HyLAA: A Tool for Computing Simulation-Equivalent Reachability for Linear Systems

Stanley Bak and Parasara Sridhar Duggirala
Overview

- Computing Simulation-Equivalent Reachability using **Linear Stars**
- Invariant Constraint Trimming / Sucessor Deaggregation
- Hylaa Tool Demonstration
Motivation

- Observation: Numerical simulations are *extremely* useful
  - High-dimension scalability
  - Tunable accuracy
  - Fast
  - Trusted in practice

- But simulation is not perfect:
  - Model fidelity issues
  - Simulation accuracy
  - Point-based analysis (not on continuous trajectories)
  - Insufficient coverage of a system's nondeterminism
    (initial states / inputs / switching / disturbances)
Motivation

- **Observation:** Numerical simulations are *extremely* useful
  - High-dimension scalability
  - Tunable accuracy
  - Fast
  - Trusted in practice

- But simulation is not perfect:
  - Model fidelity issues
  - Simulation accuracy
  - Point-based analysis (not on continuous trajectories)
  - **Insufficient coverage of a system's nondeterminism**
    (initial states / switching / disturbances)
We strive to compute **exactly the set of states that any simulation might reach**, which we call simulation-equivalent reachability.

For continuous systems, this is like discrete-time reachability. For hybrid systems, a false invariant forces a transition (no sophisticated zero-crossing).

For every state that is reachable, however, there should be a corresponding simulation which can be produced (counter-example generation).
Generalized Star Sets

- Hylaa uses a state representation which is a version of a generalized star set.

**Definition 5.** A generalized star $\Theta$ is a tuple $\langle c, V, P \rangle$ where $c \in \mathbb{R}^n$ is called the center, $V = \{v_1, v_2, \ldots, v_m\}$ is a set of $m$ ($\leq n$) vectors in $\mathbb{R}^n$ called the basis vectors, and $P : \mathbb{R}^n \rightarrow \{\top, \bot\}$ is a predicate. A generalized star $\Theta$ defines a subset of $\mathbb{R}^n$ as follows.

$$\llbracket \Theta \rrbracket = \{x \mid \exists \bar{\alpha} = [\alpha_1, \ldots, \alpha_m]^T \text{ such that } x = c + \sum_{i=1}^{n} \alpha_i v_i \text{ and } P(\bar{\alpha}) = \top\}$$

Superposition

\[ \Theta \triangleq \langle c, V, P \rangle \]
\[ |\alpha_1| \leq 1 \land |\alpha_2| \leq 1 \]

\[ \text{Reach}_i(\Theta) \triangleq \langle c', V', P \rangle \]
\[ |\alpha_1| \leq 1 \land |\alpha_2| \leq 1 \]
Point Containment

\[ \Theta \triangleq \langle c, V, P \rangle \]
\[ |\alpha_1| \leq 1 \land |\alpha_2| \leq 1 \]

\[ \text{Reach}_i(\Theta) \triangleq \langle c', V', P \rangle \]
\[ |\alpha_1| \leq 1 \land |\alpha_2| \leq 1 \]

\[ [\Theta] = \{ x \mid \exists \bar{\alpha} = [\alpha_1, \ldots, \alpha_m]^T \text{ such that } x = c + \sum_{i=1}^{n} \alpha_i v_i \text{ and } P(\bar{\alpha}) = T \} \]
\[ [\Theta] = \{ x \mid \exists \bar{\alpha} = [\alpha_1, \ldots, \alpha_m]^T \text{ such that } x = c + \sum_{i=1}^{n} \alpha_i v_i \text{ and } P(\bar{\alpha}) = T \} \]
Harmonic Oscillator Example

Dynamics: $x' = y$, $y' = -x$

Initial condition: $x(0) \in [-6, -5]$, $y(0) \in [0, 1]$

At time $\pi/4$ - basis vector #1: $(1, 0) \rightarrow (0.707, -0.707)$

basis vector #2: $(0, 1) \rightarrow (0.707, 0.707)$

Basis Matrix at $\pi/4$

\[
\begin{pmatrix}
-1 & 0 & 0.707 & 0.707 \\
0 & -1 & -0.707 & 0.707 \\
0 & 0 & -1 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & -1 \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\]

LP solver assigns:

$(x, y)$ – state at time $t$

$(\alpha_1, \alpha_2)$ – initial state
Continuous Post Scalability

- In 2 dimensions, we need to do 3 simulations (one for each basis vector, and one for the center)
- In $N$-dimensions, we need $N+1$ simulations
- Two main computations:
  - Run $n+1$ simulations
  - Solve a linear program
- Both seem scalable... how scalable is the method?
Scalability Comparison

- Comparison of Hylaa vs SpaceEx
  - Replicated Helicopter (28 dims each)
The “standard” reachability algorithm:
- Continuous Post until invariant is false
- Trim to invariant
- Discrete Post
- (repeat)
Mode Invariant Error

- The “standard” reachability algorithm:
  - Continuous Post until invariant is false
  - Trim to invariant
  - Discrete Post
  - (repeat)

Trimmed due to Invariant
(Hyla Demo)
invariant_trim.py
Aggregation Error

- Upon taking a discrete transition, successors are aggregated
- Desired operation is set union, but often only convex-hull is possible
Aggregation Error

- Upon taking a discrete transition, successors are aggregated
- Desired operation is set **union**, but often only **convex-hull** is possible
To eliminate this error, we still perform aggregation, but then deaggregate (split) upon reaching a subsequent guard.

**Example:**
- Steps 10 to 20 have a guard enabled, and get aggregated into a single set [10, 20]
- In the successor mode, continuous post for 1.5 seconds before another guard is reached
- Split into two sets, [10, 14] and [15, 20]
- Continue with each of those two sets, skipping the first 1.5 seconds
Deaggregation and Simulation-Equivalence

- With deaggregation, only states with concrete simulations can pass through guards

- Unsafe states are defined as entire modes

- Therefore, unsafe states are reachable only if a concrete simulation exists
  - *Simulation-equivalent safety*
Hylaa is a new tool that computes simulation-equivalent reachability.

The Hylaa tool code, repeatability scripts, an interactive demo, and videos are all available online:

stanleybak.com/hylaa

Our ARCH2017 paper used Hylaa to verify linear systems with over 10000 dimensions*!