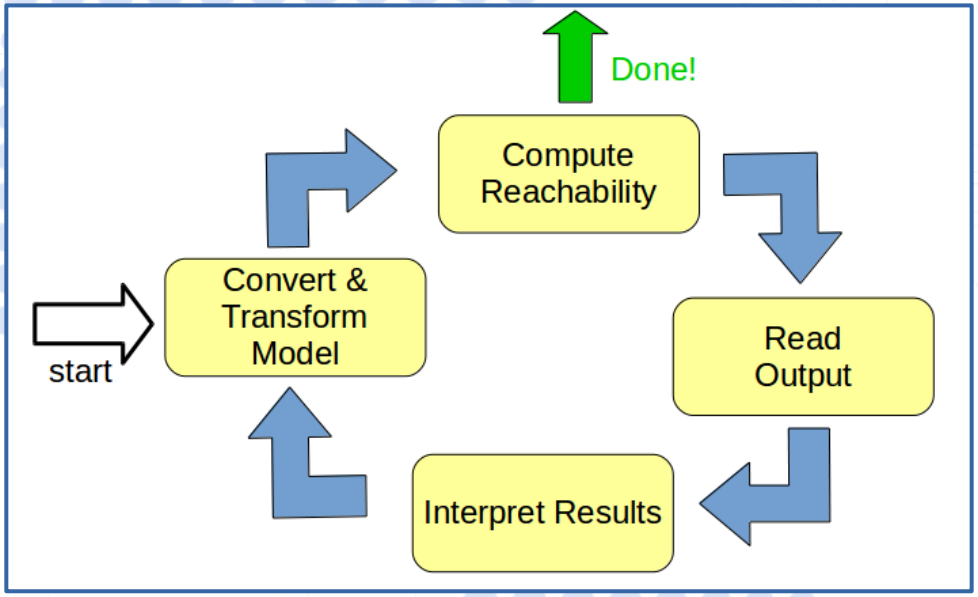
High-Level Hybrid Systems Analysis with Hypy



Stanley Bak, Sergiy Bogomolov, Christian Schilling
Air Force Research Lab (AFRL), IST Austria, University of Freiburg

ARCH 2016

Advertisement (1 of 2)

- 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

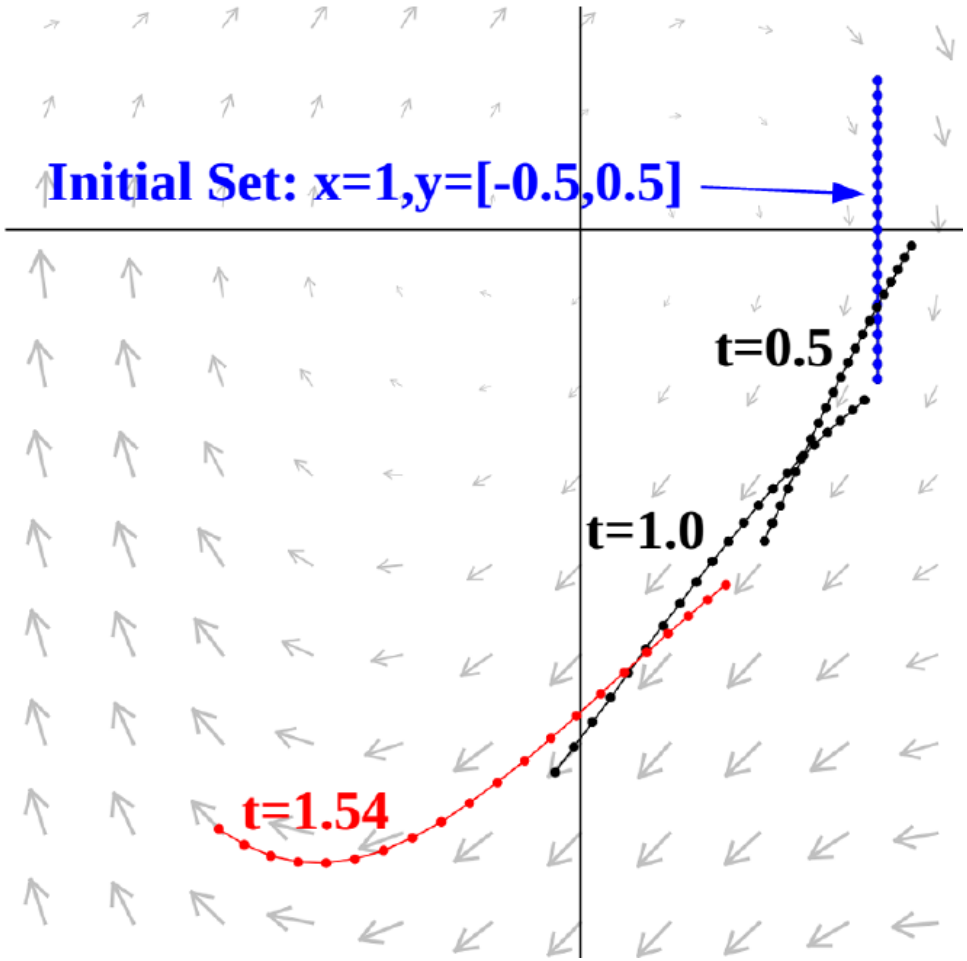
Roadmap

- Two-slide Flowpipe Construction Review
- Hyst: Model Transformation
- Hypy: High-Level Analysis

