Cubical Sets and Petri Nets: an Adjunction

Samuel Mimram

MeASI – CEA Saclay

11 January 2010
Concurrent computations

Programs tend to be concurrent

▶ processes, multi-core processors, networks, etc.

This raises new problems

▶ concurrent accesses to resources
▶ deadlocks
▶ etc.

A geometrical approach

▶ in order to regulate and verify concurrent programs, we should study their geometry
An adjunction

\[
\begin{align*}
\text{Petri nets} & \quad \leftrightarrow \quad \text{Cubical Sets} \\
a \text{ very well-known and studied model} & \quad \quad a \text{ geometrical model}
\end{align*}
\]
An adjunction

Petri nets $\leftrightarrow$ Cubical Sets

a very well-known and studied model

$\text{pn}(C) \rightarrow N$  

$C \rightarrow \text{cs}(N)$
Petri nets

An abstract representation of processes focused on resources:

Petri net: a graph whose vertices are either

- places (containing tokens)
- events (or transitions)
Petri nets

An abstract representation of processes focused on *resources*:

Petri net: a graph whose vertices are either

- *places* (containing *tokens*)
- *events* (or *transitions*)
Petri nets

An abstract representation of processes focused on *resources*:

Petri net: a graph whose vertices are either

- places (containing *tokens*)
- events (or *transitions*)
Typical situations

Petri nets can express **causality**:

Possible runs:
Typical situations

Petri nets can express **causality**:

![Petri net diagram]

Possible runs:  $a$
Typical situations

Petri nets can express **causality**:

Possible runs: $ab$
Typical situations

Petri nets can express **conflict:**

![Petri net diagram](image)

Possible runs:
Typical situations

Petri nets can express **conflict**:

Possible runs: $a$
Typical situations

Petri nets can express **conflict:**

Possible runs: $a$ or $b$
Typical situations

Petri nets can express independence:

Possible runs: $ab$ or $ba$ or $aa$ or $bb$
Typical situations

Petri nets can express **independence**:

Possible runs: $ab$ or $ba$ or $aa$ or $bb$
Typical situations

Petri nets can express **independence**:

Possible runs: $ab$ or $ba$ or $aa$ or $bb$
Typical situations

Petri nets can express **loops**:

Possible runs: aaaaaaaa…
Taking multiplicities in account

More generally we consider nets in which a transition might need or produce multiple tokens of the same place:
Taking multiplicities in account

More generally we consider nets in which a transition might need or produce multiple tokens of the same place:
Petri nets, formally

A Petri net \((P, M_0, E, \text{pre}, \text{post})\) consists of

- a set \(P\) of places
- an initial marking \(M_0 \in \mathbb{N}^P\)
- a set \(E\) of events (or transitions)
- a precondition function \(\text{pre} : E \rightarrow \mathbb{N}^P\)
- a postcondition function \(\text{post} : E \rightarrow \mathbb{N}^P\)
Transitions

States

The “state” of a Petri net is a marking \( M \in \mathbb{N}^P \).
Transitions

States
The “state” of a Petri net is a marking \( M \in \mathbb{N}^P \).

Transitions
Given an event \( e \) and two markings \( M_1 \) and \( M_2 \), there is a transition

\[
M_1 \xrightarrow{e} M_2
\]

when there exists a marking \( M \) such that

\[
M_1 = M + \text{pre}(e) \quad \text{and} \quad M_2 = M + \text{post}(e)
\]
Transitions

States
The “state” of a Petri net is a marking $M \in \mathbb{N}^P$.

Transitions
Given an event $e$ and two markings $M_1$ and $M_2$, there is a transition

\[ M_1 \xrightarrow{e} M_2 \]

when there exists a marking $M$ such that

\[ M_1 = M + \text{pre}(e) \quad \text{and} \quad M_2 = M + \text{post}(e) \]

Runs
A run

\[ M_0 \xrightarrow{e_1} M_1 \xrightarrow{e_2} M_2 \ldots M_{n-1} \xrightarrow{e_n} M_n \]

is a finite sequence of transitions from the initial marking $M_0$. 
Semantics of Petri nets

To every Petri net $N$ we want to associate a semantics $\mathcal{J}[N]$ which describes precisely the dynamic behavior of the net.
Semantics of Petri nets

To every Petri net $N$ we want to associate a semantics $\llbracket N \rrbracket$ which describes precisely the dynamic behavior of the net.

