On the Verification of Nondeterministic Automata Specifications of Probabilistic Real-Time Systems

Arnaldo V. Moura and Guilherme A. Pinto
{arnaldo,guialbu}@dcc.unicamp.br

Relatório Técnico IC–99-27

Dezembro de 1999

On the Verification of Nondeterministic Automata Specifications of Probabilistic Real-Time Systems

Arnaldo V. Moura and Guilherme A. Pinto
{arnaldo,guialbu}@dcc.unicamp.br

Abstract

In [2], Alur et al. presented an algorithm for the problem of verifying deterministic timed automata specifications of probabilistic real-time systems given as generalized semi-Markov processes; and posed the question of the verification of nondeterministic specifications. We give a partial answer to the question, extending their method so that processes, with a fairly acceptable restriction, can be tested against any timed automaton. These include, for instance, real-time models of digital circuits where the key property is that the delay distributions have non-zero lower bounds.

1 Introduction

Techniques for automatic verification of timing properties of non-probabilistic real-time systems have already been widely studied [5, 3, 12] and applied in practical problems. In many cases, however, one may want to consider the probabilistic behavior of a physical system—in these cases, the best a verification method can do is to say that the system satisfies the property with probability one. Typically, system models based on state-transition graphs can account for probabilities either only in the discrete transitions, or also in the delays of the events triggering the discrete transitions. In the latter case, when there are continuous probability distributions in the delays, there are, at least, three approaches.

In [1], Alur et al. presented a model-checking algorithm for timed computation tree logic (TCTL) formulas, where the existential and universal path quantifiers are interpreted as "with positive probability" and "with probability one" respectively, so that the verification is qualitative (in what concerns probability bounds). The system model defines a generalized semi-Markov process.

Recently, Kwiatkowska et al. [13] proposed a procedure to check systems, given by a similar model, against formulas of the probabilistic timed computation tree logic (PTCTL), which is a version of TCTL where the path quantifiers are amended with quantitative probabilistic bounds, so that the verification is quantitative, although (unavoidably) approximative.

These logics, however, are purely propositional, and do not have the counting ability, for example, of the popular timed automata (TA) formalism [12]. There are certain interesting properties that can be specified by TA, and such that TCTL or PTCTL cannot describe. An example is the *convergent bounded response* property: "two events a and b alternate

and eventually always the time difference between a and the next b is less than or equal to 2 seconds" [3, 2]. Thus, in [2], Alur et al. extended their previous method to the verification of deterministic TA specifications, which can express the convergent bounded response property, for instance; and posed the question of the verification of nondeterministic TA specifications. Besides the fact that nondeterminism facilitates the specification of properties and gives rise to, potentially, smaller models, the interest in this question also stems from the fact that nondeterministic TA are strictly more expressive than deterministic TA [3]. Unfortunately, the probabilistic verification problem, which is commonly solved by complementing the specification, resembles the language inclusion problem, which is undecidable for nondeterministic TA [3]. Nondeterministic TA are not closed under complementation [3], so that one must resort to some other technique to cope with the nondeterminism in the specification. For instance, in the discrete time domain, where the system is a Markov chain and the specification an ω-automaton, Courcoubetis and Yannakakis [7] solved a more general problem with the usual subset construction. But, for TA, it was not clear in [2] how one could apply such a construction, and also carry through the probabilistic analysis.

In this paper, we show that a subset construction on the states (location plus valuation for the clocks) of the automaton admit the definition of the integral parts/order of fractional parts equivalence relation over the set of states, so that the method in [2] can be extended in a uniform way until the point where only probabilistic information are left. At this point, the method constructs a graph, over which the probabilistic analysis is done. When we apply the subset construction, for some instances, this graph is infinite, and the method cannot be applied—the process and the automaton may synchronize is such a way that an unbounded amount of information is needed for the method to succeed. Nevertheless, we show that for fairly acceptable restrictions on the process model, the graph is guaranteed to be finite, in such a way that the powerful nondeterministic timed automata formalism can indeed be used to specify properties. For example, if there is a bound on the number of events generated by the process, in a time interval of unit length, then our method guarantees that it can be tested against any TA, be it deterministic or nondeterministic.

The paper is organized as follows. Section 2 explains the system model of probabilistic real-time processes. In Section 3 we review the specification formalism of timed automata and define the probabilistic verification problem. Section 4 presents the construction, gives an example for which the method cannot be used, and shows that most practical instances *are* solvable. Section 5 concludes with some final remarks.

2 The System Model

Our system model is inspired on the one defined in [2], and we use the same notation. Informally, the system moves probabilistically through a finite set of states S according to a sequence of discrete events. The sojourn time in a state and the event triggering the probabilistic transition are chosen in such a way that it defines a *generalized semi-Markov process* [16] over the set S. We first give the formal definition and then discuss the operation of the process.

Definition 1 We say that a probability distribution is *bounded* if it has support on a

bounded interval $[t_1, t_2]$, t_1 and t_2 in $\mathbb{Q}$ (the set of non-negative rational numbers), and probability density function f, such that, if $t_1 = t_2$, then $f(t_1) = 1$ (discrete probability distribution), and if $t_1 < t_2$, then $\int_{t_1}^{t_2} f(t)dt = 1$ (continuous probability distribution).

We denote by D the set of all bounded and exponential distributions. The algorithm needs only trivial modifications to treat bounded distributions with support of k bounded intervals, for any k in $\mathbb{N}$ (the set of natural numbers).

Definition 2 A *real-time probabilistic process* $\mathcal{M}$ is a tuple $\langle S, E, act, f_0, next, dist\rangle$, where

- S is a finite set of states;
- E is a finite set of basic events. The set of all nonempty subsets of E is denoted by Δ. An element $a \in \Delta$ is called an event of $\mathcal{M}$;
- $act : S \to \Delta$ is a function associating a set of basic events to each state, the *active* basic events of a state;
- $f_0 : S \to [0, 1]$ is a initial distribution function such that $\sum_{s \in S} f_0(s) = 1$;
- $next : S \times \Delta \times S \to [0, 1]$ is a probabilistic transition function. For every state s and every $a \in \Delta$, $\sum_{s' \in S} next(s, a, s') = 1$;
- $dist : E \to D$ is a function associating each basic event $x \in E$ to a probability distribution in D. Given $dist$, we denote by E_{e} and E_{b}, respectively, the set of basic events with exponential distribution, and the set of basic events with bounded distributions; and distinguish two functions, $l : E_{\mathrm{b}} \to \mathbb{Q}$ and $u : E_{\mathrm{b}} \to \mathbb{Q}$, giving the lower and upper bounds of the distributions, respectively. We call $x \in E_{\mathrm{b}}$ a *fixed-delay* basic event if $l_x = u_x$, and a *variable-delay* basic event if $l_x < u_x$. Let $act_{\mathrm{e}}(s) = act(s) \cap E_{\mathrm{e}}$ and $act_{\mathrm{b}}(s) = act(s) \cap E_{\mathrm{b}}$. Also, we write f_x for the probability density function associated to an exponential or variable delay basic event x.

The process works as follows. Each basic event $x \in E$ has an associated clock, also referred to as x. The process starts in some state s such that $f_0(s) > 0$. For each $x \in act(s)$, a value d_x is randomly and independently chosen from the distribution $dist(x)$ and the clock x is set to $-d_x$. The values of all the clocks increase with the real time until some clock reaches 0, when, then, a state transition occurs. Let $a \in \Delta$ be the set of basic events reaching value 0 in s. The process moves to some state s' such that $next(s, a, s') > 0$. The reading of the clocks in $act(s')$ are obtained as follows:

- Let $old(s, a, s') = act(s') \cap (act(s) \setminus a)$. The basic events in $old(s, a, s')$ are active in s and s'—the value of their clocks is not modified;
- Let $new(s, a, s') = act(s') \setminus old(s, a, s')$. Each clock x in $new(s, a, s')$ is set to $-d_x$, where d_x is, again, randomly and independently chosen from $dist(x)$.

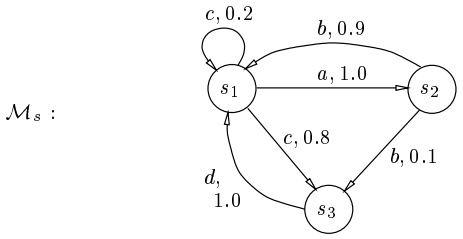

Figure 1: A real-time probabilistic process $\mathcal{M}_s$

Example 1 Consider the process $\mathcal{M}_s$ in Fig. 1, where $act(s_1) = \{a, c\}$, $act(s_2) = \{b\}$ and $act(s_3) = \{d\}$. It models two concurrent basic events a and c, with uniform distribution between 0 and 4, and between 1 and 3, respectively. The process starts in state s_1 and, if a happens before c, then the process goes to s_2 with probability 1; c is no longer active and a response basic event b is scheduled with uniform distribution between 1 and 2. If c happens before a in s_1, then, with probability 0.2, c is simply rescheduled; and with probability 0.8, the process goes to s_3, and a response basic event d is scheduled with uniform distribution between 0 and 1. Also, when b happens in s_2, with a small probability, the process goes to s_3. Note that, the probability that a and c happen simultaneously is zero. Two basic events can happen at the same time only if they are fixed-delay. □

The process is not Markov because non-exponential distributions can be associated to the delays, and is not semi-Markov because not even at the transition times the Markov property holds, since not all events in $act(s)$ are rescheduled when the process enters s. We see that $\mathcal{M}$ indeed defines a generalized semi-Markov process over S because we can retrieve the Markov property by allowing a generalized notion of state. A *generalized state* of $\mathcal{M}$ has the form $\langle s, v \rangle$, where $s \in S$ and v is a mapping from $act(s)$ to the non-positive reals, that is, a particular reading for each active clock, which we call a *clock interpretation for* E. If we now let the state space be the space of all generalized states, then we have a Markov process Y. For $t \in \mathbb{R}$ (the set of non-negative real numbers), we write $v + t$ for the clock interpretation which maps every clock x to $v(x) + t$.

