Model Checking Real-Time Systems

Vincent Trélat

Technical University of Munich

December 12, 2022

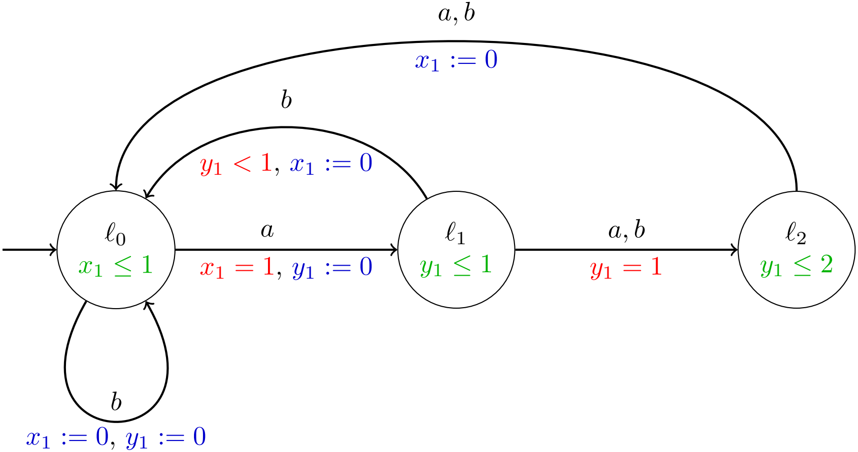
Definition 1

A *Timed Automaton* (TA) $\mathcal{A}$ is the tuple $(L, \ell_0, C, \Sigma, I, E)$ where:

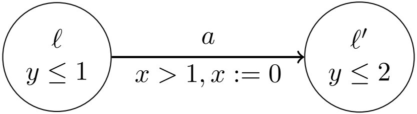
- L is a finite set of *locations* with initial location $\ell_0 \in L$
- C is a finite set of *clocks*
- Σ is a finite set of *actions*
- $I \colon L \to \Phi(C)$ is an *invariant mapping*
- $E \subseteq L \times \Phi(C) \times \Sigma \times 2^C \times L$ is a set of edges
- a set F of *target locations* is generally specified

Edges are denoted by $\ell \xrightarrow{\varphi,a,r} \ell'$.

Example of a TA with 3 locations, 2 clocks and 2 actions (letters):

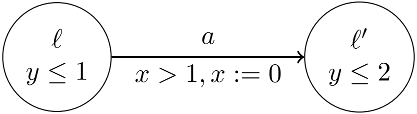


Operational Semantics



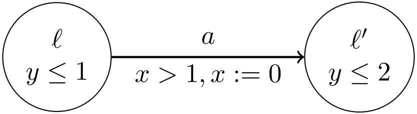
 Timed Automata

Operational Semantics



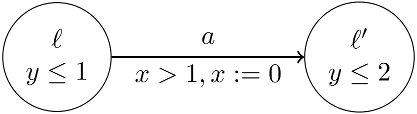
$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases}$$

Operational Semantics



$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v')$$

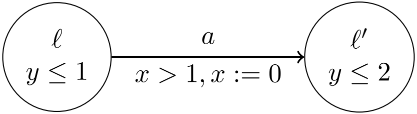
Operational Semantics



$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.9 \end{cases}$$

 Timed Automata

Operational Semantics

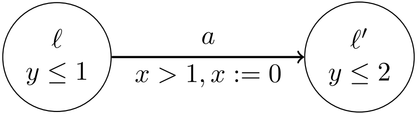


$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.9 \end{cases}$$

$$\begin{cases} v(x) = 1.2 \\ v(y) = 0.2 \end{cases}$$

 Timed Automata

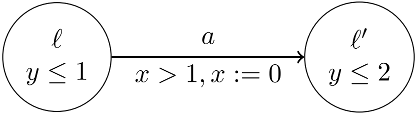
Operational Semantics



$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.9 \end{cases}$$

$$\begin{cases} v(x) = 1.2 \\ v(y) = 0.2 \end{cases} \qquad (\ell, v) \xrightarrow{0.5,\ a} (\ell', v')$$

