IA169 System Verification and Assurance

Verification of Real-Time and Hybrid systems

Jiří Barnat

Verification in Model-Based Development

Software Engineering Experience

- Employing V&V techniques too late in the development process significantly increases the cost of poor quality.
- The sooner a bug is detected the cheaper is the fix.
- Model-Based Development
- Model-Based Verification

Model-Based Development

- Consider models of the target system in order to ,e.g., simulate its behaviour in the design phase prior implementation.
- Behavioural models can be used for verification.

Hybrid Systems

Hybrid Systems

- Systems that combine multiple kinds of dynamics.
- Continuous systems driven by discrete events.

Areas of existence

- Mechanical systems
  - Continuous movement and contact with physical obstacle.
- Electrical systems
  - Continuous nature of electric charge in circuit driven by discrete switches.
- **Embedded systems**
  - Computer-driven systems in analogue environment.

Example – Bouncing Ball

System Description

- A ball released at height h bounces on a hard surface. The ball is under continuous influence of the gravity (9.8m/s). When bounces some energy is consumed by friction and elasticity and turns into heat.

Physics

- Acceleration = First derivative of speed with respect to time.
- Speed = First derivative of height with respect to time.

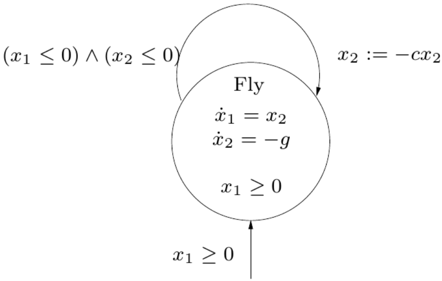
Abstraction and simplification

- Modelled with a mass point.
- Instant (time-less) bounce.

Automaton Description

- $x1$ — height
- $x2$ — vertical speed ($+$ means up, $-$ means down)
- $c \in [0, 1]$ — loss of energy (elasticity and heat)

Schema



Questions

- What time elapses between the fourth and fifth bounce?
- If given horizontal speed, will the ball jump over an obstacle?
- ...

Searching for Answers

- Need for precise formal description of hybrid system.
- Algorithmic analysis of properties of hybrid systems and controller synthesis.

Hybrid Automata

Hybrid Automata

Hybrid Automaton is a tuple

- $Q = \{q1, q2, \ldots\}$ — Set of discrete states.
- $X = \mathbf{R}^n$ — Set of continuous states.
- $f : Q \times X \to \mathbf{R}^n$ — System dynamics.
- $Init \subseteq Q \times X$ — Set of initial states.
- $Dom : Q \to PowerSet(X)$ — State invariants.
- $E \subseteq Q \times Q$ — Set of discrete transitions
- $G : E \to PowerSet(X)$ — Map of transition guards.
- $R : E \times X \to PowerSet(X)$ — Map of transition resets.

State of Hybrid Automaton

- Given by the discrete state and the current value of continuous variables: $(q, \overrightarrow{x}) \in Q \times X$.

Initial State

- Set of initial states in both the discrete and continuous part.
- $(q_0, \overrightarrow{x_0}) \in I$

Transitions of Hybrid Automaton

Transition by Time Passing

- Let $(q, \overrightarrow{x})$ be origin state.
- Continuous part for every variable x follows the system dynamics

$$\frac{dx(t)}{dt} = f(q, x), \text{ where } x(0) = x$$

- Discrete part does not change:

$$q(t) = q$$

- Time may pass only if the state invariant is valid:

$$x(t) \in Dom(q)$$

Discrete Transition

- Let $(q, \overrightarrow{x})$ be origin state.
- It is possible (but not necessary) to perform a transition

$$(q, q') \in E,$$

- if transition guard is valid, i.e.

$$\overrightarrow{x} \in G(q, q').$$

- If the transition is taken, the continuous part of the state is updated accordingly:

$$\overrightarrow{x'} := R((q, q'), \overrightarrow{x})$$

- The target state after a discrete transition is $(q', \overrightarrow{x'})$.

