Verifying Quantitative Properties of Continuous Probabilistic Timed Automata*

Marta Kwiatkowska[1], Gethin Norman[1], Roberto Segala[2] and Jeremy Sproston[2]

[1] University of Birmingham, Birmingham B15 2TT, United Kingdom
{M.Z.Kwiatkowska,G.Norman,J.Sproston}@cs.bham.ac.uk
[2] Dipartimento di Scienze dell'Informazione, Università di Bologna, Mura Anteo Zamboni 7, 40127 Bologna, Italy
segala@cs.unibo.it

Abstract. We consider the problem of automatically verifying real-time systems with *continuously* distributed random delays. We generalise *probabilistic timed automata* introduced in [19], an extension of the timed automata model of [4], with clock resets made according to continuous probability distributions. Thus, our model exhibits nondeterministic and probabilistic choice, the latter being made according to both discrete and continuous probability distributions. To facilitate algorithmic verification, we modify the standard region graph construction by subdividing the unit intervals in order to *approximate* the probability to within an interval. We then develop a model checking method for continuous probabilistic timed automata, taking as our specification language Probabilistic Timed Computation Tree Logic (PTCTL). Our method improves on the previously known techniques in that it allows the verification of *quantitative* probability bounds, as opposed to qualitative properties which can only refer to bounds of probability 0 or 1.

1 Introduction

Background: In recent years we have witnessed an increase in demand for formal models and verification techniques for real-time systems such as communication protocols, digital circuits with uncertain delay lengths, and media synchronization protocols. Automatic verification of quantitative timing constraints has particularly been subject to significant attention, as indicated by the development of associated software tools [9, 11] and their successful application in industrial case studies.

Traditional approaches to real-time systems define behaviour purely in terms of non-determinism. However, it may be desirable to express the relative *likelihood* of the occurrence of certain behaviour. For example, we may wish to model a system in which an event is triggered after a random, *continuously distributed* delay, where the distribution may be e.g. uniform, normal, or exponential. Such a framework would be particularly useful when modelling environments with

* Supported in part by EPSRC grants GR/M04617, GR/M13046 and GR/N22960.

unpredictable behaviour; for instance, those involving component failure or customer arrivals in a network. Furthermore, we may also wish to refer to the likelihood of certain temporal logic properties being satisfied by such a real-time system, and to have a model checking algorithm for verifying the truth of these assertions. The remit of this paper is to address these issues.

To provide an appropriate foundation for our work, we take the model of *timed automata* [4], a framework for modelling non-deterministic real-time systems and a focus of much attention from both theoretical researchers and verification practitioners alike. A timed automaton takes the form of a finite directed graph equipped with a set of variables referred to as *clocks*. Since clocks are real-valued, the state space of a timed automaton is infinite, and hence verification is performed by constructing a finite-state quotient of the system model, called a *region graph* [3], which is then subject to established model checking techniques. Recently, we have shown that this region graph construction can also be applied to timed automata augmented with *discrete* probability distributions [19]. This result provides a method for verifying such probabilistic timed automata against PTCTL. This result relies on the fact that all of the states encoded into a single region satisfy the same formulae. However, if our system model admits continuously distributed random delays, the latter property does not hold.

Motivating example: We illustrate why the region graph approach does not work with continuously distributed random delays by means of the example below due to Rajeev Alur [1]. Suppose in state s, at time $t = 0$, a clock d is set to values in the interval $(0, 1)$ according to some continuous density function. Now suppose that, at time $t < 1$, a transition occurs to state s' where d remains scheduled and another clock d' is newly scheduled, again set to values in the interval $(0, 1)$. Consider the three possible relationships between the clocks d and d' in state s':

$$(1)\ d' < d \qquad\qquad (2)\ d' = d \qquad\qquad (3)\ d' > d$$

The region-based approach only encodes the information that: (2) has probability 0, while (1) and (3) both have positive probability. However, the actual probabilities *depend on the value* of t (when the transition from s to s' is made).

Therefore, to perform any *exact* probability calculations with respect to continuous probabilistic real-time systems, we will require an *infinite* model, since, as can be seen in the example above, for each different value of t the probabilities of (1) and (3) will differ. On the other hand, a *finite* model is required for decidable automatic verification.

Main contribution: Our key proposal is to *refine* the region graph construction by subdividing the equivalence classes of clock values. In particular, we split the unit intervals into n subintervals of equal size. Intuitively, this process increases the granularity of the partitioning in order for the region graph to more accurately retain information concerning the continuous random delays. However, note that, for finite n, this will constitute an *approximation* of the exact probability values involved; we supplement such approximate answers with an estimate of the *error*. Further refinements into yet smaller intervals yields better approximations of the probability bounds.

The main technical challenges in our approach are threefold. Firstly, we must define the probability measure and σ-algebra of the infinite paths in the presence of continuously distributed delays and dense time. Secondly, we need to show how the refined region graph may be constructed, where the particular difficulty is due to not knowing the relative order of clocks which are set continuously at random. Thirdly, we have to estimate the error caused by the finite approximation.

Related work: There are many probabilistic verification frameworks, see e.g. [6, 16, 13, 14], most of which only handle discrete probability and time. Our results relate to those of [2, 12], which concern the model checking of probabilistic real-time systems with continuous random delays against qualitative properties that can only refer to probability bounds of 0 or 1. In [7], a quantitative model checking procedure for continuous time Markov chains is presented. Recently a method for approximating continuous Markov processes has been proposed in [15], but the relationship between their approach and ours is not yet known.

2 Preliminaries

Throughout, we use standard notation from timed automata, based on [3]. Labelled paths are non-empty finite or infinite sequences of the form: $\omega = \sigma_0 \xrightarrow{l_0} \sigma_1 \xrightarrow{l_1} \sigma_2 \xrightarrow{l_2} \cdots$ where σ_i are states and l_i are labels for transitions. The first state of ω is denoted by $\mathit{first}(\omega)$. If ω is finite then the last state of ω is denoted $\mathit{last}(\omega)$. The length of a path is defined in the standard way (∞ if the path is infinite) and is denoted $|\omega|$. The prefix relation on paths is denoted by $\leq$ and the concatenation by juxtaposition. If $k \in \mathbb{N}$ then $\omega(k)$ denotes the k-th state, $\mathit{step}(\omega, k)$ the label of the k-th step, and $\omega^{(k)}$ denotes the k-th prefix of ω.