3 Timed Automata and the Verification Problem

Timed automata were proposed in [3] as a formalism for the verification of real-time systems. Informally, a timed automaton is an ω-automaton together with a finite set of clock variables whose values increase with the real time. Every transition of the automaton has a constraint on the values of the clocks and can be taken only if the clocks satisfy the constraint. In addition, a transition may reset some of the clocks. As we will see, timed automata accept timed words instead of ω-words.

Definition 3 A *timed word* ρ over a finite alphabet Σ is a pair (σ, τ) where

- $\sigma = \sigma_1\sigma_2\cdots$ is a sequence of symbols $\sigma_i \in \Sigma$ (an ω-word over Σ);
- $\tau = \tau_1\tau_2\cdots$ is an strictly increasing sequence of time values $\tau_i \in \mathbb{R}$, $\tau_i > 0$, satisfying the *progress* property: for every $t \in \mathbb{R}$, there is some $i \geq 1$ such that $\tau_i > t$. Given $c \in \mathbb{R}$, we write $c \cdot \tau$ for the sequence obtained from τ by multiplying every τ_i by c.

In a timed word (σ, τ), the time value τ_i is interpreted as the occurrence time of the event σ_i. For example, the following is the initial prefix of a timed word ρ_s over $\{a, b\}$: $(a, 3.1) \to (b, 6) \to (a, 6.2) \to (b, 7.5) \to \cdots$.

Definition 4 Given a finite set X of clock variables, a *clock constraint* δ over X is defined inductively by $\delta := x \leq c \,|\, c \leq x \,|\, \neg\delta \,|\, \delta_1 \wedge \delta_2$, where $x \in X$ and $c \in \mathbb{Q}$. The set of all clock constraints over X is denoted by $\Phi(X)$.

Definition 5 A *timed (Büchi) automaton* $\mathcal{A}$ is a tuple $\langle \Sigma, Q, Q_0, X, T, F \rangle$, where

- Σ is a finite alphabet;
- Q is a finite set of locations;
- $Q_0 \subseteq Q$ is a set of start locations;
- X is a finite set of clocks;
- $T \subseteq Q \times Q \times \Sigma \times 2^X \times \Phi(X)$ is a set of transitions. For a transition $\langle q, q', a, \lambda, \delta \rangle$ from location q to location q', on symbol a, δ gives the constraint to be satisfied and λ the set of clocks to be reset;
- $F \subseteq Q$ is a set of accepting locations.

The operation of the automaton $\mathcal{A}$ is obtained by defining runs of $\mathcal{A}$ over timed words. For this, let a *clock interpretation for* X be a mapping from X to $\mathbb{R}$, that is, a particular reading of the clocks in X. A *generalized location* of $\mathcal{A}$ has the form $\langle q, \nu \rangle$, where $q \in Q$ and ν is a clock interpretation for X. For $t \in \mathbb{R}$, we write $\nu + t$ and $t \cdot \nu$, respectively, for the clock interpretation which maps every clock x to $\nu(x) + t$ and to $t \cdot \nu(x)$. A clock interpretation ν for X satisfies a clock constraint δ over X iff δ evaluates to true when each clock x is replaced by $\nu(x)$. Given a transition $\langle q, q', a, \lambda, \delta \rangle$ and a generalized location $\langle q, \nu \rangle$, the transition is said to be *enabled* if ν satisfies δ.

Definition 6 A *run* $r = (\overline{q}, \overline{\nu})$, of a timed automaton $\mathcal{A}$ over a timed word $\rho = (\sigma, \tau)$ is an infinite sequence of generalized locations of the form

$$r : \langle q_0, \nu_0 \rangle \xrightarrow[\tau_1]{\sigma_1} \langle q_1, \nu_1 \rangle \xrightarrow[\tau_2]{\sigma_2} \langle q_2, \nu_2 \rangle \xrightarrow[\tau_3]{\sigma_3} \cdots,$$

satisfying:

- *Initiation*: $q_0 \in Q_0$, and $\nu_0(x) = 0$ for all $x \in X$;

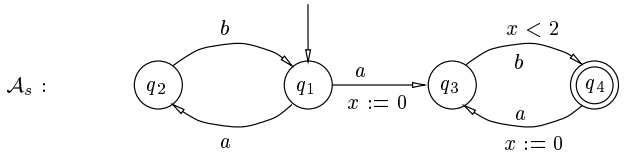

Figure 2: A nondeterministic timed automaton $\mathcal{A}_s$

- *Consecution*: for all $i \geq 1$, there exists $\langle q_{i-1}, q_i, \sigma_i, \lambda_i, \delta_i \rangle \in T$ such that $(\nu_{i-1}+\tau_i-\tau_{i-1})$ satisfies δ_i, and $\nu_i(x) = 0$ if $x \in \lambda_i$ and $\nu_i(x) = \nu_{i-1} + \tau_i - \tau_{i-1}$ otherwise ($\tau_0 = 0$, by definition).

Given a run $r = (\overline{q}, \overline{\nu})$ over a timed word $\rho = (\sigma, \tau)$, let $\mathit{inf}(r)$ be the set of locations such that $q \in \mathit{inf}(r)$ iff $q = q_i$ for infinitely many $i \geq 1$ in r. The run r over ρ is called an *accepting run* iff $\mathit{inf}(r) \cap F \neq \emptyset$. Finally, the language accepted by $\mathcal{A}$ is $L(\mathcal{A}) = \{(\sigma, \tau) \mid \mathcal{A} \text{ has an accepting run over } (\sigma, \tau)\}$.

Example 2 The automaton $\mathcal{A}_s$ in Fig. 2 expresses the convergent bounded response property mentioned in the introduction. The single clock x is reset only in the transitions from q_1 to q_3 and from q_4 to q_3. The transition from q_3 to q_4 is the only one with a clock constraint different than *true*. Note that the automaton is nondeterministic, since the transitions from q_1 to q_3 and from q_1 to q_2, which are on the same symbol, can be simultaneously enabled.

The language accepted by $\mathcal{A}_s$ is $L(\mathcal{A}_s) = \{((ab)^\omega, \tau) \mid$ there is i such that for all $j \geq i$, $(\tau_{2j} < \tau_{2j-1} + 2)\}$. The following are the two possible initial prefixes of runs of $\mathcal{A}_s$ over ρ_s ($x(d)$ means $x = d$):

$$\langle q_1, x(0)\rangle \xrightarrow[3.1]{a} \langle q_2, x(3.1)\rangle \xrightarrow[6]{b} \langle q_1, x(6)\rangle \xrightarrow[6.2]{a} \langle q_2, x(6.2)\rangle \xrightarrow[7.5]{b} \langle q_1, x(7.5)\rangle \cdots ,$$

$$\langle q_1, x(0)\rangle \xrightarrow[3.1]{a} \langle q_2, x(3.1)\rangle \xrightarrow[6]{b} \langle q_1, x(6)\rangle \xrightarrow[6.2]{a} \langle q_3, x(0)\rangle \xrightarrow[7.5]{b} \langle q_4, x(1.3)\rangle \cdots .$$

□

We note that timed Büchi automata have the same expressiveness as timed Muller automata [3]. In [2], Muller acceptance condition was used instead, since deterministic timed Muller automata are strictly more expressive than deterministic timed Büchi automata.

3.1 The Verification Problem

An instance of the verification problem is composed by a process $\mathcal{M}$ describing the probabilistic system, and a timed automaton $\mathcal{A}$ giving the specification. A particular behavior of the process $\mathcal{M}$ is described as a timed word over Δ, which gives the sequence of events and their occurrence times. From now on, we assume that the set of events Δ of $\mathcal{M}$ equals the alphabet Σ of the timed automaton $\mathcal{A}$. Let Σ^t denote the set of all timed words over Σ.

The process $\mathcal{M}$ induces a probability measure[1] over Σ^t. We say that $\mathcal{M}$ satisfies $\mathcal{A}$ iff $L(\mathcal{A})$ has measure 1 in the probability measure induced by $\mathcal{M}$. In other words, the verification problem is to decide, whether or not, the probability that $\mathcal{M}$ exhibits a behavior accepted by $\mathcal{A}$ equals 1.

As an example, if we add dummy transitions for the basic events c and d (of the process $\mathcal{M}_s$) in the automaton $\mathcal{A}_s$ ($\langle q, q, \ell, \emptyset, \mathit{true}\rangle$, for all $q \in Q$, $\ell \in \{c, d\}$), the process $\mathcal{M}_s$ satisfies $\mathcal{A}_s$.

3.2 Restriction to Integer Constants

Let c be a positive real constant. Given a timed automaton $\mathcal{A}$, let $\mathcal{A}^c$ denote the automaton obtained by multiplying all the constants appearing in all the clock constraints of $\mathcal{A}$ by c. There exists a one-to-one correspondence between the runs of $\mathcal{A}$ and the runs of $\mathcal{A}^c$: given a timed word $\rho = (\sigma, \tau)$, $(\overline{q}, \overline{\nu})$ is a run of $\mathcal{A}$ over ρ iff $(\overline{q}, c \cdot \overline{\nu})$ is a run of $\mathcal{A}^c$ over $(\sigma, c \cdot \tau)$ [3].

Given a process $\mathcal{M}$, let $\mathcal{M}^c$ denote the process obtained by replacing the function *dist* by dist^c, such that dist^c replaces the functions l by l^c, u by u^c, and, for every variable delay and exponential basic event x, the function f_x by f_x^c; where l^c, u^c and f_x^c are defined, respectively, as: $l_x^c = c \cdot l_x$, $u_x^c = c \cdot u_x$ and $f_x^c(t) = f_x(t/c)/c$. These definitions guarantee that, given any interval $[t_1, t_2]$, $\int_{t_1}^{t_2} f_x(t)\,dt = \int_{c \cdot t_1}^{c \cdot t_2} f_x^c(t)\,dt$. In particular, $\int_{l_x}^{u_x} f_x(t)\,dt = \int_{l_x^c}^{u_x^c} f_x^c(t)\,dt = 1$. Thus, we can obtain the probability measure induced by $\mathcal{M}^c$ on Σ^t from the one induced by $\mathcal{M}$ simply mapping each timed word (σ, τ) to $(\sigma, c \cdot \tau)$. The following lemma holds.