Reasonable restrictions of Hybrid Automata

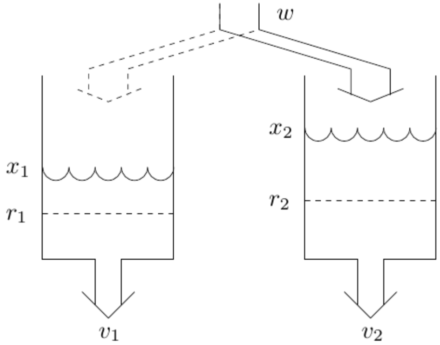
Restrictions in Continuous Part

- $f(q, \overrightarrow{x})$ is Lipschitz continuous for $\forall q \in Q$, (solution of differential equations is well defined)
- $\forall e \in E$ we assume non-empty $G(e)$
- $\forall e \in E$ and $\forall x \in Q$ we assume non-empty $R(e, x)$

Restrictions in Discrete Part

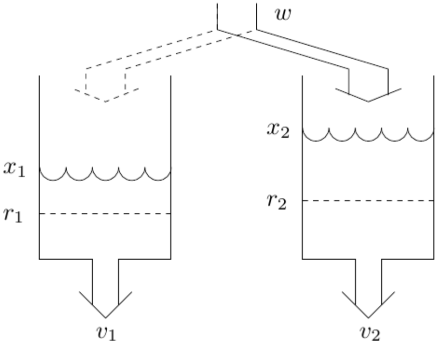
- The set of discrete state is finite.

Example 2 – Water Tank



- Two water tanks, volume of water denoted with x_1 and x_2.
- There is a constant speed leak from both tanks, v_1 and v_2.
- A hose can fill one of the tanks with speed w.
- The hose is always in exactly one of the tanks.

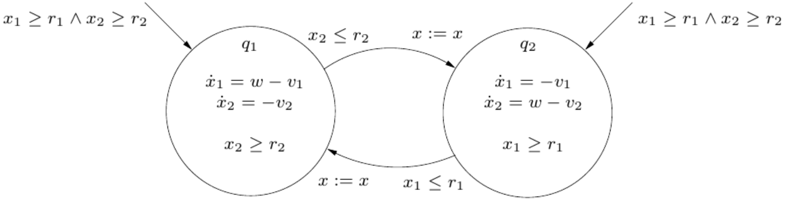
Example 2 – Water Tank



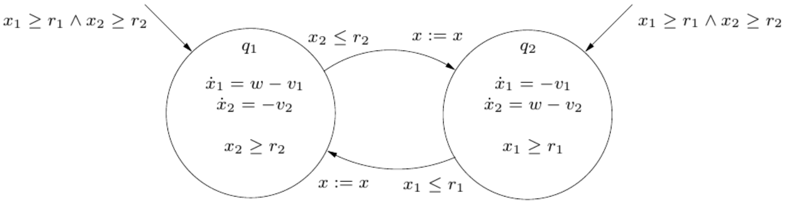
Goal

- Keep water level above the necessary minimum r_1 and r_2.
- Initially, there is enough water in both tanks.
- The hose is switched to a tank at the moment the water level in the tank drops to the required minimum.

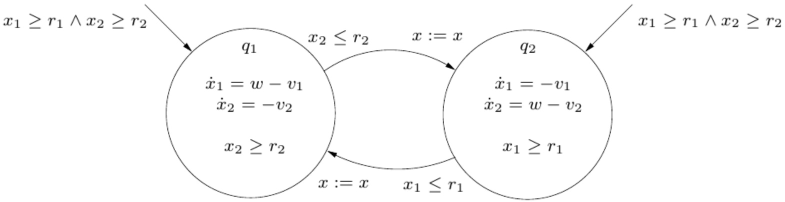
Water Tanks — Formal Definition of the System



- $Q = \{q_1, q_2\}$
- $X = \mathbf{R} \times \mathbf{R}$
- $f(q_1, x) = \begin{bmatrix} w - v_1 \\ -v_2 \end{bmatrix} \quad f(q_2, x) = \begin{bmatrix} -v_1 \\ w - v_2 \end{bmatrix}$



- $Init = \{q_1, q_2\} \times \{x \in \mathbf{R} \times \mathbf{R} \mid x_1 \geq r_1 \wedge x_2 \geq r_2\}$
- $Dom(q_1) = \{x \in \mathbf{R} \times \mathbf{R} \mid x_2 \geq r_2\}$
  $Dom(q_2) = \{x \in \mathbf{R} \times \mathbf{R} \mid x_1 \geq r_1\}$



- $E = \{(q_1, q_2), (q_2, q_1)\}$
- $G(q_1, q_2) = \{x \in \mathbf{R} \times \mathbf{R} \mid x2 \leq r2\}$
  $G(q_2, q_1) = \{x \in \mathbf{R} \times \mathbf{R} \mid x_1 \leq r_1\}$
- $R(q_1, q_2, x) = R(q_2, q_1, x) = \{x\}$

