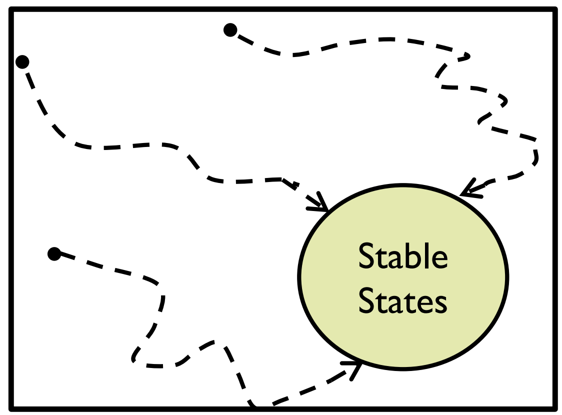
Abstraction Refinement for Stability

Parasara Sridhar Duggirala
Sayan Mitra

University of Illinois at Urbana Champaign

Stability

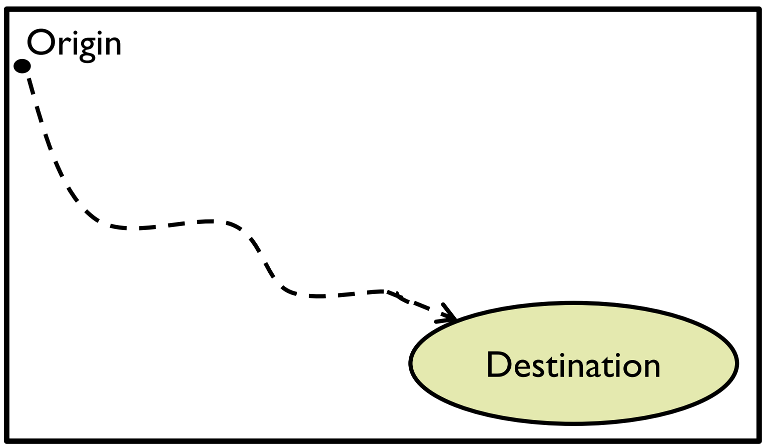
- *System eventually reaches a set of stable states and remains in them forever*

