Lecture 20: Timed automata and its reachability

Huan Zhang

huan@huan-zhang.com

Slides adapted from Prof. Sayan Mitra’s slides in Fall 2021

Review: CTL semantics

Path quantifiers

E: Exists some path

A: All paths

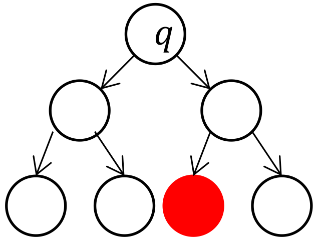
Temporal operators

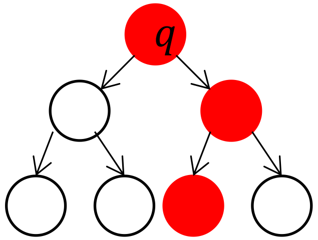
X: Next state

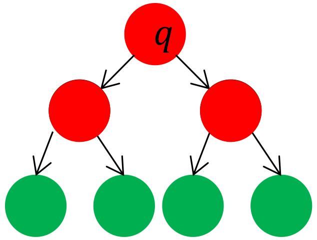
U: Until

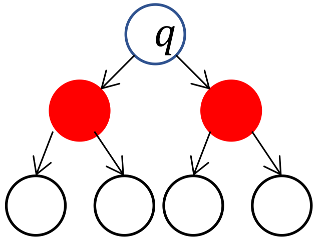
F: Eventually

G: Globally

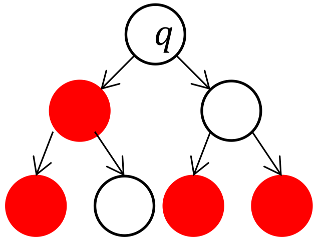


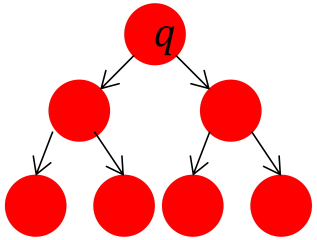


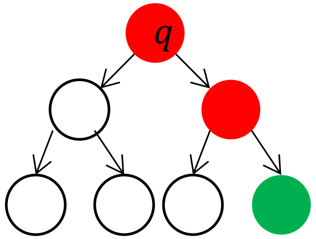


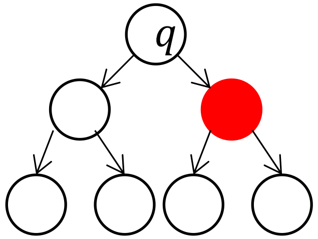


$q \vDash ?$





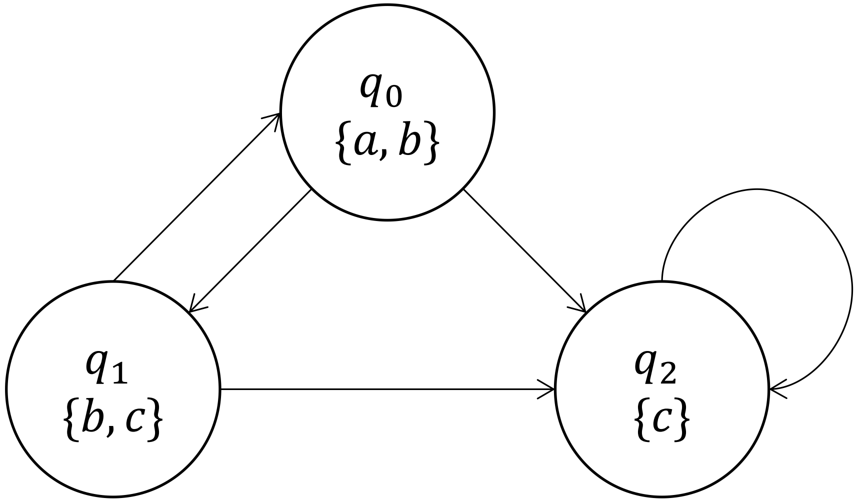




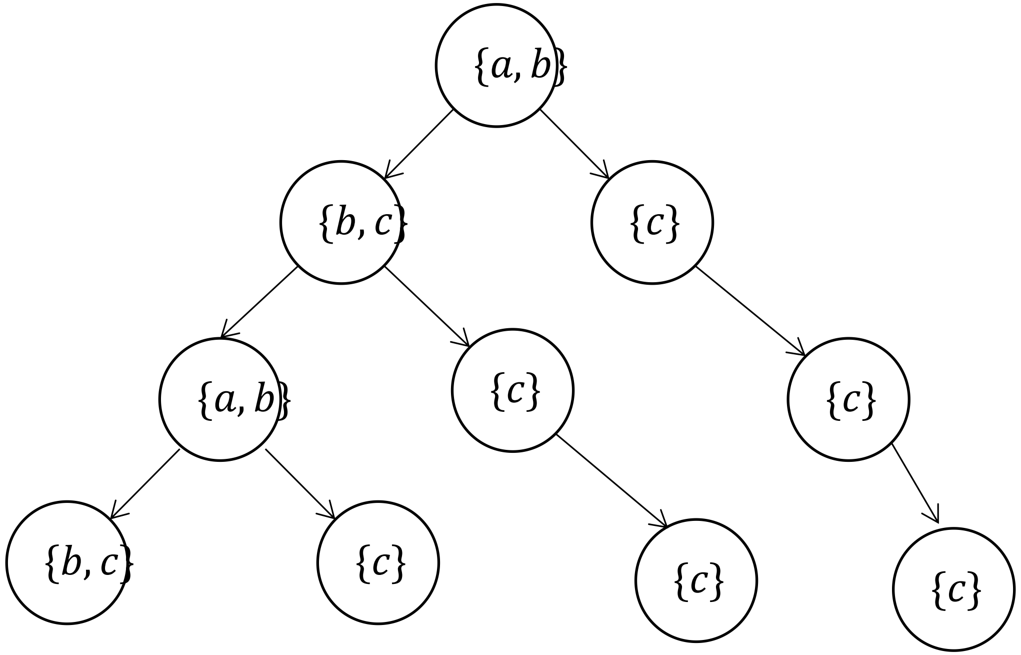
Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Review: Algorithm for deciding $\mathcal{A} \vDash f$

Since all CTL operators can be expressed by EX, EU, EG, we just need to figure out how to check these operators



$\mathcal{A}$



CTL: e.g., **AG** (a => **AF** b)

CheckEG(f_1, Q, T, L)

```
Let $Q' = \{q \in Q \mid f_1 \in label(q)\}$     $T' = \{\langle q_1, q_2\rangle \in T \mid q_1 \in Q'\}$
Let $\mathbb{C}$ be the set of nontrivial SCCs of $\langle Q', T'\rangle$
$\boldsymbol{T} = \cup_{C \in \mathbb{C}}\{q \mid q \in C\}$
for each $q \in \boldsymbol{T}$
  $label(q) := label(q) \cup \{EGf_1\}$     Everything already in the SCC satisifies
while $\boldsymbol{T} \neq \emptyset$
  for each $q \in \boldsymbol{T}$
    $\boldsymbol{T} := \boldsymbol{T} \setminus \{q\}$
    for each $q' \in Q'$ such that $(q', q) \in T'$
      if $\mathrm{E}Gf_1 \notin label(q')$ then
        $label(q') := label(q') \cup \{EGf_1\}$
         $\boldsymbol{T} := \boldsymbol{T} \cup \{q'\}$
```

Find all states in Q’ that can reach the SCCs

Proposition. For any state $label(q) \ni EGf_1$ iff $q \vDash EGf_1$.

