td>
<td>Fundam. Inform. 2004</td>
<td>yes</td>
</tr>
<tr>
<td>CTL*</td>
<td>JCST 2006</td>
<td>no</td>
</tr>
<tr>
<td>ACTL</td>
<td>ICFEM 2007</td>
<td>yes</td>
</tr>
</tbody>
</table>

Provide a sound and complete bounded semantics for CTL.

Explain that there is no such sound and complete bounded semantics for CTL*.
Bounded Semantics for CTL

CTL: a subset of CTL*.

Let $AP$ be a set of propositional symbols. The set of CTL formulas is defined as follows:

- Every member of $AP$ is a CTL formula.
- If $\varphi$ and $\psi$ are CTL formulas, then so are: $\neg \varphi$, $\varphi \land \psi$, and $\varphi \lor \psi$.
- If $\varphi$ and $\psi$ are CTL formulas, then so are: $EX \varphi$, $EF \varphi$, $EG \varphi$, $E(\varphi R \psi)$, $E(\varphi U \psi)$, $AX \varphi$, $AF \varphi$, $AG \varphi$, $A(\varphi R \psi)$, and $A(\varphi U \psi)$.

Without loss of generality, we only consider formulas in negation normal form (NNF).
Bounded Models

Model: \( M = \langle S, T, I, L \rangle \)

A \( k \)-path of \( M \) is a finite path of \( M \) with length \( k + 1 \).

\( k \)-model: \( M_k = \langle S, Ph_k, I, L \rangle \)

where \( Ph_k \) is the set of all different \( k \)-paths of \( M \).

A \( k \)-path has an internal loop, if \( \pi_i = \pi_j \) for some \( 0 \leq i < j \leq k \).

Let \( ilp(\pi) \) denote that \( \pi \) has an internal loop.

If \( \pi \) is a prefix of \( \pi' \), then \( ilp(\pi) \rightarrow ilp(\pi') \).
Formulation of the Bounded Semantics for CTL

The relation that \( \varphi \) holds on \( s \) in \( M_k \) is denoted \( M_k, s \models \varphi \). Let \( \pi = \pi_0 \cdots \pi_k \) denote a \( k \)-path of \( Ph_k \). The relation \( \models \) is defined as follows.

\[
\begin{align*}
M_k, s \models p & \iff p \in L(s) \\
M_k, s \models \neg p & \iff p \notin L(s) \\
M_k, s \models \varphi \land \psi & \iff (M_k, s \models \varphi) \text{ and } (M_k, s \models \psi) \\
M_k, s \models \varphi \lor \psi & \iff (M_k, s \models \varphi) \text{ or } (M_k, s \models \psi) \\
M_k, s \models AX \varphi & \iff k \geq 1 \land \forall \pi(s). (M_k, \pi_1 \models \varphi) \\
M_k, s \models AF \psi & \iff \forall \pi(s). (\exists i \leq k. (M_k, \pi_i \models \psi)) \\
M_k, s \models AG \psi & \iff \forall \pi(s). (ilp(\pi) \land (\forall i \leq k. (M_k, \pi_i \models \psi)))
\end{align*}
\]
Formulation of the Bounded Semantics for CTL (cont.)

\[ M_k, s \models A(\varphi U \psi) \text{ iff } \forall \pi(s). (\exists i \leq k. (M_k, \pi_i \models \psi \land \forall j < i. (M_k, \pi_j \models \varphi))) \]

\[ M_k, s \models A(\varphi R \psi) \text{ iff } \forall \pi(s). (\forall i \leq k. (M_k, \pi_i \models \psi \lor \exists j < i. (M_k, \pi_j \models \varphi)) \land (\exists j \leq k. (M_k, \pi_j \models \varphi) \lor ilp(\pi))) \]

\[ M_k, s \models EX \varphi \text{ iff } k \geq 1 \land \exists \pi(s). (M_k, \pi_1 \models \varphi) \]

\[ M_k, s \models EF \psi \text{ iff } \exists \pi(s). (\exists i \leq k. (M_k, \pi_i \models \psi)) \]

\[ M_k, s \models EG \psi \text{ iff } \exists \pi(s). (ilp(\pi) \land (\forall i \leq k. (M_k, \pi_i \models \psi))) \]

\[ M_k, s \models E(\varphi U \psi) \text{ iff } \exists \pi(s). (\exists i \leq k. (M_k, \pi_i \models \psi \land \forall j < i. (M_k, \pi_j \models \varphi))) \]

\[ M_k, s \models E(\varphi R \psi) \text{ iff } \exists \pi(s). (\forall i \leq k. (M_k, \pi_i \models \psi \lor \exists j < i. (M_k, \pi_j \models \varphi)) \land (\exists j \leq k. (M_k, \pi_j \models \varphi) \lor ilp(\pi))) \]
Soundness and Completeness of the Bounded Semantics

