An Empirical Framework for Comparing Effectiveness of Testing and Property-Based Formal Analysis

Jeremy S. Bradbury, James R. Cordy, Juergen Dingel

Software Technology Laboratory
School of Computing
Queen's University
Kingston, Canada

SUPPORTED BY NSERC CRSNG

Background

- There is an increase in the practicality of formal analysis.
  - e.g. automatic, scalable tools that can directly analyze source code
- In the next few years applications will need to be concurrent to fully exploit CPU throughput gains [Sut05].
- Formal analysis can often succeed at debugging concurrent systems while testing can be insufficient or impractical.

[Sut05] H. Sutter. The free lunch is over: A fundamental turn toward concurrency in software. *Dr. Dobb's Journal*, 30(3), Mar. 2005.

Goals of Proposed Study

- Development of a quantitative assessment framework to empirically evaluate the following open problems:
  - How good is property-based formal analysis at finding bugs in source code?
  - How efficient is a formal analysis technique at finding bugs in comparison to testing or in comparison to another formal analysis technique?
  - Can a hybrid approach that combines formal analysis and testing ever find more bugs or be more efficient than either approach used in isolation?
  - …

Experimental Setup

- **Quantity of bugs detected**
  - Mutant score = percentage of non-equivalent mutants detected (*killed*) by a test suite or property set

Selection of Metrics

Experimental Setup

- **Quantity of bugs detected**
  - Mutant score = percentage of non-equivalent mutants detected (*killed*) by a test suite or property set
- **Efficiency of bug detection**
  - Execution cost = the time to run each test case or test suite
  - Verification cost = the time to verify each property or property set
  - Cost to kill a mutant = the time to run a test case or verify a property that kills the mutant averaged over the number of mutants killed by the test case/property

Selection of Metrics

Experimental Setup

- Testing approaches
  - Test suites developed using standard coverage technique – e.g. branch coverage.
- Formal analysis approaches
  - Static analysis – Path Inspector
  - Model checking – Bogor

Selection of Metrics

Selection of Testing and Formal Analysis Techniques

Experimental Setup

- We have tried to find industrial example programs that have
  - a mature test suite
  - an existing property specification
- To start, we will use the Siemens programs used in testing community
  - a pattern replace program
  - priority schedulers
  - lexical analyzers
  - …

Selection of Metrics

Selection of Testing and Formal Analysis Techniques

Selection of Example Programs, Test Suites, and Property Sets

Experimental Setup

- We have tried to find industrial example programs that have
  - a mature test suite
  - an existing property specification

← Difficult!

