Model Checking Real-Time Systems

Vincent Trélat

Technical University of Munich

December 12, 2022

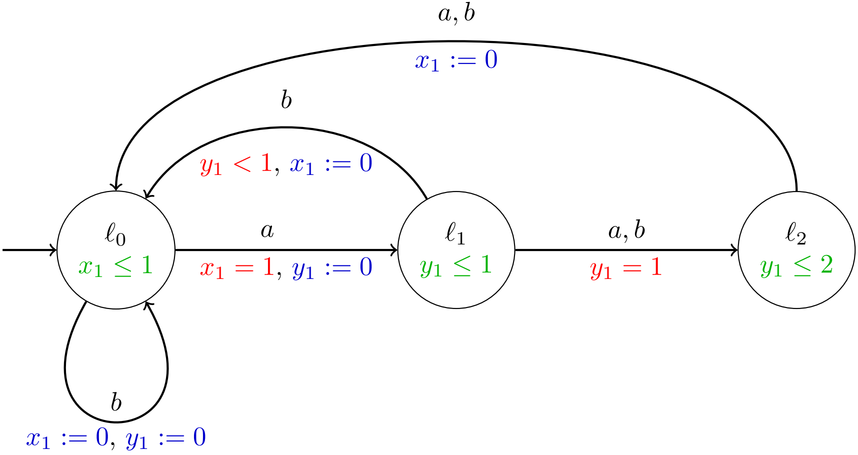
Definition 1

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

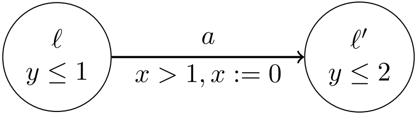
Edges are denoted by $\ell \xrightarrow{\varphi, a, r} \ell'$.

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

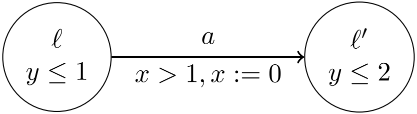


 Timed Automata

Operational Semantics



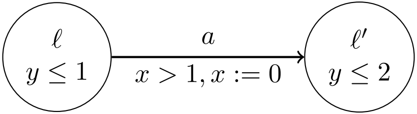
Operational Semantics



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

 Timed Automata

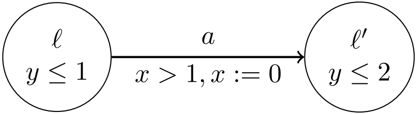
Operational Semantics



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

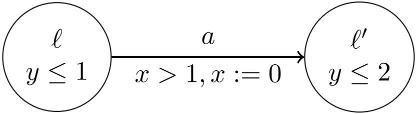
 Timed Automata

Operational Semantics



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

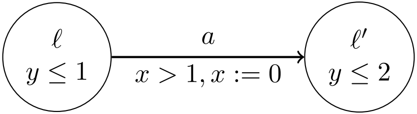
Operational Semantics



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

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

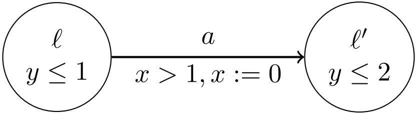
Operational Semantics



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

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

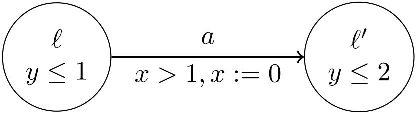
Operational Semantics



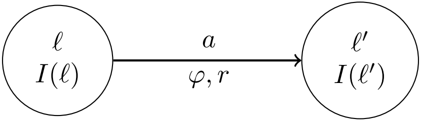
$$\begin{cases} v(x) = 3.4 \\ v(y) = 0.9 \end{cases} \qquad (\ell, v) \xrightarrow{a} (\ell', v') \qquad \begin{cases} v'(x) = 0 \\ v'(y) = 0.9 \end{cases}$$

