Using BDDs to Implement Propositional Modal Tableaux

Kerry Olesen

Supervisors:
Rajeev Goré
Jimmy Thomson

November 5, 2013
Automated Reasoning

Designing computer programs that can reason about highly complex formal systems.
Take a logical encoding of a system $\Gamma$, and some property $\varphi$. Determine whether $\varphi$ is a logical consequence of $\Gamma$.
For example, say you’ve designed a digital circuit, and now you want to verify that your circuit actually does what you think it does. That system of logic gates is hugely complex. Also formal, well-defined system. Prime example of where automated reasoning is used today.
Using BDDs to Implement Propositional Modal Tableaux

- **Task:** Investigate the viability of a novel data structure for implementing an automated reasoner.
- **Data structure:** Binary Decision Diagrams
- **Method:** The tableau method
- **Logic:** Propositional Modal Logic
Outline

1. Propositional Modal Logic
2. Tableau
3. Binary Decision Diagrams
4. Implementation
5. Results
6. Questions
Propositional Modal Logic

Syntax:

\[ p ::= p_0 \mid p_1 \mid p_2 \mid \ldots \]
\[ \varphi ::= p \mid \neg \varphi \mid \Diamond \varphi \mid \Box \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi \]

Semantics: Given by a model \( \langle W, R, \vartheta \rangle \)

- \( W \): a non-empty set of worlds
- \( R \): a binary reachability relation over the worlds of \( W \)
- \( \vartheta \): a valuation of every atom at every world of \( W \)

We write \( wRv \) to denote that \( v \) is an \( R \)-successor of \( w \).
Semantics (cont.): \( \neg, \land, \lor \) behave as in the classical propositional case at each world.

\[
\vartheta(w, \varphi \land \psi) = t \text{ iiff } \vartheta(w, \varphi) = t \text{ and } \vartheta(w, \psi) = t
\]

\[
\vartheta(w, \varphi \lor \psi) = t \text{ iiff } \vartheta(w, \varphi) = t \text{ or } \vartheta(w, \psi) = t
\]

\[
\vartheta(w, \neg \varphi) = t \text{ iiff } \vartheta(w, \varphi) = f
\]

\( \square \) and \( \diamond \) make use of the reachability relation \( R \):

\[
\vartheta(w, \square \varphi) = t \text{ iiff } \vartheta(v, \varphi) = t \text{ for every } v \in W \text{ with } wRv
\]

\[
\vartheta(w, \diamond \varphi) = t \text{ iiff } \vartheta(v, \varphi) = t \text{ for some } v \in W \text{ with } wRv
\]
Propositional Modal Logic

Semantics (cont.):

\[ \Gamma \models \varphi \iff \forall M. M \models \Gamma \Rightarrow M \models \varphi \]

\( \varphi \) is satisfiable iff \( \exists M. \exists w. \vartheta(w, \varphi) = t \)

Important relationship:

\[ \Gamma \models \varphi \iff \Gamma \wedge \neg \varphi \text{ is not satisfiable} \]
Tableau

What it is: Inference procedure that can determine satisfiability in propositional modal logic.

The gist: Take a set of formulae $Y$. Construct a tree of nodes (a tableau), by instantiating some rules of inference. See if this tree is ‘open’ (satisfiable) or ‘closed’ (unsatisfiable).
Tableau

**Node:** Set of propositional modal formulae.

The rules:

\[
(id) \quad \frac{p; \neg p; X}{X} \\
(\land) \quad \frac{\varphi \land \psi; X}{\varphi; \psi; X} \\
(\lor) \quad \frac{\varphi \lor \psi; X}{\varphi; X | \psi; X}
\]

\[
(\Diamond \Gamma) \quad \frac{\Diamond \varphi; \Box X; Z}{\varphi; X; \Gamma} \quad \forall \psi. \Box \psi \notin Z
\]

If a node matches a numerator pattern, then you can branch to the nodes in the denominator (separated by ‘|’).
If every leaf is an instance of the (id) rule, the tableau is closed, otherwise it is open.
Tableau

The rules:

\[(id) \frac{p; \neg p; X}{X} \quad (\wedge) \frac{\varphi \wedge \psi; X}{\varphi; \psi; X} \quad (\vee) \frac{\varphi \vee \psi; X}{\varphi; X | \psi; X}\]

\[(\Diamond \Gamma) \frac{\Diamond \varphi; \Box X; Z}{\varphi; X; \Gamma} \forall \psi. \Box \psi \notin Z\]

Example:

\[
\begin{align*}
(p_0 \wedge \Box p_1 \wedge \Diamond \neg p_1) & \vee (p_1 \wedge \neg p_1) \\
(p_0 \wedge \Box p_1 \wedge \Diamond \neg p_1) & \wedge (p_1 \wedge \neg p_1) \\
(p_0; \Box p_1 \wedge \Diamond \neg p_1) & \wedge (p_1; \neg p_1) \\
(p_0; \Box p_1; \Diamond \neg p_1) & \Diamond \Gamma \\
(p_1; \neg p_1) & (id) \\
\neg p_1 & (id) \\

\end{align*}
\]