Idea 1

$\llbracket N \rrbracket$ should be the set of words of events labeling a run of $N$.

\[
\llbracket N \rrbracket = \{ \varepsilon, a, b, ab, ba, aa, bb \}
\]
Semantics of Petri nets

To every Petri net $N$ we want to associate a semantics $⟦N⟧$ which describes precisely the dynamic behavior of the net.

Idea 1

$⟦N⟧$ should be the set of words of events labeling a run of $N$.

\[
⟦N⟧ = \{ \varepsilon, a, b, ab, ba, aa, bb \}
\]

We loose too much structure by forgetting about states!
Semantics of Petri nets

Idea 2

$\llbracket N \rrbracket$ should be a graph whose

- vertices are reachable markings
- edges are transitions, labelled by events
Semantics of Petri nets

Idea 2

$[N]$ should be a graph whose

- vertices are reachable markings
- edges are transitions, labelled by events

We loose structure by forgetting about concurrency!
Semantics of Petri nets

Idea 2
\[\llbracket N \rrbracket\] should be a graph whose
- vertices are reachable markings
- edges are transitions, labelled by events

We lose structure by forgetting about concurrency!
Semantics of Petri nets

Idea 2

$\llbracket N \rrbracket$ should be a graph whose

- vertices are reachable markings
- edges are transitions, labelled by events

We loose structure by forgetting about concurrency!
Taking concurrency in account

\[
(x := 3 \mid x := 4) \text{ vs. } (x := 3 \mid y := 4)
\]
Taking concurrency in account

\[ x := 3 \mid x := 4 \] vs. \[ x := 3 \mid y := 4 \]
Taking concurrency in account

\[(x := 3 \mid x := 4) \text{ vs. } (x := 3 \mid y := 4)\]
Taking concurrency in account

- We can now distinguish between an “empty square” and a “filled square”.

\[
\begin{array}{ccc}
  a & b & c \\
  \text{empty cube} & \text{vs.} & \text{filled cube}
\end{array}
\]
Taking concurrency in account

- We can now distinguish between an “empty square” and a “filled square”.
- We should also go on in higher dimensions:

- empty cube vs. filled cube
So, to every Petri net we associate a **Cubical Set**
which is like a simplicial set with squares instead of triangles
So, to every Petri net we associate a **Cubical Set**
which is like a simplicial set with squares instead of triangles
whose arrows are labeled by events
From Petri nets to Cubical Sets

So, to every Petri net we associate a Cubical Set which is like a simplicial set with squares instead of triangles whose arrows are labeled by events with an initial position.
Recall that a (augmented) simplicial set is a functor $S : \Delta^{op} \to \text{Set}$.  

$\Delta$ is the category of finite ordinals and increasing functions.

Geometric intuition:

\[
\begin{array}{cccccc}
0 & 1 & 2 & 3 & \ldots \\
. & \cdot & \cdot & \cdot & \ldots \\
\end{array}
\]

\[
\begin{array}{cccc}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\end{array}
\]
Simplicial sets

- Recall that a (augmented) simplicial set is a functor $S : \Delta^{\text{op}} \to \text{Set}$.
- $\Delta$ is the category of finite ordinals and increasing functions.
- Geometric intuition:

$$
\begin{array}{cccccc}
0 & 1 & 2 & 3 & \ldots \\
\cdot & \cdot & \cdot & \cdot & \ldots
\end{array}
$$

- The arrows of $\Delta$ are generated by

$$
\delta^n_i : n \to n + 1 \quad \text{and} \quad \sigma^n_{i+1} : n + 2 \to n + 1
$$

with $n \in \mathbb{N}$ and $0 \leq i \leq n$, subject to equations

$$
\delta^n_{i+1} \delta^n_j = \delta^n_{j+1} \delta^n_i \quad \ldots
$$
Cubical sets

- A **cubical set** is a functor $C : \Box^{\text{op}} \to \text{Set}$.
- Geometric intuition:

