CACHAN

P. Bouyer and A. Petit

Decomposition and composition of timed automata

Research Report LSV−99−1, Feb. 1999

Decomposition and composition of timed automata

Patricia Bouyer and Antoine Petit

LSV, CNRS UMR 8643, ENS de Cachan, 61 av. du Prés. Wilson,
F-94235 Cachan Cedex, France,
bouyer,petit@lsv.ens-cachan.fr

Abstract. We propose in this paper a decomposition theorem for the timed automata introduced by Alur and Dill [3,4]. To this purpose, we define a new simple and natural composition operation, indexed by the set of clocks to be reset, on timed automata generalizing the classical untimed concatenation.
Then we extend the famous Kleene's and Büchi's theorems on classical untimed automata by simply changing the basic objects to take time into account, keeping the union operation and replacing the concatenation, finite and infinite iterations by the new timed concatenations and their induced iterations.
Thus, and on the contrary of the interesting result of [8], we do not need neither intersection nor renaming. Therefore, and up to our knowledge, our result provides the simplest known algebraic characterization of recognizable timed languages.

1 Introduction

We are interested in this paper in the basic and natural model of so-called timed automata proposed by Alur and Dill [3,4] to modelize real-time systems. Since its introduction, this model has been largely studied under several aspects: determinization [5], minimization [1], power of clocks [2,15], of $\varepsilon-$transitions [12], extensions of the model [6,9] and logical characterization [18] have been considered in particular. Moreover this model has been successfully used for verification and specification of real-time systems [11,19,16].

But there is a lack of algebraic characterizations of timed languages or timed automata. On the contrary to the untimed case, there does not exist a notion of timed finite semi-groups and a notion of timed recognizing morphisms. Even the famous Kleene's theorem [17] and its extension of $\omega-$languages by Büchi [14] has only partial counterparts in the framework of timed languages. The most interesting result, due to Asarin, Caspi and Maler [8], proposes such a Kleene theorem for a slightly different notion of timed languages. But the main drawback of their characterization is that it uses in a crucial way the operations of intersection and renaming.

The aim of this paper is to propose, at least up to our knowledge for the first time, a simple Kleene-Büchi theorem for usual timed automata. Precisely, we define a natural composition operation on timed automata, indexed by a subset of clocks. Roughly, the composition of two timed automata is the classical concatenation in which the values of the specified clocks - and only these ones - are reset. We also introduce the finite and infinite iterations corresponding to this composition. Then we prove that any timed automaton is equivalent to a timed automaton built in a modular way from some basic automata using these operations and the classical union operation. Hence our result is very closed from Kleene's and Büchi's theorem, we have just changed the basic objects to take time into account, keep the union operation and replace the concatenation, finite and infinite iterations by timed concatenations based naturally on clocks and their induced iterations.

Our proof generalizes one of the proof of Kleene's theorem based on the resolution of equations on set of words, see e.g. [13]. The idea to solve equations on timed languages - in the simple case of automata with a unique clock - was already proposed by Asarin in [7] who gave a new proof of the main result of [8]. Here, we solve equations on multi-clock automata.

This paper is organized as follows. Section 2 recall the notion of timed words and timed automata. But note that, on the contrary with the classical way, we associate with a timed automaton not only one timed language but a family of languages indexed by the initial clock valuation. This idea simple, but powerful, will help deeply in the presentation of the operators. Then the technical notion of constrained generators is introduced. Section 3 is devoted to the introduction of the new composition operations and of the induced iterations. We also claim our main result at the end of this section. In section 4, we propose all the technical staff on constrained generators. We extend the operators to these objects and we show how to solve a simple equation on constrained generators. Then, Section 5 proposes a proof of our main theorem. Finally, a small conclusion proposes a promising direction of research.

For lack of space, this extended draft does not contain all the proofs.

2 Timed automata and constrained generators

In this section, we briefly recall the basic definitions and notations for timed words and finite automata with timing constraints, as well as classical properties of the corresponding families of timed languages. They come mostly from [3] and [4].

2.1 Timed words and clocks

Let Z be any set. We write Z^+ (respectively Z^ω) the set of non-empty finite (respectively infinite) sequences of elements in Z, with ε for the empty sequence, and $Z^\infty = Z^+ \cup Z^\omega$ the set of all sequences of elements in Z.

Throughout this paper, we consider a time domain $\mathbb{T}$ which is either the set of non-negative integers $\mathbb{N}$ or the set of non-negative rationals $\mathbb{Q}_+$. A time sequence over $\mathbb{T}$ is a finite or infinite divergent non decreasing sequence $\tau = (t_i)_{i \geq 1}$ of elements in $\mathbb{T}$.

The observable actions of the processes we consider are given by a finite set Σ. A timed word is an element of $(\Sigma \times \mathbb{T})^\infty$, $w = (a_i, t_i)_{i \geq 1}$, also written as a pair $w = (\sigma, \tau)$, where $\sigma = (a_i)_{i \geq 1}$ is a word in Σ^∞ and $\tau = (t_i)_{i \geq 1}$ is a timed sequence in $\mathbb{T}^\infty$ of same length than σ. For instance, $(a, 1.5)(b, 2)(a, \pi)(b, 3.9)(a, 3.9)$ is a finite timed word, while $((ab)^\omega, (i + 1/i)_{i \geq 1})$ is an infinite timed word.

The concatenation of a finite timed word $u = (a_i, t_i)_{1 \leq i \leq n}$ and of a finite or infinite timed word $v = (a'_i, t'_i)_{1 \leq i}$ is defined only if $t_n \leq t'_1$ and is equal to the timed word $u \cdot v = (a''_i, t''_i)_{1 \leq i}$ with $(a''_i, t''_i) = (a_i, t_i)$ if $1 \leq i \leq n$ and $(a''_i, t''_i) = (a'_{i-n}, t'_{i-n})$ if $n < i$.

Let X be a set of variables with values in $\mathbb{T}$, called clocks. Among X, we distinguish a special clock x_0 which will denote the absolute time.

