Model Checking
Lecture #18: Reduced Ordered Binary Decision Diagrams
[Baier & Katoen, Chapter 6.7]

Joost-Pieter Katoen
Software Modeling and Verification Group
Model Checking Course, RWTH Aachen, WiSe 2019/2020

Overview

1 Motivation
2 Switching Functions
3 Ordered Binary Decision Diagrams
4 Reduced Ordered Binary Decision Diagrams
5 Summary

State Spaces Can Be Gigantic

A model of the Hubble telescope
Treating Gigantic Models?

- Use compact data structures
- Make models smaller prior to (or: during) model checking
- Try to make them even smaller
- If possible, try to obtain the smallest possible model
- While preserving the properties of interest
- Do this all algorithmically and possibly fast

Symbolic CTL Model Checking

- Explicit representation of transition system: state explosion problem
- Idea: reformulate model-checking in a symbolic way
- Concept: represent sets of states and transitions symbolically
- Approach: binary encoding of states + switching functions for sets
- Compactly represent switching functions by binary decision diagrams
- Alternative: conjunctive normal form (used in SAT-based model checking)

Basic Approach

- let $TS = (S, \rightarrow, I, AP, L)$ be a “large” finite transition system
  - the set of actions is irrelevant here and is omitted, i.e., $\rightarrow \subseteq S \times S$

- For $n \geq \lceil \log |S| \rceil$, let injective function $enc : S \rightarrow \{0, 1\}^n$
  - the encoding of the states by bit vectors of length $n$
  - elements in $\{0, 1\}^n \setminus enc(S)$ encode unreachable pseudo states

- Identify the states $s \in S = enc^{-1}(\{0, 1\}^n)$ with $enc(s) \in \{0, 1\}^n$

- And $T \subseteq S$ by its characteristic function $\chi_T : \{0, 1\}^n \rightarrow \{0, 1\}$
  - $\chi_T(enc(s)) = 1$ if and only if $s \in T$

- And $\rightarrow \subseteq S \times S$ by the Boolean function $\Delta : \{0, 1\}^{2n} \rightarrow \{0, 1\}$
  - such that $\Delta\{enc(s), enc(s')\} = 1$ if and only if $s \rightarrow s'$

Symbolic Representation of Transition System

Function: $\Delta(x_1, x_2, x'_1, x'_2) = 1$ if and only if $s \rightarrow s'$

$$\Delta(x_1, x_2, x'_1, x'_2) = (\neg x_1 \land \neg x_2 \land \neg x'_1 \land x'_2) \lor (\neg x_1 \land \neg x_2 \land x'_1 \land x'_2) \lor (\neg x_1 \land x_2 \land x'_1 \land \neg x'_2) \lor \ldots \lor (x_1 \land x_2 \land x'_1 \land x'_2)$$
Overview

Motivation

Switching Functions

Ordered Binary Decision Diagrams

Reduced Ordered Binary Decision Diagrams

Summary

Switching Functions

Let $\text{Var} = \{z_1, \ldots, z_m\}$ be a finite set of Boolean variables, $m \geq 0$

An evaluation is a function $\eta : \text{Var} \to \{0, 1\}$

shorthand $[z_1 = b_1, \ldots, z_m = b_m]$ for $\eta(z_1) = b_1, \ldots, \eta(z_m) = b_m$

Let $\text{Eval}(\text{Var})$ denote the set of all evaluations for $\text{Var} = z_1, \ldots, z_m$

$f : \text{Eval}(\text{Var}) \to \{0, 1\}$ is a switching function for $\text{Var} = \{z_1, \ldots, z_m\}$

Logical operations and quantification are defined as:

- $f_1(\cdot) \land f_2(\cdot) = \min\{f_1(\cdot), f_2(\cdot)\}$
- $f_1(\cdot) \lor f_2(\cdot) = \max\{f_1(\cdot), f_2(\cdot)\}$
- $\exists z. f(\cdot) = f(\cdot)|_{z=0} \lor f(\cdot)|_{z=1}$, and
- $\forall z. f(\cdot) = f(\cdot)|_{z=0} \land f(\cdot)|_{z=1}$

Impossible Polynomial Data Structure

There is no polynomial-size data structure for all switching functions with $|\text{Eval}(z_1, \ldots, z_m)| = 2^n$; i.e., the number of switching functions is $2^{2^m}$.

Proof.

