Analysis of Non-Linear Probabilistic Hybrid Systems

Joseph Assouramou[*]
Université Laval, Quebec, Canada
Joseph.Assouramou.1@ulaval.ca

Josée Desharnais[*]
Université Laval, Quebec, Canada
Josee.Desharnais@ift.ulaval.ca

This paper shows how to compute, for probabilistic hybrid systems, the clock approximation and linear phase-portrait approximation that have been proposed for non probabilistic processes by Henzinger et al. The techniques permit to define a rectangular probabilistic process from a non rectangular one, hence allowing the model-checking of any class of systems. Clock approximation, which applies under some restrictions, aims at replacing a non rectangular variable by a clock variable. Linear phase-approximation applies without restriction and yields an approximation that simulates the original process. The conditions that we need for probabilistic processes are the same as those for the classic case.

Keywords: probabilistic hybrid systems, model-checking, linear and rectangular processes, approximation, linearization

1 Introduction

Hybrid processes are a combination of a process that evolves continuously with time and of a discrete component. A typical example is a physical system, such as a heating unit, that is controlled by a monitor. There are discrete changes of modes, like turning on and off the unit, and there is a continuous evolution – the change in temperature. Because of their continuous nature, model-checking hybrid systems can only be done for sub-classes of them. Especially, the largest class for which verification is decidable is the class of rectangular hybrid automata [7, 6]. Another such class for which verification is decidable is that of o-minimal hybrid automata [14], which models hybrid systems whose relevant sets and continuous behavior are definable in an o-minimal structure. In the probabilistic case, Sproston proposed methods to verify ∀-PBTL on probabilistic rectangular and o-minimal hybrid processes [11]. Probabilistic timed automata are also a subclass of such processes and have been analyzed extensively [8, 12].

In order to allow the verification of non-rectangular hybrid automata, two translation/approximation methods were proposed by Henzinger et al. [4]: clock-translation and linear phase-portrait approximation. The idea behind those methods is to transfer the verification of any hybrid automaton to the one of a rectangular hybrid automaton which exhibits the same behaviour or over approximates it. In this paper, we show how to apply these methods to probabilistic hybrid processes. We show that both methods apply with the same conditions as for the non deterministic case. The technique of approximation is based on replacing exact values by lower and upper bounds, after splitting the hybrid automaton for more precision in the approximation. Hence, we also show how to split a probabilistic hybrid automaton in order to obtain a bisimilar one. Other side contributions of this paper are: a slightly more general, yet a simpler definition of probabilistic automata than the one proposed by Sproston [11]; and the description, in the next background section, of the two translation techniques in a simpler way than what can be found in [4, 5], mostly because we take advantage of the fact that the definition of hybrid automata has been simplified since then, being presented in terms of functions instead of predicates, and being slightly less general than in the original paper [5].

[*] Research supported by NSERC

M. Massink and G. Norman (Eds.): 9th Workshop on
Quantitative Aspects of Programming Languages (QAPL 2011)
EPTCS 57, 2011, pp. 104–119, doi:10.4204/EPTCS.57.8

2 Transformation methods for hybrid automata

In this section, we describe the two methods presented by Henzinger et al [4] that will permit the verification of safety properties on any hybrid system.

Let $X = \{x_1, \dots, x_n\}$ be a finite set of real variables; we write $\dot{X} = \{\dot{x}_1, \dots, \dot{x}_n\}$ where $\dot{x}_i = \frac{dx_i}{dt}$ is the first derivative of x_i with respect to time. The set of predicates on $\dot{X} \cup X$ is denoted $G(\dot{X} \cup X)$. The set of *valuations* $\mathsf{a} : X \to \mathbb{R}$ is written $\mathbb{R}^X$ or $\mathbb{R}^n$. A set $U \subseteq \mathbb{R}^X$ is *rectangular* if there exists a family of (possibly unbounded) intervals $(I_x)_{x \in X}$ with rational endpoints such that $U = \{\mathsf{a} \in \mathbb{R}^n \mid \mathsf{a}(x) \in I_x \text{ for all } x \in X\}$. We denote by $R(X)$ the set of rectangles over X. For any set Y, we write $\mathscr{P}(Y)$ (resp. $\mathscr{P}_{fin}(Y)$) for the power set of Y (resp. finite power set of Y). For any variable x, belonging to X or not, we write $\mathsf{a}[x \mapsto r]$ for the valuation that maps x to $r \in \mathbb{R}$ and agrees with a elsewhere. Conversely, if $X' \subseteq X$, we write $\mathsf{a}|_{X'}$ for the restriction of a to X'. In the following, we use the notation Set instead of the usual slightly misleading one: Reset.

Definition 1 *[1]* $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, E, \mathit{Pre}, \mathrm{Set})$ *is a* hybrid automaton (HA) *if*

- V *is a finite set of locations or control modes;*
- $X = \{x_1, x_2, \dots, x_n\}$ *is a set of n continuous variables;*
- $\mathrm{Inv} : V \to \mathscr{P}(\mathbb{R}^X)$ *defines invariants for the variables in each location.*
- $\mathrm{Init} : V \to \mathscr{P}(\mathbb{R}^X)$ *defines initial states and satisfies* $\mathrm{Init}(v) \subseteq \mathrm{Inv}(v)$ *for all* $v \in V$.
- Act *is a finite set of actions, possibly including a silent one,* τ*;*
- $\mathrm{Flow} : V \to G(\dot{X} \cup X)$ *is a flow evolution condition;*
- $E \subseteq V \times \mathrm{Act} \times V$ *is a finite set of discrete transitions;*
- $\mathit{Pre} : E \to \mathscr{P}(\mathbb{R}^X)$ *maps to every discrete transition a set of preconditions;*
- $\mathrm{Set} : E \times \mathbb{R}^X \to \mathscr{P}(\mathbb{R}^X)$ *describes change in values of variables resulting from taking edges. We write* $\mathrm{Set}^x(e) := \{\mathsf{d}(x) \mid \exists \mathsf{a} \in \mathbb{R}^X . \mathsf{d} \in \mathrm{Set}(e, \mathsf{a})\}$.

H is said to be rectangular *if the image of* Inv*, Pre and* Set *are included in* $R(X)$ *and* $\mathrm{Flow}(v) = \wedge_{x \in X}\ \dot{x} \in I_x$ *where each* $I_x \subseteq \mathbb{R}$ *is a (possibly unbounded) interval with rational endpoints.*

The semantics of H is a labelled transition system: the set of states is $S_H := \{(v, \mathsf{a}) \mid \mathsf{a} \in \mathrm{Inv}(v)\}$. There are two kinds of transitions between states: flow transitions and discrete transitions. In a flow transition, the mode of the automaton is fixed and only the variables' values change over time. More formally, there is a flow transition of duration $\sigma \in \mathbb{R}_{\geq 0}$ between states (v, a) and (v, a'), written $(v, \mathsf{a}) \xrightarrow{\sigma} (v, \mathsf{a}')$, if either (1) $\sigma = 0$ and $\mathsf{a} = \mathsf{a}'$ or (2) $\sigma > 0$ and there exists a differentiable function $\gamma : [0; \sigma] \to \mathbb{R}^n$ with $\dot{\gamma} : (0; \sigma) \to \mathbb{R}^n$ such that γ is a solution of $\mathrm{Flow}(v)$ with $\gamma(0) = \mathsf{a}$, $\gamma(\sigma) = \mathsf{a}'$, and $\gamma(\varepsilon) \in \mathrm{Inv}(v)$ for all $\varepsilon \in (0; \sigma)$. For discrete transitions, the control mode of the automaton changes instantaneously. We write $(v, \mathsf{a}) \xrightarrow{a} (v', \mathsf{a}')$, if there exists $e = (v, a, v') \in E$ such that $\mathsf{a} \in \mathit{Pre}(e)$ and $\mathsf{a}' \in \mathrm{Set}(e, \mathsf{a})$.