Lemma 1 *$\mathcal{M}$ satisfies $\mathcal{A}$ iff $\mathcal{M}^c$ satisfies $\mathcal{A}^c$.* □

The invariance under multiplication by a constant shows that we can restrict our attention to instances with integer constants. Given a process $\mathcal{M}$ and an automaton $\mathcal{A}$, we can choose c to be the least common multiple of the denominators of all constants appearing in the clock constraints of $\mathcal{A}$ and all constants in the ranges of the functions l and u of $\mathcal{M}$, and then, use $\mathcal{M}^c$ and $\mathcal{A}^c$ instead, which have only integer constants.

4 The Algorithm

Let $\mathcal{M}$ be a real-time probabilistic process, and let $\mathcal{A}$ be a timed automaton, both with integer constants. We assume that from every location $q \in Q$ of $\mathcal{A}$ and every clock interpretation ν, there is an edge $\langle q, q', a, \lambda, \delta\rangle$, for every $a \in \Sigma$, for some q', λ and δ, such that ν satisfies δ. This can be achieved by a simple transformation: add a dummy location q_d to Q and, then, transitions $\langle q, q_d, a, X, \mathit{true}\rangle$, for all $a \in \Sigma$, and all $q \in Q$. The set F remains unchanged. Clearly, $L(\mathcal{A})$ is not altered.

In order to cope with the nondeterminism in $\mathcal{A}$, we use the standard idea of a subset construction, applied on the generalized locations of $\mathcal{A}$. As in [2], given Y, we define an extended Markov process Y^* that simulates the runs of $\mathcal{A}$ over the behavior of $\mathcal{M}$. The

[1] A formal definition of a similar probability measure can be found in [13]

process Y^* records in its states, in addition to the state of Y, a finite set of generalized locations of $\mathcal{A}$. Let A be the set of all generalized locations of $\mathcal{A}$ and let 2^A_{fin} denote the set of all finite subsets of A. Then, a state of Y^* has the form $\langle s, v, p\rangle$, where $\langle s, v\rangle$ is a state of Y and $p \in 2^A_{fin}$. The states of Y^* are updated as follows:

- *Initial states*: all states of the form $\langle s, v, p\rangle$, where $f_0(s) > 0$ and for all $x \in act(s)$, $v(x)$ is according to the probability distribution $dist(x)$ and $p = \{\langle q, \nu\rangle \mid q \in Q_0$ and $\nu(x) = 0$ for all $x \in X\}$;
- *Time-passage states*: if $Y^*_t = \langle s, v, p\rangle$, such that for some clock $x \in act(s)$, $v(x) = -\varepsilon < 0$ and $v(y) \leq v(x)$ for all $y \in act(s)$, then, for all $0 < \varepsilon' \leq \varepsilon$, $Y^*_{t+\varepsilon'} = \langle s, v+\varepsilon', p'\rangle$, where $p' = \{\langle q, \nu'\rangle \mid$ there is $\langle q, \nu\rangle \in p$ such that $\nu' = \nu + \varepsilon'\}$;
- *Transition states*: consider Y^* in a state $\langle s, v, p\rangle$, such that $v(x) = 0$ for some $x \in act(s)$. Let $a = \{x \mid v(x) = 0\}$. The process moves to some state $\langle s', v', p'\rangle$, where
  - $next(s, a, s') > 0$;
  - For all $x \in old(s, a, s')$, $v'(x) = v(x)$. For all $x \in new(s, a, s')$, $v'(x)$ is according to $dist(x)$;
  - $p' = \{\langle q', \nu'\rangle \mid$ there is $\langle q, \nu\rangle \in p$ and there is $\langle q, q', a, \lambda, \delta\rangle \in T$ such that ν satisfies δ and $\nu'(x) = 0$ if $x \in \lambda$ and $\nu'(x) = \nu(x)$ otherwise$\}$.

The Markov process Y^* induces the same probability measure as Y over Σ^t. A particular behavior ρ of Y^* is a function from $\mathbb{R}$ to the state space of Y^*, defined according to the above rules. In the next section we will introduce the idea of a *generic clock* and define the traditional (integral parts/order of fractional parts) equivalence relation on the states of Y^*. Later on we will see that for most practical processes $\mathcal{M}$, the equivalence relation allows us to analyze the process Y^* with the very same method of [2] and decide the verification problem for any timed automaton $\mathcal{A}$.

4.1 Generic Clocks

Consider Y^* in a time-passage state $\langle s, v, p\rangle$, where $|p| = n$. In the next state $\langle s, v + \varepsilon', p'\rangle$, $|p'|$ is still n. If Y^* is in a transition state, where $|p| = n$, then, in the next state, $|p'|$ can be as high as kn, where k is the degree of nondeterminism of $\mathcal{A}$, that is, the maximum number of transitions, on the same symbol, that can be simultaneously enabled. But note, however, that the number of distinct values in the ranges of the functions ν in all generalized locations of p' is, at most, one more than the number of distinct values in p. This possible additional distinct value is zero, and it corresponds to all the clocks that were reset by the transition.

Let $c_{\mathcal{A}}$ be the greatest constant appearing in the clock constraints of $\mathcal{A}$. Let $\uparrow$ be a special symbol representing any value in the interval $(c_{\mathcal{A}}, \infty)$. By definition, $\uparrow > c_{\mathcal{A}}$. Given a set p of generalized locations of $\mathcal{A}$, we define the set $R_p \subset [0, c_{\mathcal{A}}] \cup \{\uparrow\}$ as follows: let $R'_p = \{d \mid$ there is $\langle q, \nu\rangle \in p$, such that $d = \nu(x) \leq c_{\mathcal{A}}$ for some $x \in X\}$. If there is $\langle q, \nu\rangle \in p$, such that $\nu(x) > c_{\mathcal{A}}$ for some $x \in X$, then $R_p = R'_p \cup \{\uparrow\}$, otherwise $R_p = R'_p$.

In order to formalize the equivalence relation on the states of Y^*, we think of each value in R_p as being represented by a *generic* clock. We create a set of generic clock variables, $C_p = \{c_1, c_2, \ldots, c_{|R_p|}\}$ for p. We define also the bijective function $\eta_p : C_p \to R_p$ as the unique function such that $\eta_p(c_1) < \eta_p(c_2) < \cdots < \eta_p(c_{|R_p|})$. Given two sets p and p' of generalized locations of $\mathcal{A}$, if $|R_p| = |R_{p'}|$, then we interpret the two sets C_p and $C_{p'}$ as being the same set of generic clock variables.

The function η_p induces, for each $\langle q, \nu\rangle \in p$, a function $\mu : X \to C_p$ that associates to each clock $x \in X$ the generic clock which holds "the value $\nu(x)$", that is, $\mu(x) = \eta_p^{-1}(\nu(x))$ if $\nu(x) \leq c_{\mathcal{A}}$ and $\mu(x) = \eta_p^{-1}(\uparrow)$ otherwise. The generalized location $\langle q, \nu\rangle$ is, then, represented by a pair $\langle q, \mu\rangle$, which we call a *position* of $\mathcal{A}$. Note that two different generalized locations can be associated to the same position. This is because all values greater than $c_{\mathcal{A}}$ are mapped to $\uparrow$. For a set of generalized locations p, we define the set of positions of $\mathcal{A}$ as $P_p = \{\langle q, \mu\rangle \mid \langle q, \mu\rangle \text{ represents some } \langle q, \nu\rangle \in p\}$.

Example 3 Suppose $c_{\mathcal{A}} = 4$ for a timed automaton $\mathcal{A}$ with $Q = \{q_1, q_2, \ldots, q_{10}\}$, and consider the following set of generalized locations of $\mathcal{A}$:

$$p = \left\{ \langle q_2, \begin{bmatrix} x_1(3.1) \\ x_2(4.1) \\ x_3(2.9) \end{bmatrix}\rangle, \langle q_3, \begin{bmatrix} x_1(2.9) \\ x_2(5) \\ x_3(1.3) \end{bmatrix}\rangle, \langle q_8, \begin{bmatrix} x_1(2.9) \\ x_2(\pi) \\ x_3(1.3) \end{bmatrix}\rangle, \langle q_6, \begin{bmatrix} x_1(4.8) \\ x_2(2.2) \\ x_3(2.9) \end{bmatrix}\rangle \right\} ,$$

Then $C_p = \{c_1, c_2, \ldots, c_6\}$,

$$\eta_p = \begin{bmatrix} c_6(\uparrow) \\ c_5(\pi) \\ c_4(3.1) \\ c_3(2.9) \\ c_2(2.2) \\ c_1(1.3) \end{bmatrix} \text{ and } P_p = \left\{ \langle q_2, \begin{bmatrix} x_1(c_4) \\ x_2(c_6) \\ x_3(c_3) \end{bmatrix}\rangle, \langle q_3, \begin{bmatrix} x_1(c_3) \\ x_2(c_6) \\ x_3(c_1) \end{bmatrix}\rangle, \langle q_8, \begin{bmatrix} x_1(c_3) \\ x_2(c_5) \\ x_3(c_1) \end{bmatrix}\rangle, \langle q_6, \begin{bmatrix} x_1(c_6) \\ x_2(c_2) \\ x_3(c_3) \end{bmatrix}\rangle \right\} .$$

□

Given a state $\langle s, \upsilon, p\rangle$ of Y^*, we define the *clock vector* $\xi_{\langle s,\upsilon,p\rangle}$ as the mapping $\xi_{\langle s,\upsilon,p\rangle} : act(s) \cup C_p \to \mathbb{R} \cup \{\uparrow\}$, induced by υ and η_p, that is, for each $x \in act(s)$, $\xi_{\langle s,\upsilon,p\rangle}(x) = \upsilon(x)$, and for each $c \in C_p$, $\xi_{\langle s,\upsilon,p\rangle}(c) = \eta_p(c)$. A generic clock $c \in C_p$ is said to be *irrelevant* to $\xi_{\langle s,\upsilon,p\rangle}$ if $\xi_{\langle s,\upsilon,p\rangle}(c) = \uparrow$, and *relevant* otherwise. Note that at most one generic clock is irrelevant to a clock vector. The set of relevant generic clocks is denoted by C_p^{rel}. We are now ready to define the traditional [3, 2, 8] equivalence relation $\sim$ over the set of all clock vectors.

