Languages of Higher-Dimensional Timed Automata

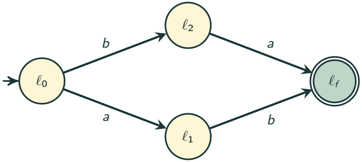
Amazigh Amrane [2] *Hugo Bazille* [2] ***Emily Clement*** [1] *Uli Fahrenberg* [2]

[1] Université Paris Cité, CNRS, IRIF, F-75013, Paris, France
[2] EPITA Research Laboratory (LRE), Paris, France

26th of May 2024

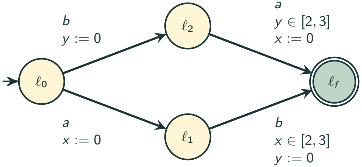
Untimed and Timed models for interleaving concurrency

Classical Automata



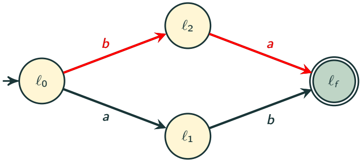
- Language : words.
  Here : $a.b + b.a$

Timed Automata

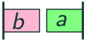


Untimed and Timed models for interleaving concurrency

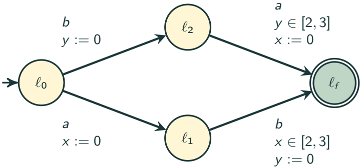
Classical Automata



- Language : words.
  Here : $a.b + b.a$

- 

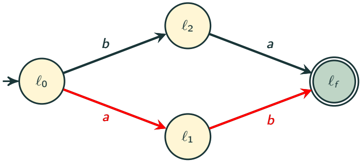
Timed Automata



- Language : Timed words

Untimed and Timed models for interleaving concurrency

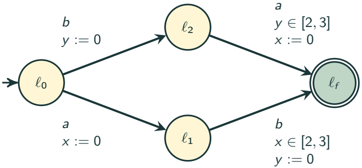
Classical Automata



- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

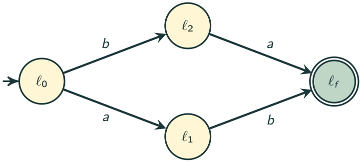
Timed Automata

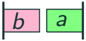
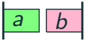


- Language : Timed words

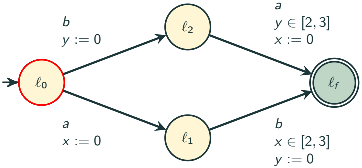
Untimed and Timed models for interleaving concurrency

Classical Automata

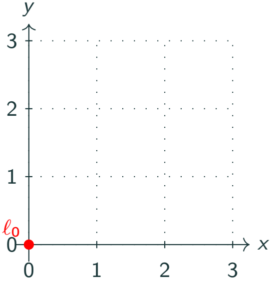


- Language : words.
  Here : $a.b + b.a$
-  or 
- Interleaving concurrency
- No timing constraints

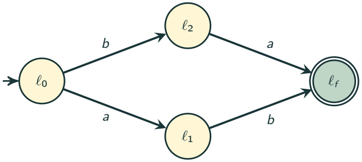
Timed Automata



- Language : Timed words
- Example : (b, (1,1)) (a, (3,2))

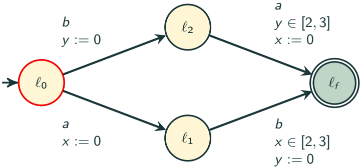


Classical Automata

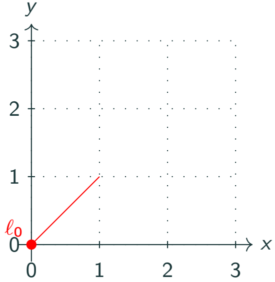


- Language : words.
  Here : $a.b + b.a$
- *b* *a* or *a* *b*
- Interleaving concurrency
- No timing constraints

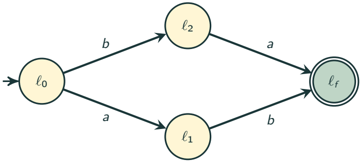
Timed Automata



- Language : Timed words
- Example : (b, (1,1)) (a, (3,2))



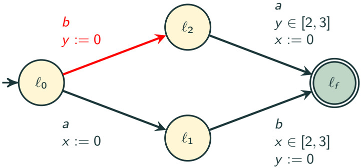
Classical Automata



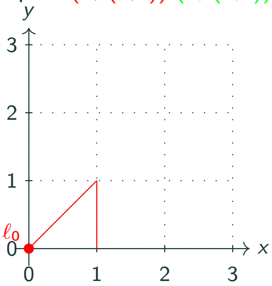
- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

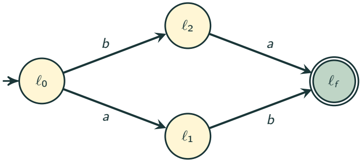
Timed Automata



- Language : Timed words
- Example : (b, (1,1)) (a, (3,2))



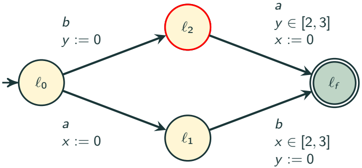
Classical Automata



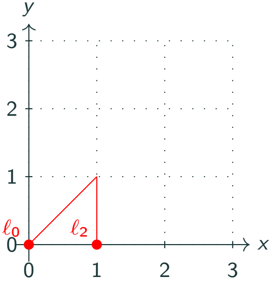
- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

Timed Automata

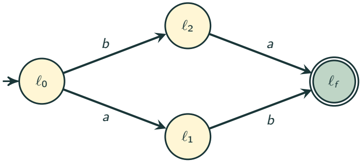


- Language : Timed words
- Example : (b, (1,1)) (a, (3,2))



Untimed and Timed models for interleaving concurrency

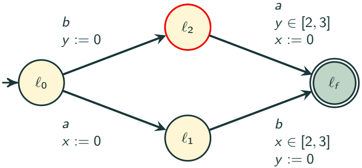
Classical Automata



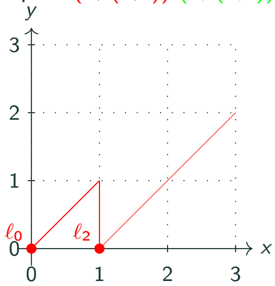
- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

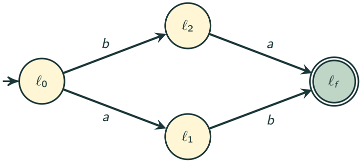
Timed Automata



- Language : Timed words
- Example : (b, (1,1)) (a, (3,2))

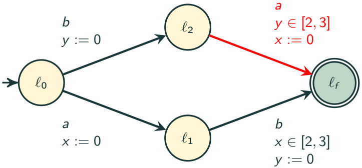


Classical Automata

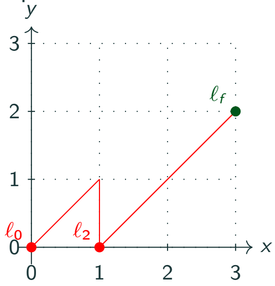


- Language : words.
  Here : $a.b + b.a$
- b a or a b
- Interleaving concurrency
- No timing constraints

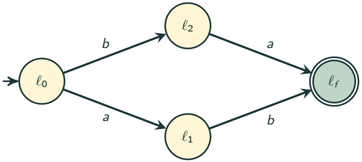
Timed Automata



- Language : Timed words
- Example :



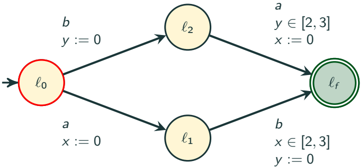
Classical Automata



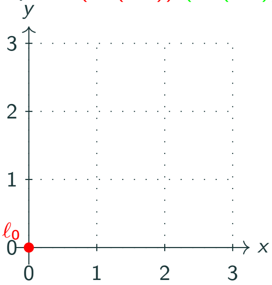
- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