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

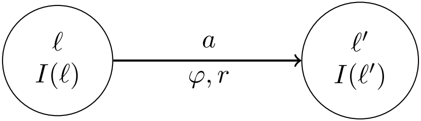
Operational Semantics



Operational Semantics

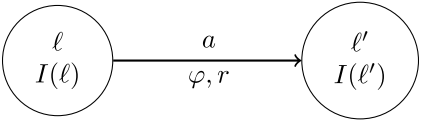


Operational Semantics



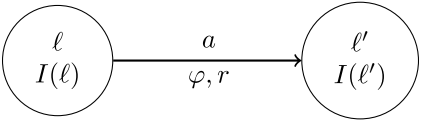
$$(\ell, v) \xrightarrow{d,\ a} (\ell', v')$$

Operational Semantics



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

Operational Semantics



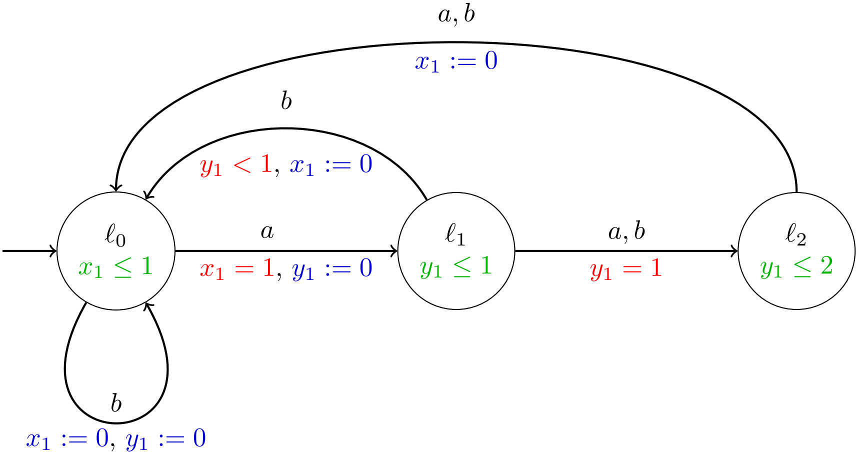
$$(\ell, v) \xrightarrow{d,\ a} (\ell', (v+d)[r])$$

provided that:

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

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

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

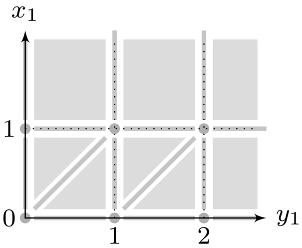


$$\left(\ell_0, \begin{bmatrix} 0.6 \\ 1 \end{bmatrix}\right) \xrightarrow{0.4,a} \left(\ell_1, \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) \xrightarrow{1,b} \left(\ell_2, \begin{bmatrix} 2 \\ 1 \end{bmatrix}\right) \xrightarrow{0.6,a} \left(\ell_0, \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}\right) \xrightarrow{0,b}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

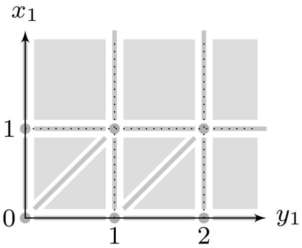
$\boxed{\text{Reachability and language emptiness for TA are PSPACE-complete}}$

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.

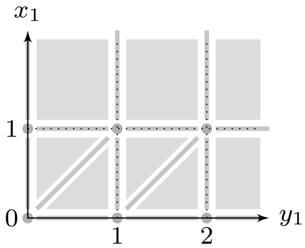


x_1
1
0
1
2
y_1



Definition 2

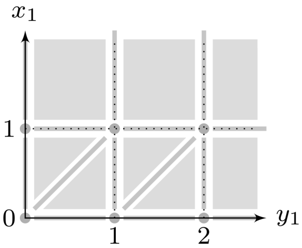
Two valuations $v, v' \in \mathbb{R}^{C}_{\geq 0}$ are region equivalent, i.e $v \cong_M v'$ iff:



Definition 2

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

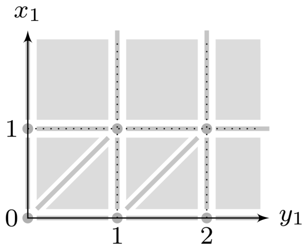
- $\forall x \in C, \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor \ \vee \ v(x), v'(x) > M_x$
  their integral parts on any clock are equal



Definition 2

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



Definition 2

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
- $\forall x \in C, \langle v(x) \rangle = 0 \Leftrightarrow \langle v'(x) \rangle = 0 \;\vee\; v(x) \geq M_x$
  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 3 (Region Automaton)

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

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

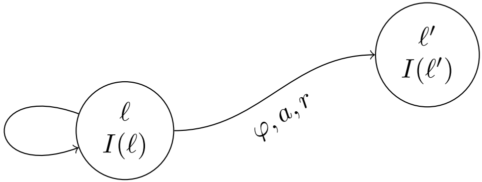
Definition 3 (Region Automaton)

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

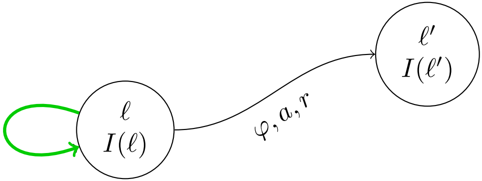
- $S := (L \times \mathbb{R}^C_{\geq 0})_{/\cong_M}$, $s_0 := [\ell_0, \mathbf{0}_C]_{/\cong_M}$
- $T := \{[\ell, v]_{\cong_M} \xrightarrow{a} [\ell', v']_{\cong_M} \mid \exists d \in \mathbb{R}_{\geq 0}, (\ell, v) \xrightarrow{d,a} (\ell', v')\}$

- $|S|$ 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. $[\![\varphi]\!]_C$.

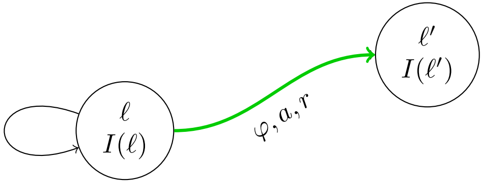


Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



- Wait in ℓ for a delay d to elapse

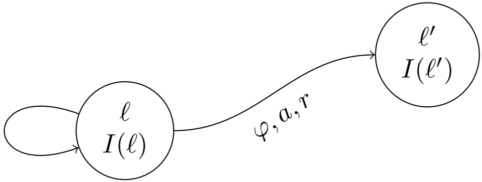
Zone: a set of valuations satisfying a constraint over C, e.g. $[\![\varphi]\!]_C$.



- Wait in ℓ for a delay d to elapse

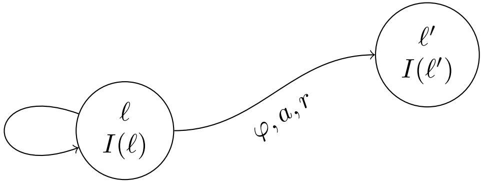
- Take the transition to ℓ' (without any delay)

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



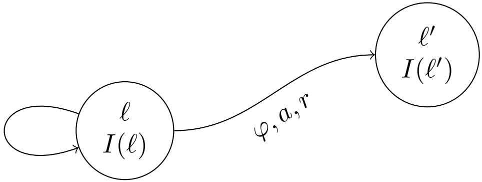
- Wait in ℓ for a delay d to elapse
  - Z must be "delayed" (transition to the upward closure of Z)
  - $I(\ell)$ must be satisfied over the whole delay
- Take the transition to ℓ' (without any delay)

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!

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 4 (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.

Definition 4 (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.

