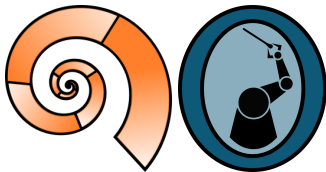


Hybrid automata and the reachability problem

Luca Geretti

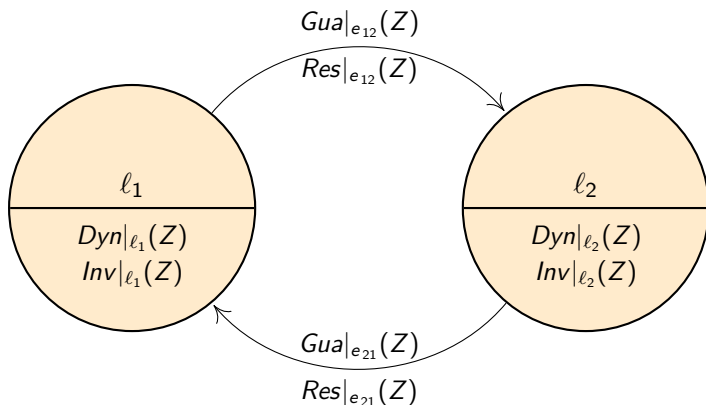
University of Verona, Italy



The intuition



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



A **state** is a couple (ℓ, r) where r is a valuation for Z

The role of functions in hybrid automata



- **dynamics** $Dyn|_e$: evolution of the variables in location ℓ
- **invariant** $Inv|_e$: conditions under which continuous evolution is allowed in location ℓ
- **guard** $Gua|_e$: conditions under which discrete evolution is allowed according to event e
- **reset** $Res|_e$: transformation of the continuous state after event e

Non-determinism



Hybrid automata may be non-deterministic since:

Non-determinism



Hybrid automata may be non-deterministic since:

- Different **locations** may partially share the **invariants**



Hybrid automata may be non-deterministic since:

- Different **locations** may partially share the **invariants**
- Different continuous **trajectories** may leave from the same admissible state



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



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



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



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
- The **dynamics** may include uncertainties



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
- The **dynamics** may include uncertainties

Probabilistic generalisations add other levels of complexity:

- Discrete transitions form a **Markov process**
- **Stochastic equations** add probability to the continuous dynamics



We are not restricted to specifying the system as one "monolithic" automaton: we can have **multiple automata** that interact between each other

- **Compositionality** refers to the capability of such interaction in a coherent way



We are not restricted to specifying the system as one "monolithic" automaton: we can have **multiple automata** that interact between each other

- **Compositionality** refers to the capability of such interaction in a coherent way
- Interaction may be through input/output variables or input/output events
 - ▶ **Input variable**: a variable for which the automaton does not specify dynamics
 - ▶ **Input event**: an event for which the activation does not depend on the automaton variables
 - ▶ In a simplified approach, everything that is not an input is an output for the automaton



We are not restricted to specifying the system as one "monolithic" automaton: we can have **multiple automata** that interact between each other

- **Compositionality** refers to the capability of such interaction in a coherent way
- Interaction may be through input/output variables or input/output events
 - ▶ **Input variable**: a variable for which the automaton does not specify dynamics
 - ▶ **Input event**: an event for which the activation does not depend on the automaton variables
 - ▶ In a simplified approach, everything that is not an input is an output for the automaton
- Composition combines multiple automata in order to construct (statically or dynamically) the fully specified system
 - ▶ Example: the soil automaton has its own dynamics, but it also reacts to the input coming from the weather automaton



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 19°C
- The thermostat switches off the radiator when the temperature increases above 21°C