The guards over X are the formulas defined by the grammar:

$$g ::= \top \mid x \sim c \mid x - y \sim c \mid g \vee g \mid g \wedge g \mid \neg g$$

where x and y are clocks, $c \in \mathbb{T}$ and $\sim$ is a binary operator in $\{<, =, >\}$. The set of all guards over X is denoted by Guards(X).

A clock valuation $\alpha : X \to \mathbb{T}$ is a mapping that assigns to each clock x a time value $\alpha(x)$. The set of all clock valuations is denoted $\mathbb{T}^X$. We write $\alpha \models g$ when the clock valuation α satisfies the clock constraint g. If Y a subset of X, the valuation $\alpha[Y := 0]$ is defined by, for each clock x in X, $(\alpha[Y := 0])(x) = 0$ if $x \in Y$, $\alpha(x)$ otherwise.

2.2 Timed automata and runs

A Büchi timed automaton over $\Sigma, \mathbb{T}$ and X is a tuple $\mathcal{A} = (Q, T, I, F, R)$, where Q is a finite set of states, $T \subseteq Q \times [\text{Guards}(X) \times \Sigma \times \mathcal{P}(X)] \times Q$ is the set of transitions, $I \subseteq Q$ is the subset of initial states, $F \subseteq Q$ is the subset of final states[1] and $R \subseteq Q$ is a subset of repeated states corresponding to a Büchi condition, as recalled below. Thus, a transition has the form (p, g, a, r, q), where g is a guard, a is an action in Σ and $r \subseteq X$ is the subset of clocks to be reset. We assume that the special clock x_0 is never reset.

The set of all timed automata over the set of actions Σ, the timed domain $\mathbb{T}$ and the set of clocks X is denoted by TA$(\Sigma, \mathbb{T}, X)$.

In order to define how a timed word is recognized by a timed automaton, we recall the notions of path and of timed run through a path. A path in $\mathcal{A}$ is a non empty finite or infinite sequence of consecutive transitions:

$$P = q_0 \xrightarrow{g_1, a_1, r_1} q_1 \xrightarrow{g_2, a_2, r_2} q_2 \cdots$$

where $(q_{i-1}, g_i, a_i, r_i, q_i) \in T$ for all $i > 0$. The path P is accepting, if it starts in an initial state and either it is finite ending in a final state or it is infinite and a

[1] For sake of simplicity, we will always assume that $I \cap F = \emptyset$

Büchi condition holds: $\mathit{inf}(P) \cap R$ is nonempty, where $\mathit{inf}(P)$ is the set of states which occur infinitely often in P.

A run of the automaton through the path P and from the initial clock valuation $\alpha \in \mathbb{T}^X$ is a triple $R = (\alpha, P, (t_i)_{i>0})$ where $(t_i)_{i>0}$ is a non-decreasing sequence of dates of same length than P; t_i is the time at which transition $q_{i-1} \xrightarrow{g_i,a_i,r_i} q_i$ has been executed. Intuitively, this transition can be performed, only if the guard g_i is satisfied by the current clock valuation and the clocks in r_i are reset at that time. More precisely, the value of clock x at time t is written $x(t)$, so that the clock valuation at time t can be written as the tuple $X(t) = (x(t))_{x \in X}$. Beginning with time $t_0 = \alpha(x_0)$ and clock values $x(t_0) = \alpha(x)$, for all x,[2] the clock valuation at time t is completely determined by the run: for each $i \geq 1$,

$$x(t_i) = \begin{cases} 0 & \text{if } x \in r_i \\ x(t_{i-1}) + t_i - t_{i-1} & \text{otherwise} \end{cases}$$

and $x(t) = x(t_{i-1}) + t - t_{i-1}$ for $t_{i-1} \leq t < t_i$. Moreover, we require that $X(t_{i-1}) + t_i - t_{i-1} \models g_i$ for all $i \geq 1$.

With every finite (infinite resp.) run we can associate in a natural way a finite (infinite resp.) timed word, designed as the label of R, by

$$\ell(R) = (a_1, t_1)(a_2, t_2) \cdots \in (\Sigma \times \mathbb{T})^\infty.$$

If R is finite of length n, we define also the clock valuation $\mathrm{Ends}(R) \in \mathbb{T}^X$ "at the end of R" in the following way:

$$\mathrm{Ends}(R)(x) = \begin{cases} 0 & \text{if } x \in r_n \\ x(t_n) & \text{otherwise} \end{cases}$$

Note that since we have assumed that the clock x_0 is never reset, it holds $\mathrm{Ends}(R)(x_0) = t_n$.

2.3 Timed automata and constrained generators

With our definition, we therefore associate not only one timed language with a timed automaton $\mathcal{A}$ but a family of timed languages indexed by the initial clock valuations. Precisely, for any clock valuation $\alpha \in \mathbb{T}^X$, we define

$$L_{\mathcal{A}}(\alpha) = \{\ell(R) \mid R = (\alpha, P, (t_i)_{i>0}) \text{ for some accepting path } P \text{ in } \mathcal{A}\}$$

For sake of simplicity, we denote $L_{\mathcal{A},+}(\alpha)$ for $L_{\mathcal{A}}(\alpha) \cap (\Sigma \times \mathbb{T})^+$ and $L_{\mathcal{A},\omega}(\alpha)$ for $L_{\mathcal{A}}(\alpha) \cap (\Sigma \times \mathbb{T})^\omega$.

[2] In most papers on timed automata, all the clocks, and in particular the absolute time, are initialized to 0. Therefore our definition generalizes the classical one.

Remark 1. It is not difficult to prove that for any timed automaton $\mathcal{A}$ and any clock valuation α, there exists a timed automaton $\mathcal{B}$ such that $L_{\mathcal{A}}(\alpha) = L_{\mathcal{B}}(\mathbf{0})$ - where $\mathbf{0}$ denotes the clock valuation defined by $\mathbf{0}(x) = 0$ for any clock x. Indeed, it suffices to add some $\varepsilon-$transitions before the initial states and to use the results of [12] to suppress these $\varepsilon-$transitions.

