On Recognizable Timed Languages

Oded Maler	Amir Pnueli
VERIMAG	NYU and Weizmann
Grenoble	New York and Rehovot
France	USA

FOSSACS 2004

Nutrition Facts

Classical (Untimed) Recognizability

Timed Languages

Deterministic Timed Automata (DTA)

Recognizable Timed Languages (REC)

DTA $\Rightarrow$ REC

REC $\Rightarrow$ DTA

Related Work and Open Problems

Classical Recognizability without Monoids

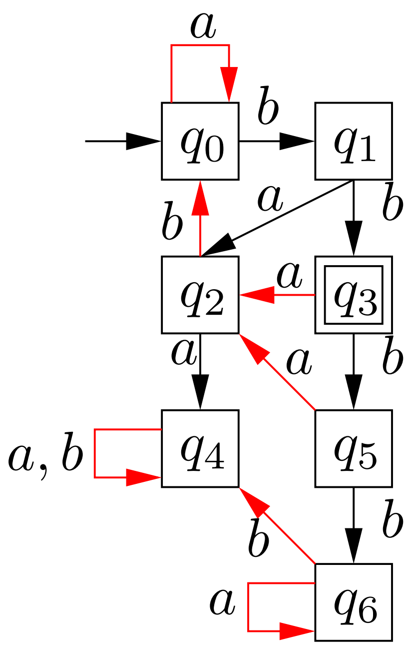
A language L is recognizable if there exists a finite prefix-closed set $P \subseteq \Sigma^*$, a "right"-morphism $\varphi : \Sigma^* \rightarrow P$ satisfying

$$\varphi(w) = w \quad \text{if } \; w \in P \qquad\qquad \varphi(w \cdot w') = \varphi(\varphi(w) \cdot w')$$

and a subset $F \subseteq P$ such that $L = \bigcup_{w \in F} \varphi^{-1}(w)$

In automata-theoretic terms the set P contains for each state a representative "minimal" word that reaches it (a spanning tree) and φ is defined via rules that rewrite words in $ext(P) = P \cdot \Sigma - P$ into words in P

Example



$$P = \{\varepsilon, b, ba, bb, baa, bbb, bbbb\} \quad F = \{bb\}$$

$$\varphi: \quad a = \varepsilon \quad bba = ba \quad bab = \varepsilon \quad bbba = ba$$

$$baaa = baab = baa \quad bbbbb = baa \quad bbbba = bbbb$$

Reading a word by the automaton is like successive rewriting into an element of P:

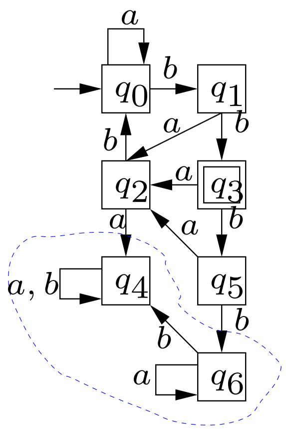
$$abbabba \longrightarrow bbabba \longrightarrow babba \longrightarrow ba$$

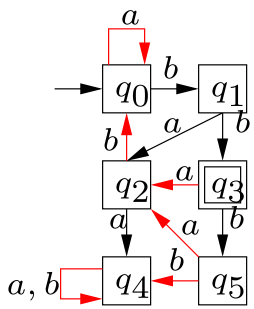
Nerode Right-congruence

$$u \sim v \text{ iff } \forall w \; uw \in L \iff vw \in L$$

Two words are equivalent if thet accept the same future.

Any automaton accepting L is inverse-homomorphic to the automaton whose states correspond to equivalence classes of $\sim$. Any set P should contain at least one representative from each equivalence class.





Timed Languages

Subsets of the time-event monoid $\mathcal{T} = \Sigma^* \uplus \mathbb{R}_+$, the free product (shuffle) of the free monoid $(\Sigma^*, \cdot, \varepsilon)$ and the commutative monoid $(\mathbb{R}_+, +, 0)$ [ACM02]

$$w = t_0 \cdot a_1 \cdot t_1 \cdot a_2 \cdot t_2 \cdots a_n \cdot t_n$$

Canonical form: $a \cdot 0 \cdot a' = a \cdot a'$ and $t \cdot \varepsilon \cdot t' = t + t'$

Untime: $\mu(w) = a_1 \cdot a_2 \cdots a_n$, Duration: $\lambda(w) = t_0 + t_1 + \cdots + t_n$

For an untimed word u, $|u|$ is the number of letters

Prefix: $u \preceq u \cdot v$ for any $u, v \in \mathcal{T}$, in particular, $w \cdot t \preceq w \cdot t'$ whenever $t \leq t'$

Finite Recognizability is Useless

Consider the language $\{5 \cdot a\}$

Its Nerode congruence has an uncountable number of classes because

$t \not\sim t'$ for every $t \neq t'$ where $t, t' < 5$

The appropriate notion is boundendess:

A language $L \subseteq \mathcal{T}$ is bounded if $\mu(L)$ is finite and $\lambda(L)$ is bounded in the usual sense of $\mathbb{R}_+$.

