Verification of hybrid dynamical systems

Jüri Vain
Tallinn Technical University/Institute of Cybernetics
vain@ioc.ee

Outline

- What are Hybrid Systems?
- Hybrid automata
- Verification of hybrid systems
- Verification by reachability analysis
  - Bisimulation of General Transition Systems
  - Bisimulation of Hybrid Systems
- Decidability and complexity results
- Open problems

- Hybrid Systems (HS)

  - Dynamical systems with interacting continuous and discrete components.
  - Continuous trajectories alternate with discrete jumps and switching.

  - Continuous dynamics:
    - robot manipulators;
    - linear circuits;
    - thermal processes.

  - Discrete dynamics:
    - collision in mechanical systems;
    - reley systems;
    - valves and pumps in chemical plants.

Bouncing ball

$q \in Q = \{0\}$, $X = \{x_1, x_2\} \in X = \mathbf{R}$

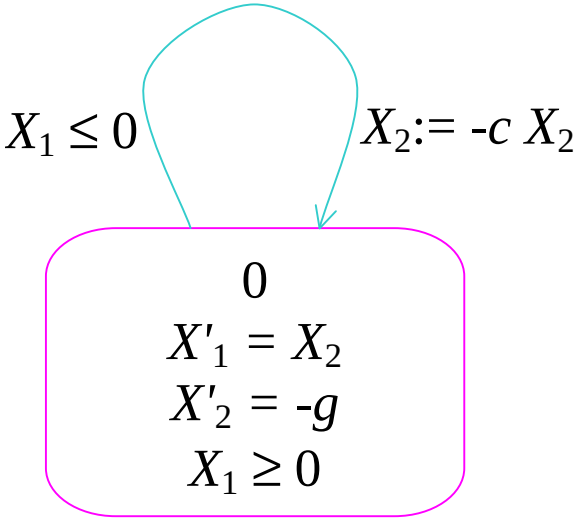
$Init = \{0\} \times \{x \in \mathbf{R}^2 : x_1 \geq 0\} \subseteq Q \times X$