Given a number $t \in \mathbb{R}$, $\lfloor t \rfloor$ denote the greatest integer smaller than or equal to t, and $\text{fr}(t) = t - \lfloor t \rfloor$ is the fractional part of t. Define $\xi \sim \xi'$ iff:

- *domain*: the domains of ξ and ξ' are equal. Let *act* denote the set of active basic events and let C denote the set of generic clocks in ξ and ξ';
- *irrelevant clock*: for each $x \in C$, $\xi(x) = \uparrow$ iff $\xi'(x) = \uparrow$;
- *exponential basic events*: for each $x \in act \cap E_{\text{e}}$, $\xi(x) = 0$ iff $\xi'(x) = 0$;

- *relevant clocks and bounded basic events*: Let $E^* = (act \cap E_b) \cup C^{\mathrm{rel}}$. (1) for each $x \in E^*$, $\lfloor \xi(x) \rfloor = \lfloor \xi'(x) \rfloor$ and $\mathrm{fr}(\xi(x)) = 0$ iff $\mathrm{fr}(\xi'(x)) = 0$; (2) and for each pair x and y in E^*, $\mathrm{fr}(\xi(x)) < \mathrm{fr}(\xi(y))$ iff $\mathrm{fr}(\xi'(x)) < \mathrm{fr}(\xi'(y))$ and $\mathrm{fr}(\xi(x)) = \mathrm{fr}(\xi(y))$ iff $\mathrm{fr}(\xi'(x)) = \mathrm{fr}(\xi'(y))$.

Finally, we can define the equivalence relation over the set of all states of Y^* as an extension of the relation $\sim$ for clock vectors.

Definition 7 Consider a process $\mathcal{M}$, a timed automaton $\mathcal{A}$, and the associated Markov process Y^*. We define $\langle s, v, p \rangle \sim^* \langle s', v', p' \rangle$ iff:

- $s = s'$;
- $\xi_{\langle s,v,p \rangle} \sim \xi_{\langle s',v',p' \rangle}$;
- $P_p = P_{p'}$.

The relation $\sim^*$ preserves enough information to decide which event $\mathcal{M}$ will deliver next and, when it occurs, which transitions of $\mathcal{A}$ will be enabled. In order to achieve this, the relation $\sim$ records, for each generic clock (condition (1)), the interval from $I^{\mathrm{c}} = \{[0,0], (0,1), [1,1], (1,2), \ldots, [c_{\mathcal{A}}, c_{\mathcal{A}}], (\uparrow)\}$ where the clock is contained. Note that any two clocks in the same interval satisfy the same set of clock constraints. For the basic events, the intervals are from $I^{\mathrm{b}} = \{[-u_g, -u_g], \ldots, (-2,-1), [-1,-1], (-1,0), [0,0]\}$, where u_g is the greatest value in the range of the function u. To correctly update this information, the relation also records (condition (2)) the order of the fractional parts. Nothing is needed, however, for clocks whose values are greater than $c_{\mathcal{A}}$, since all of them satisfy the same set of clock constraints. For exponential events, we need only the intervals $I^{\mathrm{e}} = \{(-\infty, 0), [0,0]\}$, because of the memoryless property of these distributions. Thus, an equivalence class can be specified by a tuple of the form $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, where:

- $s \in S$;
- $C = \{c_1, c_2, \ldots, c_K\}$ is a set of generic clocks;
- P is a set of positions of $\mathcal{A}$;
- $int_{\mathrm{c}} : C \to I^{\mathrm{c}}$ gives the interval of each $x \in C$;
- $int_{\mathrm{b}} : act_{\mathrm{b}}(s) \to I^{\mathrm{b}}$ gives the interval of each $x \in act_{\mathrm{b}}(s)$;
- $int_{\mathrm{e}} : act_{\mathrm{e}}(s) \to I^{\mathrm{e}}$ gives the interval of each $x \in act_{\mathrm{e}}(s)$;
- Fr is an ordering for the fractional parts of the clocks in $E^* = act_{\mathrm{b}}(s) \cup C^{\mathrm{rel}}$. It has the form[2] $Fr : 0 \diamond \mathrm{fr}(x_1) \diamond \mathrm{fr}(x_2) \diamond \cdots \diamond \mathrm{fr}(x_{|E^*|}) < 1$, where $\diamond \in \{=, <\}$ and $x_i \in E^*$. Also, int_{c} and Fr are such that $c_1 < c_2 < \cdots < c_K$.

[2]This notation comes from [8].

Note 1 Throughout the paper we use the following schematic representation for equivalence classes in $\sim^*$:

$$[s,\ \begin{bmatrix} c_4(\uparrow) \\ c(-1,0) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c_2(1) \\ b(-\infty,0) \end{bmatrix},P]\ .$$

Each clock x is annotated with its interval (we write $x(c)$ as a shortcut for $x[c,c]$). Between brackets we construct a stack of the relevant generic clocks and the bounded basic events. The stack gives the order of the fractional parts, such that the clock on the top has the greatest fractional part. If there is an irrelevant clock then we put it above the stack. If there are exponential basic events we put them under the stack.

Remark 1 The number of equivalence classes of $\sim^*$ is *not* finite, since there is no bound on the number of generic clocks in the set C. However, it is important to note that the number of equivalence classes with at most K generic clocks *is* finite. Let V_K denote the set of all equivalence classes with exactly K generic clocks, that is, $|C| = K$. The following bound holds (compare to [2, 3]):

$$|V_K| < \underbrace{2^{|Q||C|^{|X|}}}_{(1)} \cdot \underbrace{(2c_{\mathcal{A}}+2)^{|C|}}_{(2)} \cdot \sum_{s\in S} \left[\underbrace{|act_{\mathrm{b}}(s) \cup C|!}_{(3)} \cdot \underbrace{(2u_g+1)^{|act_{\mathrm{b}}(s)|}}_{(4)} \cdot \underbrace{2^{|act_{\mathrm{e}}(s)|}}_{(5)} \right],$$

where the factor (i) refers to the number of possible:

(1) sets of positions of $\mathcal{A}$;

(2) combinations of intervals for relevant generic clocks;

(3) orders of fractional parts;

(4) combinations of intervals for bounded events;

(5) combinations of intervals for exponential events.

4.2 The Graph G

Let V denote the set of all equivalence classes of $\sim^*$. If we project the states of the process Y^* onto their equivalence classes, we get a projected process Y^p over V. By projecting the states of a particular behavior ρ of Y^*, we can obtain an ω-word ρ_p over V, giving the sequence of equivalence classes visited by the behavior ρ.

The process Y^p is not Markov. However, given that Y^p is in a state $v \in V$, we can effectively compute the set of states $\{v' \in V \mid$ there is a positive probability that the next state will be $v'\}$. Thus, we can define an oriented graph G whose vertex set is a subset of V and such that there is an edge vv' iff there is a positive probability that the next state will be v', given that the present state is v. The set V_0 of initial vertices of G is composed by all the vertices of the form $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, where:

- $f_0(s) > 0$;
- $C = \{c_1\}$ and $int_{\mathrm{c}}(c_1) = (0)$;
- $P = \{\langle q, \mu\rangle \mid q \in Q_0\}$, where μ is defined as: for each $x \in X$, $\mu(x) = c_1$;
- for every $x \in act_{\mathrm{e}}(s)$, $int_{\mathrm{e}}(x) = (-\infty, 0)$;
- for every fixed-delay basic event $x \in act_{\mathrm{b}}(s)$, $int_{\mathrm{b}}(x) = (-l_x)$;
- for every variable-delay basic event $x \in act_{\mathrm{b}}(s)$, $int_{\mathrm{b}}(x) = (-c, -(c-1))$, where $c \in \mathbb{N}$, $l_x + 1 \leq c \leq u_x$;
- for any two variable-delay basic events x and y in $act_{\mathrm{b}}(s)$, $\mathrm{fr}(x) \neq \mathrm{fr}(y)$.

A state $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$ of Y^p is called *transient* iff for some $x \in act(s) \cup C^{\mathrm{rel}}$, we have $\mathrm{fr}(x) = 0$. The graph G is defined inductively, from the initial vertices, by the following rules that define the edge relation of G.

Rules. The edges are grouped in five different types:

1. Consider Y^p in a transient vertex $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, such that for each $x \in act_i(s)$, $int_i(x) \neq (0)$, $i \in \{\mathrm{b}, \mathrm{e}\}$, and for each $x \in C$, $int_{\mathrm{c}}(x) \neq (c_{\mathcal{A}})$. Then, with probability 1, the next vertex will be $[s, C, P, int'_{\mathrm{c}}, int'_{\mathrm{b}}, int_{\mathrm{e}}, Fr']$, where:
   - for each $x \in C$, $int'_{\mathrm{c}}(x) = (c, c+1)$ if $int_{\mathrm{c}}(x) = (c)$ for some $c \in \mathbb{N}$, and $int'_{\mathrm{c}}(x) = int_{\mathrm{c}}(x)$ otherwise;
   - for each $x \in act_{\mathrm{b}}(s)$, $int'_{\mathrm{b}}(x) = (-c, -(c-1))$ if $int_{\mathrm{b}}(x) = (-c)$ for some $c \in \mathbb{N}$, and $int'_{\mathrm{b}}(x) = int_{\mathrm{b}}(x)$ otherwise;
   - in this case, $Fr : 0 = \mathrm{fr}(x_1) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$. Then, $Fr' : 0 < \mathrm{fr}(x_1) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$.

