Hybrid Automata

Lecturer: Tiziano Villa[1]

[1]Dipartimento d'Informatica
Università di Verona
tiziano.villa@univr.it

Thanks to Carla Piazza, Dipartimento di Matematica ed Informatica, Università di Udine

Motivation

We will consider:

AUTOMATA

with an INFINITE number of STATES

Motivation

We will discuss:

the SPECIFICATION and ANALYSIS

of systems involving variables either

DISCRETE or CONTINUOUS

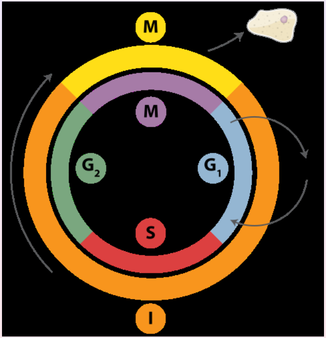
Hybrid Systems

Many real systems have a double nature. They:

- evolve in a continuous fashion
- are controlled by a discrete system

Such systems are called hybrid systems and may be modeled by hybrid automata

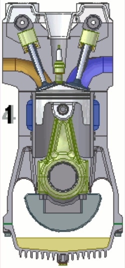
Example: Cell Cycle



- I (interphase): the cell grows cumulating nutrients needed for duplication. It contains the subphases G_1 (growth), S (DNA synthesis), G_2 (growth)
- M (mitosis): the chromosomes in the nucleus split to yield two nuclei.

It is a growth process genetically controlled

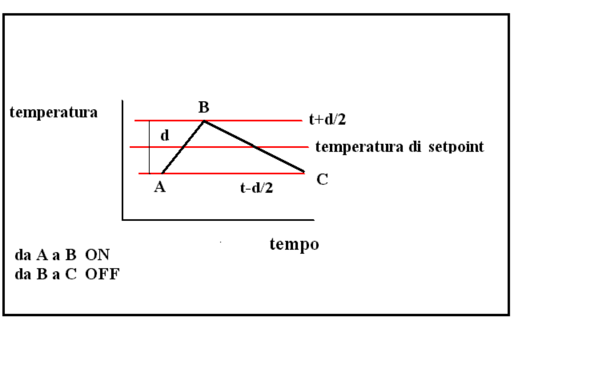
Example: 4-Strokes Engine



- Intake stroke: air and vaporized fuel are drawn in
- Compression stroke: fuel vapor and air are compressed and ignited
- Combustion stroke: fuel combusts and piston is pushed downwards
- Exhaust/Emission stroke: exhaust is driven out
- During 1st, 2nd and 4th stroke the piston is relying on the power and momentum generated by the pistons of the other cylinders

During the 4 strokes pression, temperature, . . .
vary continuously

Example: Thermostat



It is a switch controlled by a variation of temperature.
The first thermostat credited to the Scottish chemist Andrew Ure in 1830

Topics of the Lectures

- Hybrid Automata: syntax and semantics
- Finite State Systems (brief refresh)
- The Reachability problem
- Results of Undecidability
- Important Classes of hybrid automata: timed, rectangular, o-minimal, . . .
- Decidabily techniques: (Bi)Simulation, Cylindric Algebraic Decomposition, . . .
- Software Tools

Today's Topic

- Hybrid Automata: Syntax and Semantics
- Sistemi a stati finiti (breve ripasso)
- The problem of Reachability
- Results of Undecidability
- Classi notevoli di Automi Ibridi: timed, rectangular, o-minimal, . . .
- Tecniche di Decisione: (Bi)Simulazione, Cylindric Algebraic Decomposition, Teoremi di Selezione, Semantiche approssimate
- . . . e tanto altro:
  - Logiche temporali
  - Composizione di Automi
  - Il caso Stocastico
  - Stabilità, Osservabilità, Controllabilità
  - Strumenti Software
  - Applicazioni

Historical Background

- Computer scientists developed Classical Automata Theory, Temporal Logics, Model Checking for the analysis and synthesis of finite systems
- Engineers, mathematicians and physicists investigated Dynamical Systems and Control Theory for the analysis and synthesis of continuous control systems
- In the 90s, computer scientists and control specialists started to study hybrid systems with discrete and continuous features
- Some computer scientists proposed the model of Hybrid Automata (e.g., Alur, Courcobetis, Dill, Henzinger, Sifakis, and many more)