Elements in L have a finite number of events and a bounded duration.

We want to extend recognizability to timed languages using a bounded prefix-closed subset P of $\mathcal{T}$ and a morphism to it.

A Bad Language

Here is a language which cannot be recognized by any bounded structure

$$L_{bad} = \mathcal{T} \cdot \{a \cdot t_1 \cdot a \cdot t_2 \cdots t_n \cdot a : n \in \mathbb{N} \ \wedge \ \sum_{i=1}^{n} t_i = 1\}$$

This language (due to Alur) can be "accepted" by a non-deterministic timed automaton (but its complement cannot). To recognize it one should remember the occurrence times of unboundedly many events

For every n, $\quad a \cdot t_1 \cdots t_n \cdot a \not\sim a \cdot t_1 \cdots t_n \cdot a \cdot t_{n+1} \cdot a$

whenever $0 < \sum_{i=1}^{n+1} t_i < 1$

For for any P, $\mu(P)$ should contain the infinite language $\{a^n : n \in \mathbb{N}\}$ hence unbounded

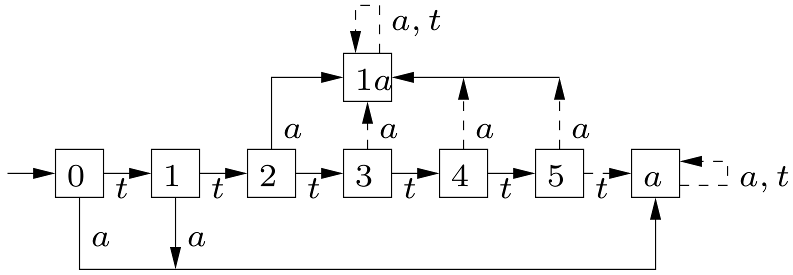
How the Timed Recognizing Structure Looks Like

$$L = \{t \cdot a \cdot w : t \in [2,5], w \in \mathcal{T}\}$$

All timed words with no letters until 2 and an occurrence of a in $[2,5]$.

$$P = \{t : t \in [0,5]\} \cup \{a\} \cup \{2 \cdot a\}, \quad F = \{2 \cdot a\}$$

$$\varphi : \quad \{t \cdot a = a : t \in [0,2)\} \quad \{t \cdot a = 2 \cdot a : t \in [2,5]\}$$
$$a \cdot a = a \cdot t = a \quad 2 \cdot a \cdot a = 2 \cdot a \cdot t = 2 \cdot a \quad 5 \cdot t = a$$



Timed Automata are n-tuples

A timed automaton is $\mathcal{A} = (\Sigma, Q, X, q_0, I, \Delta, F)$, Q: a set of states, X: a set of clocks,

I: staying condition (invariant), assigning to every q a conjunction I_q of inequalities of the form $x \leq u$, for some clock x and integer u

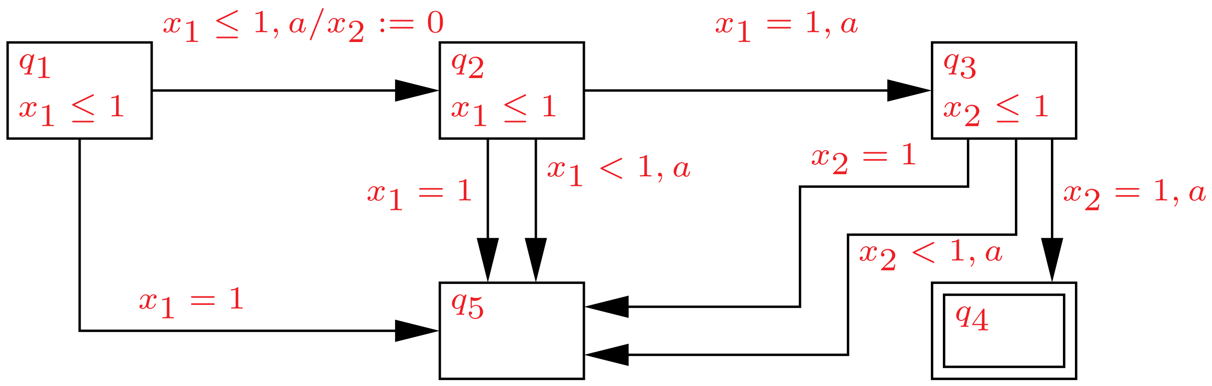
Δ: a transition relation consisting of tuples (q, a, ϕ, ρ, q') where q and q' are states, $a \in \Sigma \cup \{\varepsilon\}$

$\rho \subseteq C$ is the set of clocks reset by the transition, and

ϕ (the transition guard) is a conjunction of formulae of the form $x \geq l$ for some clock x and integer l.

