

# Decidability Results for Probabilistic Hybrid Automata

Prof. Dr. Erika Ábrahám

Informatik 2 - Theory of Hybrid Systems  
RWTH Aachen

SS09

Jeremy Sproston:

Decidable Model Checking of Probabilistic Hybrid Automata

FTRTFT'00, LNCS 1926, pp. 31–45, 2000.

Which components of a hybrid system could be probabilistic?

Which components of a hybrid system could be probabilistic?

Example applications?

Which components of a hybrid system could be probabilistic?

Example applications?

What do you expect to be a decidable class?

## Definition (Distribution)

- For a set  $Y$ , a (discrete probability) **distribution on  $Y$**  is a function  $\mu : Y \rightarrow [0, 1]$  such that  $\mu(y) > 0$  for at most countably many  $y \in Y$  and  $\sum_{y \in Y} \mu(y) = 1$ .

## Definition (Distribution)

- For a set  $Y$ , a (discrete probability) **distribution on  $Y$**  is a function  $\mu : Y \rightarrow [0, 1]$  such that  $\mu(y) > 0$  for at most countably many  $y \in Y$  and  $\sum_{y \in Y} \mu(y) = 1$ .
- We use  **$\text{Dist}(Y)$**  to denote the set of all distributions on  $Y$ .

## Definition (Distribution)

- For a set  $Y$ , a (discrete probability) **distribution on  $Y$**  is a function  $\mu : Y \rightarrow [0, 1]$  such that  $\mu(y) > 0$  for at most countably many  $y \in Y$  and  $\sum_{y \in Y} \mu(y) = 1$ .
- We use  **$\text{Dist}(Y)$**  to denote the set of all distributions on  $Y$ .
- For a distribution  $\mu$  on a set  $Y$  let  **$\text{support}(\mu)$**  be the set of elements  $y$  of  $Y$  with  $\mu(y) > 0$ .



How can probabilities and conditional jumps with effects be combined?

# Probabilistic hybrid automata

(We skip synchronization labels.)

## Definition (Probabilistic hybrid automaton)

A **probabilistic hybrid automaton**  $H$  is a hybrid automaton without the Edge set, and with

- an additional function **prob** which maps to each location a finite set of probability distributions on  $\text{Loc} \times 2^{\mathbb{R}^n} \times 2^{\text{Var}}$ , and

# Probabilistic hybrid automata

(We skip synchronization labels.)

## Definition (Probabilistic hybrid automaton)

A **probabilistic hybrid automaton**  $H$  is a hybrid automaton without the Edge set, and with

- an additional function **prob** which maps to each location a finite set of probability distributions on  $\text{Loc} \times 2^{\mathbb{R}^n} \times 2^{\text{Var}}$ , and
- a function **pre** which maps to each location  $l$  and each distribution in  $\text{prob}(l)$  a subset of  $\mathbb{R}^n$ , called the **precondition set**.

# Probabilistic hybrid automata

(We skip synchronization labels.)

## Definition (Probabilistic hybrid automaton)

A **probabilistic hybrid automaton**  $H$  is a hybrid automaton without the Edge set, and with

- an additional function **prob** which maps to each location a finite set of probability distributions on  $\text{Loc} \times 2^{\mathbb{R}^n} \times 2^{\text{Var}}$ , and
- a function **pre** which maps to each location  $l$  and each distribution in  $\text{prob}(l)$  a subset of  $\mathbb{R}^n$ , called the **precondition set**.

# Probabilistic hybrid automata

(We skip synchronization labels.)

## Definition (Probabilistic hybrid automaton)

A **probabilistic hybrid automaton**  $H$  is a hybrid automaton without the Edge set, and with

- an additional function **prob** which maps to each location a finite set of probability distributions on  $\text{Loc} \times 2^{\mathbb{R}^n} \times 2^{\text{Var}}$ , and
- a function **pre** which maps to each location  $l$  and each distribution in  $\text{prob}(l)$  a subset of  $\mathbb{R}^n$ , called the **precondition set**.

## Definition (Probabilistic rectangular automaton)

A **probabilistic rectangular automaton** is a probabilistic hybrid automaton with only rectangular sets in the definition.

- **Flows** as before.

- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff

- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff
  - there is a distribution  $\mu \in \text{prob}(l)$  such that



- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff
  - there is a distribution  $\mu \in \text{prob}(l)$  such that
  - the precondition  $\text{pre}(l)(\mu)$  is satisfied by  $\nu$ , and

- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff
  - there is a distribution  $\mu \in \text{prob}(l)$  such that
  - the precondition  $\text{pre}(l)(\mu)$  is satisfied by  $\nu$ , and
  - $\mu((l', \text{post}, X)) > 0$  for some  $\text{post} \subseteq \mathbb{R}^n$  and  $X \subseteq \text{Var}$  with

- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff
  - there is a distribution  $\mu \in \text{prob}(l)$  such that
  - the precondition  $\text{pre}(l)(\mu)$  is satisfied by  $\nu$ , and
  - $\mu((l', \text{post}, X)) > 0$  for some  $\text{post} \subseteq \mathbb{R}^n$  and  $X \subseteq \text{Var}$  with
  - $\nu' \in \text{post}$  and

- **Flows** as before.
- A **jump** can take place from a source state  $(l, \nu)$  to a target state  $(l', \nu')$  iff
  - there is a distribution  $\mu \in \text{prob}(l)$  such that
  - the precondition  $\text{pre}(l)(\mu)$  is satisfied by  $\nu$ , and
  - $\mu((l', \text{post}, X)) > 0$  for some  $\text{post} \subseteq \mathbb{R}^n$  and  $X \subseteq \text{Var}$  with
  - $\nu' \in \text{post}$  and
  - $\nu(x) = \nu'(x)$  for all  $x \in \text{Var} \setminus X$ .

What is the maximal probability of a single path?

What is the maximal probability of a single path?

What is about time divergence?

What is the maximal probability of a single path?

What is about time divergence?

What is about zeno behaviour?

# Adversaries

Intuitively, an adversary resolves all of the nondeterministic choices of a probabilistic hybrid automaton.



# Adversaries

Intuitively, an adversary resolves all of the nondeterministic choices of a probabilistic hybrid automaton.

## Definition (Adversary)

An **adversary** of a probabilistic hybrid automaton  $H$  is a function  $A$  mapping each finite path  $\omega$  with last state  $(l, \nu)$  of  $H$  to a distribution  $\mu \in \text{prob}(l)$ .

# Adversaries

Intuitively, an adversary resolves all of the nondeterministic choices of a probabilistic hybrid automaton.

## Definition (Adversary)

An **adversary** of a probabilistic hybrid automaton  $H$  is a function  $A$  mapping each finite path  $\omega$  with last state  $(l, \nu)$  of  $H$  to a distribution  $\mu \in \text{prob}(l)$ .

## Definition

An adversary  $A$  of a probabilistic hybrid automaton  $H$  is **divergent** iff for each state of  $H$  the total probability of the divergent paths under  $A$  is 1. Let  $\mathcal{A}_H$  be the set of divergent adversaries of  $H$ .

# Adversaries

Intuitively, an adversary resolves all of the nondeterministic choices of a probabilistic hybrid automaton.

## Definition (Adversary)

An **adversary** of a probabilistic hybrid automaton  $H$  is a function  $A$  mapping each finite path  $\omega$  with last state  $(l, \nu)$  of  $H$  to a distribution  $\mu \in \text{prob}(l)$ .

## Definition

An adversary  $A$  of a probabilistic hybrid automaton  $H$  is **divergent** iff for each state of  $H$  the total probability of the divergent paths under  $A$  is 1. Let  $\mathcal{A}_H$  be the set of divergent adversaries of  $H$ .

## Definition

A probabilistic hybrid automaton is **non-zero** iff it has at least one divergent adversary.

How could a logic  
arguing about timed and probabilistic behaviour  
look like?

## Definition (PTCTL Syntax)

The abstract syntax of PTCTL is as follows:

$$\Phi ::= a \mid g \mid \Phi \wedge \Phi \mid \neg\Phi \mid z.\Phi \mid P_{\sim\lambda}[\Phi\mathcal{U}\Phi]$$

with  $a$  an atomic proposition,  $g$  a clock constraint,  $z$  a formula clock,  $\sim \in \{\leq, <, >, \geq\}$ , and  $\lambda \in [0, 1]$ .

## Definition

$$\sigma, \mathcal{E} \models z.\Phi \quad \leftrightarrow \quad \sigma, \mathcal{E}[z := 0] \models \Phi$$

## Definition

$$\begin{aligned} \sigma, \mathcal{E} \models z.\Phi & \leftrightarrow \sigma, \mathcal{E}[z := 0] \models \Phi \\ \sigma, \mathcal{E} \models P_{\sim\lambda}[\Phi_1 \mathcal{U} \Phi_2] & \leftrightarrow \text{for all divergent adversaries } A \in \mathcal{A}_H, \\ & \text{the total probability of all infinite paths } \omega \\ & \text{under } A \text{ with } \omega, \mathcal{E} \models \Phi_1 \mathcal{U} \Phi_2 \text{ is } \sim \lambda. \end{aligned}$$

## Definition

$$\begin{aligned} \sigma, \mathcal{E} \models z.\Phi & \leftrightarrow \sigma, \mathcal{E}[z := 0] \models \Phi \\ \sigma, \mathcal{E} \models P_{\sim\lambda}[\Phi_1 \mathcal{U} \Phi_2] & \leftrightarrow \text{for all divergent adversaries } A \in \mathcal{A}_H, \\ & \text{the total probability of all infinite paths } \omega \\ & \text{under } A \text{ with } \omega, \mathcal{E} \models \Phi_1 \mathcal{U} \Phi_2 \text{ is } \sim \lambda. \end{aligned}$$

Remember:  $\Phi_1 \mathcal{U} \Phi_2$  in TCTL corresponds to  $(\Phi_1 \vee \Phi_2) \mathcal{U} \Phi_2$  in CTL.