The fonction Ends on runs allows also to define the set of possible clock valuations after accepting a finite timed word. Precisely, for any timed word $u \in (\Sigma \times \mathbb{T})^+$ and any initial clock valuation $\alpha \in \mathbb{T}^X$, we set

$$\begin{aligned}\mathrm{Ends}_{\mathcal{A}}(\alpha, u) = \{\mathrm{Ends}(R) \mid R = (\alpha, P, (t_i)_{i>0}) \\ \text{for some accepting path } P \text{ in } \mathcal{A} \text{ such that } \ell(R) = u\}\end{aligned}$$

Note that if $u \notin L_{\mathcal{A},+}(\alpha)$, it holds $\mathrm{Ends}_{\mathcal{A}}(\alpha, u) = \emptyset$. This new function $\mathrm{Ends}_{\mathcal{A}}$ can be viewed as a constraint on the timed words of $L_{\mathcal{A},+}(\alpha)$.

We now define the notion of constrained generator as a generalization of these functions $L_{\mathcal{A}}$ and $\mathrm{Ends}_{\mathcal{A}}$. This notion will be a fundamental tool in the sequel. A constrained generator is a couple $(\mathcal{G}, \Lambda)$ such that

1. $\mathcal{G} : \mathbb{T}^X \longrightarrow \mathcal{P}((\Sigma \times \mathbb{T})^\infty)$
2. $\Lambda : \mathbb{T}^X \times (\Sigma \times \mathbb{T})^+ \longrightarrow \mathcal{P}(\mathbb{T}^X)$
3. $\forall u \in (\Sigma \times \mathbb{T})^+, u \in \mathcal{G}(\alpha) \Longleftrightarrow \Lambda(\alpha, u) \neq \emptyset$

From the results above, it is clear that for any timed automaton $\mathcal{A}$, the couple $(L_{\mathcal{A}}, \mathrm{Ends}_{\mathcal{A}})$ is a constrained generator. It will be denoted by $\hat{\mathcal{A}}$ and refered as the constrained generator associated with $\mathcal{A}$.

Two timed automata $\mathcal{A}$ and $\mathcal{B}$ are equivalent, we then denote $\mathcal{A} \equiv \mathcal{B}$ if their associated constrained generators $\hat{\mathcal{A}}$ and $\hat{\mathcal{B}}$ are equal. In this case, it holds in particular $L_{\mathcal{A}}(\mathbf{0}) = L_{\mathcal{B}}(\mathbf{0})$.

3 Composition of timed automata

3.1 Union

The union of two timed automaton $\mathcal{A} = (Q_1, T_1, I_1, F_1, R_1)$ and $\mathcal{B} = (Q_2, T_2, I_2, F_2, R_2)$ with $Q_1 \cap Q_2 = \emptyset$ is simply the timed automaton $\mathcal{A} + \mathcal{B} = (Q_1 \cup Q_2, T_1 \cup T_2, I_1 \cup I_2, F_1 \cup F_2, R_1 \cup R2)$.

3.2 Composition

We propose in this section a basic operation of composition of timed automata. The concatenation of two usual - i.e. untimed - automata is a classical operation widely used in the modelization of untimed systems. When we deal with timed systems, the main problem comes from the clocks. Assume that we want to concatenate two timed automata $\mathcal{A}$ and $\mathcal{B}$. Then, entering in $\mathcal{B}$, the two extreme possibilities are either to reset all the clocks of $\mathcal{A}$ or on the contrary to not reset any clock of $\mathcal{A}$. For instance, it is the way chosen by Asarin, Caspi and Maler in [8] in the particular framework of automata with a unique clock.

But on many real examples, these extreme possibilities are unsatisfactory. Assume for instance that we want to built a complex product under some global timing constraints and that the construction is made in several parts, each of them having its own timing constraints. Then it is much more natural to allow in the concatenation of timing automata modelizing the subsystems to reset some clocks and to not reset some other clocks.

Following this simple, but powerful, idea, we propose to define the concatenation of two timed automata depending on a fixed set of clocks. Assume that two timed automata $\mathcal{A}$ and $\mathcal{B}$ and a set of clocks C are given. Intuitively, the concatenation $\mathcal{A} \cdot_C \mathcal{B}$ is given by the following picture (y is a new clock neither used in $\mathcal{A}$ nor $\mathcal{B}$).

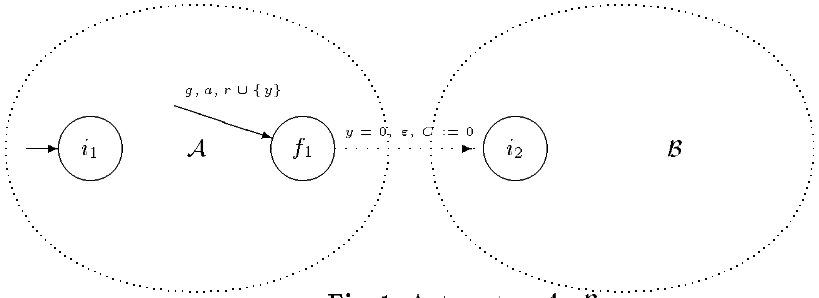

Fig. 1. Automaton $\mathcal{A} \cdot_C \mathcal{B}$

The use of $\varepsilon-$transitions is not allowed in our model. Nevertheless, it is easy to verify that $\mathcal{A} \cdot_C \mathcal{B}$ can be defined formally, without such transitions, in the following way. Let $\mathcal{A} = (Q_1, T_1, I_1, F_1, R_1)$ and $\mathcal{B} = (Q_2, T_2, I_2, F_2, R_2)$ with $Q_1 \cap Q_2 = \emptyset$. We consider a copy $\widetilde{F_1}$ of F_1, disjoint of $Q_1 \cup Q_2$. Then $\mathcal{A} \cdot_C \mathcal{B}$ is the timed automaton[3] $(Q_1 \cup Q_2 \cup \widetilde{F_1}, T, I_1, F_2, R_1 \cup R_2)$ with,

$$\begin{aligned} T = {} & T_1 \\ & \cup T_2 \\ & \cup \{(q_1, g_1, a_1, r_1 \cup C, \tilde{f_1}) \mid f_1 \in F_1, (q_1, g_1, a_1, r_1, f_1) \in T_1\} \\ & \cup \{(\tilde{f_1}, g_2, a_2, r_2, q_2) \mid \exists i_2 \in I_2, (i_2, g_2, a_2, r_2, q_2) \in T_2\} \end{aligned}$$

3.3 Iterations

We now derive in a natural way two iterations, a finite one and an infinite one, from this composition operator $\cdot_C$. Let $\mathcal{A} = (Q, T, I, F, R)$ be a timed automaton and let C be a subset of clocks. We consider a copy $\tilde{F}$ of F, disjoint from Q.

