Categorical Geometry of Interaction and Application to Higher-Order Quantum Computation

Based on: IH & N. Hoshino, **Semantics of Higher-Order Quantum Computation via Geometry of Interaction,** Proc. LICS 2011. (Extended ver. coming soon)

Ichiro Hasuo
University of Tokyo (JP)

Naohiko Hoshino
RIMS, Kyoto University (JP)

Supported by Aihara Innovative Mathematical Modelling Project, FIRST Program, JSPS/CSTP

東京大学
THE UNIVERSITY OF TOKYO

京都大学
KYOTO UNIVERSITY

GaLoP (Queen Mary) 2013/7/19

Highlights

- **Categorical GoI** [Abramsky, Haghverdi, Scott]
  - Categorical axiomatization of "when we can run a **GoI business**"
  - **not** like "category of games"

Highlights

- **Categorical GoI** [Abramsky, Haghverdi, Scott]
  - Categorical axiomatization of "when we can run a **GoI business**"
  - **not** like "category of games"
- Combined with **coalgebras** [Rutten, Jacobs, ...]
  - Nice operational flavor!

Highlights

- **Categorical GoI** [Abramsky, Haghverdi, Scott]
  - Categorical axiomatization of "when we can run a **GoI business**"
  - **not** like "category of games"
- Combined with **coalgebras** [Rutten, Jacobs, ...]
  - Nice operational flavor!
- Application: **quantum λ-calculus** [Selinger, Valiron, van Tonder, ...]
  - "The categorical GoI workflow"

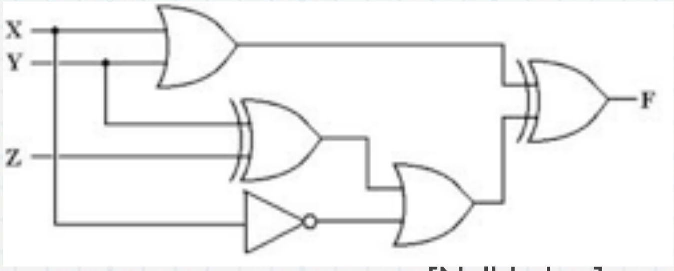
Part 1

Functional QPL: Some Contexts

Quantum Programming Language

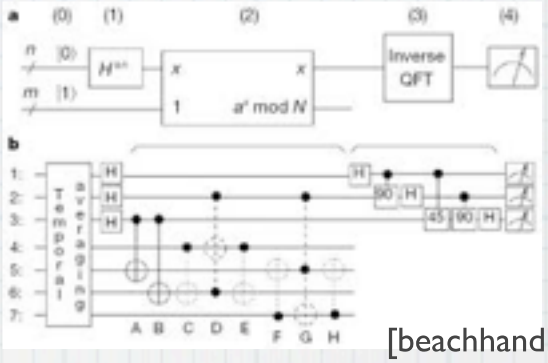
Classical	Quantum
(Boolean) circuit	Quantum circuit

(Boolean) circuit



[Null-Lobur]

Quantum circuit



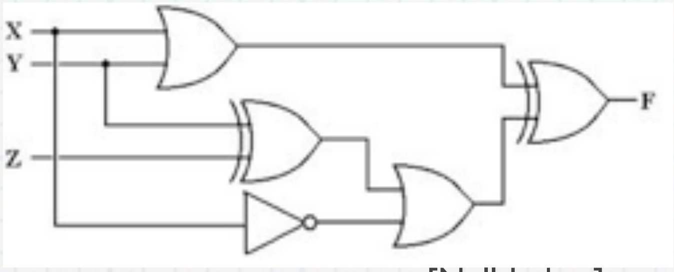
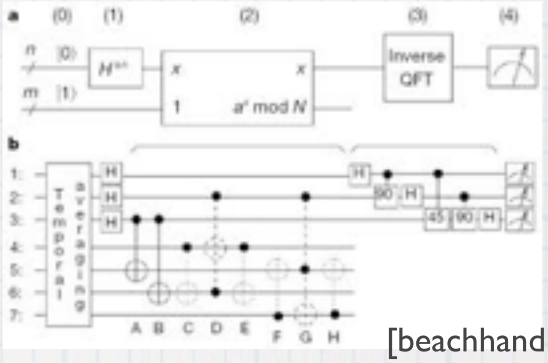
[beachhandball.es]

Programming language

```
int i,j;
int factorial(int k)
{
    j=1;
    for (i=1; i<=k; i++)
        j=j*i;
    return j;
}
```

Hasuo (Tokyo)

Quantum Programming Language

Classical		Quantum	
(Boolean) circuit	 [Null-Lobur]	Quantum circuit	 [beachhandball.es]

Programming language

```
int i,j;
int factorial(int k)
{
    j=1;
    for (i=1; i<=k; i++)
        j=j*i;
    return j;
}
```

Quantum programming language

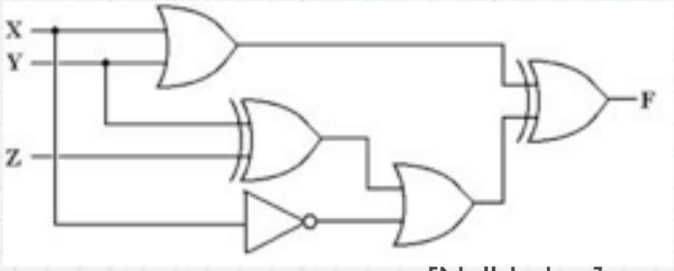
$$\begin{aligned}\mathbf{telep} = \quad & \mathit{let}\ \langle x, y\rangle = \mathbf{EPR} * \mathit{in} \\ & \quad \mathit{let}\ f = \mathbf{BellMeasure}\ x\ \mathit{in} \\ & \quad \mathit{let}\ g = \mathbf{U}\ y \\ & \quad \mathit{in}\ \langle f, g\rangle.\end{aligned}$$

[Selinger-Valiron]

Quantum Programming Language

Classical

(Boolean) circuit



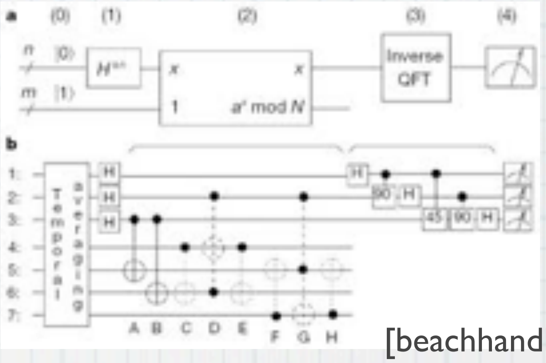
[Null-Lobur]

Programming language

```
int i,j;
int factorial(int k)
{
    j=1;
    for (i=1; i<=k; i++)
        j=j*i;
    return j;
}
```

Quantum

Quantum circuit



[beachhandball.es]

Quantum programming language

$$
\begin{aligned}
\mathbf{telep} = \quad & \mathit{let}\ \langle x, y\rangle = \mathbf{EPR} * \mathit{in} \\
& \quad \mathit{let}\ f = \mathbf{BellMeasure}\ x\ \mathit{in} \\
& \quad \mathit{let}\ g = \mathbf{U}\ y \\
& \quad \mathit{in}\ \langle f, g\rangle.
\end{aligned}
$$

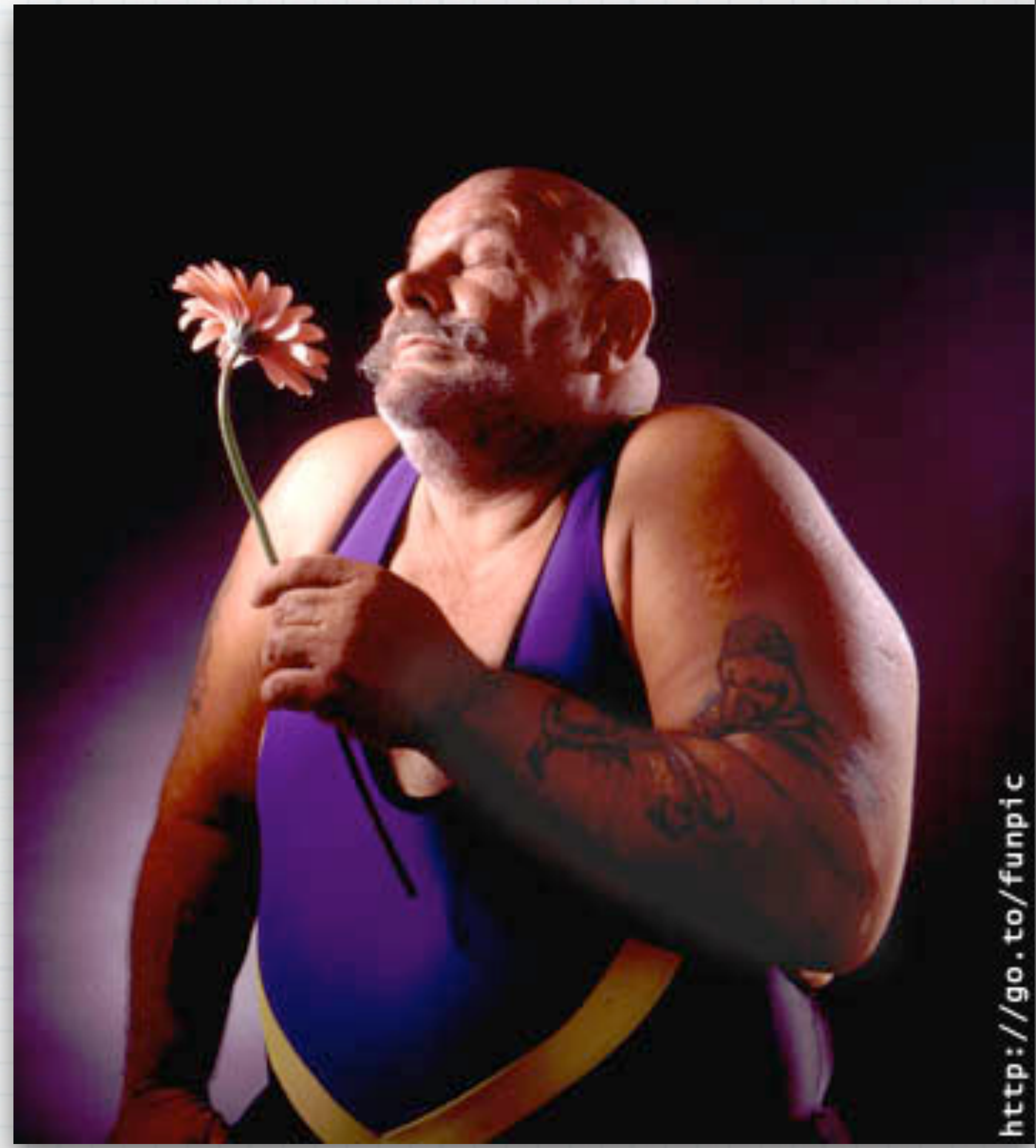
[Selinger-Valiron]

- For discovery of **algorithms**
- For **reasoning, verification**

Functional Quantum Programming Language

Functional Quantum Programming Language

- A **real man**'s programming style



http://go.to/funpic

Hasuo (Tokyo)

Functional Quantum Programming Language

- A **real man's** programming style
- Heavily used in the **financial sector**
- ...

ICFP'11 Sponsers (Tokyo, Sep 2011)

Functional Quantum Programming Language

- A **real man's** programming style
- Heavily used in the **financial sector**
- ...
- **Mathematically nice and clean**
  - Aids semantical study
  - **Transfer** from classical to quantum

ICFP'11 Sponsers (Tokyo, Sep 2011)

Functional QPL: Syntax

- **Linear λ-calculus**

  + quantum primitives [van Tonder, Selinger, Valiron, ...]

- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable:
  - Duplicable (classical) data: by the **!-modality**

Functional QPL: Syntax

- **Linear λ-calculus**

  + quantum primitives

Preparation/Unitary transformation/Measurement

[van Tonder, Selinger, Valiron, ...]

- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable:
  - Duplicable (classical) data: by the **!-modality**

Hasuo (Tokyo)

Friday, July 19, 13

Functional QPL: Syntax

- **Linear λ-calculus**
  + quantum primitives

Preparation/Unitary transformation/Measurement

[van Tonder, Selinger, Valiron, ...]

- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable:

$$\lambda x.\ \langle \mathtt{meas}\, x,\ \mathtt{meas}\, x \rangle$$

Functional QPL: Syntax

- **Linear λ-calculus**

  + quantum primitives

Preparation/Unitary transformation/Measurement

[van Tonder, Selinger, Valiron, ...]

- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable:

$$\boxed{\lambda x.\ \langle \mathtt{meas}\ x,\ \mathtt{meas}\ x \rangle}$$

Functional QPL: Syntax

- Linear λ-calculus
  + quantum primitives

Preparation/Unitary transformation/Measurement

[van Tonder, Selinger, Valiron, ...]

- Linearity for no-cloning
  - "Input can be used only once"
  - Not allowed/typable: $\lambda x.\ \langle \mathtt{meas}\, x,\ \mathtt{meas}\, x \rangle$
  - Duplicable (classical) data: by the !-modality

$$\vdash \mathtt{tt} : {!}\,\mathtt{bit}$$

Functional QPL: Syntax

- **Linear λ-calculus**
  + quantum primitives

Preparation/Unitary transformation/Measurement

[van Tonder, Selinger, Valiron, ...]

- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable: $\lambda x.\ \langle \mathtt{meas}\ \overset{\searrow}{x},\ \mathtt{meas}\ \overset{\searrow}{x} \rangle$
  - Duplicable (classical) data: by the **!-modality**

$$\vdash \mathtt{tt} : {!}\,\mathtt{bit}$$

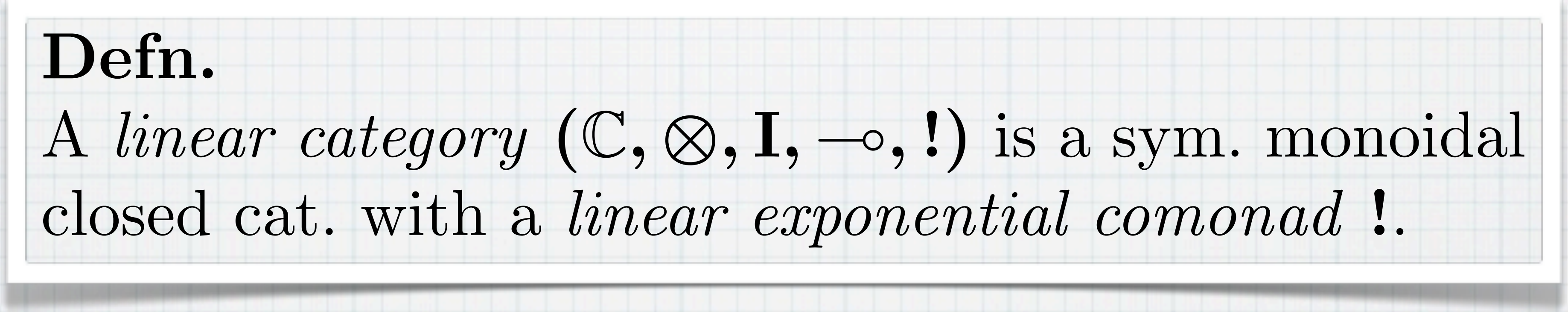
"arbitrary many copies"

Functional QPL: Semantics

Functional QPL: Semantics

- Denotational semantics

  - **Linear category**: [Benton & Wadler, Bierman]
    (axioms for) a categorical model of linear λ-calculus

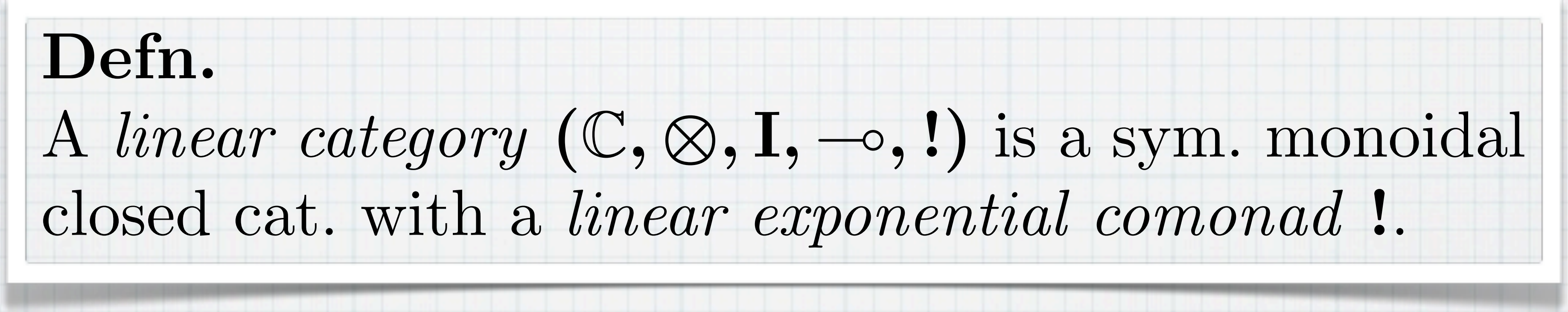


Defn.
A *linear category* $(\mathbb{C}, \otimes, \mathbf{I}, \multimap, \mathbf{!})$ is a sym. monoidal closed cat. with a *linear exponential comonad* $\mathbf{!}$.

Functional QPL: Semantics

- Denotational semantics

  - **Linear category**: [Benton & Wadler, Bierman] (axioms for) a categorical model of linear λ-calculus



Defn.
A *linear category* $(\mathbb{C}, \otimes, \mathbf{I}, \multimap, !)$ is a sym. monoidal closed cat. with a *linear exponential comonad* $!$.

  - For functional QPL? Is **Hilb** (or alike) a linear cat.?

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category

- **Challenge**: coexistence of **quantum** and **classical** data
- Only partial results
  - [Selinger & Valiron, '08]:
    for strictly linear fragmant (w/o !)

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category

monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

! (for duplicable data)

- **Challenge**: coexistence of **quantum** and **classical** data
- Only partial results
  - [Selinger & Valiron, '08]: for strictly linear fragmant (w/o **!**)

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category

monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

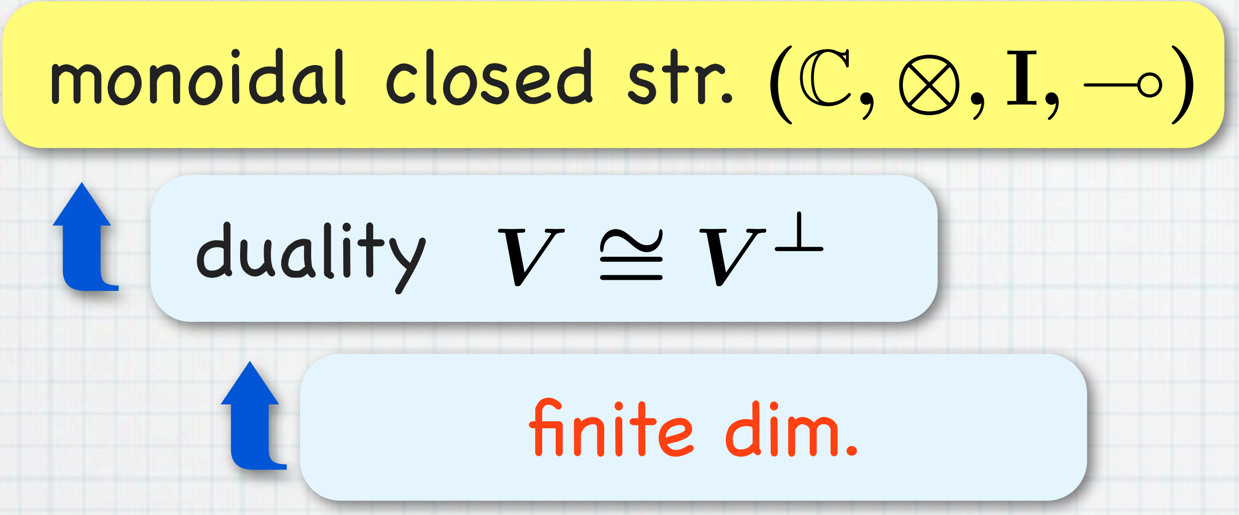
! (for duplicable data)

duality $V \cong V^{\perp}$

- **Challenge**: coexistence of **quantum** and **classical** data
- Only partial results
  - [Selinger & Valiron, '08]:
    for strictly linear fragmant (w/o **!**)

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category



monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

duality $V \cong V^{\perp}$

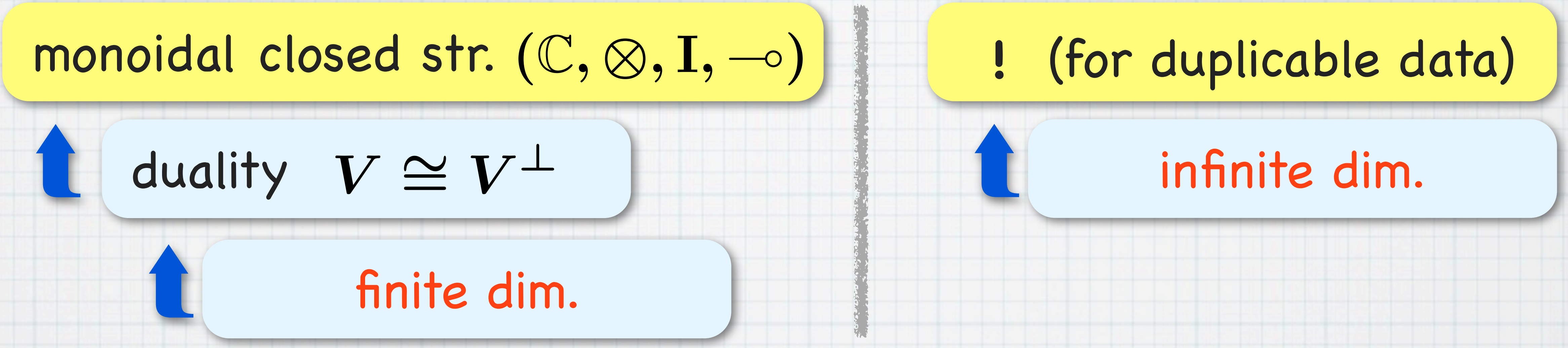
finite dim.