   For example:

$$[s, \begin{bmatrix} b(-1,0) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c_2(1) \end{bmatrix}, P] \xrightarrow{\text{type 1}} [s, \begin{bmatrix} b(-1,0) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c_2(1,2) \end{bmatrix}, P] \ .$$

2. Consider Y^p in a transient vertex $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, such that for each $x \in act_i(s)$, $int_i(x) \neq (0)$, $i \in \{\mathrm{b}, \mathrm{e}\}$, and there is $y \in C$, such that $int_{\mathrm{c}}(y) = (c_{\mathcal{A}})$. Note that there can be only one such y. Then, there are two cases:
   (a) there is no irrelevant generic clock. Then $y{=}c_{|C|}$, and, with probability 1, the next vertex will be $[s, C, P, int'_{\mathrm{c}}, int'_{\mathrm{b}}, int_{\mathrm{e}}, Fr']$, where:
      - $int'_{\mathrm{c}}(y) = (\uparrow)$, and $int'_{\mathrm{c}}(x) = int_{\mathrm{c}}(x)$ for each $x \in C$, $x \neq y$;

- for each $x \in act_{\rm b}(s)$, $int'_{\rm b}(x) = (-c, -(c-1))$ if $int_{\rm b}(x) = (-c)$ for some $c \in \mathbb{N}$, and $int'_{\rm b}(x) = int_{\rm b}(x)$ otherwise;
- in this case, $Fr : 0 = \mathrm{fr}(y) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$. Then, $Fr' : 0 < \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$.

For example, suppose $c_{\mathcal{A}} = 4$:

$$[s, \begin{bmatrix} b(-1,0) \\ c_2(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c_3(4), c(-1) \end{bmatrix}, P] \quad \xrightarrow{\text{type 2(a)}} \quad [s, \begin{matrix} c_3(\uparrow) \\ \begin{bmatrix} b(-1,0) \\ c_2(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c(-1,0) \end{bmatrix} \end{matrix}, P] \ .$$

(b) there is an irrelevant generic clock. Then $y{=}c_{|C|-1}$, and, with probability 1, the next vertex will be $[s, C', P', int'_{\rm c}, int'_{\rm b}, int_{\rm e}, Fr']$, where:

- $C' = \{c_1, c_2, \ldots, c_{|C|-1}\}$;
- $int'_{\rm c}(y) = (\uparrow)$, and $int'_{\rm c}(x) = int_{\rm c}(x)$ for each $x \in C'$, $x \neq y$;
- for each $x \in act_{\rm b}(s)$, $int'_{\rm b}(x) = (-c, -(c-1))$ if $int_{\rm b}(x) = (-c)$ for some $c \in \mathbb{N}$, and $int'_{\rm b}(x) = int_{\rm b}(x)$ otherwise;
- in this case, $|P'| \leq |P|$. $P' = \{\langle q, \mu' \rangle \mid$ there is $\langle q, \mu \rangle \in P$ such that for every $x \in X$, either $\mu'(x) = \mu(x)$ or $\mu'(x) = c_{|C'|}$ and $\mu(x) = c_{|C|}\}$;
- in this case, $Fr : 0 = \mathrm{fr}(y) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$. Then, $Fr' : 0 < \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_{|E^*|} \mathrm{fr}(x_{|E^*|}) < 1$.

For example, let $c_{\mathcal{A}} = 4$ and $X = \{x, y\}$:

$$[s, \begin{matrix} c_4(\uparrow) \\ \begin{bmatrix} b(-1,0) \\ c_2(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ c_3(4) \end{bmatrix} \end{matrix}, \left\{ \langle q_2, \begin{bmatrix} x(c_1) \\ y(c_2) \end{bmatrix} \rangle, \langle q_4, \begin{bmatrix} x(c_4) \\ y(c_1) \end{bmatrix} \rangle, \langle q_4, \begin{bmatrix} x(c_3) \\ y(c_1) \end{bmatrix} \rangle \right\}]$$

$$\Big\downarrow \text{type 2(b)}$$

$$[s, \begin{matrix} c_3(\uparrow) \\ \begin{bmatrix} b(-1,0) \\ c_2(2,3) \\ a(-2,-1) \\ c_1(0,1) \end{bmatrix} \end{matrix}, \left\{ \langle q_2, \begin{bmatrix} x(c_1) \\ y(c_2) \end{bmatrix} \rangle, \langle q_4, \begin{bmatrix} x(c_3) \\ y(c_1) \end{bmatrix} \rangle \right\}] \ .$$

3. Consider Y^p in a nontransient vertex $[s, C, P, int_{\rm c}, int_{\rm b}, int_{\rm e}, Fr]$, such that $|E^*| \geq 1$. Then, with positive probability, the next vertex will be $[s, C, P, int'_{\rm c}, int'_{\rm b}, int_{\rm e}, Fr']$, where:

- for each $y \in C$, $int'_{\rm c}(y) = (c+1)$ if $int_{\rm c}(y) = (c, c+1)$, for some $c \in \mathbb{N}$ and for every $x \in E^*$, $\mathrm{fr}(x) \leq \mathrm{fr}(y)$. Otherwise, $int'_{\rm c}(y) = int_{\rm c}(y)$;
- for each $y \in act_{\rm b}(s)$, $int'_{\rm b}(y) = (-(c-1))$ if $int_{\rm b}(y) = (-c, -(c-1))$, for some $c \in \mathbb{N}$ and for every $x \in E^*$, $\mathrm{fr}(x) \leq \mathrm{fr}(y)$. Otherwise, $int'_{\rm b}(y) = int_{\rm b}(y)$;

- in this case, $Fr : 0 < \mathrm{fr}(x_1) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_M \mathrm{fr}(x_M) < \mathrm{fr}(y_1) = \mathrm{fr}(y_2) = \cdots = \mathrm{fr}(y_N) < 1$. Then, $Fr' : 0 = \mathrm{fr}(y_1) = \mathrm{fr}(y_2) = \cdots = \mathrm{fr}(y_N) < \mathrm{fr}(x_1) \diamond_2 \mathrm{fr}(x_2) \diamond_3 \cdots \diamond_M \mathrm{fr}(x_M) < 1$.

For example:

$$[s,\ \begin{array}{c} c_4(\uparrow) \\ \left[\begin{array}{l} b(-1,0), c_2(0,1) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \end{array}\right] \\ c(-\infty,0) \end{array},P] \quad \xrightarrow{\text{type 3}} \quad [s,\ \begin{array}{c} c_4(\uparrow) \\ \left[\begin{array}{l} c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \\ b(0), c_2(1) \end{array}\right] \\ c(-\infty,0) \end{array},P] \ .$$

4. Consider Y^p in a nontransient vertex $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$. Then, for each $y \in act_{\mathrm{e}}(s)$, there is a positive probability that the next vertex will be $[s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int'_{\mathrm{e}}, Fr]$, where:

   - $int'_{\mathrm{e}}(y) = (0)$, and $int'_{\mathrm{e}}(x) = (-\infty, 0)$ for every $x \in act_{\mathrm{e}}(s)$, $x \neq y$;

For example:

$$[s,\ \begin{array}{c} c_4(\uparrow) \\ \left[\begin{array}{l} b(-1,0), c_2(0,1) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \end{array}\right] \\ c(-\infty,0) \end{array},P] \quad \xrightarrow{\text{type 4}} \quad [s,\ \begin{array}{c} c_4(\uparrow) \\ \left[\begin{array}{l} b(-1,0), c_2(0,1) \\ c_3(2,3) \\ a(-2,-1) \\ c_1(0,1) \end{array}\right] \\ c(0) \end{array},P] \ .$$

5. Consider Y^p in a transient vertex $v = [s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$ such that $|a| > 0$, $a = \{x \,|\, x \in act_i(s) \text{ and } int_i(x) = (0),\ i \in \{\mathrm{b}, \mathrm{e}\}\}$. Then, a state transition occurs is $\mathcal{M}$. We need a few definitions.

   Given a position $\langle q, \mu\rangle \in P$, let the clock interpretation ν_μ over X be defined as: $\nu_\mu(x) = (l + u)/2$, where $l = u = c$ if $int_{\mathrm{c}}(\mu(x)) = (c)$, and $l = c$ and $u = c + 1$ if $int_{\mathrm{c}}(\mu(x)) = (c, c + 1)$, for some $c \in \mathbb{N}$. We say that μ satisfies a clock constraint δ iff ν_μ satisfies δ.

   We say that v is *reseting* iff there is $\langle q, \mu\rangle \in P$ and $\langle q, q', a, \lambda, \delta\rangle \in T$ such that μ satisfies δ and $|\lambda| > 0$. This means that a new generic clock will be needed to represent the value 0 in the next state. Define $D \subseteq \{1, 2, \ldots, |C|\}$ as $D = \{i \,|$ there is $\langle q, \mu\rangle \in P$ such that, for some $x \in X$, $\mu(x) = c_i$ and there is $\langle q, q', a, \lambda, \delta\rangle \in T$ such that $x \notin \lambda$ and μ satisfies $\delta\}$, that is, D contains the indices of all generic clocks whose values will still represent some clock in the next state. Let d be the unique function $d : \{1, 2, \ldots, |D|\} \to D$ satisfying $d(1) < d(2) < \cdots < d(|D|)$.

