Decidability issues for timed models

an application of computer algebra techniques

Béatrice Bérard

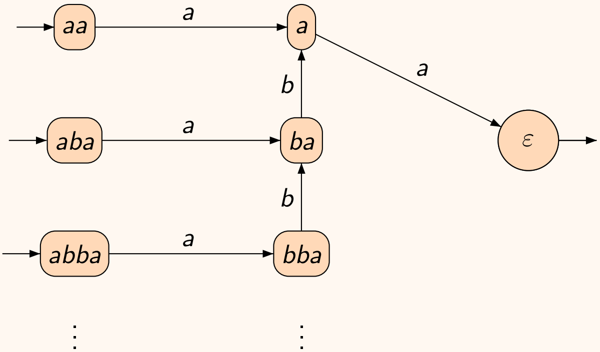
Université Pierre & Marie Curie, LIP6/MoVe, CNRS UMR 7606

Based on joint work with S. Haddad, C. Picaronny, M. Safey El Din, M. Sassolas

EJCIM, mars 2015

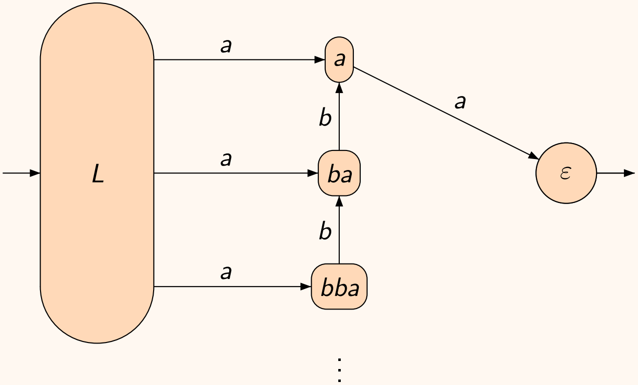
An infinite transition system

for the set of words $L = ab^*a = \{ab^na \mid n \in \mathbb{N}\}$
over alphabet $\Sigma = \{a, b\}$



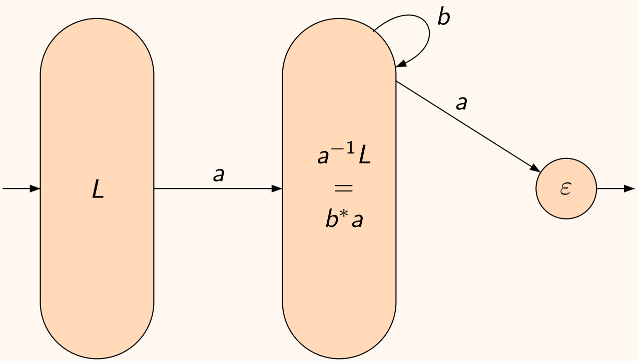
An infinite transition system

for the set of words $L = ab^*a = \{ab^n a \mid n \in \mathbb{N}\}$
over alphabet $\Sigma = \{a, b\}$



An infinite transition system

for the set of words $L = ab^*a = \{ab^na \mid n \in \mathbb{N}\}$
over alphabet $\Sigma = \{a, b\}$



... and its finite quotient

Quotients

Σ alphabet, Σ^* set of words over Σ, language : subset of Σ^*

For a language $M \subseteq \Sigma^*$ and a word $u \in \Sigma^*$

$$u^{-1}M = \{v \in \Sigma^* \mid uv \in M\}$$

$u^{-1}M$, also noted $M \setminus u$, is a quotient of M.

For the example $L = ab^*a$

$a^{-1}L = b^*a$ $\qquad$ $b^{-1}L = \emptyset = (bu)^{-1}L$ for any u

A partition of Σ^* is obtained by quotient under $\sim_L$:
$u_1 \sim_L u_2$ if $u_1^{-1}L = u_2^{-1}L$.

Quotients

Σ alphabet, Σ^* set of words over Σ, language : subset of Σ^*

For a language $M \subseteq \Sigma^*$ and a word $u \in \Sigma^*$

$$u^{-1}M = \{v \in \Sigma^* \mid uv \in M\}$$

$u^{-1}M$, also noted $M \setminus u$, is a quotient of M.

For the example $L = ab^*a$

$a^{-1}L = b^*a$ $\qquad$ $b^{-1}L = \emptyset = (bu)^{-1}L$ for any u

A partition of Σ^* is obtained by quotient under $\sim_L$:
$u_1 \sim_L u_2$ if $u_1^{-1}L = u_2^{-1}L$.

[Nerode, 1958]

A language is accepted by a finite automaton if and only if it has a finite number of quotients.

Quotients and finite automata

States = quotients, with transitions:

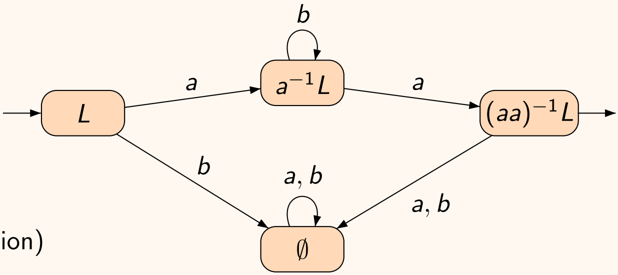
$$u^{-1}L \xrightarrow{a} (ua)^{-1}L$$

initial state: $L = \varepsilon^{-1}L$

final states : those containing ε

$L = ab^*a$

$a^{-1}L = b^*a$ and $b^{-1}L = \emptyset$
$(ab)^{-1}L = b^{-1}(a^{-1}L) = b^{-1}(b^*a) = b^*a = a^{-1}L$
$(aa)^{-1}L = a^{-1}(b^*a) = \{\varepsilon\}$
$a^{-1}\{\varepsilon\} = b^{-1}\{\varepsilon\} = \emptyset$



(the completed version)

Quotients for infinite transition systems

or the reductionist approach [Henzinger, Majumdar, Raskin, 2003]

A transition system

$\mathcal{T} = (S, E)$ with

- S set of configurations
- $E \subseteq S \times S$ set of transitions

An equivalence $\sim$ over S producing a quotient

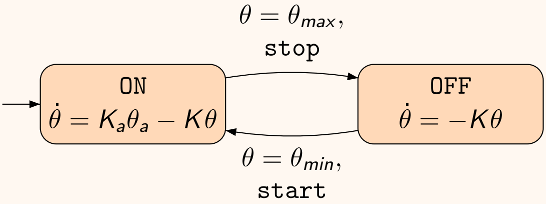
$\mathcal{T}_\sim = (S/\sim, E_\sim)$ with

- $S/\sim$ set of equivalence classes
- $E_\sim \subseteq Q/\sim \times Q/\sim$
  such that $P \to P'$ if $q \to q'$ in E for some $q \in P$ and $q' \in P'$

Adding propositions on states or labels on transitions,
Goal: build finite quotients preserving specific classes of properties like accepted language, reachability, LTL, CTL or μ-calculus model checking, ...