! (for duplicable data)

- **<u>Challenge</u>**: coexistence of **quantum** and **classical** data
- Only partial results
  - [Selinger & Valiron, '08]:
    for strictly linear fragmant (w/o **!**)

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category



monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

duality $V \cong V^{\perp}$

finite dim.

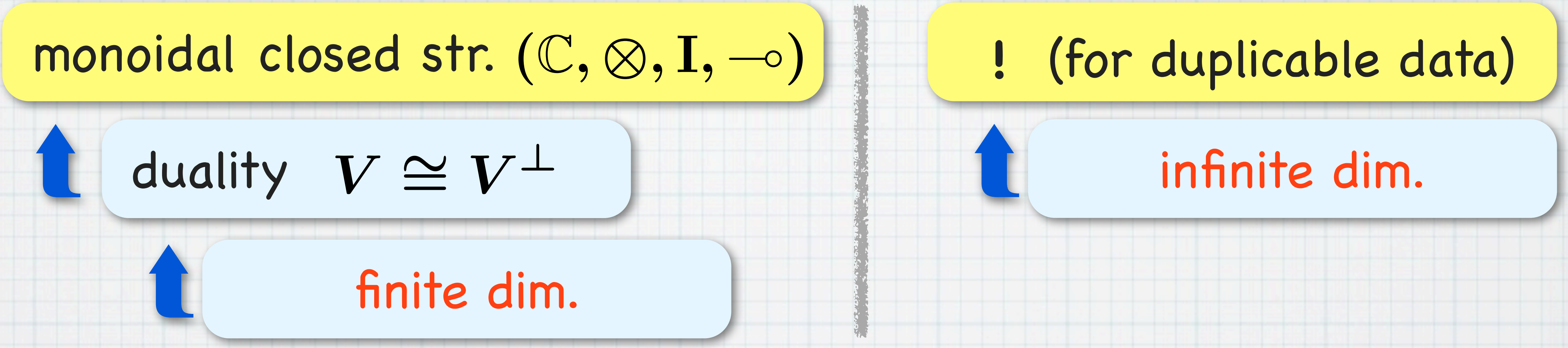
! (for duplicable data)

infinite dim.

- **Challenge**: coexistence of **quantum** and **classical** data
- Only partial results
  - [Selinger & Valiron, '08]:
    for strictly linear fragmant (w/o **!**)

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category



monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

duality $V \cong V^{\perp}$

finite dim.

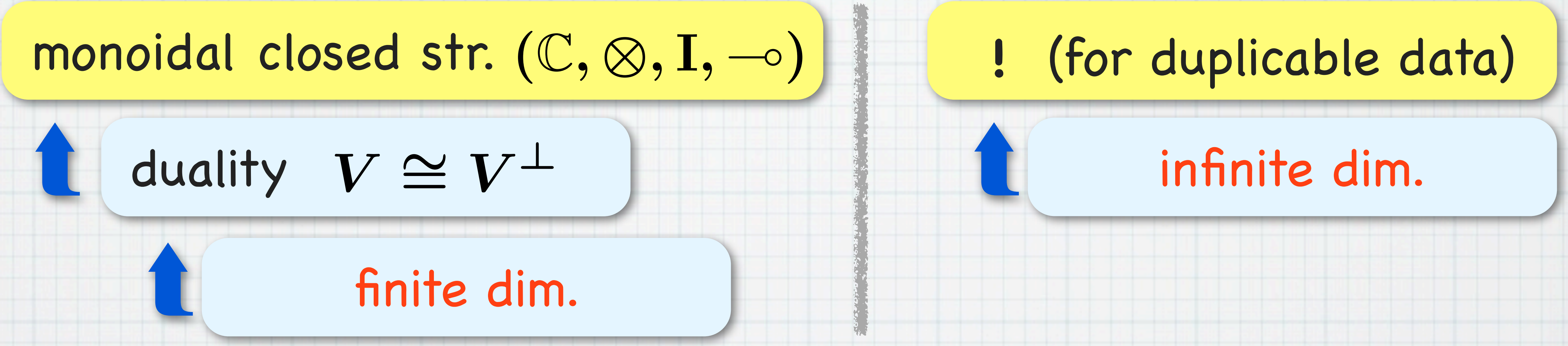
$!$ (for duplicable data)

infinite dim.

- **Challenge**: coexistence of **quantum** and **classical** data

Functional QPL: Semantics

- **Hilb** (or alike) is **not** a linear category



monoidal closed str. $(\mathbb{C}, \otimes, \mathbf{I}, \multimap)$

duality $V \cong V^{\perp}$

finite dim.

! (for duplicable data)

infinite dim.

- **<u>Challenge</u>**: coexistence of **quantum** and **classical** data

- Only partial results

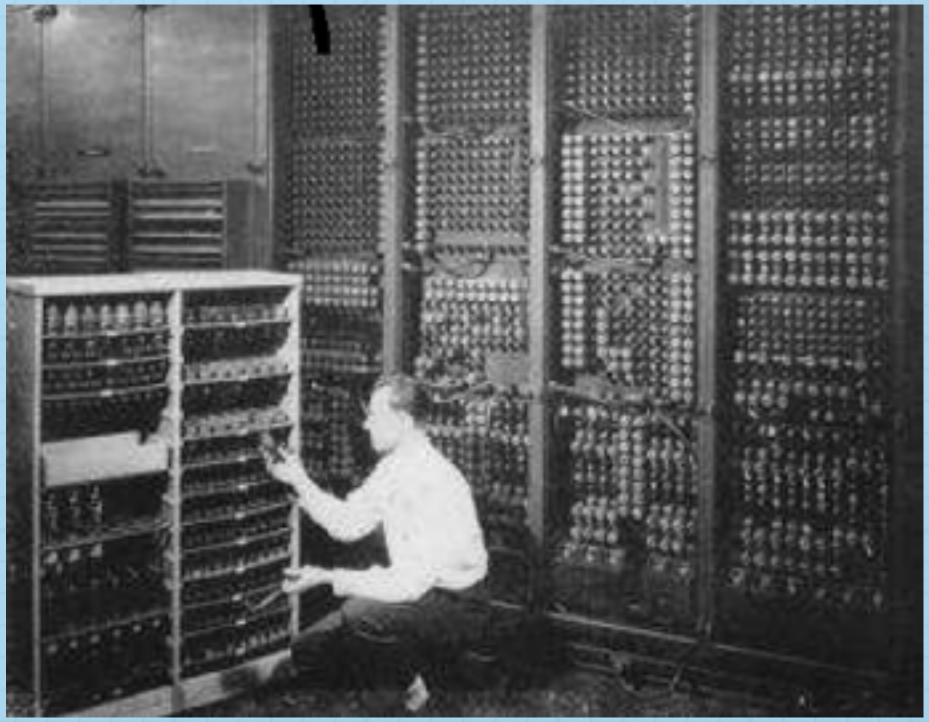
  - [Selinger & Valiron, '08]:
    for strictly linear fragmant (w/o **!**)

“Quantum Data, Classical Control”

Illustration by N. Hoshino

Quantum data

Classical control



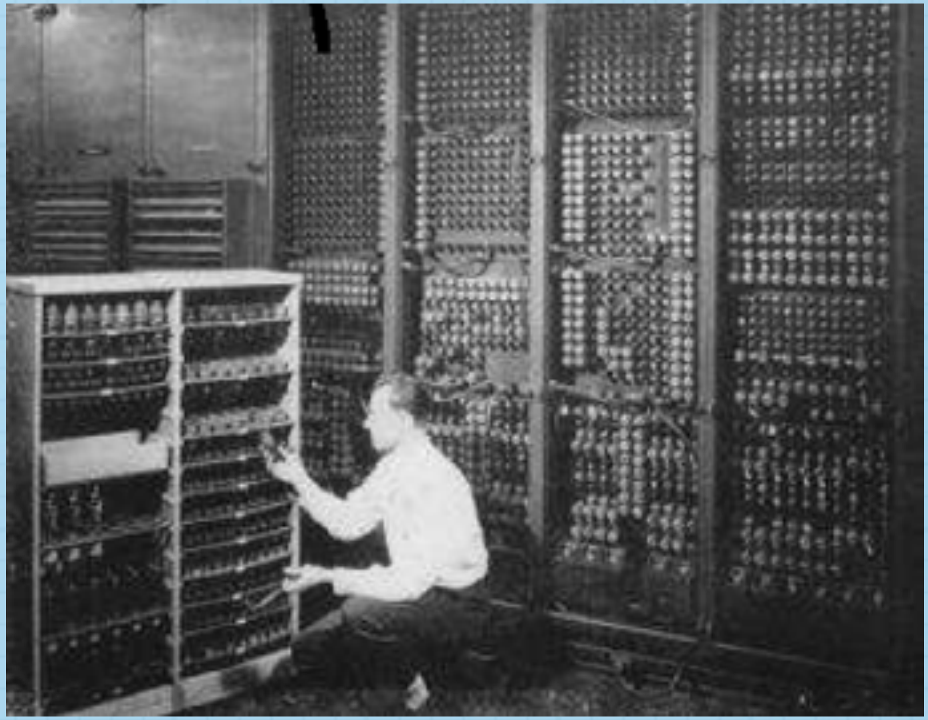
“Quantum Data, Classical Control”

Illustration by N. Hoshino

Quantum data

$$\frac{1}{\sqrt{2}}$$

Classical control



“Quantum Data, Classical Control”

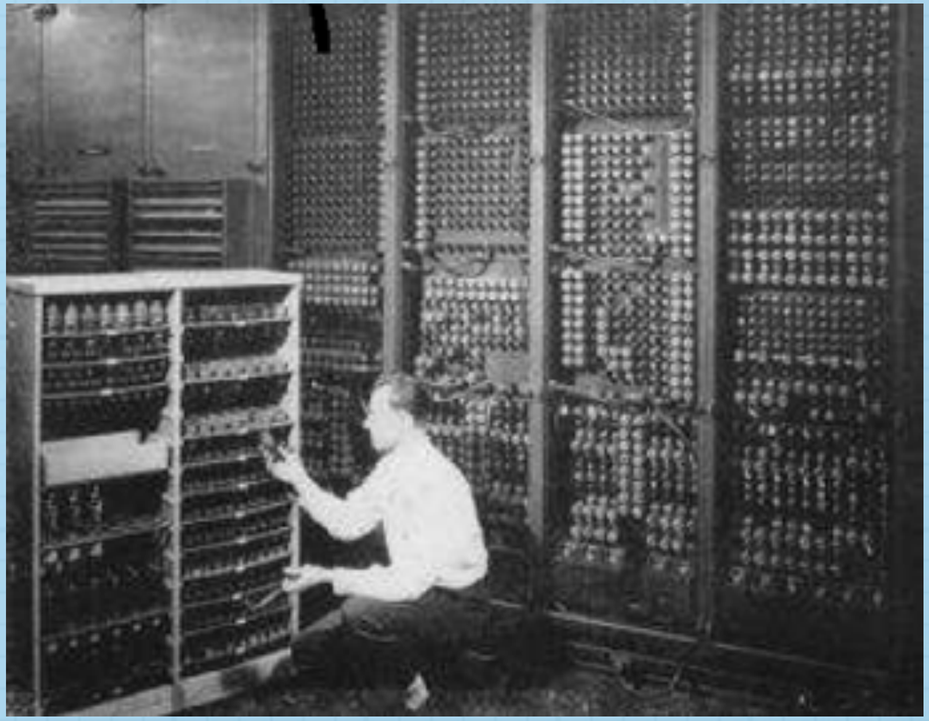
Illustration by N. Hoshino

Quantum data

$$\frac{1}{\sqrt{2}}$$

$$+\frac{1}{\sqrt{2}}$$

Classical control



What We Do

- **GoI (Geometry of Interaction)** [Girard '89]

  An "implementation" of **classical control**

$$\mathbf{tr}(f) =$$

$$f_{XY} \sqcup \left(\coprod_{n\in\mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

What We Do

$$\mathbf{tr}(f) = f_{XY} \sqcup \left(\coprod_{n\in\mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

- **GoI (Geometry of Interaction)** [Girard '89]

  An "implementation" of **classical control**

What We Do

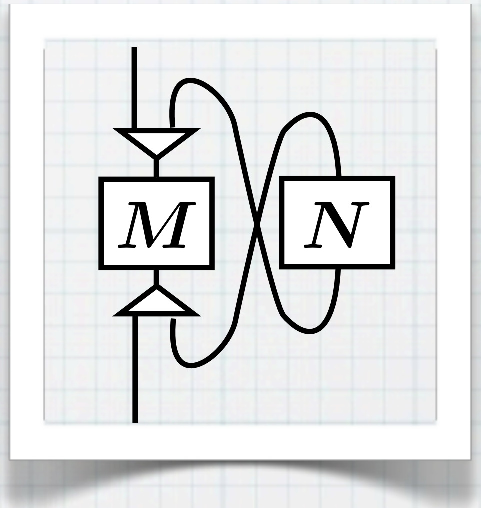
- **GoI (Geometry of Interaction)** [Girard '89]

  An "implementation" of **classical control**

$$\mathbf{tr}(f) = f_{XY} \sqcup \left(\coprod_{n\in\mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

- **Categorical GoI** [Abramsky, Haghverdi, Scott '02]

  Its categorical axiomatics



What We Do

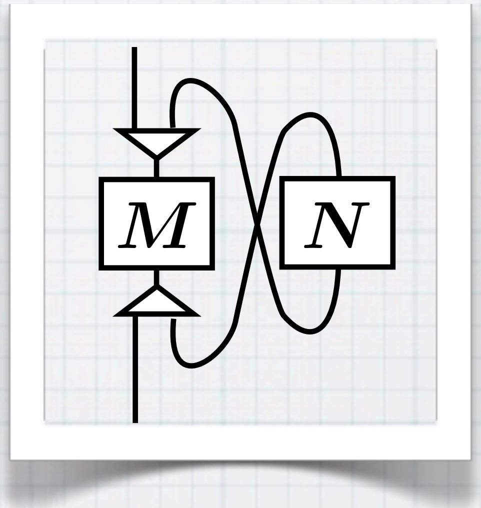
- **GoI (Geometry of Interaction)** [Girard '89]

  An "implementation" of **classical control**

$$\mathbf{tr}(f) = f_{XY} \sqcup \left(\coprod_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

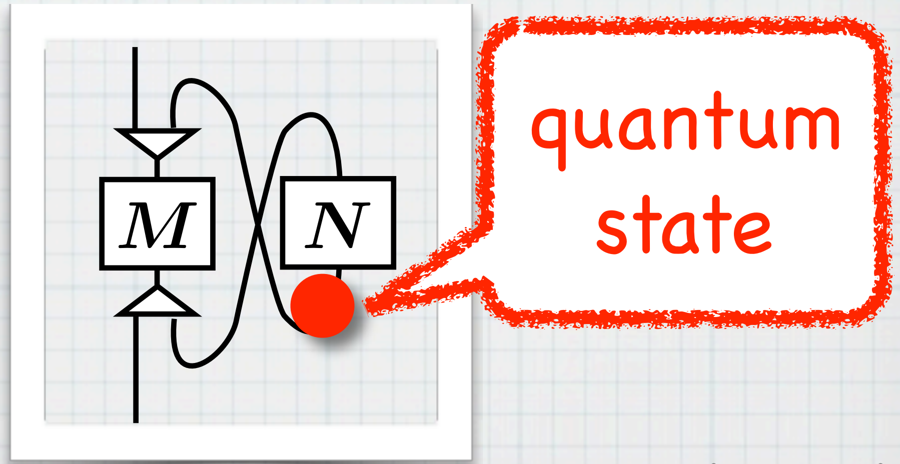
- **Categorical GoI** [Abramsky, Haghverdi, Scott '02]

  Its categorical axiomatics



- We add a **quantum layer** to GoI
  - ➜ "Quantum data, classical control"
  - Used: theory of coalgebra

    [Hasuo, Jacobs, Sokolova '07] [Jacobs '10]



Part 2

The Categorical GoI Workflow

GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88

GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M

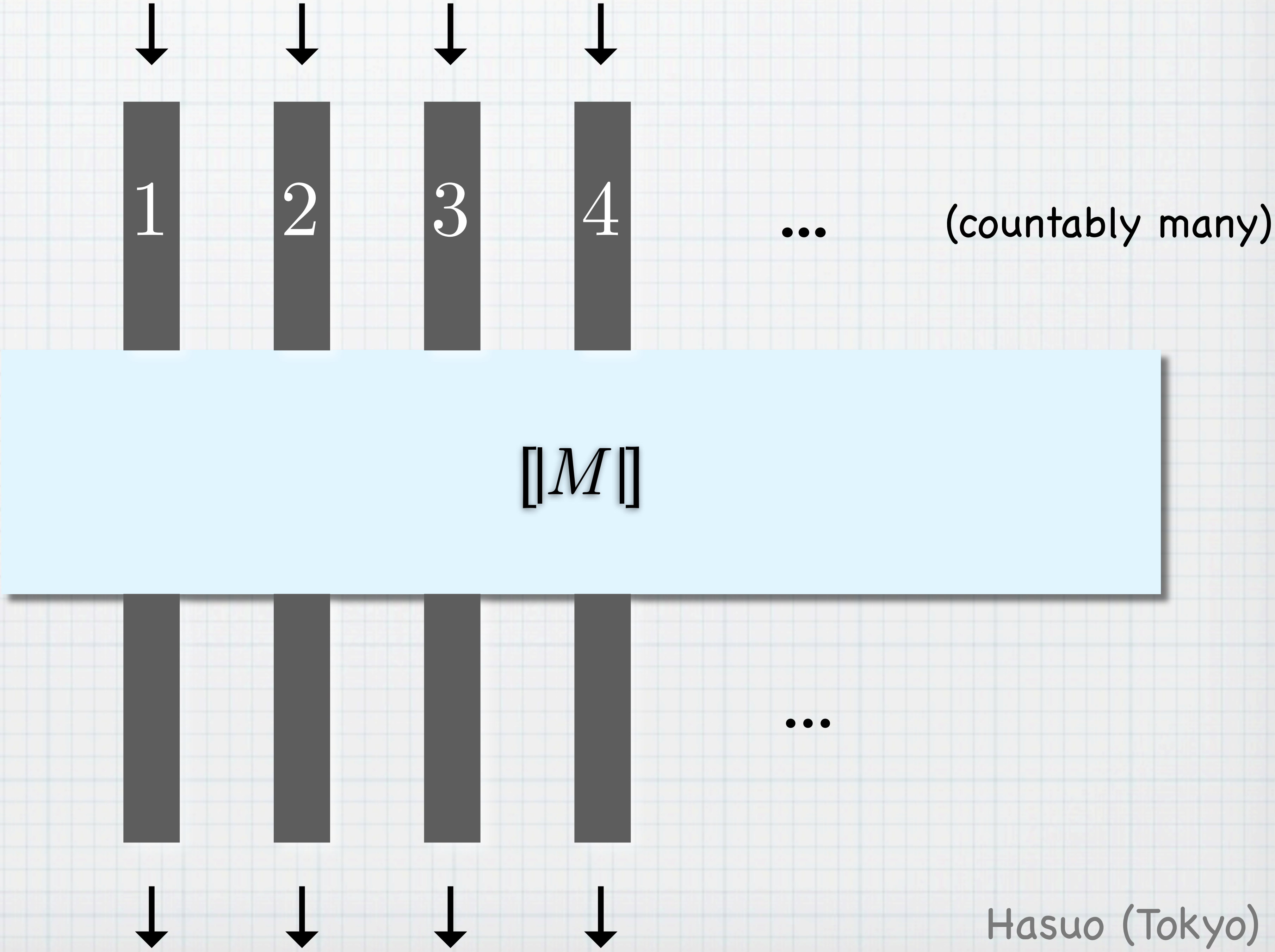
GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
- In this talk:
  - Its categorical formulation [Abramsky, Haghverdi, Scott '02]
  - "The GoI Animation"