   Then, there is a positive probability that the next vertex will be any vertex $[s', C', P', int'_{\mathrm{c}}, int'_{\mathrm{b}}, int'_{\mathrm{e}}, Fr']$, where:

   - $next(s, a, s') > 0$;

- $C' = \{c_1, c_2, \ldots, c_N\}$, where $N = |D| + 1$ if v is reseting and $N = |D|$ otherwise;
- $P' = \{\langle q', \mu' \rangle \mid$ there is $\langle q, \mu \rangle \in P$ and $\langle q, q', a, \lambda, \delta \rangle \in T$ such that μ satisfies δ, and for each $x \in \lambda$, $\mu'(x) = c_1$, and for each $x \notin \lambda$, $\mu'(x) = c_i$, and $\mu(x) = c_{d(i-1)}$ if v is reseting, and $\mu(x) = c_{d(i)}$ otherwise$\}$;
- if v is reseting then $int'_{\mathrm{c}}(c_1) = (0)$, and for $2 \leq i \leq N$, $int'_{\mathrm{c}}(c_i) = int_{\mathrm{c}}(c_{d(i-1)})$. Otherwise, for $1 \leq i \leq N$, $int'_{\mathrm{c}}(c_i) = int_{\mathrm{c}}(c_{d(i)})$;
- for each $x \in old(s, a, s') \cap act_i(s')$, $int'_i(x) = int_i(x)$, $i \in \{\mathrm{b}, \mathrm{e}\}$;
- for each $x \in new(s, a, s')$:
  - if $x \in act_{\mathrm{e}}(s')$, $int'_{\mathrm{e}}(x) = (-\infty, 0)$;
  - if $x \in act_{\mathrm{b}}(s')$ and is fixed-delay, $int'_{\mathrm{b}}(x) = (-l_x)$;
  - if $x \in act_{\mathrm{b}}(s')$ and is variable-delay, $int'_{\mathrm{b}}(x) = (-c, -(c-1))$, where $c \in \mathbb{N}$, $l_x + 1 \leq c \leq u_x$;
- Let $C^s = C' \setminus \{c_1\}$ if v is reseting, and $C^s = C'$ otherwise. Let $O = (old(s, a, s') \cap act_{\mathrm{b}}(s')) \cup C^s$. Given a clock $x \in O$, if v is reseting, then, $o(x) = c_{d(i-1)}$ if $x = c_i$ and $o(x) = x$ if $x \notin C^s$. Otherwise, $o(x) = c_{d(i)}$ if $x = c_i$ and $o(x) = x$ if $x \notin C^s$. Fr' is such that:
  - for any two clocks x and y in O, $\mathrm{fr}(x) < \mathrm{fr}(y)$ in Fr' iff $\mathrm{fr}(o(x)) < \mathrm{fr}(o(y))$ in Fr and $\mathrm{fr}(x) = \mathrm{fr}(y)$ in Fr' iff $\mathrm{fr}(o(x)) = \mathrm{fr}(o(y))$ in Fr.
  - for any two variable-delay x and y in $new(s, a, s')$, $\mathrm{fr}(x) \neq \mathrm{fr}(y)$, and for any variable-delay $x \in new(s, a, s')$ and $y \in O$, $\mathrm{fr}(x) \neq \mathrm{fr}(y)$.

As an example, consider Y^p in the following state and suppose that there are two edges $\langle q_1, q_2, a, \{x\}, \delta_1 \rangle$ and $\langle q_1, q_3, a, \emptyset, \delta_2 \rangle$ enabled, that $next(s, a, s') = 1$, and $act(s') = \{b, c\}$, $c \in act_{\mathrm{b}}(s')$ and $l_c = 1$ and $u_c = 2$:

$$[s, \begin{bmatrix} c_2(2,3) \\ b(-3,-2) \\ c_1(0,1) \\ a(0) \end{bmatrix}, \left\{ \langle q_1, \begin{bmatrix} x(c_1) \\ y(c_2) \end{bmatrix} \rangle \right\}] \ .$$

Then Y^p moves, with positive probability, to each state of the form

$$[s', \begin{bmatrix} \longrightarrow \\ c_3(2,3) \\ \longrightarrow \\ b(-3,-2) \\ \longrightarrow \\ c_2(0,1) \\ \longrightarrow \\ c_1(0) \end{bmatrix}, \left\{ \langle q_2, \begin{bmatrix} x(c_1) \\ y(c_3) \end{bmatrix} \rangle, \langle q_3, \begin{bmatrix} x(c_2) \\ y(c_3) \end{bmatrix} \rangle \right\}] \ ,$$

where the arrows indicate the four possible places for $c(-2, -1)$. □

Note that the definition of G implies that, if there is no edge between v and v', then, the probability that the next state will be v' given that the present state is v, is zero. The argument that, indeed, for each edge of G, this probability is positive, also comes from the definition for edges of the types 1, 2 and 5. For the types 3 and 4, the argument is straightforward and is given in the proof of the following lemma in [2].

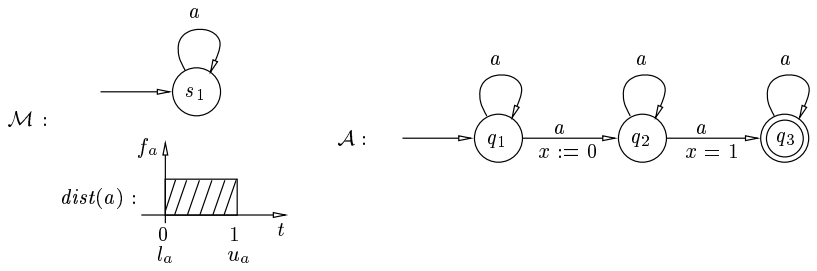

Figure 3: An instance for which G is infinite

Lemma 2 (Lemma 2 in [2]) *For each vertex $v \in V$, the set of behaviors ρ of Y^* such that v appears on ρ_p, has positive probability iff there is a finite path in G from some vertex $v_0 \in V_0$ to v.* □

To solve the verification problem, nevertheless, we need to analyze the *ergodic* behavior of Y^*. A strongly connected component $W \subseteq V$ of G is called a *bottom strongly connected component* (b.s.c. component) of G iff for every $v \in W$, if there is an edge vv', then $v' \in W$. The method in [2] relates the ergodic behavior of Y^* to the b.s.c. components of G. Every b.s.c. component must satisfy a certain condition, and the algorithm, in order to verify this condition, must visit every vertex of the component. Thus, if G is not finite, the method cannot be applied. We now give an example of an instance for which G is infinite. In the next section, we show that for most interesting practical instances, G *is* finite. Then, in Sect. 4.4, we will resume the presentation of the algorithm, for these instances for which we can give a guarantee of the finiteness of G.

Example 4 Consider the instance in Fig. 3. The process $\mathcal{M}$ has a single basic event a with uniform distribution between 0 and 1. Every particular behavior of $\mathcal{M}$ is characterized by a timed word $\rho = (\sigma, \tau)$ such that $\sigma = a^\omega$ and $\tau_i - \tau_{i+1} \leq 1$, $i \geq 0$. The specification $\mathcal{A}$ is the traditional [3] example of a noncomplementable timed automaton. It accepts the language $L(\mathcal{A}) = \{(a^\omega, \tau) \mid$ there are $i \geq 1$ and $j > i$ such that $(\tau_j = \tau_i + 1)\}$. Note that, since $\mathcal{A}$ cannot be complemented and the class of languages defined by deterministic timed automata is closed by complementation, there is no deterministic timed automaton accepting $L(\mathcal{A})$. Clearly, $\mathcal{M}$ does not satisfy $\mathcal{A}$. The probability that a particular behavior of $\mathcal{M}$ satisfies the condition $(\tau_j = \tau_i + 1)$ is zero.

If we try to construct the graph G, we encounter the following infinite sequence of vertices beginning in the unique initial vertex. The label $\delta + a$ in the arrows represents a time passage plus the state transition in $\mathcal{M}$, that is, actually, between any two consecutive vertices in this sequence there are two other vertices corresponding to the passage of time.

$$[s_1,\ \begin{bmatrix} a(-1,0) \\ c_1(0) \end{bmatrix},\ \{\langle q_1, x(c_1)\rangle\}]$$
$$\downarrow \delta + a$$
$$[s_1,\ \begin{bmatrix} a(-1,0) \\ c_2(0,1) \\ c_1(0) \end{bmatrix},\ \{\langle q_1, x(c_2)\rangle, \langle q_2, x(c_1)\rangle\}]$$
$$\downarrow \delta + a$$
$$[s_1,\ \begin{bmatrix} a(-1,0) \\ c_3(0,1) \\ c_2(0,1) \\ c_1(0) \end{bmatrix},\ \{\langle q_1, x(c_3)\rangle, \langle q_2, x(c_1)\rangle, \langle q_2, x(c_2)\rangle\}]$$
$$\downarrow \delta + a$$
$$[s_1,\ \begin{bmatrix} a(-1,0) \\ c_4(0,1) \\ c_3(0,1) \\ c_2(0,1) \\ c_1(0) \end{bmatrix},\ \{\langle q_1, x(c_4)\rangle, \langle q_2, x(c_1)\rangle, \langle q_2, x(c_2)\rangle, \langle q_2, x(c_3)\rangle\}]$$
$$\vdots$$

The number of relevant generic clocks needed to keep track of all the possible runs of $\mathcal{A}$ over this behavior of $\mathcal{M}$ is not bounded, because the clock x is reset, in each run, at different times, so that we need to add one generic clock for each time. In addition, the process can schedule the basic event a arbitrarily close to zero in the interval $(-1, 0)$, such that no generic clock becomes irrelevant. But note that this infinite sequence only exists in G because the process can generate arbitrarily many events in a finite interval of time. We show, in the next section, that this is a necessary condition for G to be infinite.