Hybrid automata

A heating device controller

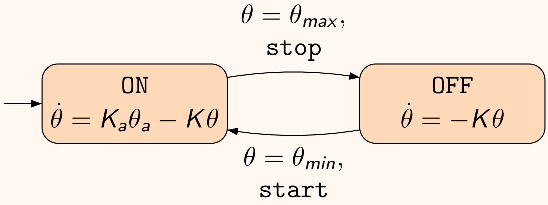


Configurations in S: $(q, v(\theta))$, with $q \in \{\mathtt{ON}, \mathtt{OFF}\}$ and $v(\theta)$ the temperature value.
Evolution: continuous for θ in a fixed q (following the differential equation),
discrete when firing a transition.

With n real variables, flows and invariants on control states Q, guards and updates on transitions, configurations : $Q \times \mathbb{R}^n$.

Hybrid automata

A heating device controller



Configurations in S: $(q, v(\theta))$, with $q \in \{\mathtt{ON}, \mathtt{OFF}\}$ and $v(\theta)$ the temperature value.
Evolution: continuous for θ in a fixed q (following the differential equation),
discrete when firing a transition.

With n real variables, flows and invariants on control states Q, guards and updates on transitions, configurations : $Q \times \mathbb{R}^n$.

Verification problems are mostly undecidable

Decidability requires restricting either the flows [Henzinger, Kopke, Puri Varayia, 1998] or the jumps [Alur, Henzinger, Lafferrière, Pappas, 2000] for flows $\dot{x} = Ax$

Outline

Timed Automata

Interrupt Timed Automata

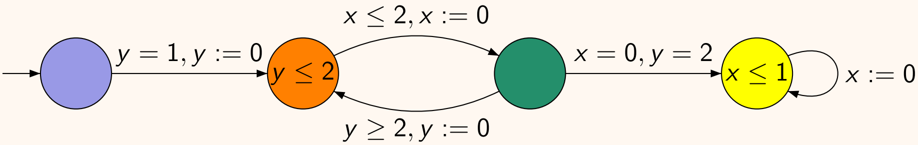
Using Cylindrical Decomposition

Timed automata

Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

Ex. 1

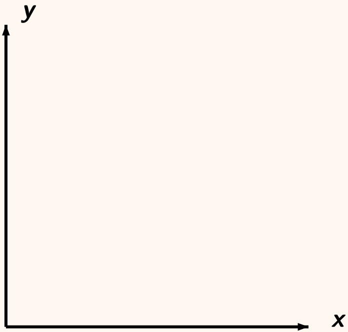


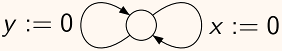
Timed automata

Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

Ex. 2: A geometric view of a trajectory





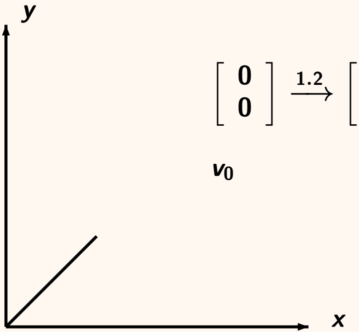
Timed automata

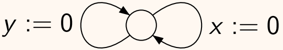
Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

Ex. 2: A geometric view of a trajectory

$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{1.2} \begin{bmatrix} \mathbf{1.2} \\ \mathbf{1.2} \end{bmatrix}$$





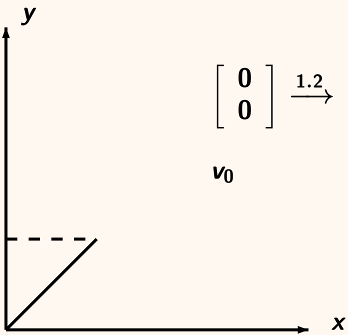
Timed automata

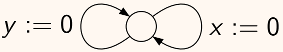
Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

Ex. 2: A geometric view of a trajectory

$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1.2}} \begin{bmatrix} \mathbf{1.2} \\ \mathbf{1.2} \end{bmatrix} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{1.2} \end{bmatrix}$$



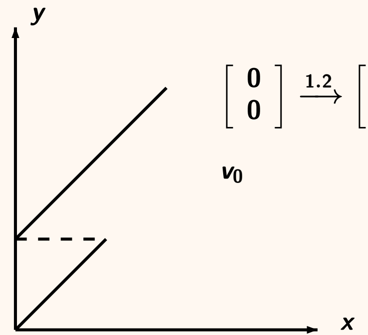


Timed automata

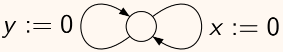
Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

Ex. 2: A geometric view of a trajectory



$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{1.2} \begin{bmatrix} \mathbf{1.2} \\ \mathbf{1.2} \end{bmatrix} \xrightarrow{x:=0} \begin{bmatrix} \mathbf{0} \\ \mathbf{1.2} \end{bmatrix} \xrightarrow{2} \begin{bmatrix} \mathbf{2} \\ \mathbf{3.2} \end{bmatrix}$$

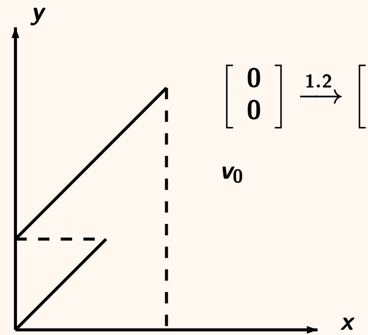


Timed automata

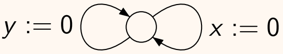
Variables: clocks with flow $\dot{x} = 1$ for each $x \in X$
Guards: conjunctions of $x - c \bowtie 0$, with $c \in \mathbb{Q}$ and $\bowtie$ in $\{<, \leq, =, \geq, >\}$
Updates: conjunctions of reset $x := 0$
Clock valuation: $v = (v(x_1), \ldots, v(x_n)) \in \mathbb{R}_+^n$ if $X = \{x_1, \ldots, x_n\}$

Examples (with two clocks x and y)

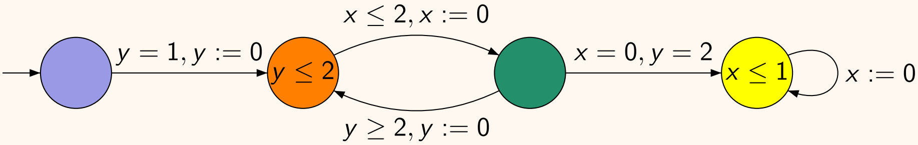
Ex. 2: A geometric view of a trajectory



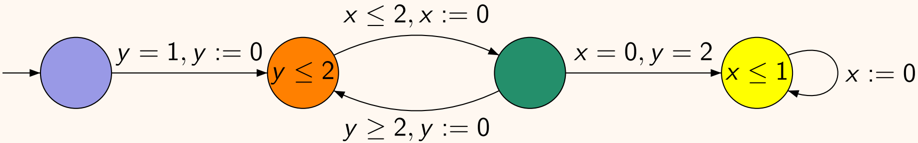
$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{1.2} \begin{bmatrix} \mathbf{1.2} \\ \mathbf{1.2} \end{bmatrix} \xrightarrow{x:=0} \begin{bmatrix} \mathbf{0} \\ \mathbf{1.2} \end{bmatrix} \xrightarrow{2} \begin{bmatrix} \mathbf{2} \\ \mathbf{3.2} \end{bmatrix} \xrightarrow{y:=0} \begin{bmatrix} \mathbf{2} \\ \mathbf{0} \end{bmatrix}$$