The tableau is closed, so \((p_0 \wedge \Box p_1 \wedge \Diamond \neg p_1) \vee (p_1 \wedge \neg p_1)\) is unsatisfiable.
Tableau

The rules:

\[(id) \frac{p; \neg p; X}{\times} \quad (\wedge) \frac{\varphi \wedge \psi; X}{\varphi; \psi; X} \quad (\vee) \frac{\varphi \vee \psi; X}{\varphi; X \mid \psi; X}\]

\[(\Diamond \Gamma) \frac{\Diamond \varphi; \Box X; Z}{\varphi; X; \Gamma} \forall \psi. \Box \psi \notin Z\]

**Phases:** The \((id)\), \((\wedge)\) and \((\vee)\) rules are static rules. Both the numerator and denominator refer to the same world. The \((\Diamond \Gamma)\) rule is a transitional rule. The denominator refers to an \(R\)-successor of the numerator world.

Saturation phase: only static rules are instantiated.

Modal jump phase: the \((\Diamond \Gamma)\) rule is instantiated.
Binary Decision Diagrams

**High level:** A compact representation of a boolean function of boolean variables \( f : \text{Var} \mapsto \{ t, f \} \)

**Low level:** Directed acyclic graph of BDD-nodes: either the true-node, false-node or a variable-node. true- and false-nodes represent true and false.

Variable nodes: \( \langle v, high, low \rangle \)
- \( v \): a variable number.
- \( high, low \): branches to other bdd-nodes. Represent assignment of true and false to \( v \) respectively.
Binary Decision Diagrams

Example:

BDD for $f = v_a \lor v_b$, with high represented by a solid line, low by a dashed line. Each path to the true-node from the root, is a valuation on which the function is true.

Shorthand:

$$v = BDD\langle v, \text{true-node, false-node}\rangle$$
$$\neg v = BDD\langle v, \text{false-node, true-node}\rangle$$
Binary Decision Diagrams

BDDs of modal formulae: Define a mapping $[\cdot]$

$[p] = v_p$
$[\Box \varphi] = v_{\Box \varphi}$
$[\Diamond \varphi] = \neg [\Box \neg \varphi]$
$[\neg \varphi] = \neg [\varphi]$
$[\varphi \land \psi] = [\varphi] \land [\psi]$
$[\varphi \lor \psi] = [\varphi] \lor [\psi]$

Propositionally straightforward.
Modally shallow.
$\Diamond \varphi$ translated to equivalent $\neg \Box \neg \varphi$. 

BDDs of modal formulae (cont.): $[\varphi]$ is then a function that is true for every valuation that satisfies $\varphi$ at a propositional level.

**BDDs and tableau:** The set of all true valuations in $[\varphi]$ represents the set of open leaves of a fully saturated tableau for $\varphi$.

**In practice:** Can compute the saturation phase by constructing $[\varphi]$. 
Implementation

**Goal:** Determine whether $\Gamma \models \varphi$ or not.

**Procedure overview:** Translate to $\Gamma \land \neg \varphi$, and determine satisfiability.
- Construct a tableau in phases.
- Use BDDs to compute the saturation phase whenever it is needed.
- Recursive, depth-first approach.
**Refining a BDD:** When a modal jump closes, we need to pick another leaf to explore. Simple method would be to simply traverse our BDD representation of the tableau. Instead, we modify the BDD, removing that leaf, and then get a new leaf.

\[
(\diamond \Gamma) \frac{\diamond \varphi; \Box X; Z}{\varphi; X; \Gamma} \forall \psi. \Box \psi \notin Z
\]

\[BDD_f = BDD_i \land \neg (\diamond \varphi \land \Box x_1 \land \Box x_2 \land \ldots)\]
Optimisations

Idea: $\neg(v_0)$ is a stronger refinement than $\neg(v_0 \land v_1)$.

Implementation: Where possible, minimise the set of variables to refine over.
   Track ‘responsible’ variables.

Idea: Tableau may contain repetitions.

Implementation: Cache results of satisfiable and unsatisfiable.
# Results - LWB

<table>
<thead>
<tr>
<th>subclass</th>
<th>BDD-based</th>
<th>FaCT++</th>
<th>InKreSAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>branch_n</td>
<td>16</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>branch_p</td>
<td>21</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>d4_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>d4_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>dum_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>dum_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>grz_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>grz_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>lin_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>lin_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>path_n</td>
<td>21</td>
<td>21</td>
<td>10</td>
</tr>
<tr>
<td>path_p</td>
<td>21</td>
<td>21</td>
<td>10</td>
</tr>
<tr>
<td>ph_n</td>
<td>9</td>
<td>13</td>
<td>21</td>
</tr>
<tr>
<td>ph_p</td>
<td>9</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>poly_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>poly_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>t4p_n</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>t4p_p</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
</tbody>
</table>
Results - LWB

K_d4_n

K_path_n
Results

3CNF<sub>k</sub>

MQBF

instances solved

time (s)

BDD
FaCT++
InKreSAT

instances solved

time (s)

BDD
FaCT++
InKreSAT
Questions?

Any questions?