Proposition. Finite Q therefore terminates and in $O(|Q| + |T|)$ steps.

CheckEU(f_1, f_2, Q, T, L)

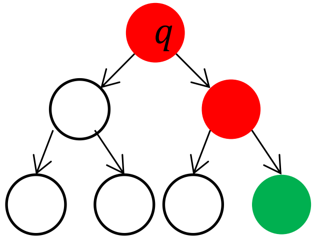
```
Let $S = \{q \in Q \mid f_2 \in label(q)\}$
for each $q \in S$
  $label(q) := label(q) \cup \{E[f_1 U f_2]\}$
```

all states where f_2 is true already satsifies

```
while $S \neq \emptyset$
  for each $q' \in S$
    $S := S \setminus \{q'\}$
    for each $q \in T^{-1}(q')$
      if $f_1 \in label(q)$ then
        $label(q) := label(q) \cup \{E[f_1 U f_2]\}$
          $S := S \cup \{q\}$
```

Check all states whose next state is q'

This if statement will be always true for EF f_2



$E[f_1 U f_2]$

Proposition. For any state $label(q) \ni E[f_1 U f_2]$ iff $q \vDash E[f_1 U f_2]$.

Proposition. Finite Q therefore terminates and in $O(|Q| + |T|)$ steps.

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Review: Structural **induction** on formula

Six cases to consider based on structure of f

$f = p,$ for some $p \in AP,$ $\forall q,\ label(q) \coloneqq label(q) \cup f$

$f = \neg f_1$ if $f_1 \notin label(q)$ then $label(q) \coloneqq label(q) \cup f$

$f = f_1 \wedge f_2$ if $f_1,\ f_2 \in label(q)$ then $label(q) \coloneqq label(q) \cup f$

$f = EXf_1$ if $\exists q' \in \mathrm{Q}$ such that $(q, q') \in T$ and $f_1 \in label(q'),$ then $label(q) \coloneqq label(q) \cup f$

$f = E[f_1 U f_2]$ $\mathrm{CheckEU}(f_1, f_2, Q, T, L)$

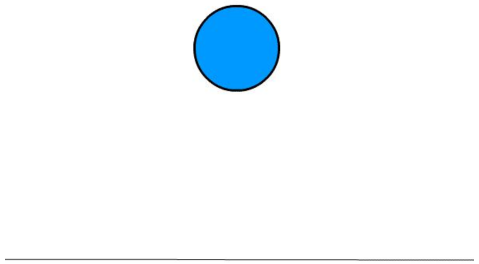
$f = EGf_1$ $\mathrm{CheckEG}(f_1, Q, T, L)$

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Today: Timed Automata & Reachability

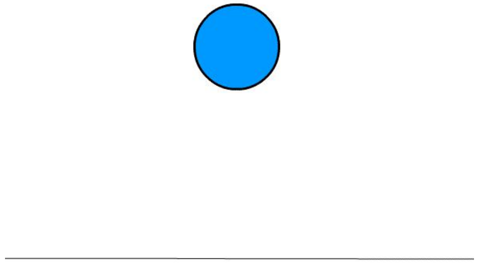
- We have studied hybrid automaton

```
automaton Bouncingball(c,h,g)
    variables: x: Reals := h, v: Reals := 0
    actions: bounce
    transitions:
        bounce
            pre $x = 0 \wedge v < 0$
            eff v := -cv
    trajectories:
            freefall
            evolve d(x) = v; d(v) = -g
            invariant $x \geq 0$
```



Timed Automata & Reachability

- We have studied hybrid automaton
- However, verification for general hyrbid automaton is in general difficult



- Special classes of hybrid automaton:
  - (Alur-Dill's) Timed Automata
  - Rectangular initialized hybrid automata
  - Linear hybrid automata
- Verification is feasible for these classes
  - Today and next a few lectures

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 ∈ $\tau.dom,\ (\tau \downarrow x)(t) = t.$
- In other words, d(x) = 1

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

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

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

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

Example: “smart” light switch

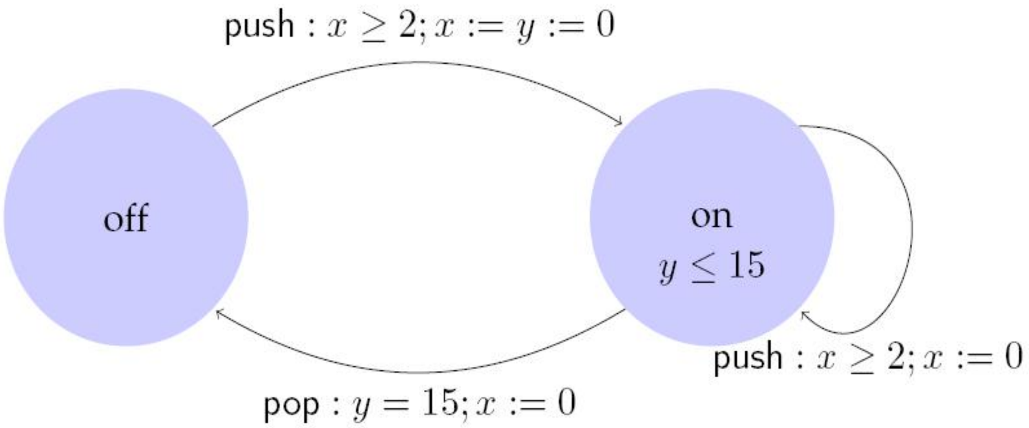
```
automaton Switch
    variables x, y:Real := 0, loc: {on,off} := off

    transitions
      push
         pre x ≥ 2
         eff if loc = off then x,y := 0; loc := on
            else x := 0
      pop
         pre y = 15 /\ loc = on
         eff x := 0; loc = off

    trajectories
         invariant  loc = off \/ y ≤ 15
         evolve d(x) = 1; d(y) = 1
```

integral clock constraints

clock guard

clock variables

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



Integral Timed Automata (ITA)

- **Definition.** A **integral timed automaton** $\mathcal{A} = \langle V,\ \Theta,\ A,\ \mathcal{D},\ \mathcal{T} \rangle$ where
  - V = X ∪ $\{l\}$, where X is a set of n clocks and l is a discrete state variable of finite type L**.** The **stata space is** val(V) × L
  - A is a finite set
  - $\mathcal{D}$ is a set of transitions such that
    - The preconditions are described by **integral** clock constrainsg $\Phi(X)$
    - $\langle x, l \rangle_a \rightarrow \langle x', l' \rangle$ implies either $x' = x$ or $x' = 0$ (time is reset to 0, or no change)
  - $\mathcal{T}$ set of clock trajectories for the clock variables in X

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
  - How many states in $\mathcal{A}$ (in general)?