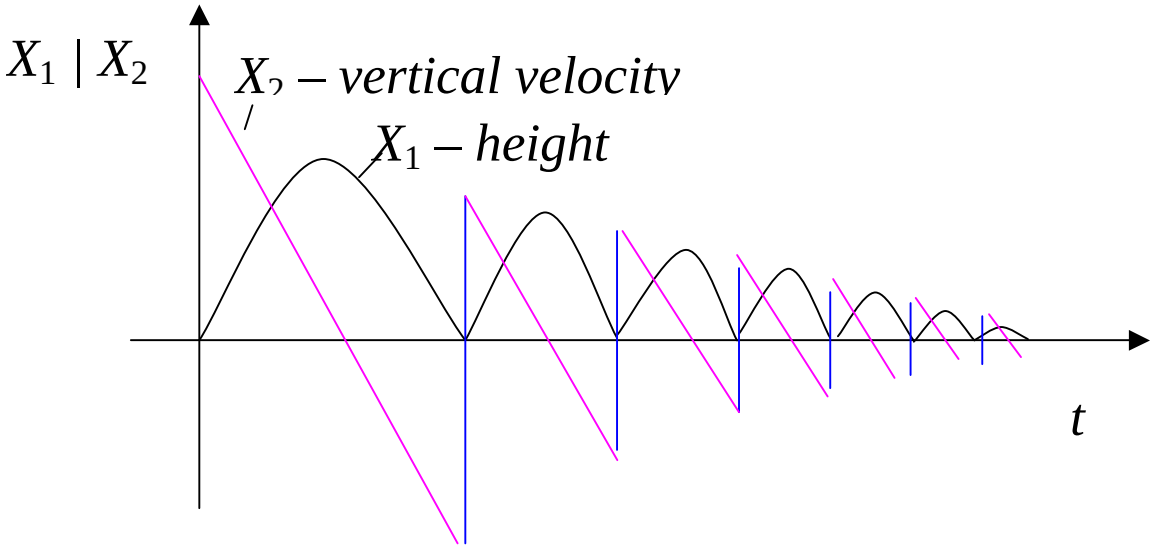
$X' = F(x) = (x_2, -g)$

$Inv(q) = (x_1 \geq 0) \subseteq X$

$G(q, q) = (x_1 \leq 0) \subseteq X$

$R(q, q, x) = (x_1, -c\, x_2) \subseteq X \quad c \in [0,1]$

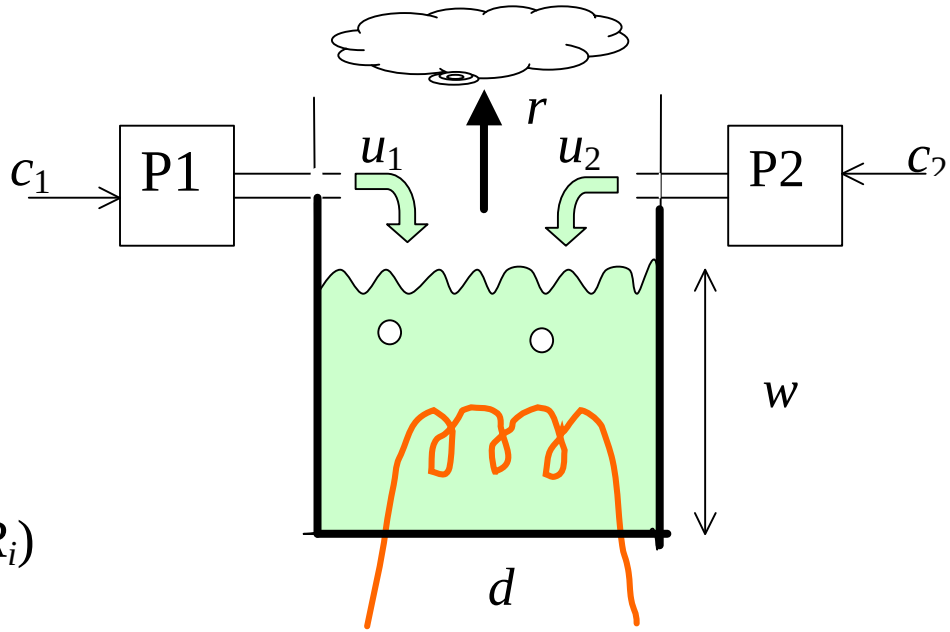


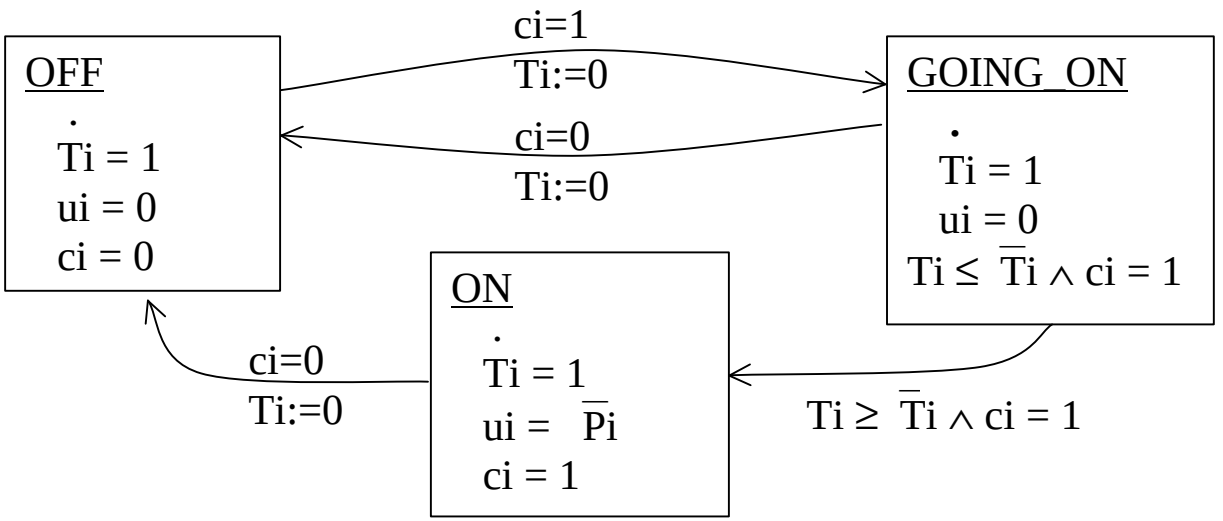


The Steam Boiler

w – water level ($w > 0$)
$u_1(t)$, $u_2(t)$ – pumping rates of P1 and P2
r – rate of evaporation $\dot{r} = d$
d – wattage of the heater
P1, P2 – Pumps

Pump automaton
$HA_{Pi} = (Q_i, X_i, V_i, Y_i, Init_i, f_i, h_i, Inv_i, E_i, G_i, R_i)$





Hybrid Automaton

Definition (*Hybrid Automaton*):

$H = (Q, X, Init, F, Inv, E, G, R)$,

Q – set of discrete variables

X – set of continuous variables

$Init \subseteq Q \times X$ - set of initial states

$F: Q \times X \to TX$ - vector field ($F(q, x) \subseteq \mathbf{R}^n$)

$Inv: Q \to 2^X$ – assigns to each $q \in Q$ an invariant set

$E \subset Q \times Q$ – collection of discrete transitions

$G: E \to 2^X$ – assigns to each $e = (q, q') \in E$ a guard

$R: E \times X \to 2^X$ assigns to each $e = (q, q') \in E$ and $x \in X$ a *reset*-relation

Definition (*Hybrid time trajectory*)
Hybrid time trajectory t is a finite or infinite sequence of intervals of the real line, $t = \{I_i\}$, $i \in \mathbf{N}$, s.t.:

- I_i is closed unless t is a finite sequence and I_i is the last interval. Then it can be right open.
- Let $I_i = [t_i, t'_i]$, then $(\forall i: t_i \leq t'_i)$ and $(\forall i > 0: t_i = t'_{i-1})$.