Historical Background

- Computer scientists developed Classical Automata Theory, Temporal Logics, Model Checking for the analysis and synthesis of finite systems
- Engineers, mathematicians and physicists investigated Dynamical Systems and Control Theory for the analysis and synthesis of continuous control systems
- In the 90s, computer scientists and control specialists started to study hybrid systems with discrete and continuous features
- Some computer scientists proposed the model of Hybrid Automata (e.g., Alur, Courcobetis, Dill, Henzinger, Sifakis, and many more)

Historical Background

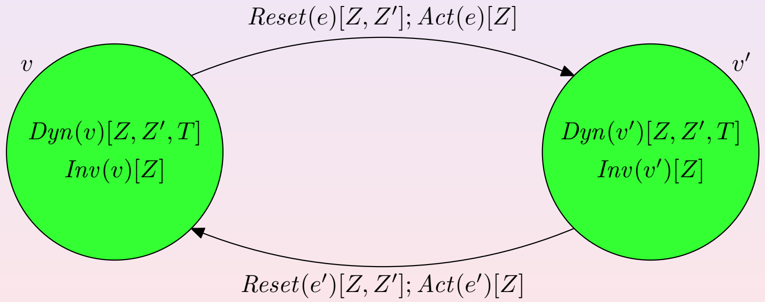
- Computer scientists developed Classical Automata Theory, Temporal Logics, Model Checking for the analysis and synthesis of finite systems
- Engineers, mathematicians and physicists investigated Dynamical Systems and Control Theory for the analysis and synthesis of continuous control systems
- In the 90s, computer scientists and control specialists started to study hybrid systems with discrete and continuous features
- Some computer scientists proposed the model of Hybrid Automata (e.g., Alur, Courcobetis, Dill, Henzinger, Sifakis, and many more)

Historical Background

- Computer scientists developed Classical Automata Theory, Temporal Logics, Model Checking for the analysis and synthesis of finite systems
- Engineers, mathematicians and physicists investigated Dynamical Systems and Control Theory for the analysis and synthesis of continuous control systems
- In the 90s, computer scientists and control specialists started to study hybrid systems with discrete and continuous features
- Some computer scientists proposed the model of Hybrid Automata (e.g., Alur, Courcobetis, Dill, Henzinger, Sifakis, and many more)

Hybrid Automata - The Intuition

An hybrid automaton *H* is
a finite-state automaton with continuous variables *Z*



A state is a couple $\langle v, r \rangle$ where *r* is a valuation for *Z*

Hybrid Automata - Syntax

Definition (Hybrid Automata (Piazza et al.))

A k-hybrid automaton $H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ consists of the following components:

1. $Z = (Z_1, \dots, Z_k)$ and $Z' = (Z'_1, \dots, Z'_k)$ are two vectors of variables ranging over the reals;
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ is a finite directed graph;
3. Each $v \in \mathcal{V}$ is labeled by the two formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T]$ such that if $Inv(v)[p]$ holds then $Dyn(v)[p, p, 0]$ holds as well;
4. Each $e \in \mathcal{E}$ is labeled by the formulæ $Act(e)[Z]$ and $Reset(e)[Z, Z']$.

Hybrid Automata - Syntax

Definition (Hybrid Automata (Piazza et al.))

A k-hybrid automaton $H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ consists of the following components:

1. $Z = (Z_1, \ldots, Z_k)$ and $Z' = (Z'_1, \ldots, Z'_k)$ are two vectors of variables ranging over the reals;
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ is a finite directed graph;
3. Each $v \in \mathcal{V}$ is labeled by the two formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T]$ such that if $Inv(v)[p]$ holds then $Dyn(v)[p, p, 0]$ holds as well;
4. Each $e \in \mathcal{E}$ is labeled by the formulæ $Act(e)[Z]$ and $Reset(e)[Z, Z']$.

