Introduction

- Model Checking
  - Exhaustive verification.
  - Difficult to scale (state explosion).

- Bounded model checking (BMC) is targeted to find bugs, not to achieve the complete correctness proof.
  - Finds bugs in a bounded number of executions.
  - Can discover shallow bugs quickly.
  - Based on the latest advances in Boolean satisfiability solving.
  - State explosion is alleviated, but runtime may be a serious problem.
BMC vs SMC

- Both operate on Boolean manipulations.
- BMC uses SAT solving while SMC uses OBDDs.
  - Both are exponential procedures (time or space).
- BMC can better solve some problems that cannot be solved by SMC, and vice versa.
- BMC cannot prove the absence of errors in large cases.
  - May require a very large bound $k$. 
Check if the circuit satisfies $AG\neg q$.

Initial state: $x=0$, $y=0$. 
Circuit State after Cycle 1

\[ w^0 = 1 \]
\[ y^0 = 0 \]
\[ x^0 = 0 \]
\[ q^0 = 0 \]
\[ y^1 = 1 \]
\[ x^1 = 0 \]
Circuit State after Cycle 2

$q = 1$ if $w = 1$ in cycle 1 and $w = 0$ in cycle 2.

A counter-example to $\text{AG}\neg q$ is a three state sequence.
Big Picture of Bounded Model Checking

Comb. Logic

Comb. Logic

Comb. Logic

Comb. Logic

I

I

I

I

S

S

S

S

O

O

O

O
How BMC Works

$k = 0$

**Stop**

- **Resource exhausted**

**BMC**(\(M, \neg f, k\))

**SAT**

- **fail**

**UNSAT**

- **k++**

**k \geq CT**

**Yes**

- **pass**
A $k$-bounded path is a sequence of $k$ state transitions.

A finite path is infinite if it has a back loop.

A $(k, l)$-loop is a $k$-bounded path $\rho$ such that $R(s_k, s_l)$ holds.

A path $\rho$ is a $k$-loop if there exists $0 \leq l \leq k$ such that $\rho$ is a $(k, l)$-loop.
Bounded Semantics of LTL Formulas

Let $\rho |\models_k f$ denote the truth of the LTL formula $f$ over the $k-$bounded path $\rho$.

Evaluate $f$ only in the first $k + 1$ states on $\rho$.

Let $\rho(i)$ denote the $i^{th}$ state on $\rho$.

Let $\rho |\models^i_k f$ denote the truth of $f$ over the path from state $\rho(i)$ to $\rho(k)$.

If a path $\rho$ is a $k-$loop,

$$\rho |\models_k f \iff \rho |\models f$$
Bounded Semantics of LTL Formulas (2)

\( \rho \models_k f \iff \rho \models^0_k \) where

\( \rho \models^i_k p \iff p \in L(\rho(i)) \)

\( \rho \models^i_k \neg p \iff p \notin L(\rho(i)) \)

\( \rho \models^i_k f \land g \iff \rho \models^i_k f \text{ and } \rho \models^i_k g \)

\( \rho \models^i_k f \lor g \iff \rho \models^i_k f \text{ or } \rho \models^i_k g \)

\( \rho \models^i_k G f \iff \text{false} \)

\( \rho \models^i_k F f \iff \exists i \leq j \leq k. \rho \models^j_k f \)

\( \rho \models^i_k X f \iff i < k \text{ and } \rho \models^{i+1}_k f \)

\( \rho \models^i_k f U g \iff \exists i \leq j \leq k. \rho \models^j_k f \text{ and } \forall i \leq n \leq j. \rho \models^n_k f \)
Let $M \models_k f$ denote a $k$–bounded model checking problem for the LTL formula $f$.

Formula $f$ is evaluated on all $k$–bounded path.

Let $f$ be a LTL formula and $\rho$ a path. $\rho \models_k f \Rightarrow \rho \models f$.

If for each $\rho$ in $M$ such that $\rho \models_k f$, then $M \models f$ holds.

If there is a $\rho$ in $M$ such that $\rho \models_k f$, then $M \models \neg f$ does not hold.

$M \models f \iff \exists k \geq 0. M \models_k f$.

There always exists a $k$ such that the result of bounded model checking is equivalent to that of the complete one.
An BMC Example

- $M \models \neg (a \land b)$.
- BMC checks if there is a bounded path on which $F(a \land b)$ holds.
An BMC Example (2)

\[ M \models_{k=1} G \neg (a \land b). \]

\[ M \models_{k=2} G \neg (a \land b). \]

\[ k = 1 \]
\[ k = 2 \]
Bounded Model Checking: Overview

System Model

Specification in LTL

BMC

Propositional formula

SAT Solver

Specification violation

Satisfied

Specification holds

Unsatisfied
Boolean Encoding of Bounded Model Checking

Given a $M$, an LTL formula $f$ and a bound $k$, generate a Boolean formula $[[M, f]]_k$.

