Higher Dimensional Timed Automata

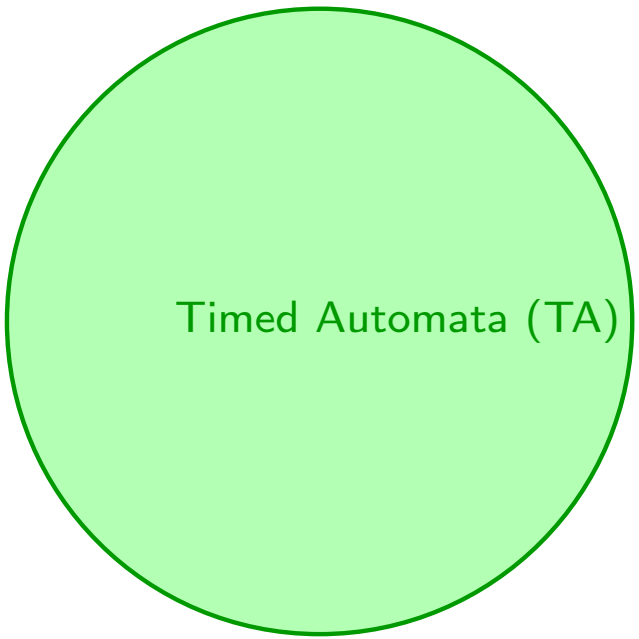
Amazigh Amrane [2] Hugo Bazille [2] **Emily Clement** [1] Uli Fahrbenberg [2]