Hybrid Automata - Syntax

Definition (Hybrid Automata (Piazza et al.))

A k-hybrid automaton $H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ consists of the following components:

1. $Z = (Z_1, \ldots, Z_k)$ and $Z' = (Z'_1, \ldots, Z'_k)$ are two vectors of variables ranging over the reals;
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ is a finite directed graph;
3. Each $v \in \mathcal{V}$ is labeled by the two formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T]$ such that if $Inv(v)[p]$ holds then $Dyn(v)[p, p, 0]$ holds as well;
4. Each $e \in \mathcal{E}$ is labeled by the formulæ $Act(e)[Z]$ and $Reset(e)[Z, Z']$.

Hybrid Automata - Syntax

Definition (Hybrid Automata (Piazza et al.))

A k-hybrid automaton $H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ consists of the following components:

1. $Z = (Z_1, \ldots, Z_k)$ and $Z' = (Z'_1, \ldots, Z'_k)$ are two vectors of variables ranging over the reals;
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ is a finite directed graph;
3. Each $v \in \mathcal{V}$ is labeled by the two formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T]$ such that if $Inv(v)[p]$ holds then $Dyn(v)[p, p, 0]$ holds as well;
4. Each $e \in \mathcal{E}$ is labeled by the formulæ $Act(e)[Z]$ and $Reset(e)[Z, Z']$.

Hybrid Automata - Syntax

Definition (Hybrid Automata (Piazza et al.))

A k-hybrid automaton $H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ consists of the following components:

1. $Z = (Z_1, \dots, Z_k)$ and $Z' = (Z'_1, \dots, Z'_k)$ are two vectors of variables ranging over the reals;
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ is a finite directed graph;
3. Each $v \in \mathcal{V}$ is labeled by the two formulæ $Inv(v)[Z]$ and $Dyn(v)[Z, Z', T]$ such that if $Inv(v)[p]$ holds then $Dyn(v)[p, p, 0]$ holds as well;
4. Each $e \in \mathcal{E}$ is labeled by the formulæ $Act(e)[Z]$ and $Reset(e)[Z, Z']$.

Comments on the Definition

- *Inv*, *Dyn*, *Act*, *Reset* are sets of formulae in a first-order language $\mathcal{L}$
  E.g., $\mathcal{L} = (+, *, <, 0, 1)$
- the formulae are evaluated over a model $\mathcal{M}$ of $\mathcal{L}$ in the domain $\mathbb{R}$
  E.g., $\mathcal{M} = (\mathbb{R}, +, *, <, 0, 1)$
- the nodes $\mathcal{V}$ are called locations (or *control modes*), the arcs $\mathcal{E}$ are called control switches
- the variable T represents time
- $p \in \mathbb{R}^k$

Comments on the Definition

- *Inv*, *Dyn*, *Act*, *Reset* are sets of formulae in a first-order language $\mathcal{L}$
  E.g., $\mathcal{L} = (+, *, <, 0, 1)$
- the formulae are evaluated over a model $\mathcal{M}$ of $\mathcal{L}$ in the domain $\mathbb{R}$
  E.g., $\mathcal{M} = (\mathbb{R}, +, *, <, 0, 1)$
- the nodes $\mathcal{V}$ are called locations (or *control modes*), the arcs $\mathcal{E}$ are called control switches
- the variable T represents time
- $p \in \mathbb{R}^k$

Comments on the Definition

- *Inv*, *Dyn*, *Act*, *Reset* are sets of formulae in a first-order language $\mathcal{L}$
  E.g., $\mathcal{L} = (+, *, <, 0, 1)$
- the formulae are evaluated over a model $\mathcal{M}$ of $\mathcal{L}$ in the domain $\mathbb{R}$
  E.g., $\mathcal{M} = (\mathbb{R}, +, *, <, 0, 1)$
- the nodes $\mathcal{V}$ are called locations (or *control modes*), the arcs $\mathcal{E}$ are called control switches
- the variable T represents time
- $p \in \mathbb{R}^k$