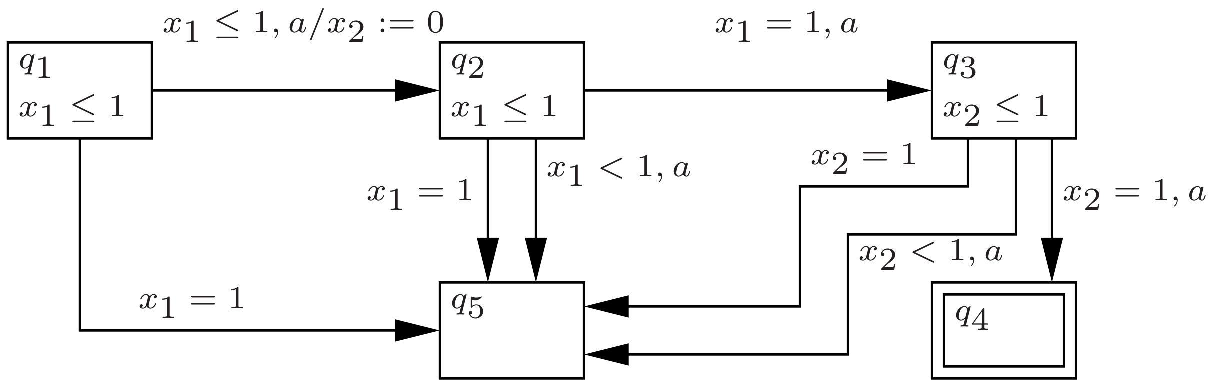
A clock valuation is a function $\mathbf{v} : C \to \mathbb{R}_+$ and a configuration is a pair $(q, \mathbf{v})$ consisting of a discrete state (location) and a clock valuation.

Example



The automaton accepts any word with 3 a's such that the second occurs 1 time after the beginning and the third — 1 time after the first.

Runs



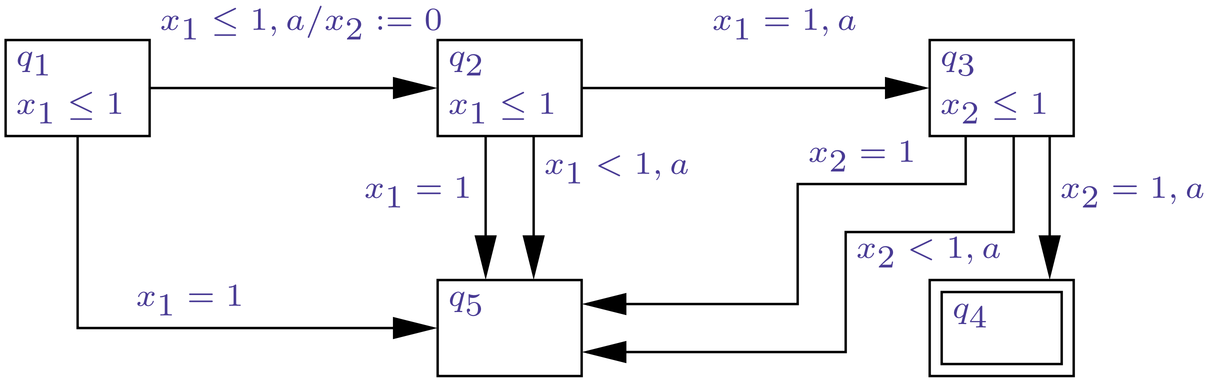
2	$(q_1, 0, 0) \xrightarrow{1} (q_1, 1, 1) \xrightarrow{\varepsilon} (q_5, 1, 1)$
$0.4a0.7$	$(q_1, 0, 0) \xrightarrow{0.4} (q_1, 0.4, 0.4) \xrightarrow{a} (q_2, 0.4, 0) \xrightarrow{0.6} (q_2, 1, 0.6) \xrightarrow{\varepsilon} (q_5, 1, 0.6)$
$0.4a0.3a$	$(q_1, 0, 0) \xrightarrow{0.4} (q_1, 0.4, 0.4) \xrightarrow{a} (q_2, 0.4, 0) \xrightarrow{0.3} (q_2, 0.7, 0.3) \xrightarrow{a} (q_5, 0.7, 0.3)$
$0.4a0.6a0.5$	$(q_1, 0, 0) \xrightarrow{0.4} (q_1, 0.4, 0.4) \xrightarrow{a} (q_2, 0.4, 0) \xrightarrow{0.6} (q_2, 1, 0.6) \xrightarrow{a} (q_3, 1, 0.6) \xrightarrow{0.4}$ $(q_3, 1.4, 1) \xrightarrow{\varepsilon} (q_5, 1.4, 1)$
$0.4a0.6a0.4a$	$(q_1, 0, 0) \xrightarrow{0.4} (q_1, 0.4, 0.4) \xrightarrow{a} (q_2, 0.4, 0) \xrightarrow{0.6} (q_2, 1, 0.6) \xrightarrow{a} (q_3, 1, 0.6) \xrightarrow{0.4}$ $(q_3, 1.4, 1) \xrightarrow{a} (q_4, 1.4, 1)$

