Static Estimation of Test Coverage

Tiago Alves & Joost Visser

May 29, 2008  Arent Janszoon Ernststraat 595-H  
NL-1082 LD Amsterdam  
info@sig.nl  
www.sig.nl
Introduction

Background

• 2004 - Computer Science and Systems Engineering, University of Minho, Braga
• 2006 - MSc in Informatics
  • Grammar engineering (ISO VDM-SL Grammar in SDF)
  • SdfMeta: Metrication of syntax formalisms (SDF, DMS, Antlr, and Bison)
  • VooDooM: model transformation and code generation (VDM-SL -> SQL)
  • Teaching compiler course 3rd year students
• 2006 - European Space Operations Center (ESA), Damstadt, Germany
  • Team member responsible for managing development of a prototype system
  • Requirements specification for new software system
  • Acceptance test of a communication system.
• 2007 - PhD at University of Minho and Software Improvement Group
  • 2LT Extensions: Constraint-aware transformations
  • Static estimation of test coverage
Measuring testing coverage

Pros:
- Indicator for test quality
- Indicator for quality of the software under test
  - Higher coverage => better software quality (in principle)

Cons:
- Tied with software development process
  - Full installation required (sources + libraries)
- Instrumentation of source/byte code (problematic in embedded systems)
- Execution (Hardware or time constraints)

- Not appropriate to compute in the context of software quality assessment!!
Research Challenge

13th Testdag, Delft, November 2007

- I. Heitlager, T. Kuipers, J. Visser “Observing unit test maturity in the wild”

Research questions:

- Is it possible to determine test coverage without running tests?
- What trade-offs can be made between sophistication and accuracy?

Requirements

- Use only static analysis
- Scale to large systems
- Robust against incomplete systems

Static estimation of test coverage
Solution sketch

1. **Extract**
   - Extract structural and call information
   - Determine set of test classes

2. **Slice (modified)**
   - Slice graph starting from the test methods
   - Set of methods reached from test code
   - Take into account class initializer calls

3. **Count (per class)**
   - Determine number of defined methods
   - Determine number of covered methods

4. **Estimate**
   - Class coverage
   - Package coverage
   - System coverage
Modified slicing

Binary relational expression

\[ \begin{align*}
    n \xrightarrow{\text{call}} m & \quad m \xleftarrow{\text{def}} c \\
    n \xrightarrow{\text{init}} m = n \xrightarrow{\text{call}} m_i \xleftarrow{\text{def}} c \xrightarrow{\text{call}} m \\
    n \xrightarrow{\text{invoke}} m = n \xrightarrow{\text{call}} m & \mid n \xrightarrow{\text{init}} m \\
    n \xrightarrow{\text{invoke}} + m
\end{align*} \]

SemmleCode implementation

\[
\text{predicate isTestCovered()} \{ \\
    \exists(\text{TestClass } tc, \text{ Callable } tm \mid \text{tc.contains(tm)} \land \text{tm.polyCalls+}(\text{this})) \lor \exists(\text{Callable } m \mid m.\text{hasName}("<\text{clinit}>") \land m.\text{polyCalls+}(\text{this})) \\
\}
\]
What can go wrong?  
(Sources of imprecision)

Java language

- Control flow
  - Conditional statements (if-then, if-then-else)
  - Switch statements (switch, case)
  - Looping statements (for, while, do-while)
  - Branching statements (break, continue, return)
- Dynamic dispatching
  - Inheritance
- Overloading

General issues

- Frameworks / Libraries call backs
- Identification of test code
  - Test code is recognized by determining JUnit dependencies
- ///CLOVER:OFF flags

Static estimation of test coverage
Dealing with imprecision

Pessimistic approach
- Report only what can be determined to be true
- False negatives
- Estimates lower bound for coverage

Optimistic approach
- Report everything that might be true
- False positives
- Estimates upper bound for coverage

Pessimistic vs. Optimistic (software assessment context)
- Pessimistic will always report low coverage
- Optimistic will be sensitive to lack of coverage
Experimental design

Data set selection and characterization
- Open-source and proprietary Java systems
- Available clover report (XML or HTML)

Execution of experiment
- SemmleCode execution (text file export + scripts for CSV conversion)
- XML Clover extraction (XSLT transformations to CSV conversion)
- HTML Clover extraction (grep, sed, awk, wc scripts to CSV conversion)
- Custom built java tool to read CSV files and XLS creation

Statistical analysis
- Histograms (distribution)
- Scatter charts (correlation)
- Spearman (correlation)
- Inter-quartile ranges (dispersion)
### Data set characterization

