Real-Time Systems

Lecture 18: The Universality Problem of Timed Büchi Automata

2018-01-23

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

A Theory of Timed Automata [1]

Rajeev Alur [2] David L. Dill [3]
Computer Science Department, Stanford University
Stanford, CA 94305.

Abstract. We propose *timed (finite) automata* to model the behavior of real time systems over time. Our definition provides a simple, and yet powerful, way to annotate state transition graphs with timing constraints using finitely many real valued *clocks*. A timed automaton accepts *timed words* — infinite sequences in which a real valued time of occurrence is associated with each symbol. We study timed automata from the perspective of formal language theory: we consider closure properties, decision problems, and subclasses. We consider both nondeterministic and deterministic transition structures, and both Büchi and Muller acceptance con ditions. We show that nondeterministic timed automata are closed under union and intersection, but not under complementation, whereas deterministic timed Muller automata are closed under all Boolean operations. The main construction of the paper is an (PSPACE) algorithm for checking the emptiness of the language of a (nondeterministic) timed automaton. We also prove that the universality problem and the language inclusion problem are solvable only for the deterministic automata: both problems are undecidable (Π^1_1 hard) in the nondeterministic case and PSPACE complete in the deterministic case. Finally, we discuss the application of this theory to automatic verification of real time requirements of finite state systems.

Keywords: Real time systems, automatic verification, formal languages and au tomata theory.

[1] Preliminary versions of this paper appear in the *Proceedings of the 17th International Colloquium on Automata, Languages, and Programming* (1990), and in the *Proceedings of the REX workshop "Real-time: theory in practice"* (1991).

[2] Current address: AT&T Bell Laboratories, 600 Mountain Avenue, Room 2D-144, Murray Hill, NJ 07974.

[3] Supported by the National Science Foundation under grant MIP-8858807, and by the United States Navy, Office of the Chief of Naval Research under grant N00014-91-J-1901. This publication does not necessarily reflect the position or the policy of the U.S. Government, and no official endorsement of this work should be inferred.

Alur and Dill (1994)

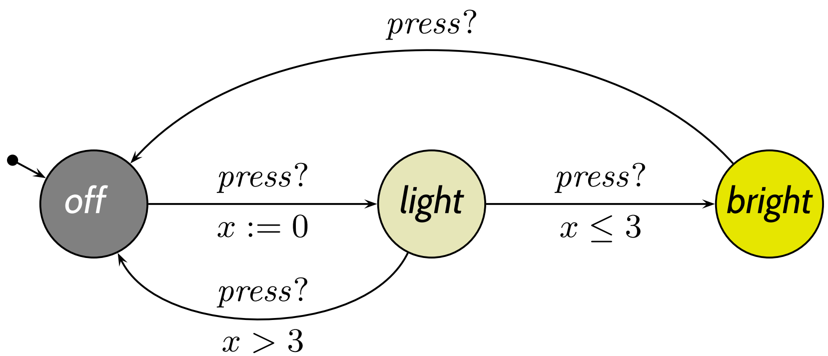
Content

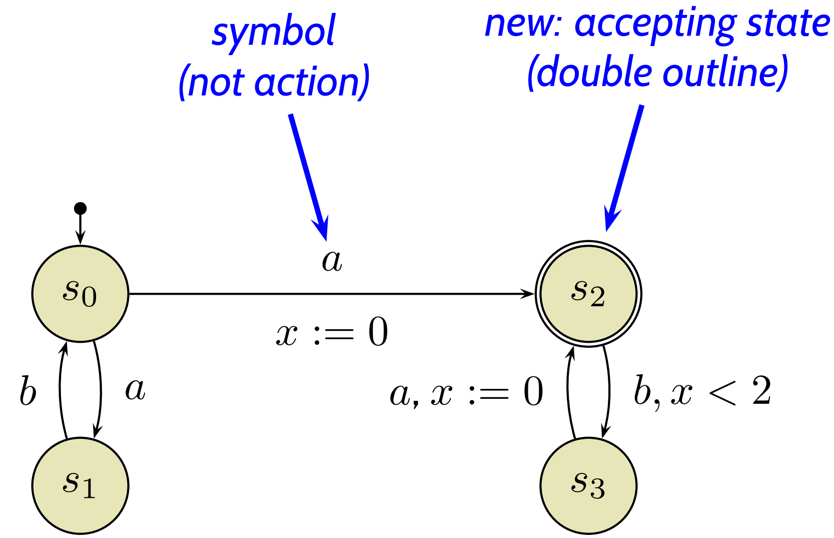
- **Timed Büchi Automata**
  - vs. **Pure/Extended Timed Automata**
  - **timed word**, **timed language**
  - **accepting** TBA runs
  - **language** of a TBA
- **The Universality Problem of TBA**
  - **definition**: universality problem
  - **undecidability claim**
  - **proof idea**: 2-counter machines again
  - **construct observer** for non-recurring computations
- **Consequences**
  - the **language inclusion** problem
  - the **complementation** problem
- **Beyond Timed Regular**

Timed Büchi Automata

Alur and Dill (1994)

$\ldots$ vs. Timed Automata





timed automaton $\mathcal{A}$ **induces**
computation paths and **runs** such as

$$\xi = \langle \mathit{off}, 0 \rangle, 0 \xrightarrow{1} \langle \mathit{off}, 1 \rangle, 1 \xrightarrow{press?} \langle \mathit{light}, 0 \rangle, 1 \xrightarrow{3} \langle \mathit{light}, 3 \rangle, 4 \xrightarrow{press?} \langle \mathit{bright}, 3 \rangle, 4 \rightarrow \ldots$$

Behaviour of $\mathcal{A}$:
set of computation paths / runs.

Timed Büchi Automaton $\mathcal{A}$ **accepts**
timed words such as

$$(a, 1), (b, 2), (a, 3), (b, 4), (a, 5), (b, 6), \ldots$$

Language of $\mathcal{A}$:
set of accepted timed words.