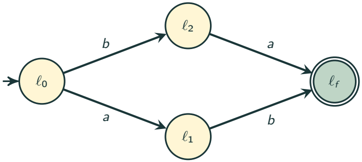
Timed Automata



- Language : Timed words
- Example : (a, (1,1)) (b, (2,3))

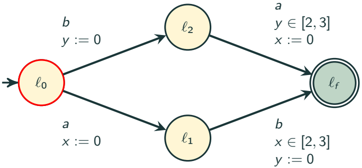


Classical Automata

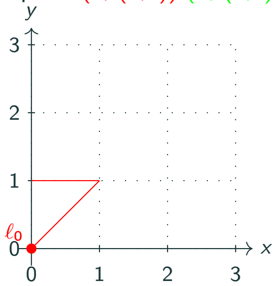


- Language : words.
  Here : $a.b + b.a$
- b a or a b
- Interleaving concurrency
- No timing constraints

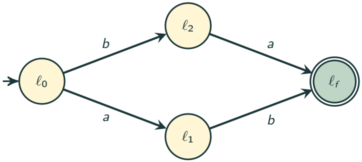
Timed Automata



- Language : Timed words
- Example : (a, (1,1)) (b, (2,3))

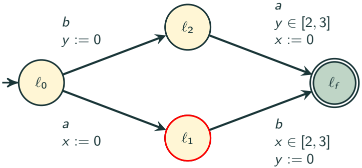


Classical Automata

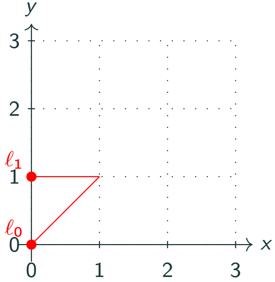


- Language : words.
  Here : $a.b + b.a$
- b a or a b
- Interleaving concurrency
- No timing constraints

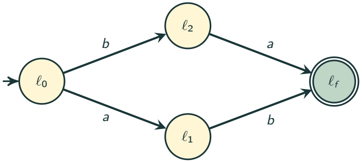
Timed Automata

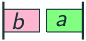


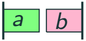
- Language : Timed words
- Example : (a, (1,1)) (b, (2,3))



Classical Automata

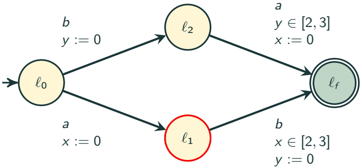


- Language : words.
  Here : $a.b + b.a$
- 

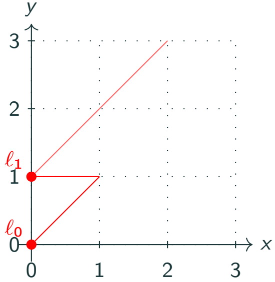
  or 

- Interleaving concurrency
- No timing constraints

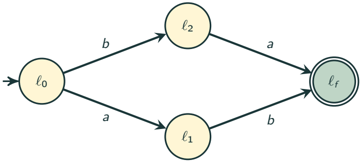
Timed Automata



- Language : Timed words
- Example : (a, (1,1)) (b, (2,3))



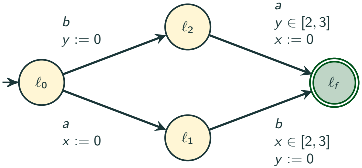
Classical Automata



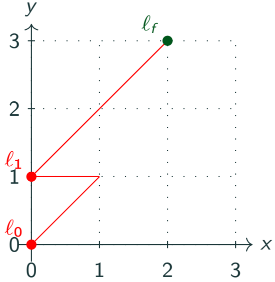
- Language : words.
  Here : $a.b + b.a$
- 

- Interleaving concurrency
- No timing constraints

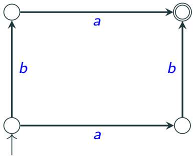
Timed Automata

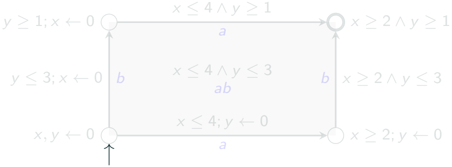


- Language : Timed words
- Example : (a, (1,1)) (b, (2,3))

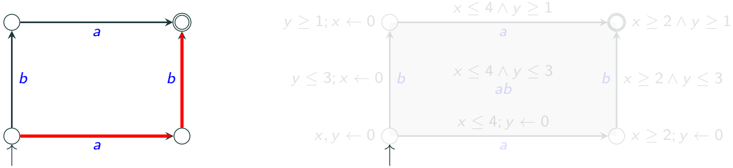


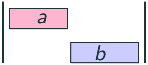
- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :





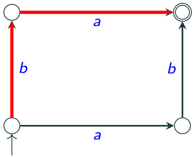
- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :

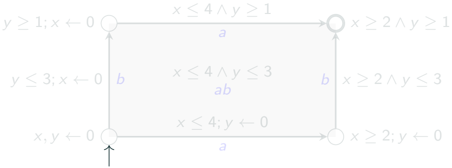


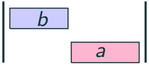
- Example : 

- HDA Language : $L(\mathcal{A}) = \{(\ a \rightarrow b \ ), ?\}$

- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :

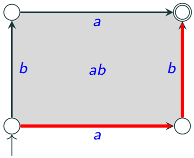


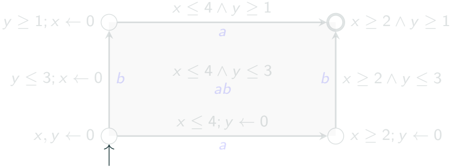


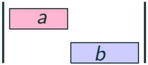
- Example : 

- HDA Language : $L(\mathcal{A}) = \{(\ a \rightarrow b\ ), (\ b \rightarrow a\ )\}$

- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :



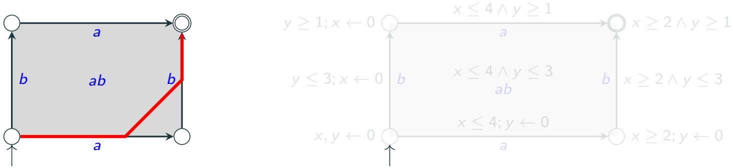


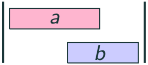
- Example : 

- HDA Language : $L(\mathcal{A}) = \{(\ a \rightarrow b\ ), (\ b \rightarrow a\ ), ?\}$

- **True concurrency** : $a||b \neq a.b + b.a$

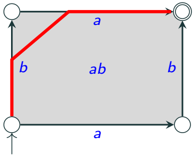
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :

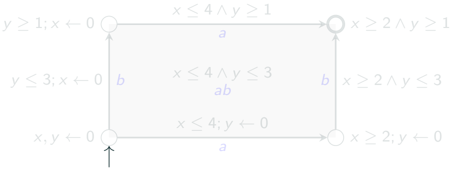


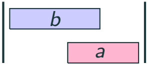
- Example : 

- HDA Language : $L(\mathcal{A}) = \left\{ \left(\ a \rightarrow b \ \right), \left(\ b \rightarrow a \ \right), \left(\begin{matrix} a \\ b \end{matrix} \right) \right\}$

- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :

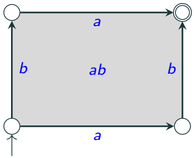


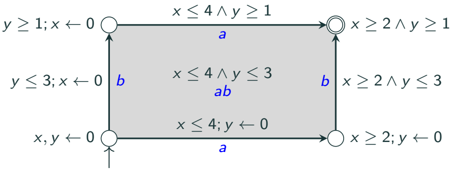


- Example : 

- HDA Language : $L(\mathcal{A}) = \left\{ \left(\; a \rightarrow b \; \right), \left(\; b \rightarrow a \; \right), \left(\begin{array}{c} a \\ b \end{array} \right) \right\}$

- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :

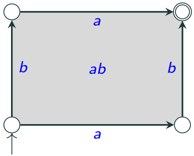


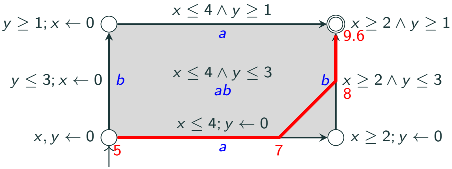


- Example :

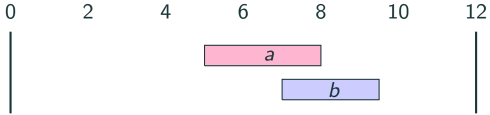
Untimed and Timed models for true concurrency

- **True concurrency** : $a||b \neq a.b + b.a$
- Higher Dimensional Automata ($\mathcal{A}$, HDA, left), Higher Dimensional Timed Automata ($\mathcal{A}_t$, HDTA, right) :





- Example :



Contribution of Petri Nets 2024

- Express **Language** of Higher Dimensional Timed Automata

- Express **Language** of Higher Dimensional Timed Automata
- Decidability/Undecidability results :

Undecidable ✗

Inclusion of HDTA language

Decidable ✓

Inclusion of the closed (subsumption) of untimed Language of HDTA

Events representation : interval pomset with interfaces (iiPomset)

Interval Pomset with Interface

- $<$: precedence order (rep with $\longrightarrow$) , $\dashrightarrow$: event order.
- $< \cup \dashrightarrow$: **total** relation.

Events representation : interval pomset with interfaces (iiPomset)

Interval Pomset with Interface

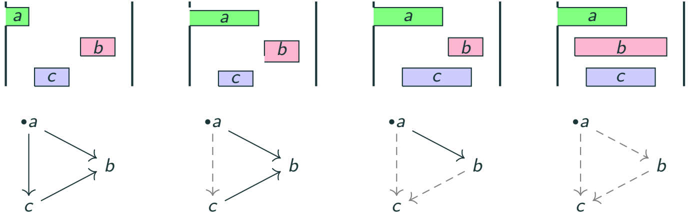
- $<$: precedence order (rep with $\longrightarrow$) , $\dashrightarrow$: event order.
- $< \cup \dashrightarrow$: **total** relation.
- **Source/Target interfaces** : S/T : $<$-minimal/maximal.

Events representation : interval pomset with interfaces (iiPomset)

Interval Pomset with Interface

- $<$: precedence order (rep with $\longrightarrow$) , $\dashrightarrow$: event order.
- $< \cup \dashrightarrow$: **total** relation.
- **Source/Target interfaces** : S/T : $<$-minimal/maximal.

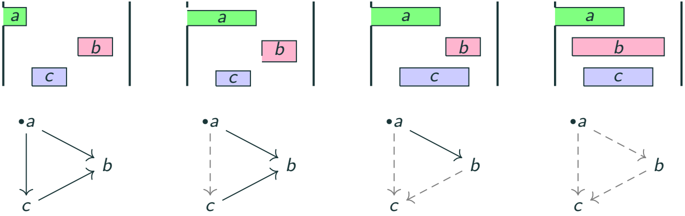
Representation of events as interval



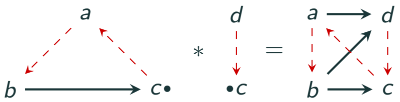
Interval Pomset with Interface

- ▷ $<$: precedence order (rep with $\longrightarrow$) , $\dashrightarrow$: event order.
- ▷ $< \cup \dashrightarrow$: **total** relation.
- ▷ **Source/Target interfaces** : S/T : $<$-minimal/maximal.

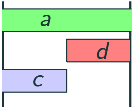
Representation of events as interval



Gluing composition :



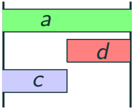
Timed iiPomsets : add the duration information



Timed interval pomsets with interface

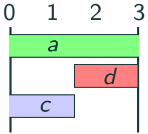
Timed iiPomsets : add the duration information

- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0, 3) \\ \sigma(d) & = (1.5, 3) \\ \sigma(c) & = (0, 1.5) \end{cases}$



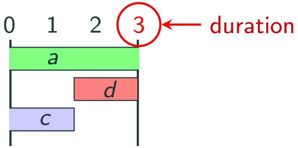
Timed iiPomsets : add the duration information

- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0,3) \\ \sigma(d) & = (1.5,3) \\ \sigma(c) & = (0,1.5) \end{cases}$



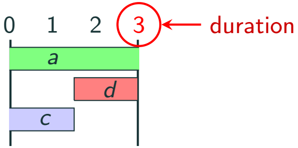
Timed iiPomsets : add the duration information

- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0,3) \\ \sigma(d) & = (1.5,3) \\ \sigma(c) & = (0,1.5) \end{cases}$

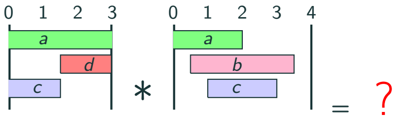


Timed iiPomsets : add the duration information

- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0,3) \\ \sigma(d) & = (1.5,3) \\ \sigma(c) & = (0,1.5) \end{cases}$



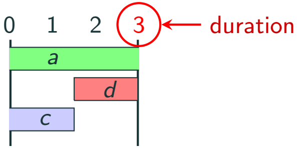
Gluing operation extends naturally :



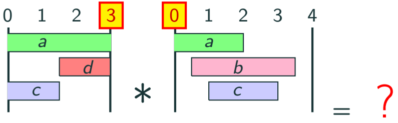
Timed interval pomsets with interface

Timed iiPomsets : add the duration information

- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0,3) \\ \sigma(d) & = (1.5,3) \\ \sigma(c) & = (0,1.5) \end{cases}$

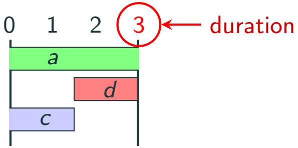


Gluing operation extends naturally :

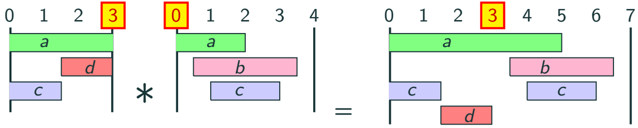


Timed iiPomsets : add the duration information

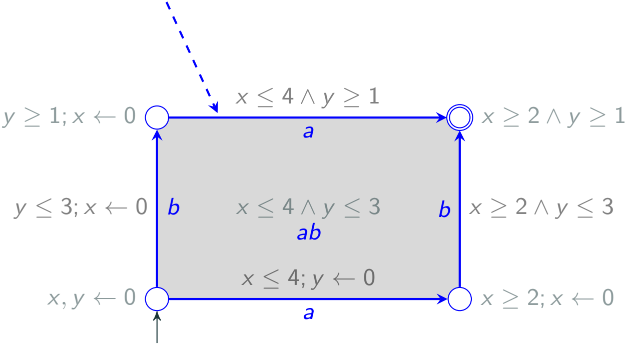
- σ : events $\mapsto$ time intervals : $\begin{cases} \sigma(a) & = (0,3) \\ \sigma(d) & = (1.5,3) \\ \sigma(c) & = (0,1.5) \end{cases}$



Gluing operation extends naturally :

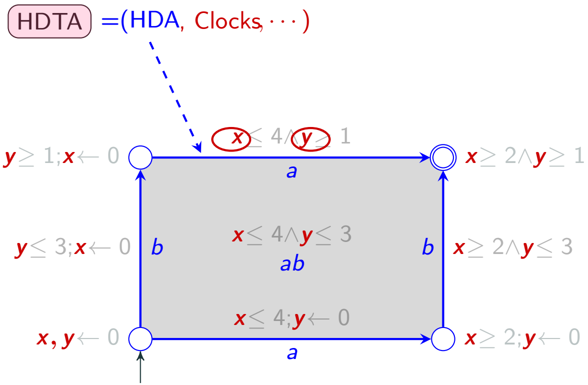


HDTA =(HDA, ⋯)