Hybrid Time Sequence (HTS)

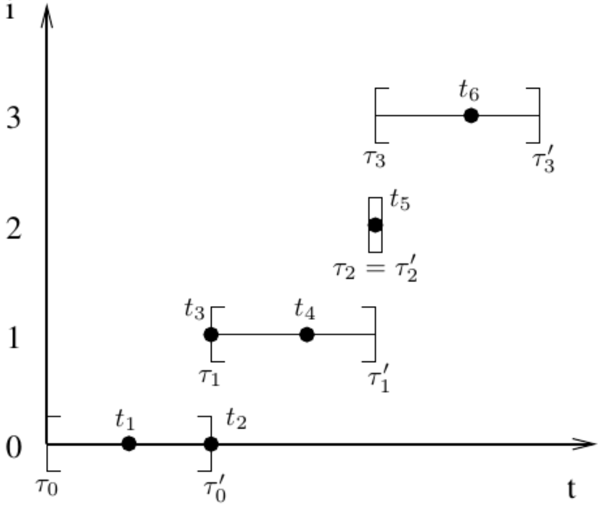
Informally

- A run of hybrid automaton proceeds in a sequence of continuous time intervals. Discrete transitions happen on the boundaries of the intervals in instant time.
- The time characteristic of a run of hybrid automaton is formalised with the usage of the so called **Hybrid Time Sequence**.

Definitions

- Hybrid Time Sequence is a (finite or infinite) sequence of intervals $\tau = \{I_0, I_1, \ldots, I_N\} = \{I_i\}_{i=0}^{N}$ such that:
  - $I_i = [\tau_i, \tau_i']$ for all $i < N$
  - If $N < \infty$ then either $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$
  - $\tau_i \leq \tau_i' = \tau_{i+1}$ for all $0 \leq i < N$.

Graphical Representation of Hybrid Time Sequence



Ordering of Time Moments

Observation

- If every time moment is related with an interval of HTS ...
- ... then time moments can be linearly ordered.

Ordering $\prec$

- $t_1 \in I_i,\ t_2 \in I_j$
- $t_1 \prec t_2 \stackrel{def}{=} (t_1 < t_2) \vee (t_1 = t_2 \wedge i < j)$

Generalisation

- Every hybrid time sequence is linearly ordered with $\prec$ relation.

Prefix Of Hybrid Time Sequence

- $\tau = \{I_i\}_{i=0}^{N}$
- $\hat{\tau} = \{\hat{I}_i\}_{i=0}^{M}$
- We say that τ is a prefix of $\hat{\tau}$ (denoted with $\tau \sqsubseteq \hat{\tau}$), if

  $\tau = \hat{\tau}$, or

  N is finite $\wedge\ I_N \subseteq \hat{I}_N \wedge \forall i \in [0, N) : I_i = \hat{I}_i$

Proper Prefix

- $\tau \sqsubset \hat{\tau} \quad \equiv \quad \tau \sqsubseteq \hat{\tau} \ \wedge \ \tau \neq \hat{\tau}$

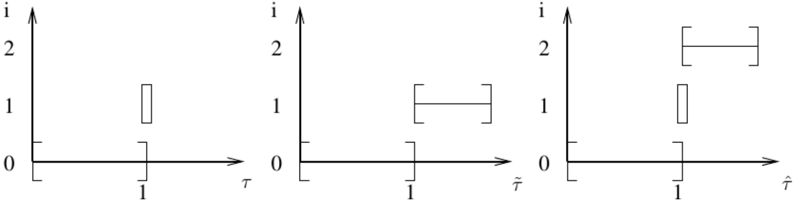
Relation $\sqsubseteq$ is a Partial Ordering

- There exist τ and $\widehat{\tau}$ such that $\tau \not\sqsubseteq \widehat{\tau}$ and $\widehat{\tau} \not\sqsubseteq \tau$.