Suppose there is a data structure that can represent $K_m$ switching functions by at most $2^{m-1}$ bits.

Then $K_m \leq \sum_{i=0}^{2^m-1} 2^i = 2^{2^m-1} - 1 < 2^{2^m+1}$

But then there are at least

$$2^n - 2^{2^m-1} = 2^{2^m-1} \cdot 2^{2^m-1} - 1 = 2^{2^m+1} \cdot 2^{2^m-1} - 1$$

switching functions whose representation needs more than $2^{m-1}$ bits.

Representing Switching Functions

- Truth tables
  - very space inefficient: $2^n$ entries for $n$ variables
  - satisfiability and equivalence check: easy; boolean operations also easy
  - ... but have to consider exponentially many lines (so are hard)

- Disjunctive Normal Form (DNF)
  - satisfiability is easy: find a disjunct with complementary literals
  - negation and conjunction complicated
  - equivalence checking ($f = g$?) is coNP-complete

- Conjunctive Normal Form (CNF)
  - satisfiability problem is NP-complete (Cook’s theorem)
  - negation and disjunction complicated
Representing Switching Functions

<table>
<thead>
<tr>
<th>representation</th>
<th>compact?</th>
<th>sat</th>
<th>equiv</th>
</tr>
</thead>
<tbody>
<tr>
<td>propositional formula</td>
<td>often</td>
<td>hard</td>
<td>hard</td>
</tr>
<tr>
<td>DNF</td>
<td>sometimes</td>
<td>easy</td>
<td>hard</td>
</tr>
<tr>
<td>CNF</td>
<td>sometimes</td>
<td>hard</td>
<td>easy</td>
</tr>
<tr>
<td>(ordered) truth table</td>
<td>never</td>
<td>hard</td>
<td>hard</td>
</tr>
<tr>
<td>reduced ordered binary decision diagram</td>
<td>often</td>
<td>easy</td>
<td>easy</td>
</tr>
</tbody>
</table>

* provided appropriate implementation techniques are used

Binary Decision Tree

- The BDT for function $f$ on $\text{Var} = \{ z_1, \ldots, z_m \}$ has depth $m$
  - outgoing edges for node at level $i$ stand for $z_i = 0$ (dashed) and $z_i = 1$ (solid)
- For evaluation $s = [z_1 = b_1, \ldots, z_m = b_m]$, $f(s)$ is the value of the leaf
  - reached by traversing the BDT from the root using branch $z_i = b_i$ at level $i$
- The sub-tree of node $v$ at level $i$ for variable ordering $z_1 < \ldots < z_m$ represents
  $$f_v = f|_{z_1=b_1,\ldots,z_{i-1}=b_{i-1}}$$
  - which is a switching function over $\{ z_i, \ldots, z_m \}$
  - where $z_1 = b_1, \ldots, z_{i-1} = b_{i-1}$ is the sequence of decisions made along the path from the root to node $v$
Symbolic Representation of Transition System

Switching function: \( \Delta(x_1, x_2, x'_1, x'_2) = 1 \) if and only if \( s \to s' \)

\[
\Delta(x_1, x_2, x'_1, x'_2) = \\
(\neg x_1 \land \neg x_2 \land \neg x'_1 \land x'_2) \\
\lor (\neg x_1 \land \neg x_2 \land x'_1 \land x'_2) \\
\lor (\neg x_1 \land x_2 \land x'_1 \land \neg x'_2) \\
\lor \ldots \\
\lor (x_1 \land x_2 \land x'_1 \land x'_2)
\]

Facts About BDTs

- BDTs are not compact
  - a BDT for switching function \( f \) on \( n \) variables has \( 2^n \) leafs
  - they are as space inefficient as truth tables!

- BDTs contain quite some redundancy
  - all leafs with value one (zero) could be collapsed into a single leaf
  - a similar scheme could be adopted for isomorphic subtrees

- The size of a BDT does not change if the variable order changes

Transition Relation as a BDT

A BDT representing \( \Delta \) for our example using ordering \( x_1 < x_2 < x'_1 < x'_2 \)
Ordered Binary Decision Diagram

- OBDDs rely on compactifying BDT representations
- Idea: skip redundant fragments of BDT representations
- Collapse sub-trees with all terminals having same value
- Identify nodes with isomorphic sub-trees
- This yields directed acyclic graphs with out-degree two
- Inner nodes are labeled with variables
- Leaf nodes are labeled with function values (zero and one)

Definition: Ordered BDDs

Let $\mathcal{p} = (z_1, \ldots, z_m)$ be a (total) variable ordering for $\text{Var} = \{z_1, \ldots, z_m\}$, where, i.e., $z_1 \prec_\mathcal{p} \cdots \prec_\mathcal{p} z_m$.

