Is Mutation an Appropriate Tool for Testing Experiments?

J.H. Andrews, L.C. Briand, Y. Labiche

ICSE '05

Mutants = real faults?

Problem: I want to compare testing techniques, but I don't have subject programs with lots of known faults.

Workaround: Automatically generate "mutants" of subject programs.

Question: Do results based on mutants generalize to programs with real faults?

What's a mutant?

A mutant of a program is that program with a small automatic change:

- add/subtract 1 from integer constants
- change * to /
- change TRUE to FALSE
- delete a statement
- other similar changes...

Mutants are easy to create in large numbers.

Experiment (1:3)

1. Take 8 programs with multiple known faulty versions and big pools of test cases.
2. Make mutants.
3. Eliminate mutants not detected by any test case.
4. Run randomly-chosen test suites on faulty versions.
5. Run same test suites on mutants.

Experiment (2:3)

For each faulty version or mutant of a given program:

```
+---+           #
|###|          #
|###| ------>   # #  ----->  8^P
|###| random   #     apply
+---+            #
Big            5000         Faulty
test           test         version
pool           suites       or mutant
```

Experiment (3:3)

```
COMPARE
             Mean # of faulty versions detected
             by each test suite
          -----------------------------------------
                  # of faulty versions
WITH
                Mean # of mutants detected
                by each test suite
          -----------------------------------------
                     # of mutants
```

Hypothesis: Detection ratios will be equal.

What is being measured?

```
| faults not |    | mutants not | -- not
| detected   |    | detected    | -- tested
+------------+    +-------------+
| faults     |    | mutants     | \  tested
| detected   |    | detected    |  | against
| by big     |    | by big      |  | subset
| test pool  |    | test pool   | /  of pool
+------------+    +-------------+
```

- What if each suite caught every fault?
- What if each suite caught at most one fault?
- Is this what we want?

The test applications

- ESA "space" program, 6KLOC, real faults.
- 7 "Seimens programs" $\leq$ 500LOC each with *hand-seeded faults*.
- Experiment treats real and hand-seeded faults as equivalent.

Mutants = real faults needs an experiment.

Hand-seeded faults = real faults can just be assumed?

Empirical Results

- Median detection ratios for "space":
  - mutants: 75%
  - real faults: 76%
- Median detection ratios for 7 "Seimens" programs:
  - mutants: about 96%
  - hand-seeded faults: about 70%

Extra bonus analysis

Why not just calculate and compare:

```
                 # of test cases in pool that
 Mean over all   detected this faulty version
 faulty versions: ----------------------------
                 total number of test cases

WITH

                 # of test cases in pool that
     Mean over   detected this mutant
     all mutants: ----------------------------
                 total number of test cases
```

"Ease of detection"

Extra bonus results

Program	Faulty Versions	Mutants
SPACE	15%	10%
others	5%	30%

(Values estimated by eye from paper's graphs.)

Authors' conclusions

1. Mutants = real faults
   - Supported by space case in experiment,
   - but what about the other 7 cases?
   - And what about the "ease of detection" calculation?
2. Hand-seeded faults are harder to detect than real faults.
   - Note on page 8 reveals original hand-seeded fault authors discarded any fault detected by 350 or more of their test cases.

And what about this?

Recall the Graves 2001 empirical regression test selection technique study:

- Used same programs as this experiment...
- ... plus one more: the Player program.
- Player was the only example with an actual history of real feature additions.
- Player results said “minimization” technique was good, the other cases said bad.
- Conclusion: minimization bad.