Sending Programs “Back to the Future” with DeLorean

Branden Kroske, Broward College

FACULTY ADVISOR: Professor Matthew Caesar

GRADUATE ADVISOR: Jason Croft

Background

Home automation systems are becoming more widespread for several reasons. The two primary ones are that it is becoming cheaper for users to construct them (the network-enabled devices, such as locks and thermostats, are becoming less expensive) and the average user is more likely to have the knowledge to create and administrate an HA system. However, there can be problems, because it is extremely difficult to validate HA systems either by hand or programmatically. There are several reasons, but the most prominent are the interconnected rules and the necessary dependence on time to validate the systems.

It is for those reasons that we proposed DeLorean, a validation system that takes advantage of timed automata and symbolic execution to validate HA systems. However, the current version of DeLorean can only validate programs that it is hard-coded to be able to validate, and it is difficult to install.

Because it is necessary for users to be able to use the system easily, I was in charge of constructing a web application for the system.

Goals

- Validate programs written in C#, ISY, and Elk.
- Allow users to set the time for the program to run and its starting conditions.
- Output the possible final states for a house.
- Take a list of invariants and make sure that the program cannot violate them.
- Make the program easy to use.

Fundamental Challenges

- Absolute and relative timing is vital to HA programs so it can’t be abstracted away.
- Run time must be reasonable; this entails splitting of a program into control loops that are tested individually.
- Once all of the control loops have been tested, they need to be recombined to find the configurations of the entire house.
- When recombining the configurations, it isn’t a cross product of the sets of configurations from the control loops, because some actions affect all devices.
- Inputs can occur at any time, so time cannot be restricted to discrete values.
- The current version of DeLorean is difficult to install and run, and it can only validate ten different programs.
- How should the possible configurations be displayed?
  - There can be many devices with differing possible states
  - It can be tough to understand the configuration of a house with just a list of devices and states.
  - There can be over one thousand unique ending configurations for a program; a list of them would be extremely long.

Research Plan

- Research C# and the Razor syntax of ASP.NET, the languages needed to build a web app.
- Research timed automata: how they work and are represented.
- Research the work already done on DeLorean to understand how it works and how to modify it.
- Research HA systems: how they are written and their format.
- Build a web application to allow users to test DeLorean (delorean.cs.illinois.edu).
- Find a way to display the possible configurations clearly.
- Find a way to recombine the configurations derived from the separate control loops.

Research Results

Improvements are still being made to DeLorean, but current testing is extremely positive.

Early test results:

- Execution of the programs can be accelerated from 3.6 to 36,000 times.
- The results were more accurate than the ones provided by randomized or untimed testing.
- We discovered that two of the programs had states that appeared to be unintended.
- Model checking: Current systems cannot handle checking models that depend on time.
- Timed automata-based model checking: DeLorean doesn’t require the full TA, unlike many systems.
- Symbolic execution: Testing a range of equivalent values at once instead of testing every value individually.

Related Work/Interaction with Other Projects

- Model checking: Current systems cannot handle checking models that depend on time.
- Timed automata-based model checking: DeLorean doesn’t require the full TA, unlike many systems.
- Symbolic execution: Testing a range of equivalent values at once instead of testing every value individually.