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

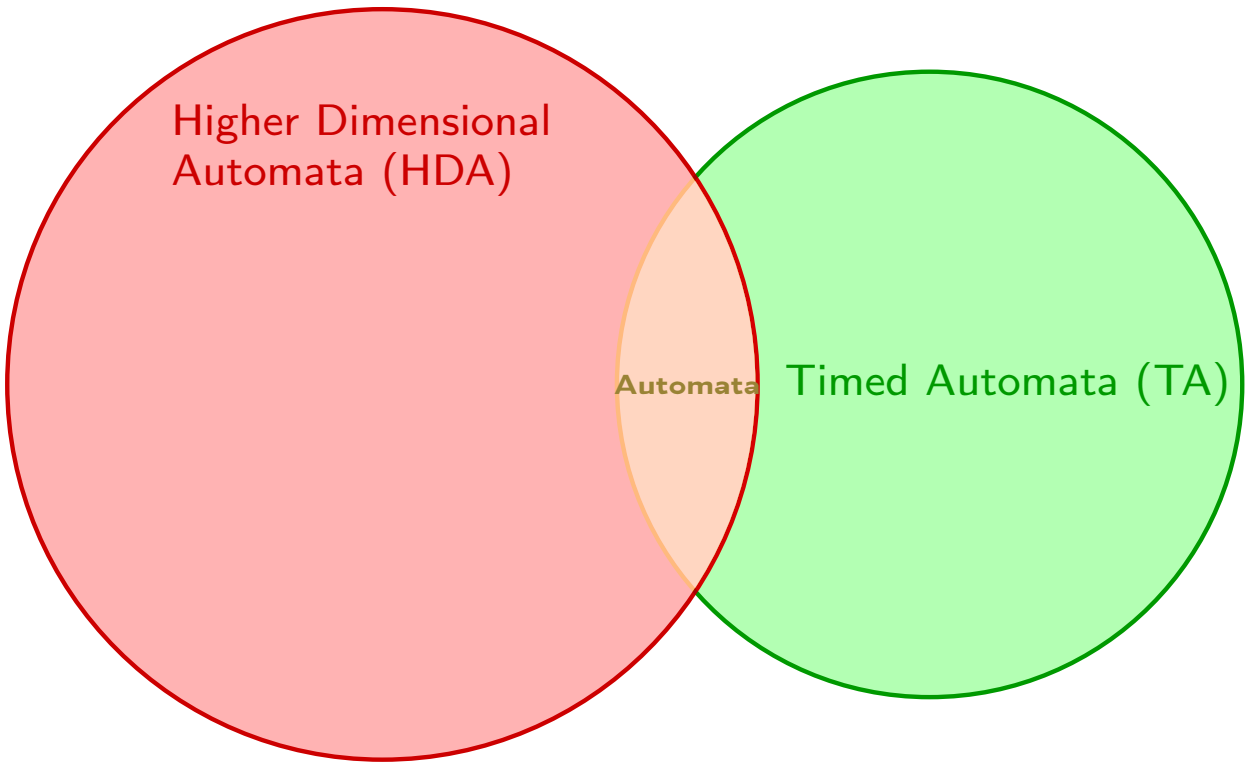
14th of December 2023



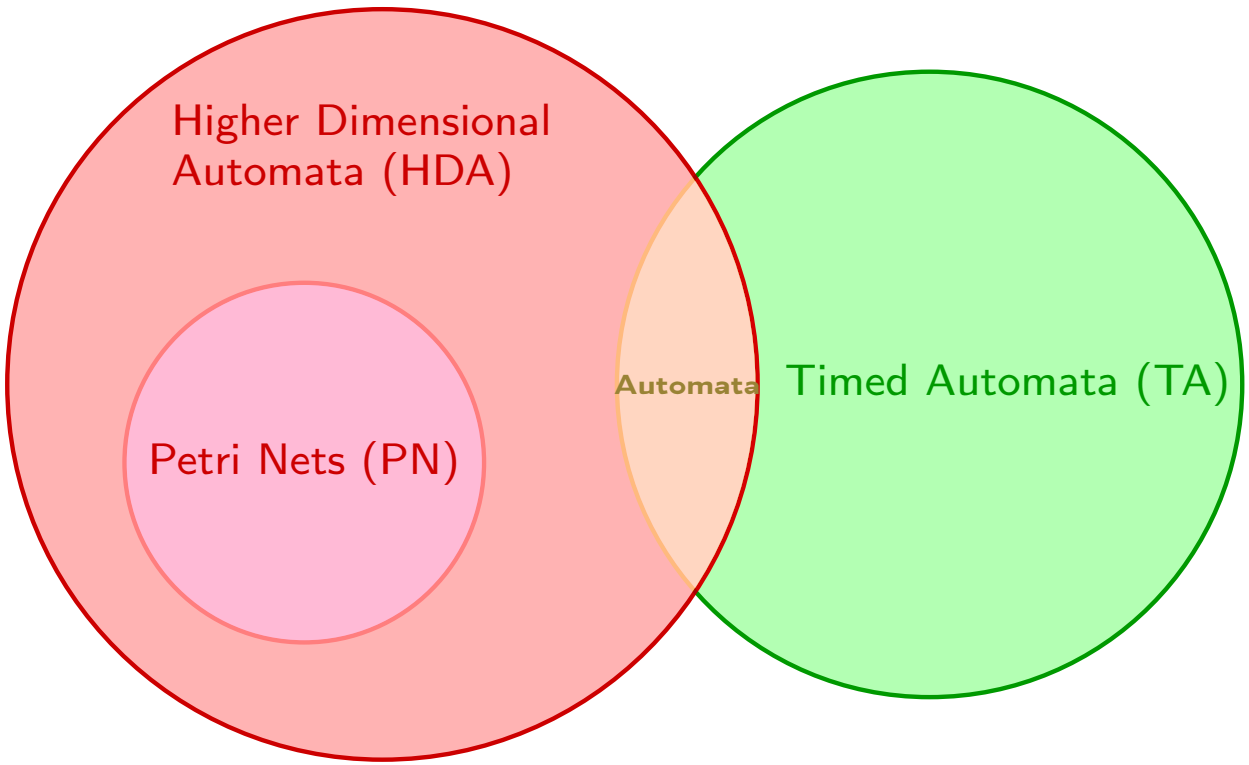

Motivation



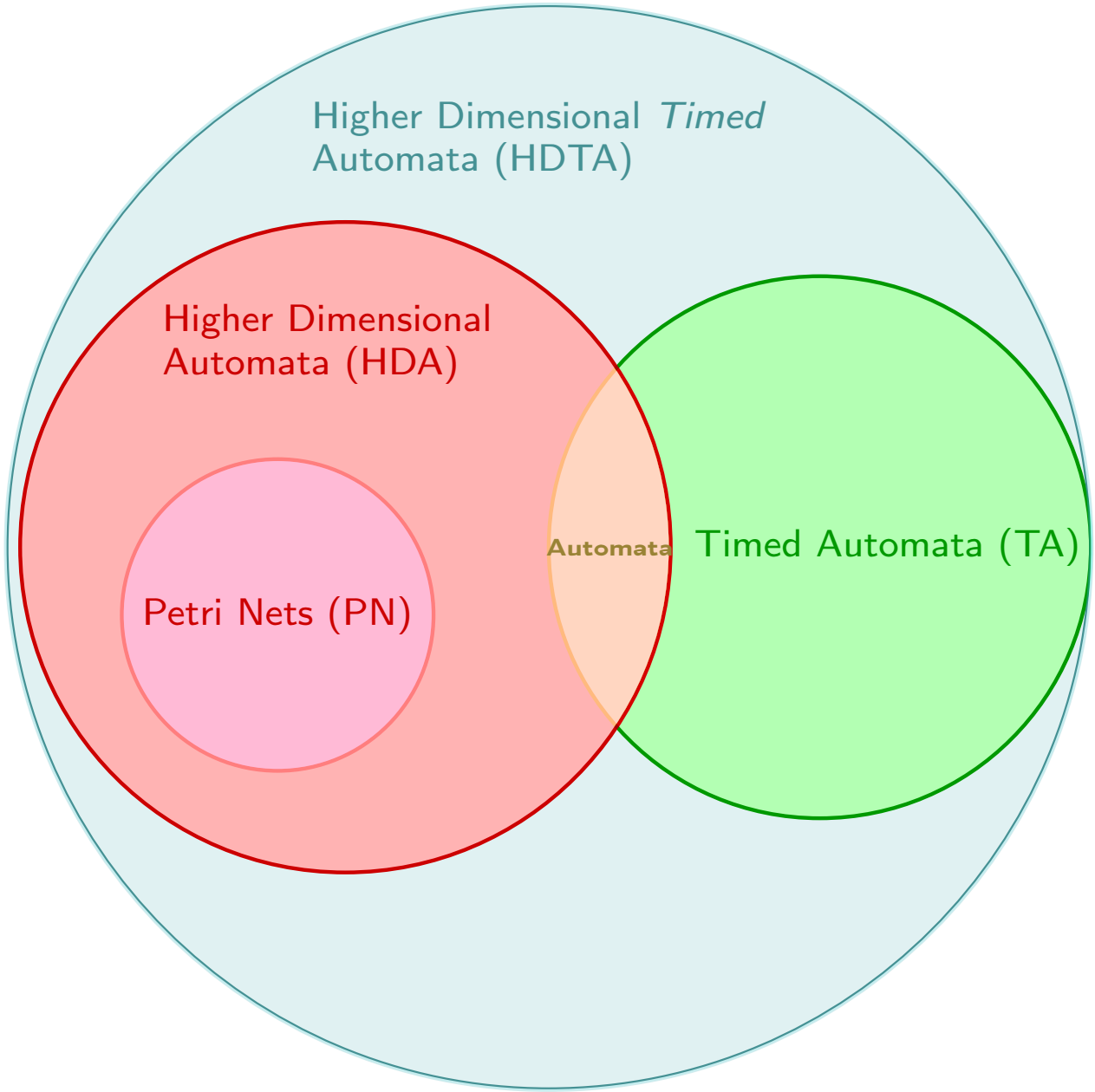
Motivation


Higher Dimensional
Automata (HDA)
Automata
Timed Automata (TA)

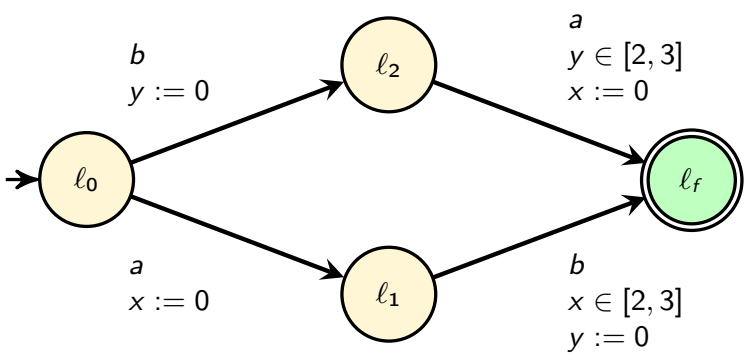
Motivation


Higher Dimensional
Automata (HDA)
Petri Nets (PN)
Automata
Timed Automata (TA)

Motivation

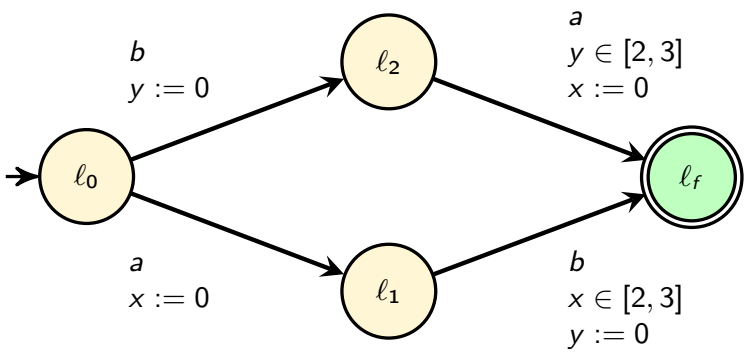

Higher Dimensional *Timed* Automata (HDTA)
Higher Dimensional Automata (HDA)
Petri Nets (PN)
Automata
Timed Automata (TA)

- Timed Automata
  - **Alur & al**: 1994
  - **Application**: synchronous real-time systems
  - **Decidable Problems**: Reachability, Emptriness, LTL MC PSPACE-Complete...
  - **Undecidable Problem**: Universality.

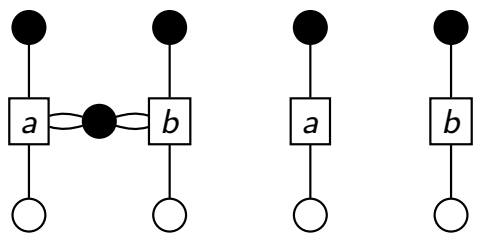


Models and applications

- Timed Automata
  - **Alur & al**: 1994
  - **Application**: synchronous real-time systems
  - **Decidable Problems**: Reachability, Emptriness, LTL MC PSPACE-Complete...
  - **Undecidable Problem**: Universality.

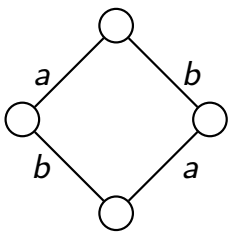


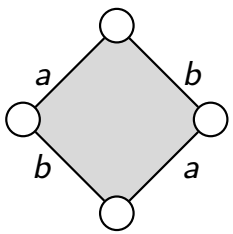
- Petri Nets
  - **Petri**: 1962
  - **Applications**: Discrete events, concurrency.



Our goal/motivation

- Concurrency in Automata and their variants
  - **Timed Automata/Automata**: $b.a + a.b$.
  - **Higher dimensional Automata**: $a||b$.

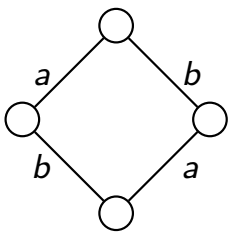


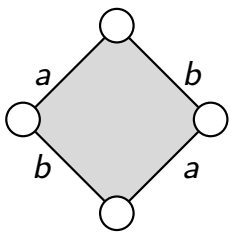


- Goal: model Time duration of events & concurrency

Our goal/motivation

- Concurrency in Automata and their variants
  - **Timed Automata/Automata**: $b.a + a.b$.
  - **Higher dimensional Automata**: $a||b$.





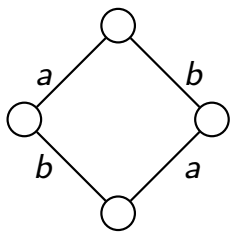
- Goal: model Time duration of events & concurrency

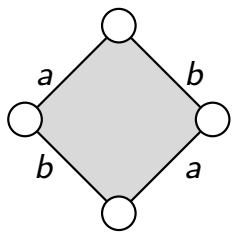
- Application: Real-time concurrent systems

Ex: distributed cyber-physical systems

Our goal/motivation

- Concurrency in Automata and their variants
  - **Timed Automata/Automata**: $b.a + a.b$.
  - **Higher dimensional Automata**: $a||b$.





- Goal: model Time duration of events & concurrency

- Application: Real-time concurrent systems

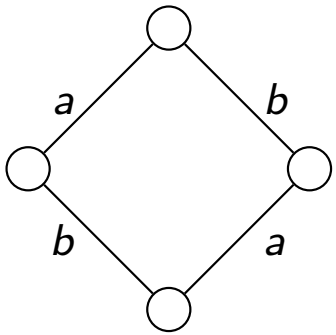
Ex: distributed cyber-physical systems

- Method: Higher Dimensional Timed Automata

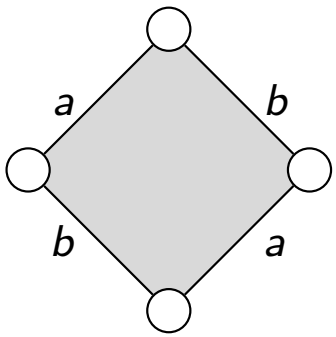
Augment Higher Dimensional Automata with **time duration** of events.

Higher Dimensional Automata: Intuition

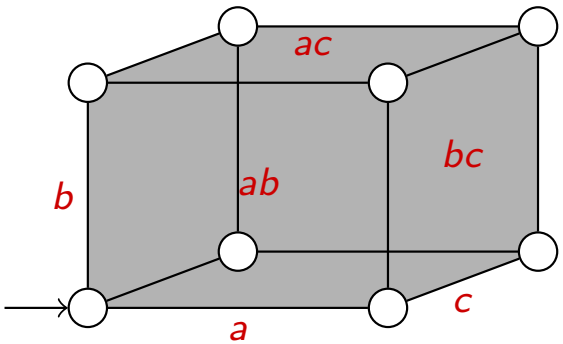
- Examples with two events
  - $a.b + b.a$:



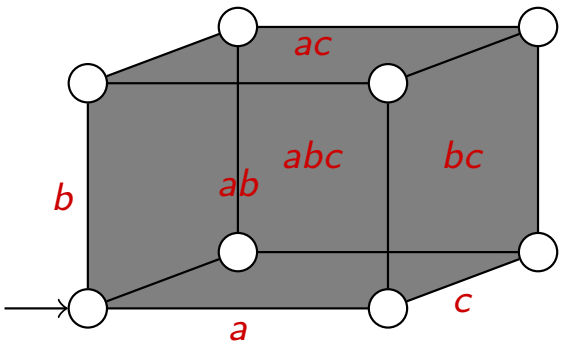
  - $a||b$:



- Examples with three events
  - $a||b + b||c + a||c$:



  - All events independent:

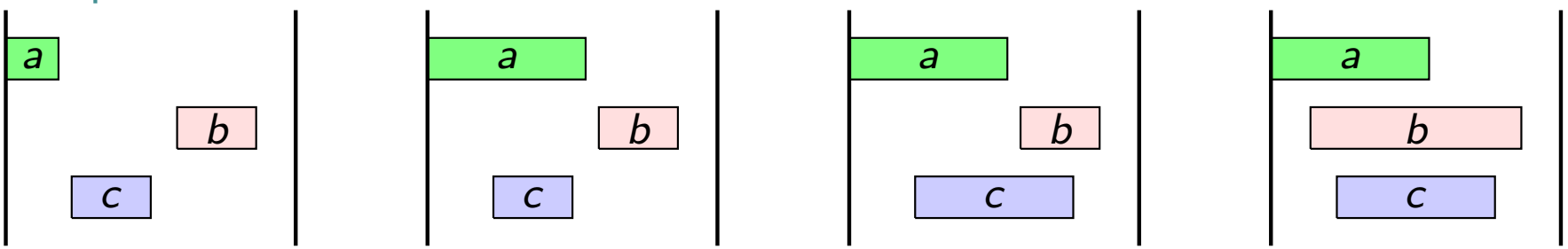


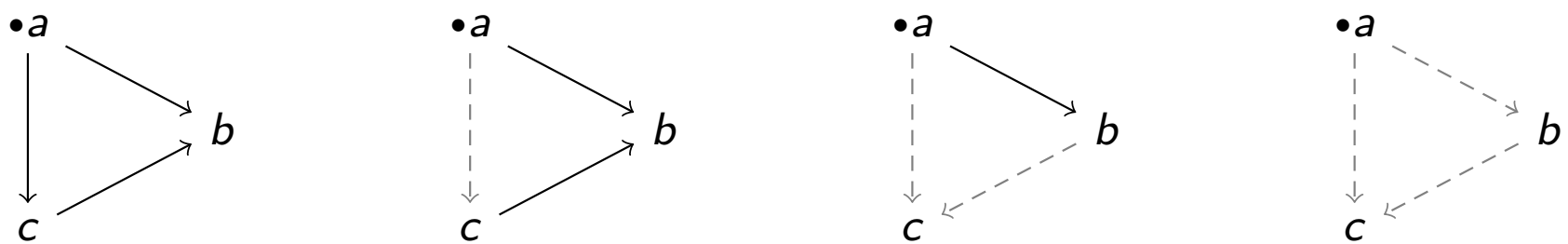
Events representation: pomset with interface

- Two partial order events
  - $<$: precedence order (rep with $\longrightarrow$)
  - $\dashrightarrow$: event order.
  - $< \cup \dashrightarrow$: **total**.
- Interfaces

Source/Target interfaces: S/T: $<$ −minimal/maximal.

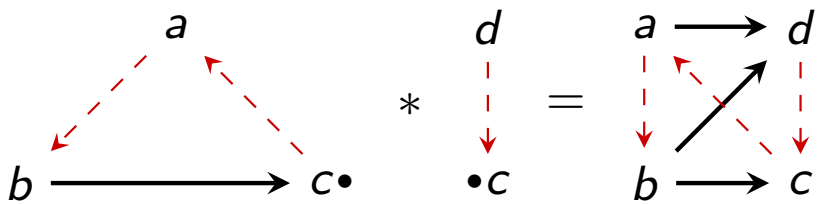
- Representation of events as interval



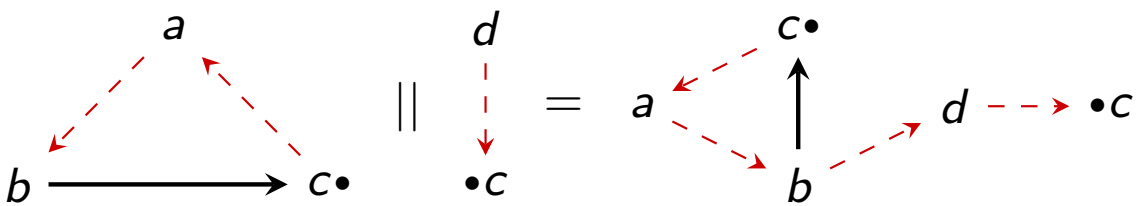


Example of operations on ipomsets

- Gluing composition:



- Parallel composition



Definition of Higher Dimensional Automata

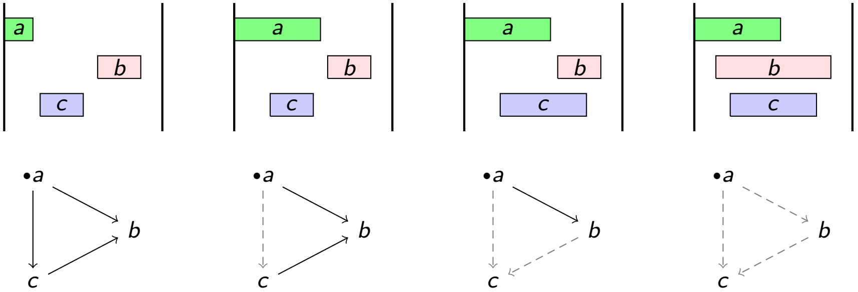
- An Higher Dimensional Automata A:
  - A tuple $(X, X_\perp, X_\top)$ where X is a finite precubical set, $X_\perp$ (*resp.* $X_\top$) $\subseteq X$ a start (*resp.* accept) cell and λ a labelling function.
- List of events
  - a conclist (concurrent list): a finite, totally ordered ($\dashrightarrow$) $\Sigma-$labelled set.
- Precubical set X:
  - a set of cells X, each $x \in X$ associated with a conclist $ev(x)$ (a list of active events in cell x)
  - $X[U] = \{x \in X | ev(x) = U\}$ represents the cell of type U, for a conclist U.
  - For each conclist U and $A \subseteq U$, we can define: lower\upper face map ($\delta^0_A \backslash \delta^1_A$): that represents unstarting\terminating events A:

$$\delta^0_A : X[U] \to X[U - A], \delta^1_A : X[U] \to X[U - A]$$

  - Respects the identity:

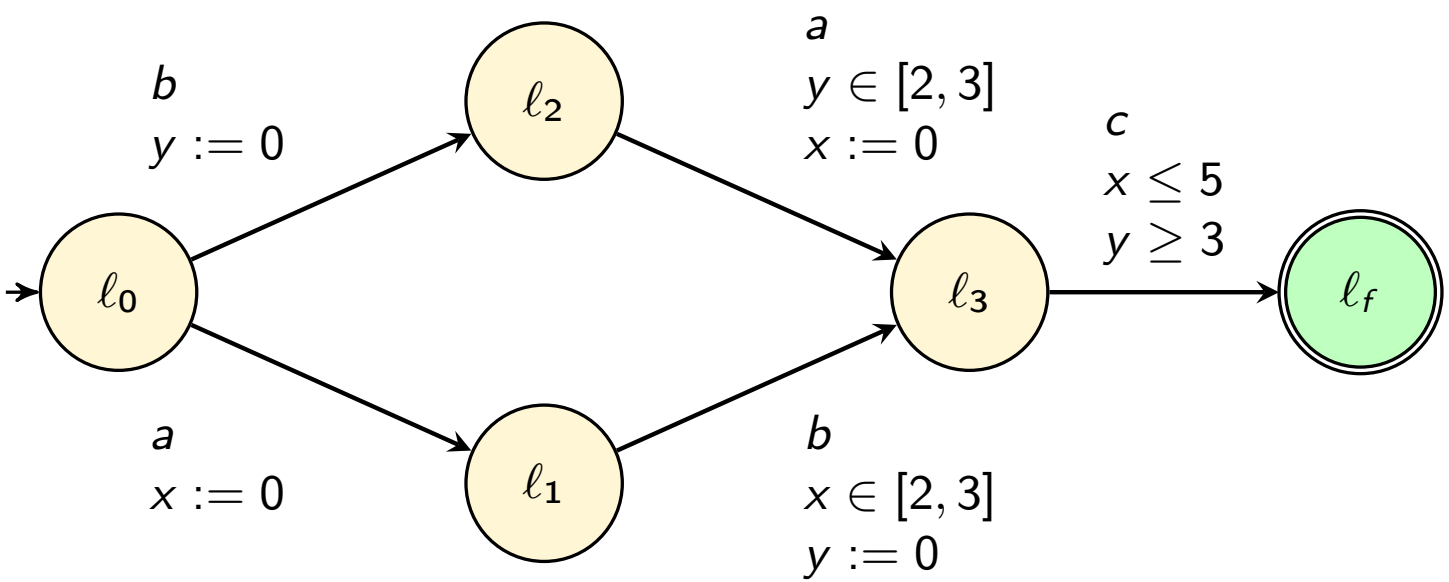
$$\boxed{\delta^\mu_A \delta^\nu_B = \delta^\nu_B \delta^\mu_A \ : \ \forall A, B \ s.t. \ A \cap B = \varnothing, \forall \mu, \nu \in \{0, 1\}}$$

What about time?