Remarks:

- Time trajectories are infinite if t is an infinite sequence or it is a finite sequence ending with interval $[t_N, \infty)$.
- T – the set of all hybrid time trajectories.
- For a topological space K and a t, a map $k: t \rightarrow K$ – assigns a value from K to each $t \in t$.

Definition (*Execution*)

An execution χ of a HA H is a collection:

$\chi = (t, q, x)$, with $t \in \mathcal{T}$, $q: t \to Q$ and $x: t \to X$, satisfying

- initial condition: $(q(t_0), x(t_0)) \in$ Init;
- continuous evolution: $\forall i: t_i \le t'_i$, x and q are continuous over $[t_i, t'_i)$ and
  - $\forall t \in [t_i, t'_i)$, $x(t) \in Inv(q(t))$
  - $\frac{d}{dt} x(t) = f(q(t), x(t))$
- discrete evolution:

$\forall i: e = (q(t'_i), q(t_{i+1})) \in E$, $x(t'_i) \in G(e)$ and $x(t_{i+1}) \in R(e, x(t'_i))$.

Remarks:

- χ is a prefix of χ' ($\chi \le \chi'$), if $t \le t'$ and $\forall t \in t: (q(t), x(t)) = (q'(t), x'(t))$.
- An execution is maximal if it is not a strict prefixs of any other execution.
- The set of executions is prefix closed.

Definition (*Types of execution*)
An execution χ of a hybrid automaton H is

- *Finite*, if t is a finite sequence ending in a right closed interval;
- *Infinite*, if t is an infinite sequece or $\Sigma_i(t'_i - t_i) = \infty$;
- *Admissible*, if it is finite or $\Sigma_i(t'_i - t_i) = \infty$;
- *Zeno*, if it is infinite and not admissible. (Zeno time: $t_{¥} = \Sigma_i(t'_i - t_i)$).

Assumption: $f(q, x)$ is globally Lipschitz continuous in x.

Definitsioon (*Reachable State*)
A state $(q^*, x^*) \in Q \times X$ is reachable by H if there exists a finite execution $\chi = (t, q, x)$, with $t = \{[t_i, t'_i]\}^{N}_{i=0}$ and $(q(t'_N), x('_N)) = (q^*, x^*)$.

Verification of HS

Verification: Prove that a HA satisfies a sequence property.

Notation:

- W – set of (discete and/or continuous) variables
- $Hyb(W)$ – set of hybrid sequences on W

$$Hyb(W) = \{(t, w) : t \in T, w: t \to \mathbf{W}\}$$

Example: For an open HA $H = (Q, X, V, Y, \text{Init}, f, h, I, E, G, R)$

$$\text{Var}(H) = Q \cup X \cup V \cup Y$$

Execution: $H \subseteq Hyb(Q \cup X \cup V \cup Y)$,

$$\text{Trace}(H) \subseteq Hyb(V \cup Y)$$

- $H|_W$ – set of sequences of H restricted to variables in W.
- $\text{Trace}(H) = H|_{(V \cup Y)}$

Sequence properties

Definition (Sequence Property)
A sequence property is a pair (W, P) of a collection of variables W, and a map P:

$$P: Hyb(W) \rightarrow \mathbf{B}$$

- Execution χ satisfies property (W, P) ($\chi \models P$), if $\chi \in Hyb(W)$ and $\chi \models P$
- HA H satisfies property (W, P), denote $H \models (W, P)$, if
  - $W \subseteq \mathrm{Var}(H)$
  - $\forall \chi \in H: \chi_{|W} \models P$

LTL – Linear time temporal logic for specification of sequence properties .

Example: Consider HA $H = (Q, X, \mathrm{Init}, f, I, E, G, R)$ and a subset $F \subseteq \mathbf{Q} \times \mathbf{X}$.

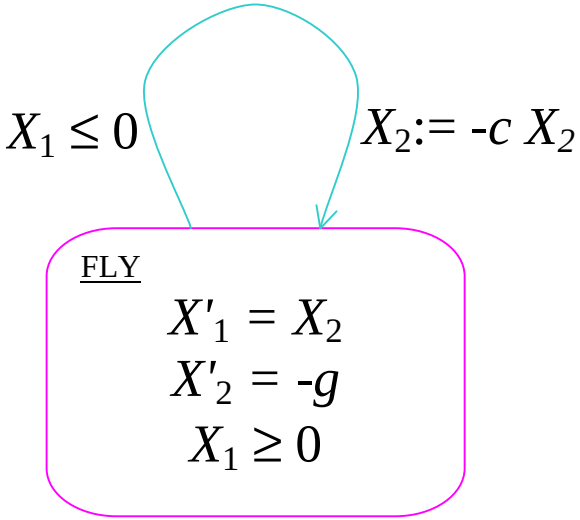
- "*always F*": $(Q \cup X, \pounds F)$, where $\chi \models \pounds F$ iff $\forall t \in \ t : (q(t), x(t)) \in F$.
- "*eventually F*": $(Q \cup X, \Diamond F)$, where $\chi \models \Diamond F$ iff $\exists\, t \in \ t : (q(t), x(t)) \in F$.
- "*responsiveness*": $(Q \cup X, \pounds \Diamond F)$ always, eventually in F.
- "*persistence*": $(Q \cup X, \Diamond \pounds F)$: eventually, always in F.