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

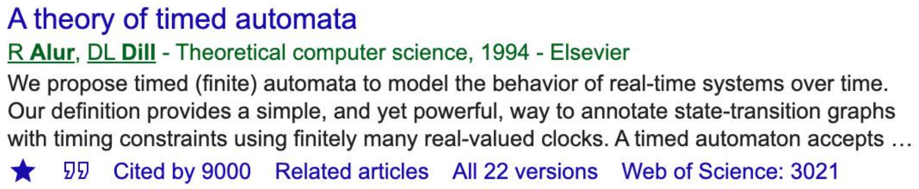
- cannot just enumerate all states - uncontable many states!

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 …

★ 99 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 $\mathbf{x_1}$ and $\mathbf{x_2}$ of the automaton $\mathcal{A}$ behave identically with respect to control state reachability (CSR)?

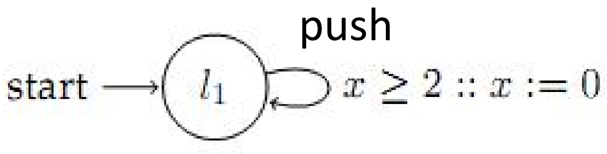
When do they satisfy the same set of clock constraints?

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

Goal: infinite number of states -> finite number of states (possible because some states are **identical**)

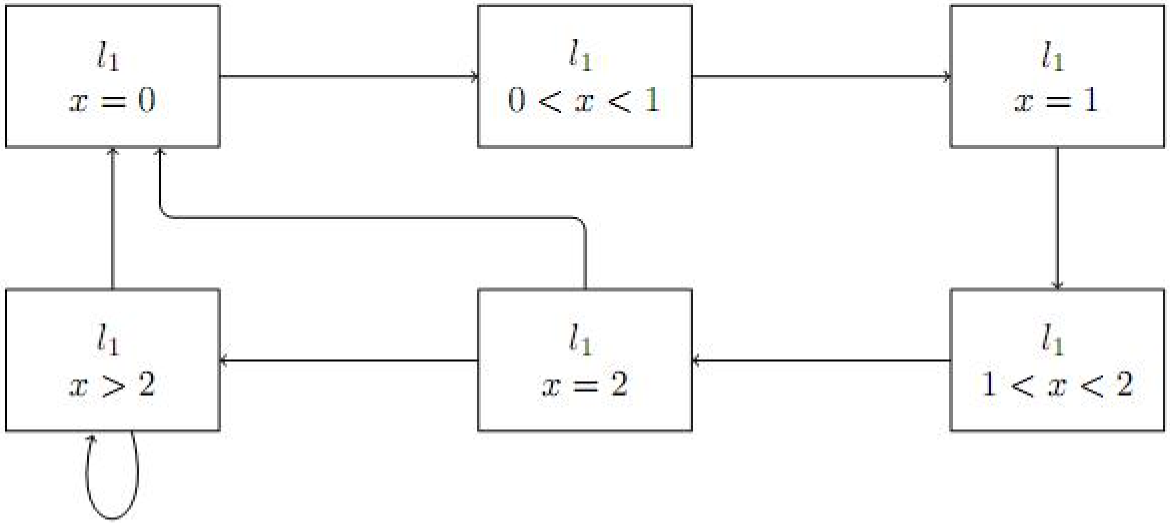
Example 1: Region Automata

ITA



Corresponding FA

Insight: there are infinite many states in ITA, but some of the are “equivalent” and can be “merged” into a single state



Given two valuations (states) $\mathbf{x_1}$ and $\mathbf{x_2}$, they are equivalent when:

$\mathbf{x_1}.loc = \mathbf{x_2}.loc$ and

$\mathbf{x_1}$ and $\mathbf{x_2}$ satisfy the same set of clock constraints (3 conditions must be all satisfied):

- int($\mathbf{x_1}.y$) = int($\mathbf{x_2}.y$), or int($\mathbf{x_1}.y$) $\geq c_{\mathcal{A}y}$ and int($\mathbf{x_2}.y$) $\geq c_{\mathcal{A}y}$
    - ($c_{\mathcal{A}y}$ is the maxium clock guard of y)
- For all clock y with $\mathbf{x_1}.y \leq c_{\mathcal{A}y}$: frac($\mathbf{x_1}.y$) $= 0$ iff frac($\mathbf{x_2}.y$) $= 0$
- For any two clocks y and z with $\mathbf{x_1}.y \leq c_{\mathcal{A}y}$ and $\mathbf{x_1}.z \leq c_{\mathcal{A}z}$: frac($\mathbf{x_1}.y$) $\leq$ frac($\mathbf{x_1}.z$) iff frac($\mathbf{x_2}.y$) $\leq$ frac($\mathbf{x_2}.z$)

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**

Equivalence relation

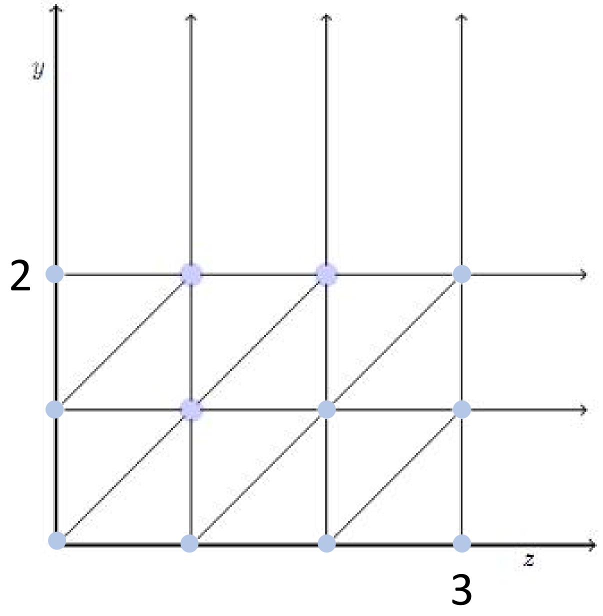
- int($\mathbf{x}_1.y$) = int($\mathbf{x}_2.y$), or int($\mathbf{x}_1.y$) $\geq c_{\mathcal{A}y}$ and int($\mathbf{x}_2.y$) $\geq c_{\mathcal{A}y}$
  - this gives the “grids”
- For all clock y with $\mathbf{x}_1.y \leq c_{\mathcal{A}y}$: frac($\mathbf{x}_1.y$) $= 0$ iff frac($\mathbf{x}_2.y$) $= 0$
  - this gives the isolated points ●
- For any two clocks y and z with $\mathbf{x}_1.y \leq c_{\mathcal{A}y}$ and $\mathbf{x}_1.z \leq c_{\mathcal{A}z}$: frac($\mathbf{x}_1.y$) $\leq$ frac($\mathbf{x}_1.z$) iff frac($\mathbf{x}_2.y$) $\leq$ frac($\mathbf{x}_2.z$)
  - this gives the “/” in each grid