We assume some familiarity with probability and measure theory, see e.g. [5]. Consider a set Ω. A *σ-field* on Ω, denoted $\mathcal{F}$, is a field closed under countable union. The elements of a σ-field are called the *measurable sets*, and $(\Omega, \mathcal{F})$ is called a *measurable space*. Let $(\Omega, \mathcal{F})$ be a measurable space. A function $P : \mathcal{F} \to [0, 1]$ is a *probability measure* on $(\Omega, \mathcal{F})$, and $\mathcal{P} = (\Omega, \mathcal{F}, P)$ a *probability space*, if P satisfies the following properties: $P(\Omega) = 1$, and if $A_1, A_2, \ldots$ is a disjoint sequence of elements of $\mathcal{F}$, then $P(\cup_i A_i) = \sum_i P(A_i)$.

A *continuous density function* (cdf) on $\mathbb{R}$ is a function f such that $f(x) \geq 0$ for all $x \in \mathbb{R}$ and $\int_{-\infty}^{+\infty} f(x)dx = 1$. Furthermore, f has *support* $A \subseteq \mathbb{R}$ if $f(x) = 0$ for all $x \in \mathbb{R} \setminus A$. We define a cdf f to be *positive bounded* if its support lies within a closed interval of $\mathbb{R}^{\geq 0}$. We denote by PB the set of positive bounded cdfs, and the set of *discrete probability distributions* over a (finite) set S by $\mu(S)$.

2.1 Dense Markov Processes

Definition 1. *A* dense Markov Process M *is a tuple* $(Q, \mathcal{F}, q_0, \{P_q\}_{q \in Q})$*, where* Q *is a set of* states*,* $\mathcal{F}$ *is a σ-field over* Q*,* q_0 *is the* initial state*, and each* P_q *is a probability measure on* $(Q, \mathcal{F})$*.*

Observe that we do not impose any limit on the cardinality of Q and that the probability spaces associates with the states are not necessarily discrete. For notational convenience we denote the dense Markov process $(Q, q, \mathcal{F}, \{P_{q'}\}_{q' \in Q})$ by M_q. Our objective is to define a probability space, $\mathcal{P}_{M_q} = (\Omega_{M_q}, \mathcal{F}_{M_q}, P_{M_q})$, for the infinite sequences of states that can be generated by a dense Markov process.

The sample set Ω_{M_q} is the set of infinite sequences qQ^ω of states starting in q. For the σ-field we generalise the cone construction for ordinary Markov processes. The generalisation of a cone is a set of sequences of Ω_{M_q} that extend some appropriate set of finite sequences. Formally, given a dense Markov process M_q and a finite sequence $X_1 X_2 \dots X_k$ of elements of $\mathcal{F}$, the set of sequences

$$B_{X_1 X_2 \dots X_k} = \{q q_1 q_2 \dots q_k \alpha \mid \alpha \in Q^\omega \text{ and } q_i \in X_i \text{ for all } 1 \leq i \leq k\}$$

is called a *basic set*. Special basic sets are $B_\epsilon = \Omega_{Mq}$ (ϵ denotes the empty sequence) and $B_\perp = \emptyset$. We use β to range over sequences of elements of $\mathcal{F}$.

The next step is to assign a measure to basic sets. It turns out that we cannot assign a measure to all basic sets in general. We define basic measurable sets together with their measures by induction.

Definition 2 (Basic measurable sets). *The basic sets $B_\perp$ and B_ϵ are measurable. The measure of $B_\perp$ is $P_{M_q}[B_\perp] = 0$ and the measure of B_ϵ is $P_{M_q}[B_\epsilon] = 1$.*

A basic set $B_{X\beta}$ is a basic measurable set if

1. *B_β is a basic measurable set; and*
2. *the function $f_{X\beta}$ that maps the state q to $P_{M_q}[B_{X\beta}]$ is measurable from $(Q, \mathcal{F})$ to the Borel σ-field over the interval $[0,1]$, where $P_{M_q}[B_{X\beta}]$ is defined to be $\int_Q f_\beta I_X dP_q$ and I_X denotes the indicator function of X.*

Note that in the integral above f_β is measurable because B_β is a basic measurable set, and I_X is measurable because $X \in \mathcal{F}$. Thus the above integral is well defined by [5, Theorem 1.5.9].

Following an argument similar to [20], we can show that the measure P_{M_q} is σ-additive on the basic measurable sets. If all basic sets are basic measurable, then we can generate the minimum field that contains the basic measurable sets and show that there is a unique extension of the measure P_{M_q}. Thus, we can simply define the σ-field $\mathcal{F}_{M_q}$ to be the σ-field generated by the basic measurable sets, and extend the measure P_{M_q} using [5, Theorem 1.3.6].

3 Definition of the model

Let AP be a set of atomic propositions. A *clock* x is a non-negative real-valued variable which increases at the same rate as real-time. Let $\mathcal{X}$ be a set of clocks, and let $\nu : \mathcal{X} \to \mathbb{R}^{\geq 0}$ be a function assigning a non-negative real value to each of the clocks in this set. Such a function is called a *clock assignment*. For any $X \subseteq \mathcal{X}$ and $t \in \mathbb{R}^{\geq 0}$, we write $\nu[X := t]$ for the clock assignment that assigns t

to all clocks in X, and agrees with ν for all clocks in $\mathcal{X} \setminus X$. In addition, $\nu + t$ denotes the clock assignment for which all clocks x in $\mathcal{X}$ take the value $\nu(x) + t$.

As with standard timed automata [4, 17], the conditions for progress between nodes of the graph are described in terms of *clock constraints*.

Definition 3 (Clock Constraints). *Let X be a set of clocks. The set of* clock constraints *of clocks in X, $\mathcal{C}_X$, is defined inductively by the syntax:*

$$\zeta ::= x \leq k \,|\, x \geq k \,|\, x - y \leq k \,|\, x - y \geq k \,|\, \neg\zeta \,|\, \zeta \vee \zeta,$$

where $x, y \in X$ and $k \in \mathbb{N}$.

We say that a clock assignment ν *satisfies* the clock constraint ζ (also denoted $\nu \models \zeta$) if substitution of each $x \in \mathcal{X}$ by $\nu(x)$ results in ζ resolving to true.

