# Discrete and Continuous Models for Concurrent Systems

## 3. Languages of Higher-Dimensional Automata

Uli Fahrenberg

LMF, Université Paris-Saclay, France

ISIMM, April 2026

LMF

Laboratoire Méthodes Formelles

# Higher-Dimensional Automata

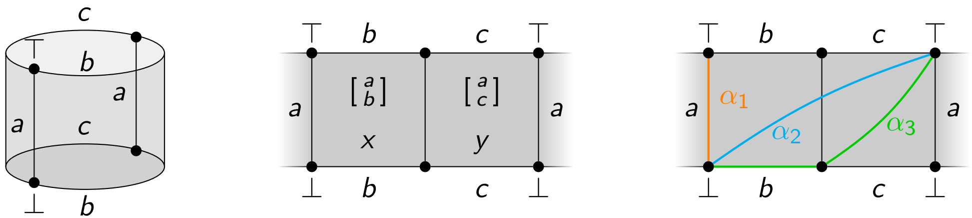
A conclist is a finite, totally ordered, $\Sigma$-labeled set. (a list of labeled events)

A precubical set $X$ consists of:

- A set of cells $X$ (cubes)
- Every cell $x \in X$ has a conclist $\text{ev}(x)$ (list of events active in $x$)
- We write $X[U] = \{x \in X \mid \text{ev}(x) = U\}$ for a conclist $U$ (cells of type $U$)
- For every conclist $U$ and $A \subseteq U$ there are:
  upper face map $\delta_A^1 : X[U] \to X[U \setminus A]$ (terminating events $A$)
  lower face map $\delta_A^0 : X[U] \to X[U \setminus A]$ ("unstarting" events $A$)
- Precube identities: $\delta_A^\mu \delta_B^\nu = \delta_B^\nu \delta_A^\mu$ for $A \cap B = \emptyset$ and $\mu, \nu \in \{0, 1\}$

A higher dimensional automaton (HDA) is a precubical set $X$ with initial cells $\bot \subseteq X$ and accepting cells $\top \subseteq X$ (not necessarily vertices)

# Example

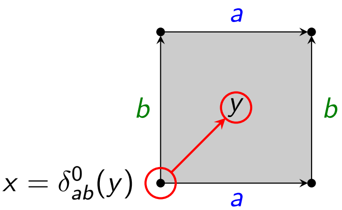



$a \parallel (bc)^*$

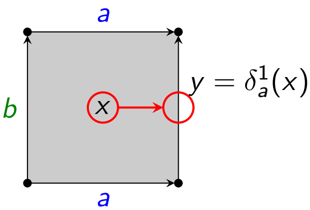
# Computations of HDAs

An HDA computes by starting and terminating events in sequence:

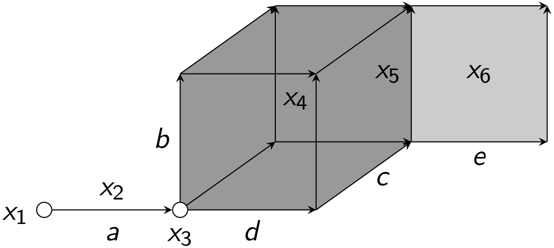
upstep $x \nearrow y$, starting $[\,{}^{a}_{b}\,]$:




downstep $x \searrow y$, terminating $a$:



# Example




$$\left(x_1 \nearrow^{a} x_2 \searrow_{a} x_3 \nearrow^{\{b,c,d\}} x_4 \searrow_{\{c,d\}} x_5 \nearrow^{e} x_6\right)$$

# Precubical Sets As Presheaves

A presheaf over a category $\mathcal{C}$ is a functor $\mathcal{C}^{\text{op}} \to \mathsf{Set}$ (contravariant functor on $\mathcal{C}$)

The precube category $\square$ has conclists as objects.
Morphisms are coface maps $d_{A,B} : U \to V$, where

- $A, B \subseteq V$ are disjoint subsets,
- $U \simeq V \setminus (A \cup B)$ are isomorphic conclists,
- $d_{A,B} : U \to V$ is the unique label-preserving monotonic map with image $V \setminus (A \cup B)$.

Composition of coface maps $d_{A,B} : U \to V$ and $d_{C,D} : V \to W$ is

$$d_{\partial(A) \cup C, \partial(B) \cup D} : U \to W,$$

where $\partial : V \to W \setminus (C \cup D)$ is the unique conclist isomorphism.