Deterministic Timed Automata (DTA)

For every two distinct transitions $(q, a, \phi_1, \rho_1, q_1)$ and $(q, a, \phi_2, \rho_2, q_2)$, ϕ_1 and ϕ_2 have an empty intersection (event determinism)

For every transition $(q, \varepsilon, \phi, \rho, q') \in \Delta$, the intersection of ϕ with I_q is, at most, a singleton(time determinism)

In deterministic automata any word is carried by exactly one run



The Region Automaton I

The uncountable clock space can be partitioned into equivalence classes called regions. Given n clocks and an integer constant κ, an (n, κ)-condition is either one of

$$x_i = d, \quad d < x_i < d+1, \quad x_i - x_j = c \ \text{ or } \ c < x_i - x_j < c+1$$

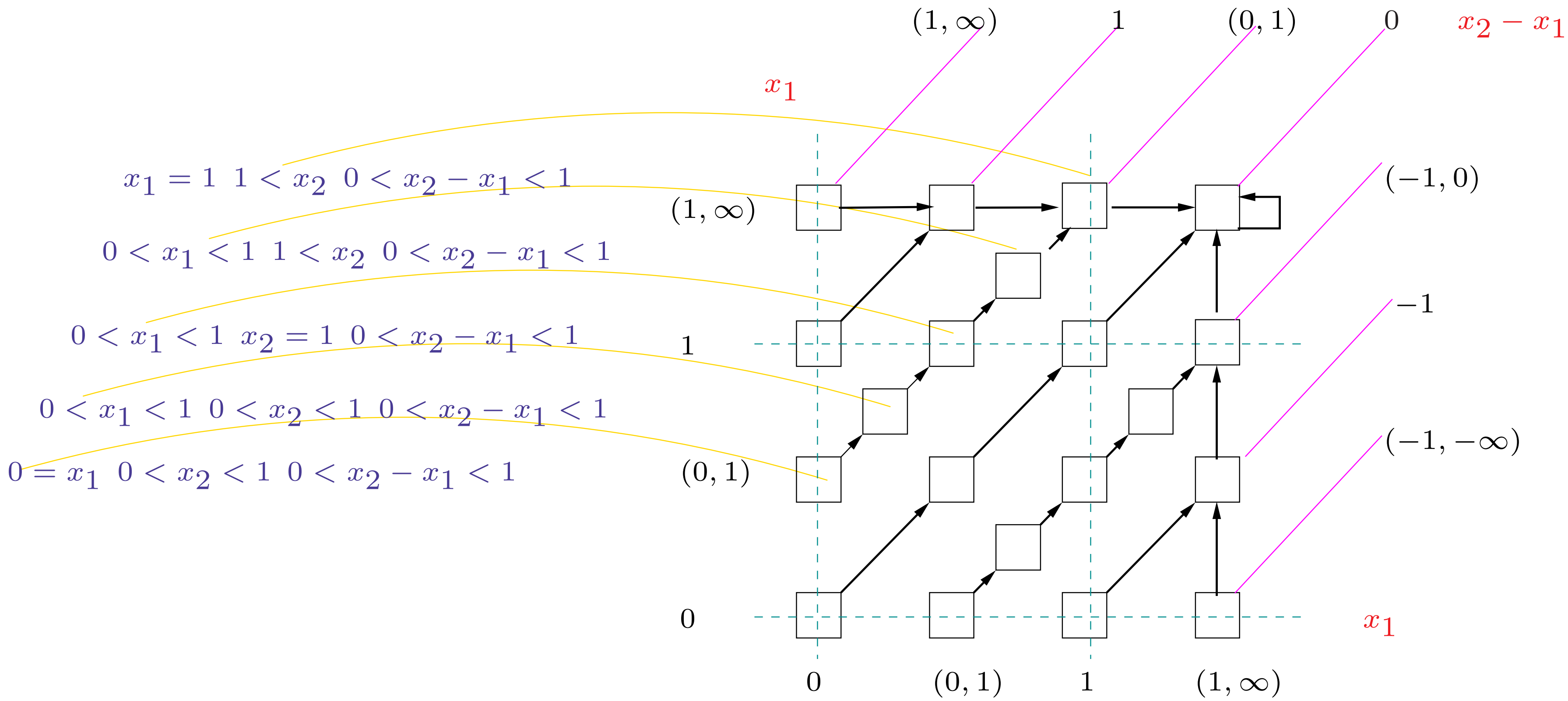
for integers c, d such that $d \in [0, \kappa]$ and $c \in [-\kappa, \kappa]$

