High-Level Hybrid Systems Analysis with Hypy

Stanley Bak, Sergiy Bogomolov, Christian Schilling
Air Force Research Lab (AFRL), IST Austria, University of Freiburg
ARCH 2016
• AFRL-Hosted Workshop: 2016 Safe and Secure Systems and Software Symposium (S5)

• Dayton, Ohio (birthplace of aviation)

• No registration costs. The week before CAV: July 12-14, 2016
Roadmap

• Two-slide Flowpipe Construction Review

• Hyst: Model Transformation

• Hypy: High-Level Analysis
Continuous Post

- A **currently-tracked set of states** is propagated through different instants in time

\[
\begin{align*}
\dot{x} &= y \\
\dot{y} &= (1 - x^2) \times y - x
\end{align*}
\]

*Initial Set: $x=1, y=[-0.5, 0.5]$*
Discrete Post

- A different set representation might be necessary to compute guard / invariant intersections
High-Level Analysis

- High-level analysis of hybrid systems uses multiple iterations, or combines different methods to analyze a system.

- If reachability fails (or error is too large), other strategies can be applied:
  - Change tool parameters
  - Try a different tool
  - Simplify or modify the model
  - Break up the computation into parts
Input Format Problem

SpaceEx

Flow*

```plaintext
hybrid reachability
{
    # Vars
    state var x, t

    setting
    {
        fixed steps 0.001
        time 25
        remainder estimation 1e-4
        QR precondition
        ...
    }

    modes
    {
        off
        {
            nonpoly ode
            {
                x' = -0.1 * x
                t' = 1
            }
            inv
            {
                x >= 18
                0 - (t) <= 0
                t <= 50
            }
        }
    }
```
Input Format Solutions

- Interchange Formats
  - Compositional Interchange Format (CIF)
  - Hybrid Systems Interchange Format (HSIF)
  - Relies on each tool developer to add support
  - Common semantics issues?

- Conversion
  - Hyst
  - Conversion can be done external to each tool
  - Semantics issues?
Hyst

- Hyst: A Hybrid Source Transformation and Translation Tool
Rapid Model Translation

On/off heater in a room
hybrid automaton
A Pseudo-Invariant Transformation Pass splits a single mode of a hybrid automaton into two based on a user-provided hyperplane.
Effect On Reachability

Original

Area: 2.00

Pseudo–Invariant

Area: 0.64

After Transition:
x=[-5,-3.87], y=1

Pseudo Invariant:
y < 1
Pseudo-Invariant Params

- The previous pseudo-invariant transformation pass needs a user-provided hyperplane (defined using a point + normal direction)

- A second pseudo-invariant transformation pass was written which runs a simulation of the system in order to create PI-hyperplanes
  - User provides only the times at which to create the hyperplane parameters
Other Passes

• Other Transformation Passes available in Hyst:
  - Add Identity Resets
  - Scale Time
  - Substitute Constants
  - Simplify Expressions
  - Remove Unsat Invariants
  - Shorten Mode Names
  - Flatten Network Automaton
  - Convert Look-up-Table Dynamics (submission in progress)
  - Model Order Reduction (submission in progress)
  - Continuization (RTSS 2015)
  - Mixed-Triggered Hybridization (HSCC 2016)

- HSCC Wednesday Afternoon (right after lunch), Sergiy will present

SpaceEx Runtime: 10 seconds
Outline

• Background: Hybrid Automata Verification

• Hyst: Model Translation and Transformation

• Scalable Static Hybridization

• Hypy: Automated High-Level Analysis
Why Hypy?

• Techniques like Pseudo-Invariants, Continuization, or Mixed-Triggered Hybridization can be used to analyze models
  - But is anyone actually using them?

• To be truly usable, they must be fully automated

• To evaluate if they actually make a difference (on a large set of benchmarks), they must be fully automated
Hypy: Python Library

- Hypy attempts to present a generic interface to run a number of tools, in order to enable high-level analysis approaches

- Tool-specific scripts are needed to:
  - Run each tool
  - Produce a plot
  - Create a python output object from the tool's output files and/or stdout

- Hyst's model conversion and transformation can be used from Hypy
Hypy: Overview

- Convert & Transform Model
- Read Output
- Interpret Results
- Compute Reachability