The GoI Animation

$$[\![M]\!] \;=\; (\,\mathbb{N} \rightharpoonup \mathbb{N},\;\text{a partial function}\,)$$

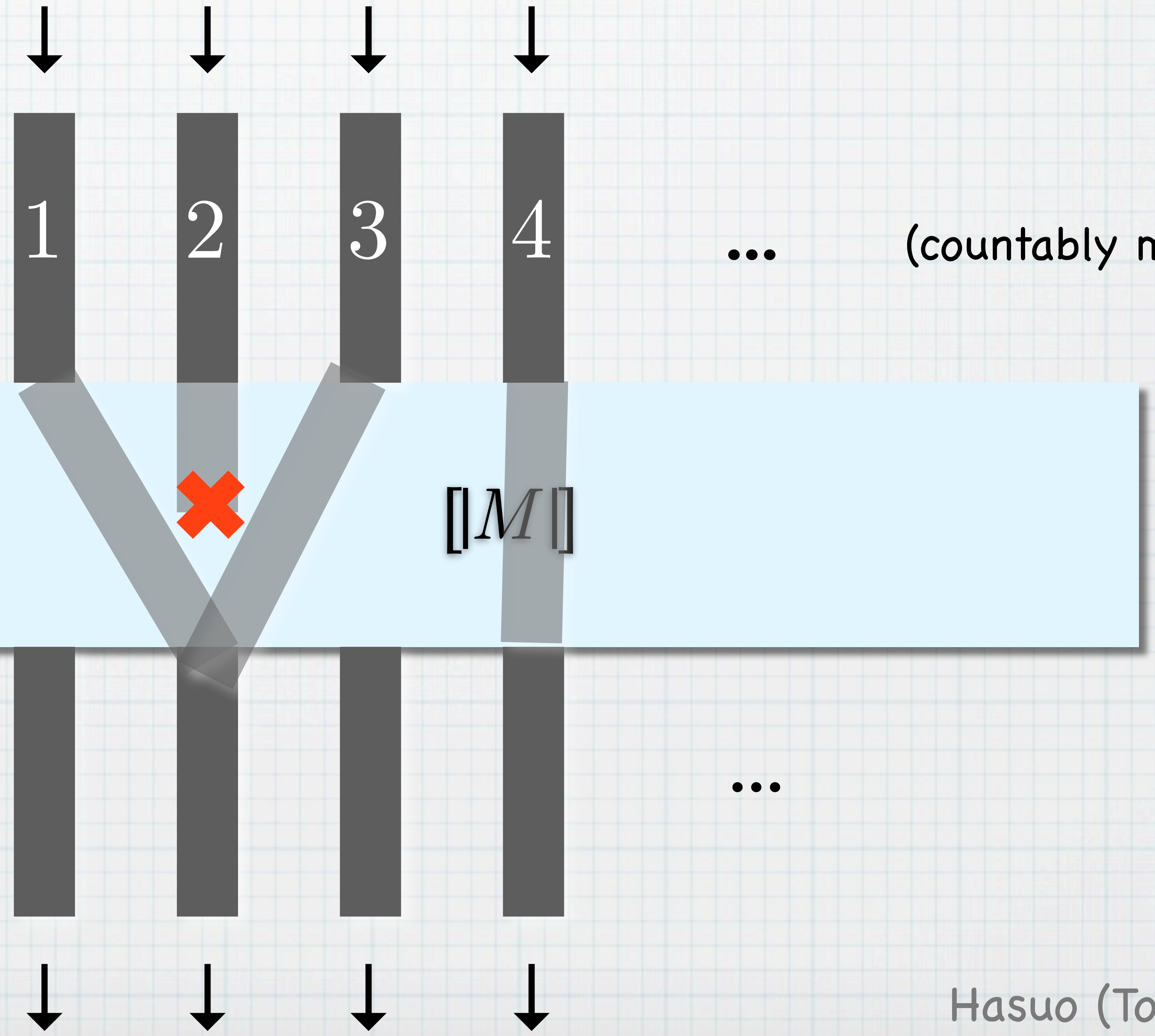
$$=\;\text{“piping”}$$



The GoI Animation

$$[\![M]\!] \;=\; (\,\mathbb{N} \rightharpoonup \mathbb{N},\ \text{a partial function}\,)$$

$$=\ \text{“piping”}$$

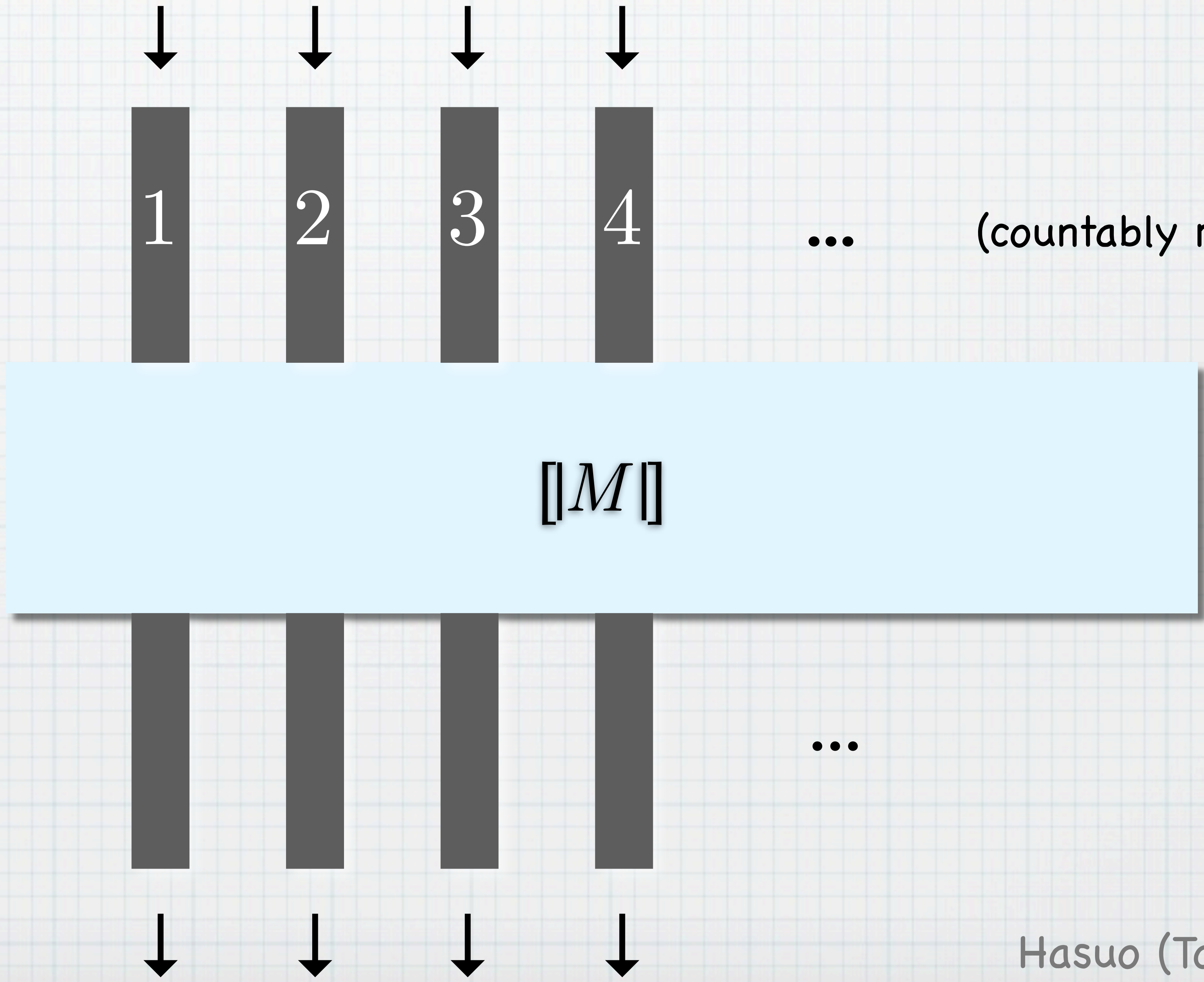


(countably many)

The GoI Animation

$$[\![M]\!] = (\ \mathbb{N} \rightharpoonup \mathbb{N},\ \text{a partial function}\ )$$

$$= \text{“piping”}$$

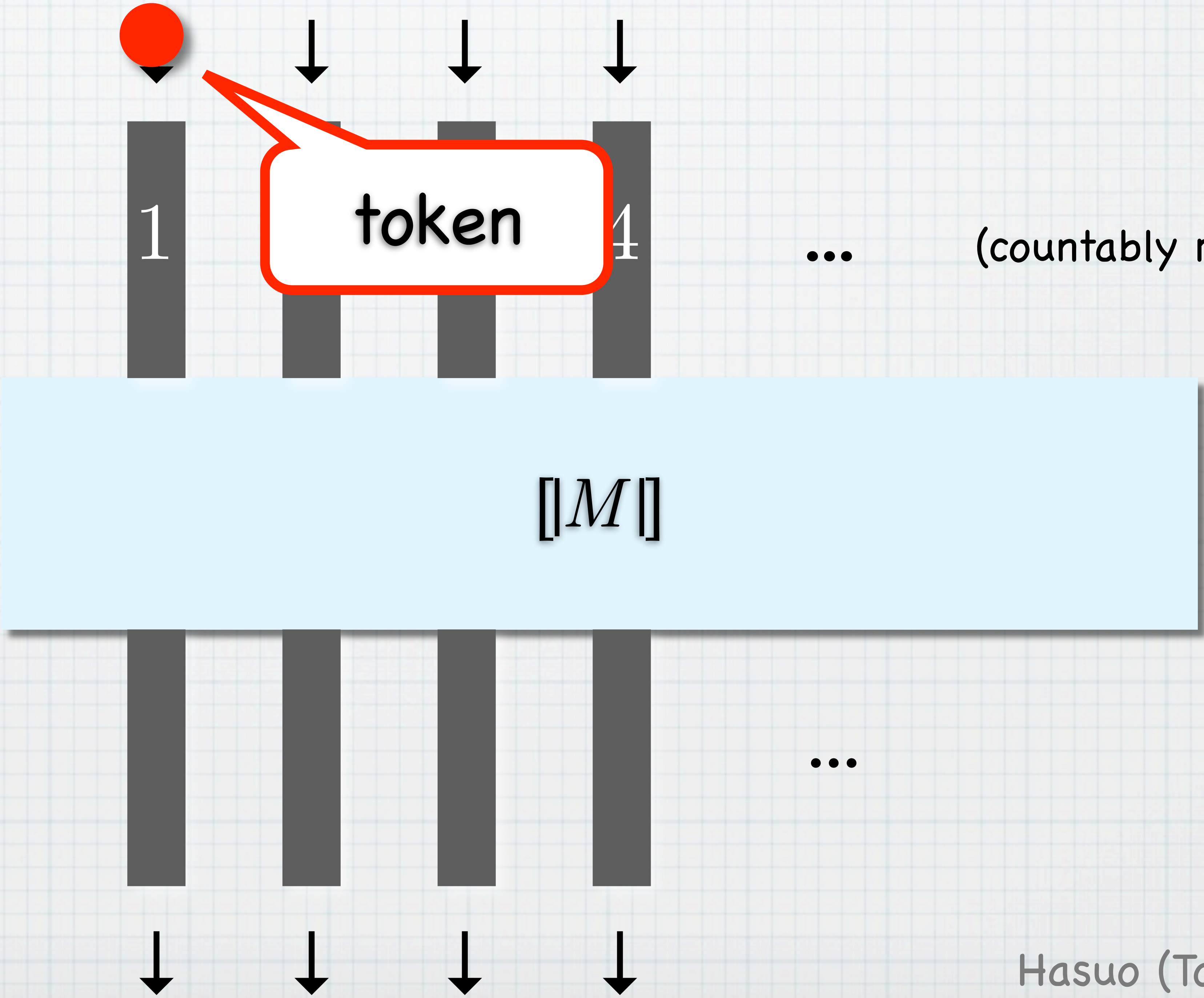


(countably many)

The GoI Animation

$$[\![M]\!] \;=\; (\,\mathbb{N} \rightharpoonup \mathbb{N},\;\text{a partial function}\,)$$

$$=\;\text{“piping”}$$

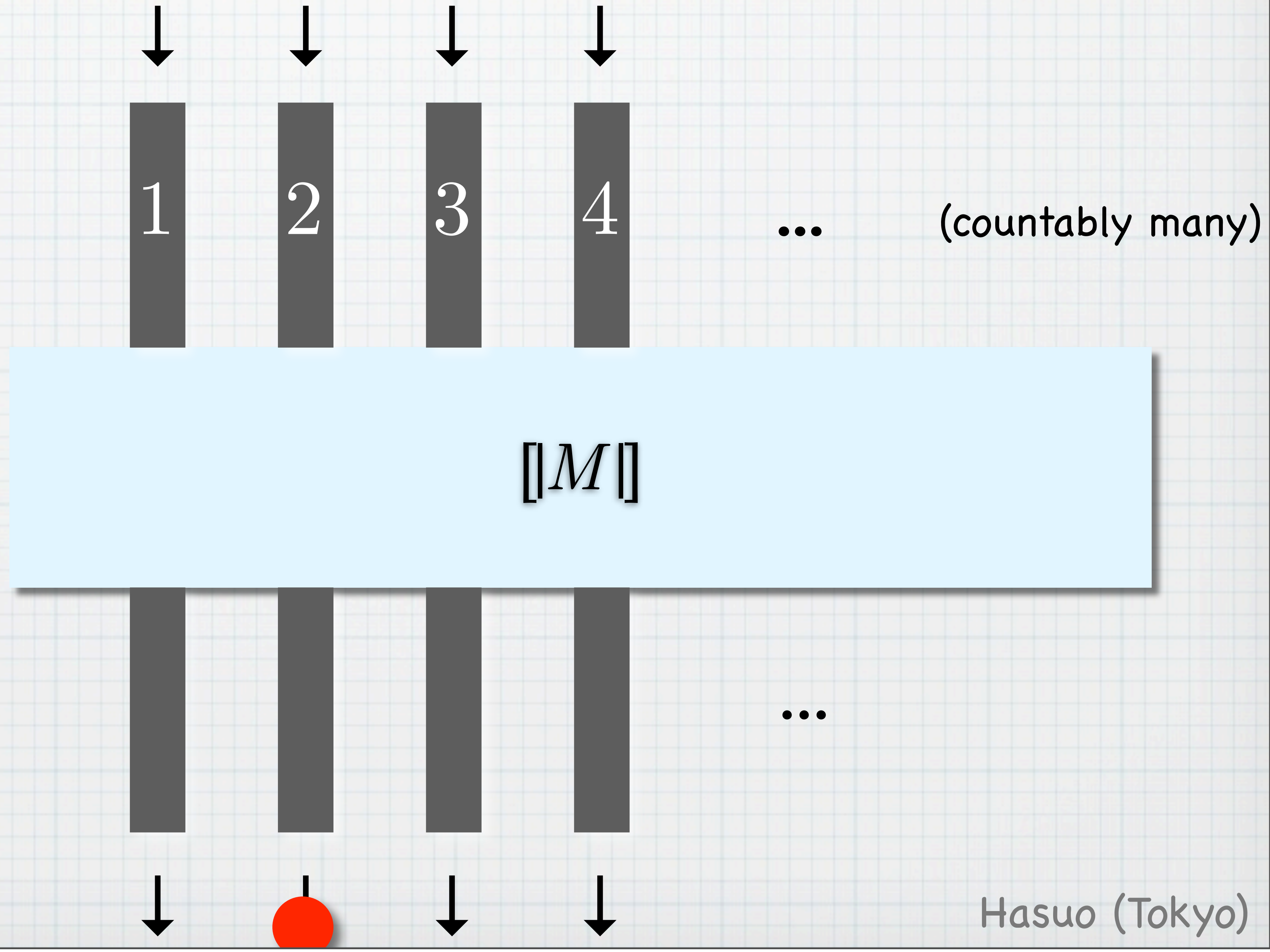


(countably many)

The GoI Animation

$$[\![M]\!] \;=\; (\, \mathbb{N} \rightharpoonup \mathbb{N},\ \text{a partial function}\,)$$

$$=\ \text{“piping”}$$



1

2

3

4

...

(countably many)

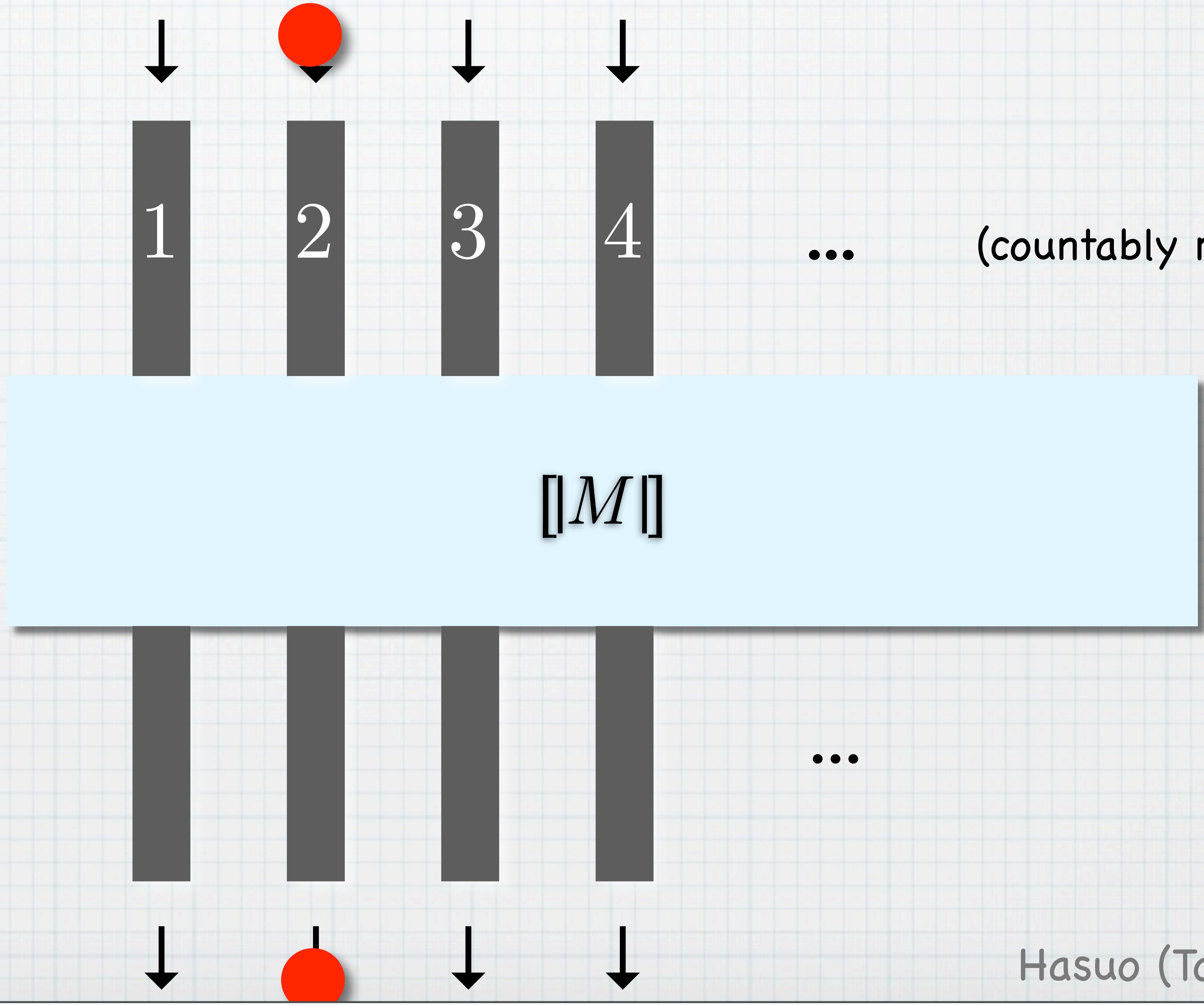
$[\![M]\!]$

...

