Demo: System Testing with S-TaLiRo: Recent Functionality and Additions

Bardh Hoxha, Adel Dokhanchi, and Georgios Fainekos

Arizona State University,
Tempe, AZ, USA
{bhoxha, adokhanc, fainekos}@asu.edu

Abstract

In this demo, we will demonstrate the latest features of S-TaLiRo, a modular software tool that provides various methods of testing and verification of hybrid systems.
Demo Setup: We will bring a laptop and show how to use S-TaLiRo with different benchmark examples. We will also demonstrate the specification elicitation tool ViSpec.

1 Introduction

One of the main challenges in software development for safety-critical Cyber-Physical Systems (CPS) lies in achieving a certain level of confidence in the system correctness and robustness. In order to perform formal monitoring, testing and verification of CPS, the fully modular tool S-TaLiRo is presented. The tool is designed for seamless integration with the Model Based Design (MBD) process in Matlab/Simulink^{TM}. S-TaLiRo performs robustness guided Metric Temporal Logic (MTL) testing and monitoring. Since writing specifications in MTL is an error prone task that requires expert temporal logic users, a graphical formalism for the development and visualization of specifications is presented. The demo provides an up-to-date overview of S-TaLiRo and its newer features.

2 Features

2.1 Falsification

Temporal logic verification involves the ability to prove as well as to falsify temporal logic properties of systems. S-TaLiRo searches for counterexamples to Metric Temporal Logic (MTL) properties for non-linear hybrid systems through global minimization of a robustness metric.

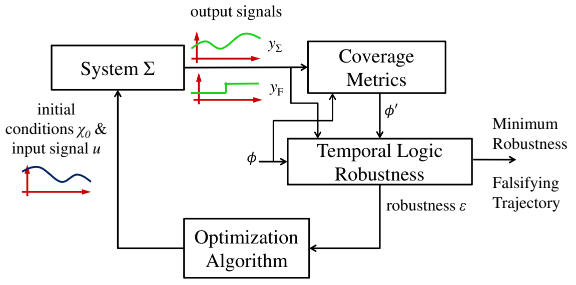

Figure 1: Overview of the basic components of the coverage guided falsification framework.

At its core, it integrates robustness computation for traces of hybrid systems (TaLiRo) with stochastic simulation. The search returns the simulation trace with the smallest robustness value that was found. Traces with positive - but low - robustness values are closer in distance to falsifying traces, using a mathematically well-defined notion of distance between trajectories and temporal logic properties. Such traces provide valuable insight to the developer on why a given property holds, or how to refocus a search for a counter-example.

2.1.1 Coverage Guided Falsification

Coverage guided falsification [3] utilizes coverage metrics on the state space of hybrid systems in order to improve the performance of the falsification methods. As the search process evolves, coverage statistics are collected for the finite space of the output Y_F in addition to the original output space Y_Σ (See Fig. 1). If the falsification process fails, then the search algorithm switches to the coverage guided search, where we modify our specification to bias the search towards the desired hybrid locations. In particular, if the original specification is φ and the least visited (or not visited at all) set of hybrid locations is Q_{Des}, then we define a new MTL formula φ' to incorporate the following requirement "never visit Q_{Des}". The formula φ' now becomes the target for the falsification process. As the falsification algorithm tries to minimize the robustness metric, it indirectly pushes the system to go to the locations provided in φ'. This is due to the fact that the robustness value will be minimized if the hybrid location becomes Q_{Des}.

2.2 Multi-Parameter Mining

In Model Based Development (MBD) of embedded systems, it is often desirable to not only verify/falsify certain formal system specifications, but also to automatically explore the properties that the system satisfies. Namely, given a parametric specification with one or more state or timing parameters, we would like to automatically infer the ranges of parameters for which the property does not hold on the system. We consider parametric specifications in Metric Temporal Logic (MTL). Using robust semantics for MTL, the parameter mining problem can be converted into an optimization problem which can be solved by utilizing stochastic optimization methods.

We will demonstrate how S-TaLiRo can be used to mine the set of falsifying parameters for various benchmarks and specifications. The method is explained in detail in [4].

2.3 Elicitation of Formal Requirements

Developing MTL specifications requires a level of mathematical training that many users may not have. Furthermore, the training required takes a certain amount of time and effort. This, coupled with the fact that writing formal specifications is an error prone task has decreased the willingness of the industry to utilize formal specifications. Therefore, making MTL accessible for widespread use is an important problem. In this demo, we will present the tool VisPec which enables the elicitation of formal requirements.

2.3.1 Debugging MTL specifications

In addition to the development of specifications, we can also debug issues in specifications [2]. In more detail, given an MITL formula, we can detect the following logical issues: 1) Validity: the specification is unsatisfiable or a tautology. 2) Redundancy: the formula has redundant conjuncts. 3) Vacuity: some subformulas do not contribute to the satisfiability of the formula.

2.4 MTL Monitoring

On-line monitoring enables users to observe the CPS behavior in real-time and report potential violations to a supervisor for further control actions. For on-line monitoring of MTL robustness, we can also provide a metric that shows by how far the specification is satisfied or falsified during simulation/execution.

S-TaLiRo provides an on-line monitoring tool as a Simulink block that can run as an integrated module in the simulation process [1]. The user provides the required specification as a bounded future and/or unbounded past MTL formula. The monitor block checks the Simulink generated traces with respect to the required MTL specification. The monitor block then computes the instant robustness estimate at each simulation step. The output of the monitor block can be used on a feedback loop in control applications.

References

[1] Adel Dokhanchi, Bardh Hoxha, and Georgios Fainekos. On-line monitoring for temporal logic robustness. In *Runtime Verification*, pages 231–246. Springer, 2014.

[2] Adel Dokhanchi, Bardh Hoxha, and Georgios Fainekos. Metric interval temporal logic specification elicitation and debugging. In *Formal Methods and Models for Codesign (MEMOCODE), 2015 ACM/IEEE International Conference on*, pages 70–79. IEEE, 2015.

[3] Adel Dokhanchi, Aditya Zutshi, Rahul T Sriniva, Sriram Sankaranarayanan, and Georgios Fainekos. Requirements driven falsification with coverage metrics. In *Proceedings of the 12th International Conference on Embedded Software*, pages 31–40. IEEE Press, 2015.

[4] Bardh Hoxha, Adel Dokhanchi, and Georgios Fainekos. Querying parametric temporal logic properties in model based design. *arXiv preprint arXiv:1512.07956*, 2015.