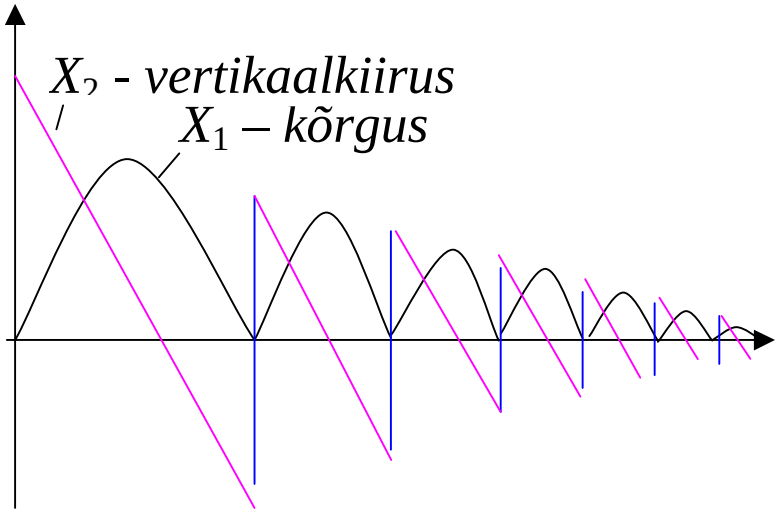
Verification of Sequence properties

Problem (*verification of HA*)

- Given: HA H and a sequence property (W, P), where $W \subseteq \mathrm{Var}(H)$
- Show:
  1) $H \models (W, P)$
  2) If $H \not\models (W, P)$, find a witness χ (*diagnostic trace*), s.t. $\chi|_W \models \neg P$.

Example 1: For bouncing ball automaton $H_{BB} \models (X, \mathbf{\varepsilon}\ (x_1 \geq -1))$:





Example 2: $H_{BB} \models (X, \Diamond (x_1 = 0))$

Proof: After at most one discrete transition continuous evolution starts. Along continuous evolution $x_1(t) = x_1(0) + x_2(0)\, t - gt^2/2$. Therefore, eventually $x_1 = 0$.

Safety and liveness properties

Definition (*Safety Property*):

A sequence property (W, P) is called a *safety property* if it is:

- *Non-empty*: $\{\chi \in \mathrm{Hyb}(W) : P(\chi)\} \neq \varnothing$
- *Prefix closed*: $\forall\ \overline{\chi} \leq \chi : P(\chi) \Rightarrow P(\overline{\chi})$
- *Limit closed*: $\forall i \in (1,\infty): \chi_i \leq \chi_{i+1} \leq \ldots \wedge P(\chi_i) \wedge \lim_{i\to\infty} \chi_i = \chi \Rightarrow P(\chi)$

"*if something bad happens in a sequence, it has to happen after finite "time"* ".

Proposition:

$(W, \pounds\ F)$, for $F \subseteq \mathbf{W}$ with $F \neq \varnothing$ is a safety property.

Definition (*Liveness Property*)
A sequence property is called (W, P) is called a *liveness property* if for all finite sequences $w \in \text{Hyb}(W)$ there exists $\overline{w} \in \text{Hyb}(W)$ s.t.

- $w \leq \overline{w}$
- $\overline{w} \models P$

Proposition:

(W, $\Diamond F$) for $F \subseteq \mathbf{W}$ with $F \neq \emptyset$ is a liveness property.

Example: Liveness properties are $\pounds \Diamond F$ and $\Diamond \pounds F$.

Theorem
Let (W, P) be a sequence property s.t. $\{\chi \in \text{Hyb}(W): P(\chi)\} \neq \emptyset$. Then there exists a safety property (W, P_1) and a liveness property (W, P_2) s.t. $P(\chi) \Leftrightarrow (W, P_1) \wedge (W, P_2)$.

Sequence properties are verified by reachability analysis

Reachability problem

Given a HA H compute Reach(H) $\subseteq \mathbf{Q} \times \mathbf{X}$.

Proposition:

$H \models \mathfrak{L} G$ iff Reach(H) $\subseteq G$.

Model checking by reachability analysis

To compute Reach(H) – requires "computing" with (possibly infinite) sets of states!

Bisimulation (of General Transition Systems)

Definition (*Transition System*) A transition system is a collection $T = (S, \Sigma, \rightarrow, S_0, S_F)$

- S - set of states
- Σ - alfabet of events
  $\rightarrow \subseteq S \times \Sigma \times S$ transition relation
- $S_0 \subseteq S$ set of intial states
- $S_F \subseteq S$ set of final states