This problem is somehow expected, since the fact that the automaton can cycle arbitrarily many times in q_2, without reseting x, before passing to q_3, is precisely one of the reasons why this automaton is not complementable [3]. □

4.3 Instances with Finite G

One may argue that a system model which can generate arbitrarily many discrete events in a finite interval of time is not realistic, since this is not physically realizable. In addition, allowing this property in the model may drastically affect the complexity of the decision problems—it is an essential property in the proofs of the undecidability results about nondeterministic timed automata [3, 11]. Indeed, nondeterministic timed automata has been recognized as *too* expressive—to the point where the important (for automatic verification) problem of language inclusion is undecidable [4]. There has been much discussion about the adequacy of the various models (see [12] for a start). We will not pursue this discussion here. We show that our use of generic clocks to "encode" the nondeterminism of the automaton suffices to decide the verification problem for processes $\mathcal{M}$ satisfying the following (*K-transitions*) assumption: there is a constant $K \in \mathbb{N}$ such that at most K state transitions can happen, in $\mathcal{M}$, in a time interval of unit length. Thus, one can use the full

expressiveness power of nondeterministic timed automata to specify properties as long as the process satisfies this assumption.

This assumption is commonly adopted in an important application of real-time verification techniques: the analysis of digital circuits [14, 6, 15]. For instance, in [6, 15] a timed automaton model for asynchronous circuits is proposed. Every logical gate is followed by a delay element constraining, between lower and upper bounds, the rising and falling (which are the discrete events) of the digital signals. The lower bound is a positive constant, such that any cycle in the model takes at least k time units to complete, for some positive constant k, and so the assumption is met. It is worth noting that the K-transitions assumption does not affect the dense time assumption. In fact, one of the results in [6] is that cyclic circuits in that model, which meets the K-transitions assumption, in general, do not admit discretization.

Lemma 3 *G is infinite iff, for every $k \in \mathbb{N}$, $k > 0$, a vertex with exactly k generic clocks is reachable from some initial vertex.*

Proof 1 The "if" part is trivial. For the other direction, recall that every initial vertex has exactly one generic clock, and that the number of vertices with at most k generic clocks is finite. Thus, if G is infinite, for every $l \in \mathbb{N}$, a vertex with more than l generic clock is reachable. But, by definition, if there is an edge vv', and v has m generic clocks, then v' has, at most, $m + 1$ generic clocks. This means that a vertex with n generic clocks is reachable only if a vertex with $n - 1$ is also reachable. □

Recall the rule for edges of type 5 in G. Given the transient vertex $v = [s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, we defined a set D containing the indices of the generic clocks in C which represent at least one clock $x \in X$, in some position of $\mathcal{A}$, which is not reset by the transition. We say that a generic clock c_i *survives* the transition if $i \in D$. Its value will be represented, in the next state, by a generic clock c_j, such that j is, at most, $i + 1$ ($j = i + 1$ exactly in the case when all generic clocks c_h, $h \leq i$, survive the transition and, in addition, v is reseting). We say that c_j *survived* the transition when it represents some clock $x \in X$, in some position of $\mathcal{A}$, which was not reset by the last transition. Extending this discussion for more than one transition, we see that the index of a generic clock gives a lower bound on the number of transitions that it has survived, and the following lemma holds.

Lemma 4 *Consider a generic clock c_i in any vertex v of G. Then, in any finite initial path of G ending in v, the generic clock c_i survives for, at least, the last $i - 1$ transitions.* □

This lemma has also the following interpretation: if a vertex v has k generic clocks, then any finite behavior of $\mathcal{M}$ ending in v has, at least, one run of $\mathcal{A}$ over it, such that some clock $x \in X$ is not reset in this run, for, at least, the last $k - 1$ transitions.

Theorem 1 *If $\mathcal{M}$ satisfies the K-transitions assumption, then G is finite.*

Proof 2 Since $\mathcal{M}$ satisfies the K-transitions assumption, we can find a constant N such that any sequence of N transitions in $\mathcal{M}$ takes *more* than $c_{\mathcal{A}}$ time units. Assume that a

vertex v with $N+2$ generic clocks is reachable in G from some initial vertex. Then, c_{N+1} is relevant in v, by definition. But this is a contradiction, since Lemma 4 implies that c_{N+1} survives at least the last N transitions, so it cannot be relevant. Since no vertex with $N+2$ generic clocks can be reached, G is finite by Lemma 3. □

Remark 2 The K-transitions assumption is a sufficient but not necessary condition for the finiteness of G. The process may have a cycle where the lower bound of all basic events is zero, but this cycle may not "synchronize" (as it did in Example 4) with a similar cycle in $\mathcal{A}$, where some clock is never reset. Also, clearly, if $\mathcal{A}$ happens to be such that every clock is reset in every cycle, then G is finite for any $\mathcal{M}$.

4.4 Ergodic Components and Recurrent Vertices

We now finish the presentation of the algorithm. This final part is a combination of the method in [2] and the results of [7, Section 4]. From now on, assume that G is finite and let V_{G} be its set of vertices (V_{G} is a finite subset of V). Every b.s.c. component of G is classified as either *accepting* or *rejecting*, so that $\mathcal{M}$ satisfies $\mathcal{A}$ iff all b.s.c. components of G are accepting.

Given a particular behavior ρ of Y^*, let $inf(\rho_p) \subseteq V_{\mathrm{G}}$ be the set of vertices of G appearing infinitely often in ρ_p. The main result in [2], which applies directly to our construction, is the following:

Lemma 5 (Lemma 3 in [2]) *Let $W \subseteq V_{\mathrm{G}}$ be a set of vertices of G. Assume that the process Y^* starts in some state $\langle s, v, p\rangle \in v$, for some $v \in W$. The set of behaviors ρ of Y^* such that $inf(\rho_p) = W$ has positive measure iff W is a b.s.c. component of G.* □

We discuss the difficulty behind this lemma. In the discrete time model [7], where the system is given as a Markov chain and the specification as an ω-automaton, the analogous lemma follows trivially from this property: if a vertex v is visited infinitely often by the process, then, each vertex v', such that there is an edge vv', is also visited infinitely often. This is because, if there is an edge from v to v', then the probability that the next vertex will be v' given that the present vertex is v is not only positive, but is a positive constant. In our case, this does not hold. Recall the example for edges of type 5:

$$v = [s,\ \begin{bmatrix} c_2(2,3) \\ b(-3,-2) \\ c_1(0,1) \\ a(0) \end{bmatrix},\ \left\{\langle q_1, \begin{bmatrix} x(c_1) \\ y(c_2) \end{bmatrix}\rangle\right\}]$$

$$\downarrow \text{ type 5}$$

$$v' = [s,\ \begin{bmatrix} c_3(2,3) \\ c(-2,-1) \\ b(-3,-2) \\ c_2(0,1) \\ c_1(0) \end{bmatrix},\ \left\{\langle q_2, \begin{bmatrix} x(c_1) \\ y(c_3) \end{bmatrix}\rangle, \langle q_3, \begin{bmatrix} x(c_2) \\ y(c_3) \end{bmatrix}\rangle\right\}] \ .$$

If the process Y^p is in v, then the probability that the next vertex will be v' is positive, but its exact value depends on the difference $[\mathrm{fr}(c_2) - \mathrm{fr}(b)]$ in v. There is no a priori

guarantee that, if the process visits v infinitely often, then it will visit v' also infinitely often, because the difference $[\mathrm{fr}(c_2) - \mathrm{fr}(b)]$ could converge to zero.

Given a vertex $v = [s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, consider the set $E^* = act_{\mathrm{b}}(s) \cup C^{\mathrm{rel}}$. We say that a visit of Y^p to v is *δ-separated* iff, for every pair x and y in E^*, if $\mathrm{fr}(x) \neq \mathrm{fr}(y)$ then $|\mathrm{fr}(x) - \mathrm{fr}(y)| \geq \delta$. We will need the following two facts, which are demonstrated in the course of the proof of this Lemma 5:

Fact 1 *Consider a vertex v' such that there is the edge vv'. There exists a positive ε and a positive δ' such that, for any δ-separated visit of Y^p to v, the probability that the next state will be a δ'-separated visit to v', is bounded from below by ε.*

Fact 2 *Given any particular behavior ρ of Y^*, if $inf(\rho_p) = W$, then for each $v \in W$, there is a positive δ_v, such that ρ makes infinitely many δ_v-separated visits to v.*

Lemma 5 gives the relationship between the ergodic behavior of Y^* and the b.s.c. components of G. In order to classify these components, we need some definitions. A vertex $v \in V$, $v = [s, C, P, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, is called a *unit* vertex, iff $|P| = 1$. A unit vertex $[s, C, \{\langle q, \mu\rangle\}, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$ is said to be *accepting* iff $q \in F$. We say that a unit vertex v is *contained* in a vertex $v' \in V$ iff:

- Given two states of Y^*, $\langle s, \upsilon, p\rangle \in v$ and $\langle s', \upsilon', p'\rangle \in v'$, there is $\langle q', \nu'\rangle \in p'$ such that, $\langle s, \upsilon, p\rangle \sim^* \langle s', \upsilon', \{\langle q', \nu'\rangle\}\rangle$. That is, informally, there is a position in v' such that we can obtain v from v' by deleting all the other positions (and modifying the set of generic clocks accordingly).

For example, v is contained in v':

$$v = [s,\ \begin{bmatrix} c_2(2,3) \\ c(-2,-1) \\ b(-3,-2) \\ c_1(0,1) \end{bmatrix},\ \left\{\langle q_3, \begin{bmatrix} x(c_1) \\ y(c_2) \end{bmatrix}\rangle\right\}]$$

$$\cap$$

$$v' = [s,\ \begin{bmatrix} c_3(2,3) \\ c(-2,-1) \\ b(-3,-2) \\ c_2(0,1) \\ c_1(0) \end{bmatrix},\ \left\{\langle q_2, \begin{bmatrix} x(c_1) \\ y(c_3) \end{bmatrix}\rangle, \langle q_3, \begin{bmatrix} x(c_2) \\ y(c_3) \end{bmatrix}\rangle\right\}] \ .$$

Given a unit vertex $v \in V$, let G_v denote the graph obtained inductively from v by applying the same rules for the edge relation of G. It is important to note that, if v is contained in some vertex of G, then G_v is guaranteed to be finite.