Continuous Post

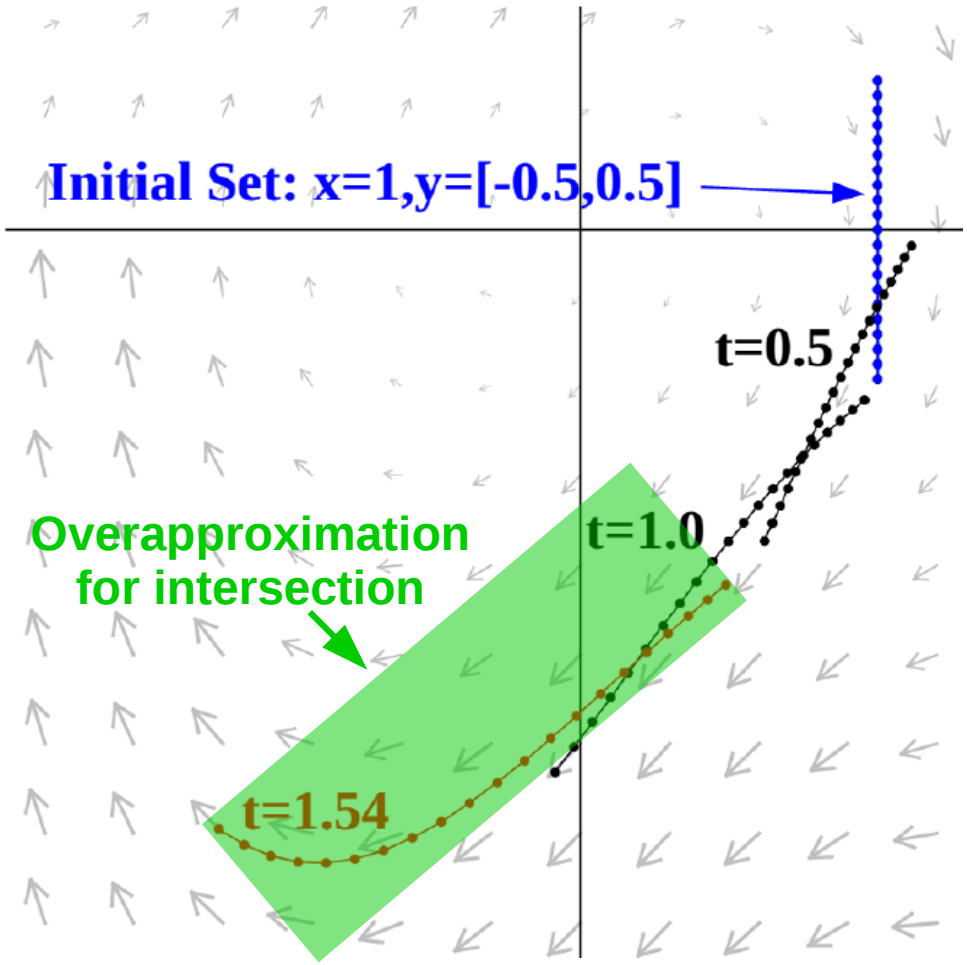
- A **currently-tracked set of states** is propagated through different instants in time