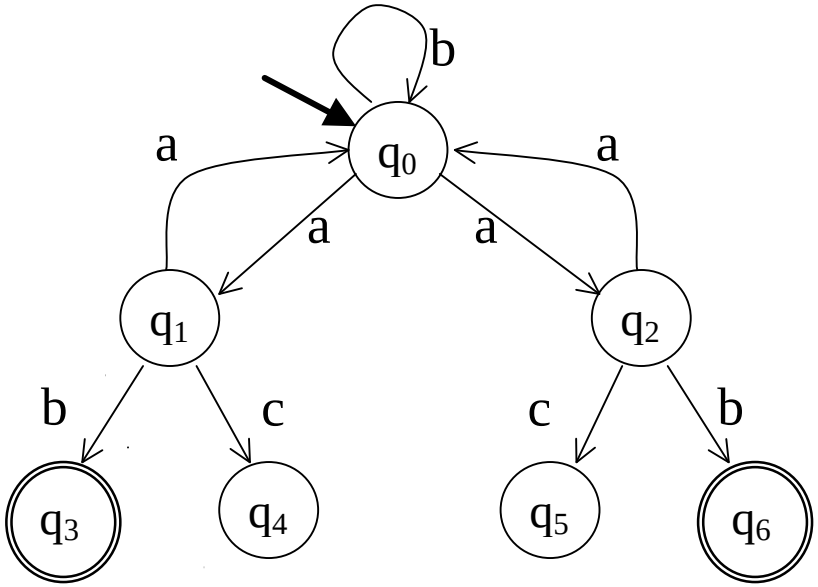
Problem (*Reachability of transition system*):
Given a transition system T, is any state $s_F \in S_F$ reachable from a state $s_0 \in S_0$ by a sequence of T transitions?

Algorithm (*Reachability for TS*)
Initialization
$\text{Reach}_0 := S_0$
$\text{Reach}_{-1} := \varnothing$
$i = 0$
while $\text{Reach}_i \neq \text{Reach}_{i-1}$ **do**
begin
$\text{Reach}_{i+1} := \text{Reach}_i \cup \{s' \in S: \exists s \in \text{Reach}_i,\ s \in \Sigma$, with $(s, \sigma, s') \in \rightarrow\}$
$i := i + 1$
end

!For FSA the reacability algorithm always terminates!
What about infinite state systems?

Decidability of reachability problem bases on *bisimulation*!

Example FSA:



Let $P = \{q_3, q_4, q_5, q_6\}$
$Pre_\sigma(P) = \{ q_1, q_2\}$

Observation:
q_1 and q_2 are very similar, let's make this more precise!

Definition (*equivalence relation*):
A relation $\sim \subseteq S \times S$ is called an equivalence relation if it is

1. Reflexive: $\forall s \in S$: $(s, s) \in \sim$
2. Symmetric: $(s, s') \in \sim \Rightarrow (s', s) \in \sim$
3. Transitive: $(s, s') \in \sim \wedge (s', s'') \in \sim \Rightarrow (s, s'') \in \sim$

- An equivalence relation partitions S to a number of *equivalence classes*:
  $$S = \cup_i S_i$$
  s.t. $\forall s, s' \in S$: $s, s' \in S_i$ iff $(s, s') \in \sim$

- Given an equivalence relation $\sim$, let $S/\sim = \{ S_i \}$ denote the *quotient space*.
- Given a set $P \subseteq S$, let $P/\sim$ represent the part of the quotient space with which P overlaps $P/\sim = \{ S_i : S_i \cap P \neq \emptyset \} \subseteq S/\sim$.

- Let S are the states of a transition system $T = (S, \Sigma, \rightarrow, S_0, S_F)$. The *quotient transition system* of T is $T/\sim = (S/\sim, \Sigma, \rightarrow/\sim, S_0/\sim, S_F/\sim)$, where
  for $S_1, S_2 \in S/\sim$, $(S_1 \sigma S_2) \in \rightarrow/\sim \Leftrightarrow \exists s_1 \in S_1, s_2 \in S_2$: $(s_1 \sigma s_2) \in \rightarrow$

- For $s \in \Sigma$ define $\mathrm{Pre}_\sigma: 2^S \to 2^S$:

$$\mathrm{Pre}_\sigma(P) = \{s \in S: \exists\, s' \in P : (s_1\ \sigma\ s_2) \in \to\}$$

Definition (*Bisimulation*)

Given $T = (S, \Sigma, \to, S_0, S_F)$, and ~ an equivalence relation over S, ~ is bisimulation if:

1. S_0 is union of equivalence classes
2. S_F is union of equivalence classes
3. $\forall\, s \in \Sigma$: if P is union of equivalence classes then $\mathrm{Pre}_\sigma(P)$ is union of equivalence classes

If ~ is a bisimulation, T and $T/\sim$ are called bisimilar.

Proposition

~ is a bisimulation iff $\forall(\, s_1 \sim s_2)$:

1. $s_1 \in S_0 \Rightarrow s_2 \in S_0$
2. $s_1 \in S_F \Rightarrow s_2 \in S_F$
3. $((s_1\ \sigma\ s'_1) \in \to) \Rightarrow \exists\, s'_2 : ((s'_1, s'_2) \in \sim) \ \wedge\ ((s_2\ \sigma\ s'_2) \in \to)$.

- More generally, two transition systems T and T' are called *bisimilar* if there exists a relation $\sim \subseteq S \times S'$ s.t. T is bisimulation of T' and T' is bisimulation of T.
- Bisimulations are *reachability* (and language) *preserving* quotient systems.
- How to find a *finite* bisimulation?