The GoI Animation

$[\![M]\!] \;=\; (\,\mathbb{N} \rightharpoonup \mathbb{N},\;$ a partial function $)$

$= \;$ "piping"



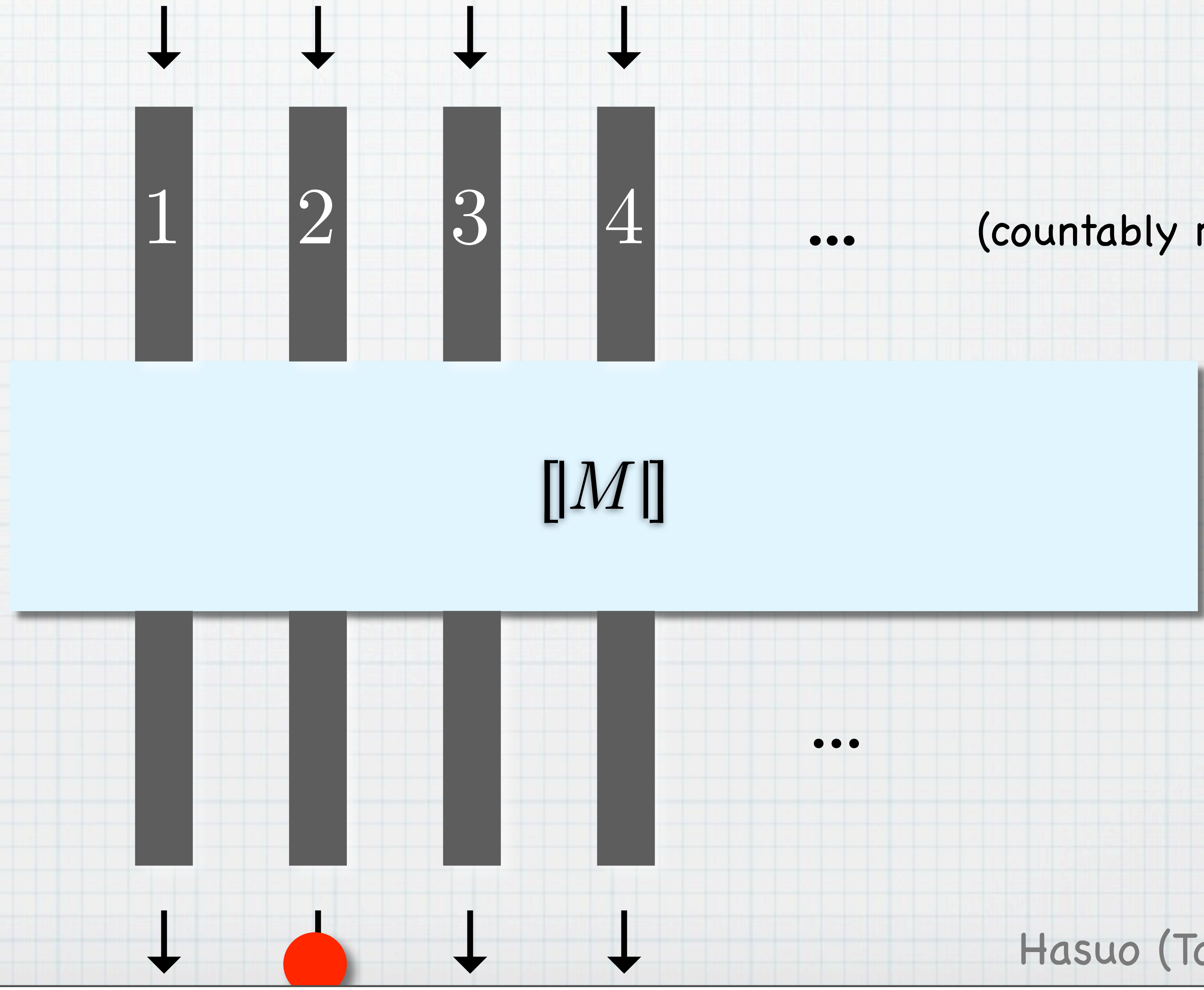
(countably many)

Hasuo (Tokyo)

The GoI Animation

$$[\![M]\!] \;=\; (\,\mathbb{N} \rightharpoonup \mathbb{N},\ \text{a partial function}\,)$$

$$=\;\text{“piping”}$$



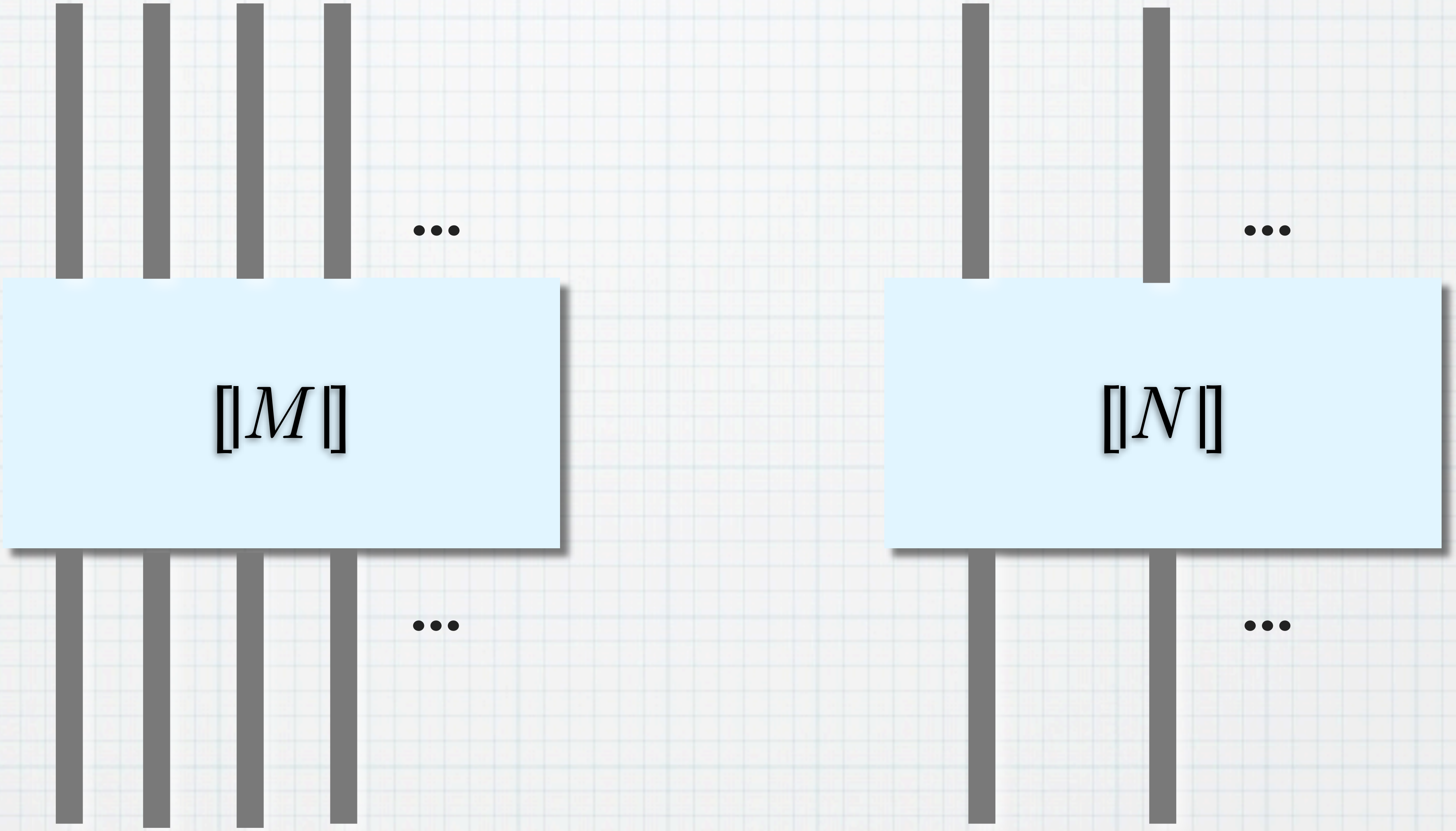
(countably many)

The GoI Animation

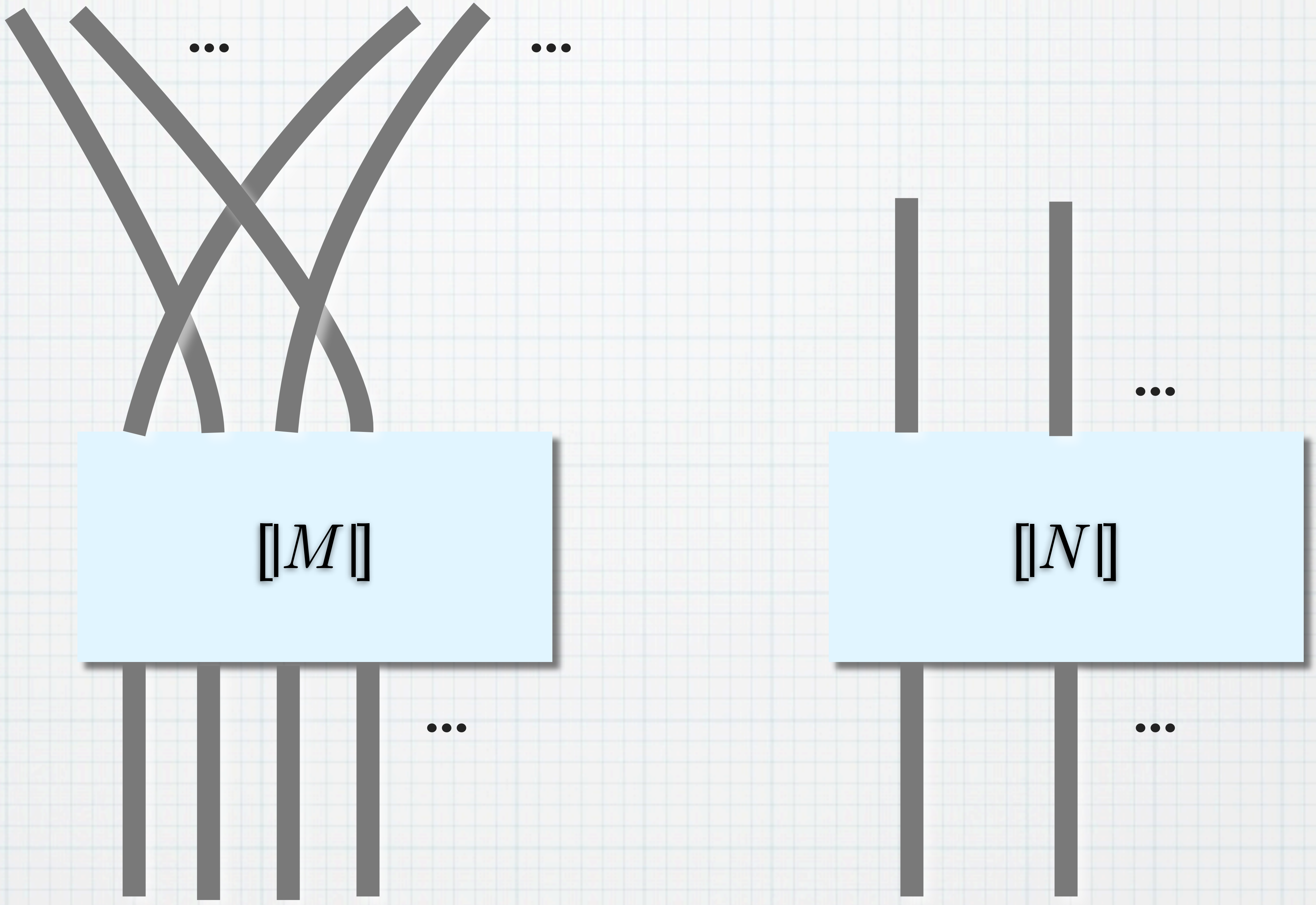
- Function application $[\![MN]\!]$
  - by “parallel composition + hiding”

Hasuo (Tokyo)

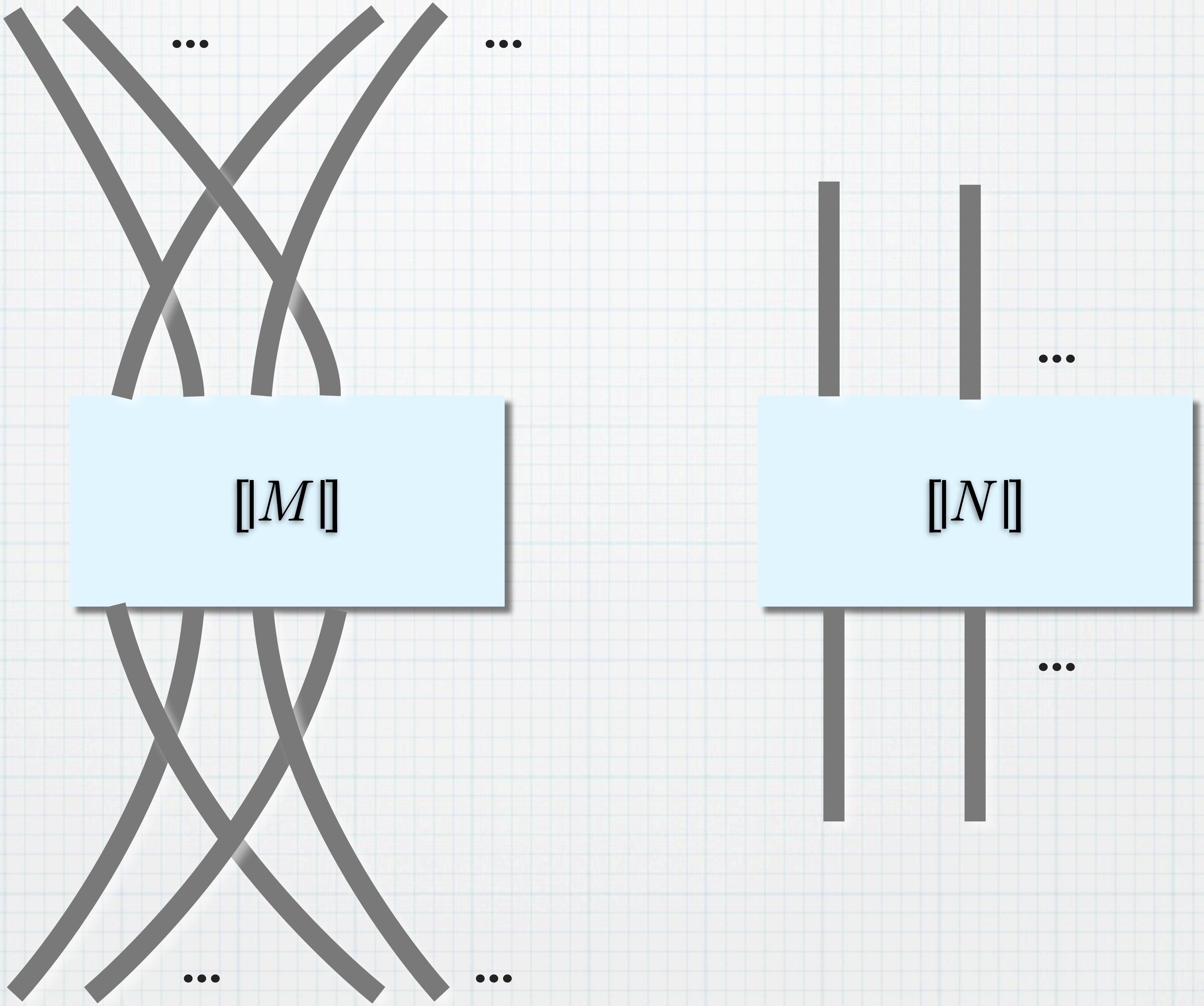
$$[\![\boldsymbol{M}\boldsymbol{N}]\!] =$$



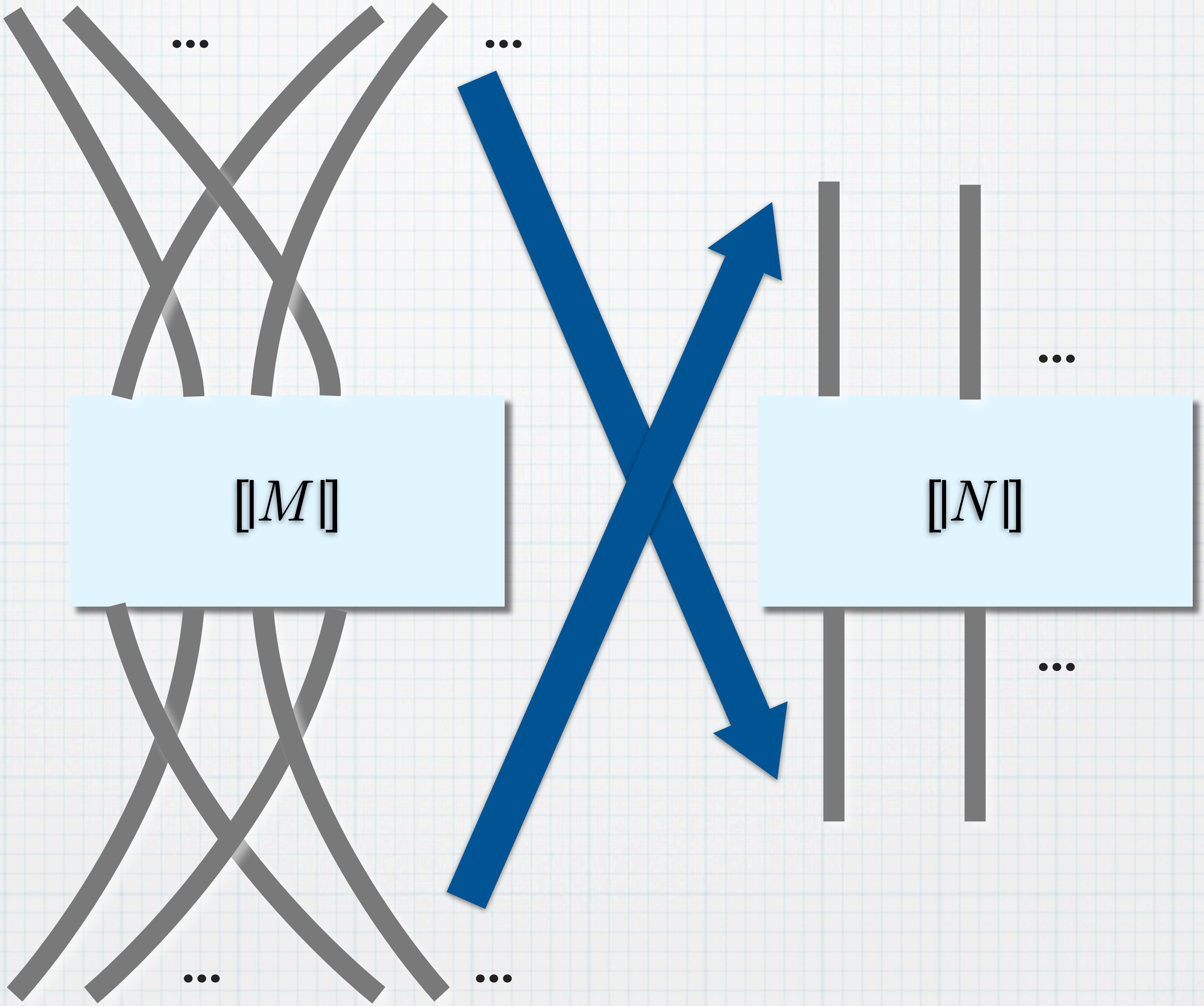
$$[\![\boldsymbol{M N}]\!] =$$



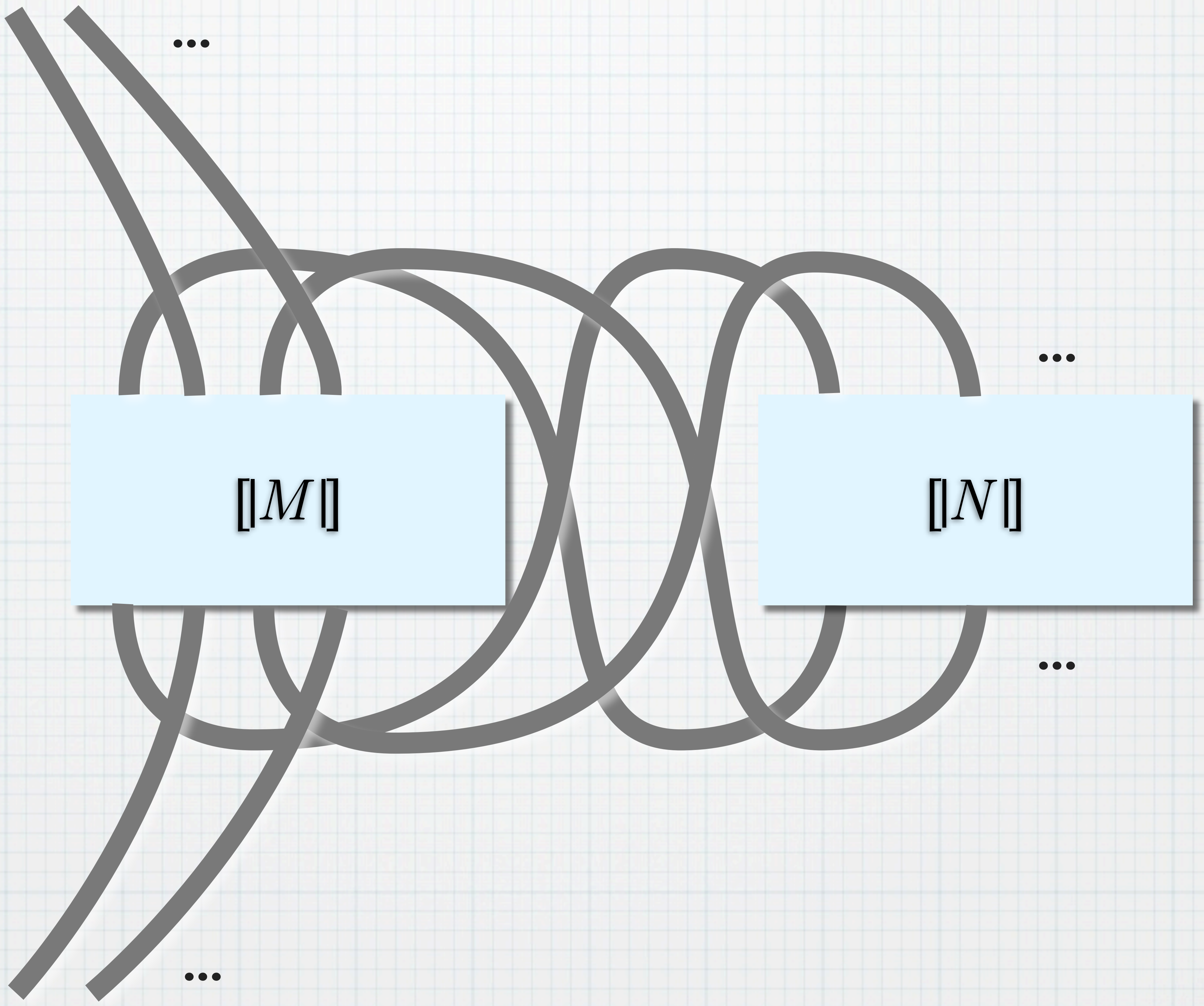
$$[\![\boldsymbol{M}\boldsymbol{N}]\!] =$$



