Formal Methods
Lecture VII

Symbolic Model Checking

Alessandro Artale

Faculty of Computer Science – Free University of Bozen-Bolzano
artale@inf.unibz.it http://www.inf.unibz.it/∼artale/

Academic Year: 2006/07

Some material (text, figures) displayed in these slides is courtesy of:
M. Benerecetti, A. Cimatti, M. Fisher, F. Giunchiglia, M. Pistore, M. Roveri, R.Sebastiani.

1. Representing Set of States as OBDD's

2. Symbolic Model-Checking Algorithm

Main Ideas

- OBDD's allow systems with a large state space to be verified.
- The Labeling algorithm takes a CTL formula and returns a **set of states** manipulating intermediate **set of states**.
- The algorithm is changed by storing set of states as OBDD's and then manipulating them.
- Model checking using OBDD's is called **Symbolic Model Checking**.

Symbolic Representation of States

Example:

- Three state variables x_1, x_2, x_3:
  $\{000, 001, 010, 011\}$ represented as "first bit false": $\neg x_1$
- With five state variables x_1, x_2, x_3, x_4, x_5:
  $\{00000, 00001, 00010, 00011, 00100, 00101, 00110,$
  $00111, \ldots, 01111\}$ still represented as "first bit false": $\neg x_1$

Symbolic Representation of States (Cont.)

- Let $M = (S, I, R, L, AP)$ be a Kripke structure
- States $s \in S$ are described by means of a vector $V = (v_1, v_2, \ldots, v_n)$ of boolean values: One for each $x_i \in AP$.
  - A **state**, s, is a **truth assignment** to each variable in AP such that $v_i = 1$ iff $x_i \in L(s)$.
  - **Example**: **0100** represents the state s where only $x_2 \in L(s)$.

Symbolic Representation of States (Cont.)

- Boolean vectors can be represented by boolean formulas
  - **Example**: **0100** can be represented by the formula $\xi(s) = (\neg x_1 \wedge x_2 \wedge \neg x_3 \wedge \neg x_4)$
- We call $\xi(s)$ the formula representing the state $s \in S$ (Intuition: $\xi(s)$ holds iff the system is in the state s)
- A **set of states**, $Q \subseteq S$, can be represented by the formula – **Characteristic Function of Q**:

$$\xi(\mathbf{Q}) = \bigvee_{\mathbf{s} \in \mathbf{Q}} \xi(\mathbf{s})$$

- Thus, (set of) states can be encoded as OBDD's!

Remark

- Any propositional formula is a (typically very compact) representation of the set of assignments satisfying it
- **Any formula equivalent to $\xi(Q)$ is a representation of Q**
  $\Rightarrow$ Typically Q can be encoded by much smaller formulas than $\bigvee_{s \in Q} \xi(s)$!
- **Example**: $Q = \{00000, 00001, 00010, 00011, 00100, 00101, 00110, 00111, \ldots, 01111\}$ represented as "first bit false": $\neg x_1$

$$\bigvee_{s \in Q} \xi(s) = \left.\begin{array}{l} (\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4 \wedge \neg x_5) \vee \\ (\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4 \wedge x_5) \vee \\ (\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge x_4 \wedge \neg x_5) \vee \\ \ldots \\ (\neg x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5) \end{array}\right\} 2^4 \textit{disjuncts}$$

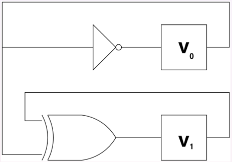
Symbolic Representation of Transitions

- The transition relation R is a set of pairs of states: $R \subseteq S \times S$.
- Then, a single transition is a pair of states $(\mathbf{s}, \mathbf{s}')$.
- A new vector of variables V' (the **next state vector**) represents the value of variables after the transition has occurred.
- $\xi(s, s')$ **defined as** $\xi(s) \wedge \xi(s')$.
- The transition relation R can be (naively) represented by

$$\bigvee_{(s,s')\in R} \xi(s, s') = \bigvee_{(s,s')\in R} \xi(s) \wedge \xi(s')$$

Remark

- **Any formula equivalent to $\xi(R)$ is a representation of R**
  $\Rightarrow$ Typically R can be encoded by a much smaller formula than $\bigvee_{(s,s')\in R}\xi(s)\wedge\xi(s')$!
- **Example**: a synchronous sequential circuit



v_1	v_0	v_1'	v_0'
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

$$
\begin{aligned}
\xi(R) &= (v_0' \Leftrightarrow \neg v_0) \wedge (v_1' \Leftrightarrow v_0 \bigoplus v_1)\\
\bigvee_{(s,s')\in R}\xi(s)\wedge\xi(s') &= (\neg v_0 \wedge \neg v_1 \wedge v_0' \wedge \neg v_1') \vee\\
&\quad (v_0 \wedge \neg v_1 \wedge \neg v_0' \wedge v_1') \vee\\
&\quad (\neg v_0 \wedge v_1 \wedge v_0' \wedge v_1') \vee\\
&\quad (v_0 \wedge v_1 \wedge \neg v_0' \wedge \neg v_1')
\end{aligned}
$$

Summary

- Representing Set of States as OBDD's.
- **Symbolic Model-Checking Algorithm.**

Intro

Problem: $M \models \varphi$?