Two clock valuation $\mathbf{x}, \mathbf{x}' \in \mathbb{R}_+^n$ are region equivalent if they agree on any (n, κ)-condition. The following holds for such **x** and $\mathbf{x}'$:

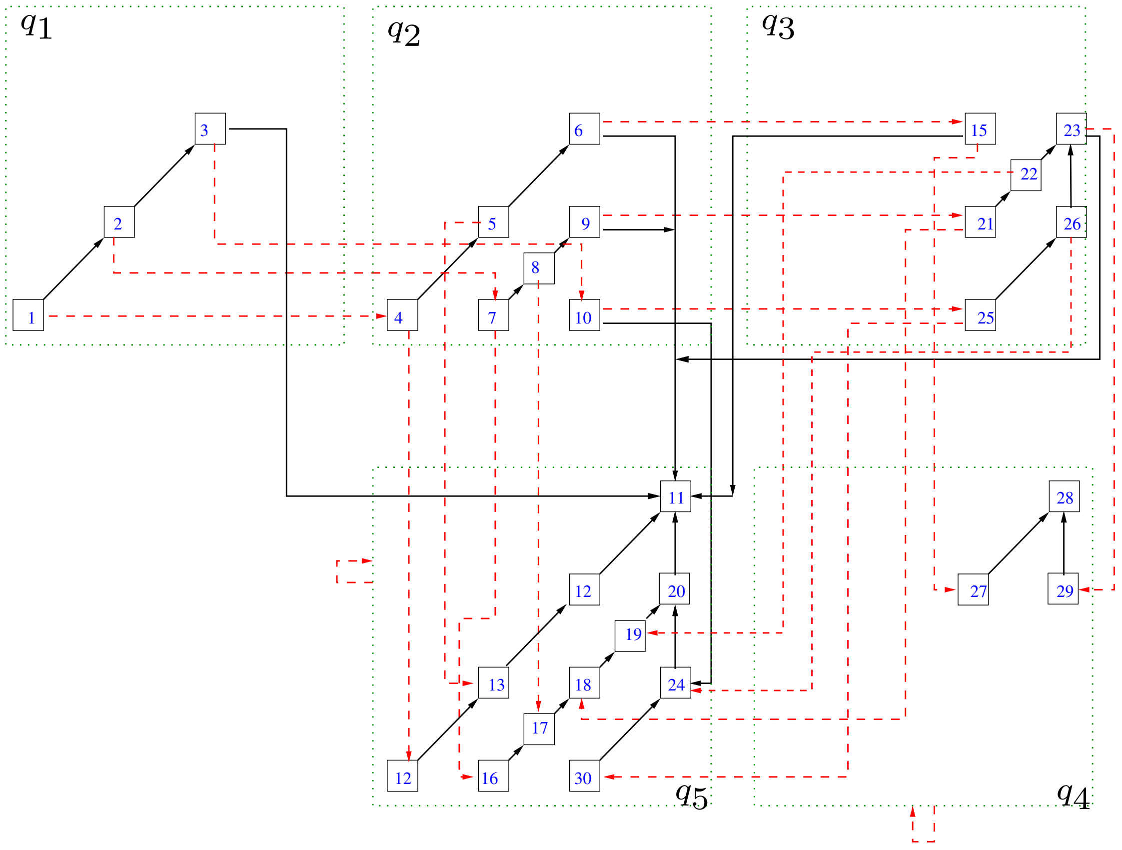
1) They satisfy the same guards and staying conditions
2) By letting time pass from both, the same next region is encountered
3) Applying a reset to both leads to the same region

The Region Automaton II

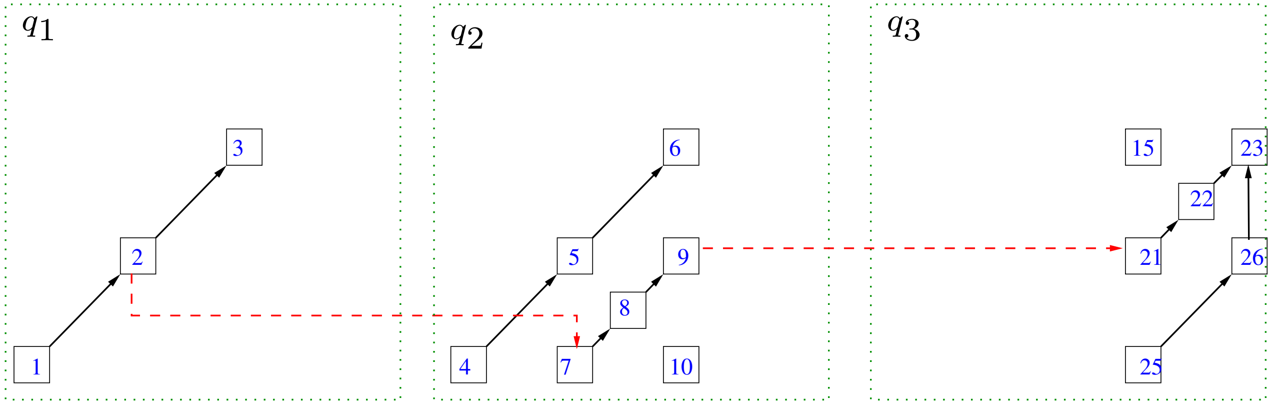
The dynamics of time for $n = 2$, $\kappa = 1$