Timed Languages

Definition. A **time sequence** $\tau = \tau_1, \tau_2, \ldots$ is an infinite sequence of time values $\tau_i \in \mathbb{R}_0^+$, satisfying the following constraints:

(i) **Monotonicity**: τ increases strictly monotonically, i.e. $\tau_i < \tau_{i+1}$ for all $i \geq 1$.

(ii) **Progress**: For every $t \in \mathbb{R}_0^+$, there is some $i \geq 1$ such that $\tau_i > t$.

Definition. A **timed word** over an alphabet Σ is a pair (σ, τ) where

- $\sigma = \sigma_1, \sigma_2, \dots \in \Sigma^\omega$ is an infinite word over Σ, and
- τ is a time sequence.

Definition. A **timed language** over an alphabet Σ is a set of timed words over Σ.

Example: Timed Language

Timed word over alphabet Σ: a pair (σ, τ) where

- $\sigma = \sigma_1, \sigma_2, \ldots$ is an infinite word over Σ, and
- τ is a time sequence (strictly (!) monotonic, non-Zeno).

$(ab)^*$ infinite sequence abab...

$$\Sigma = \{a, b\}$$

$$L_{crt} = \{((ab)^\omega, \tau) \mid \exists\, i \in \mathbb{N}^+\ \forall\, j \geq i : (\tau_{2j} < \tau_{2j-1} + 2)\}$$

$\tau_{2j} - \tau_{2j-1} < 2$

$$\begin{array}{cccccccccccccc}
\sigma = & a & b & a & b & a & b & \ldots & a & b & a & b & \cdots \\
 & \tau_1 & \tau_2 & \tau_3 & \tau_4 & \tau_5 & \tau_6 & & \tau_{2i-1} & \tau_{2i} & \cdots & & \\
 & \underbrace{3}_{\tau_{2j-1}} & \underbrace{6}_{\tau_{2j}} & 9 & 12 & 15 & 18 & \cdots & 6i-3 & 6i & \cdots & &
\end{array}$$

3

$\notin L_{crt}$

Example: Timed Language

Timed word over alphabet Σ: a pair (σ, τ) where

- $\sigma = \sigma_1, \sigma_2, \dots$ is an infinite word over Σ, and
- τ is a time sequence (strictly (!) monotonic, non-Zeno).

$$\Sigma = \{a, b\}$$

$$L_{crt} = \{((ab)^\omega, \tau) \mid \exists\, i \in \mathbb{N}^+ \, \forall\, j \geq i : (\tau_{2j} < \tau_{2j-1} + 2)\}$$

i $\quad j \longrightarrow$

$$\begin{array}{lccccccccccc} \sigma = & a & b & a & b & a & b & \dots & a & b & a & b & \cdots \\ & \tau_1 & \tau_2 & \tau_3 & \tau_4 & \tau_5 & \tau_6 & & \tau_{2i-1} & \tau_{2i} & \cdots & & \\ & 3 & 6 & 13 & 27 & 27.1 & 29 & \cdots & 100 & 100.2 & \cdots & & \end{array}$$

Timed Büchi Automata

not simple! (negation is in, clock difference not)

Definition. The set $\Phi(X)$ of **clock constraints** over X is defined inductively by

$$\delta ::= x \leq c \mid c \leq x \mid \neg\delta \mid \delta_1 \wedge \delta_2, \qquad \text{where } x \in X, c \in \mathbb{Q}.$$

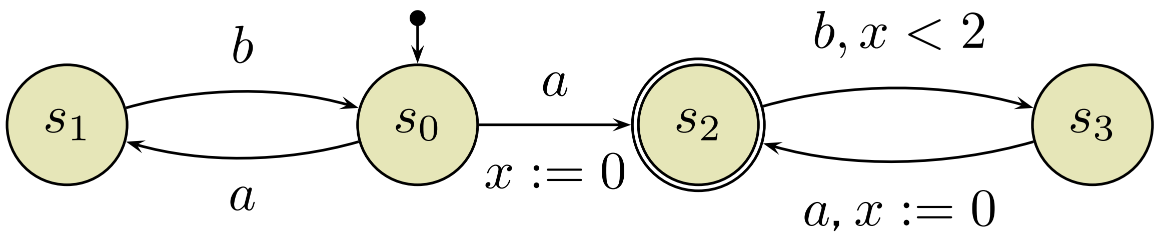
Definition.
A **timed Büchi automaton** (TBA) $\mathcal{A}$ is a tuple $(\Sigma, S, S_0, X, E, F)$, where

- Σ is an **alphabet**,
- S is a finite set of **states**, $S_0 \subseteq S$ is a set of **start states**,
- X is a finite set of **clocks**, and
- $E \subseteq S \times S \times \Sigma \times 2^X \times \Phi(X)$ gives the set of **transitions**.

  An **edge** $(s, s', a, \lambda, \delta)$ represents a **transition** from state s to state s' on input symbol a. The set $\lambda \subseteq X$ gives the clocks to be reset with this transition, and δ is a clock constraint over X.
- $F \subseteq S$ is a set of **accepting states**.

Example: TBA

$$\mathcal{A} = (\Sigma, S, S_0, X, E, F)$$
$$(s, s', a, \lambda, \delta) \in E$$



$\Sigma = \{a, b\}$

$S = \{s_0, s_1, s_2, s_3\}$ $\quad F = \{s_2\}$

$S_0 = \{s_0\}$

$X = \{x\}$

$E = \{ (s_0, s_1, a, \emptyset, true), \dots \}$

(Accepting) TBA Runs

Definition. A **run** r, denoted by $(\bar{s}, \bar{\nu})$, of a TBA $(\Sigma, S, S_0, X, E, F)$ over a timed word (σ, τ) is an **infinite** sequence of the form

$$r : \langle s_0, \nu_0 \rangle \xrightarrow[\tau_1]{\sigma_1} \langle s_1, \nu_1 \rangle \xrightarrow[\tau_2]{\sigma_2} \langle s_2, \nu_2 \rangle \xrightarrow[\tau_3]{\sigma_3} \dots$$

with $s_i \in S$ and $\nu_i : X \to \mathbb{R}_0^+$, satisfying the following requirements:

(Accepting) TBA Runs

Definition. A **run** r, denoted by $(\bar{s}, \bar{\nu})$, of a TBA $(\Sigma, S, S_0, X, E, F)$ over a timed word (σ, τ) is an **infinite** sequence of the form

$$r : \langle s_0, \nu_0 \rangle \xrightarrow[\tau_1]{\sigma_1} \underbrace{\langle s_1, \nu_1 \rangle \xrightarrow[\tau_2]{\sigma_2} \langle s_2, \nu_2 \rangle}_{} \xrightarrow[\tau_3]{\sigma_3} \dots$$

with $s_i \in S$ and $\nu_i : X \to \mathbb{R}_0^+$, satisfying the following requirements:

- **Initiation**: $s_0 \in S_0$ and $\nu(x) = 0$ for all $x \in X$.
- **Consecution**: for all $i \geq 1$, there is $(s_{i-1}, s_i, \sigma_i, \lambda_i, \delta_i)$ in E such that
  - $(\nu_{i-1} + (\tau_i - \tau_{i-1}))$ satisfies δ_i, and
  - $\nu_i = (\nu_{i-1} + (\tau_i - \tau_{i-1}))[\lambda_i := 0]$.

(Accepting) TBA Runs

Definition. A **run** r, denoted by $(\bar{s}, \bar{\nu})$, of a TBA $(\Sigma, S, S_0, X, E, F)$ over a timed word (σ, τ) is an **infinite** sequence of the form

$$r : \langle s_0, \nu_0 \rangle \xrightarrow[\tau_1]{\sigma_1} \langle s_1, \nu_1 \rangle \xrightarrow[\tau_2]{\sigma_2} \langle s_2, \nu_2 \rangle \xrightarrow[\tau_3]{\sigma_3} \dots$$

with $s_i \in S$ and $\nu_i : X \to \mathbb{R}_0^+$, satisfying the following requirements:

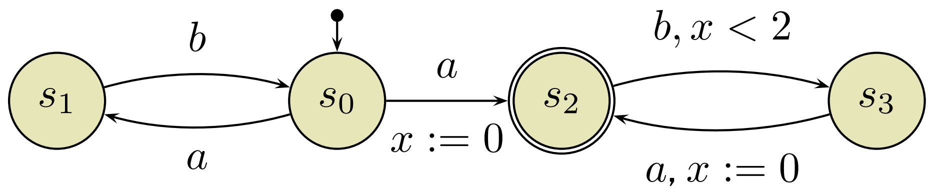
- **Initiation**: $s_0 \in S_0$ and $\nu(x) = 0$ for all $x \in X$.
- **Consecution**: for all $i \geq 1$, there is $(s_{i-1}, s_i, \sigma_i, \lambda_i, \delta_i)$ in E such that
  - $(\nu_{i-1} + (\tau_i - \tau_{i-1}))$ satisfies δ_i, and
  - $\nu_i = (\nu_{i-1} + (\tau_i - \tau_{i-1}))[\lambda_i := 0]$.

The set $\mathit{inf}(r) \subseteq S$ consists of those states $s \in S$ such that $s = s_i$ for infinitely many $i \geq 0$.

Definition. A run $r = (\bar{s}, \bar{\nu})$ of a TBA over timed word (σ, τ) is called (an) **accepting** (run) if and only if $\mathit{inf}(r) \cap F \neq \emptyset$.

Example: (Accepting) Runs

$r : \langle s_0, \nu_0 \rangle \xrightarrow[\tau_1]{\sigma_1} \langle s_1, \nu_1 \rangle \xrightarrow[\tau_2]{\sigma_2} \langle s_2, \nu_2 \rangle \xrightarrow[\tau_3]{\sigma_3} \dots$ initial and $(s_{i-1}, s_i, \sigma_i, \lambda_i, \delta_i) \in E$, s.t. $(\nu_{i-1} + (\tau_i - \tau_{i-1})) \models \delta_i, \nu_i = (\nu_{i-1} + (\tau_i - \tau_{i-1}))[\lambda_i := 0]$. Accepting iff $\mathit{inf}(r) \cap F \neq \emptyset$.



Timed word: $(a, 1), (b, 2), (a, 3), (b, 4), (a, 5), (b, 6), \dots$

- Can we construct **any run**? Is it accepting?

$r : \langle s_0, 0 \rangle \xrightarrow[1]{a} \langle s_2, 0 \rangle \xrightarrow[2]{b} \langle s_3, 1 \rangle \xrightarrow[3]{a} \langle s_2, 0 \rangle \dots$ $\quad \mathit{inf}(r) = \{s_3, s_2\} \cap \{s_2\} \neq \emptyset$ ✓

- Can we construct a **non-run**?

No. BUT $(a,1), (b,10), (a,11), (b,12), \dots$ $\quad \langle s_0, 0 \rangle \xrightarrow[1]{a} \langle s_2, 0 \rangle$ ↯ got stuck

$\langle s_0, 0 \rangle \xrightarrow[1]{a} \langle s_1, 1 \rangle \xrightarrow[10]{b} \langle s_0, 10 \rangle \xrightarrow[11]{a} \langle s_2, 0 \rangle \dots$

- Can we construct a **(non-)accepting run**?

$\langle s_0, 0 \rangle \xrightarrow[1]{a} \langle s_1, 1 \rangle \xrightarrow[2]{b} \langle s_0, 2 \rangle \xrightarrow[3]{a} \langle s_1, 3 \rangle \dots$

Definition. For a TBA $\mathcal{A}$,
the **language** $L(\mathcal{A})$ of timed words it accepts is defined to be the set

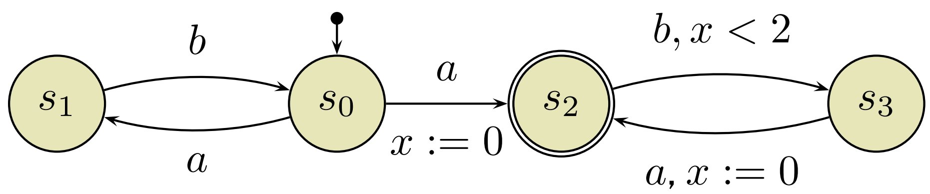
$$\{(\sigma, \tau) \mid \mathcal{A} \text{ has an accepting run over } (\sigma, \tau)\}.$$

For short: $L(\mathcal{A})$ is the **language of $\mathcal{A}$**.