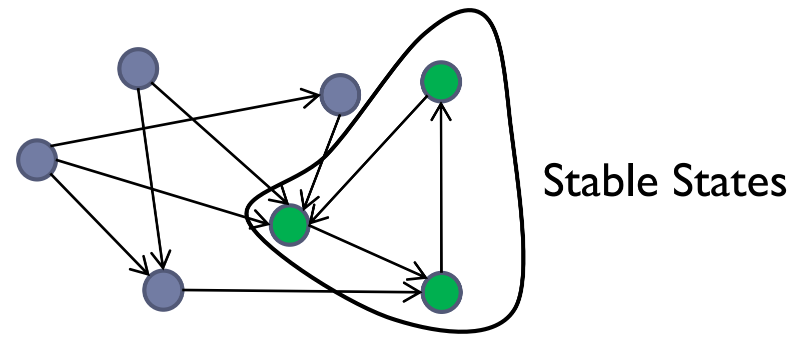


- Also called *Practical Stability* or *Region Stability*

Stability

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

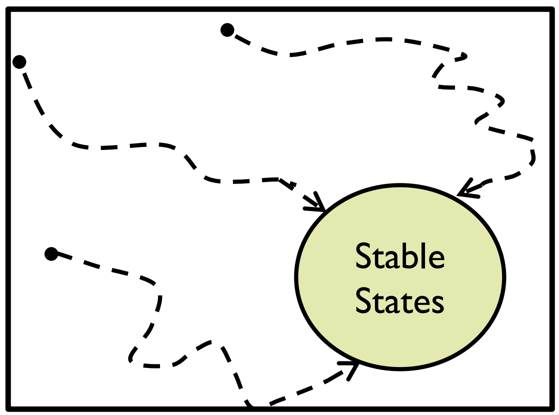


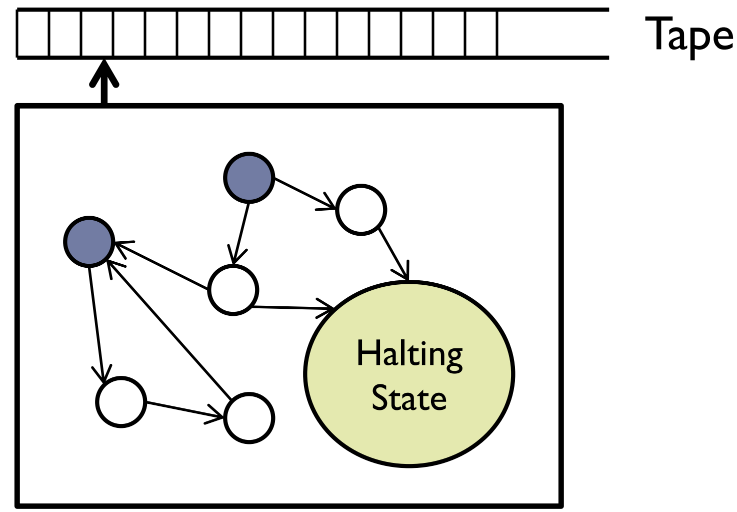


- *Self Stability* – Distributed Systems
- Related to Control Theory

Stability

- *Similar* to Halting Problem





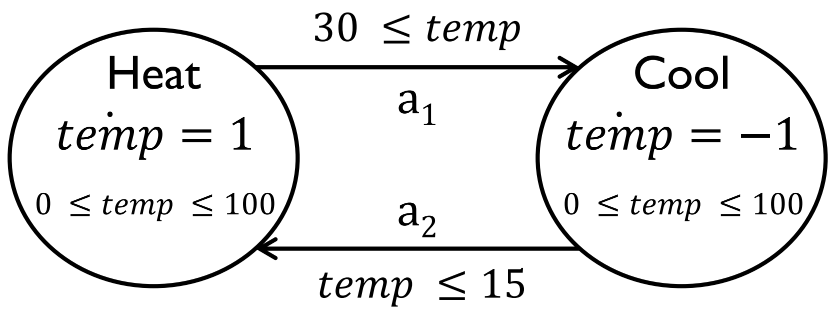
- Techniques for proving termination
- *Terminator* project from Microsoft Research
- *Well-Founded Relations: Partial Order Relations with no infinite chains*

Goal

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

Hybrid Systems

- Mix of continuous and discrete dynamics

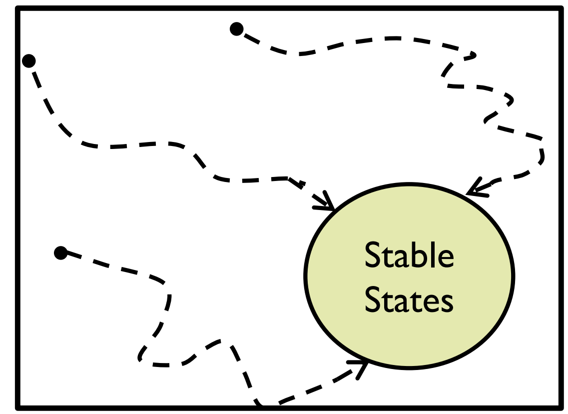


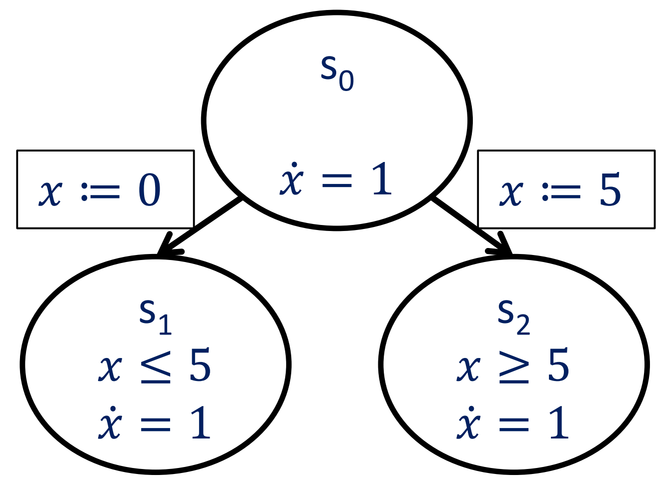
- Several *modes* of operation
- System switches *modes* based on constraints
- *Trajectories* (τ) and *Discrete Transitions*
- Execution sequences – $\tau_0 a_1 \tau_1 a_2 \tau_2$...
- Thermostat example:

$$temp = 20 \rightarrow temp = 30 \rightarrow_{a_1} temp = 30 \rightarrow temp = 15 \ldots$$

(Region) Stability and Blocking

- 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

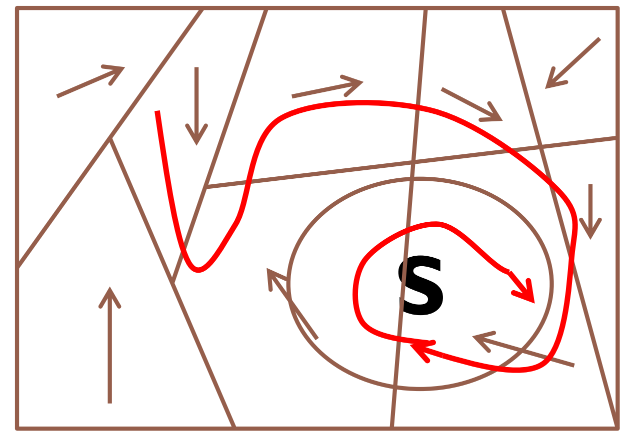




Relating Stability and Blocking

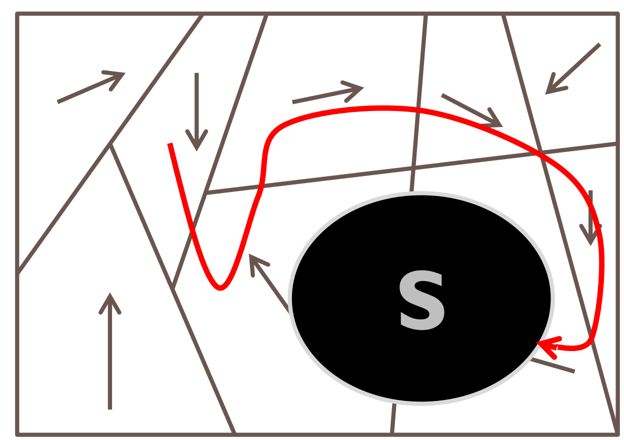
- $A_{\overline{S}}$: HA obtained by removing **S** from A
- If $A_{\overline{S}}$ is blocking then **S** is inevitable for A

  In addition if **S** is closed then **S** is stable for A
- Conversely, if **S** is stable for A then $A_{\overline{S}}$ is blocking