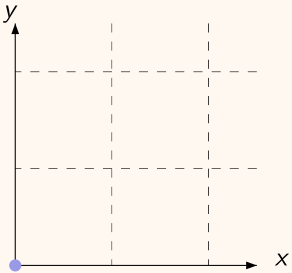


Zones for timed automata



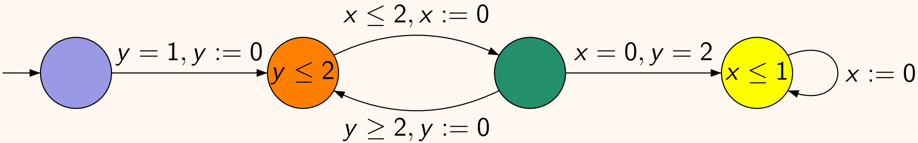
Zones for timed automata

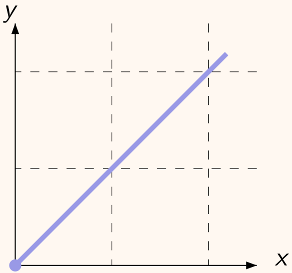




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

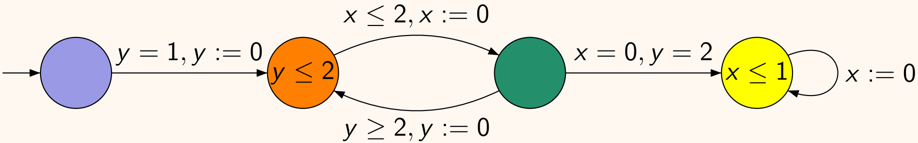
Zones for timed automata

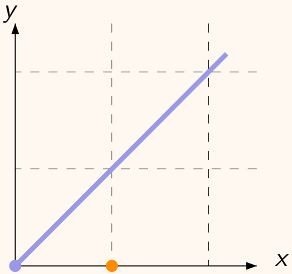




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{1} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix}$$

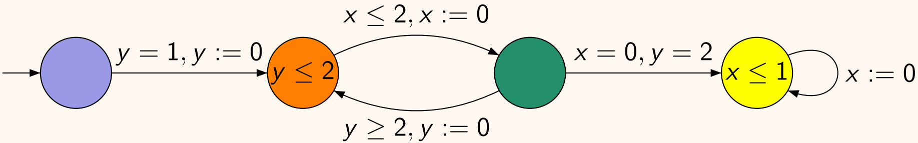
Zones for timed automata

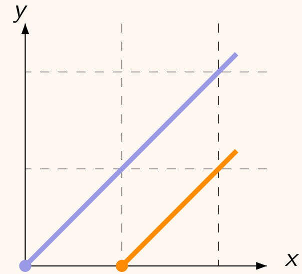




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{y:=\mathbf{0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}$$

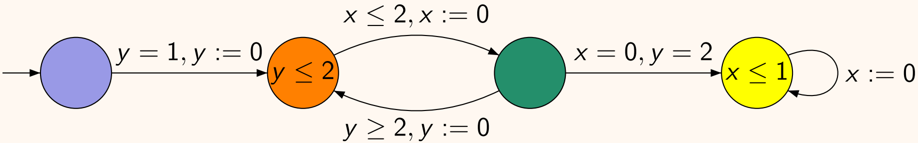
Zones for timed automata

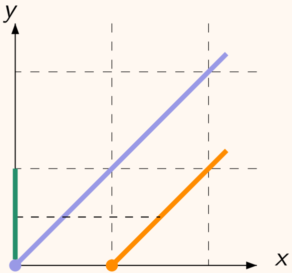




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{y:=\mathbf{0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix}$$

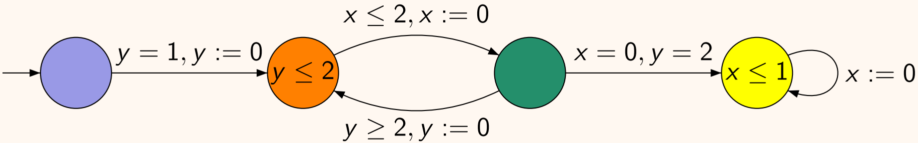
Zones for timed automata

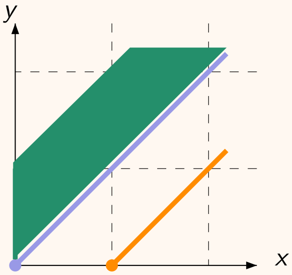




$$\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix}\mathbf{1}\\ \mathbf{1}\end{bmatrix} \xrightarrow{y:=\mathbf{0}} \begin{bmatrix}\mathbf{1}\\ \mathbf{0}\end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix}\mathbf{1.5}\\ \mathbf{0.5}\end{bmatrix} \xrightarrow{x:=\mathbf{0}} \begin{bmatrix}\mathbf{0}\\ \mathbf{0.5}\end{bmatrix} \cdots$$

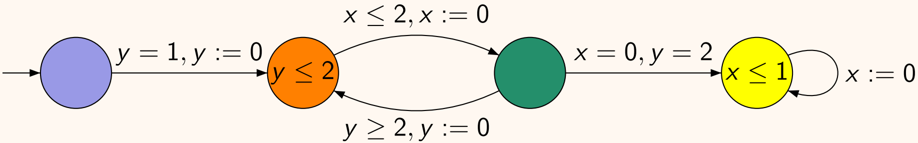
Zones for timed automata

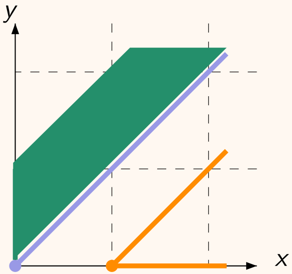




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\mathbf{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\mathbf{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \cdots$$

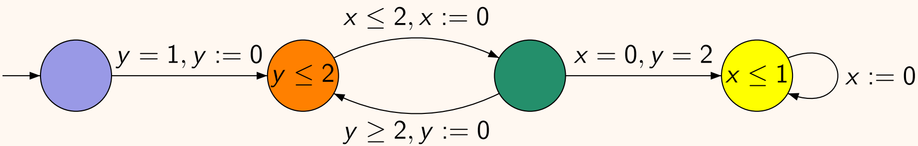
Zones for timed automata

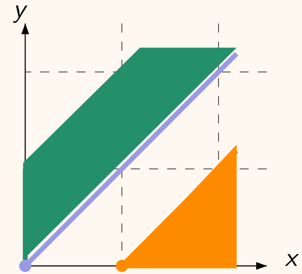




$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}^{\bullet} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix}^{\bullet} \xrightarrow{\boldsymbol{y:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}^{\bullet} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix}^{\bullet} \xrightarrow{\boldsymbol{x:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix}^{\bullet} \cdots$$

Zones for timed automata





$$\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{1}} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix} \xrightarrow{\boldsymbol{y}\mathbf{:=0}} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix} \xrightarrow{\mathbf{0.5}} \begin{bmatrix} \mathbf{1.5} \\ \mathbf{0.5} \end{bmatrix} \xrightarrow{\boldsymbol{x}\mathbf{:=0}} \begin{bmatrix} \mathbf{0} \\ \mathbf{0.5} \end{bmatrix} \cdots$$

A finite quotient for timed automata

[Alur, Dill, 1990]

From $\mathcal{A}$, build a finite automaton $Reg(\mathcal{A})$ preserving reachability of a control state and accepting the untimed part of the language (with labels).