Definition 4 (Continuous Probabilistic Timed Automaton). *A continuous probabilistic timed automaton is a tuple $G = (\mathcal{S}, \bar{s}, L, \mathcal{X}, dens, inv, prob, \langle\tau_s\rangle_{s\in\mathcal{S}})$ consisting of*

- *a finite set $\mathcal{S}$ of nodes, including a initial node $\bar{s}$;*
- *a function $L : \mathcal{S} \to 2^{\mathrm{AP}}$ assigning to each node of the graph the set of atomic propositions that are true in that node;*
- *a finite set $\mathcal{X}$ of clocks;*
- *a partial function $dens : \mathcal{S} \times \mathcal{X} \to PB$ assigning to pairs of nodes and clocks a positive bounded density function;*
- *a function $inv : \mathcal{S} \to \mathcal{C}_{\mathcal{X}}$ assigning to each node an invariant condition;*
- *a function $prob : \mathcal{S} \to \mathcal{P}_{fn}(\mu(\mathcal{S}))$, assigning to each node a (finite non-empty) set of discrete probability distributions on $\mathcal{S}$;*
- *a family of functions $\langle\tau_s\rangle_{s\in\mathcal{S}}$ where, for any $s \in \mathcal{S}$, $\tau_s : prob(s) \to \mathcal{C}_{\mathcal{X}}$ assigns an enabling condition to each discrete probability distribution in $prob(s)$.*

Continuous probabilistic timed automata generalise the probabilistic timed automata of [19] through the addition of the partial function *dens*. Whenever defined for a node s and a clock x, this function yields a cdf, say f, which captures the *reset* of x upon entry into s. Such a reset results in a *random* assignment to x (for f a general cdf in PB) or an assignment of an *exact value* (if f is a point distribution). If *dens* is undefined, the clocks keep their old values, as in [2].

The behaviour of the model is as follows. It starts in node $\bar{s}$ with all clocks in $\mathcal{X}$ initialized to 0. The values of all the clocks increase uniformly with time. At any point in time, if the system is in node s then it can behave in one of two ways depending on the values of the clocks in $\mathcal{X}$. It can either let *time advance* such that the invariant condition $inv(s)$ does not become violated, or make a *state transition*, subject to certain conditions given below. State transitions are instantaneous, and generalise the state transitions of the (discrete-)probabilistic timed automata of [19] in the following sense:

- a distribution $p_s \in prob(s)$, whose corresponding enabling condition $\tau_s(p_s)$ is satisfied by the current values of the clocks, is selected *nondeterministically*;

– then, supposing p_s is chosen, for any $s' \in \mathcal{S}$, with probability $p_s(s')$ the system will make a transition to node s' and reassign values to all the clocks x for which $dens(s', x)$ is defined according to the cdfs given by $dens(s', x)$.

For notational convenience, for each node $s \in \mathcal{S}$, we denote by $O(s)$ the set of clocks x for which $dens(s, x)$ is not defined, i.e. those clocks that keep their old value when s is reached, and by $N(s)$ the set of clocks x for which $dens(s, x)$ is defined, i.e. those clocks that are reassigned a new value when s is reached.

Let G be a continuous probabilistic timed automaton. We now define formally the behaviour of G as a probabilistic timed structure. We let $\Gamma(G)$ denote the set of all clock assignments for all the clocks in $\mathcal{X}$.

Definition 5 (State). *A* state *of G is a pair $\langle s, \nu \rangle$ where $s \in \mathcal{S}$, $\nu \in \Gamma(G)$ such that $inv(s)$ is satisfied by ν.*

Definition 6 (Path). *A path of G is an infinite or finite sequence*

$$\omega = \langle s_0, \nu_0 \rangle \xrightarrow{t_0, p_0} \langle s_1, \nu_1 \rangle \xrightarrow{t_1, p_1} \langle s_2, \nu_2 \rangle \xrightarrow{t_2, p_2} \cdots$$

such that, for each $i \in \mathbb{N}$:

1. *$s_i \in \mathcal{S}$, $\nu_i \in \Gamma(G)$, $t_i \in \mathbb{R}^{\geq 0}$ and $p_i \in prob(s_i)$;*
2. *the invariant condition $inv(s_i)$ is satisfied by $(\nu_i + t)$ for all $0 \leq t \leq t_i$;*
3. *the clock assignment $(\nu_i + t_i)$ satisfies $\tau_{s_i}(p_i)$;*
4. *$p_i(s_{i+1}) > 0$, $\nu_{i+1}(x)$ is in the support of $dens(s_{i+1}, x)$ for all $x \in N(s_{i+1})$ and $\nu_{i+1}(x) = \nu_i(x) + t_i$ for all $x \in O(s_{i+1})$.*

For all $0 \leq i \leq |\omega|$, define $\mathcal{T}_\omega(i)$, the elapsed time *until the i^{th} transition, as follows: put $\mathcal{T}_\omega(0) = 0$, and for any $1 \leq i \leq |\omega|$, let $\mathcal{T}_\omega(i) = \sum_{k=0}^{i-1} t_k$.*

Consider an infinite path ω of G. A *position* of ω is a pair (i, t'), where $i \in \mathbb{N}$ and $t' \in \mathbb{R}$ such that $0 \leq t' \leq t_i$. The *state at position* (i, t'), denoted by $\omega(i + t')$, is given by $\langle s_i, \nu_i + t' \rangle$. Given a path ω, $i, j \in \mathbb{N}$ and $t, t' \in \mathbb{R}$ such that $t \leq t_i$ and $t' \leq t_j$, then we say that the position (j, t') *precedes* the position (i, t), written $(j, t') \prec (i, t)$, if and only if $j < i$, or $j = i$ and $t' < t$.

For simplicity, as in [19], we add time successor transitions from each state of the probabilistic timed structure determined by G in which time may diverge. We omit the details of this approach to time divergence from this extended abstract.

Due to the presence of both non-deterministic and probabilistic choice, we use the notion of an adversary, based on e.g. [8]. The role of an adversary is to select, for each finite path of G, the time point t and one of the probability distributions p enabled in the last state of the path.

Definition 7 (Adversary of G). *An* adversary *(or scheduler) of G is a function A mapping every finite path ω of G to a pair (t, p) where $t \in \mathbb{R}^{\geq 0}$ and $p \in \mu(\mathcal{S})$ such that if $last(\omega) = \langle s, \nu \rangle$ then $p \in prob(s)$, $\nu + t'$ satisfies $inv(s)$ for all $0 \leq t' \leq t$, and $\nu + t$ satisfies $\tau_s(p)$.*

For an adversary A of a continuous probabilistic timed automaton G we define the following sets of paths: $Path^A_{fin}\langle s,\nu\rangle$ ($Path^A_{ful}\langle s,\nu\rangle$) is the set of finite (infinite) paths such that $step(\omega,i) = A(\omega^{(i)})$ for all $1 \leq i < |\omega|$ and $first(\omega) = \langle s,\nu\rangle$.

