A Tool Prototype for Model-Based Testing of Cyber-Physical Systems: Manual

1 Introduction

This prototype tool is a Model-Based Testing tool for hybrid systems implemented in Matlab. This tool makes use of hybrid-system models described in a domain-specific language called Acumen Modeling Language [3]. Note that we use "model" and "implementation" to refer to the hybrid system specification model and implementation respectively. The goal of the tool is to generate online test cases, execute them and perform a conformance analysis in order to verify whether the implementation conforms to the specification model.

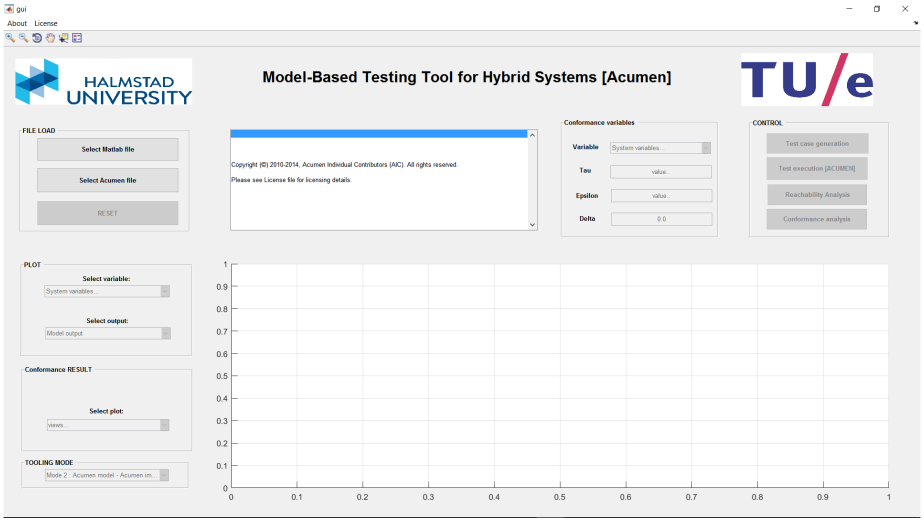
2 Installation Requirements

No installation steps are required to use this software. However, in order to execute the Reachability Analysis procedure, it is necessary to have CORA installed[1]. Our only requirement is that the path to the CORA installation directory should be set in the MATLAB enviroment.

3 Main Routine

The four main steps of the conformance method are test-case generation, test-case execution, reachability analysis and conformance analysis. The application of test-case generation and execution methods results in generating input-output data for both the model and the implementation under test. Application of the conformance analysis, subsequently, results in a conformance judgment possibly accompanied with an additional witness for conformance violation, which is fed into the GUI for visualization purposes. In 1, a preview of the tool GUI is provided. Note that the four buttons in the CONTROL box refers to each one of the steps.

[1] http://www6.in.tum.de/Main/SoftwareCORA


Model-Based Testing Tool for Hybrid Systems [Acumen]

Copyright (©) 2010-2014, Acumen Individual Contributors (AIC). All rights reserved.

Please see License file for licensing details.

Variable: System variables...
Tau: value..
Epsilon: value..
Delta: 0.0

Select variable: System variables...
Select output: Model output
Select plot: views...
TOOLING MODE: Mode 2 : Acumen model - Acumen im...

Figure 1: Tool GUI

3.1 Test Case Generation

In 1, before the test-case generation algorithm is executed, the simulation parameters as specified in the Acumen module are loaded into Matlab. This process is performed by an Acumen file parser which extracts the specified simulation parameters and all model variables from the implementation modeled in Acumen. We focus on a subset of hybrid systems in which, firstly, the guards are not time-dependent and secondly, the invariants are only specified as intervals of input variable valuations.

3.2 Test Case Execution

The test-case execution refers to the application of generated test cases on the implementation modeled in Acumen. In this step, a combination of Java and Matlab code is used in order to execute test cases / inputs on an Acumen (hybrid-system) model. This process involves communication between the Matlab tooling and the Acumen runtime environment.

As soon as the Matlab tooling initiates the socket connection by running its embedded Javaclient, the (initial) simulator state is send over to Matlab. When Matlab returns the simulator state to Acumen, one simulation step of the implementation (in Acumen) is performed. This process repeats itself for every timestep of the full simulation duration as specified in the Acumen model file.

3.3 Reachability Analysis

The reachability Analysis procedure computes the Specification Maximum Change [2], which represents the maximum variation of the specification dynamics within the specified sampling rate. It is recommended to perform this step before the Conformance Analysis in order to guarantee sound verdicts [2]. To execute this step, it is required to have CORA installed and a CORA model of the specification available. The result of this computation is displayed in the ”Delta” box and this value is automatically added to the value of ”Epsilon” when performing the Conformance Analysis.

3.4 Conformance Analysis

The conformance analysis is an implementation of the (τ, ϵ)-conformance relation [1]. In addition to providing a yes/no answer, the tool provides a visualization of the counter-example in case the conformance relation does not hold. This is achieved by plotting both the specification and the implementation trajectories and depicting the case of violation by a (τ, ϵ) box around the specification point which does not find a counterpart in the implementation (or the other wayaround).

References

[1] Houssam Abbas, Bardh Hoxha, Georgios Fainekos, Jyotirmoy V Deshmukh, James Kapinski, and Koichi Ueda. Conformance testing as falsification for cyber-physical systems. *arXiv preprint arXiv:1401.5200*, 2014.

[2] Morteza Mohaqeqi and Mohammadreza Mousavi. Sound test-suites for cyber-physical systems. In *TASE 2016, July 17–19, Shanghai, China*. IEEE Computer Society, 2016.

[3] Walid Taha, Paul Brauner, Robert Cartwright, Verónica Gaspes, Aaron Ames, and Alexandre Chapoutot. A core language for executable models of cyber physical systems. *Rice University*, 2010.