Verification and Analysis of Simulink/Stateflow Models

Daniel Grier, University of South Carolina - Columbia

ADVISOR: Sayan Mitra

Background

Most embedded computing systems are designed with simulation tools such as Mathwork’s Simulink and Stateflow.

While these tools provide powerful simulation engines for the constructed models, they do not have well-defined semantics, and verifying such models is a challenge.

A hybrid automaton (HA) is a popular mathematical formalism for modeling interaction of physical processes with computations (e.g., an embedded system). It combines discrete state transitions with differential equations.

Considerable progress has been made in verifying safety properties of HA. HyTech is such a tool that computes the reach set of rectangular initialized HA.

Goals

Bring HA verification technologies closer to Simulink/Stateflow (SLSF) models: Develop a tool that will translate SLSF models to rectangular, initialized HA and then use HyTech for verification (compute the reachable set)

Perform a case study of the Small Aircraft Transportation System (SATS) protocol using the tool to verify the safety property: no two planes will enter a zone in which they are considered dangerously close

Fundamental Questions/Challenges

• How does one model nondeterminism in SLSF, which allows only deterministic models?

• How does one model deterministic transitions whose guards are expressed as conjunctions in HyTech, which is unable to handle disjunctive invariants?

• Under which conditions is the translation of the specified system identical to the one defined in HyTech?

Research Results

Found counterexample in simplified SATS model. Traceback generated identified the bug in the Stateflow model representing the simplified protocol.

Future Work/Related Work

• Generating the switching set for a system that is controlled by a safety controller and an optimized controller. This involves computing the backward reachability of the unsafe states with the safety controller. The more exactly one can compute this region, the longer the optimized controller can be used.

• Counterexample-driven debugging for design of embedded controllers. This involves parsing the counterexample that HyTech generates for unsafe systems.

• Approximating linear and nonlinear systems as rectangular, initialized hybrid automata so that the reachable set can be identified with HyTech.

Thanks to Karthikeyan Manamcheri for useful discussions on this research.