Example 1 *Figure 1 shows the graphical representation of a thermostat [5] that controls the variation of temperature in a room through a radiator. The whole system has three modes representing that the radiator is either on, off, or down; there is one initial state, where the radiator is on and the temperature has value 2. When the radiator is on the temperature increases with respect to the equation* $\dot{x} = -x + 5$ *whereas it decreases with respect to* $\dot{x} = -x$ *when the radiator is off. When the whole system is down, no variation of the temperature is modeled. The values of the temperature evolve in the range* $[1; 3]$. *The radiator must switch off when it is on and the temperature reaches* 3 *units and on when it is off and the temperature is* 1. *Finally, when we try to turn on the radiator, it might turn on or down.*

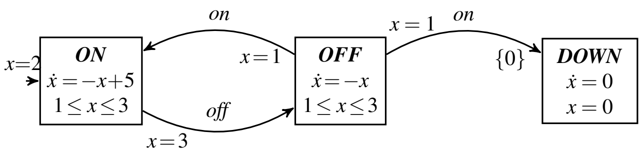

Figure 1: A graphical representation of the thermostat hybrid automaton

Because we need a notion of weak simulation, we define weak transitions through stuttering. Hence, let τ be the (usual) silent action. We write $s \overset{a}{\twoheadrightarrow} s'$ if there exists a finite sequence $s \xrightarrow{\tau} s_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} s_k \xrightarrow{a} s'$. Similarly, we write $s \overset{\sigma}{\twoheadrightarrow} s'$ if there exists a finite sequence $s \xrightarrow{\sigma_1} s_1 \xrightarrow{\tau} s_2 \xrightarrow{\sigma_2} \dots \xrightarrow{\tau} s_k \xrightarrow{\sigma_k} s'$ such that $\sum_i \sigma_i = \sigma \in \mathbb{R}_{\geq 0}$. *Simulation* and *bisimulation* are defined on the underlying infinite transition system (and are rather called *time bi/simulation in Henzinger et al. [4]*).

Definition 2 *[4] Let H and H' be two hybrid automata. A relation $\preceq \subseteq S_H \times S_{H'}$ is a* simulation *of H by H' if every initial state of H is related by $\preceq$ to an initial state of H' and if whenever $s \preceq s'$, then for each $a \in \mathrm{Act} \setminus \{\tau\} \cup \mathbb{R}_{\geq 0}$ and each transition $s \overset{a}{\twoheadrightarrow} s_1$, there exists a transition $s' \overset{a}{\twoheadrightarrow} s'{}_1$ such that $s_1 \preceq s'_1$. If $\preceq^{-1}$ is also a simulation, then $\preceq$ is called a bisimulation. If there is a simulation between H and H' (resp. a bisimulation), we write $H \preceq H'$ (resp. $H \equiv H'$).*

2.1 Clock-translation

The *clock-translation* method is based on the substitution of non-rectangular variables by clocks. Let H be a non-rectangular hybrid automaton. The substitution of a variable x of H by a clock t_x is possible only if, at any time, the value of t_x can be determined by the one of x (i.e., x is solvable).

2.1.1 Preliminaries

We say that a predicate is *simple* if it is a positive boolean combination of predicates of the form $x \sim c$ where $c \in \mathbb{R}$ and $\sim \in \{<, \leq, =, \geq, >\}$. We say that x is *solvable* in H if

- every initial condition, invariant condition, and precondition of H defines simple predicates for x and each flow condition of x in $\mathrm{Flow}(v)$ has the form $(\dot{x} = f_x^v(x)) \wedge P_x$, where P_x is a simple predicate on x; flow evolutions of other variables must not depend on x nor $\dot{x}$;
- the initial-value problem $\dot{y}(t) = f_x^v(y(t))$; $y(0) = c$ has a unique, continuous and strictly monotone solution g_c;
- H is initialised with respect to x. That is, for any transition $e \in E$, x must either stay unchanged in any valuation or get assigned only one value r for all valuations; this will happen if $\mathrm{Set}^x(e)$ is a singleton with the help of the following notation: we will write

  $$\mathrm{Set}^x(e) = \{r\} \subseteq \mathbb{R}_* := \mathbb{R} \cup \{*\},$$

  where r is either the unique value $r \in \mathbb{R}$, in which case we say that x is reset to r by e, or a special character, $*$, which will represent stability in the value of x. If $r = *$ we must also have that $f_x^v = f_x^{v'}$.

Example 2 *The thermostat automaton of Figure 1 is* solvable *as the flow evolution equation of variable x is solvable in all the modes: in mode ON, the differential equation $\dot{x} = -x+5$ with the initial condition $x(0) = 2$ has the function $x(t) = -3e^{-t}+5$ where $t \in \mathbb{R}_+$ as solution.*

Suppose that $x \in X$ is solvable in the hybrid automaton $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, E, Pre, \mathrm{Set})$, and let $c \in \mathbb{R}$ be a constant. We say that c is a *starting value* for a variable x if c is either: the initial value of x in some mode v, that is, $c = \mathsf{a}(x)$ for $\mathsf{a} \in \mathrm{Init}(v)$; or the unique value of $\mathrm{Set}^x(e)$ for some edge $e \in E$ if this value is not $*$. Let $D_v(x)$ be the finite set of starting values of x in v.

Transformation from $x \sim l$ to $t_x \sim' g_c^{-1}(l)$. To simplify the presentation below, we show how predicates on x are transformed into predicates on t_x [4].

Let $g_c(t)$ be the unique solution of the initial-value problem $\dot{y}(t) = f_x^v(y(t)); y(0) = c$, where $c \in \mathbb{R}$. As $g_c(t)$ is strictly monotone, there exists at most one $t \in \mathbb{R}_+$ such that $g_c(t) = l$ for each $l \in \mathbb{R}$. Let $g_c^{-1}(l) = t$ if $g_c(t) = l$ and $g_c^{-1}(l) = -$ if $g_c(t) \neq l$ for all $t \in \mathbb{R}_+$. Let $O := \{<, \leq, =, \geq, >\}$. The transformation from simple atomic predicates over $\{x\}$ to simple atomic predicates over $\{t_x\}$ is the function α_c defined using $\sim \in O$, $lt : O \to O$ and $gt : O \to O$, as follows:

$$\alpha_c(x \sim l) = \begin{cases} \mathit{true} & \text{if } g_c^{-1}(l) = - \text{ and } c \sim l. \\ \mathit{false} & \text{if } g_c^{-1}(l) = - \text{ and } c \nsim l. \\ t_x\ lt(\sim)\ g_c^{-1}(l) & \text{if } g_c^{-1}(l) \neq - \text{ and } c \sim l. \\ t_x\ gt(\sim)\ g_c^{-1}(l) & \text{if } g_c^{-1}(l) \neq - \text{ and } c \nsim l. \end{cases}$$

$\sim$	$lt(\sim)$	$gt(\sim)$
$<$	$<$	$>$
$\leq$	$\leq$	$\geq$
$=$	$=$	$=$
$\geq$	$\leq$	$\geq$
$>$	$<$	$>$

For each (v, c_i) of the hybrid automaton, every predicate $x \sim l$ is replaced by the predicate $\alpha_{c_i}(x \sim l)$, except the invariant predicate which is replaced by $\alpha_{c_i}(x \sim l)$ if $c_i \sim l$, and by *false* otherwise((v, c_i) may be removed in the latter case).

2.1.2 Clock-translation

We are now ready to define clock-translation.

Definition 3 *[4] If $x \in X$ is solvable in $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, E, Pre, \mathrm{Set})$, then the clock-translation of H with respect to x is*

$$T = (V_T, X_T, \mathrm{Init}_T, \mathrm{Act}, \mathrm{Inv}_T, \mathrm{Flow}_T, E_T, Pre_T, \mathrm{Set}_T),$$

the hybrid system obtained from the following algorithm:

Step 1: adding the clock t_x.