$$\dot{x} = y$$
$$\dot{y} = (1 - x^2) * y - x$$



Discrete Post

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

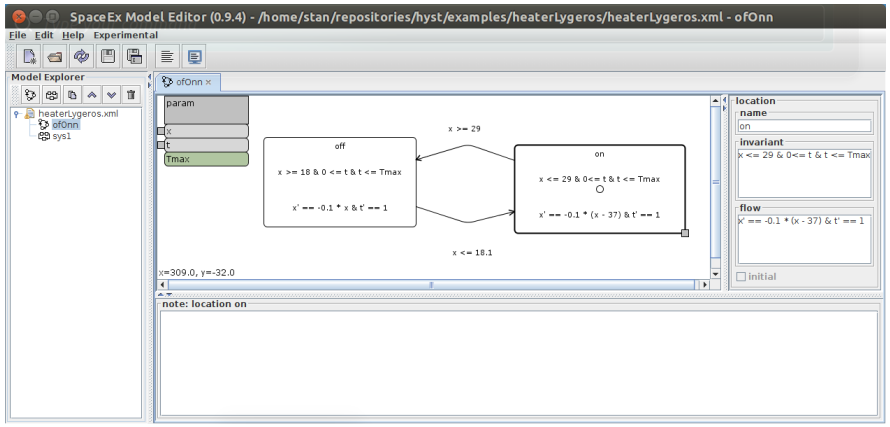


High-Level Analysis

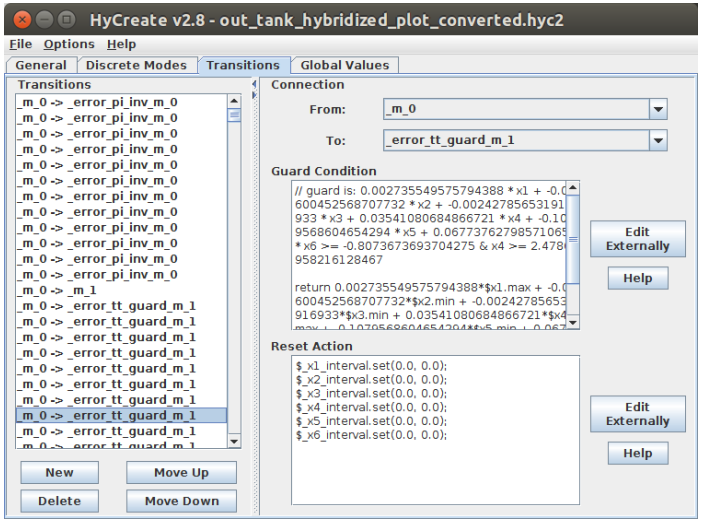
- High-level analysis of hybrid systems uses multiple iterations, or combines different methods to analyze a system

- 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



HyCreate2

```
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
            }
        }
```

Flow*

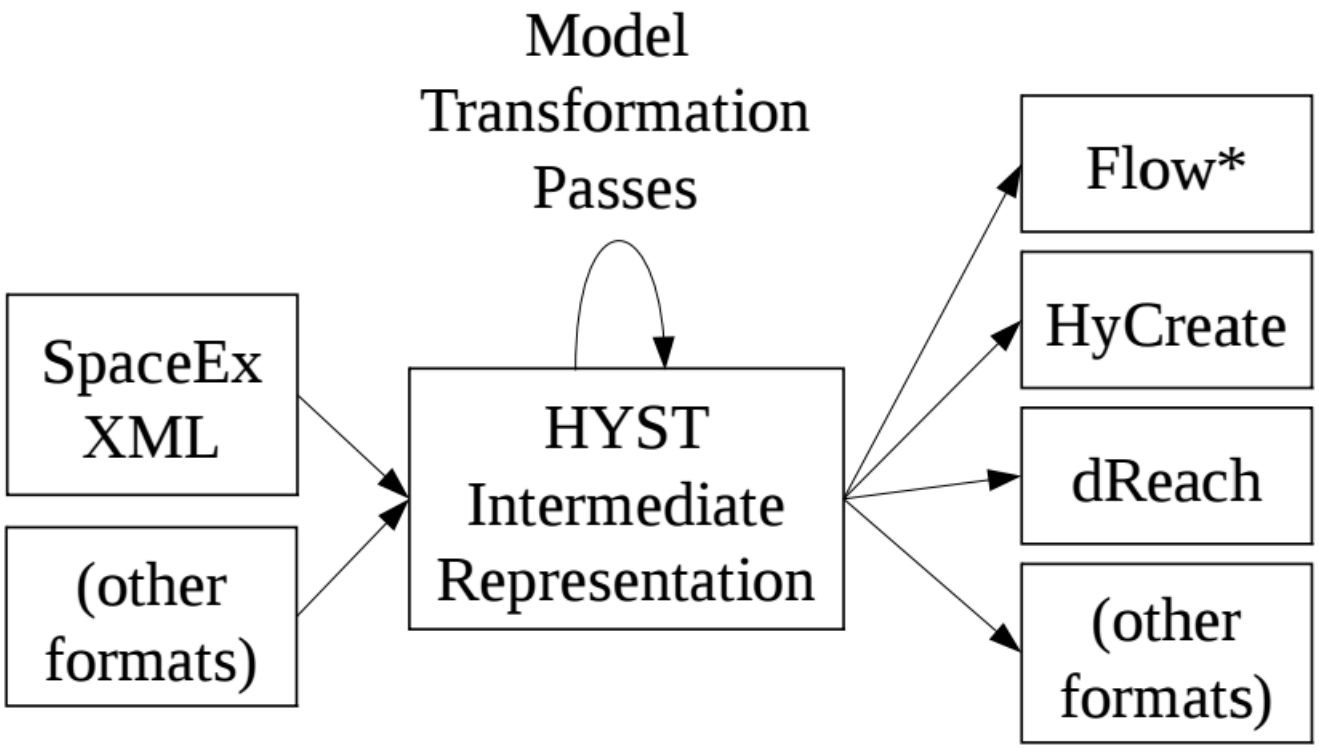
Input Format Solutions

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

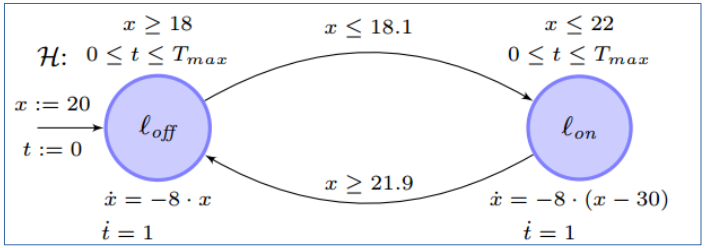
- Conversion
  - **Hyst**
  - Conversion can be done external to each tool
  - Semantics issues?