  
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>...</th>
</tr>
</thead>
</table>

  ![Diagram of cubical set]

  ![Diagram of higher-dimensional cube]

  ![Diagram of even higher-dimensional cube]

  ...
Cubical sets

- A **cubical set** is a functor $C : \square^{\text{op}} \to \text{Set}$.
- Geometric intuition:

  $0 \quad 1 \quad 2 \quad 3 \quad \ldots$

  $\ldots$

  $\ldots$

  $\ldots$

- The category $\square$ is generated by

  $\varepsilon_i^- : n \to n + 1 \quad \varepsilon_i^+ : n \to n + 1 \quad \eta_i : n + 1 \to n$
Cubical sets

- **A cubical set** is a functor $C : \square^{\text{op}} \to \text{Set}$.

- Geometric intuition:

  0 1 2 3 ... 
  \[
  \begin{array}{cccc}
  & & & \\
  & & & \\
  & & & \\
  & & & \\
  & & & \\
  & & & \\
  & & & \\
  \end{array}
  \]

- The category $\square$ is generated by

  - $\varepsilon_i^- : n \to n + 1$
  - $\varepsilon_i^+ : n \to n + 1$
  - $\eta_i : n + 1 \to n$

  source target degeneracy
The cubical category

The category □ is the category generated by

\[ \varepsilon_i^- : n \to n + 1 \quad \varepsilon_i^+ : n \to n + 1 \quad \eta_i : n + 1 \to n \]

subject to the equations

\[ \varepsilon_j^\alpha \varepsilon_i^\beta = \varepsilon_i^\beta \varepsilon_j^\alpha \quad \text{with } i < j, \alpha, \beta \in \{-, +\} \]

\[ \eta_j \eta_i = \eta_{i-1} \eta_j \quad \text{with } i > j \]

\[ \eta_j \varepsilon_i^\alpha = \begin{cases} \varepsilon_i^\alpha \eta_{j-1} & \text{if } i < j \\ \text{id} & \text{if } i = j \\ \varepsilon_i^\alpha \eta_j & \text{if } i > j \end{cases} \quad \text{with } \alpha \in \{-, +\} \]
Labeled cubical sets

A labeled cubical set on an alphabet $\Sigma$ is

- a cubical set $C : \square^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to !\Sigma$
Labeled cubical sets

A labeled cubical set on an alphabet $\Sigma$ is

- a cubical set $C : \square^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to !\Sigma$

What should $!\Sigma$ look like if $\Sigma = \{ a, b \}$?

![Diagram of labeled cubical sets]

In the diagram, $z \sim y_1 \sim y_2 \sim x$, with $b : y_1 \to y_2$ and $a : z \to x$. The arrows represent the labels for the morphisms.
Labeled cubical sets

A labeled cubical set on an alphabet $\Sigma$ is

- a cubical set $C : \square^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to ! \Sigma$

What should $! \Sigma$ look like if $\Sigma = \{ a, b \}$?

![](diagram.png)

$! \Sigma(0)$

$\{ \ast \}$
Labeled cubical sets

A labeled cubical set on an alphabet $\Sigma$ is

- a cubical set $C : \square^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to ! \Sigma$

What should $! \Sigma$ look like if $\Sigma = \{a, b\}$?

![Diagram of labeled cubical set]

$! \Sigma(0)$ $! \Sigma(1)$

$\{\ast\}$ $\{\ast, a, b\}$
Labeled cubical sets

A **labeled cubical set** on an alphabet $\Sigma$ is

- a cubical set $C : \square^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to ! \Sigma$

What should $! \Sigma$ look like if $\Sigma = \{ \ a, b \ \}$?