Transition system $\mathcal{T}_{\mathcal{A}}$

with clocks $X = \{x_1, \ldots, x_n\}$, set of control states Q, set of transitions E:

- configurations $S = Q \times \mathbb{R}_+^n$
- time steps $(q, v) \xrightarrow{d} (q, v + d)$
- discrete steps $(q, v) \xrightarrow{e} (q', v')$ for a transition $e = q \xrightarrow{g,u} q'$ in E if clock values v satisfy the guard g and $v' = v[u]$

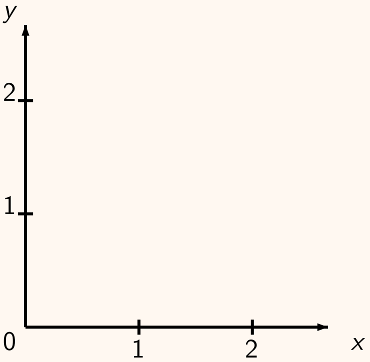
Equivalence $\sim$ over $\mathbb{R}_+^n$ producing a quotient $Reg(\mathcal{A})$

- $Q \times \mathcal{R}$, for a set $\mathcal{R}$ of **regions** partitioning $\mathbb{R}_+^n$,
- abstract time steps $(q, R) \rightarrow (q, succ(R))$
- discrete steps $(q, R) \xrightarrow{e} (q', R')$

both steps consistent with $\sim$

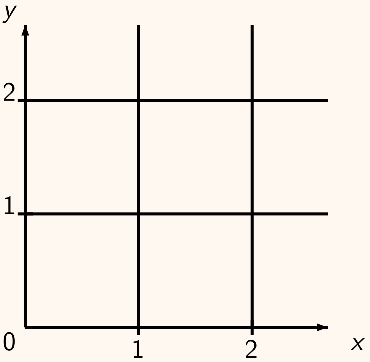
Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



Quotient construction

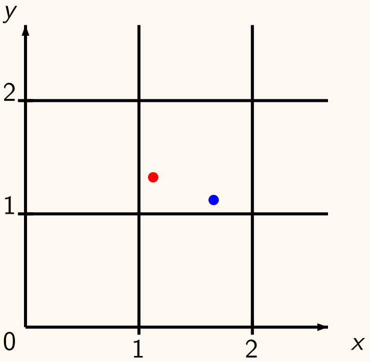
A geometric view with two clocks x and y, maximal constant $m = 2$



- Equivalent valuations must be consistent with constraints $x \bowtie k$

Quotient construction

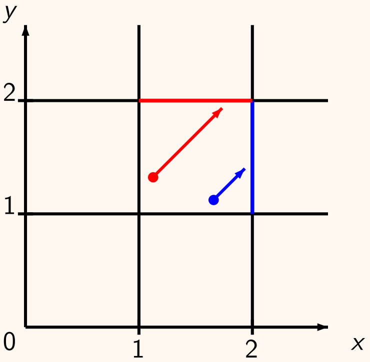
A geometric view with two clocks x and y, maximal constant $m = 2$



- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

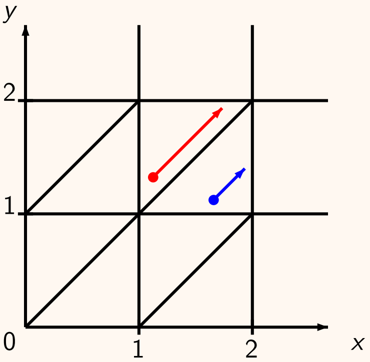
A geometric view with two clocks x and y, maximal constant $m = 2$



- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

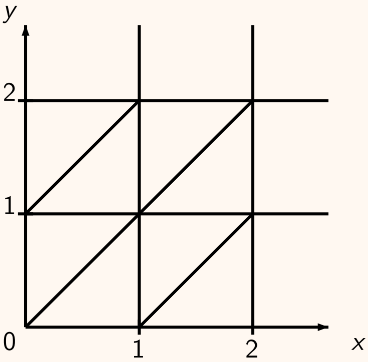
A geometric view with two clocks x and y, maximal constant $m = 2$



- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

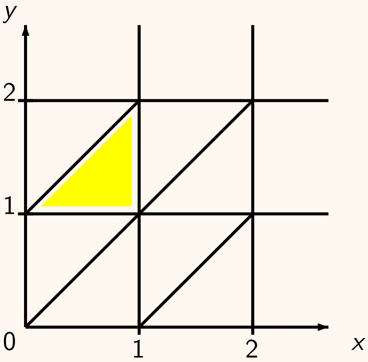
A geometric view with two clocks x and y, maximal constant $m = 2$



- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$

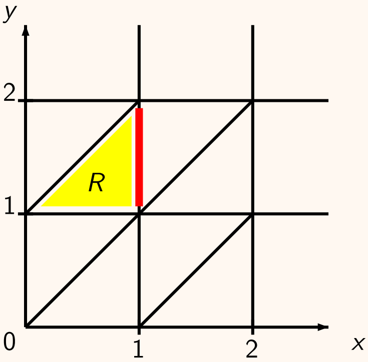


region R defined by
$0 < x < 1$ and $1 < y < 2$
and $\mathit{frac}(x) > \mathit{frac}(y)$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



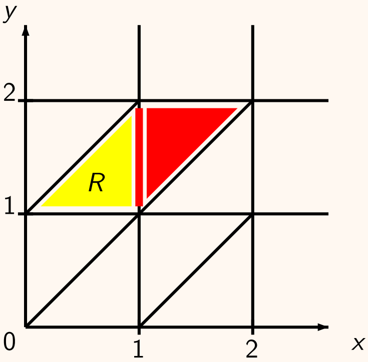
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



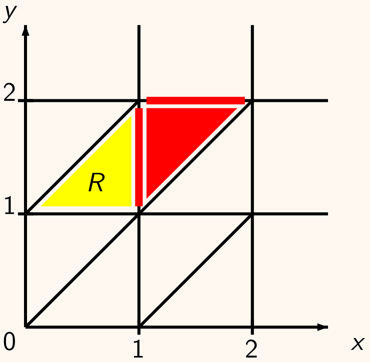
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



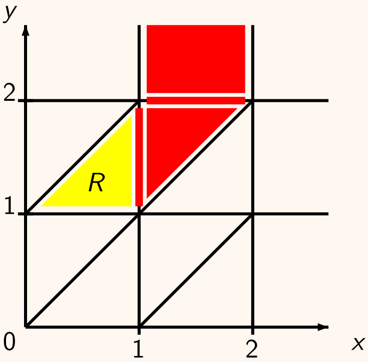
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



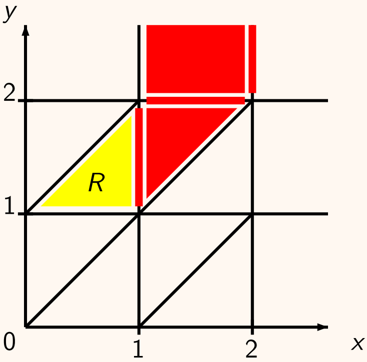
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



