Real-time automata and the Kleene algebra of sets of real numbers

Cătălin Dima*

Bucharest University, Department of Fundamentals of Computer Science

Abstract. A commutative complemented Kleene algebra of sets of (positive) real numbers is introduced. For the subalgebra generated by finite unions of rational intervals a normal form is found. These are then applied to the complementation problem for real-time automata.

1 Introduction

Computing with intervals has sometimes been a by-product whenever a quantitative study of time and succession was of interest. Examples are constraint networks [DMP91], timed automata [AD94] and logics of real-time [AH96]. It is trivial that finite unions of intervals form a boolean algebra and in [CG98] it has been noticed that "periodic" unions of intervals form a boolean algebra, but the next and simple step of looking at a Kleene algebra of sets of positive reals has not yet (up to our knowledge) been an issue of interest.

We make this step here by defining star with the usual least fixpoint construction based on addition of *sets* of numbers, hence transforming sets of real numbers into a Kleene algebra. This algebra naturally arises in the study of the so-called *real-time automata* (RTA) [DW96], which are timed automata with a single clock which is reseted at each transition. In this model removing silent steps (a necessary step for transforming nondeterministic automata into deterministic ones and then "complementing" them) involves computing stars of unions of intervals. Hence even at this lowest level of generalization from the untimed finite automata removing silent transitions [BDGP98] is a problem.

We solve this problem here by studying the sub-Kleene algebra generated by finite unions of intervals with rational bounds. We prove a normal form theorem for the elements of this subalgebra, result which is based on properties of integer division. We then find a normal form for circuits with

* This research was done during the author's visit to TIFR, Bombay, supported by an extension of a UNU/IIST fellowship. Mailing address: str. Academiei 14, Bucharest, Romania, R0-70109, e-mail: `cdima@funinf.math.unibuc.ro`

silent transitions which allows complementation. Actually this is done by allowing clock constraints of the form $x \in X$ where X is no longer a finite union of intervals but some "periodic" union of intervals.

We note that, though RTA cannot be used for modeling distributed real-time systems, they show sufficient theoretical interest as they are incomparable to event-clock automata[AFH94,HRS98] which are so far the largest determinizable subclass of timed automata. This situation occurs as RTA may accept languages consisting of signals whose lengths are natural numbers while event-clock automata may not.

The rest of the paper is divided as follows: in the next section we remind some definitions and state the problem of complementation of real-time automata. In the third section the Kleene algebra of sets of positive reals is introduced and the normal form theorem is proved. The fourth section is for the determinization construction and the last section is for short comments and directions of further study.

2 Real-time automata

A **signal** over a finite alphabet V is a function $\sigma : [0, e) \longrightarrow V$ where e is a nonnegative number, function which has finitely many discontinuities, all of them being left discontinuities. Hence the domain of a signal σ splits into finitely many intervals $[e_{i-1}, e_i)$ on which σ is constant. We denote by $dom(\sigma)$ the domain of σ, and $Sig(V)$ the set of signals over V.

For $\sigma_1, \sigma_2 \in Sig(V)$ with $dom(\sigma_i) = [0, e_i)$ $(i = 1, 2)$ define their *concatenation* $\sigma_1; \sigma_2 = \sigma$ as the signal with $dom(\sigma) = [0, e_1 + e_2)$ and such that for $t \in [0, e_1)$, $\sigma(t) = \sigma_1(t)$ and for $[e_1, e_1 + e_2)$, $\sigma(t) = \sigma_2(t - e_1)$. Hence $(Sig(V), \text{``;''}, \sigma_\epsilon)$ becomes a noncommutative monoid whose unit is the signal σ_ϵ with $dom(\sigma_\epsilon) = [0, 0)$. Then concatenation can be extended for sets of signals and gives rise to star: for $\Sigma \subseteq Sig(V)$ put $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$. Here $\Sigma^0 = \{\sigma_\epsilon\}$ and $\Sigma^{n+1} = \Sigma^n; \Sigma$.

For the sequel, $Int_{\mathbb{Q}}^{\geq 0}$ denotes the set of intervals with positive rational or infinite bounds. Its elements are called *rational intervals.*

Definition 1. *A* ***real-time automaton*** *(RTA) over the alphabet V is a tuple $\mathcal{A} = (Q, V, \lambda, \iota, \delta, Q_0, F)$ where Q is the (finite) set of* states, *$\delta \subseteq Q \times Q$ is the* transition relation, *$Q_0, F \subseteq Q$ are the sets of* initial, *resp.* final *states, $\lambda : Q \longrightarrow V$ is the* state labeling function *and $\iota : Q \longrightarrow Int_{\mathbb{Q}}^{\geq 0}$ is the* interval labeling function*; also call q an a-*state *iff $\lambda(q) = a$.*

RTA work over signals: a *run* of length n is a sequence of states $(q_i)_{i \in [n]}$ connected by δ, i.e. $(q_{i-1}, q_i) \in \delta, \forall i \in [n]$. A run is **accepting** iff it starts in Q_0 and ends in F. A run is **associated to** a signal σ with $dom(\sigma) = [0,e)$ iff there exist some "splitting" points $0 = e_1 \leq \ldots \leq e_{n+1} = e$ such that $e_{i+1} - e_i \in \iota(q_i)$ and $\sigma(t) = \lambda(q_i)$ for all $t \in (e_i, e_{i+1})$ and all $i \in [n]$. Note that the "splitting" points must contain all the discontinuities but the reverse does not hold.

The **language** of some RTA $\mathcal{A}$ is the set of signals associated to some *accepting run* of $\mathcal{A}$ and is denoted $L(\mathcal{A})$. Two RTA are *equivalent* iff they have the same language. Define then the class of **timed recognizable languages** as $TRec(V) = \{\Sigma \in Sig(V) \mid \exists \mathcal{A} \text{ s.t. } L(\mathcal{A}) = \Sigma\}$.

Proposition 1. *The emptiness problem for RTA is decidable.*

The proof relies on the algorithms for computing the accessible (or coaccessible) states which can be done in linear time (w.r.t. $card(Q)$).

Definition 2. *The set* **ETRE**(V) *of* ***elementary timed regular expressions*** *over* V *is defined by the rules* $E ::= 0 \mid [a]_I \mid E{+}E \mid E;E \mid E^*$ *where the* atoms $[a]_I$ *are all expressions with* $a \in V$ *and* $I \in Int_{\mathbb{Q}}^{\geq 0}$.

Semantics of elementary timed regular expressions is the following:

$$|[a]_I| = \{\sigma \in Sig(V) \mid dom(\sigma) = [0,e), e \in I \text{ and } \forall t \in [0,e) \;\; \sigma(t) = a\}$$
$$|E{+}F| = |E| \cup |F| \qquad |E;F| = |E|;|F| \qquad |E^*| = |E|^* \qquad |0| = \emptyset$$

Define the class of **timed regular languages** as $TReg(V) = \{\Sigma \in Sig(V) \mid \exists E \in ETRE(V) \text{ such that } |E| = \Sigma\}$.

Theorem 1 (Kleene theorem for RTA). *$TRec(V) = TReg(V)$.*

This theorem is proved in [Dim99]. Note that the regular expressions defined here are weaker than the ones of [ACM97]. A class of regular expressions equivalent[1] to [ACM97] involve using *sets* of letters in the atoms, e.g. $[A]_{(1,2)}$ where $A \subseteq V$, see [Dim99].

We want to show that the class of timed recognizable languages is closed under complementation, hence we need a subclass of RTA in which each word has *a single run*, for then complementation would be accomplished simply by complementation of the set of final states.

Definition 3. *A RTA $\mathcal{A}$ is* ***language deterministic*** *iff each signal in $L(\mathcal{A})$ is associated to a unique run. $\mathcal{A}$ is* ***stuttering-free*** *iff it does not have time labels which contain 0 and transitions (q,r) with $\lambda(q) = \lambda(r)$.*

[1] This is actually the reason for denoting the expressions introduced here as *elementary*

*$\mathcal{A}$ is **state-deterministic** iff initial states have disjoint labels and transitions starting in the same states have disjoint labels too, i.e. whenever $r \neq s$ and either $r, s \in Q_0$ or $(q, r), (q, s) \in \delta$ then $\lambda(r) \neq \lambda(s)$ or $\iota(r) \cap \iota(s) = \emptyset$. $\mathcal{A}$ is simply called **deterministic** iff it is both state-deterministic and stuttering-free.*

A first observation is that determinism implies language determinism while state-determinism itself does not. But a more important observation is that deterministic RTA are *strictly less expressive* than general RTA: Consider the language $L_{\mathbb{N}} = \{\sigma : [0, n) \longrightarrow \{a\} \mid n \in \mathbb{N}\}$ of constant signals with integer length; it is accepted by the RTA in the figure $1(a)$.

Proposition 2. *$L_{\mathbb{N}}$ cannot be accepted by any stuttering-free RTA.*

The proof is based on the intuition that a stuttering-free RTA for $L_{\mathbb{N}}$ would need an infinite number of states. Note that this proof works for event-clock automata too [AFH94,HRS98], hence RTA and event-clock automata are incomparable.

However there is no problem for building a RTA for the complement of $L_{\mathbb{N}}$, as we see from the next figure. Also in this figure we find out that some stuttering RTA can still be transformed into stuttering-free RTA.

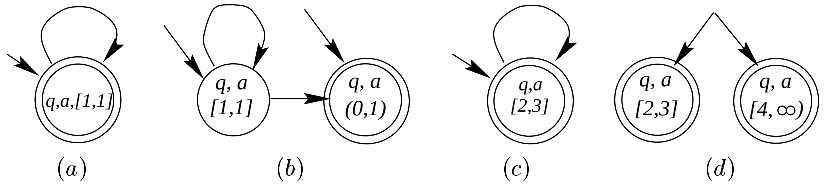

Fig. 1. The automaton at a accepts $L_{\mathbb{N}}$ while the complement of $L_{\mathbb{N}}$ is accepted by the RTA at (b). The stuttering RTA at (c) is equivalent to the stuttering-free one at (d).

Hence we discover the need of computing the "sum" of two intervals and the "star" of some interval, i.e. some operations that satisfy

$$\mathbb{R} \setminus \{1\}^* = \{1\}^* + (0, 1) \quad \text{and} \quad [2, 3]^* = [2, 3] \cup [4, \infty)$$

relations which are suggested by figure 1.

3 Operations with subsets of the real numbers

The powerset of the positive numbers $\mathcal{P}(\mathbb{R}_{\geq 0})$ is naturally endowed with an operation of "concatenation": it is addition extended over sets:

$$X + Y = \{x + y \mid x \in X, y \in Y\} \text{ for all } X, Y \subseteq \mathbb{R}$$

whose unit is $\mathbf{0} = \{0\}$.

Moreover we can define star via the usual least fixpoint construction $X^* = \bigcup_{n\in\mathbb{N}} nX$, where the multiples of X are defined as usual: $0X = \mathbf{0}$ and $(n+1)X = nX + X$.

The following theorem can then be easily verified:

Theorem 2. *The structure $\mathcal{P}(\mathbb{R}_{\geq 0}) = (\mathcal{P}(\mathbb{R}_{\geq 0}), \cup, +, \cdot^*, \emptyset, \mathbf{0})$ is a commutative Kleene algebra, i.e. $(\mathcal{P}(\mathbb{R}_{\geq 0}), +, \mathbf{0})$ is a commutative monoid, $+$ distributes over $\cup$ and $\cdot^*$ satisfies the following equations [Con71,Koz94]:*

$$X + Y \leq Y \Rightarrow X^* + Y \leq Y \tag{1}$$

$$\mathbf{0} \cup (X + X^*) \leq X^* \tag{2}$$

$$X^* + Y^* = (X+Y)^* + (X^* \cup Y^*) \tag{3}$$

where $X \leq Y$ denotes $X \cup Y = Y$.

Because a complement operation is available too: $\neg X = \mathbb{R}_{\geq 0} \setminus X$, we actually get a **commutative complemented Kleene algebra**, i.e. a boolean algebra which is also a commutative Kleene algebra.

Denote $\mathcal{K}(Int_{\mathbb{Q}}^{\geq 0})$ the sub-(commutative complemented Kleene) algebra generated by $Int_{\mathbb{Q}}^{\geq 0}$ in $\mathcal{P}(\mathbb{R}_{\geq 0})$.

Definition 4. *A set $X \in \mathcal{K}(Int_{\mathbb{Q}}^{\geq 0})$ can be written in* ***normal form*** *(NF) iff there exist finite unions of rational intervals $X_1, X_2 \in Int_{\mathbb{Q}}^{\geq 0}$ and some $k \in \mathbb{Q}_{\geq 0}$ such that*

$$X = X_1 \cup (X_2 + \{k\}^*) \tag{4}$$

with the requirement that there exists some $N \in \mathbb{N}$ such that $X_1 \subseteq [0, Nk)$ and $X_2 \subseteq [Nk, (N+1)k)$. We call this N the ***bound*** *of the NF.*

Normal forms are not unique: for the NF in the definition and some $p \in \mathbb{N}$, the following expression:

$$X = (X_1 \cup (X_2 + \{0, k, 2k, \ldots, (p-1)k\})) \cup (X_2 + \{pk\} + \{k\}^*)$$

is a NF too, but with bound $N + p$.

Clearly a finite union of rational intervals $X \in Int_{\mathbb{Q}}^{\geq 0}$ can always be put into NF. Also note that some NF $X = X_1 \cup (X_2 + \{k\}^*)$ has $X = \emptyset$ iff both X_1 and X_2 are empty.

Sometimes when applying different operations to NFs we might not be able to get very easily a new NF; instead, we might get a **weak normal form**, which is a decomposition like equation 4 but without the additional requirement on the existence of the bound. However we have the following:

Lemma 1. *Weak NFs can be transformed into NFs.*

The key result for NFs is the following:

Theorem 3. *Each $X \in \mathcal{K}(Int_{\mathbb{Q}}^{\geq 0})$ can be written in normal form.*

Proof. We must show that the result of any operation applied to NF can be put into NF. We first list some useful identities valid in $\mathcal{P}(\mathbb{R})$ [Con71]:

$$X^{**} = X^* \tag{5}$$

$$(X \cup Y)^* = X^* + Y^* \tag{6}$$

$$(X^* + Y)^* = \{0\} \cup (X^* + Y^* + Y) \tag{7}$$

Also note the following *ultimately periodicity property*:

> Given n distinct positive rationals $a_i \in \mathbb{Q}_{\geq 0}$ we have that $\{a_1, \dots, a_n\}^*$ is *ultimately periodic*, i.e. there exist some *finite* set of rationals B and some rationals $q, r \in \mathbb{Q}_{\geq 0}$ such that
>
> $$\{a_1, \dots, a_n\}^* = B \cup (\{q\} + \{r\}^*) \tag{8}$$

This result can be seen as a corollary of the normal form theorem of the regular languages over a one letter alphabet.

Fix now two NFs $X = X_1 \cup (X_2 + \{k\}^*)$ with bound M and $Y = Y_1 \cup (Y_2 + \{l\}^*)$ with bound N and denote $m = lcm(k, l)$. We then get the following form for $X \cup Y$:

$$X_1 \cup Y_1 \cup \left(\left(\bigcup_{i=0}^{m/k-1} (X_2 + \{ik\}) \cup \bigcup_{i=0}^{m/l-1} (Y_2 + \{il\}) \right) + \{m\}^* \right)$$

This is a WNF and Lemma 1 helps transforming it into NF.

For $X + Y$ distributivity of $+$ over $\cup$ transforms it into:

$$(X_1 + Y_1) \cup (X_1 + Y_2 + \{l\}^*) \cup (X_2 + Y_1 + \{k\}^*) \cup (X_2 + Y_2 + \{k\}^* + \{l\}^*)$$

An instantiation of identity 6 gives $\{k\}^* + \{l\}^* = \{k, l\}^*$. The ultimately periodicity property 8 gives a NF for the last and thence we have above a union of (weak) NFs which we already know how to bring to NF.

For X^* we have two cases. The first one occurs when one of X_1 and X_2 contains a nonpoint interval. Then *the set X^* is a finite union of rational intervals*, so it is in NF. To prove this claim, note that for each nonpoint interval, say $(a, b]$ (that is $b - a > 0$), denoting $m_0 = \left\lceil \frac{a}{b-a} \right\rceil$, we have that

$(a,b]^* = \mathbf{0} \cup \bigcup_{i=1}^{m_0-1} (ia, ib] \cup (m_0 a, \infty)$ since the choice of m_0 assures that $(m_0+1)a < m_0 b$. Hence from the m_0-th iteration the intervals start to overlap. This observation can be easily adapted to prove our claim.

The second case for X^* is when both X_1 and X_2 consist of point intervals. Applying identity 6 we get that $X^* = X_1^* + (X_2 + \{k\}^*)^*$. Then by the ultimately periodicity property 8 X_1 can be written into NF, so we concentrate on $(X_2 + \{k\}^*)^*$. Therefore, by identities 7 and 6 we get

$$(X_2 + \{k\}^*)^* = \mathbf{0} \cup (X_2 + X_2^* + \{k\}^*) \ = \ \mathbf{0} \cup (X_2 + (X_2 \cup \{k\})^*)$$

Now the ultimately periodicity property 8 tells us that $(X_2 \cup \{k\})^*$ can be put into NF, so we can also find a NF for X^* by the previous cases.

For $\neg$, note first that $\neg[a,b) = [0,a) \cup [b,\infty)$ (when $b < \infty$), and $\neg\{(a,\infty)\} = \{[0,a)\}$. Also, by De Morgan laws, $\neg(I_1 \cup \ldots \cup I_n) = \neg I_1 \cap \ldots \cap \neg I_n$ is a NF and, for any NF, $\neg X = \neg X_1 \cap \neg(X_2 + \{k\}^*)$. Then, by distributivity of $\cap$, we may restrict ourselves to the case $X_1 = [a,b)$ with $a, b \in \mathbb{R}_{>0}$ and $k \neq 0$ (the cases with other parentheses are similar). But since $X_2 \cup [0, Nk) \cup [Nk+k, \infty)$ is a finite union of rational intervals (hence we know how to write NFs for the complement of it) the following is a NF for $\neg X$:

$$[0,a) \cup [b, Nk) \cup (\neg(X_2 \cup [0,Nk) \cup [Nk+k,\infty)) + \{k\}^*) \qquad \square$$

Note that in [CG98] a weaker version of this theorem is proved, roughly saying that the set of finite unions of n-dimensional NFs forms a boolean algebra.

Though the theorem is based on the same technique that gives the normal form of regular languages over a one-letter alphabet it cannot be a simple corollary of that. Even if we restrict attention to the algebra generated by intervals with natural bounds, denote it $\mathbb{N}Int$, this has *two* generators: the point set $\{1\}$ and the nonpoint interval $(0,1)$. Neither of these generators alone may generate the whole $\mathbb{N}Int$: $\{1\}$ generates just sets with isolated points or complements of such sets (i.e. countable or co-countable sets) and hence does not generate $(0,1)$, while $(0,1)$ generates just finite unions of intervals (it cannot generate $(0,1) + \{1\}^*$).

One might also think that the result follows from Eilenberg's theory of automata with multiplicities [Ei74]. But this is not the case either since in that work star is defined via some formal power series and one cannot prove, unless defining some suitable equivalence on power series, that e.g. $[0,1)^* = [0,\infty)$.

Finally note the interesting relation which holds between the two generators of $\mathbb{N}Int$:

$$(0,1)^* = (\mathbf{0} \cup (0,1)) + \{1\}^*$$

At the end of this section we make a brief excursion into matrix theory. We construct, as in [Koz94] the Kleene algebra of matrices over $\mathcal{P}(\mathbb{R}_{\geq 0})$ whose operations are defined as follows:

$$(A \cup B)_{ij} = A_{ij} \cup B_{ij} \qquad (A + B)_{ij} = \bigcup\nolimits_{k=1}^{n} (A_{ik} + B_{kj})$$
$$A^* = \bigcup\nolimits_{n \in \mathbb{N}} nA$$

The star of matrix A can be computed by the usual Floyd-Warshall-Kleene algorithm [Con71,Ei74]: we recursively define a sequence of $n+1$ matrices $A(k)$ $(0 \leq k \leq n)$ with $A(0) = A$ and

$$A(k)_{ij} = A(k-1)_{ij} \cup (A(k-1)_{ik} + (A(k-1)_{kk})^* + A(k-1)_{kj}) \tag{9}$$

Proposition 3 ([Ei74]). *$A(n) = A^*$ for any matrix over $\mathcal{P}(\mathbb{R}_{\geq 0})$.*

Corollary 1. *If A is a matrix of normal forms then A^* is a matrix of normal forms too which can be computed by the above algorithm.*

4 Determinization and complementation of RTA

Definition 5. *An **augmented real-time automaton** (ARTA for short) over V is a tuple $\mathcal{A} = (Q, V, \delta, \lambda, \iota, Q_0, F)$ where Q, Q_0, F, δ and λ are the same as for RTA while $\iota : Q \longrightarrow \mathcal{K}(Int_{\mathbb{Q}}^{\geq 0})$ (actually ι gives a NF).*

ARTA work similarly to RTA: *runs* have the same definition and a signal σ with $dom(\sigma) = [0, e)$ is *associated to* a run of length n iff there exist $0 = t_1 \leq \ldots \leq t_{n+1} = e$ with $t_{i+1} - t_i \in \iota(q_i)$ and $\sigma(t) = \lambda(q_i)$ for all $t \in [t_i, t_{i+1})$ and all $i \in [n]$. The emptiness problem is again decidable in linear time w.r.t. $card(Q)$. Note that we need a preliminary step in which states q whose $\iota(q) = \emptyset$ are removed. Also the different notions of determinism also apply to ARTA with the same definitions; hence we will speak of state-deterministic ARTA and stuttering-free ARTA.

The following theorem says that usage of NFs instead of just intervals do not add to the expressive power of RTA:

Theorem 4. *$TReg(V)$ equals the class of languages accepted by ARTA.*

The proof is very close to the one of Theorem 1 and is based on the following property of regular expressions:

$$|[a]_{(a,b)\cup((c,d)+\{k\}^*)}| = |[a]_{(a,b)} + [a]_{(c,d)};[a]^*_{\{k\}}|$$

which can be easily extended to any atomic ETRE.

Hence we may focus on the problem of complementing ARTA, since ARTA and RTA are equally expressive by Theorem 4.

Theorem 5. *Each ARTA is equivalent to some deterministic ARTA.*

Proof. As a preliminary step, in the given ARTA we modify all states q in which $0 \in \iota(q)$ by removing 0 from $\iota(q)$ and adding some new transitions, the whole process resembling very closely to the removal of ϵ-transitions from finite automata. Also we assume that all states with empty time label have been removed.

We first achieve **stuttering freeness** by removing all stuttering transitions between a-states, for some $a \in V$, and repeating this for all the other letters in V. The idea is to find, for each pair of states (q, r) the set of positive numbers which are the duration of a signal that is associated to some run starting in q, ending in r and containing only a-states. For this we need to recursively add all the intervals of the states that may lie on such a run. This is one of the places where we apply the normal form theorem 3 and the algorithm for computing the star of a matrix of sets of positive numbers. The formalization is the following:

Start with some ARTA $\mathcal{A} = (Q, V, \delta, \lambda, \iota, Q_0, F)$ and number its transitions as $\delta = \{t_1, \ldots, t_p\}$ (denote $t_i = (in_i, out_i)$). Construct a matrix A whose elements are the sets of the a-labeled states: $A_{ij} = X$ iff $out_i = in_j$, $\lambda(out_i) = a$ and $\iota(out_i) = X$, otherwise $A_{ij} = \emptyset$. Then we add two more rows and columns to A (the $p+1$-th and the $p+2$-th) which intuitively record the time labels of initial, resp. final a-states: for $j \in [p]$ put $A_{p+1,j} = X$ for all j with $in_j \in Q_0$, $\lambda(in_j) = a$ and $\iota(in_j) = X$, otherwise $A_{p+1,j} = \emptyset$ and $A_{j,p+2} = X$ for all j with $out_j \in F$, $\lambda(out_j) = a$ and $\iota(out_j) = X\}$; moreover put $A_{p+1,p+2} = \bigcup\{X \mid \exists q \in Q_0 \cap F, \lambda(q) = a \text{ and } \iota(q) = X\}$.

Then A^* holds *the lengths of all signals associated to runs consisting of a-states only.* Hence $(A^*)_{ij}$ consists of the lengths of signals associated to runs starting in out_i, ending in in_j and consisting of a-states only. Also $(A^*)_{p+1,j}$ consists of the lengths of all runs that start in Q_0, end in in_i and consist of a-states only. Similarly for $(A^*)_{i,p+2}$ and $(A^*)_{p+1,p+2}$. Computation of A^* is done by the Floyd-Warshall-Kleene algorithm (9). Note here the importance of Corollary 1: the elements of A^* are still NFs, hence they may be used for labeling some new states of an ARTA.

Hence, while non-a-states will be preserved, the *nonempty* components of A^* will replace all a-states: their time label will be $(A^*)_{ij}$ and they will be connected only to non-a-states. Formally, denote $Q_{\neg a} = \{q \in A \mid \lambda(q) \neq a\}$ and $Q_a = \{(i,j) \mid (A^*)_{ij} \neq \emptyset\}$ and define $\mathcal{B} = (Q_{\neg a} \cup Q_a, V, \overline{\delta}, \overline{\lambda}, \overline{\iota}, T_0, T_f)$ where

$$\begin{aligned}
&\overline{\lambda}(s) = \lambda(s) \text{ for } s \in Q_{\neg a} \text{ and } \overline{\lambda}((i,j)) = a \text{ for } (i,j) \in Q_a;\\
&\overline{\iota}(s) = \iota(s) \text{ for } s \in Q_{\neg a} \text{ and } \overline{\iota}((i,j)) = (A^*)_{ij} \text{ for } (i,j) \in Q_a;\\
&T_0 = (Q_{\neg a} \cap Q_0) \cup \{(p+1,j) \in Q_a\} \cup \{(p+1,p+2) \mid A^*_{p+1,p+2} \neq \emptyset\};\\
&T_f = (Q_{\neg a} \cap F) \cup \{(i,p+2) \in Q_a\} \cup \{(p+1,p+2) \mid A^*_{p+1,p+2} \neq \emptyset\};\\
&\overline{\delta} = \{(in_i,(i,j)) \mid \lambda(in_i) \neq a\} \cup \{((i,j),out_j) \mid \lambda(out_j) \neq a\} \cup\\
&\qquad \cup (\delta \cap (Q_{\neg a} \times Q_{\neg a}))
\end{aligned}$$

The proof that $L(\mathcal{B}) = L(\mathcal{A})$ relies on Theorem 3. Note that by construction no two a-states are directly connected and by the exclusion of zeroes from intervals in the beginning all components of A^* will not contain 0. Also, all transitions between non a-states are preserved, hence no stuttering transitions are added. This shows that after applying it for all letters in V we get a stuttering-free ARTA.

The **determinization** construction is an adaptation of the subset construction. Start with some ARTA $\mathcal{B} = (Q, V, \delta, \lambda, \iota, Q_0, F)$ assumed stuttering-free. If the time labels did not count then the states of the deterministic automaton were sets of *identically state-labeled states* and we would draw a transition from some S_1 with $\lambda(S_1) = \{a\}$ to some S_2 with $\lambda(S_2) = \{b\}$ iff $S_2 = \{r \in Q \mid \exists q \in S_1 \text{ s.t. } (q,r) \in \delta\}$. Taking into account the time labels is done by splitting S_2 into several "smaller" sets such that the time labels of these parts give a partition of $\mathbb{R}_{>0}$.

Therefore we start with $\mathcal{T}$, the set of triples (S, S', a) where $a \in V$, $S' \subseteq S \subseteq Q$ with $\lambda(S) = \{a\}$. and $\overline{\lambda}((S,S',a)) = a$. Intuitively the control passes through (S,S',a) iff in $\mathcal{B}$ the control may pass through some state in S' but not through any of the states in $S \setminus S'$. Formally we associate to each $U \subseteq Q$ with $\lambda(U) = \{a\}$ the set $Tl(U) = \{X \in \mathcal{K}(Int_{\mathbb{Q}}^{\geq 0}) \mid \exists q \in U \text{ s.t. } \iota(q) = X\}$ and define

$$\overline{\iota}((S,S',a)) = \mathbb{R}_{>0} \cap \left(\bigcap Tl(S')\right) \cap \neg\left(\bigcup Tl(S \setminus S')\right)$$

where the usual conventions $\bigcap \emptyset = \mathbb{R}_{\geq 0}$ and $\bigcup \emptyset = \emptyset$ apply. Note that we put $\mathbb{R}_{>0}$ in front of $\overline{\iota}((S,S',a))$ because otherwise we lose stuttering-freeness.

It is also important to note that it is *here* where we need the result that NFs are closed under complementation, because we need to put $\overline{\iota}((S,S',a))$ in NF and $\overline{\iota}((S,S',a))$ contains complementation.

Though $\mathcal{T}$ is not what we need: it might still happen that $\bar{\iota}(S, S', a) \cap \bar{\iota}(S, S'', a) \neq \emptyset$, but only when $\bar{\iota}(S, S', a) = \bar{\iota}(S, S'', a)$ Hence we can define an equivalence on $\mathcal{T}$ as $(S, S', a) \simeq (S, S'', a)$ iff $\bar{\iota}(S, S', a) = \bar{\iota}(S, S'', a)$. The quotient set $\mathcal{T}/\simeq$ is our desired set of states as we have that for any $S \subseteq Q$ the time labels of the classes $[S, S', a]$ for all $S' \subseteq S$ give a partition of $\mathbb{R}_{>0}$.

Hence we build $\mathcal{C} = (\mathcal{T}/\simeq, V, \tilde{\delta}, \tilde{\lambda}, \tilde{\iota}, \mathcal{T}_0, \mathcal{T}_f)$, where $\tilde{\lambda}([S, S', a]) = a$ and $\tilde{\iota}([S, S', a]) = \bar{\iota}((S, S', a))$. $\tilde{\delta}$ consists of transitions from $[S, S', a]$ to each $[U, U', b]$ where $U = \{q \in Q \mid \exists r \in S' \text{ s.t. } (r, q) \in \delta \text{ and } \lambda(q) = b\}$ and $U' \subseteq U$. Case $U' = \emptyset$ stands for the situation when the length of the current state in the signal is not in any of the sets from $Tl(U)$. (Note how states $[\emptyset, \emptyset, a]$ time-labeled with $\mathbb{R}_{>0}$ play the role of the trap states in dfa). Moreover put

$$\mathcal{T}_0 = \{[S, S', a] \in \overline{Q} \mid \emptyset \neq S \subseteq Q_0, a \in V\} \cup \{[\emptyset, \emptyset, a] \mid \forall q \in Q_0, \lambda(q) \neq a\}$$
$$\mathcal{T}_f = \{[S, S', a] \in \overline{Q} \mid S' \cap F \neq \emptyset, a \in V\}$$

The proof that $\mathcal{C}$ is equivalent to $\mathcal{B}$ follows the classical pattern. □

Theorem 6. *$TRec(V)$ is closed under complementation.*

Proof. This is a corollary of the above theorem: in the ARTA $\mathcal{C}$ constructed above, each signal *is associated to a unique run that starts in T_0*. Hence, the ARTA that accepts $Sig(S) \setminus L(\mathcal{B})$ is the automaton obtained from $\mathcal{C}$ by complementing its set of final states. □

We actually have a normal form for RTAs resulting from this, namely that each RTA is equivalent to a RTA in which any circuit composed of stuttering transitions is a loop at a state labeled with a point interval, stuttering steps may only start from loop states and any chain of *different* stuttering steps has at most two transitions. An example for this is in figure 1(b).

5 Conclusions and further work

We have presented here a Kleene algebra of sets of positive numbers where elements have a finite representation and a class of real-time languages which is closed under complementation and is defined by some automata and by regular expressions.

We briefly note two problems for further study: the first one concerns the possible application of this theory to removing of circuits of silent transitions in timed automata [BDGP98], possibly using the Kleene theorem of [BP99]. The second is whether a normal form can be found for unions of intervals which have *any* bounds (not necessarily rational) and are generated from finite sets of intervals by $\cup$, complementation, concatenation and star.

Acknowledgments

I am grateful to Xu Qiwen, Paritosh Pandya and the anonymous reviewers for their helpful suggestions.

References

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

[AFH94] R. Alur, L. Fix and T.A. Henzinger. A determinizable class of timed automata, in *Computer-Aided Verification*, LNCS 818:1-13, Springer Verlag, 1994.

[AH96] R. Alur and T. Henzinger. The benefits of relaxing punctuality, *Journal of ACM*, 43:116-146, 1996.

[ACM97] E. Asarin, P. Caspi and O. Maler, A Kleene Theorem for Timed Automata, in G. Winskel (Ed.) *Proc. LICS'97*, p.160-171, 1997.

[BDGP98] 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.

[BP99] P. Bouyer and A. Petit, Decomposition and composition of timed automata, *Proceedings of ICALP'99*, LNCS series, 1999.

[CG98] Ch. Choffrut and M. Goldwurm. Timed automata with periodic clock constraints, LIAFA report, 1998.

[Con71] J. H. Conway. *Regular Algebra and Finite Machines*, Chapman and Hall, 1971.

[DW96] Dang Van Hung and Wang Ji, On the design of Hybrid Control Systems Using Automata Models, in V. Chandru and V. Vinay (Eds.) *FSTTCS*, LNCS 1180:156-167, 1996.

[DMP91] R. Dechter, I. Meiri and J. Pearl, Temporal constraint networks, *Artificial Intelligence*, 49:61-95, 1991.

[Dim98] C. Dima. *Timed Regular Expressions, Real-Time Automata and Simple Duration Calculus*, draft of a UNU/IIST report, 1998. Available at `http://funinf.math.unibuc.ro/~cdima/work/report.ps.gz`.

[Dim99] C. Dima. Automata and regular expressions for real-time languages, *Proceedings of the AFL'99 workshop*, Vasszeczeny, Hungary, 9-13 august 1999.

[Ei74] S. Eilenberg. *Automata, Languages, and Machines, Vol. A*, Academic Press, 1974.

[HRS98] T.A. Henzinger, J.-F. Raskin and P.-Y. Schobbens. The regular real-time languages, in *Proc. 25-th ICALP*, LNCS series, Springer Verlag, 1998.

[HU] John E. Hopcroft and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley/Narosa Publishing House, eighth edition, New Delhi, 1992.

[Koz94] D. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events, *Information and Computation*, 110:366-390, 1994.