- *$V_T := \cup_{v \in V} \{v\} \times D_v(x)$, that is, each mode v of H is split. $X_T := X \cup \{t_x\}$.*
- $\mathrm{Init}_T(v, c) := \{\mathsf{a}[t_x \mapsto 0] \mid \mathsf{a} \in \mathrm{Init}(v)$ *and* $\mathsf{a}(x) = c\}$.
- *E_T contains two kinds of control switches; for $c \in D_v(x)$ and $e = (v, a, v') \in E$*

  – *if* $\mathrm{Set}^x(e) = \{r\} \subseteq \mathbb{R}$, *$E_T$ contains the edge $e_T := ((v, c), a, (v', r))$, with $Pre_T(e_T) := Pre(e)$ and* $\mathrm{Set}_T(e_T, \mathsf{a}) := \{\mathsf{d}[t_x \mapsto 0] \mid \mathsf{d} \in \mathrm{Set}(e, \mathsf{a}|_X)\}$.

  – *if* $\mathrm{Set}^x(e) = \{*\}$, *$E_T$ contains the edge $e_T := ((v, c), a, (v', c))$ with $Pre_T(e_T) := Pre(e)$ and* $\mathrm{Set}_T(e_T, \mathsf{a}) := \{\mathsf{d}[t_x \mapsto \mathsf{a}(t_x)] \mid \mathsf{d} \in \mathrm{Set}(e, \mathsf{a}|_X)\}$.

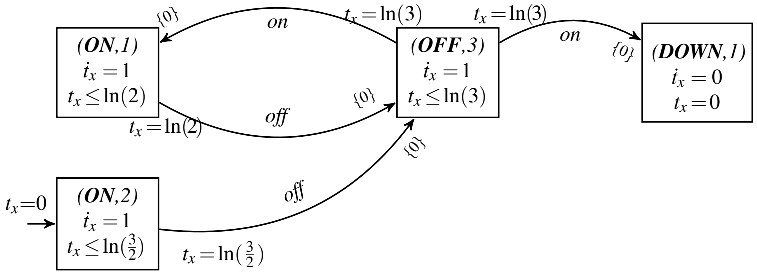

Figure 2: The clock-translation of the thermostat automaton

Step 2: moving to conditions on t_x. We view the images of Init, Inv*,* *Pre*, Set *and* Flow *as predicates instead of sets of valuations. We replace these predicates over x of the form $x \sim r$ where $\sim \in \{<, \leq, =, >, \geq\}$ and $r \in \mathbb{R}$ in T by predicates over t_x of the form $t_x \sim' g_c^{-1}(r)$ as described above. Finally, the variable x can be removed from X_T.*

Example 3 *The timed automaton of Figure 2 is obtained by applying the clock-translation on the thermostat automaton. Each mode v of the automaton is split into $|D_v(x)|$ modes. Since $D_{ON}(x) = \{1,2\}$, we get the modes $(ON, 1)$ and $(ON, 2)$. For these two modes, the differential equations are respectively "$\dot{x} = -x+5$, where $x(0) = 2$" and "$\dot{x} = -x+5$, where $x(0) = 1$", and then we have the solutions "$x(t) = -3e^{-t}+5$" and "$x(t) = -4e^{-t}+5$" respectively. Next, we substitute the variable x by the clock t_x in the preconditions, and the invariants. Then, the constraint $x \leq 3$ becomes $t \leq \ln(2)$ and $t \leq \ln(\frac{3}{2})$ respectively. The two automata are bisimilar, by the following theorem.*

Theorem 1 *[4] H is bisimilar to its clock-translation T. The relation is given by the graph of the projection $\eta : S_T \to S_H$, defined as $\eta((v,c),\mathsf{a}) := (v,\mathsf{a}')$, where a' satisfies $\mathsf{a}'|_X = \mathsf{a}|_X$ and $\mathsf{a}'(x) = g_c(\mathsf{a}(t_x))$ where g_c is the solution of the initial-value problem $[\dot{y}(t) = f_x^v(y(t)); y(0) = c]$.*

As a corollary, H and T satisfy the same properties of usual temporal logics.

2.2 Linear phase-portrait approximation

We now present the second method which allows the translation of any hybrid automaton into a rectangular one. The linear phase-portrait approximation method can be applied to any hybrid automaton, yielding an *approximation* of the original process which simulates the original automaton (instead of being bisimilar to it, as for clock-translation). This implies that if a safety property is verified on the approximation, then it holds in the original system [5].

The general method is to first split the automaton and then approximate the result. Approximation is done by replacing non-rectangular flow equations by lower and upper bounds on the variables, hence forgetting the true details of the equations. By splitting more finely, one obtains a better approximation.

2.2.1 Splitting a hybrid automaton

Let H be a hybrid automaton with invariant function Inv. A *split function* is a map θ that returns to each mode v of H a finite open cover $\{\text{inv}_1^v, \ldots, \text{inv}_m^v\} \subseteq \mathscr{P}(\mathbb{R}^X)$ of $\text{Inv}(v)$. In splitting, a mode v will be split into several modes according to the cover $\theta(v)$. The fact that $\cup_i \text{inv}_i^v = \text{Inv}(v)$ makes sure that states are preserved whereas the evolution inside mode v is preserved through silent transitions between copies of v, which is possible because $\theta(v)$'s components overlap.

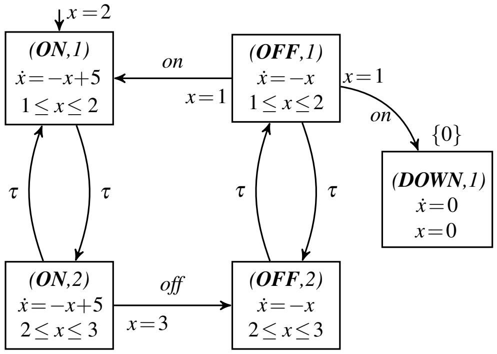

Figure 3: A split of the thermostat

Definition 4 *Let $H=(V,X,\mathrm{Init},\mathrm{Act},\mathrm{Inv},\mathrm{Flow},E,Pre,\mathrm{Set})$ be a hybrid automaton. The split of H by θ is the hybrid automaton $\theta(H)=(V_\theta,X,\mathrm{Init}_\theta,\mathrm{Act}_\theta,\mathrm{Inv}_\theta,\mathrm{Flow}_\theta,E_\theta,Pre_\theta,\mathrm{Set}_\theta)$ defined as:*

- $V_\theta=\{(v,i)\mid v\in V$ *and* $1\le i\le|\theta(v)|\}$
- $\mathrm{Init}_\theta((v,i))=\mathrm{Init}(v)\cap\mathrm{inv}_i^v$
- $\mathrm{Act}_\theta=\mathrm{Act}\cup\{\tau\}$
- $\mathrm{Inv}_\theta(v,i)=\mathrm{inv}_i^v$
- $\mathrm{Flow}_\theta(v,i)=\mathrm{Flow}(v)$
- $E_\theta=E_1\cup E_2$, *where E_1 contains the control switch $((v,i),a,(v',j))$ for each $(v,a,v')\in E$, whereas $E_2=\{((v,i),\tau,(v,j))\mid(v,i),(v,j)\in V_\theta\}$ allows the automaton to transit silently between the different copies of v.*
- *If $e_\theta=((v,i),a,(v',j))\in E_1$, we set $Pre_\theta(e_\theta)=Pre(v,a,v')$ and $\mathrm{Set}_\theta(e_\theta,\mathsf{a})=\mathrm{Set}((v,a,v'),\mathsf{a})$. If $e_\theta=((v,i),\tau,(v,j))\in E_2$, we set $Pre_\theta(e_\theta)=\mathbb{R}^X$ and $\mathrm{Set}_\theta(e_\theta,\mathsf{a})=\{\mathsf{a}\}$.*

Note that the cover $\theta(v)$ need not really be open. What is important is that the evolution within any mode be preserved, as pointed out in [4]. This is the case in the following example, where components of the cover are closed and intersect in exactly one point, which is sufficient to allow evolution.

Example 4 *The automaton in Figure 3 is a split of the thermostat automaton with function $\theta(ON)=\theta(OFF)=\{\{x\mid 1\le x\le 2\},\ \{x\mid 2\le x\le 3\}\}$, and $\theta(DOWN)=\{\{x\mid x=0\}\}$. Note the silent transitions between states $((ON,i),2)$, $i=1,2$, the latter being duplicates of the original thermostat's state $(ON,\{x\mapsto 2\})$, that preserve the evolution within mode ON.*

2.2.2 Approximating a hybrid automaton

An (over) approximation of a HA is obtained by weakening all predicates of its evolution.