Definition 8 A unit vertex $v \in V$ is called *recurrent* iff, the graph G_v has a b.s.c. component M such that there is a vertex $v' \in M$, and v is contained in v'. We associate with a given recurrent unit vertex v, a finite path γ_v of G_v, going from v to some vertex in M (any such path).

Finally, a b.s.c. component W of G is accepting iff some vertex $v' \in W$ contains an accepting recurrent unit vertex v. Otherwise, it is rejecting.

Lemma 6 (Analogous to Lemma 4.1.2. in [7]) *Let $W \subseteq V_{\mathrm{G}}$ be an accepting b.s.c. component of G. Any behavior ρ of Y^* such that $inf(\rho_p) = W$ is accepted by $\mathcal{A}$ with probability one.*

Proof 3 (Outline) Given a particular behavior ρ' of Y^*, consider the sequence $\rho'_p = v_1 v_2 \cdots$ of vertices of G traversed by ρ'. We construct a run of $\mathcal{A}$ for ρ' by fixing finite segments of the run, repeating the following procedure: choose an index i and a position $\langle q, \mu \rangle$ in v_i. By the definition of Y^*, any finite behavior of Y^* traversing $v_1 v_2 \cdots v_i$ has an initial finite run r of $\mathcal{A}$ ending in a generalized location $\langle q, \nu \rangle$, for some ν. We fix this finite run by deleting all the other positions in v_i, obtaining a unit vertex u, and letting the process Y^* continues from the vertex u. That is, the remaining sequence $v_i v_{i+1} v_{i+2} \cdots$ is *projected* on the graph G_u, yielding a new sequence $\rho''_p = u u_2 u_3 \cdots$ of vertices of G_u.

As W is accepting, there is a vertex $v' \in W$ containing an accepting recurrent unit vertex $v = [s, C, \{\langle q, \mu \rangle\}, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$, such that G_v has a b.s.c. component M, and there is a vertex $v'' \in M$, and v is contained in v''. The idea is to construct an accepting run for the behavior ρ repeating the location q infinitely many times.

Consider the finite path $\gamma_v = v v_2 v_3 \cdots v_{|\gamma_v|}$ in G_v (from Definition 8). If Y^* is in a δ-separated visit to v, then, by applying Fact 1 repeatedly, the probability that the process Y^* follows the path γ_v is bounded from below by a positive constant ε. Also, as v is contained in v', a δ-separated visit to v' can be viewed as a δ-separated visit to v.

Now, consider the sequence $\rho_p = v_1 v_2 \cdots$. By Fact 2, for infinitely many $i \geq 1$, the visit of the behavior ρ to v_i is a $\delta_{v'}$-separated visit to v'. Since these are also $\delta_{v'}$-separated visits to v, then, with probability one, there is a finite index n, such that $v_n = v'$, and, if we project the sequence $v_n v_{n+1} \cdots$ on G_v, the process Y^* follows the path γ_v and is absorbed by the b.s.c. component M. Thus, we fix the first finite segment of the accepting run by choosing v_n and deleting positions in v' so as to obtain v.

The remaining sequence $v_n v_{n+1} \cdots$ is projected on G_v, and the new sequence is $\gamma_v \cdots$, that is, $v v_2 v_3 \cdots v_{|\gamma_v|} \cdots$. But now, we can apply Lemma 5 and Fact 2 on the graph G_v (with v'' playing the role of v') and repeat the procedure infinitely many times. Since every time the procedure is repeated, we can fix the finite segment of the run with probability one, the whole run is constructed with probability one. □

Lemma 7 (Analogous to Lemma 4.1.3. in [7]) *Let $W \subseteq V_{\mathrm{G}}$ be a rejecting b.s.c. component of G. Any behavior ρ of Y^* such that $inf(\rho_p) = W$ is rejected by $\mathcal{A}$ with probability one.*

Proof 4 (Outline) If no vertex $v' \in W$ contains an accepting unit vertex v, then by Lemma 5, clearly, ρ is rejected by $\mathcal{A}$ with probability one. Suppose there is an accepting unit vertex $v = [s, C, \{\langle q, \mu \rangle\}, int_{\mathrm{c}}, int_{\mathrm{b}}, int_{\mathrm{e}}, Fr]$ contained in some vertex v' of W. The idea is to show that, for every such v, the probability that $\mathcal{A}$ has a run over ρ repeating q infinitely often is zero.

By applying Lemma 5 to the graph G_v, v is contained in the vertex v_i of any sequence $vv_1v_2\cdots$ of G_v, for only finitely many $i \geq 1$, since any such sequence is eventually absorbed by some b.s.c. component of G_v and v is *not* recurrent. Hence, the probability that a behavior of Y^*, traversing the sequence $vv_1v_2\cdots$ in G_v, has a run of $\mathcal{A}$ repeating q infinitely often is zero.

Now, consider the sequence $\rho_p = v_1v_2\cdots$. By Lemma 5, for infinitely many $i \geq 1$, $v_i = v'$. But, for *any* such v_i, if we fix the first finite segment of the run by choosing v_i, then the probability that a run with this initial segment repeats q infinitely often is zero by the last paragraph. □

Putting everything together we have the following theorem, giving the algorithm:

Theorem 2 *Given a real-time probabilistic process $\mathcal{M}$ and a timed automaton $\mathcal{A}$, such that the graph G is finite, $\mathcal{M}$ satisfies $\mathcal{A}$ iff, for every b.s.c. component W of G, there is at least one vertex $v' \in W$, such that v' contains a recurrent accepting unit vertex v.* □

5 Conclusions

In this paper, we presented an extension to the method in [2], for the verification of deterministic timed automata specifications of real-time probabilistic processes, giving a partial answer to the question about the verification of nondeterministic timed automata specifications. Our construction gives a semi-decision procedure, which is guaranteed to finish, for instance, when there is a bound on the number of events that the process can generated in a finite interval of time. This assumption is frequently adopted in practical applications of verifications techniques [14, 6, 15]. Hence, the very expressive formalism of nondeterministic timed automata can be used to specify properties for these probabilistic processes.

The method *is* quite expensive, as can be seen in the bound on the number of equivalence classes with K generic clocks in Section 4.1. In particular, it is doubly exponential in the number of clocks of the automaton, and exponential in the number of locations of the automaton. Nevertheless, it has been shown, in similar contexts [9, 10], that heuristic methods and symbolic techniques can sometimes yield useful implementations for such expensive (or undecidable, for that matter) problems. Thus, one direction for future work is the development of such methods and techniques for the probabilistic verification problem.

Another direction is, of course, the theoretical question of the decidability of the general verification problem, which remains open. From this theoretical point of view, it is interesting to note that, given an arbitrary process $\mathcal{M}$ and an arbitrary automaton $\mathcal{A}$, such that the procedure does not finish, the present method can be used to decide the verification problem for a modified process $\mathcal{M}_\varepsilon$, which can be arbitrarily "close" to $\mathcal{M}$. First note that the exact form of the probability density functions of the basic events is not relevant, since we are doing qualitative probabilistic verification. Given any rational constant ε, we obtain $\mathcal{M}_\varepsilon$ from $\mathcal{M}$ by:

- replacing the lower bound l_x of every bounded basic event x, such that $l_x = 0$, by ε;

- shifting every exponential distribution to the right by ε. The algorithm can be easily modified to deal with shifted exponential distributions.

Now, we use the transformation of section 3.2 to get only integer constants; and then use the instance composed of $\mathcal{M}_\varepsilon$ and $\mathcal{A}$, for which the procedure is guaranteed to finish. Moreover, the behavior of $\mathcal{M}_\varepsilon$ approaches the behavior of $\mathcal{M}$ when ε tends to zero; and yet, for any $\varepsilon > 0$, the problem is decidable.

References

[1] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for probabilistic real-time systems. In *Proceedings of the 18th ICALP*, number 510 in LNCS, pages 115–136. Springer-Verlag, 1991.

[2] R. Alur, C. Courcoubetis, and D. Dill. Verifying automata specifications of probabilistic real-time systems. In *Real-Time: Theory in Practice*, number 600 in LNCS, pages 28–44. Springer-Verlag, 1992.

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

[4] R. Alur, L. Fix, and T. Henzinger. Event-clock automata: A determinizable class of timed automata. In *CAV'94*, number 818 in LNCS, pages 1–13, 1994.

[5] R. Alur and T. Henzinger. Logics and models of real-time: A survey. In *Real-Time: Theory and Practice*, number 600 in LNCS, pages 74–106, 1992.

[6] E. Asarin, O. Maler, and A. Pnueli. On discretization of delays in timed automata and digital circuits. In *CONCUR'98*, number 1466 in LNCS, pages 470–484, 1998.

[7] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, 1995.

[8] A. Göllü, A. Puri, and P. Varaiya. Discretization of timed automata. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, 1994.

[9] N. Halbwachs, Y. Proy, and P. Raymond. Verification of linear hybrid systems by means of convex approximations. In *Int. Symposium on Static Analysis*, number 864 in LNCS, 1994.

[10] T. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: the next generation. In *Proceedings fo the 16th Annual IEEE Real-Time Systems Symposium*, pages 56–65, 1995.

[11] T. Henzinger and J. Raskin. Robust undecidability of timed and hybrid systems. Technical Report USB/CSD-99-1073, Berkeley Univ., October 1999.

[12] T. Henzinger, J. Raskin, and P. Schobbens. The regular real-time languages. In *ICALP'98*, number 1443 in LNCS, pages 580–591, 1998.

[13] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Verifying quantitative properties of continuous probabilistic real-time graphs. Technical Report CSR-98-11, Univ. of Birmingham, 1998.

[14] H. Lewis. Finite-state analysis of asynchronous circuits with bounded temporal uncertainty. Technical Report TR-15-89, Harvard Univ., 1989.

[15] O. Maler and A. Pnueli. Timing analysis of asynchronous circuits using timed automata. In *CHARME'95*, number 987 in LNCS, pages 189–205, 1995.

[16] G. Shedler. *Regeneration and Network of Queues.* Springer-Verlag, 1987.