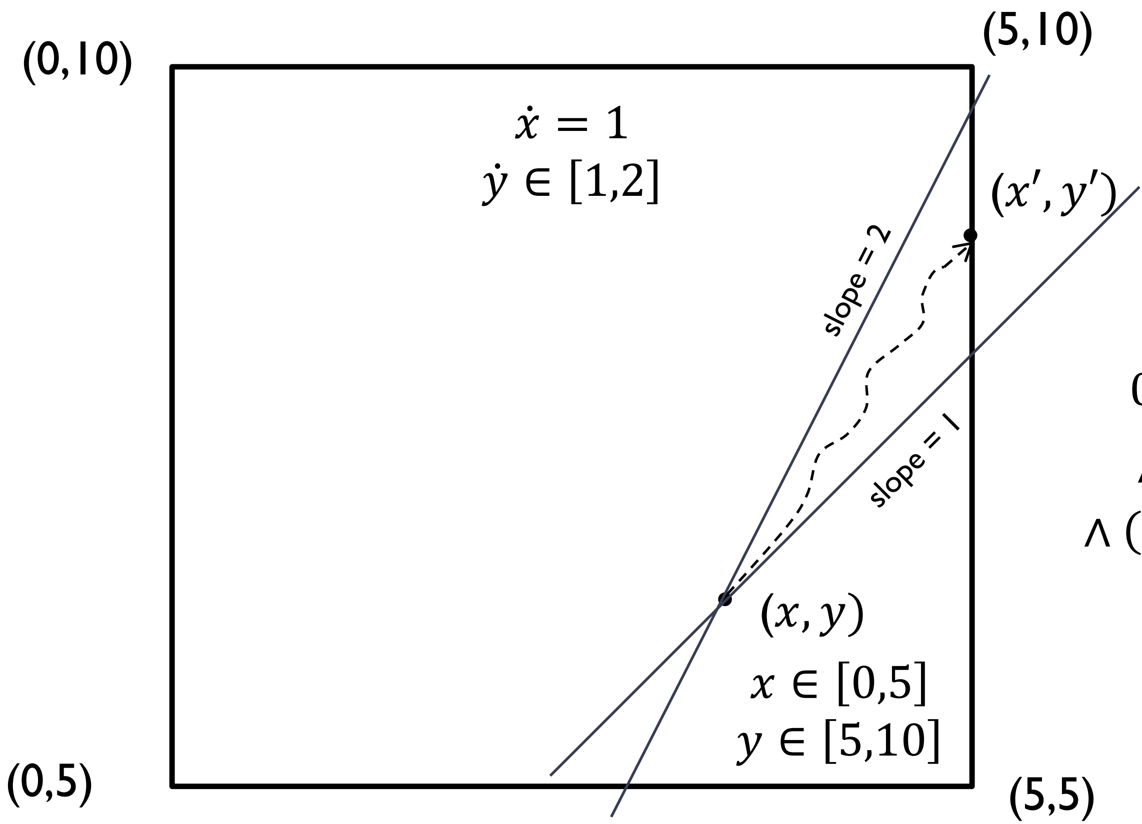
- Relate stability verification to blocking property
- Trouble: Dealing with the dense time

Solution : Hybrid Step Relation



Hybrid Step Relation

- $H_r \subseteq Q \times Q$ is called Hybrid step relation
- $(q,q') \in H_r$ iff $\exists\, q''\ q \rightarrow_{\tau} q'' \wedge q'' \rightarrow_{a} q'$



Hybrid Step relation

$$0 \le x \le 5 \ \wedge\ 5 \le y \le 10$$
$$\wedge\ x' = 5\ \wedge 5\ \le y' \le 10$$
$$\wedge\, (x' - x) \le (y' - y) \le 2(x' - x)$$