Example of Two Clocks

X = {y,z}

$c_{\mathcal{A}y}$ = 2

$c_{\mathcal{A}z}$ = 3



Equivalence relation

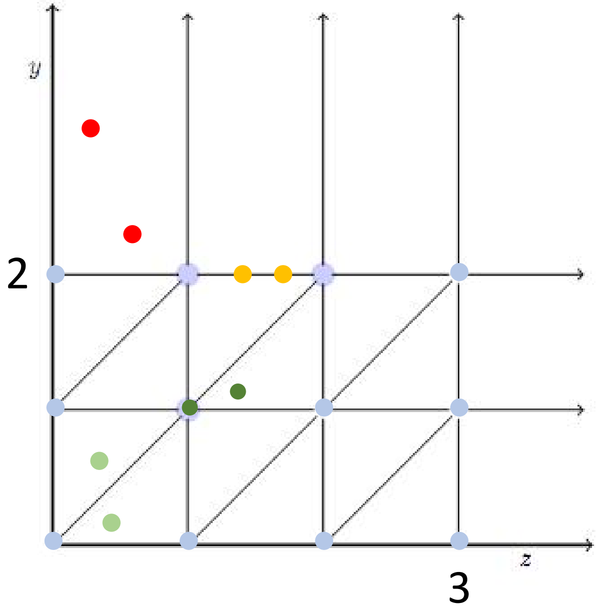
- int($\mathbf{x_1}.y$) = int($\mathbf{x_2}.y$), or int($\mathbf{x_1}.y$) $\geq c_{\mathcal{A}y}$ and int($\mathbf{x_2}.y$) $\geq c_{\mathcal{A}y}$
  - this gives the “grids”
- For all clock y with $\mathbf{x_1}.y \leq c_{\mathcal{A}y}$: frac($\mathbf{x_1}.y$) $= 0$ iff frac($\mathbf{x_2}.y$) $= 0$
  - this gives the isolated points ●
- For any two clocks y and z with $\mathbf{x_1}.y \leq c_{\mathcal{A}y}$ and $\mathbf{x_1}.z \leq c_{\mathcal{A}z}$: frac($\mathbf{x_1}.y$) $\leq$ frac($\mathbf{x_1}.z$) iff frac($\mathbf{x_2}.y$) $\leq$ frac($\mathbf{x_2}.z$)
  - this gives the “/” in each grid

Consider the two pairs of states.
Why are the states in each pair
eqiuvalent?

● ●
● ●

Consider the two pairs of states.
Why are the states in each pair
NOT eqiuvalent?

● ●
● ●



Complexity

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

X is a set of clocks

L is the type of discrete states

$c_{\mathcal{A}z}$ is the clock guard

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

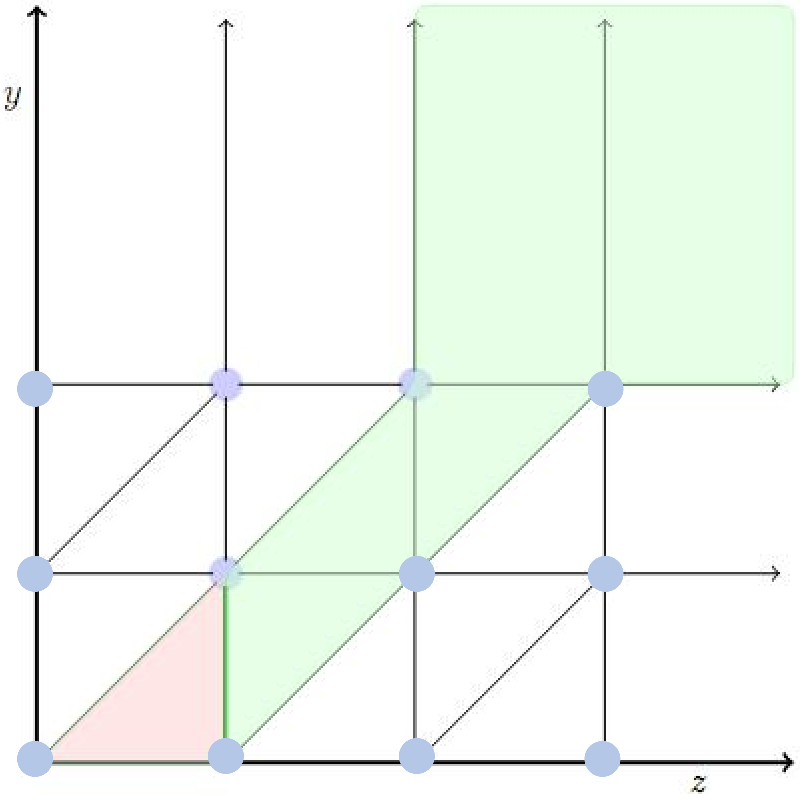
(ii) $\mathrm{R}(\mathcal{A})$ is finite state machine.

- 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 $\mathcal{A}$)