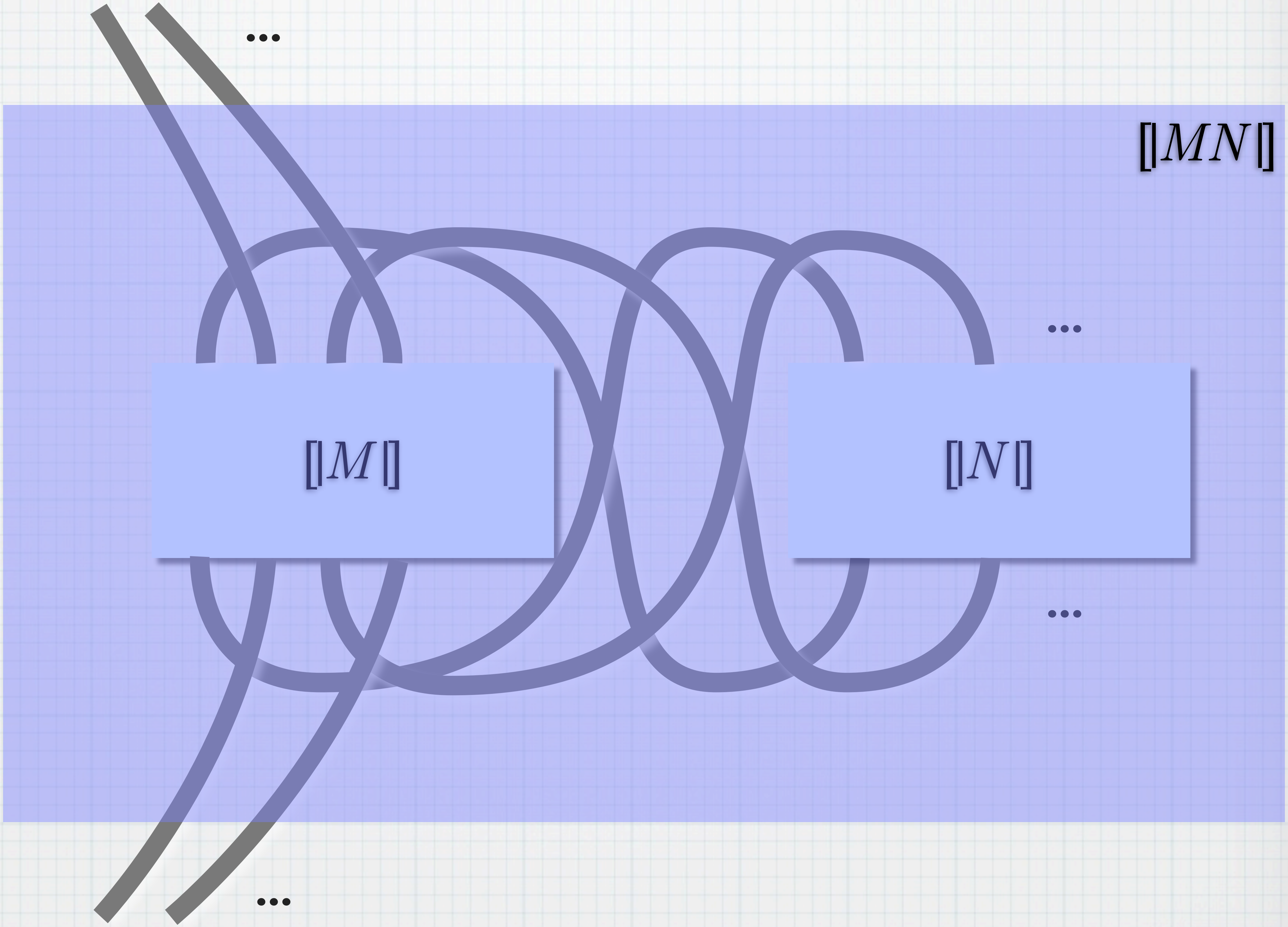
$$\llbracket \boldsymbol{M}\boldsymbol{N} \rrbracket =$$



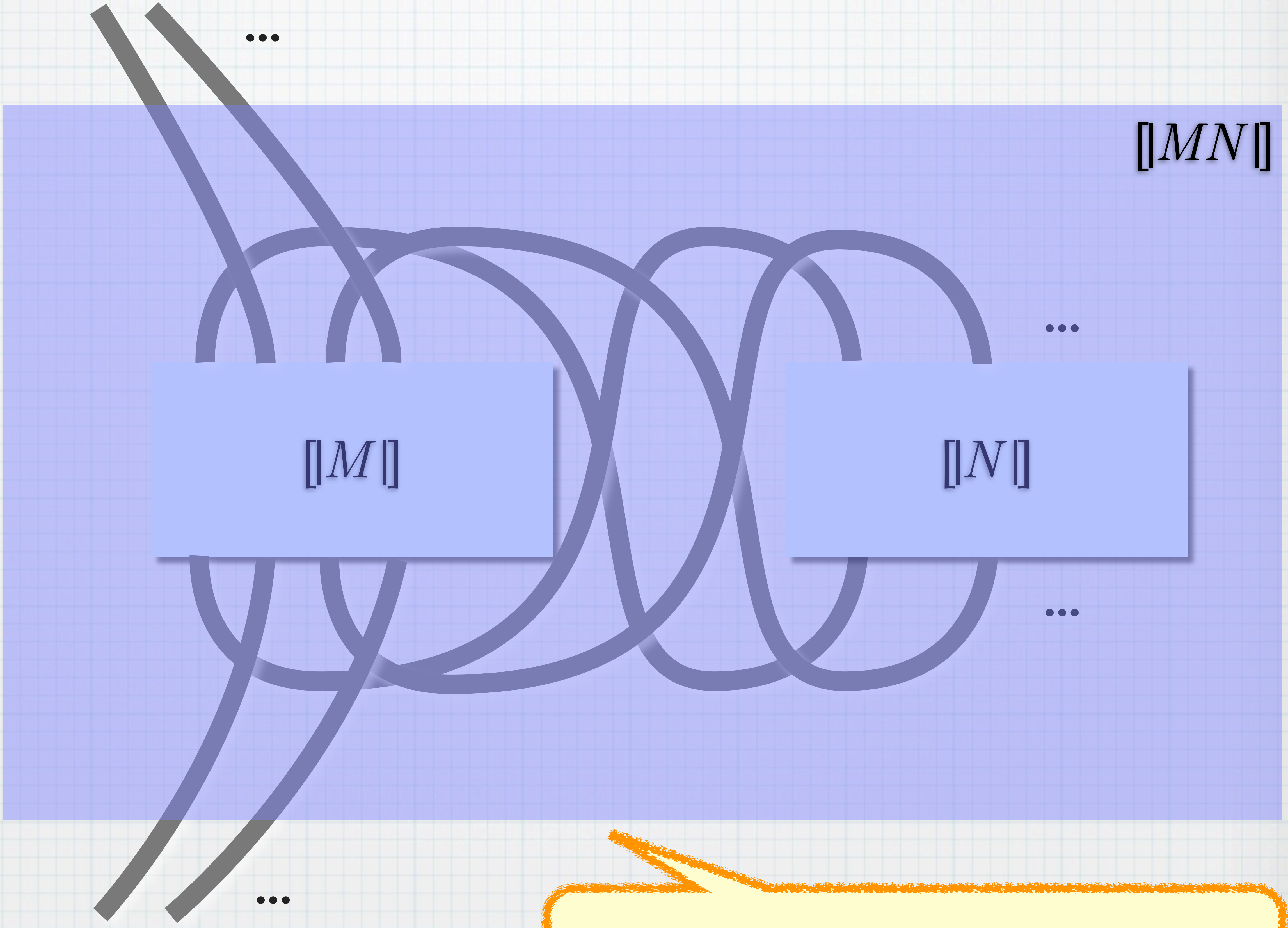
$$[\![\boldsymbol{M}\,\boldsymbol{N}]\!] =$$



$$[\![\boldsymbol{M}\boldsymbol{N}]\!] =$$

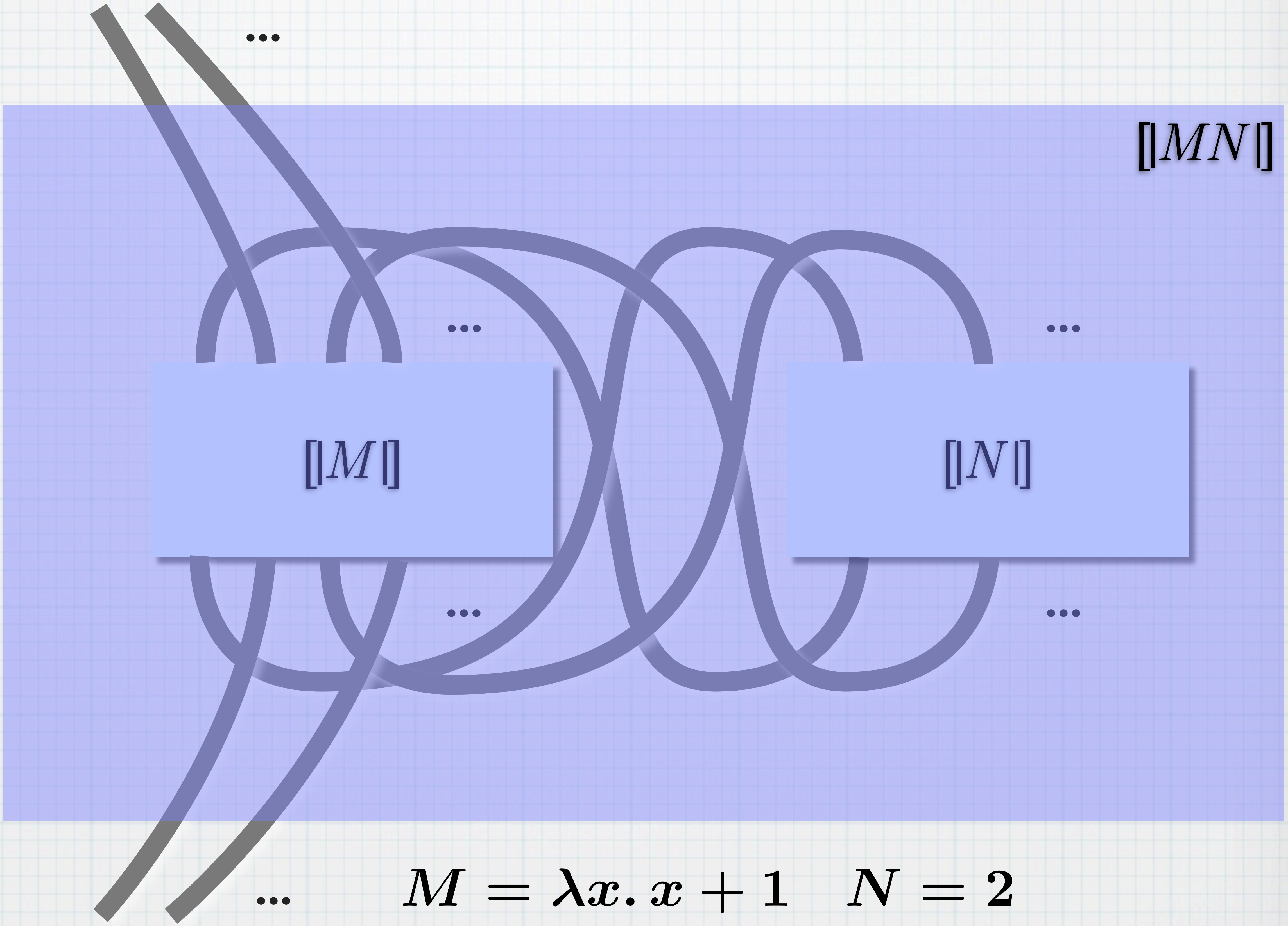


$$[\![MN]\!] =$$



"parallel composition + hiding"
(cf. AJM games)

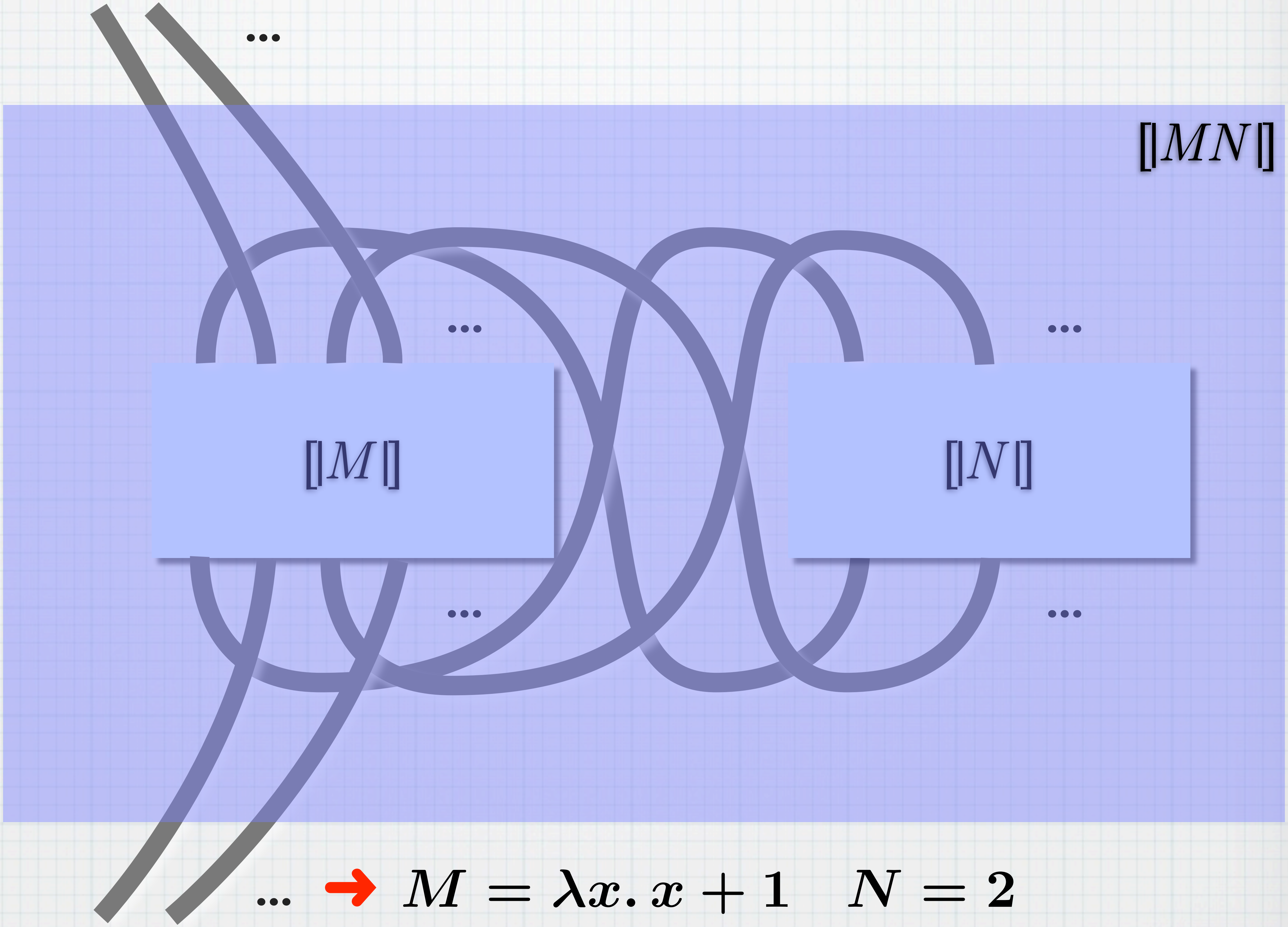
$$[\![\boldsymbol{MN}]\!] =$$



$$\boldsymbol{M = \lambda x.\, x + 1} \qquad \boldsymbol{N = 2}$$
$$\boldsymbol{M = \lambda x.\, 1} \qquad \boldsymbol{N = 2}$$
$$\boldsymbol{M = \lambda f.\, f 1} \qquad \boldsymbol{N = \lambda x.\, (x + 1)}$$

$[\![MN]\!]$

$=$



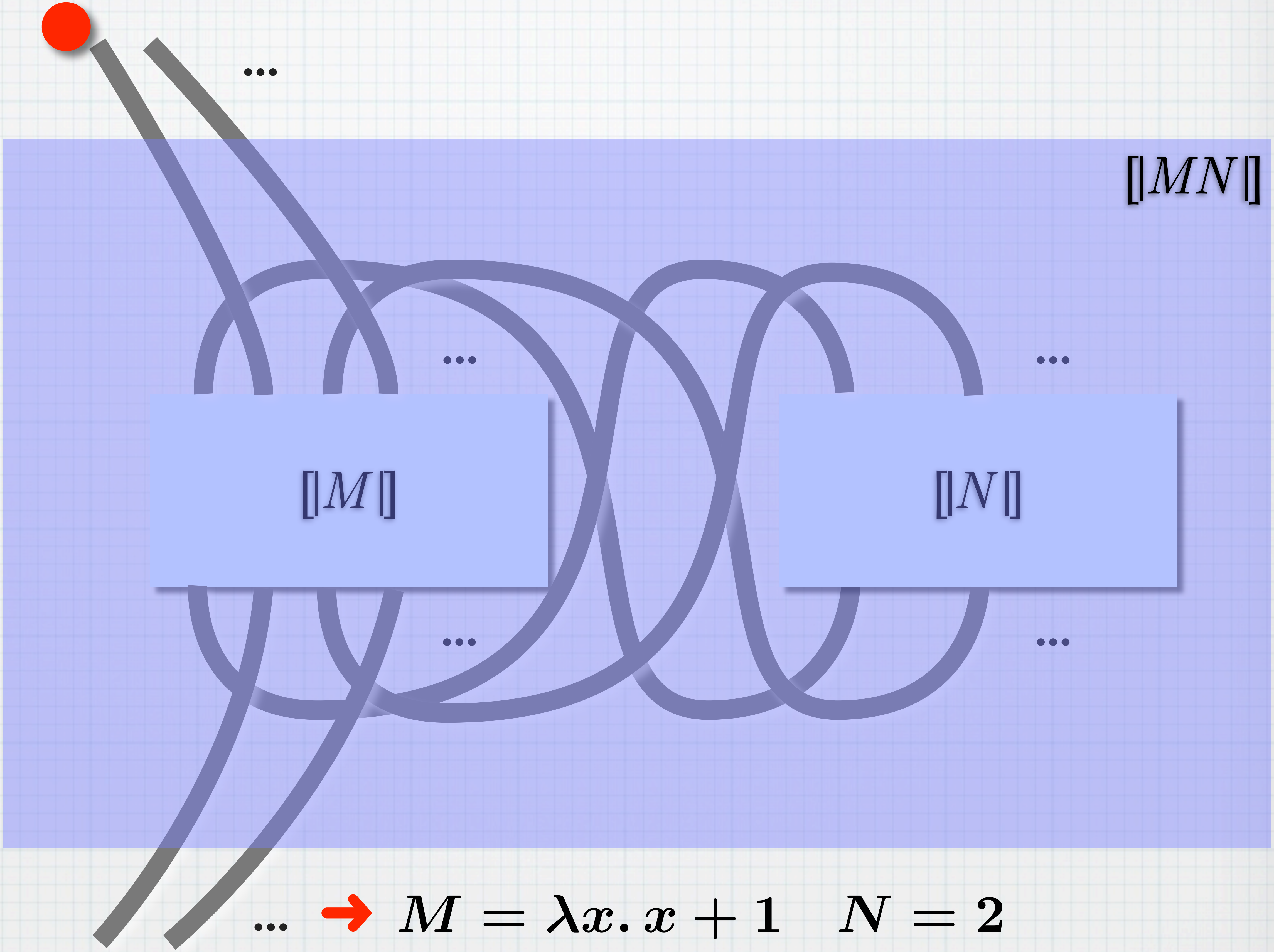
➜ $\boldsymbol{M = \lambda x.\, x + 1} \quad \boldsymbol{N = 2}$

