Reachability analysis

Sayan Mitra

Verifying cyberphysical systems

mitras@illinois.edu

Next few lectures

Focus on specific classes of hybrid automata for which safety properties (invariants) can be verified completely automatically

- Finite state machines
- Alur-Dill's Timed Automata[1] (Today)
- Rectangular initializaed hybrid automata
- Linear hybrid automata
- …

We will introduce ***abstractions:*** Simplifying or approximating one automaton A with another automaton B

[1] Rajeev Alur et al. The Algorithmic Analysis ofHybrid Systems. Theoretical Computer Science, volume 138, pages 3-34, 1995.

Today

- Finite state machines
- Algorithmic analysis of (Alur-Dill's) Timed Automata[1]
    - A restricted class of what we call hybrid automata in this course with only clock variables

[1] Rajeev Alur and David L. Dill. A theory of timed automata. Theoretical Computer Science, 126:183-235, 1994.

Reachability of Finite Automata

An **finite automaton** is a tuple $\mathcal{A} = \langle Q, \mathrm{Q}_0, \mathcal{D} \rangle$ where

- Q is a finite set of states
- $Q_0 \subseteq Q$ is the set of **initial** or **start states**
- $\mathcal{D} \subseteq Q \times Q$ is the set of **transitions**

An *execution* of $\mathcal{A}$ is an alternating sequence of states and actions $\alpha = q_0 q_1 q_2 \ \ldots q_k$such that:

1. $q_0 \in \mathrm{Q}_0$ 2. $\forall\, i$ in the sequence, $(q_i, q_{i+1}) \ \in \mathcal{D}$

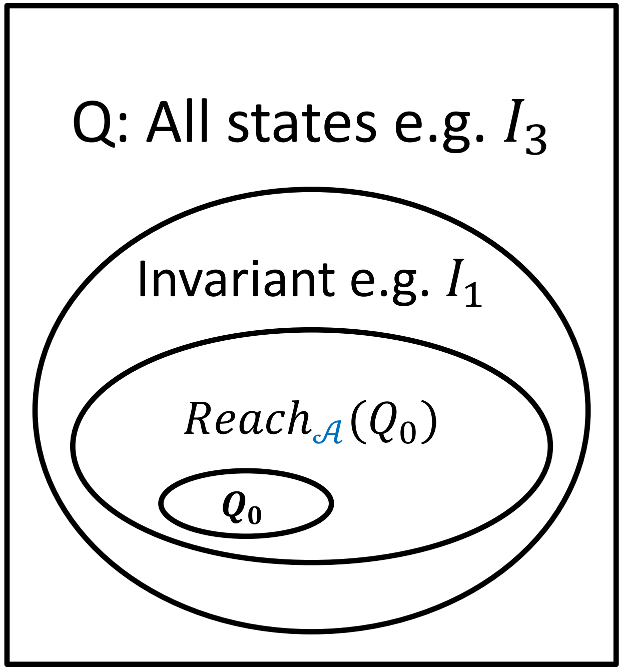
A state $\boldsymbol{u}$ is *reachable* if there exists an execution α such that $\alpha.lstate = q_k = \boldsymbol{u}$

Reachability in finite state machines

$Reach_{\mathcal{A}}(\Theta)$: set of states reachable from Θ by automaton $\mathcal{A}$

An *invariant* is a set of states I such that $Reach_{\mathcal{A}} \subseteq I$

How to check whether $\boldsymbol{u}$ is *reachable* ?



Reachability as graph search

Q1. Given $\mathcal{A}$, is a state $\boldsymbol{u} \in Q$ reachable?
Define a graph $G_{\mathcal{A}} = \langle V, E \rangle$ where
$V = Q$
$E = \{(q, q') | q \rightarrow q'\}$

Q2. Does there exist a path in $G_{\mathcal{A}}$ from any state in Θ to u ?

Perform Depth First or Breadth First Search on $G_{\mathcal{A}}$ from Q_0

Time complexity of BFS $O(| Q | + |D|)$
Space complexity is $O(| Q |)$

Nondeterministic reachability

```
Input: G = (V, E), $Q_0, U \subseteq V$
$n := |V|$
vcurrent := choose $Q_0$
If vcurrent ∈ $T$ return ‘‘yes”
Else For i = 1 to $n$:
   vnext := choose V
   If (vcurrent, vnext) ∉E break
   If vnext ∈ T return ‘‘yes”
   vcurrent := vnext
Return ‘‘no”
```

Requires only O(log |Q|) bits of memory

Using Savitch’s construction we get a deterministic algorithm that uses O($\log^2$|Q|) bits

Adding Clocks and Clock Constraints

- A clock variable x is a continuous (analog) variable of type real such that along any trajectory τ of x, for all t ∈ $\tau.dom, (\tau \downarrow x)(t) = t$.

- For a set X of clock variables, the set Φ(X) of integral clock constraints are expressions defined by the syntax:

  g ::= x ≤ $q \mid x \geq q \mid \neg\, g \ \mid g_1 \wedge \ \ g_2$

  where $x \in X \ and \ q \in \ \mathbb{Z}$

- Examples: x = 10; x ∈ [2, 5); true are valid clock constraints
- What do clock constraints look like?

- Semantics of clock constraints $[g]$

Integral Timed Automata

Definition. A integral timed automaton is a HIOA $\mathcal{A} = \langle V, \Theta, A, \mathcal{D}, \mathcal{T} \rangle$ where

- V = X ∪ $\{l\}$, X is a set of n clocks and l is a discrete state variable of finite type L; **stata space** $val(X) \times L$
- A is a finite set
- $\mathcal{D}$ is a set of transitions such that
  - The guards are described by clock constrains $\Phi(X)$
  - $\langle x, l \rangle - a \rightarrow \langle x', l' \rangle$ implies either $x' = x$ or $x = 0$
- $\mathcal{T}$ set of clock trajectories for the clock variables in X

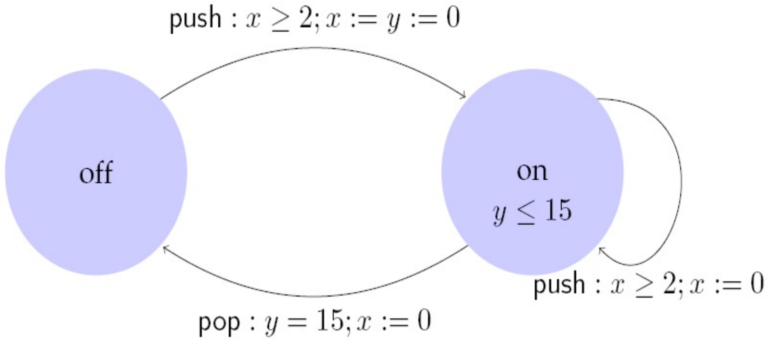
Example: Light switch

Math Formulation

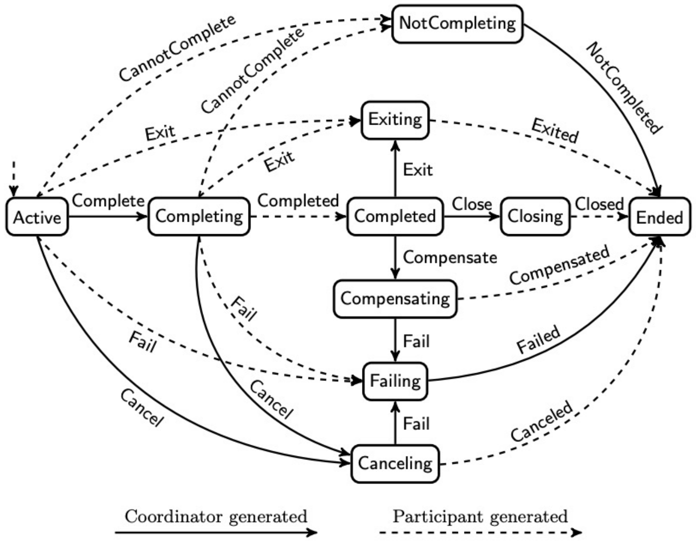
```
automaton Switch
    variables
    internal x, y:Real := 0, loc: {on,off} := off

    transitions
    internal push
        pre $x \geq 2$
        eff if loc = on then x := 0
           else x,y := 0; loc := off
    internal pop
        pre $y = 15 \wedge loc = off$
        eff x := 0

    trajectories
        invariant loc = off => $y \leq 15$
        evolve d(x) = 1; d(y) = 1
```

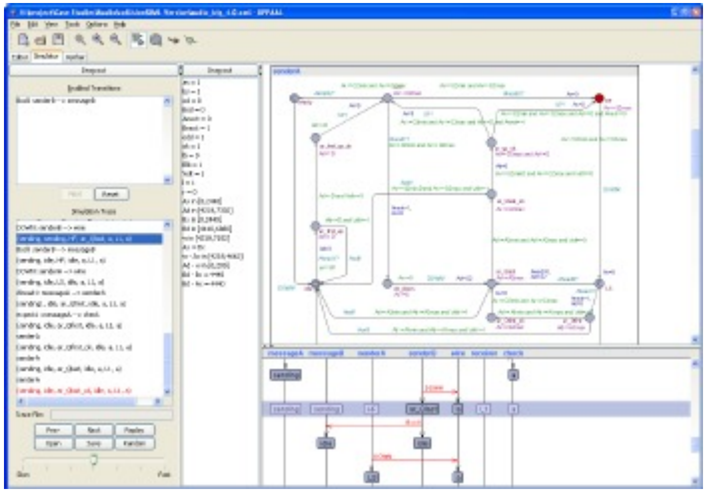
Description

Switch can be turned on whenever at least 2 time units have elapsed since the last turn on. Switches off automatically 15 time units after the last on.



Timed Automaton application in Web Services (WS)





WS-Coordination describes a framework for coordinating transactional web services

Network protocol described in state tables

600+ lines of C-like code in the protocol model

Modeled and Verified using the UPPAAL tool

Analysis considers different channel models

The main safety property: protocol **does not enter invalid state**

Property violated in all but the FIFO channel model

Modelling and Verification of Web Services Business Activity Protocol Anders P. Ravn, Jiri Srba, and Saleem Vighio, RV 2010

Control State (mode) Reachability Problem

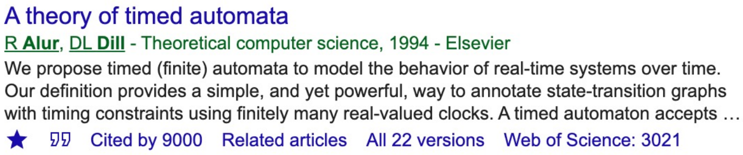
- Given an ITA $\mathcal{A}$, check if a particular (mode) control state $l^* \in L$ is **reachable** from the initial states
- Why is mode reachability good enough even if we are interested in checking reachability of $X^* \subseteq val(X)$?

Model Reachability of Integral Timed Automata is Decidable [Alur Dill 94]

That is, there is an algorithm that takes in $\mathcal{A}, l^*$ and terminates with the correct answer.