We now turn to the definition of a probability space over the set of infinite paths $Path^A_{ful}\langle s,\nu\rangle$ of a given adversary A and state $\langle s,\nu\rangle$. When we use an adversary to resolve the nondeterminism we obtain a dense Markov process, whose path space is defined in Section 2.1. Each element of the sample set is an infinite chain of paths; however, it is easy to see that such chains can be replaced by their limits under prefix, i.e. an infinite path. We denote by $\mathcal{P}_{A,\langle s,\nu\rangle}$ the probability space over $Path^A_{ful}\langle s,\nu\rangle$.

Below we give an idea of how an adversary A and state $\langle s,\nu\rangle$ generate a dense Markov process $(Q, \mathcal{F}, q_0, \{P_q\}_{q\in Q})$. The set of states Q is the set $Path^A_{fin}\langle s,\nu\rangle$ and the initial state q_0 is $\langle s,\nu\rangle$. The σ-field $\mathcal{F}$ is the σ-field generated by all the sets of the form

$$C_{q,s',\mathcal{I}} = \{q \xrightarrow{t,p} \langle s',\nu'\rangle \in Q \mid \nu'(x) \in \mathcal{I}(x) \text{ for all } x \in N(s')\},$$

where $q \in Q$, $s' \in \mathcal{S}$, $A(q) = (t,p)$ and $\mathcal{I}$ denotes a function mapping clocks to closed intervals. Finally, if $q \in Q$ and $A(q) = (t,p)$, then P_q is defined on the sets $C_{q',s',\mathcal{I}}$ as follows:

$$P_q(C_{q',s',\mathcal{I}}) = \begin{cases} p(s') \cdot \left(\prod_{x\in N(s')} \int_{\mathcal{I}(x)} dens(s',x)\,dx \right) & \text{if } q = q' \\ 0 & \text{otherwise} \end{cases}$$

and then extended to $\mathcal{F}$ using [5, Theorem 1.3.6].

4 Probabilistic Timed Computation Tree Logic (PTCTL)

We now introduce Probabilistic Timed Computation Tree Logic (PTCTL) as our logic for the specification of properties of probabilistic timed automata. Before we can define our logic formally, we will need to appropriately restrict the notion of an adversary of a continuous probabilistic timed automaton G. Due to the unlimited power that we have given to an adversary, it is easy to provide adversaries that would not guarantee the measurability of trivial events such as the occurrence of a single action. On the other hand, such adversaries would be extremely unnatural, and therefore we think it reasonable to rule out such pathological adversaries by definition. Thus, for the rest of this paper, we consider only *feasible* adversaries, that is, those that ensure the *measurability* of all the events identified by PTCTL formulae.

A further restriction on adversaries that we shall require is that of *time-divergence*; it is commonly imposed in real-time systems so that unrealisable behaviour (i.e. corresponding to time not advancing beyond a time bound) is disregarded during analysis. We say that an infinite path ω is *divergent* if for any $t \in \mathbb{R}^{\geq 0}$, there exists $j \in \mathbb{N}$ such that $\mathcal{T}_\omega(j) > t$.

Definition 8 (Divergent adversary). *An adversary A for a continuous probabilistic timed automaton G is* divergent *if and only if for each state $\langle s, \nu \rangle$ of G the probability $P_{A,\langle s,\nu\rangle}$ of the divergent paths of $Path^A_{ful}\langle s, \nu \rangle$ is 1. Let $\mathcal{A}_G$ denote the set of all divergent adversaries of G.*

We now define the syntax and semantics of PTCTL. We have omitted the treatment of reset quantifiers and clock constraints, the addition of which is straightforward, see [19].

Definition 9 (Syntax of PTCTL). *The syntax of* PTCTL *is defined as follows:*

$$\phi ::= \texttt{true} \;\mid\; a \;\mid\; \phi \wedge \phi \;\mid\; \neg\phi \;\mid\; [\phi \,\exists\, \mathcal{U}_{\sim k}\, \phi]_{\sqsupseteq \delta} \;\mid\; [\phi \,\forall\, \mathcal{U}_{\sim k}\, \phi]_{\sqsupseteq \delta}$$

where $a \in \mathrm{AP}$, $k \in \mathbb{N}$, $\delta \in [0,1]$, $\sim \in \{\leq, <, \geq, >\}$, and $\sqsupseteq$ is either $\geq$ or $>$.

Definition 10 (Satisfaction Relation). *For any continuous probabilistic timed automaton G, state $\langle s, \nu \rangle$ of G, set of divergent adversaries $\mathcal{A}_G$ of G, and* PTCTL *formula ϕ, the satisfaction relation $\langle s, \nu \rangle \models_{\mathcal{A}_G} \phi$ is defined inductively as follows:*

$\langle s, \nu \rangle \models_{\mathcal{A}_G} \texttt{true}$	for all $s \in \mathcal{S}$ and $\nu \in \Gamma(G)$
$\langle s, \nu \rangle \models_{\mathcal{A}_G} a$	$\Leftrightarrow a \in L(s)$
$\langle s, \nu \rangle \models_{\mathcal{A}_G} \phi_1 \wedge \phi_2$	$\Leftrightarrow \langle s, \nu \rangle \models_{\mathcal{A}_G} \phi_i$ for all $i \in \{1, 2\}$
$\langle s, \nu \rangle \models_{\mathcal{A}_G} \neg\phi$	$\Leftrightarrow s \not\models_{\mathcal{A}_G} \phi$
$\langle s, \nu \rangle \models_{\mathcal{A}_G} [\phi_1 \,\exists\, \mathcal{U}_{\sim k}\, \phi_2]_{\sqsupseteq \delta}$	$\Leftrightarrow P_{A,\langle s,\nu\rangle}\{\omega \in Path^A_{ful}\langle s, \nu\rangle \mid \omega \models_{\mathcal{A}_G} \phi_1 \,\mathcal{U}_{\sim k}\, \phi_2\} \sqsupseteq \delta$ for some $A \in \mathcal{A}_G$
$\langle s, \nu \rangle \models_{\mathcal{A}_G} [\phi_1 \,\forall\, \mathcal{U}_{\sim k}\, \phi_2]_{\sqsupseteq \delta}$	$\Leftrightarrow P_{A,\langle s,\nu\rangle}\{\omega \in Path^A_{ful}\langle s, \nu\rangle \mid \omega \models_{\mathcal{A}_G} \phi_1 \,\mathcal{U}_{\sim k}\, \phi_2\} \sqsupseteq \delta$ for all $A \in \mathcal{A}_G$
$\omega \models_{\mathcal{A}_G} \phi_1 \,\mathcal{U}_{\sim k}\, \phi_2$	$\Leftrightarrow$ there exists a position (i, t) of ω such that $\mathcal{T}_\omega(i) + t \sim k$, $\omega(i + t) \models_{\mathcal{A}_G} \phi_2$, and for all positions (j, t') of ω such that $(j, t') \prec (i, t)$, $\omega(j + t') \models_{\mathcal{A}_G} \phi_1 \vee \phi_2$.