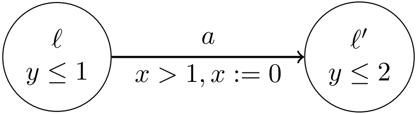
Operational Semantics



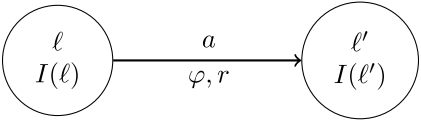
$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.9 \end{cases}$$

$$\begin{cases} v(x) = 1.2 \\ v(y) = 0.2 \end{cases} \qquad (\ell, v) \xrightarrow{0.5,\ a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.7 \end{cases}$$

Operational Semantics

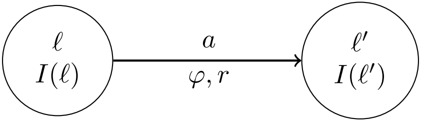


Operational Semantics



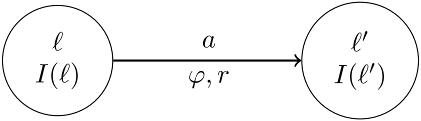
 Timed Automata

Operational Semantics



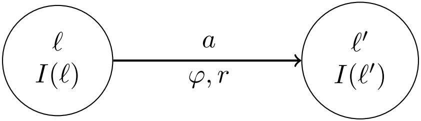
$$(\ell, v) \xrightarrow{d,\ a} (\ell', v')$$

Operational Semantics



$$(\ell, v) \xrightarrow{d,\ a} (\ell', (v+d)[r])$$

Operational Semantics



$$(\ell, v) \xrightarrow{d,\ a} (\ell', (v+d)[r])$$

provided that:

$$\ell \xrightarrow{\varphi,a,r} \ell' \text{ is a transition in the TA}$$

$$\forall t \in [0,d],\ v + t \models I(\ell)$$

$$(v+d)[r] \models I(\ell')$$

Operational Semantics

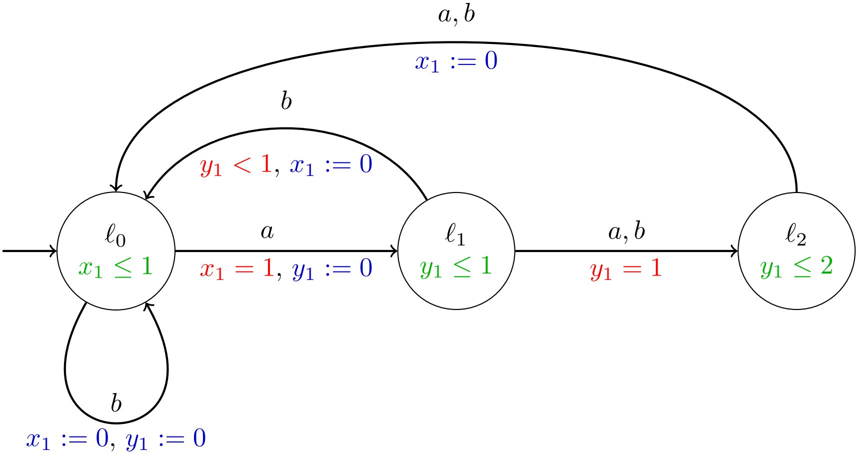
Definition 2

The *operational semantics* of a TA $A = (L, \ell_0, C, \Sigma, I, E)$ is the infinite-state timed transition system $[\![A]\!] = (S, s_0, \Sigma \times \mathbb{R}_{\geq 0}, T)$, where

$$S := \{(\ell, v) \in L \times \mathbb{R}^C_{\geq 0} \mid v \models I(\ell)\}, \quad s_0 := (\ell_0, \mathbf{0}_C),$$

$$T := \{(\ell, v) \xrightarrow{d,a} (\ell', (v+d)[r]) \mid d \in \mathbb{R}_{\geq 0},$$
$$\forall d' \in [0, d], v + d' \models I(\ell) \land \exists \ell \xrightarrow{\varphi,a,r} \ell' \in E, v + d \models \varphi\}$$

