Automated Detection of Zeno Sets in Models by an OpenModelica Addon

Marcel Gehrke, Sibylle Schupp, Gerhard Schmitz

Institute for Software Systems
Hamburg University of Technology (TUHH)
Outline

1. Zeno Behaviour
2. Tool for Detecting Zeno Sets
3. State of the Tool
Bouncing Ball Model Falsified by Zenoness

Zenoness = infinite transitions in a bounded and finite length of time

Figure: Bouncing Ball Simulation in OpenModelica

- Zenoness falsifies the result of the simulation.
- The ball drops below the surface.
Water Tank

Figure: Water Tank System

\[ w, \text{ one water source} \]

\[ v_i, \text{ hole in tank } i, \text{ from which water drains} \]

\[ x_i, \text{ current water level in tank } i \]

\[ r_i, \text{ minimal required water level in tank } i \]

\[ w < v_1 + v_2 \text{ is the interesting case} \]
Hybrid Automaton

\[ \begin{align*}
\dot{x}_1 &= w - v_1 \\
\dot{x}_2 &= -v_2 \\
x_2 &\geq r_2 \\
x_1 &> r_1 \\
\end{align*} \]

\[ \begin{align*}
\dot{x}_1 &= -v_1 \\
\dot{x}_2 &= w - v_2 \\
x_1 &\geq r_1 \\
x_2 &> r_2 \\
\end{align*} \]

Figure: A Hybrid Automaton of the Water Tank System

- \( Q \), discrete states: \( \{ q_1, \ldots, q_n \} \)
- \( X \), continuous variables from \( \mathbb{R}^n \) with \( n \geq 1 \)
- \( Init \), initial states from \( Q \)
- \( D : Q \to \mathcal{P}(X) \), domain for each discrete state
- \( E \subset Q \times Q \), transition from one state to another: \( q_i \to q_j \)
- Guards \( G : E \to \mathcal{P}(X) \)
- Reset map \( R : E \times X \to \mathcal{P}(X) \)
- \( D(q)^0 \), interior of the domain
- \( \partial D(q) \), boundary of the domain
Requirements

A cycle is a necessary condition for a hybrid automaton to accept zeno executions [3].

**Theorem** (*Non-expanding* reset map (*|R| ≤ 1*), zeno [3])

- If *G*(q, q′) ∩ *D*(q)⁰ = ∅, ∀(q, q′) ∈ *E* with q, q′ ∈ *Q*∞
- then *x*ᵢ ∈ ∂*D*(qᵢ) for all i = 1, ..., m.

**Corollary** (*Identity* reset map (*R = 1*), zeno free [3])

- *G*(q, q′) ∩ *D*(q)⁰ = ∅, ∀(q, q′) ∈ *E*,
- for all cycles {qᵢ}ᵢ=1^K with qᵦ = q₁ and (qᵢ, qᵢ₊₁) ∈ *E*,
  1 ≤ i ≤ K − 1, ∩ᵢ=1^K−₁∂*D*(qᵢ) = ∅.
Steps of the Tool

Input: The hybrid automaton of the system, provided by the user

- Parse input file based on a predefined grammar
- Detect cycles
  - Robert Tarjan “Enumeration of the Elementary Circuits of a Directed Graph”[2]
- Detect the zeno sets
- Return results
Input File of Water Tank Automaton

Automaton;
State, q1, True, x2 >= r2, x1 = w - v1, x2 = - v2;
State, q2, True, x1 >= r1, x1 = - v1, x2 = w - v2;
Transition, q1, q2, x2 <= r2, 1;
Transition, q2, q1, x1 <= r1, 1;
Water Tank

\[
\begin{align*}
\dot{x}_1 &= w - v_1 \\
\dot{x}_2 &= -v_2 \\
x_2 &\geq r_2 \\
x_1 &> r_1 \land x_2 > r_2
\end{align*}
\]

\[
\begin{align*}
\dot{x}_1 &= -v_1 \\
\dot{x}_2 &= w - v_2 \\
x_1 &\geq r_1 \\
x_2 &\leq r_2 \land x_1 > r_1 \land x_2 > r_2
\end{align*}
\]

Figure: A Hybrid Automaton of the Water Tank System

Output of the Tool

```
*Zeno> main "watertank.txt"
The zeno set for the cycle [q1 q2 q1] is:
In state `q1` for the continuous variable x1 the zeno point(s) are: r1; for the continuous variable x2 the zeno point(s) are: r2;
In state `q2` for the continuous variable x1 the zeno point(s) are: r1; for the continuous variable x2 the zeno point(s) are: r2.
```
Validation of the Tool

Tested all combinations of guards and domains to validate that:

- Neither the order of the states and transitions nor the order in the restrictions matters.
- Only transitions of the cycles are considered.
- Intervals are correctly generated.
- Distinct cycles are correctly detected.
- Tool works correctly for either variables or constants as bounds.
State of the Tool

- Command-line prototype
- Written in Haskell

Next steps:
- Always handle reflexive edges correctly
- Automatic generation of hybrid automaton (use the HyAuLib?)
- Prevent zeno behavior
  - Introduce delay
Conclusion

- Zenoness is a modeling artifact.
- Prototype detects zeno sets off-line and automatically; requires transformed model.
- Zeno sets are important for validating the simulation.
- Validating the simulations of hybrid systems is crucial for validate hybrid systems.
Bibliography