Definition. A timed language L is a **timed regular language** if and only if $L = L(\mathcal{A})$ for **some** TBA $\mathcal{A}$.

Example: Language of a TBA

$$L(\mathcal{A}) = \{(\sigma, \tau) \mid \mathcal{A} \text{ has an accepting run over } (\sigma, \tau)\}.$$



Claim: $L(\mathcal{A}) = L_{crt}$ $(= \{((ab)^\omega, \tau) \mid \exists\, i\, \forall\, j \geq i : (\tau_{2j} < \tau_{2j-1} + 2)\})$

- $(\sigma, \tau) \in L(\mathcal{A}) \implies (\sigma, \tau) \in L_{crt}$: ✓

- $(\sigma, \tau) \in L_{crt} \implies (\sigma, \tau) \in L(\mathcal{A})$: ✓

Question: Is L_{crt} timed regular or not? YES

The Universality Problem is Undecidable for TBA

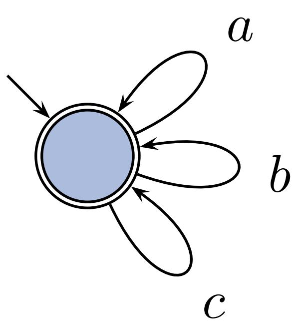
Alur and Dill (1994)

The Universality Problem

- **Given:** A TBA $\mathcal{A}$ over alphabet Σ.
- **Question:** Does $\mathcal{A}$ **accept all timed words** over Σ?

  In other words: Is $L(\mathcal{A}) = \{(\sigma, \tau) \mid \sigma \in \Sigma^\omega, \tau$ time sequence$\}$.

- **Obvious examples exist**: Let $\Sigma = \{a, b, c\}$, then



  accepts all timed words over Σ.
- In general **not that obvious**.

The Universality Problem

- **Given:** A TBA $\mathcal{A}$ over alphabet Σ.
- **Question:** Does $\mathcal{A}$ **accept all timed words** over Σ?

  In other words: Is $L(\mathcal{A}) = \{(\sigma, \tau) \mid \sigma \in \Sigma^\omega, \tau$ time sequence$\}$.

Theorem 5.2. The problem of deciding whether a timed automaton over alphabet Σ accepts all timed words over Σ is Π_1^1-hard.

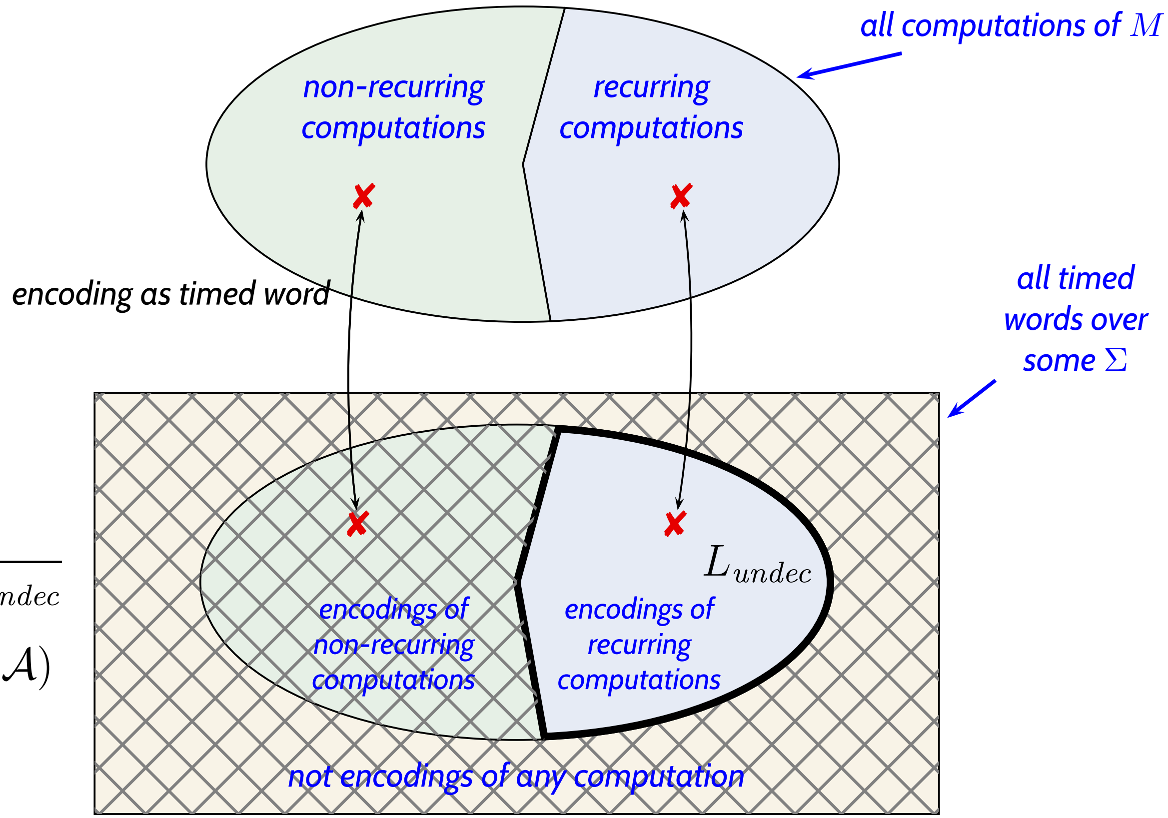
("The class Π_1^1 consists of highly undecidable problems, including some nonarithmetical sets (for an exposition of the analytical hierarchy consult, for instance [Rogers, 1967].)

Recall: With classical (untimed) Büchi Automata, this is different:

- Let $\mathcal{B}$ be a **Büchi Automaton** over Σ.
- $\mathcal{B}$ is **universal** if and only if $\overline{L(\mathcal{B})} = \emptyset$.
- $\mathcal{B}'$ such that $L(\mathcal{B}') = \overline{L(\mathcal{B})}$ is **effectively computable**.
- **Language emptyness** is **decidable** for Büchi Automata.

