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

Petri nets	$\longleftrightarrow$	**Cubical Sets**
a very well-known and studied model		a geometrical model

An adjunction

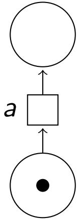
Petri nets $\longleftrightarrow$ **Cubical Sets**

a very well-known and studied model	a geometrical model

$$\frac{\mathrm{pn}(C) \to N}{C \to \mathrm{cs}(N)}$$

Petri nets

An abstract representation of processes focused on *resources*:

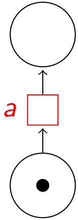


Petri net: a graph whose vertices are either

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

Petri nets

An abstract representation of processes focused on *resources*:

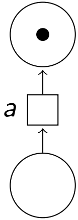


Petri net: a graph whose vertices are either

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

Petri nets

An abstract representation of processes focused on *resources*:

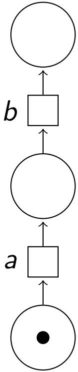


Petri net: a graph whose vertices are either

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

Typical situations

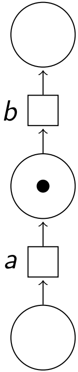
Petri nets can express **causality**:



Possible runs:

Typical situations

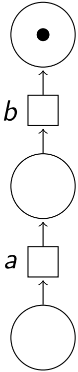
Petri nets can express **causality**:



Possible runs: a

Typical situations

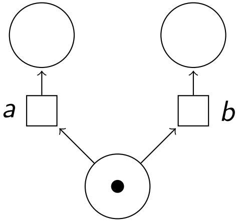
Petri nets can express **causality**:



Possible runs: *ab*

Typical situations

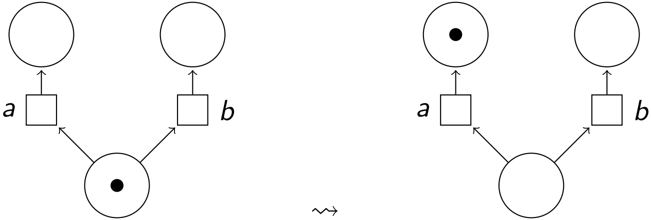
Petri nets can express **conflict**:



Possible runs:

Typical situations

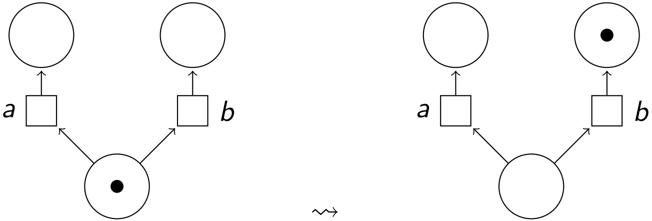
Petri nets can express **conflict**:



Possible runs: a

Typical situations

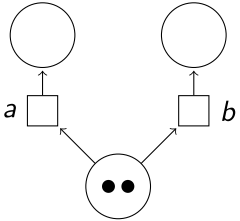
Petri nets can express **conflict**:



Possible runs: a or b

Typical situations

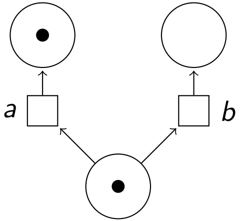
Petri nets can express **independence**:



Possible runs: *ab* or *ba* or *aa* or *bb*

Typical situations

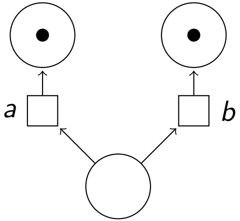
Petri nets can express **independence**:



Possible runs: *ab* or *ba* or *aa* or *bb*

Typical situations

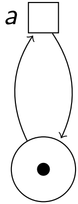
Petri nets can express **independence**:



Possible runs: *ab* or *ba* or *aa* or *bb*

Typical situations

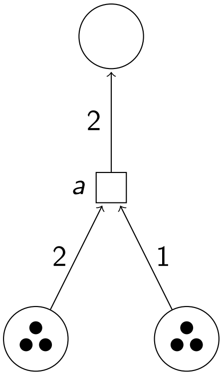
Petri nets can express **loops**:



Possible runs: $aaaaaaa\ldots$

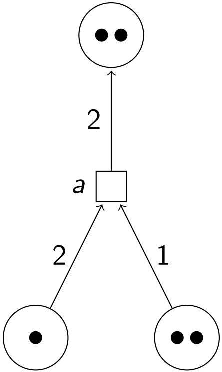
Taking multiplicities in account

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* $(P, M_0, E, \mathrm{pre}, \mathrm{post})$ 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 $\mathrm{pre} : E \to \mathbb{N}^P$
- a *postcondition* function $\mathrm{post} : E \to \mathbb{N}^P$

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

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

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

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

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

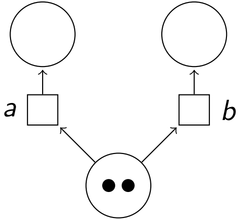
To every Petri net N we want to associate a *semantics* $[\![N]\!]$ which describes precisely the dynamic behavior of the net.