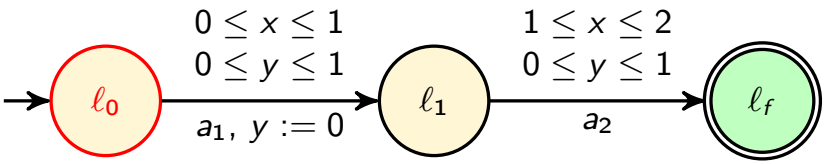
Timed Automata: example of scheduling

- 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
  - Resets ($x := 0$) of clocks

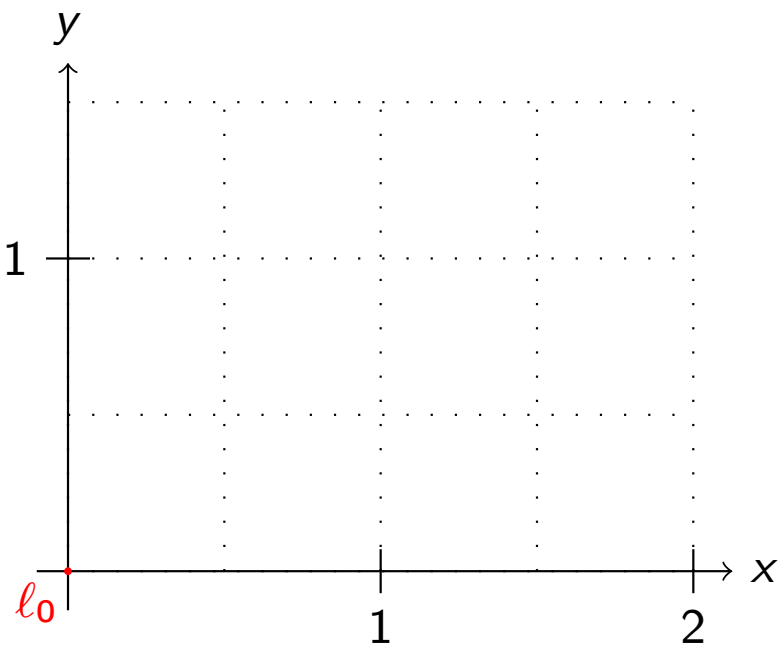


Clocks evolution example

- Timed automaton $\mathcal{A}$:

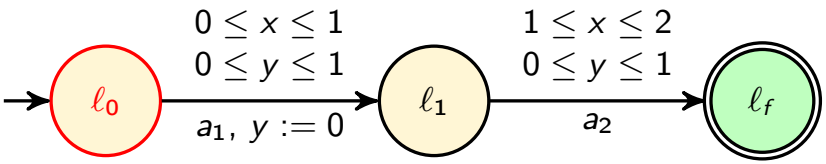


- Evolution of clocks x and y during the run

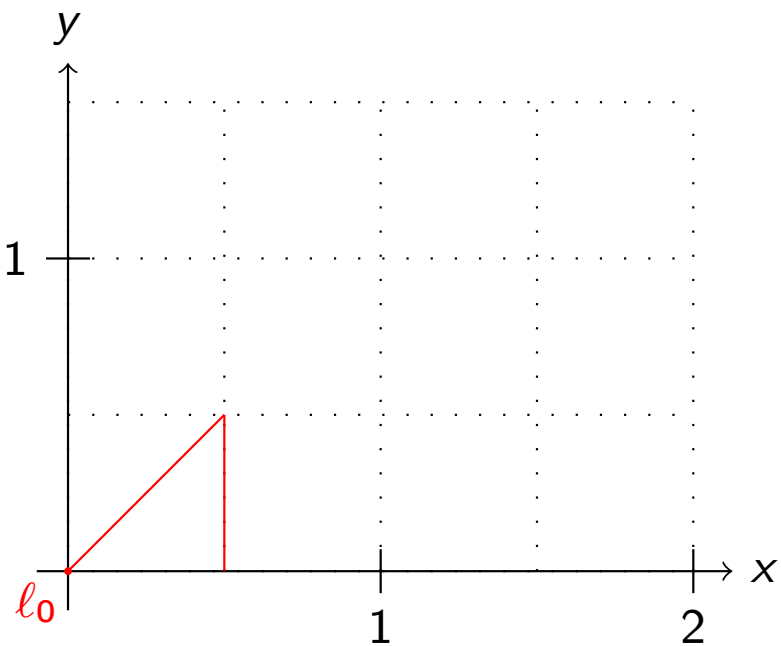


Clocks evolution example

- Timed automaton $\mathcal{A}$:

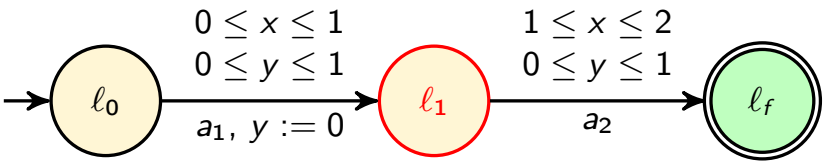


- Evolution of clocks x and y during the run

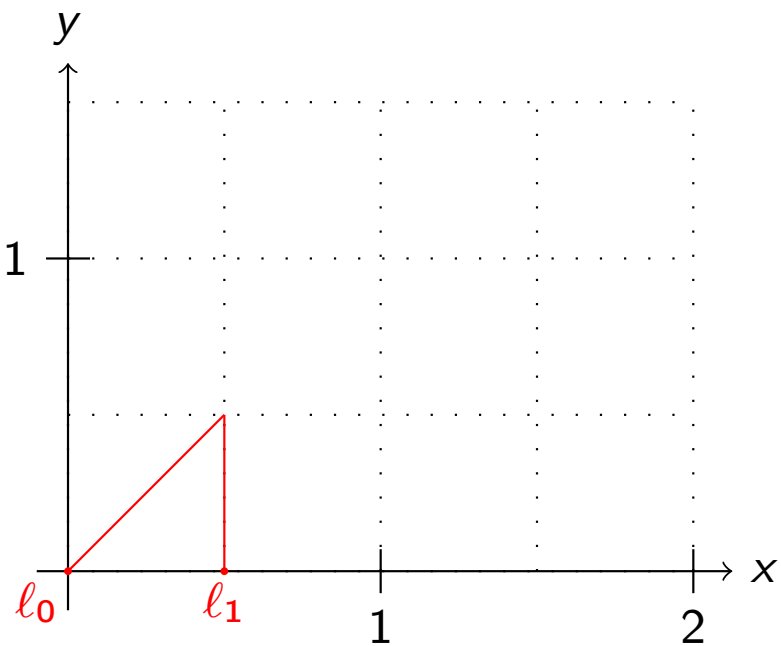


Clocks evolution example

- Timed automaton $\mathcal{A}$:

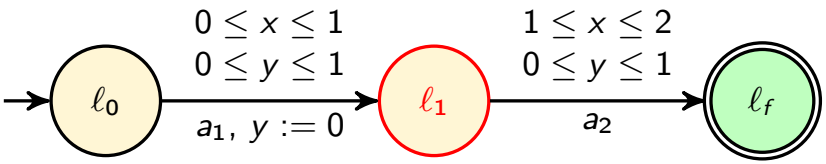


- Evolution of clocks x and y during the run

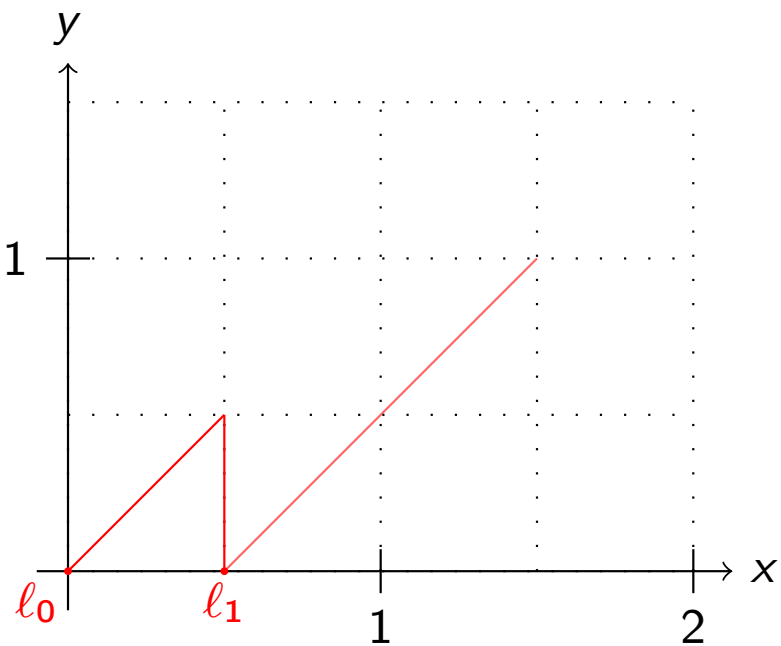


Clocks evolution example

- Timed automaton $\mathcal{A}$:

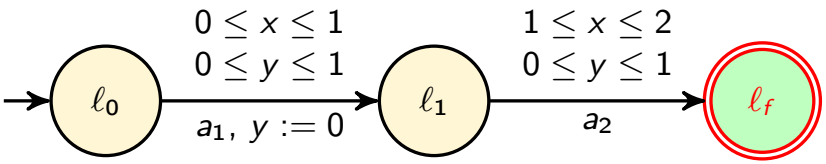


- Evolution of clocks x and y during the run

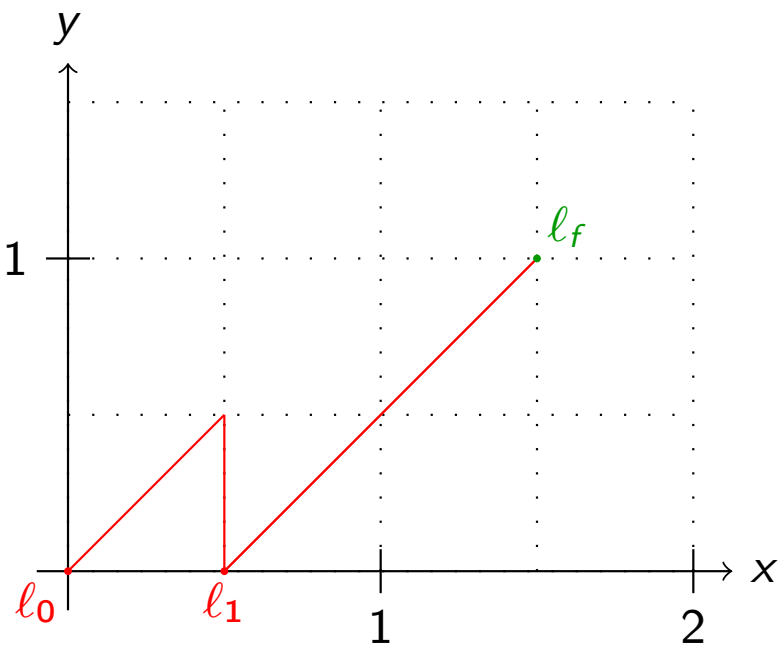


Clocks evolution example

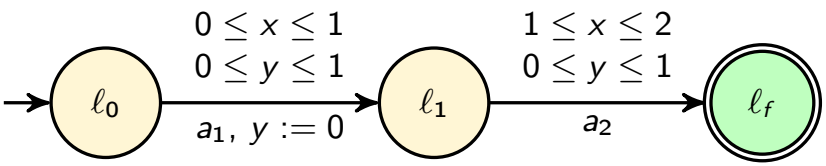
- Timed automaton $\mathcal{A}$:



- Evolution of clocks x and y during the run



Events: semantics of transitions



- Delay-based transitions

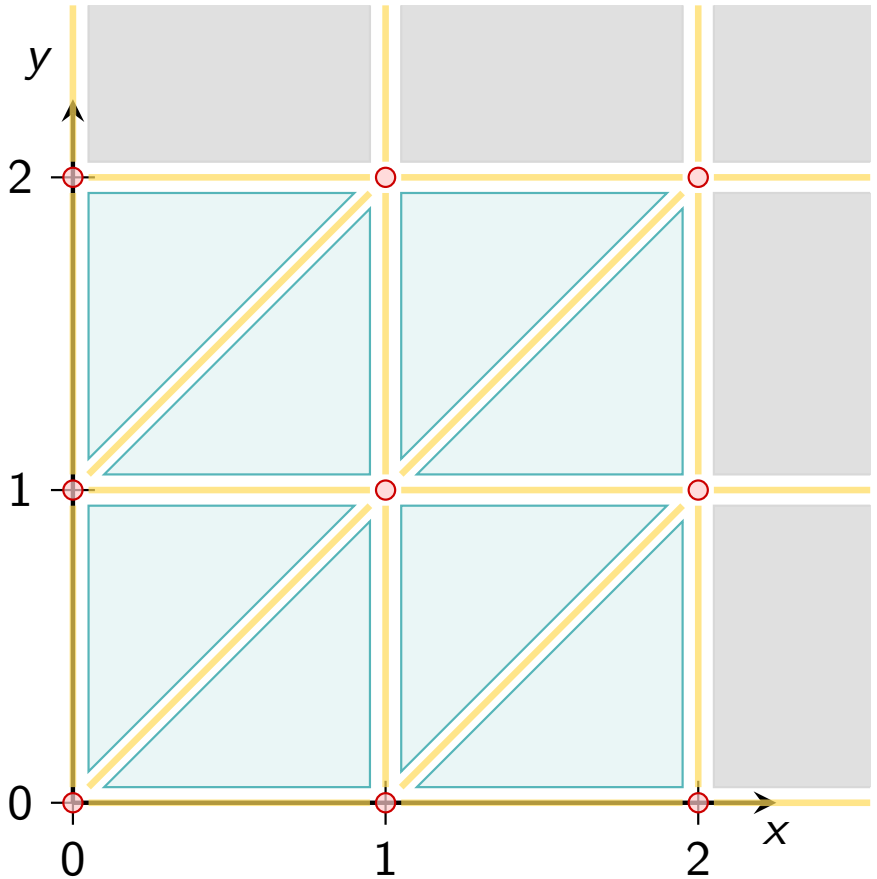
$$(\ell, v) \xrightarrow{\delta} (\ell, v + \delta)$$

- Action-based transitions

$$(\ell, v) \xrightarrow{a_1} (\ell_1, v[y := 0])$$

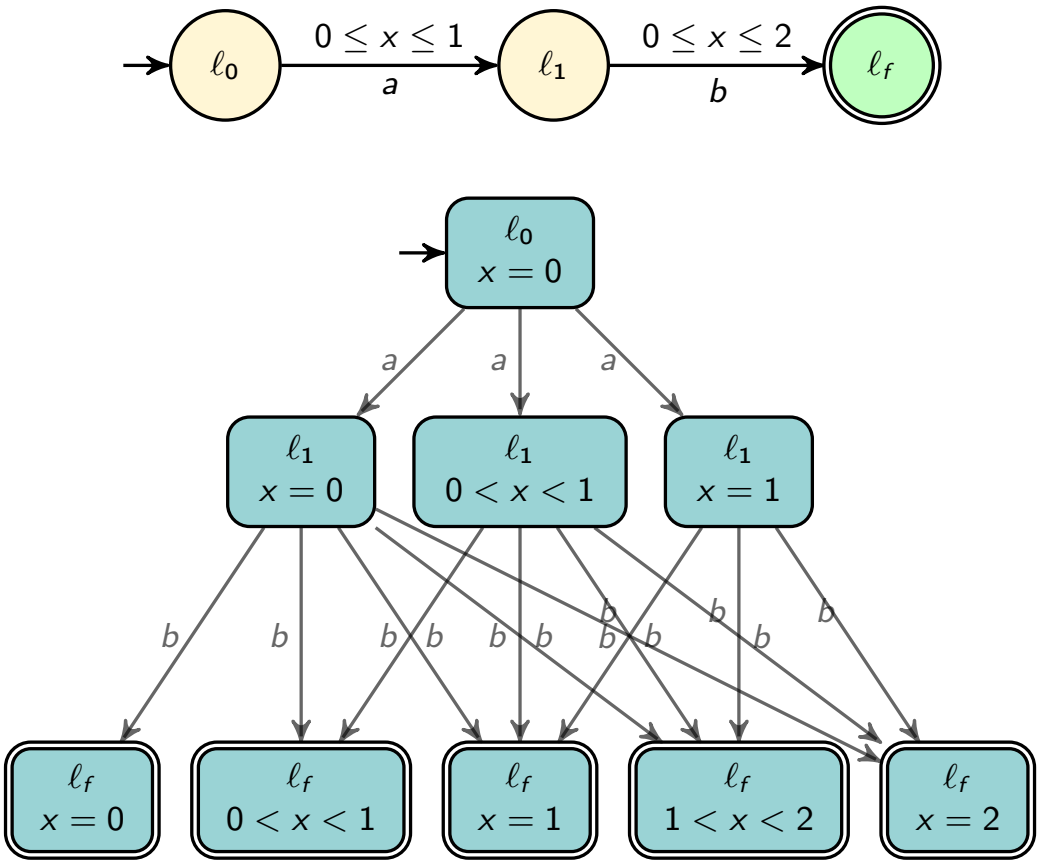
Region

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



Region Automaton: example for one-clock TA

- A timed Automaton and its region automaton

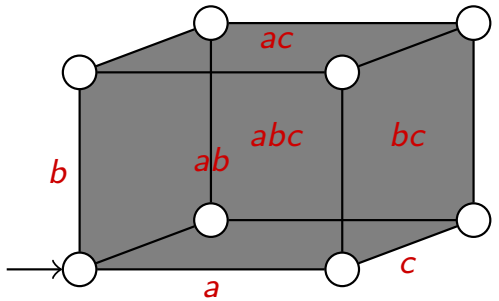


- Reachability problem for TA

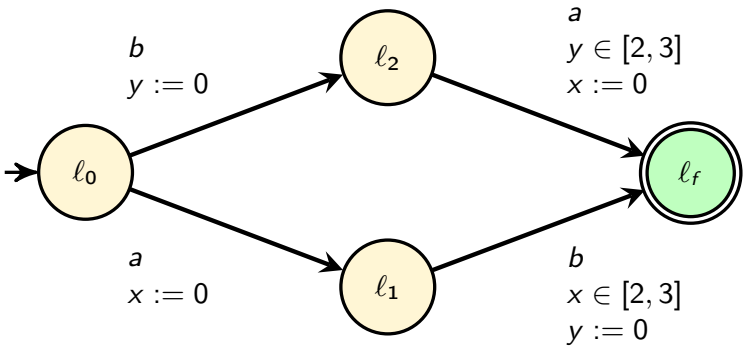
PSPACE (Alur et al, 1994): correspondance between runs of TA and the one of the corresponding region automata.

Motivation

- Higher Dimensional Automata:
  - Events can happen in parallel, but...
  - No timing duration information nor timing constraints

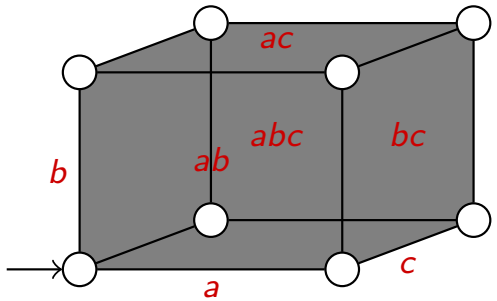


- Timed Automata
  - Timing constraints, but...
  - Instantaneous events
  - interleaving concurrency

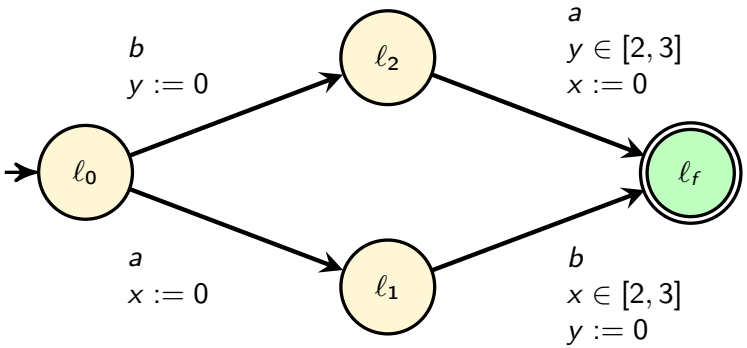


Motivation

- Higher Dimensional Automata:
  - Events can happen in parallel, but...
  - No timing duration information nor timing constraints