- precubical sets: presheaves over $\square$
- HDAs: precubical sets with initial and accepting cells

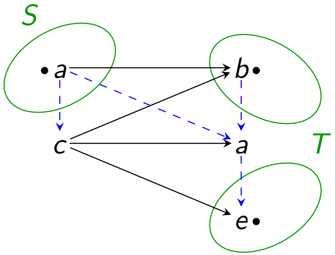
1. Higher-Dimensional Automata
2. Languages of Higher-Dimensional Automata
3. Concurrent Semantics of Petri Nets
4. Exercises

# Pomsets with Interfaces

## Definition

A pomset with interfaces (ipomset): $(P, <, \dashrightarrow, S, T, \lambda)$:

- finite set $P$;
- two partial orders $<$ (precedence order), $\dashrightarrow$ (event order)
  - s.t. $< \cup \dashrightarrow$ is a *total relation*;
- $S, T \subseteq P$ source and target interfaces
  - s.t. $S$ is $<$-minimal and $T$ is $<$-maximal.

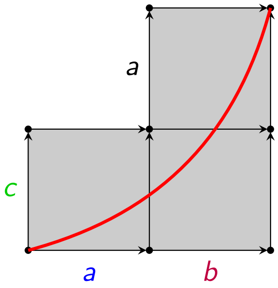


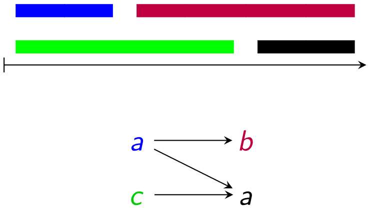
# Interval Orders

## Definition

An ipomset $(P, <_P, \dashrightarrow, S, T, \lambda)$ is interval if $(P, <_P)$ has an interval representation: functions $b, e : P \to \mathbb{R}$ s.t.

- $\forall x \in P : b(x) \leq_\mathbb{R} e(x)$;
- $\forall x, y \in P : e(x) <_\mathbb{R} b(y) \iff x <_P y$



# Fishburn's Theorem

## Definition (restated)

An ipomset $(P, <_P, \dashrightarrow, S, T, \lambda)$ is interval if $(P, <_P)$ is an interval order.

## Definition

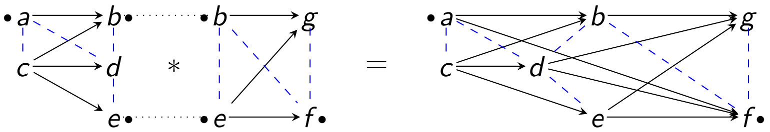
A poset $(P, <_P)$ is an interval order if it has an interval representation: functions $b, e : P \to \mathbb{R}$ s.t.

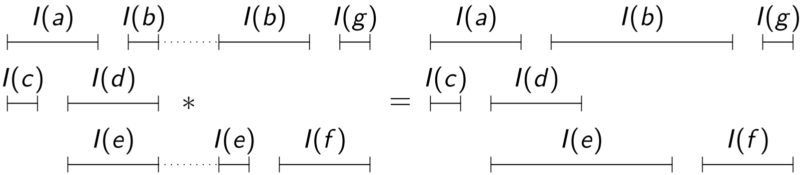
- $\forall x \in P : b(x) \leq_{\mathbb{R}} e(x)$;
- $\forall x, y \in P : e(x) <_{\mathbb{R}} b(y) \iff x <_P y$

## Theorem (Fishburn 1970)

*A poset $(P, <_P)$ is an interval order iff it does not contain $\left[\begin{smallmatrix} \cdot \longrightarrow \cdot \\ \cdot \longrightarrow \cdot \end{smallmatrix}\right]$ as induced subposet. Equivalently: if $a \longrightarrow b$ and $c \longrightarrow d$, then also $a \longrightarrow d$ or $c \longrightarrow b$.*

# Gluing Composition




- Gluing $P * Q$: $P$ before $Q$, except for interfaces (which are identified)
- (also have parallel composition $P \parallel Q$: disjoint union)

# Gluing Composition, Formally

## Definition

Let $P$ and $Q$ be ipomsets such that

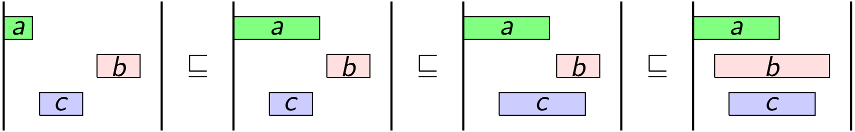
- $T_P = S_Q$ and $(P \setminus T_P) \cap (Q \setminus S_Q) = \emptyset$,
- $x \dashrightarrow_P y$ iff $x \dashrightarrow_Q y$ for all $x, y \in T_P = S_Q$, and
- the restrictions $\lambda_P \restriction T_P = \lambda_Q \restriction S_Q$.