$\boldsymbol{M = \lambda x.\, 1} \quad \boldsymbol{N = 2}$

$\boldsymbol{M = \lambda f.\, f 1} \quad \boldsymbol{N = \lambda x.\, (x + 1)}$

$[\![MN]\!]$

$=$

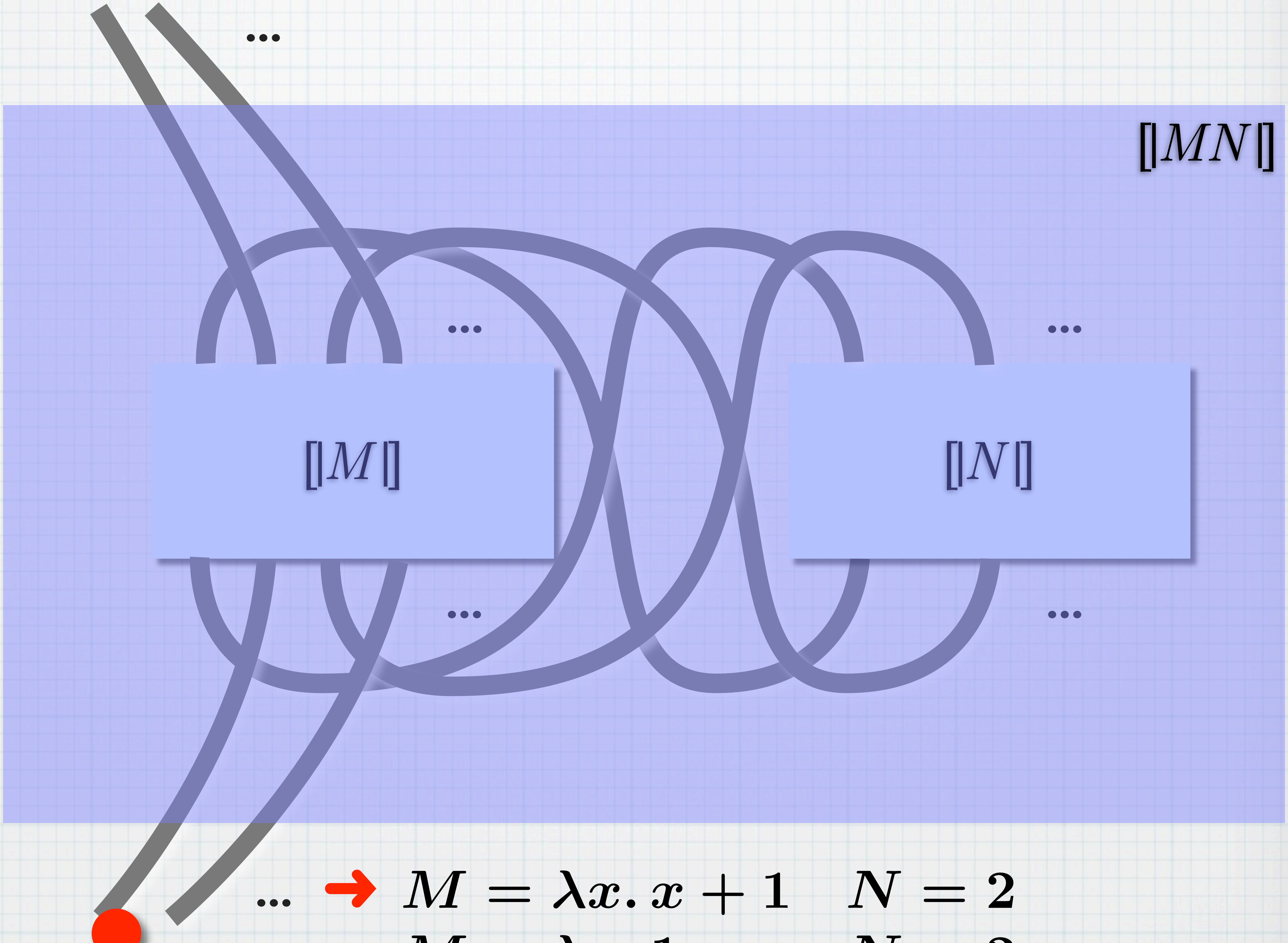


➜ $M = \lambda x.\, x + 1 \qquad N = 2$

$M = \lambda x.\, 1 \qquad N = 2$

$M = \lambda f.\, f 1 \qquad N = \lambda x.\, (x + 1)$

$[\![MN]\!] =$

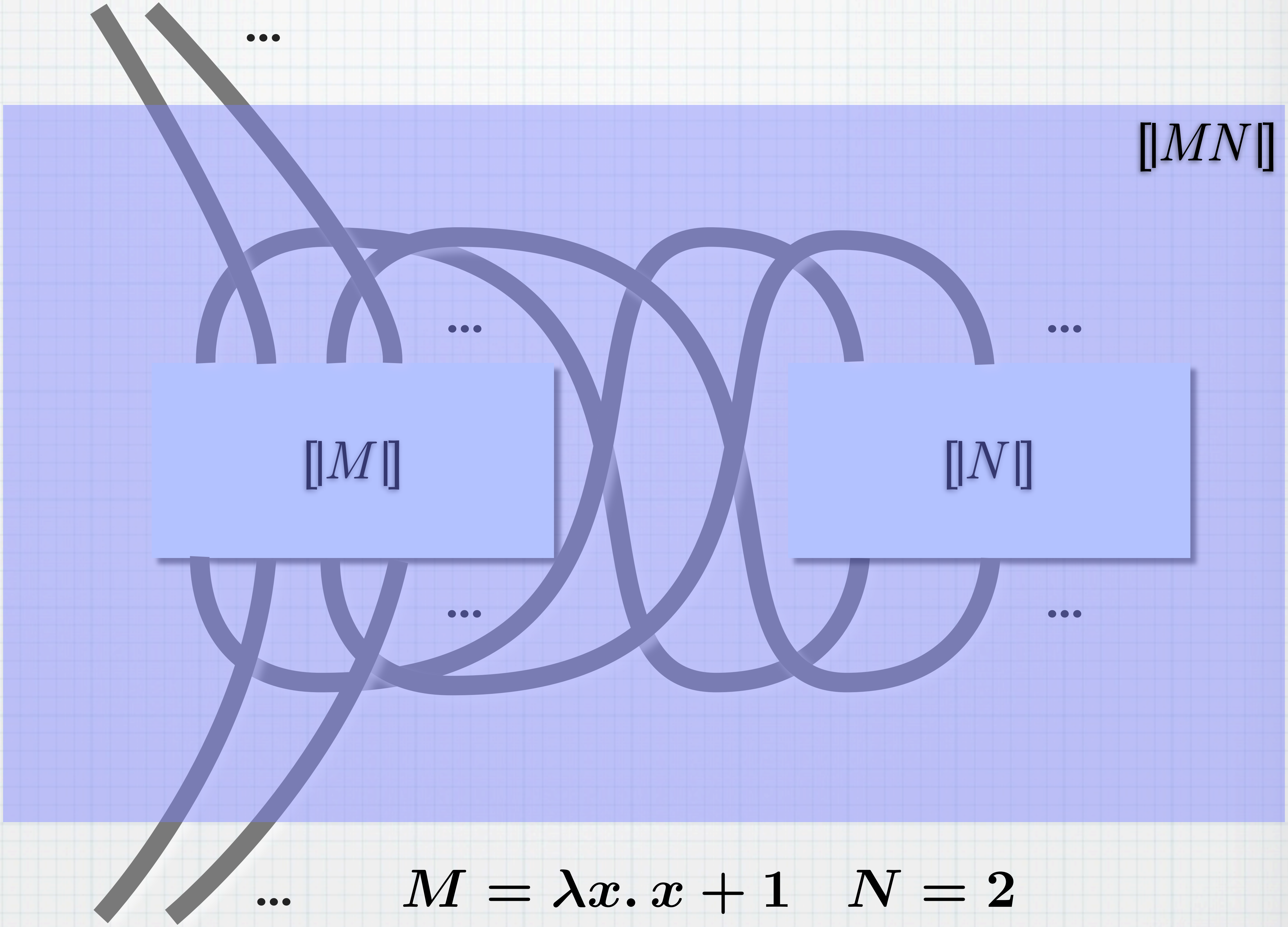


→ $M = \lambda x.\, x + 1 \quad N = 2$

$M = \lambda x.\, 1 \quad N = 2$

$M = \lambda f.\, f 1 \quad N = \lambda x.\, (x + 1)$

$$[\![\boldsymbol{MN}]\!] =$$

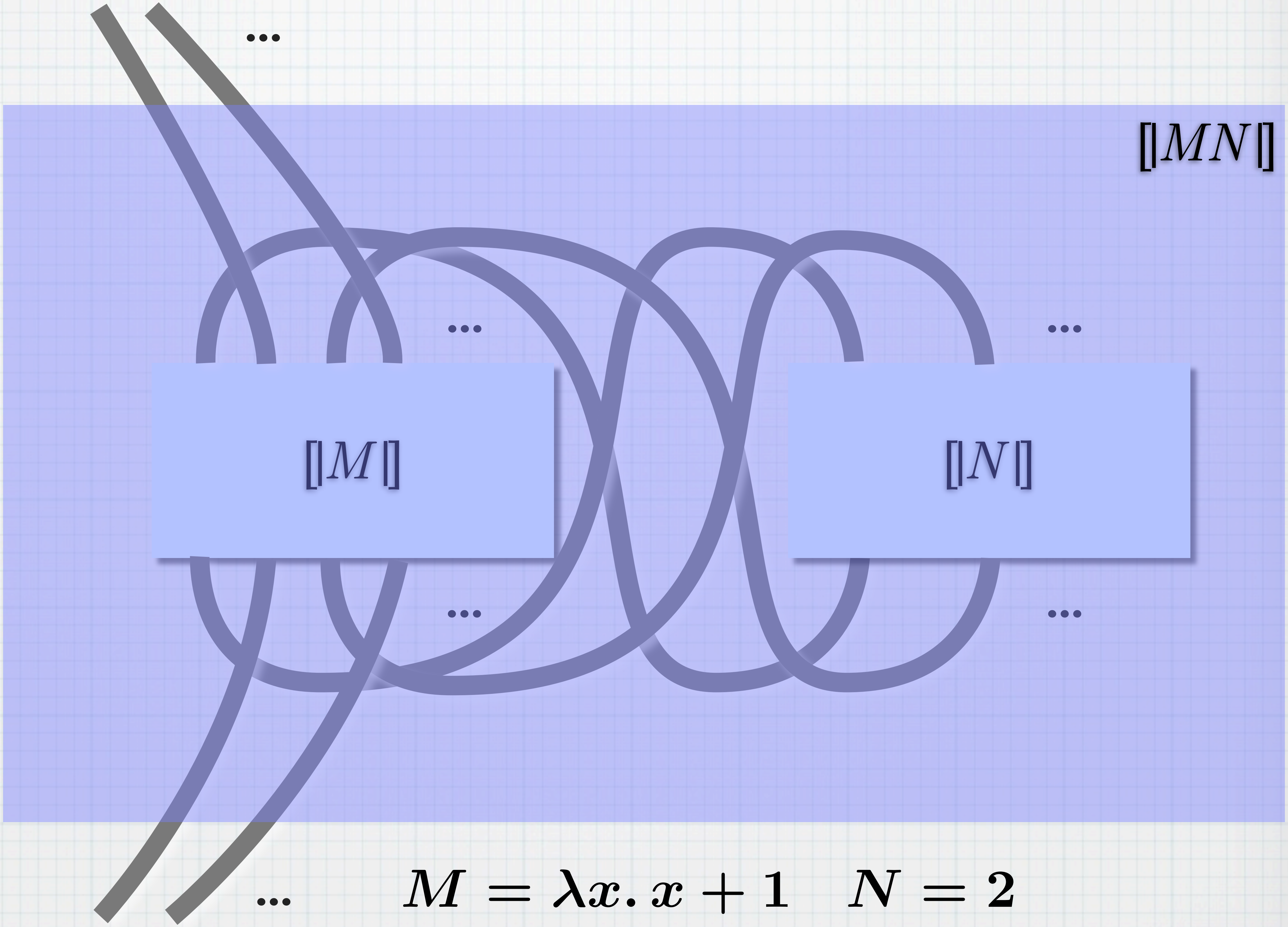


$$\boldsymbol{M = \lambda x.\, x + 1 \quad N = 2}$$

$$\boldsymbol{M = \lambda x.\, 1 \quad N = 2}$$

$$\boldsymbol{M = \lambda f.\, f1 \quad N = \lambda x.\, (x + 1)}$$

$[\![MN]\!] =$



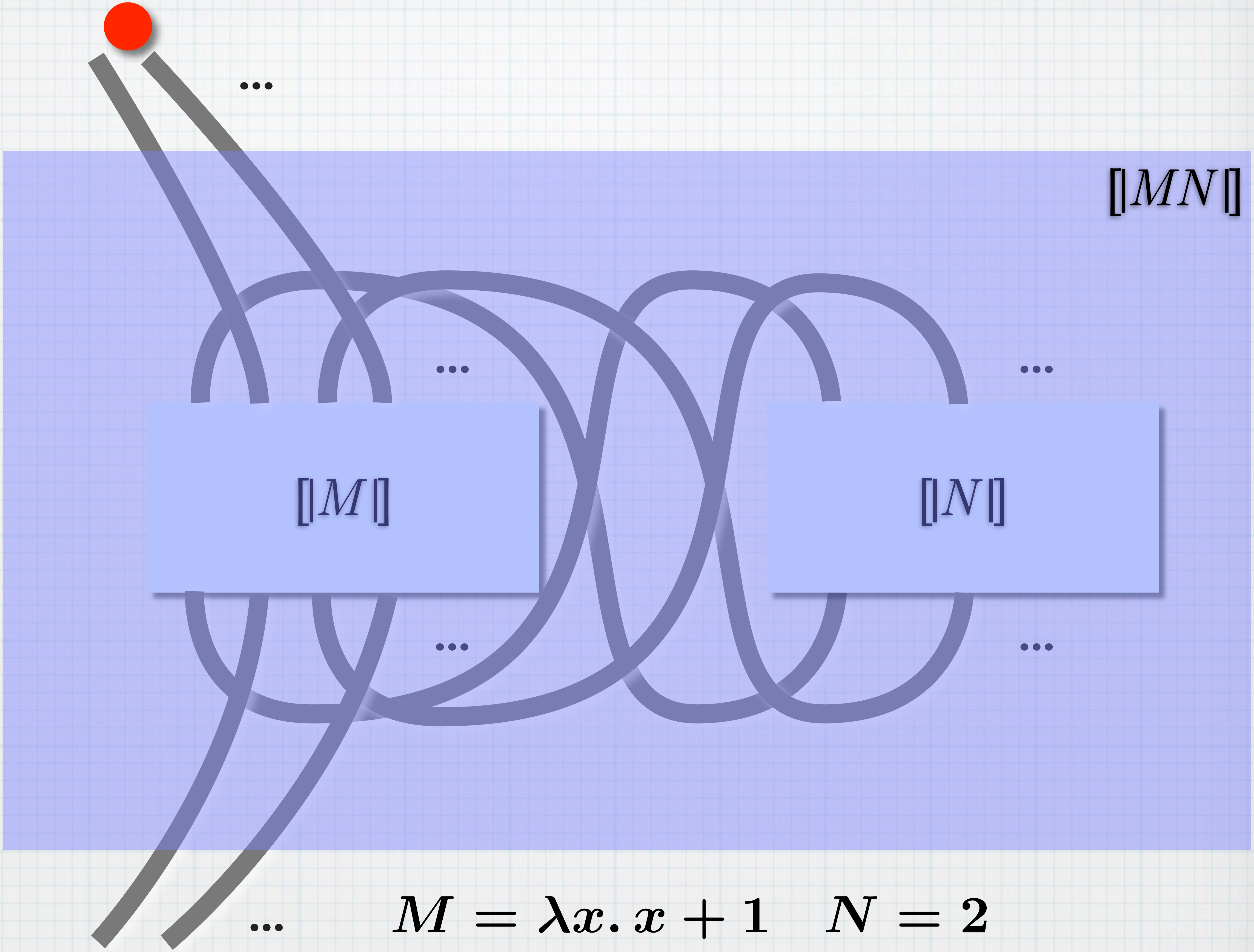
$\boldsymbol{M = \lambda x.\, x + 1} \quad \boldsymbol{N = 2}$

➜ $\boldsymbol{M = \lambda x.\, 1} \quad \boldsymbol{N = 2}$

$\boldsymbol{M = \lambda f.\, f 1} \quad \boldsymbol{N = \lambda x.\, (x + 1)}$

$[\![MN]\!]$

$=$



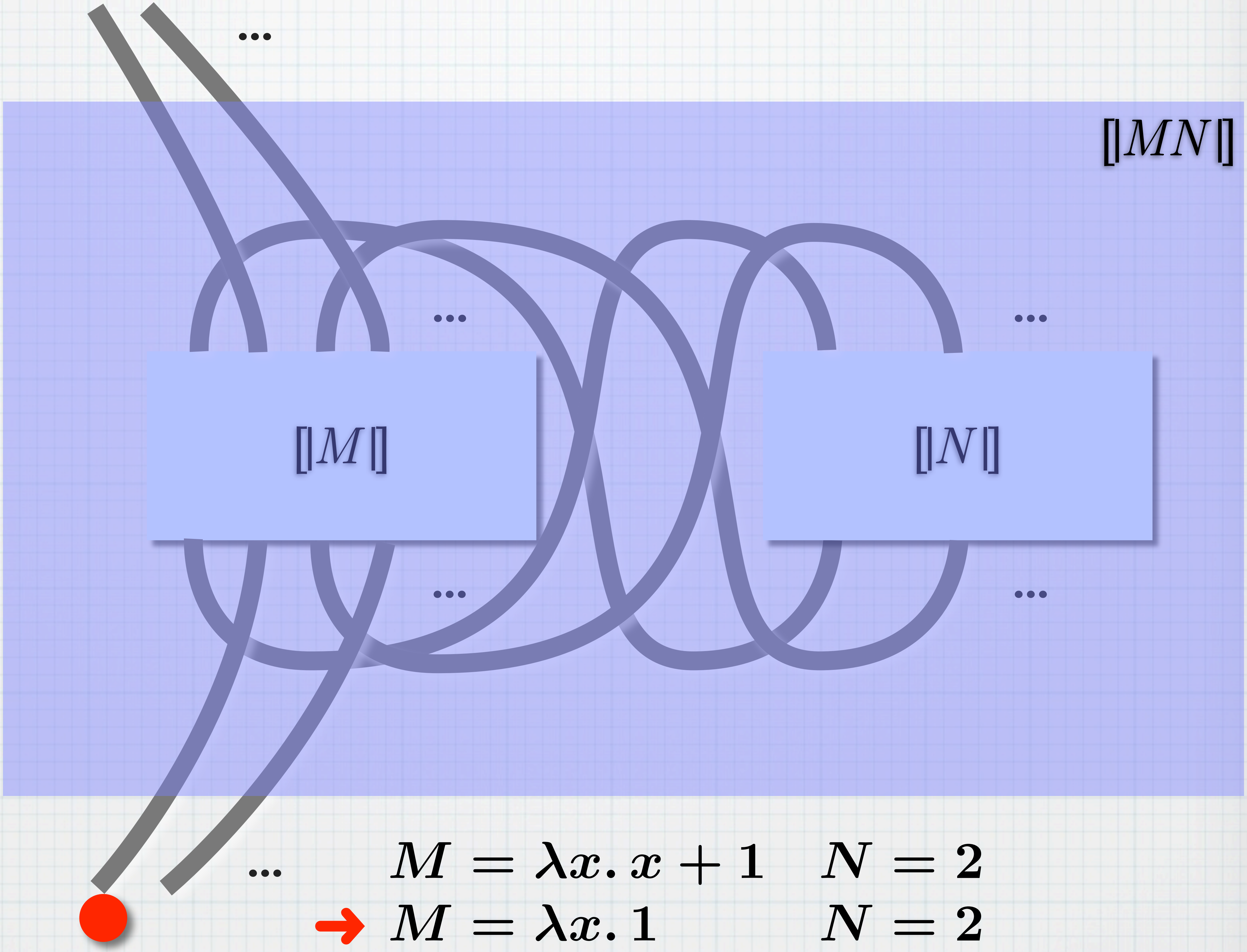
$\boldsymbol{M = \lambda x.\, x + 1} \quad \boldsymbol{N = 2}$