Start → Convert & Transform Model → Compute Reachability → Read Output → Interpret Results → Done!
import hybridpy.hypy as hypy

e = hypy.Engine()
e.set_model("toy.xml")
e.set_tool("spaceex")
e.set_output_image("result.png")

if e.run() != hypy.RUN_CODES.SUCCESS:
    print "engine.run() returned error: " + str(code)
    exit(1)

<table>
<thead>
<tr>
<th>Engine Method Name</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>set_tool</td>
<td>selects which tool to run</td>
</tr>
<tr>
<td>set_model</td>
<td>sets the input model path</td>
</tr>
<tr>
<td>set_timeout</td>
<td>sets a timeout while running</td>
</tr>
<tr>
<td>set_save_model_path</td>
<td>sets whether to save converted model output by HyST</td>
</tr>
<tr>
<td>set_output_image</td>
<td>sets the output image path</td>
</tr>
<tr>
<td>set_print_terminal_output</td>
<td>sets whether to print stdout</td>
</tr>
<tr>
<td>set_save_terminal_output</td>
<td>sets whether to save stdout</td>
</tr>
<tr>
<td>set_create_result</td>
<td>sets whether the tool-specific output object should be created</td>
</tr>
<tr>
<td>set_tool_params</td>
<td>sets HyST parameters</td>
</tr>
</tbody>
</table>
Automated Pseudo-Invariants

- Encoded in a ~120 line python script
Automated Result

- Vanderpol system with large initial set, $w = 300$
  
  $x_0 \in 1.25 + w \cdot [0, 0.01], y_0 \in [2.28; 2.32]$

\begin{align*}
(a) \text{ Initial run} & & (b) \text{ Iteration 1} & & (c) \text{ Iteration 2} \\
(d) \text{ Iteration 3} & & (e) \text{ Iteration 4}
\end{align*}

\[w = 75\] case, Fabian Immler, "Verified reachability analysis of continuous systems", Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2015
for splits in [100, 400, 800, 3200, 12800]:

    if done or timeout_reached:
        break

    step = max_time / splits

    for order in ["2", "4", "6"]:

        if done or timeout_reached:
            break

        for rem in ["1e-9", "1e-5", "1e-2"]:  
            print "trying '" + model + '" with step = " + str(step) + ", order = " + order + ", rem = " + rem

            before_last_run = time.time()
            e = hypy.Engine()
            e.set_model(MODEL_DIR + model + ".xml")
            e.set_tool('flowstar')
            e.set_timeout(timeout_sec)
            e.set_create_result(True)
            e.set_tool_params(['-tp', 'plot=gnuplot interval:orders=' + order + ':time=' + str(max_time) + ':step=' + str(step) + ':remainder=' + rem])
Autotune Sample Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Tune (sec)</th>
<th>Run (sec)</th>
<th>Step</th>
<th>Order</th>
<th>Rem</th>
</tr>
</thead>
<tbody>
<tr>
<td>brusselator</td>
<td>3.90</td>
<td>18.47</td>
<td>0.15</td>
<td>6</td>
<td>1e-2</td>
</tr>
<tr>
<td>lorenz_short</td>
<td>2.48</td>
<td>12.26</td>
<td>0.01</td>
<td>4</td>
<td>1e-2</td>
</tr>
<tr>
<td>bio9</td>
<td>15.19</td>
<td>10.14</td>
<td>0.0005</td>
<td>2</td>
<td>1e-2</td>
</tr>
<tr>
<td>3dstable</td>
<td>7.82</td>
<td>110.64</td>
<td>0.025</td>
<td>4</td>
<td>1e-2</td>
</tr>
<tr>
<td>tank</td>
<td>50.90</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>lorenz_long</td>
<td>7787.46</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>bio7</td>
<td>3604.07</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

• Failures:
  - lorenz_long reached 1 hour timeout
  - bio7 appeared to be stuck (bug?)
  - tank hand-tuned model used per-variable remainder estimates
Summary

• Hyst:
  – Model Input Format Conversion
  – Model Transformation

• Hypy:
  – Run Hyst
  – Run Reachability
  – Interpret Tool Output
Long-Term Vision

- **Comparison / Competition**
  - Different tools / tuning approaches / automated application of model transformations

- **Generalizability / Repeatability**
  - Model Transformations coded using Hyst
  - Results plots made using Hypy scripts

- **Thought Transformation**
  - Start thinking in terms of model transformations (tactics) and high-level analysis (strategies)