Comments on the Definition

- *Inv*, *Dyn*, *Act*, *Reset* are sets of formulae in a first-order language $\mathcal{L}$
  E.g., $\mathcal{L} = (+, *, <, 0, 1)$
- the formulae are evaluated over a model $\mathcal{M}$ of $\mathcal{L}$ in the domain $\mathbb{R}$
  E.g., $\mathcal{M} = (\mathbb{R}, +, *, <, 0, 1)$
- the nodes $\mathcal{V}$ are called locations (or *control modes*), the arcs $\mathcal{E}$ are called control switches
- the variable T represents time
- $p \in \mathbb{R}^k$

Comments on the Definition

- *Inv*, *Dyn*, *Act*, *Reset* are sets of formulae in a first-order language $\mathcal{L}$
  E.g., $\mathcal{L} = (+, *, <, 0, 1)$
- the formulae are evaluated over a model $\mathcal{M}$ of $\mathcal{L}$ in the domain $\mathbb{R}$
  E.g., $\mathcal{M} = (\mathbb{R}, +, *, <, 0, 1)$
- the nodes $\mathcal{V}$ are called locations (or *control modes*), the arcs $\mathcal{E}$ are called control switches
- the variable T represents time
- $p \in \mathbb{R}^k$

An Example: Thermostat

Example (Thermostat)

Let us consider a room heated by a radiator controlled by a thermostat

- When the thermostat is on the temperature increases exponentially in time
- When the thermostat is off the temperature decreases exponentially in time
- The thermostat switches on the radiator when the temperature decreases below $19C$
- The thermostat switches off the radiator when the temperature increases above $21C$

An Example: Thermostat

Let us model the behaviour of the temperature in time by an hybrid automaton H with:

- 2 locations ON and OFF
- 2 arcs that join the two locations
- 1 continuous variable Z that represents the temperature

An Example: Thermostat

$H = \langle Z, Z', \mathcal{V}, \mathcal{E}, Inv, Dyn, Act, Reset \rangle$ such that:

- Z e Z' are two variables
- $\mathcal{V} = \{ON, OFF\}$ and $\mathcal{E} = \{(ON, OFF), (OFF, ON)\}$
- $Inv(ON)[Z] := Z \leq 22$ and
  $Dyn(ON)[Z, Z', T] := Z' = Z * e^T$
- $Inv(OFF)[Z] := Z \geq 18$ and
  $Dyn(OFF)[Z, Z', T] := Z' = Z/e^T$
- $Act((ON, OFF))[Z] := Z \geq 21$ and
  $Reset((ON, OFF))[Z, Z'] := Z' = Z$
- $Act((OFF, ON))[Z] := Z \leq 19$ and
  $Reset((OFF, ON))[Z, Z'] := Z' = Z$

...it is better to draw it on the blackboard or on paper

Hybrid Automata - Definitions of Syntax from Literature

T. A. Henzinger

Definition 1.1 [Hybrid automata] [5, 43, 3] A *hybrid automaton* H consists of the following components.

Variables. A finite set $X = \{x_1, \ldots, x_n\}$ of real-numbered variables. The number n is called the *dimension* of H. We write $\dot{X}$ for the set $\{\dot{x}_1, \ldots, \dot{x}_n\}$ of dotted variables (which represent first derivatives during continuous change), and we write X' for the set $\{x'_1, \ldots, x'_n\}$ of primed variables (which represent values at the conclusion of discrete change).

Control graph. A finite directed multigraph (V, E). The vertices in V are called *control modes*. The edges in E are called *control switches*.

Initial, invariant, and flow conditions. Three vertex labeling functions *init*, *inv*, and *flow* that assign to each control mode $v \in V$ three predicates. Each initial condition $init(v)$ is a predicate whose free variables are from X. Each invariant condition $inv(v)$ is a predicate whose free variables are from X. Each flow condition $flow(v)$ is a predicate whose free variables are from $X \cup \dot{X}$.

Jump conditions. An edge labeling function *jump* that assigns to each control switch $e \in E$ a predicate. Each jump condition $jump(e)$ is a predicate whose free variables are from $X \cup X'$.