![Diagram of labeled cubical set]

$\begin{align*}
! \Sigma(0) & \quad \quad \quad \quad \quad \quad ! \Sigma(1) & \quad \quad \quad \quad \quad \quad ! \Sigma(2) \\
\{ \ * \ \} & \quad \quad \quad \quad \quad \quad \{ \ *, a, b \ \} & \quad \quad \quad \quad \quad \quad \{ \ *, a, b, ab, ba \ \}
\end{align*}$
Labeled cubical sets

A labeled cubical set on an alphabet $\Sigma$ is

- a cubical set $C : \Box^{\text{op}} \to \text{Set}$
- together with a labeling morphism $\lambda : C \to \! \Sigma$

What should $\! \Sigma$ look like if $\Sigma = \{a, b\}$?

\begin{align*}
!\Sigma(0) & : \{\ast\} \\
!\Sigma(1) & : \{\ast, a, b\} \\
!\Sigma(2) & : \{\ast, a, b, ab, ba\} \\
\ldots & \\
\end{align*}
Technically

- Defining $\Sigma$ involves
  - defining all the $\Sigma(n)$
  - defining the generators for maps
  - verifying the equations.

- We have two possible labels for the preceding square.
A monoidal definition of cubical sets

The cubical category □ is a monoidal category:

- We have a tensor product ⊗
A monoidal definition of cubical sets

The cubical category \( \square \) is a **monoidal category**:

- We have a tensor product \( \otimes \)

\[
\begin{array}{ccc}
m_1 & \xrightarrow{f} & n_1 \\
\end{array}
\]
A monoidal definition of cubical sets

The cubical category □ is a **monoidal category**:  
- We have a tensor product \( \otimes \)

\[
\begin{array}{c}
m_2 \xrightarrow{g} n_2 \\
m_1 \xrightarrow{f} n_1
\end{array}
\]
A monoidal definition of cubical sets

The cubical category $\Box$ is a **monoidal category**:

- We have a tensor product $\otimes$

$$m_1 + m_2 \xrightarrow{f \otimes g} n_1 + n_2$$
A monoidal definition of cubical sets

The cubical category \( \square \) is a **monoidal category**:

- We have a tensor product \( \otimes \)

\[
m_1 + m_2 \xrightarrow{f \otimes g} n_1 + n_2
\]

- We also have a unit object: 0
A monoidal definition of cubical sets

The category \( \square \) is the category generated by

\[
\varepsilon_i^- : n \to n + 1 \quad \varepsilon_i^+ : n \to n + 1 \quad \eta_i : n + 1 \to n
\]

subject to the equations

\[
\varepsilon_j^\alpha \varepsilon_i^\beta = \varepsilon_i^\beta \varepsilon_j^{\alpha-1} \quad \text{with } i < j, \alpha, \beta \in \{-, +\}
\]

\[
\eta_j \eta_i = \eta_{i-1} \eta_j \quad \text{with } i > j
\]

\[
\eta_j \varepsilon_i^\alpha = \begin{cases} 
\varepsilon_i^\alpha \eta_{j-1} & \text{if } i < j \\
\text{id} & \text{if } i = j \\
\varepsilon_i^\alpha \eta_j & \text{if } i > j
\end{cases} \quad \text{with } \alpha \in \{-, +\}
\]
A monoidal definition of cubical sets

The category $\Box$ is the monoidal category generated by

$$
\varepsilon^- : 0 \to 1 \quad \varepsilon^+ : 0 \to 1 \quad \eta : 1 \to 0
$$

subject to the equations

$$
\eta \circ \varepsilon^- = \text{id}_0 = \eta \circ \varepsilon^+
$$
A monoidal definition of cubical sets

- A **monoidal functor** between monoidal categories is a functor which preserves tensor product.
A monoidal definition of cubical sets

- **A monoidal functor** between monoidal categories is a functor which preserves tensor product.

- In particular, functors from □ are often monoidal: consider the functor $F : □ \to \textbf{Top}$ defined by

  \[
  F(2 + 1) = F(2) \times F(1) = 2 \times 3 \quad \ldots
  \]
A monoidal definition of cubical sets

- A **monoidal functor** between monoidal categories is a functor which preserves tensor product.
- In particular, functors from □ are often monoidal: consider the functor $F : □ \to \textbf{Top}$ defined by

$$
\begin{array}{cccccc}
0 & 1 & 2 & 3 & \ldots \\
\end{array}
$$

We have

$$F(2 + 1) = F(2) \times F(1)$$
Cubical objects

A cubical set is a functor

\[ C : \square^{\text{op}} \rightarrow \textbf{Set} \]

When this functor is monoidal, this is exactly the same as a cubical object.
Cubical objects

A cubical set is a functor

\[ C : \square^{\text{op}} \to \text{Set} \]

When this functor is monoidal, this is exactly the same as a **cubical object**.

Cubical objects

A *cubical object* \((A, \varepsilon^-, \varepsilon^+, \eta)\) in a monoidal category \(C\) is an object \(A\) of \(C\) together with morphisms

\[ \varepsilon^- : A \to I \quad \varepsilon^+ : A \to I \quad \eta : I \to A \]

such that

\[ \varepsilon^- \circ \eta = \text{id}_I = \varepsilon^+ \circ \eta \]
Cubical objects

A cubical object \((A, \varepsilon^-, \varepsilon^+, \eta)\) in a monoidal category \(C\) is an object \(A\) of \(C\) together with morphisms

\[
\varepsilon^- : A \to I \quad \varepsilon^+ : A \to I \quad \eta : I \to A
\]

such that

\[
\varepsilon^- \circ \eta = \text{id}_I = \varepsilon^+ \circ \eta
\]
Cubical objects

A cubical object $(A, \varepsilon^-, \varepsilon^+, \eta)$ in a monoidal category $C$ is an object $A$ of $C$ together with morphisms

$$\varepsilon^- : A \to I \quad \varepsilon^+ : A \to I \quad \eta : I \to A$$

such that

$$\varepsilon^- \circ \eta = \text{id}_I = \varepsilon^+ \circ \eta$$

The labeling cubical set

$(\text{Set}, \times, 1)$ is a monoidal category.

The object $1 = \{\ast\}$ is terminal in $\text{Set}$. Take

- $\eta : 1 \to (1 + \Sigma)$ the injection
- $\varepsilon^-, \varepsilon^+ : (1 + \Sigma) \to 1$ the terminal arrow

This defines the cubical set $! \Sigma$. 
The labeling cubical set

We can give an explicit description of $! \Sigma$:

- the elements of $! \Sigma(n)$ are words $a_1 \cdot a_2 \cdots a_n$
  where $a_i \in \Sigma \cup \{\ast\}$
- $\varepsilon^-_i, \varepsilon^+_i$ remove the $i$-th letter
- $\eta_i$ inserts a $\ast$ at the $i$-th position
Symmetric cubical sets

Should we label the tile by $ab$ or by $ba$?

In fact, we should keep both possibilities and remember that they are "almost the same":

Set is a symmetric monoidal category $A \times B \sim B \times A$.
Symmetric cubical sets

Should we label the tile by \( ab \) or by \( ba \)?

In fact, we should keep both possibilities and remember that they are “almost the same”: \( \textbf{Set} \) is a symmetric monoidal category

\[
A \times B \cong B \times A
\]
Symmetric cubical sets

A **symmetric cubical set** is a symmetric monoidal functor

\[ C : \square_S^{\text{op}} \rightarrow \text{Set} \]

where \( \square_S \) is the free symmetric monoidal category on \( \square \).
Symmetric cubical sets

The category $\square_S$ is the symmetric monoidal category generated by

$$\varepsilon^- : 0 \rightarrow 1 \quad \varepsilon^+ : 0 \rightarrow 1 \quad \eta : 1 \rightarrow 0$$

subject to the equations

$$\eta \circ \varepsilon^- = \text{id}_0 = \eta \circ \varepsilon^+$$
Symmetric cubical sets

The category $\square_S$ is the monoidal category generated by

$$
\varepsilon^- : 0 \to 1 \quad \varepsilon^+ : 0 \to 1 \quad \eta : 1 \to 0 \quad \gamma : 2 \to 2
$$

subject to the equations

$$
\eta \circ \varepsilon^- = \text{id}_0 = \eta \circ \varepsilon^+
$$

$$(\gamma \otimes 1) \circ (1 \otimes \gamma) \circ (\gamma \otimes 1) = (1 \otimes \gamma) \circ (\gamma \otimes 1) \circ (1 \otimes \gamma)$$

$$
\gamma \circ \gamma = 2
$$

$$(\eta \otimes 1) \circ \gamma = 1 \otimes \eta$$

$$(1 \otimes \eta) \circ \gamma = \eta \otimes 1$$

$$\ldots$$
Higher-dimensional automata

To every Petri net $N$ we associate a **higher-dimensional automaton** $\text{hda}(N)$ consisting of

- a symmetric cubical set $C$
- labeled by events of the net $\lambda : C \to ! E$
- with an initial position $M_0$
Morphisms of Petri nets

- A morphism of cubical sets $\varphi : C \to C'$ sends $n$-cells to $n$-cells respecting source and target.

