A Critical Analysis of Software Testing Research

Lionel C. Briand, P. Eng.
Simula Research Laboratory
and
University of Oslo
Testing Software Systems

• Testing: Controlled, systematic software execution and verification of results (outputs, system state)

• Systems are increasingly complex: 30% to 60% devoted to testing

• Testing cannot guarantee the absence of faults – no technique can

• Resources available for testing are limited

• Goals:
  – Effective at uncovering faults
  – Automated so as to lower the cost and timescale
  – Help locate faults for debugging
  – Repeatable so that a precise understanding of the fault can be gained and to determine whether it was successfully corrected
  – Systematic so as to be predictable in terms of its effect on dependability
Testing Facets

• Focus
  – Reliability
  – Robustness
  – Safety
  – Security
  – Performance (e.g., response time)

• Mode
  – Typical conditions
  – Stress conditions

A tough and varied engineering problem
Abstract View

Software Representation (Test Model)

Adequacy / Coverage Criteria
- Test cases must cover all the (features) in the model

Based on Roper, 1994

Automation?
Effectiveness?
Cost?

Test Cases
Oracle?
State of Testing Research

• > 30 years, a huge body of research

• Main historical focus:
  – Control and data flow testing (including regression testing)
  – Mutation testing
  – Testing based on finite-state machines
  – Combinatorial designs
  – Predicate testing

• Past trends:
  – Small scale testing, mostly code based
  – Anecdotal empirical evidence
  – Studies based on small artifacts
Current Trends

- Small scale => Large scale
- Code based => Model based
- Functional => Non functional
  - GUI, databases
  - Load, real-time, concurrency
- Algorithmic => Search based
- Analytical => Empirical
Empirical Testing Research

• Understand what the most important problems are in testing practice and determine what should be the priorities of research

• Assess the cost effectiveness of testing techniques, their combination, and the factors affecting it

• For any testing technique to transfer to practice based on a body of evidence about what works and what does not
Empirical Issues

• Many validity issues

• Standard classification
  – Construct validity
  – Internal validity
  – External validity
  – Conclusion validity

We also need to learn how to perform better empirical studies
Construct Validity
Typical Cost-Effectiveness Curve

Andrews et al. 2006

• Type
• Severity
• Detectability

• Uncertainty
• Actual cost?
The Cost Components of Testing

- Test identification
  - Test Model
    - State model
    - IPCFG
    - Activity diagram
  - Requirements
    - Paths
    - Permutations
    - Combinations
    - Value ranges
- Test harness
  - Driver
  - Stubs
- Test execution
- Test diagnosis
  - Oracle
    - Contract assertions
    - State invariants
    - Legal ordering
    - Exact outputs
  - Fault Location
    - Ranking statements
    - Ranking methods
    - Statement or method subset
Measuring Testing Cost

• Test suite size often used as surrogate measure for cost: we assume cost to be proportional to size

• Test suite size can be measured in different ways depending on context, e.g., number of test cases, public method calls

• But using size to compare the cost of test techniques is often inherently flawed, regardless of the size measure

• It does account for all differences across all components in the cost breakdown

• Differences in terms of:
  – Initial test model development cost
  – Automation
  – Error-proneness in devising test cases
  – Etc.
The Effectiveness of Testing

- Percentage of faults detected (distribution)
- Actual versus seeded faults (e.g., mutation)
- Assume a fault population target:
  - Detectability (e.g., detection probability)
  - Type (e.g., integration)
  - Severity
  - Distribution across components (e.g., fault location profile)
Example: Detectability

- Block coverage
- Increasing coverage levels

Andrews et al. 2006
Using Mutation

• Mutation is used in many studies to seed faults
  – many faults, inexpensive (e.g., MuJava)
  – Increased statistical power
  – Independent, objective mechanism

• Can it yield realistic results?