The Region Automaton III



Associating Timed Words with Regions



Which words reach region R_{21}?

$$\{t_0 \cdot a \cdot t_1 \cdot a \cdot t_2 : 0 < t_0 < 1, 0 < t_1 < 1, t_2 = 0\}$$

Recognizable Timed Languages I

$T_n = \{t_0, \ldots, t_n\}$: a set of non-negative real variables.

A contiguous sum over T_n is $S_{j..k} = t_j + t_{j+1} + \cdots + t_{k-1} + t_k$

A timed inequality on T_n is a condition of the form $S_{i..j} \in J$ where J is an interval.

A timed condition is a conjunction of timed inequalities.

A timed language L is elementary if $\mu(L) = \{u\}$ with $u = a_1 \cdots a_n$ and the set $\{(t_0, \ldots t_n) : t_0 \cdot a_1 \cdots a_n \cdot t_n \in L\}$ is definable by a timed condition Λ.

We denote such a language by (u, Λ)

Recognizable Timed Languages II

The immediate exterior $ext(L)$ of an elementary language $L = (u, \Lambda)$:

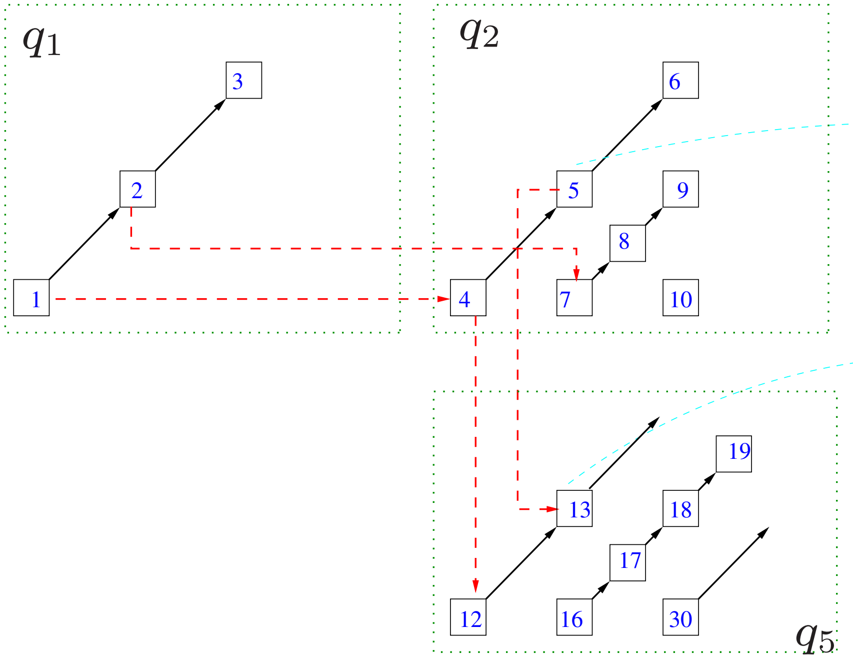
For every $a \in \Sigma$, $ext^a(L)$ is $(u \cdot a, \Lambda^a)$ where $\Lambda^a = \Lambda \cup \{t_{n+1} = 0\}$

$ext^t(L) = (u, \Lambda^t)$ where Λ^t is obtained from Λ as follows. If Λ contains one or more equality constraints of the form $S_{j..n} = d$, these constraints are replaced by constraints of the form $d < S_{j..n}$. Otherwise, let j be the smallest number such that a constraint of the form $S_{j..n} < d$ appears in Λ. This constraint is replaced by $S_{j..n} = d$ (region intuition)

A subset P of $\mathcal{T}$ is chronometric if it can be written as a finite union of disjoint elementary languages

Morphism - Intuition

What happens to the words when, after an a, region R_5 goes to R_{13} (already reached from R_{12})? How to express $R_5 \cdot a = R_{12} \cdot t = R_{13}$?