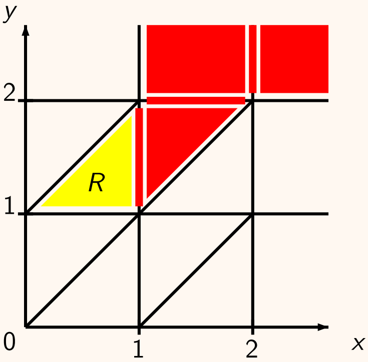
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



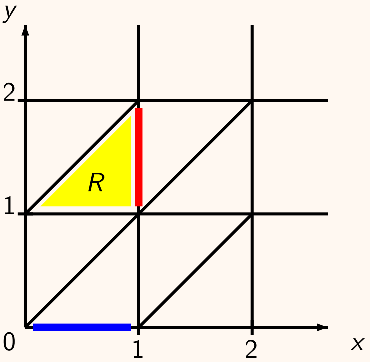
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Quotient construction

A geometric view with two clocks x and y, maximal constant $m = 2$



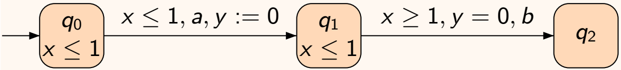
region R defined by
$0 < x < 1$ and $1 < y < 2$
and $frac(x) > frac(y)$

Time successor of R
$x = 1$ and $1 < y < 2$

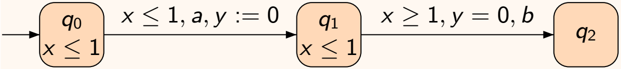
Discrete step from R
with $y := 0$
$0 < x < 1$ and $y = 0$

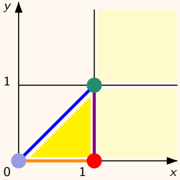
- Equivalent valuations must be consistent with constraints $x \bowtie k$
- Equivalent valuations must be consistent with time elapsing

Example of quotient

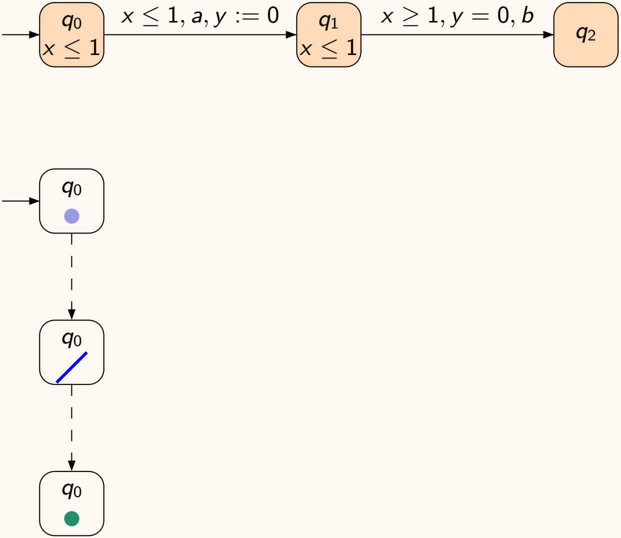


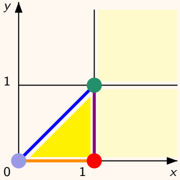
Example of quotient



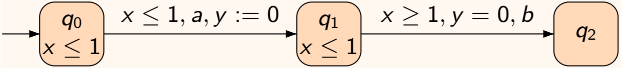


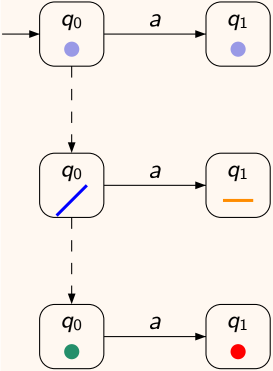
Example of quotient

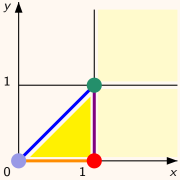




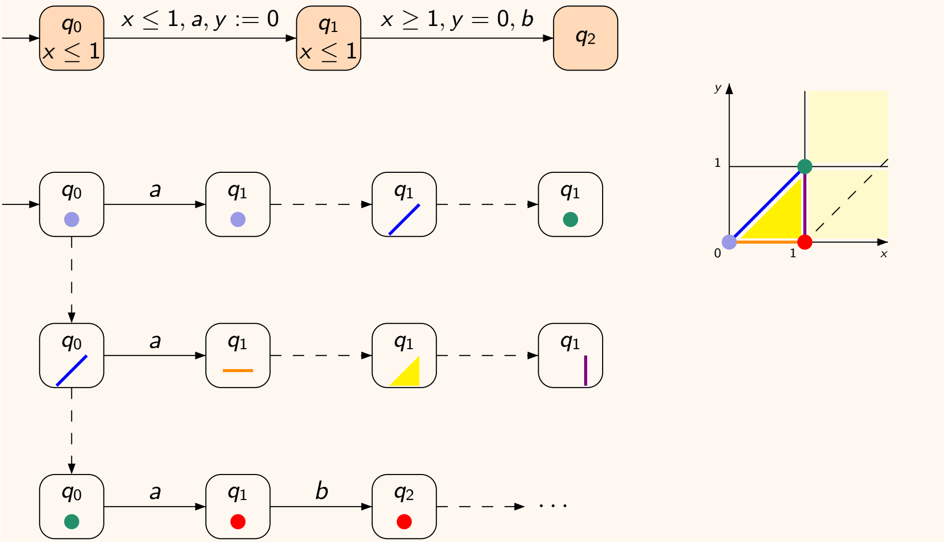
Example of quotient



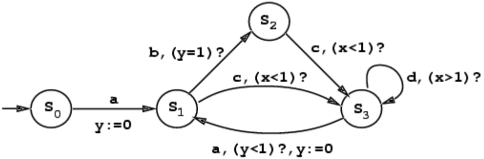


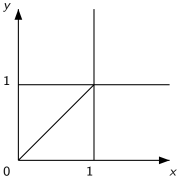


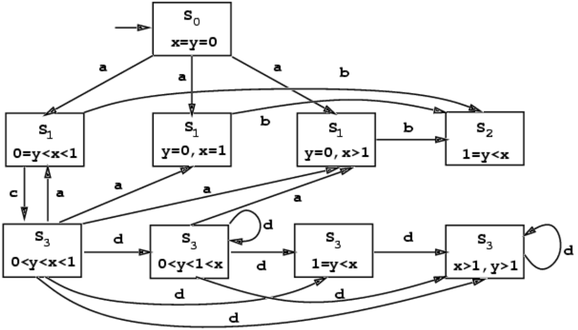
Example of quotient



Exemple from [Alur et Dill, 1990]

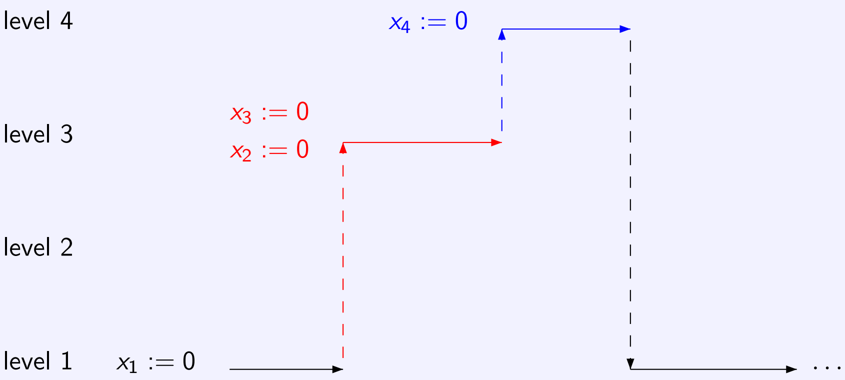






Interrupt Timed Automata (ITA)