[3] Recall that by hypothesis, $I_1 \cap F_1 = I_2 \cap F_2 = \emptyset$

Then the timed automaton $\mathcal{A}^{\overset{+}{C}}$ is defined as follows, $\mathcal{A}^{\overset{+}{C}} = (Q \cup \tilde{F}, T', I, \tilde{F}, R)$ with

$$\begin{aligned} T' &= T \\ &\cup \{(q, g, a, r \cup C, \tilde{f}) \mid f \in F, (q, g, a, r, f) \in T\} \\ &\cup \{(\tilde{f}, g, a, r, q) \mid \exists i \in I, (i, g, a, r, q) \in T\} \end{aligned}$$

In a similar way, the infinite iteration of $\mathcal{A}$, denoted $\mathcal{A}^{\overset{\omega}{C}}$ is defined by $\mathcal{A}^{\overset{\omega}{C}} = (Q \cup \tilde{F}, T', I, \emptyset, R \cup \tilde{F})$ where T' is defined as just above.

3.4 Modular constructions of timed automata

For any guard $g \in \text{Guards}(X)$ and any action $a \in \Sigma$, we denote by $\mathcal{A}_{\langle \delta a \rangle_g}$ the simple timed automaton described in Figure 3.4. For any clock valuation $\alpha \in \mathbb{T}^X$ and any $t \in \mathbb{T}$, denote by $\alpha + t$ the clock valuation defined by, for any clock x, $(\alpha + t)(x) = \alpha(x) + t$. Then it is immediate to verify that $L_{\mathcal{A}_{\langle \delta a \rangle_g}}(\alpha)$ is equal to $\{(a, t) \mid \alpha + t \models g\}$. Moreover $\text{Ends}_{\mathcal{A}_{\langle \delta a \rangle_g}}(\alpha, a) = \{\alpha + t \mid \alpha + t \models g\}$.

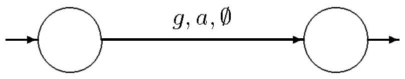

Fig. 2. Automate $\mathcal{A}_{\langle \delta a \rangle_g}$

From these basic automata and the composition operators define below, we define now a modular family of timed automata as follows.

Definition 2. Let $\text{Mod}(\Sigma, \mathbb{T}, X)$ be the smallest subset of timed automata of $\text{TA}(\Sigma, \mathbb{T}, X)$ generated by the following grammar:

$$\mathcal{A} ::= \mathcal{A}_{\langle \delta a \rangle_g} \mid \mathcal{A} + \mathcal{A} \mid \mathcal{A} \underset{C}{\cdot} \mathcal{A} \mid \mathcal{A}^{\overset{+}{C}} \mid \mathcal{A}^{\overset{\omega}{C}}$$

with a an action in Σ, $g \in \text{Guards}(X)$ a guard over X and $C \subseteq X$ a subset of clocks such that $x_0 \notin C$.

The main result of this paper is to prove that any timed automaton is equivalent to some timed automaton of $\text{Mod}(\Sigma, \mathbb{T}, X)$. This result can be seen as a Kleene-Büchi theorem for timed automata. Note that contrary to the results of the paper of Asarin, Caspi and Maler [8], we do not need neither intersection nor renaming.

Theorem 3. *Any timed automaton of $TA(\Sigma, \mathbb{T}, X)$ is equivalent to an automaton of $Mod(\Sigma, \mathbb{T}, X)$.*

This theorem will be proved by solving systems of equations on constrained generators. Since we can not assume, a priori, that the solution is always a constraint generator associated with some timed automaton, we need first some technical material on operations on constrained generators.

4 Composition of constrained generators

This section is devoted to the definitions of the composition of constrained generators. Even if these definitions are a bit technical, they are easy to understand keeping in mind that the goal is to obtain definitions such that the following proposition holds.

Proposition 4. *Let $\mathcal{A}$ and $\mathcal{B}$ be two timed automata in $TA(\Sigma, \mathbb{T}, X)$ and $C \subseteq X$ be a subset of clocks. Then it holds:*

1. $\widehat{\mathcal{A}+\mathcal{B}} = \hat{\mathcal{A}} + \hat{\mathcal{B}}$
2. $\widehat{\mathcal{A} ;_C \mathcal{B}} = \hat{\mathcal{A}} ;_C \hat{\mathcal{B}}$
3. $\widehat{\mathcal{A}^{\dot{+}}_{C}} = \hat{\mathcal{A}}^{\dot{+}}_{C}$
4. $\widehat{\mathcal{A}^{\dot{\omega}}_{C}} = \hat{\mathcal{A}}^{\dot{\omega}}_{C}$

Throughout this section, $(\mathcal{G}, \Lambda)$ and $(\mathcal{G}', \Lambda')$ are two constrained generators and $C \subseteq X$ is a subset of clocks.

4.1 Union of constrained generators

The union of $(\mathcal{G}, \Lambda)$ and $(\mathcal{G}', \Lambda')$, denoted by $(\mathcal{G}, \Lambda) + (\mathcal{G}', \Lambda')$ is naturally the constrained generator $(\mathcal{G}'', \Lambda'')$ where, for any clock valuation $\alpha \in \mathbb{T}^X$ and any finite timed word $w \in (\Sigma \times \mathbb{T})^+$, $\mathcal{G}''(\alpha) = \mathcal{G}(\alpha) \cup \mathcal{G}'(\alpha)$ and $\Lambda''(\alpha, u) = \Lambda(\alpha, u) \cup \Lambda'(\alpha, u)$.