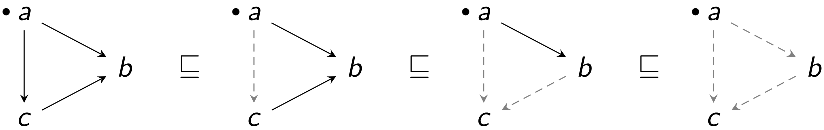
The gluing of $P$ with $Q$ is $P * Q = (P \cup Q, <, \dashrightarrow, S_P, T_Q, \lambda)$, where

- $x < y$ if $x <_P y$, $x <_Q y$, or $x \in P - T_P$ and $y \in Q - S_Q$,
- $\dashrightarrow$ is the transitive closure of $\dashrightarrow_P \cup \dashrightarrow_Q$,
- $\lambda(x) = \lambda_P(x)$ if $x \in P$, and $\lambda(x) = \lambda_Q(x)$ if $x \in Q$.

- defined if the targets of $P$ are equal to the sources of $Q$ as conclists
- events in $P$ precede events in $Q$, except when in the interfaces

# Subsumption




$P$ refines $Q$ / $Q$ subsumes $P$ / $P \sqsubseteq Q$ iff

- $P$ and $Q$ have same interfaces
- $P$ has more $<$ than $Q$
- $Q$ has more $\dashrightarrow$ than $P$

# Languages of HDAs

## Definition

The language of an HDA $X$ is the set of event ipomsets of all accepting paths:

$$L(X) = \{\mathrm{ev}(\pi) \mid \pi \in \mathrm{Paths}(X), \mathrm{src}(\pi) \in \bot_X, \mathrm{tgt}(\pi) \in \top_X\}$$

- $L(X)$ contains only interval ipomsets,
- is closed under subsumption,
- and has finite width

## Definition

A language $L \subseteq \mathrm{iiPoms}$ is regular if there is an HDA $X$ with $L = L(X)$.

# Theorems

**Definition (Rational Languages over $\Sigma$)**

- Generated by $\emptyset$, $\{\epsilon\}$, and all $\{[a]\}$, $\{[\bullet\, a]\}$, $\{[a\, \bullet]\}$, $\{[\bullet\, a\, \bullet]\}$ for $a \in \Sigma$
- under operations $\cup$, $*$, $\parallel$ and (Kleene plus) $^+$
- (these need to take subsumption closure into account)

**Definition (Monadic Second-Order Logics over Ipomsets)**

$$\psi ::= a(x) \mid \mathsf{s}(x) \mid \mathsf{t}(x) \mid x < y \mid x \dashrightarrow y \mid x \in X \mid$$
$$\exists x.\, \psi \mid \forall x.\, \psi \mid \exists X.\, \psi \mid \forall X.\, \psi \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \neg \psi$$

**Theorem (à la Kleene)**: regular $\iff$ rational

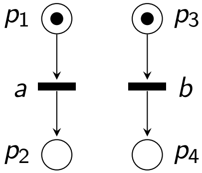
**Theorem (à la Myhill-Nerode)**: regular $\iff$ finite prefix quotient

**Theorem (à la Büchi-Elgot-Trakhtenbrot)**:

regular $\iff$ MSO-definable, of finite width, and subsumption-closed

# Concurrent Semantics of Petri Nets

Petri net $(S, T, F)$: places $S$; transitions $T$;
weighted flows $F : S \times T \cup T \times S \to \mathbb{N}$




Interleaved semantics $(V, E)$: $V = \mathbb{N}^S$; $E \subseteq V \times T \times V$