Properties of the bounded semantics:

If $M_k, s \models \varphi$, then $M_{k+1}, s \models \varphi$.

**Theorem (Soundness and Completeness)**

$M, s \models \varphi$ iff $M_k, s \models \varphi$ for some $k \geq 0$. 
A Simple Bounded Model Checking Principle for CTL

Let $M$ be a model, $s$ a state and $\varphi$ a CTL formula. A simple bounded model checking principle may be formulated as follows.

Let $MAX$ be a given number.

\[
\begin{align*}
\text{Let } k &= 0; \\
\text{If } M_k, s \models \varphi \text{ holds, report that } \varphi \text{ holds;} \\
\text{If } k = MAX, \text{ exit without a conclusion;} \\
\text{Increase } k, \text{ go to the first “if”-test;}
\end{align*}
\]
Completeness Threshold

The completeness threshold of the problem $M, s \models \varphi$ is defined as the least $k$ such that if $M_k, s \models \varphi$ does not hold then $M_{k'}, s \models \varphi$ does not hold for all $k' > k$.

The completeness threshold of the problem $M, s \models \varphi$ exists.

Corollary

Let $ct_0$ be an over-approximation of the completeness threshold of $M, s \models \varphi$. Then $M, s \models \varphi$ iff $M_k, s \models \varphi$ for some $k \leq ct_0$. 
Bounded Model Checking Principle for CTL

Let $M$ be a model, $s$ a state and $\varphi$ a CTL formula. The bounded model checking principle may be formulated as follows.

Let $ct_0$ be an over-approximation of $ct(M, s, \varphi)$.

\[
\begin{align*}
\text{Let } k &= 0; \\
\text{If } M_k, s \models \varphi \text{ holds, report that } \varphi \text{ holds;} \\
\text{If } k = ct_0, \text{ report that } \varphi \text{ does not hold;} \\
\text{Increase } k, \text{ go to the first "if"-test;} \\
\end{align*}
\]
Exploiting the Negation

CTL is closed under negation.

Let $\neg \varphi$ be for the corresponding NNF formula equivalent to $\neg \varphi$.

**Corollary**

$M, s \models \varphi$ iff

there is a $k$ such that $M_k, s \models \varphi$ and
there is no $k$ such that $M_k, s \models \neg \varphi$. 
Let $M$ be a model, $s$ a state and $\varphi$ a CTL formula. The bounded model checking and verification principle may be formulated as follows.

Let $k = 0$;

If $M_k, s \models \varphi$ holds, report that $\varphi$ holds;

If $M_k, s \models \neg \varphi$ holds, report that $\varphi$ does not hold;

Increase $k$, go to the first “if”-test;

Advantage: avoided the use of the completeness threshold.
OnCTL*

What we have:
a bounded semantics and bounded model checking and verification principle for CTL.

It would be natural to think about:
an extension of the above bounded semantics and bounded model checking and verification principle to CTL*.
On Bounded Semantics of CTL*

Theorem

*There is no sound and complete bounded semantics with respect to the semantics of CTL*.
Proof Ideas

Suppose that there is such a bounded semantics defined by $|=k$.

Consider a property $A(Gp \lor Fr)$ and the following models:

Then the bounded semantics defined by $|=k$ will not be able to distinguish the truth of $A(Gp \lor Fr)$ on two models.