An $\mathcal{p}$-OBDD is a tuple $\mathcal{B} = (V, V_I, V_T, \text{succ}_0, \text{succ}_1, \text{var}, \text{val}, v_0)$ with:

- A finite set $V$ of nodes, partitioned into $V_I$ (inner) and $V_T$ (terminals)
- And a distinguished root (node) $v_0 \in V_I$
- Successor functions $\text{succ}_0, \text{succ}_1 : V_I \to V$
- Such that each node $v \in V \setminus \{v_0\}$ has at least one predecessor
- I.e., all nodes of the OBDD $\mathcal{B}$ are reachable from the root
- Labeling functions $\text{var} : V_I \to \text{Var}$ and $\text{val} : V_T \to \{0, 1\}$ satisfying for $v \in V_I$ and $w \in \{\text{succ}_0(v), \text{succ}_1(v)\}$:

$$\text{var}(v) = z_i \land w \in V_I \Rightarrow \text{var}(w) = z_j \text{ with } z_i \prec_\mathcal{p} z_j$$

Example: Transition Relation as OBDD

Example OBDD representing $f_\to$ for our example with $x_1 \prec_\mathcal{p} x_2 \prec_\mathcal{p} x'_1 \prec_\mathcal{p} x'_2$
OBDD Semantics

**Definition: OBDD semantics**

The semantics of \( \mathcal{P}-\text{OBDD} \ B \) is the switching function \( f_B \) where

\[
f_B([z_1 = b_1, \ldots, z_m = b_m])
\]

is the value of the resulting leaf when traversing \( B \) starting in \( v_0 \) and branching according to the evaluation \([z_1 = b_1, \ldots, z_m = b_m]\).

Intermezzo: OBDDs and DFA

Each OBDD \( B \) is a deterministic finite-state automaton \( \mathcal{A}_B \) with \( f_B^{-1}(1) = L(\mathcal{A}_B) \).

Consistent Co-factors in OBDDs

**Definition: consistent co-factors**

Let \( f \) be a switching function for \( \text{Var} \) and let \( \mathcal{P} = (z_1, \ldots, z_m) \) be a variable ordering for \( \text{Var} \), i.e., \( z_1 \preceq_p \ldots \preceq_p z_m \).

The switching function \( g \) is a \( \mathcal{P} \)-consistent cofactor of \( f_B \) if

\[
g = f \big|_{z_1 = b_1, \ldots, z_i = b_i} \quad \text{for some} \ i \in \{0, 1, \ldots, m\}.
\]

It holds that:

\begin{itemize}
  \item for each node \( v \) of a \( \mathcal{P} \)-OBDD \( B \), \( f_v \) is a \( \mathcal{P} \)-consistent cofactor of \( f_B \)
  \item for each \( \mathcal{P} \)-consistent cofactor \( g \) of \( f_B \), \( f_v = g \) for some node \( v \in B \)
\end{itemize}
Overview

1. Motivation
2. Switching Functions
3. Ordered Binary Decision Diagrams
4. Reduced Ordered Binary Decision Diagrams
5. Summary

Reduced Ordered Binary Decision Diagrams

Definition: reduced OBDD

A \( \wp \)-OBDD \( \mathcal{B} \) is reduced if for every pair \((v, w)\) of nodes in \( \mathcal{B} \) it holds:

\[ v \neq w \implies f_v \neq f_w. \]

A reduced \( \wp \)-OBDD is abbreviated as \( \wp \)-ROBDD.

In \( \wp \)-ROBDDs every \( \wp \)-consistent cofactor is represented by exactly one node.

Example Reduced OBDDs

Example: Transition Relation as OBDD

An example OBDD representing \( f_\rightarrow \) for our example using \( x_1 < x_2 < x'_1 < x'_2 \).
Example: Transition Relation as ROBDD

Universality and Canonicity Theorem

\[\text{[Fortune, Hopcroft & Schmidt, 1978]}\]

For finite set \(\text{Var}\) of Boolean variables and variable ordering \(p\) for \(\text{Var}\):

(a) For each switching function \(f\) on \(\text{Var}\), \(f = f_B\) for some \(p\)-ROBDD \(B\).

