An Efficient Algorithm for the Analysis of Cyclic Circuits

Osama Neiroukh  Stephen A. Edwards  Xiaoyu Song

Intel  Columbia University  Portland State University
What is a Combinational Circuit?

Malik [ICCAD 93]: “A circuit is combinational for an input pattern if three-valued simulation starting from Xs converges to 0s and 1s.”

Shiple [96]: “Equivalent to stability in Brzozowski and Seger’s [95] model.”
**Goal**

Given a cyclic circuit that is combinational for some inputs, create an acyclic circuit that computes the same combinational function.

<table>
<thead>
<tr>
<th>abcd</th>
<th>wxyz</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0001</td>
<td>0000</td>
</tr>
<tr>
<td>0010</td>
<td>0010</td>
</tr>
<tr>
<td>0011</td>
<td>1011</td>
</tr>
<tr>
<td>0100</td>
<td>0000</td>
</tr>
<tr>
<td>0101</td>
<td></td>
</tr>
<tr>
<td>0110</td>
<td>0010</td>
</tr>
<tr>
<td>0111</td>
<td>1111</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>1001</td>
<td>1000</td>
</tr>
<tr>
<td>1010</td>
<td>1010</td>
</tr>
<tr>
<td>1011</td>
<td>1011</td>
</tr>
<tr>
<td>1100</td>
<td>1110</td>
</tr>
<tr>
<td>1101</td>
<td>1111</td>
</tr>
<tr>
<td>1110</td>
<td>1110</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
</tr>
</tbody>
</table>

This input not combinational
Applications

Fixing cyclic circuits from high-level synthesis

Stok [ICCAD 92]: cycles from resource sharing

Berry [92]: cycles from Esterel programs

Acyclic circuits easier to simulate
Related Work

Malik [ICCAD 93]: basic definitions, unrolling
Edwards [DAC 03]: basis of our work
Gupta and Selvidge [ICCAD 05]: fix single loops
Riedel [DAC 03]: a technique for creating them
Approach [Edwards, DAC 03]

Original Circuit

An Efcient Algorithm for the Analysis of Cyclic Circuits – p. 6/1
Approach [Edwards, DAC 03]

Original Circuit

Acyclic Fragments
Approach [Edwards, DAC 03]

Original Circuit

Acyclic Fragments

Merged
Approach [Edwards, DAC 03]

Original Circuit

Acyclic Fragments

Merged

Simplified
First Observation [Edwards 2003]

For an input pattern to be combinational, at least one input coming from outside each strongly-connected component must have a controlling value.

If all external inputs were non-controlling, the gates in the SCC would stay at X.
Second Observation

Frontier gate: some inputs dened, output remains X. Input is combinational, frontier is empty.
Second Observation

An Efficient Algorithm for the Analysis of Cyclic Circuits – p. 8/12
Second Observation

Frontier gate: some inputs defined, output remains X
Second Observation

*Frontier gate*: some inputs defined, output remains $X$

Input is combinational $\Leftrightarrow$ frontier is empty
Our Algorithm Step 1: Apply a Controlling Value to Each Input
Our Algorithm Step 1: Apply a Controlling Value to Each Input

\[
a = 0 \quad \text{acyclic}
\]
Our Algorithm Step 1: Apply a Controlling Value to Each Input

a = 0  acyclic

b = 0
Our Algorithm Step 1: Apply a Controlling Value to Each Input

\[ a = 0 \quad \text{acyclic} \]

\[ b = 0 \]
Our Algorithm Step 1: Apply a Controlling Value to Each Input

\[ a = 0 \quad \text{acyclic} \]

\[ b = 0 \]

\[ c = 0 \]
Our Algorithm Step 1: Apply a Controlling Value to Each Input

\[ a = 0 \quad \text{acyclic} \quad d = 1 \]

\[ b = 0 \]

\[ c = 0 \]
Our Algorithm Step 1: Apply a Controlling Value to Each Input
Our Algorithm Step 2: Attack Frontier Gates with Combinations

\[ b = 0 \]
\[ c = 0 \]
\[ d = 0 \]
\[ e = 0 \]
\[ f = 0 \]
\[ g = 0 \]

\[ b = 0 + c = 0 \]
Our Algorithm Step 2: Attack Frontier Gates with Combinations

\[
\begin{align*}
V &= b = 0 \quad V + c = 0 \\
V &= b = 0 \quad V + d = 1
\end{align*}
\]
Our Algorithm Step 2: Attack Frontier Gates with Combinations

\[ b = 0 \]
\[ c = 0 \]
\[ d = 1 \]
\[ f = 1 \]
\[ g = 0 \]
## Experimental Results

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Netlist</th>
<th>SCC</th>
<th>DAC 03</th>
<th>Ours</th>
<th>Acyclic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gates</td>
<td>SCC</td>
<td>PAs</td>
<td>time</td>
<td>PAs</td>
</tr>
<tr>
<td>arbiter5</td>
<td>213</td>
<td>25</td>
<td>257</td>
<td>1.3</td>
<td>25</td>
</tr>
<tr>
<td>arbiter6</td>
<td>248</td>
<td>30</td>
<td>745</td>
<td>8</td>
<td>29</td>
</tr>
<tr>
<td>arbiter7</td>
<td>283</td>
<td>35</td>
<td>2205</td>
<td>69</td>
<td>33</td>
</tr>
<tr>
<td>arbiter8</td>
<td>318</td>
<td>40</td>
<td>6581</td>
<td>656</td>
<td>37</td>
</tr>
<tr>
<td>exp</td>
<td>124</td>
<td>69</td>
<td>54517</td>
<td>2868</td>
<td>23260</td>
</tr>
<tr>
<td>ex1</td>
<td>150</td>
<td>47</td>
<td>43777</td>
<td>2341</td>
<td>232</td>
</tr>
<tr>
<td>gary</td>
<td>177</td>
<td>32</td>
<td>∞</td>
<td></td>
<td>290</td>
</tr>
<tr>
<td>planet</td>
<td>253</td>
<td>51</td>
<td>∞</td>
<td></td>
<td>1489</td>
</tr>
<tr>
<td>s1488</td>
<td>272</td>
<td>61</td>
<td>∞</td>
<td></td>
<td>588</td>
</tr>
<tr>
<td>table3</td>
<td>311</td>
<td>49</td>
<td>∞</td>
<td></td>
<td>3604</td>
</tr>
</tbody>
</table>
Conclusions

- More focused exploration of search space
- Idea: combine partial assignments to attack frontier gates
- Exponential improvement compared to Edwards [DAC 03]
- Future work
  - Even better pruning
  - Symbolic approach?