We cannot unfold Petri nets!
Morphisms of Petri nets

- A morphism of cubical sets \( \varphi : C \rightarrow C' \)
sends \( n \)-cells to \( n \)-cells respecting source and target.

- A Petri net \( N = (P, M_0, E, \text{pre}, \text{post}) \) consists of
  - a set \( P \) of places
  - an initial marking \( M_0 \in \mathbb{N}^P \)
  - a set \( E \) of events
  - a precondition function \( \text{pre} : E \rightarrow \mathbb{N}^P \)
  - a postcondition function \( \text{post} : E \rightarrow \mathbb{N}^P \)
Morphisms of Petri nets

- A morphism of cubical sets $\varphi : C \to C'$ sends $n$-cells to $n$-cells respecting source and target.

- A Petri net $N = (P, M_0, E, \text{pre}, \text{post})$ consists of
  - a set $P$ of places
  - an initial marking $M_0 \in \mathbb{N}^P$
  - a set $E$ of events
  - a precondition function $\text{pre} : E \to \mathbb{N}^P$
  - a postcondition function $\text{post} : E \to \mathbb{N}^P$

A morphism of Petri nets $\varphi : N \to N'$ should be a pair of functions
- $\varphi_P : P \to P'$
- $\varphi_E : E \to E'$

preserving the initial marking, pre- and postconditions.
Morphisms of Petri nets

- A morphism of cubical sets \( \varphi : C \to C' \)
sends \( n \)-cells to \( n \)-cells respecting source and target.
  If \( a \) and \( b \) are independent in \( C \),
  \( \varphi(a) \) and \( \varphi(b) \) should be independent in \( C' \)

- A Petri net \( N = (P, M_0, E, \text{pre}, \text{post}) \) consists of
  - a set \( P \) of places
  - an initial marking \( M_0 \in \mathbb{N}^P \)
  - a set \( E \) of events
  - a precondition function \( \text{pre} : E \to \mathbb{N}^P \)
  - a postcondition function \( \text{post} : E \to \mathbb{N}^P \)

A morphism of Petri nets \( \varphi : N \to N' \) should be a pair of functions
- \( \varphi_P : P \to P' \)
- \( \varphi_E : E \to E' \)

preserving the initial marking, pre- and postconditions.
Morphisms of Petri nets

- A morphism of cubical sets \( \varphi : C \to C' \) sends \( n \)-cells to \( n \)-cells respecting source and target. If \( \varphi(a) \) and \( \varphi(b) \) are causally dependent in \( C' \), \( a \) and \( b \) should be causally dependent in \( C \).

- A Petri net \( N = (P, M_0, E, \text{pre}, \text{post}) \) consists of:
  - a set \( P \) of \texttt{places}
  - an initial marking \( M_0 \in \mathbb{N}^P \)
  - a set \( E \) of \texttt{events}
  - a precondition function \( \text{pre} : E \to \mathbb{N}^P \)
  - a postcondition function \( \text{post} : E \to \mathbb{N}^P \)

A morphism of Petri nets \( \varphi : N \to N' \) should be a pair of functions

- \( \varphi_P : P \to P' \)
- \( \varphi_E : E \to E' \)

preserving the initial marking, pre- and postconditions.
Morphisms of Petri nets

▶ A morphism of cubical sets $\varphi : C \rightarrow C'$ sends $n$-cells to $n$-cells respecting source and target. If $\varphi(a)$ and $\varphi(b)$ are causally dependent in $C'$, $a$ and $b$ should be causally dependent in $C$.

