Abstraction Refinement for Stability

Parasara Sridhar Duggirala
Sayan Mitra

University of Illinois at Urbana Champaign
**Stability**

- System eventually reaches a set of stable states and remains in them forever

- Also called *Practical Stability* or *Region Stability*
Stability

- Practical Application: *Automotive control protocol ensures that destination is reached eventually*

  ![Diagram of stability](image)

  - *Self Stability* – Distributed Systems
  - Related to Control Theory
Stability

- Similar to Halting Problem

- Techniques for proving termination
- Terminator project from Microsoft Research
- Well-Founded Relations: Partial Order Relations with no infinite chains
Goal

To use abstraction refinement techniques from software Verification to verifying stability of Hybrid Systems
Hybrid Systems

- Mix of continuous and discrete dynamics

- Several modes of operation

- System switches modes based on constraints

- Trajectories ($\tau$) and Discrete Transitions

- Execution sequences $- \tau_0 a_1 \tau_1 a_2 \tau_2 ...$

- Thermostat example:

  $$\text{temp} = 20 \rightarrow \text{temp} = 30 \rightarrow a_1 \text{temp} = 30 \rightarrow \text{temp} = 15 \ldots$$
A set of states $S$ is **stable** for $A$ if
- $S$ is closed and
- $S$ is inevitable

Examples: Vehicle **reaches** destination, protocol **recovers** from failures

- $A$ is **nonblocking** if time can diverge along every execution starting from every state
- $A$ is **blocking** if time stops along every execution starting from every state

\[
\begin{align*}
S_0 & : \quad x \leq 5 \quad \dot{x} = 1 \\
S_1 & : \quad \dot{x} = 1 \\
S_2 & : \quad x \geq 5 \quad \dot{x} = 1 \\
\end{align*}
\]
Relating Stability and Blocking

- $A_S$: HA obtained by removing $S$ from $A$

- If $A_S$ is blocking then $S$ is inevitable for $A$

- In addition if $S$ is closed then $S$ is stable for $A$

- Conversely, if $S$ is stable for $A$ then $A_S$ is blocking

- Relate stability verification to blocking property

- Trouble: Dealing with the dense time

- Solution: Hybrid Step Relation
Hybrid Step Relation

- \( H_r \subseteq Q \times Q \) is called Hybrid step relation
- \((q, q') \in H_r \) iff \( \exists q'' : q \xrightarrow{\tau} q'' \land q'' \xrightarrow{a} q' \)

\[
\begin{align*}
0 \leq x & \leq 5 \land 5 \leq y \leq 10 \\
\land x' = 5 \land 5 \leq y' & \leq 10 \\
\land (x' - x) & \leq (y' - y) \leq 2(x' - x)
\end{align*}
\]
Hybrid Step relation and Blocking

- Prove blocking property using hybrid step relation

  Intuition: If the hybrid system is blocking, then there are no infinite chains of hybrid step relations

- Well-founded relations do not have infinite chains

  \[ x' = x + 1 \text{ - not well founded} \]
  \[ x' = x + 1 \land x' < 5 \text{ - well founded} \]

A non-Zeno Hybrid System \( \mathbf{A} \) is blocking iff the Hybrid step relation \( H_r \) is well-founded

- To verify blocking property of \( \mathbf{A} \): Compute \( H_r \) and check whether it is well-founded
Stability (Overview)

- Stability of Hybrid System $A$
  - stable set $S$

- Blocking Property of $A_S$

- Hybrid Step relation $H_r$

Software Verification

- Well Founded Relations for proving termination

- Yes/No
Abstraction Refinement - Need

- Coming up with one well-founded relation for the whole system is impractical
- Similar to proving termination of programs

Ex: 

\[ x R y \leftrightarrow \exists n, x - y = 10n \]
\[ x R' y \leftrightarrow \exists n, x - y = n \]

Advantage: Divide the task of proving that \( H_r \) has no infinite chains by giving more than one well founded relation
Hybrid Step Relation – well foundedness

- For a state transition system \((s,t)\)
  No infinite chains \(s_1 \rightarrow s_2 \rightarrow \ldots\) if
  \(t^+ \subseteq R_1 \cup R_2 \cup \ldots \cup R_n\)
  where \(R_i\) is well founded [Podelski & Rybalchenko 2004]

- Similarly if \(H_r^+ \subseteq R_1 \cup R_2 \cup \ldots \cup R_n\) then \(H_r\) is well founded

- \((q,q') \in H_r^+\) if \(q \rightarrow_{\tau_1} q_1 \rightarrow a_1 q_2 \rightarrow \ldots \rightarrow a_m q'\)

- if \(q.\text{mode} \neq q'.\text{mode}\) then well founded

- Suffices to consider only loops
Abstraction Refinement (sketch)

- For every loop $L$ check whether the corresponding loop transition relation $H_L$ is well founded

- Abstraction: We abstract $H_L$ by a more “general” transition relation
  ex: $x' = x + 10n$ can be abstracted by $x' = x + n$

- Given $\mathcal{P} = \{P_1, \ldots, P_m\}$,

- $\text{abs}_\mathcal{P}(H_L) \supseteq H_L$ is defined as the smallest superset of $H_L$ constructed by taking conjunctions of predicates in $\mathcal{P}$

- Locally blocking, non-Zeno
  $A$ is blocking if there exist predicates $\mathcal{P} = \{P_1, \ldots, P_m\}$ and well-formed relations $\mathcal{R} = \{R_1, \ldots, R_n\}$ such that for every loop $L$, $\text{abs}_\mathcal{P}(H_L) \subseteq R_i$
Abstraction refinement algorithm

\[ \mathcal{P} = \emptyset \quad \mathcal{R} = \emptyset \]

\[ \exists \mathcal{L}, \text{abs}_{\mathcal{P}}(H_{\mathcal{L}}) \not\subseteq \mathcal{R}_i \]

\[ \exists \mathcal{R}_i \in \mathcal{R}, H_{\mathcal{L}} \subseteq \mathcal{R}_i \]

\[ \mathcal{P} = \mathcal{P} \cup F(\mathcal{L}, \mathcal{R}) \]

\[ \exists \mathcal{R} \notin \mathcal{R}, H_{\mathcal{L}} \subseteq \mathcal{R} \]

\[ \mathcal{R} = \mathcal{R} \cup \mathcal{R} \]

O is an infinite execution
Requirements

- Compose hybrid step relations to construct $H_L$
- Check $\exists R \not\in H_L \subseteq R$
  - RankFinder
- Sound and complete for initialized rectangular HA
- Terminates for many rectangular HA in practice
Summary and Future Work

- Well founded relations can be used to prove blocking property of hybrid systems
- Hybrid systems with positive average dwell time
- Complete for Initialized rectangular hybrid automata

Future Work

- Extend the technique for Linear Hybrid Systems
- Use Lyapunov functions effectively