Removing silent transitions from event-clock automata

Cătălin Dima

Bucharest University, Department of Fundamentals of Computer Science,
14-th Academiei str., Bucharest, RO70109

Abstract. We prove that ϵ-transitions can be removed from event-clock automata. Thus event-clock automata with ϵ-transitions can be determinized and hence shown to be closed under complementation by the determinization procedure of Alur, Fix and Henzinger.
Keywords: timed automata, event-clock automata, ϵ-transitions.
Categories: real-time systems, formal languages.

1 Introduction

Safe system design is crucial wherever their failure may lead to disasters, like in nuclear plants or outer space exploration. Such systems are a mixture of analog and digital, so they have to be fully analyzed not only on their analog or digital properties, but also on the interactions between these two. In the past decade an emerging interest has been shown on the formalization of hybrid (i.e. mixed digital/analog) systems, with a particular interest on adapting tools from automata theory. This trend is easily justified by the success reported by automata theory, in conjunction with temporal and modal logics, on the specification and verification of discrete systems.

Many automata models have emerged, all showing problems specific to the use of the reals, instead of the naturals, as the time axis. The main problem is undecidability, which shows up in very simple models and is due to the density of the reals. The largest class of automata with timing constraints that has a decidable emptiness problem are the so-called *timed automata* of Alur and Dill [1]. However even this class has its drawback: it has an undecidable complementation problem. And since complementation is always used in specifying safety properties there has been an active search for subclasses of timed automata that are sufficiently expressive and are closed under complementation.

One such class is the class of so-called *event-clock automata* [2]. This class is able to express properties like bounded response ("action a takes place at some time point t which is between 2 and 3 time units from now") and is closed under complementation, fact which was proved using a modified version of the usual subset construction [7]. However there is a drawback in the original paper: the automata are not allowed to have "inobservable" (or ϵ-) transitions. One solution was found in [6] by shifting constraints from the transitions into the states. In this article we show that ϵ-transitions may be removed without changing the definition of event-clock automata in the sense of [6].

The main idea is similar to [3]: we shift the constraints from the ϵ-transitions to be removed to the neighboring transitions. This idea is initially applied to some simple automata, which have only a chain of ϵ-transitions. We also show that circuits with ϵ-transitions can be removed without other modifications. The full property is then obtained via a notion of "hierarchical cooperation". Roughly speaking, this notion says that some timed word is accepted by $m+1$ automata (where the first automaton is the "chief") iff it happens that each time the "chief" automaton passes through a certain state, some designated automaton has to pass through some critical subset of its states. This notion allows us to apply the following technique in removing ϵ-transitions: we first decompose the given automaton in a hierarchical cooperation of $m+1$ automata, where m is the number of distinct chains of ϵ-transitions, such that the "chief" automaton is ϵ-free and each of the other m automata contains only one chain of ϵ-transitions. Then transform each of the m automata into a ϵ-free one. Then we recompose from these m new automata and the "chief" automaton a new ϵ-free automaton, which may be thus passed to the determinization procedure of [2].

The paper runs as follows: in the following section we remind the notions of timed words, the use of event-clocks in accordance to these and also the definitions of event-clock automata, their regular expressions and the Kleene theorems that connect these from [4]. These are used to reduce the problem to the case when there is a finite number of chains of ϵ-transitions (this reduction is shown by defining extended automata in the style of [8]). The third section contains our main constructions. We end with a short section of conclusions.

2 Preliminaries

We describe first some notation: for a word $w \in \Sigma^*$ we denote by $|w|$ its length. $[n]$ is the set of natural numbers $\{1, \dots, n\}$. Notations $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_+$ are for the sets of nonnegative, resp. positive reals. Similarly $\mathbb{Q}_+$ stands for the set of nonnegative rationals.

A **timed word** [1] (TW for short) is a pair (w, θ) consisting of a word $w \in \Sigma^*$ and a strictly increasing function $\theta : [|w|] \longrightarrow \mathbb{R}_{\geq 0}$. The set of timed words over Σ is denoted $TW(\Sigma)$. A **timed word with silent transitions and limited observation** (LTW for short) is a tuple (w, τ, I, t_1, t_2) where

- $w \in \Sigma^*$ is a finite word and $0 \leq t_1 \leq t_2 < \infty$;
- $\tau : [m] \longrightarrow \mathbb{R}_{\geq 0}$ is a strictly increasing function with $m \geq |w|$ which is called the **sequence of time points of the actions** and
- $I \subseteq [m]$ with $card(I) = |w|$ is the set of time points of the **observable actions**.

