An Animation Module For TLA+

William Schultz

May 13, 2018

Abstract

TLA+ is an excellent tool for communicating how a system or algorithm works in a precise manner. Understanding a new TLA+ specification thoroughly, however, can be time consuming and require a certain amount of careful, focused study, depending on the complexity of the system being modeled. Visualizing the behaviors of a system can provide intuition about how a system works, and help provide insights on behaviors that may not be obvious or clear by examining a textual specification alone. The existing TLA+ tools are lacking in their capabilities for interactive visualization. To address this shortcoming, the *Animation* module is a new TLA+ module which allows for the creation of interactive visualizations of TLC execution traces that can be run inside a web browser. Many of the ideas behind the *Animation* module were inspired by the ProB animator tool and the Runway specification language, both of which provide built-in visualization tools. The ProB animator is to a certain extent compatible with TLA+ but specs must be translated first. The Runway visualization tool uses a different specification language and the project does not appear to be actively maintained. The goal of the *Animation* module is to provide a visualization tool that is simple to use and comfortable for existing TLA+ users. To create an animation, one can start with an existing spec and define a view expression, which is a TLA+ expression that produces a set of graphical elements based on the values of the variables declared in a specification. The *Animation* module, when used in conjunction with TLC, constructs a sequence of frames, where each frame is the value of the view expression at a step in an execution trace. The module uses SVG elements as its graphical primitives, which allows for flexibility and variety in the visualizations that can be produced. Additionally, the fact that view expressions are written directly in TLA+ means that they can take advantage of the expressiveness of the language. It also eliminates the need to work in multiple languages when creating a visualization; for example, writing a spec in TLA+ and then defining visualization elements in Javascript.