→ $\boldsymbol{M = \lambda x.\, 1} \quad \boldsymbol{N = 2}$

$\boldsymbol{M = \lambda f.\, f 1} \quad \boldsymbol{N = \lambda x.\, (x + 1)}$

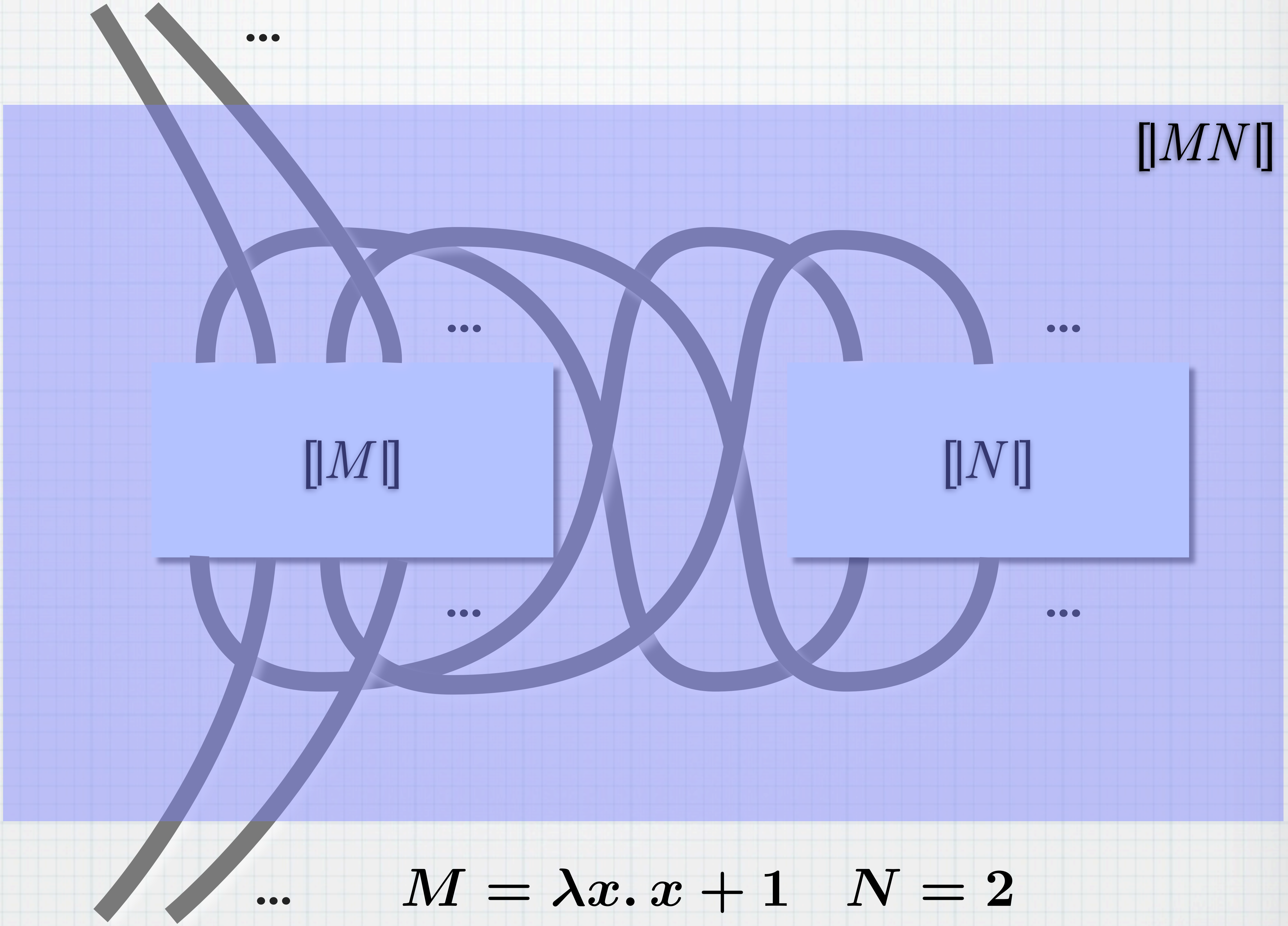
$[\![MN]\!]$

$=$



$$\begin{aligned} M &= \lambda x.\, x+1 & N &= 2\\ \rightarrow M &= \lambda x.\,1 & N &= 2\\ M &= \lambda f.\, f1 & N &= \lambda x.\,(x+1)\end{aligned}$$

$$[\![\boldsymbol{MN}]\!] =$$

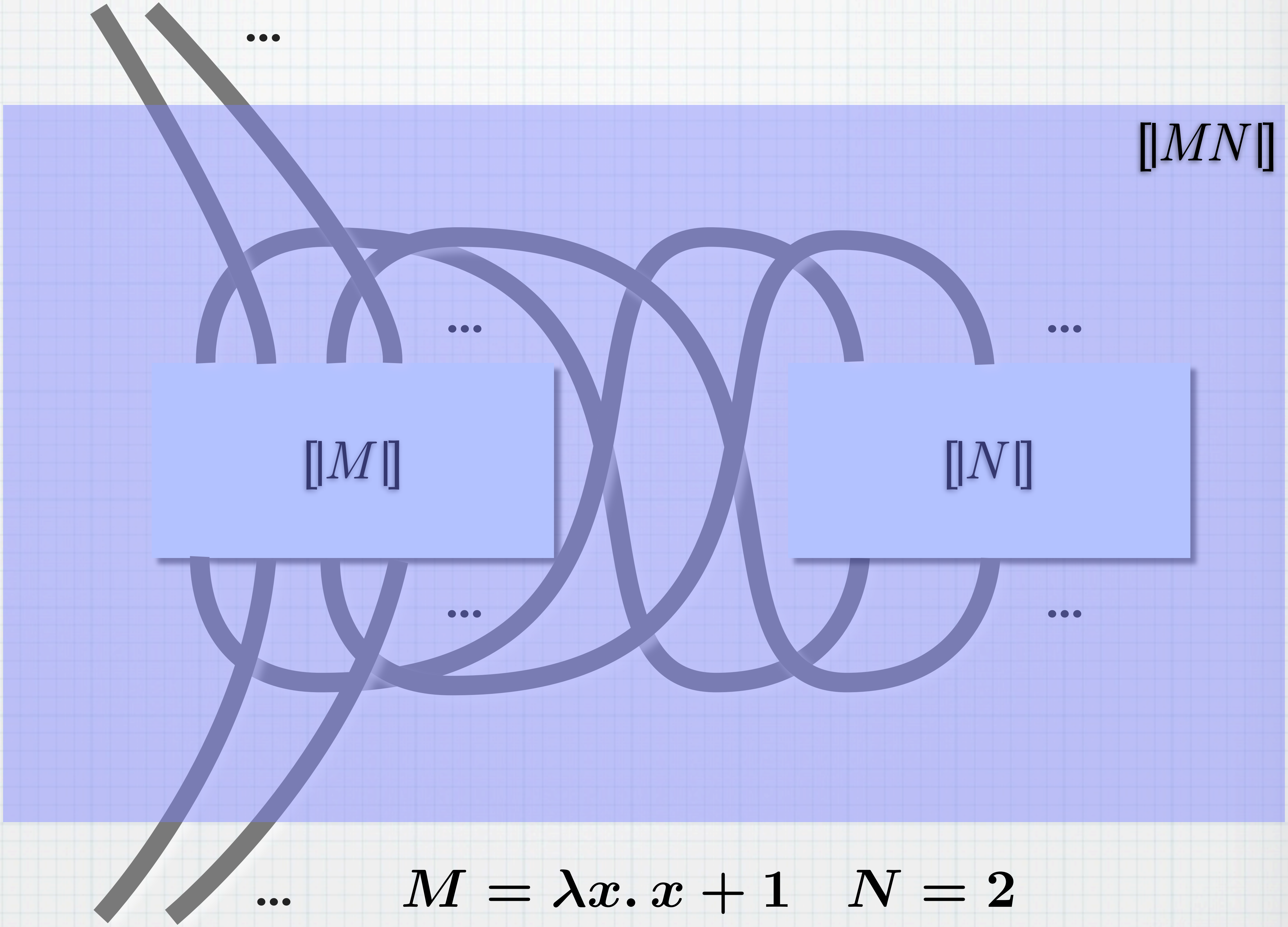


$$\boldsymbol{M = \lambda x.\, x + 1 \quad N = 2}$$

$$\boldsymbol{M = \lambda x.\, 1 \quad N = 2}$$

$$\boldsymbol{M = \lambda f.\, f 1 \quad N = \lambda x.\, (x + 1)}$$

$$[\![\boldsymbol{MN}]\!] =$$



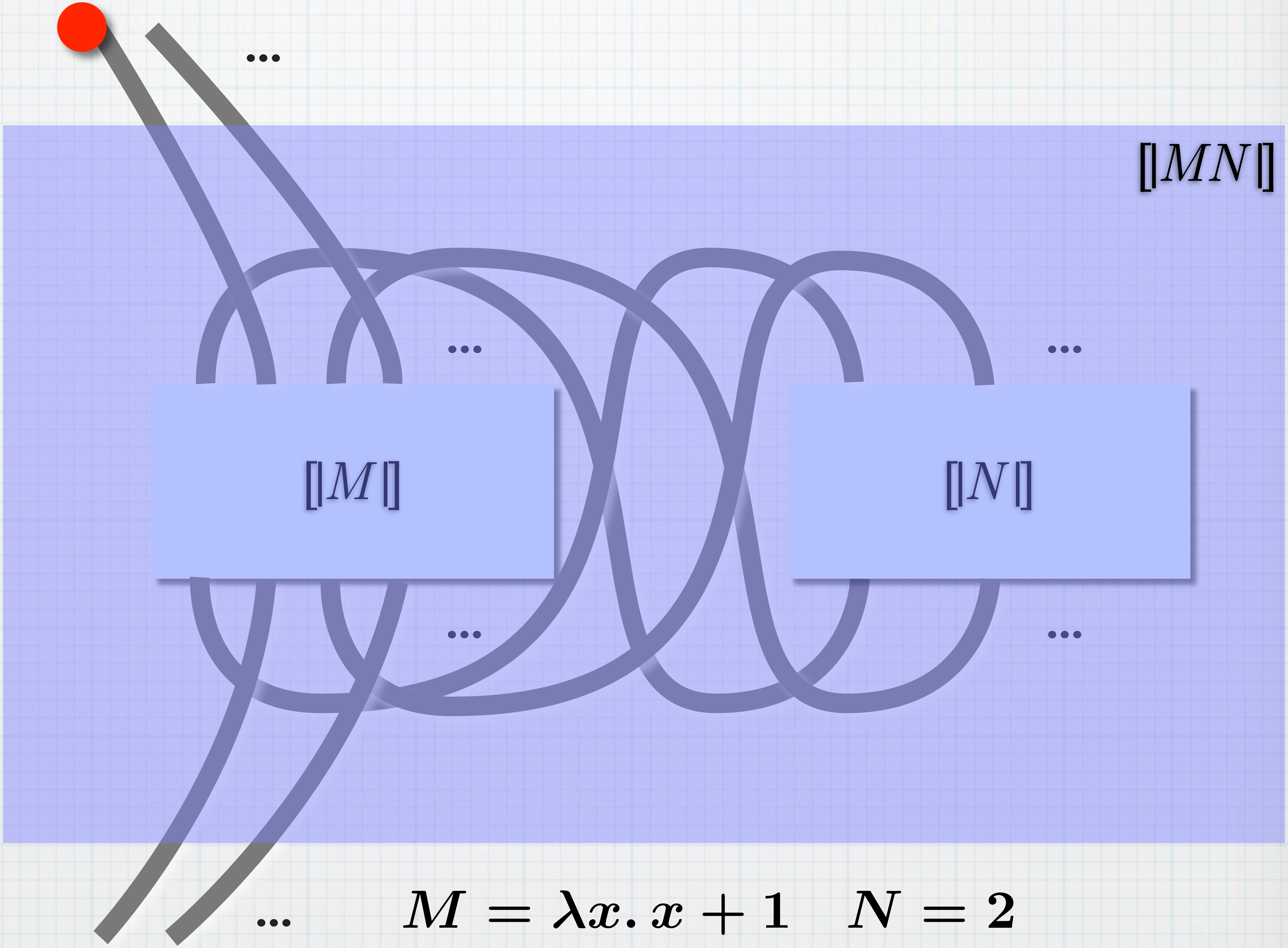
$\boldsymbol{M = \lambda x.\, x + 1} \quad \boldsymbol{N = 2}$

$\boldsymbol{M = \lambda x.\, 1} \quad \boldsymbol{N = 2}$

➜ $\boldsymbol{M = \lambda f.\, f 1} \quad \boldsymbol{N = \lambda x.\, (x + 1)}$