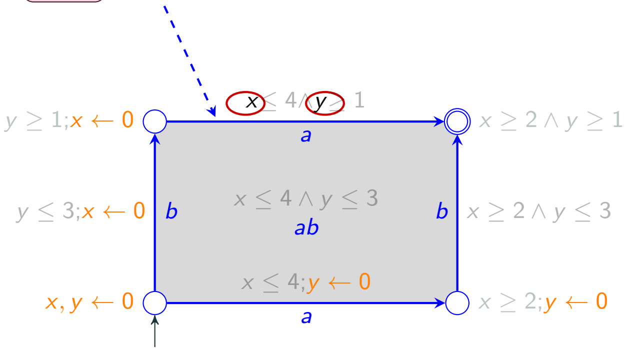


$y \geq 1; x \leftarrow 0$
$x \leq 4 \wedge y \geq 1$
a
$x \geq 2 \wedge y \geq 1$
$y \leq 3; x \leftarrow 0$
b
$x \leq 4 \wedge y \leq 3$
ab
b
$x \geq 2 \wedge y \leq 3$
$x \leq 4; y \leftarrow 0$
$x, y \leftarrow 0$
a
$x \geq 2; x \leftarrow 0$

Higher Dimensional Timed Automata : intuition

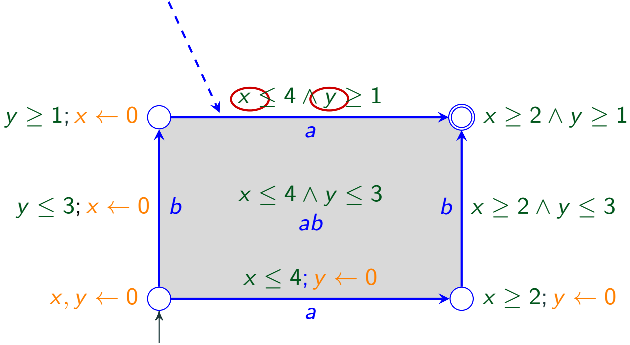


HDTA =(HDA, Clocks, Exit conditions, $\cdots$)



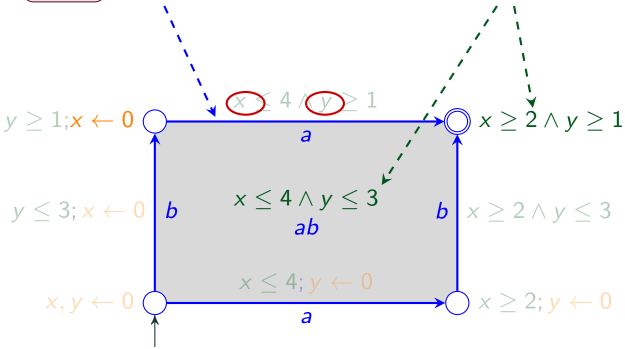
Higher Dimensional Timed Automata : intuition

HDTA =(HDA, Clocks, Exit conditions , Invariants)



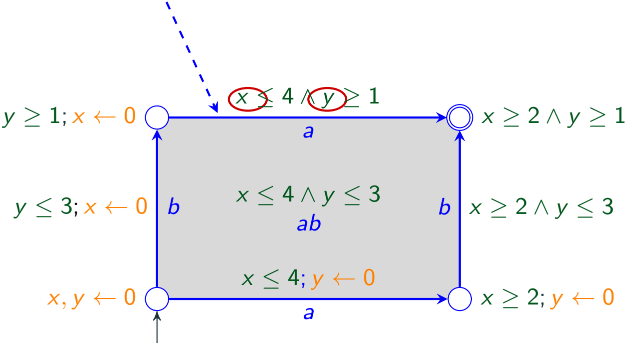
Higher Dimensional Timed Automata : intuition

HDTA =(HDA, Clocks, Exit conditions , Invariants)

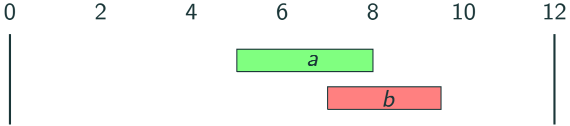


Higher Dimensional Timed Automata : intuition

HDTA =(HDA, Clocks, Exit conditions , Invariants)



Accepting path



Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of $a : [2, 4]$ and $b : [1, 3]$ separately :

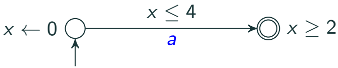
Timing duration of events :

- $a : [2, 4]$ time units
- $b : [1, 3]$ time units

Examples of HDTA

Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of $a : [2,4]$ and $b : [1,3]$ separately :



Timing duration of events :

- $a : [2,4]$ time units
- $b : [1,3]$ time units

Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of $a : [2,4]$ and $b : [1,3]$ separately :

Timing duration of events :

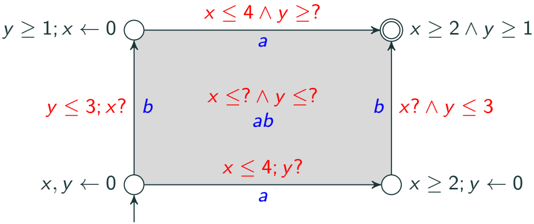
- $a : [2,4]$ time units
- $b : [1,3]$ time units

Examples of HDTA

Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of $a : [2, 4]$ and $b : [1, 3]$ separately :

Let's put them together

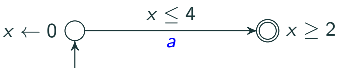


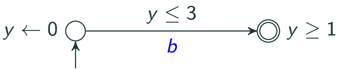
Timing duration of events :

- $a : [2, 4]$ time units
- $b : [1, 3]$ time units

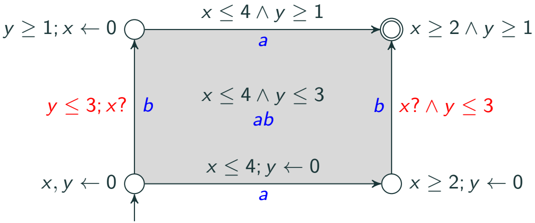
Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of a : $[2,4]$ and b : $[1,3]$ separately :





Let's put them together



Timing duration of events :

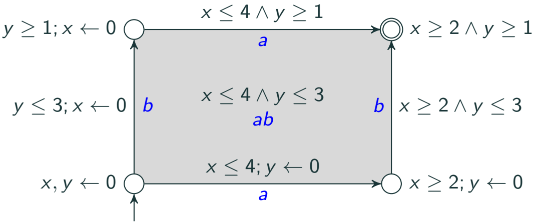
- a : $[2,4]$ time units
- b : $[1,3]$ time units

Examples of HDTA

Quizz : suppose that a and b are not in concurrency

Let us draw the HDTA of $a : [2,4]$ and $b : [1,3]$ separately :

Let's put them together

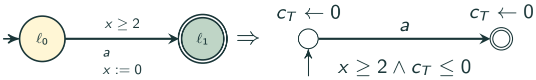


Timing duration of events :

- a : $[2,4]$ time units
- b : $[1,3]$ time units

Contribution : **Embedding of TA into HDTA**

- **Forcing immediate transition** : add a global clock c_T, for any transition
- Examples : TA(left), HDTA (right)

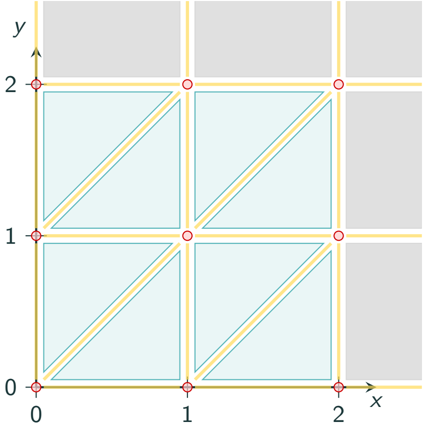


Corollary

Undecidable ✗

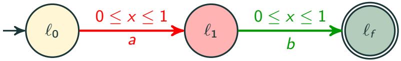
Inclusion of HDTA language

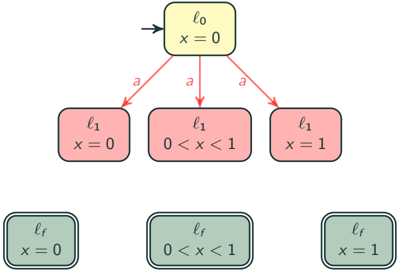
Region of the constraint $0 \leq x, y \leq 2$