Definition 5 *Let $H=(V,X,\mathrm{Init},\mathrm{Act},\mathrm{Inv},\mathrm{Flow},E,\ Pre,\mathrm{Set})$ be a HA. Another hybrid automaton $A=(V,X,\mathrm{Init}_A,\mathrm{Act},\mathrm{Inv}_A,\ \mathrm{Flow}_A,E,Pre_A,\mathrm{Set}_A)$ is a* basic approximation *of H if:*

- *for all $v\in V$, $\mathrm{Inv}(v)\Rightarrow\mathrm{Inv}_A(v)$, $\mathrm{Flow}(v)\wedge\mathrm{Inv}(v)\Rightarrow\mathrm{Flow}_A(v)\wedge\mathrm{Inv}_A(v)$, $\mathrm{Init}(v)\Rightarrow\mathrm{Init}_A(v)$;*
- *for every discrete transition $e\in E$, $Pre(e)\Rightarrow Pre_A(e)$ and $\mathrm{Set}(e)\Rightarrow\mathrm{Set}_A(e)$;*

where sets of valuations are viewed as predicates. If there exists a split θ on H such that A is a basic approximation of H_θ then A is a phase-portrait approximation *of H. If the lower and upper bounds of all the predicates in A are rational then A is a (rational)* linear *phase-portrait approximation of H.*

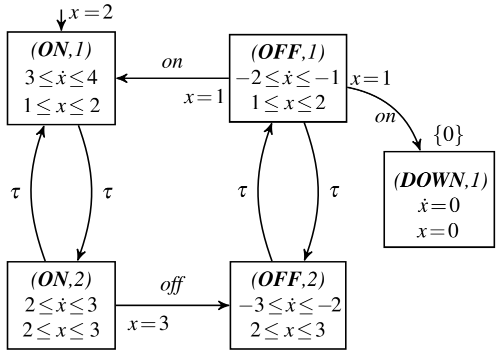

Figure 4: A linear phase-portrait approximation of the thermostat

A straightforward linear phase-portrait approximation is obtained by replacing the invariant in each mode v by a product of rational intervals that contains $\mathrm{Inv}(v)$ and all other predicates, including the flow evolution, by the rational lower and upper bounds implied by the invariant on v.

Example 5 *The automaton of Figure 4 is the linear phase-portrait approximation of the thermostat (with the same split as in Figure 3). Every predicate on $\dot{x}$ is replaced by a predicate that specifies lower and upper bounds on it. For example, the approximation of $\dot{x}$ in mode (ON,1) yields the set* $\{\dot{x} \mid 3 \leq \dot{x} \leq 4\}$.

The following theorem implies that if a safety property is verified for an approximation A, it holds also for H [4, 5].

Theorem 2 *[4] If A is a linear phase-portrait approximation of H, then A simulates H. If it is just a split of H then $A \equiv H$. In both cases, the state $((v,i),\mathsf{a})$ of A is related to (v,a) in H.*

The automaton of Figure 4 simulates the split of the automaton (Figure 3), and then, by transitivity of simulation, simulates the thermostat hybrid automaton.

The verification of initialized rectangular hybrid automata has been widely discussed, particularly in [7] and [9] and it is proved that their verification is decidable. Hence, a non-rectangular hybrid automata can be verified (for satisfaction of safety properties) if an initialized linear phase-portrait approximation can be defined from it.

3 Analysis of probabilistic hybrid automata

In this section, we show how the two methods presented above can be used for probabilistic hybrid automata. Our definition of a probabilistic hybrid automaton (PHA) is close to but slightly more general than the one of Sproston [11]. We also add the definition of finitely branching PHAs. A (discrete) probability distribution over a set C is a function $\mu : C \to [0,1]$ such that $\sum_{c \in C} \mu(c) \leq 1$; the support of μ is defined as $\mathrm{supp}(\mu) := \{c \in C \mid \mu(c) > 0\}$, and it is countable. For $U \subseteq C$, we sometimes write $\mu(U) := \sum_{c \in U} \mu(c)$. Let $\mathrm{Dist}(C)$ be the set of all (discrete) distributions over C.

Definition 6 *A tuple* $H = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, prob, \langle pre_{v,a} \rangle_{v \in V, a \in \mathrm{Act}}, \langle pos_{v,a} \rangle_{v \in V, a \in \mathrm{Act}})$ *is a* probabilistic hybrid automaton (PHA) *if* V, X Init, Act, Inv *and* Flow *are as in Def. 1 and*

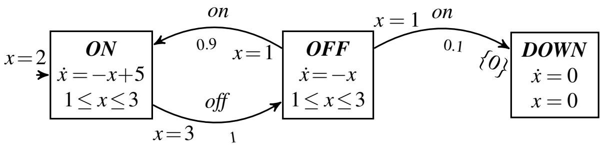

Figure 5: A probabilistic version of the thermostat

- *prob* $: V \times \mathrm{Act} \to \mathscr{P}_{\mathit{fin}}(\mathrm{Dist}(V \times \mathscr{P}(\mathbb{R}^X_*)))$ *encodes probabilistic transitions. If* $\tau \in \mathrm{Act}$*, we require that every* $\mu \in prob(v,\tau)$ *is concentrated in a unique pair of the form* $(v, \{\mathsf{d}\})$.
- $pre_{v,a} : prob(v,a) \to \mathscr{P}(\mathbb{R}^X)$ *defines preconditions for distributions from* $v \in V$ *and* $a \in \mathrm{Act}$.
- $pos_{v,a} : prob(v,a) \times V \to \mathscr{P}(\mathbb{R}^X)$ *defines postconditions for distributions associated with* $v \in V$ *and* $a \in \mathrm{Act}$.

We say that H is finitely branching *if for every* $v \in V$, $a \in \mathrm{Act}$, $\mu \in prob(v,a)$, μ *is* finitely branching*, that is,* $\mathrm{supp}(\mu)$ *is finite and every set* post *such that* $(v', \mathsf{post}) \in \mathrm{supp}(\mu)$ *is also finite.*

To simplify the notation, we drop the subscripts of *pre* and *pos* when there is no ambiguity.

The semantics of PHAs, is given by probabilistic transition systems. States are defined in the same way. As for non-probabilistic hybrid automata, we distinguish two kinds of transitions in PHAs. Flow transitions are the same, but discrete transitions are now probabilistic and hence defined from a state (v, a) to a distribution. To define transitions, we need some notations on valuations. For $\mathsf{d} \in \mathbb{R}^X_*$, $\mathsf{a} \in \mathbb{R}^X$, $\mathsf{post} \subseteq \mathbb{R}^X_*$, $\mathsf{A} \subseteq \mathbb{R}^X$ and $x \in X$, let

$$\mathsf{d}[\mathsf{a}](x) := \begin{cases} \mathsf{d}(x) & \text{if } \mathsf{d}(x) \neq * \\ \mathsf{a}(x) & \text{if } \mathsf{d}(x) = *, \end{cases} \qquad \text{and} \qquad \begin{array}{l} \mathsf{post}[\mathsf{A}] := \{\mathsf{d}[\mathsf{a}] \mid \mathsf{d} \in \mathsf{post}, \mathsf{a} \in \mathsf{A}\} \\ \mathsf{post}(x) := \{\mathsf{d}(x) \mid \mathsf{d} \in \mathsf{post}\}. \end{array}$$

Transitions of action a from a state (v, a) in the underlying PTS are as follows. Let $\mu \in prob(v,a)$, $\mathsf{a} \in pre_{v,a}(\mu)$, $\mathrm{supp}(\mu) = \{(v_i, \mathsf{post}_i)\}_{i=1}^m$. Each combination of $\mathsf{d}_i \in \mathsf{post}_i$, $i = 1, \ldots, m$, such that $\mathsf{d}_i[\mathsf{a}] \in pos(\mu, v_i)$, defines a transition

$$(v, \mathsf{a}) \xrightarrow{a} \mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle},$$

where $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ is positive on (arrival) states $(v_i, \mathsf{d}_i[\mathsf{a}])$; the probability that the automaton transits to a state (v', a') is

$$\mu_{\mathsf{a}}{}^{\langle \mathsf{d}_i \rangle}(v', \mathsf{a}') := \begin{cases} \displaystyle\sum_{i=1}^{m} \{\mu(v_i, \mathsf{post}_i) \mid v_i = v', \mathsf{d}_i[\mathsf{a}] = \mathsf{a}'\} & \text{if } \mathsf{a}' \in pos(\mu, v') \\ 0 & \text{otherwise.} \end{cases}$$

Example 6 *A probabilistic version of the thermostat is shown in Figure 5. Each discrete transition is labeled by a probability value and an action. Since there is only one variable, a valuation can be represented as a real number. For mode OFF, we have* $prob(OFF, on) = \{\mu\}$ *where* $pre(\mu) = \{1\}$*, such that* $\mu(ON, \{*\}) = 0.9$*, that is, the temperature is unchanged, and* $\mu(DOWN, \{0\}) = 0.1$*. Here, defining* $pos(\mu, -)$ *is not necessary since it is encoded in* μ*. Suppose that in another example one would set* $\mu(ON, [1;3]) = 0.9$*. This would mean that the temperature would end up in the interval* $[1;3]$ *and that the exact value would happen non deterministically. This example would not be finitely branching.*

We now discuss how Definition 6 of PHA slightly differs from previous ones [11]. First note that non deterministic transitions in PHA arise in two ways: when the image of *prob* is not a singleton, and from the possible combinations $\mathsf{d}_i \in \mathsf{post}_i$, $i=1,\dots,m$, that we can obtain. Hence, the expression *finitely branching* is well chosen because every set post, such that $(v,\mathsf{post}) \in \mu$, being finite in the underlying probabilistic transition system, every state (v,a) will have finitely many distributions $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ associated to any action.

The star notation is more general than the reset set of Sproston [11] if there is more than one variable. In the latter, the target of a transition is a distribution over $V \times \mathscr{P}(\mathbb{R}^n) \times \mathscr{P}(X)$, where the third component of a tuple (v,post,X'), called a *reset set*, represents the set of variables that can change value in the transition, with respect to valuations of post. The star notation allows to state, for example, that valuation (x,y) will be modified to $(x,0)$, $(x,2)$, $(0,y)$, $(1,1)$ with probability 1 non deterministically. The corresponding set would be $\mathsf{post}_0 := \{(*,0),(*,2),(0,*),(1,1)\}$. This is an important feature to describe the transitions of the clock-translation (see Section 3.1) and is not possible with the reset set because there is no uniform reset of any variable in post_0. Note in passing that the star notation avoids a third component in the notation and will allow to state very simply the notion of initialised PHA. The use of a postcondition function together with the star notation allows to define distributions on complex sets, such as: $\mathsf{post}_0 \cap ([0;3] \times [1;4])$, the distribution being defined on post_0 and the rectangle being the postcondition. We could not transfer the latter into the distributions by defining μ, for example, as $\mu(v',\mathsf{post}_0 \cap ([0;3] \times [1;4]))$ since post_0 may contain valuations that assign $*$ to a variable which implies that its value depends on the actual state, or valuation for which the distribution μ will be used. Postconditions are necessary for the splitting of PHAs, if one wants to avoid the even more general model that consists in defining probabilistic transitions *prob* from $S_H \times A$ to $\mathscr{P}_{\mathit{fin}}(\mathrm{Dist}(V \times \mathscr{P}(\mathbb{R}^X)))$. This is too general for practical purposes: indeed, any description of a system must be finite and hence states must be described in a parametric (or generic) way.

The condition on τ transitions is to simplify the definition of weak transitions, since it permits to write $s \xrightarrow{\tau} s'$. There may be more than one τ transition from a mode v, but for each $\mu \in \mathit{prob}(v,\tau)$, there is a unique pair $(v',\{\mathsf{d}\})$ such that $\mu(v',\{\mathsf{d}\}) = 1$. Then, following definition of $\mu_{\mathsf{a}}^{\langle \mathsf{d}_i \rangle}$ above, there is a τ-transition to state $(v',\mathsf{d}[\mathsf{a}])$ if and only if $\mathsf{d}[\mathsf{a}] \in \mathit{pos}(\mu,v')$ (that is, $\mu_{\mathsf{a}}(v',\mathsf{d}[\mathsf{a}]) = 1$). Weak flow transitions are then defined between states as for hybrid automata, and we write $s \overset{a}{\twoheadrightarrow} \mu$ if there exists a finite sequence of transitions $s \xrightarrow{\tau} s_1 \xrightarrow{\tau} s_2 \dots \xrightarrow{\tau} s_k \xrightarrow{a} \mu$.

We now define a notion of weak simulation between PHAs. Let $\preceq \subseteq S \times T$ be a relation between two sets S and T. For $X \subseteq S$, we use the notation $\preceq(X) := \{t \in T \mid \exists s \in X . s \preceq t\}$.

Definition 7 *Let H_1, H_2 be two probabilistic hybrid automata. A relation $\preceq \subseteq S_{H_1} \times S_{H_2}$ is a* simulation *if any initial state of H_1 is related to an initial state of H_2 and whenever $s_1 \preceq s_2$, we have:*

- *if $s_1 \overset{a}{\twoheadrightarrow} \mu_1$, for $a \in \Sigma \setminus \{\tau\}$ then $s_2 \overset{a}{\twoheadrightarrow} \mu_2$ and $\mu_1(X) \leq \mu_2(\preceq(X))$ for every X;*
- *if $s_1 \overset{\sigma}{\twoheadrightarrow} s_1'$, for $\sigma \in \mathbb{R}_{\geq 0}$, $s_2 \overset{\sigma}{\twoheadrightarrow} s_2'$ and $s_1' \preceq s_2'$.*

Then we say that H_1 is simulated by H_2, written $H_1 \preceq H_2$. If $\preceq^{-1}$ is also a simulation, it is a bisimulation. Equivalently, an equivalence relation is a bisimulation *if in the condition above we have $\mu_1(X) = \mu_2(X)$ for each equivalence class X.*

This definition is known to be equivalent to the one using weight functions: see Desharnais et al. [3] for a proof that the inequality between μ_1 and μ_2 above is equivalent to the existence of a network flow between them; it is well-known [2], in turn, that the flow condition is equivalent to the existence of a weight function between μ_1 and μ_2.

3.1 Probabilistic clock-translation

In this section, we prove that clock-translation [4], can be applied to a PHA and that it results in a bisimilar PHA, as expected. The general method is to first compute a non probabilistic hybrid automaton from a PHA. Then we apply clock-translation and finally add probabilities in order to obtain a probabilistic clock-translation. There is no condition for computing the underlying non probabilistic HA but we need a notion of solvability so that we will be able to use the clock-translation method thereafter. A variable x of a PHA H is *solvable* if

- the two first conditions of solvability for non probabilistic systems are satisfied
- for every state $s \in S_H$, if $s \xrightarrow{a} \mu$, and $\mu(v', \mathsf{post}) > 0$, then $\mathsf{post}(x) = \{r\}$ for some $r \in \mathbb{R}_*$. Moreover, if $\mathsf{post}(x) = \{*\}$, then we must have $f_x^v = f_x^{v'}$.

Let $H_p = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, \mathit{prob}, \langle \mathit{pre} \rangle, \langle \mathit{pos} \rangle)$ be a PHA. The algorithm has three steps:

Step 1: Define $H{:=}(V, X, \mathrm{Init}, A, \mathrm{Inv}, \mathrm{Flow}, E, \mathit{Pre}, \mathrm{Set})$ the *underlying non probabilistic* HA of H_p as:

- $A := \{\ a_\mu^{\mathsf{post}} \mid \exists v, v'\ \in V$ such that $\mu \in \mathit{prob}(v, a)$ and $\mu(v', \mathsf{post}) > 0\}$.
- $E := \{(v, a_\mu^{\mathsf{post}}, v') \mid \mu \in \mathit{prob}(v, a)$ and $\mu(v', \mathsf{post}) > 0\}$. Finally, for $e := (v, a_\mu^{\mathsf{post}}, v') \in E$, we set $\mathit{Pre}(e) := \mathit{pre}_{v,a}(\mu)$ and $\mathrm{Set}(e, \mathsf{a}) := \mathsf{post}[\mathsf{a}]$.

Note that solvability of H_p implies that $\mathrm{Set}^x(e) = \{r\} \subseteq \mathbb{R}_*$ and hence H is also solvable.

Step 2: Since H is solvable, let $T = (V_T, X_T, \mathrm{Init}_T, A, \mathrm{Inv}_T, \mathrm{Flow}_T, E_T, \mathit{Pre}_T, \mathrm{Set}_T)$ be the clock-translation of H w.r.t. x. Hence, each transition $e = (v, a_\mu^{\mathsf{post}}, v')$ of H becomes a transition $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', r))$ in T, where $r = c$ if $\mathsf{post}(x) = \{*\}$, otherwise $\mathsf{post}(x) = \{r\}$.

Step 3: Finally, we build $T_p = (V_T, X_T, \mathrm{Init}_T, \mathrm{Act}, \mathrm{Inv}_T, \mathrm{Flow}_T, \mathit{Prob}, \langle \mathit{Pre} \rangle, \langle \mathit{Pos} \rangle)$, the probabilistic clock-translation of H_p from T as follows. Let (v, c) be in V_T, and a in Act. $\mathit{Prob}((v,c), a)$ contains all distributions ν_μ, defined from some $\mu \in \mathit{prob}(v, a)$ as follows:

- for each edge $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', r))$ such that $\mathrm{Set}_T^{t_x}(e_T) = \{0\}$ (i.e., $\mathrm{Set}^x(e) = \{r\}$), let

$$\nu_\mu((v', r), \mathsf{post}[t_x \mapsto 0]) := \mu(v', \mathsf{post});$$

- for each transition $e_T = ((v,c), a_\mu^{\mathsf{post}}, (v', c))$ such that $\mathrm{Set}_T^{t_x}(e_T) = \mathrm{Set}^x(e) = \{*\}$, let

$$\nu_\mu((v', c), \mathsf{post}[t_x \mapsto *]) := \mu(v', \mathsf{post}).$$

For both cases, $\mathit{Pre}(\nu_\mu) := \mathit{Pre}_T(e_T)$, and $\mathit{Pos}(\nu_\mu, (v', r)) := \{\mathsf{a} \in \mathbb{R}_*^{X \cup \{t\}} \mid \mathsf{a}|_X \in \mathit{pos}(\mu, v')\}$.

Example 7 *The clock-translation of the probabilistic thermostat automaton is a slight modification of the HA of Figure 2; all transitions get probability* 1 *except for on-transitions from* $(OFF, 3)$ *which have probability* 0.9 *to* $(ON, 1)$ *and* 0.1 *to* $(DOWN, 1)$.

We should now prove that the construction yields a valid PHA: this will be a consequence of the following theorem.

Theorem 3 *If T_p is the clock-translation of H_p, then H_p and T_p are bisimilar.*

Proof. Let H be the underlying non probabilistic automaton of H_p and T its clock-translation. By Theorem 1, H and T are bisimilar through $\eta : S_T \to S_H$ which, being a function, returns a unique state for any state of T. As H_p and H have the same state space, similarly for T and T_p, η can be seen as a function between states of T_p and H_p. We prove that η is a bisimulation between T_p and H_p.

Let $s = ((v,c),\mathsf{a})$ be a state of T_p. There are two kinds of transitions to check in the definition of simulation. For flow transitions, let $s \overset{\sigma}{\twoheadrightarrow} s'$, where $\sigma \in \mathbb{R}_{>0}$. Then since η is a bisimulation between H and T, we obtain $\eta(s) \overset{\sigma}{\twoheadrightarrow} \eta(s')$, as wanted. For discrete transitions, we have to prove that for all $\nu_\mu \in Prob((v,c),a)$ defined from $\mu \in prob(v,a)$, for $\mathsf{a} \in Pre(\nu_\mu)$ and any combination $\langle \mathsf{d}_i \rangle$ from the support of ν_μ, we have $\nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}(\eta^{-1}(U)) = \mu_{\mathsf{a}|_X}^{\langle \mathsf{d}_i|_X \rangle}(U)$ for all $U \subseteq S_{H_p}$. In fact, we need only to prove it for U equals to some state (v',a') since $\{(v',\mathsf{a}')\} \cup \eta^{-1}((v',\mathsf{a}'))$ is an equivalence class.

$$\begin{aligned}
\nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}(\eta^{-1}((v',\mathsf{a}'))) &= \sum_{\mathsf{b},r} \{\, \nu_{\mu,\mathsf{a}}^{\langle \mathsf{d}_i \rangle}((v',r),\mathsf{b}) \mid \underbrace{\mathsf{b}|_X = \mathsf{a}',\ g_r(\mathsf{b}(t_x)) = \mathsf{a}'(x)}_{P(\mathsf{b},u)} \} \\
&= \sum_{\mathsf{b},r} \sum_{i=1}^{m} \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v', r_i = r,\ \mathsf{d}_i[\mathsf{a}] = \mathsf{b}, \\ \mathsf{post}_i(x) = \{r_i\}, P(\mathsf{b},r_i) \end{array} \} \\
&\qquad \cup \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v', r_i = r = c,\ \mathsf{d}_i[\mathsf{a}] = \mathsf{b}, \\ \mathsf{post}_i(x) = \{*\}, P(\mathsf{b},c) \end{array} \} \\
&= \sum_{i=1}^{m} \{\, \nu_\mu((v_i,r_i),\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{r_i\},\ \mathsf{b}(t_x) = 0 \end{array} \} \\
&\qquad \cup \{\, \nu_\mu((v_i,c),\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{*\},\ \mathsf{b}(t_x) = \mathsf{a}(t_x) \end{array} \} \\
&= \sum_{i=1}^{m} \{\, \mu(v_i,\mathsf{post}_i) \mid \begin{array}{l} v_i = v',\ \mathsf{d}_i[\mathsf{a}]|_X = \mathsf{a}', \\ \mathsf{post}_i(x) = \{r_i\} \text{ or } \{*\} \end{array} \} \\
&= \mu_{\mathsf{a}|_X}^{\langle \mathsf{d}_i|_X \rangle}(v',\mathsf{a}').
\end{aligned}$$

In the third equality, the double sum is reduced to a single one because i determines b and r. □

Corollary 1 *The clock-translation of a solvable PHA is a PHA.*

Proof. We only need to prove that every defined ν_μ is a distribution, that is, $\nu_\mu(S_{T_p})$ is 1. We do so by showing that elements of $\mathsf{supp}(\nu_\mu)$ are in bijection with elements of $\mathsf{supp}(\mu)$. By construction, for every $\mathsf{d} \in \mathsf{post}$ such that $\nu_\mu(s,\mathsf{post}) > 0$, we have $\mathsf{d}(t_x) = \{0\}$ if and only if $\mathsf{d}(x) = r$ and $\mathsf{d}(t_x) = *$ if and only if $\mathsf{d}(x) = *$. This implies that $\mathsf{d}|_X = \mathsf{d}'|_X$ if and only if $\mathsf{d} = \mathsf{d}'$, as wanted. Another proof is obtained by taking $U = S_{T_p}$ in the proof of Theorem 3. □

If all variables of H_p are solvable, then its clock-translation with respect to all its variables will yield a probabilistic timed automaton. Knowing that the model-checking of probabilistic timed automata is decidable [10], it implies that the model-checking of solvable PHAs is decidable.

3.2 Probabilistic linear phase-portrait approximation

In this section, we show how to apply the linear phase-approximation method to a probabilistic hybrid automaton, and that it results in a rectangular hybrid automaton which simulates it.

Let $H_p = (V, X, \mathrm{Init}, \mathrm{Act}, \mathrm{Inv}, \mathrm{Flow}, prob, \langle pre \rangle, \langle pos \rangle)$ be a finitely branching PHA. The method of approximation in a probabilistic context follows the same kind of steps as the clock-translation, by going through an underlying non probabilistic hybrid automaton (this approach is also the one of Zhang et al. [13]). However, this translation is simpler, as well as smaller, here as no condition has to be satisfied by the non probabilistic automaton.

Step 1: We define, $H = (V, X, \mathrm{Init}, A, \mathrm{Inv}, \mathrm{Flow}, E, \mathit{Pre}, \mathrm{Set})$, the *underlying non probabilistic* hybrid automaton of H_p as follows:

- $A := \{a_\mu \mid \exists v \in V \text{ such that } \mu \in \mathit{prob}(v,a)\}$.
- For each $\mu \in \mathit{prob}(v,a)$: E will contain all $e = (v, a_\mu, v')$ such that there is some $(v', \mathsf{post}) \in \mathrm{supp}(\mu)$. $\mathit{Pre}(e) = \mathit{pre}(\mu)$. $\mathrm{Set}(e, \mathsf{a}) = \bigcup_{(v', \mathsf{post}) \in \mathrm{supp}(\mu)} \mathsf{post}[\mathsf{a}] \cap \mathit{pos}(\mu, v')$.

Step 2: We build the linear phase-portrait approximation of H:

$$T = (V_\theta, X_\theta, \mathrm{Init}_\theta, A, \mathrm{Inv}_\theta, \mathrm{Flow}_\theta, E_\theta, \mathit{Pre}_\theta, \mathrm{Set}_\theta).$$

Step 3: Finally, we build

$$T' := (V_\theta, X_\theta, \mathrm{Init}_\theta, \mathrm{Act}, \mathrm{Inv}_\theta, \mathrm{Flow}_\theta, \mathit{Prob}, \langle \mathit{Pre} \rangle, \langle \mathit{Pos} \rangle)$$

the probabilistic linear phase-portrait approximation of H_p from T as follows.

Let (v,i) be in V_θ, and a in Act. $\mathit{Prob}((v,i),a)$ contains two kinds of distributions:

- μ_θ defined as follows, for each $\mu \in \mathit{prob}(v,a)$. For each $e_\theta = ((v,i), a_\mu, (v', j)) \in E_\theta$ and post such that $(v', \mathsf{post}) \in \mathrm{supp}(\mu)$, we define

  $$\mu_\theta((v', j), \mathsf{post}) := \mu(v', \mathsf{post}),$$

  and μ_θ is zero elsewhere. Preconditions and postconditions are
  - $\mathit{Pre}(\mu_\theta) := \mathit{Pre}_\theta(e_\theta) \cap \mathrm{inv}_i^v$ and
  - $\mathit{Pos}(\mu_\theta, (v', j)) := \overline{\mathrm{inv}}_j^{v'} \cap \mathit{Pos}(\mu, v')$,

  where $\overline{\mathrm{inv}}_j^{v'} := \mathrm{inv}_j^{v'} \setminus (\cup_{k<j} \mathrm{inv}_k^{v'})$ defines a partition of $\mathrm{Inv}(v')$.
- if $a = \tau$, all μ_τ^j defined as

  $$\mu_\tau^j((v,j), \{*\}^X) := 1,$$

  that is, the valuation is unchanged during the silent transition from a copy of v to another. These transitions correspond to the edges $((v,i), \tau, (v,j)) \in E_\theta$. There is no special precondition or postcondition, and hence we set $\mathit{Pre}(\mu_\tau^j) := \mathrm{inv}_i^v$ and $\mathit{Pos}(\mu_\tau^j, (v', j)) := \mathrm{inv}_j^{v'}$.

Remark 1 *The use of postconditions, in presence of the star notation, is crucial in the apparently simple definition of μ_θ above, both to make it correct, and to indeed permit a simple and clean formulation. This is on one hand because of the reasons mentioned after Definition 6. On another hand, if we used the less general syntax involving reset sets instead of the star notation in distributions, the preconditions of μ_θ would have to take into account the invariant of the arrival state: if a probability is assigned to a pair that ends up not being valid because of the splitting of transitions, the probability ends up missing, i.e., μ_θ would not sum up to 1: the machinery to overcome this loss of probability would complicate a lot the notation. The use of $\overline{\mathrm{inv}}_j^{v'}$ is also crucial in the definition: it makes sure that we do not use, in μ_θ, a probability value from μ more than once. Indeed, some states get duplicated in the split (if some*

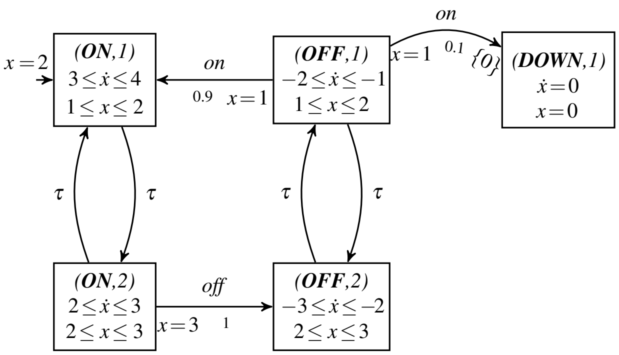

Figure 6: A phase-portrait approximation of the probabilistic thermostat

a *belongs to more than one element of* $\theta(v)$*), which has no negative impact in a non probabilistic HA, since duplicated transitions define bisimilar systems. However, in the probabilistic case, we must make sure that we do not give to both copies the probability value that was meant for one copy: by duplicating the value, we would lose the correspondence between copies and the original mode and we could also end up with a weight of more than one: this would result in a function that is not a distribution. We chose to put the probability on one of the duplicates because silent transitions make sure that the behavior is preserved. We could also have chosen to spread the probability to all duplicates uniformly.*

Example 8 *A linear phase-portrait approximation of the probabilistic thermostat automaton is obtained by slightly modifying the HA of Figure 3 and is illustrated in Figure 6. All transitions get probability* 1 *except for on-transitions from* $(OFF,1)$ *which have probability* 0.9 *to* $(ON,1)$ *and* 0.1 *to* $(DOWN,1)$*. Had we not restricted the postcondition of* $\mu_\theta((ON,j),-)$ *to* $\overline{\mathrm{inv}}_j^{ON}$*, it would give probability one to* $(ON,2)$ *and hence the distribution* $\mu_\theta \in prob((OFF,1),on)$ *would sum up to* 2*.*

We should now prove that the construction yields a valid PHA: this will be a consequence of the following theorem.

Theorem 4 *Any linear phase-approximation* H_θ *of a probabilistic PHA* H_p *simulates it: i.e.,* $H_\theta \preceq H_p$*. The split of a PHA is bisimilar to it.*

Proof. Let R be the relation that relates any $(v,\mathsf{a}) \in S_{H_p}$ to every $((v,i),\mathsf{a})$, with $1 \le i \le |\theta(v)|$. Let $(v,\mathsf{a}) \in S_{H_p}$, $a \in \mathrm{Act}$, $1 \le i \le |\theta(v)|$ and $\mu \in prob(v,a)$. We have to prove that for all $\mathsf{a} \in Pre(\mu_\theta)$ and any combination $\langle \mathsf{d}_j \rangle$ from μ_θ, we have $\mu_{\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(U) \le \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(R(U))$ for all $U \subseteq S_{H_p}$. In fact, we show equality. This does not give us a bisimulation because of the flow transitions which only satisfy simulation. The second equality below relies on the $\overline{\mathrm{inv}}_k^{v'}$'s being disjoint and cumulating to $\mathrm{inv}(v)$: indeed, for each k, there is only one $\mathsf{a}' \in \overline{\mathrm{inv}}_k^{v'}$ and all $\mathsf{a} \in \mathrm{inv}(v)$ is in one of those. Consequently the summation over k can be inserted freely.