Algorithm (*Bisimulation*)

Initialization: $S/\sim = \{ S_0, S_F, S \setminus (S_0 \cup S_F)\}$
while $\exists P, P' \in S/\sim\ ,\ s \in \Sigma$: $P \cap \mathrm{Pre}_\sigma(P') \neq \emptyset \wedge P \cap \mathrm{Pre}_\sigma(P') \neq P$ **do**
begin
$P_1 := P \cap \mathrm{Pre}_\sigma(P')$
$P_2 := P \setminus \mathrm{Pre}_\sigma(P')$
$S/\sim := (S/\sim \setminus \{P\} \cup \{ P_1 , P_2 \}$
end

- If the algorithm terminates, then $\sim$ is a bisimulation since $S_0 \subseteq S/\sim$, $S_F \subseteq S/\sim$, and $\mathrm{Pre}_\sigma(P') \subseteq S/\sim$.
- For FSM the algorithm always terminates!

- Why is this an improvement?
  - Computational advantage: to check reachability, we just search through the equivalence classes instead of single states;
  - Extends to systems with infinite states: if the bisimulation quotient can be computed and it is finite, then the reachability problem is decidable.

- How to find finite quotient spaces for hybrid systems?

Finite bisimulation exists for following subclasses of HA:

- timed automata;
- initialized rectangular automata;
- linear hybrid automata.

Initialized Rectangular Automata

- A set $R \subset \mathbf{R}^n$ is called a *rectangle* if $R = \Pi^n_{i=1} R_i$, where R_i are intervals whose finite end points are rational.

Examples: $R_1 = (1,\infty)$, $R_2 = [-3, 3/4)$, $R_2 = \{3\}$.

Definition (*Rectangular Automaton*):
A rectangular automaton is a HA $H = (Q, X, \text{Init}, f, I, E, G, R)$, where

- Q – set of discrete variables, $\mathbf{Q} = \{q_1, \ldots, q_m\}$
- $X = \{ x_1, \ldots, x_n \}$, $\mathbf{X} = \mathbf{R}^n$;
- $Init = \cup^m_{i=1} (\{q_i\} \times Init\ q_i)$, where $Init(q_i) = Init_1(q_i) \times \ldots \times Init_n(q_i)$ is a rectangle;
- $\forall (q, x)$: $f(q, x) = F(q)$, where $F(q) = F_1(q) \times \ldots \times F_n(q)$ is a rectangle;
- $\forall q \in \mathbf{Q}$: $I(q)$ is a rectangle;
- $E \subset \mathbf{Q} \times \mathbf{Q}$;
- $\forall e = (q, q') \in E$: $G(e) = G_1(e) \times \ldots \times G_n(e)$ is a rectangle
- $\forall e$: $R(e, x) = R_1(e, x) \times \ldots \times R_n(e, x)$, where

$$R_i(e, x) = \begin{cases} \{x_i\}, & \text{or} \\ \text{a fixed (independent of } x\text{) interval} \end{cases}$$

Proposition (Puri, Varaiya, Borkar, 95):

Reachable set of a Lipschitz differential equation over a finite time horizon can be approximated arbitrarily closely by the reach set of a rectangular automaton.

Differential inclusion:

- Differential inclusion is an abstraction of O.D.E. for reachability computations.

  $\dot{x} = f(x) \in F(x)$, where $F(x)$ – convex subset of $\mathbf{R}^n$.

- Differential inclusion is *non-deterministic*, i.e., many executions may exist for a single initial condition;

An *execution of the differential inclusion* $\dot{x} \in F(x)$, $x(0) = x_0 \in \mathbf{R}^n$ on $[0, T] \subset \mathbf{R}^+$ is a function x: $[0, T] \to \mathbf{R}^n$ with $x(0) = x_0$ s.t. $\forall t \in [0, T]$: $\dot{x} \in F(x)$,

Definition (*Initialized Rectangular Automaton*)

A rectangular automaton is called *initialized* if for all transitions $e = (q, q') \in E$:

$$F_i(q) \neq F_i(q') \Rightarrow R_i(e, x) \neq \{x_i\}.$$

Bisimulation of HS

Consider HS as a transition system with:

- Discrete Transitions: $(q, x) \rightarrow^{e} (q', x')$ where $(q, x) \in G(e)$ and $(q', x') \in R(e)$
- Continuous Transitions: $(q_1, x_1) \rightarrow^{t} (q_2, x_2)$ iff
  - $q_1 = q_2$
  - exist $\delta \geq 0$ and curve $x: [0,\delta] \rightarrow X$ with $x(0) = x_1$, $x(\delta) = x_2$
  - $\forall t \in [0,\delta]: \dot{x} = F(q_1, x(t)) \wedge x(t) \in I(q_1)$

Given $e \in E$ and any region $P \subseteq X$, define $Pre_t(P)$ and $Pre_e(P)$:

$$Pre_e(P) = \begin{cases} \emptyset & \text{if } P \cap R(e) = \emptyset \\ G(e) & \text{if } P \cap R(e) \neq \emptyset \end{cases}$$