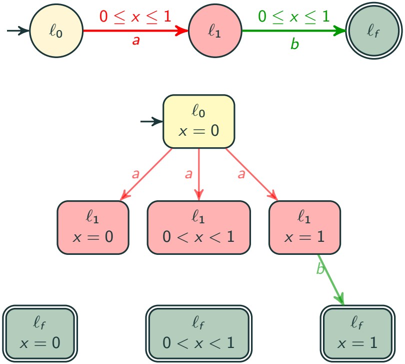
Region Automaton : example for one-clock TA

A timed Automaton $\mathcal{A}$ and its region automaton $R(\mathcal{A})$

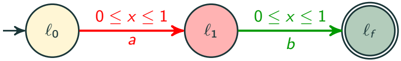


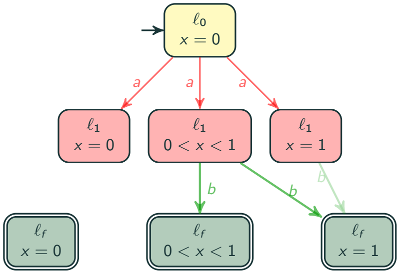


A timed Automaton $\mathcal{A}$ and its region automaton $R(\mathcal{A})$

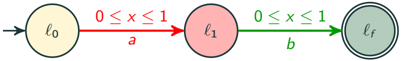


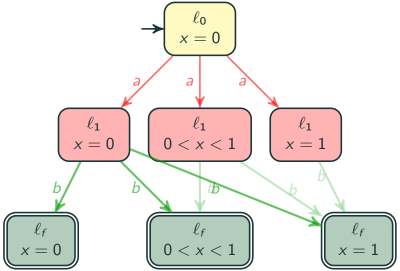
A timed Automaton $\mathcal{A}$ and its region automaton $R(\mathcal{A})$



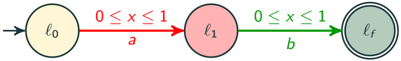


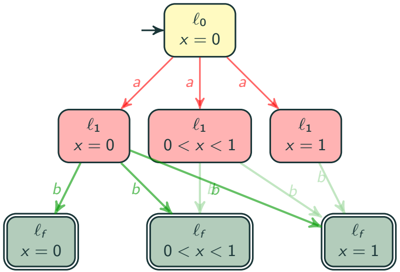
A timed Automaton $\mathcal{A}$ and its region automaton $R(\mathcal{A})$





A timed Automaton $\mathcal{A}$ and its region automaton $R(\mathcal{A})$





Reachability problem for TA : PSPACE (Alur et al, 1994)

Correspondance between runs of TA and the one of the corresponding region automata.

Region equivalence

Let $A = (\Sigma, C, L, \bot_L, \top_L, inv, exit)$ be an HDTA

- $M :=$ the maximal constant appearing in *inv*
- $\cong$: region equivalence on $\mathbb{R}^C_{\geq 0}$ defined as follows : for any $v, v' : C \to \mathbb{R}_{\geq 0}$:
  - $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ or $v(x), v'(x) > M$, $\forall x \in C$,
  - $\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0, \forall x \in C$
  - $\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\}$

Region equivalence

Let $A = (\Sigma, C, L, \bot_L, \top_L, inv, exit)$ be an HDTA

- $M :=$ the maximal constant appearing in *inv*
- $\cong$: region equivalence on $\mathbb{R}^C_{\geq 0}$ defined as follows : for any $v, v' : C \to \mathbb{R}_{\geq 0}$:
  - $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ or $v(x), v'(x) > M$, $\forall x \in C$,
  - $\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0, \forall x \in C$
  - $\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\}$
- For any HDTA A : $unt(\mathcal{L}(A)) = \mathcal{L}(R(A))$

Untimed language inclusion is decidable

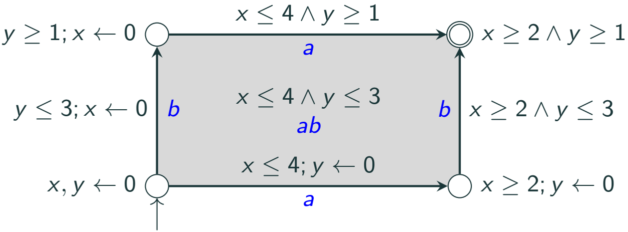
Region equivalence

Let $A = (\Sigma, C, L, \bot_L, \top_L, inv, exit)$ be an HDTA

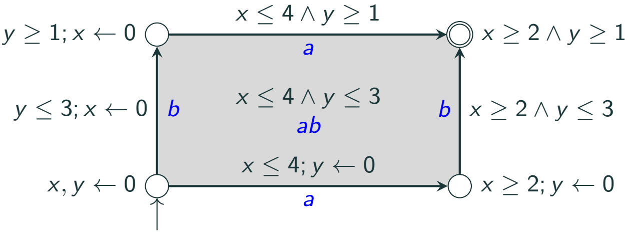
- $M :=$ the maximal constant appearing in *inv*
- $\cong$: region equivalence on $\mathbb{R}^C_{\geq 0}$ defined as follows : for any $v, v' : C \to \mathbb{R}_{\geq 0}$:
  - $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ or $v(x), v'(x) > M$, $\forall x \in C$,
  - $\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0, \forall x \in C$
  - $\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\}$
- For any HDTA A : $unt(\mathcal{L}(A)) = \mathcal{L}(R(A))$
- Consequence :

Decidable ✓

Inclusion of the closed (subsumption) of untimed Language of HDTA



- **Express the Language of Higher Dimensional Timed Automata**



- **Express the Language of Higher Dimensional Timed Automata**
- Decidability/Undecidability results :

Undecidable ✗

Inclusion of HDTA language

Decidable ✓

Inclusion of the closed (subsumption) of untimed Language of HDTA

Current & Future work

Higher Dimensional Automata : **properties, temporal logic**

- **Temporal logic for HDA** : (Erlich, Ledent)

Current & Future work

Higher Dimensional Automata : **properties, temporal logic**

- **Temporal logic for HDA** : (Erlich, Ledent)

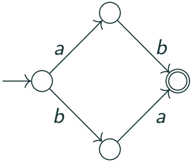
Robustness for Higher Dimensional Timed Automata

- Distance between words
- Guard enlargment
- Delay perturbation
- Topological point of view

Appendix

Two-events HDA

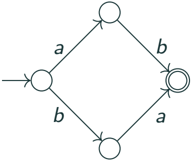
- $a.b + b.a$:



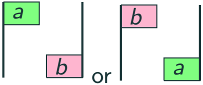
Higher Dimensional Automata : 2 and 3-dimension examples

Two-events HDA

- $a.b + b.a$:

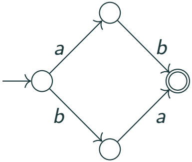


Example of traces :

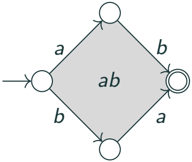


Two-events HDA

- $a.b + b.a$:



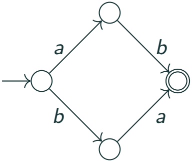
- $a||b$:



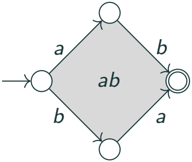
Higher Dimensional Automata : 2 and 3-dimension examples

Two-events HDA

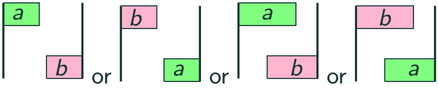
- $a.b + b.a$:



- $a||b$:



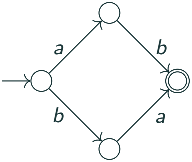
Example of traces :