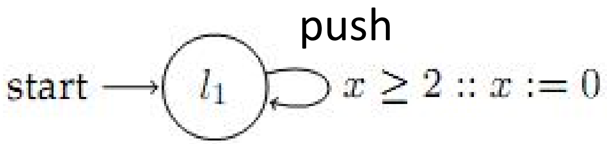
Time successors



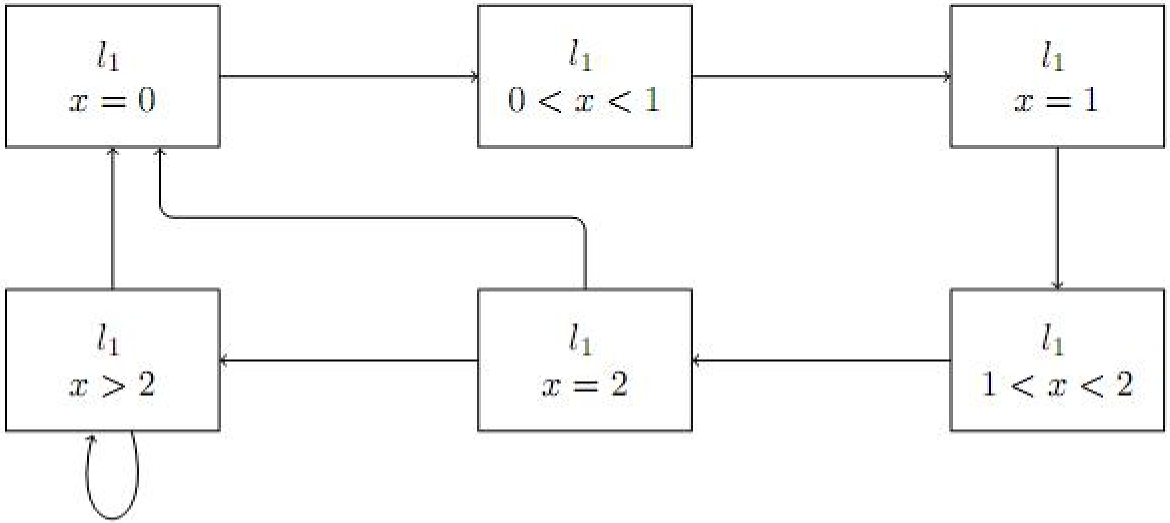
The clock regions in green are time successors of the clock region in red.