Events. A finite set Σ of events, and an edge labeling function $event: E \to \Sigma$ that assigns to each control switch an event. □

Hybrid Automata - Definitions of Syntax from Literature

J. Lygeros et al.

Definition 3.1 (Hybrid Automaton) *A* hybrid automaton H *is a collection* $H = (Q, X, \text{Init}, f, I, E, G, R)$, *where*

- Q *is a set of discrete variables and* Q *is countable;*
- X *is a set of continuous variables;*
- $\text{Init} \subseteq \mathbf{Q} \times \mathbf{X}$ *is a set of initial states;*
- $f : \mathbf{Q} \times \mathbf{X} \to T\mathbf{X}$ *is a vector field;*
- $\text{Inv} : \mathbf{Q} \to P(X) := 2^{\mathbf{X}}$ *assigns to each* $q \in \mathbf{Q}$ *an invariant set;*
- $E \subset \mathbf{Q} \times \mathbf{Q}$ *is a collection of discrete transitions;*
- $G : E \to P(X)$ *assigns to each* $e = (q, q') \in E$ *a guard; and*
- $R : E \times \mathbf{X} \to P(X)$ *assigns to each* $e = (q, q') \in E$ *and* $x \in \mathbf{X}$ *a reset relation.*

Why . . .

. . . in the proposed definition there are no differential equations?

- to be more general allowing any kind of solvable/approximable equations
- to avoid making differential equations the only culprits of undecidability and complexity results

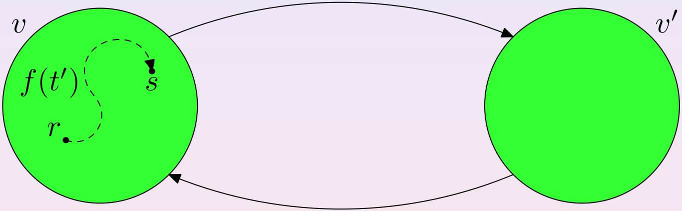
Why . . .

. . . in the proposed definition there are no differential equations?

- to be more general allowing any kind of solvable/approximable equations
- to avoid making differential equations the only culprits of undecidability and complexity results

Hybrid Automata - Semantics

$\ell = \langle v, r \rangle$ is *admissible* if $Inv(v)[r]$ holds

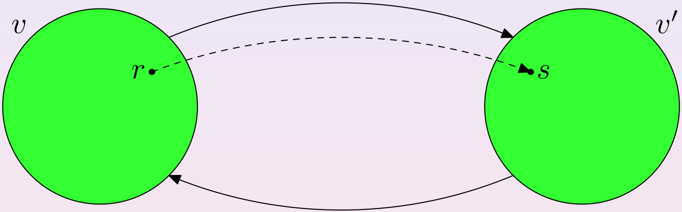


Definition (Continuous Transitions)

$\langle v, r \rangle \xrightarrow{t}_C \langle v, s \rangle \iff$ There exists a continuous function $f : \mathbb{R}^+ \mapsto \mathbb{R}^k$ such that $r = f(0)$, $s = f(t)$ and for each $t' \in [0, t]$ the formulæ $Inv(v)[f(t')]$ and $Dyn(v)[r, f(t'), t']$ hold

Hybrid Automata - Semantics

$\ell = \langle v, r \rangle$ is *admissible* if $Inv(v)[r]$ holds



Definition (Discrete Transitions)

$\langle v, r \rangle \xrightarrow{\langle v, v' \rangle}_D \langle v', s \rangle \iff$ $\langle v, v' \rangle \in \mathcal{E}$, $Inv(v)[r]$, $Act(\langle v, v' \rangle)[r]$, $Reset(\langle v, v' \rangle)[r, s]$ and $Inv(v')[s]$ hold

Comments on the Definition

- As a fact, we defined an infinite graph with two types of arcs

$$(\mathcal{V} \times \mathbb{R}^k, \xrightarrow{\langle\_,\_\rangle}_D, \xrightarrow{\_}_C)$$