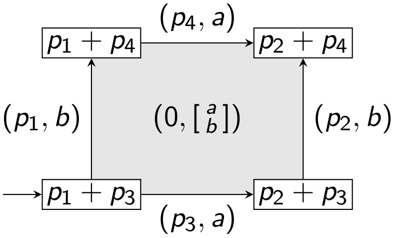
- $E = \{(m, t, m') \mid {}^\bullet t \le m, m' = m - {}^\bullet t + t^\bullet\}$

Concurrent semantics as HDA:
$\square = \square(T)$, $X = \mathbb{N}^S \times \square$, $\text{ev}(m, \tau) = \tau$

- for $x = (m, \tau) \in X[\tau]$ with $\tau = (t_1, \dots, t_n)$:

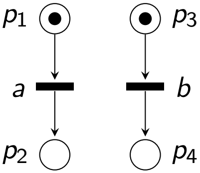
$$\delta^0_{t_i}(x) = (m + {}^\bullet t_i, (t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n))$$
$$\delta^1_{t_i}(x) = (m + t_i^\bullet, (t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n))$$

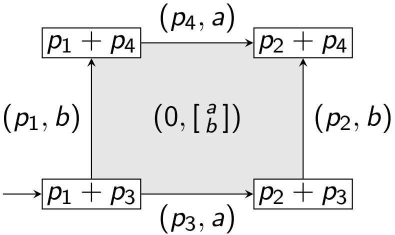
- initial marking $\implies$ initial cell; take reachable part
- (no accepting cells)



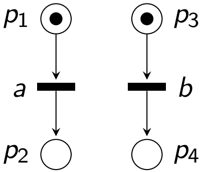
# Event Order

- trouble with symmetry:
  have a cell $(0, [\begin{smallmatrix} a \\ b \end{smallmatrix}])$, but also $(0, [\begin{smallmatrix} b \\ a \end{smallmatrix}])$ (not shown)
- solution: fix an arbitrary order $\preccurlyeq$ on $T$
- and use $\square = \left\{ \begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix} \,\middle|\, \forall i = 1, \dots, n-1 : t_i \preccurlyeq t_{i+1} \right\}$
  instead of $\square(T)$
- order $\preccurlyeq$ may be chosen (and re-chosen) at will
- here: lexicographic $a \prec b \prec \dots$

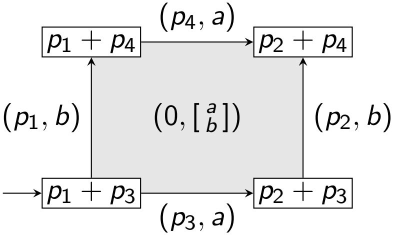


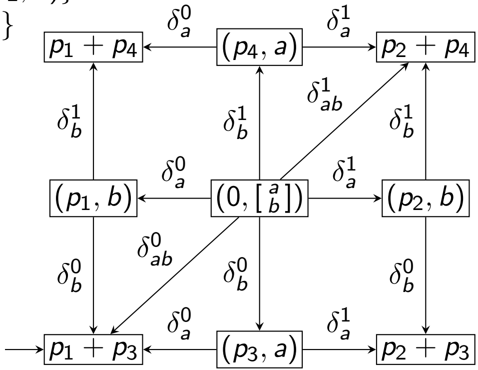



# Example, Complete




$$X[\emptyset] = \{p_1 + p_3, p_2 + p_3, p_1 + p_4, p_2 + p_4\}$$
$$X[a] = \{(p_3, a), (p_4, a)\}$$
$$X[b] = \{(p_1, b), (p_2, b)\}$$
$$X[[\begin{smallmatrix} a \\ b \end{smallmatrix}]] = \{(0, [\begin{smallmatrix} a \\ b \end{smallmatrix}])\}$$

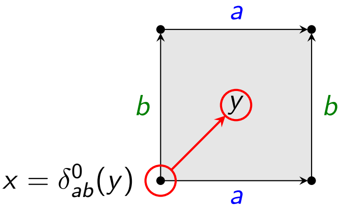


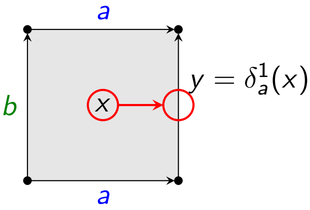
# Computations of HDAs

An HDA computes by starting and terminating events in sequence:

upstep $x \nearrow y$, starting $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right]$:




downstep $x \searrow y$, terminating $a$:




Idea: Use this to define an automata-like operational semantics for HDAs

- an ST-automaton (def. next slide) has
  - transitions which start and terminate events
  - states which remember which events are currently running

