Analyzing Subclasses of Probabilistic Hybrid Automata

Jeremy Sproston*

School of Computer Science, University of Birmingham,
Birmingham B15 2TT, United Kingdom
Email: J.Sproston@cs.bham.ac.uk

Abstract

Hybrid automata offer a framework for the description of systems with both discrete and continuous components, such as digital technology embedded in an analogue environment. Traditional uses of hybrid automata express choice of transitions purely in terms of nondeterminism, abstracting potentially significant information concerning the relative likelihood of certain behaviours. To model such probabilistic information, we present a variant of hybrid automata augmented with discrete probability distributions. We concentrate on classes of the model which have restricted continuous dynamics in order to obtain, via appropriate notions of simulation and bisimulation, decidable model checking algorithms for properties expressed in a probabilistic temporal logic.

1 Introduction

The proliferation of digital technology embedded in real-life environments has lead to increased interest in systems which are defined in terms of interaction between discrete and continuous components. Examples of such *hybrid systems* include robots, autonomous vehicles and chemical plants [VvS99]. In order to reason about the behaviour of hybrid systems in a precise manner, it is advantageous to utilize formal methods. An example of a framework for the formal description of hybrid systems which has been proposed by the computer science community is that of *hybrid automata* [ACH$^+$95], which represents the discrete component of the system as a finite, directed graph, and the continuous component as a multidimensional, real-valued space, the dynamics of which are defined by a set of differential equations or differential inequalities. Given such a formal description of a hybrid system, it may be possible to automatically verify that it satisfies requirements specified in a temporal logic. This *model checking* approach has received advanced support in the context of hybrid systems, both in terms of theoretical and practical developments [AHH96, HHMWT99]. Model checking is decidable only for certain, restricted classes of hybrid automata, although it has been noted that in many cases useful results can be obtained by verifying instances of the model for which no a priori guarantees for termination of the model checking algorithm exist.

Traditional approaches to the formal description of hybrid systems express the system model purely in terms of nondeterminism. However, it may be desirable to express the

*Supported in part by the Esprit working group FIREworks (23531).

relative likelihood of the system exhibiting certain behaviour. This notion is particularly important when considering fault-tolerant systems, in which the occurrence of the discrete event *malfunction* is less likely than the event *correct_behaviour*. Furthermore, it may be appropriate to model the likelihood of an event changing with respect to the continuous behaviour of the environment; for example, *malfunction* may become more likely if the system is operating at extreme temperatures or at high speeds. We may also wish to refer to the likelihood of certain temporal logic properties being satisfied by the hybrid system, and to have a model checking algorithm for verifying automatically the truth of these assertions.

Therefore, we present a model for hybrid automata which are described partially in terms of discrete probability distributions. This approach is inspired by the work of [KNSS99], which presents firstly a model of timed automata (a highly restricted class of hybrid automata) extended with such distributions, and secondly a decidable algorithm for verifying instances of this model against formulae of a probabilistic temporal logic, based on a similar algorithm for the non-probabilistic case presented in [ACD93]. Our new model, *probabilistic hybrid automata*, differs from traditional hybrid automata in that the edge relation of the graph representing the system's discrete component is both nondeterministic and probabilistic in nature. More precisely, instead of making a purely nondeterministic choice over the set of currently enabled edges, we choose amongst the set of enabled discrete probability distributions, each of which is defined over a finite set of edges. We then make a probabilistic choice as to which edge to take according to the selected distribution.

A substantial body of work has been devoted to exploring notions of *decidability* of non-probabilistic hybrid automata, particularly with regard to problems of reachability ("does a behaviour of the hybrid automaton reach a certain state?") and language emptiness ("does the hybrid automaton generate an admissible behaviour?"), both of which underly verification procedures. Two hybrid automata subclasses have received much attention in this context. *Rectangular automata* [HKPV98] feature differential inequalities which describe the continuous evolution of system variables taking place within piecewise-linear, convex envelopes, and can be used to state, for example, that a system variable increases between 1 and 3 units per second. In contrast, *multisingular automata* [ACH$^+$95] employ differential equations to describe the trajectories of system variables taking the form of piecewise constant rates (observe that timed automata are multisingular automata for which all system variables increase at the same rate as real-time). Decidability issues for these subclasses are addressed by utilizing refinement relations such as simulation and bisimulation in order to introduce a notion of state equivalence. Such relations can be categorized in one of two ways. Firstly, *time-abstract* simulation and bisimulation relations are obtained if the exact durations of continuous transitions are not considered. Certain classes of hybrid automata can be shown to be time-abstract similar or bisimilar to finite-state systems, and, as such, have a decidable model checking problem. In particular, timed automata and types of multisingular automata have finite time-abstract bisimulation quotients [AD94, ACH$^+$95], and a restricted subclass of rectangular automata with two continuous variables has finite time-abstract simulation quotients [HHK95]. Secondly, consideration of exact time durations results in *timed* simulation and bisimulation relations [LV96, Cer92]. It is possible to show the existence of timed simulation and bisimulation relations between classes of hybrid automata [HKPV98], although naturally such relations do not result in finite quotients. Furthermore, both types of simulation and bisimulation relations can be shown to preserve certain temporal logic formulae. The remit of this paper is to adapt such results for the case of probabilistic hybrid automata.

Noting that the definitions of simulation and bisimulation have been extended to mixed nondeterministic-probabilistic systems [SL95], we define such state equivalences in our context, and show that the resulting relations preserve certain properties of PBTL (Probabilistic Branching Time Logic) [BK98], which can be used to specify properties of probabilistic systems. The results of [HKPV98] concerning the transformation of multisingular automata into timed bisimilar timed automata are then extended to the probabilistic case. A model checking result can then be obtained by subsequent use of the algorithm of [KNSS99]. Furthermore, using the results of [HHK95], we show how to obtain a finite time-abstract simulation quotient of probabilistic rectangular automata with two continuous variables, thereby making model checking feasible for this class of model against certain PBTL formulae.

As far as we are aware, these are the first verification results for hybrid automata extended with probability. However, this probabilistic extension is relatively limited, particularly as probability has no direct effect on the continuous dynamics of the model. This means that probabilistic rectangular automata are suited to the modelling of systems for which relative likelihoods can be associated with discrete behaviour (such as the aforementioned fault-tolerant systems, or embedded technology operating according to randomized algorithms) rather than continuous behaviour (such as the air traffic control system affected by probabilistically determined wind conditions of [PLNS99]).

The paper proceeds as follows. Section 2 introduces probabilistic rectangular automata, and shows how the framework can be used to model a simple production line which is prone to malfunction. Section 3 explains how the semantics of probabilistic rectangular automata can be presented in terms of infinite-state, nondeterministic-probabilistic transition systems, and also defines simulation, bisimulation and PBTL in our context. Strategies for model checking probabilistic multisingular automata, via a translation into equivalent probabilistic timed automata, are presented in section 4, as is a method for the verification of probabilistic rectangular automata with two continuous variables against properties of a fragment of PBTL. To conclude, section 5 suggests further directions of research, including strategies to extend the verification results to more general classes of model, and to address the inefficiencies inherent in our verification methods.

2 Probabilistic rectangular automata

The purpose of this section is to present a model for probabilistic hybrid systems using the framework of hybrid automata. Noting that [KNSS99] presents a decidable model checking algorithm for timed automata augmented with discrete probability distributions, we focus on probabilistic extensions of subclasses of hybrid automata that have been shown to be equivalent, via bisimulation and simulation relations, to timed automata [OSY94, HKPV98]. We adopt the definition of hybrid automata for the maximal of these classes, namely that of *rectangular automata*, and introduce further subclasses as restrictions of this definition.

For a set Y, a *distribution* on Y is a function $p : Y \to [0,1]$ such that $p(y) \neq 0$ for at most countably many $y \in Y$, and $\sum_{y \in Y} p(y) = 1$. We use $\mu(Y)$ to denote the set of all distributions on Y. If Y contains on element, then a distribution over Y is called a *Dirac* *distribution*, and is denoted $\mathcal{D}(y)$ where $Y = \{y\}$.

Let $\mathcal{X} = \{x_1, ..., x_n\}$ be a set of real-valued variables. We write $\mathbf{a} \in \mathbb{R}^n$ for a vector of length n which assigns a *valuation* $\mathbf{a}_i$ to each variable $x_i \in \mathcal{X}$. A *rectangular inequality* over $\mathcal{X}$ is of the form $x_i \sim k$, where $x_i \in \mathcal{X}$, $\sim \in \{\leq, =, \geq\}$ and $k \in \mathbb{Q}$. A *rectangular predicate* over $\mathcal{X}$ is a conjunction of rectangular inequalities over $\mathcal{X}$. The set of rectangular predicates over $\mathcal{X}$ is denoted $\mathit{Rect}(\mathcal{X})$. Given the rectangular predicate R, the conjunction of rectangular inequalities that refer to variable x_i only is denoted by R_i. Then, for any $R \in \mathit{Rect}(\mathcal{X})$, $[\![R]\!]$ is the set of valuations for which R is true when each $x_i \in \mathcal{X}$ is replaced by its corresponding valuation $\mathbf{a}_i$. Intuitively, $[\![R]\!]$ describes the set of points in $\mathbb{R}^n$ such that R is true, and if $\mathbf{a} \in [\![R]\!]$, we say that $\mathbf{a}$ *satisfies* R. Furthermore, we call such a set $[\![R]\!]$ a *rectangle*, and a subset Z of $\mathbb{R}^n$ is referred to as *rectangular* if there exists some rectangular predicate R such that $Z = [\![R]\!]$. If a rectangle $[\![R]\!]$ contains a single point, then $[\![R]\!]$ is called a *singleton*. We also denote by $\dot{\mathcal{X}}$ the set of first derivatives of variables of $\mathcal{X}$ with respect to time, and, for each $x \in \mathcal{X}$, $\dot{x}$ is the first derivative of x with respect to time (that is, $\dot{x} = \frac{dx}{dt}$ where t represents time).

Following the approach of [KNSS99] for extending timed automata with discrete probability distributions, we now extend rectangular automata with the same type of probabilistic information.

Definition 1 (Probabilistic rectangular automaton) *A* probabilistic rectangular automaton $H = (\mathcal{X}, V, L, \Theta, \mathit{init}, \mathit{inv}, \mathit{flow}, \mathit{prob}, \mathit{pre})$ *comprises of the following components:*

Variables. $\mathcal{X}$ *is a finite set of real-valued* variables. *If $\mathcal{X}$ contains n elements, we say that H is an n-dimensional (or nD) probabilistic rectangular automaton.*

Control modes. V *is a finite set of* control modes.

Labelling function. $L : V \to 2^{\mathrm{AP}}$ *is a function assigning a finite set of atomic propositions to each control mode.*

Observations. Θ *is a finite set of* observations.

Initial conditions. $\mathit{init} : V \to \mathit{Rect}(\mathcal{X})$ *is a function that maps every control mode to an* initial condition *in $\mathit{Rect}(\mathcal{X})$.*

Invariant conditions. $\mathit{inv} : V \to \mathit{Rect}(\mathcal{X})$ *is a function that maps every control mode to an* invariant condition *in $\mathit{Rect}(\mathcal{X})$. The probabilistic rectangular automaton may remain in a control mode only if its invariant condition is satisfied by the current values of the variables.*

Flow conditions. $\mathit{flow} : V \to \mathit{Rect}(\dot{\mathcal{X}})$ *is a function that maps every control mode to a* flow condition *in $\mathit{Rect}(\dot{\mathcal{X}})$. Such conditions describe the form of the continuous evolution of the model.*

Probability distributions. $\mathit{prob} : V \to \mathcal{P}_{fn}(\mu(V \times \mathit{Rect}(\mathcal{X}) \times 2^{\mathcal{X}} \times \Theta))$ *is a function that maps every control mode to a finite, non-empty set of discrete probability distributions over the set of control modes, the set of rectangular predicates of $\mathcal{X}$, the powerset of $\mathcal{X}$ and the observation set. Therefore, each control mode v will have a set of associated probability distributions, denoted by $\mathit{prob}(v) = \{p_v^1, ..., p_v^m\}$ for some $m \in \mathbb{N}$.*

Pre-conditions. *$pre : prob(v) \to Rect(\mathcal{X})$ is a function that maps every probability distribution associated with a control mode to a* pre-condition *in $Rect(\mathcal{X})$.*

As with non-probabilistic hybrid automata, control of the model commences in a mode v with a variable valuation $\mathbf{a}$ such that $\mathbf{a}$ satisfies $init(v)$. When control of a probabilistic rectangular automaton remains in a given mode $v \in V$, the values of the real-valued variables in $\mathcal{X}$ change continuously with respect to time. Such continuous evolution is determined by the mode's flow condition $flow(v)$, which, for each variable $x_i \in \mathcal{X}$, describes the rectangle $[\![flow(v)_i]\!]$, which in turn gives (possibly unbounded) upper and lower limits on the values that the first derivative $\dot{x_i}$ may take in v. A discrete transition, henceforth called a *control switch*, from v to another mode may take place if the pre-condition $pre(p_v)$ of a probability distribution p_v in the set $prob(v)$ is satisfied by the current values of the variables (that is, if a distribution associated with the current mode is *enabled*). Conversely, such a control switch *must* take place if the invariant condition of the current mode, $inv(v)$, is no longer satisfied by the current variable values. Observe that, if an invariant is violated when no probability distribution is enabled, then a pathological situation in which time cannot advance (also referred to as *timelock*) will arise. Therefore, the invariant and pre-conditions are subject to the assumption that, if allowing any amount of time to elapse would violate the invariant condition of the current mode v, a pre-condition of at least one probability distribution in $prob(v)$ must be satisfied.

Given that it has been decided to make a control switch via a particular distribution $p_v \in prob(v)$, where the current valuation $\mathbf{a}$ satisfies $pre(p_v)$, then a probabilistic choice as to which mode will be switched to, and the discrete changes to the continuous variables, is made over a finite set of possibilities. More precisely, with probability $p_v(w, post, X, \theta)$, a transition corresponding to the observation θ is made to mode $w \in V$ with the valuation $\mathbf{b}$, such that $\mathbf{b}$ satisfies *post* and, for every variable $x_i \in \mathcal{X} \setminus X$, $\mathbf{b}_i = \mathbf{a}_i$. Rectangular predicates such as *post* are referred to as *post-conditions*, and variable sets such as X are referred to as *reset sets*. Together, post-conditions and reset sets determine the effects that a probabilistic rectangular automaton's control switches have on its continuous variables.

Example. As an example of a probabilistic hybrid system, consider a production line which may be subject to faults as a job is processed. When no job is being processed, an item can arrive at the production line at any time. It is then placed on a conveyor belt comprising of two sections, the length of each being three metres. The item proceeds at between 2 and 3 metres per minute in the first section, and between 1 and 4 metres per minute in the second. As an item passes between the sections of the conveyor belt, there is a 1% chance of the occurrence of a fault, which necessitates production being halted for two minutes. After this time, or if a fault did not occur originally, production continues until the item reaches the end of the second conveyor belt. An example of a property that we may want this probabilistic hybrid system to satisfy is, "with probability 0.6 and within 5 minutes, a job should have progressed at least 4 metres along the production line".

The system can be modelled as the probabilistic rectangular automaton H_1, as shown in figure 1. The control modes are *idle*, which represents the situation in which no job is being processed, *segment1* and *segment2*, in which an item is being successfully processed along the first and second sections of the conveyor belt respectively, and *delay*, in which the production line is halted because of a fault. The continuous variable x represents the distance in metres which the current item has progressed along the production line, and

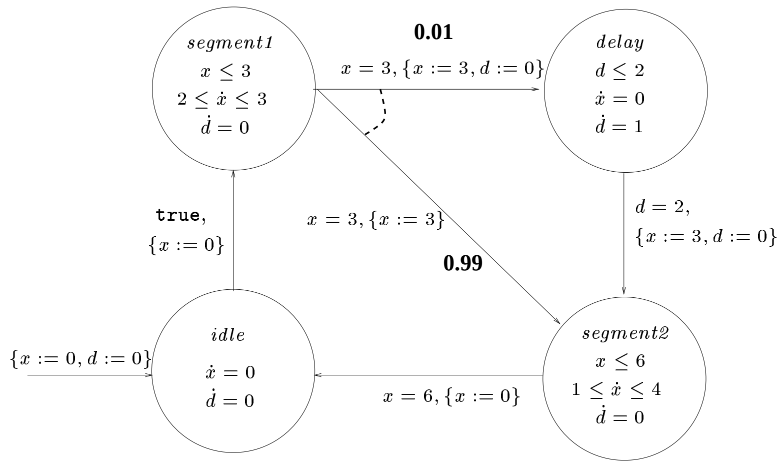

Figure 1. The probabilistic rectangular automaton H_1.

d is used in the presence of a fault to model the time elapsed since production has been halted. In the usual manner for hybrid automata, invariant conditions (such as $x \leq 3$) and flow conditions (such as $2 \leq \dot{x} \leq 3$) are written in the body of control modes. Edges are labelled with their pre-conditions (for example, $d = 2$) and a special notation combining post-conditions and reset sets (for example, $\{x := 0\}$ means that x takes a value in the interval $[0, 0]$, with all other variables remaining the same, and corresponds to the reset set X and a post-condition $post = (x = 0)$). The edges joined by the dotted arc are the only two that are probabilistic; the edge whose target is *delay* represents the occurrence of a fault, and as such is labelled with probability 0.01, whereas the edge with *segment2* as its target represents successful progression to the next stage of the production process, and is labelled with the probability 0.99. Intuitively, when control is in *segment1* and x reaches 3, a probabilistic choice is made over these two edges. All other probability distributions assign probability 1 to a single edge, and therefore the probability label has been omitted. Also omitted for simplicity are observations, and invariants of the form `true`.

Next, we present a number of subclasses of rectangular hybrid automata, where, in each case, the criterion for classification is the type of continuous evolution permitted. A *probabilistic multisingular automaton* M is a probabilistic rectangular automaton such that, for each $v \in V$ and for each $x_i \in \mathcal{X}$, $\mathit{flow}(v)_i$ is $\dot{x}_i = k$ for some $k \in \mathbb{N}$. A *probabilistic stopwatch automaton* S is a probabilistic multisingular automaton such that, for each $v \in V$ and for each $x_i \in \mathcal{X}$, $\mathit{flow}(v)_i$ is $\dot{x}_i = 1$ or $\dot{x}_i = 0$. A *probabilistic timed automaton* T is a probabilistic stopwatch automaton such that, for each $v \in V$ and for each $x_i \in \mathcal{X}$, $\mathit{flow}(v)_i$ is $\dot{x}_i = 1$. If we further restrict the *post* conditions of all control switches of a probabilistic timed automaton T to the assignment of 0 for all variables in the corresponding reset set X, then the model agrees with the probabilistic timed graph of [KNSS99].

The results presented in the remainder of this paper are dependent on the imposition of two restrictions, which are called *initialization* and *deterministic jumps* [HKPV98]. The probabilistic rectangular automaton H is *initialized* if, for every pair of modes $v, w \in V$, and every $x_i \in \mathcal{X}$ for which $[\![\mathit{flow}(v)_i]\!] \neq [\![\mathit{flow}(w)_i]\!]$, then if there exists $p_v \in \mathit{prob}(v)$, $\mathit{post} \in \mathit{Rect}(\mathcal{X})$, $X \subseteq \mathcal{X}$, and $\theta \in \Theta$ such that $p_v(w, \mathit{post}, X, \theta) > 0$, then $x_i \in X$. Intuitively, if the execution of a control switch results in a variable changing the condition on its continuous evolution, then the value of that variable must be reinitialized. The probabilistic rectangular automaton H has *deterministic jumps* if (1) for a control mode $v \in V$, the rectangle $[\![init(v)]\!]$ is a singleton, and for all other $v' \in V \setminus \{v\}$, $init(v') = \mathtt{false}$, [1] and (2) for every $v, w \in V$, and $p_v \in prob(v)$, if there exists $post \in Rect(\mathcal{X})$, $X \subseteq \mathcal{X}$ and $\theta \in \Theta$ such that $p_v(w, post, X, \theta) > 0$, then, for every $x_i \in X$, $[\![post_i]\!]$ is a singleton. Intuitively, the second requirement states that, for every control switch, each variable either remains unchanged or is deterministically reset to a new value. Henceforth, we assume that all probabilistic rectangular automata are initialized and have deterministic jumps. Note that H_1 is a probabilistic rectangular automaton which satisfies these assumptions.

Remark 1. Although many published case studies using non-probabilistic hybrid automata to model real-life systems feature models with deterministic jumps (for example, see [Cor94, SMF97]), the combination of this assumption and that of initialization is more severe. In particular, together they compromise the use of rectangular automata to conservatively over-approximate the continuous evaluation of hybrid systems that exhibit 'non-linear' behaviour [HHWT98], such as that defined by the differential equation $\dot{x} = -x + 5$. Although it is straightforward to construct an initial, approximative rectangular automaton of a non-linear hybrid system, if the verification result is inconclusive, then the method for the construction of a tighter approximation presented in [HHWT98] cannot be used in our context, as it requires additional control switches that do not satisfy both the assumptions of initialization and deterministic jumps.

Remark 2. A probabilistic rectangular automaton with rational constants, $H^{\mathbb{Q}}$, can be transformed into an equivalent probabilistic rectangular automaton with integer constants, $H^{\mathbb{N}}$, by simply multiplying each constant used in the description of $H^{\mathbb{Q}}$ by the least common multiple of the denominators of all such constants [AD94].

3 Semantics of probabilistic rectangular automata

This section presents a class of discrete structures that make transitions according to both nondeterministic and probabilistic choice. Firstly, a number of standard concepts relating to probabilistic-nondeterministic structures are revised, including the resolution of nondeterminism, obtaining probability measures over behaviours, and state equivalence. Secondly, the probabilistic temporal logic PBTL is presented, and its relationship with various notions of state equivalence are explored. Finally, this section shows how probabilistic-nondeterministic structures can be used as underlying models of probabilistic rectangular automata.

3.1 Probabilistic structures

The underlying transition systems of probabilistic hybrid automata will take the form of *probabilistic structures.* These are identical to the probabilistic automata of [SL95]; we have changed the name to avoid confusion with that of the 'higher-level' model of hybrid automata.

Definition 2 (Probabilistic structure) *A* probabilistic structure $\mathcal{S}$ *is a tuple* $(Q, Q^0, \Sigma,$ *Steps*$)$*, where* Q *is a (possibly infinite) set of states,* $Q^0 \subseteq Q$ *is a non-empty set of initial*

[1] For simplicity, this clause is stronger than that of [HKPV98], and guarantees that there is a single initial state of H.

states, Σ is a set of events, and Steps is a function which assigns to each state a non-empty set $Steps(q)$ of distributions on $\Sigma \times Q$.

A *transition* of $\mathcal{S}$ from state q, denoted by $q \xrightarrow{p}_{\sigma} q'$, comprises of a nondeterministic choice of a probability distribution $p \in Steps(q)$, followed by a probabilistic choice of an event-state pair (σ, q') according to p such that $p(\sigma, q') > 0$. Paths of a probabilistic structure arise by resolving both the nondeterministic and probabilistic choices, and take the form of sequences of transitions. Formally, a *path* of $\mathcal{S}$ is a non-empty finite or infinite sequence of transitions:

$$\omega = q_0 \xrightarrow{p_0}_{\sigma_0} q_1 \xrightarrow{p_1}_{\sigma_1} q_2 \xrightarrow{p_2}_{\sigma_2} \cdots .$$

Note that the special case $\omega = q$, for some $q \in Q$, is also a path.

The following notation is employed when reasoning about paths. Take any path ω; the first state of ω is denoted by $first(\omega)$. The length of a path, $|\omega|$ is defined in the usual way: if ω is the path $\omega = q$, then $|\omega| = 0$; if ω is the finite path $\omega = q_0 \xrightarrow{p_0}_{\sigma_0} q_1 \xrightarrow{p_1}_{\sigma_1} \cdots \xrightarrow{p_{n-1}}_{\sigma_{n-1}} q_n$, then $|\omega| = n$; if ω is an infinite path, then we let $|\omega| = \infty$. If $k \leq |\omega|$ then $\omega(k)$ denotes the k-th state of ω and if $k < |\omega|$ then $step(\omega, k)$ is the probability distribution associated with the k-th transition (that is, $\omega(k) = q_k$ and $step(\omega, k) = p_k$). If $k < |\omega|$, then let $\omega^{(k)} = q_0 \xrightarrow{p_0}_{\sigma_0} q_1 \xrightarrow{p_1}_{\sigma_1} \cdots \xrightarrow{p_{k-1}}_{\sigma_{k-1}} q_k$ ($\omega^{(k)}$ is the k-th prefix of ω).

Sets of labelled paths are denoted in the following way. $Path_{fin}$ is the set of finite paths, and $Path_{fin}(q)$ is the set of paths ω in $Path_{fin}$ such that $first(\omega) = q$. $Path_{ful}$ is the set of infinite paths. $Path_{ful}(q)$ is the set of paths ω in $Path_{ful}$ such that $first(\omega) = q$.

3.1.1 Adversaries of probabilistic structures

We now introduce *adversaries* of probabilistic structures as functions which resolve all the nondeterministic choice of the model.

Definition 3 (Adversary) *An* adversary *of a probabilistic structure $\mathcal{S} = (Q, Q^0, \Sigma, Steps)$ is a function A mapping every finite path ω of $\mathcal{S}$ to a probability distribution p such that $A(\omega) \in Steps(last(\omega))$. Let $\mathcal{A}$ be the set of all adversaries of $\mathcal{S}$.*

For an adversary A of a probabilistic structure $\mathcal{S} = (Q, Q^0, \Sigma, Steps)$, we define $Path^A_{fin}$ to be the set of finite paths such that $step(\omega, i) = A(\omega^{(i)})$ for all $0 \leq i < |\omega|$, and $Path^A_{ful}$ to be the set of paths in $Path_{ful}$ such that $step(\omega, i) = A(\omega^{(i)})$ for all $i \in \mathbb{N}$. Furthermore, $Path^A_{ful}(q)$ is defined to be the set of paths of $Path^A_{ful}$ such that, for all $\omega \in Path^A_{ful}(q)$, $first(\omega) = q$.

With each adversary we associate a sequential (in general, infinite-state) Markov chain, which can be viewed as a set of paths in $\mathcal{S}$. Formally, if A is an adversary of $\mathcal{S}$, then $MC^A = (Path^A_{fin}, \mathbf{P}^A)$ is a Markov chain where:

$$\mathbf{P}^A(\omega, \omega') = \begin{cases} p(\sigma, q) & \text{if } A(\omega) = p \text{ and } \omega' = \omega \xrightarrow{p}_{\sigma} q \\ 0 & \text{otherwise.} \end{cases}$$

For any probabilistic structure, let $\mathcal{F}_{Path^A}$ be the smallest σ-algebra on $Path^A_{ful}$ which contains the sets $\{\omega \,|\, \omega \in Path^A_{ful}$ and ω' is a prefix of $\omega\}$ for all $\omega' \in Path_{fin}$.

We now define a measure $Prob^A$ on the σ-algebra $\mathcal{F}_{Path^A}$, by first defining the following function on the set of finite paths $Path^A_{fin}$.

Definition 4 *Let $Prob^A_{fin} : Path^A_{fin} \to [0,1]$ be the mapping inductively defined on the length of paths in $Path^A_{fin}$ as follows. If $|\omega| = 0$, then $Prob^A_{fin}(\omega) = 1$.*

Let ω' be a path in $Path^A_{fin}$. If $\omega' = \omega \xrightarrow{p}_\sigma q$ for some $\omega \in Path^A_{fin}$, $\sigma \in \Sigma$, and $q \in Q$, then we let:

$$Prob^A_{fin}(\omega') = Prob^A_{fin}(\omega) \cdot \mathbf{P}^A(\omega, \omega') \ .$$

Definition 5 *The measure $Prob^A$ on $\mathcal{F}_{Path^A}$ is the unique measure such that:*

$$Prob^A(\{\omega \,|\, \omega \in Path^A_{ful} \text{ and } \omega' \text{ is a prefix of } \omega\}) = Prob^A_{fin}(\omega').$$

3.1.2 State equivalence for probabilistic structures

Next, we introduce two notions of equivalence for probabilistic structures, namely those of *probabilistic bisimulation and simulation.* In the standard manner, the concept of weight functions [JL91] is used to provide the basis of the definition of simulation, and bisimulation is then defined as a symmetric simulation [SL95, Bai98]. We write $q \xrightarrow{p}$ if there exists a transition $q \xrightarrow{p}_\sigma q'$ for some $(\sigma, q') \in \Sigma \times Q$.

Definition 6 (Weight function) *Let $\mathcal{R} \subseteq Q_1 \times Q_2$ be a relation between the two sets Q_1, Q_2, Σ a finite set, and p_1, p_2 distributions such that $p_1 \in \mu(\Sigma \times Q_1)$ and $p_2 \in \mu(\Sigma \times Q_2)$. A* weight function *for (p_1, p_2) with respect to $\mathcal{R}$ is a function $w : Q_1 \times \Sigma \times Q_2 \to [0,1]$ such that, for all $\sigma \in \Sigma$ and $q_1 \in Q_1$, $q_2 \in Q_2$:*

1. *if $w(q_1, \sigma, q_2) > 0$, then $(q_1, q_2) \in \mathcal{R}$, and*
2. *$\sum_{q' \in Q_2} w(q_1, \sigma, q') = p_1(\sigma, q_1)$, and $\sum_{q' \in Q_1} w(q', \sigma, q_2) = p_2(\sigma, q_2)$.*

Definition 7 (State equivalence for a single probabilistic structure) *Let $\mathcal{S} = (Q, Q^0, \Sigma, Steps)$ be a probabilistic structure.*

- *A* simulation *of $\mathcal{S}$ is a relation $\mathcal{R} \subseteq Q \times Q$ such that, for each $(q_1, q_2) \in \mathcal{R}$, if $q_1 \xrightarrow{p_1}$, then $q_2 \xrightarrow{p_2}$ for some p_2 such that there exists a weight function w for (p_1, p_2) with respect to $\mathcal{R}$. We say that q_2* simulates *q_1, denoted $q_1 \sqsubseteq q_2$, iff there exists a simulation which contains (q_1, q_2).*

- *A* mutual simulation *of $\mathcal{S}$ is an equivalence relation $\mathcal{R}$ over Q such that, for each $(q_1, q_2) \in \mathcal{R}$, $q_1 \sqsubseteq q_2$. Two states q_1, q_2 are called* mutually similar, *denoted by $q_1 \approx q_2$, iff there exists a mutual simulation which contains (q_1, q_2).*

- *A* bisimulation *of $\mathcal{S}$ is an equivalence relation $\mathcal{R}$ over Q such that, for each $(q_1, q_2) \in \mathcal{R}$:*

  1. *if $q_1 \xrightarrow{p_1}$, then $q_2 \xrightarrow{p_2}$ for some p_2 such that there exists a weight function w for (p_1, p_2) with respect to $\mathcal{R}$; and,*
  2. *if $q_2 \xrightarrow{p'_2}$, then $q_1 \xrightarrow{p'_1}$ for some p'_1 such that there exists a weight function w' for (p'_1, p'_2) with respect to $\mathcal{R}$.*

  Two states q_1, q_2 are called bisimilar, *denoted by $q_1 \simeq q_2$, iff there exists a bisimulation which contains (q_1, q_2).*

The notions of mutual simulation and bisimulation can also be extended to relations between probabilistic structures.

Definition 8 (Simulation and bisimulation between probabilistic structures) *Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two probabilistic structures with the same event set Σ. A* mutual simulation (bisimulation) between $\mathcal{S}_1$ and $\mathcal{S}_2$ *is an equivalence relation $\mathcal{R}$ over $Q_1 \cup Q_2$ such that:*

1. *for each $q_1 \in Q_1^0$, there exists a $q_2 \in Q_2^0$ such that $(q_1, q_2) \in \mathcal{R}$, and vice versa, and*
2. *for each $(q_1, q_2) \in \mathcal{R}$, $q_1 \approx q_2$ ($q_1 \simeq q_2$).*

Two probabilistic structures, $\mathcal{S}_1 = (Q_1, Q_1^0, \Sigma, Steps_1)$ and $\mathcal{S}_2 = (Q_2, Q_2^0, \Sigma, Steps_2)$, are called mutually similar (bisimilar), denoted by $\mathcal{S}_1 \approx \mathcal{S}_2$ ($\mathcal{S}_1 \simeq \mathcal{S}_2$), iff there exists a mutual simulation (bisimulation) between $\mathcal{S}_1$ and $\mathcal{S}_2$.

3.2 Probabilistic Branching Time Logic

We now present the probabilistic temporal logic PBTL (Probabilistic Branching Time Logic) [BK98], which can be used to specify properties of probabilistic-nondeterministic systems such as probabilistic rectangular automata. In brief, PBTL is an extension of the branching temporal logic CTL in which the until operator includes a bound on probability. For example, the property, "with probability 0.1 or greater, the job will be processed along the first conveyor belt until the production line is subject to a delay", is represented by the PBTL formula $[(\mathit{segment1})\forall\mathcal{U}(\mathit{delay})]_{\geq 0.1}$, where *segment1* and *delay* are atomic propositions labelling the appropriate states. Note that PBTL is essentially identical to the logic pCTL presented in [BdA95].

Formulae of PBTL are interpreted over probabilistic structures augmented with state labelling functions.

Definition 9 (Labelled probabilistic structure) *A* labelled probabilistic structure *is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{L})$, where $\mathcal{S} = (Q, Q^0, \Sigma, Steps)$ is a probabilistic structure, and $\mathcal{L} : Q \to 2^{\mathrm{AP}}$ is a function assigning a finite set of atomic propositions to each state.*

Both simulation and bisimulation can be extended in a straightforward manner to labelled probabilistic structures. We merely impose the additional requirement that related states are labelled with the same set of atomic propositions.

Definition 10 (State equivalence for labelled probabilistic structures) *Let $\mathcal{S} = (Q, Q^0, \Sigma, Steps)$ be a probabilistic structure, and $\mathcal{M} = (\mathcal{S}, \mathcal{L})$ be a labelled probabilistic structure. Then, given the relation $\mathcal{R} \subseteq Q \times Q$, we let the* relation $\mathcal{R}$ with respect to AP *be the greatest relation $\mathcal{R}' \subseteq \mathcal{R}$ such that, for each $(q_1, q_2) \in \mathcal{R}'$, $\mathcal{L}(q_1) = \mathcal{L}(q_2)$.*

Two states q_1, q_2 of a labelled probabilistic structure are similar (mutually similar; bisimilar) with respect to AP, denoted $q_1 \sqsubseteq_{\mathrm{AP}} q_2$ ($q_1 \approx_{\mathrm{AP}} q_2$; $q_1 \simeq_{\mathrm{AP}} q_2$) iff there exists a simulation (mutual simulation; bisimulation) of $\mathcal{M}$ with respect to AP which contains (q_1, q_2).

We say that two labelled probabilistic structures, $\mathcal{M}_1$ and $\mathcal{M}_2$, are mutually similar (bisimilar) with respect to AP, denoted $\mathcal{M}_1 \approx_{\mathrm{AP}} \mathcal{M}_2$ ($\mathcal{M}_1 \simeq_{\mathrm{AP}} \mathcal{M}_2$), iff there exists a mutual simulation (bisimulation) between $\mathcal{M}_1$ and $\mathcal{M}_2$ with respect to AP.

The syntax and semantics of PBTL are now defined. Note that we omit the 'next step' and 'bounded until' operators of [BK98], because the dense time domain of hybrid automata renders their standard interpretations meaningless.

Definition 11 (Syntax of PBTL**)** *The syntax of* PBTL *is defined as follows:*

$$\Phi ::= \texttt{true} \;\mid\; a \;\mid\; \Phi \wedge \Phi \;\mid\; \neg\Phi \;\mid\; [\Phi \,\exists\,\mathcal{U}\, \Phi]_{\sqsupseteq\lambda} \;\mid\; [\Phi \,\forall\,\mathcal{U}\, \Phi]_{\sqsupseteq\lambda}$$

where $a \in \mathrm{AP}$ *is an atomic proposition,* $\lambda \in [0,1]$*, and* $\sqsupseteq$ *is either* $\geq$ *or* $>$*.*

Definition 12 (Satisfaction Relation for PBTL**)** *Given a labelled probabilistic structure* $\mathcal{M}$ *and a set* $\mathcal{A}$ *of adversaries on* $\mathcal{M}$*, then for any state* q *of* $\mathcal{M}$*, and* PBTL *formula* Φ*, the satisfaction relation* $q \models_{\mathcal{A}} \Phi$ *is defined inductively as follows:*

$$\begin{array}{lll}
q \models_{\mathcal{A}} \texttt{true} & \multicolumn{2}{l}{\text{for all } q \in Q} \\
q \models_{\mathcal{A}} a & \Leftrightarrow & a \in \mathcal{L}(q) \\
q \models_{\mathcal{A}} \Phi_1 \wedge \Phi_2 & \Leftrightarrow & q \models_{\mathcal{A}} \Phi_1 \text{ and } q \models_{\mathcal{A}} \Phi_2 \\
q \models_{\mathcal{A}} \neg\Phi & \Leftrightarrow & q \not\models_{\mathcal{A}} \Phi \\
q \models_{\mathcal{A}} [\Phi_1 \,\exists\mathcal{U}\, \Phi_2]_{\sqsupseteq\lambda} & \Leftrightarrow & \mathit{Prob}^{A}(\{\omega \mid \omega \in \mathit{Path}^{A}_{\mathit{ful}}(q) \;\&\; \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}) \sqsupseteq \lambda \\
 & & \;\text{for some } A \in \mathcal{A} \\
q \models_{\mathcal{A}} [\Phi_1 \,\forall\mathcal{U}\, \Phi_2]_{\sqsupseteq\lambda} & \Leftrightarrow & \mathit{Prob}^{A}(\{\omega \mid \omega \in \mathit{Path}^{A}_{\mathit{ful}}(q) \;\&\; \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}) \sqsupseteq \lambda \\
 & & \;\text{for all } A \in \mathcal{A} \\
\omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2 & \Leftrightarrow & \text{there exists } i \in \mathbb{N}, \text{ such that } \omega(i) \models_{\mathcal{A}} \Phi_2, \\
 & & \text{and, for all } j \in \mathbb{N} \text{ such that } 0 \leq j < i,\ \omega(j) \models_{\mathcal{A}} \Phi_1
\end{array}$$

If a probabilistic structure $\mathcal{M}$ has a unique initial state (that is $Q^0 = \{q^0\}$), then we say that $\mathcal{M}$ satisfies the PBTL formula Φ, written $\mathcal{M} \models_{\mathcal{A}} \Phi$, iff $q^0 \models_{\mathcal{A}} \Phi$.

Observe that the set $\mathcal{A}$ of adversaries used in the satisfaction relation need not be the full set of adversaries of the labelled probabilistic structure $\mathcal{M}$. Instead, it may be appropriate to use a subset of $\mathcal{A}$; indeed, in section 3.3.1, we explain how only a subset of adversaries correspond to realisable behaviour of a hybrid automaton.

We now introduce $\forall$PBTL as a certain fragment of PBTL, the properties of which are preserved by simulation.

Definition 13 (Syntax of $\forall$PBTL**)** *The syntax of* $\forall$PBTL *is defined as follows:*

$$\Phi ::= \texttt{true} \;\mid\; \texttt{false} \;\mid\; a \;\mid\; \Phi \wedge \Phi \;\mid\; \Phi \vee \Phi \;\mid\; [\Phi \,\forall\,\mathcal{U}\, \Phi]_{\sqsupseteq\lambda}$$

where $a \in \mathrm{AP}$ *is an atomic proposition,* $\lambda \in [0,1]$*, and* $\sqsupseteq$ *is either* $\geq$ *or* $>$*.*

Definition 14 (Satisfaction Relation for $\forall$PBTL**)** *Given a labelled probabilistic structure* $\mathcal{M}$ *and a set* $\mathcal{A}$ *of adversaries on* $\mathcal{M}$*, then for any state* q *of* $\mathcal{M}$*, and* $\forall$PBTL *formula* Φ*, the satisfaction relation* $q \models_{\mathcal{A}} \Phi$ *is defined inductively according to the rules in Definition 12 and the following:*

$$\begin{array}{lll}
q \not\models_{\mathcal{A}} \texttt{false} & \multicolumn{2}{l}{\text{for all } q \in Q} \\
q \models_{\mathcal{A}} \Phi_1 \vee \Phi_2 & \Leftrightarrow & q \models_{\mathcal{A}} \Phi_1 \text{ or } q \models_{\mathcal{A}} \Phi_2
\end{array}$$

The following theorem states that simulation with respect to atomic propositions preserves $\forall$PBTL formulae.

Theorem 15 *Let $\mathcal{M}$ be a labelled probabilistic structure with an associated set $\mathcal{A}$ of adversaries, and let Φ be a $\forall$PBTL formula. If $q_1 \sqsubseteq_{\mathrm{AP}} q_2$, for $q_1, q_2 \in Q$, then $q_2 \models_{\mathcal{A}} \Phi$ implies $q_1 \models_{\mathcal{A}} \Phi$.*

Proof. The proof borrows ideas from a similar result in [SL95], and proceeds by induction on the structure of Φ. However, the cases for atomic propositions and boolean connectives are trivial, and hence we concentrate on $[\Phi_1 \forall\mathcal{U} \Phi_2]_{\sqsupseteq\lambda}$.

For any adversary A, let $\xi_A = \mathit{Prob}^A(\{\omega \,|\, \omega \in \mathit{Path}^A_{\mathit{ful}}(q_1) \,\&\, \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\})$. The idea underlying the proof is that, for every adversary A_1, we can find an adversary A_2 such that $\xi_{A_2} \leq \xi_{A_1}$. Such an adversary A_2 is constructed from A_1 using the simulation relation; however, note that, for a given adversary A_1, there may be more than one adversary that can be built from $\sqsubseteq_{\mathrm{AP}}$. We denote the set of such adversaries as $\mathrm{ADV}(A_1)$. To explain how the construction of adversaries in $\mathrm{ADV}(A_1)$ takes place, first assume that $\mathrm{ADV}(A_1)$ contains only one adversary, and consider the states q_1, q_2. In q_1, A_1 makes the choice of a particular probability distribution p_1. By $q_1 \sqsubseteq_{\mathrm{AP}} q_2$, there exists $m \geq 1$ distributions p_2 enabled in q_2 such that there exists a weight function for (p_1, p_2). Therefore, we split the adversary in $\mathrm{ADV}(A_1)$ into m adversaries, each with a different choice of probability distribution for their first transition. Now take a particular $A_2 \in \mathrm{ADV}(A_1)$, and a state q'_2 such that $A_2(q_2) = p_2$ and $p_2(\sigma, q') > 0$ for some $\sigma \in \Sigma$. Then, by Definition 6, there exists a q'_1 such that $p_1(\sigma, q'_1) > 0$ for $A_1(q_1) = p_1$ and where $q'_1 \sqsubseteq_{\mathrm{AP}} q'_2$. Let $p'_1 = A_1(q'_1)$. Similarly as the case for q_1, q_2, we know that there exists $m' \geq 1$ distributions p'_2 enabled in q'_2 such that there exists a weight function for (p'_1, p'_2). Therefore, we split A_2 into m' further adversaries, again each with a different nondeterministic choice of probability distribution. This process is then repeated for all such q'_2, and then for all such adversaries such as A_2, to obtain the second transition of all adversaries in $\mathrm{ADV}(A_1)$. Then the process is repeated ad infinitum to obtain all transitions of all of these adversaries.

Take any $A_2 \in \mathrm{ADV}(A_1)$. Next, we show that the probability measure over paths of A_1 satisfying $\Phi_1 \mathcal{U} \Phi_2$ is greater than or equal to that over the paths of A_2 which satisfy $\Phi_1 \mathcal{U} \Phi_2$. We deal with a number of cases.

Case 1. If q_1 and q_2 both satisfy Φ_2, $\xi_{A_1} = \xi_{A_2} = 1$ and our task is completed.

Case 2. Otherwise, for $i \in \{1, 2\}$:

- let $\Omega^l_{A_i}$ be the set of finite paths such that, for all $\omega \in \Omega^l_{A_i}$, $\omega(l) \models_{\mathcal{A}} \Phi_2$, for all $0 \leq j < l$, $\omega(j) \models_{\mathcal{A}} \Phi_1$, and $\omega(0) = q_i$,
- let $Q_{A_i}(l-1)$ be the set of states reached by paths in $\Omega^l_{A_i}$ at step $l-1$, and
- let $Q^{\Phi'}_{A_i}$ be the set of states reachable by adversary A_i that satisfy the $\forall$PBTL formula Φ'.

Then it follows that, for every state $q'_2 \in Q_{A_2}(l-1)$, there exists a state $q'_1 \in Q_{A_1}(l-1)$ such that (1) $q'_1 \sqsubseteq_{AP} q'_2$, and (2) p'_1 is chosen in q'_1 and p'_2 is chosen in q'_2 such that there exists a weight function w for (p'_1, p'_2). Now take any $q' \in Q^{\Phi_2}_{A_2}$ and $\sigma \in \Sigma$. It follows from Definition 6 and induction that $\sum_{q \in Q} w(q, \sigma, q') = \sum_{q \in Q^{\Phi_2}_{A_1}} w(q, \sigma, q')$. Then, from Definition 6 and this fact, for each $\sigma \in \Sigma$:

$$\sum_{q' \in Q^{\Phi_2}_{A_2}} p'_2(\sigma, q') = \sum_{q' \in Q^{\Phi_2}_{A_2}} \sum_{q \in Q} w(q, \sigma, q') = \sum_{q' \in Q^{\Phi_2}_{A_2}} \sum_{q \in Q^{\Phi_2}_{A_1}} w(q, \sigma, q') \,.$$

Note that it is possible that $w(q, \sigma, q') > 0$ for some $q \in Q^{\Phi_2}_{A_1}, q' \in Q \setminus Q^{\Phi_2}_{A_2}$. Then:

$$\sum_{q' \in Q^{\Phi_2}_{A_2}} \sum_{q \in Q^{\Phi_2}_{A_1}} w(q, \sigma, q') \leq \sum_{q' \in Q} \sum_{q \in Q^{\Phi_2}_{A_1}} w(q, \sigma, q') = \sum_{q \in Q^{\Phi_2}_{A_1}} \sum_{q' \in Q} w(q, \sigma, q') = \sum_{q \in Q^{\Phi_2}_{A_1}} p'_1(\sigma, q)$$

by Definition 6. Hence $\sum_{q' \in Q^{\Phi_2}_{A_2}} p'_2(\sigma, q') \leq \sum_{q \in Q^{\Phi_2}_{A_1}} p'_1(\sigma, q)$. Therefore, it is clear that, for the distributions p_1 and p_2 related via a weight function, the probability that p_1 assigns to next states that satisfy Φ_2 is greater than the probability that p_2 assigns to states that satisfy Φ_2.

Case 2.1. If $l = 1$ then this stage of this proof is completed.

Case 2.2. If $l > 1$, we then take the sets $Q_{A_i}(l-2)$ and $Q^{\Phi_1}_{A_i}$ for $i \in \{1, 2\}$, for which it can be shown that, for distributions of A_1 and A_2 related by a weight function, the probability that the distribution of A_1 assigns to $Q^{\Phi_1}_{A_1}$ is at least the probability that the distribution of A_2 assigns to $Q^{\Phi_1}_{A_2}$. This is done by simply substituting $Q_{A_i}(l-2)$ for $Q_{A_i}(l-1)$ and $Q^{\Phi_1}_{A_i}$ for $Q^{\Phi_2}_{A_i}$ in the process above, for $i \in \{1, 2\}$. Then the whole process is repeated until we have traced the paths in $\Omega^l_{A_1}$ and $\Omega^l_{A_2}$ back to the states q_1 and q_2 respectively.

Next, we can repeat the process in case 2 for paths of a different length l'. Then it follows that $\xi_{A_2} \leq \xi_{A_1}$.

Now let $A_1^{\min}$ be the adversary such that no other adversary A_1 assigns a greater probability to the set $\{\omega \,|\, \omega \in \mathit{Path}^{A_1}_{\mathit{ful}}(q_1) \,\&\, \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}$, and let this probability be $\xi_{A_1^{\min}}$. Then it follows that all adversaries A_2 in the set $\mathrm{ADV}(A_1^{\min})$ assign probability less than or equal to $\xi_{A_1^{\min}}$ to the set $\{\omega \,|\, \omega \in \mathit{Path}^{A_2}_{\mathit{ful}}(q_2) \,\&\, \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}$. □

Corollary 16 *Let $\mathcal{M}$ be a labelled probabilistic structure with an associated set $\mathcal{A}$ of adversaries. If $q_1 \approx_{\mathrm{AP}} q_2$, for $q_1, q_2 \in Q$, and Φ is a $\forall$PBTL formula, then $q_1 \models_{\mathcal{A}} \Phi$ iff $q_2 \models_{\mathcal{A}} \Phi$.*

Theorem 17 *Let $\mathcal{M}$ be a labelled probabilistic structure with an associated set $\mathcal{A}$ of adversaries. If $q_1 \simeq_{\mathrm{AP}} q_2$, for $q_1, q_2 \in Q$, and Φ is a PBTL formula, then $q_1 \models_{\mathcal{A}} \Phi$ iff $q_2 \models_{\mathcal{A}} \Phi$.*

Proof sketch. Bisimulation preserves properties of the full logic PBTL for the following reasons. Let $q_1, q_2 \in Q$ be such that $q_1 \simeq_{\mathrm{AP}} q_2$. Firstly, note that, for each probability distribution p_1 available in q_1, there will exist a distribution p_2 enabled in q_2 such that there exists a weight function for (p_1, p_2), and vice versa. This means that, for each adversary A_1 , there will exists a *single* corresponding adversary A_2 that is generated via $\simeq_{\mathrm{AP}}$ in the manner given in the proof of Theorem 15. Secondly, probability distributions such as p_1 and p_2 assign the same *total* probability to transitions whose target is a state in a particular equivalence class of $\simeq_{\mathrm{AP}}$. Therefore, at each step, the probability of being in an equivalence class in which states satisfy Φ_1, or Φ_2, is the same. It then follows that:

$$\begin{aligned} &\mathit{Prob}^{A_2}(\{\omega \,|\, \omega \in \mathit{Path}^{A_2}_{\mathit{ful}}(q_2) \,\&\, \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}) \\ &\quad = \mathit{Prob}^{A_1}(\{\omega \,|\, \omega \in \mathit{Path}^{A_1}_{\mathit{ful}}(q_1) \,\&\, \omega \models_{\mathcal{A}} \Phi_1 \,\mathcal{U}\, \Phi_2\}) \,. \end{aligned}$$

Thus bisimulation with respect to atomic propositions preserves PBTL formulae. □

3.3 Semantics of probabilistic rectangular automata

The semantics of a given probabilistic rectangular automaton H can be represented in terms of variants on probabilistic structures in the following way. First, we define the *timed* and *time-abstract* probabilistic structures, $\mathcal{S}_H^{\mathcal{T}}$ and $\mathcal{S}_H^{Abs}$ respectively, that correspond to H, and then observe that labelled versions of these structures can be defined simply.

Let R be a rectangular predicate such that $[\![R]\!]$ is a singleton. Then $\mathbf{a}[X := [\![R]\!]]$ denotes the valuation $(\mathbf{a}'_1, ..., \mathbf{a}'_n)$ such that, for all $x_i \in X$, $\mathbf{a}'_i = [\![R_i]\!]$, and for all other $x_i \in \mathcal{X} \setminus X$, $\mathbf{a}'_i = \mathbf{a}_i$.

Definition 18 *The* timed probabilistic structure $\mathcal{S}_H^{\mathcal{T}} = (Q_H, Q_H^0, \Sigma_H^{\mathcal{T}}, Steps_H^{\mathcal{T}})$ *of the probabilistic rectangular automaton* $H = (\mathcal{X}, V, L, \Theta, init, inv, flow, prob, pre)$ *is defined as follows:*

- $Q_H \subseteq V \times \mathbb{R}^n$ *such that* $(v, \mathbf{a}) \in Q_H$ *iff* $\mathbf{a} \in [\![inv(v)]\!]$*;*
- $Q_H^0 \subseteq Q_H$ *such that* $(v, \mathbf{a}) \in Q_H^0$ *iff* $\mathbf{a} \in [\![init(v)]\!]$*;*
- $\Sigma_H^{\mathcal{T}} = \Theta \cup \mathbb{R}_{\geq 0}$*;*
- $Steps_H^{\mathcal{T}} = TimeSteps_H^{\mathcal{T}} \cup SwitchSteps_H$*, where:*
  - *for each* $(v, \mathbf{a}) \in Q_H$ *and* $\delta \in \mathbb{R}_{\geq 0}$*, there exists* $\mathcal{D}(\delta, (v, \mathbf{b})) \in TimeSteps_H^{\mathcal{T}}(v, \mathbf{a})$ *iff either (a)* $\delta = 0$ *and* $\mathbf{a} = \mathbf{b}$*, or (b)* $\delta > 0$ *and* $(\mathbf{b} - \mathbf{a})/\delta \in [\![flow(v)]\!]$*;*
  - *for each* $(v, \mathbf{a}) \in Q_H$ *and event* $\theta \in \Theta$*, there exists* $p \in SwitchSteps_H(q)$ *iff there exists* $p_v \in prob(v)$ *such that*
    1. $\mathbf{a} \in [\![pre(p_v)]\!]$*, and*
    2. *for any* $q' \in Q_H$*, where* $q' = (w, \mathbf{b})$*,*

$$p(\sigma, q') = \sum_{\substack{X \subseteq \mathcal{X},\, post \in Rect(\mathcal{X}) \\ \& \, \mathbf{b} = \mathbf{a}[X := [\![post]\!]]}} p_v(w, post, X, \sigma).$$

Definition 19 *The* time-abstract probabilistic structure $\mathcal{S}_H^{Abs} = (Q_H, Q_H^0, \Sigma_H^{Abs}, Steps_H^{Abs})$ *of the probabilistic rectangular automaton* $H = (\mathcal{X}, V, L, \Theta, init, inv, flow, prob, pre)$ *is defined as follows:*

- Q_H *and* Q_H^0 *are as in Definition 18;*
- $\Sigma_H^{Abs} = \Theta \cup \{\tau\}$*;*
- $Steps_H^{Abs} = TimeSteps_H^{Abs} \cup SwitchSteps_H$*, where:*
  - *for each state* $q \in Q_H$ *there exists* $\mathcal{D}(\tau, q') \in TimeSteps_H^{Abs}(q)$ *iff there exists a non-negative real* $\delta \in \mathbb{R}_{\geq 0}$ *such that* $\mathcal{D}(\delta, q') \in TimeSteps_H^{\mathcal{T}}(q)$*;*
  - *we define* $SwitchSteps_H$ *as in Definition 18.*

Remark. The restriction of the initial condition of probabilistic rectangular automata H to a singleton in one control mode means that the initial states of $\mathcal{S}_H^{\mathcal{T}}$ and $\mathcal{S}_H^{Abs}$ are unique.

Definition 20 *The* labelled probabilistic structure $\mathcal{M}_H = (\mathcal{S}_H, \mathcal{L}_H)$ of the probabilistic rectangular automaton $H = (\mathcal{X}, V, L, \Theta, \mathit{init}, \mathit{inv}, \mathit{flow}, \mathit{prob}, \mathit{pre})$ *comprises of a probabilistic structure* $\mathcal{S}_H = (Q_H, Q_H^0, \Sigma_H, \mathit{Steps}_H)$ *of* H*, and a function* $\mathcal{L}_H : Q \to 2^{\mathrm{AP}}$ *such that, for each* $(v, \mathbf{a}) \in Q_H$*,* $\mathcal{L}_H(v, \mathbf{a}) = L(v)$*.*

Then we define the labelled timed probabilistic structure of H as $\mathcal{M}_H^{\mathcal{T}} = (\mathcal{S}_H^{\mathcal{T}}, \mathcal{L}_H)$, and the labelled time-abstract probabilistic structure of H as $\mathcal{M}_H^{Abs} = (\mathcal{S}_H^{Abs}, \mathcal{L}_H)$.

3.3.1 Time divergence

When reasoning about timed and hybrid automata, it is common to have a means to express the notion of the *divergence of time* of their infinite paths. In particular, it is clear that a path along which time converges represents unrealisable behaviour, and therefore can be disregarded during analysis. In our context, we associate a notion of divergence with adversaries, rather than individual paths. Observe that only a subset of paths associated with an adversary A will be such that time diverges without bound. A is then described as being divergent if the probability measure over this subset is 1.

Definition 21 (Divergent adversary) *Let* $\omega = q_0 \xrightarrow{p_0}_{\sigma_0} q_1 \xrightarrow{p_1}_{\sigma_1} q_2 \xrightarrow{p_2}_{\sigma_2} \cdots$ *be a path of a timed probabilistic structure* $\mathcal{S}_H^{\mathcal{T}}$*. For any* $i \in \mathrm{I\!N}$*, let* $\mathrm{Dur}_\omega(i) = \sum_{j=0}^{i-1}\{\sigma_j \mid \sigma_j \in \mathrm{I\!R}_{\geq 0}\}$*. Then an adversary* A *of* $\mathcal{S}_H^{\mathcal{T}}$ *is* divergent *iff, for all states* $q \in Q_H$*:*

$$\mathit{Prob}^A(\{\omega \mid \omega \in \mathit{Path}_{\mathit{ful}}^A(q) \text{ and } \forall t \in \mathrm{I\!R}_{\geq 0}.\exists j \in \mathrm{I\!N}.\mathrm{Dur}_\omega(j) > t\}) = 1.$$

Let $\mathcal{A}_{\mathit{div}}$ *be the set of divergent adversaries.*

It then follows that, instead of defining the satisfaction relation of PBTL and ∀PBTL in terms of an arbitrary set of adversaries, it is possible to define it with respect to the set of divergent adversaries $\mathcal{A}_{\mathit{div}}$. Furthermore, given that divergent adversaries have been defined on the timed probabilistic structure $\mathcal{S}_H^{\mathcal{T}}$ of H, it is straightforward to define divergent adversaries on the associated time-abstract probabilistic structure $\mathcal{S}_H^{Abs}$, using the correspondence between timed and time-abstract paths of hybrid automata presented in [Hen96].

3.3.2 State equivalence for timed and time-abstract structures

Observe that it follows that the definitions of simulation and bisimulation are applicable to both timed and time-abstract probabilistic structures in the standard manner. Note that the equivalence relations for timed probabilistic structures correspond to timed simulation [LV96] and timed bisimulation [Cer92], and the relations for the time-abstract case correspond to time-abstract simulation and bisimulation (defined in the context of hybrid automata in [Hen96]). In the sequel, our focus will be exclusively on the labelled versions of the structures, $\mathcal{M}_H^{\mathcal{T}}$ and $\mathcal{M}_H^{Abs}$. The following notation is introduced: $\simeq^{\mathcal{T}}$ denotes a bisimulation relation on $\mathcal{M}_H^{\mathcal{T}}$, whereas $\simeq_{\mathrm{AP}}^{\mathcal{T}}$ denotes the relation of bisimulation with respect to AP of $\mathcal{M}_H^{\mathcal{T}}$.

4 Model checking subclasses of probabilistic rectangular automata

This section focuses on model checking strategies for subclasses of probabilistic rectangular automata. In particular, we show that both probabilistic multisingular and probabilistic stopwatch automata are equivalent to probabilistic timed automata, where the notion of equivalence preserves PBTL properties. Therefore, the verification result of [KNSS99], which presents an algorithm for model checking probabilistic timed automata against a superset of PBTL properties, can be easily applied to verify probabilistic multisingular and stopwatch automata against such properties. However, the type of equivalence that exists between probabilistic rectangular automata and probabilistic timed automata is weaker, and does not preserve the full set of PBTL properties. Our solution is to focus on simulation relations of 2D probabilistic rectangular automata, which preserve $\forall$PBTL properties, and explore the adjustments necessary to the algorithm of [KNSS99] to facilitate verification of such models, an instance of which is H_1 in section 2.

4.1 Model checking probabilistic timed automata

Firstly, we remind ourselves of the results of [KNSS99], and highlight several of their characteristics that are pertinent in this context. In brief, [KNSS99] utilizes the so-called 'region construction' for non-probabilistic timed automata, which involves the use of an equivalence to collapse an infinite number of states into a finite number of equivalence classes [ACD93, AD94]. Each class becomes a state of a finite probabilistic structure referred to as the 'region graph', the transitions of which are defined by the possible movement between equivalence classes in the original probabilistic timed automaton. This probabilistic structure can then be model checked against formulae of a quantitative, probabilistic temporal logic using the established methods of [BdA95, BK98]. Observe that model checking in this context is polynomial in the size of the region graph, which in turn is exponential in the number of variables and the magnitude of their upper bounds, and linear in the size of the PBTL formula.

We now introduce *region equivalence* formally. Let T be the probabilistic timed automaton $(\mathcal{X}, V, L, \Theta, \mathit{init}, \mathit{inv}, \mathit{flow}, \mathit{prob}, \mathit{pre})$, such that all constants in rectangular inequalities used in the description of T are integers. For each variable $x_i \in \mathcal{X}$, let $\max_i \in \mathbb{N}$ be the largest constant in an inequality referring to x_i that appears in an invariant, pre- or post-condition of T.

Definition 22 (Region equivalence) *For any $t \in \mathbb{R}$, let $\lfloor t \rfloor$ denote its integral part and $\mathit{frac}(t)$ its fractional part. For a vector $\mathbf{a}$, let $\lfloor \mathbf{a} \rfloor$ denote the vector whose ith coordinate is $\lfloor \mathbf{a}_i \rfloor$, and $\mathit{frac}(\mathbf{a})$ the vector whose ith coordinate is $\mathit{frac}(\mathbf{a}_i)$. Let $\mathbf{b}$ also be a vector. Define $\mathbf{a} \equiv \mathbf{b}$ iff, for each $x_i \in \mathcal{X}$, (1) $\mathit{frac}(\mathbf{a}_i) = 0$ iff $\mathit{frac}(\mathbf{b}_i) = 0$, and (2) for each $x_j \in \mathcal{X}$, $x_j \neq x_i$, $\mathit{frac}(\mathbf{a}_i) < \mathit{frac}(\mathbf{a}_j)$ iff $\mathit{frac}(\mathbf{b}_i) < \mathit{frac}(\mathbf{b}_j)$. Then two states $(v, \mathbf{a}), (v', \mathbf{b})$ are* region equivalent*, denoted by $(v, \mathbf{a}) \equiv^R (v', \mathbf{b})$ if (1) $v = v'$, (2) for all $x_i \in \mathcal{X}$, either $\lfloor \mathbf{a}_i \rfloor = \lfloor \mathbf{b}_i \rfloor$ or both $\lfloor \mathbf{a}_i \rfloor$ and $\lfloor \mathbf{b}_i \rfloor$ are greater than* $\max_i$*, and (3) $\mathit{frac}(\mathbf{a}) \equiv \mathit{frac}(\mathbf{b})$.*

A fundamental observation about region equivalence in the non-probabilistic setting is that it is a finite bisimulation of the time-abstract transition system of a timed automaton [AD94], which inspires the following lemma.

Lemma 23 *Let T be a probabilistic timed automata and* AP *be a set of atomic propositions. Region equivalence $\equiv^R$ is a finite time-abstract bisimulation of $\mathcal{M}_T^{Abs}$ with respect to* AP.

The proof of this lemma follows from the fact that all states within a given region equivalence class must satisfy the same atomic formulae on variable values, and therefore enable the same probability distributions for choice.

The model for probabilistic timed automata presented here differs from that of [KNSS99] in two respects. Firstly, the previous paper only permitted constraints which compared variables to *integers*; however, as explained in remark 2 of section 2, a probabilistic hybrid automaton with rational constants used in variable constraints can easily be transformed into an equivalent model including integer comparisons simply by scaling up the values of all such constants. Secondly, [KNSS99] considered probabilistic timed automata for which variables could be reset to 0 only. Our approach of permitting assignments to other rationals (given the assumption of deterministic jumps) does not add any conceptual complexity to the model checking algorithm of [KNSS99]. All that is necessary is to consider post-conditions (in addition to other conditions) when determining the maximal constant used in the probabilistic timed automaton description and to adjust the transition relation of the region graph in the obvious way to take account of such resets.

Furthermore, we adopt a more restricted logic than that of [KNSS99]. The previous paper presented a model checking algorithm for probabilistic timed automata against formulae of the timed, probabilistic logic PTCTL (Probabilistic Timed Computation Tree Logic). PTCTL is more expressive than PBTL in two respects: firstly, it admits subformulae that refer directly to values of the variables of the system model in question (in the case of timed automata, these variables are exclusively clocks); secondly, and more significantly, PTCTL utilizes the 'reset quantifier' mechanism [HNSY94] in order to reason about time quantitatively in its formulae. Neither aspect poses conceptual difficulties when focusing on probabilistic rectangular automata rather than probabilistic timed automata, and have been omitted from the logic used in this paper for reasons of space only.

4.2 Translation from probabilistic stopwatch automata to probabilistic timed automata

As noted in [HKPV98], a non-probabilistic stopwatch automaton can be transformed into a timed bisimilar non-probabilistic timed automaton. The intuition underlying this result is that the values of stopped variables (those with $\dot{x} = 0$) are encoded into the control modes of the timed automaton. Because the assumptions of initialization and deterministic jumps mean that there are only a finite number of rational values that stopped variables can take, such an encoding always results in a finite number of control modes. We now extend this result to the probabilistic case.

From a probabilistic stopwatch automaton $S = (\mathcal{X}, V_S, L_S, \Theta, \mathit{init}_S, \mathit{inv}_S, \mathit{flow}_S, \mathit{prob}_S, \mathit{pre}_S)$, we construct the probabilistic timed automaton $T_S = (\mathcal{X}, V_{T_S}, L_{T_S}, \Theta, \mathit{init}_{T_S}, \mathit{inv}_{T_S}, \mathit{flow}_{T_S}, \mathit{prob}_{T_S}, \mathit{pre}_{T_S})$ in the following way.

Control modes. Let $\mathcal{K}_S$ be the set of integer constants used in the definition of S, and let $\mathcal{K}_- = \mathcal{K}_S \cup \{-\}$. Then $V_{T_S} = V_S \times \mathcal{K}_-^{\mathcal{X}}$; therefore, each control mode of T_S consists of a control mode of S and a function $f : \mathcal{X} \to \mathcal{K}_-$. For a given control mode of S, say $v \in V_S$, a function f of $(v, f) \in V_{T_S}$ will have the following form: for each $x_i \in \mathcal{X}$, if

$\mathit{flow}(v)_i$ is $\dot{x}_i = 1$, then $f(x_i) = -$; otherwise, $f(x_i) \in \mathcal{K}_S$. [2]

Labelling conditions. For each $(v, f) \in V_{T_S}$, $L_{T_S}(v, f) = L_S(v)$.

Initial, invariant and flow conditions. For all $(v, f) \in V_{T_S}$, if, for each $x_i \in \mathcal{X}$, either $f(x_i) = -$ or $f(x_i) = [\![\mathit{init}_S(v)_i]\!]$, then $\mathit{init}_{T_S}(v, f) = \mathit{init}_S(v)$, otherwise $\mathit{init}_{T_S}(v, f) = \mathtt{false}$. Next, for all $(v, f) \in V_{T_S}$, $\mathit{inv}_{T_S}(v, f) = \mathit{inv}_S(v)$. Finally, by the definition of probabilistic timed automata, for all $(v, f) \in V_{T_S}$ and all $x_i \in \mathcal{X}$, $\mathit{flow}_{T_S}(v)_i$ is $\dot{x}_i = 1$.

Probability distributions. Recall that, for each control mode $v \in V_S$, $\mathit{prob}_S(v) = \{p_v^1, ..., p_v^l\}$ for some finite $l \in \mathbb{N}$. Then, for each control mode of the form $(v, f) \in V_{T_S}$, let $\mathit{prob}_{T_S}(v, f) = \{p_{(v,f)}^1, ..., p_{(v,f)}^l\}$, where, for each $1 \le j \le l$, $p_{(v,f)}^j$ is derived from p_v^j in the following manner. For each tuple $(w, \mathit{post}, X, \theta) \in V_S \times \mathit{Rect}(\mathcal{X}) \times 2^{\mathcal{X}} \times \Theta$, there exists $(w, g) \in V_{T_S}$ such that $p_{(v,f)}^j((w, g), \mathit{post}, X, \theta) = p_v^j(w, \mathit{post}, X, \theta)$, and, for every $x_i \in \mathcal{X}$, either:

1. $\mathit{flow}_S(w)_i$ is $\dot{x}_i = 1$ and $g(x_i) = -$,
2. $\mathit{flow}_S(v)_i$ is $\dot{x}_i = 1$, $\mathit{flow}_S(w)_i$ is $\dot{x}_i = 0$ and $g(x_i) = [\![\mathit{post}_i]\!]$,
3. $\mathit{flow}_S(v)_i$ is $\dot{x}_i = 0$, $\mathit{flow}_S(w)_i$ is $\dot{x}_i = 0$, $x_i \notin X$ and $g(x_i) = f(x_i)$,
4. $\mathit{flow}_S(v)_i$ is $\dot{x}_i = 0$, $\mathit{flow}_S(w)_i$ is $\dot{x}_i = 0$, $x_i \in X$ and $g(x_i) = [\![\mathit{post}_i]\!]$.

Let $p_{(v,f)}^j((w', g'), \mathit{post}, X, \theta) = 0$ for all other $(w', g') \in V_{T_S}$.

Pre-conditions. For all $(v, f) \in V_{T_S}$, $1 \le j \le l$, and $x_i \in \mathcal{X}$, if $f(x_i) = -$, then $\mathit{pre}_{T_S}(p_{(v,f)}^j)_i = \mathit{pre}_S(p_v^j)_i$, otherwise we obtain $\mathit{pre}_{T_S}(p_{(v,f)}^j)_i$ by substituting the value of $f(x_i)$ for x_i in $\mathit{pre}_S(p_v^j)_i$, which then resolves to a boolean value.

4.2.1 Equivalence of S and T_S

Observe that the timed probabilistic structures, $\mathcal{S}_S^{\mathcal{T}}$ and $\mathcal{S}_{T_S}^{\mathcal{T}}$, and the labelled timed probabilistic structures, $\mathcal{M}_S^{\mathcal{T}} = (\mathcal{S}_S^{\mathcal{T}}, \mathcal{L}_S)$ and $\mathcal{M}_{T_S}^{\mathcal{T}} = (\mathcal{S}_{T_S}^{\mathcal{T}}, \mathcal{L}_{T_S})$, of S and T_S can be obtained by straightforward use of Definition 18 and Definition 20. We now present the function α of [HKPV98] in our context, and prove that its associated equivalence relation, $\hat{\alpha}$, is a *probabilistic* bisimulation in the style of Definition 7. More precisely, $\alpha : Q_{T_S} \to Q_S$ is a surjective function relating states of the underlying model $\mathcal{S}_{T_S}^{\mathcal{T}}$ of the probabilistic timed automaton T_S, to those of the underlying model $\mathcal{S}_S^{\mathcal{T}}$ of the associated probabilistic stopwatch automaton S, and $\hat{\alpha}$ is the equivalence relation on states induced by α.

Definition 24 *$\alpha : Q_{T_S} \to Q_S$ is the function such that, for each $q = ((v, f), \mathbf{a}) \in Q_{T_S}$, $\alpha(q) = (v, \mathbf{b})$, where, for each $x_i \in \mathcal{X}$, $\mathbf{a}_i = \mathbf{b}_i$ if $f(x_i) = -$, and $\mathbf{b}_i = f(x_i)$ otherwise.*

$\hat{\alpha}$ is an equivalence relation on $Q_{T_S} \cup Q_S$ such that, for $q_{T_S} \in Q_{T_S}$ and $q_S \in Q_S$, $(q_{T_S}, q_S) \in \hat{\alpha}$ iff $\alpha(q_{T_S}) = q_S$.

Proposition 25 *$\hat{\alpha}$ is a bisimulation between $\mathcal{S}_S^{\mathcal{T}}$ and $\mathcal{S}_{T_S}^{\mathcal{T}}$.*

[2] Equivalently, the control modes of T_S can be obtained by equipping S with extra variables which take values over a *finite* set of possibilities. More precisely, there will be $|\mathcal{X}|$ such extra variables, which take values from the set $\mathcal{K}_-$, and which remain constant while control remains in a particular mode. The facility for reasoning about such variables exists in the real-time model checking tools UPPAAL [LPY97], and OPEN-KRONOS (an adaptation of KRONOS) [Tri98].

Proof. The proposition can be stated alternatively as: if $((v,f),\mathbf{a}) \in Q_{T_S}$ and $(v,\mathbf{b}) \in Q_S$ are such that $(((v,f),\mathbf{a}),(v,\mathbf{b})) \in \hat{\alpha}$, then $((v,f),\mathbf{a}) \simeq^{\mathcal{T}} (v,\mathbf{b})$. Firstly, it will be shown that, for the set of probability distributions that can be nondeterministically chosen in $(v,\mathbf{b})$, the set of distributions available for choice in $((v,f),\mathbf{a})$ are exactly those that are derived from the former set. That is, if $prob_S(v) = \{p_v^1,...,p_v^l\}$ and $prob_{T_S}(v,f) = \{p^1_{(v,f)},...,p^l_{(v,f)}\}$, then, for all $1 \leq j \leq l$, if $\mathbf{b}$ satisfies $pre_S(p_v^j)$, then $\mathbf{a}$ satisfies $pre_{T_S}(p^j_{(v,f)})$. This fact follows immediately from the definitions of α and the pre-conditions of the distributions of T_S.

Secondly, take a particular j, where $1 \leq j \leq l$. We now show that p_v^j and $p^j_{(v,f)}$ assign the same total probability to equivalence classes of $\hat{\alpha}$. Take a tuple $(w, post, X, \theta) \in V_S \times Rect(\mathcal{X}) \times 2^{\mathcal{X}} \times \Theta$, with an associated tuple $((w,g), post, X, \theta) \in V_{T_S} \times Rect(\mathcal{X}) \times 2^{\mathcal{X}} \times \Theta$ such that $p^j_{(v,f)}((w,g),post,X,\theta) = p_v^j(w,post,X,\theta)$ (by the definition of the translation from S to T_S, such a tuple will exist). Then the state $(w,\mathbf{b}')$, which is reached from $(v,\mathbf{b})$ via nondeterministic choice of p_v^j and then probabilistic choice of $(w,post,X,\theta)$, and the state $((w,g),\mathbf{a}')$, reached from $((v,f),\mathbf{a})$ via nondeterministic choice of $p^j_{(v,f)}$ and then probabilistic choice of $((w,g),post,X,\theta)$, are such that $(((w,g),\mathbf{a}'),(w,\mathbf{b}')) \in \hat{\alpha}$. To see this, note that it follows from the definition of α that, if $(((v,f),\mathbf{a}),(v,\mathbf{b})) \in \hat{\alpha}$, then for each variable $x_i \in \mathcal{X}$ such that $f(x_i) \neq -$, $\mathbf{a}_i = \mathbf{b}_i$. Observe that the control switches described above will mean that each variable $x_i \in \mathcal{X}$ is either reset to $[\![post_i]\!]$, or retains the same value in $\mathbf{a}$ and $\mathbf{b}$. Thus it immediately follows that, for each variable $x_i \in \mathcal{X}$ such that $g(x_i) \neq -$, $\mathbf{a}'_i = \mathbf{b}'_i$, and for all other variables $x_i \in \mathcal{X}$ such that $g(x_i) \in \mathcal{K}_S$, $g(x_i) = \mathbf{b}_i$ by the construction of T_S. Therefore, $(((w,g),\mathbf{a}'),(w,\mathbf{b}')) \in \hat{\alpha}$, Next, observe that it must be the case that $\mathbf{a}' = \mathbf{a}[X := [\![post]\!]]$ and $\mathbf{b}' = \mathbf{b}[X := [\![post]\!]]$. Then, by Definition 18, it follows that:

$$\sum_{\substack{X \subseteq \mathcal{X},\, post \in Rect(\mathcal{X}) \\ \&\, \mathbf{b}' = \mathbf{b}[X := [\![post]\!]]}} p_v^j(w,post,X,\sigma) = \sum_{\substack{X \subseteq \mathcal{X},\, post \in Rect(\mathcal{X}) \\ \&\, \mathbf{a}' = \mathbf{a}[X := [\![post]\!]]}} p^j_{(v,f)}((w,g),post,X,\sigma).$$

Hence, there exists $p_v \in Steps_S(v,\mathbf{b})$ and $p_{(v,f)} \in Steps_{T_S}((v,f),\mathbf{a})$ such that $p_v(\theta,(w,\mathbf{b}'))$ $= p_{(v,f)}(\theta,((w,g),\mathbf{a}'))$. That is, the same probability is assigned to states that are equivalent according to $\hat{\alpha}$. We can repeat this process for all $1 \leq j \leq l$. Therefore, if follows from Definition 7 that $((v,f),\mathbf{a}) \simeq (v,\mathbf{b})$ □

Corollary 26 *$\hat{\alpha}$ is a bisimulation with respect to* AP *between $\mathcal{M}_S^{\mathcal{T}}$ and $\mathcal{M}_{T_S}^{\mathcal{T}}$.*

Proof. Observe that the corollary can be stated alternatively as: if $((v,f),\mathbf{a}) \in Q_{T_S}$ and $(v,\mathbf{b}) \in Q_S$ are such that $(((v,f),\mathbf{a}),(v,\mathbf{b})) \in \hat{\alpha}$, then $((v,f),\mathbf{a}) \simeq^{\mathcal{T}}_{\mathrm{AP}} (v,\mathbf{b})$. By the definition of the translation from S to T_S, $L_{T_S}(v,f) = L_S(v)$, and, from this fact and Definition 20, $\mathcal{L}_S(v,f) = \mathcal{L}_{T_S}(v)$. Therefore, $\hat{\alpha}$ is a bisimulation with respect to AP. □

Verification of S against the PBTL property Φ is then performed by transforming S to T_S, and then model checking T_S against Φ using the algorithm of [KNSS99]. Then, if T_S satisfies the property Φ, by Theorem 17 we conclude that S satisfies Φ.

4.3 Translation from probabilistic multisingular automata to probabilistic stopwatch automata

This section presents a method for the transformation of probabilistic multisingular automata into equivalent probabilistic stopwatch automata, again with the notion of equivalence being a type of bisimulation that preserves PBTL properties. The transformation relies on appropriate scaling of the constants used in the constraints of the model, so that, for each variable $x_i \in \mathcal{X}$, the associated flow condition is either $\dot{x}_i = 1$ or $\dot{x}_i = 0$, and is based on the results for non-probabilistic multisingular automata presented in [OSY94, ACH$^+$95, HKPV98]. Take a particular probabilistic multisingular automaton $M = (\mathcal{X}, V, L, \Theta, \mathit{init}_M, \mathit{inv}_M, \mathit{flow}_M, \mathit{prob}_M, \mathit{pre}_M)$.

Definition 27 (Scaling factor.) *For all $v \in V$ and $x_i \in \mathcal{X}$, if $\mathit{flow}(v)_i$ takes the form $\dot{x}_i = k$ for some $k \in \mathbb{Q}$, then we define the* scaling factor of x_i in v *by k_i^v, where $k_i^v = k$ if $k \neq 0$, and $k_i^v = 1$ if $k = 0$.*

From M we construct the probabilistic stopwatch automaton $S_M = (\mathcal{X}, V, L, \Theta, \mathit{init}_{S_M}, \mathit{inv}_{S_M}, \mathit{flow}_{S_M}, \mathit{prob}_{S_M}, \mathit{pre}_{S_M})$ in the following way.

Initial, invariant and flow conditions. For all $v \in V$ and $x_i \in \mathcal{X}$, $\mathit{init}_{S_M}(v)_i$ is obtained from $\mathit{init}_M(v)_i$ by replacing all constants $k \in \mathbb{Q}$ appearing in $\mathit{init}_M(v)_i$ by $\frac{k}{k_i^v}$. Similarly, $\mathit{inv}_{S_M}(v)_i$ is obtained from $\mathit{inv}_M(v)_i$ by replacing all $k \in \mathbb{Q}$ appearing in $\mathit{inv}_M(v)_i$ by $\frac{k}{k_i^v}$. Furthermore, for all $v \in V$ and $x_i \in \mathcal{X}$, if $\mathit{flow}_M(v)_i$ is $\dot{x}_i = 0$, then $\mathit{flow}_{S_M}(v)_i$ is $\dot{x}_i = 0$, otherwise $\dot{x}_i = 1$.

Probability distributions. The distributions used in the description of S_M are similar to those used in M, except that the post-conditions of discrete transitions must be re-scaled. Recall that, for each $v \in V$, $\mathit{prob}_M(v) = \{p_M^1, ..., p_M^l\}$ for some finite $l \in \mathbb{N}$. Then we let $\mathit{prob}_{S_M}(v) = \{p_{S_M}^1, ..., p_{S_M}^l\}$, where, for all $1 \leq j \leq l$, $p_{S_M}^j$ is derived from p_M^j in the following manner. Take the tuple $(w, \mathit{post}_M, X, \theta) \in V_S \times \mathit{Rect}(\mathcal{X}) \times 2^{\mathcal{X}} \times \Theta$, which is such that $p_M^j(w, \mathit{post}_M, X, \theta) > 0$. Then we derive the post-condition post_{S_M} from post_M by replacing all constants $k \in \mathbb{Q}$ in $(\mathit{post}_M)_i$ by $\frac{k}{k_i^w}$, for each $x_i \in \mathcal{X}$. Next, we let $p_{S_M}^j(w, \mathit{post}_{S_M}, X, \theta) = p_M^j(w, \mathit{post}_M, X, \theta)$. This procedure is repeated for all tuples $(w, \mathit{post}_{S_M}, X, \theta)$ derived from those $(w, \mathit{post}_M, X, \theta)$ such that $p_M^j(w, \mathit{post}_M, X, \theta) > 0$, and we let $p_{S_M}^j$ assign 0 to all other tuples.

Pre-conditions. For all $v \in V$, where $\mathit{prob}_M(v) = \{p_M^1, ..., p_M^l\}$, and for all $1 \leq j \leq l$ and $x_i \in \mathcal{X}$, we obtain $\mathit{pre}_{S_M}(p_{S_M}^j)_i$ from $\mathit{pre}_M(p_M^j)_i$ by replacing all constants $k \in \mathbb{Q}$ in $\mathit{pre}_M(p_M^j)_i$ by $\frac{k}{k_i^v}$.

4.3.1 Equivalence of M and S_M

Again, it is clear that the timed probabilistic structures, $\mathcal{S}_M^{\mathcal{T}}$ and $\mathcal{S}_{S_M}^{\mathcal{T}}$, and the labelled timed probabilistic structures, $\mathcal{M}_M^{\mathcal{T}}$ and $\mathcal{M}_{S_M}^{\mathcal{T}}$, of M and S_M can be obtained using Definition 18 and Definition 20. Furthermore, the function β of [HKPV98], which relates states of S_M and M, induces the equivalence relation $\hat{\beta}$ that is a bisimulation on $\mathcal{S}_M^{\mathcal{T}}$ and $\mathcal{S}_{S_M}^{\mathcal{T}}$, in the same way that α induced the bisimulation $\hat{\alpha}$ in section 4.2.1.

Definition 28 *$\beta : Q_{S_M} \rightarrow Q_M$ is the bijection such that, for each $q = (v, \mathbf{a}) \in Q_{S_M}$, $\beta(q) = (v, \mathbf{b})$, where, for each $x_i \in \mathcal{X}$, $\mathbf{b}_i = k_i^v.\mathbf{a}_i$.*

$\hat{\beta}$ is an equivalence relation on $Q_{S_M} \cup Q_M$ such that, for $q_{S_M} \in Q_{S_M}$ and $q_M \in Q_M$, $(q_{S_M}, q_M) \in \hat{\beta}$ iff $\beta(q_{S_M}) = q_M$.