Note that the feasibility condition we impose on adversaries ensures that the set $\{\omega \mid \omega \in Path^A_{ful}(\langle s, \nu \rangle) \;\&\; \omega \models_{\mathcal{A}_G} \phi_1 \,\mathcal{U}_{\sim k}\, \phi_2\}$ is measurable with respect to the probability space $P_{A,\langle s,\nu\rangle}$ induced by A and $\langle s, \nu \rangle$.

5 The Refined Region Graph

As already observed, the standard region construction applied to a continuous probabilistic timed automaton fails in the case of quantitative probabilistic temporal properties. We propose to quotient over *smaller* intervals of clock values, and to this end subdivide each unit interval into n intervals of the same size for some $n \in \mathbb{N}$. The intuition is that, as we subdivide into smaller regions, we obtain an improvement of the minimum/maximum probability bounds, which in the limit tend to the exact bounds as the number of subdivisions increases.

We deal with the inevitable loss of information caused by the finiteness of the construction by providing a bound on the error.

We first refine the equivalence relation of [4] to intervals of size $\frac{1}{n}$.

Definition 11. *For any $x \in \mathcal{X}$, let k_x be the largest constant x is compared to in any of the invariant and enabling conditions of G. For any $\nu \in \Gamma(G)$ and $x \in \mathcal{X}$, define x to be* relevant *for ν if $\nu(x) \leq k_x$.*

Definition 12 (Clock equivalence). *For clock assignments ν and ν' in $\Gamma(G)$ and $n \in \mathbb{N}$, $\nu \cong_n \nu'$ if and only if the following conditions are satisfied:*

1. *$\forall x \in \mathcal{X}$ either $\lfloor n \cdot \nu(x) \rfloor = \lfloor n \cdot \nu'(x) \rfloor$ or x is not relevant for ν and ν';*
2. *$\forall x, x' \in \mathcal{X}$ relevant for ν:*
   (i) $fract(\nu(x)) < fract(\nu(x'))$ if and only if $fract(\nu'(x)) < fract(\nu'(x'))$.
   (ii) $fract(\nu(x)) > fract(\nu(x'))$ if and only if $fract(\nu'(x)) > fract(\nu'(x'))$.

Let $[\nu]_n$ denote the equivalence class to which ν belongs under $\cong_n$. The following lemma allows us to extend the notion of satisfaction of clock constraints to equivalence classes of clocks.

Lemma 1 ([4]). *Let $\nu, \nu' \in \Gamma(G)$ such that $\nu \cong_n \nu'$. Then, for any clock constraint $\zeta \in \mathcal{C}_{\mathcal{X}}$, ν satisfies ζ if and only if ν' satisfies ζ.*

We now define a probabilistic graph $R_n(G, \phi)$ (where ϕ is a PTCTL formula) whose vertices are pairs consisting of the nodes of G and the equivalence classes with respect to $\cong_n$. As in [3], to improve the complexity of the model checking algorithm, we keep track of the time elapsed when passing through sequences of regions by adding an extra clock $\mathbf{x}$ to $\mathcal{X}$ and setting $k_{\mathbf{x}}$ to be the maximal time-bound appearing in the formula ϕ. We start with some preliminary definitions following the construction in [3, 19].

Definition 13. *Let α and β be distinct equivalence classes of $\Gamma(G)$.*

- *The equivalence class β is said to be the* successor *of α, denoted $succ(\alpha)$, if for all $\nu \in \alpha$, there exists $t > 0$ such that $\nu + t \in \beta$ and $\nu + t' \in \alpha \cup \beta$ for all $0 \leq t' < t$.*
- *The class α is said to be an* invariant class of *s if $succ(\alpha)$ violates the invariant condition $inv(s)$.*
- *The class α is an* end class *if, for all $x \in \mathcal{X}$, x is not relevant for α. If α is an end class then, for any $s \in \mathcal{S}$, $\langle s, \alpha \rangle$ is an* end region.

Thus, if we are in an *invariant class* of s then we cannot let time advance sufficiently to move into a new equivalence class without the invariant condition being violated. If we are in an *end class* then we can remain in this region and let time diverge.

The next step is to define the transition relation over regions. As in the standard approach, there are two types of transitions, due to passage of time and change of state respectively, which we consider in turn.

Transitions due to *passage of time* are straightforward using Definition 13: the region that can be reached from $\langle s, \alpha \rangle$ due to passage of time is $\langle s, succ(\alpha) \rangle$. *State* *transitions* are more complex to deal with. Suppose that we are in a region $\langle s, \alpha \rangle$ and a state transition occurs. Then, by definition of the model, the following two choices are made in succession:

– a discrete probability distribution $p_s \in \mathit{prob}(s)$, where $p_s \in \mu(\mathcal{S})$, such that the enabling condition $\tau_s(p_s)$ is satisfied by α, is selected non-deterministically;
– then, supposing p_s is chosen, a transition is made according to p_s.

In order to establish which equivalence classes the system moves to, we consider what happens to the values of all the clocks when the transition is made. Consider a transition from a region $\langle s, \alpha \rangle$ to some node s'. To understand the possible equivalence classes of clock assignments associated with s', we consider the equivalence classes separately for the clocks of $O(s')$, denoted α'_O and of $N(s')$, denoted α'_N.

The equivalence class α'_O is the restriction of α to the clocks of $O(s')$, since the clocks of $O(s')$ remain unchanged. The clocks of $N(s')$ are assigned new values at random, and thus the new equivalence class α'_N is determined by a probability distribution that can be computed through simple integrations. We call $P_{N(s')}$ the *joint probability measure* for the clocks of $N(s')$.

The problem is how to combine the equivalence classes of α'_O and α'_N to obtain a unique equivalence class of clock assignments, since we have no way of stating the relative orders between the clocks in $O(s')$ and $N(s')$. Indeed, there are several possible equivalence classes that work correctly. Since we have no way of determining which one is correct, we introduce a *nondeterministic choice* between them all, and consequently *an error* which we analyze in Section 6. The definition below can be used to determine all the possible equivalence classes that are consistent with α'_O and α'_N.