Key idea:

- Construct a finite automaton B that is a ***time-abstract bisimilar*** to the given ITA $\mathcal{A}$
- That is, FA B behaves identically to ITA $\mathcal{A}$ w.r.t. control state reachability, but does not preserve timing information
- Check reachability of FA B



A theory of timed automata

R **Alur**, DL **Dill** - Theoretical computer science, 1994 - Elsevier

We propose timed (finite) automata to model the behavior of real-time systems over time. Our definition provides a simple, and yet powerful, way to annotate state-transition graphs with timing constraints using finitely many real-valued clocks. A timed automaton accepts …

★ ” Cited by 9000 Related articles All 22 versions Web of Science: 3021

An equivalence relation with a finite quotient

Under what conditions do two states x_1 and x_2 of the automaton $\mathcal{A}$ behave identically with respect to control state reachability (CSR)?

- When do they satisfy the same set of clock constraints?
- When would they continue to satisfy the same set of clock constraints?

An equivalence relation with a finite quotient

Under what conditions do two states x_1 and x_2 of the automaton $\mathcal{A}$ behave identically with respect to mode reachability ?

When do they satisfy the same set of clock constraints?

When would they continue to satisfy the same set of clock constraints?

$x_1.loc = x_2.loc$ and

x_1 and x_2 satisfy the same set of clock constraints

For each clock y int($x_1.y$) = int($x_2.y$) or int($x_1.y$) $\geq c_{\mathcal{A}y}$ and int($x_2.y$) $\geq c_{\mathcal{A}y}$. ($c_{\mathcal{A}y}$ **is the maxium clock guard** of y)

For each clock y with $x_1.y \leq c_{\mathcal{A}y}$, frac($x_1.y$) $= 0$ iff frac($x_2.y$) $= 0$

For any two clocks y and z with $x_1.y \leq c_{\mathcal{A}y}$ and $x_1.z \leq c_{\mathcal{A}z}$, frac($x_1.y$) $\leq$ frac($x_1.z$) iff frac($x_2.y$) $\leq$ frac($x_2.z$)

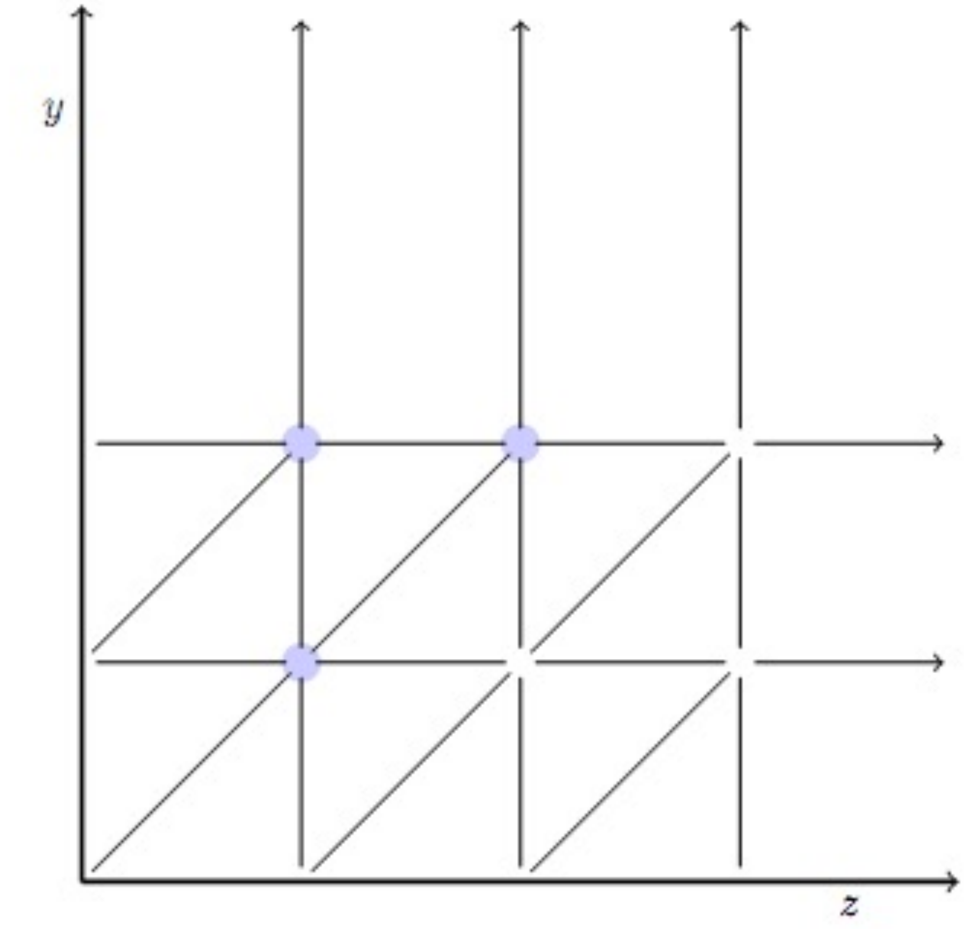
Lemma. This is an **equivalence relation** on *val(V)* the states of $\mathcal{A}$

The partition of *val(V)* induced by this relation is are called **clock regions**

What do the clock regions look like?

Example of
Two Clocks

X = {y,z}
$c_{\mathcal{A}y}$ = 2
$c_{\mathcal{A}z}$ = 3



Complexity

Lemma. The number of clock regions is bounded by |L| |X| ! $2^{|X|}\prod_{z\in X}(2c_{\mathcal{A}z}+2)$.

Region automaton $\mathrm{R}(\mathcal{A})$

Given an ITA $\mathcal{A} = \langle V, \Theta, \mathcal{D}, \mathcal{T} \rangle$, we construct the corresponding **Region Automaton** $\mathrm{R}(\mathcal{A}) = \langle Q_R, \Theta_R, D_R \rangle$.

(i) $\mathrm{R}(\mathcal{A})$ visits the same set of modes (but does not have timing information) and

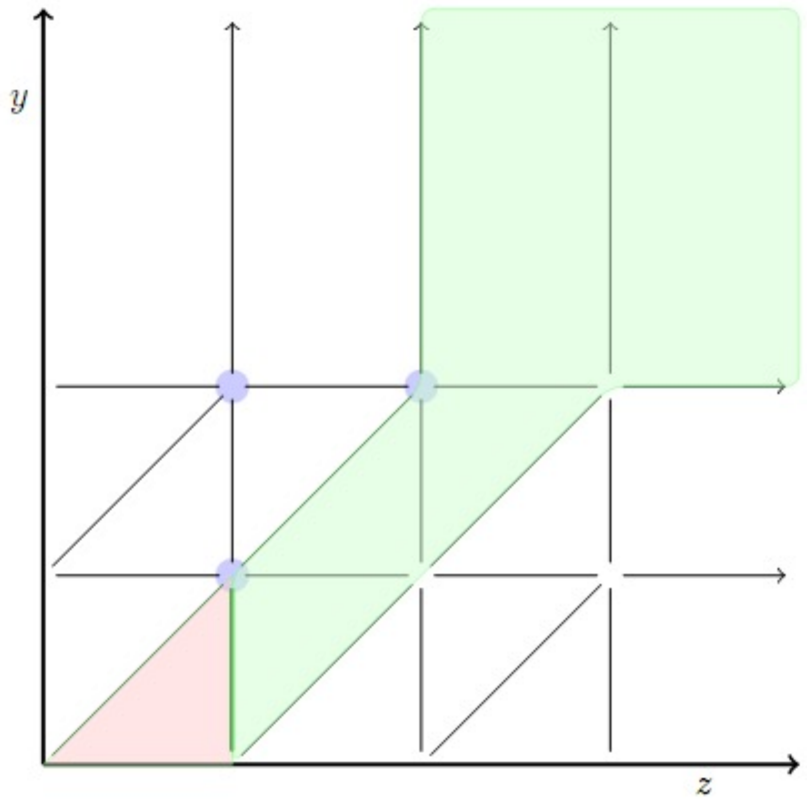
(ii) $\mathrm{R}(\mathcal{A})$ is finite state machine.

- ITA (clock constants) defines a set of clock regions, say $\mathrm{C}_{\mathcal{A}}$. The set of states $Q_R = C_{\mathcal{A}} \times L$
- $Q_0 \subseteq Q$is the set of states contain initial set Θ of $\mathcal{A}$
- D: We add the transitions between Q (regions)
  - **Time successors**: Consider two clock regions γ and γ', we say that γ' is a time successor of γ if there exits a trajectory of ITA starting from γ that ends in γ'
  - **Discrete transitions**: Same as the ITA

Theorem. A mode of ITA $\mathcal{A}$ is reachable iff it is also reachable in $\mathrm{R}(\mathcal{A})$.

(we say that $\mathrm{R}(\mathcal{A})$ is *time abstract bisimilar* to $\mathcal{A}$)

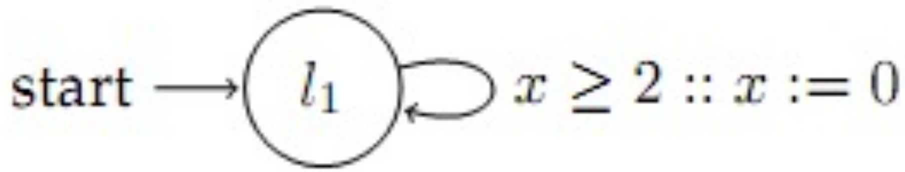
Time successors



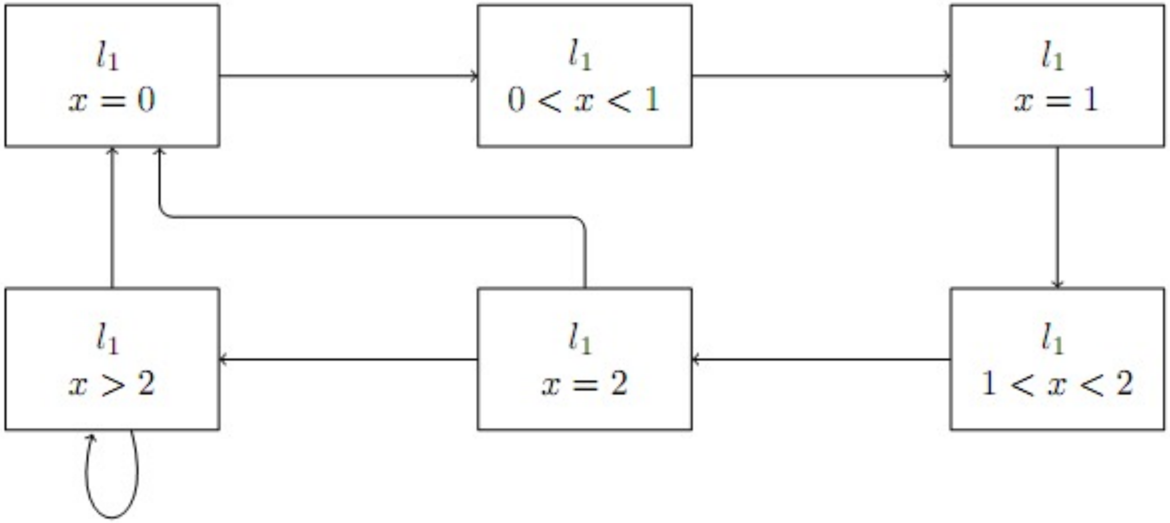
The clock regions in blue are time successors of the clock region in red.