Semantics of Petri nets

To every Petri net N we want to associate a *semantics* $[\![N]\!]$ 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.



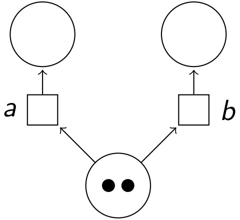
$$[\![N]\!] \quad = \quad \{ \, \varepsilon, a, b, ab, ba, aa, bb \, \}$$

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.



$$[\![N]\!] \quad = \quad \{\, \varepsilon, a, b, ab, ba, aa, bb \,\}$$

- We loose too much structure by forgetting about states!

Semantics of Petri nets

Idea 2

$[\![N]\!]$ should be a graph whose

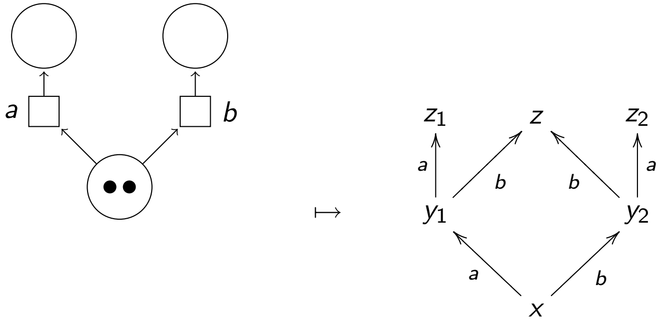
- vertices are reachable markings
- edges are transitions, labelled by events

Semantics of Petri nets

Idea 2

$[\![N]\!]$ should be a graph whose

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

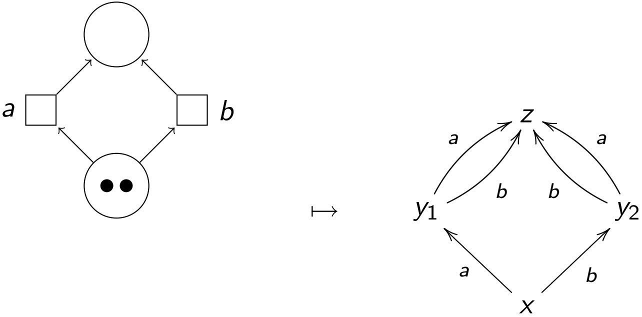


Semantics of Petri nets

Idea 2

$[\![N]\!]$ should be a graph whose

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

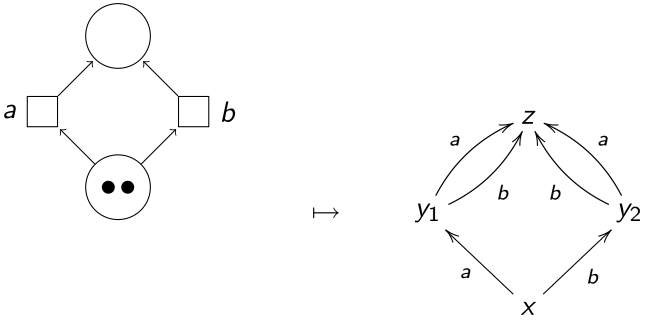


Semantics of Petri nets

Idea 2

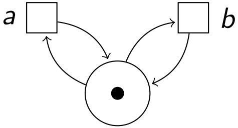
$[\![N]\!]$ should be a graph whose

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

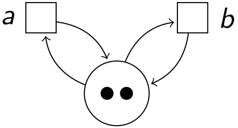


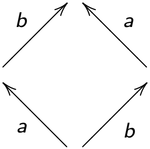
- We loose structure by forgetting about concurrency!

Taking concurrency in account

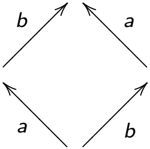


vs.

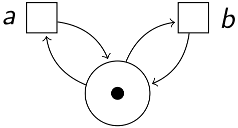




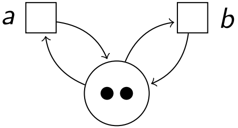
vs.

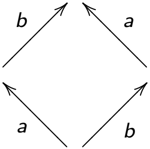


Taking concurrency in account

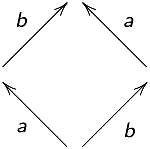


vs.