- Could I have been more precise ?
  I could have recorded explicitly also the continuous function f
- Could I have been less precise ?
  I could have considered only one type of arcs

$$\rightarrow = \xrightarrow{\langle\_,\_\rangle}_D \cup \xrightarrow{\_}_C$$

untimed semantics

Comments on the Definition

- As a fact, we defined an infinite graph with two types of arcs

$$(\mathcal{V} \times \mathbb{R}^k, \xrightarrow{\langle \_,\_ \rangle}_D, \xrightarrow{\_}_C)$$

- Could I have been more precise ?
  I could have recorded explicitly also the continuous function f
- Could I have been less precise ?
  I could have considered only one type of arcs

$$\rightarrow = \xrightarrow{\langle \_,\_ \rangle}_D \cup \xrightarrow{\_}_C$$

untimed semantics

Comments on the Definition

- As a fact, we defined an infinite graph with two types of arcs

$$(\mathcal{V} \times \mathbb{R}^k, \xrightarrow{\langle \_,\_ \rangle}_D, \xrightarrow{\_}_C)$$

- Could I have been more precise ?
  I could have recorded explicitly also the continuous function f
- Could I have been less precise ?
  I could have considered only one type of arcs

$$\rightarrow = \xrightarrow{\langle \_,\_ \rangle}_D \cup \xrightarrow{\_}_C$$

untimed semantics

Comments on the Definition

- As a fact, we defined an infinite graph with two types of arcs

$$(\mathcal{V} \times \mathbb{R}^k, \xrightarrow{\langle -,- \rangle}_D, \xrightarrow{-}_C)$$

- Could I have been more precise ?
  I could have recorded explicitly also the continuous function f
- Could I have been less precise ?
  I could have considered only one type of arcs

$$\rightarrow = \xrightarrow{\langle -,- \rangle}_D \cup \xrightarrow{-}_C$$

untimed semantics

Comments on the Definition

- As a fact, we defined an infinite graph with two types of arcs

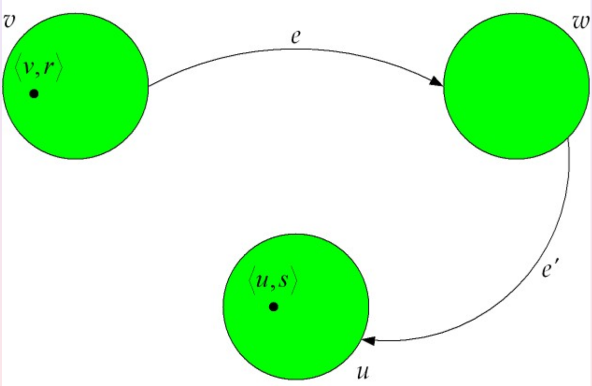
$$(\mathcal{V} \times \mathbb{R}^k, \xrightarrow{\langle\_,\_\rangle}_D, \xrightarrow{\_}_C)$$

- Could I have been more precise ?
  I could have recorded explicitly also the continuous function f
- Could I have been less precise ?
  I could have considered only one type of arcs

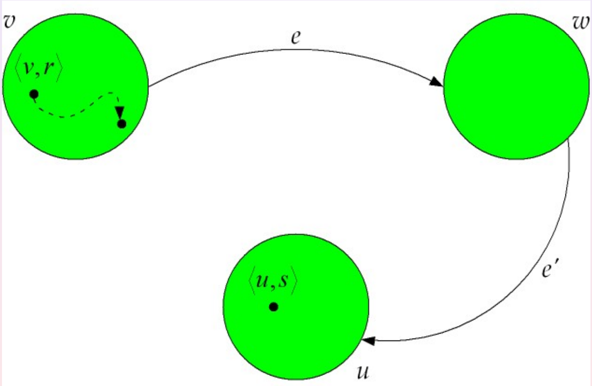
$$\rightarrow = \xrightarrow{\langle\_,\_\rangle}_D \cup \xrightarrow{\_}_C$$