Hyst

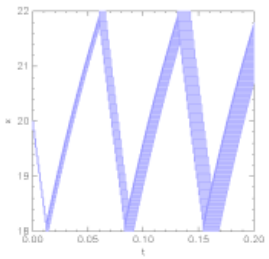
- Hyst: A Hybrid Source Transformation and Translation Tool



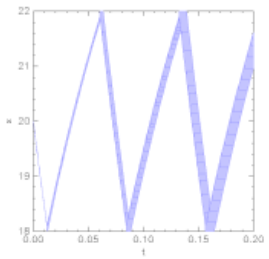
Rapid Model Translation



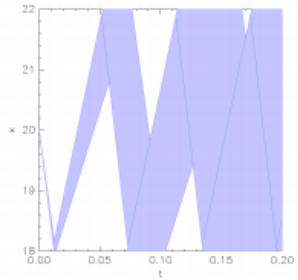
On/off heater in a room
hybrid automaton



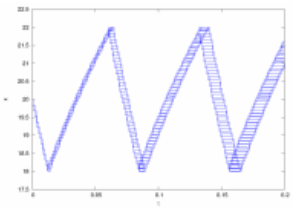
(a) SpaceEx LGG



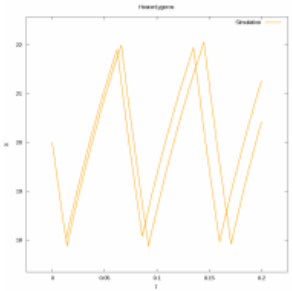
(b) SpaceEx STC