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

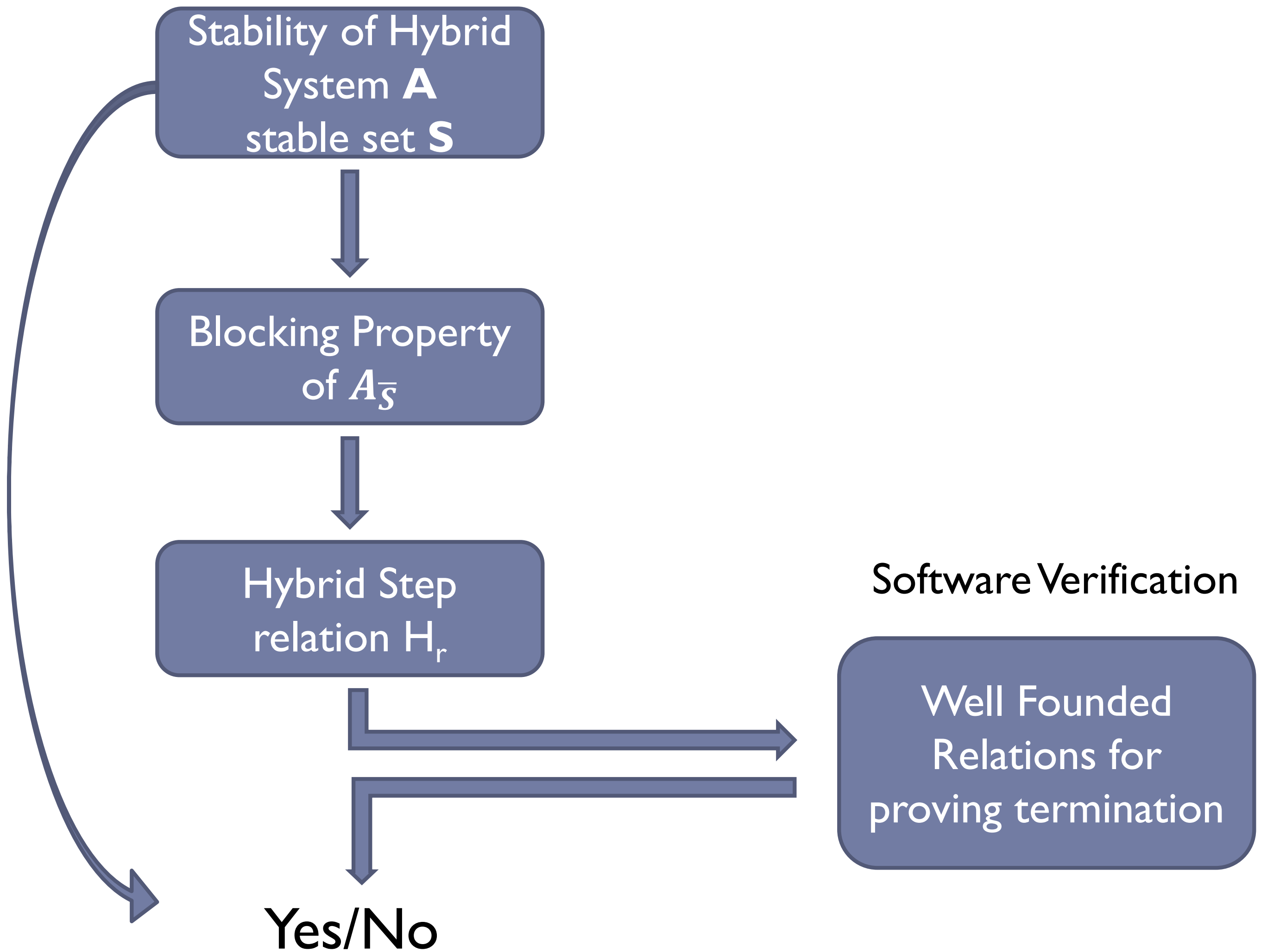
- *Well-founded* relations do not have infinite chains

$x' = x + 1$ – *not well founded*
$x' = x + 1 \wedge x' < 5$ – *well founded*

A non-Zeno Hybrid System **A** is blocking iff the
Hybrid step relation H_r is well-founded

- To verify blocking property of **A** : Compute H_r and check whether it is well-founded

Stability (Overview)



Abstraction Refinement - Need

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

Abstraction: We abstraction a transition relation R with an abstraction transition relation R' if R ⊆ R'

- 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
  $t^+ \subseteq \mathbf{R_1} \cup \mathbf{R_2} \cup \ldots \mathbf{R_n}$
  where $\mathbf{R_i}$ is well founded [Podelski & Rybalchenko 2004]

- Similarly if $H_r^+ \subseteq \mathbf{R_1} \cup \mathbf{R_2} \cup \ldots \mathbf{R_n}$ then H_r is well founded
- $(q,q') \in H_r^+$ if $q \rightarrow_{\tau_1} q_1 \rightarrow_{a_1} q_2 \ldots \rightarrow_{a_m} q'$
- if q.mode $\neq$ q'.mode then well founded
- Suffices to consider only loops

Abstraction Refinement (sketch)