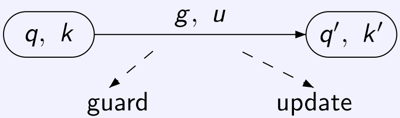
Control states on levels $\{1, \ldots, n\}$, a single clock x_k active on level k

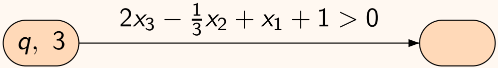


$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{1.5} \begin{bmatrix} 1.5 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{2.1} \begin{bmatrix} 1.5 \\ 0 \\ 2.1 \\ 0 \end{bmatrix} \xrightarrow{1.7} \begin{bmatrix} 1.5 \\ 0 \\ 2.1 \\ 1.7 \end{bmatrix} \xrightarrow{2.2} \begin{bmatrix} 3.7 \\ 0 \\ 2.1 \\ 1.7 \end{bmatrix}$$

ITA: syntax

- Variables: stopwatches with flow $\dot{x} = 1$ or $\dot{x} = 0$,
  clock x_k active at level $k \in \{1, \dots, n\}$
- Guards: conjunctions of linear constraints with rational coefficients
  $\sum_{j=1}^{k} a_j x_j + b \bowtie 0$ at level k, with $\bowtie$ in $\{<, \leq, =, \geq, >\}$
- Clock valuation: $v = (v(x_1), \dots, v(x_n)) \in \mathbb{R}^n$
- $\lambda : Q \to \{1, \dots, n\}$ state level, with $x_{\lambda(q)}$ the active clock in state q
- Transitions:





ITA: updates

From level k to k'

increasing level $k \leq k'$

Level higher than k': unchanged
Level from $k+1$ to k': reset
Level $i \leq k$: unchanged or linear update $x_i := \sum_{j<i} a_j x_j + b$.

ITA: updates

From level k to k'

increasing level $k \leq k'$

Level higher than k': unchanged
Level from $k+1$ to k': reset
Level $i \leq k$: unchanged or linear update $x_i := \sum_{j<i} a_j x_j + b$.

Example

$$\boxed{q_1,\ 2} \xrightarrow{\substack{x_1 := 1 \\ x_2 > 2x_1,\quad x_2 := 2x_1 \\ (x_3 := 0, x_4 := 0)}} \boxed{q_2,\ 4}$$

ITA: updates

From level k to k'

increasing level $k \leq k'$

Level higher than k': unchanged
Level from $k+1$ to k': reset
Level $i \leq k$: unchanged or linear update $x_i := \sum_{j<i} a_j x_j + b$.

Example

$$
\begin{tikzcd}[column sep=huge]
(q_1,\ 2) \arrow[r, "\begin{array}{l} x_2 > 2x_1,\ \begin{array}{l} x_1 := 1 \\ x_2 := 2x_1 \\ (x_3 := 0, x_4 := 0) \end{array} \end{array}"] & (q_2,\ 4) \arrow[r, "\begin{array}{l} x_4 = 3x_1 + x_2,\ \begin{array}{l} x_1 := 0 \\ x_2 := x_1 + 1, \\ x_3 := 2x_2 \end{array} \end{array}"] & (q_3,\ 3)
\end{tikzcd}
$$

Decreasing level

Level higher than k': unchanged
Otherwise: linear update $x_i := \sum_{j<i} a_j x_j + b$.

In a state at level k, clocks from higher levels are irrelevant.

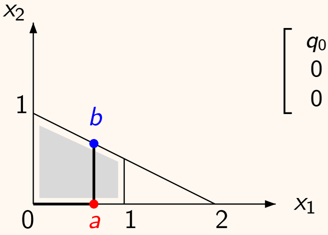
ITA: semantics

A transition system $\mathcal{T}_\mathcal{A}$

- configurations $S = Q \times \mathbb{R}^n$
- time steps from q at level k: only x_k is active, $(q, v) \xrightarrow{d} (q, v +_k d)$, with all clocks in $v +_k d$ unchanged except $(v +_k d)(x_k) = v(x_k) + d$
- discrete steps $(q, v) \xrightarrow{e} (q', v')$ for a transition $e : q \xrightarrow{g,u} q'$ if v satisfies the guard g and $v' = v[u]$.

Example: trajectories

$$\to (q_0, 1) \xrightarrow{x_1 < 1,\ a,\ (x_2 := 0)} (q_1, 2) \xrightarrow{x_1 + 2x_2 = 2,\ b} (q_2, 2) \to$$



$$\begin{bmatrix} q_0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{0.6} \begin{bmatrix} q_0 \\ 0.6 \\ 0 \end{bmatrix} \xrightarrow{a} \begin{bmatrix} q_1 \\ 0.6 \\ 0 \end{bmatrix} \xrightarrow{0.7} \begin{bmatrix} q_1 \\ 0.6 \\ 0.7 \end{bmatrix} \xrightarrow{b} \begin{bmatrix} q_2 \\ 0.6 \\ 0.7 \end{bmatrix}$$

grey zone for state q_1:

$0 < x_1 < 1$ and $0 < x_2 < -\frac{1}{2}x_1 + 1$

A finite quotient for ITA

[BH 2009]

From $\mathcal{A}$, build a finite automaton $Reg(\mathcal{A})$ preserving reachability of a control state and accepting the untimed part of the language.

A finite quotient for ITA

[BH 2009]

From $\mathcal{A}$, build a finite automaton $Reg(\mathcal{A})$ preserving reachability of a control state and accepting the untimed part of the language.

Principle - 1

Build sets of linear expressions E_k for each level k, starting from $\{0, x_k\}$ iteratively downward:

- adding the *complements* of x_k in guards from level k,
- saturating E_k by applying updates of appropriate transitions to expressions of E_k,
- saturating E_j $(j < k)$ by applying updates of appropriate transitions to differences of expressions of E_k.

$$\longrightarrow \boxed{q_0, 1} \xrightarrow{x_1 < 1,\ a,\ (x_2 := 0)} \boxed{q_1, 2} \xrightarrow{x_1 + 2x_2 = 2,\ b} \boxed{q_2, 2} \longrightarrow$$

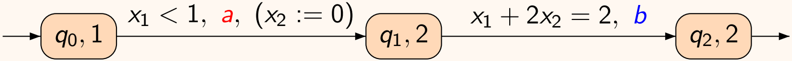
Starting from $E_2 = \{0, x_2\}$ and $E_1 = \{0, x_1\}$, first add $-\frac{1}{2}x_1 + 1$ to E_2 and 2 to E_1. Then add 1 to E_1.