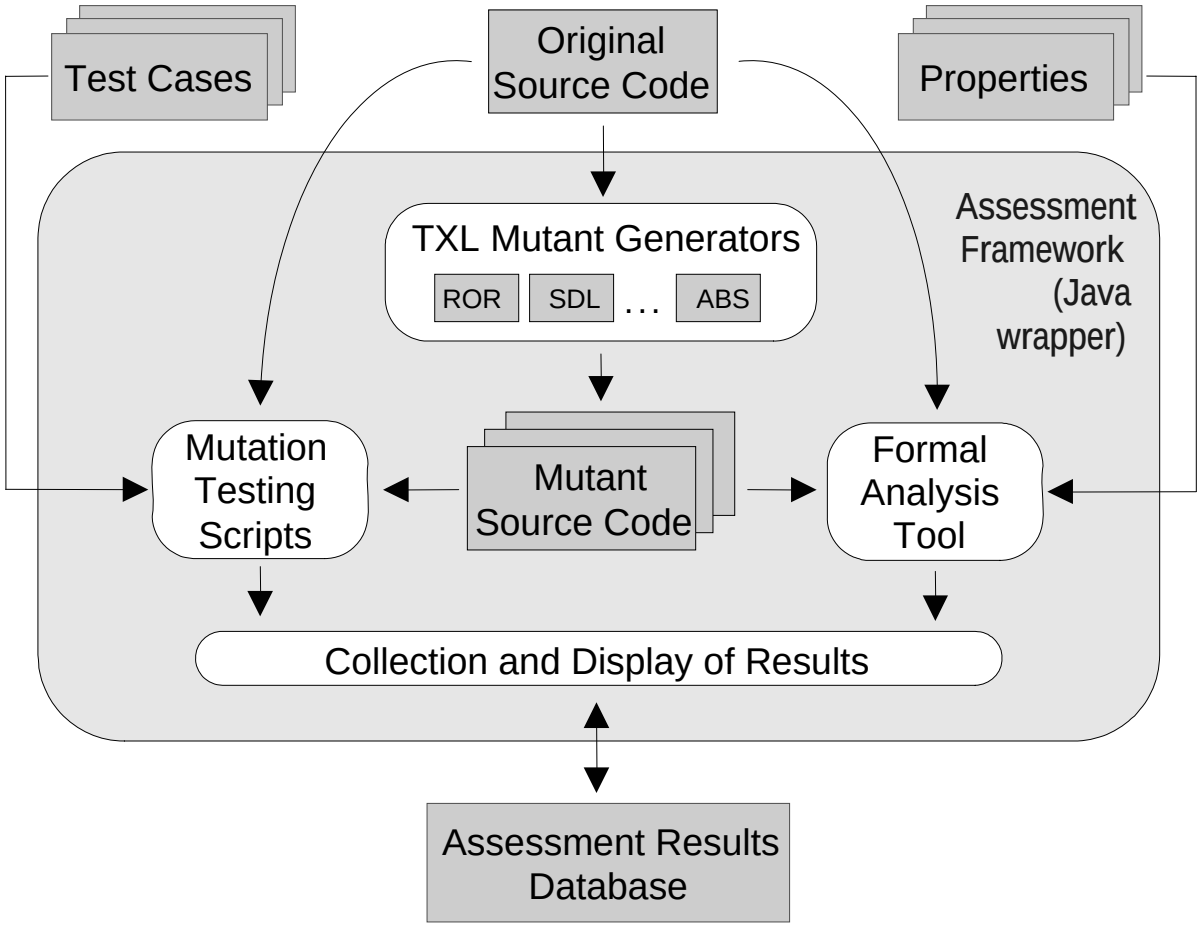
- To start, we will use the Siemens programs used in testing community
  - a pattern replace program
  - priority schedulers
  - lexical analyzers
  - …

Selection of Metrics

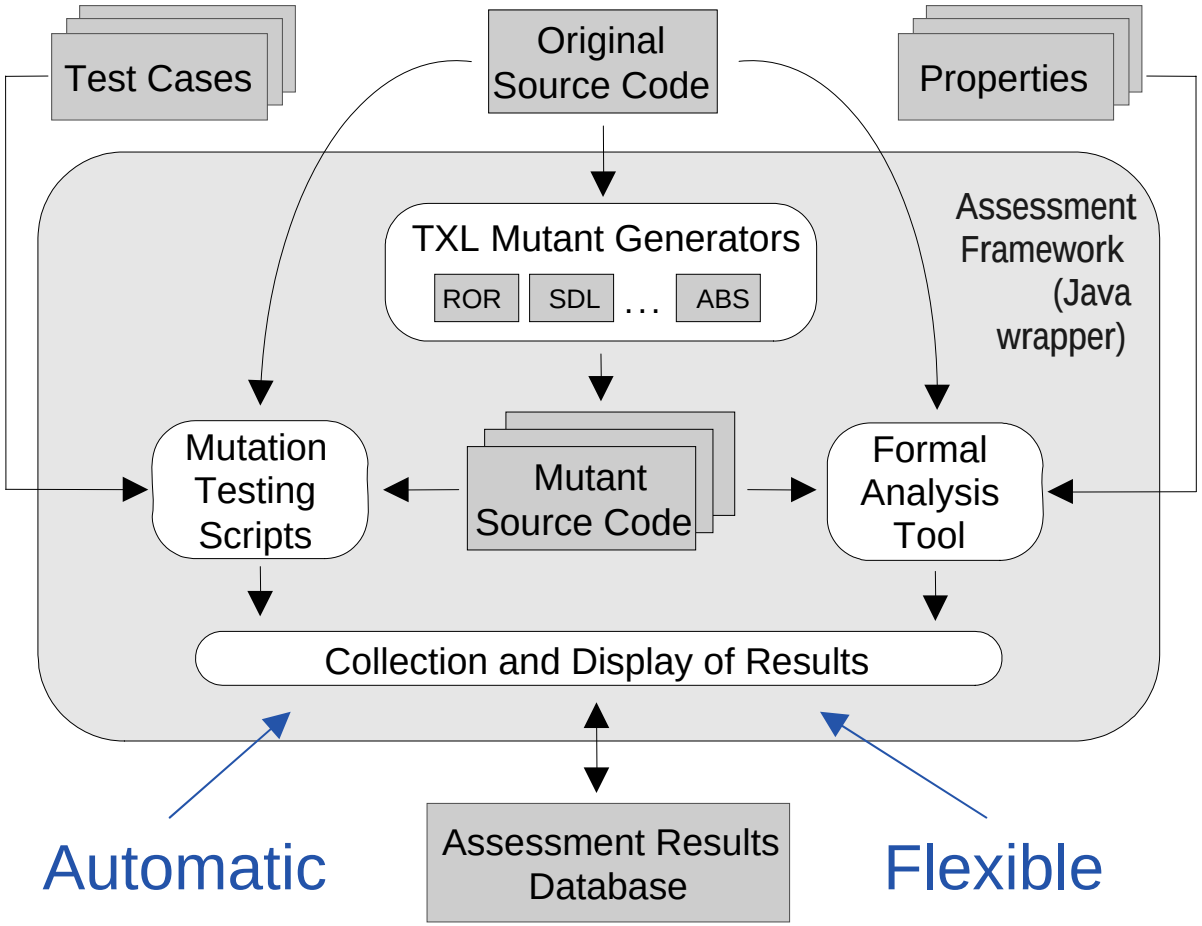
Selection of Testing and Formal Analysis Techniques