Then we have a contradiction, because a sound and complete bounded semantics must be able to make difference evaluations of the formula on the models.
Lessons Learned

In order to develop a sound and complete bounded semantics for CTL*, we have to sacrifice some of the “good” properties of semantics relations.
Applications of the Bounded Semantics of CTL

The bounded semantics of CTL may be used to develop:

- QBF-based bounded model checking and verification algorithms for CTL
- SAT-based bounded model checking and verification algorithms for ACTL

Taking into consideration of the computational efficiency and the expressiveness of the logics, we have further looked into the second issue.
SAT-based Bounded Model Checking and Verification

What we need to do:

- Represent bounded models by propositional formulas
- Represent the bounded semantics by propositional formulas

One problem is that a $k$-model would have too many paths. The number of $k$-paths may be of the order

$$O(|S|^{k+1})$$

It is infeasible to check a $k$-model, for a reasonably large $k$.

Solution: Consider submodels and symbolic representation of submodels.
Submodels

Let $M_k = \langle S, Ph_k, I, L \rangle$ be the $k$-model of $M$.

$M^b_k = \langle S, Ph^b_k, I, L \rangle$ is a submodel of $M_k$, if $Ph^b_k \subseteq Ph_k$ where $b$ denotes the size of $Ph^b_k$.

We call $M^b_k$ a $(k, b)$-submodel of $M_k$.

Let the relation $M^b_k, s \models \varphi$ be defined similar to the relation $M_k, s \models \varphi$, only with paths restricted to that of $M^b_k$. 
Sufficient Number of Paths to be used in the Submodels

Let $\varphi$ be an ACTL formula and $\psi$ be an ECTL formula.

Let $n^a_k(\varphi)$ be the least number such that for all $s$, $M_k, s \models \varphi$ iff $M'_k, s \models \varphi$ for all $(k, n^a_k(\varphi))$-submodels $M'_k$.

Let $n^e_k(\psi)$ be the least number such that for all $s$, $M_k, s \models \psi$ iff $M'_k, s \models \psi$ for some $(k, n^e_k(\psi))$-submodel $M'_k$.

We consider over-approximations of $n^a_k(\varphi)$ and $n^e_k(\psi)$. 
Over-Approximations of $n_k^a(\varphi)$ and $n_k^e(\psi)$.

Let $\varphi$ be an ACTL formula. $f_k(\varphi)$ is defined as follows.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_k(p)$</td>
<td>$0$ if $p \in AP$</td>
</tr>
<tr>
<td>$f_k(\neg p)$</td>
<td>$0$ if $p \in AP$</td>
</tr>
<tr>
<td>$f_k(\varphi_0 \land \varphi_1)$</td>
<td>$\max(f_k(\varphi_0), f_k(\varphi_1))$</td>
</tr>
<tr>
<td>$f_k(\varphi_0 \lor \varphi_1)$</td>
<td>$f_k(\varphi_0) + f_k(\varphi_1)$</td>
</tr>
<tr>
<td>$f_k(AX\varphi)$</td>
<td>$f_k(\varphi) + 1$</td>
</tr>
<tr>
<td>$f_k(AF\varphi)$</td>
<td>$(k + 1) \cdot f_k(\varphi) + 1$</td>
</tr>
<tr>
<td>$f_k(AG\varphi)$</td>
<td>$f_k(\varphi) + 1$</td>
</tr>
<tr>
<td>$f_k(A(\varphi_0 U \varphi_1))$</td>
<td>$k \cdot \max(f_k(\varphi_0), f_k(\varphi_1)) + f_k(\varphi_0) + f_k(\varphi_1) + 1$</td>
</tr>
<tr>
<td>$f_k(A(\varphi_0 R \varphi_1))$</td>
<td>$k \cdot f_k(\varphi_0) + \max(f_k(\varphi_0), f_k(\varphi_1)) + 1$</td>
</tr>
</tbody>
</table>
Correctness of the Over-Approximations

Properties of $f_k(\varphi)$:

Let $\varphi$ be an ACTL formula. $n^a_k(\varphi) \leq f_k(\varphi)$.
Let $\psi$ be an ECTL formula. $n^e_k(\psi) \leq f_k(\neg \psi)$.

Theorem
Let $\varphi$ be an ACTL formula.
$M, s \models \varphi$ iff there is a $k$ such that $M'_k, s \models \varphi$ for all $(k, f_k(\varphi))$-submodels $M'_k$ and there is no $k$ such that $M''_k, s \models \neg \varphi$ for some $(k, f_k(\varphi))$-submodel $M''_k$. 
Bounded Model Checking and Verification for ACTL

Let $M$ be a model and $\varphi$ an ACTL formula. The corresponding bounded model checking and verification approach is as follows.