Example 1: Region Automata

ITA

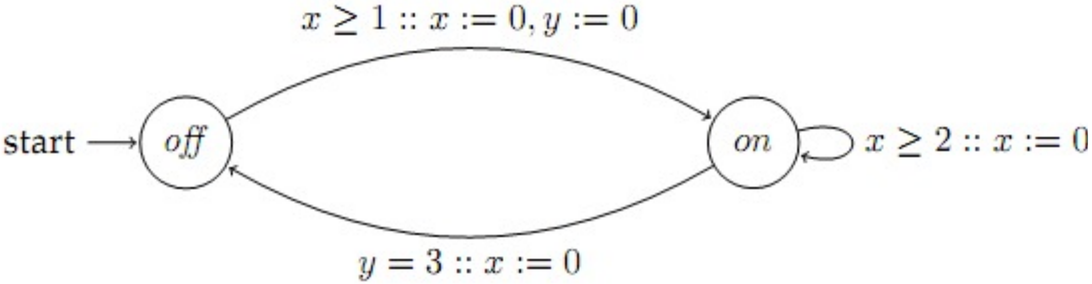


Corresponding FA

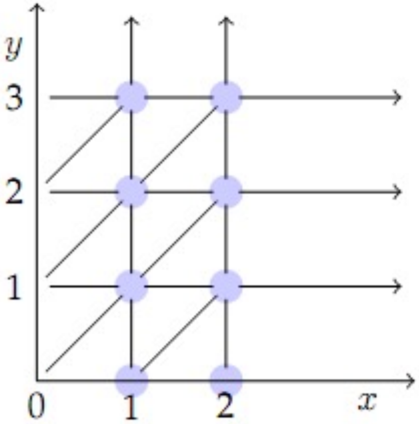


Example 2

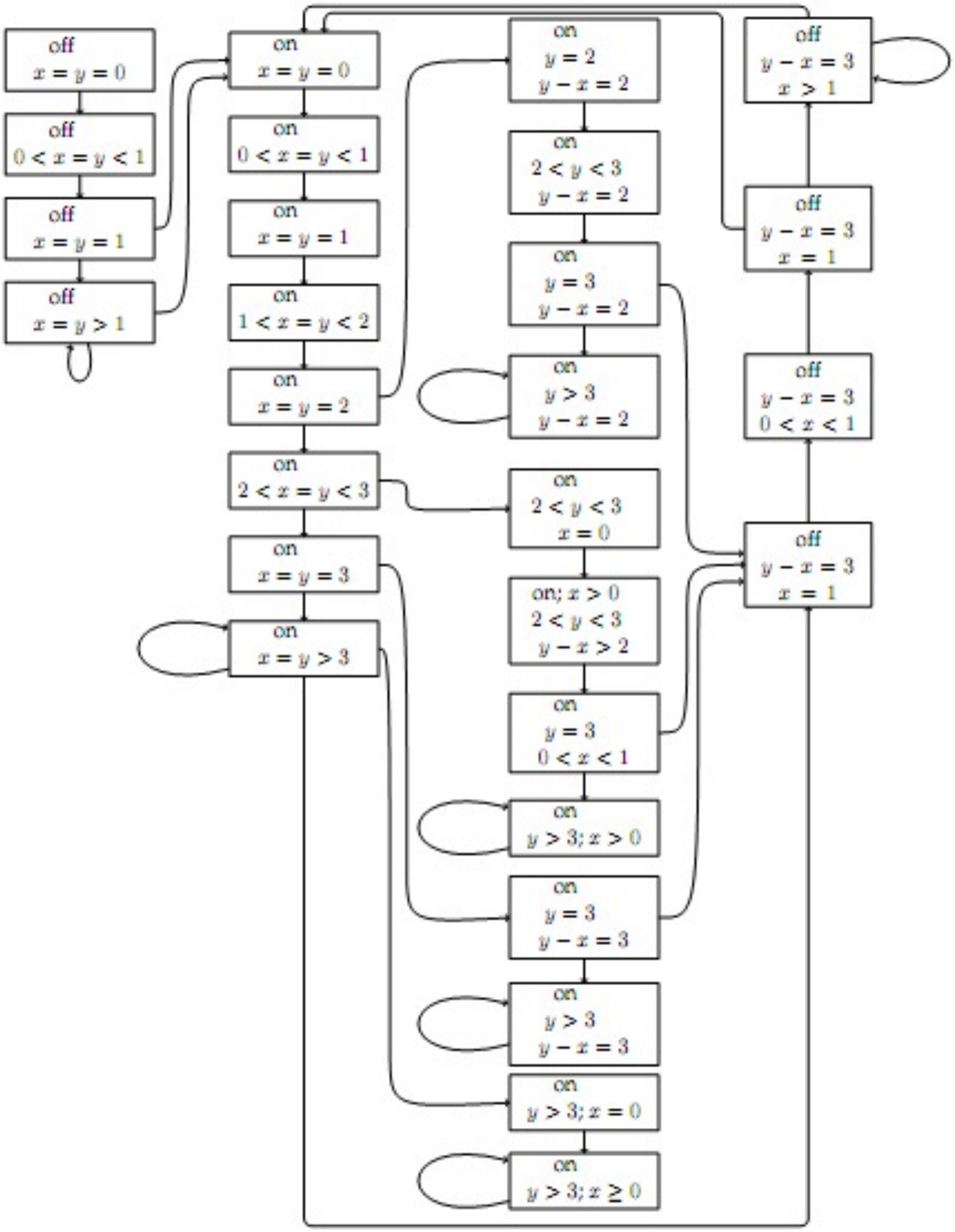
ITA



Clock Regions



Corresponding FA



$$|X|!\,2^{|X|}\prod_{z \in X}(2c_{\mathcal{A}z} + 2)$$

Drastically increasing with the number of clocks

Special Classes of Hybrid Automata

- Finite Automata
- Integral Timed Automata ←
- Rational time automata
- Multirate automata
- Rectangular Initialized HA

- Rectangular HA

- Linear HA

- Nonlinear HA

ACM NEWS: In Space, No One Can Fix Your Sign Errors--- Paul Cheng & Peter Carian

15,000 satellite launches planned for the decade

5.3% satellites are lost in the first year, 42% of those in first 2 months

Most common cause **sign errors**: SW/HW parameter used the wrong way

- fitting acceleration sensors the wrong way
- wrong usage of negative instead of positive parameters
- switching current in wrong direction in a circuit
- inverting the orientation of the electromagnets used for positioning

Genesis (2001) for capturing particles from the solar wind, pounded into the Utah desert unbraked because a pencil-eraser-sized deceleration sensor was mounted upside-down.

Clocks and **Rational** Clock Constraints

- A **clock variable** x is a continuous (analog) variable of type real such that along any trajectory τ of x, for all t ∈ $\tau.dom, (\tau \downarrow x)(t) = t$.

- For a set X of clock variables, the set Φ(X) of ***rational* clock constraints** are expressions defined by the syntax:

  g ::= x $\leq q$ | $x \geq q$ | $\neg\ g$ | $g_1 \wedge\ \ g_2$

  where $x \in X\ and\ q \in\ \mathbb{Q}$

- Examples: x = 10.125; x ∈ [2.99, 5); true are valid rational clock constraints

- Semantics of clock constraints $[g]$

Step 1. Rational Timed Automata

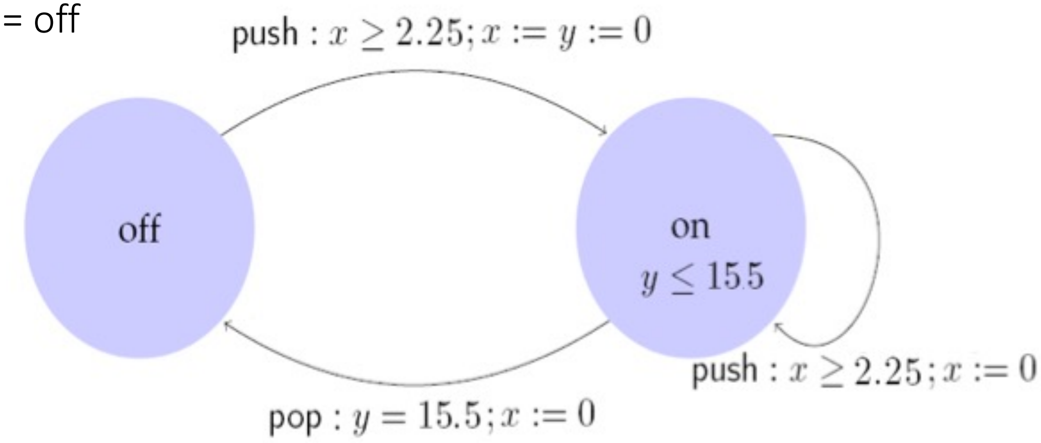
Definition. A *rational timed automaton* is a HA $\boldsymbol{\mathcal{A}}$ = $\langle V, \Theta, A, \mathcal{D}, \mathcal{T} \rangle$ where

- V = X ∪ $\{loc\}$, where X is a set of n clocks and l is a discrete state variable of finite type Ł
- A is a finite set
- $\mathcal{D}$ is a set of transitions such that
  - The guards are described by rational clock constraings $\Phi(X)$
  - $\langle x, l \rangle - a \rightarrow \langle x', l' \rangle$ implies either $x' = x$ or $x = 0$
- $\mathcal{T}$ set of clock trajectories for the clock variables in X

Example: Rational Light switch

Switch can be turned on whenever at least 2.25 time units have elapsed since the last turn off or on. Switches off automatically 15.5 time units after the last on.

```
automaton Switch
 internal push; pop
  variables
    internal x, y:Real := 0, loc:{on,off} := off
   transitions
     push
       pre x >=2.25
       eff if loc = on then y := 0 fi; x := 0; loc := off
     pop
       pre y = 15.5 ∧ loc = off
       eff x := 0
   trajectories
     invariant loc = on ∨ loc = off
     stop when y = 15.5 ∧ loc = off
     evolve d(x) = 1; d(y) = 1
```


push : $x \geq 2.25; x := y := 0$
off
on
$y \leq 15.5$
push : $x \geq 2.25; x := 0$
pop : $y = 15.5; x := 0$

Control State (Location) Reachability Problem

- Given an RTA, check if a particular mode is reachable from the initial states
- Is problem decidable?
- Yes
- Key idea:
  - Construct a ITA that has exactly same mode reachability behavior as the given RTA (timing behavior may be different)
  - Check mode reachability for ITA

Construction of ITA from RTA

- Multiply all rational constants by a factor q that make them integral
- Make d(x) = q for all the clocks

- RTA Switch reaches the same control locations as the ITA Iswitch

- Simulation relation R is given by
- (**u**,**s**) ∈ R iff **u**.x = 4 **s**.x and **u**.y = 4 **s**.y

```
automaton ISwitch
internal push; pop
variables
  internal x, y:Real := 0, loc:{on,off} := off
transitions
  push
    pre x >=  9
    eff if loc = on then y := 0 fi; x := 0; loc := off
   pop
     pre y = 62 ∧ loc = off
     eff x := 0
trajectories
   invariant loc = on ∨ loc = off
   stop when y = 62 ∧ loc = off
   evolve d(x) = 4; d(y) = 4
```

