Actual Test Coverage

Mark Timmer, University of Twente

September 9, 2008

Testing is inherently incomplete; no test suite will ever be able to test all possible usage scenarios of a system. Therefore, in the past decades many coverage measures have been developed. These measures denote the portion of a system that is tested, that way providing a quality criterion for test suites.

Known examples of coverage measures in white box testing are statement coverage and path coverage. In black box testing much research has been devoted to state and transition coverage using finite state machines. Potential drawbacks of finite state machines, though, are that they are deterministic and not compositional. This limits their usability for embedded systems.

Recently a coverage measure that is suitable for embedded systems has been introduced by Brandán Briones, Brinksma and Stoelinga: semantic coverage. It measures how many erroneous traces might be uncovered by a given test suite. A possible disadvantage of semantic coverage is that it focusses on which faults can *potentially* be detected. In practice, however, a single execution or even several executions of a test suite will almost never detect all faults that could potentially be detected, due to non-determinism of the system outputs.

In this talk we introduce a framework on *actual test coverage*. This measure denotes the number of faults *actually* shown present or absent. Our framework provides a method to *evaluate* the actual coverage of a given set of test suite executions after testing has taken place, and also one to *predict* the actual coverage a certain number of executions will yield. Both the evaluation afterwards and the prediction in advance are quite efficient, making it feasible to implement the theory in a tool and use it in a practical context.

Our methods are behavioural, in the sense that actual coverage is invariant to syntactic changes that do not affect system behaviour. Moreover, faults can be given a weight, incorporating their severity.

The framework is based on a probabilistic execution model, describing the probabilistic output behaviour of a system. Using this information, we can estimate how many faults on average will be detectable during a testing process. Moreover, we use estimations of the probability with which faults occur in case they are present, enabling us to calculate the probability of their absence when a given number of executions did not show their presence.

In conclusion, our framework can evaluate the effect of a test suite after testing has taken place and predict the effect of a given number of test suite executions. It is therefore useful for both test evaluation and test selection.

We will illustrate our methods using a small example of a chemical dispenser.