Lecture 10
Symbolic (BDD-based) Model Checking for CTL
BDD-based Model Checking

- Given: Kripke structure $M$, CTL formula $f$
- Returns: $S_f$ - the set of states satisfying $f$

$M$ is given by:
- BDD $R(V, V')$, representing the transition relation
- BDD $p(V)$, for every $p \in AP$, representing $S_p$
  - the set of states satisfying $p$
- $V = (v_1, ... v_n)$
BDD-based Model Checking

• For \( f = \text{EX} \, f_1 \) return

\[
f(V) = \exists V' \ [ \ f_1(V') \land R(V,V') \ ]
\]

• This BDD represents all (encoding V of) states that have a successor (with encoding V’) in \( f_1 \)
• Defined as a new BDD operator:
  \[ EX \ f_1(V) = \exists V' \ [ f_1(V') \land R(V,V') ] \]

• This operation is also called pre-image

• Important:
  the formula defines a sequence of BDD operations and therefore is considered as a symbolic algorithm
Model Checking $f = \text{EF } g$

Given: a model $M$ and the set $S_g$ of states satisfying $g$ in $M$

procedure $\text{CheckEF } (S_g)$

1. $Q := \text{emptyset}$; $Q' := S_g$
2. while $Q \neq Q'$ do
   1. $Q := Q'$
   2. $Q' := Q \cup \text{Pred}(Q)$
end while
3. $S_f := Q$; $\text{return}(S_f)$

Least fixpoint
Model Checking \( f = EG \ g \)

Given: BDDs \( R(V, V') \), \( g(V) \)

procedure \text{CheckEG} (g) 
\[
\begin{align*}
Q & := S; \\
Q' & := g; \\
\text{while } Q \neq Q' \text{ do} \\
Q & := Q'; \\
Q' & := Q \land \text{EX} (Q)
\end{align*}
\]
end while 
\[
f := Q; \quad \text{return}( f )
\]
Example: \( f = EG g \)
Bounded (SAT-based) Model Checking
State explosion problem - revisited

- state of the art BDD-based symbolic model checking can handle effectively designs with a few hundreds of Boolean variables

Other solutions for the state explosion problem are needed!
SAT-based model checking

- Translates the model and the specification to a propositional formula
- Uses efficient tools (SAT solvers) for solving the satisfiability problem

Since the satisfiability problem is **NP-complete**, SAT solvers are based on heuristics.
SAT tools

• Using heuristics, SAT tools can solve very large problems fast.
• They can handle systems with 1000 variables that create formulas with a few millions of variables.

GRASP (Silva, Sakallah)
Prover (Stalmark)
Chaff (Malik)
MiniSAT
Glucose
Bounded model checking (BMC) for checking AGp

• Given
  - A finite system $M$
  - A safety property AGp
  - A bound $k$

• Determine
  - Does $M$ contain a counterexample to AGp of $k$ transitions (or fewer)?
Bounded Model Checking (BMC) for checking $AGp$

- **Unwind** the model for $k$ levels, i.e., construct all computations of length $k$

- If a state satisfying $\neg p$ is encountered, produce a counterexample;
  Otherwise, **increase** $k$

[BCCZ 99]
Bounded Model Checking

Terminates
• with a counterexample or
• with time- or memory-out

The method is suitable for **falsification**, not verification
BMC for checking AGp (EF¬p)

Input to BMC:
A system over variables \( V = \{v_1, \ldots, v_n\} \), where
- \( \text{INIT}(V) \) is a propositional formula representing the set of initial states
- \( R(V, V') \) is a propositional formula representing the transition relation

A specification:
- \( \neg p(V) \) is a propositional formula representing the set of states satisfying \( \neg p \)
BMC for checking $\varphi = \text{EF} \neg p$

1. $k=1$
2. Build a propositional formula $f_M^k$ describing all prefixes of length $k$ of paths of $M$ from an initial state
3. Build a propositional formula $f_{\varphi}^k$ describing all prefixes of length $k$ of paths satisfying $\varphi$
4. If $(f_M^k \land f_{\varphi}^k)$ is satisfiable, return the satisfying assignment as a counterexample
5. Otherwise, increase $k$ and return to 2.
• If \((f_M^k \land f_\varphi^k)\) is unsatisfiable: 
  \(M\) has no counterexample of length \(k\)

• If \(k = 2^{|V|}\) then we can conclude \(M \models AGp\)
  - Too big - not practical

• The method is suitable for refutation
  - Bug finding
BMC for checking $\varphi = EF\neg p$

- $f_M^k (V_0, ..., V_k) =$
  
  \[ INIT(V_0) \land R(V_0, V_1) \land ... \land R(V_{k-1}, V_k) \]

- Uses $k+1$ copies of $V = \{ v_1, ..., v_n \}$
- $V_i$ represents the state after $i$ transitions
BMC for checking $\varphi = \text{EF} \neg p$

- To check if $p$ is violated within $k$ steps:

$$f^k_\varphi (V_0, \ldots, V_k) = \neg p(V_0) \lor \ldots \lor \neg p(V_k) = \lor_{i=0\ldots k} \neg p(V_i)$$

- To check if $p$ is violated exactly on state $k$:

$$f^k_\varphi (V_0, \ldots, V_k) = \neg p(V_k)$$

  - Useful when working iteratively on $k=0,1,2,\ldots$
BMC for checking $\varphi = \text{EF}\neg p$

- The iterative algorithm:

\[
\begin{align*}
\text{INIT}(V_0) & \land \neg p(V_0) \\
\text{INIT}(V_0) & \land R(V_0, V_1) \land \neg p(V_1) \\
\text{INIT}(V_0) & \land R(V_0, V_1) \land R(V_1, V_2) \land \neg p(V_2) \\
& \quad \ldots \\
& \quad \ldots
\end{align*}
\]

\[
\text{INIT}(V_0) \land R(V_0, V_1) \land R(V_1, V_2) \land K \land R(V_{k-1}, V_k) \land \neg p(V_k)
\]
Example - shift register

Shift register of 3 bits: \(<x, y, z>\)

Transition relation:

\[ R(x, y, z, x', y', z') = x' = y \land y' = z \land z' = 1 \]

Initial condition:

\[ INIT(x, y, z) = x = 0 \lor y = 0 \lor z = 0 \]

Specification: \( AG (x = 0 \lor y = 0 \lor z = 0) \)
Propositional formula for $k=2$

$$f_{M,2} = (x_0=0 \lor y_0=0 \lor z_0=0) \land$$
$$((x_1=y_0 \land y_1=z_0 \land z_1=1) \land$$
$$((x_2=y_1 \land y_2=z_1 \land z_2=1)$$

$$f_{\varphi,2} = \bigvee_{i=0,\ldots,2} (x_i=1 \land y_i=1 \land z_i=1)$$

Satisfying assignment: 101 011 111
This is a counterexample!
BMC for checking $\text{AFp (}\varphi=\text{EG} \neg p)$

- Is there an infinite path in $M$
  - From an initial state
  - all of its states satisfying $\neg p$
  - Over $k+1$ states?

- Must be a lasso
BMC for checking $\text{AFp} (\varphi=\text{EG} \neg p)$

An infinite path in $M$, from an initial state, over $k+1$ states, all satisfying $\neg p$:

- $f^k_M (V_0,\ldots,V_k) =$
  \begin{align*}
  & \text{INIT}(V_0) \land \land_{i=0,\ldots,k-1} R(V_i,V_{i+1}) \land \land_{i=0,\ldots,k-1} (V_k=V_i)
  \end{align*}

- $V_k=V_i$ means bitwise equality: $\land_{j=0,\ldots,n} (v_{kj} \leftrightarrow v_{ij})$

- $f^k_\varphi (V_0,\ldots,V_k) = \land_{i=0,\ldots,k} \neg p (V_i)$
A remark

In order to describe a computation of length $k$ by a propositional formula we need $k+1$ copies of the state variables.

With BDDs we use only two copies of current and next states.
Bounded model checking

- Can handle all of LTL formulas
- Can be used for verification by choosing \( k \) which is large enough
  - Need bound on length of the shortest counterexample.
    - diameter bound. The diameter is the maximum length of the shortest path between any two states.
- Using such \( k \) is often not practical due to the size of the model
  - Computing worst case diameter is exponential. Obtaining better bounds is sometimes possible, but generally intractable.
SAT-based verification

- Induction
- Interpolation
- IC3