If sets $R(e)$, $G(e)$ are blocks of any partition of X then no partition refinement is necessary due to $e \in E$,

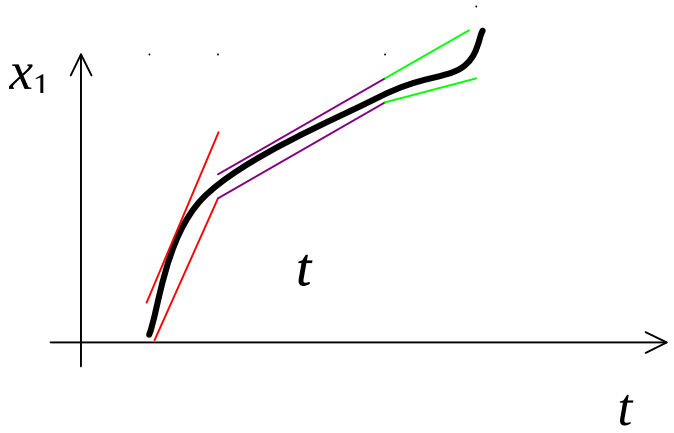
i.e., initial partition in BS algorithm should contain X_0, X_F, and for each $q \in$ Q, a collection of sets

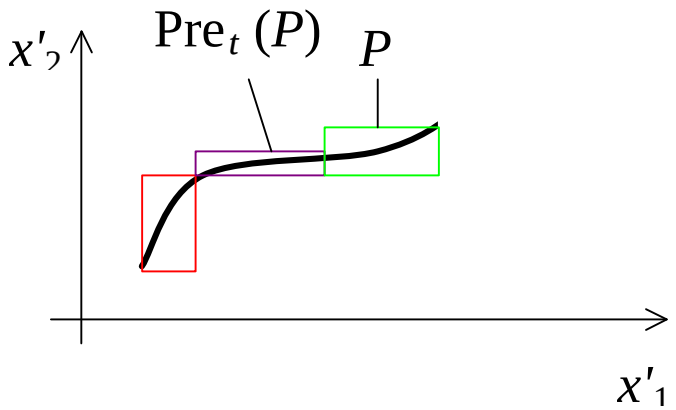
$$A_q = \{I(q), (X_0)_q, (X_F)_q\} \cup \{G(e)_q, R(e)_q : e \in E\}$$

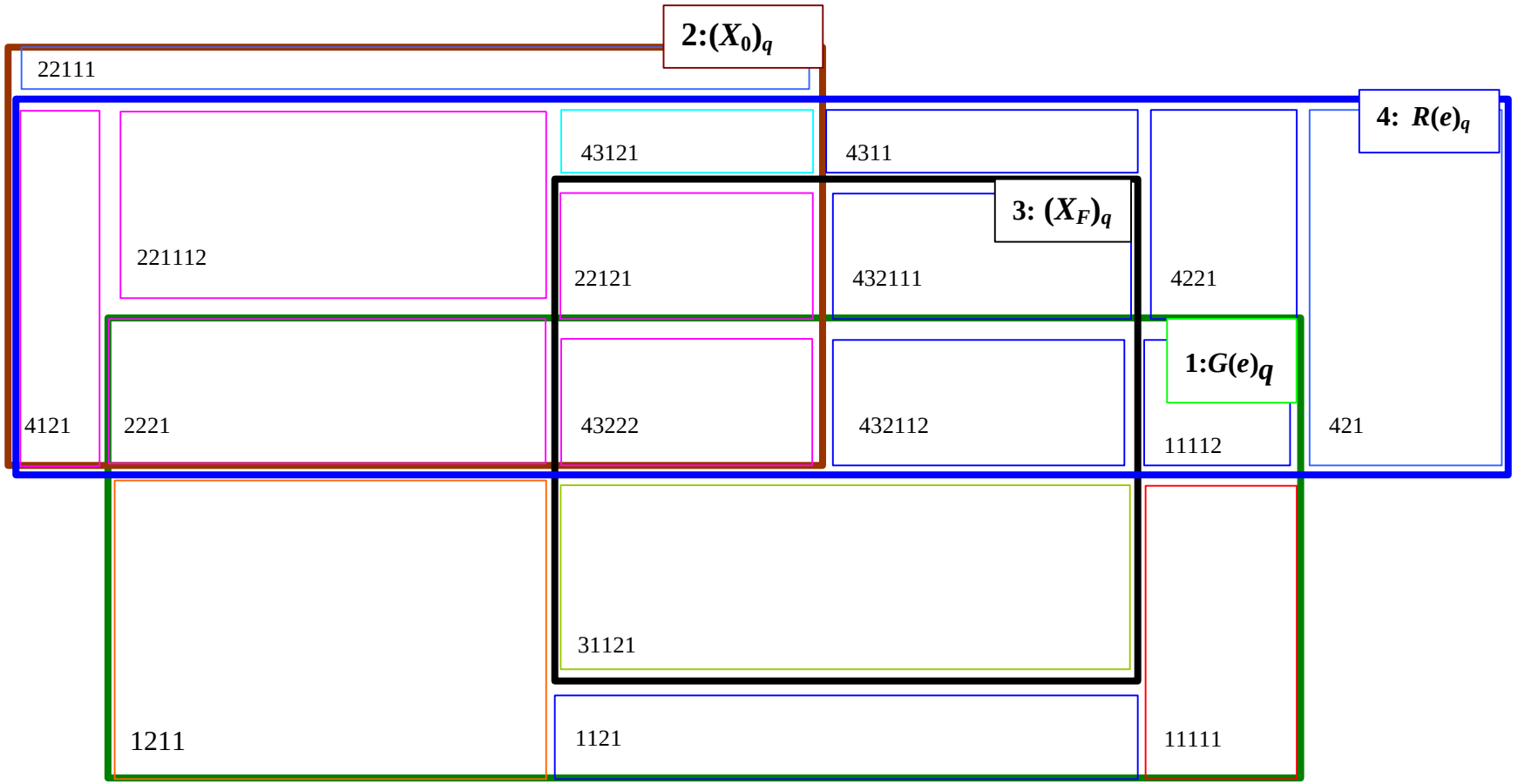
Let S_q be the coarsest partition of X_c compatible with A_q, i.e., each set in A_q is a union of set in S_q.