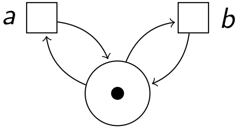


vs.

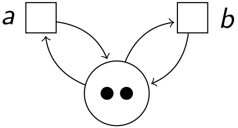


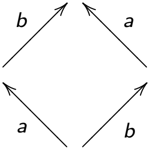
$(x := 3 \mid x := 4)$ vs. $(x := 3 \mid y := 4)$

Taking concurrency in account

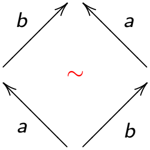


vs.





vs.



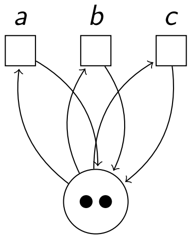
$(x := 3 \mid x := 4)$ vs. $(x := 3 \mid y := 4)$

Taking concurrency in account

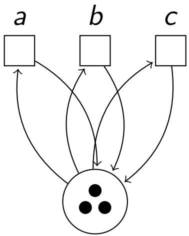
- We can now distinguish between an “empty square” and a “filled square”.

Taking concurrency in account

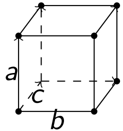
- We can now distinguish between an "empty square" and a "filled square".
- We should also go on in higher dimensions:



vs.



empty cube vs. filled cube



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

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

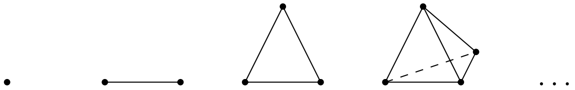
From Petri nets to Cubical Sets

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.

Simplicial sets

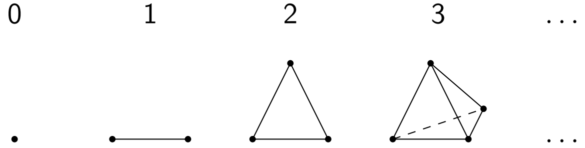
- Recall that a (augmented) **simplicial set** is a functor $S : \Delta^{\mathrm{op}} \to \mathbf{Set}$.
- Δ is the category of finite ordinals and increasing functions.
- Geometric intuition:

0 1 2 3 ...



Simplicial sets

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



- The arrows of Δ are generated by

$$\delta_i^n : n \to n+1 \quad \text{and} \quad \sigma_i^{n+1} : n+2 \to n+1$$

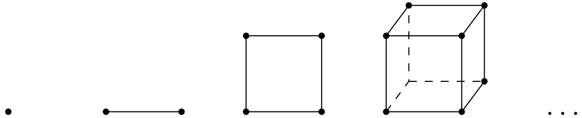
with $n \in \mathbb{N}$ and $0 \leq i \leq n$, subject to equations

$$\delta_i^{n+1}\delta_j^n = \delta_{j+1}^{n+1}\delta_i^n \qquad \ldots$$

Cubical sets

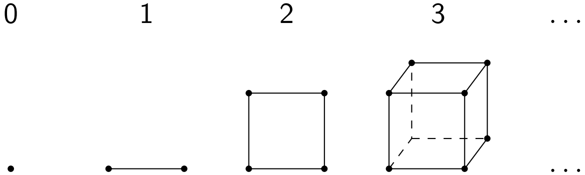
- A **cubical set** is a functor $C : \square^{\mathrm{op}} \to \mathbf{Set}$.
- Geometric intuition:

0 1 2 3 ...



Cubical sets

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

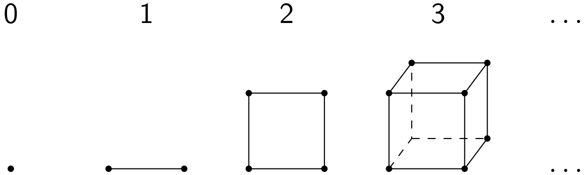


- The category $\square$ is generated by

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

Cubical sets

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



- The category $\square$ is generated by

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

source target degeneracy

The cubical category

The category $\square$ is the category generated by

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

subject to the equations

$$\begin{aligned}
\varepsilon_j^\alpha \varepsilon_i^\beta &= \varepsilon_i^\beta \varepsilon_{j-1}^\alpha && \text{with } i < j,\ \alpha, \beta \in \{-,+\} \\
\eta_j \eta_i &= \eta_{i-1} \eta_j && \text{with } i > j \\
\eta_j \varepsilon_i^\alpha &= \begin{cases} \varepsilon_i^\alpha \eta_{j-1} & \text{if } i < j \\ \mathrm{id} & \text{if } i = j \\ \varepsilon_i^\alpha \eta_j & \text{if } i > j \end{cases} && \text{with } \alpha \in \{-,+\}
\end{aligned}$$