▶ A Petri net $N = (P, M_0, E, \text{pre}, \text{post})$ consists of
  ▶ a set $P$ of places
  ▶ an initial marking $M_0 \in \mathbb{N}^P$
  ▶ a set $E$ of events
  ▶ a precondition function $\text{pre} : E \rightarrow \mathbb{N}^P$
  ▶ a postcondition function $\text{post} : E \rightarrow \mathbb{N}^P$

A morphism of Petri nets $\varphi : N \rightarrow N'$ should be a pair of functions
  ▶ $\varphi_P : P \leftarrow P'$
  ▶ $\varphi_E : E \rightarrow E'$

preserving the initial marking, pre- and postconditions.
Morphisms of Petri nets

- A morphism of cubical sets $\varphi : C \rightarrow C'$ sends $n$-cells to $n$-cells respecting source and target. If $\varphi(a)$ and $\varphi(b)$ are causally dependent $C'$, $a$ and $b$ should be causally dependent in $C$

- A Petri net $N = (P, M_0, E, \text{pre}, \text{post})$ consists of
  - a set $P$ of places
  - an initial marking $M_0 \in \mathbb{N}^P$
  - a set $E$ of events
  - a precondition function $\text{pre} : E \rightarrow \mathbb{N}^P$
  - a postcondition function $\text{post} : E \rightarrow \mathbb{N}^P$

A morphism of Petri nets $\varphi : N \rightarrow N'$ should be a pair of functions

- $\varphi_P : P \leftarrow P'$
- $\varphi_E : E \rightarrow E'$

preserving the initial marking, pre- and postconditions.

$\Rightarrow$ We cannot unfold Petri nets!
The adjunction

This way we get two categories

- higher-dimensional automata
- Petri nets

and an adjunction between them

\[
\begin{array}{c}
\text{pn}(C) \rightarrow N \\
C \rightarrow \text{hda}(N)
\end{array}
\]

with

\[
\begin{array}{c}
\text{HDA} \\
\text{PN}
\end{array}
\]

\[
\begin{array}{c}
\text{pn} \\
\text{hda}
\end{array}
\]
From HDA to Petri nets

To every HDA $C$, we associate a Petri net $\text{pn}(C)$ whose

- events are labels of $C$
From HDA to Petri nets

To every HDA $C$, we associate a Petri net $\text{pn}(C)$ whose

- events are labels of $C$
- places are regions $R$ of $C$:
  - for every 0-cell $x$, an integer $R(x)$
  - for every label $a$, a pair of integers $(R'(a), R''(a))$ such that for every 1-cell $y$,

$$R'(\lambda(y)) = R(\varepsilon^-(y)) \quad R''(\lambda(y)) = R(\varepsilon^+(y)) \quad \ldots$$

\begin{figure}
\centering
\begin{tikzpicture}
  \node (y1) at (0,0) {$y_1$};
  \node (y2) at (2,0) {$y_2$};

  \node (x) at (-2,-2) {$x$};
  \node (z) at (0,2) {$z$};
  \node (4) at (-4,0) {$4$};
  \node (2) at (4,0) {$2$};

  \path[->,thick]
    (z) edge node[above] {$a$} (y2)
    (z) edge node[below] {$b$} (y1)
    (y2) edge node[above] {$-1, +0$} (4)
    (y1) edge node[below] {$-2, +3$} (4)
    (y2) edge node[below] {$-2, +3$} (2)
    (y1) edge node[above] {$-1, +0$} (2)
    (z) edge node[below] {$-2, +3$} (x)
    (z) edge node[above] {$a$} (x)
    (4) edge node[below] {$-2, +3$} (x)
    (2) edge node[above] {$-1, +0$} (x);
\end{tikzpicture}
\end{figure}
Results

An adjunction

- An extension Winskel’s “2-dimensional” adjunction between safe Petri nets and asynchronous transition systems
- A cleaner setting (no partial functions for example)
- This adjunction is not very “precise”
- Project: relate models of parallelism in higher dimension (Petri nets, HDA, event structures, . . . )
Results

An adjunction

- An extension Winskel’s “2-dimensional” adjunction between safe Petri nets and asynchronous transition systems
- A cleaner setting (no partial functions for example)
- This adjunction is not very “precise”
- Project: relate models of parallelism in higher dimension (Petri nets, HDA, event structures, . . . )

Future works

We can apply methods from topology:

- category of components
- homology
- . . .

and from Petri nets

- semi-linear invariants on places
- . . .