Step 2. Multi-Rate Automaton

- **Definition.** A multirate automaton is $\boldsymbol{\mathcal{A}}$ = $\langle V, Q, \Theta, A, \mathcal{D}, \mathcal{T} \rangle$ where
  - V = X ∪ $\{loc\}$, where X is a set of n continuous variables and loc is a discrete state variable of finite type Ł
  - A is a finite set of actions
  - $\mathcal{D}$ is a set of transitions such that
    - The guards are described by rational clock constraings $\Phi(X)$
    - $\langle x, l \rangle - a \rightarrow \langle x', l' \rangle$ implies either $x' = c \; or \; x' = x$
  - $\mathcal{T}$ set of trajectories such that

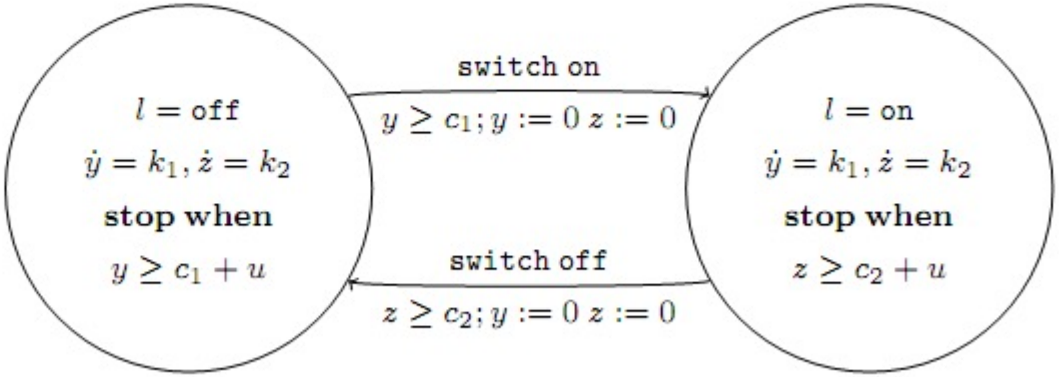
for each variable $x \in X \; \exists k \; such \; that \; \tau \in \mathcal{T}, t \in \tau.dom$

$$\tau(t).x = \tau(0).x + k\,t$$

Control State (Location) Reachability Problem

- Given an MRA, check if a particular location is reachable from the initial states
- Is problem is decidable? Yes
- Key idea:
  - Construct a RTA that is bisimilar to the given MRA

Example: Multi-rate to rational TA



$l = \text{off}$

$\dot{y} = k_1, \dot{z} = k_2$

stop when

$y \geq c_1 + u$

switch on

$y \geq c_1; y := 0\ z := 0$

switch off

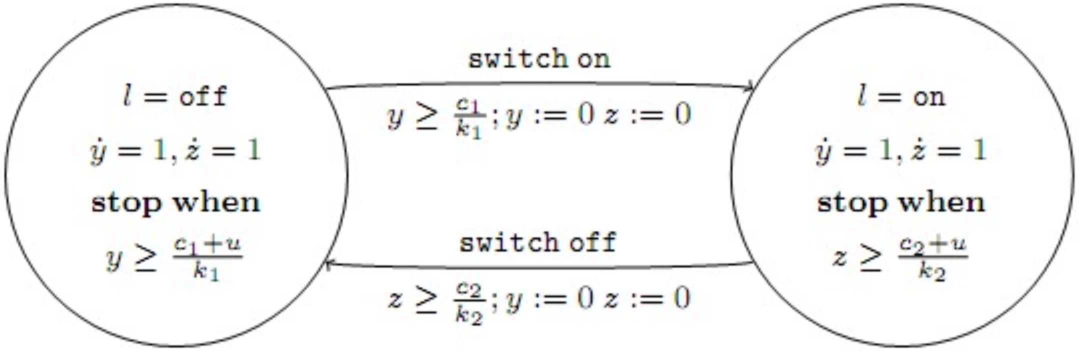
$z \geq c_2; y := 0\ z := 0$

$l = \text{on}$

$\dot{y} = k_1, \dot{z} = k_2$

stop when

$z \geq c_2 + u$



$l = \text{off}$

$\dot{y} = 1, \dot{z} = 1$

stop when

$y \geq \frac{c_1+u}{k_1}$

switch on

$y \geq \frac{c_1}{k_1}; y := 0\ z := 0$

switch off

$z \geq \frac{c_2}{k_2}; y := 0\ z := 0$

$l = \text{on}$

$\dot{y} = 1, \dot{z} = 1$

stop when

$z \geq \frac{c_2+u}{k_2}$

Step 3. Rectangular HA

Definition. A rectangular hybrid automaton (RHA) is a HA $\boldsymbol{\mathcal{A}} = \langle V, A, \mathcal{T}, \mathcal{D} \rangle$ where

- V = X ∪ $\{loc\}$, where X is a set of n continuous variables and loc is a discrete state variable of finite type Ł
- A is a finite set
- $\mathcal{T} = \cup_{\ell} \mathcal{T}_{\ell}$ set of trajectories for X
  - For each $\tau \in \mathcal{T}_{\ell}, x \in X$ either (i) $d(x) = k_{\ell}$ or (ii) $d(x) \in [k_{\ell 1}, k_{\ell 2}]$
  - Equivalently, (i) $\tau(t)\lceil x = \tau(0)\lceil x + k_{\ell} t$

    (ii) $\tau(0)\lceil x + k_{\ell 1} t \leq \tau(t)\lceil x \leq \tau(0)\lceil x + k_{\ell 2} t$
- $\mathcal{D}$ is a set of transitions such that
  - Guards are described by rational clock constraings
  - $\langle x, l \rangle \rightarrow_a \langle x', l' \rangle$ implies $x' = x\ or x' \in [c_1, c_2]$

CSR Decidable for RHA?

- Given an RHA, check if a particular location is reachable from the initial states?
- Is this problem decidable? No
  - **[Henz95]** Thomas Henzinger, Peter Kopke, Anuj Puri, and Pravin Varaiya. What's Decidable About Hybrid Automata?. Journal of Computer and System Sciences, pages 373–382. ACM Press, 1995.
  - CSR for RHA reduction to Halting problem for 2 counter machines
  - Halting problem for 2CM known to be undecidable
  - Reduction in next lecture

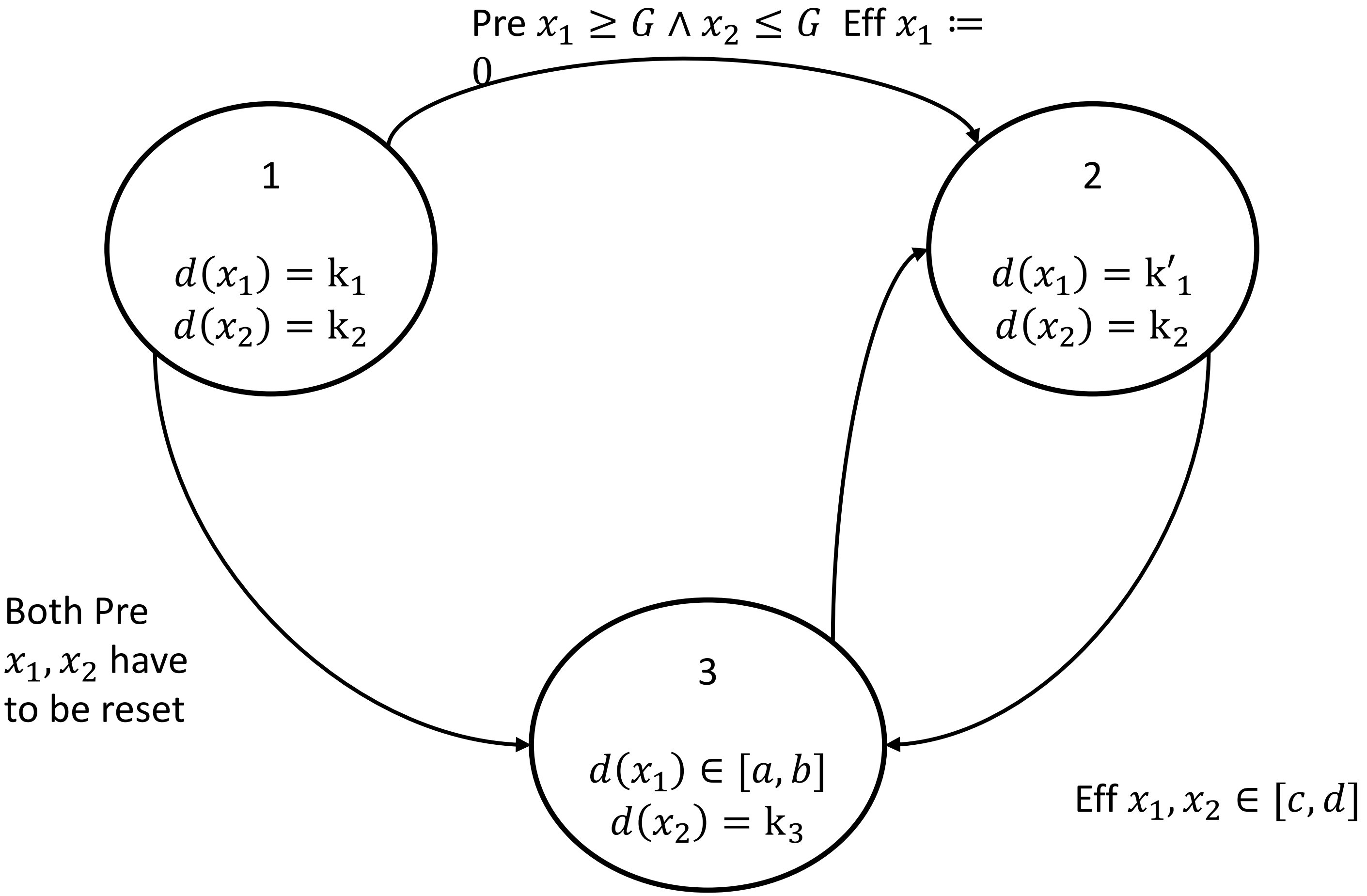
Step 4. Initialized Rectangular HA

Definition. *An initialized rectangular hybrid automaton* (IRHA) is a RHA $\boldsymbol{\mathcal{A}}$ where

- V = X ∪ $\{loc\}$, where X is a set of n continuous variables and $\{loc\}$ is a discrete state variable of finite type Ł
- A is a finite set
- $\mathcal{T}$ =∪$_\ell$ $\mathcal{T}_\ell$ set of trajectories for X
  - For each $\tau \in \mathcal{T}_\ell, x \in X$ either (i) $d(x) = k_\ell$ or (ii) $d(x) \in [k_{\ell 1}, k_{\ell 2}]$
  - Equivalently, (i) $\tau(t)\lceil x = \tau(0)\lceil x + k_\ell t$

    (ii) $\tau(0)\lceil x + k_{\ell 1} t \leq \tau(t)\lceil x \leq \tau(0)\lceil x + k_{\ell 2} t$
- $\mathcal{D}$ is a set of transitions such that
  - Guards are described by rational clock constrings
  - $\langle x, l\rangle \to_a \langle x', l'\rangle$ implies if dynamics changes from ℓ to ℓ' then $x' \in [c_1, c_2]$, otherwise $x' = x$

Example: Rectangular Initialized HA



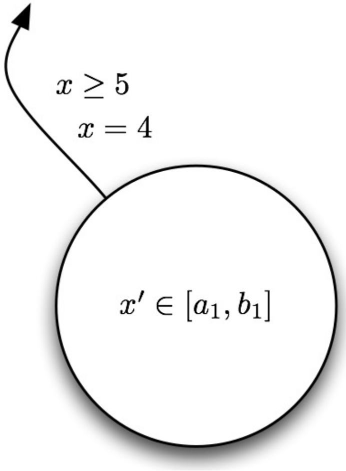
CSR Decidable for IRHA?