- Timed Automata
  - Timing constraints, but...
  - Instantaneous events
  - interleaving concurrency

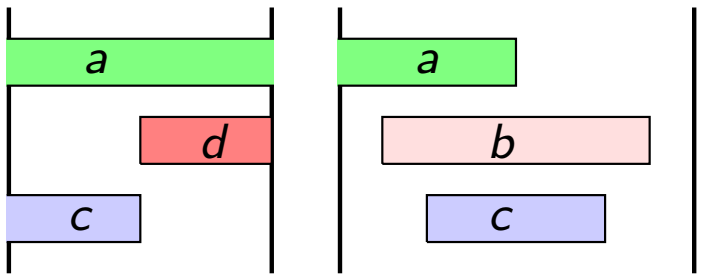


- Our goal: study of Higher Dimensional *Timed* Automata
  - Represent timing constraints
  - Consider timing duration of events

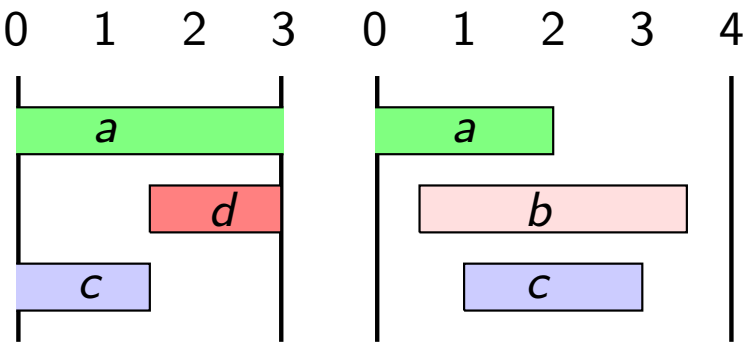
Represent ipomset with timing information

- Timed ipomsets is composed of:
  - An ipomset
  - A duration
  - A map labelling all events to time intervals

- Ipomsets in HDA:



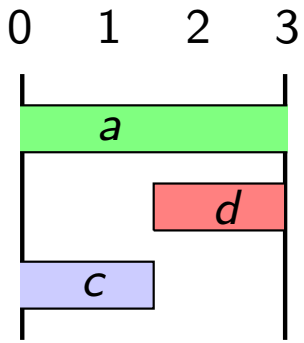
- Timed ipomsets:

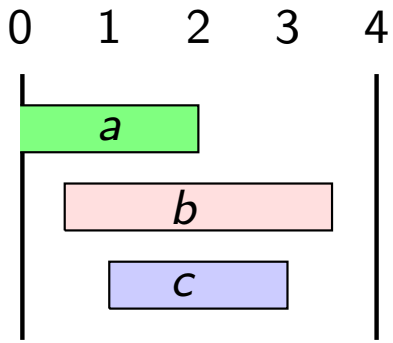


- Example of Timed Ipomset:
  - ipomset: $(\{x_1, x_2, x_3\}, <, x_1 \dashrightarrow x_2 \dashrightarrow x_3, \{x_1, x_3\}, \{x_2\}, \lambda_1)$
  - $\lambda(x_1) = a, \lambda(x_2) = d, \lambda(x_3) = c$
  - $d = 3$
  - $\sigma(a) = (0, 3), \sigma(c) = (0, 1.5), \sigma(d) = (1.5, 3)$

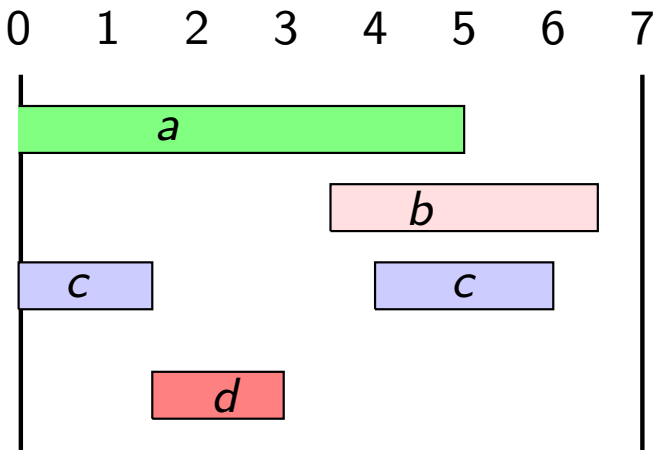
Operation on Timed Ipomsets

- Timed ipomsets T_1, T_2:





- Gluing of T_1 and T_2



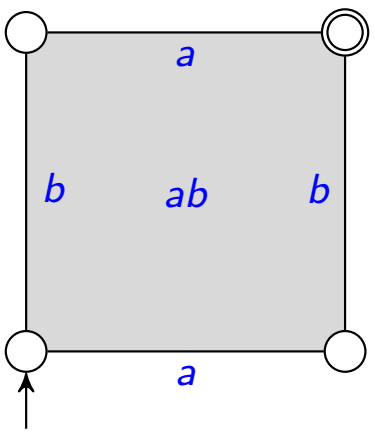
Higher Dimensional Timed Automata: definition

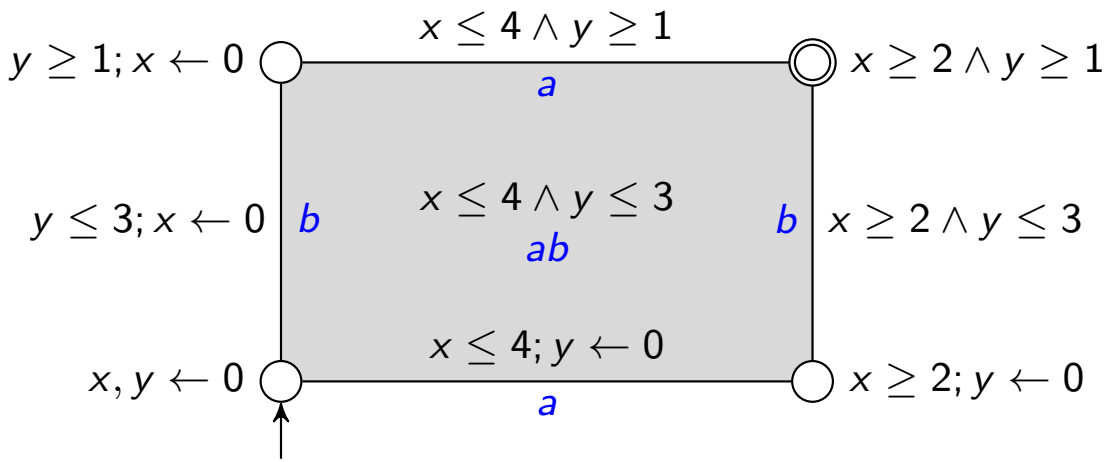
- Definition:

A HDTA is a tuple $(X, X_\perp, X_\top, \lambda, \mathcal{C}, \text{inv}, \text{exit})$ where:

  - $(X, X_\perp, X_\top, \lambda)$ is an HDA
  - $\mathcal{C}$: set of clocks
  - $\text{inv} : X \to \phi(\mathcal{C})$ (*resp.* $\text{exit} : X \to 2^{\mathcal{C}}$) assign invariant (*resp.* exit) conditions to cells.

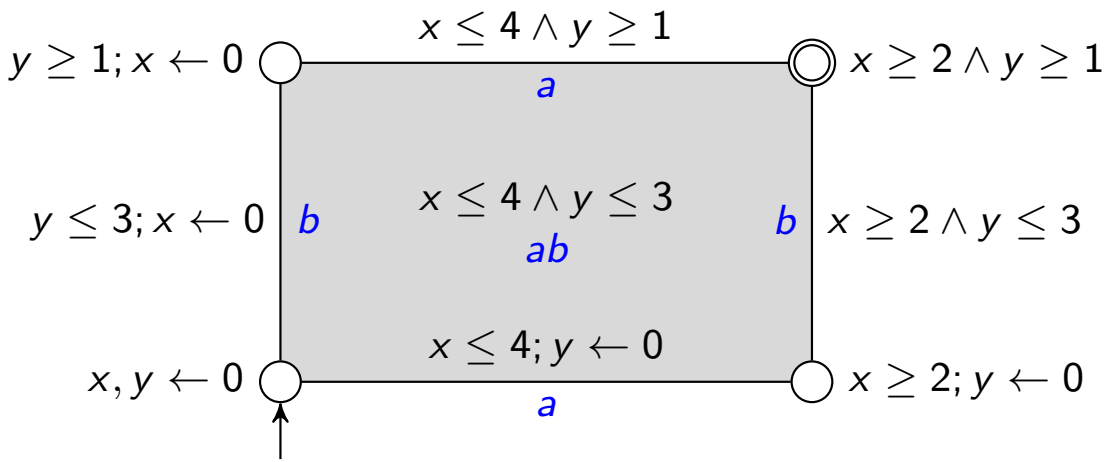
- Example with events a and b: HDA (left) of the HDTA (right)





Examples of HDTA

- First example: adding timing duration to events

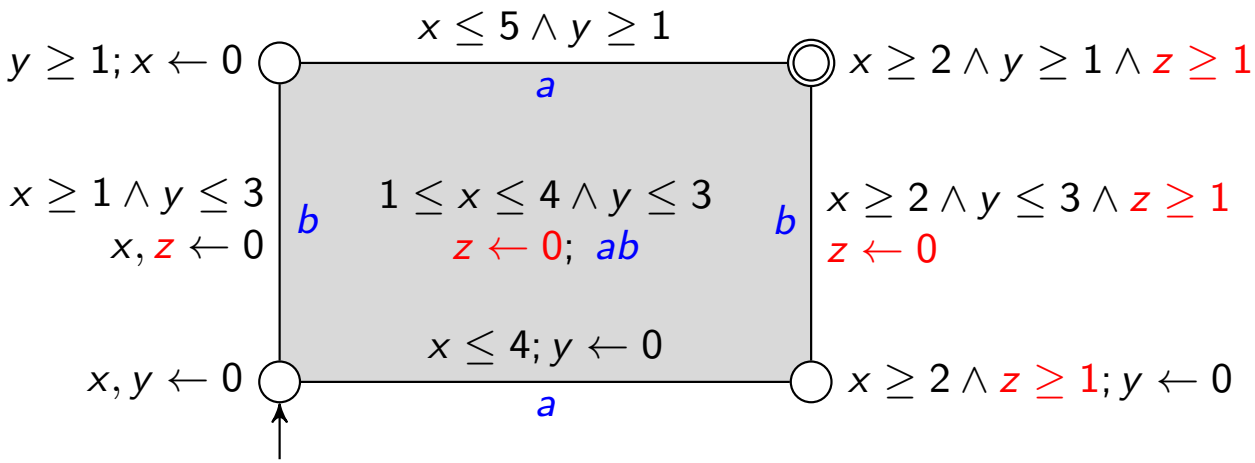


Timing duration of events:

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

Example of HDTA

- Second example: adding timing constraints between events...



