Search-Based Testing of Complex Simulink Models containing Stateflow Diagrams

Andreas Windisch
Technische Universität Berlin
Daimler Center for Automotive IT Innovations
Berlin, Germany
tu-berlin.windisch@daimler.com

1 Motivation

Evolutionary structural testing is an approach to generate test cases automatically that achieve high structural code coverage. This white box approach has been established for being used on the code level, but so far only little work has been done to apply it to system models, such as MATLAB/SIMULINK/STATEFLOW models.

The position taken in the research summarized here is that structural testing of real-world SIMULINK models containing STATEFLOW charts is possible. Ideas for necessary problem solutions are proposed.

2 Proposed Solution

Considering the task of structural testing of SIMULINK models containing STATEFLOW diagrams, a number of difficulties arise that need to be addressed. One of the main problems is the size of the model under test and the resulting *complexity*. Other major challenges are the *signal generation* as well as the *coverage of stateflow diagrams* in general.

Coverage of Stateflow Diagrams The principles of white box structural testing of procedural code (e.g. [2]) may also be applicable to STATEFLOW diagrams. In the context of STATEFLOW diagrams, one test goal would aim at the achievement of a particular state or transition segment or the execution of a specific transition action. An objective function is constructed for each test goal which guides the evolutionary input sigal generator to identify input that triggers transitions of the STATEFLOW model so as to eventually achieve the test goal. For the construction of an appropriate objective function, transition-specific distance functions are applied similar to the distance functions used for procedural software.

Dealing with Complexity The complexity of the model under test has a big influence on the success of the test due to increasing execution times. Therefore, submodels of the model should be optimized separately. Models usually are already structured hierarchically by using SIMULINK *subsystem* blocks. If not, complexity measures can be used to create reasonable subsystems in various ways. Subsequently, this information can be used to process the subsystems in a reasonable way. Depending on the current test goal and the coverage criterion chosen the necessary input signals can then be determined in a way similar to Zhan's approach [3].

Signal Generation Due to the size of the inputs to be optimized when using continuous signals, a differentiation between optimization and simulation sequences needs to be accomplished. The optimization sequence consists of only a small number of parameters to be used for the optimization engine and to be transformed into a simulation sequence by interpolation [1]. Another way to represent simulation sequences may be an approximation approach using trigonometrical polynomials of variable degrees. The polynomial parameters are to be optimized, the resulting signal is supposed to be used for the simulation.

References

[1] A. B. et. al. Structural and functional sequence test of dynamic and state-based software with evolutionary algorithms. In *GECCO'03*, pages 2428–2441, 2003.

[2] J. W. et. al. Evolutionary test environment for automatic structural testing. *Information and Software Technology*, 43(1):841–854, 2001.

[3] Y. Zhan. *A Search-Based Framework for Automatic Test-Set Generation for MATLAB/Simulink Models*. PhD thesis, University of York, December 2005.