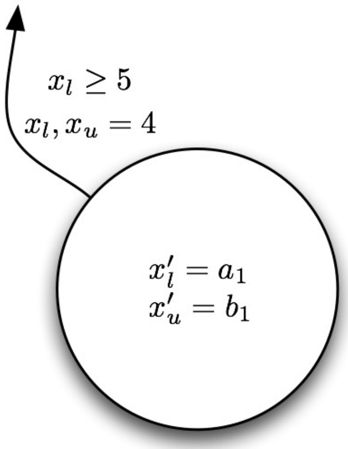
- Given an IRHA, check if a particular location is reachable from the initial states
- Is this problem decidable? Yes
- Key idea:
  - Construct a 2n-dimensional **initialized m**ulti-rate automaton that is bisimilar to the given IRHA
  - Construct a ITA that is bisimilar to the Singular TA

From IRHA to Singular HA conversion

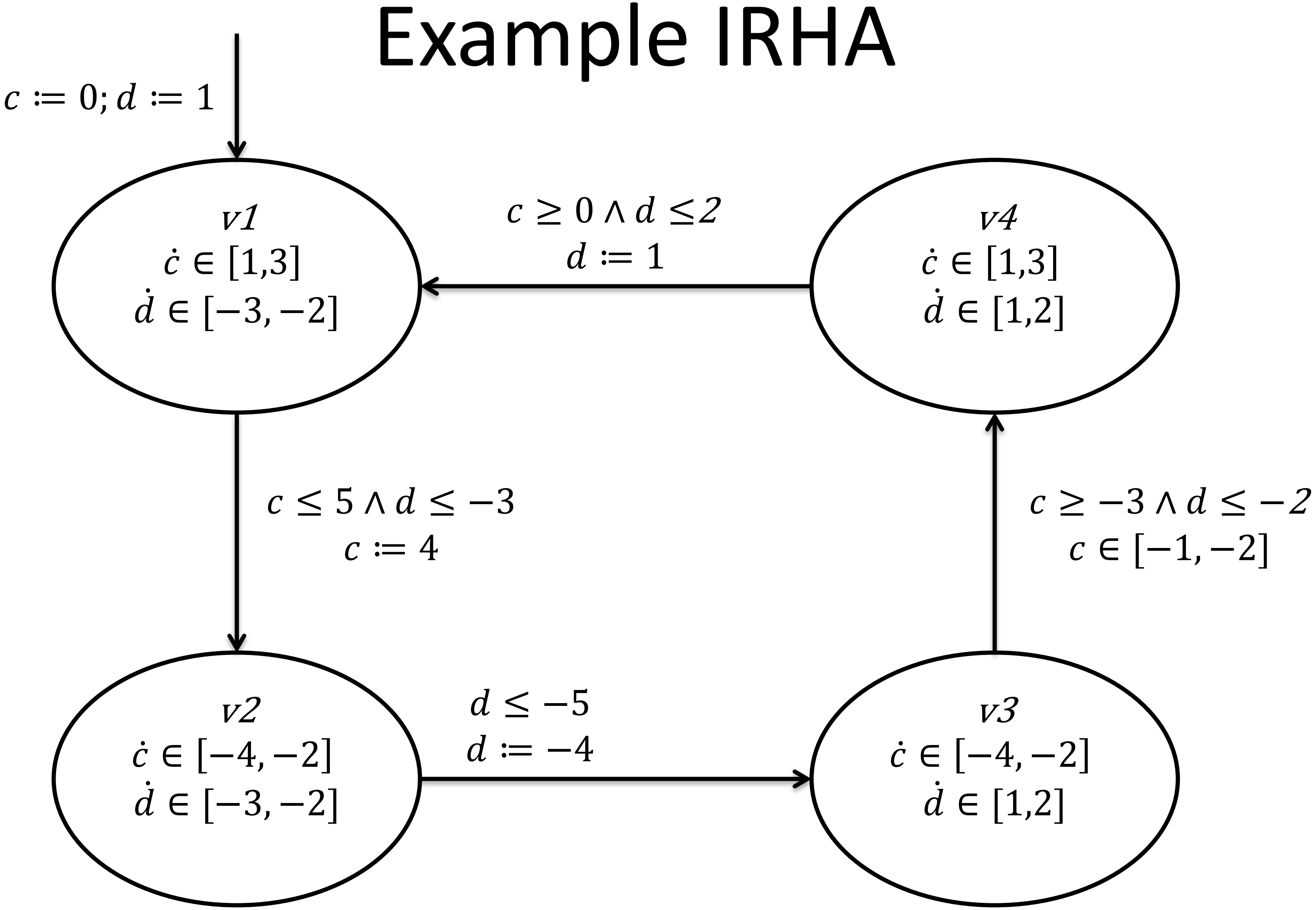
For every variable create two variables---tracking the upper and lower bounds

IRHA	MRA
x	x_ℓ ; x_u
Evolve: $d(x) \in [a_1, b_1]$	Evolve: $d(x_\ell) = a_1; d(x_u) = b_1$
Eff: $x' \in [a_1, b_1]$	Eff: $x_\ell = a_1; x_u = b_1$
$x' = c$	$x_\ell = x_u = c$
Guard: $x \geq 5$	$x_l \geq 5$
	$x_l < 5 \wedge x_u \geq 5$ Eff $x_l = 5$

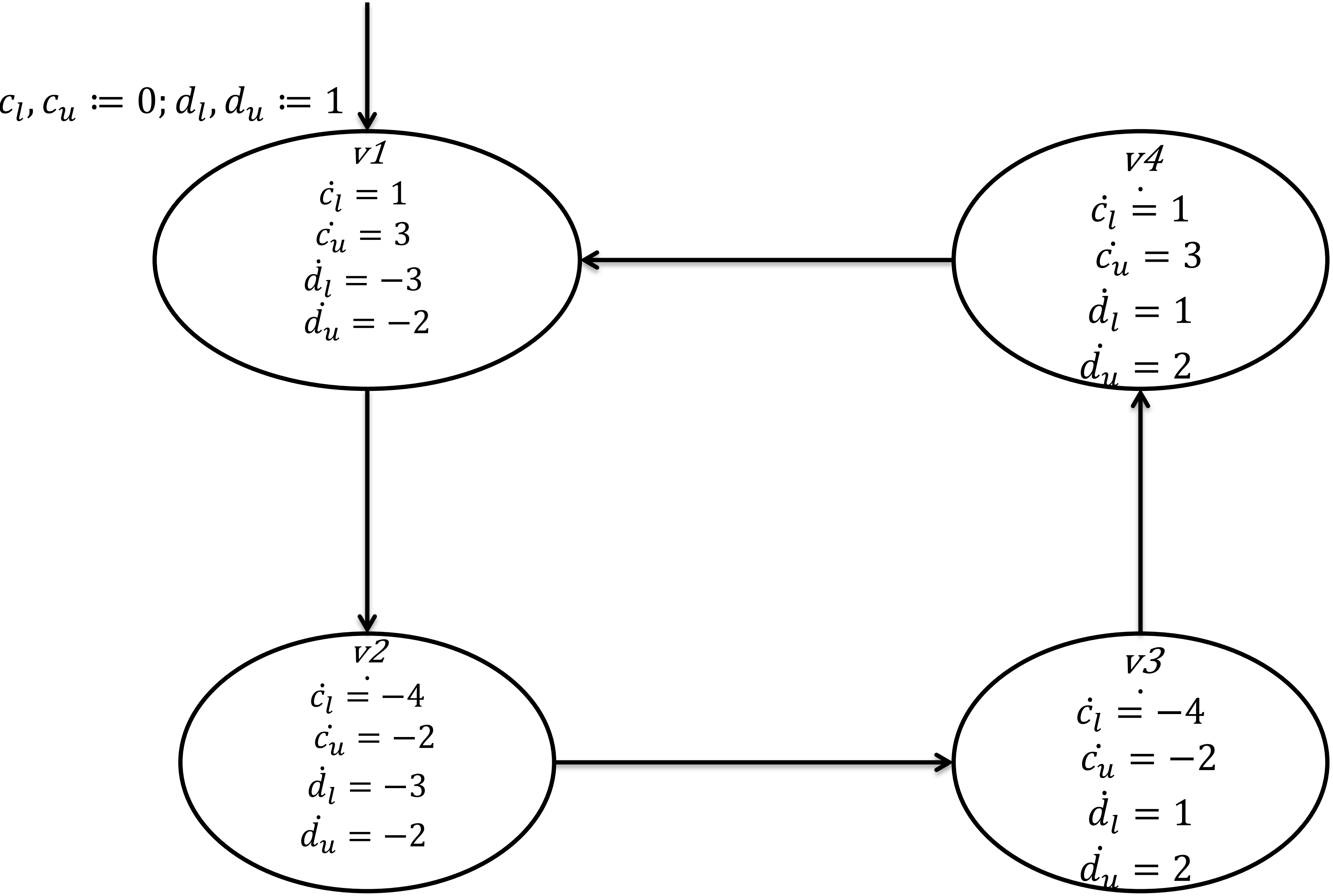




Example IRHA



Initialized Singular HA



$c_l, c_u \coloneqq 0; d_l, d_u \coloneqq 1$

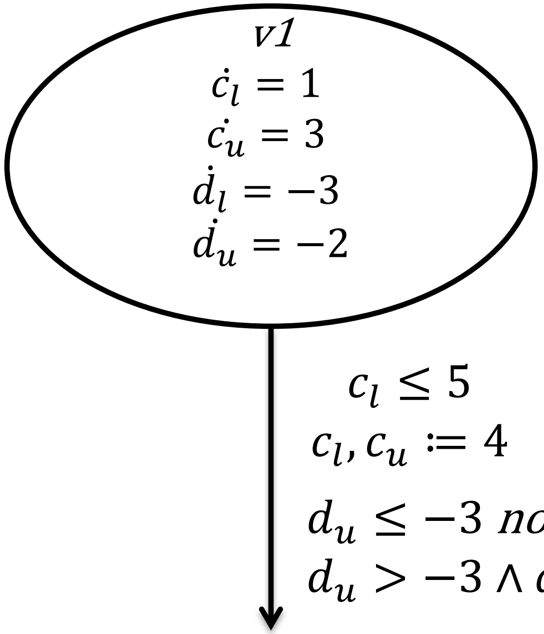
v1
$\dot{c_l} = 1$
$\dot{c_u} = 3$
$\dot{d_l} = -3$
$\dot{d_u} = -2$