$$R_5 a = \{t_0 a t_1 a t_2 : t_0 = 0, 0 < t_1 < 1, t_2 = 0\}$$

$$R_{13} = \{t_0 a t_1 \cdot a t_2 : t_0 = 0, t_1 = 0, 0 < t_2 < 1\}$$

Morphism - Definition

A chronometric (relational) morphism Φ from $\mathcal{T}$ to bounded and prefix-closed subset P is a relation definable by a finite set of tuples $(u, \Lambda, u', \Lambda', E)$ such that each $(u, \Lambda) \in ext(P)$, each (u', Λ') is an elementary language contained in P, and E is a set of equalities of the form $\sum_{i=j}^{n} t_i = \sum_{i=k}^{n'} t'_i$, where $n = |u|$ and $n' = |u'|$

All (u, Λ) are disjoint and their union is equal to $ext(P)$

For every $w = t_0 \cdot a_1 \cdots a_n \cdot t_n$ and $w' = t'_0 \cdot b_1 \cdots b_{n'} \cdot t'_{n'}$, $(w, w') \in \Phi$ iff there exists a tuple $(u, \Lambda, u', \Lambda', E)$ in Φ such that $w \in (u, \Lambda)$, $w' \in (u', \Lambda')$ and the respective time values for w and w' satisfy the equalities in E

for words outside $ext(P)$, $\Phi(u \cdot v) = \Phi(\Phi(u) \cdot v)$

Example

$$(u, \Lambda) = (t_0 \cdot a \cdot t_1 \cdot a \,, \{0 < t_0 < 1, 0 < t_1 < 1, 0 < t_0 + t_1 < 1\})$$

$$(u', \Lambda') = (r_0 \cdot a \cdot r_1 \cdot a \cdot r_2 \,, \{0 < r_0 < 1, r_1 = 0, 0 < r_2 < 1 \,, 0 < r_0 + r_1 + r_2 < 1\})$$

$$E \quad = \quad t_0 + t_1 = r_0 + r_1 + r_2$$

We can derive from Φ a (functional) chronometric morphism $\varphi : \mathcal{T} \rightarrow P$

$\varphi(t_0 \cdot a \cdot t_1 \cdot a) = t_0 \cdot a \cdot a \cdot t_1$ or

$$\varphi(t_0 \cdot a \cdot t_1 \cdot a) = a \cdot a \cdot (t_0 + t_1)$$

Recognizable timed Languages (REC)

A timed language L is recognizable if there is a chronometric prefix-closed set P, a chronometric subset F of P and a chronometric relational morphism $\Phi : \mathcal{T} \to P$ such that $L = \bigcup_{w \in F} \Phi^{-1}(w)$

The main result of this work: the recognizable languages are exactly those accepted by deterministic timed automata

The main opinion of this work: these are the interesting timed languages

From Deterministic Automata to Recognizable Languages

The idea: take a spanning tree of the region automaton and assign an elementary timed language to every region in a prefix-closed manner

The relation between the words in the language and the elements of the region is done via a clock binding

A clock binding over (X, T_n) is a function β associating with every clock x a contiguous sum of the form $S_{j..n}$

A region is a conjunction of single clock constraints and clock difference constraints. By substituting $\beta(x)$ for x, the former become timed inequalities and the latter become inequalities on $S_{j..n} - S_{k..n} = S_{j..k}$ which are timed inequalities as well

More Formally

We associate with each region a triple (u, Λ, β) where (u, Λ) is an elementary timed language with $|u| = n$ and β is a clock binding on (X, T_n).

We decompose Λ into two sets of timed inequalities:

Λ_-: “anachronistic” inequalities not involving t_n

Λ_+: “live” constraints involving t_n

Transitions may change the binding and may move some inequalities from Λ_+ to Λ_-

For the initial region $\mathcal{R}_0 = (q, \mathbf{0})$, $u = \varepsilon$, $\Lambda = \Lambda_+$ is $t_0 = 0$, and all the clocks are bound to t_0

The Inductive Step I

Given a region $\mathcal{R}$ with (u, Λ, β) we compute (u', Λ', β') for its successor (via a spanning transition) $\mathcal{R}'$. Two cases:

$\mathcal{R}'$ is a time successor of $\mathcal{R}$: $u' = u$ and $\beta' = \beta$. We let $\Lambda'_- = \Lambda_-$ and obtain Λ'_+ from the region formula ψ' by replacing every clock x by $\beta(x)$

$R_8 = (q_2, 0 < x_2 < 1, 0 < x_1 < 1, 0 < x_1 - x_2 < 1)$, $u = a$
$\beta(x_1) = t_0 + t_1 \; \beta(x_2) = t_1 \qquad \Lambda = 0 < t_1 < 1, 0 < t_0 + t_1 < 1 < 1$

