Open Challenges in Mutation Testing

Panel Discussion
Objectives

“In 2000, I thought we were finished with mutation.”
Objectives

- Why is mutation testing not done in industry? (Or is it?)
- Where is research leading to?
- Where should research be leading to?
Panelists

Mark Hampton  Mark Harman  Jeff Offutt
Panelists

Mark Hampton
Mark Harman
Jeff Offutt

4th Order Mutants?
Survey Results
Mutation testing in practice

- Mutation is mainly used for research experiments
- Equivalent mutants are not ignored
- Equivalent mutants are mainly detected manually
- Mutation researchers like their own tools?
Industry Acceptance

- Tools
- Scalability
- Marketing
- Equivalent mutants
- Understandability
- Data
- Real faults
Industry Acceptance

- Lack of automatic test data generation
- Lack of cooperation between industry and academia
- Industry has many pressing problems
- Huge effort and costs compared to other techniques
- Benefits unclear for evolving systems
Research Challenges

- Tools
- Equivalent mutants
- Real world projects
- Efficiency
- New operators
- HOM
- Languages
Research Challenges

• Test (data) generation

• Use mutation for other purposes such as repairing faults, automatic patching

• Developers don’t care how tests are built - they just want good tests

• Handle larger code with various features (databases, network, ...)

Mark Hampton
previously CTO & co-founder at Certess

Certess developed a Mutation Analysis tool.

Languages supported: VHDL, Verilog, SystemVerilog, C, C++

Company grew from 3 to 25 in 5 years
Sold to a much larger company in early 2009
Major clients: CISCO, Intel, Sony, Toshiba, Bosch etc
Open Challenges in Mutation Testing

Some common assumptions need to be questioned (5 in this presentation)

To be adopted MA needs to offer a convincing story regarding ROI
Assumption #1

Don't assume test checking is based on golden results that are manually checked

Lets assume the test environment is automated

an oracle exists

e.g. model based testing
Assumption #2

- Killing a mutant should require a failing testcase
- If we change assumption #1 then this means the testcase includes an oracle and code for checking the expected results

This means redefining strong mutation

To include propagation through the test environment until the indication of a passing/failing testcase
Assumption #3

- Lets redefine the mutation analysis process to include humans interacting with the results.
  - No need to collect all the mutants status before using the information.
  - The order of the mutants is important.
  - The process for results analysis is important.
Assumption #4

- Weak mutation is not a good solution to the performance issue

  the principle value of the Certess product is in finding mutants that are “weak” dead

  mutation is complimentary to coverage information

  focus on the strengths of mutation compared to code coverage – finding problems in the propagation and detection of bugs
Assumption #5

• A mutation analysis metric, based on sampling, can be many orders of magnitude faster than the published results

The Certess product provides a metric in a reasonable runtime even for large systems (100's of CPUs days of testcase execution)
ROI 1/2

• Can an ROI perspective help with the adoption of MA in industry
  
  are we marketing the technology well?

  which niche should be the entry point for software testing?

  where is the most pain?
ROI 2/2

• Is mutation highlighting a more fundamental issue that industry is ignoring?

  measuring the quality of testing should be a separate discipline from testing

  in the same way measuring that quality of the design has been specialized into testing

  should there be "qualifiers" in the same way we have coders and testers?
Mark Harman

King’s College and
University College London
New challenges

• Finding faults is **easy**;
• … there are too many to count …

• So we need to
  • Take fault severity into account
  • Develop test cases
  • Develop fixes
HOM Testing

- First order mutants are not necessarily realistic
- We need tailored adaptive higher order mutation
- We need to take account of fault models
- We need non functional mutants
Jeff Offutt

George Mason University
Jeff Offutt
George Mason University

• Hampton – ROI
• Harman – HOM
Jeff Offutt  
George Mason University

- Hampton – ROI
- Harman – HOM
- Offutt – YAMGG
What ?????
What does that mean?
We can’t even pronounce “YAMGGG”!

• Hampton – ROI
• Harman – HOM
• Offutt – YAMGGG
Research into Mutation Operators
Research into Mutation Operators

Mutation 2000
Research into Mutation Operators

Mutation 2000

Here is the path to build a commercial mutation system.

Jeff
Research into Mutation Operators

**Mutation 2000**

Here is the path to build a commercial mutation system.

I quit.

Jeff
Research into Mutation Operators

Mutation 2000

Here is the path to build a commercial mutation system.

I quit.

Jeff

Yu-Seung Ma
Research into Mutation Operators

Mutation 2000

Here is the path to build a commercial mutation system.

I quit.

I have an idea! OO mutation operators ... muJava

Jeff

Yu-Seung Ma
• Since then, we have had a veritable explosion in languages we can mutate
• We needed mutation operators for each language
  • Java OO, XML, JSPs, HTML, SQL, XML, PhP, inter-class, statecharts, dynamic typing, WS-BPEL, Agents, Actors, …
  • Most of today’s papers present new mutation operators
• And we need a mutation engine for each language!
Mutant Generators
Mutant Generators

Language

Mutation Operators
Mutant Generators

Language

Mutation Operators

Program Object (Base String)
Mutant Generators

- Language
- Mutation Operators
- Program Object (Base String)

Special purpose tool to create mutants
Mutant Generators

- Language
- Mutation Operators
- Program Object (Base String)
- Special purpose tool to create mutants
- Mutants
Mutant Generators

Language

Mutation Operators

Special purpose tool to create mutants

Program Object (Base String)

Mutants

This is a lot of work!
Mutant Generators
Mutant Generators

Language

Mutation Operators

Program Object (Base String)

Special purpose tool to create mutants

Mutants
Mutant Generators

Language

Mutation Operators

Program Object (Base String)

Special purpose tool to create mutants

Mutants
Yet Another Mutant Generator Generator

- Language
- Mutation Operators
- Program Object (Base String)
- Special purpose tool to create mutants
- Mutants
Yet Another Mutant Generator Generator

- Language
- Mutation Operators
- Program Object (Base String)
- Special purpose tool to create mutants
- Mutants
- Grammar for Language
Yet Another Mutant Generator Generator

Grammar for Language

Rules for Mutation Operators

Program Object (Base String)

Mutant Generators

Mutation 2010 Panel
Yet Another Mutant Generator Generator

- Language
- Special purpose tool to create mutants
- Mutants
- Rules for Mutation Operators
- Grammar for Language
- YAMGG

Mutant Generators
Mutation 2010 Panel
Yet Another Mutant Generator Generator

- Language
- Mutation Operators
- Special purpose tool to create mutants
- Mutants
- Program Object (Base String)
- Rules for Mutation Operators
- Grammar for Language
- YAMGG
- Tool to create mutants
Yet Another Mutant Generator Generator

- **Language**
- **Mutation Operators**
- **Program Object (Base String)**
- **Special purpose tool to create mutants**
- **Rules for Mutation Operators**
- **Grammar for Language**
- **YAMGG**
- **Tool to create mutants**
- **Mutants**

**Mutations Generators**

**Operator Mutations**

**Program Object (Base String)**

**Grammar for Language**

**Rules for Mutation Operators**

**Mutants**

**Special purpose tool to create mutants**

**Yet Another Mutant Generator Generator (YAMGG)**
Open Challenges in Mutation Testing

- Hampton – ROI
- Harman – HOM
- Offutt – YAMGG
Open Challenges in Mutation Testing

- Hampton – ROI
- Harman – HOM
- Offutt – YAMGG

Your turn!!