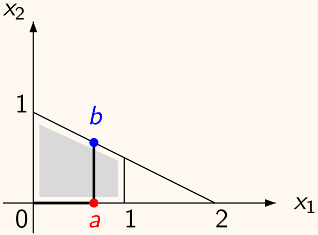
A finite quotient for ITA

Principle - 2

Two valuations are equivalent in state q at level k if they produce the same preorders for linear expressions in each E_i, $i \leq k$.

- a class is a pair $C = (q, \{\preceq_k\}_{k \leq \lambda(q)})$ where $\preceq_k$ is a total preorder on E_k
- abstract time steps $(q, R) \to (q, succ(R))$ and discrete steps $(q, R) \xrightarrow{a} (q', R')$ consistent with preorders.





Level 1: $E_1 = \{x_1, 0, 1, 2\}$

Initial class $C_0 = (q_0, x_1 = 0 < 1 < 2) = (q_0, R_0)$

$succ(C_0) = C_0^1 = (q_0, 0 < x_1 < 1 < 2) = (q_0, R_0^1)$

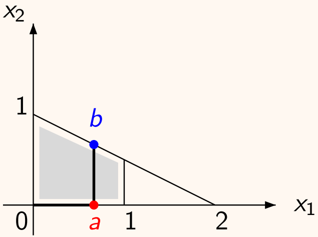
$succ(C_0^1) = C_0^2 = (q_0, 0 < x_1 = 1 < 2)$

$\ldots C_0^5 = (q_0, 0 < 1 < 2 < x_1)$

Discrete transitions a from C_0 and C_0^1

Example (cont.)

$$\longrightarrow \boxed{q_0, 1} \xrightarrow{x_1 < 1,\ a,\ (x_2 := 0)} \boxed{q_1, 2} \xrightarrow{x_1 + 2x_2 = 2,\ b} \boxed{q_2, 2} \longrightarrow$$



Level 2: $E_2 = \{x_2, 0, -\frac{1}{2}x_1 + 1\}$

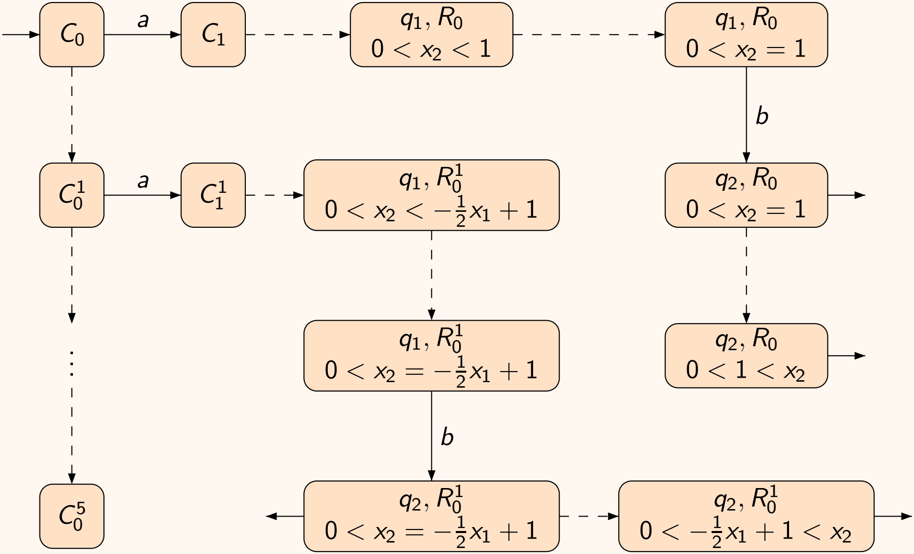
$C_0 \xrightarrow{a} C_1 = (q_1, R_0, x_2 = 0 < \frac{1}{2})$ with $x_1 = 0$

$C_0^1 \xrightarrow{a} C_1^1 = (q_1, R_0^1, x_2 = 0 < -\frac{1}{2}x_1 + 1)$

with $0 < x_1 < 1$

Discrete transitions b : from classes such that $x_2 = -\frac{1}{2}x_1 + 1$.

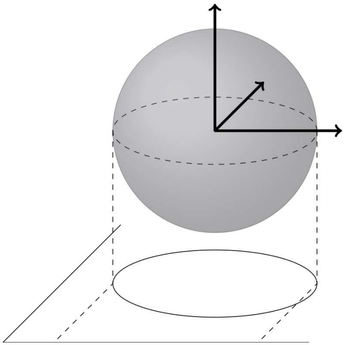
Example: class automaton



Cylindrical decomposition

Example for polynomial $P_3 = X_1^2 + X_2^2 + X_3^2 - 1$

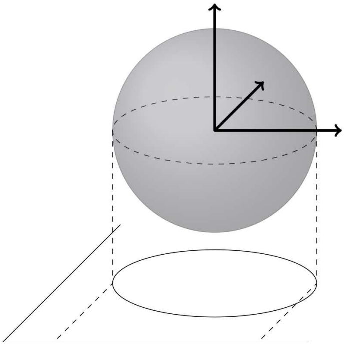
- Elimination phase produces the polynomials $P_2 = X_1^2 + X_2^2 - 1$ and $P_1 = X_1^2 - 1$
- Lifting phase produces partitions of $\mathbb{R}$, $\mathbb{R}^2$ and $\mathbb{R}^3$ organized in a tree of cells where the signs of these polynomials (in $\{-1, 0, 1\}$) are constant.



Cylindrical decomposition

Example for polynomial $P_3 = X_1^2 + X_2^2 + X_3^2 - 1$

- Elimination phase produces the polynomials $P_2 = X_1^2 + X_2^2 - 1$ and $P_1 = X_1^2 - 1$
- Lifting phase produces partitions of $\mathbb{R}$, $\mathbb{R}^2$ and $\mathbb{R}^3$ organized in a tree of cells where the signs of these polynomials (in $\{-1, 0, 1\}$) are constant.



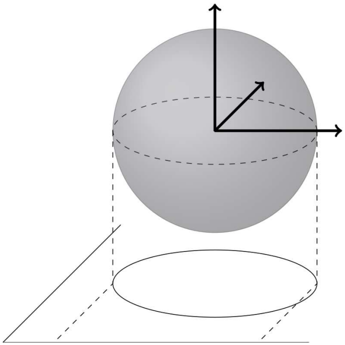
Level 1 : partition of $\mathbb{R}$ in 5 cells

$C_{-\infty} =]-\infty, -1[$, $C_{-1} = \{-1\}$, $C_0 =]-1, 1[$, $C_1 = \{1\}$, $C_{+\infty} =]1, +\infty[$

Cylindrical decomposition

Example for polynomial $P_3 = X_1^2 + X_2^2 + X_3^2 - 1$

- Elimination phase produces the polynomials $P_2 = X_1^2 + X_2^2 - 1$ and $P_1 = X_1^2 - 1$
- Lifting phase produces partitions of $\mathbb{R}$, $\mathbb{R}^2$ and $\mathbb{R}^3$ organized in a tree of cells where the signs of these polynomials (in $\{-1, 0, 1\}$) are constant.