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

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

$$\begin{tikzcd} & z & \\ y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\ & x \arrow[ul, "a"] \arrow[ur, "b"'] & \end{tikzcd}$$

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

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

$$
\begin{tikzcd}
 & z & \\
y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\
 & x \arrow[ul, "a"] \arrow[ur, "b"'] &
\end{tikzcd}
$$

$!\Sigma(0)$

$\{ * \}$

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

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

$$\begin{tikzcd}
 & z & \\
y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\
 & x \arrow[ul, "a"] \arrow[ur, "b"'] &
\end{tikzcd}$$

$!\,\Sigma(0)$	$!\,\Sigma(1)$
$\{\, * \,\}$	$\{\, *, a, b \,\}$

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

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

$$\begin{tikzcd}
& z & \\
y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\
& x \arrow[ul, "a"] \arrow[ur, "b"'] &
\end{tikzcd}$$

$!\,\Sigma(0)$	$!\,\Sigma(1)$	$!\,\Sigma(2)$
$\{\, * \,\}$	$\{\, *, a, b \,\}$	$\{\, *, a, b, ab, ba \,\}$

Labeled cubical sets

A **labeled cubical set** on an alphabet Σ is

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

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

$$\begin{tikzcd}
& z & \\
y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\
& x \arrow[ul, "a"] \arrow[ur, "b"'] &
\end{tikzcd}$$

$$\begin{array}{cccc}
{!}\,\Sigma(0) & {!}\,\Sigma(1) & {!}\,\Sigma(2) & \dots \\
\{ * \} & \{ *, a, b \} & \{ *, a, b, ab, ba \} & \dots
\end{array}$$

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 $\square$ is a **monoidal category**:

- We have a tensor product $\otimes$

A monoidal definition of cubical sets

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

- We have a tensor product $\otimes$

$$m_1 \xrightarrow{f} n_1$$

A monoidal definition of cubical sets

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

- We have a tensor product $\otimes$

$$m_2 \xrightarrow{g} n_2$$

$$m_1 \xrightarrow{f} n_1$$

A monoidal definition of cubical sets

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

- We have a tensor product $\otimes$

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

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

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

subject to the equations

$$\begin{aligned}
\varepsilon_j^\alpha \varepsilon_i^\beta &= \varepsilon_i^\beta \varepsilon_{j-1}^\alpha && \text{with } i < j,\ \alpha, \beta \in \{-,+\} \\
\eta_j \eta_i &= \eta_{i-1} \eta_j && \text{with } i > j \\
\eta_j \varepsilon_i^\alpha &= \begin{cases} \varepsilon_i^\alpha \eta_{j-1} & \text{if } i < j \\ \mathrm{id} & \text{if } i = j \\ \varepsilon_i^\alpha \eta_j & \text{if } i > j \end{cases} && \text{with } \alpha \in \{-,+\}
\end{aligned}$$

A monoidal definition of cubical sets

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

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

subject to the equations

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

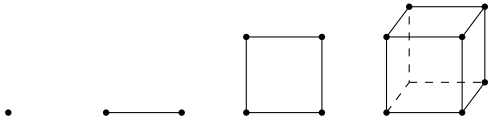
A monoidal definition of cubical sets

- 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 $\square$ are often monoidal: consider the functor $F : \square \to \mathbf{Top}$ defined by

0	1	2	3	...

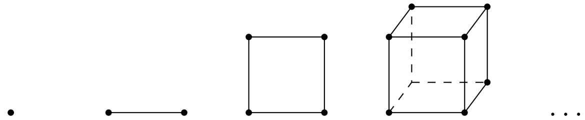


...

A monoidal definition of cubical sets

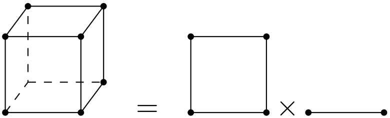
- A **monoidal functor** between monoidal categories is a functor which preserves tensor product.
- In particular, functors from $\square$ are often monoidal: consider the functor $F : \square \to \mathbf{Top}$ defined by

$$0 \qquad 1 \qquad 2 \qquad 3 \qquad \ldots$$



We have

$$F(2+1) \quad = \quad F(2) \times F(1)$$



Cubical objects

A cubical set is a functor

$$C : \square^{\mathrm{op}} \to \mathbf{Set}$$

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

Cubical objects

A cubical set is a functor

$$C : \square^{\mathrm{op}} \to \mathbf{Set}$$

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 $\mathcal{C}$ is an object A of $\mathcal{C}$ together with morphisms