Proof Idea

- **2-counter machines** (once again).



: $\overline{L_{undec}}$ $= L(\mathcal{A})$

- Consider a language L_{undec} consisting of the **recurring** computations of a **2-counter machine** M.
- Construct a TBA $\mathcal{A}$ from M which accepts **the complement** of L_{undec}, i.e. with $L(\mathcal{A}) = \overline{L_{undec}}$.
- Then $\mathcal{A}$ is universal if and only if L_{undec} is empty… …if and only if M **doesn't have** a recurring computation.
- Thus if **universality** of TBA would be decidable, we had a decision procedure for **recurrence** of 2-counter machines.

Once Again: Two Counter Machines (Different Flavour)

A **two-counter machine** M

- has two **counters** C, D and
- a finite **program** consisting of n **instructions** $\{b_1, \dots, b_n\}$.

  An **instruction increments** or **decrements** one of the counters, or **jumps**, here even **non-deterministically**.

A **configuration** of M is a triple $\langle i, c, d \rangle \in \{1, \dots, n\} \times \mathbb{N}_0 \times \mathbb{N}_0$:

- **program counter** $i \in \{1, \dots, n\}$,
- **values** $c, d \in \mathbb{N}_0$ **of counters** C and D.

A **computation** of M is an infinite, initial, consecutive sequence

$$\langle 1, 0, 0 \rangle = \langle i_0, c_0, d_0 \rangle, \langle i_1, c_1, d_1 \rangle, \langle i_2, c_2, d_2 \rangle, \dots \text{ where}$$

- $\langle i_0, c_0, d_0 \rangle = \langle 1, 0, 0 \rangle$,
- $\langle i_{j+1}, c_{j+1}, d_{j+1} \rangle$ is a result **executing instruction** b_{i_j} at $\langle i_j, c_j, d_j \rangle$ for all $j \in \mathbb{N}_0$.

A computation of M is called **recurring** iff $i_j = 1$ for infinitely many $j \in \mathbb{N}_0$.

Step 1: Choose Alphabet

- **Given**: Let M be a 2-counter machine with n instructions $\{b_1, \dots, b_n\}$.
- **Wanted**: a Timed Büchi Automaton $\mathcal{A}$ which accepts timed words which **do not** encode a **recurring** computation of M.

  That is, $\mathcal{A}$ should accept the complement of the set of timed words which do encode a **recurring** computation of M.
- **Choose** alphabet $\Sigma = \{b_1, \dots, b_n, a_1, a_2\}$.
- A configuration

$$\langle i, c, d \rangle \in \{1, \dots, n\} \times \mathbb{N}_0 \times \mathbb{N}_0$$

  of M is **represented** by the letter sequence

$$b_i \underbrace{a_1 \dots a_1}_{c \text{ times}} \underbrace{a_2 \dots a_2}_{d \text{ times}} = b_i a_1^c a_2^d$$

Construction Idea

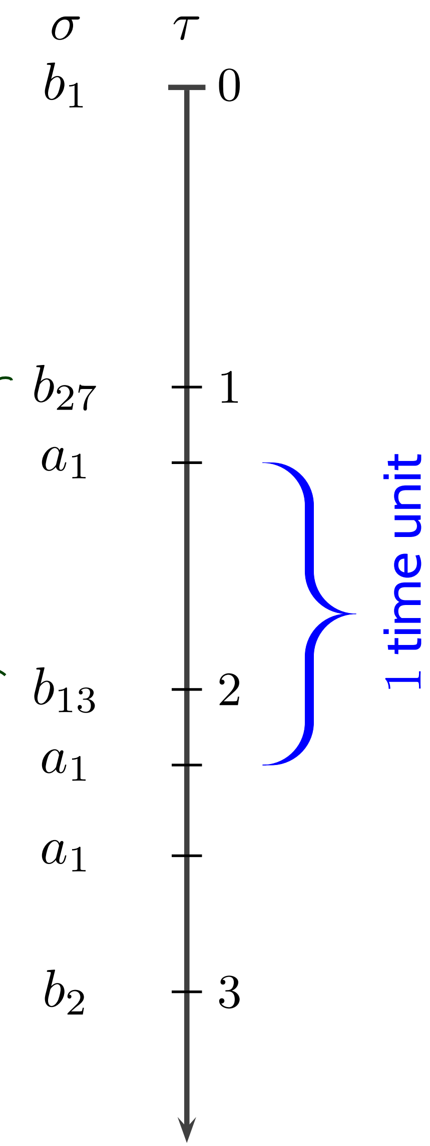
(σ, τ) **is in** L_{undec} iff:

- $\sigma = b_{i_1} a_1^{c_1} a_2^{d_1} b_{i_2} a_1^{c_2} a_2^{d_2} \ldots$, **and**
- the prefix of σ with times $0 \leq t < 1$ encodes configuration $\langle 1, 0, 0 \rangle$, **and**
- the time of b_{i_j} is j, **and**
- For all $j \in \mathbb{N}_0$,
  - ~~the time of b_{i_j} is j~~
  - if $c_{j+1} = c_j$: for every a_1 at time t in the interval $[j, j+1]$ there is an a_1 at $t+1$,
  - if $c_{j+1} = c_j + 1$: for every a_1 at time t in the interval $[j+1, j+2]$, except for the last one, there is an a_1 at time $t-1$,
  - if $c_{j+1} = c_j - 1$: for every a_1 at time t in the interval $[j, j+1]$, except for the last one, there is an a_1 at time $t+1$,

  and analogously for the a_2's, **and**
- $\langle i_1, c_1, d_1 \rangle, \langle i_2, c_2, d_2 \rangle, \ldots$ is a recurring computation of M, thus b_1 occurs infinitely often.