The intuition is that observation of the system is restricted to the interval $[t_1, t_2]$ and that only at the time points in I the letters of w occur, in between them occuring inobservable steps. We denote $dom(\tau)$ the domain $[m]$ of the sequence of time points τ and $range(\tau)$ its range. The set of LTWs over Σ is denoted $LTW(\Sigma)$.

The underlying TW of a LTW $\alpha = (w, \tau, I, t_1, t_2)$ is $[\alpha] = (w, \theta)$ where if we list I in increasing order as $I = \{j_1, \dots, j_{|w|}\}$, $\theta : [|w|] \longrightarrow \mathbb{R}_+$ is defined as $\theta(k) = \tau(j_k)$ for all $k \in [|w|]$. Hence we get an *abstraction* operation $[\cdot] : LTW(\Sigma \longrightarrow TW(\Sigma)$.

We have a *partial* operation of concatenation of LTWs, defined when both operands have the same first three components and when the endpoint of the interval of observation of the first equals the initial point of the interval of observation of the second:

$$(w, \tau, I, t_1, t_2); (w', \tau', I', u_1, u_2) = (w, \tau, I, t_1, u_2) \text{ iff } w = w', \tau = \tau', I = I', t_2 = u_1$$

and is undefined otherwise. This operation is extended to a *total* operation on sets of LTW:

$$T_1; T_2 = \{(w, \tau, I, t_1, t_3) \mid \exists t_2 \in (t_1, t_3) \text{ such that } (w, \tau, I, t_1, t_2) \in T_1 \text{ and } (w, \tau, I, t_2, t_3) \in T_2\}$$

with unit being $T_0 = \{(w, \tau, I, t, t) \mid t \in \mathbb{R}, w, \tau, I \text{ arbitrary }\}$. Concatenation on sets of LTWs gives rise to a star: $T^* = \bigcup_{n \in \mathbb{N}} T^n$ where $T^0 = T_0$ and $T^{n+1} = T^n; T$.

In the sequel we remind the use of clocks in association with a LTW: We associate with each letter a of Σ two symbols, a **history clock** x_a and a **prophecy clock** y_a. Denote $H(\Sigma) = \{x_a \mid a \in \Sigma\}$ and $P(\Sigma) = \{y_a \mid a \in \Sigma\}$ the copies of Σ which arise this way.

Each LTW (w,τ,I,t_1,t_2) defines a **trajectory of clocks** which is a function $Ck = Ck_{((w,\tau,I),t_1)} : (H(\Sigma) \cup P(\Sigma)) \times \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}_{\geq 0} \cup \{\bot\}$ defined as:

$$Ck(x_a,t) = \begin{cases} t-\tau(i) & \text{iff for } A = \{k \in I \mid w[k]=a \text{ and } \tau(k)<t\} \text{ we have } A \neq \emptyset \text{ and } i = \max A \\ \bot & \text{otherwise (i.e. } A = \emptyset) \end{cases}$$

$$Ck(y_a,t) = \begin{cases} \tau(i)-t & \text{iff for } B = \{k \in I \mid w[k]=a \text{ and } \tau(k)>t\} \text{ we have } B \neq \emptyset \text{ and } i = \min B \\ \bot & \text{otherwise (i.e. } B = \emptyset) \end{cases}$$

Hence, at each time point $t \in \mathbb{R}_{\geq 0}$ the value of the clock x_a holds the time elapsed since the last occurrence of letter a in the LTW; if no such occurrences, $x_a = \bot$. Similarly, y_a holds the time remaining till the next occurrence of a or is $\bot$ if there is no such occurrence.

A **clock constraint** is a boolean combination of atomic formulas of the type $x_a < c$, $x_a \leq c$, $x_a = \bot$, and $y_a < c$, $y_a \leq c$ or $y_a = \bot$ where c is some positive rational $c \in \mathbb{Q}_+$. The set of clock constraints over Σ is denoted $Constr(\Sigma)$. For each LTW (w,τ,I,t_1,t_2) the trajectory of clocks Ck defines at each time point $t \in \mathbb{R}_{\geq 0}$ an interpretation of the clock constraints. We use the notation $(w,\tau,I,t) \models C$ to encode that the interpretation $Ck(\cdot,t)$, maps the constraint C to true.