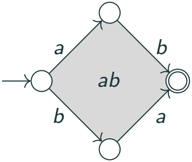
Higher Dimensional Automata : 2 and 3-dimension examples

Two-events HDA

- $a.b + b.a$:

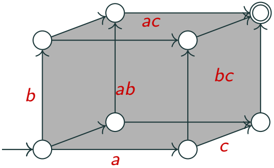


- $a||b$:



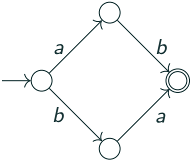
Three-events HDA

- $a||b + b||c + a||c$:

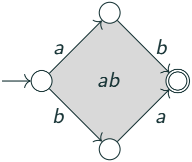


Two-events HDA

- $a.b + b.a$:

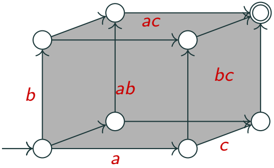


- $a||b$:



Three-events HDA

- $a||b + b||c + a||c$:

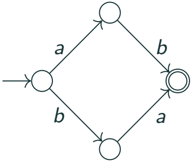


Examples of traces :

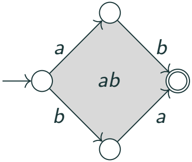
Higher Dimensional Automata : 2 and 3-dimension examples

Two-events HDA

- $a.b + b.a$:

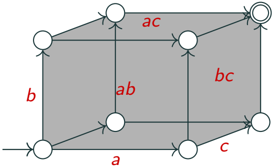


- $a||b$:

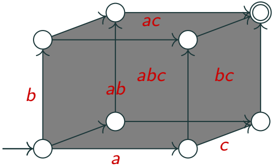


Three-events HDA

- $a||b + b||c + a||c$:



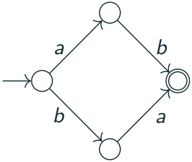
- $a||b||c$:



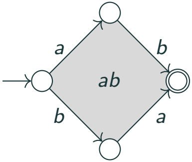
Higher Dimensional Automata : 2 and 3-dimension examples

Two-events HDA

- $a.b + b.a$:

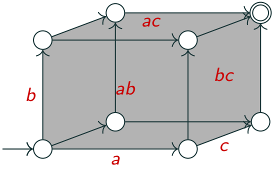


- $a||b$:

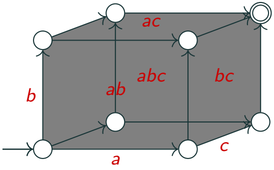


Three-events HDA

- $a||b + b||c + a||c$:

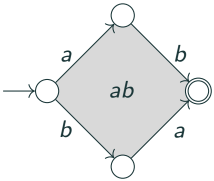


- $a||b||c$:



Examples of traces :

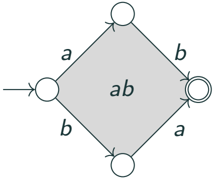
Definition of Higher Dimensional Automata



Higher Dimensional Automata A :

- A tuple $(X, X_\perp, X_\top)$ where X is a finite **precubical set** and $X_\perp$ (*resp.* $X_\top$) $\subseteq X$ a **start (*resp.* accept) cell**.
- Ex : start cell $X_\perp$:  , accept cell $X_\top$: 
  X : {  } $\cup \left\{ \xrightarrow{\lambda} | \lambda \in \{a, b\} \right\}$

Definition of Higher Dimensional Automata



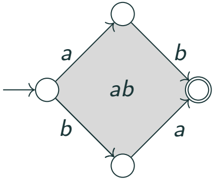
Higher Dimensional Automata A :

- A tuple $(X, X_\perp, X_\top)$ where X is a finite **precubical set** and $X_\perp$ (*resp.* $X_\top$) $\subseteq X$ a **start (*resp.* accept) cell**.
- Ex : start cell $X_\perp$: $\longrightarrow\bigcirc$, accept cell $X_\top$: $\circledcirc$
  $X : \{ \longrightarrow\bigcirc \, , \circledcirc, \bigcirc \, ,$  $\} \cup \left\{ \xrightarrow{\lambda} | \lambda \in \{a, b\} \right\}$

List of events

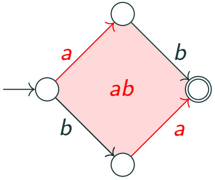
- A **conclist** (concurrent list) : a finite, totally ordered ($\dashrightarrow$) $\Sigma-$labelled set.
- Ex : $\{a, b\}$

Definition of Higher Dimensional Automata



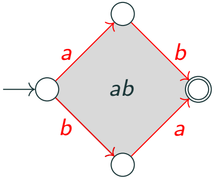
Precubical set X :

- A set of cells $\boldsymbol{X}$.
- **List of active events** of a cell $x \in X$: a conclist $ev(x)$.
  Ex : $\{a\}$**, or** $\{b\}$ **or** $\{a, b\}$**.**



Precubical set X :

- A set of cells $\boldsymbol{X}$.
- **List of active events** of a cell $x \in X$: a conclist $ev(x)$.
  Ex : $\{a\}$, **or** $\{b\}$ **or** $\{a, b\}$.
- **The cells of a list of events** U : $\boldsymbol{X[U] = \{x \in X | ev(x) = U\}}$.
  Ex : $\boldsymbol{X[a]}$



Precubical set X :

- A set of cells $\boldsymbol{X}$.
- **List of active events** of a cell $x \in X$: a conclist $ev(x)$.
  Ex : $\{a\}$**, or** $\{b\}$ **or** $\{a, b\}$**.**
- **The cells of a list of events** U : $X[U] = \{x \in X | ev(x) = U\}$.
  Ex : $X[a]$
- **Lower & Upper faces** : Let U and $A \subseteq U$ be conclists.
  $\delta_A^0 \backslash \delta_A^1$ represent **unstarting\terminating events** A :

$$\boxed{\delta_A^0 : X[U] \to X[U - A], \delta_A^1 : X[U] \to X[U - A]}$$

Paths in an HDA

Sequence $p = (x_0, \varphi_1, x_1, \cdots, x_{n-1}, \varphi_n, x_n)$ *s.t.*

- $x_i \in X$, where x_0 : start cell, x_n : accept cells
- φ : face map type.
- $ev(p_1 * p_2 * \cdots * p_n) = ev(p_1) * \cdots * ev(p_n)$

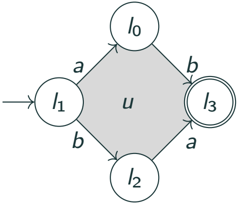
Path of HDA

Paths in an HDA

Sequence $p = (x_0, \varphi_1, x_1, \cdots, x_{n-1}, \varphi_n, x_n)$ *s.t.*

- $x_i \in X$, where x_0 : start cell, x_n : accept cells
- φ : face map type.
- $ev(p_1 * p_2 * \cdots * p_n) = ev(p_1) * \cdots * ev(p_n)$

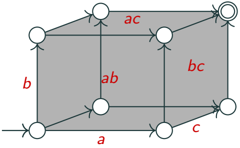
Example of a 2−events HDA



Example of an accepting path :

$$\alpha_0 = l_0 \nearrow^{ab} u \searrow_{ab} l_3,\; ev(\alpha_0) = \left(\begin{bmatrix} a\bullet \\ b\bullet \end{bmatrix} * \begin{bmatrix} \bullet a \\ \bullet b \end{bmatrix} \right)$$

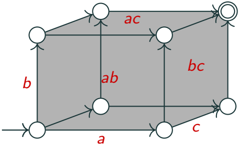
Example of languages

- A_1 : 

$$L_1 = \{abc, acb, bac, bca, cab, cba\} \cup \left\{ \begin{pmatrix} a \\ b \to c \end{pmatrix}, \begin{pmatrix} a \\ c \to b \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ c \to a \end{pmatrix}, \begin{pmatrix} c \\ a \to b \end{pmatrix}, \begin{pmatrix} c \\ b \to a \end{pmatrix} \right\}$$

