Extended Resolution Proofs for Symbolic SAT Solving

Toni Jussila, Carsten Sinz, and Armin Biere
Johannes Kepler University Linz, Austria
Why Propositional Logic Proofs?

- SAT-solvers and BDDs commercially employed
  - Hardware verification (Bounded Model Checking)
  - Product configuration

- Yes/No answer of solvers not sufficient
  - Counterexample or proof needed
  - Used for abstraction refinement, interpolant computation, proof checking, diagnosis, ...
Symbolic SAT-Solving

- **Given:** \( F = C_1 \land \ldots \land C_n \), a formula in CNF
- **Method:** Build a BDD \( B \) for \( F \) by BDD–and and BDD–exists operations as follows:
  - take a variable ordering
  - put all clauses \( C_i \) to buckets (one bucket for each variable)
  - process buckets (variables) one by one
    - build conjunction of clauses (BDD–and)
    - eliminate variable by existential quantification (BDD–exists)
    - put resulting BDD to the right bucket
Symbolic SAT Solving (II)

- **Fact:** $B=0$ iff $F$ unsatisfiable
- **Question:** How to build refutation proof for $F$ if $B=0$?
- **Solution:** Use Extended Resolution as proof system.
Extended Resolution (ER)

Resolution calculus: one inference rule

\[
\frac{C \cup \{l\} \quad \{\bar{l}\} \cup D}{C \cup D}
\]

- \(C, D\): clauses
- \(l\): literal occurring positively in \(C\) and negatively in \(D\)

Extended Resolution: adds extension rule

- Introduces new variable and clauses.
- „Definitions“

\[
\frac{\text{CNF}(x \leftrightarrow F)}{x: \text{new variable (neither occurring in } F \text{ nor in current clause set)}}
\]

- \(F\): arbitrary formula

Goal: derive empty clause

[Tseitin, 1970]
What Definitions?

- Add a new variable for every BDD node that occurs in the computation.

For BDD node $f$, definition is

- $f \leftrightarrow (x \ ? f_1 : f_0)$

- where $f_1$ and $f_0$ are the children of $f$.

- as formula: $(x \rightarrow f_1) \land (\neg x \rightarrow f_0)$

- as clauses: $(\neg f \ \neg x \ f_1), (\neg f \ x \ f_0), (f \ \neg x \ \neg f_1), (f \ x \ \neg f_0)$
ER Proof Generation Outline
(for unsatisfiable $F = C_1 \land \ldots \land C_n$)

1. Take first bucket $U$.
2. Compute BDDs $B_i$ for all clauses $C_i$ in $U$.
3. Add definitions for all BDD nodes occurring in any $B_i$.
   (convention: let $b_i$ be ER variable of the top node of $B_i$)
4. Produce ER proofs $F \vdash b_i$ for all clauses in $U$.
5. Compute the BDD of the conjunction of the clauses of $U$. $H_2 = \text{BDD-} \land (B_1, B_2)$ $H_i = \text{BDD-} \land (B_i, H_{i-1})$
6. Produce ER proofs $F \vdash h_i$ for all $h_i$. 
ER Proof Generation Outline (II)

7. Eliminate root variable, ie. compute BDD 
   \[ H_i' = \text{BDD-exists}(H_i) \].

8. Produce ER proofs \( F \vdash h_i' \) for all \( h_i' \).

9. Let \( U = \text{next_bucket}() \) and go to 2.
ER Proofs from BDDs:
Conjunctions (BDD–and)

- Build proof of $f \land g \rightarrow h$ recursively
  - from $f_0 \land g_0 \rightarrow h_0$ and $f_1 \land g_1 \rightarrow h_1$.

\[
\begin{array}{c}
\frac{(\bar{f}x_0)}{(\bar{f}_0 g_0 h_0)} \quad \frac{(\bar{f}x_1 h_1)}{(\bar{f}_1 g_1 h_1)} \\
\frac{(\bar{g}x_0)}{(\bar{f}x_0 h_0)} \quad \frac{(\bar{f}x_1 h_1)}{(\bar{g}x_1 g_1 h_1)} \\
\frac{(h_0 \bar{x}_0)}{(\bar{f}g x_0)} \quad \frac{(h_1 \bar{x}_1)}{(\bar{f}g x_0 h_1)} \\
\frac{(\bar{f}g x)}{(h_0 \bar{x}_0 h_0)} \quad \frac{(\bar{f}g x h_1)}{(h_1 \bar{x}_1 h_1)} \\
\frac{(\bar{f}g)}{(\bar{f}g x)} \quad \frac{(\bar{f}g h)}{(\bar{f}g h x)} \\
\end{array}
\]