Example 1: Region Automata

ITA

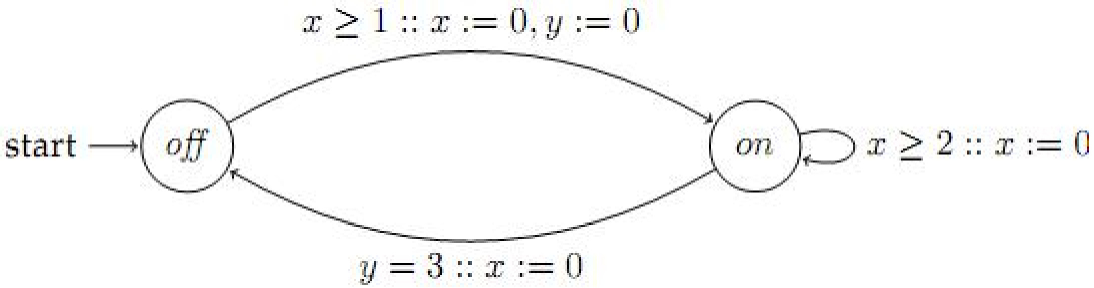


Corresponding FA

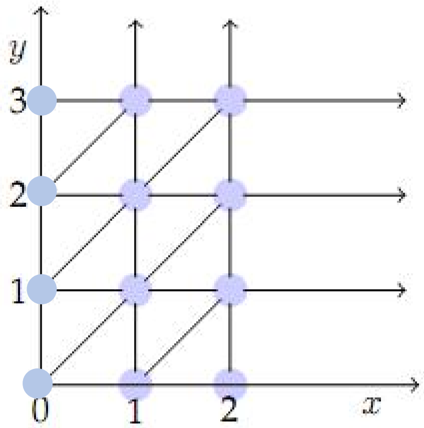


Example 2

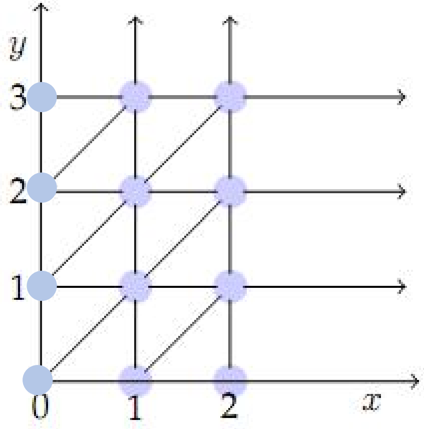
ITA

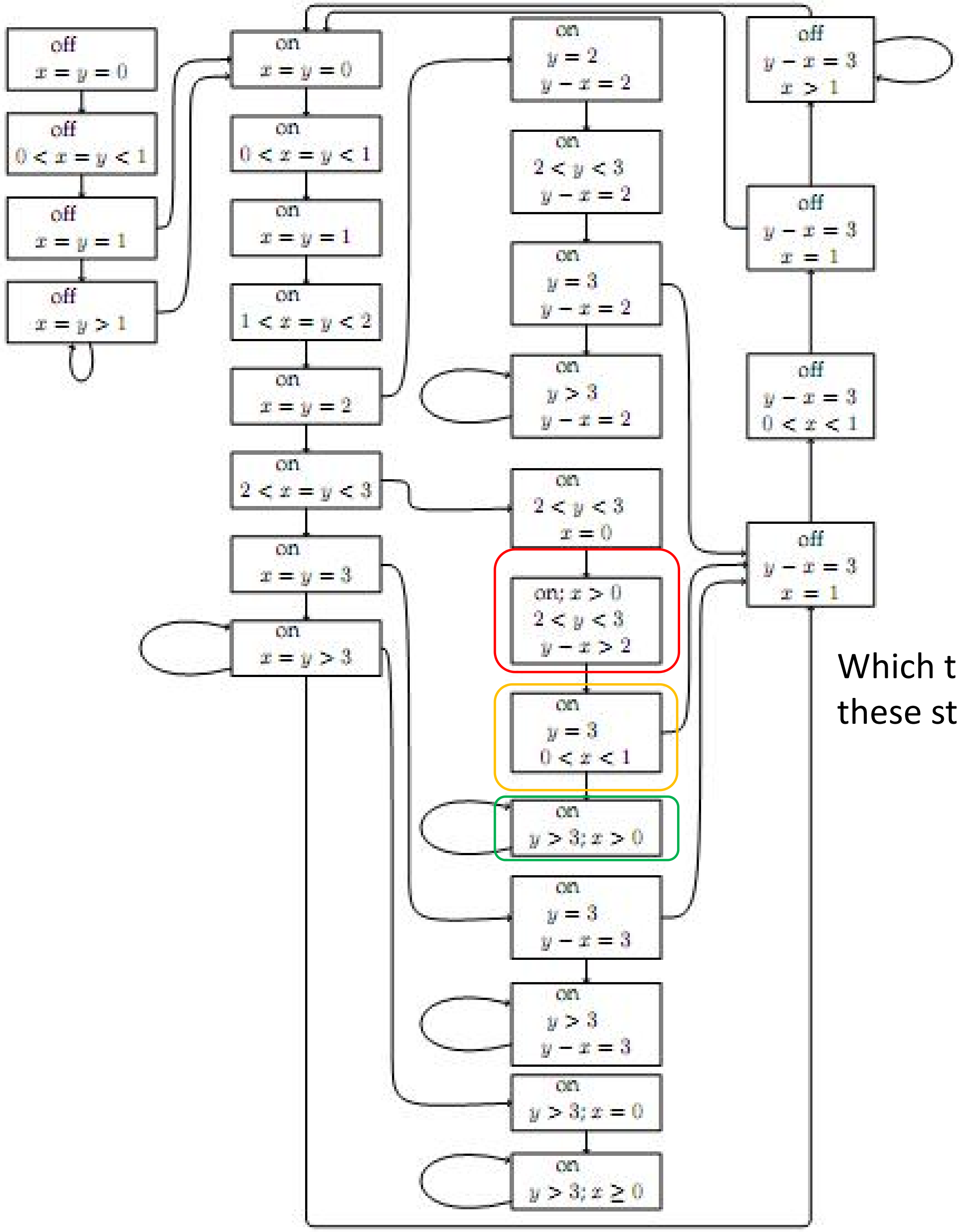


Clock Regions



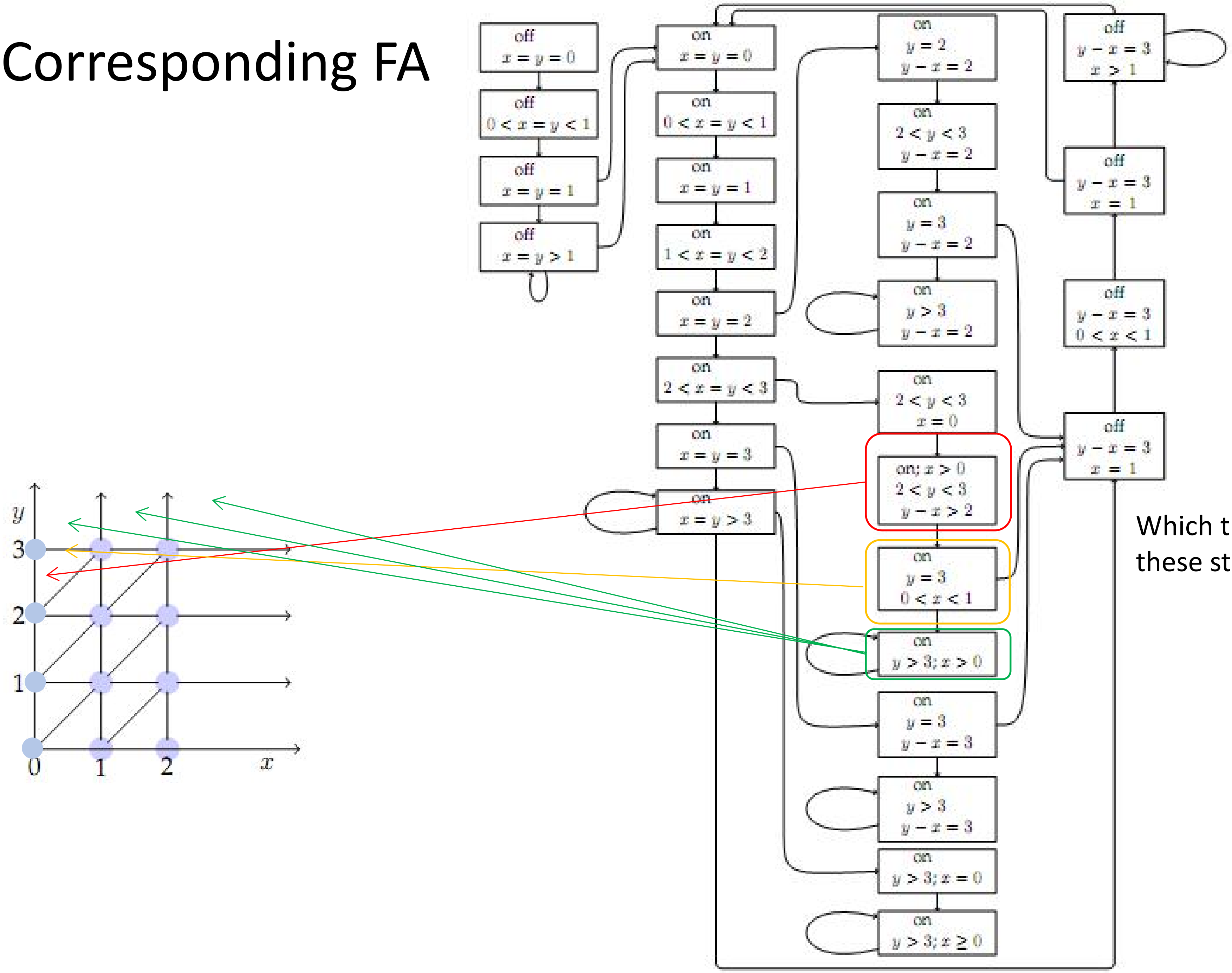
Corresponding FA





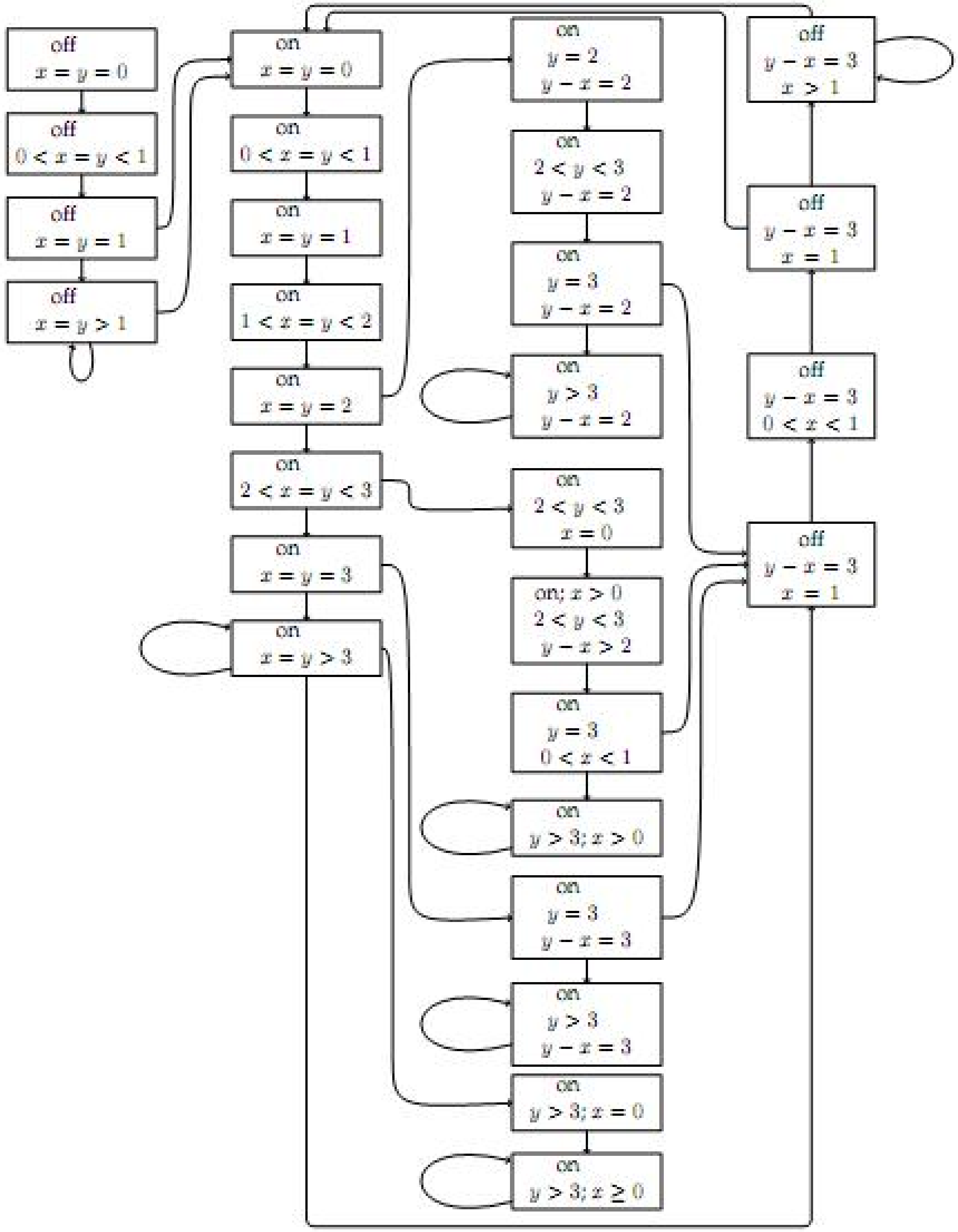
Which three clock regions do these state correspond to?