- For every loop **L** check whether the corresponding loop transition relation $\mathrm{H}_{\mathbf{L}}$ is well founded
- *Abstraction:* We abstract $\mathrm{H}_{\mathbf{L}}$ by a more "general" transition relation
ex: x' = x + 10n can be abstracted by x' = x + n

- Given $\boldsymbol{\mathcal{P}} = \{P_1, \dots, P_m\}$,
- $abs_{\mathcal{P}}(\mathrm{H}_{\mathbf{L}}) \supseteq \mathrm{H}_{\mathbf{L}}$ is defined as the smallest superset of $\mathrm{H}_{\mathbf{L}}$ constructed by taking conjunctions of predicates in $\boldsymbol{\mathcal{P}}$

- Locally blocking, non-Zeno

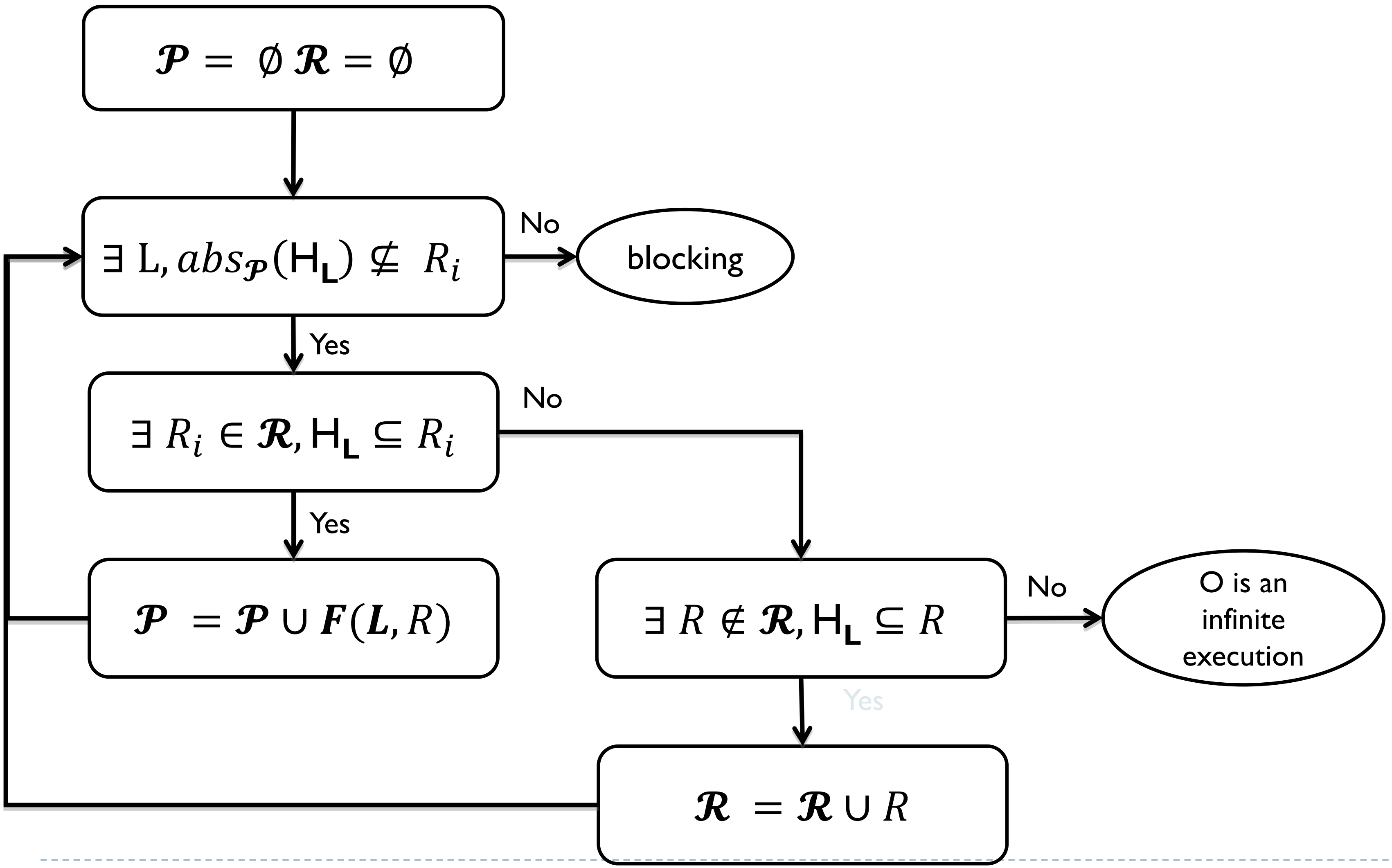
  $\boldsymbol{A}$ is blocking if there exist