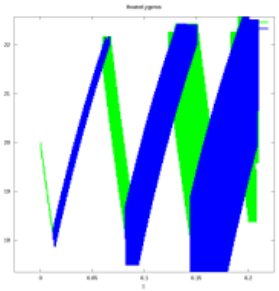
(c) SpaceEx PHAVer



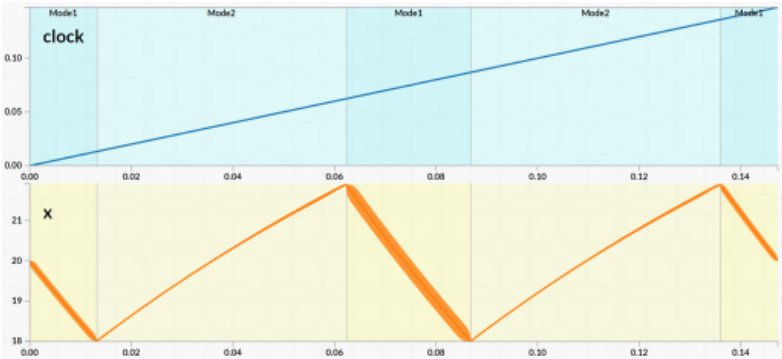
(d) Flow*



(e) HyCreate2 Sim



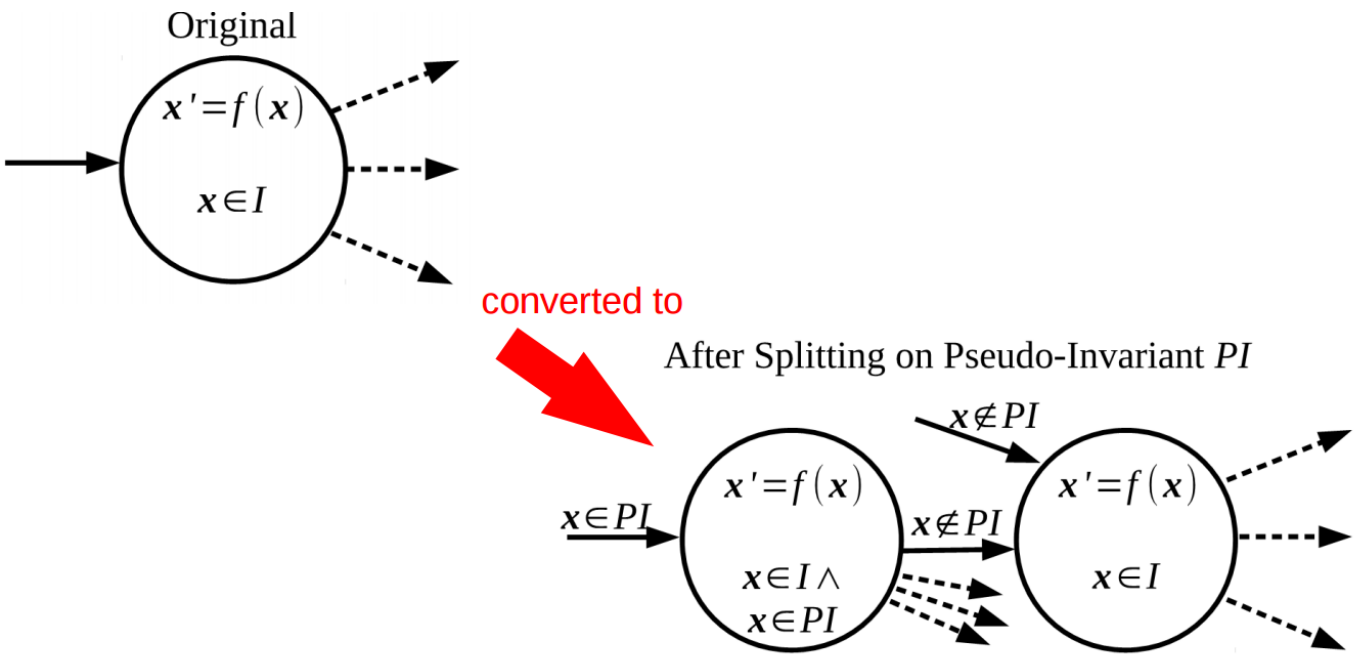
(f) HyCreate2



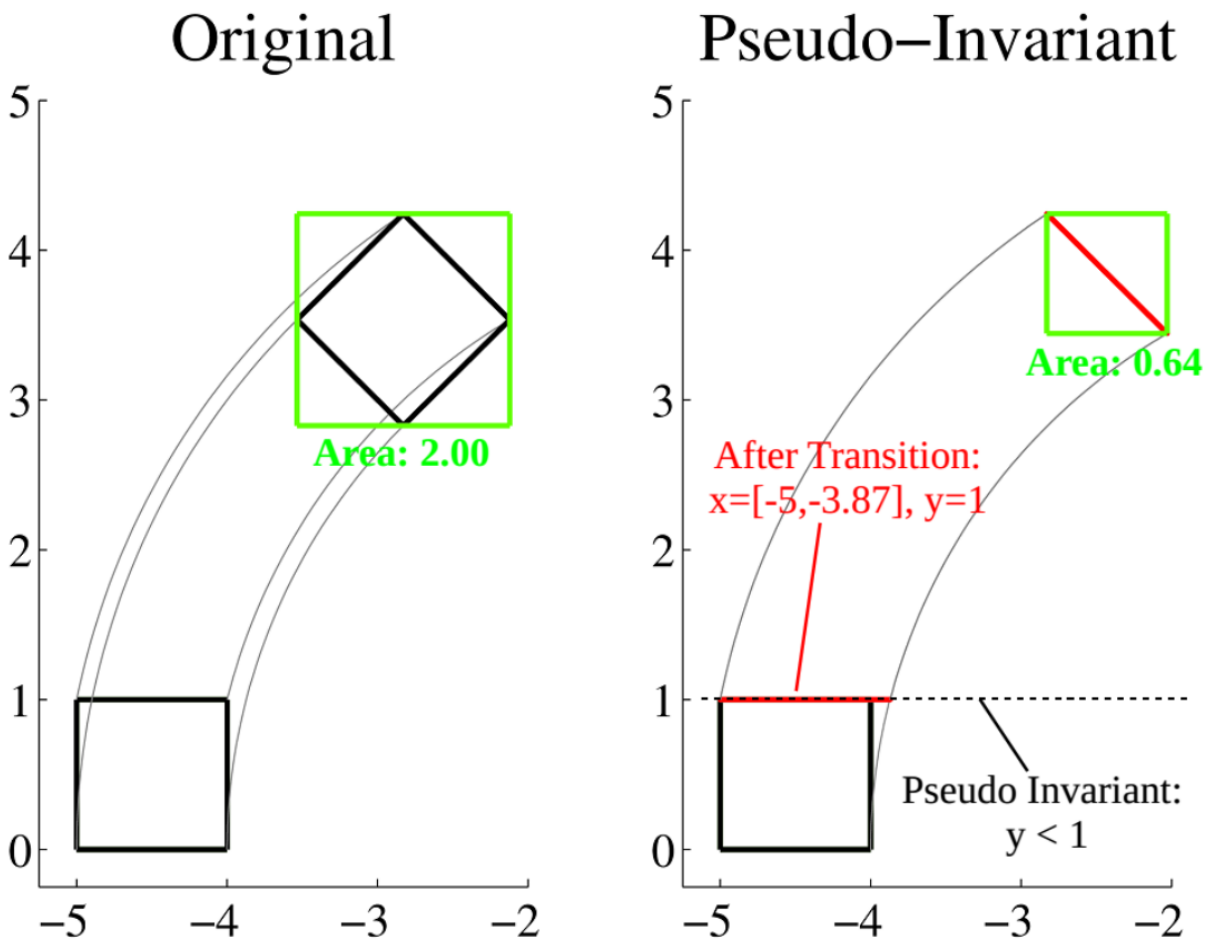
(g) dReach

Example Transformation

- A Pseudo-Invariant Transformation Pass splits a single mode of a hybrid automaton into two based on a user-provided hyperplane