4.2 Composition of constrained generators

We define the composition $(\mathcal{G}, \Lambda) ;_C (\mathcal{G}', \Lambda')$ as the constrained generator $(\mathcal{G}'', \Lambda'')$ where, for any clock valuation $\alpha \in \mathbb{T}^X$ and any finite timed word[4] $w \in (\Sigma \times \mathbb{T})^+$,

$$\mathcal{G}''(\alpha) = \{u \cdot v \mid u \in \mathcal{G}(\alpha) \cap (\Sigma \times \mathbb{T})^+ \text{ and } v \in \mathcal{G}'(\beta[C := 0]) \cap (\Sigma \times \mathbb{T})^\infty \text{ for some } \beta \in \Lambda(\alpha, u)\}$$

and

$$\Lambda''(\alpha, w) = \{\Lambda'(\beta[C := 0], v) \mid \text{ there exisst some } u \in \mathcal{G}(\alpha) \text{ with } w = u \cdot v \text{ and } \beta \in \Lambda(\alpha, u)\}$$

[4] Recall that the concatenation of two timed words is a partial operation which has been defined in Section 2.1

4.3 Iterations of a constrained generator

We define now inductively, the constrained generator, $(\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^i}$, for $i \geq 1$, by

$$(\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^1} = (\mathcal{G},\Lambda) \text{ and } (\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^{i+1}} = (\mathcal{G},\Lambda) \underset{C}{\cdot} (\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^i}$$

and we set $(\mathcal{G},\Lambda)^{\overset{+}{C}} = \Sigma_{i\geq 1}(\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^i}$.

Finally, let $(\mathcal{G},\Lambda)$ be a constrained generator and let $(\mathcal{G}_1,\Lambda_1) = (\mathcal{G},\Lambda)^{\overset{+}{C}}$. We define $(\mathcal{G},\Lambda)^{\overset{\omega}{C}} = (\mathcal{G}'',\Lambda'')$ by, for any clock valuation $\alpha \in \mathbb{T}^X$ and any finite timed word $w \in (\Sigma \times \mathbb{T})^+$,

$$\begin{aligned}
\mathcal{G}''(\alpha) = (\mathcal{G}_1(\alpha) \cap (\Sigma \times \mathbb{T})^\omega)\ \cup \\
\{u_0 \cdot u_1 \cdots \mid \text{ there exists a sequence } (\alpha_i)_{i\geq 0} \text{ of clock valuations} \\
\text{with } \alpha_0 = \alpha \text{ and for any } i \geq 1,\ u_{i+1} \in \mathcal{G}(\alpha_i) \cap (\Sigma \times \mathbb{T})^+ \\
\text{and } \alpha_{i+1} \in \Lambda(\alpha_i, u_i)\}
\end{aligned}$$

and $\Lambda''(\alpha, w) = \emptyset$.

With these definitions, the proof of Proposition 4 is now technical but without major difficulty.

4.4 Basic properties

The following result summarizes the properties of the operators $\underset{C}{\cdot}$ and $\overset{+}{C}$ needed in the following.

Proposition 5. *Let $(\mathcal{G},\Lambda)$, $(\mathcal{G}',\Lambda')$ and $(\mathcal{G}'',\Lambda'')$ three constrained generators and let C, D two subsets of clocks. Then it holds:*

1. $(\mathcal{G},\Lambda) \underset{C}{\cdot} ((\mathcal{G}',\Lambda') + (\mathcal{G}'',\Lambda'')) = ((\mathcal{G},\Lambda) \underset{C}{\cdot} (\mathcal{G}',\Lambda')) + ((\mathcal{G},\Lambda) \underset{C}{\cdot} (\mathcal{G}'',\Lambda''))$
2. $((\mathcal{G},\Lambda) + (\mathcal{G}',\Lambda')) \underset{C}{\cdot} (\mathcal{G}'',\Lambda'') = ((\mathcal{G},\Lambda) \underset{C}{\cdot} (\mathcal{G}'',\Lambda'')) + ((\mathcal{G}',\Lambda') \underset{C}{\cdot} (\mathcal{G}'',\Lambda''))$
3. $((\mathcal{G},\Lambda) \underset{C}{\cdot} (\mathcal{G}',\Lambda')) \underset{D}{\cdot} (\mathcal{G}'',\Lambda'') = (\mathcal{G},\Lambda) \underset{C}{\cdot} ((\mathcal{G}',\Lambda') \underset{D}{\cdot} (\mathcal{G}'',\Lambda''))$
4. $(\mathcal{G},\Lambda)^{\overset{+}{C}} \underset{C}{\cdot} (\mathcal{G}',\Lambda') = \sum_{i\geq 1}(\mathcal{G},\Lambda)^{(\underset{C}{\cdot})^i} \underset{C}{\cdot} (\mathcal{G}',\Lambda')$

4.5 Equations on constrained generators

We consider two constrained generators $(\mathcal{G}_1,\Lambda_1)$ and $(\mathcal{G}_2,\Lambda_2)$ and a subset C of clocks. The following lemma is the fundamental result allowing to solve systems of equations on constrained generators.

Lemma 6. *The equation on constrained generators*

$$(\mathcal{G},\Lambda) = (\mathcal{G}_1,\Lambda_1) \underset{C}{\cdot} (\mathcal{G},\Lambda) + (\mathcal{G}_2,\Lambda_2)$$

has for unique solution the constrained generator $(\mathcal{G}_1,\Lambda_1)^{\overset{+}{C}} \underset{C}{\cdot} (\mathcal{G}_2,\Lambda_2) + (\mathcal{G}_2,\Lambda_2)$.

Note that even if the constrained generators $(\mathcal{G}_1, \Lambda_1)$ and $(\mathcal{G}_2, \Lambda_2)$ are associated with some timed automata, we can not assume, a priori, that a solution of the equation, if any, is also associated with a time automaton. It is the reason why we had to introduce all the technical staff on constrained generators.

The proof of this lemma can be found in Appendix A.