  predicates $\boldsymbol{\mathcal{P}} = \{P_1, \dots, P_m\}$ and

  well-formed relations $\boldsymbol{\mathcal{R}} = \{R_1, \dots, R_n\}$ such that

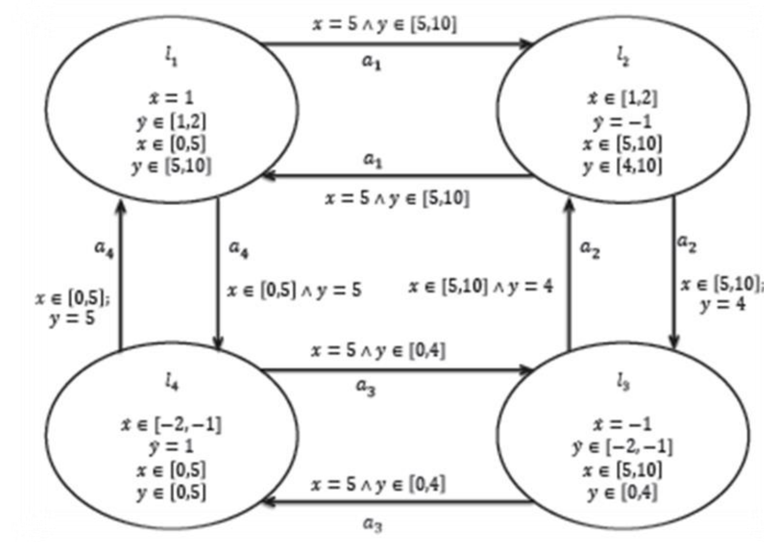
  for every loop **L,** $abs_{\boldsymbol{P}}(\mathrm{H}_{\mathbf{L}}) \subseteq R_i$

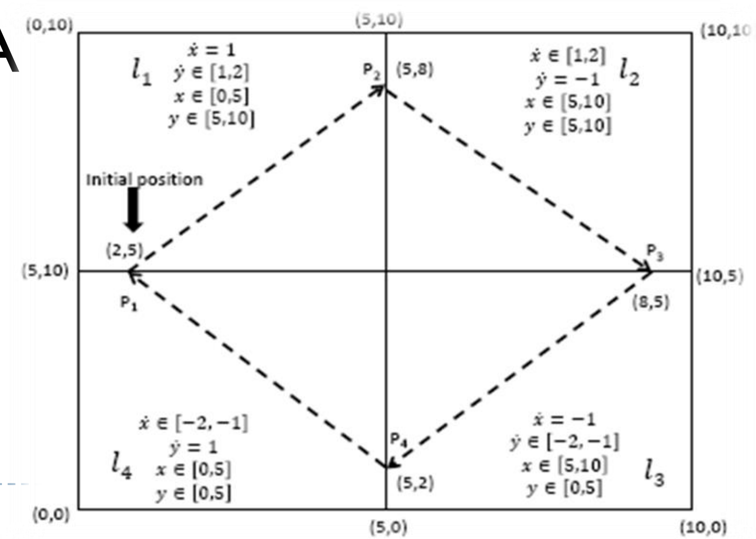
Abstraction refinement algorithm



Requirements

- Compose hybrid step relations to construct H_L
- Check $\exists\ R \notin, H_L \subseteq R$
  - 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