Task – Find $\tau, \widetilde{\tau}$ and $\widehat{\tau}$ such that

- $\tau \sqsubseteq \widetilde{\tau}$
- $\tau \sqsubseteq \widehat{\tau}$
- $\widetilde{\tau} \not\sqsubseteq \widehat{\tau} \wedge \widehat{\tau} \not\sqsubseteq \widetilde{\tau}$

Solution



Hybrid Trajectories

Definition

- Hybrid trajectory is a triple (τ, q, x), where τ is hybrid time sequence $\tau = \{I\}_0^N$ and q, x are two sequences of functions $q = \{q_i\}_0^N$ and $x = \{x_i\}_0^N$ such that $q_i : I_i \to Q$ and $x_i : I_i \to \mathbf{R}^n$, respectively.

Intuition

- Continuous part flows within individual time intervals of hybrid time sequence.
- Discrete state within a single interval does not change.
- Discrete transitions realise transitions from the end of one interval to the beginning of the succeeding interval.

Run of Hybrid Automaton

- Let $\mathcal{H} = (Q, X, f, Init, Dom, E, G, R)$ be hybrid automaton.
- Let (τ, q, x) be hybrid trajectory.
- Trajectory (τ, q, x) is a run of automaton $\mathcal{H}$, if it is compliant with $\mathcal{H}$ in: initial condition, discrete behaviour and continuous behaviour.

Initial Condition

- $(q_0(0), x_0(0)) \in Init$

Discrete Behaviour – For all $i < N$ it holds that

- $(q_i(\tau_i'), q_{i+1}(\tau_{i+1})) \in E$
- $x_i(\tau') \in G(q_i(\tau_i'), q_{i+1}(\tau_{i+1}))$
- $x_{i+1}(\tau_{i+1}) \in R(q_i(\tau_i'), q_{i+1}(\tau_{i+1}), x_i(\tau_i'))$

Run of Hybrid Automaton

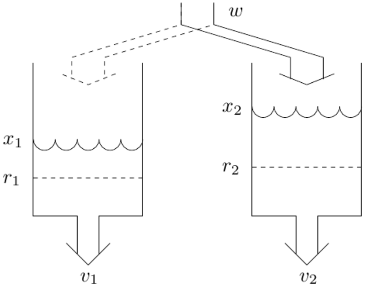
- Let $\mathcal{H} = (Q, X, f, Init, Dom, E, G, R)$ be hybrid automaton.
- Let (τ, q, x) be hybrid trajectory.
- Trajectory (τ, q, x) is a run of automaton $\mathcal{H}$, if it is compliant with $\mathcal{H}$ in: initial condition, discrete behaviour and continuous behaviour.

Continuous Behaviour – For all $i \leq N$ it holds that

- $q_i : I_i \to Q$ is constant over $t \in I_i$,
- $x_i : I_i \to X$ is a solution to differential equation

$$\frac{dx_i(t)}{dt} = f(q_i(t), x_i(t))$$

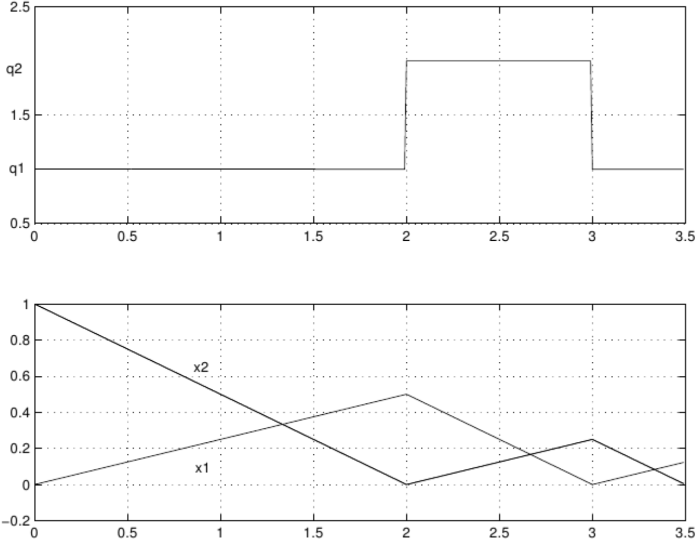
  over I_i beginning in $x_i(\tau_i)$,
- For all $t \in [\tau_i, \tau_i')$ it holds that $x_i(t) \in Dom(q_i(t))$.



Specification

- $\tau = \{[0, 2], [2, 3], [3, 3.5]\}$
- Constants $r_1 = r_2 = 0$, $v_1 = v_2 = 0.5$, $w = 0.75$
- Initial state $q = q_1$, $x_1 = 0$, $x_2 = 1$.

Water Tanks – Trajectories



Classification of Runs (τ, q, x)

Finite

- If τ is finite and the last interval of τ is closed.

