Incremental Test Case Generation for UML-RT Models Using Symbolic Execution

**Motivation and Goals**

The Iterative Nature of Model-Driven Development (MDD)

Understanding and Classifying the Effects of Model Evolution on Execution and Testing

Improving Efficiency of Test Case Generation Tools by Reducing Redundancy

**Background**

Real-Time Software Modeling (UML-RT)

Symbolic Execution of UML-RT State Machines

**Process**

1. **Differencing Symbolic Execution Trees (SETs)**
   - Original Symbolic Execution Tree
   - Modified Symbolic Execution Tree
   - Determine Paths (in blue) to Tree Differences (in yellow)

2. **Initial Test Case Generation & Examination**
   - Path Coverage Based Test Case Generation
   - Keep Test Cases That Are Not Affected by Differences (in green) – Remove Others

3. **Incremental Generation of New Test Cases**
   - Generate New Paths From Highest Difference (in purple)
   - Add Prefix (blue from Step 1) to New Paths to Generate Full Tests & Add to Final Test Suite

**Evolution Steps**

Three Main Areas of Focus: Additions, Modifications and Deletions

Looking at States, Transitions, Parameters, Attributes, Action Code, and Hierarchy

Having examples from each area for each artifact ensures a coverage of all logical evolution steps

**Planned Work**

Using Empirical Results, Create a Set of Classifications of the effects of Model Evolution on Test Cases


Use the Classifications of the Effects to Minimize the Need for Symbolic Execution

**Resources**