Combinatorial Coverage Measurement

Rick Kuhn
NIST
Software Failure Analysis

• We studied software failures in a variety of fields including 15 years of FDA medical device recall data
• What causes software failures?
  • logic errors?
  • calculation errors?
  • inadequate input checking?
  • interaction faults? Etc.

Interaction faults: e.g., failure occurs if

- pressure < 10
- pressure < 10 && volume > 300

(1-way <= all-values testing catches)
(2-way <= all-pairs testing catches)

Example:

Failure when “altitude adjustment set on 0 meters and total flow volume set at delivery rate of less than 2.2 liters per minute.”

=> 2-way interaction
Software Failure Internals

How does an interaction fault manifest itself in code?

Example: `altitude_adj == 0 && volume < 2.2` (2-way interaction)

```java
if (altitude_adj == 0) {
    // do something
    if (volume < 2.2) { faulty code! BOOM! }
    else { good code, no problem}
} else {
    // do something else
}
```

A test that included `altitude_adj == 0` and `volume = 1` would trigger this failure
How about flaws that are harder to find?

- Interactions e.g., failure occurs if
  - pressure < 10 (1-way interaction)
  - pressure < 10 & volume > 300 (2-way interaction)
  - pressure < 10 & volume > 300 & velocity = 5 (3-way interaction)
  - The most complex failure reported required 4-way interaction to trigger
What about other applications?

Server (green)

These faults more complex than medical device software!! Why?

Why?
Others?

Browser (magenta)
Still more?

NASA Goddard distributed database  (light blue)

Note: development data, others are released products
Even more?

FAA Traffic Collision Avoidance System module (seeded errors)  (purple)
Finally

Network security (Bell, 2006) (orange)

Curves appear to be similar across a variety of application domains.
Fault curve pushed down and right as faults detected and removed?

![Graph showing fault curve pushed down and right as faults detected and removed.](image)

- **App users**: Database 10s (testers)
- **Med. 100s**
- **Server**: 10s of mill.
- **Browser**: 10s of mill.
- **TCP/IP**: 100s of mill.
Interaction Rule

- How many parameters involved in faults? => *interaction rule*: most failures are triggered by one or two parameters, and progressively fewer by three, four, or more parameters, and the maximum interaction degree is small.

- Maximum interactions for fault triggering was 6
- Popular “pairwise testing” not enough
- More empirical work needed
- Reasonable evidence that maximum interaction strength for fault triggering is relatively small
How do we use this knowledge in testing?
A simple example
How Many Tests Would It Take?

- There are 10 effects, each can be on or off
- All combinations is $2^{10} = 1,024$ tests
- What if our budget is too limited for these tests?
- Instead, let’s look at all 3-way interactions ...
Now How Many Would It Take?

- There are $\binom{10}{3} = 120$ 3-way interactions.
- Naively $120 \times 2^3 = 960$ tests.
- Since we can pack 3 triples into each test, we need no more than 320 tests.
- Each test exercises many triples:

\[
\begin{array}{cccccccc}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
\end{array}
\]
A covering array

All triples in only 13 tests, covering \( \binom{10}{3}2^3 = 960 \) combinations

Each row is a test:

- Developed 1990s
- Extends Design of Experiments concept
- NP hard problem but good algorithms now
How does this knowledge help?

If all faults are triggered by the interaction of \( t \) or fewer variables, then testing all \( t \)-way combinations can provide strong assurance.

(taking into account: value propagation issues, equivalence partitioning, timing issues, more complex interactions, . . . )
Test coverage measurement

Path coverage

• Many varieties, studied for decades
• Path, branch, condition coverage, plus many variations

Combinatorial coverage

• The subject of this talk, new
• How should we measure it?
## Combinatorial Coverage Measurement

<table>
<thead>
<tr>
<th>Tests</th>
<th>Variables</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a b c d</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable pairs</th>
<th>Variable-value combinations covered</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ab</td>
<td>00, 01, 10</td>
<td>.75</td>
</tr>
<tr>
<td>ac</td>
<td>00, 01, 10</td>
<td>.75</td>
</tr>
<tr>
<td>ad</td>
<td>00, 01, 11</td>
<td>.75</td>
</tr>
<tr>
<td>bc</td>
<td>00, 11</td>
<td>.50</td>
</tr>
<tr>
<td>bd</td>
<td>00, 01, 10, 11</td>
<td>1.0</td>
</tr>
<tr>
<td>cd</td>
<td>00, 01, 10, 11</td>
<td>1.0</td>
</tr>
</tbody>
</table>