Definition 14. *If α_1 and α_2 are equivalence classes of clock assignments defined on some subset of clocks X_1 and X_2 respectively such that $X_1 \cap X_2 = \emptyset$, then we let $\alpha_1 \cup \alpha_2$ be the set of equivalence classes over $X_1 \cup X_2$ such that $\gamma \in \alpha_1 \cup \alpha_2$ if and only if $\gamma \restriction X_1 = \alpha_1$, $\gamma \restriction X_2 = \alpha_2$, where $\restriction$ denotes restriction.*

Based on the discussion above, we introduce the notion of a *union region*, which is a triple $\langle s, \alpha_O, \alpha_N \rangle$, where α_O is an equivalence class of $O(s)$, and α_N is an equivalence class of $N(s)$.

We are now ready to formulate the region graph for a continuous probabilistic timed automaton G and PTCTL formula ϕ.

Definition 15 (Region Graph). *The region graph $R_n(G, \phi)$ is defined to be the graph (V, Rstep). The vertex set V is the set of regions and union regions (satisfying the corresponding invariant condition). The probabilistic edge function $\mathit{Rstep} : V \to \mathcal{P}_f(\mu(V))$ consists of three types of steps:*

– (**passage of time**) *if $\langle s, \alpha \rangle \in V$ and α is not an invariant class of s, then the point distribution over $\langle s, \mathit{succ}(\alpha) \rangle$ is an element of $\mathit{Rstep}\langle s, \alpha \rangle$.*
– (**transitions of** G) *if $\langle s, \alpha \rangle \in V$, $p_s \in \mathit{prob}(s)$ and $\tau_s(p_s)$ is satisfied by α, then the distribution p such that the probability of each union region $\langle s', \alpha'_1, \alpha'_2 \rangle$ is $p_s(s') P_{N(s')}(\alpha'_2)$ when $\alpha'_1 = \alpha \restriction O(s)$ and is 0 otherwise is an element of $\mathit{Rstep}\langle s, \alpha \rangle$.*

– (**division**) *if* $\langle s, \alpha_1, \alpha_2 \rangle \in V$*, then for each* $\alpha' \in \alpha_1 \cup \alpha_2$*, the point distribution over* $\langle s, \alpha' \rangle$ *is an element of* $Rstep\langle s, \alpha_1, \alpha_2 \rangle$*.*

The definition of a path for a region graph is similar to the definition of a path for a continuous probabilistic timed automaton with the exception that the labels of the arrows do not contain time values.

Definition 16 (Adversary of R_n). *A (randomized) adversary B on the region graph is a function B mapping every finite path π of $R_n(G, \phi)$ to a distribution over $Rstep(last(\pi))$.*

The definition of a probability space $\mathcal{P}_{B,v}$ on $Path^B_{ful}(v)$, given a randomized adversary B and a region v, is standard [10, 20]. The definition of divergent adversaries can also be adapted easily to region graphs (see [19]).

6 Model Checking Continuous Probabilistic Timed Automata

The aim of this paper is to extend the result of [19], which we now recall. Suppose G is a *discrete* probabilistic timed automaton and the mapping $\phi \mapsto \Phi$ from PTCTL to PBTL [8] is as defined in [19]. Then, for any $\langle s, \nu \rangle \in G$ and $\phi \in$ PTCTL, we have

$$\langle s, \nu \rangle \models_{\mathcal{A}_G} \phi \text{ if and only if } R_1\langle s, \nu \rangle \models_{\mathcal{A}_{R_1}} \Phi$$

where $R_1\langle s, \nu \rangle$ denotes the unique state $\langle s', \alpha' \rangle \in R_1(G, \phi)$ of the region graph such that $s' = s$ and $\alpha = [\nu[\mathbf{x} := 0]]_1$. In particular, the formula $[\phi_1 \,\forall \mathcal{U}_{\sim k}\, \phi_2]_{\sqsupseteq \delta}$ is mapped to $[\Phi_1 \,\forall \mathcal{U}\, (\Phi_2 \wedge a_{\mathbf{x} \sim k})]_{\sqsupseteq \delta}$, where $a_{\mathbf{x} \sim k}$ is the atomic proposition which encodes the time bound subscript $\sim k$, and labels a region $\langle s, \alpha \rangle$ if and only if $\alpha \models \mathbf{x} \sim k$. By abuse of notation, we abbreviate $\Phi_1 \,\mathcal{U}\, (\Phi_2 \wedge a_{\mathbf{x} \sim k})$ to $\Phi_1 \,\mathcal{U}_{\sim k}\, \Phi_2$.

Let G be a continuous probabilistic timed automaton, ϕ be a PTCTL formula, and $R_n(G, \phi)$ be the region graph for G and ϕ with each unit interval refined into n parts. The region graph $R_n(G, \phi)$ does not preserve the validity of ϕ in general since its construction does not preserve the probabilities of events. In particular, it is not the case that a state $\langle s, \nu \rangle \in G$ satisfies ϕ if and only if the corresponding state $R_n\langle s, \nu \rangle$ of $R_n(G, \phi)$ satisfies Φ.

To understand the problem better, let $\phi = [\phi_1 \,\forall \mathcal{U}_{\sim k}\, \phi_2]_{\geq \delta}$. Suppose that there is a known upper bound λ on the error that we introduce by evaluating the probability of $\Phi_1 \,\mathcal{U}_{\sim k}\, \Phi_2$ on $R_n\langle s, \nu \rangle$ rather than on $\langle s, \nu \rangle$ (see Section 6.1 for the method to compute λ), and that the minimum, over all $B \in \mathcal{A}_{R_n(G,\phi)}$, of the probability of the paths of B starting from $R_n\langle s, \nu \rangle$ and satisfying $\Phi_1 \,\mathcal{U}_{\sim k}\, \Phi_2$ is p_1. Then we can deduce that, from $\langle s, \nu \rangle$, there exists an adversary for which the probability of paths from $\langle s, \nu \rangle$ satisfying $\phi_1 \,\mathcal{U}_{\sim k}\, \phi_2$ is in the interval $[p_1, p_1 + \lambda]$.

If $\delta \leq p_1$, then we can conclude that ϕ is valid; if $\delta > p_1 + \lambda$, then we can conclude that ϕ is not valid. If δ is in the interval $(p_1, p_1 + \lambda]$, then Φ is valid in $R_n\langle s, \nu \rangle$; however, ϕ may or may not be valid in $\langle s, \nu \rangle$. In this case we have three possible choices for how to proceed:

1. consider a more refined graph in the hope of solving the uncertainty;
2. say that we do not know the correct answer ("don't know");
3. say that the formula is valid and warn the user that there may be an error of λ in the determination of the probability bound.