For each clock $K \in H(\Sigma) \cup P(\Sigma)$ define the **elementary clock constraints** as all the formulas $K = \bot$, $c < K$, $c \leq K$, $c_1 < K \wedge K < c_2$, $c_1 \leq K \wedge K < c_2$, $c_1 < K \wedge K \leq c_2$ or $c_1 \leq K \wedge K \leq c_2$, for some $c, c_1, c_2 \in \mathbb{Q}_+$. A **simple constraint** is a formula $\bigwedge_{a\in\Sigma}(C(x_a) \wedge C(y_a))$ where $C(x_a)$ and $C(y_a)$ are elementary constraints. We denote $SC(\Sigma)$ the set of simple constraints.

It is clear that any clock constraint can be brought to a disjunction of simple clock constraints (a normal form without negation). E.g. $\neg(x_a < 2) \equiv (x_a \geq 2) \vee (x_a = \bot)$.

Definition 1. *An **event-clock automaton** (ECA for short) is a tuple $\mathcal{A} = (Q,\Sigma,\delta,q_0,F)$ where Q is the set of* states*, $Q_0, F \subseteq Q$ are the sets of* initial *resp.* final states *and δ is the* transition relation *$\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Constr(\Sigma) \times Q$ with $|\delta| < \infty$.*

A **run** of length $n \geq 1$ is a sequence of transitions $((q_{i-1}, x_i, C_i, q_i))_{i\in[n]}$ $(x_i \in \Sigma \cup \{\epsilon\})$. A zero-length run consists of some initial and final state. An **accepting run** is a sequence of transitions which starts in Q_0 and ends in F.

We have two types of laguages for ECA. The basic one consists of LTW: a LTW (w,τ,I,t_1,t_2) with $dom(\tau) = [n]$ is accepted by $\mathcal{A}$ iff there is an accepting run $((q_{i-1}, x_i, C_i, q_i))_{i\in[m]}$ such that m is the number of actions that occur within the interval $[t_1,t_2]$ and at the i-th action occurring in $[t_1,t_2]$ constraint C_i is satisfied. Formally, $m = card(\{i \in [n] \mid \tau(i) \in [t_1,t_2]\}$ and, denoting $j = \min\{i \in [m] \mid \tau(i) \in [t_1,t_2]\}$ we have that $(w,\tau,I,\tau(j+i-1)) \models C_i$ for all $i \in [n]$. We also say that the transition (q_{i-1}, x_i, C_i, q_i) is **taken** at the moment $\tau(j+i-1)$. We denote $L_l(\mathcal{A})$ the set of LTW accepted by $\mathcal{A}$ and call it the **limited observation language**.

The other semantics, the **language of abstract limited observation** is $L_{al}(\mathcal{A}) = [L_l(\mathcal{A})] = \{[(w,\tau,I,t_1,t_2)] \mid (w,\tau,I,t_1,t_2) \in L_l(\mathcal{A})\}$.

Proposition 1. *For each ECA $\mathcal{A}$ there exist an ECA $\mathcal{B}$ such that with $L_l(\mathcal{A}) = L_l(\mathcal{B})$ and such that all transitions in $\mathcal{B}$ have simple constraints (call this automaton **simple ECA**).*

Definition 2. *An ECA is **ϵ-free** iff there is no tuple in the transition relation labeled with the empty word. An ϵ-free ECA is **state-deterministic** iff it has a single initial state and for each two distinct transitions having the same source and labeled with the same symbol the conjunction of their constraints is a nonsatisfiable formula. An ECA which is both ϵ-free and state deterministic is simply called **deterministic**.*

Definition 3. *The set $\epsilon RE(\Sigma)$ of* ***ϵ-regular expressions*** *over Σ is defined by the following grammar where $a \in \Sigma$ and $C \in Constr(\Sigma)$:*

$$E ::= \emptyset \mid () \mid (\epsilon, C) \mid (a, C) \mid E + E \mid E; E \mid E^*$$

The rules without ϵ give the set of ***regular expressions****.*

Atomic expressions are (x, C) with $x \in \Sigma \cup \{\epsilon\}$ while $()$ is called the *empty expression*.

We define two types of semantics of ϵ-regular expressions, namely one in terms of LTW (the *concrete* semantics) and another one in terms of TW (the *abstract* semantics). The *concrete semantics* of an ϵ-regular expression E is denoted $|E|$ and for atoms is:

- $|()|$ contains all the LTW (w, τ, I, t_1, t_2) such that the interval $[t_1, t_2]$ does not contain any action (i.e. $range(\tau) \cap [t_1, t_2] = \emptyset$).
- $|(\epsilon, C)|$ contains some LTW (w, τ, I, t_1, t_2) iff there is a unique time point $\tau(i) \in [t_1, t_2]$ at which the inobservable action occurs, C holds there and no other action occurs in $[t_1, t_2]$ (i.e. $range(\tau) \cap [t_1, t_2] = \{\tau(i)\}$, $(w, \tau, I, \tau(i)) \models C$ and $w[i] = \epsilon$).
- $|(a, C)|$ contains some LTW (w, τ, I, t_1, t_2) iff there is a unique time point $\tau(i) \in [t_1, t_2]$ at which the observable action a occurs, C holds there and no other action occurs in the interval $[t_1, t_2]$ (i.e. $range(\tau) \cap [t_1, t_2] = \{\tau(i)\}$, $(w, \tau, I, \tau(i)) \models C$ and $w[i] = a$).

The semantics is then easily extended to ϵ-regular expressions by structural induction:

$$|E_1; E_2| = |E_1|; |E_2| \qquad |E_1 + E_2| = |E_1| \cup |E_2| \qquad |\emptyset| = \emptyset$$
$$|E^*| = \bigcup_{n \in \mathbb{N}} |E^n| \qquad \text{where } E^0 = () \text{ and } E^{n+1} = E^n; E$$

The *abstract* semantics, denoted $\|E\|$ comes via the map $[\cdot] : LTW(\Sigma) \longrightarrow TW(\Sigma)$: $\|E\| \stackrel{def}{=} [|E|]$.

Proposition 2. *Each ϵ-regular expression is concrete equivalent to an ϵ-regular expression in which all constraints are simple.*

Proposition 3. *For each ϵ-regular expression E containing only ϵ-atoms, $\|E^*\| = \|()\| = TW(\Sigma)$. Each ϵ-regular expression is equivalent to an ϵ-regular expression in which all stars of expressions containing ϵ-atoms have been removed.*

The proof is straightforward: since $|()| \subseteq |E^*|$ we also have $\|()\| \subseteq \|E^*\|$. But as one may easily see, $\|()\| = TW(\Sigma)$, hence the reverse inclusion holds too. The last part of the proof follows by an easy generalization.

Theorem 1 (Concrete Kleene theorem). *The class of sets of LTW which are the limited observation language of some ECA equals the class of LTW which are the concrete semantics of some ϵ-regular expression.*

Theorem 2 (Abstract Kleene theorem). *The class of sets of TW which are the abstract language of some ECA equals the class of sets of TW which are the abstract semantics of some ϵ-regular expression.*

These Kleene theorems suggest that, like in the case of finite automata [8], ECA can be extended such that transitions can be ϵ-regular expressions:

Definition 4. *An **extended ECA** is a tuple $\mathcal{E} = (Q, \Sigma, \delta, Q_0, F)$ where $Q, Q_0, F \subseteq Q$ are the sets of states, resp. initial and final states and the transition relation δ is a finite subset of $Q \times \epsilon RE \times Q$. An ϵ-transition is some $(q, E, r) \in \delta$ which contains only ϵ-atoms.*

A run in $\mathcal{E}$ is, as one expects, a finite sequence of transitions $((q_{i-1}, E_i, q_i))$. It is accepting whenever it starts in Q_0 and ends in F. A LTW (w, τ, I, t_1, t_2) with $dom(\tau) = [n]$ is accepted by $\mathcal{E}$ iff there is an accepting run $((q_{i-1}, E_i, q_i))_{i\in[m]}$ such that $m = card(\{i \in [n] \mid \tau(i) \in [t_1, t_2]\}$ and, denoting $j = \min\{i \in [m] \mid \tau(i) \in [t_1, t_2]\}$ we have that $(w, \tau, I, \tau_{j+i}) \in C_i$ for all $i \in [n]$. $L_l(\mathcal{E})$ consists then of the set of LTW accepted by $\mathcal{E}$ while $L_{al}(\mathcal{E}) = [L_l(\mathcal{E})]$.

Definition 5. *An **ECA with canonical ϵ-transitions** is an extended ECA in which:*

- *transitions may be labeled with atomic expressions whose atoms use letters or ϵ-regular expressions containing only concatenations of ϵ-atoms which have simple constraints;*
- *if a state q has an ϵ-transition leaving it then only that transition leaves it and the dual property, if q has an incoming ϵ-transition then only that transition enters q;*
- *no two ϵ-transitions are consecutive;*

Proposition 4. *Given some ECA we can build an ECA with canonical ϵ-transitions having the same language of abstract limited observation.*

The proof of this property relies on Proposition 3: denoting $\mathcal{A}$ the ECA in discussion, for each pair of states $q, r \in Q$ put E^{ϵ}_{qr} the ϵ-regular expression whose concrete semantics contains all the LTW that have a run in the ECA that starts in q, ends in r and consists only of ϵ-transitions. We may assume that expressions have only simple constraints by proposition 2. The second step is to remove stars in each E^{ϵ}_{qr} (with the aid of Proposition 3 and the abstract Kleene theorem) transforming it to, say, $\overline{E}^{\epsilon}_{qr}$ (this way only the language of abstract limited observation is preserved!). Also by distributivity of $+$ we put each such expression into a sum of concatenations of ϵ-regular expressions. Then replace in $\mathcal{A}$ all ϵ-transitionwith some new ones: for each pair q, r put the transition $(q, \overline{E}^{\epsilon}_{qr}, r)$ iff $\overline{E}^{\epsilon}_{qr}$ is nonempty. Finally we split states which do not meet the last two requirements by the usual state-splitting technique.

3 Removing the ϵ-transitions

We first study the case of an extended ECA $\mathcal{A} = (\{q_1, q_2\}, \Sigma, \delta, \{q_1\}, \{q_2\})$ where $\delta = \{(q_1, a, true, q_1), (q_2, a, true, q_2) \mid a \in \Sigma\} \cup \{(q_1, \epsilon, E, q_2)\}$ where $E = (\epsilon, C_1); \dots; (\epsilon, C_n)$ with all constraints C_i assumed *simple*, $C_i \in SC(\Sigma)$. Observe that if the language of $\mathcal{A}$ is nonempty then $x_a = \bot$ appears in C_i iff $x_a = \bot$ appears in C_j for all $i, j \in [n]$ and similarly for prophecy clocks. Hence Σ is partitioned it two ways: $\Sigma = \Sigma_1 \cup \Sigma_2 = \Sigma_3 \cup \Sigma_4$ with $\Sigma_1 \cap \Sigma_2 = \Sigma_3 \cap \Sigma_4 = \emptyset$ and $\Sigma_1 = \{a \in \Sigma \mid x_a = \bot \text{ appears in some } C_i\}$, $\Sigma_1 = \{a \in \Sigma \mid y_a = \bot \text{ appears in some } C_i\}$. Also denote $\Sigma_2 = \{a_1, \dots, a_{k_1}\}$, $\Sigma_4 = \{b_1, \dots, b_{k_2}\}$ and $x_{a_j} \in (\alpha^i_j, \beta^i_j)$ the constraint on x_{a_j} appearing in C_i, respectively $y_{b_j} \in (\mu^i_j, \nu^i_j)$ the constraint on y_{b_j} appearing in C_i. Call this kind of automata **basic**.

An important observation is that for each $1 \le i < j \le n$ we may replace any α^j_p with $\max\{\alpha^i_p, \alpha^j_p\}$, hence obtaining $\alpha^1_p \le \dots \le \alpha^n_p$ and similarly for β's, μ's and ν's.

The idea of elimination of ϵ-transitions from $\mathcal{A}$ is to shift all the constraints on the ϵ-transition to non-ϵ-transitions appearing before or after the ϵ-transition.

Thinking of an accepting trajectory in $\mathcal{A}$, denote s_i the first moment after taking the ϵ-transition where letter b_i occurs (for all $i \in [k_2]$) and s_0 the last moment before the ϵ-transition where some letter occurs (we assume $\Sigma_2 \neq \emptyset \neq \Sigma_4$). For some $i \in [n]$ denote r_i the point where C_i is checked during the trajectory, $r_i \in (s_0, \min\{s_i \mid i \in [k_2]\}$. Checking C_i at r_i implies the following (we use round parentheses):

$$s_j - r_i \in (\mu_j^i, \nu_j^i) \forall\, j \in [k_2] \tag{1}$$

Proposition 5. *The existence of $r_1 < \ldots < r_n$ with $r_1 > s_0$ and the requirements 1 is equivalent to the following relations between the points $(s_i)_{i\in[n]}$:*

$$s_p - s_q \in (\mu_p^i - \nu_q^i, \nu_p^i - \mu_q^i) \quad \forall\, i \in [n],\ p, q \in [k_2],\ p \neq q \tag{2}$$

$$s_p - s_0 \in [\mu_p^i, \infty) \quad \forall\, i \in,\ p \in [k_2] \tag{3}$$

Note that the information in 2 is twice as much as needed, both constraints on $s_p - s_q$ and $s_q - s_p$ appearing. We will drop relations containing negative intervals (the interval of the opposite relation is positive then) and combine opposite relations with $\nu_q^i > \mu_p^i$ and $\nu_p^i > \mu_q^i$ (i.e. intervals overlapping 0) into the following:

$$s_p - s_q \in (0, \nu_p^i - \mu_q^i) \quad \text{and} \quad s_q - s_p \in (0, \nu_q^i - \mu_p^i)$$

The idea of shifting the constraints is accomplished by labeling the first b_p-transition after the ϵ-transition with $K_p = \bigwedge_{q\in[k_2], i\in[n]} K_{pq}^i$ where K_{pq}^i carries the information about the difference between s_p and s_q:

$$K_{pq}^i = \begin{cases} true & \text{iff } \nu_q^i \leq \mu_p^i \\ y_{b_q} \in (0, \nu_q^i - \mu_p^i) & \text{iff } \nu_q^i > \mu_p^i \text{ and } \nu_p^i > \mu_q^i \\ y_{b_q} \in (\mu_q^i - \nu_p^i, \nu_q^i - \mu_p^i) & \text{iff } \nu_p^i \leq \mu_q^i \end{cases}$$

Moreover the last transition before the ϵ-transition will be labeled with

$$C_{bef} = \bigwedge_{i\in[n], p\in[k_2]} y_{b_p} \in [\mu_p^i, \infty) \tag{4}$$

Analogously we define appropriate constraints on the history clocks L_p and C_{aft} (whose effective construction is skipped here). The ϵ-free ECA is then $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, \delta, Q_0, F)$ where $Q_{\mathcal{B}} = \{(S,1) \mid \emptyset \neq S \subseteq \Sigma_2\} \cup \{(S,2) \mid \emptyset \neq S \subseteq \Sigma_4\} \cup \{\epsilon\}$ where ϵ encodes the passage through the ϵ-transition, states $(S,1)$ appear before and $(S,2)$ after state ϵ, and when in a state $(S,1)$ letters from S are still bound to appear before passing through ϵ in any accepting trajectory, while for a state $(S,2)$ letters in S have already appeared after passing through ϵ. With this in mind we put $Q_0 = \{(\Sigma_2, 1)\}$ and $F = \{(\Sigma_4, 2)\}$, while the transition relation has:

- loops at $(S,1) \in Q_{\mathcal{B}}$ with letters in S: $((S,1), a, true, (S,1))$ for $a \in S$ and similarly for $(S,2) \in Q_{\mathcal{B}}$, $((S,2), a, true, (S,2))$ with $a \in S$;
- transitions with the last occurrence of some $a_p \in \Sigma_2$: for $\emptyset \neq S \backslash \{a_p\} \subseteq \Sigma_2$, $((S,1), a_p, L_p, (S \backslash \{a_p\}, 1))$ and similarly for $b_p \in \Sigma_4$: for $\emptyset \neq S \setminus \{b_p\} \subseteq \Sigma_4$, $((S,2), b_p, K_p, (S \setminus \{a_p\}, 2))$;
- transitions just before the state ϵ: $(\{a_p\}, a_p, L_p \wedge C_{bef}, \epsilon)$ and transitions just after ϵ: $(\epsilon, b_p, K_p \wedge C_{bef}, \{b_p\})$.

Note again that $\mathcal{B}$ was built in case $\Sigma_2 \neq \emptyset$, $\Sigma_4 \neq \emptyset$. But the other cases involve just simple modifications of the $\mathcal{B}$: e.g. when $\Sigma_2 = \emptyset$ $Q_{\mathcal{B}}$ does not contain states $(S,1)$ and $Q_0 = \{\epsilon\}$, hence there is no transition before ϵ.

3.1 The general case

First we define what does it mean, for some ECA $\mathcal{A}$, some run of it $((q_{i-1}, x_i, C_i, q_i))_{i\in[n]}$ and LTW $\alpha = (w, \tau, I, t_1, t_2)$ associated to the run, that at the moment $t \in [t_1, t_2]$ the automaton $\mathcal{A}$ **is in state** q: it means that there is some $\tau(j+i-1) \leq t \leq \tau(j+i)$ such that the i-th and $i+1$-th transitions are taken at these moments and the intermediary state is q, i.e. $q = q_i$. We then say that $[u_1, u_2] \subseteq [t_1, t_2]$ is a **maximal interval** in which $\mathcal{A}$ stays in state q iff $\mathcal{A}$ is in state q at any moment $t \in [u_1, u_2]$ and there exist η such that for any $t \in [u_1 - \eta, u_1] \cup [u_2, u_2 + \eta]$ the automaton is not in state q at moment t.

Definition 6. *Given $m+1$ ECA $\mathcal{A}$ and $\mathcal{A}_1, \ldots, \mathcal{A}_m$, m distinct states $r_j \in Q_{\mathcal{A}}$ and in each automaton $\mathcal{A}_j$ a subset of states Q_j. The* ***language accepted by hierarchical cooperation*** *between $\mathcal{A}$ and $\mathcal{A}_1, \ldots, \mathcal{A}_m$ constrained by the states $(r_j)_{j\in[m]}$ and the tuple of sets of states $(Q_1, \ldots, Q_m)$ is the set of TWs (w, τ) which is the underlying TW of some LTW $\alpha_0 = (w, \tau^0, I^0, t_1, t_2)$ (i.e. $[\alpha_0] = (w, \tau)$) such that*

- *α_0 is associated to some run of $\mathcal{A}$ $((q^0_{i-1}, x^0_i, C^0_i, q^j_i))_{i\in[n_0]}$ having the property that*
- *for each maximal interval $[u_1, u_2] \subseteq [t_1, t_2]$ for which $\mathcal{A}$ is in some state r_j at any moment $t \in [u_1, u_2]$,*
- *there exist a second LTW $(w, \tau^j, I^j, t_1, t_2)$ which is associated to some accepting run $((q^j_{i-1}, x^j_i, C^j_i, q^j_i))_{i\in[n_0]}$ of $\mathcal{A}_j$ such that $\mathcal{A}_j$ is in one of the states of Q_j at all the moments $t \in [u_1, u_2]$.*

Note that the accepted language is *abstract* in the sense we have employed this notion.

Proposition 6. *In the framework of definition 6, suppose an ECA $\mathcal{A}'_j$ which is abstract equivalent to $\mathcal{A}_j$ has the property that for some subset $\overline{Q_j}' \subseteq Q_{\mathcal{A}'_j}$ the following requirement is met:*

> *For each accepting run $((q_{i-1}, x_i, C_i, q_i))_{i\in[n]}$ of $\mathcal{A}_j$ associated to some LTW (w, τ, I, t_1, t_2) there exists some accepting run $((q'_{i-1}, x'_i, C'_i, q'_i))_{i\in[n]}$ of $\mathcal{A}'_j$ such that*
> *if $\mathcal{A}_j$ stays in Q_j in some moment $t \in [t_1, t_2]$ of this run then $\mathcal{A}'_j$ stays in $\overline{Q_j}'$ at this moment too.*

Then $\mathcal{A}_j$ can be replaced with $\mathcal{A}'_j$ in the hierarchical cooperation without any change of the accepted language.

Going back to the given ECA $\mathcal{A}$ we built, for each ϵ-transition (q, ϵ, E_{qr}, r) of $\mathcal{A}$ the respective basic ECA (denote it $\mathcal{A}_{qr}$) and fuse the states q and r (denote the resulting state qr). State qr will bear all transitions of q and r excepting the ϵ-transition. Note that the requirements on ECA with canonical ϵ-transitions avoid situations where we want to fuse q with r, then r with s. Also no loop is added at any state qr.