- Let $M = \langle S, I, R, L, AP \rangle$ be a Kripke structure and φ be a CTL formula.
- The Symbolic Model-Checking algorithm is a Labeling algorithm that makes use of OBDD.
- It is implemented by a recursive procedure CHECK with:
  - **Input:** φ, the formula to be checked;
  - **Output:** B_φ, the OBDD representing the states satisfying φ.

Intro (Cont.)

- To check whether $I \subseteq [\![\varphi]\!]$:

$$(\mathbf{B_I} \Rightarrow \mathbf{B}_\varphi) \equiv \mathbf{B}_\top$$

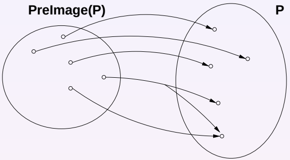
i.e.,

$$\text{APPLY}(\Rightarrow, \mathbf{B_I}, \mathbf{B}_\varphi) \equiv \mathbf{B}_\top$$

- To compute OBDD's for CTL formulas we need to understand how to compute them in case of the temporal operators: $\Diamond\!\!\!\!\!\scriptstyle{P}\;\; \bigcirc$, $\Diamond\!\!\!\!\!\scriptstyle{P}\;\; \mathcal{U}$, $\Diamond\!\!\!\!\!\scriptstyle{P}\;\; \Box$.

PreImage

- Backward (pre) image of a set:



- Evaluate all transitions ending in the states of the set.
- Set theoretic view:

$$\mathbf{PreImage}(\mathbf{P}, \mathbf{R}) := \{\mathbf{s} \in \mathbf{S} \mid \exists \mathbf{s}'.(\mathbf{s}, \mathbf{s}') \in \mathbf{R} \text{ and } \mathbf{s}' \in \mathbf{P}\}$$

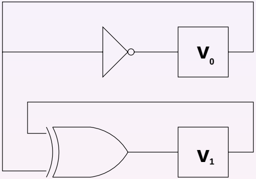
- Logical Characterization:

$$\xi(\mathbf{PreImage}(\mathbf{P}, \mathbf{R})) \ := \ \exists \mathbf{V}'.(\xi(\mathbf{P})[\mathbf{V}'] \wedge \xi(\mathbf{R})[\mathbf{V}, \mathbf{V}'])$$

- N.B.: quantification over propositional variables

PreImage: An Example

- **Example**: A synchronous sequential circuit



v_1	v_0	v_1'	v_0'
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

$\xi(R) = (v_0' \Leftrightarrow \neg v_0) \wedge (v_1' \Leftrightarrow v_0 \bigoplus v_1)$
$\xi(P) := (v_0 \Leftrightarrow v_1)$ (i.e., $P = \{00, 11\}$)

- Pre Image:

$$\begin{aligned} \xi(PreImage(P,R)) &= \exists V'.(\xi(\mathbf{P})[\mathbf{V'}] \wedge \xi(\mathbf{R})[\mathbf{V},\mathbf{V'}]) \\ &= \exists V'.((\mathbf{v_0'} \Leftrightarrow \mathbf{v_1'}) \wedge (\mathbf{v_0'} \Leftrightarrow \neg \mathbf{v_0}) \wedge (\mathbf{v_1'} \Leftrightarrow \mathbf{v_0} \bigoplus \mathbf{v_1})) \end{aligned}$$

OBDD for PreImages

- $B_{\langle P \rangle \bigcirc \varphi}$, the OBDD for $\langle P \rangle \bigcirc \varphi$, is computed starting from the OBDD's for both φ, B_φ, and the transition relation, B_R.
- $\xi(\mathbf{PreImage}(\varphi, \mathbf{R})) := \exists \mathbf{V}'.(\xi(\varphi)[\mathbf{V}'] \wedge \xi(\mathbf{R})[\mathbf{V}, \mathbf{V}'])$, then:
  1. Rename the variable in B_φ to their primed version, $B_{\varphi'}$
  2. Compute $B_{(\varphi' \wedge R)} = \textsc{apply}(\wedge, B_{\varphi'}, B_R)$;
  3. $B_{\langle P \rangle \bigcirc \varphi}$ is a sequence of:

     $$\textsc{apply}(\vee, \textsc{restrict}(0, x_i', B_{(\varphi' \wedge R)}), \textsc{restrict}(1, x_i', B_{(\varphi' \wedge R)}))$$

     where $x_i' \in V'$
- We call $\textsc{Pre}(B_\varphi)$ the procedure that computes $B_{\langle P \rangle \bigcirc \varphi}$.

