Interactive Coverage Effectiveness Multiplots for Evaluating Prioritized Regression Test Suites
Adam M. Smith,1, Joshua J. Geiger,1 Gregory M. Kapfhammer, Manos Renieris,2 and G. Elisabeta Maral1
1Department of Computer Science, University of Pittsburgh 2Department of Computer Science, Allegheny College 3Google
Presented at the IEEE Information Visualization Conference (IEEE InfoVis 2009), Atlantic City, NJ

SOFTWARE TESTING CHALLENGES

- Complex source code, databases, files, and network communication
- Defects may exist in the individual components or their interactions
- Testing isolates defects and establishes confidence in the correctness of software

LIMITATIONS FOR TESTERS

- Many prioritization methods exist because finding the highest CE by evaluating all orderings of a test suite is too expensive
- Each of these prioritization techniques can have many configurations from which to choose
- Testers relying on static coverage effectiveness multiplots, such as Figure 4, and/or large tables of CE scores and test orders can be easily overwhelmed
- Existing visualizations assist during different development processes such as manual debugging and automated fault localization [2, 5]

REGRESSION TESTING

When a test case is modified, new tests run in addition to the old, increasing the test suite size
Execution time of a test suite may be prohibitive
Prioritization techniques re-order the tests to locate defects earlier in the test execution process
Coverage reports identify points in the source code, executed, or covered, by each test case
Prioritizers must analyze the requirements covered by each test case to effectively re-order the test suite

EVALUATING TEST SUITES

- Coverage Effectiveness (CE) rates test suites by how quickly they cover each requirement [4]
- Prioritize to increase the CE of a test suite where CE = Actual/Expected [0, 1]

Figure 1: What is a test case? Each test case invokes a method within the program and compares the actual and expected output values.

A sequence of test cases is a test suite
A test suite executor such as JUnit runs each test case independently

Figure 2: Regression testing. A test suite will be executed repeatedly throughout development, searching for faults introduced by changes made to the software.

Prioritizer

Set Up
Input
Method Under Test
Test Oracle
Output
Verdict
Test Data

Visualization Using Interactive Multiplots and Interactive Multiplots

- Enable software testers to quickly find the best test suite order for their own applications
- Interactively pick prioritizers, comparing CE values and the actual ordering of the tests
- Utilize prioritization techniques such as greedy (GRD), 2-optimal greedy (2OPT), delayed greedy (DGR), and Harold Gupta Softa (HGS) which use greedy choice metrics (GCMs) to efficiently construct new test orders [7]

Figure 3: Calculating Coverage Effectiveness (CE). The CE score is the area under C(T,J) divided by the area under the ideal test suite function (dashed line). Cover (T,J) denotes the set of requirements covered by test T

Figure 4: Static Coverage Effectiveness Multiplot. Multiple lines severely clutter the visualization making evaluation and comparison of prioritized test suites nearly impossible.

Figure 5: Greedy approaches to test prioritization. Re-order the test suites by repeatedly performing reduction.

- Make use of the potential power of reverse and random prioritizations [7]
- Encourage empirical study on the use of visualization during test suite prioritization

REFERENCES

Visualization Features

- Multiplot of cumulative coverage step functions of all selected prioritization methods
- Toggle display of cumulative coverage functions for each prioritization method
- Color coding of prioritization techniques for easy identification
- Adjust the number of random prioritizations
- Display cumulative averages and standard deviations of random prioritizations

Figure 6: Interactive Coverage Effectiveness Multiplot in RAISE. Visualization using Interactive Multiplots and details on demand allows the users to quickly filter, evaluate and compare prioritized test suites.

Left Panel
- Displays information about the test suite and controls multiplot display
- Toggle display of cumulative coverage functions for each prioritization method
- Color coding of prioritization techniques for easy identification
- Adjust the number of random prioritizations
- Display cumulative averages and standard deviations of random prioritizations

Right Panel
- Multiplot of cumulative coverage step functions of all selected prioritization methods
- Mouse-over of plots highlights the line and shades the area under the line
- Mouse-over a line reveals its corresponding prioritizer, GCM, and CE score
- Vertical axis to display the number of covered requirements and horizontal axis to show the test suite execution time

Figure 7: http://raise.googlecode.com/ provides tools, data sets and resources.

CONCLUSIONS AND FUTURE WORK

- An interactive visualization that enables the evaluation of prioritized regression test suites
- Free and open source Reduce And prioritize SuiteEs (RAISE) system available for download
- Intend to add new features and conduct more experimental studies
- Will extend RAISE to support other metrics like average percentage of faults detected (APFD) and average percentage of requirements covered (APRC)
- RAISE will serve as a simple and valuable tool in a comprehensive framework supporting all of the phases in the regression testing process