(b) For any \(p\)-ROBDDs \(B\) and \(C\) with \(f_B = f_C\), \(B\) and \(C\) are isomorphic\(^1\).

\(^1\)agree up to renaming of nodes

The Importance of Canonicity

- **Absence of redundant vertices**
  - if \(f_B\) does not depend on \(x_i\), ROBDD \(B\) does not contain an \(x_i\) node

- Test for **equivalence**: \(f(x_1, \ldots, x_n) \equiv g(x_1, \ldots, x_n)\)?
  - generate ROBDDs \(B_f\) and \(B_g\), and check isomorphism

- Test for **validity**: for all \(x_1, \ldots, x_n\), is \(f(x_1, \ldots, x_n) = 1\)?
  - generate ROBDD \(B_f\) and check whether it only consists of a 1-leaf

- Test for **implication**: \(f(x_1, \ldots, x_n) \rightarrow g(x_1, \ldots, x_n)\)?
  - generate ROBDD \(B_f \land \neg g\) and check if it just consists of a 0-leaf

- Test for **satisfiability**
  - \(f\) is satisfiable if and only if \(B_f\) has a reachable 1-leaf
Reduced Ordered Binary Decision Diagrams

Minimality of ROBDDs

Let $\mathcal{B}$ be an $\mathcal{p}$-OBDD for $f$. Then: $\mathcal{B}$ is reduced iff $\text{size}(\mathcal{B}) \leq \text{size}(\mathcal{C})$ for each $\mathcal{p}$-OBDD $\mathcal{C}$ for $f$.

**Proof.**

This follows from the fact that:

1. Each $\mathcal{p}$-consistent cofactor of $f$ is represented in any $\mathcal{p}$-OBDD for $f$ by at least one node, and
2. A $\mathcal{p}$-OBDD $\mathcal{B}$ for $f$ is reduced iff there is a 1-to-1 correspondence between the nodes in $\mathcal{B}$ and the $\mathcal{p}$-consistent cofactors of $\mathcal{B}$.

Reducing OBDDs

- Generate an OBDD (or BDT) for a boolean expression, then reduce by means of a recursive descent over the OBDD.
  - Elimination of duplicate leaves
    - For a duplicate 0-leaf (or 1-leaf), redirect all in-edges to just one of them.
  - Elimination of “don’t care” (non-leaf) vertices
    - If $\text{succ}_0(v) = \text{succ}_1(v) = w$, delete $v$ and redirect all its in-edges to $w$.
  - Elimination of isomorphic subtrees
    - If $v \neq w$ are roots of isomorphic subtrees, remove $w$ and redirect all incoming edges to $w$ to $v$.

How to Reduce an OBDD?

(special case of) isomorphism rule

isomorphism rule
How to Reduce an OBDD?

![Diagram showing reduction from OBDD to another OBDD]

elimination rule

Soundness:

if \( C \) arises from a \( \mathcal{P} \)-OBDD \( B \) by the elimination or isomorphism rule, then \( C \) is a \( \mathcal{P} \)-OBDD with \( f_B = f_C \).

Proof:

Elimination rule for \( v \) with \( \text{var}(v) = z \), and \( w = \text{succ}_0(v) = \text{succ}_1(v) \):

\[
f_v = (\neg z \land f_{\text{succ}_0(v)}) \lor (z \land f_{\text{succ}_1(v)}) = (\neg z \land f_w) \lor (z \land f_w) = f_w
\]

Isomorphism rule for \( v, w \) with \( \text{var}(v) = \text{var}(w) = z \) yields:

\[
f_v = (\neg z \land f_{\text{succ}_0(v)}) \lor (z \land f_{\text{succ}_1(v)}) = (\neg z \land f_{\text{succ}_0(w)}) \lor (z \land f_{\text{succ}_1(w)}) = f_w
\]

as each reduction rule decreases the \# nodes, repeatedly applying them terminates

Completeness of Reduction Rules:

\( \mathcal{P} \)-OBDD \( B \) is reduced iff no reduction rule is applicable to \( B \).
ROBDDs are directed acyclic graphs aimed at succinctly representing switching functions.

- They provide a compact representation for many switching functions.
- In an ROBDD, each co-factor is represented by exactly one node.
- ROBDDs are canonical for a given variable ordering.
- Any OBDD can be reduced by two reduction rules (applied in any order).