$\langle b_{27}, 1, 0 \rangle$

$\langle b_{13}, 2, 0 \rangle$



Construction Idea

(σ, τ) **is in** L_{undec} iff:

- $\sigma = b_{i_1} a_1^{c_1} a_2^{d_1} b_{i_2} a_1^{c_2} a_2^{d_2} \dots$, **and**
- the prefix of σ with times $0 \leq t < 1$ encodes configuration $\langle 1, 0, 0 \rangle$, **and**
- the time of b_{i_j} is j, **and**
- For all $j \in \mathbb{N}_0$,
  - the time of b_{i_j} is j.
  - if $c_{j+1} = c_j$: for every a_1 at time t in the interval $[j, j+1]$ there is an a_1 at $t+1$,
  - if $c_{j+1} = c_j + 1$: for every a_1 at time t in the interval $[j+1, j+2]$, except for the last one, there is an a_1 at time $t-1$,
  - if $c_{j+1} = c_j - 1$: for every a_1 at time t in the interval $[j, j+1]$, except for the last one, there is an a_1 at time $t+1$,

  and analogously for the a_2's, **and**
- $\langle i_1, c_1, d_1 \rangle, \langle i_2, c_2, d_2 \rangle, \dots$ is a recurring computation of M, thus b_1 occurs infinitely often.

(σ, τ) **is not in** L_{undec} (i.e. $(\sigma, \tau) \in \overline{L_{undec}}$) iff:

(i) the prefix of σ with times $0 \leq t < 1$ **doesn't** encode $\langle 1, 0, 0 \rangle$, **or**

(ii) b_i at time $j \in \mathbb{N}$ is **missing**, **or** there is a spurious b_i at time $t \in]j, j+1[$, **or**

(iii) the configuration encoded in

$$[j+1, j+2[$$

doesn't faithfully represent the **effect of instruction** b_{i_j} on the configuration encoded in $[j, j+1[$, **or**

(iv) the timed word is not recurring, i.e. it has only **finitely many** b_i.

Step 2: Construct "Observer" for $\overline{L_{undec}}$

Wanted: A TBA $\mathcal{A}$ such that

$$L(\mathcal{A}) = \overline{L_{undec}},$$

i.e., $\mathcal{A}$ accepts a timed word (σ, τ) if and only if $(\sigma, \tau) \notin L_{undec}$.

Plan: Construct a TBA

- $\mathcal{A}_0$ for case (ii)
  [missing b_i at time j, or spurious b_i],
- $\mathcal{A}_{init}$ for case (i)
  [initial configuration not encoded],
- $\mathcal{A}_{recur}$ for case (iv)
  [not recurring], and
- $\mathcal{A}_i$ for each instruction b_i for case (iii)
  [instruction effect not encoded].

Then set

$$\mathcal{A} = \mathcal{A}_0 \cup \mathcal{A}_{init} \cup \mathcal{A}_{recur} \cup \bigcup_{1 \leq i \leq n} \mathcal{A}_i$$

Step 2.(ii): Construct $\mathcal{A}_0$

(ii) The b_i at time $j \in \mathbb{N}$ is missing, or there is a spurious b_i at time $t \in]j, j+1[$.

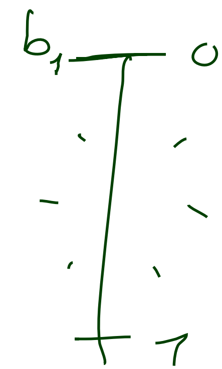
Alur and Dill (1994): "It is easy to construct such a timed automaton."

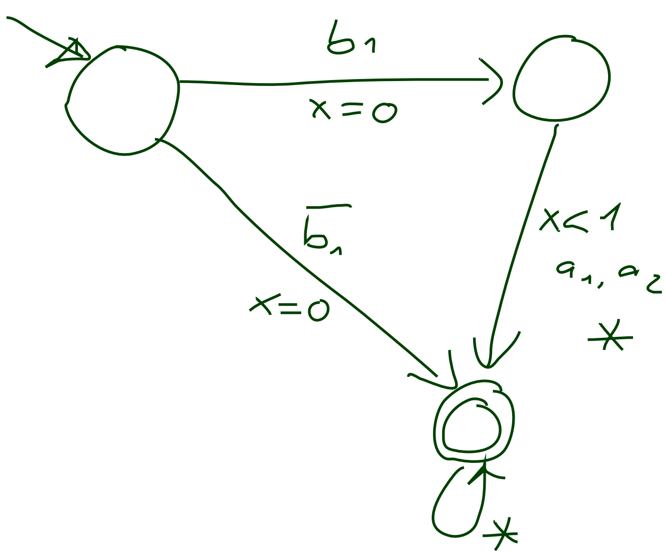
Step 2.(i): Construct $\mathcal{A}_{init}$

(i) The prefix of the timed word with times $0 \leq t < 1$ doesn't encode $\langle 1, 0, 0 \rangle$.

- It accepts

$$\{(\sigma_j, \tau_j)_{j \in \mathbb{N}_0} \mid (\sigma_0 \neq b_1) \vee (\tau_0 \neq 0) \vee (\tau_1 \neq 1)\}.$$