<table>
<thead>
<tr>
<th>System name</th>
<th>LOC</th>
<th>#Packages</th>
<th>#Classes</th>
<th>#Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pacman</td>
<td>2987</td>
<td>2</td>
<td>20</td>
<td>181</td>
</tr>
<tr>
<td>G System</td>
<td>6265</td>
<td>15</td>
<td>53</td>
<td>385</td>
</tr>
<tr>
<td>Utils</td>
<td>23604</td>
<td>35</td>
<td>260</td>
<td>2571</td>
</tr>
<tr>
<td>Dom4j</td>
<td>42863</td>
<td>14</td>
<td>144</td>
<td>2481</td>
</tr>
<tr>
<td>PMD</td>
<td>51219</td>
<td>40</td>
<td>455</td>
<td>3398</td>
</tr>
<tr>
<td>Architect</td>
<td>58477</td>
<td>17</td>
<td>220</td>
<td>2781</td>
</tr>
<tr>
<td>DepFinder</td>
<td>73861</td>
<td>12</td>
<td>261</td>
<td>4686</td>
</tr>
<tr>
<td>R System</td>
<td>79776</td>
<td>62</td>
<td>600</td>
<td>5620</td>
</tr>
</tbody>
</table>

Static estimation of test coverage
Statistical analysis
(System coverage comparison)

<table>
<thead>
<tr>
<th>System</th>
<th>Static</th>
<th>Clover</th>
<th>Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pacman</td>
<td>84.53%</td>
<td>90.61%</td>
<td>-6.08%</td>
</tr>
<tr>
<td>G System</td>
<td>88.37%</td>
<td>94.81%</td>
<td>-6.44%</td>
</tr>
<tr>
<td>Utils</td>
<td>72.95%</td>
<td>68.73%</td>
<td>4.22%</td>
</tr>
<tr>
<td>Dom4j</td>
<td>60.69%</td>
<td>45.20%</td>
<td>15.49%</td>
</tr>
<tr>
<td>PMD</td>
<td>77.77%</td>
<td>65.50%</td>
<td>12.27%</td>
</tr>
<tr>
<td>Architect</td>
<td>48.98%</td>
<td>35.30%</td>
<td>13.68%</td>
</tr>
<tr>
<td>DepFinder</td>
<td>62.14%</td>
<td>70.08%</td>
<td>-7.94%</td>
</tr>
<tr>
<td>R System</td>
<td>64.54%</td>
<td>72.46%</td>
<td>-7.92%</td>
</tr>
</tbody>
</table>
## Statistical Analysis
(Class and package coverage comparison)

<table>
<thead>
<tr>
<th>System name</th>
<th>Spearman</th>
<th>Median</th>
<th>Inter-quartile range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Class</td>
<td>Package</td>
</tr>
<tr>
<td>Pacman</td>
<td>0.275</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>G System</td>
<td>0.777**</td>
<td>0.694**</td>
<td>0</td>
</tr>
<tr>
<td>Utils</td>
<td>0.737**</td>
<td>0.825**</td>
<td>0</td>
</tr>
<tr>
<td>Dom4j</td>
<td>0.557**</td>
<td>0.625*</td>
<td>0.17</td>
</tr>
<tr>
<td>PMD</td>
<td>0.702**</td>
<td>0.693**</td>
<td>0</td>
</tr>
<tr>
<td>Architect</td>
<td>0.504**</td>
<td>0.5*</td>
<td>0</td>
</tr>
<tr>
<td>DepFinder</td>
<td>0.659**</td>
<td>0.396</td>
<td>0</td>
</tr>
<tr>
<td>R System</td>
<td>0.752**</td>
<td>0.652**</td>
<td>0</td>
</tr>
</tbody>
</table>

Static estimation of test coverage
R System: detailed statistical analysis
(Class coverage histograms comparison)
R System: detailed statistical analysis
(Class coverage comparison + differences)

Static estimation of test coverage
R System: detailed statistical analysis
(Package coverage histograms comparison)

Clover

Static

Static estimation of test coverage
R System: detailed statistical analysis
(Package coverage comparison + differences)

Static estimation of test coverage

Static and clover coverage at package level
Conclusion

Is it possible to determine test coverage without running tests?

- Yes!!!
- Spearman: high correlation between static and clover coverage
- In general static coverage identifies the same values as clover

What trade-offs can be made between sophistication and accuracy?

- Average absolute difference for system coverage: 9%
- Class and Package coverage needs further improvement

Implementation

- SemmleCode: 92 LOC = 76 LOC (extensions) + 16 LOC (3 Queries)
- SIG Monitor: 265 LOC = 136 + 56 + 22 + 23 + 14 + 15 (6 classes)
Future work

Implementation:
- Add analysis to production at SIG (done)
- Add tests (in progress)

Research:
- Use LOC as a weight for better estimation of coverage
- Compute static levels of testing
  - T. Kanstrén. *Towards a deeper understanding of test coverage*
- Investigate the use of McCabe + #Tests + #asserts + Test(LOC) / Code(LOC)
Questions?