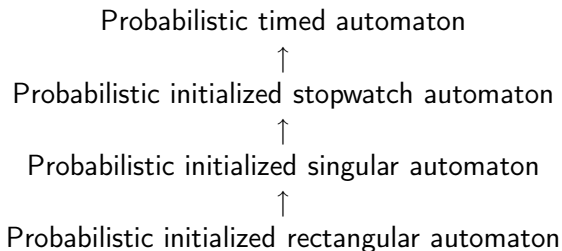


## Lemma

*The PTCTL model checking problem for initialized probabilistic rectangular automata is decidable.*

## Lemma

*The PTCTL model checking problem for initialized probabilistic rectangular automata is decidable.*



A **probabilistic timed automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable is a clock, i.e.,  $Act(l)(x) = [1, 1]$  for all locations  $l$  and variables  $x$ .

A **probabilistic timed automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable is a clock, i.e.,  $Act(l)(x) = [1, 1]$  for all locations  $l$  and variables  $x$ .

## Lemma

*The PTCTL model checking problem for probabilistic timed automata is decidable.*

A **probabilistic timed automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable is a clock, i.e.,  $Act(l)(x) = [1, 1]$  for all locations  $l$  and variables  $x$ .

## Lemma

*The PTCTL model checking problem for probabilistic timed automata is decidable.*

Model checking as for timed automata with

- summing up probabilities for distributions, and
- taking minimum/maximum of all distributions within a location.

A **probabilistic stopwatch automaton** is a probabilistic rectangular automaton with deterministic jumps and stopwatch variables only.

A **probabilistic stopwatch automaton** is a probabilistic rectangular automaton with deterministic jumps and stopwatch variables only.

Probabilistic timed automaton



Probabilistic initialized stopwatch automaton

A **probabilistic stopwatch automaton** is a probabilistic rectangular automaton with deterministic jumps and stopwatch variables only.

Probabilistic timed automaton



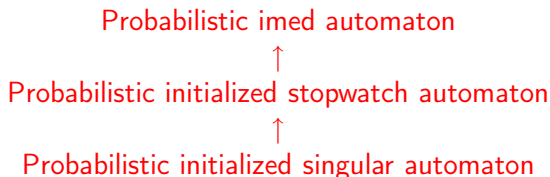
Probabilistic initialized stopwatch automaton

**Construction** is similar as for non-probabilistic automata (probabilistic setting: adapt preconditions).

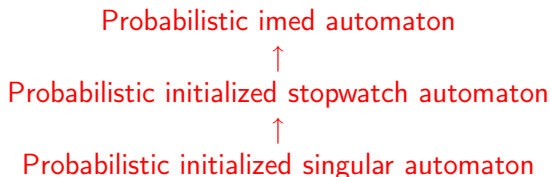


A **probabilistic singular automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable of the automaton is a finite-slope variable.

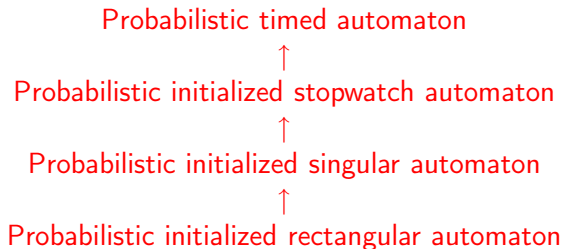
A **probabilistic singular automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable of the automaton is a finite-slope variable.

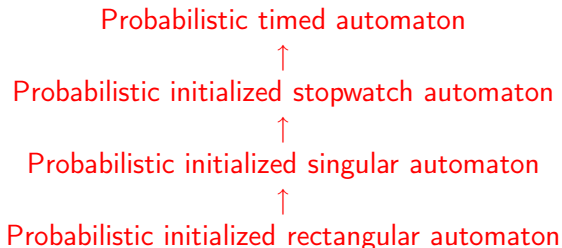


A **probabilistic singular automaton** is a probabilistic rectangular automaton with deterministic jumps such that every variable of the automaton is a finite-slope variable.



**Construction** is similar as for non-probabilistic automata (probabilistic setting: adapt pre- and postconditions).





**Construction** is similar as for non-probabilistic automata (probabilistic setting: adapt all conditions, copies of distributions).