<table>
<thead>
<tr>
<th>Let $k = 0$;</th>
</tr>
</thead>
<tbody>
<tr>
<td>If $M'_k \models \varphi$ for all $(k, f_k(\varphi))$-models $M'_k$, report that the property holds;</td>
</tr>
<tr>
<td>If $M'_k \models \neg \varphi$ for some $(k, f_k(\varphi))$-model $M'_k$, report that the property does not hold;</td>
</tr>
<tr>
<td>Increase $k$, go to the first “if”-test;</td>
</tr>
</tbody>
</table>

The main difference is the use of $(k, f_k(\varphi))$-models instead of $k$-models.
SAT-based Implementation

Based on the above approach, a tool is implemented:

<table>
<thead>
<tr>
<th>Name of the tool:</th>
<th>VERBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location of the tool:</td>
<td><a href="http://lcs.ios.ac.cn/~zwh/verbs/">http://lcs.ios.ac.cn/~zwh/verbs/</a></td>
</tr>
</tbody>
</table>

The tool has a built-in SAT-solver, and it may also be linked to external SAT-solvers.

In particular, we have used MiniSat-1.14 in our experimental case studies.
Experimental Case Studies

1. A mutual exclusion algorithm with properties: mutual exclusion, liveness and non-starvation.

2. An asynchronous communication mechanism with rereading and overwriting, with a coherence property.

3. A set of examples for making a comparison of VERBS and SMV on their relative efficiency.
Experimental Case Study 1

This case study has been carried out with a mutual exclusion algorithm, with two processes. The three properties (mutual exclusion, liveness, non-starvation) are considered:

\[
\begin{align*}
AG(\neg(p1.cri \land p2.cri)) \\
AG((p1.req \lor p2.req) \rightarrow AF(p1.cri \lor p2.cri)) \\
AG((p1.req \rightarrow AF(p1.cri)) \land (p2.req \rightarrow AF(p2.cri)))
\end{align*}
\]

The verification process correctly verified the first two properties and falsified the last one.

The bounds for verification and falsification are respectively 35, 35, 4 for the three properties in the experiment.
Experimental Case Study 2

This case study has been carried out with an asynchronous communication mechanism (ACM) with rereading and overwriting. The memory of the ACM of the instance of our model has length 6. Then two instances of the property are specified as follows.

\[
AG(s = 6 \rightarrow (ra \in x \rightarrow ra = a[0] \land (wa \in x \rightarrow wa = a[5])))
\]
\[
AG(s = 6 \rightarrow (ra \in x \rightarrow ra = a[0] \land (wa \in x \rightarrow wa = a[4])))
\]

The verification process correctly verified the first property and falsified the second one.

The bounds reached are respectively 42, 12 for the two properties.
Experimental Case Study 3: Model

There are $3n$ global boolean variables $p[0],..., p[n-1], q[0],..., q[n-1], r[0],..., r[n-1]$.
Three processes $p, q, r$, each of which has in addition one local variable and has $n$ transitions. The transitions of $p$ are as follows:

\[
\begin{align*}
ss = a_0 & \quad \longrightarrow \quad (p[0], ss) := (\neg p[0], a_1); \\
ss = a_1 & \quad \longrightarrow \quad (p[1], ss) := (\neg p[1], a_2); \\
& \vdots \\
ss = a_{n-2} & \quad \longrightarrow \quad (p[n-2], ss) := (\neg p[n-2], a_{n-1}); \\
ss = a_{n-1} & \quad \longrightarrow \quad (p[n-1], ss) := (\neg p[n-1], a_0);
\end{align*}
\]

Within the process, the variables $p[i]$ are initially set to 0 for all $i \in \{0, ..., n-1\}$, and the variable $ss$ (acting as the program counter, which takes one of the values of $\{a_0, ..., a_{n-1}\}$) is initially $a_0$ (in practice, $a_i$ is interpreted as number $i$).
Experimental Case Study 3: Properties

Let $\varphi(i)$ be $\neg p[i] \land \neg q[i] \land \neg r[i]$. The following types of properties are considered.

\begin{align*}
PT1 & : A(\neg \varphi(i) R A(\neg \varphi(j) R \varphi(k))) \\
PT2 & : A(\neg \varphi(i) R A(\varphi(j) U \neg \varphi(k))) \\
PT3 & : A(\varphi(i) U A(\neg \varphi(j) R \varphi(k))) \\
PT4 & : A(\varphi(i) U A(\varphi(j) U \neg \varphi(k)))
\end{align*}

For each type, there are $n^3$ properties.

We have tried $n = 9$ and $n = 13$. 
Experimental Case Study 3: Data for $n = 9$

The number of properties for each type = 729.

Results on false properties (error detection):

<table>
<thead>
<tr>
<th></th>
<th>PT1</th>
<th>PT2</th>
<th>PT3</th>
<th>PT4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMV</td>
<td>525</td>
<td>324</td>
<td>405</td>
<td>204</td>
</tr>
<tr>
<td>VERBS</td>
<td>9 - 12</td>
<td>21 - 29</td>
<td>13 - 18</td>
<td>37 - 48</td>
</tr>
<tr>
<td></td>
<td>94.8%</td>
<td>100.0%</td>
<td>66.1%</td>
<td>100.0%</td>
</tr>
</tbody>
</table>

Results on true properties (verification):

<table>
<thead>
<tr>
<th></th>
<th>PT1</th>
<th>PT2</th>
<th>PT3</th>
<th>PT4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMV</td>
<td>204</td>
<td>405</td>
<td>324</td>
<td>525</td>
</tr>
<tr>
<td>VERBS</td>
<td>8 - 8</td>
<td>13 - 16</td>
<td>11 - 13</td>
<td>13 - 26</td>
</tr>
<tr>
<td></td>
<td>22.5%</td>
<td>18.2%</td>
<td>67.5%</td>
<td>58.8%</td>
</tr>
</tbody>
</table>
Experimental Case Study 3: Data for $n = 13$

The number of properties for each type = 2197.

Results on false properties (error detection):

<table>
<thead>
<tr>
<th></th>
<th>PT1</th>
<th>PT2</th>
<th>PT3</th>
<th>PT4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1547</td>
<td>1014</td>
<td>1183</td>
<td>650</td>
</tr>
<tr>
<td>SMV</td>
<td>59 - 81</td>
<td>232 - 286</td>
<td>89 - 127</td>
<td>419 - 507</td>
</tr>
<tr>
<td>VERBS</td>
<td>0 - 340</td>
<td>0 - 600+</td>
<td>0 - 600+</td>
<td>0 - 600+</td>
</tr>
<tr>
<td></td>
<td>96.8%</td>
<td>99.6%</td>
<td>72.7%</td>
<td>96.3%</td>
</tr>
</tbody>
</table>

Results on true properties (verification):

<table>
<thead>
<tr>
<th></th>
<th>PT1</th>
<th>PT2</th>
<th>PT3</th>
<th>PT4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>650</td>
<td>1183</td>
<td>1014</td>
<td>1547</td>
</tr>
<tr>
<td>SMV</td>
<td>53 - 54</td>
<td>131 - 156</td>
<td>76 - 89</td>
<td>131 - 266</td>
</tr>
<tr>
<td>VERBS</td>
<td>0 - 600+</td>
<td>0 - 600+</td>
<td>0 - 600+</td>
<td>0 - 600+</td>
</tr>
<tr>
<td></td>
<td>24.3%</td>
<td>18.1%</td>
<td>72.8%</td>
<td>59.5%</td>
</tr>
</tbody>
</table>
Experimental Case Study 3: Summary

This set of examples has illustrated that:

- With respect to ACTL properties, VERBS and SMV have their own advantages both for verification and falsification.
- The former has advantage when a small $k$ is sufficient for either verification or falsification.
- The latter has advantage on the opposite situations.

It is expected that when $n$ is increased to a certain number, SMV will be infeasible for all of the problem instances in the set of examples, while VERBS will still be feasible for some of the problem instances, for both of verification problems and error detection problems.
Concluding Remarks

General experience on bounded model checking:

- may win on examples with counter-examples, always loose on examples without counter-examples
- good for bug finding, not for verification

Our experience:

- may win on examples with counter-examples, may also win on examples without counter-examples
- good for bug finding, may also be useful for verification

Our experience is very limited, only on some artificial examples and properties, it remains to be seen whether this experience can be extended to practical applications, and there is still a lot of work that needs to be done in this area.
Thanks!