v2
$\dot{c_l} = -4$
$\dot{c_u} = -2$
$\dot{d_l} = -3$
$\dot{d_u} = -2$

v3
$\dot{c_l} = -4$
$\dot{c_u} = -2$
$\dot{d_l} = 1$
$\dot{d_u} = 2$

v4
$\dot{c_l} = 1$
$\dot{c_u} = 3$
$\dot{d_l} = 1$
$\dot{d_u} = 2$

Transitions



$v1$

$\dot{c_l} = 1$

$\dot{c_u} = 3$

$\dot{d_l} = -3$

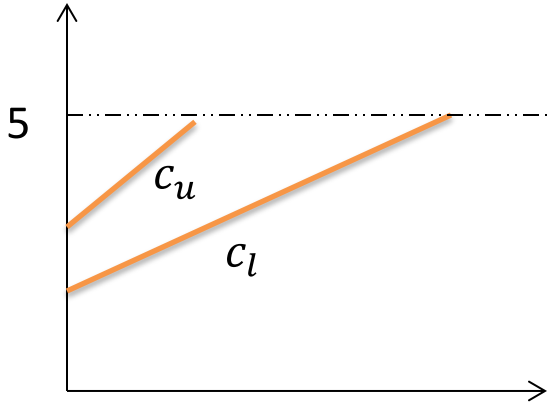
$\dot{d_u} = -2$

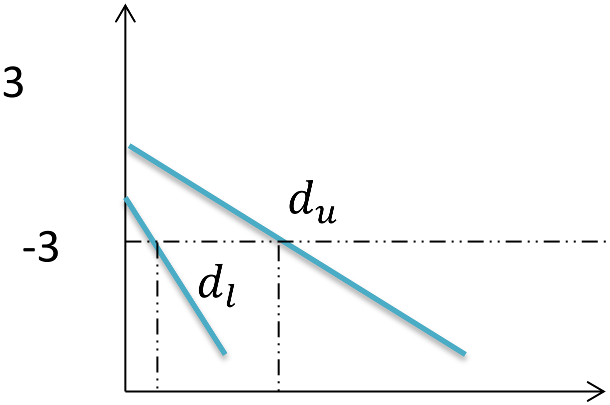
$c_l \leq 5$

$c_l, c_u := 4$

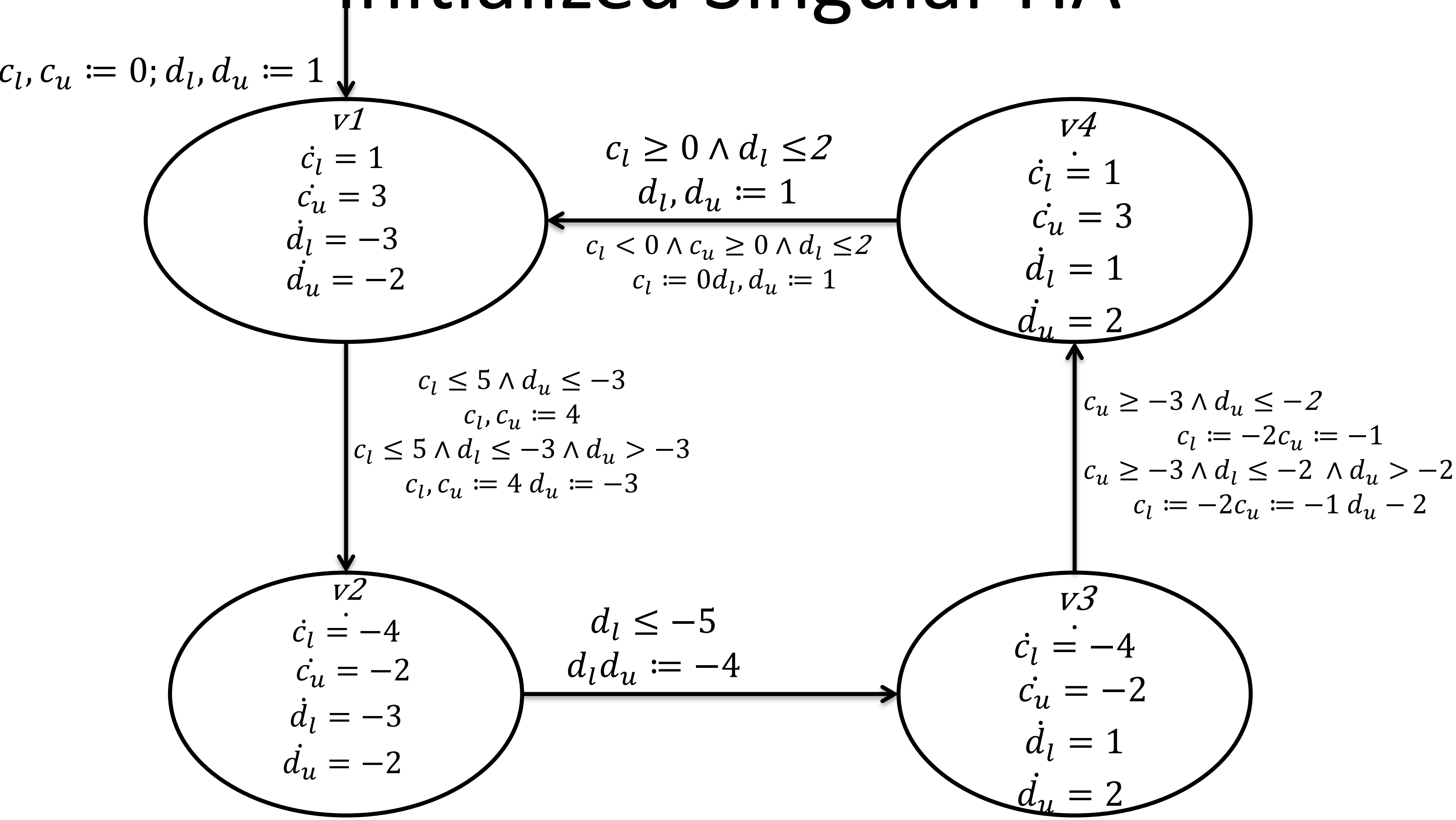
$d_u \leq -3$ *no reset*

$d_u > -3 \wedge d_l \leq -3$ d_u :=-3





Initialized Singular HA



$c_l, c_u \coloneqq 0; d_l, d_u \coloneqq 1$

v1

$\dot{c_l} = 1$

$\dot{c_u} = 3$

$\dot{d_l} = -3$

$\dot{d_u} = -2$

$c_l \geq 0 \wedge d_l \leq 2$

$d_l, d_u \coloneqq 1$

$c_l < 0 \wedge c_u \geq 0 \wedge d_l \leq 2$

$c_l \coloneqq 0 d_l, d_u \coloneqq 1$

v4

$\dot{c_l} = 1$

$\dot{c_u} = 3$

$\dot{d_l} = 1$

$\dot{d_u} = 2$

$c_l \leq 5 \wedge d_u \leq -3$

$c_l, c_u \coloneqq 4$

$c_l \leq 5 \wedge d_l \leq -3 \wedge d_u > -3$

$c_l, c_u \coloneqq 4 \; d_u \coloneqq -3$

v2

$\dot{c_l} = -4$

$\dot{c_u} = -2$

$\dot{d_l} = -3$

$\dot{d_u} = -2$

$d_l \leq -5$

$d_l d_u \coloneqq -4$

v3

$\dot{c_l} = -4$

$\dot{c_u} = -2$

$\dot{d_l} = 1$

$\dot{d_u} = 2$

$c_u \geq -3 \wedge d_u \leq -2$

$c_l \coloneqq -2 c_u \coloneqq -1$

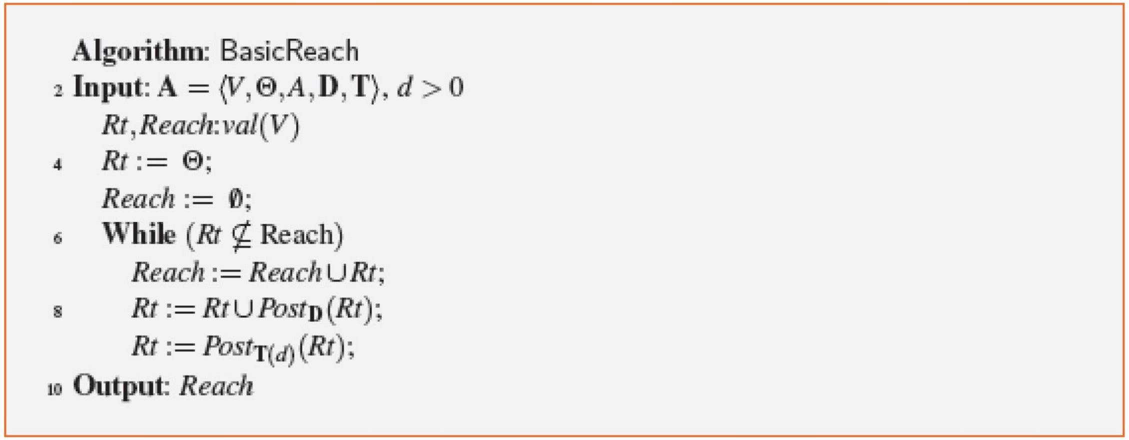
$c_u \geq -3 \wedge d_l \leq -2 \wedge d_u > -2$

$c_l \coloneqq -2 c_u \coloneqq -1 \; d_u - 2$

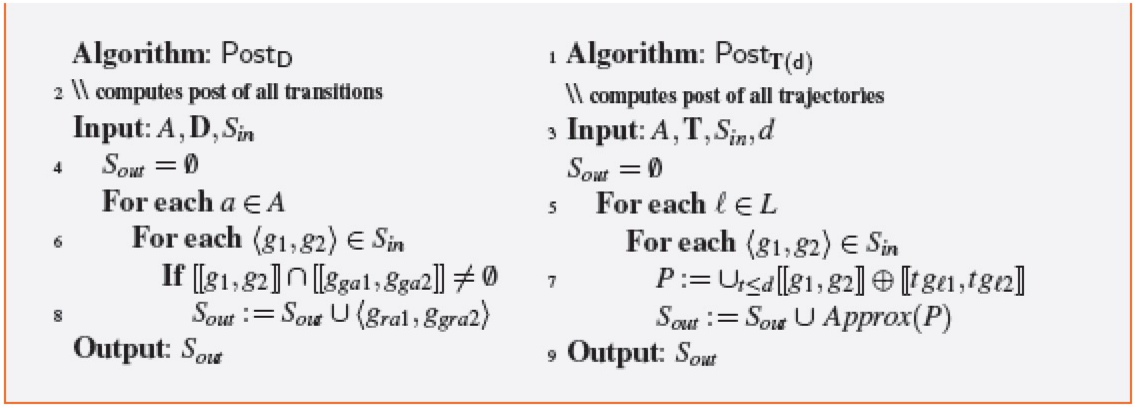
Can this be further generalized ?

