Minimal test sets, defect coverage, and other random ramblings: My First RATS Talks

Ernesto Staroswiecki
Outline

- Introduction
- Goal
- Minimal Test Sets
- Exhaustive Tests
- Questions? Ideas?
Compaction vs. Compression

**Compaction**: reduce the size of a test set while maintaining coverage
- Less vectors
- Less “don’t cares”

**Compression**: reduce the storage requirements of a test set without changing the “cares” of the vectors
- Fill the “don’t cares” and compress.
Compaction vs. Compression

- Compaction ≠ Compression

- They share some issues:
  - Loss of defect coverage

- They share some questions:
  - Is there a “cheaper” way to obtain the same results?
Outline

- Introduction
- Goal
- Minimal Test Sets
- Exhaustive Tests
- Questions? Ideas?
Goal

- Better understanding of implications of test set compaction
  - Quantify defect coverage loss

- Compare the compaction performance of commercial tools with the maximum achievable results

- Work on methods to achieve comparable results
Outline

- Introduction
- Goal
- Minimal Test Sets
- Exhaustive Tests
- Questions? Ideas?
Minimal Test Set

- We want to see how well do the commercial tools compact
- We also want to see how much defect coverage do we loose with compaction
- Best way to do both: Find the minimal test set
Minimal Cover

- Can solve exactly a minimal cover problem
- Usually used for logic minimization

<table>
<thead>
<tr>
<th>Implicants</th>
<th>Minterms</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Minimal Cover

- We can solve this problem through tree iterative steps:
  - Select essential implicants
  - Remove dominating columns
  - Remove dominated rows

- We are left with "cyclic core"
  - Use Petrick’s method
## Minimal Test Set

- Consider for now only single stuck-at faults
- Finding a test set is a covering problem

<table>
<thead>
<tr>
<th>Vectors</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Minimal Test Set

- Replace “Minterms” with “Faults,” and “Implicants” with “Vectors”

- The solutions of the minimal cover problem are the minimal subsets of the starting vector set that cover all the faults in the starting fault list
“Ultimate” Test Set

- Start with all the vectors \( (2^N) \) and all detectable faults

- Solution is the minimal, 100% fault coverage, test set for single stuck-at faults
“Ultimate” Test Set

- Can use this to evaluate the overall performance of commercial tools regarding test generation and compaction

- Can use this to find the highest cost of compaction (most escapes)
Other Minimal Test Sets

- Start with all detectable faults and tool generated 100% fault coverage vector set
- Can use this to evaluate tool compaction algorithm
- Also evaluate starting set dependence on minimal test set size
Other Minimal Test Sets

- Start with tool generated X% test coverage vector set and the detected faults
- Can use this to evaluate tool compaction algorithm
- NOTE: Solution need NOT be minimal X% fault coverage test set
  - Problem is even more intractable
Other Minimal Test Sets

■ Could this be a valid comparison of sets generated by different tools?
  – Do we care about how “compactable” a test set is?

■ Maybe we can find an algorithm to solve for the minimal X% fault coverage test set
Issues with Minimal Test

- Solution is not unique
  - How do we choose one?
  - Are we looking for best case or worse case?
  - Add compression to the picture

- Problem is intractable
Outline

- Introduction
- Goal
- Minimal Test Sets
- Exhaustive Tests
- Questions? Ideas?
Exhaustive Tests

- Way to evaluate test sets before going to the tester

- Create a TIC fault dictionary for our chips (ELF35, Murphy, etc.) to find “interesting” vector sets
Exhaustive Tests

- Run exhaustive \( (2^N) \) test set on our combinational CUTs in several different orderings (in-order, reversed, pseudo-random, PR reversed, etc.)

- Record ALL vectors that fail for a defective chip
Exhaustive Tests

- Consider only the vectors that failed in all the different orderings

- Presume this vectors are order-independent

- Generate Vector/Defect table
Evaluate Test Sets

- If my test set is a "cover" of the Vector/Defect table, presume it will find all defects (?)

- Minimal test sets that are NOT covers might be the interesting ones (?)
Evaluate Test Sets

- I can compare the defect coverage of a minimal X% fault coverage test set with the defect coverage of a tool generated Y% fault coverage test sets.

- Y < X => cheaper to generate Y% test set
Evaluate Test Sets

- And how about the size?
- Compare size of $Y\%$ test set with tool compacted $X\%$ test set
- Compare size of $Y\%$ test set with minimal $X\%$ test set
- (Remember $Y < X$)
Issues

- Can we study CUT dependence?
- Do we have enough defective chips?
- Will results be statistically significant?
Outline

- Introduction
- Goal
- Minimal Test Sets
- Exhaustive Tests
- Questions? Ideas?
Questions? Ideas?
Thanks!
Extra topics

- Other fault models
- Minimal N-detect test set
Other Fault Models

- Can I find Minimal Test Sets for other fault models?

- Iddq, Bridging Faults, Transition Faults, Delay Faults, etc.

- Can I combine them?
N-detect Minimal Test

- I can find Minimal Test Sets for N-detect
- Need to re-define column and row dominance concepts, and keep track of the number of times a fault has been covered
- Can’t find ALL Minimal Test Sets for N-detect
N-detect Minimal Test

- Compare size of Minimal N-detect Test Set with Minimal 100% ss-@ coverage Test Set

- Once again, I can evaluate SOME minimal set
References


References