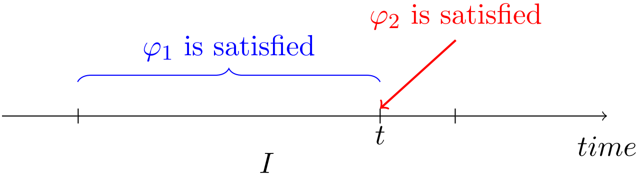
$\boxed{\text{What is } \mathbf{U}_I\text{?!}}$

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:

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

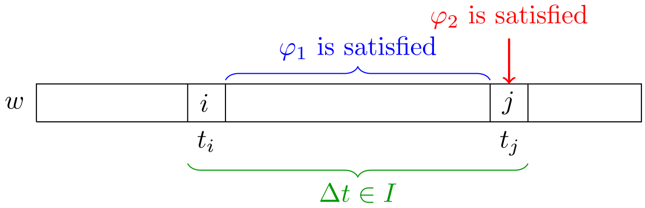


Pointwise semantics: Let $w = ((a_i, t_i))_{i \in \mathbb{N}}$ be a timed word over 2^P.

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

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

- MC and SAT for “classical” LTL are PSPACE-complete over both semantics

- MC and SAT for MTL in the pointwise semantics are decidable over finite words only, and are undecidable in the continuous semantics

  $\rightarrow$ In some subsets of MTL, we can recover decidability

- MC and SAT for “classical” LTL are PSPACE-complete over both semantics

- MC and SAT for MTL in the pointwise semantics are decidable over finite words only, and are undecidable in the continuous semantics

  $\rightarrow$ In some subsets of MTL, we can recover decidability

- MC is PSPACE-complete and SAT is undecidable for TCTL

Timed games by example:

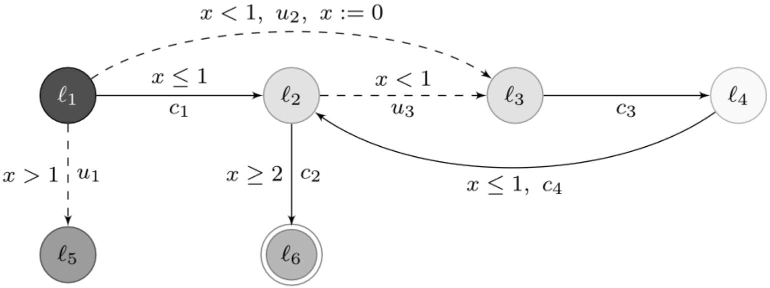


Figure taken from [2]

Timed games by example:

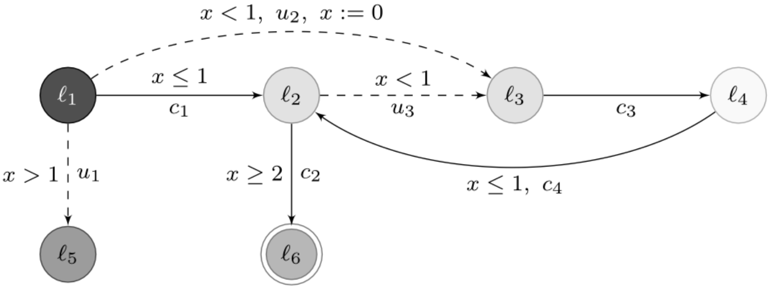


Figure taken from [2]

$(\ell_1, v) \quad \rightarrow \quad \begin{cases} \delta \text{ if } v(x) \leq 1 \\ c_1 \text{ if } v(x) = 1 \end{cases}$	$(\ell_2, v) \quad \rightarrow \quad \begin{cases} \delta \text{ if } v(x) \leq 2 \\ c_2 \text{ if } v(x) \geq 2 \end{cases}$
$(\ell_3, v) \quad \rightarrow \quad \begin{cases} \delta \text{ if } v(x) < 1 \\ c_3 \text{ if } v(x) \geq 1 \end{cases}$	$(\ell_4, v) \quad \rightarrow \quad \begin{cases} \delta \text{ if } v(x) \neq 1 \\ c_4 \text{ if } v(x) = 1 \end{cases}$

Remarks:

- The (time-optimal) reachability and safety games are decidable for timed games. They are EXPTIME-complete.
- Restriction to non-Zeno is still EXPTIME-complete.
- TG can be used to check time bisimilarity

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.