$$\begin{aligned}
\mu_{\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(U) &= \sum_{(v',\mathsf{a}')\in U} \sum_{j=1}^{m} \{\, \mu(v_j, \mathsf{post}_j) \mid v_j = v',\ \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \textit{ where } m = |\mathrm{supp}(\mu)| \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in\overline{\mathrm{inv}}_k^{v'}}}^{|\theta(v')|} \sum_{j=1}^{m} \{\, \mu(v_j, \mathsf{post}_j) \mid v_i = v',\ \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in\mathrm{inv}_k^{v'}}}^{|\theta(v')|} \sum_{j=1}^{m} \{\, \mu_\theta((v_j,k), \mathsf{post}_j) \mid v_i = v', \mathsf{d}_j[\mathsf{a}] = \mathsf{a}' \} \\
&= \sum_{(v',\mathsf{a}')\in U} \sum_{\substack{k=1 \\ \mathsf{a}'\in\mathrm{inv}_k^{v'}}}^{|\theta(v')|} \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_k \rangle}((v',k),\mathsf{a}') \;=\; \mu_{\theta\mathsf{a}}^{\langle \mathsf{d}_j \rangle}(R(U))
\end{aligned}$$

This completes the proof of the first claim. The second claim follows easily. □

Example 9 *The linear phase-portrait approximation of the probabilistic thermostat automaton of Figure 6 simulates the thermostat automaton of Figure 5. In particular, consider the states* $s_1 = (OFF, 1)$ *and* $s_2 = (ON, 1)$ *of the original thermostat automaton, such that* $\mu(ON, 1) = 0.9$ *for* $\mu \in prob(OFF, on)$*. The states ((OFF,1), 1) and ((ON,1),1) simulate respectively* s_1 *and* s_2 *since* $\mu_\theta((ON, 1), 1) = 0.9$ *for* $\mu_\theta \in Prob((OFF, 1), on)$*.*

Corollary 2 *Phase-portrait approximation and splitting of finitely branching PHAs are PHAs.*

Proof. We only need to prove that every defined $\mu_{\theta\mathsf{a}}$ is a distribution, that is, the total probability out of $\mu_{\theta\mathsf{a}}$ is 1. This is guaranteed by μ_{a} being a distribution and by taking $U := S_\theta$ in the proof of Theorem 4; one obtains that $\mu_{\mathsf{a}}(S_H) = \mu_{\theta\mathsf{a}}(S_\theta)$. Since the distribution has nothing to do with the flow evolution, it is the same argument for both cases of approximation and splitting. □

Since an approximation of a probabilistic hybrid automaton is a rectangular PHA, its model-checking is decidable [11]. Therefore, by taking the right approximation to it, any probabilistic hybrid automaton can be verified.

4 Conclusion

In this paper, we proved that the two methods of Henzinger et al. [4], clock-translation and linear phase-portrait approximation, can also be applied in the probabilistic context to verify non-rectangular PHAs. The adaptation of the methods to PHAs were facilitated by a modification of the syntax over PHAs: a star notation to represent stability of a variable after a transition and postcondition functions. The advantage of adapting the methods instead of defining them from scratch is mainly that proving bi/simulation had only to be checked for discrete transitions. The correctness of the constructions is ensured by bi/simulation relations.

The first method, when it is applicable, results in a probabilistic timed automaton which satisfies exactly the same properties as the original PHA. The inconvenience of this method is that it requires, as in the non-probabilistic case, that the non-rectangular variables be solvable: all the equations induced by the flow evolution have solutions in $\mathbb{R}$.

For linear phase-portrait approximation, there is no restriction on the PHA, and its application results in a rectangular PHA that simulates the original one. When a safety property is satisfied by the rectangular approximation, we can assert that the original PHA satisfies the same property. However, in the case when a safety property is not satisfied by the approximation, more splits should be done; this could be costly in time depending on the property and the size of the PHA.

Side contributions of this paper are also: new additions to the definition of PHA that make them more general; a splitting construction on PHAs that results in a bisimilar PHA; a simpler description of the two techniques than what can be found in the original papers [4, 5]. About the additions to the definition of PHAs, it is interesting to note that the star notation and the postcondition function on distributions permit to represent constraints on variables in terms of set of valuations instead of predicates. When working with distributions, set of valuations are more natural than predicates.

Let us discuss how our approximation relates to the construction of a recent paper by Zhang et al [13], which also defines approximations for probabilistic hybrid automata. That paper also gives a method to over approximate the original automaton. It is clear that the latter is less abstract than the former since Zhang et al. define a *finite* approximation. As in the construction of Henzinger et al. [4], they start from a cover of the state space and abstract according to it. The difference is that the cover is over states instead of over the variables' space of values. More importantly, whereas we use the cover to "linearize" each piece that the cover defines, they group together all these states into one single state. The result is a finite probabilistic transition system, whereas we obtain a PHA. They prove, as we do, that their abstraction simulates the original PHA. However, their approximation is more abstract, which has the advantage of being smaller but of course with less information.

Future work includes the implementations of the technique into a probabilistic model checker and taking advantage of other approximation techniques that have been developed for probabilistic systems in order to widen the class of PHAs for which model-checking is supported.

Acknowledgement

J. Desharnais wishes to thank Marta Kwiatkowska and the Computing Laboratory of Oxford University for welcoming her during year 2009-2010.

References

[1] Rajeev Alur, Thomas A. Henzinger, Gerardo Lafferriere and George J. Pappas, *Discrete abstractions of hybrid systems*, Proc. IEEE, 2000, pp. 971–984. doi:10.1109/5.871304

[2] Christel Baier, *Polynomial time algorithms for testing probabilistic bisimulation and simulation*, Proc. 8th International Conference on Computer Aided Verification (CAV'96), LNCS, vol. 1102, Springer-Verlag, 1996, pp. 38–49. doi:10.1007/3-540-61474-5_57

[3] Josée Desharnais, François Laviolette, and Mathieu Tracol, *Approximate analysis of probabilistic processes: logic, simulation and games*, Proc. 5th International Conference on Quantitative Evaluation of Systems (QEST '08), IEEE Press, 2008, pp. 264–273. doi:10.1109/QEST.2008.42

[4] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-toi, *Algorithmic analysis of nonlinear hybrid systems*, IEEE Transactions on Automatic Control **43** (1996), 225–238. doi:10.1109/9.664156

[5] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi, *Hytech: A model checker for hybrid systems*, Int. Journal Software Tools for Technology Transfer (STTT) **1** (1997), no 1–2, 110–122. doi:10.1007/s100090050008

[6] Thomas A. Henzinger and Peter W. Kopke, *Discrete-time control for rectangular hybrid automata*, Theor. Comput. Sci. **221** (1999), no. 1-2, 369–392. doi:10.1016/S0304-3975(99)00038-9

[7] Isabella Kotini and George Hassapis, *Verification of rectangular hybrid automata models*, Journal of Systems and Software **79** (2006), no. 10, 1433–1443. doi:10.1016/j.jss.2006.01.016

[8] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang, *Symbolic model checking for probabilistic timed automata*, Information and Computation **205** (2007), no 7, 1027–1077. doi:10.1016/j.ic.2007.01.004

[9] Jr. Peter William Kopke, *The theory of rectangular hybrid automata*, Ph.D. thesis, Cornell University, Faculty of the Graduate School of Cornell University, 1996.

[10] Jeremy Sproston, *Analyzing subclasses of probabilistic hybrid automata.*, Proc. 2nd International Workshop on Probabilistic Methods in Verification, Eindhoven, University of Birmingham, Technical Report, CS-99-8, August 1999.

[11] Jeremy Sproston, *Decidable model checking of probabilistic hybrid automata*, Proc. 6th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT '00), LNCS, vol. 1926, Springer-Verlag, 2000, pp. 31–45. doi:10.1007/3-540-45352-0

[12] Jeremy Sproston, *Model checking of probabilistic timed and hybrid systems*, Ph.D. thesis, University of Birmingham, Faculty of Science, 2000.

[13] Lijun Zhang, Zhikun She, Stefan Ratschan, Holger Hermanns, and Ernst Moritz Hahn, *Safety verification for probabilistic hybrid systems*, Proc. 22nd International Conference on Computer Aided Verification (CAV'10), LNCS, vol. 6174, Springer-Verlag, 1996, pp. 196–211. doi:10.1007/978-3-642-14295-6_21

[14] G. Lafferriere, G. Pappas, and S. Yovine. *A new class of decidable hybrid systems*, Proc. 2nd International Workshop Hybrid Systems: Computation and Control (HSCC'99), LNCS, vol. 1569, Springer-Verlag, 1999, pp. 137–151. doi:10.1007/3-540-48983-5