Corresponding FA



Which three clock regions do these state correspond to?

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
  - continuous variables re-initialized on each transition (no memory)
- Rectangular HA
  - continuous variables evolve in rectangular regions
- Linear HA
  - dynamics, invariants, and transitions are defined using linear constraints
- Nonlinear HA

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

  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

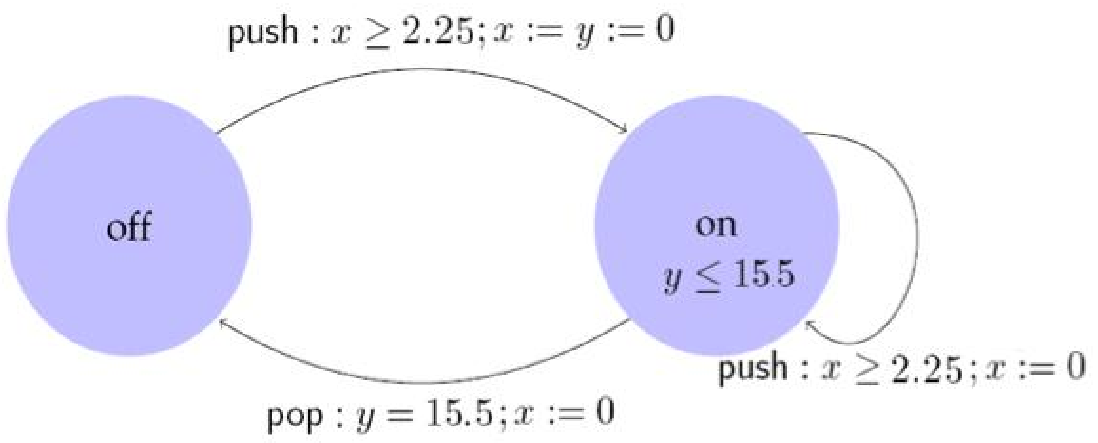
Definition. A ***rational timed automaton*** is a HA $\boldsymbol{\mathcal{A}}$ = $\langle V,\ \Theta,\ A,\ \mathcal{D},\ \mathcal{T}\rangle$ where

- V = X ∪ $\{loc\}$, where X is a set of n clocks and l is a discrete state variable of finite type L
- A is a finite set
- $\mathcal{D}$ is a set of transitions such that
  - The guards are described by **rational** clock constrainqs $\Phi(X)$
  - $\langle x, l\rangle - a \rightarrow \langle x', l'\rangle$ 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. Switch can be turn off 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; loc := off
   trajectories
     invariant loc = on ∨ loc = off
     stop when y = 15.5
     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 : $y = 15.5; x := 0$

Slides adapted from Prof. Sayan Mitra’s slides in Fall 2021

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

```
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
   evolve d(x) = 4; d(y) = 4
```

Slides adapted from Prof. Sayan Mitra’s slides in Fall 2021

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 L
  - 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 $x \in X\ \exists k \ such \ that \ \tau \in \mathcal{T},\ t \in \tau.dom$

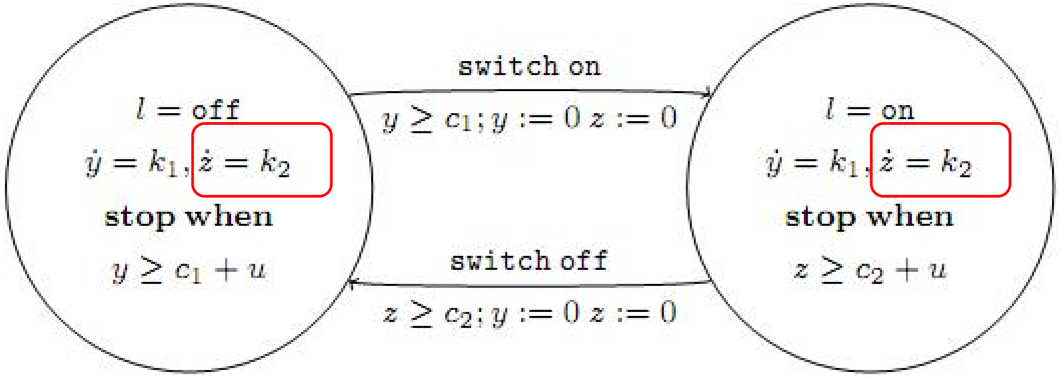
$$\tau(t).x = \tau(0).x + k\ t$$

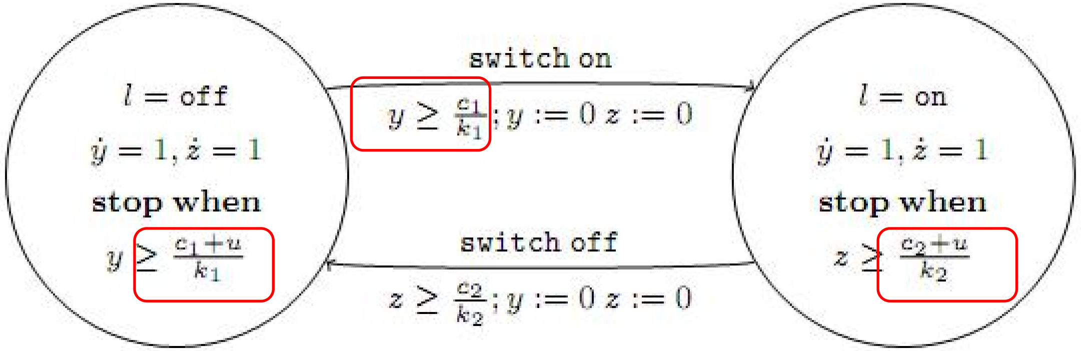
Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