Timing duration of events:

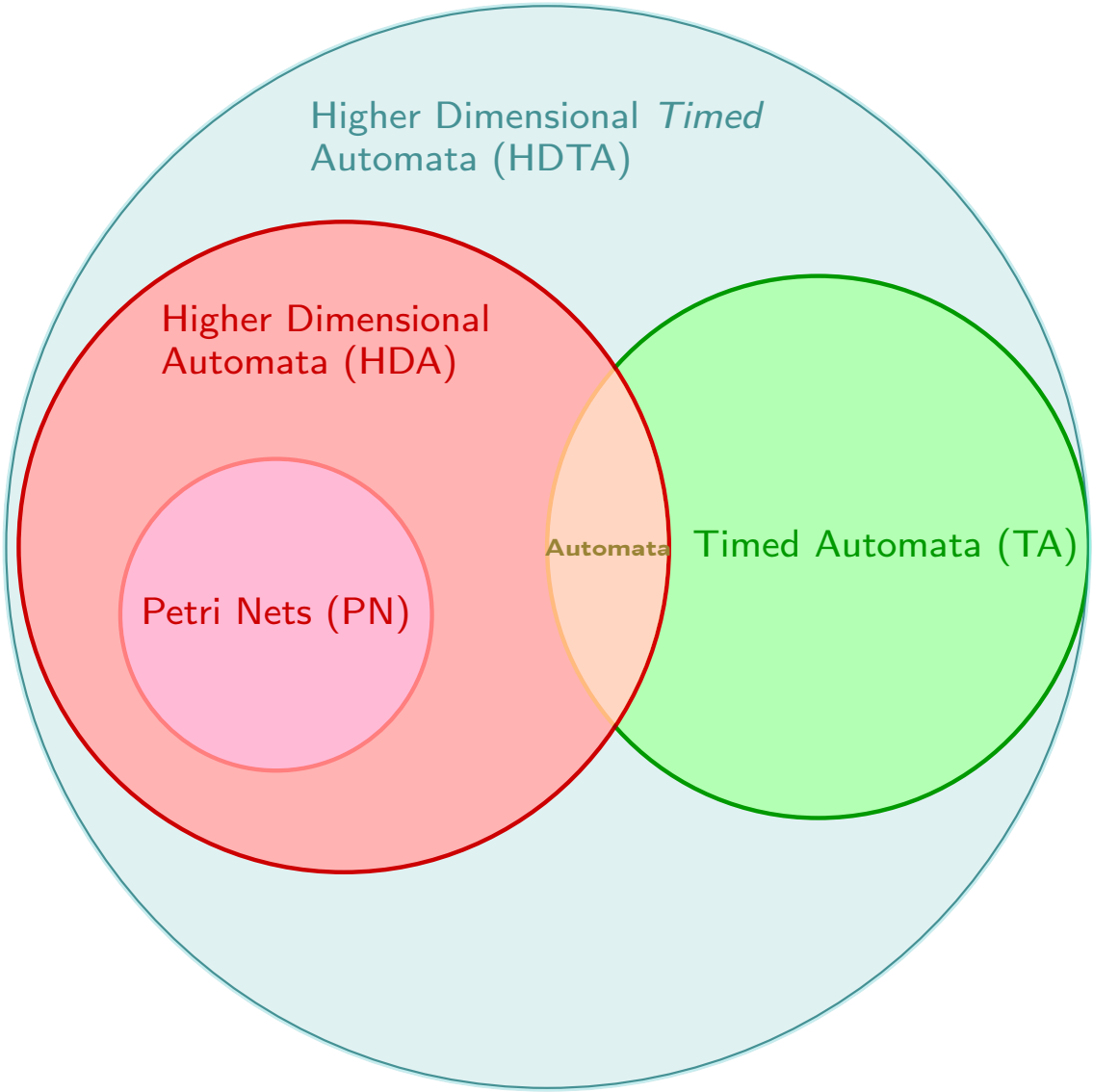
- a: $[2, 4]$ time units
- b: $[1, 3]$ time units

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

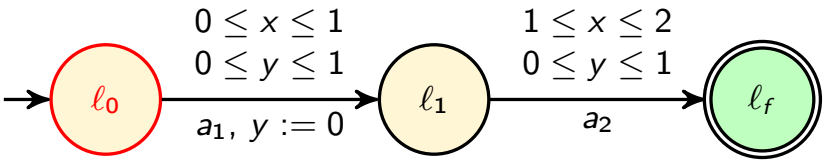
Current work: Extending properties from other models

- Language generated by HDTA
  - **TA**: timed words $((a, 0.7), (b, 1.5), (c, 2.0))...$
  - **HDA**: labeled interval orders
- Region HDTA
  - **TA**: Region Automata
- Timed bisimulation
- Future work: Robustness?