Infinite

- If τ is infinite sequence, or the sum of time intervals in τ is infinite, i.e.

$$\Sigma_{i=0}^{N}(\tau_i' - \tau_i) = \infty.$$

Zeno

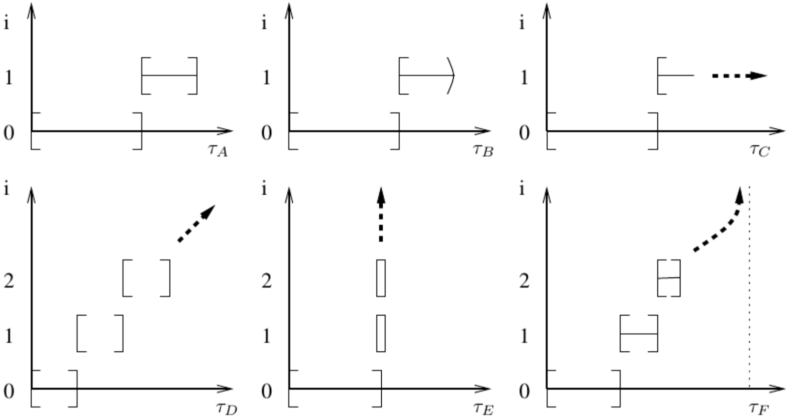
- If τ is infinite, but

$$\Sigma_{i=0}^{N}(\tau_i' - \tau_i) < \infty.$$

Maximal

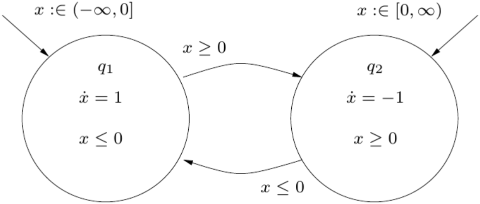
- If τ is no proper suffix of any other run τ' of $\mathcal{H}$.

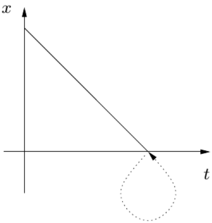
Classification of Runs



τ_A finite; τ_C and τ_D infinite; τ_E and τ_F Zeno.
What class is run τ_B?

Examples of ZENO Runs



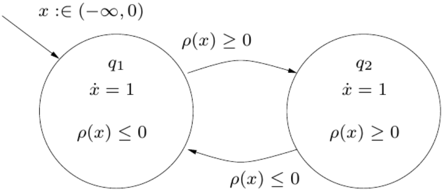


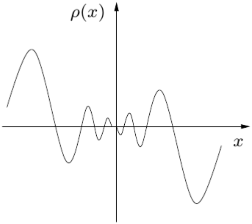
Let

$$\rho(x) = \begin{cases} \sin\left(\frac{1}{x^2}\right)\exp\left(-\frac{1}{x^2}\right) & \text{if } x \neq 0 \\ 0 & \text{if } x = 0 \end{cases}$$

Then

- the following hybrid system has infinitely many intersections with x axis in the interval $(-\epsilon, 0]$.





Modelling Hybrid Systems

Observation

- Hybrid automata are meant to describe real hybrid systems.
- Due to abstraction and simplification, it is possible to specify unrealistic situation.

Risk of Modelling

- Can create system that have no solutions.
- Can create system that have only unrealistic solutions.
- Can create system that have non-deterministic solutions.

Terminology

- System that has no solution (no run exist) is called *blocking* system.

Unrealistic Runs

Observation

- Non-blocking system does not guarantee that some runs are realistic.
- Non-blocking system does not guarantee that some runs are time divergent.

Unrealistic Runs

- Runs that perform infinitely many discrete transitions in finite time are called **ZENO** runs.
- Created by abstraction and simplification in modelling.

Discussion

- Why the ball does not bounce forever?
- It is important to see which simplification lead to ZENO runs.

Non-determinism

Non-determinism

- In general can be described as absence of unique solutions, i.e. that a hybrid automaton accepts multiple different runs emanating from the same initial conditions.
- When limited to Lipschitz continuous functions with unique solution, reason for non-determinism comes from discrete transitions.

Non-deterministic on Purpose

- Can be used to model uncertainty.
- Modeller should make difference between non-determinism due to simplification, and non-determinism used on purpose.

Observation

- Non-determinism is a real cause of troubles in both analysis and controller synthesis of hybrid systems.

