Equivalence Checking By Logic Relaxation

Eugene Goldberg

FMCAD,
Mountain View, CA, USA
October 3-6, 2016
Outline

• Introduction

• Equivalence checking by logic relaxation

• Experimental results and conclusions
Motivation

- Equivalence Checking (EC) is an important part of formal verification
- Any progress in EC empowers logic synthesis
- Short EC proofs for structurally similar circuits
- Ideas of EC of combinational circuits can be reused in EC of sequential circuits and software
Solving EC

Prove

\[ EQ \land G_{rlx} \Rightarrow (z' \equiv z''), \]

where

\[ G_{rlx} = F_{N'} \land F_{N''} \]

This reduces to proving

\[ EQ \land G_{rlx} \land \neg(z' \equiv z'') \]

UNSAT

\[ EQ(x', x'') = 1, \text{ iff } x' = x'' \]
Let $\textit{Img}_{\text{cut}}$ specify the cut image

$\textit{Img}_{\text{cut}}(q',q'')=0$, iff there is no input $(x',x'')$, $x' = x''$ for which $N',N''$ produce $(q',q'')$

Let $\textit{Cut} = \{z',z''\}$. $N'$ and $N''$ are equivalent iff $\textit{Img}_{\text{cut}} \Rightarrow (z' \equiv z'')$, 

$\text{EQ}(X',X'')$
Problem To Solve: Finding an Inductive Proof Of Equivalence

Given combin. circuits \( N' \) and \( N'' \), find formulas \( H_i \) such that

- \( \text{Img}_i \Rightarrow H_i, \ 0 \leq i < k \)
- \( H_i \) are as simple as possible
- \( H_i \) can be derived from \( H_{i-1} \)
- \( H_k \equiv \text{Img}_k(z',z'') \)

A simple inductive proof should exist if \( N' \) and \( N'' \) are struct. similar
Some Background

Building inductive proofs of equivalence

- Berman, Trevillyan 1988
- Brand 1993
- Kuehlmann, Krohm 1996
- Goldberg, Prasad, Brayton 2001
- Mishchenko, Chatterjee, Brayton, Een 2006

Proofs are based on derivation of pre-defined relations e.g. equivalences
Outline

• Introduction
• Equivalence checking by logic relaxation
• Experimental results and conclusions
Structure Of Cut Image

Assignments excluded from cut image: $S_{\text{excl}} = S_{\text{rlx}} \cup S_{\text{imp}}$

$S_{\text{rlx}} = \{(q', q'') \mid \text{only relaxed inputs } (x', x'') \text{ where } x' \neq x'' \text{ can produce } (q', q'') \}$

$S_{\text{imp}} = \{(q', q'') \mid \text{no input } (x', x'') \text{ can produce } (q', q'') \}$

$(q', q'') \in S_{\text{imp}} \iff$

- $q'$ cannot be produced in $N'$ and/or
- $q''$ cannot be produced in $N''$
Definition Of Boundary Formulas

EC by Logic Relaxation:

“replace” $Img_{cut}$ with boundary formula $H_{cut}$

Boundary formula $H_{cut}$:

1. If $(q', q'') \in S_{rlx}$, then $H_{cut}(q', q'') = 0$
2. If $(q', q'') \in S_{imp}$, then $H_{cut}(q', q'')$ can take an arbitrary value
3. $Img_{cut} \Rightarrow H_{cut}$
Boundary Formula for \( \text{Cut} = \{ z', z'' \} \)

Assume that \( N' \) and \( N'' \) are not constants.

\[
S_{\text{imp}} = \emptyset \quad \iff \quad S_{\text{excl}} = S_{\text{rlx}}
\]

\[
H_{\text{cut}} \equiv Img_{\text{cut}}
\]

Testing if \( N' \) is a constant: two easy SAT checks.
Boundary Formula And Partial Quantifier Elimination

Complete Quantif. Elimin.

\[ \text{Img}_{\text{cut}} \equiv \exists W \ [ \text{EQ} \land F_{M} ] \]
\[ W = \text{Vars}(F_{M}) \setminus \text{Vars}(\text{Cut}) \]

Partial Quantif. Elimin.