untimed semantics

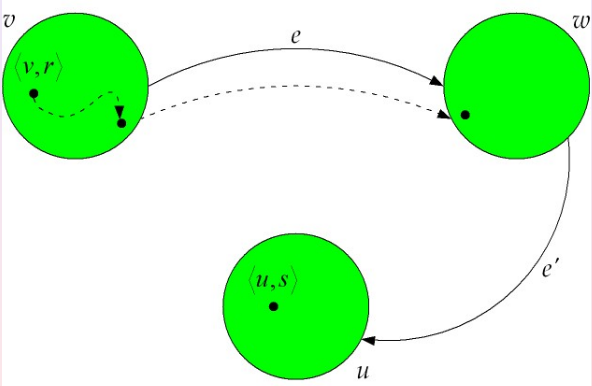
Hybrid Automata - Reachability



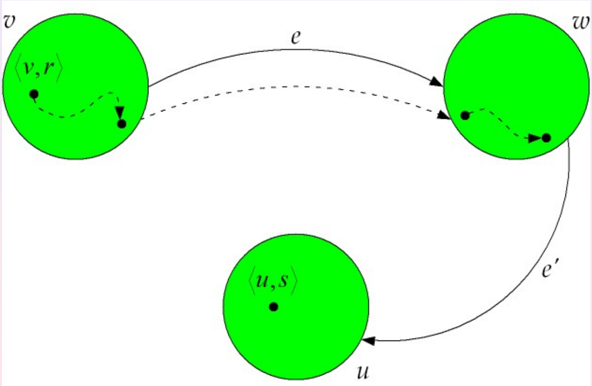
Hybrid Automata - Reachability



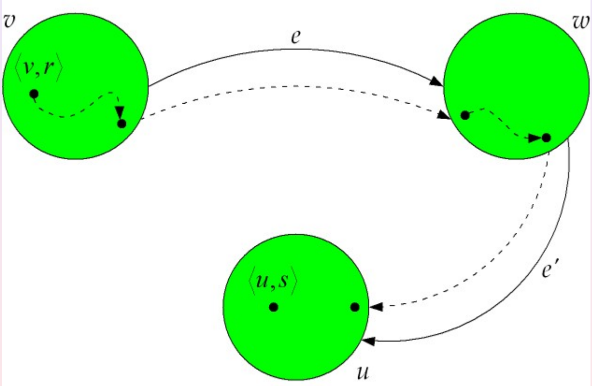
Hybrid Automata - Reachability



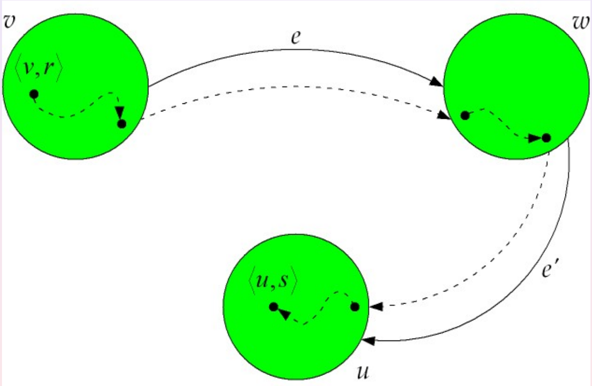
Hybrid Automata - Reachability



Hybrid Automata - Reachability

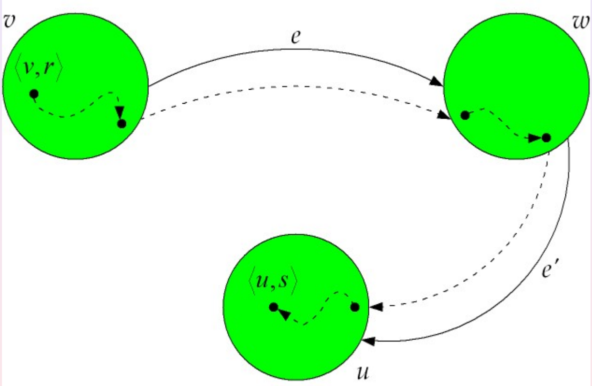


Hybrid Automata - Reachability



?

Hybrid Automata - Reachability



Let $I, F \in \mathbb{R}^k$. Can we reach $\langle u, F\rangle$ from $\langle v, I\rangle$?

Trace and Reachability

A trace of H is a sequence of admissible states $[\ell_0, \ell_1, \ldots, \ell_i, \ldots, \ell_n]$ such that $\ell_{i-1} \rightarrow \ell_i$ holds $\forall i \in [1, n]$.