Problems of Simulations and Analysis of Hybrid Systems

Existence of Solution

- How to detect existence of non-blocking run?
- How to detect ZENO behaviour?

Uniqueness

- How to perform simulation of non-deterministic system?
  - Discrete transition vs. continuous time evolution.
  - Discrete transition vs. discrete transition.
- As-soon-as semantics.

Discontinuity

- How to detect satisfiability of transition guards?
- What if state invariant ends with open interval $[a, b)$ and the succeeding transition is allowed to execute at time $[b]$?

Composition

- How to compose hybrid automata?

Non-blocking and Deterministic Hybrid Automaton

Non-blocking Hybrid Automaton

- Hybrid automaton H is called non-blocking if for all initial states $(\hat{q}, \hat{x}) \in \mathit{Init}$ there exist an infinite run emanating from $(\hat{q}, \hat{x})$.

Deterministic Hybrid Automaton

- Hybrid automaton H is called deterministic, if for all initial states $(\hat{q}, \hat{x}) \in \mathit{Init}$ there exist at most one maximal run emanating from $(\hat{q}, \hat{x})$.

Using Hybrid Automata

Motivation for Modelling

- The goal of modelling of HS is to deduce properties of, or synthesise controllers for real HS from properties of, or controllers for modelled HS.

Verification

- Does hybrid system exhibits declared behaviour (does it satisfy specification)?

Synthesis

- There are number of choices to build a HS, using models it is possible to decide which choices are good and which are bad prior the construction of the real HS.

Validation

- Check that the design described as a hybrid automaton and the real product produced behave accordingly.
- Some system modelled with hybrid automata may be unrealistic (and cannot be built) due to simplifications and abstractions used during modelling phase.

Usual Work-flow

- Synthesis (of model)
- Verification (of model)
- Validation (equivalence of model and real product)

Specification

Stability

- Typical property of purely continuous systems.
- To request stability for hybrid systems requires to specify what does the stability means with respect to the discrete part of the system.

Specification by Set of States

- Specification of safety and forbidden areas.

Specification by Set of Trajectories

- Properties of hybrid systems that can be expressed as properties of runs.
- Set of allowed runs of a hybrid automaton.
- Formally described using modal and temporal logic, such as (CTL, LTL, CTL*).

Methods of Analysis of HS

Deductive Methods

- Using math reasoning, such as math induction, to deduce properties of hybrid systems.

Model Checking

- Algorithmic procedure for deciding formally specified properties of hybrid systems.
- Decidable only for limited sub-classes of hybrid automata.

Simulations

- Used to estimate the set of reachable states.
- The precision of estimation is difficult to say.

Deductive Methods – Invariant Set

Typical Goal

- Typical goal for deductive methods is to set boundaries of the *reach* set using the so called **Invariant Set.**
- Invariant set is a set of states for which it holds that if a run of hybrid system is initiated at the state of the set it only visits states that are in the set (i.e. never leaves invariant set).

Formal Definition of Invariant Set

- Set of state $M \subseteq Q \times X$ of hybrid automaton H is called *invariant* if for all $(\hat{q}, \hat{x}) \in M$, all solutions (τ, q, x) starting from $(\hat{q}, \hat{x})$, all $I_i \in \tau$ and all $t \in I_i$ it holds that $(q_i(t), x_i(t)) \in M$.

Deductive Methods – Properties of Invariant Set

Observation

- Union and Intersection of Invariant Sets of hybrid automaton H is also an invariant set for H.
- If M is an invariant set and $Init \subseteq M$, then $Reach \subseteq M$.

Consequence

- To approximate the *Reach* set it is possible to deduce a number of invariant sets that contain initial state and are at the same time below the set of all states of hybrid automaton (here denoted by F)

$$Init \subseteq M \subseteq F$$

and intersect them.

Simplification

- For hybrid automata we restrict ourselves in the course to algorithmic test of reachability of a given state.

Considered Sub-classes of Hybrid Automata

- Timed Automata (TA).
- Rectangular Hybrid Automata (RHA).
- Linear Hybrid Automata (LHA).