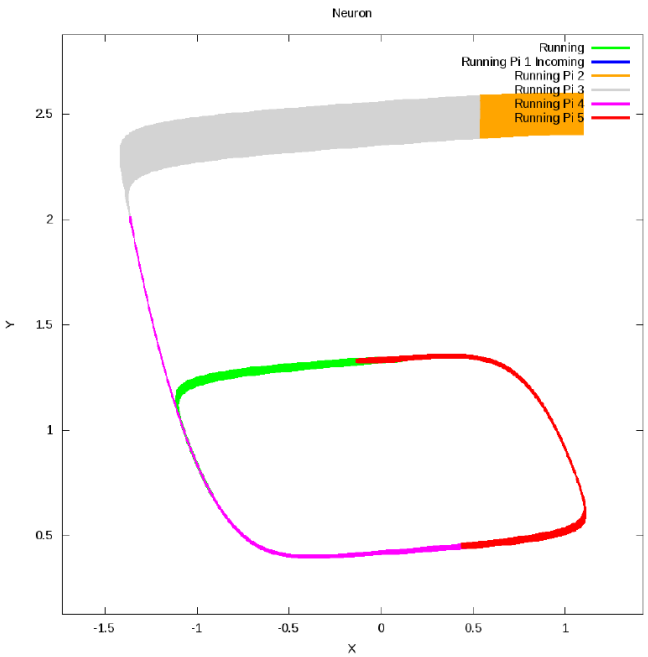
Effect On Reachability



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



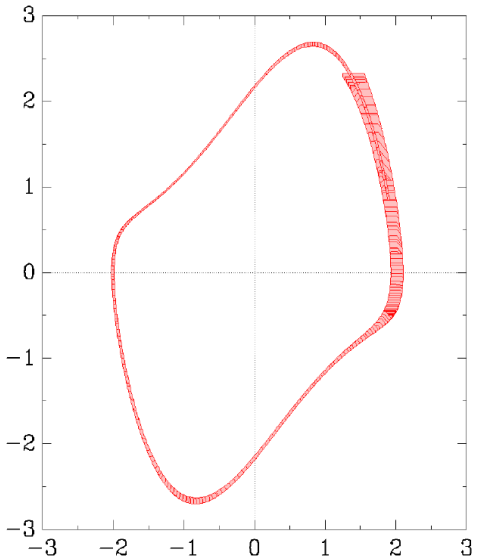
Other Passes

- 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)

Advertisement (2 of 2)

- **"Scalable Static Hybridization Methods for Analysis of Nonlinear Systems"**, S. Bak, S. Bogomolov, T. Henzinger, T. Johnson, P. Prakash, 19th International Conference on Hybrid Systems: Computation and Control (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 <u>must be fully automated</u>

- To evaluate if they actually make a difference (on a large set of benchmarks), they <u>must be fully automated</u>