**Complexity:** Requires 7 resolutions for each recursive step.
ER Proofs from BDDs:
Quantification (BDD-exists)

- Given \( f \) (children \( f_0 \) and \( f_1 \)), let \( \exists f \) be the BDD where root variable of \( f \) existentially quantified.
- First prove \( f_0 \lor f_1 \rightarrow \exists f \), clauses \((\neg f_0 \exists f), (\neg f_1 \exists f)\).
- Then prove \( f \rightarrow \exists f \), i.e. \((\neg f \exists f)\).

\[
\frac{\bar{f}x f_0 \quad \bar{f}_0 \exists f \quad \bar{f}_1 \exists f \quad \bar{f} x f_1}{\bar{f} x \exists f \quad \bar{f} x f_1 \quad \bar{f} x \exists f} \quad \frac{\bar{f} \exists f}{\bar{f} \exists f}
\]
Implementation: EBDDRES

- Performs BDD computations.
- Generates extended resolution proofs fully automatically.
- Good performance on some SAT instances that are hard for DPLL/resolution-based provers (e.g. pigeon hole).
- Proof-checker for resolution-based solvers can easily be adapted for ER proofs.
  - Only non-cyclicity test for extension rule applications has to be added.
<table>
<thead>
<tr>
<th></th>
<th>MINISAT</th>
<th></th>
<th>EBDDRES</th>
<th></th>
<th>EBDDRES, quantification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>solve</td>
<td>trace</td>
<td>solve</td>
<td>trace</td>
<td>bdd</td>
</tr>
<tr>
<td>1</td>
<td>resources</td>
<td>size</td>
<td>resources</td>
<td>gen ASCII</td>
<td>bin</td>
</tr>
<tr>
<td>2</td>
<td>sec</td>
<td>MB</td>
<td>sec</td>
<td>MB</td>
<td>sec</td>
</tr>
<tr>
<td>ph7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>ph8</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ph9</td>
<td>6</td>
<td>4</td>
<td>11</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ph10</td>
<td>44</td>
<td>4</td>
<td>63</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>ph11</td>
<td>84</td>
<td>6</td>
<td>929</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>ph12</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>22</td>
</tr>
<tr>
<td>ph13</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>126</td>
</tr>
<tr>
<td>ph14</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>9</td>
<td>111</td>
</tr>
<tr>
<td>mutcb8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>mutcb9</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>mutcb10</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>mutcb11</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>mutcb12</td>
<td>8</td>
<td>4</td>
<td>22</td>
<td>2</td>
<td>32</td>
</tr>
<tr>
<td>mutcb13</td>
<td>112</td>
<td>5</td>
<td>244</td>
<td>7</td>
<td>126</td>
</tr>
<tr>
<td>mutcb14</td>
<td>488</td>
<td>8</td>
<td>972</td>
<td>14</td>
<td>250</td>
</tr>
<tr>
<td>mutcb15</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>36</td>
<td>498</td>
</tr>
<tr>
<td>mutcb16</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>urq35</td>
<td>95</td>
<td>4</td>
<td>218</td>
<td>2</td>
<td>22</td>
</tr>
<tr>
<td>urq45</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>urq55</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>urq65</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>urq75</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>urq85</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>fpga108</td>
<td>0</td>
<td>2</td>
<td></td>
<td>6</td>
<td>47</td>
</tr>
<tr>
<td>fpga109</td>
<td>0</td>
<td>0</td>
<td></td>
<td>3</td>
<td>44</td>
</tr>
<tr>
<td>fpga1211</td>
<td>0</td>
<td>0</td>
<td></td>
<td>53</td>
<td>874</td>
</tr>
<tr>
<td>add16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>add32</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>add64</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>146</td>
</tr>
<tr>
<td>add128</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>-</td>
<td>*</td>
</tr>
</tbody>
</table>
Summary

- Extends work of Biere & Sinz 2006 with existential quantification.
- Extended resolution proofs as generic proof format.
- Enabler for further applications of extended resolution.