5 Decomposition of timed automata

We are now ready to prove our main result, Theorem 3. Let $\mathcal{A} = (Q, T, I, F, R)$ be a timed automaton. We assume that for any state q there exists some subset $C_q \subseteq X$ such that for any transition (q', g, a, r, q), it holds $r = C_q$. Note that, changing the set of states Q into the cartesian product $Q \times \mathcal{P}(X)$, it is easy to transform any timed automaton into an equivalent timed automaton verifying this property. We will propose now an algorithm to find a timed automaton $\mathcal{B}$ in $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$ which is equivalent to $\mathcal{A}$.

For any states $i, f \in Q$, we set $\mathcal{A}_{i,f} = (Q, T, \{i\}, \{f\}, \emptyset)$ and we consider the constrained generator $\widehat{\mathcal{A}_{i,f}}$ associated with $\mathcal{A}_{i,f}$. From the definition of run in a timed automaton (see Section 2), it is easy to verify that the following equation holds

$$\hat{\mathcal{A}} = \sum_{i\in I, f\in F} \widehat{\mathcal{A}_{i,f}} + \sum_{i \in I, q \in R} \widehat{\mathcal{A}_{i,q}} \cdot_{C_q} \widehat{\mathcal{A}_{q,q}}^{\overset{\omega}{C_q}}$$

which, in terms of timed automata and using Proposition 4, can be rewritten equivalently in

$$\mathcal{A} \equiv \sum_{i\in I, f\in F} \mathcal{A}_{i,f} + \sum_{i \in I, q \in R} \mathcal{A}_{i,q} \cdot_{C_q} \mathcal{A}_{q,q}^{\overset{\omega}{C_q}} \qquad (\dagger)$$

Therefore, in order to prove that $\mathcal{A}$ is equivalent to some automaton in $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$, it suffices to prove that for any $q, q' \in Q$, the automaton $\mathcal{A}_{q,q'}$ belongs to $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$.

Assume now that $f \in Q$ is fixed, then the family of constrained generators $(\widehat{\mathcal{A}_{q,f}})_{q \in Q}$ verifies the system of equations on constrained generators - where $\langle \delta a \rangle_g$ denotes the constrained generator associated with the basic timed automaton $\mathcal{A}_{\langle \delta a \rangle_g}$ defined in Section 3.4:

$$\begin{cases} \widehat{\mathcal{A}_{q,f}} = \varepsilon + \sum_{(q,g,a,C_{q'},q') \in T} \langle \delta a \rangle_g \cdot_r \widehat{\mathcal{A}_{q',f}} \text{ if } q = f \\ \widehat{\mathcal{A}_{q,f}} = \sum_{(q,g,a,C_{q'},q') \in T} \langle \delta a \rangle_g \cdot_r \widehat{\mathcal{A}_{q',f}} \text{ otherwise} \end{cases}$$

Consider now an arbitrary order $q_1 < q_2 < \ldots < q_n$ on the elements of Q. Then the equation with left member $\widehat{\mathcal{A}_{q_n,f}}$ is solved, with $\widehat{\mathcal{A}_{q_n,f}}$ as unknown, using the fundamental Lemma 6 - ε has to be added if $q_n = f$:

$$\widehat{\mathcal{A}_{q_n,f}} = \left(\sum_{(q_n,g,a,C_{q_n},q_n) \in T} \langle \delta a \rangle_g \right)^{\overset{+}{C_{q_n}}} \cdot_{C_{q_n}} \left(\varepsilon_{q,f} + \sum_{(q_n,g,a,C_{q'},q') \in T, q' \neq q_n} \widehat{\mathcal{A}_{q',f}} \right)$$

We thus replace $\widehat{\mathcal{A}_{q_n,f}}$ by this formula in the $n-1$ other equations. Step by step, we solve the system using the fundamental Lemma 6. The last step proves that the constrained generator $\widehat{\mathcal{A}_{q_1,f}}$ can be expressed using the elementary constrained generators $\langle \delta a \rangle_g$ and the composition operators, which is similar to say that the automaton $\mathcal{A}_{q_1,f}$ is equivalent to some automaton of $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$. We thus deduce that $\mathcal{A}_{q_2,f}$ and then $\mathcal{A}_{q_3,f}, \ldots, \mathcal{A}_{q_n,f}$ are also equivalent to some automata of $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$.

Finally, for any $q, q' \in Q$, every automaton $\mathcal{A}_{q,q'}$ is equivalent to some automaton of $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$ and thus, using the equation (†), $\mathcal{A}$ is also equivalent to some automaton of $\mathrm{Mod}(\Sigma, \mathbb{T}, X)$. Theorem 3 is therefore proved.

This method is illustrated on an example in Appendix B.

6 Conclusion

We have proposed in this paper a Kleene-Büchi's theorem for timed languages. We have precisely prove that any timed automaton is equivalent to a timed automaton built on a modular way form basic objects using the operations of union, timed concatenations through subsets of clocks and their induced finite and infinite iterations. For such an automaton constructed in a modular way, the classical emptiness procedure [3,4] can also be done in a modular way in parallel with the construction of the automaton, see [10]. Even if the complexity in the worst case is PSPACE (Alur and Dill have shown that the problem is PSPACE-complete), this procedure has given promising much simpler results on some non trivial examples.

References

1. R. Alur, C. Courcoubetis, D.L. Dill, N. Halbwachs, and H. Wong-Toi. Minimization of timed transition systems. In *Proceedings of CONCUR'92*, number 630 in Lecture Notes in Computer Science. Springer Verlag, 1992.
2. R. Alur, C. Courcoubetis, and T.A. Henzinger. The observational power of clocks. In *Proceedings of CONCUR'94*, number 836 in Lecture Notes in Computer Science, pages 162–177. Springer Verlag, 1994.
3. R. Alur and D.L. Dill. Automata for modeling real-time systems. In *Proceedings of ICALP'90*, number 443 in Lecture Notes in Computer Science, pages 322–335. Springer Verlag, 1990.
4. R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.
5. R. Alur, L. Fix, and T.A. Henzinger. A determinizable class of timed automata. In *Proceedings of CAV'94*, number 818 in Lecture Notes in Computer Science, pages 1–13. Springer Verlag, 1994.
6. R. Alur and T.A. Henzinger. Back to the future: towards a theory of timed regular languages. In *Proceedings of FOCS'92*, Lecture Notes in Computer Science, pages 177–186. Springer Verlag, 1992.