- For initialized Rectangular HA, control state reachability is decidable
  - Can we drop the initialization restriction?
    - No, problem becomes undecidable (next time)
  - Can we drop the rectangular restriction?
    - No, problem becomes undecidable

Practical reachability



```
Algorithm: BasicReach
Input: $\mathbf{A} = \langle V, \Theta, A, \mathbf{D}, \mathbf{T} \rangle$, $d > 0$
  $Rt, Reach{:}val(V)$
  $Rt := \Theta$;
  $Reach := \emptyset$;
  While ($Rt \not\subseteq$ Reach)
    $Reach := Reach \cup Rt$;
    $Rt := Rt \cup Post_{\mathbf{D}}(Rt)$;
    $Rt := Post_{\mathbf{T}(d)}(Rt)$;
Output: $Reach$
```



```
Algorithm: $\mathsf{Post}_{\mathsf{D}}$
\\ computes post of all transitions
Input: $A, \mathbf{D}, S_{in}$
  $S_{out} = \emptyset$
  For each $a \in A$
    For each $\langle g_1, g_2 \rangle \in S_{in}$
      If $[[g_1, g_2]] \cap [[g_{ga1}, g_{ga2}]] \neq \emptyset$
        $S_{out} := S_{out} \cup \langle g_{ra1}, g_{gra2} \rangle$
Output: $S_{out}$
```

```
Algorithm: $\mathsf{Post}_{\mathbf{T}(\mathsf{d})}$
\\ computes post of all trajectories
Input: $A, \mathbf{T}, S_{in}, d$
$S_{out} = \emptyset$
  For each $\ell \in L$
    For each $\langle g_1, g_2 \rangle \in S_{in}$
      $P := \cup_{t \leq d} [[g_1, g_2]] \oplus [[t g_{\ell 1}, t g_{\ell 2}]]$
      $S_{out} := S_{out} \cup Approx(P)$
Output: $S_{out}$
```

Tools:

SpaceEX

CORA

C2E2

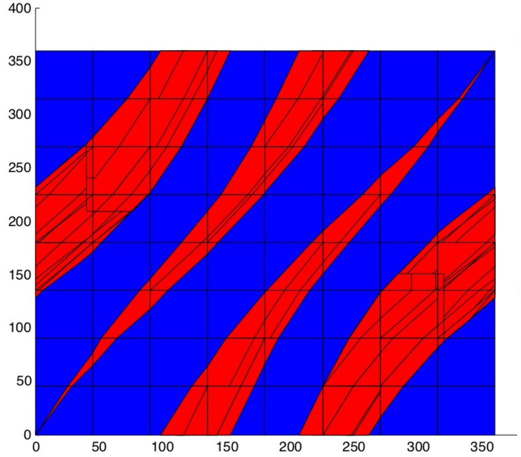
Flow*

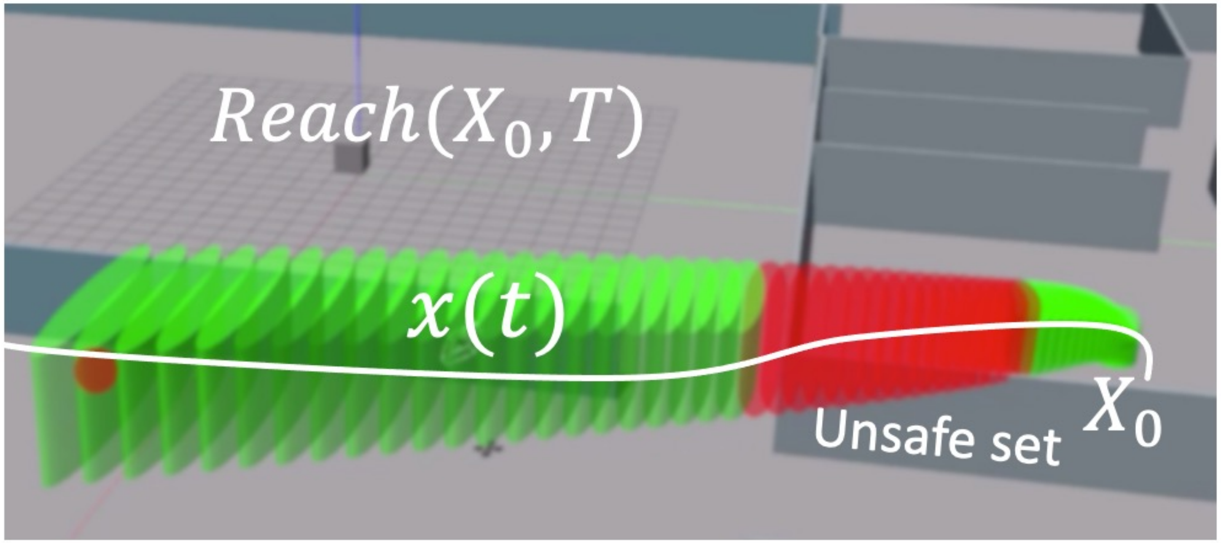
DryVR

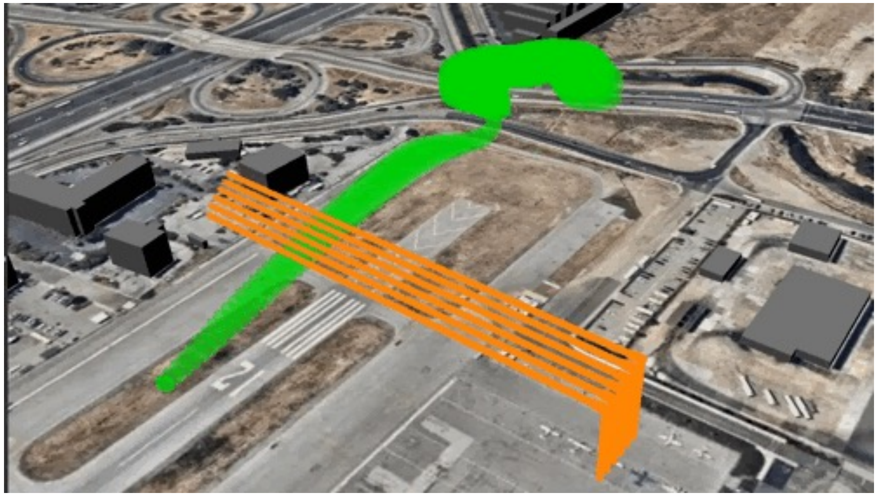
Data structures critical for reachability

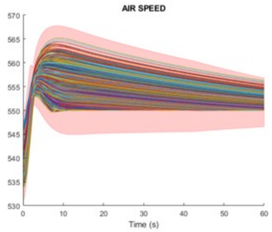
- Hyperrectangles
  - $[[\mathrm{g}_1;\mathrm{g}_2]] = \{x \in R^n \mid \ ||\mathrm{x}-\mathrm{g}_1||_\infty \leq ||\mathrm{g}_2-\mathrm{g}_1||_\infty\} = \Pi_i[g_{1i}, g_{2i}]$
- Polyhedra
- Zonotopes [Girard 2005]
- Ellipsoids [Kurzhanskiy 2001]
- Support functions [Guernic et al. 2009]
- Generalized star set [Duggirala and Viswanathan 2018]