• Initial studies show that yes, if historical fault data is used to select an appropriate subset of mutants

• This needs to be further explored to provide precise guidelines on how to seed faults in test empirical studies
Internal Validity
Many Confounded Factors

- Automation
  - Test identification
  - Test driver
  - Test stubs
  - Test oracles

- SUT
  - Exception Handling
  - Concurrency, distribution

- Fault Profile
  - Type
  - Detectability
  - Severity

- Training & Skills
  - Complexity
  - Correctness
  - Completeness

- Test Model
  - Test Strategy/Criteria
  - Cost-Effectiveness
Interaction Effects

- Interaction Factors
  - Testing Strategy / Criteria
- Independent variable
  - Automation
  - Training & Skills
- Dependent variable
  - Fault Profile
  - SUT
  - Effectiveness
External Validity
Characterizing the Target

• Fault profile
  – Type: state, integration, concurrency, real-time, distribution
  – Location: more faults in larger, more complex components
  – Probability of fault detection

• System
  – Size
  – Distribution, concurrency, real time constraints
  – Exception handling

• Those factors will have a significant impact and must be explicitly characterized to define a target and ensure a clear external validity within an explicit though limited realm
Qualitative Analysis of Undetected Faults

• Investigating why faults aren’t detected provide relevant insights in terms of the conditions that make a testing technique effective.

• One example: Statechart-based testing, round-trip paths criterion fault detection effectiveness depends on (Briand et al., ICSE 2004, Mouchawrab et al., ESEM 2007)
  – Statechart characteristics, e.g., guard conditions
  – The usage of contract assertion oracles
  – Real-time behavior of the system
Conclusion Validity
Random Variation

• Typically many test suites are adequate for a test coverage criterion (e.g., transition coverage in statecharts)
• Test suites may lead to different cost or effectiveness scores
• This random variation must be assessed in a realistic manner
• Two common types of empirical studies
  – Simulations: Variability in test pools
  – Experiments: Multiple testers/teams
• Also random variation due to selected sample of faults
Simulations versus Experiments

- Large pool of test cases is built beforehand
- Systematic, random sampling process to form large number of adequate test suites
- Increased power
- Adequate test suites
- Realism of results unclear (external validity)

- Human factors
- Test suites are formed by subjects
- Constrained by experiment time
- Not always fully adequate test suites
- Analysis more complex
- When adequate training, representative tooling: realistic
Typical Problems in Studies

• Test suite size = cost
• Undefined fault targets (severity, type, detectability)
• Human factors not accounted for
• No thorough investigation of uncertainty, random variation
Beyond Assessing Test Suites

• Evolutionary testing
  – Using search techniques (e.g., genetic algorithms) to automate testing (e.g., test case generation)
  – Heuristics => therefore need empirical assessments

• Debugging aid
  – Help locating faults
  – Heuristics based on coverage of failed and successful test cases (e.g., Tarantula)
  – Heuristics => therefore need empirical assessments
Stress Testing for RT Systems

Goal: to automate, based on a model the system task architecture, the derivation of test cases that maximize the chances of critical deadline misses within real-time systems.

[Briand, Labiche, Shousha, 2003, 2006]
Debugging Example

Tarantula

RUBAR: Using Machine learning

Briand et al., 2007
State of Testing Practice

- Little automation, except for mechanical tasks
- Little effort is invested in test automation endeavors
- No systematic strategies
- Even basic techniques, e.g., control flow coverage, are rarely used
- Testers are still considered, in many organizations, as second-class developers
- Why is there such a gap between practice and research?
Root Causes of the Practice/Research Fracture

• Education (most testers have not been formally trained)

• Little is known about the cost-effectiveness of most (if not all) techniques

• Research focus on techniques that do not scale up

• Commercial tools are far behind research, e.g., many coverage tools don’t implement even branch/edge coverage correctly

• Techniques without proper automation are essentially not applicable

• Cost of late fault discovery and failures not known in most organizations: the case for more investment in testing is often hard to make
Remedies

• Field studies: Real problems
  – What faults
  – Why
• Abstraction: model based testing
  – Scale up
  – Focus
  – Conformance
• Automation through search
  – Pushing the limits of automation
  – Evolutionary algorithms
• Quantitative methods, Empirical studies
  – Realistic cost effectiveness analysis
  – Quantitative management
Thank you!
Comparison of Mutant and Fault Detection Ratios
Random Variation in Fault Detection Effectiveness: Statechart Example
Interaction Effect Example

[ simula.research.laboratory ]

Back
IEEE int. conference on Software Testing, Verification, and Validation

- Lillehammer, Norway
- Submission deadline: October 12
- Both research and industry tracks
- Charter ensures transparent, open management and renewal