$R_9 = (q_2, 0 < x_2 < 1, x_1 = 1, 0 < x_1 - x_2 < 1)$, $u = a$
$\beta(x_1) = t_0 + t_1 \; \beta(x_2) = t_1 \qquad \Lambda = 0 < t_1 < 1, 0 < t_0 + t_1 = 1$

The Inductive Step II

If $\mathcal{R}'$ is a transition successor of $\mathcal{R}$ via an a-labeled transition: $u' = u \cdot a \cdot t_{n+1}$, the (T^{n+1}, X) binding β' is derived from β and from the corresponding transition as follows. If a clock x is not reset by the transition then $\beta'(x) = S_{i..n+1}$ whenever $\beta(x) = S_{i..n}$. If x is reset then $\beta'(x) = t_{n+1}$

To compute Λ'_- we add to Λ_- the substitution of $\beta(x)$ for x in ψ and let Λ'_+ be the substitution of $\beta'(x)$ in ψ'. Example: $R_2 \cdot a = R_7$

$R_2 = (q_1, 0 < x_1 = x_2 < 1)$, $u = \varepsilon$
$\beta(x_1) = \beta(x_2) = t_0 \qquad \Lambda = 0 < t_0 < 1$

$R_7 = (q_2, 0 < x_1 < 1, x_2 = 0$, $u = a$
$\beta(x_1) = t_0 + t_1, \beta(x_2) = t_1 \qquad \Lambda = 0 < t_0 < 1, t_1 = 0$

And the Non-spanning Transitions

Consider a transition from $\mathcal{R}$ with characteristic (u, Λ, β) into $\mathcal{R}'$ with characteristic (u', Λ', β')

Let $(u'', \Lambda'', \beta'')$ be the result of applying the above procedure.

Add to Φ the tuple $(u'', \Lambda'', u', \Lambda', E)$:

For each clock x which is not reset by the transition, E contains the equality $\beta'(x) = \beta''(x)$

If x is reset by the transition, then E contains the equality $\beta'(x) = 0$

QED

The Other Direction I

Given P, Φ and F construct from it a deterministic timed automaton such that two words w, w' reach the same configurations only if $(w, w') \in \Phi$

We use a variant of TA where in addition to resets one can have assignment of the form $x_i := x_j$ (same expressive power [SV96])

Start with a tree-like untimed automaton whose states are $\mu(P)$ and $\delta(u, a) = u \cdot a$ whenever $u \cdot a$ is in $\mu(P)$

Decorate with resets, staying conditions and guards

With every transition we reset a new clock so that for every word $t_0 a_1 \cdots a_n t_n$, the value of clock x_i at any state $a_1 \cdots a_j$, $i \leq j$ is bound to $S_{i..j}$

The Other Direction II

For every state $u = a_1 \cdots a_n \in \mu(P)$ let

$$\Lambda(u) = \{(t_0, \ldots, t_n) : t_0 \cdot a_1 \cdots a_n \cdot t_n \in P\}.$$

decompose $\Lambda(u)$ into anachronistic (Λ_-) and live (Λ_+) constraints and substitute x_i instead of every $S_{i..n}$ in Λ_+ to obtain the staying condition for state u

For every u and a such that $u \cdot a$ is in $\mu(P)$ let

$$H_{u,a} = \{(t_0, \ldots, t_n) : t_0 \cdot a_1 \cdots a_n \cdot t_n a \in P\}.$$

Every expression $S_{i..j}$ in it can be replaced by $x_j - x_i$ and the whole condition can becomes the guard of the transition between u and $u \cdot a$

The Other Direction III

So far we have an automaton in which every element of P reaches a distinct configuration.

Consider an element $(u \cdot a, \Lambda, u', \Lambda', E) \in \Phi$, such that $(u \cdot a, \Lambda) \in ext^a(P)$ and $|u| = n$ and $|u'| = n'$

For every constraint of the form $S_{j..k} \in J$ included in Λ, add to the transition guard the constraint $x_j - x_k \in J$

For every equality $S'_{j..n'} = S_{k..n}$ included in E, we add to the reset function the assignment $x_j := x_k$.

$ext^t(P)$ is treated in a similar way

QED

Related Work

Event-clock automata [Alur, Fix Henxinger] (more restricted)

[Henzinger, Raskin, Schobbens] (more general but very complicated)

[Asarin, Caspi, Maler]: timed regular expression, time-event monoid (treatment of time different than timed traces)

[Springintveld and Vaandrager]: semantic minimization of time automata

[Bouyer, Petit, Therien]: data languages, monoids, too general

[Tripakis]: testing, diagnosability and synthesis with partial information

Open Questions

What if contiguous sums are replaced by sums (stopwatch automata) or by (positive) linear combinations ("linear" hybrid automata)?

Any matching syntax (fragment of timed logic, timed regular expressions)?

Application to learning and minimization?

Relation to the growth of the size of discretized automaton as a function of the time step?