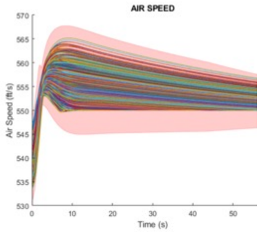
Reachability in practice



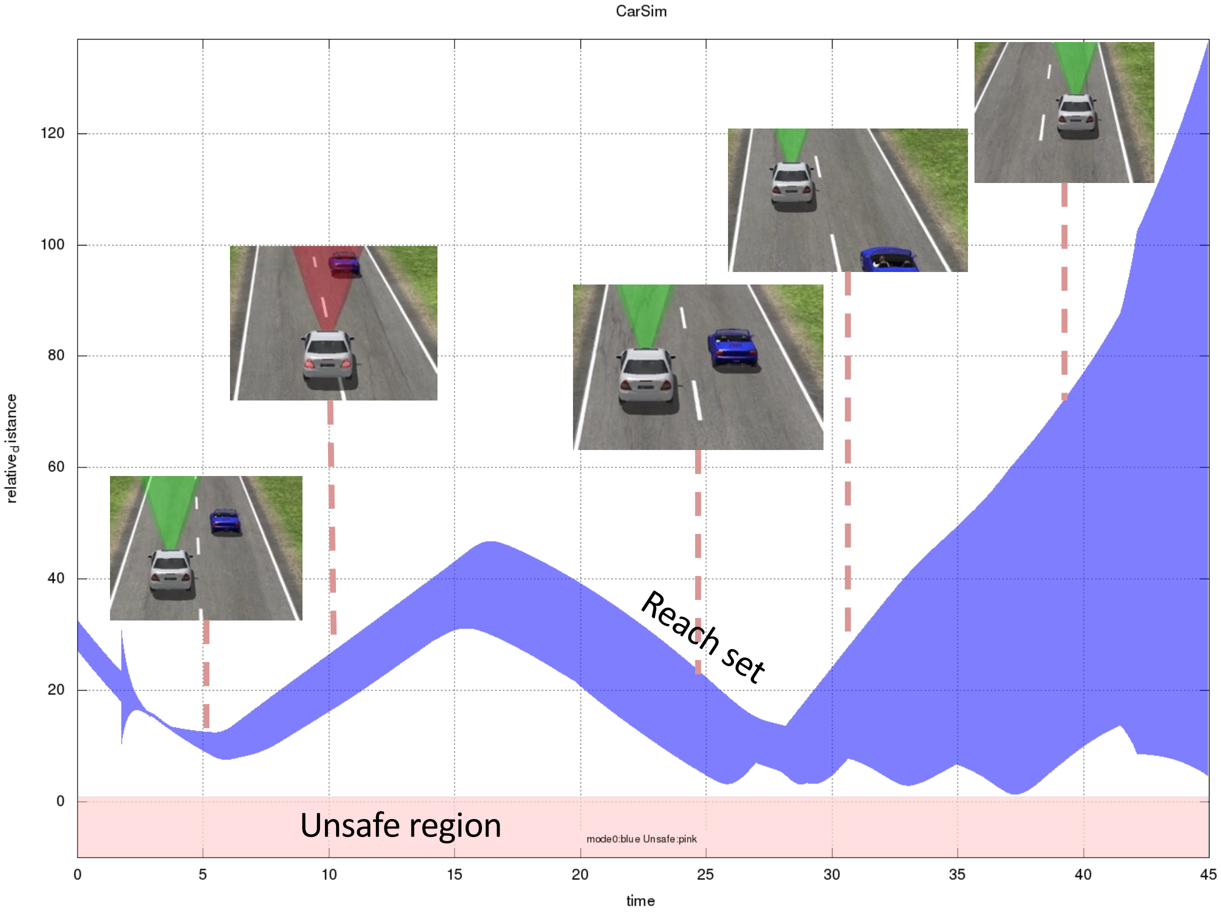


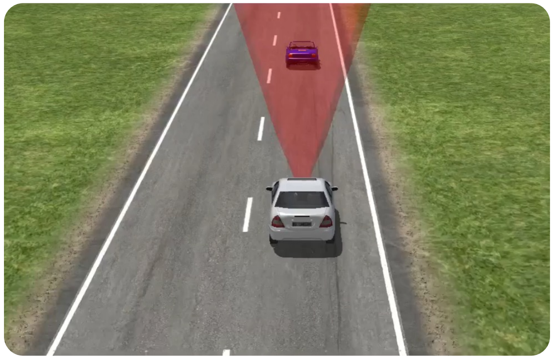






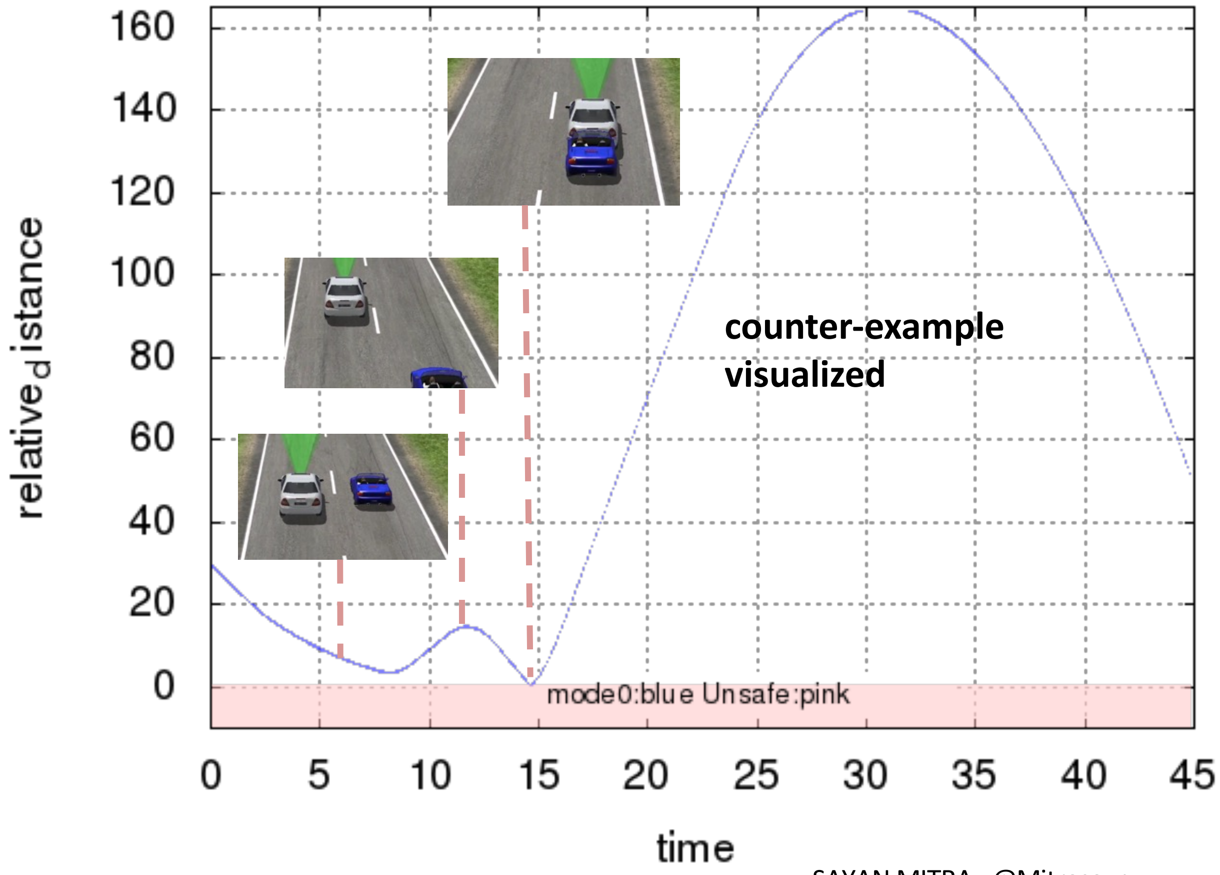
C2E2 generated safety certificate for a given user model



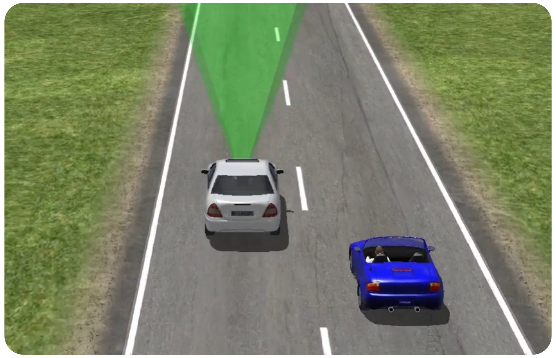


Verify no collision with **uncertainties**: speeds in [70, 85] mph and acceleration range of NPC

For a different user model C2E2 finds a corner case

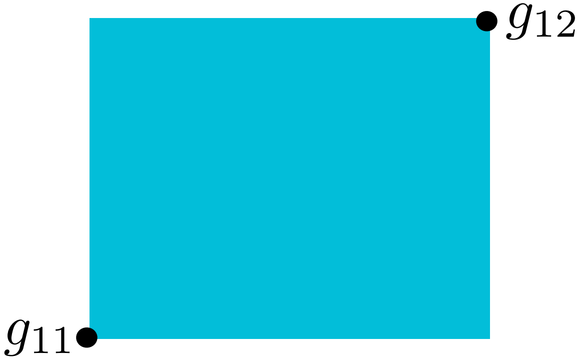


counter-example visualized

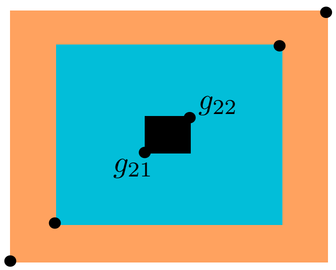


Verify no collision with **uncertainties** like speeds in [70, 85] mph and **bigger** acceleration range of NPC

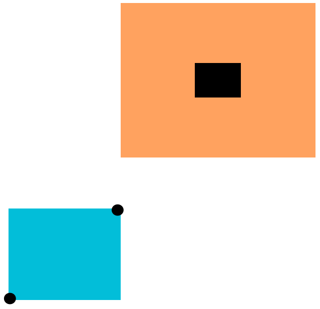
Data structures: rectangles and ellipsoids



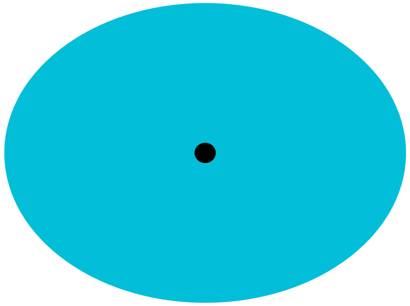
$[[g_{11}, g_{12}]]$



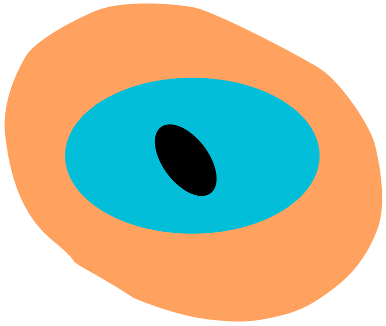
$[[g_{11}, g_{12}]] \oplus [[g_{21}, g_{22}]]$
$= [[g_{11} + g_{12}, g_{21} + g_{22}]]$



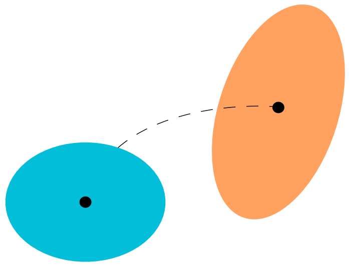
$[[g_{11}, g_{12}]] \oplus [[t.g_1, t.g_2]]$



$[[c_1, Q]]$

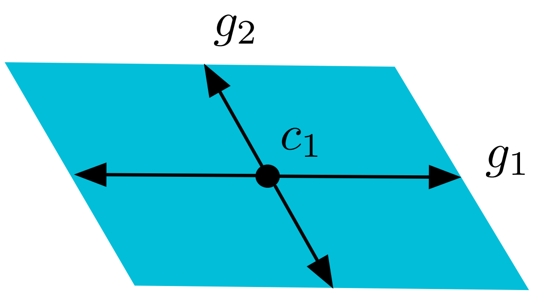


$[[c_1, Q_1]] \oplus [[c_2, Q_2]] \neq [[c_3, Q_3]]$

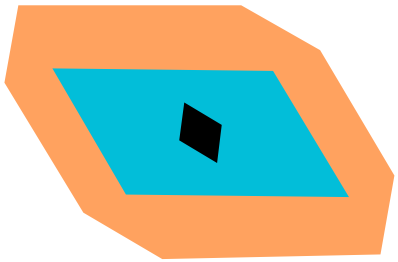


$[[Ac_1, AQA^T]]$

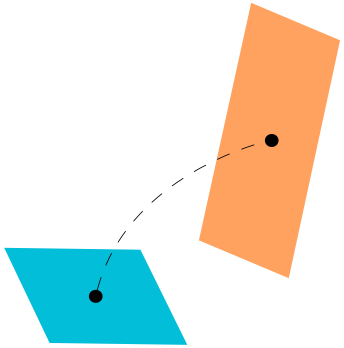
Zonotopes and polytopes



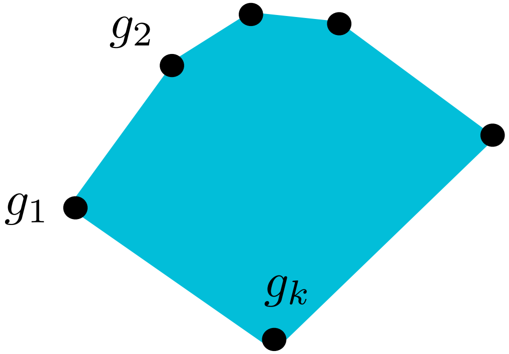
$[[c_1, \langle g_1, g_2 \rangle]]$



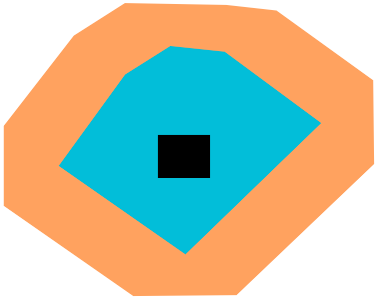
$[[c_1, \langle g_1, g_2 \rangle]] \oplus [[c_2, \langle g_1', g_2' \rangle]]$
$= [[c_1 + c_2, \langle g_1, g_1', g_2, g_2' \rangle]]$

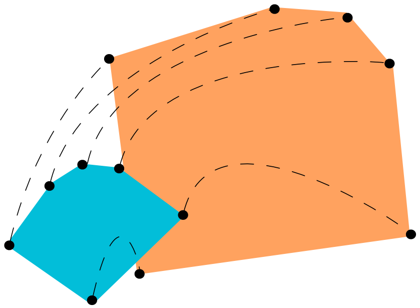


$[[Ac_1, \langle Ag_1, Ag_2 \rangle]]$



$[[A, b]]$
$[[g_1, \ldots, g_k]]$





$[[\xi(g_1, t), \ldots, \xi(g_k, t)]]$

Takeaway messages

- For restricted classes of HA, e.g., ITA, IRHA, Control state reachability is decidable (Alur-Dill)
- The problem becomes undecidable for RHA (Henzinger et al.)
  - Important message to re-focus on relaxed problem
  - Bounded time, approximate reachability
- Many tools and successful applications using iterative *Post* computations
- Choice of data-structure critical for practical performance