Rapid Evolution of Software Tests

Kevin L. Mills

Information Technology Laboratory
Rapid Evolution of Software Tests (REST)

Evolutionary Test Generator

Fitness Function

Surviving Test Campaign

Test Language

Population of Test Campaigns

Parallel Execution of Test Campaigns on Software Under Test

1/21/99

Information Technology Laboratory
Rapid Evolution of Software Tests (REST)

Goal: Develop software to automatically generate measurably effective tests for information technology standards.

New Ideas

- Use evolutionary algorithm to generate tests from test language.

- Evolve automatically from previous best tests to generate tests when system changes.

- Use software test coverage against reference implementation to evaluate tests.

- Execute generated tests in parallel on network of workstations.
Rapid Evolution of Software Tests (REST)

RESEARCH ISSUES

- Identifying Effective Genetic Operators
  - Cross-over: single slicing versus multiple slicing and recombination of slices versus riffling recombination
  - Mutation: statement mutation, sequence insertion, or sequence deletion

- Discovering Effective Genetic Representations
  - Structure: sequences or trees of sequences
  - Code: binary or syntax-driven

- Evaluating Alternative Fitness Functions
  - Coverage Metrics: code coverage, data-flow coverage, or mutation coverage
  - Other Metrics: cost and combinations of coverage and cost
Rapid Evolution of Software Tests (REST)

IMPACT

REDUCE COST TO TEST SYSTEM

Practice: 25-50% Cost of Software
Goal: 5-10% Cost of Software

IMPROVE QUALITY OF TESTS

Practice: Test Suite Composition is a Judgment Call
Goal: Coverage and Cost Metrics Used to Evaluate

REDUCE TIME TO TEST AVAILABILITY

Practice: Months to produce tests
Goal: Days to produce tests

REDUCE TIME TO ADAPT TESTS

Practice: Weeks to revise tests
Goal: Hours to revise tests
### Rapid Evolution of Software Tests (REST)

#### SCHEDULE

<table>
<thead>
<tr>
<th>YEAR 1 - BUILD PROOF-OF-CONCEPT PROTOTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverables:</td>
</tr>
<tr>
<td>• Prototype evolutionary test generation software</td>
</tr>
<tr>
<td>• Demonstration of application to test a real-time application of 2.5KLOCs</td>
</tr>
<tr>
<td>Decision: Generated tests achieve at least 90% branch coverage.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>YEAR 2 - EXPLORE GENETIC OPERATORS AND FITNESS FUNCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverables:</td>
</tr>
<tr>
<td>• Robust evolutionary test generator software</td>
</tr>
<tr>
<td>• Evaluation of various genetic operators and fitness functions</td>
</tr>
<tr>
<td>• Demonstration of application to an ITL problem</td>
</tr>
<tr>
<td>Decision: Generated tests achieve feasible coverage against selected ITL problem.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>YEAR 3 - APPLY TO INDUSTRIAL-STRENGTH PROBLEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverables:</td>
</tr>
<tr>
<td>• Final evolutionary test generator software available for download</td>
</tr>
<tr>
<td>• Demonstration of application to an industry problem</td>
</tr>
</tbody>
</table>
Rapid Evolution of Software Tests (REST)  
RESOURCES

YEAR 1 - BUILD PROOF-OF-CONCEPT PROTOTYPE

1 Staff Researcher
2 Half-time programmers

Access to Network of Workstations

YEAR 2 - EXPLORE GENETIC OPERATORS AND FITNESS FUNCTIONS

2 Staff Researchers
2 Half-time programmers

Access to Network of Workstations and Available ITL Problem

YEAR 3 - APPLY TO INDUSTRIAL-STRENGTH PROBLEM

1 Staff Researchers
2 Full-time programmers

Access to Network of Workstations and Available Industry Problem