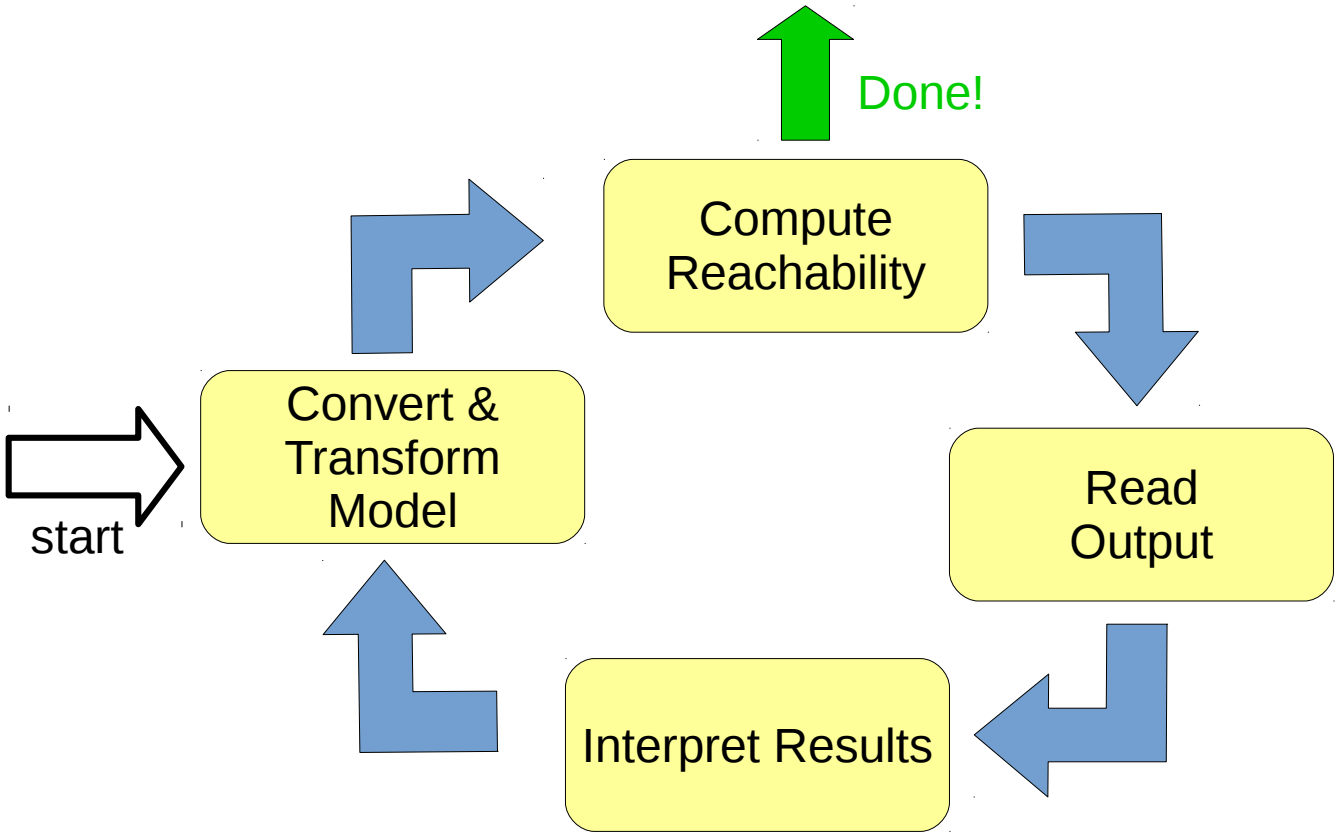
Hypy: Python Library

- 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



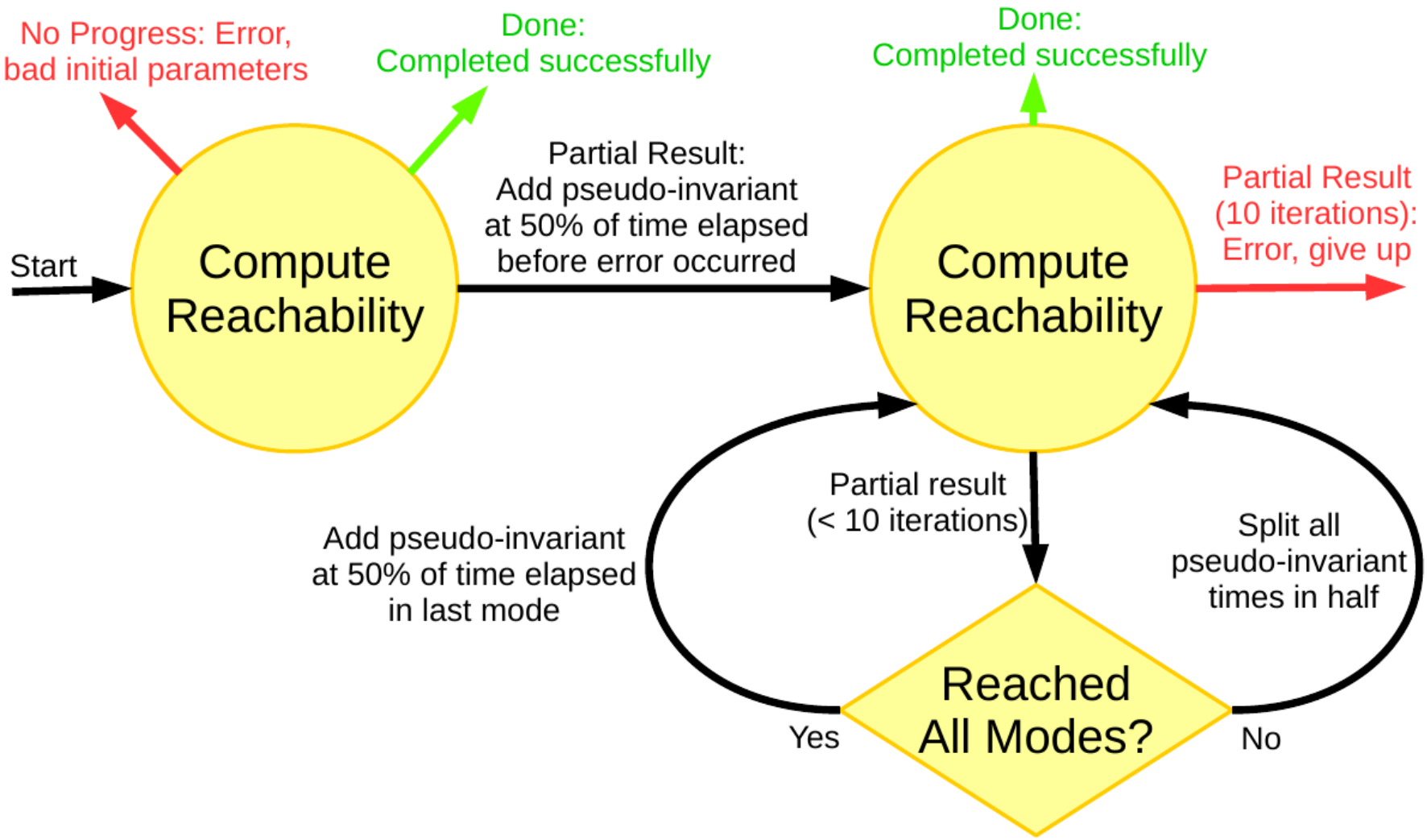
Hypy Example

```
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)
```