Timed Automata

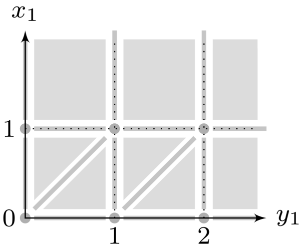


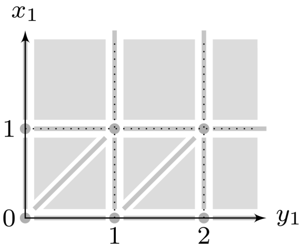
$$\left(\ell_0, \begin{bmatrix} 0.6 \\ 1 \end{bmatrix}\right) \xrightarrow{0.4,a} \left(\ell_1, \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) \xrightarrow{1,b} \left(\ell_2, \begin{bmatrix} 2 \\ 1 \end{bmatrix}\right) \xrightarrow{0.6,a} \left(\ell_0, \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}\right) \xrightarrow{0,b} \left(\ell_0, \begin{bmatrix} 0 \\ 0 \end{bmatrix}\right)$$

$\boxed{\text{Reachability and language emptiness for TA are PSPACE-complete}}$

Universality, inclusion and equivalence are all undecidable.

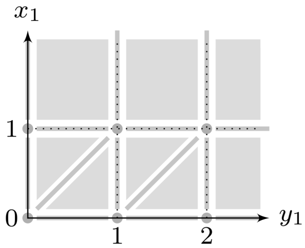
It can be proved with the *region automaton* construction: it has exponentially larger size, but checking a reachability property can be done on the fly, hence this can be done in polynomial space.


x_1
1
0
1
2
y_1



Definition 3

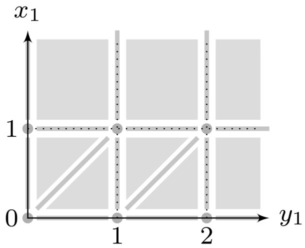
Two valuations $v, v' \in \mathbb{R}_{\geq 0}^{C}$ are region equivalent, i.e $v \cong_M v'$ iff:



Definition 3

Two valuations $v, v' \in \mathbb{R}^C_{\geq 0}$ are region equivalent, i.e $v \cong_M v'$ iff:

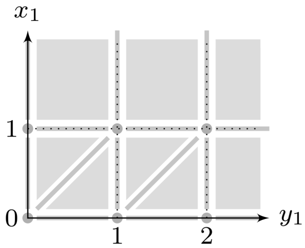
- $\forall x \in C, \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor \ \vee \ v(x), v'(x) > M_x$
  their integral parts on any clock are equal



Definition 3

Two valuations $v, v' \in \mathbb{R}_{\geq 0}^{C}$ are region equivalent, i.e $v \cong_M v'$ iff:

- $\forall x \in C, \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor \;\vee\; v(x), v'(x) > M_x$
  their integral parts on any clock are equal
- $\forall x \in C, \langle v(x) \rangle = 0 \Leftrightarrow \langle v'(x) \rangle = 0 \;\vee\; v(x) \geq M_x$
  their fractional parts on any clock are simultaneously equal to zero



Definition 3

Two valuations $v, v' \in \mathbb{R}_{\geq 0}^{C}$ are region equivalent, i.e $v \cong_M v'$ iff:

- $\forall x \in C, \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor \ \vee \ v(x), v'(x) > M_x$
  their integral parts on any clock are equal
- $\forall x \in C, \langle v(x) \rangle = 0 \Leftrightarrow \langle v'(x) \rangle = 0 \ \vee \ v(x) \geq M_x$
  their fractional parts on any clock are simultaneously equal to zero
- $\forall x, y \in C, \langle v(x) \rangle \leq \langle v(y) \rangle \Leftrightarrow \langle v'(x) \rangle \leq \langle v'(y) \rangle \ \vee \ v(x) > M_x \ \vee \ v(y) > M_y$
  the order of their fractional parts on any two clocks is preserved

Definition 4 (Region Automaton)

$\mathcal{R}_{\cong_M}(A) = (S, s_0, \Sigma, T)$ is the *region automaton* of A, where:

- $S := (L \times \mathbb{R}^C_{\geq 0})_{/\cong_M}$, $s_0 := [\ell_0, \mathbf{0}_C]_{/\cong_M}$
- $T := \{[\ell, v]_{\cong_M} \xrightarrow{a} [\ell', v']_{\cong_M} \mid \exists d \in \mathbb{R}_{\geq 0}, (\ell, v) \xrightarrow{d,a} (\ell', v')\}$