The $\textsc{Check}$ Symbolic M.C. Algorithm

$\textsc{Check}(\varphi)$ {
case φ **of**

true:	**return** $B_\top$;
false:	**return** $B_\bot$;
an atom x_i:	**return** B_{x_i};
$\neg\varphi_1$:	**return** $\textsc{Invert}(\textsc{Check}(\varphi_1))$;
$\varphi_1 \wedge \varphi_2$:	**return** $\textsc{Apply}(\wedge, \textsc{Check}(\varphi_1), \textsc{Check}(\varphi_2))$;
$\Diamond_P \bigcirc \varphi_1$:	**return** $\textsc{Pre}(\textsc{Check}(\varphi_1))$;
$\Diamond_P (\varphi_1 \mathcal{U} \varphi_2)$:	**return** $\textsc{Check\_EU}(\textsc{Check}(\varphi_1), \textsc{Check}(\varphi_2))$;
$\Diamond_P \Box \varphi_1$:	**return** $\textsc{Check\_EG}(\textsc{Check}(\varphi_1))$;

}

CHECK_EG

$$[\![\diamondsuit_P \, \Box \varphi]\!] = [\![\varphi]\!] \cap \mathrm{PRE}([\![\diamondsuit_P \, \Box \varphi]\!])$$

```
CHECK_EG(B_φ){
    var X, OLD-X;
    X := B_φ;
    OLD-X := B_⊥;
    while X ≠ OLD-X
    begin
        OLD-X := X;
        X := APPLY(∧, X, PRE(X))
    end
    return X
}
```

Check_EU

$$[\![\Diamond_{P} (\varphi\, \mathcal{U} \psi)]\!] = [\![\psi]\!] \cup ([\![\varphi]\!] \cap \textsc{Pre}([\![\Diamond_{P} (\varphi\, \mathcal{U} \psi)]\!]))$$

$\textsc{Check\_EU}(B_\varphi, B_\psi)\{$
  var $X, \mathit{OLD\text{-}X}$;
  $X := B_\psi$;
  $\mathit{OLD\text{-}X} := B_\top$;
  while $X \neq \mathit{OLD\text{-}X}$
  begin
    $\mathit{OLD\text{-}X} := X$;
    $X := \textsc{Apply}(\vee, X, \textsc{Apply}(\wedge, B_\varphi, \textsc{Pre}(X)))$
  end
  return X
$\}$

CTL Symbolic Model Checking–Summary

- Based on fixed point CTL M.C. algorithms
- Kripke structure encoded as boolean formulas (OBDDs)
- All operations handled as (quantified) boolean operations
- **Avoids building the state graph explicitly**
- Reduces dramatically the state explosion problem
  $\Rightarrow$ problems of up to 10^{120} states handled!!

Partitioned Transition Relations

- There may be significant efficiency problems:
  - The transition relation may be too large to construct
  - Intermediate OBDDs may be too large to handle.
- IDEA: Partition conjunctively the transition relation:

$$\mathbf{R(V, V') \leftrightarrow \bigwedge_i R_i(V_i, V'_i)}$$

- Trade one "big" quantification for a sequence of "smaller" quantifications
  - $\exists \mathbf{V}'_1 \ldots \mathbf{V}'_n.(\mathbf{R}_1(\mathbf{V}_1, \mathbf{V}'_1) \land \ldots \land \mathbf{R}_n(\mathbf{V}_n, \mathbf{V}'_n) \land \mathbf{Q}(\mathbf{V}'))$
    by pushing quantifications inward can be reduced to
  - $\exists \mathbf{V}'_1.(\mathbf{R}_1(\mathbf{V}_1, \mathbf{V}'_1) \land \ldots \land \exists \mathbf{V}'_n(\mathbf{R}_n(\mathbf{V}_n, \mathbf{V}'_n) \land \mathbf{Q}(\mathbf{V}')))$
    which is typically much smaller

Symbolic Model Checkers

- **Most hardware design companies have their own Symbolic Model Checker(s)**
  - Intel, IBM, Motorola, Siemens, ST, Cadence, ...
  - very advanced tools
  - proprietary technolgy!
- **On the academic side**
  - CMU SMV [McMillan]
  - VIS [Berkeley, Colorado]
  - Bwolen Yang's SMV [CMU]
  - NuSMV [CMU, IRST, UNITN, UNIGE]
  - ...

Summary of Lecture VII

- Representing Set of States as OBDD's.
- Symbolic Model-Checking Algorithm.