7. E. Asarin. Equations on timed languages. In *Proceedings of Hybrid'98*, LNCS, 1998.
8. E. Asarin, P. Caspi and O. Maler. A Kleene theorem for timed automata. In *Proceedings of LICS'97*, 1997.
9. B. Bérard. Untiming timed languages. *Information Processing Letters*, 55:129–135, 1995.
10. P. Bouyer. Automates temporisés et modularité Rapport de stage de DEA, LSV, ENS de Cachan, 1998.
11. C. Courcoubetis and M. Yannakakis. Minimum and maximum delay problems in real-time systems. In *Proceedings of CAV'91*, number 575 in Lecture Notes in Computer Science, pages 399–409. Springer Verlag, 1991.
12. B. Bérard, V. Diekert, P. Gastin, and A. Petit. Characterization of the expressive power of silent transitions in timed automata. To appear in *Fundamenta Informaticae*, available at http://www.lsv.ens-cachan.fr/Publis/, 1998.
13. J.A. Brzozowski. A survey of regular expressions and their applications. In *IRE transactions on Electronic Computers*, EC-11(3): 324–335, 1962.
14. J.R. Büchi. A decision method in restricted arithmetic. In Proc *Int Congr. on Logic, Methodology and Philosophy of Science*, Stanford University, 1960.
15. T.A. Henzinger, P.W. Kopke, and H. Wong-Toi. The expressive power of clocks. In *Proceedings of ICALP'95*, number 944 in Lecture Notes in Computer Science, pages 335–346. Springer Verlag, 1995.
16. T.A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111(2):193–244, 1994.
17. S.C. Kleene. Representations of events in nerve and finite automata. *Automata Studies*, Princeton University Press, 3–42, 1956.
18. T. Wilke. Specifying timed state sequences in powerful decidable logics and timed automata. In H. Langmaack, W.-P. de Roever, and J. Vytopil, editors, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 863 of *Lecture Notes in Computer Science*. Springer Verlag, 1994.
19. H. Wong-Toi and G. Hoffmann. The control of dense real-time discrete event systems. In *Proceedings of the 30th IEEE Conf. on Decision and Control*, pages 1527–1528, 1991.

A Proof of Lemma 6

Proof. First let us show that $(\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)$ is indeed a solution. Using Proposition 5 1), 2) and 3), it holds

$$\begin{aligned}
&(\mathcal{G}_1, \Lambda_1) \cdot_C ((\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)) + (\mathcal{G}_2, \Lambda_2) \\
&= ((\mathcal{G}_1, \Lambda_1) \cdot_C (\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} + (\mathcal{G}_1, \Lambda_1)) \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2) \\
&= (\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)
\end{aligned}$$

Now let $(\mathcal{G}, \Lambda)$ be a solution, then $(\mathcal{G}_2, \Lambda_2) \subseteq (\mathcal{G}, \Lambda)$ [5]. And thus $(\mathcal{G}_1, \Lambda_1) \cdot_C (\mathcal{G}_2, \Lambda_2) \subseteq (\mathcal{G}, \Lambda)$ and by an immediate induction, for any $i \geq 1$, $(\mathcal{G}_1, \Lambda_1)^{(\cdot_C)^i} \cdot_C (\mathcal{G}_2, \Lambda_2) \subseteq (\mathcal{G}, \Lambda)$. Therefore $\sum_{i \geq 1}((\mathcal{G}_1, \Lambda_1)^{(\cdot_C)^i} \cdot_C (\mathcal{G}_2, \Lambda_2)) \subseteq (\mathcal{G}, \Lambda)$. Hence using Proposition 5 4), we deduce that $(\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) \subseteq (\mathcal{G}, \Lambda)$. Finally, any solution contains $(\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)$.

Conversely assume that $(\mathcal{G}, \Lambda)$ is a solution which contains strictly $(\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)$. We introduce the following notations $(\mathcal{G}', \Lambda') = (\mathcal{G}_1, \Lambda_1) \cdot_C (\mathcal{G}, \Lambda)$, $(\mathcal{G}_4, \Lambda_4) = (\mathcal{G}_1, \Lambda_1)^{\overset{+}{C}} \cdot_C (\mathcal{G}_2, \Lambda_2) + (\mathcal{G}_2, \Lambda_2)$ and $(\mathcal{G}_3, \Lambda_3) = (\mathcal{G}_1, \Lambda_1) \cdot_C (\mathcal{G}_4, \Lambda_4)$. Note that since $(\mathcal{G}_4, \Lambda_4)$ is a solution of the equation, it holds $(\mathcal{G}_3, \Lambda_3) \subseteq (\mathcal{G}_4, \Lambda_4)$. Several cases are possible.