Level 2 : partition of $\mathbb{R}^2$

Above $C_{-\infty}$: a single cell $C_{-\infty} \times \mathbb{R}$

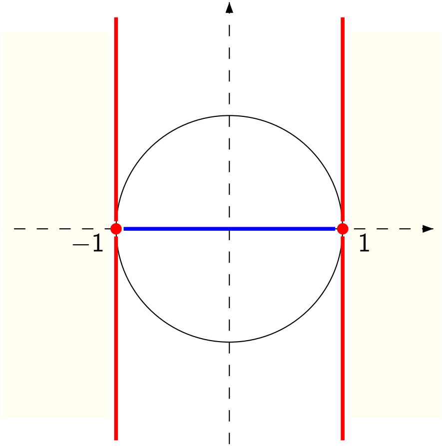
Above C_{-1}: three cells

$\{-1\}\times]-\infty, 0[$, $\{(-1, 0)\}$, $\{-1\}\times]0, +\infty[$

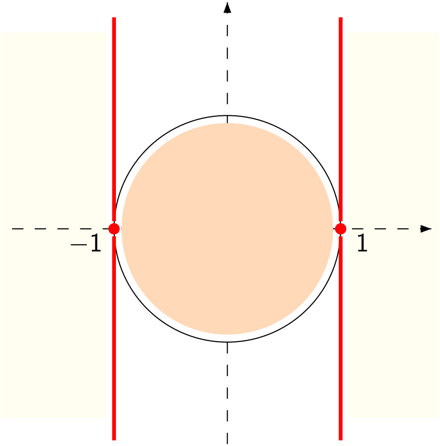
Level 1 : partition of $\mathbb{R}$ in 5 cells

$C_{-\infty} =]-\infty, -1[$, $C_{-1} = \{-1\}$, $C_0 =]-1, 1[$, $C_1 = \{1\}$, $C_{+\infty} =]1, +\infty[$

Level 2 above C_0

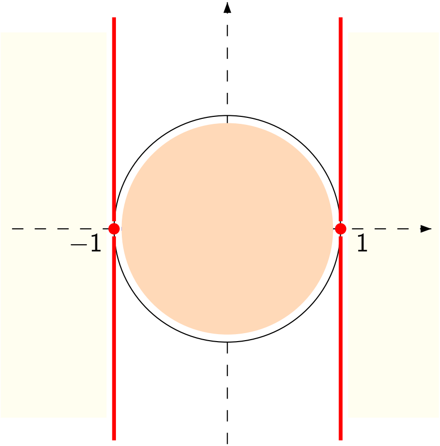


Level 2 above C_0



$$\boxed{C_{0,0}} \quad \begin{cases} -1 < x_1 < 1 \\ -\sqrt{1-x_1^2} < x_2 < \sqrt{1-x_1^2} \end{cases}$$

Level 2 above C_0

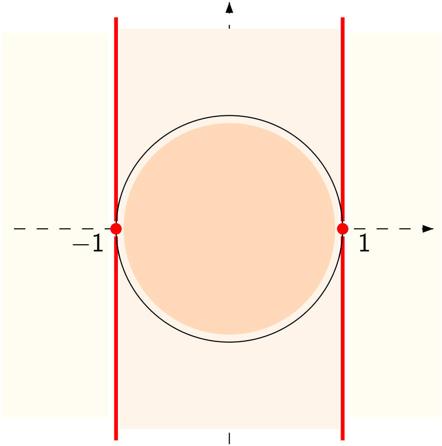


$C_{0,1}$ $\left\{ \begin{array}{l} -1 < x_1 < 1 \\ x_2 = \sqrt{1 - x_1^2} \end{array} \right.$

$\boxed{C_{0,0}}$ $\left\{ \begin{array}{l} -1 < x_1 < 1 \\ -\sqrt{1 - x_1^2} < x_2 < \sqrt{1 - x_1^2} \end{array} \right.$

$C_{0,-1}$ $\left\{ \begin{array}{l} -1 < x_1 < 1 \\ x_2 = -\sqrt{1 - x_1^2} \end{array} \right.$

Level 2 above C_0



$\boxed{C_{0,+\infty}}$ $\begin{cases} -1 < x_1 < 1 \\ x_2 > \sqrt{1-x_1^2} \end{cases}$

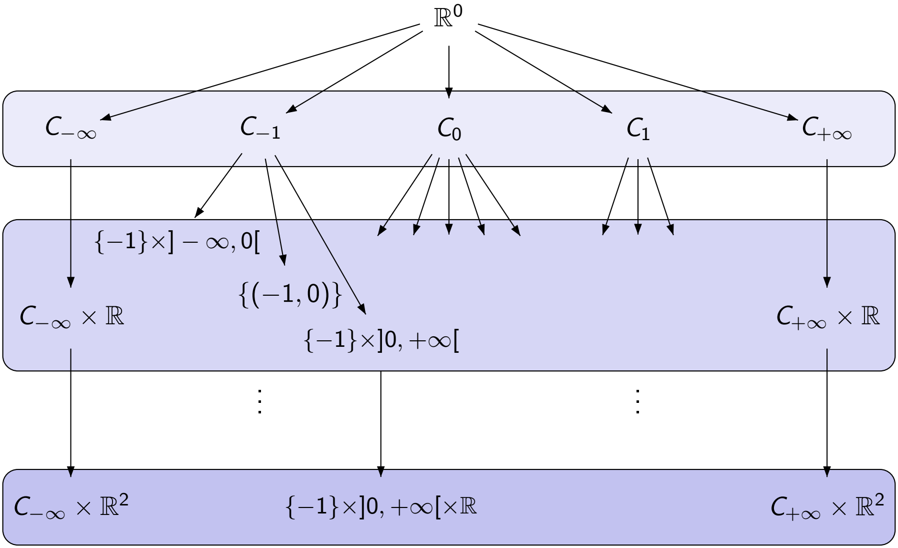
$C_{0,1}$ $\begin{cases} -1 < x_1 < 1 \\ x_2 = \sqrt{1-x_1^2} \end{cases}$

$\boxed{C_{0,0}}$ $\begin{cases} -1 < x_1 < 1 \\ -\sqrt{1-x_1^2} < x_2 < \sqrt{1-x_1^2} \end{cases}$

$C_{0,-1}$ $\begin{cases} -1 < x_1 < 1 \\ x_2 = -\sqrt{1-x_1^2} \end{cases}$

$\boxed{C_{0,-\infty}}$ $\begin{cases} -1 < x_1 < 1 \\ x_2 < -\sqrt{1-x_1^2} \end{cases}$

The tree of cells



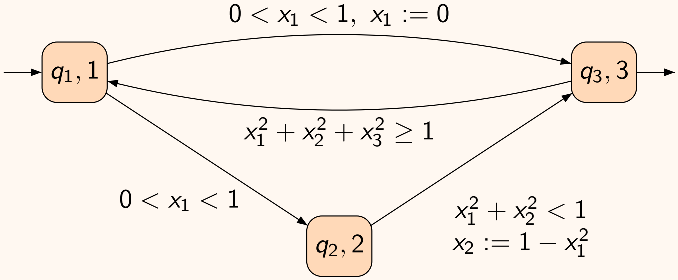
Polynomial ITA

An extension using cylindrical decomposition (work in progress)

Principle

- Replacing linear expressions on clocks by polynomials
- Replacing the saturation procedure by the elimination step
- Using the lifting step to build the class automaton

A PolITA



Conclusion

When computer algebra meets model checking... new decidability questions can be solved.

Complexity questions are next!

Thank you