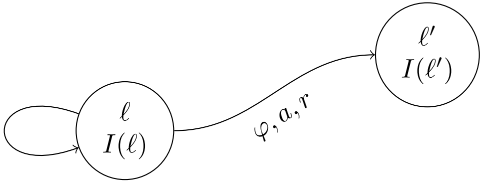
Definition 4 (Region Automaton)

$\mathcal{R}_{\cong_M}(A) = (S, s_0, \Sigma, T)$ is the *region automaton* of A, where:

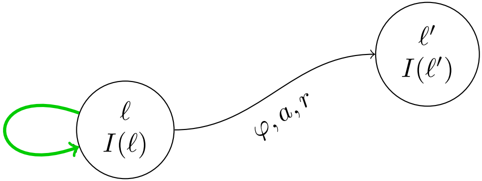
- $S := (L \times \mathbb{R}^C_{\geq 0})_{/\cong_M}, \; s_0 := [\ell_0, \mathbf{0}_C]_{/\cong_M}$
- $T := \{[\ell, v]_{\cong_M} \xrightarrow{a} [\ell', v']_{\cong_M} \mid \exists d \in \mathbb{R}_{\geq 0}, (\ell, v) \xrightarrow{d,a} (\ell', v')\}$

- $|S|$ is exponential in the number of clocks and in the maximal constants of the timed automaton
- Is there a way to reduce the number of states?

Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.

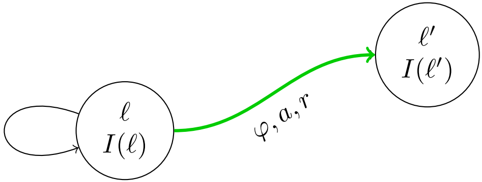


Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



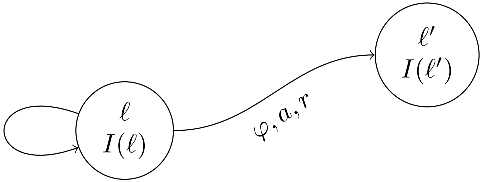
- Wait in ℓ for a delay d to elapse

Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



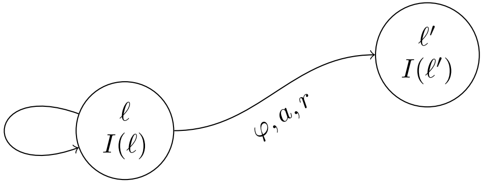
- Wait in ℓ for a delay d to elapse
- Take the transition to ℓ' (without any delay)

Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



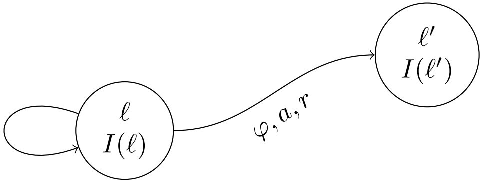
- Wait in ℓ for a delay d to elapse
  - Z must be "delayed" (transition to the upward closure of Z)
  - $I(\ell)$ must be satisfied over the whole delay
- Take the transition to ℓ' (without any delay)

Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



- Wait in ℓ for a delay d to elapse
    - Z must be "delayed" (transition to the upward closure of Z)
    - $I(\ell)$ must be satisfied over the whole delay
- Take the transition to ℓ' (without any delay)
    - φ must be satisfied
    - Z must be "reset" w.r.t. r
    - $I(\ell')$ must be satisfied eventually

Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



- Wait in ℓ for a delay d to elapse
  - Z must be "delayed" (transition to the upward closure of Z)
  - $I(\ell)$ must be satisfied over the whole delay
- Take the transition to ℓ' (without any delay)
  - φ must be satisfied
  - Z must be "reset" w.r.t. r
  - $I(\ell')$ must be satisfied eventually

Need to be normalised!