Hence the language accepted by hierarchical cooperation between the modified $\mathcal{A}$ (where all states q, r with $E_{qr} \neq \emptyset$ are fused) and all basic ECAs $\mathcal{A}_{qr}$, constrained by the tuples $(qr, Q_{\mathcal{A}_{qr}})$ is exactly the language of the initial ECA $\mathcal{A}$. Note that the requirements from Proposition 6 are met if we replace any automaton $\mathcal{A}_{qr}$ with its ϵ-free equivalent built up in the previous section, the set of states which matches $Q_{\mathcal{A}_{qr}}$ being ϵ, the state which marked the passage through the ϵ-transition(see the construction).

Finally we build an automaton $\mathcal{C}$ for the hierarchical cooperation of $\mathcal{A}, \mathcal{A}_1, \ldots \mathcal{A}_m$ when all automata are ϵ-free, hence all $\overline{Q_j}$ are singletons $\{\overline{q_j}\}$. States of $\mathcal{C}$ are tuples $(s, T_1, \ldots, T_m)$

with $s \in Q_{\mathcal{A}}$ and $T_j \subseteq Q_{\mathcal{A}_j}$ with the constraint that if $s = r_j$ then $\overline{r_j} \in T_i$. Initial states of $\mathcal{C}$ are all tuples involving only initial states of all automata and similarly for final states of $\mathcal{C}$. Transitions from $(s, T_1, \ldots, T_m)$ to $(s', T_1', \ldots, T_m')$ are taken with (a, C) iff:

- $(s, a, C', s') \in \delta_{\mathcal{A}}$ for some constraint C';
- for each $t_j \in T_j$ the set $\{t_j' \in T_j' \mid \exists\, (t_j, a, C_j, t_j') \in \delta_{\mathcal{A}_i}\}$ is nonempty and the conjunction of all constraints appearing in this set is a satisfiable constraint;
- for each $t_j' \in T_j'$ the set $\{t_j \in T_j \mid \exists\, (t_j, a, C_j, t_j') \in \delta_{\mathcal{A}_i}\}$ is nonempty and the conjunction of all constraints appearing in this set is a satisfiable constraint;
- C is the conjunction of all the above constraints.

A run in $\mathcal{C}$ is a guess of how many times $\mathcal{A}$ passes through each q_i and an appropriate match of T_j each time this happens. Note that some of the states in $T_j \setminus \{\overline{q_j}\}$ are used for future passing through q_j.

The resulting automaton is still state-nondeterministic but the algorithm from [2] brings it to a deterministic one. Hence

Theorem 3. *For each ECA there exists an equivalent ϵ-free ECA and, hence, an equivalent deterministic ECA. For each ϵ-regular expression E there exists a regular expression E' which is has the same abstract semantics as E.*

4 Conclusions

We have seen in this paper that ϵ-transitions can be removed from event-clock automata too, but with some more involved constructions than in the untimed case [7]. This problem shows to be specific to automata with timing constraints, see also [3, 5].

An observation to be made here is that if we would have defined the language of an ECA in terms of initialized TW then initial ϵ-transitions could not have been removed. Also note that checking emptiness of the language of some basic ECA is done in polynomial time w.r.t. the length of the ϵ-regular expression labeling the ϵ-transition(see [2]).

References

1. R. Alur and D.L. Dill. A theory of timed automata, *Theoretical Computer Science*, 126, 183-235, 1994.
2. R. Alur, L. Fix and T.A. Henzinger. A determinizable class of timed automata, *Theoretical Computer Science*, 211, 253-273, 1999.
3. B. Bérard, V. Diekert, P. Gastin and A. Petit. Characterization of the expressive power of silent transitions in timed automata, *Fundamenta Informaticæ*, 36:145-182, 1998.
4. C. Dima. Two Kleene theorems for event-clock automata, in *Proceedings of FCT'99, G. Ciobanu and Gh. Păun, (eds.)*, LNCS 1684:215-225, 1999.
5. C. Dima. Real-time automata and the Kleene algebra of sets of real numbers, to appear in *Proceedings of STACS'2000*, LNCS series, 2000.
6. T.A. Henzinger, J.-F. Raskin and P.-Y. Schobbens. The regular real-time languages, in *Proceedings of ICALP'98*, LNCS series, Springer Verlag, 1998.
7. John E. Hopcroft and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley/Narosa Publishing House, eighth edition, New Delhi, 1992.
8. Sheng Yu. Regular languages. Chapter 2 of G. Rozenberg and A. Salomaa (eds.), *Handbook of Formal Languages*, Vol. 1, Springer Verlag, 1997.