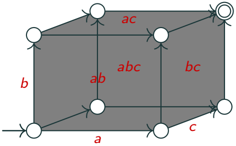
Example of languages

- A_1 :



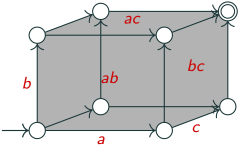
$$L_1 = \{abc, acb, bac, bca, cab, cba\} \cup \left\{ \begin{pmatrix} a \\ b \to c \end{pmatrix}, \begin{pmatrix} a \\ c \to b \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ c \to a \end{pmatrix}, \begin{pmatrix} c \\ a \to b \end{pmatrix}, \begin{pmatrix} c \\ b \to a \end{pmatrix} \right\}$$

- A_2 :

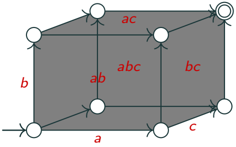


$$: L_2 = L_1 \cup \left\{ \begin{pmatrix} a \\ b \\ c \end{pmatrix} \right\}$$

Example of languages

- A_1 : 

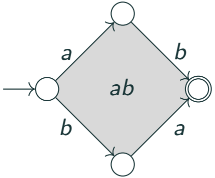
$$L_1 = \{abc, acb, bac, bca, cab, cba\} \cup \left\{ \begin{pmatrix} a \\ b \to c \end{pmatrix}, \begin{pmatrix} a \\ c \to b \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ a \to c \end{pmatrix}, \begin{pmatrix} b \\ c \to a \end{pmatrix}, \begin{pmatrix} c \\ a \to b \end{pmatrix}, \begin{pmatrix} c \\ b \to a \end{pmatrix} \right\}$$

- A_2 : 

$$: L_2 = L_1 \cup \left\{ \begin{pmatrix} a \\ b \\ c \end{pmatrix} \right\}$$

The language of an HDA $A = (X, X_\perp, X_\top)$ is :

$$\boldsymbol{L(A) = \{ev(\alpha) | \alpha \text{ accepting path in } X\}}$$



Precubical set

- Sets $(X_n)_n$
- A set of functions $\left(\delta_{i,n}^{\varepsilon} : X_n \mapsto X_{n-1}\right)_{n>0, i\in\{1,\cdots,n\}, \varepsilon\in\{0,1\}}$ such that :

$$\boxed{\delta_{j,n}^{\varepsilon'} \circ \delta_{i,n+1}^{\varepsilon} = \delta_{i-1,n}^{\varepsilon} \circ \delta_{j,n+1}^{\varepsilon}, \forall i,j}$$

.

Application in HDA

A precubical set on a finite alphabet Σ :

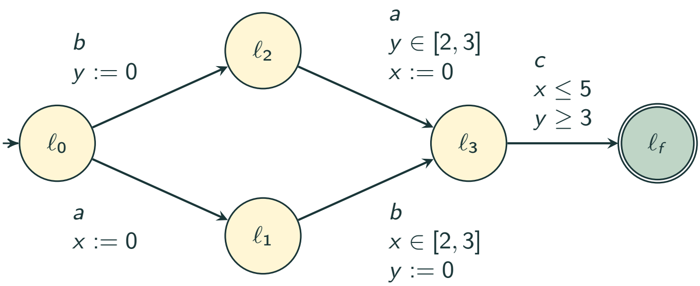
$$X = \left(X, ev, \{\delta_{A,U}^{0}, \delta_{A,U}^{1} | U \in C, A \subseteq U\}\right)$$

where C is the set of conclist over Σ

Timed Automata [1] : example of scheduling

Example of Scheduling of events a, b, c

Time constraints impose that between event a and b, at least (*resp.* at most) 2 (*resp.* 3) time units elapses



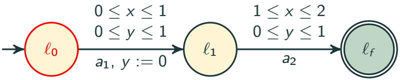
Semantics of transitions

- Delay transitions $(\ell, v) \xrightarrow{\delta} (\ell, v + \delta)$
- Action transitions : $(\ell, v) \xrightarrow{a_1} (\ell_1, v[y := 0])$

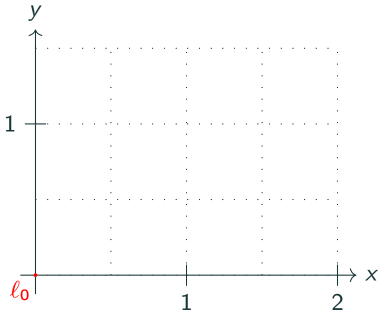
1. AlurD94.

Clocks evolution example

Timed automaton $\mathcal{A}$:

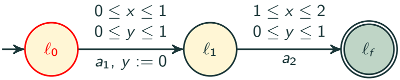


- Evolution of clocks x and y during the run

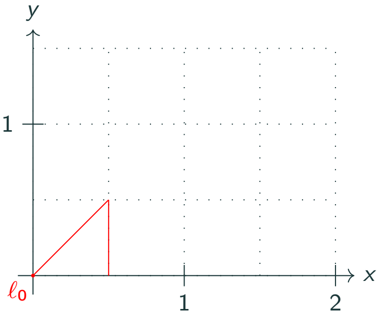


Clocks evolution example

Timed automaton $\mathcal{A}$:

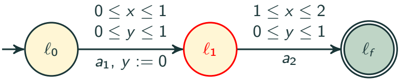


- Evolution of clocks x and y during the run

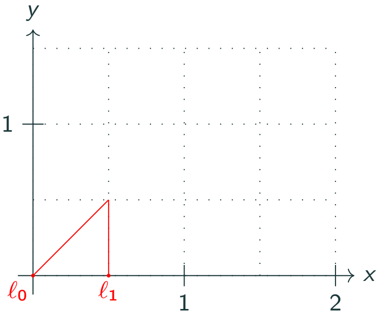


Clocks evolution example

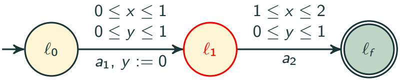
Timed automaton $\mathcal{A}$:



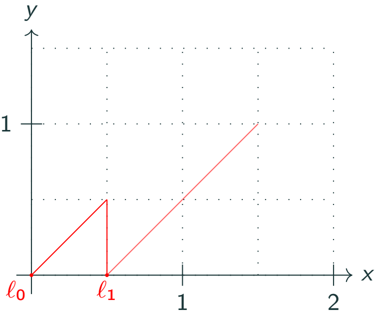
- Evolution of clocks x and y during the run



Timed automaton $\mathcal{A}$:

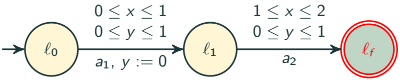


- Evolution of clocks x and y during the run

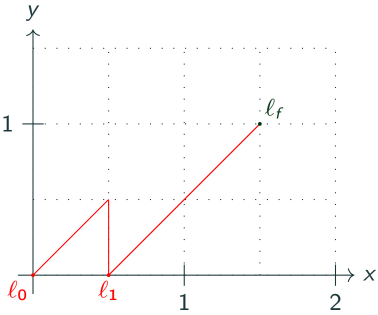


Clocks evolution example

Timed automaton $\mathcal{A}$:



- Evolution of clocks x and y during the run



Delay move, timed words

Delay words

Let us take a run $\pi = (\ell_0, v_0) \cdots \rightarrow \cdots (\ell_i, v_i) \cdots \rightarrow \cdots (\ell_n, v_n)$

- Delay move : $\delta : (\ell, v) \xrightarrow{d} (\ell, v + d)$
  Label of delay move : $ev(\delta) = d$
- Action move : $\delta : (\ell, v) \xrightarrow{a_1} (\ell_1, v[y := 0])$
  Label of action move : $ev(\delta) = a$
- Label of a run π :

$$ev((\ell_0, v_0) \rightarrow (\ell_1, v_1)) \cdots ev((\ell_{n-1}, v_{n-1}) \rightarrow (\ell_n, v_n))$$

Timed words

- Definition : $TW = \{w = (a_0, t_0) \cdots (a_n, t_n) t_{n+1} | \forall i = 0, \cdots n, t_i \leq t_{i+1}\} \subseteq (\Sigma \times \mathbb{R}_{\geq 0})^* \mathbb{R}_{\geq 0}$
- Concatenation : let $w = (a_0, t_0) \cdots (a_n, t_n) t_{n+1} w' = (a'_0, t'_0) \cdots (a'_n, t'_n) t'_{n+1} \in TW$ then :

$$\boxed{ww' := (a_0, t_0) \cdots (a_n, t_n)(a'_0, t'_0) \cdots (a'_n, t'_n)(t_{n+1} + t'_{n+1}) \in TW}$$

Finally : $\mathcal{L}(A)$: the set of delay words labeling accepting path in the transition system.