Software Tools

- UPPAAL – Timed Automata
- PHaVer – RHA, partially LHA (HyTech)

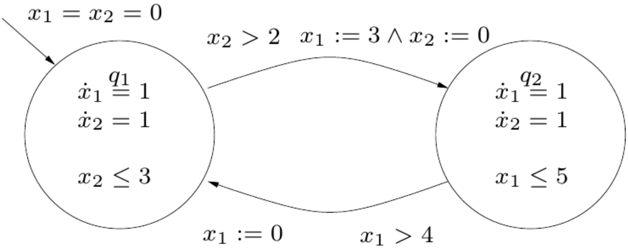
Restriction

- All derivations to drive continuous evolution of the automaton has the form of:
$$\frac{dx_i(t)}{dt} = 1$$
- Resets R of discrete transitions are allowed either to keep the value of the continuous variable, or to reset it to 0.
- *Dom* and G are defined only using relations $\leq$ and $\geq$ with respect to integral values.

Intuition

- Finite automaton with a set of continuous variables to measure elapsed time.
- Measured time values may be reset to 0 using discrete transition.

Example of Timed Automaton



Exercise

- In two-dimensional graph with axes x_1 and x_2 show how the values of continuous variables change.

Region Abstraction

Key Observation

- With respect to the restriction that comparisons are made only against integral values, two floating point values that have the same integral part cannot be differentiated.

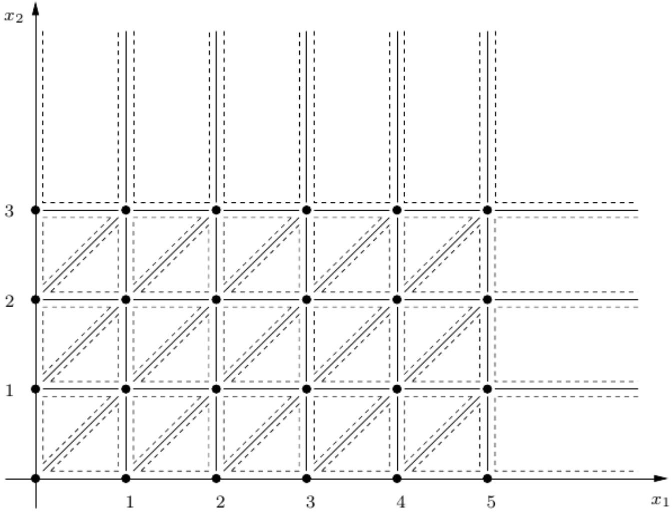
Equivalence Classes on the Continuous Domain

- If c is the greatest integral number used in a guard of timed automaton then the continuous domain can be represented with a finite set of intervals as follows:

$$[0], (0,1), [1], (1,2), [2], \dots [c-1], (c-1,c), [c], (c,\infty)$$

- Abstracted domain is finite for every continuous variable.
- It is possible to construct finite-state automaton that faithfully simulates behaviour of the timed automaton.
- This can be used for verification purposes.

Region Abstraction



Rectangular Hybrid Automata (RHA)

Restriction

- All derivations to drive continuous evolution of the automaton has the form of:
$$a \leq \frac{dx_i(t)}{dt} \leq b,$$
where a and b are rational constants.
- When specifying RHA no derivation equations are given, just the boundary constants a and b.

Exercise

- Consider a RHA with two continuous variables x_1 and x_2.
- On two-dimensional graph with axes x_1 and x_2 show the evolution of values of the continuous variables.
- Guess the origin of the name of this particular sub-class of hybrid automata.

Reachability is Decidable for RHA

Reachability

- Reachability problem for RHA is decidable if there are only finitely many values to which a continuous variable may be reset by a discrete transition.
- The most general sub-class of hybrid automata for which reachability is still decidable.

Going Beyond Means Undecidability

- Relaxation from restriction of resets is known to result in sub-class of hybrid automata for which the reachability problem is undecidable.

Definition

- Let $k_0, \ldots, k_m$ be numeric constants and $x_1, \ldots, x_m$ variables. An expression in the form of $k_0 + k_1x_1 + k_2x_2 + \cdots + k_mx_m$ is called a *linear expression*.
- Let t_1, t_2 be linear expressions. An expression of the form $t_1 \leq t_2$ is called linear inequality.
- Hybrid automaton H is called **linear hybrid automaton** (LHA), if *Init*, *Dom*, G and f are defined as Boolean combinations of linear inequalities.

Undecidability

- The reachability problem for LHA is undecidable.
- Algorithms implemented for the LHA sub-class are incomplete (HyTech).

Homework

- Will Lake Mead go dry? (SPACEex tool).
  `http://spaceex.imag.fr/documentation/tutorials`