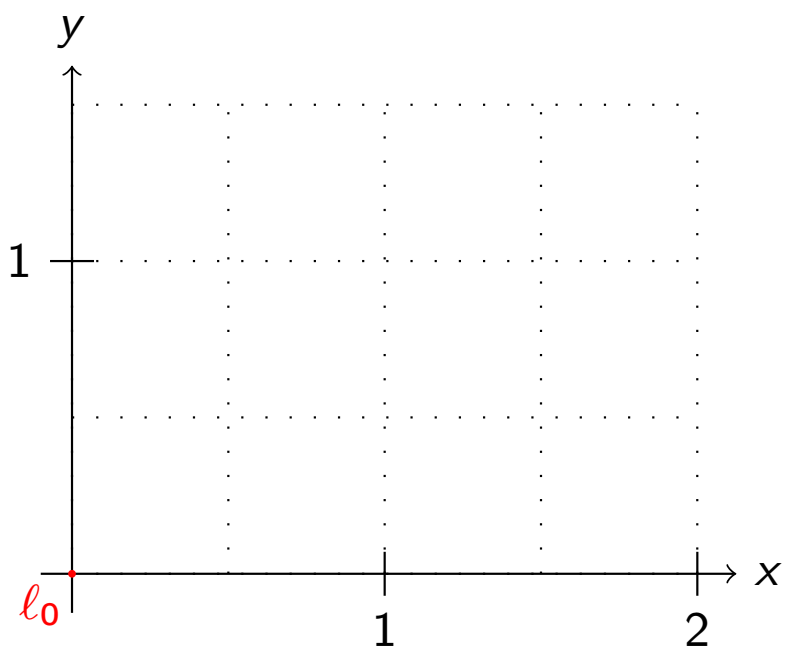


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

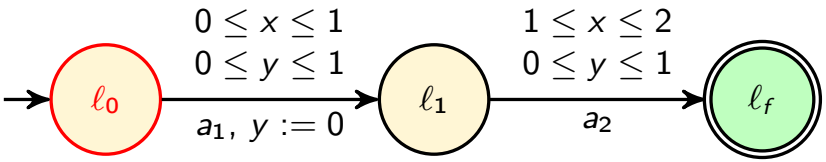


- Run with delay perturbations of at most $\delta = 0.2$

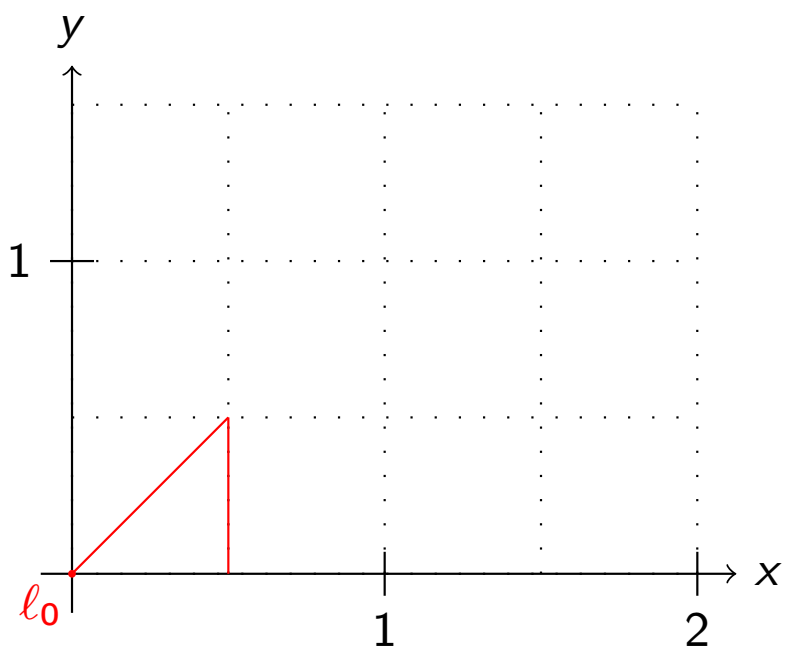


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

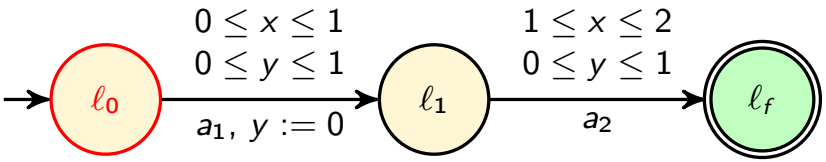


- Run with delay perturbations of at most $\delta = 0.2$

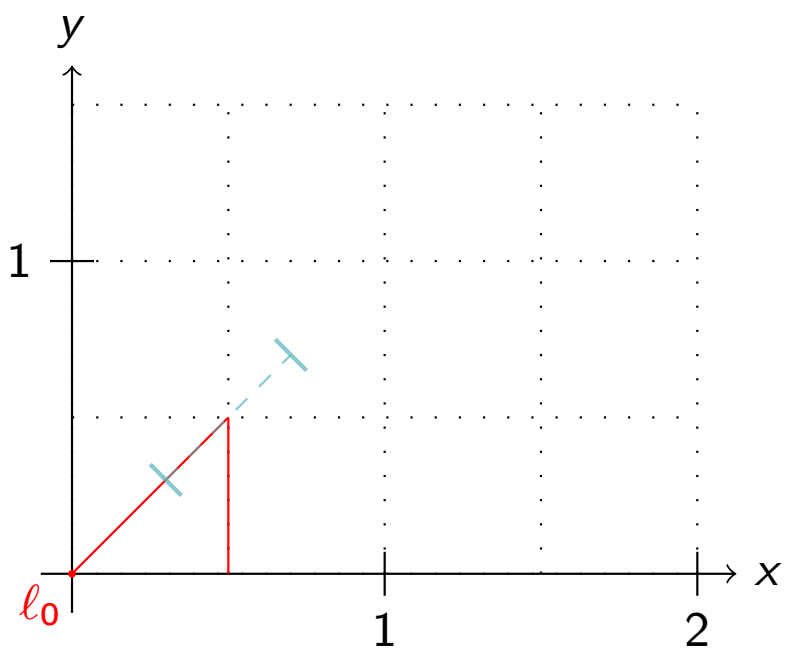


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

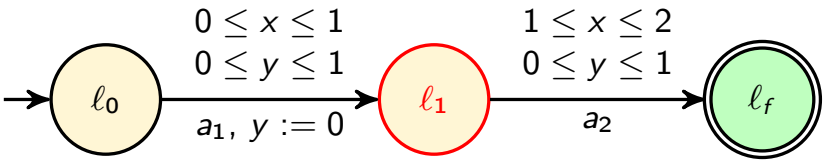


- Run with delay perturbations of at most $\delta = 0.2$

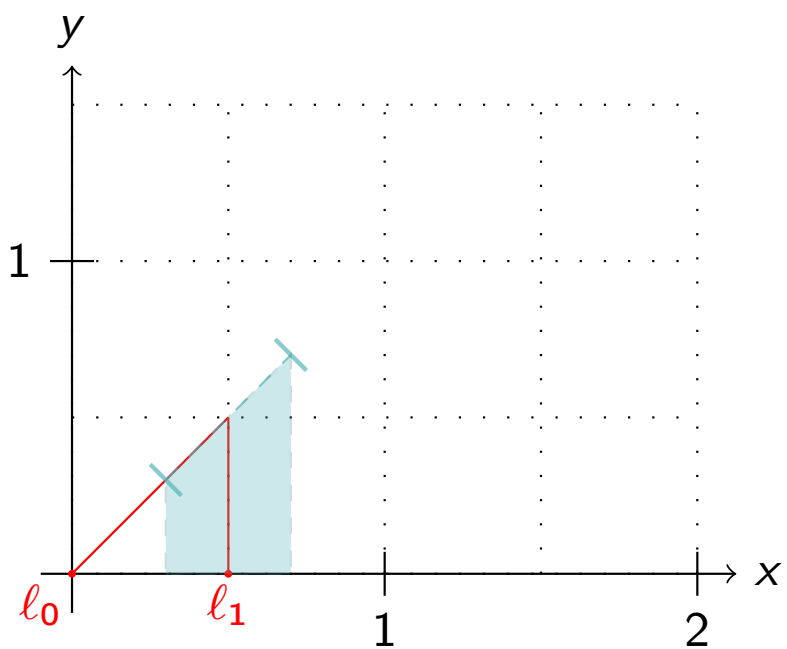


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

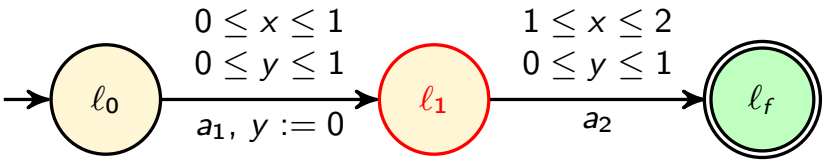


- Run with delay perturbations of at most $\delta = 0.2$

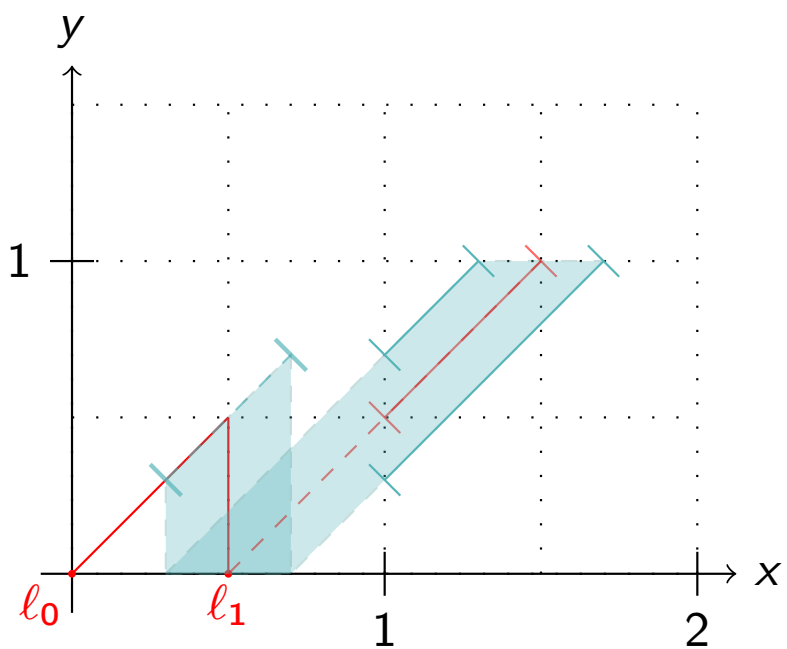


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

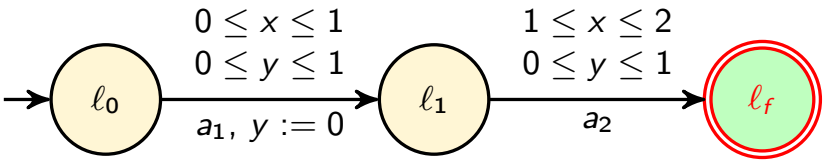


- Run with delay perturbations of at most $\delta = 0.2$

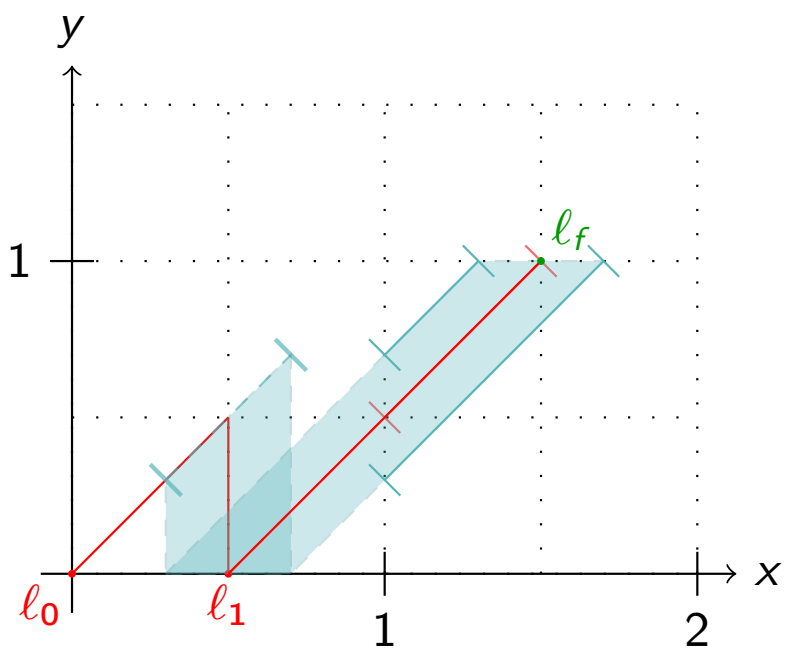


Future work: What about the robustness?

- Timed automaton $\mathcal{A}$:

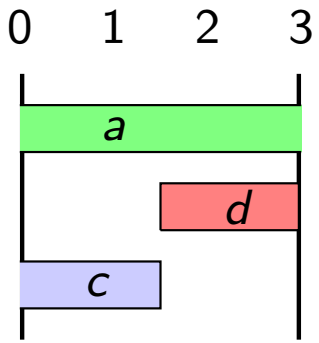


- Run with delay perturbations of at most $\delta = 0.2$



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: