Bounded Model Checking using SAT Solving

Shoham Ben-David
Model Checking

System $S$

Specification $\varphi$

Model Checker

$S \models \varphi$?

YES.
Model Checking

System $S$  \rightarrow  Model Checker

Specification $\varphi$  \rightarrow  $S \models \varphi$?

\textbf{NO:}
Here is a counter example.

req  
\begin{tabular}{c}
\hline
\hline
\hline
\end{tabular}

gnt  
\begin{tabular}{c}
\hline
\hline
\hline
\end{tabular}
What is $S$? What is $\phi$? How is model checking performed?

- $S$ is a program, software or hardware
- $\phi$ is a temporal logic specification
- The model checking method depends on $S$ and $\phi$. 
Roughly speaking, model checking algorithms are divided into:

- **Explicit** methods
  - applied mainly to software programs
- **Symbolic** methods
  - applied mainly to hardware

In the context of model checking, **Symbolic** means manipulating sets of states.

The two branches of model checking use different sets of methods:

- Both use SAT solving
- In this talk: Symbolic Model Checking
Temporal Logic Specifications

• Linear Temporal Logic (LTL)
  • Allows specifying events over time
    • $G(\text{req} \rightarrow F(\text{ack}))$
    • $G(\text{ack} \rightarrow X(\neg\text{ack}))$

• Other languages exist: CTL, PSL and more

• For model checking purposes:
  • Translated into automata + a very simple formula
    • Invariant : $G(p)$ type formula, with $p$ being a Boolean formula
    • Most of the specifications are translated in this way

• In this talk: symbolic model checking of $G(p)$ formulas
  • OR: Symbolic reachability analysis :
    • is $p$ invariant?
    • is $\neg p$ reachable?
History of symbolic Model Checking

- 1977: Temporal Logic
  - Pnueli, "The temporal logic of programs"
- 1981/1982: Symbolic Model Checking
  - Clarke, Emerson: "Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic"
  - Queille, Sifakis, "Specification and verification of concurrent systems in CESAR"
- 1992: Symbolic Model Checking using BDDs
  - Burch, Clarke, McMillan, Dill, Hwang: "Symbolic Model Checking: $10^{20}$ States and Beyond".
    - McMillan also wrote the first symbolic model checker SMV
- 1999: Bounded Model Checking using SAT
  - Biere, Cimatti, Clarke, Zhu: "Symbolic Model Checking without BDDs"
Improvement in Symbolic Model Checking

No. of state variables

- 1982: 70
- 1992: 300
- 1999: 1000
- 2001: 10000
- 2013: 10000
History of Symbolic Model Checking

- BMC

SAT/SMT Solver Research Story
A 1000x Improvement

- Solver-based programming languages
- Compiler optimizations using solvers
- Solver-based debuggers
- Solver-based type systems
- Solver-based concurrency bugfinding
- Solver-based synthesis
- Bio & Optimization

- Concolic Testing
- Program Analysis
- Equivalence Checking
- Auto Configuration

- Bounded MC
- Program Analysis
- AI

1,000,000 Constraints
100,000 Constraints
10,000 Constraints
1,000 Constraints


Jay Ganesh
Example: a Simple Model

Three Boolean variables: $V_1, V_2, V_3$

**INITIAL ASSIGNMENT**

$\text{init}(V_1) := 1; \text{init}(V_2) := 1; \text{init}(V_3) := 0;$

**NEXT STATE ASSIGNMENT**

$\text{next}(V_1) := \begin{cases} 
V_1 \& V_2 : 0; \\
V_3 : 1; \\
\text{else} : \{0,1\}; \\
\end{cases} \text{else} : \{0,1\};$

$\text{next}(V_3) := V_1;$

$\text{next}(V_2) := \begin{cases} 
V : \{0,1\}; \\
\text{else} : 0; \\
\end{cases} \text{else} : \{0,1\};$

**SPECIFICATION**

$G(! (V_1 \& V_2 \& V_3))$
init(V_1) := 1; init(V_2) := 1; init(V_3) := 0;

next(V_1) := case
    V_1 & V_2 : 0;
    V_3 : 1;
    else : \{0,1\};
    esac;
next(V_2) := case
    V : \{0,1\};
    else : 0;
    esac;
next(V_3) := V_1;

G(!\{V_1 & V_2 & V_3\})
Bounded Model Checking using SAT

- Biere, Cimatti, Clarke, Zhu: Symbolic Model Checking without BDDs. TACAS 1999

- Bounded reachability: main idea
  - Let $S$ be a model, $G(p)$ a specification, and $k$ a natural number
  - Build a Boolean formula $B(S,p,k)$, such that
    - if $B(S,p,k)$ is satisfiable, then $S \not\models G(p)$, and the satisfying assignment is a counterexample
    - Otherwise ($B(S,p,k)$ is not satisfiable), no counterexample of length $k$ or less exists in the model
Bounded Model Checking (BMC)

K = 1
Bounded Model Checking using SAT

- Let $I$ be a Boolean formula representing the set of initial states.
- Introduce $k$ new sets of variables $V^1, \ldots, V^k$.
  - Use $V^0$ for the original set of variables.
- Let $T(V^i, V^{i+1})$ represent the transition relation, in terms of the variables $V^i, V^{i+1}$.
- Let $p^i$ represent $p$ written in terms of $V^i$.
- Define $B(S, p, k)$ to be:

$$I \land T(V^0, V^1) \land T(V^1, V^2) \land \ldots \land T(V^{k-1}, V^k) \land (\neg p^0 \lor \neg p^1 \lor \ldots \lor \neg p^k)$$
Bounded Model Checking using SAT

- $B(S,p,k) =$
  \[ I \land T(V^0,V^1) \land T(V^1,V^2) \land \ldots \land T(V^{k-1},V^k) \land (\neg p \lor \neg p^1 \lor \ldots \lor \neg p^k) \]

- What if $B(S,p,k)$ is satisfiable?
- What if $B(S,p,k)$ is unsatisfiable?
BMC: Example

K = 1

\[
\begin{align*}
\text{init}(V_1) & := 1; \quad \text{init}(V_2) := 1; \quad \text{init}(V_3) := 0; \\
\text{next}(V_1) & := \text{case} \\
& \quad V_1 \& V_2 : 0; \\
& \quad V_3 : 1; \\
& \quad \text{else} : \{0,1\}; \\
& \quad \text{esac}; \\
\text{next}(V_2) & := \text{case} \\
& \quad V : \{0,1\}; \\
& \quad \text{else} : 0; \\
& \quad \text{esac}; \\
\text{next}(V_3) & := V_1;
\end{align*}
\]

\[
G(! (V_1 \& V_2 \& V_3))
\]

\[
\begin{align*}
I & = (1)(2)(-3) \\
P & = (-1, -2, -3) \\
\neg P & = (1)(2)(3) \\
\text{Introduce variables} \\
1`, 2`, 3` \\
T & = ((1)(2) \rightarrow (-1`)) ((-1, -2)(3) \rightarrow (1`)) \\
& \quad (-2) \rightarrow (-2`)(1) \rightarrow (3`) ((-1) \rightarrow (-3`)) \\
\text{Converting to CNF:} \\
T & = (-1, 3`)(1, -3`), (2, -2`)(-1, -2, -1`) \\
& \quad (1, -3, 1´)(2, -3, 1`) \\
B(S,p,1) & = I \land T(V^0, V^1) \land (\neg p^0 \lor \neg p^1) \\
(\neg p^0 \lor \neg p^1) & = (1)(2)(3) V(1`)(2`)(3`) \\
\text{Converting to CNF:} \\
& \quad (4, 5)(-4, 1)(-4, 2)(-4, 3)(-5, 1´)(-5, 2´)(-5, 3´)
\end{align*}
\]

\[
B(S,p,1) = (1)(2)(-3) (-1, 3`)(1, -3`), (2, -2´)(-1, -2, -1`) (1, -3, 1´)(2, -3, 1`) \\
(4, 5)(-4, 1)(-4, 2)(-4, 3)(-5, 1´)(-5, 2´)(-5, 3´)
\]
BMC -- Summary

• Algorithm
  • Pick an initial \( k \) and increasing integer \( i \)
  • Loop:
    1. Build \( B(S,p,k) \)
    2. Check: is \( B(S,p,k) \) satisfiable?
      • If it is: return \( S \neq G(p) + \text{counterexample} \)
    3. Set \( k := k+i \)
Many solutions exist, notably

- Induction

- Interpolation
  - McMillan 2003: "Interpolation and SAT-Based Model Checking"