Definition 5 (Zone)

A set of valuations $Z \subseteq \mathbb{R}^C_{\geq 0}$ is a zone iff: $\exists \varphi \in \Phi_d(C), Z = [\![\varphi]\!]_C$
In this case, we define:

- the *delay* of Z: $Z^{\uparrow} \triangleq \{v + d \mid v \in Z \land d \in \mathbb{R}_{\geq 0}\}$
- the *reset* of Z: $Z[r] \triangleq \{v[r] \mid v \in Z\}$ for $r \subseteq C$

Definition 6 (Zone automaton)

The *zone automaton* $[\![A]\!]_Z$ of A is the tuple $(S, s_0, \Sigma \cup \{\delta\}, T)$, where:

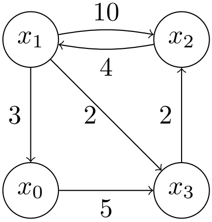
$$S := \{(\ell, Z) \mid \ell \in L, Z \in \mathbb{R}^C_{\geq 0} \text{ is a zone}\}, \quad s_0 := (\ell_0, \{\mathbf{0}_C\})$$

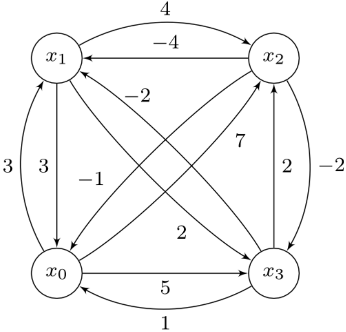
$$T := \{(\ell, Z) \overset{\delta}{\rightsquigarrow} (\ell, Z^{\uparrow} \cap [\![I(\ell)]\!]_C)\} \cup$$

$$\{(\ell, Z) \overset{a}{\rightsquigarrow} (\ell', (Z \cap [\![\varphi]\!]_C)[r] \cap [\![I(\ell')]\!]_C) \mid \ell \xrightarrow{\varphi, a, r} \ell' \in E\}$$

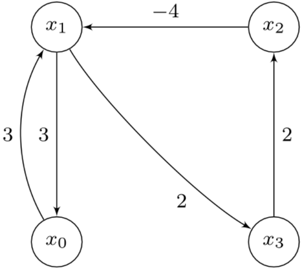
$\rightarrow$ Normalisation ($\simeq$ quotienting), shortest-path-closed DBM extrapolation

$$Z = \begin{cases} x_1 & \leq 3 \\ x_1 - x_2 & \leq 10 \\ x_1 - x_2 & \geq 4 \\ x_1 - x_3 & \leq 2 \\ x_3 - x_2 & \leq 2 \\ x_3 & \geq -5 \end{cases}$$





Shortest-path closure



Shortest-path reduction

Decidable extensions:

- *diagonal constraints*: $x - y \odot k$
- *updatable* TA: clocks can be reset to any natural number ($x := k \in \mathbb{Z}$) or can be synchronized with another clock ($x := y$)
- *urgency* constraints: some locations must be left immediately

Undecidable extensions:

- *linear* constraints: $\lambda x + \mu y \leq k$
- updates such as $x := x + k$ or $x :> k$ (non-determinism)
- *stopwatch* automata
- *hybrid* automata
- ...

Decidable extensions:

- *diagonal constraints*: $x - y \odot k$
- *updatable* TA: clocks can be reset to any natural number ($x := k \in \mathbb{Z}$) or can be synchronized with another clock ($x := y$)
- *urgency* constraints: some locations must be left immediately

Undecidable extensions:

- *linear* constraints: $\lambda x + \mu y \leq k$
- updates such as $x := x + k$ or $x :> k$ (non-determinism)
- *stopwatch* automata
- *hybrid* automata
- pretty much everything else you can think of

Definition 7 (Metric Temporal Logic)

Given a set of atomic propositions P, the formulas of MTL are defined for any time interval I with the *time-constrained until* operator $\mathbf{U}_I$ as follows:

$$\varphi ::= p \in P \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \mathbf{U}_I \varphi$$

The *constrained always* $\Box_I$ and *constrained eventually* $\Diamond_I$ operators can be defined with $\mathbf{U}_I$ in a similar way as in LTL, namely:

$$\Diamond_I \varphi \triangleq \top \mathbf{U}_I \varphi$$
$$\Box_I \varphi \triangleq \neg \Diamond_I \neg \varphi$$

Definition 7 (Metric Temporal Logic)