100% coverage of 33% of combinations  
75% coverage of half of combinations  
50% coverage of 16% of combinations
Graphing Coverage - graphing

33% of combinations covered 100%

Bottom line: All combinations covered to at least 50%

100% coverage of 33% of combinations
75% coverage of half of combinations
50% coverage of 16% of combinations
Adding a test

Coverage after adding test [1,1,0,1]
Adding another test

Coverage after adding test [1,0,1,1]
Additional test completes coverage

Coverage after adding test [1,0,1,0]
All combinations covered to 100% level,
so this is a covering array.
Coverage Measurement Tool

Combinatorial Coverage Measurement

NIST

Coverage for file: 7489x82
Total 3-way: 0.000

Coverage:
- Cov >= 0.00 - 88508948960 = 1.000
- Cov >= 0.05 - 88508948960 = 1.000
- Cov >= 0.10 - 88508948960 = 1.000
- Cov >= 0.15 - 88508948960 = 1.000
- Cov >= 0.20 - 88508948960 = 1.000
- Cov >= 0.25 - 88508948960 = 1.000
- Cov >= 0.30 - 88508948960 = 1.000
- Cov >= 0.35 - 88508948960 = 0.999
- Cov >= 0.40 - 88508948960 = 0.999
- Cov >= 0.45 - 88508948960 = 0.999
- Cov >= 0.50 - 88508948960 = 0.999
- Cov >= 0.55 - 88508948960 = 0.999
- Cov >= 0.60 - 88508948960 = 0.999
- Cov >= 0.65 - 88508948960 = 0.999
- Cov >= 0.70 - 88508948960 = 0.999
- Cov >= 0.75 - 88508948960 = 0.999
- Cov >= 0.80 - 88508948960 = 0.999
- Cov >= 0.85 - 88508948960 = 0.999
- Cov >= 0.90 - 88508948960 = 0.999
- Cov >= 0.95 - 88508948960 = 0.999
- Cov >= 1.00 - 88508948960 = 0.999

Chart:
- X = proportion of combinations
- Y = combination variable-value coverage

2-way stats:
- Combinations: 3,321
- Var/val cons: 14,761
- Total coverage: 0.940

3-way stats:
- Combinations: 88,560
- Var/val cons: 828,135
- Total coverage: 0.831
4 variables, mixed level

- Line graph for 2-way coverage shows 100% for half, 75% for half; 3-way coverage (blue line) at 75% for 25% of combinations, 40% coverage for 75% of combinations
- Number of 2-way combinations = $C(4,2) = 6$
Comparing with line graph:
- Line graph shows 3-way coverage (blue line) at 75% for 25% of combinations, 40% coverage for 75% of combinations.
- 3d graph shows one combination with 60%-80% coverage (green), and three with 40%-60% coverage (yellow).
- Number of 3-way combinations = C(4,3) = 4.
7 variables, mixed level
Two views of the 3-way graph. x, y, z are variable indices; color is coverage level.

What does this mean?
• Compared w/ 2-way, far fewer combinations with >80% coverage (blue), more with 60% .. 80% (green) than for 2-way
• Relatively few w/ <60% (red, orange, or yellow)
• One variable involved in low-coverage (orange) combinations, as seen by single line of markers
• Number of points = C(7,3) = 35
Random values, 0..3
Same data, with one interaction

\[ p_6 = (p_5 + 2) \mod 4 \]
Spacecraft tests, 82 variables, mostly binary

- Line graph shows 2-way (red), 3-way (blue), and 4-way (green) combination coverage.
- Heat map shows 2-way combination coverage; percentage coverage shown in color key above chart.
Heat map style graph of 3-way coverage

What does this mean?
- Compared w/ 2-way, far fewer combinations with >80% coverage (blue), more with 60% .. 80% (green)
- Relatively few w/ <60% (red, orange, or yellow)
- Small number of individual variables involved in low-coverage (orange) combinations
- Number of points = C(82,3) = 82,560

x, y, z are variable indices; color is coverage level.
Summary

• Combinatorial coverage is an additional measurement that may be applied to system tests
  • applies to test data, rather than source code
  • may have utility for other data analysis?

• Has been applied to tests for NASA spacecraft
  • identify interactions that may not be tested sufficiently
  • can be used to automatically generate new tests to supplement coverage

• Part of overall combinatorial testing approach to software assurance

• Further information: Rick Kuhn - kuhn@nist.gov