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


$l = \text{off}$
$\dot{y} = k_1, \boxed{\dot{z} = k_2}$
stop when
$y \geq c_1 + u$
switch on
$y \geq c_1; y := 0\ z := 0$
switch off
$z \geq c_2; y := 0\ z := 0$
$l = \text{on}$
$\dot{y} = k_1, \boxed{\dot{z} = k_2}$
stop when
$z \geq c_2 + u$


$l = \text{off}$
$\dot{y} = 1, \dot{z} = 1$
stop when
$y \boxed{\geq \frac{c_1+u}{k_1}}$
switch on
$\boxed{y \geq \frac{c_1}{k_1}}; y := 0\ z := 0$
switch off
$z \geq \frac{c_2}{k_2}; y := 0\ z := 0$
$l = \text{on}$
$\dot{y} = 1, \dot{z} = 1$
stop when
$z \boxed{\geq \frac{c_2+u}{k_2}}$

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

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 L
- 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) $\tau(0)\lceil x + k_{\ell 1} t \leq \tau(t)\lceil x \leq \tau(0)\lceil x + k_{\ell 2} t$

d(x) can vary with time, as long as it is bounded

- $\mathcal{D}$ is a set of transitions such that
  - Guards are described by **rational** clock constrainqs
  - $\langle x, l \rangle \rightarrow_a \langle x', l' \rangle$ implies $x' = x$ or $x' \in [c_1, c_2]$

bounded in a rectangle

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

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

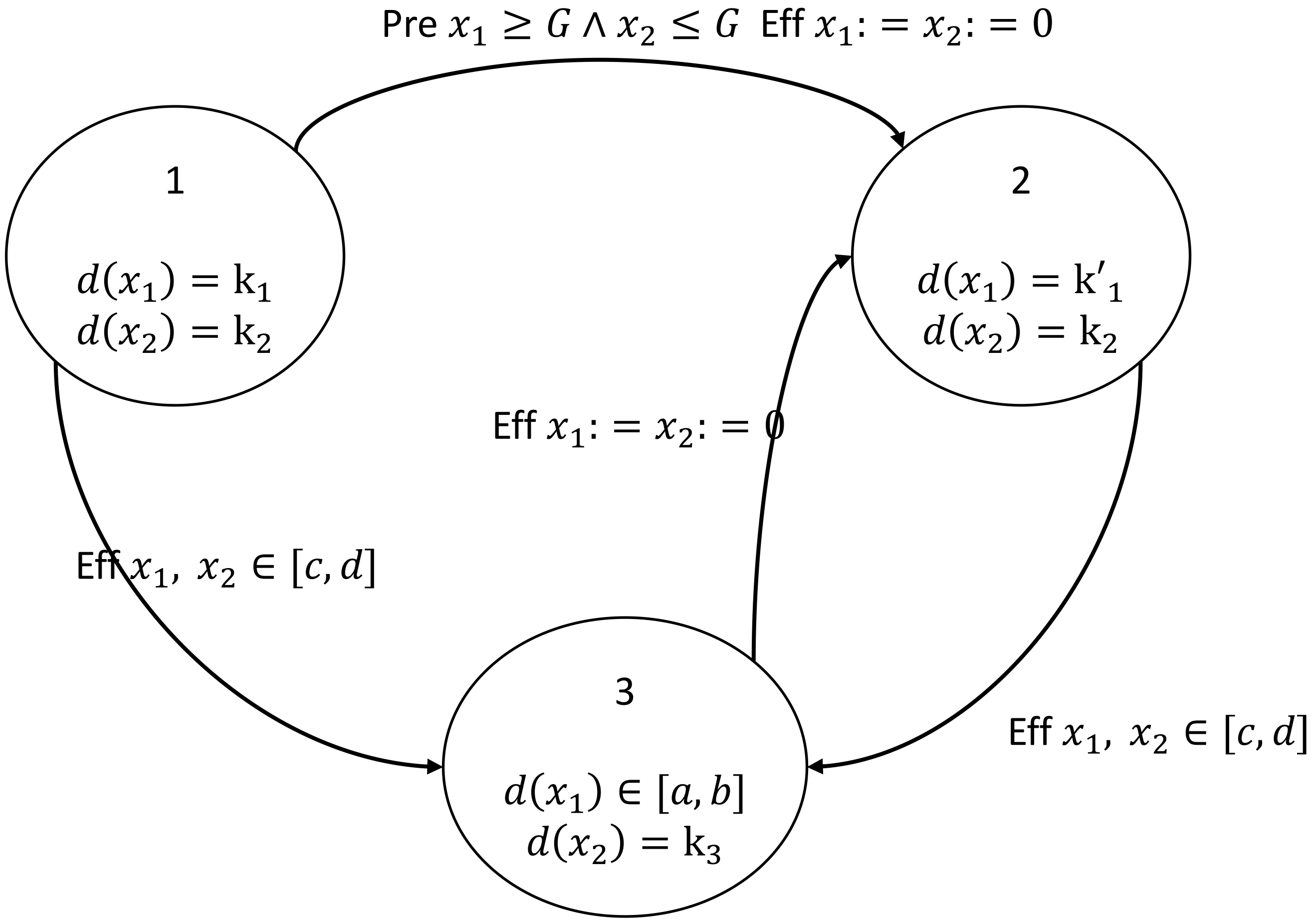
Definition. ***An initialized rectangular hybrid automaton*** **(IRHA)** is a RHA $\mathcal{A}$ where

- 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
- $\mathcal{T} = \cup_{\ell}\mathcal{T}_{\ell}$ 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 constrings
  - $\langle x, l\rangle \rightarrow_a \langle x', l'\rangle$ implies if dynamics changes from ℓ to another $\ell' \neq \ell$ then $x' \in [c_1, c_2]$ (**initialized, rather than keeping the old value**), otherwise $x' = x$ (keep the old value)

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Example: Initialized Rectangular HA



Both $x_1,\ x_2$ have to be reset on transaction to a different mode

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

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** multi-rate automaton that is bisimilar to the given IRHA
  - Construct a ITA that is bisimilar to the Singular TA

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