Given a set of atomic propositions P, the formulas of MTL are defined for any time interval I with the *time-constrained until* operator $\mathbf{U}_I$ as follows:

$$\varphi ::= p \in P \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \mathbf{U}_I \varphi$$

The *constrained always* $\Box_I$ and *constrained eventually* $\Diamond_I$ operators can be defined with $\mathbf{U}_I$ in a similar way as in LTL, namely:

$$\Diamond_I \varphi \triangleq \top \mathbf{U}_I \varphi$$
$$\Box_I \varphi \triangleq \neg \Diamond_I \neg \varphi$$

$\boxed{\text{What is } \mathbf{U}_I\text{?!}}$

Definition 7 (Metric Temporal Logic)

Given a set of atomic propositions P, the formulas of MTL are defined for any time interval I with the *time-constrained until* operator $\mathbf{U}_I$ as follows:

$$\varphi ::= p \in P \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \mathbf{U}_I \varphi$$

The *constrained always* $\Box_I$ and *constrained eventually* $\Diamond_I$ operators can be defined with $\mathbf{U}_I$ in a similar way as in LTL, namely:

$$\Diamond_I \varphi \triangleq \top \mathbf{U}_I \varphi$$

$$\Box_I \varphi \triangleq \neg \Diamond_I \neg \varphi$$

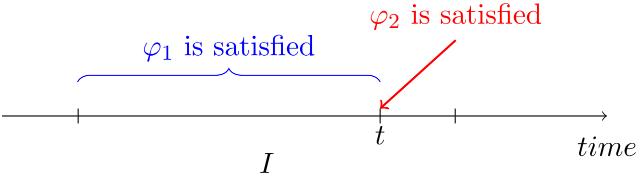
$\boxed{\text{What is } \mathbf{U}_I\text{?!}}$ $\rightarrow$ give its semantics

Continuous semantics:

$$f \models \varphi_1 \mathbf{U}_I \varphi_2 \quad \text{iff} \quad \exists t \in I, f^t \models \varphi_2 \wedge \forall u \in (0,t), f^u \models \varphi_1$$

Continuous semantics:

$$f \models \varphi_1 \mathbf{U}_I \varphi_2 \quad \text{iff} \quad \exists t \in I, f^t \models \varphi_2 \wedge \forall u \in (0,t), f^u \models \varphi_1$$

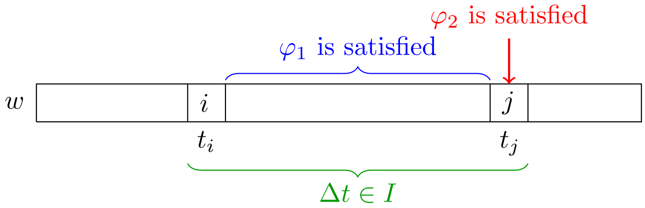


Pointwise semantics: Let $w = ((a_i, t_i))_{i \in \mathbb{N}}$ be a timed word over 2^P.

$$w, i \models \varphi_1 \mathbf{U}_I \varphi_2 \quad \text{iff} \quad \exists\ i < j < |w|, \begin{cases} w, j \models \varphi_2 & \wedge \\ t_j - t_i \in I & \wedge \\ \forall\ i < k < j,\ w, k \models \varphi_1 & \end{cases}$$

Pointwise semantics: Let $w = ((a_i, t_i))_{i \in \mathbb{N}}$ be a timed word over 2^P.

$$w, i \models \varphi_1 \mathbf{U}_I \varphi_2 \quad \text{iff} \quad \exists\ i < j < |w|, \begin{cases} w, j \models \varphi_2 & \wedge \\ t_j - t_i \in I & \wedge \\ \forall\ i < k < j,\ w, k \models \varphi_1 \end{cases}$$



- MC and SAT for "classical" LTL are PSPACE-complete over both semantics
  The PSPACE upper bound can be established by translating the negated formula to a Büchi automaton, and performing an on-the-fly reachability check on the product of this automaton with the region graph of the model.
- MC and SAT for MTL in the pointwise semantics are decidable over finite words only, and are undecidable in the continuous semantics
  Decidability results are essentially obtained by translating MTL formulas into one-clock alternating timed automata, and rephrasing the model-checking or satisfiability problems as instances of language emptiness in one-clock alternating timed automata. Continuous semantics is strictly more expressive!
  - MC and SAT for MTL over bounded time and MITL are EXPSPACE-complete for both semantics (over finite and infinite words)
  - MC and SAT for ECTL are PSPACE-complete for both semantics
- MC is PSPACE-complete and SAT is undecidable for TCTL
  The idea is to associate universal path-quantifiers with each temporal modality.