\[ H_{\text{cut}} \land \exists W [ F_{M} ] \equiv \exists W [ \text{EQ} \land F_{M} ] \]

\[ \text{EQ} \land \text{G}_{\text{rlx}} \land \sim(z' \equiv z'') \] is equisat. with
\[ H_{\text{cut}} \land \text{G}_{\text{rlx}} \land \sim(z' \equiv z'') \]

where \( \text{G}_{\text{rlx}} = F_{N'} \land F_{N''} \)
Contrasting Cut Image And Boundary Formulas

\[ N' \quad N'' \]

\[ \text{Cut} \]

\[ \text{Img}_{\text{cut}} \]

\[ \text{Cut} \]

\[ \text{H}_{\text{cut}} \]

\[ \text{EQ}(X', X'') \]

\[ \text{EQ}(X', X'') \]
Boundary Formulas Of Structurally Similar Circuits

Suppose, \( \forall \nu \in Cut' \)
\[ \nu = g_\nu(L_\nu) \text{ where } L_\nu \subseteq Cut'' \]

Let \( \text{Max}_{\text{cut}} \) be the largest value of \( |L_\nu| \), \( \forall \nu \in Cut' \)

Then \( H_{\text{cut}} \) can be built from
\( (\text{Max}_{\text{cut}} + 1) \)-literal clauses
EC By Logic Relaxation

Cut₀ = X' ∪ X'',..., Cutₖ={z',z''}

Compute H₀,..,Hₖ

where H₀ = EQ(X',X'')

Hᵢ ∧ ∃Wᵢ [ Fᵢ ] ≡ ∃Wᵢ [ Hᵢ₋₁ ∧ Fᵢ ]

Wᵢ = Vars(Fᵢ) \ Vars(Cutᵢ)

If Hₖ ⇒ (z' ≡ z''),
N' and N'' are equivalent

If, say, Hₖ(z' =0,z''=1)=1 and N', N'' can produce 0 and 1, they are inequivalent
Outline

• Introduction
• Equivalence checking by logic relaxation
• Experimental results and conclusions
Non-Trivial Example Of EC

Mlp_s computes a median bit of an s-bit multiplier

Operands A and B where
A={a_1,..,a_s}, B={b_1,..,b_s}

h is an additional input variable

If h=1, then N' and N'' compute Mlp_s
if h=0, then N' and N'' evaluate to 0
Comparison With *ABC*

- Partial Quantifier Elimination (a variation of HVC-14 algorithm) is based on machinery of D-sequents (FMCAD-12, FMCAD-13)
- *ABC* is a high-quality tool developed at UC, Berkeley

<table>
<thead>
<tr>
<th>val. of s in <em>Mlp</em></th>
<th>#cuts</th>
<th>EC by LoR (s.)</th>
<th><em>ABC</em> (s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>37</td>
<td>4.5</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>41</td>
<td>7.1</td>
<td>38</td>
</tr>
<tr>
<td>12</td>
<td>45</td>
<td>11</td>
<td>142</td>
</tr>
<tr>
<td>13</td>
<td>49</td>
<td>16</td>
<td>757</td>
</tr>
<tr>
<td>14</td>
<td>53</td>
<td>25</td>
<td>3,667</td>
</tr>
<tr>
<td>15</td>
<td>57</td>
<td>40</td>
<td>11,237</td>
</tr>
<tr>
<td>16</td>
<td>61</td>
<td>70</td>
<td>&gt; 6 h</td>
</tr>
</tbody>
</table>

Formulas $H_i$ were computed approximately

$$H_i \land \exists W_i [ F_{Mi} ] \equiv \exists W_i [ H_{i-1} \land F_{Mi} ]$$

$F_{Mi}$ specifies logic below $i$-th cut

Only a subset of clauses of $F_{Mi}$ was used
Proving Inequivalence

Formula $\alpha$

$$EQ(X',X'') \land F_{N'} \land F_{N''} \land \sim(z' \equiv z'')$$

Formula $\beta$

$$H_3 \land F_{N'} \land F_{N''} \land \sim(z' \equiv z'')$$

Formula $H_3$ was computed precisely

Sat-solver: Minisat 2.0, Time limit: 600 s

<table>
<thead>
<tr>
<th>Form. type</th>
<th>#solved</th>
<th>total time (s)</th>
<th>median time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>95</td>
<td>$&gt; 3,490$</td>
<td>4.2</td>
</tr>
<tr>
<td>$\beta$</td>
<td>100</td>
<td>1,030</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Conclusions

- Relative complexity($N', N''$) $<<$ Absolute complexity($N', N''$)
- EC by logic relaxation gives a general solution
- It can be extended to sequential circuits/programs
- Efficient partial quantifier elimination is of great value