Encoding the constraints on $k$–paths in $M$ such that these $k$–paths, if satisfiable, are witnesses of $f$.

Three components of $[[M, f]]_k$:

- $[[M]]_k$: all $k$–paths in $M$.
- $[[f]]_k$: encoding of $f$ on $k$–paths.
- $l[[f]]_k$: encoding of $f$ on $k$–loops.
Encoding of $[M]_k$

Unfolding of the transition relation

$$[M]_k = I(s_0) \land \bigwedge_{i=0}^{k-1} R(s_i, s_{i+1}).$$

$$\bigwedge_{i=0}^{k-1} R(s_i, s_{i+1}) : \text{encoding all } k\text{-paths of } M.$$

$I(s_0) : \text{constraints on the } k\text{-paths from the initial states.}$
Encoding of $f$ on $k$-Paths $[f]_k$

- **Inductive case:** $\forall i \leq k$

  \[
  [p]_k^i \equiv p(s_i) \\
  [f \land g]_k^i \equiv [f]_k^i \land [g]_k^i \\
  [Gf]_k^i \equiv [f]_k^i \land [Gg]_k^{i+1} \\
  [f \lor g]_k^i \equiv [f]_k^i \lor [g]_k^i \\
  [Ff]_k^i \equiv [f]_k^i \lor [Ff]_k^{i+1} \\
  [f \lor g]_k^i \equiv [f]_k^i \lor ( [f]_k^i \land [f \lor g]_k^{i+1}) \\
  [Xf]_k^i \equiv [f]_k^{i+1}
  \]

- **Base case:**

  \[
  [f]_k^{k+1} \equiv \text{false}
  \]
Encoding of $f$ on $k$–Loops $l[f]_k$

For $k, l, i \geq 0$ and $l, i \leq k$

$$l[p]^i_k \equiv p(s_i)$$

$$l[f \land g]^i_k \equiv l[f]^i_k \land l[g]^i_k$$

$$l[Gf]^i_k \equiv l[f]^i_k \land l[Gg]^{\text{succ}(i)}_k$$

$$l[f \mathcal{U} g]^i_k \equiv l[g]^i_k \lor (l[f]^i_k \land l[f \mathcal{U} g]^{\text{succ}(i)}_k)$$

$$l[-p]^i_k \equiv \neg p(s_i)$$

$$l[f \lor g]^i_k \equiv l[f]^i_k \lor l[g]^i_k$$

$$l[Ff]^i_k \equiv l[f]^i_k \lor l[Ff]^{\text{succ}(i)}_k$$

$$l[Xf]^i_k \equiv l[f]^{\text{succ}(i)}_k$$

where $\text{succ}(i)$ is defined as follows.

$$\text{succ}(i) = \begin{cases} 
  i + 1 & \text{if } i < k \\
  l & \text{if } i = k 
\end{cases}$$
Encoding of the Looping Conditions

A loop forms if there is a transition from $s_k$ back to $s_i$ for $i \leq k$.

There are $k$ states $s_i$, the looping condition of $k$-path is a disjunction of $k$ different looping conditions.

$$L_k = \bigvee_{i=0}^{k} iL_k$$

where

$$iL_k = R(s_k, s_i).$$
Let $M$ be a Kripke structure, $f$ an LTL formula, and $k \geq 0$ a bound.

$$[[M, f]]_k = [[M]]_k \land \left( (\neg L_k \land [[f]]_k^0) \land \bigvee_{i=0}^{k} (L_k \land [[f]]_k^0) \right)$$

$[[M, f]]_k$ is satisfiable if, and only if, $M \models_k E(f)$.

$[[M, f]]_k$ is satisfiable if, and only if, $M \models_k \neg f$ does not hold.
An BMC Example: Translation

- $M \models G\neg(a \land b)$ for $k = 2$.
- $M = (I, R)$ where

\[
I = \neg a \land \neg b \\
R = (\neg a \land \neg b \land a' \land \neg b') \lor (\neg a \land \neg b \land \neg a' \land b') \lor \\
(\neg a \land b \land \neg a' \land \neg b') \lor (a \land \neg b \land \neg a' \land \neg b') \lor \\
(a \land \neg b \land a' \land b') \lor (a \land b \land \neg a' \land \neg b')
\]