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

such that

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

Cubical objects

Cubical objects

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

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

such that

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

Cubical objects

Cubical objects

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

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

such that

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

The labeling cubical set

$(\mathbf{Set}, \times, 1)$ is a monoidal category.
The object $1 = \{*\}$ is terminal in **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 $a_i \in \Sigma \uplus \{*\}$
- $\varepsilon_i^-, \varepsilon_i^+$ remove the i-th letter
- η_i inserts a $*$ at the i-th position

Symmetric cubical sets

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

$$\begin{tikzcd} & z & \\ y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\ & x \arrow[ul, "a"] \arrow[ur, "b"'] & \end{tikzcd}$$

Symmetric cubical sets

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

$$\begin{tikzcd}
 & z & \\
y_1 \arrow[ur, "b"] & \sim & y_2 \arrow[ul, "a"'] \\
 & x \arrow[ul, "a"] \arrow[ur, "b"'] &
\end{tikzcd}$$

In fact, we should keep both possibilities and remember that they are "almost the same": **Set** is a symmetric monoidal category

$$A \times B \quad \cong \quad B \times A$$

Symmetric cubical sets

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

$$C : \square_S^{\mathrm{op}} \to \mathbf{Set}$$

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

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

subject to the equations

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

Symmetric cubical sets

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

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

subject to the equations

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

$$\begin{aligned}
(\gamma \otimes 1) \circ (1 \otimes \gamma) \circ (\gamma \otimes 1) \quad &= \quad (1 \otimes \gamma) \circ (\gamma \otimes 1) \circ (1 \otimes \gamma) \\
\gamma \circ \gamma \quad &= \quad 2 \\
(\eta \otimes 1) \circ \gamma \quad &= \quad 1 \otimes \eta \\
(1 \otimes \eta) \circ \gamma \quad &= \quad \eta \otimes 1 \\
&\dots
\end{aligned}$$

Higher-dimensional automata

To every, Petri net N we associate a **higher-dimensional automaton** $\mathrm{hda}(N)$ 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 $\varphi : C \longrightarrow C'$
  sends n-cells to n-cells respecting source and target.

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

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

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

  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 C',
  a and b should be causally dependent in C
- A *Petri net* $N = (P, M_0, E, \mathrm{pre}, \mathrm{post})$ consists of
  - a set P of *places*
  - an *initial marking* $M_0 \in \mathbb{N}^P$
  - a set E of *events*
  - a *precondition* function $\mathrm{pre} : E \to \mathbb{N}^P$
  - a *postcondition* function $\mathrm{post} : E \to \mathbb{N}^P$

  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 C',
  a and b should be causally dependent in C
- A *Petri net* $N = (P, M_0, E, \mathrm{pre}, \mathrm{post})$ consists of
  - a set P of *places*
  - an *initial marking* $M_0 \in \mathbb{N}^P$
  - a set E of *events*
  - a *precondition* function $\mathrm{pre} : E \to \mathbb{N}^P$
  - a *postcondition* function $\mathrm{post} : E \to \mathbb{N}^P$

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

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

  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 C', a and b should be causally dependent in C
- A *Petri net* $N = (P, M_0, E, \mathrm{pre}, \mathrm{post})$ consists of
  - a set P of *places*
  - an *initial marking* $M_0 \in \mathbb{N}^P$
  - a set E of *events*
  - a *precondition* function $\mathrm{pre} : E \to \mathbb{N}^P$
  - a *postcondition* function $\mathrm{post} : E \to \mathbb{N}^P$

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

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

  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

$$\frac{\mathrm{pn}(C) \to N}{C \to \mathrm{hda}(N)}$$

with

$$\begin{tikzcd} \mathbf{HDA} \arrow[r, bend left, "\mathrm{pn}"] \arrow[r, phantom, "\perp"] & \mathbf{PN} \arrow[l, bend left, "\mathrm{hda}"] \end{tikzcd}$$

From HDA to Petri nets

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

- events are labels of C

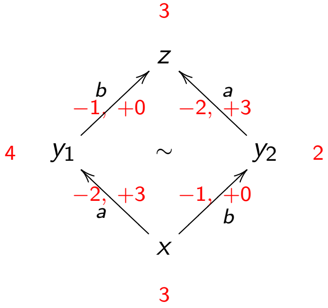
From HDA to Petri nets

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

$$R'(\lambda(y)) = R(\varepsilon^-(y)) \qquad R''(\lambda(y)) = R(\varepsilon^+(y)) \qquad \dots$$



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
- . . .