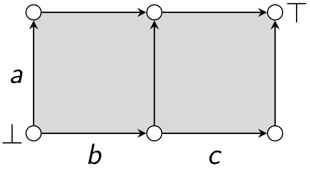
# ST-Automata

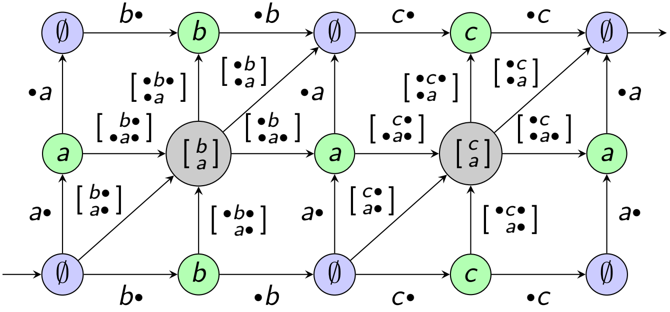
- a starter $(A, U)$: conclist $U$, subset $A \subseteq U$
- a terminator $(U, B)$: conclist $U$, subset $B \subseteq U$
- starting $A$; terminating $B$: written ${}_A{\uparrow}U$ resp. $U{\downarrow}_B$
- Let ST denote the (infinite) set of starters and terminators

An ST-automaton $(Q, \bot, \top, E, \lambda)$:

- $Q$ set of states; $\bot, \top \subseteq Q$ initial resp. accepting states
- $E \subseteq Q \times \text{ST} \times Q$ transitions
- $\lambda : Q \to \square$ state labeling, such that for all $(p, x, q) \in E$:
    - if $x = {}_A{\uparrow}U$, then $\lambda(p) = U \setminus A$ and $\lambda(q) = U$;
    - if $x = U{\downarrow}_B$, then $\lambda(p) = U$ and $\lambda(q) = U \setminus B$.

# Translation



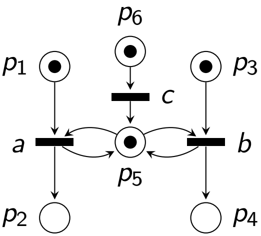



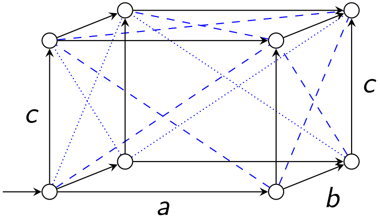

from HDA $(X, \bot, \top)$ to ST-automaton $(Q, \bot, \top, E, \lambda)$:

- $Q = X$, $\lambda = \text{ev}$, $E = \{\delta^0_A(x) \xrightarrow{A \uparrow \text{ev}(x)} x \mid A \subseteq \text{ev}(x)\} \cup \{x \xrightarrow{\text{ev}(x) \downarrow_A} \delta^1_A(x) \mid A \subseteq \text{ev}(x)\}$

from ST-automata to HDAs: complicated; we've lost geometric information

# One Last Example




- initially, $p_5$ is a mutex place: it disables concurrency of $a$ and $b$
- after $c$ fires, $p_5$ holds two tokens, so $a$ and $b$ become concurrent

- semantically, a hollow cube without bottom face
- the five faces: front: $(p_3, [\begin{smallmatrix} a \\ c \end{smallmatrix}])$, back: $(p_4, [\begin{smallmatrix} a \\ c \end{smallmatrix}])$
  left: $(p_1, [\begin{smallmatrix} b \\ c \end{smallmatrix}])$, right: $(p_2, [\begin{smallmatrix} b \\ c \end{smallmatrix}])$
  top: $(0, [\begin{smallmatrix} a \\ b \end{smallmatrix}])$

# Exercises

**Exercise 2**: Implement the “Bakery” Petri net from the lecture in `wolfgang` and simulate its steps. Save it as `bakery.pnml` and have a look at that file in a text editor. Run the net until you have explored all reachable markings. Draw the reachability graph of the net and its HDA semantics. Use `pn2hda` to confirm your results.

**Exercise 3**: Implement the last example of the lecture in `wolfgang` and translate it to an ST-automaton. Add a new place and a transition *d* which *disables* the concurrency of *a* and *b*. Repeat the translation to an ST-automaton. What does the corresponding HDA look like?