The first case has obvious complexity implications. In the second case we need to deal with a three-valued logic, which would involve propagating the "don't know" values to the higher levels of the parse tree of the PTCTL formula in question. In the third case the difficulty is that we cannot quantify the propagation of the error to super-formulae of ϕ. This is because in the worst case we may estimate wrongly the validity of ϕ on most of the states of G. Thus, the only thing that we can say safely in this case is that at each level we may be wrong by some value λ in the estimation of probabilities.

The results we obtain allow us to adopt the third solution, namely, to calculate an interval of probabilities to which the actual probability bound belongs, together with an estimate of error, for a given number of subdivisions n, and refine the region graph further in case "don't know" outcomes have resulted.

6.1 Main Results

Before we can state our results we need some auxiliary definitions. For the rest of the discussion we fix a continuous probabilistic timed automaton G and a formula ϕ. Let s, s' be nodes of G and let α, α' be sets of clock assignments for the clocks of G. We say that $\langle s, \alpha\rangle$ is *contained* in $\langle s', \alpha'\rangle$, denoted $\langle s, \alpha\rangle \leq \langle s', \alpha'\rangle$, if $s = s'$ and $\alpha \subseteq \alpha'$. Given two region graphs R_m, R_n, we say that R_m *refines* R_n, denoted by $R_m \leq R_n$, if each region of R_n is contained in a region of R_m. The next lemma implies that the probability bounds do not increase with further subdivisions of the region graph.

Lemma 2. *$R_m(G, \phi) \leq R_n(G, \phi)$ if n divides m.*

The next notion plays an important part in estimating the error. Let R_m, R_n be two region graphs such that $R_m \leq R_n$, and $v = \langle s, \alpha_1, \alpha_2\rangle$ a union region of R_m. We say that v is *homogeneous* with respect to R_n if there exists a unique region v' of R_n that contains each region of $\{\langle s, \gamma\rangle \mid \gamma \in \alpha_1 \cup \alpha_2\}$.

Fix a refined region graph R_n for G, ϕ and some $n \in \mathbb{N}$. For any $A \in \mathcal{A}_G$, $\langle s, \nu\rangle \in G$, $B \in \mathcal{A}_{R_n}$, $\langle s, \alpha\rangle \in R_n$ and $\phi_1, \phi_2 \in$ PTCTL we let:

$$P^A_{\phi_1 \mathcal{U}_{\sim k} \phi_2}\langle s, \nu\rangle \stackrel{\text{def}}{=} P_{A,\langle s,\nu\rangle}\{\omega \mid \omega \in \mathit{Path}^A_{\mathit{ful}}\langle s, \nu\rangle \text{ and } \omega \models_{\mathcal{A}_G} \phi_1 \,\mathcal{U}_{\sim k}\, \phi_2\}$$
$$P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha\rangle \stackrel{\text{def}}{=} P_{B,\langle s,\alpha\rangle}\{\pi \mid \pi \in \mathit{Path}^B_{\mathit{ful}}\langle s, \alpha\rangle \text{ and } \pi \models_{\mathcal{A}_{R_n}} \Phi_1 \,\mathcal{U}_{\sim k}\, \Phi_2\}.$$

Suppose that $\phi_1, \phi_2 \in$ PTCTL are such that for any $\langle s, \nu\rangle \in G$:

$$\langle s, \nu\rangle \models_{\mathcal{A}_G} \phi_1 \Leftrightarrow R_n\langle s, \nu\rangle \models_{\mathcal{A}_{R_n}} \Phi_1 \text{ and } \langle s, \nu\rangle \models_{\mathcal{A}_G} \phi_2 \Leftrightarrow R_n\langle s, \nu\rangle \models_{\mathcal{A}_{R_n}} \Phi_2.$$

Then we can show the following correspondence holds between adversaries of the automaton G and its region graph R_n (see [18]).

Proposition 1. *For any $A \in \mathcal{A}_G$, $\langle s, \nu \rangle \in G$ and $n \in \mathbb{N}$, there exists $B \in \mathcal{A}_{R_n}$ such that $P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle = P^A_{\phi_1 \mathcal{U}_{\sim k} \phi_2}\langle s, \nu \rangle$ where $R_n\langle s, \nu \rangle = \langle s, \alpha \rangle$.*

Proposition 2. *For any $n \in \mathbb{N}$, $B \in \mathcal{A}_{R_n}$, $\langle s, \alpha \rangle \in R_n$ and $\langle s, \nu \rangle \in G$ with $R_n\langle s, \nu \rangle = \langle s, \alpha \rangle$, there exists $A \in \mathcal{A}_G$ such that $P^A_{\phi_1 \mathcal{U}_{\sim k} \phi_2}\langle s, \nu \rangle$ and $P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle$ differ by at most the probability of reaching a non-homogeneous region before satisfying or violating $\Phi_1 \, \mathcal{U}_{\sim k} \, \Phi_2$.*

As a corollary of Proposition 1 and Proposition 2, we obtain the following crucial correspondence between the probability bounds calculated on R_n and those on G. Its importance is in stating that the probabilities of a PTCTL until formula over the divergent adversaries of G are bounded by the probabilities for the corresponding PBTL formula over the divergent adversaries of the region graph. The latter probability calculation is standard and proceeds via reduction to a linear programming problem [10, 8]. Moreover, since the difference in these values can be no more than the probability of reaching a non-homogeneous region before satisfying or violating $\Phi_1 \, \mathcal{U}_{\sim k} \, \Phi_2$, this yields the estimate of error. This error can also be calculated by standard methods [10, 8].