Definition 8

A *timed game* is basically a deterministic TA $A_G = (L, \ell_0, C, \Sigma_c \cup \Sigma_u, I, E)$ where Σ_c and Σ_u are disjoint.

A *strategy* over $[\![A_G]\!]$ is a mapping from finite runs to $\Sigma_c \cup \{\delta\}$.

The outcome of a strategy is *maximal* either if it stops in a target location or or if no controllable actions are available at its end.

A strategy is *winning for the reachability game* if any of its maximal outcome and leads to a target location, and *winning for the safety game* if none of its outcomes is accepting.

Timed games by example:

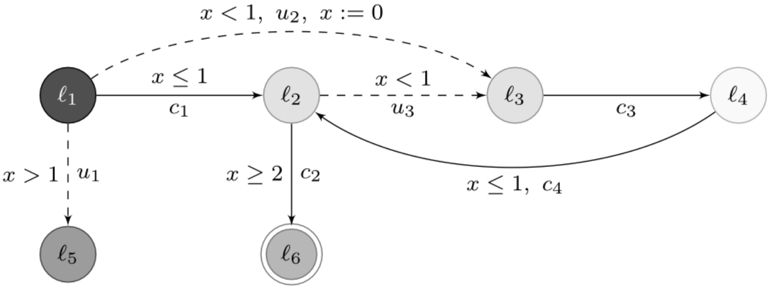


Figure taken from [2]

Timed games by example:

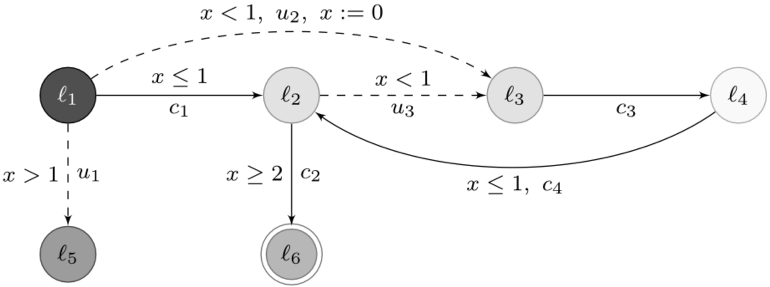


Figure taken from [2]

(ℓ_1, v)	$\rightarrow$	$\begin{cases} \delta \text{ if } v(x) \leq 1 \\ c_1 \text{ if } v(x) = 1 \end{cases}$	(ℓ_2, v)	$\rightarrow$	$\begin{cases} \delta \text{ if } v(x) \leq 2 \\ c_2 \text{ if } v(x) \geq 2 \end{cases}$
(ℓ_3, v)	$\rightarrow$	$\begin{cases} \delta \text{ if } v(x) < 1 \\ c_3 \text{ if } v(x) \geq 1 \end{cases}$	(ℓ_4, v)	$\rightarrow$	$\begin{cases} \delta \text{ if } v(x) \neq 1 \\ c_4 \text{ if } v(x) = 1 \end{cases}$

Remarks:

- The (time-optimal) reachability and safety games are decidable for timed games. They are EXPTIME-complete.

  For reachability as well as safety games, it is sufficient to consider memoryless strategies! Exponential time essentially comes from the region abstraction.

- Restriction to non-Zeno is still EXPTIME-complete.
- TG can be used to check time bisimilarity in a TA with a simple construction

Ongoing challenges

- Robustness and implementability: reconcile the semantics of TA with the models they represent
- Statistical model checking: stochastic TA
- Timed Games: extensions, e.g. where the players have their own objectives

References

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

[2] E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem.
Handbook of Model Checking.
Springer Publishing Company, Incorporated, 1st edition, 2018.