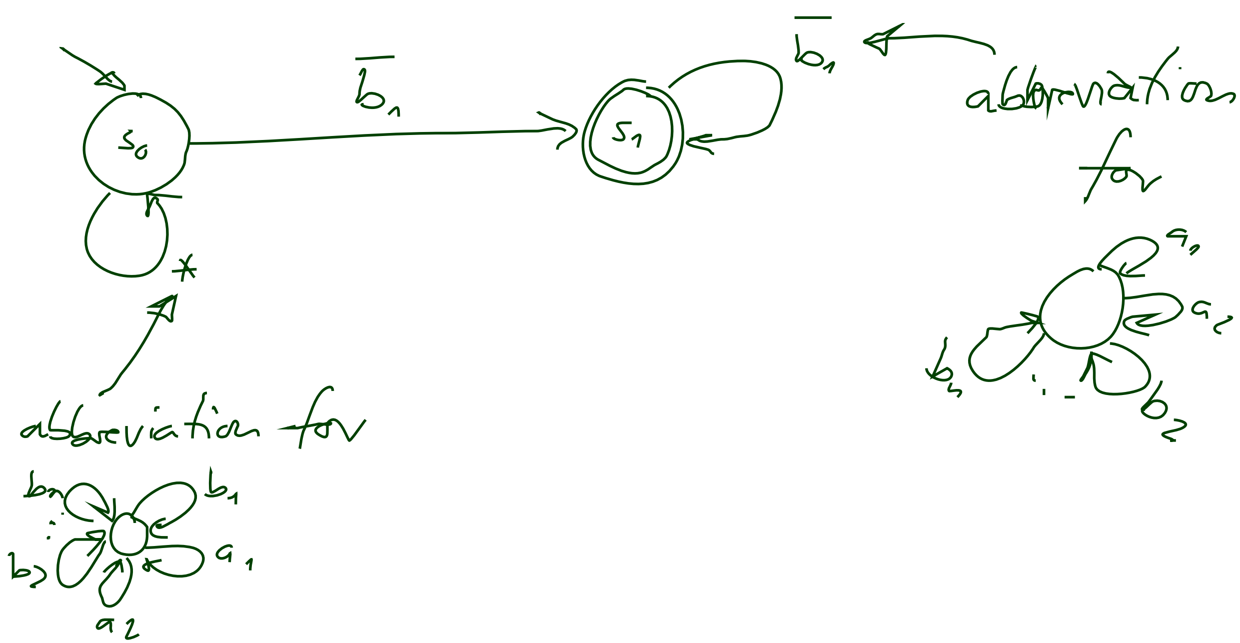




Step 2.(iv): Construct $\mathcal{A}_{recur}$

(iv) The timed word is not recurring, i.e. it has only finitely many b_1.

- $\mathcal{A}_{recur}$ accepts words with only finitely many b_1.



Step 2.(iii): Construct $\mathcal{A}_i$

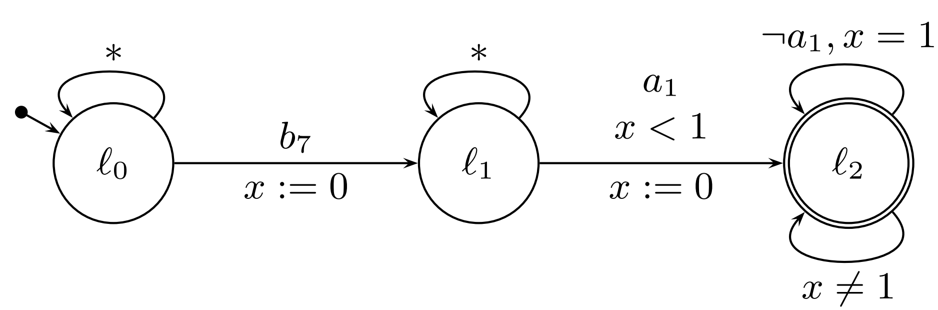
(iii) The configuration encoded in $[j+1, j+2[$ doesn't faithfully represent the effect of instruction b_i on the configuration encoded in $[j, j+1[$.

Example: assume instruction 7 is:

Increment counter D and jump non-deterministically to instruction 3 or 5.

Once again: stepwise. $\mathcal{A}_7$ is $\mathcal{A}_7^1 \cup \dots \cup \mathcal{A}_7^6$.

- $\mathcal{A}_7^1$ accepts words with b_7 at time j but neither b_3 nor b_5 at time $j+1$.
  "Easy to construct."
- $\mathcal{A}_7^2$ is



- $\mathcal{A}_7^3$ accepts words which encode unexpected change of counter C.
- $\mathcal{A}_7^4, \dots, \mathcal{A}_7^6$ accept words with missing increment of D.

Content

- **Timed Büchi Automata**
  - vs. **Pure/Extended Timed Automata**
  - **timed word**, **timed language**
  - **accepting** TBA runs
  - **language** of a TBA
- **The Universality Problem of TBA**
  - **definition**: universality problem
  - **undecidability claim**
  - **proof idea**: 2-counter machines again
  - **construct observer** for non-recurring computations
- **Consequences**
  - the **language inclusion** problem
  - the **complementation** problem
- **Beyond Timed Regular**

Aha, And...?

Consequences: Language Inclusion

- **Given:** Two TBAs $\mathcal{A}_1$ and $\mathcal{A}_2$ over alphabet B.
- **Question:** Is $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$?

Possible applications of a decision procedure:

- Characterise the **allowed behaviour** as $\mathcal{A}_2$ and model **design behaviour** as $\mathcal{A}_1$.
- Automatically decide $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$, that is,
  whether the **behaviour of the design** is a subset of the **allowed behaviour**.
- If yes, design is correct wrt. requirement.

- If **language inclusion** was decidable, then we could use it to decide universality of $\mathcal{A}$ by checking
  $$\mathcal{L}(\mathcal{A}_{univ}) \subseteq \mathcal{L}(\mathcal{A})$$
  where $\mathcal{A}_{univ}$ is **any** universal TBA (which is easy to construct).

Consequences: Complementation

- **Given:** A timed regular language W over B

  (that is, there is a TBA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = W$).
- **Question:** Is $\overline{W}$ timed regular?

Possible applications of a decision procedure:

- Characterise the **allowed behaviour** as $\mathcal{A}_2$ and model **design behaviour** as $\mathcal{A}_1$.
- Automatically construct $\mathcal{A}_3$ with $L(\mathcal{A}_3) = \overline{L(\mathcal{A}_2)}$ and check

  $$L(\mathcal{A}_1) \cap L(\mathcal{A}_3) = \emptyset,$$

  that is, whether the design has any non-allowed behaviour.
- Taking for granted that:
  - The intersection automaton is effectively computable.
  - The emptyness problem for Büchi automata **is decidable**.
    (Proof by construction of region automaton Alur and Dill (1994).)

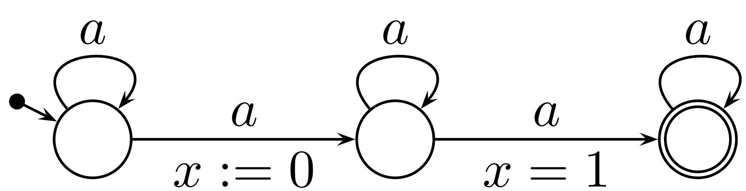
Consequences: Complementation

- **Given:** A timed regular language W over B

  (that is, there is a TBA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = W$).
- **Question:** Is $\overline{W}$ timed regular?