Corollary 1. *For any $\langle s, \nu \rangle \in G$ and $n \in \mathbb{N}$, if $R_n\langle s, \nu \rangle = \langle s, \alpha \rangle$ and the maximum probability of reaching a non-homogeneous region before satisfying or violating $\Phi_1 \, \mathcal{U}_{\sim k} \, \Phi_2$ from $R_n\langle s, \nu \rangle$ is λ, then*

$$\inf_{A \in \mathcal{A}_G} P^A_{\phi_1 \mathcal{U}_{\sim k} \phi_2}\langle s, \nu \rangle \in \left[\min_{B \in \mathcal{A}_{R_n}} P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle, \min_{B \in \mathcal{A}_{R_n}} P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle + \lambda \right]$$

$$\sup_{A \in \mathcal{A}_G} P^A_{\phi_1 \mathcal{U}_{\sim k} \phi_2}\langle s, \nu \rangle \in \left[\max_{B \in \mathcal{A}_{R_n}} P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle - \lambda, \max_{B \in \mathcal{A}_{R_n}} P^B_{\Phi_1 \mathcal{U}_{\sim k} \Phi_2}\langle s, \alpha \rangle \right] .$$

6.2 Example

We illustrate the working of our method with the help of an example. Consider the automaton H in Figure 1. From s_0 we enable a transition that moves to s_1 and sets x uniformly in the interval $[0, 1]$. From s_1 we enable two transitions: one transition, T_1, moves to node s_2 and sets y uniformly in the interval $[0, 1]$, while the other transition, T_2, moves to s_3 with probability $\frac{2}{3}$ and to s_4 with probability $\frac{1}{3}$. From s_2 we enable a transition to s_4 if $y > x$. We consider the upper bound on the probability of reaching of s_4, i.e. the formula $[\mathtt{true} \, \forall \, \mathcal{U}_{\geq 0} \, a_{s_4}]_{\sqsupseteq \delta}$. The adversary that gives the highest probability $(\frac{5}{9})$ is obtained by scheduling T_1 immediately in s_1 if $x < \frac{2}{3}$ and T_2 otherwise.

From s_0 to s_1 the possible regions that can be reached are $\frac{k}{n} < x < \frac{k+1}{n}$ for $k = 0, \ldots, n-1$, each with probability $\frac{1}{n}$. In the region $\langle s_1, \frac{k}{n} < x < \frac{k+1}{n} \rangle$ there is a choice between letting time advance, taking the transition T_1 or the transition T_2. It follows that the maximum probability of reaching s_4 equals: $\sum_{k=0}^{n-1} \frac{1}{n} \cdot \max\left(\frac{n-k}{n}, \frac{1}{3}\right)$.

To reach a non-homogeneous region, y must be set, then supposing x has been set already, this has probability $\frac{1}{n}$. Therefore, the maximum probability of reaching a non-homogeneous region is $\frac{1}{n}$, which yields the following:

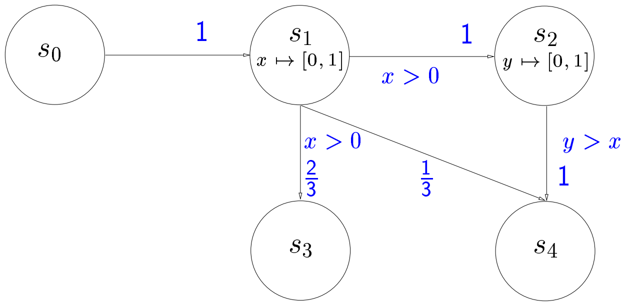

Fig. 1. The continuous probabilistic timed automaton H

- R_1: upper bound is 1, error $\frac{4}{9}$ and estimate of error 1;
- R_2: upper bound is $\frac{3}{4}$, error $\frac{7}{36}$ and estimate of error $\frac{1}{2}$;
- R_4: upper bound is $\frac{31}{48}$, error $\frac{13}{144}$ and estimate of error $\frac{1}{4}$;
- R_{100}: upper bound is $\frac{5589}{10000}$, error $\frac{301}{90000}$ and estimate of error $\frac{1}{100}$.

7 Conclusions

We have proposed a model checking method for continuous probabilistic timed automata against PTCTL specifications. In the formalism we propose, we can specify timing properties such as "at least 80% of packets will be delivered within k units of time assuming the packets arrive according to f" where f is a continuous-time probability distribution (for example, uniform or normal) with support within a closed interval of $\mathbb{R}^{\geq 0}$. The model checking algorithm runs on a finite region-like graph, obtained through subdividing the unit intervals. We show how to approximate the probability to within an interval, where approximations improve with further subdivisions, and estimate the error of the approximation.

It is known that the complexity of the verification of real-time systems is expensive, and the method proposed here is no exception. Research into improving the complexity of our procedure, for example using symbolic methods, would be necessary before it can be applied to real-world problems.

8 Acknowledgements

We thank Pedro D'Argenio for pointing out a flaw in our previous attempt to solve this problem. We also thank the anonymous referees for their helpful comments.

References

[1] R. Alur. Private communication. 1998.
[2] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for probabilistic real-time systems. In *Proc. ICALP'91*, volume 510 of *LNCS*. Springer, 1991.

[3] R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1), 1993.

[4] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126, 1994.

[5] R. B. Ash. *Real Analysis and Probability*. Academic Press, 1972.

[6] C. Baier, E. Clarke, V. Hartonas-Garmhausen, M. Kwiatkowska, and M. Ryan. Symbolic model checking for probabilistic processes. In *Proc. ICALP'97*, volume 1256 of *LNCS*. Springer, 1997.

[7] C. Baier, J.-P. Katoen, and H. Hermanns. Approximate symbolic model checking of continuous-time Markov chains. In *CONCUR'99*, volume 1664 of *LNCS*. Springer, 1999.

[8] C. Baier and M. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11, 1998.

[9] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, W. Yi, and C. Weise. New generation of UPPAAL. In *Proc. International Workshop on Software Tools for Technology Transfer*, 1998.

[10] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *FST and TCS*, volume 1026 of *LNCS*. Springer, 1995.

[11] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: a model-checking tool for real-time systems. In *Proc. CAV'98*, volume 1427 of *LNCS*. Springer, 1998.

[12] P. D'Argenio, J.-P. Katoen, and E. Brinksma. Specification and analysis of soft real-time systems: Quantity and quality. In *Proc. IEEE Real-Time Systems Symposium*. IEEE Computer Society Press, 1999.

[13] L. de Alfaro. How to specify and verify the long-run average behaviour of probabilistic systems. In *Proc. LICS'98*. IEEE Computer Society Press, 1998.

[14] L. de Alfaro. Stochastic transition systems. In *Proc. CONCUR'98*, volume 1466 of *LNCS*. Springer, 1998.

[15] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Approximating labelled Markov processes. To appear in LICS'2000.

[16] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(4), 1994.

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

[18] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of quantitative properties of continuous probabilistic real-time automata. Technical Report CSR-00-06, University of Birmingham, 2000.

[19] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. Technical Report CSR-00-02, University of Birmingham, 2000. Accepted for a Special Issue of Theoretical Computer Science. Preliminary version of this paper appeared in Proc. ARTS'99, LNCS vol 1601, 1999.

[20] R. Segala. *Modelling and Verification of Randomized Distributed Real Time Systems.* PhD thesis, Massachusetts Institute of Technology, 1995.