Differences between TA and HDTA

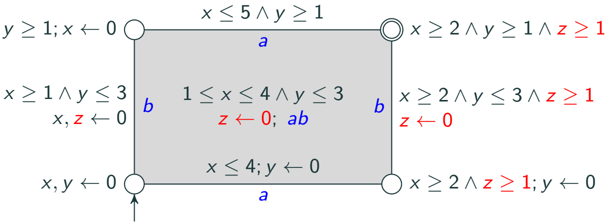
Cells

- 0-cells : location,
- $1-$cell : edges,
- $d-$cell, $d > 1$.

Differences

	TA	HDTA
Difference between locations, edges	Yes	No
Exit conditions	Edges	On $d-$cells, $\forall d$
Invariants	Locations	On $d-$cells, $\forall d$
Reset	Edges	On $d-$cells, $\forall d$
Events	Instantaneous	With duration
Concurrency	Interleaving	Possibly simultaneous

Example of 2-dimension HDTA with 3 clocks



Timing duration of events :

- a : $[2, 4]$ time units
- b : $[1, 3]$ time units

Constraints between starting/ending dates

- 1 time unit should elapse between b's starting date and a's starting date
- 1 time unit should elapse between b's ending date and a's ending date

Interval Delay words, untimed of Timed Ipomsets

Timed Ipomsets and Interval delay words

- Timed Ipomsets : (P, σ_P, d_P).

Timed Ipomsets and Interval delay words

- Timed Ipomsets : (P, σ_P, d_P).
- Steps sequence (HDA)

$$(S_{Q_\mathbf{0}}, Q_0, T_{Q_\mathbf{0}}) * (S_{Q_\mathbf{1}}, Q_1, T_{Q_\mathbf{1}}) * \cdots * (S_{Q_n}, Q_n, T_{Q_n}) \ s.t \ T_{Q_i} = S_{Q_{i+\mathbf{1}}}$$

Interval Delay words, untimed of Timed Ipomsets

Timed Ipomsets and Interval delay words

- Timed Ipomsets : (P, σ_P, d_P).
- Steps sequence (HDA)

$$(S_{Q_0}, Q_0, T_{Q_0}) * (S_{Q_1}, Q_1, T_{Q_1}) * \cdots * (S_{Q_n}, Q_n, T_{Q_n}) \ s.t \ T_{Q_i} = S_{Q_{i+1}}$$

- Interval delay words : steps sequence interspersed with delays (start/termination of events).

Interval Delay words, untimed of Timed Ipomsets

Timed Ipomsets and Interval delay words

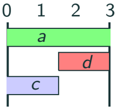
- Timed Ipomsets : (P, σ_P, d_P).
- Steps sequence (HDA)

$$(S_{Q_0}, Q_0, T_{Q_0}) * (S_{Q_1}, Q_1, T_{Q_1}) * \cdots * (S_{Q_n}, Q_n, T_{Q_n}) \ s.t \ T_{Q_i} = S_{Q_{i+1}}$$

- Interval delay words : steps sequence interspersed with delays (start/termination of events).

Untimed of Timed Ipomsets

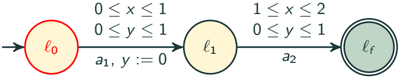
- Untimed $unt((P, \sigma_P, d_P)) = (P, <_P, \dashrightarrow_P, S, T, \lambda)$

Ex : T = 

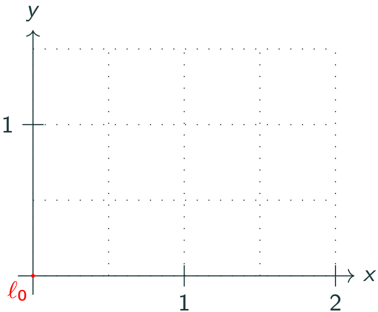
, $unt(T) = \begin{bmatrix} \bullet a \bullet \\ \bullet c \to d \end{bmatrix}$

Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

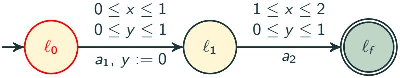


- Run with delay perturbations of at most $\delta = 0.2$

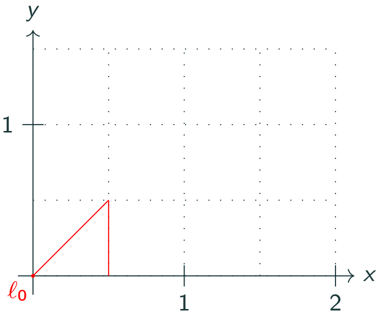


Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

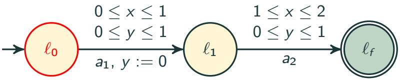


- Run with delay perturbations of at most $\delta = 0.2$

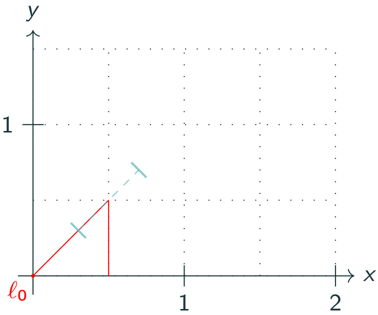


Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

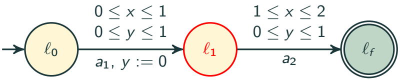


- Run with delay perturbations of at most $\delta = 0.2$

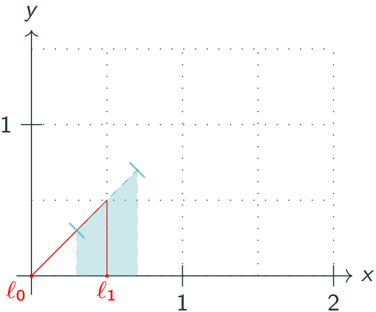


Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

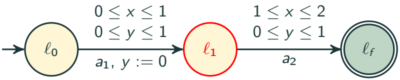


- Run with delay perturbations of at most $\delta = 0.2$

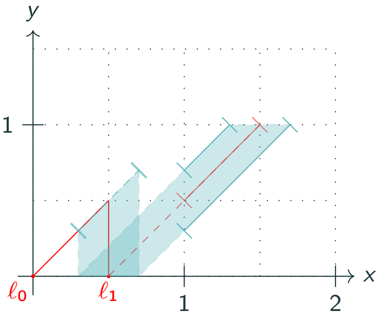


Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

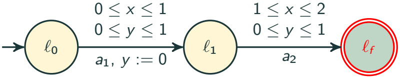


- Run with delay perturbations of at most $\delta = 0.2$

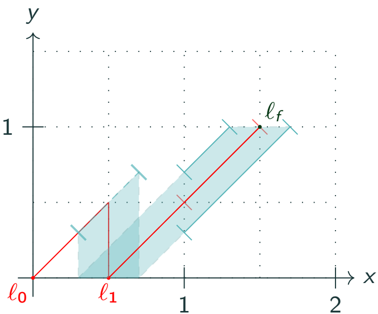


Future work : What about the robustness ?

Timed automaton $\mathcal{A}$:

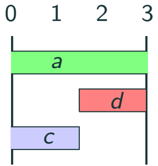


- Run with delay perturbations of at most $\delta = 0.2$



Future work : robustness on HDTA

No timing perturbation : c and d are not in concurrency



timing perturbation. Let us introduce a 0.1 delay on the end date of c :