Engine Method Name	Purpose
set_tool	selects which tool to run
set_model	sets the input model path
set_timeout	sets a timeout while running
set_save_model_path	sets whether to save converted model output by HyST
set_output_image	sets the output image path
set_print_terminal_output	sets whether to print stdout
set_save_terminal_output	sets whether to save stdout
set_create_result	sets whether the tool-specific output object should be created
set_tool_params	sets HyST parameters

Automated Pseudo-Invariants

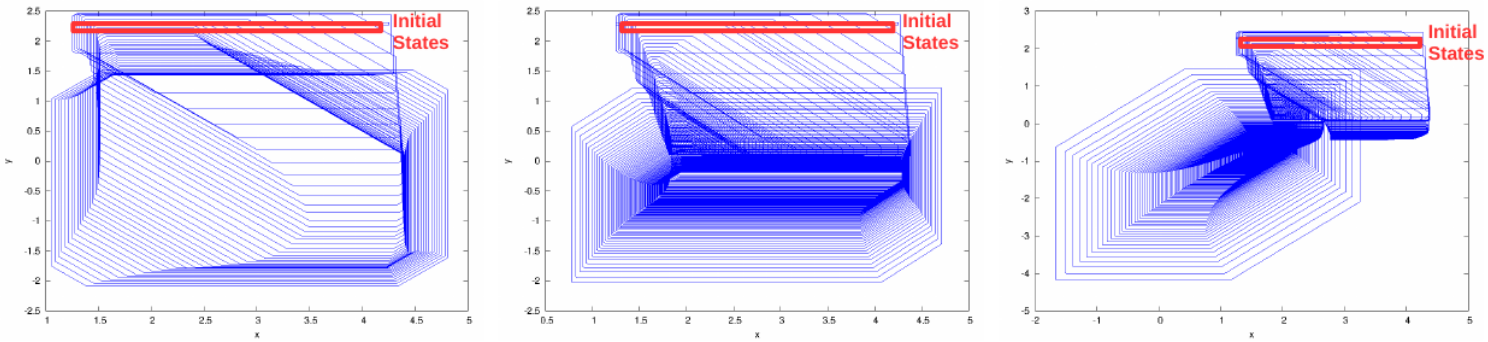
- Encoded in a ~120 line python script



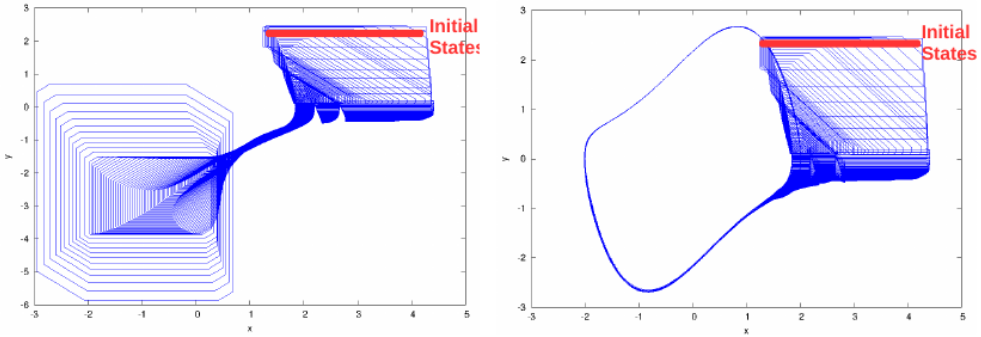
Automated Result

- Vanderpol system with large initial set, w= 300

$$x_0 \in 1.25 + w \cdot [0, 0.01], y_0 \in [2.28; 2.32]$$



(a) Initial run (b) Iteration 1 (c) Iteration 2



(d) Iteration 3 (e) Iteration 4

w = 75 case, Fabian Immler, "Verified reachability analysis of continuous systems", Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2015

Automated Parameter Tuning for Flow*

```
...

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

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

- 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 <u>model transformations</u> (tactics) and <u>high-level analysis</u> (strategies)