Example: thermostat



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

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

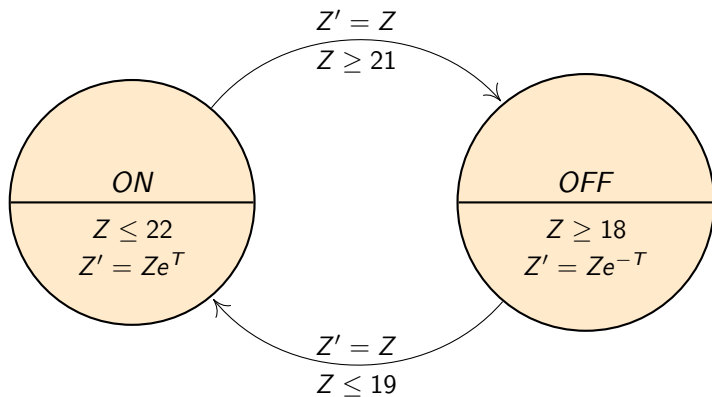
Example: thermostat



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

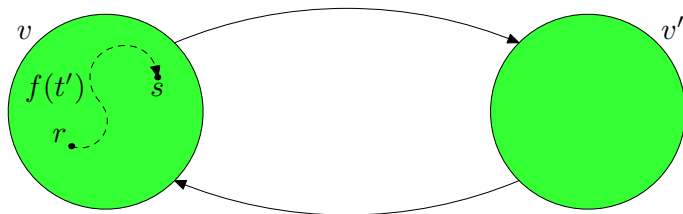
- Z e Z' are two variables
- $\mathcal{V} = \{ON, OFF\}$ and $\mathcal{E} = \{(ON, OFF), (OFF, ON)\}$
- $Inv(ON)[Z] := Z \leq 22$ and
 $Dyn(ON)[Z, Z', T] := Z' = Z * e^T$
- $Inv(OFF)[Z] := Z \geq 18$ and
 $Dyn(OFF)[Z, Z', T] := Z' = Z / e^T$
- $Act((ON, OFF))[Z] := Z \geq 21$ and
 $Reset((ON, OFF))[Z,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

Example: thermostat





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



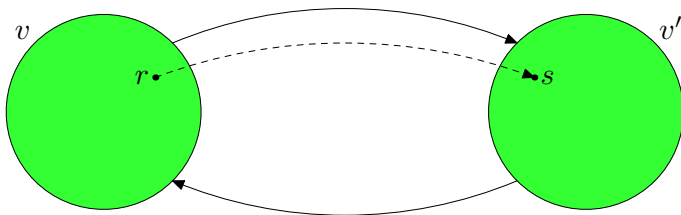
Definition (Continuous transitions)

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

$$\langle v, r \rangle \xrightarrow{t}_C \langle v, s \rangle \iff$$

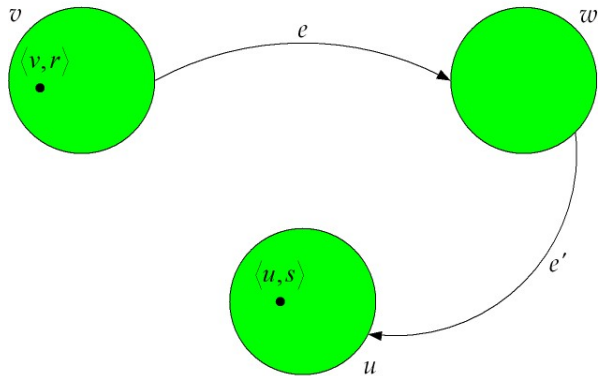


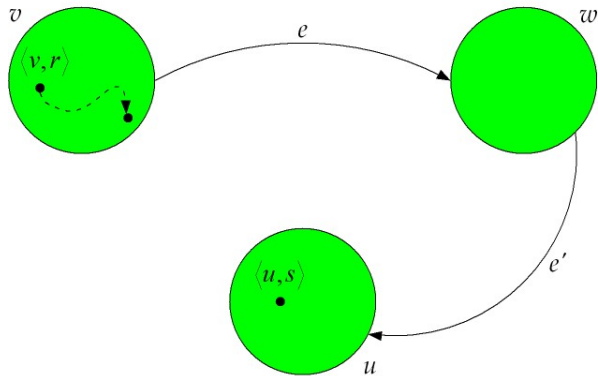
$\ell = \langle v, r \rangle$ is **admissible** if $Inv(v)[r]$ holds

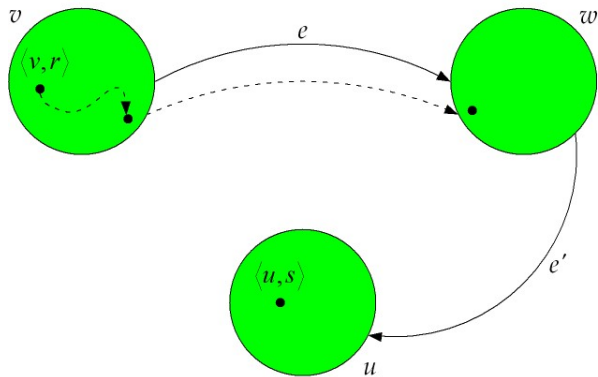


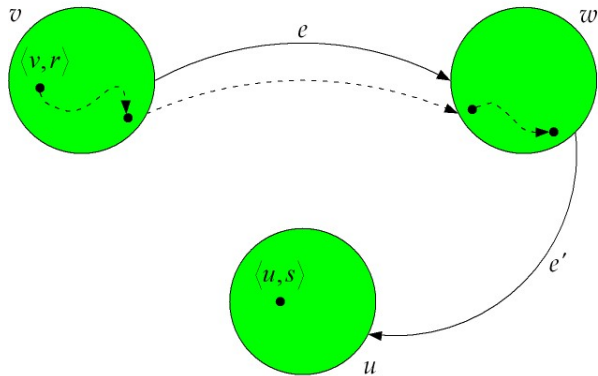
Definition (Discrete transitions)

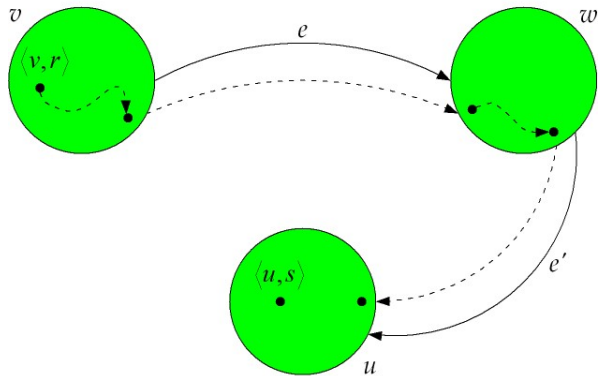
$$\langle v, r \rangle \xrightarrow{\langle v, v' \rangle} \rightarrow_D \langle v', s \rangle \iff \begin{array}{l} \langle v, v' \rangle \in \mathcal{E}, \quad Inv(v)[r], \\ Act(\langle v, v' \rangle)[r], \\ Reset(\langle v, v' \rangle)[r, s] \quad \text{and} \\ Inv(v')[s] \text{ hold} \end{array}$$



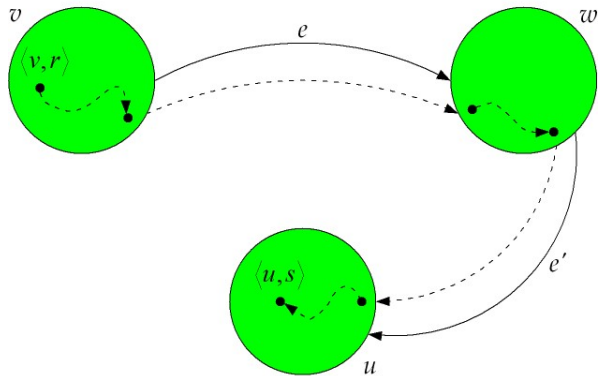


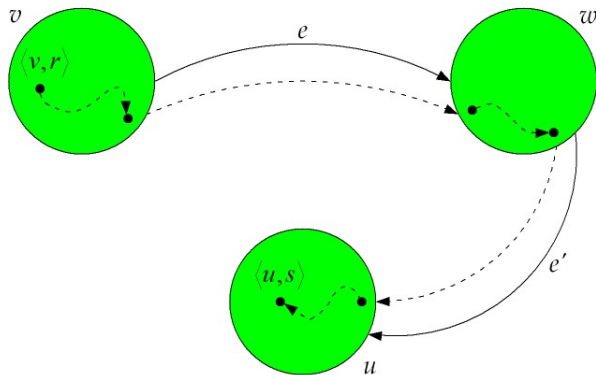






Reachability





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



A **trace** of H is a sequence of admissible states $[\ell_0, \ell_1, \dots, \ell_i, \dots, \ell_n]$ 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 $tr = [\ell_0, \dots, \ell_n]$ 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.



Reachability can be used for example to verify safety properties

φ is always true in H
if and only if
all states reachable from the initial states of H
are included in the safe set $Sat(\varphi)$.



Reachability can be used for example to verify safety properties

φ is always true in H

if and only if

all states reachable from the initial states of H
are included in the safe set $Sat(\varphi)$.

Question

Is the reachability problem for Hybrid Automata **decidable**?



Reachability can be used for example to verify safety properties

φ is always true in H
if and only if

all states reachable from the initial states of H
are included in the safe set $Sat(\varphi)$.

Question

Is the reachability problem for Hybrid Automata **decidable**?

Answer

No (Alur et al. 1995).



If we restrict the possible behaviors of the automaton, decidability can be recovered.

- Restrictions mainly involve the dynamics, in the following expressed by the derivatives \dot{x} of the continuous variables.
- Other restrictions may be placed on the transitions, in particular the reset value x' .



If we restrict the possible behaviors of the automaton, decidability can be recovered.

- Restrictions mainly involve the dynamics, in the following expressed by the derivatives \dot{x} of the continuous variables.
- Other restrictions may be placed on the transitions, in particular the reset value x' .

We also distinguish decidability for **time-bounded** reachability and **time-unbounded** reachability (also called finite-time and infinite-time reachabilities).

Recover decidability by reducing expressivity

A table



automata	derivatives	conditions	bounded	unbounded
timed	$\dot{x} = 1$	$x' = x$ or $x' = 0$	Yes	Yes
initialized rectangular	$\dot{x} \in [k_1, k_2]$	$x' \in [c_1, c_2]$ when \dot{x} changes otherwise $x' = x$	Yes	Yes
rectangular	$\dot{x} \in [k_1, k_2]$	$x' \in [c_1, c_2]$ or $x' = x$	Yes	No
linear	$\dot{x} = c$		Yes	No
affine	$\dot{x} = Ax + b$		No	No
nonlinear	$\dot{x} = f(x)$		No	No