- If the class of timed regular languages were closed under **complementation**, "the complement of the inclusion problem is recursively enumerable. This contradicts the Π^1_1-hardness of the inclusion problem." Alur and Dill (1994)

A **non-complementable TBA** $\mathcal{A}$:



$$\mathcal{L}(\mathcal{A}) = \{(a^\omega, (t_i)_{i \in \mathbb{N}_0}) \mid \exists\, i \in \mathbb{N}_0\ \exists\, j > i : (t_j = t_i + 1)\}$$

Complement language:

$$\overline{\mathcal{L}(\mathcal{A})} = \{(a^\omega, (t_i)_{i \in \mathbb{N}_0}) \mid \text{no two } a \text{ are separated by distance 1}\}.$$

Content

- **Timed Büchi Automata**
  - vs. **Pure/Extended Timed Automata**
  - **timed word**, **timed language**
  - **accepting** TBA runs
  - **language** of a TBA
- **The Universality Problem of TBA**
  - **definition**: universality problem
  - **undecidability claim**
  - **proof idea**: 2-counter machines again
  - **construct observer** for non-recurring computations
- **Consequences**
  - the **language inclusion** problem
  - the **complementation** problem
- **Beyond Timed Regular**

Beyond Timed Regular

Beyond Timed Regular

With clock constraints of the form

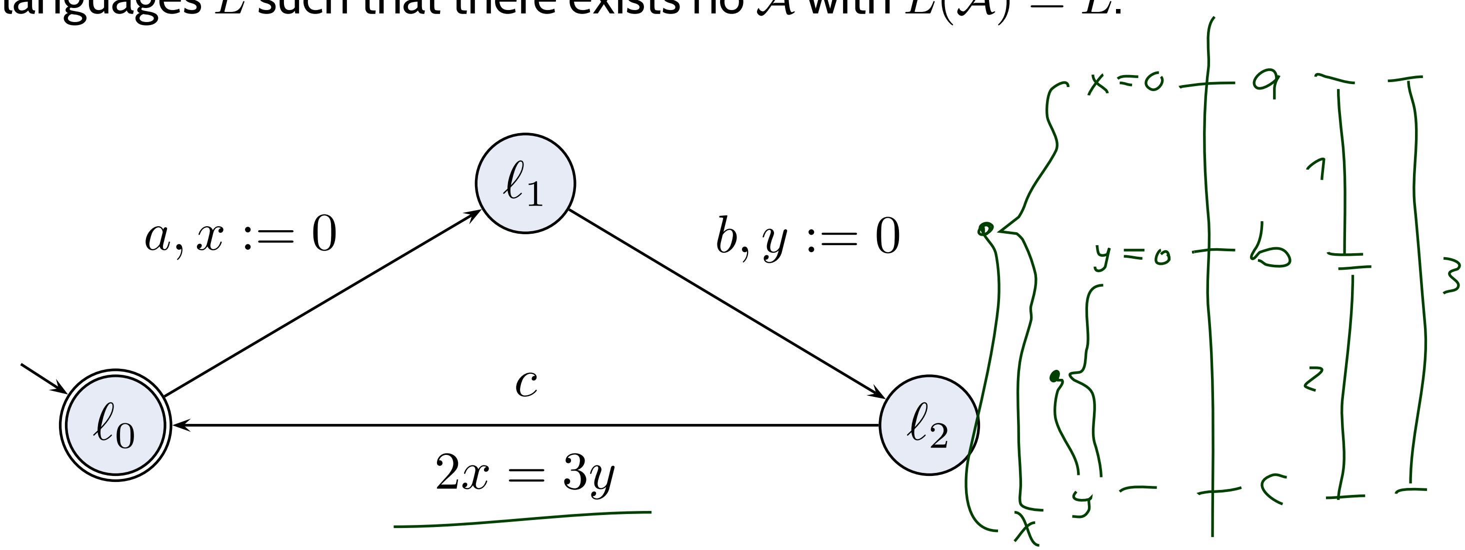
$$x + y \le x' + y'$$

we can describe timed languages which are **not timed regular**.

In other words:

- There are strictly more timed languages than timed regular languages.
- There exists timed languages L such that there exists no $\mathcal{A}$ with $L(\mathcal{A}) = L$.

Example:



$$\{((abc)^\omega, \tau) \mid \forall\, j \,.\, (\tau_{3j} - \tau_{3j-1}) = 2(\tau_{3j-1} - \tau_{3j-2})\}$$

Content

- **Timed Büchi Automata**
  - vs. **Pure/Extended Timed Automata**
  - **timed word**, **timed language**
  - **accepting** TBA runs
  - **language** of a TBA
- **The Universality Problem of TBA**
  - **definition**: universality problem
  - **undecidability claim**
  - **proof idea**: 2-counter machines again
  - **construct observer** for non-recurring computations
- **Consequences**
  - the **language inclusion** problem
  - the **complementation** problem
- **Beyond Timed Regular**

Tell Them What You've Told Them...

- **Timed Büchi Automata** accept timed words,
  Pure / Extended Timed Automata
  "produce" computation paths.
  - Different views on the same phenomenon.
- A **set of timed words** L is called **timed regular**
  if there exists a TBA whose language is L.
- **Decidability** results for Timed Büchi Automata
  - **Emptyness**: **decidable** (region construction)
  - **Universality**: **undecidable** (2-counter automata)
  - **Language Inclusion**: **undecidable** (universality)
  - **Complementation**: **undecidable** (non-compl'able TBA)
- **Beyond Timed Regular**
  - with more expressive **clock constraints**,
  - automata can accept **non-timed regular languages**.

References

References

Alur, R. and Dill, D. L. (1994). A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235.

Olderog, E.-R. and Dierks, H. (2008). *Real-Time Systems - Formal Specification and Automatic Verification*. Cambridge University Press.