Proposition 29 *$\hat{\beta}$ is a bisimulation between $\mathcal{S}_M^{\mathcal{T}}$ and $\mathcal{S}_{S_M}^{\mathcal{T}}$.*

Proof. The proposition can be stated alternatively as: if $q \in Q_{S_M}$ and $q' \in Q_M$ such that $(q, q') \in \hat{\beta}$, then $q \simeq^{\mathcal{T}} q'$. It follows immediately from the definition of the transformation from M to S_M that, for all states $q, r \in Q_{S_M}$ and every $\sigma \in \Sigma$, there exists $p \in \mathit{Steps}_{S_M}(q)$ such that $p(\sigma, r) > 0$ iff there exists $p' \in \mathit{Steps}_M(\beta(q))$ such that $p'(\sigma, \beta(r)) > 0$. Furthermore, $p(\sigma, r) = p'(\sigma, \beta(r))$. The proposition follows from this fact and Definition 7. □

Corollary 30 *$\hat{\beta}$ is a bisimulation with respect to* AP *between $\mathcal{M}_M^{\mathcal{T}}$ and $\mathcal{M}_{S_M}^{\mathcal{T}}$.*

The proof of Corollary 30 is similar to that of Corollary 26. The model checking process for a probabilistic multisingular automaton M then comprises of transforming M to S_M, scaling up the rational constants used in the description of S_M to integers in order to obtain S'_M, and then transforming S'_M to the probabilistic timed automaton $T_{S'_M}$.

4.4 Model checking 2D probabilistic rectangular automata

Recall that the theory of state equivalence for non-probabilistic rectangular automata shows that the existence of a finitary time-abstract bisimilarity quotient does not generalize from multisingular to rectangular automata [Hen95]. It follows that we cannot use the region equivalence technique for the verification of probabilistic rectangular automata as it has currently been defined, and therefore direct utilization of the results of [KNSS99] is impossible. However, finite quotients of rectangular automata do exist if the notion of equivalence is weakened. In particular, for the class of 2D rectangular automata for which variables do not decrease continuously, the 'double-region' equivalence has been shown to have a finite mutual time-abstract simulation quotient [HHK95], albeit one which is exponential in the largest constant used in the description of the model. Therefore, we define double-region equivalence in our context, and observe the fact that this equivalence preserves ∀PBTL properties.

Double-region equivalence is characterized by taking the intersection of two *scaled vector* equivalences. [3]

Definition 31 (Scaled vector equivalence [HHK95]) *Let $\eta = (\eta_1, \eta_2)$ be a tuple of integers, and $\mathbf{a}, \mathbf{b}$ be the vectors $(\mathbf{a}_1, \mathbf{a}_2), (\mathbf{b}_1, \mathbf{b}_2)$ respectively. Define $\mathbf{a} \equiv_\eta \mathbf{b}$ iff, for $i \in \{1, 2\}$, (1) $\lfloor \eta_i \mathbf{a}_i \rfloor = \lfloor \eta_i \mathbf{b}_i \rfloor$, (2) $\mathit{frac}(\eta_i \mathbf{a}_i) = 0$ iff $\mathit{frac}(\eta_i \mathbf{b}_i) = 0$, and (3) for $j \in \{1, 2\}$, $j \neq i$, $\mathit{frac}(\eta_i \mathbf{a}_i) < \mathit{frac}(\eta_j \mathbf{a}_j)$ iff $\mathit{frac}(\eta_i \mathbf{b}_i) < \mathit{frac}(\eta_j \mathbf{b}_j)$.*

The intersection of two such scaled vector equivalences $\equiv_\eta$ and $\equiv_\zeta$ is denoted by $\equiv_{\eta,\zeta}$. We now introduce formally our restricted model of probabilistic rectangular automata.

Definition 32 *A* positive 2D probabilistic rectangular automaton *is a probabilistic rectangular automaton $H^{2D} = (\{x_1, x_2\}, V, L, \Theta, \mathit{init}, \mathit{inv}, \mathit{flow}, \mathit{prob}, \mathit{pre})$ such that, for each control mode $v \in V$, $\mathit{flow}(v)$ is of the form $(l_1 \leq \dot{x}_1 \leq u_1) \wedge (l_2 \leq \dot{x}_2 \leq u_2)$, and $l_1, l_2 \geq 0$.*

[3] Note that scaled vector equivalences, when used singly, can be used to define directly time-abstract bisimilarity quotients of multisingular automata [ACH⁺95]. We have concentrated on *timed* bisimulation for such hybrid automata because it is the stronger equivalence.

For each $x_i \in \mathcal{X}$, let $\max_i$ be the largest rational constant that x_i is compared to in the definition of the positive 2D probabilistic rectangular automaton H^{2D}.

Definition 33 (Double-region equivalence) *Two states* $(v, \mathbf{a}), (v', \mathbf{b})$ *are* double-region equivalent *with respect to* Φ, *denoted by* $(v, \mathbf{a}) \equiv^{2D} (v', \mathbf{b})$ *if (1)* $v = v'$, *(2) for all* $i \in \{1, 2\}$, *either* $\lfloor \mathbf{a}_i \rfloor = \lfloor \mathbf{b}_i \rfloor$ *or both* $\lfloor \mathbf{a}_i \rfloor$ *and* $\lfloor \mathbf{b}_i \rfloor$ *are greater than* $\max_i$, *and (3)* $frac(\mathbf{a}) \equiv_{\eta_1,\eta_2} frac(\mathbf{b})$, *where* $flow(v)$ *is* $(l_1 \leq \dot{x}_1 \leq u_1) \wedge (l_2 \leq \dot{x}_2 \leq u_2)$ *and* $\eta_1 = (l_2, u_1), \eta_2 = (u_2, l_1)$

The result of [HHK95], which shows that, in the non-probabilistic context, double-region equivalence is a simulation with respect to a set of atomic formulae on the time-abstract transition system of a 2D rectangular automata, inspires the following lemma.

Lemma 34 *Let* H^{2D} *be a positive 2D probabilistic rectangular automaton, and* AP *be a set of atomic propositions. The double-region equivalence* $\equiv^{2D}$ *is a finite time-abstract simulation of* $\mathcal{M}^{Abs}_{H^{2D}}$ *with respect to* AP.

Using Theorem 15, and the observation of [Hen96] that, in the non-probabilistic context, model checking against universal or existential fragments of TCTL is decidable for 2D rectangular automata for which variables do not decrease continuously, we conclude the following.

Corollary 35 *The* $\forall$*PBTL model checking problem for the positive 2D probabilistic rectangular automaton* H^{2D} *is decidable.*

5 Conclusions

Verification methods for hybrid systems are well known to be expensive, and the model checking strategies presented in this paper are no exception. Recall that, for timed automata and 2D rectangular automata, the size of the region or double-region quotient is exponential in the number of variables used and the magnitude of their upper bounds. Furthermore, the probabilistic verification algorithm is polynomial in the size of this quotient and linear in the size of the PBTL formula. Therefore, further work could address the inefficiencies of this method, exploiting symbolic methods [HNSY94], on-the-fly verification, or coarser time-abstract bisimulation and simulation relations [Tri98].

The results of this paper could be expanded to cater for more general classes of model in two ways. Firstly it is intended that the two restrictions involved in the study of the verification problem for 2D probabilistic rectangular automata, namely the restriction to non-negative flows and to only two continuous variables, can be lifted to obtain a model checking result against properties of a quantitative, probabilistic variant of the temporal logic LTL [Bai98]. In particular, two results for the non-probabilistic context imply that obtaining such results is possible. The use of both positive and negative flows in 2D rectangular automata induces an *infinite* time-abstract simulation quotient, albeit one with sufficient structure to permit decidable model checking for LTL properties via the appropriate definition of the verification problem as one of language inclusion for pushdown automata and finite automata [HHK95]. Furthermore, [HKPV98] shows how to transform an nD rectangular automaton into a $2n$D multisingular automaton such that both accept the same language, and therefore satisfy the same LTL formulae.

Secondly, the semi-decidable algorithm for model checking *linear hybrid automata* [AHH96], which have more general continuous dynamics than rectangular automata, could be adapted to cater for this class of model augmented with discrete probability distributions.

References

[ACD93] R. Alur, C. Courcoubetis, and D.L. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1):2–34, 1993.

[ACH$^+$95] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AHH96] R. Alur, T.A. Henzinger, and P.-H. Ho. Automatic symbolic verification of embedded systems. *IEEE Transactions on Software Engineering*, 22(3):181–201, 1996.

[Bai98] C. Baier. On algorithmic verification methods for probabilistic systems, 1998. Habilitation thesis. University of Mannheim.

[BdA95] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. *Lecture Notes in Computer Science*, 1026:499–513, 1995.

[BK98] C. Baier and M.Z. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11:125–155, 1998.

[Cer92] K. Čerāns. Decidability of bisimulation equivalence for parallel timer processes. In *Proc. 4th Workshop on Computer-Aided Verification*, Montreal, Canada, 1992.

[Cor94] J. C. Corbett. Modeling and analysis of real-time ADA tasking programs. In Krithivasan Ramamritham, editor, *Proceedings of the Real-Time Systems Symposium*. IEEE Computer Society Press, 1994.

[Hen95] T.A. Henzinger. Hybrid automata with finite bisimulations. In Z. Fülöp and F. Gécseg, editors, *ICALP 95: Automata, Languages, and Programming*, Lecture Notes in Computer Science 944, pages 324–335. Springer-Verlag, 1995.

[Hen96] T.A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pages 278–292. IEEE Computer Society Press, 1996.

[HHK95] M.R. Henzinger, T.A. Henzinger, and P.W. Kopke. Computing simulations on finite and infinite graphs. In *Proceedings of the 36rd Annual Symposium on Foundations of Computer Science*, pages 453–462. IEEE Computer Society Press, 1995.

[HHMWT99] T.A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi. Beyond HyTech: Hybrid systems analysis using interval numerical methods. In Gautam Biswas and Sheila McIlraith, editors, *Proceedings of the AAAI 1999 Spring Symposium on Hybrid Systems and AI*, 1999.

[HHWT98] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE Transactions on Automatic Control*, 43:540–554, 1998.

[HKPV98] T.A. Henzinger, P.W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57(1):94–124, August 1998.

[HNSY94] T.A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111(2):193–244, 1994.

[JL91] B. Jonsson and K.G. Larsen. Specification and refinement of probabitistic processes. In A.R. Meyer, editor, *Proceedings of the 6th Annual IEEE Symposium on Logic in Computer Science*, pages 266–279, Amsterdam, The Netherlands, July 1991. IEEE Computer Society Press.

[KNSS99] M.Z. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. In J.-P. Katoen, editor, *ARTS'99: Proceedings of the 5th International AMAST Workshop on Real-Time and Probabilistic Systems*, volume 1601 of *Lecture Notes in Computer Science*, pages 75–95. Springer-Verlag, 1999.

[LPY97] K.G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *Springer International Journal of Software Tools for Technology Transfer*, 1(1+2), 1997.

[LV96] Nancy Lynch and Frits Vaandrager. Forward and backward simulations, II: Timing-based systems. *Information and Computation*, 128(1):1–25, 10 July 1996.

[OSY94] A. Olivero, J. Sifakis, and S. Yovine. Using abstractions for the verification of linear hybrid systems. In David L. Dill, editor, *Proceedings of the sixth International Conference on Computer-Aided Verification CAV*, volume 818 of *Lecture Notes in Computer Science*, pages 81–94, Stanford, California, USA, June 1994. Springer-Verlag.

[PLNS99] M. Prandini, J. Lygeros, A. Nilim, and S. Sastry. A probabilistic framework for aircraft conflict detection. In *Proceedings of the AIAA Guidance, Navigation and Control Conference*, August 1999.

[SL95] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[SMF97] T. Stauner, O. Müller, and M. Fuchs. Using HyTech to verify an automotive control system. In O. Maler, editor, *HART'97, Proceedings of the 1st*

International Workshop on Hybrid and Real-Time Systems, Lecture Notes in Computer Science 1201. Springer-Verlag, 1997.

[Tri98] S. Tripakis. *L'Analyse Formelle des Systèmes Temporisés en Pratique*. PhD thesis, Université Joseph Fourier, 1998.

[VvS99] F. Vaandrager and J. van Schuppen, editors. *HSCC 99: Hybrid Systems—Computation and Control*. Lecture Notes in Computer Science 1569. Springer-Verlag, 1999. Proceedings of the Second International Workshop.