S_q is the starting partition of the BS algorithm.

How to compute $Pre_t\ (P)$?







Example of S_q partition refinement for a q

Algorithm (bisimulation for HS)

set $X/\sim = \cup_q (q, S_q)$
for $q \in X_D$
while $\exists P, P' \in S_q$ *s.t.* $\emptyset \neq P \cap Pre_q(P') \neq P$ **do**
set $P_1 = P \cap Pre_q(P')$, $P_1 = P \setminus Pre_q(P')$
refine $S_q = (S_q \setminus \{P\}) \cup \{P_1, P_2\}$
end while
end for

Observation:

- iteration is carried out independently for each location;
- algorithm terminates if it terminates for for each discrete location q.

Decidability results

Theorem

The reachability problem for initialized rectangular automata is complete for PSPACE.

- Good: Reachability is decidable
- Bad: The computation scales very badly

  e.g., the number of equivalence classes in TA: $m(n!)(2^n)\,\Pi^n_{i=1}(2c_i+2)$,

  n – number of clocks!!!

  m – number of discrete states

  c_i - largest constant in i^{th} clock conditions

→ generally, the number of continuous state variables is critical!

- The reachability is undecidable for HA if:
  1. comparisons between x_i with different rates;
  2. non-initialized variables;
  3. assignement with continuous variables: $x_i := x_j$
- Decidability results are generalized for classes of HS where sets are not rectangular

O-Minimal Hybrid Systems

Termination of the bisimulation algorithm critically depends on whether the intersection of trajectories and sets consists of a finite number of connected components.

Definition (*O-Minimal Hybrid System*)
Hybrid system $H = (X, X_0, X_F, F, I, E, G, R)$ is o-minimal if

- $X_C = \mathrm{R}^n$
- for $\forall q \in X_D$ the flow of $F(q, .)$ is complete
- for $\forall q \in X_D$ the family of sets $A_q = \{I(q), (X_0)_q, (X_F)_q\} \cup \{G(e)_q, R(e)_q : e \in E\}$ and the flow of $F(q, .)$ are definable in the same o-minimal theory of **R.**

Definition (*O-Minimal Theories*)
A theory of the reals is o-minimal if every definable subset for **R** is a finite union of points and intervals (possibly unbounded).

Examples of o-minimal theories:

Theory	Model	Definable Sets	Definable Flows
Lin (**R**)	(**R**, +, -, <, 0, 1)	Semilinear sets	Linear flows
OF(**R**)	(**R**, +, -, ×, <, 0, 1)	Semialgebraic sets	Polynomial flows
OF_{an}(**R**)	(**R**, +, -, ×, <, 0, 1,{*f*})	Subanalytic sets	Polynomial flows
OF_{exp} (**R**)	(**R**, +, -, ×, <, 0, 1, e^x)	Semialgebraic sets	Exponential flows
$OF_{exp,an}$(**R**)	(**R**, +, -, ×, <, 0, 1, e^x,{*f*})	Subanalytic sets	Exponential flows

Theorem (*Finite Bisimulation*)
Every o-minimal HS admits a finite bisimulation. Equivalently, the bisimulation algorithm terminates for all o-minimal HSs.

References

1. Thomas A. Henzinger The theory of hybrid automata. *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science, pp. 278-292, 1996.*
2. R. Alur, T. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE, volume 88, number 7, July 2000.*
3. A. Chutinan and B.H. Krogh. Verification of Hybrid Systems Using Polygonal Flowpipe Approximations. *Hybrid Systems: Computation and Control, Lecture Notes in Computer Science 1999.*
4. R. Alur. Timed Automata. *NATO-ASI 1998 Summer School on Verification of Digital and Hybrid Systems.*
5. G.Pappas. Hybrid systems: Computation and abstraction. PhD Thesis. Univ. of California at Berkeley. 1998.
6. John Lygeros & Shankar Sastry. Hybrid Systems: Modeling Analysis & Control (Course notes ee291E). Spring 1999.