1. There exists some clock valuation α and some timed word u such that $u \in \mathcal{G}(\alpha)$ but $u \notin \mathcal{G}_4(\alpha)$.
   Assume that u is minimal among these words i.e., for any clock valuation β and any timed word v such that $|v| < |u|$, if $v \in \mathcal{G}(\beta)$ then $v \in \mathcal{G}_4(\beta)$.
   Since $(\mathcal{G}, \Lambda)$ is solution of the equation, it holds either $u \in \mathcal{G}_2(\alpha)$ or $u \in \mathcal{G}'(\alpha)$. The case $u \in \mathcal{G}_2(\alpha)$ is impossible since $\mathcal{G}_2(\alpha) \subseteq \mathcal{G}_4(\alpha)$. If $u \in \mathcal{G}'(\alpha)$, it holds $u = v \cdot w$ with $v \in \mathcal{G}_1(\alpha)$ and $w \in \mathcal{G}(\beta[C := 0])$ for some $\beta \in \Lambda_1(v)$. But since $\mathcal{G}_1(\alpha)$ does not contain the empty word, $|w| < |u|$. Therefore, from the minimality of u, we deduce that $w \in \mathcal{G}_4(\beta[C := 0])$. Thus $u = v \cdot w \in \mathcal{G}_3(\alpha) \subseteq \mathcal{G}_4(\alpha)$ which is in contradiction with the hypothesis. This first case is thus impossible.
2. There exist some clock valuations α, β and some timed word u such that $\beta \in \Lambda(\alpha, u)$ but $\beta \notin \Lambda_4(\alpha, u)$.
   Assume that u is minimal among these words i.e., for any clock valuation γ and any timed word v such that $|v| < |u|$, $\Lambda(\gamma, v) = \Lambda_4(\gamma, v)$.
   Since $(\mathcal{G}, \Lambda)$ is solution of the equation, it holds either $\beta \in \Lambda_2(\alpha, u)$ or $\beta \in \Lambda'(\alpha, u)$. The case $\beta \in \Lambda_2(\alpha, u)$ is impossible since $\Lambda_2(\alpha, u) \subseteq \Lambda_4(\alpha, u)$. If $\beta \in \Lambda'(\alpha, u)$, $\beta \in \Lambda(\gamma[C := 0], w)$ for some word $w \in \mathcal{G}(\gamma[C := 0])$ such that there exists some $v \in \mathcal{G}_1(\alpha)$ with $u = v \cdot w$ and $\gamma \in \Lambda_1(\alpha, v)$. But

[5] With the natural definition that a constrained generator $(\mathcal{G}, \Lambda)$ is included in another one $(\mathcal{G}', \Lambda')$ if for any clock valuation α and any finite timed word u, it holds $\mathcal{G}(\alpha) \subseteq \mathcal{G}'(\alpha)$ and $\Lambda(\alpha, u) \subseteq \Lambda'(\alpha, u)$.

since $\mathcal{G}_1(\alpha)$ does not contain the empty word, $|w| < |u|$. Therefore, from the minimality of u, we deduce that $\beta \in \Lambda_4(\gamma[C := 0], w)$. But from the first case, it holds $w \in \mathcal{G}(\alpha) = \mathcal{G}_4(\alpha)$. Therefore $\beta \in \Lambda_3(\alpha, u)$ and since $(\mathcal{G}_4, \Lambda_4)$ is solution of the equation, we deduce that $\beta \in \Lambda_4(\alpha, u)$ which is in contradiction with the hypothesis. This second case is thus also impossible.

B Example

We illustrate the algorithm on the following example of automaton.

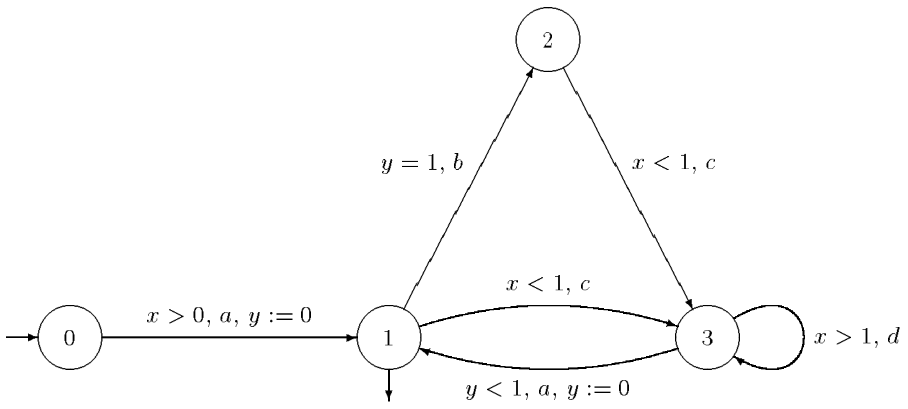

Fig. 3. Automate $\mathcal{A}$

This automaton defines the following system of equations on constrained generators:

$$\begin{cases} X_0 = \langle \delta a \rangle_{\mathbb{Q}_*^+ \times \mathbb{Q}^+} {}_{\dot{\{y\}}} X_1 \\ X_1 = \epsilon + \langle \delta b \rangle_{\mathbb{Q}^+ \times \{1\}} {}_{\dot{\emptyset}} X_2 + \langle \delta c \rangle_{[0;1[\times \mathbb{Q}^+} {}_{\dot{\emptyset}} X_3 \\ X_2 = \langle \delta c \rangle_{[0;1[\times \mathbb{Q}^+} {}_{\dot{\emptyset}} X_3 \\ X_3 = \langle \delta d \rangle_{]1;+\infty[\times \mathbb{Q}^+} {}_{\dot{\emptyset}} X_3 + \langle \delta a \rangle_{\mathbb{Q}^+ \times [0;1[} {}_{\dot{\{y\}}} X_1 \end{cases}$$

Using the algorithm presented in Section 5, we obtain as solution:

$$X_0 = \langle \delta a \rangle_{\mathbb{Q}_*^+ \times \mathbb{Q}^+} {}_{\dot{\{y\}}} \left(\epsilon + \left(\langle \delta b \rangle_{\mathbb{Q}^+ \times \{1\}} {}_{\dot{\emptyset}} \langle \delta c \rangle_{[0;1[\times \mathbb{Q}^+} {}_{\dot{\emptyset}} Z + \langle \delta c \rangle_{[0;1[\times \mathbb{Q}^+} {}_{\dot{\emptyset}} Z \right)^{\overset{+}{\{y\}}} \right)$$

with

$$Z = \left(\epsilon + \left(\langle \delta d \rangle_{]1;+\infty[\times \mathbb{Q}^+} \right)^{\overset{+}{\emptyset}} \right) {}_{\dot{\emptyset}} \langle \delta a \rangle_{\mathbb{Q}^+ \times [0;1[}$$

This defines a decomposition of $\mathcal{A}$ in basic timed automata as $\mathcal{A}_{\langle \delta a \rangle_{\mathbb{Q}_*^+ \times \mathbb{Q}^+}}$, $\mathcal{A}_{\langle \delta c \rangle_{[0;1[\times \mathbb{Q}^+}}$, ...