Definition (Reachability)

The automaton H reaches $\langle u, s \rangle$, $s \in \mathbb{R}^k$, from $\langle v, r \rangle$, $r \in \mathbb{R}^k$, if there exists a trace $tr = [\ell_0, \ldots, \ell_n]$ of H such that $\ell_0 = \langle v, r \rangle$ and $\ell_n = \langle u, s \rangle$.

Definition (Reachability Problem)

Given an automaton H, a set of starting points $\langle v, I \rangle$, $I \subseteq \mathbb{R}^k$, and a set of ending points $\langle u, F \rangle$, $F \subseteq \mathbb{R}^k$, decide whether there exists a point in $\langle v, I \rangle$ from which a point in $\langle u, F \rangle$ is reachable.

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

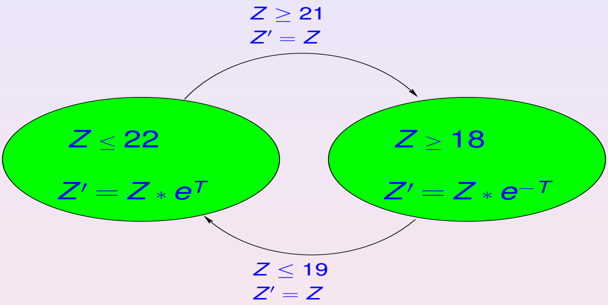
- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Many Sources of Non-Determinism

Hybrid automata may be non-deterministic since:

- Different locations may partially share the invariants
- Different continuous trajectories may leave from the same admissible state
- There may be arcs that go to different locations but partially share the activation functions
- The activation functions are not necessarily on the frontiers of the invariants
- The reset functions are not necessarily deterministic

Example: Thermostat



- $\langle ON, 15\rangle \xrightarrow{0.1}_C \langle ON, 16.57\rangle \xrightarrow{0.25}_C \langle ON, 21.28\rangle \xrightarrow{\langle ON, OFF\rangle}_D \langle OFF, 21.28\rangle \ldots$
- $\langle ON, 15\rangle \xrightarrow{0.35}_C \langle ON, 21.28\rangle \xrightarrow{\langle ON, OFF\rangle}_D \langle OFF, 21.28\rangle \ldots$
- $\langle OFF, 18.5\rangle \xrightarrow{\langle OFF, ON\rangle}_D \langle ON, 18.5\rangle \ldots$
- $\langle OFF, 18.5\rangle \xrightarrow{0.01}_C \langle OFF, 18.31\rangle \xrightarrow{\langle OFF, ON\rangle}_D \langle ON, 18.31\rangle \ldots$

Example: Thermostat

Observe that:

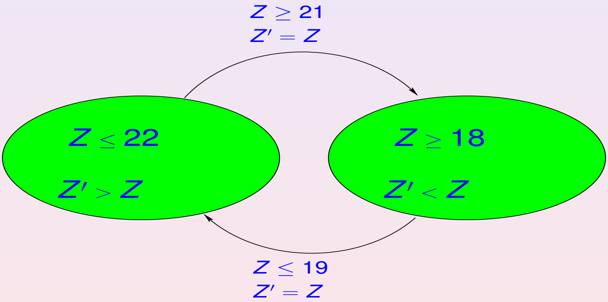
- From every point leaves an infinite number of trajectories
- Some of them are substantially ”equivalent”
- Some are not !

Example: Thermostat

What model could I have built with less information ?

Example: Thermostat

What model could I have built with less information ?



This one has more traces than the previous one!

References (from which to start)

- Automata on Infinite Objects.
  W. Thomas. Handbook of TCS 1990.
- A Theory of Timed Automata.
  R. Alur and D. Dill. TCS 1994.
- Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems.
  R. Alur et al. HS 1993.
- The Theory of Hybrid Automata.
  T. A. Henzinger. LICS 1996.
- Hybrid Systems: Modeling, Analysis and Control.
  J. Lygeros, C. Tomlin, and S. Sastry. 2008.