Selection of Example Programs, Test Suites, and Property Sets

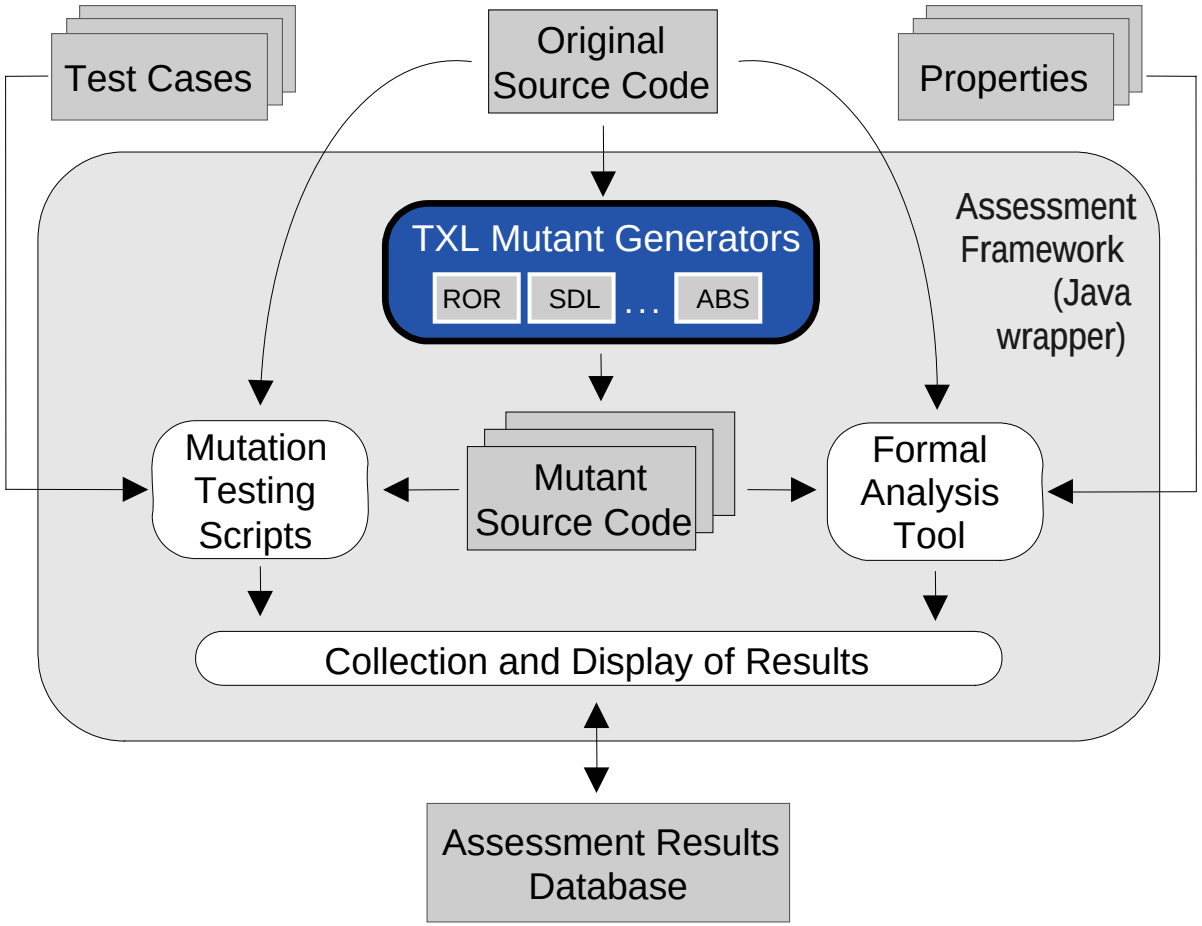
Experimental Procedure



Experimental Procedure

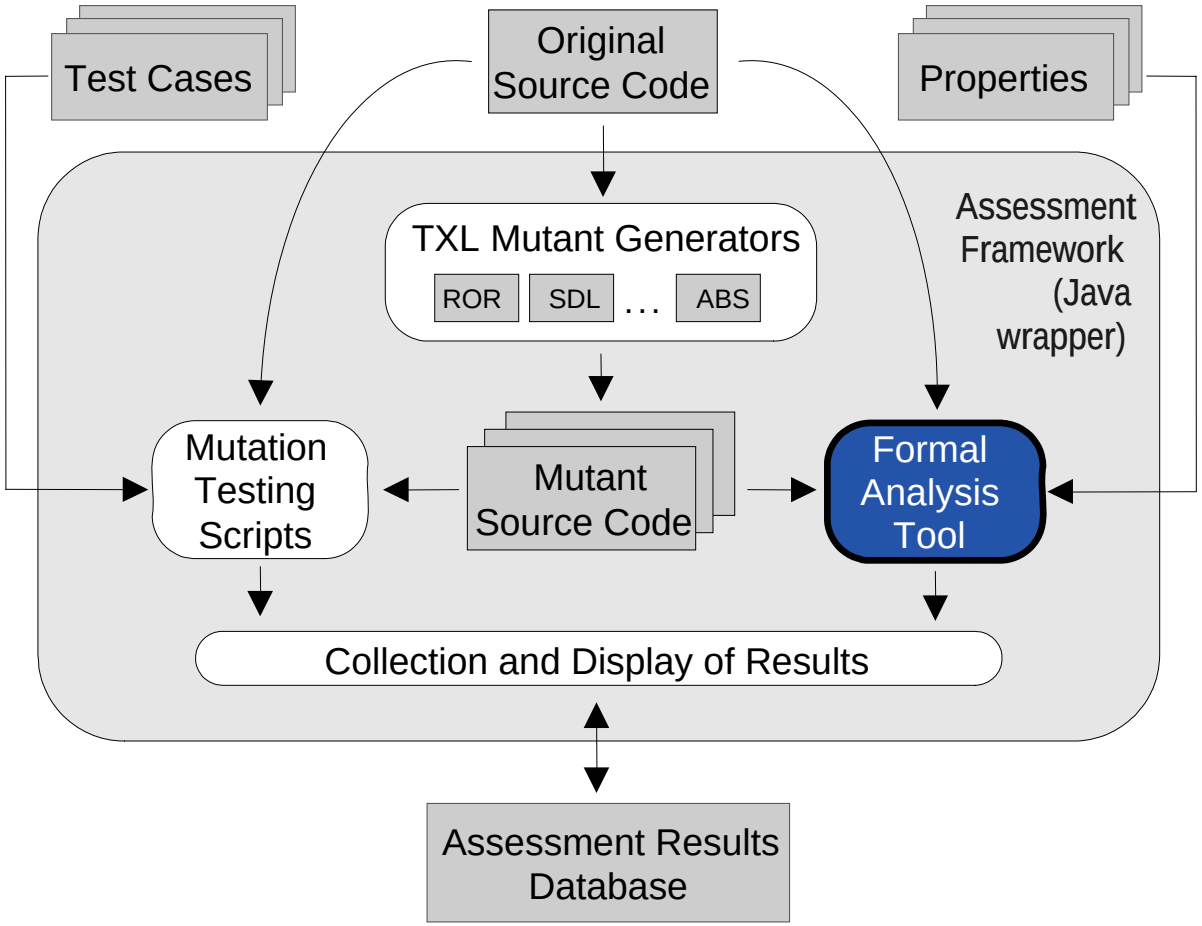


Experimental Procedure



Mutation generation

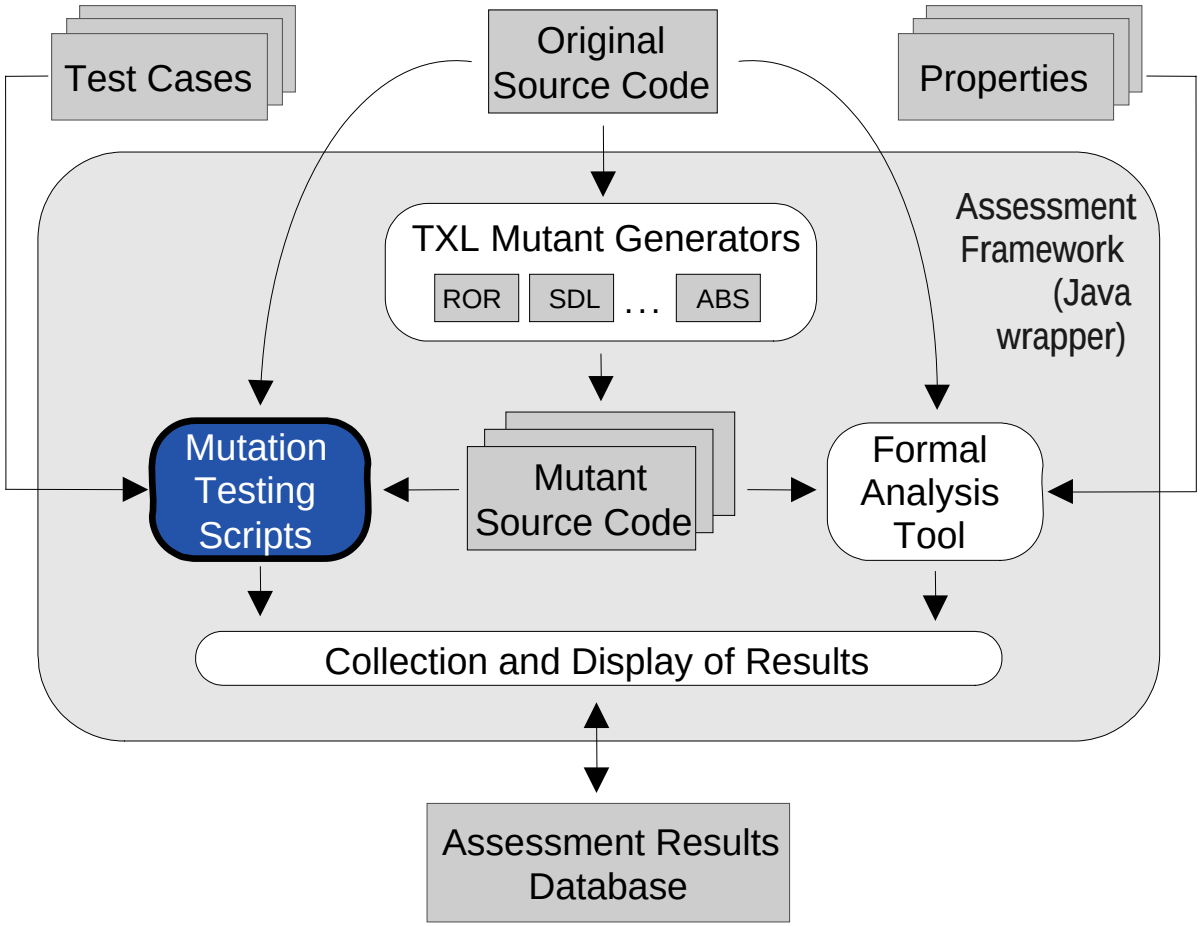
Experimental Procedure



Mutation generation

Formal Analysis

Experimental Procedure

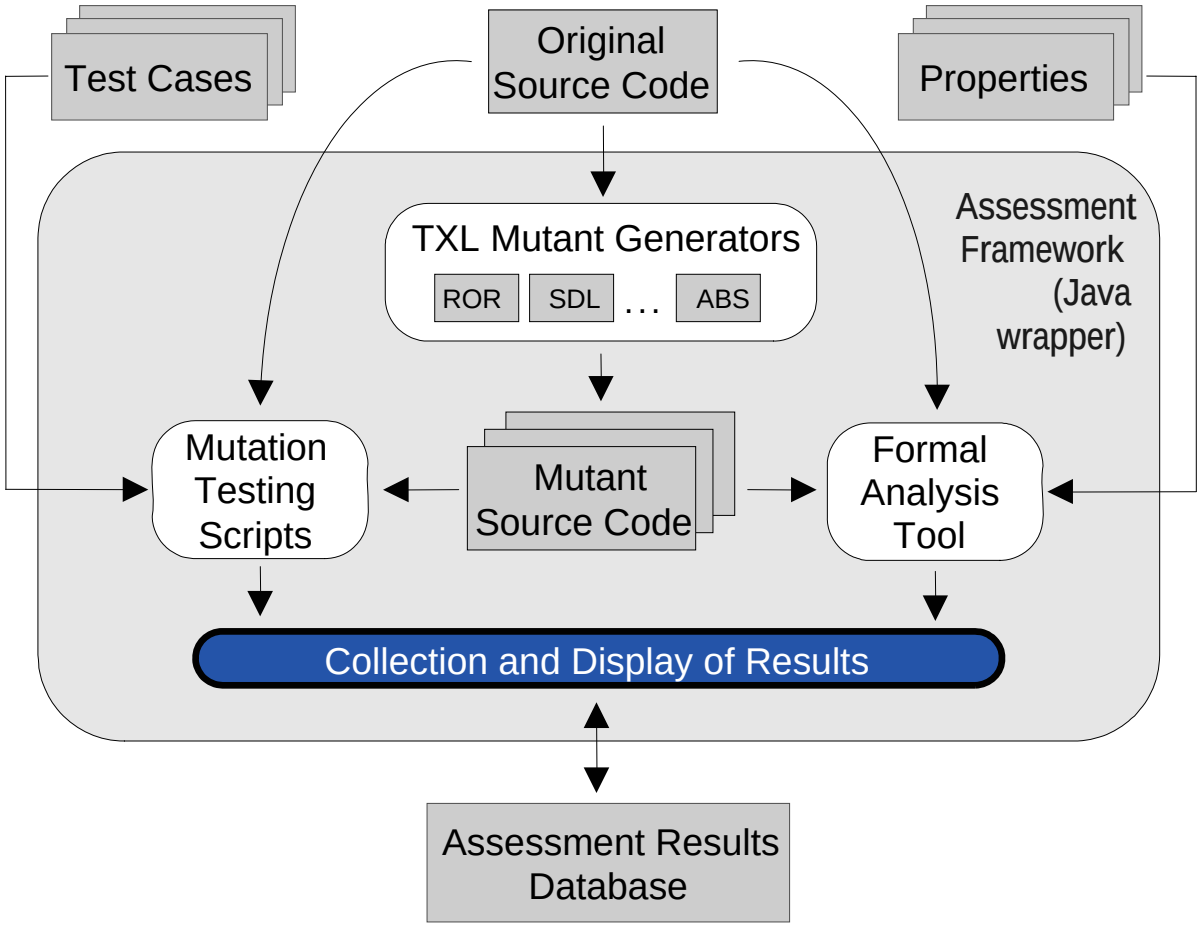


Mutation generation

Formal Analysis

Testing

Experimental Procedure



Mutation generation

Formal Analysis

Testing

Collection and display of results

Experimental Procedure

- For properties we report:
  - mutant score, verification cost, number of properties that kill each mutant.
  - the relationship between mutants killed vs. property patterns.

Mutation generation

Formal Analysis

Testing

Collection and display of results

Experimental Procedure

- For properties we report:
  - mutant score, verification cost, number of properties that kill each mutant.
  - the relationship between mutants killed vs. property patterns.
- For tests we report:
  - mutant score, execution cost, number of test cases that kill each mutant.

Mutation generation

Formal Analysis

Testing

Collection and display of results

Experimental Procedure

- For properties we report:
  - mutant score, verification cost, number of properties that kill each mutant.
  - the relationship between mutants killed vs. property patterns.
- For tests we report:
  - mutant score, execution cost, number of test cases that kill each mutant.
- For hybrid approaches we examine sets of tests and properties with:
  - highest mutant score
  - lowest execution cost (or smallest set) given a mutant score.

Mutation generation

Formal Analysis

Testing

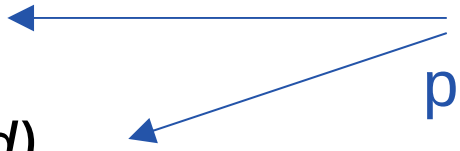
Collection and display of results

Contributions

- A mutation-based method for quantitatively evaluating bug detection with respect to:
  - property-based analysis and testing
  - different property-based analysis techniques
  - different sets of properties
  - different types of properties (assertions vs. LTL)
  - …
- Automatic experimental assessment framework
- Empirical data *(expected)*

Contributions

- A mutation-based method for quantitatively evaluating bug detection with respect to:
  - property-based analysis and testing
  - different property-based analysis techniques
  - different sets of properties
  - different types of properties (assertions vs. LTL)
  - …
- Automatic experimental assessment framework
- Empirical data *(expected)*



Will be made publicly available

An Empirical Framework for Comparing Effectiveness of Testing and Property-Based Formal Analysis

Jeremy S. Bradbury, James R. Cordy, Juergen Dingel

Software Technology Laboratory
School of Computing
Queen's University
Kingston, Canada

SUPPORTED BY NSERC CRSNG