$[\![MN]\!]$

$=$

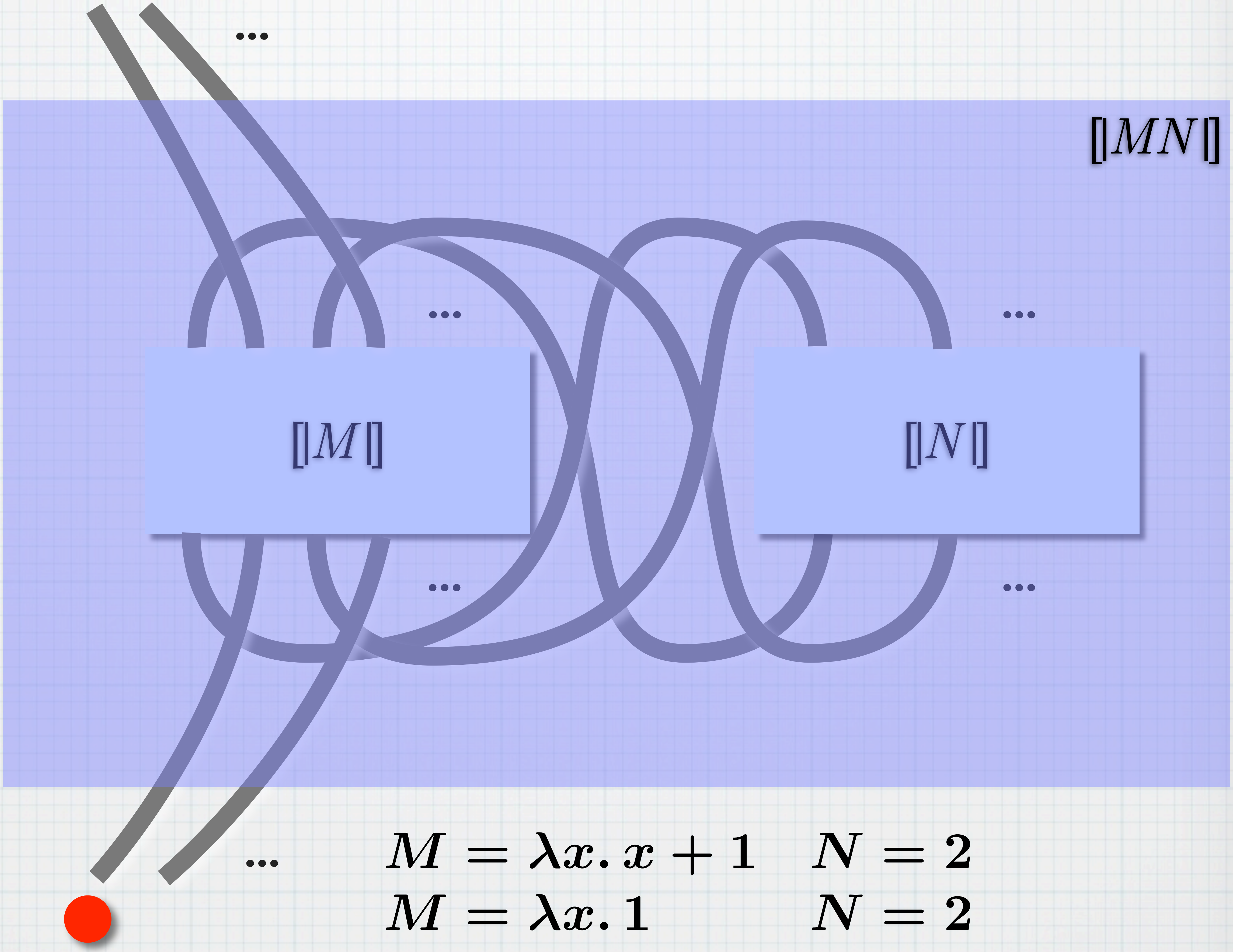


$M = \lambda x.\, x + 1 \quad N = 2$

$M = \lambda x.\, 1 \quad N = 2$

➜ $M = \lambda f.\, f1 \quad N = \lambda x.\, (x + 1)$

$[\![MN]\!] =$

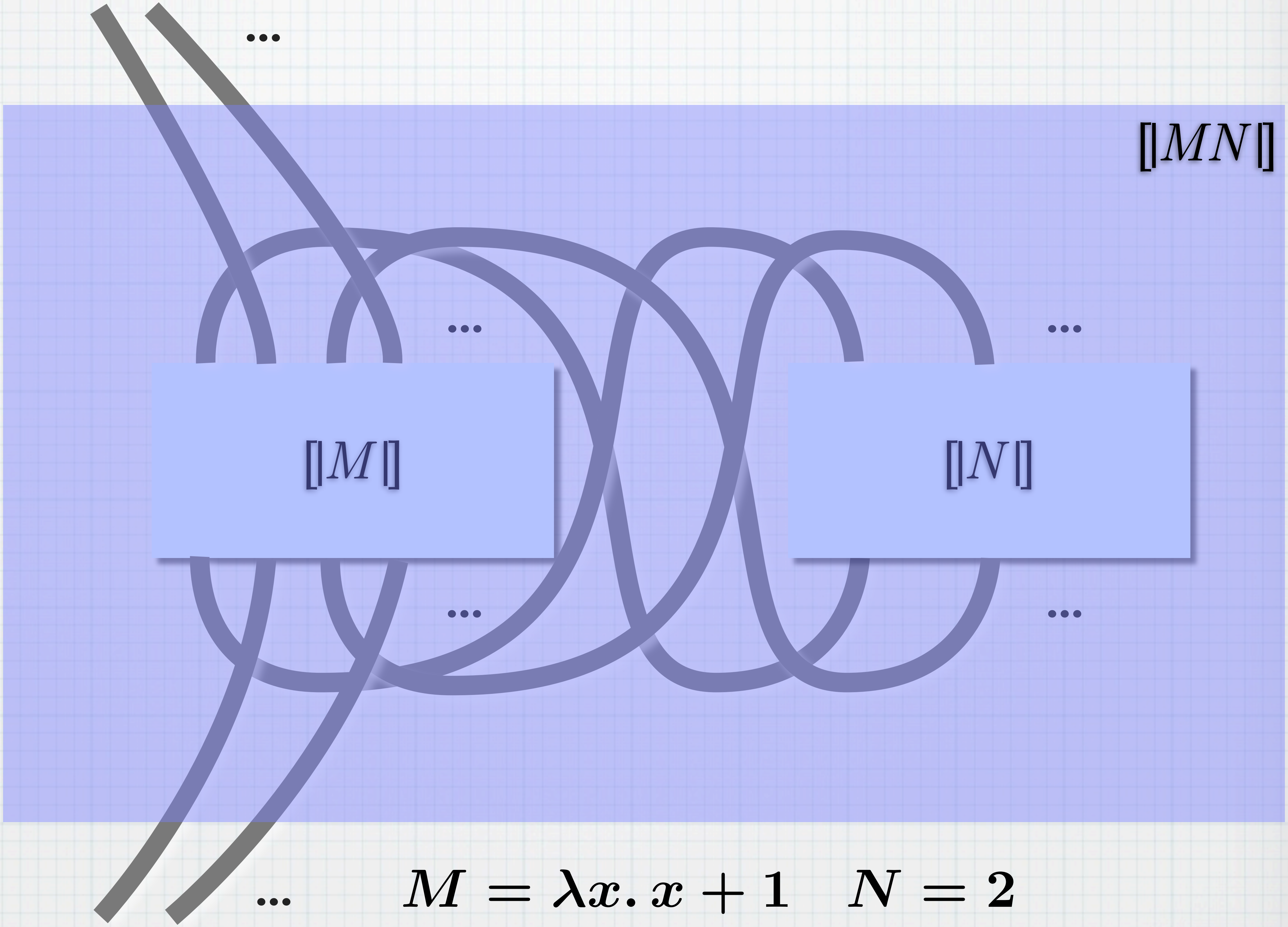


$M = \lambda x.\, x + 1 \quad N = 2$

$M = \lambda x.\, 1 \quad N = 2$

→ $M = \lambda f.\, f 1 \quad N = \lambda x.\, (x + 1)$

$$[\![MN]\!] =$$

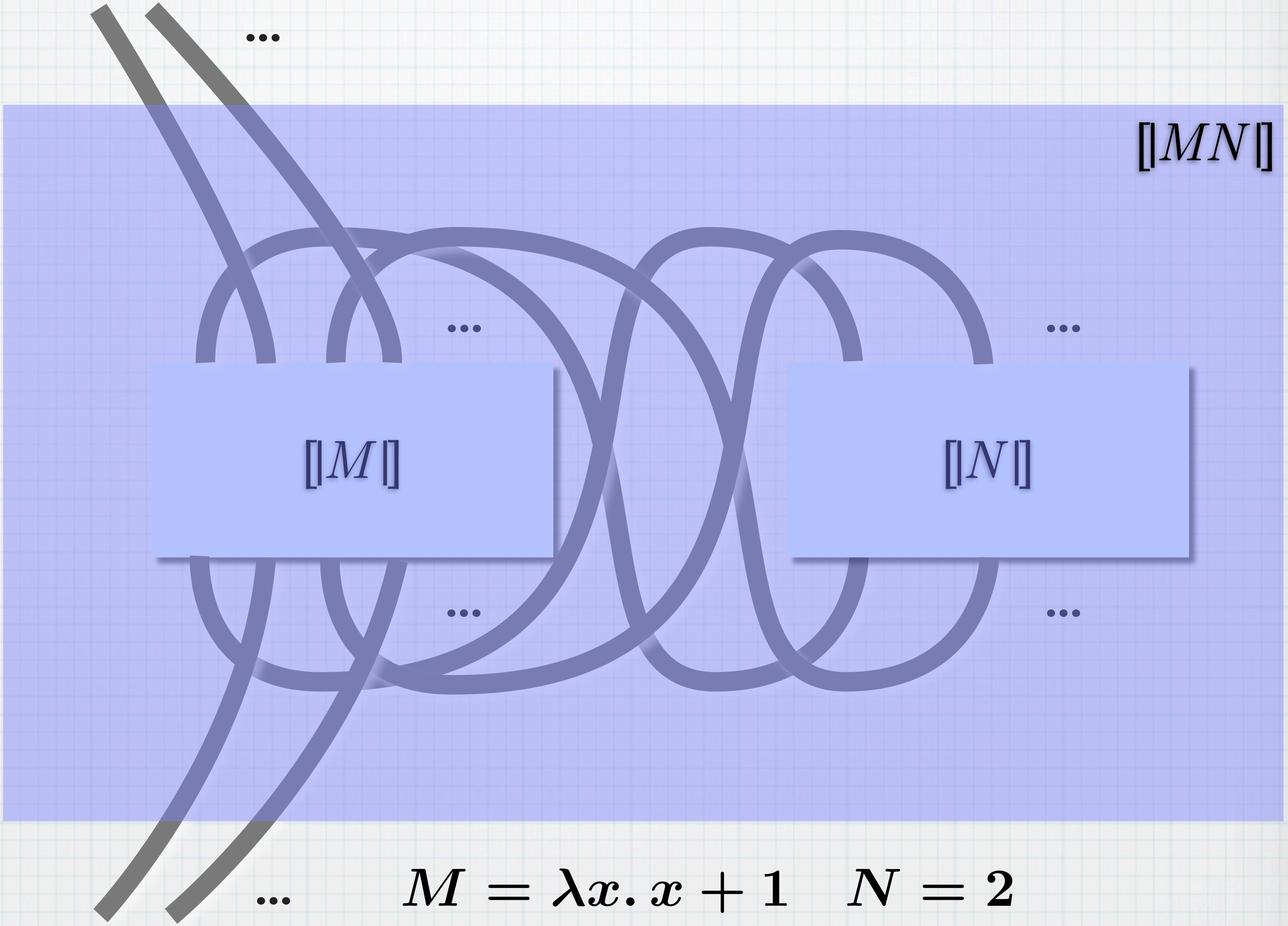


$M = \lambda x.\, x + 1 \quad N = 2$

$M = \lambda x.\, 1 \quad N = 2$

➜ $M = \lambda f.\, f 1 \quad N = \lambda x.\, (x + 1)$

$$\llbracket \boldsymbol{MN} \rrbracket =$$



$\llbracket MN \rrbracket$
$\llbracket M \rrbracket$
$\llbracket N \rrbracket$
...
...
...
...
...
...

$$\boldsymbol{M = \lambda x.\, x+1 \qquad N = 2}$$
$$\boldsymbol{M = \lambda x.\, 1 \qquad N = 2}$$
$$\boldsymbol{M = \lambda f.\, f1 \qquad N = \lambda x.\, (x+1)}$$

GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88

GoI: Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M

GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
  - Similar to **game semantics** [AJM/HO]

GoI: Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
  - Similar to **game semantics** [AJM/HO]
  - Linearity: simplicity; no-cloning

GoI: Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
  - Similar to **game semantics** [AJM/HO]
  - Linearity: simplicity; no-cloning
  - Girard translation

$A \to B$ as $!A \multimap B$

Hasuo (Tokyo)

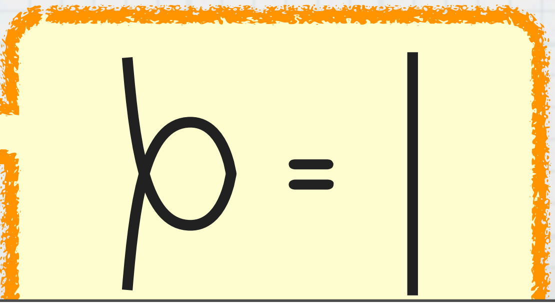
GoI:
Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
  - Similar to **game semantics** [AJM/HO]
  - Linearity: simplicity; no-cloning
  - Girard translation

$A \to B$

as $!A \multimap B$

- "Geometry": invariant under β-reductions



(Tokyo)

GoI:
Geometry of

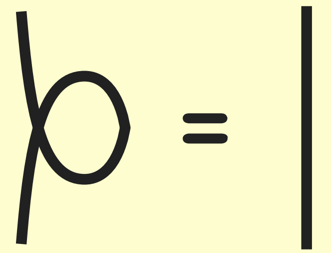
- **C*-algebra** presentation
- **Token machine** presentation

- J.-Y. Girard, at Logic Colloquium '88
- Provides denotational semantics $[\![M]\!]$ for linear λ-term M
  - Similar to **game semantics** [AJM/HO]
  - Linearity: simplicity; no-cloning
  - Girard translation

$A \to B$

as $!A \multimap B$

- "Geometry": invariant under β-reductions



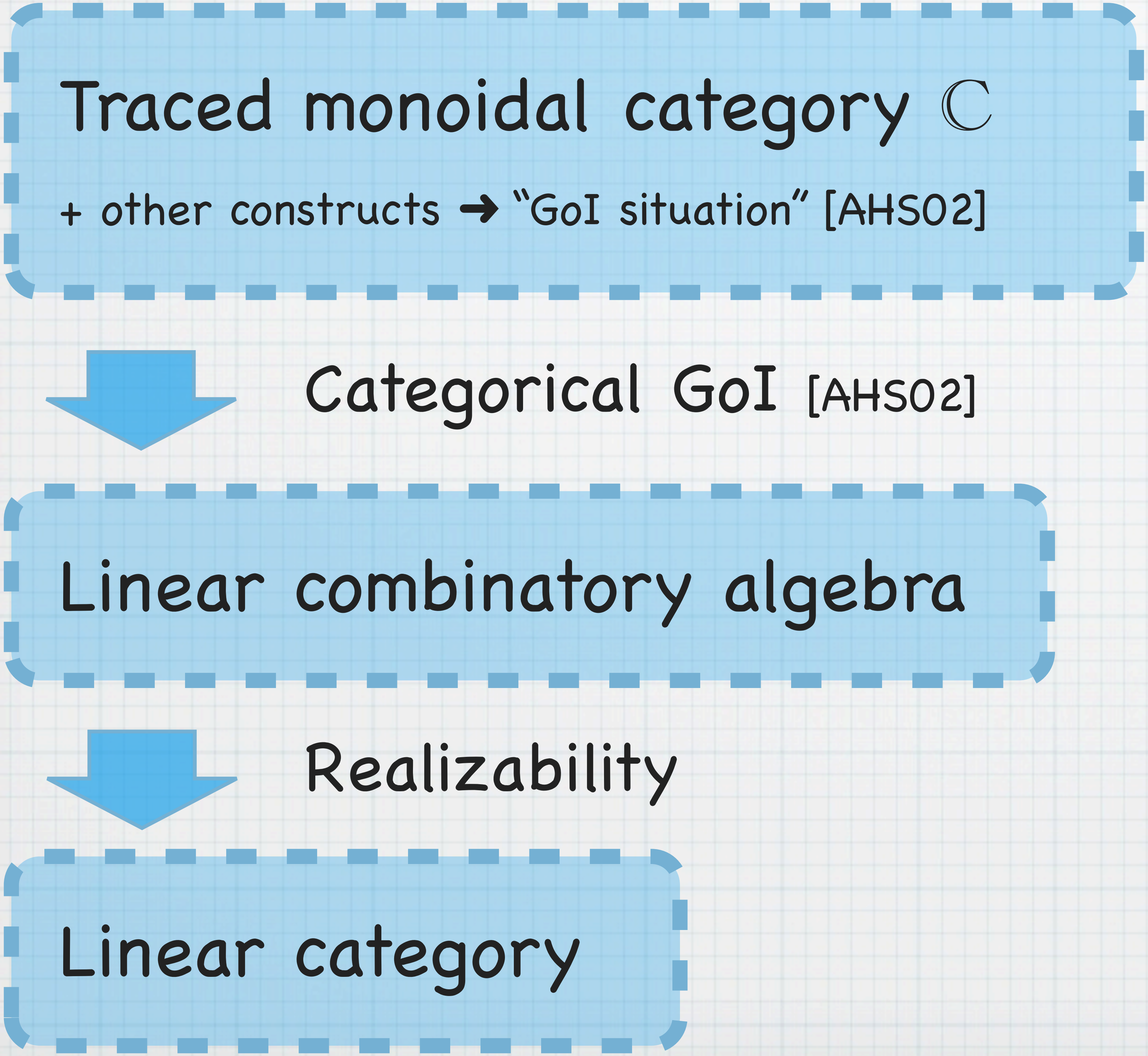
(Tokyo)

Categorical GoI

- Axiomatics of GoI in the categorical language

- Our main reference:
  - [AHS02] S. Abramsky, E. Haghverdi, and P. Scott, "Geometry of interaction and linear combinatory algebras," MSCS 2002
  - Especially its technical report version (Oxford CL), since it's a bit more detailed

The Categorical GoI Workflow



Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

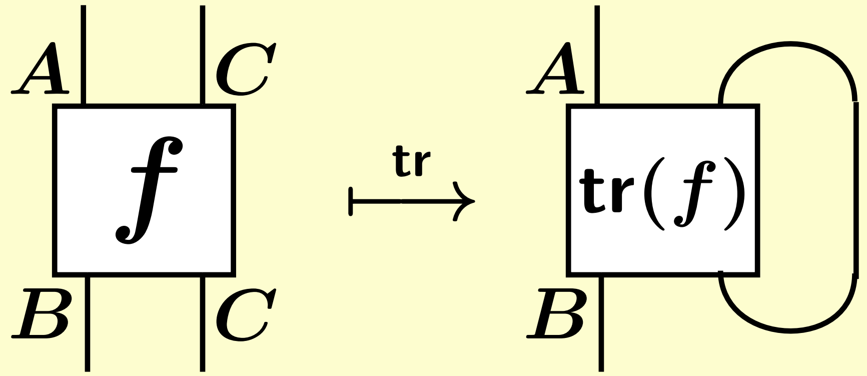
Realizability

Linear category

The Categorical GoI Workflow

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ “GoI situation” [AHS02]



⬇ Categorical GoI [AHS02]

Linear combinatory algebra

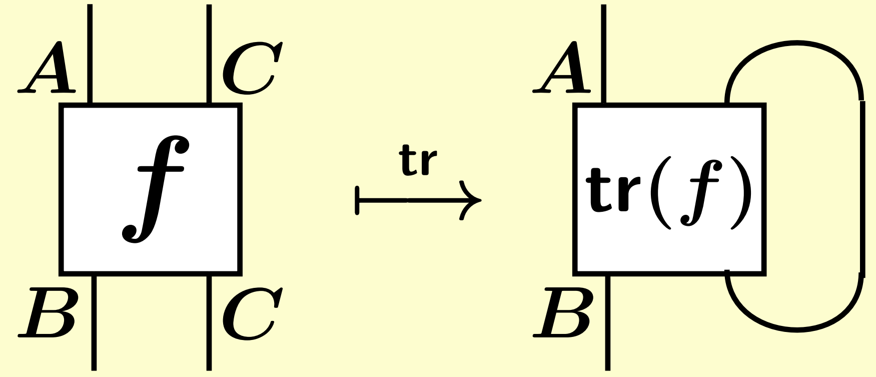
⬇ Realizability

Linear category

The Categorical GoI Workflow

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



Categorical GoI [AHS02]

Linear combinatory algebra

- ✱ Applicative str. + combinators
- ✱ Model of untyped calculus

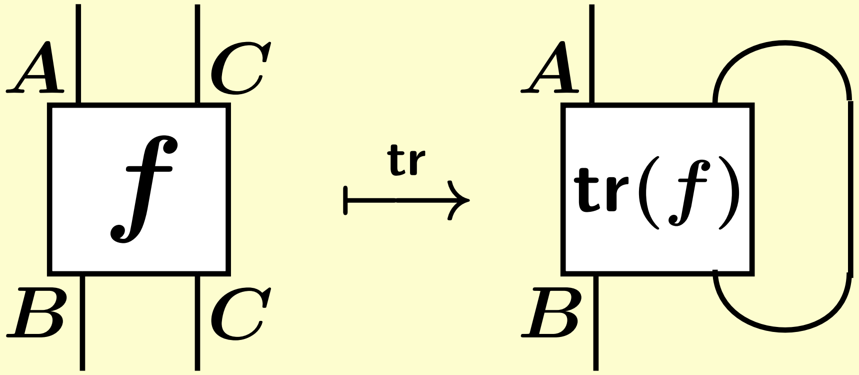
Realizability

Linear category

The Categorical GoI Workflow

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



Categorical GoI [AHS02]

Linear combinatory algebra

- Applicative str. + combinators
- Model of untyped calculus

Realizability

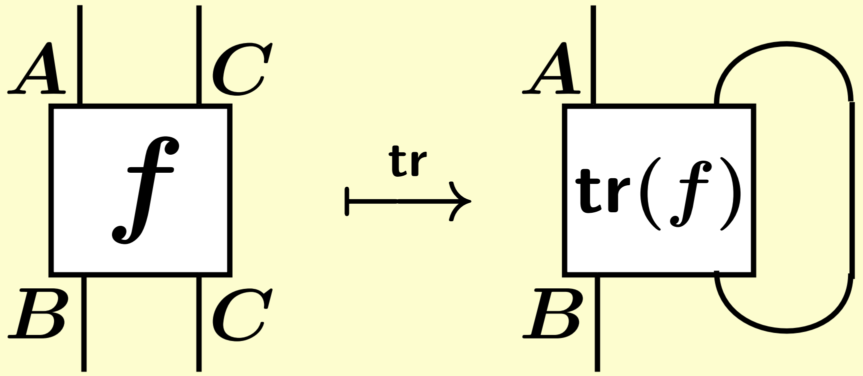
Linear category

Model of typed calculus

The Categorical GoI Workflow

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



Categorical GoI [AHS02]

- ✱ Applicative str. + combinators
- ✱ Model of untyped calculus

Linear combinatory algebra

Realizability

- ✱ PER, ω-set, assembly, ...
- ✱ "Programming in untyped λ"

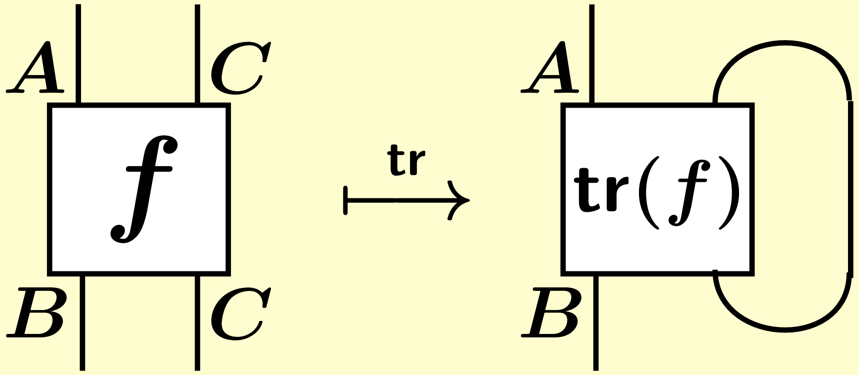
Linear category

Model of typed calculus

The Categorical GoI Workflow

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



Categorical GoI [AHS02]

Linear combinatory algebra

- Applicative str. + combinators
- Model of untyped calculus

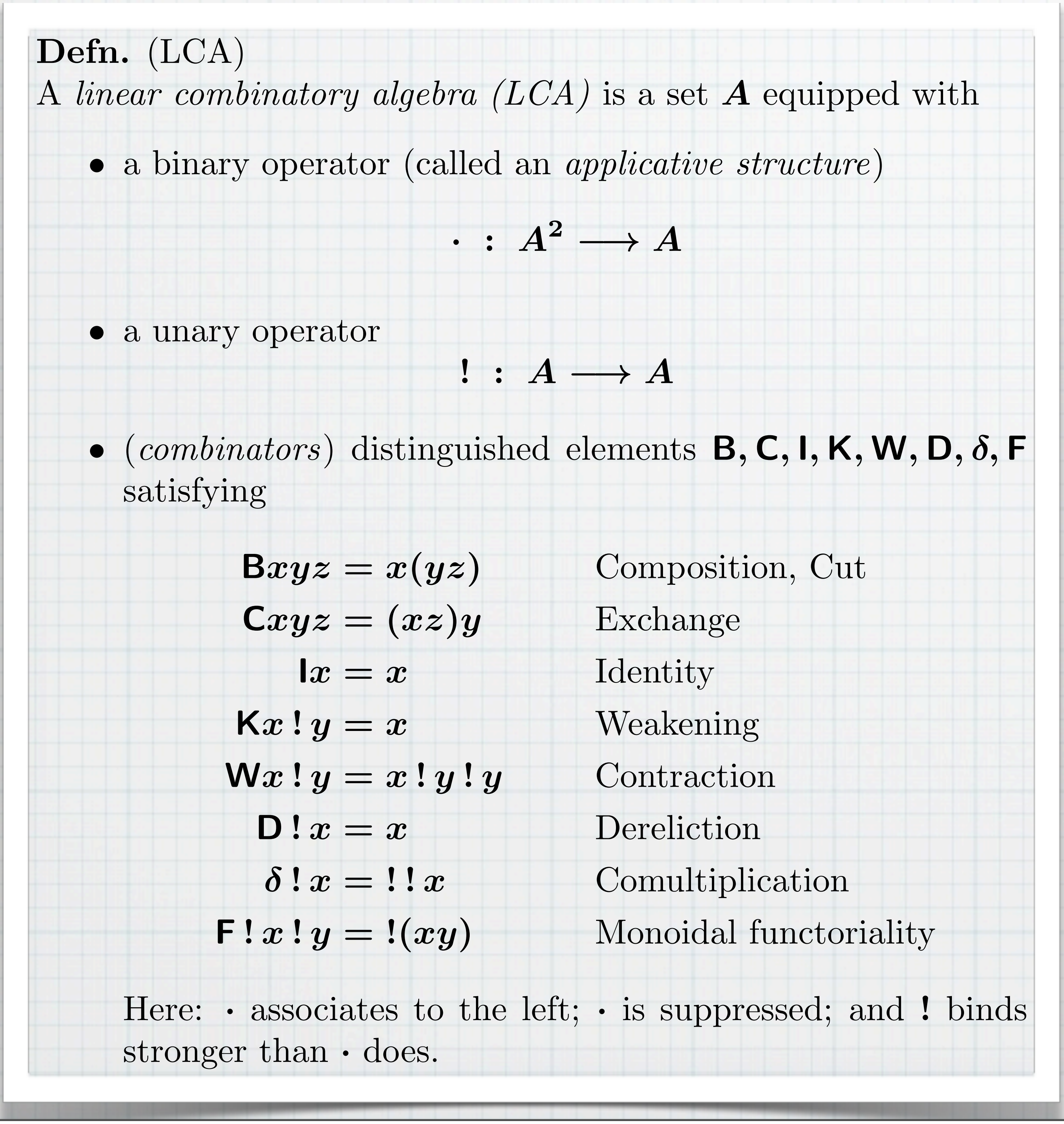
Realizability

- PER, ω-set, assembly, ...
- "Programming in untyped λ"

Linear category

Model of typed calculus

Linear Combinatory Algebra (LCA)



Defn. (LCA)
A *linear combinatory algebra (LCA)* is a set $\boldsymbol{A}$ equipped with

- a binary operator (called an *applicative structure*)

$$\cdot \;:\; \boldsymbol{A}^2 \longrightarrow \boldsymbol{A}$$

- a unary operator

$$! \;:\; \boldsymbol{A} \longrightarrow \boldsymbol{A}$$

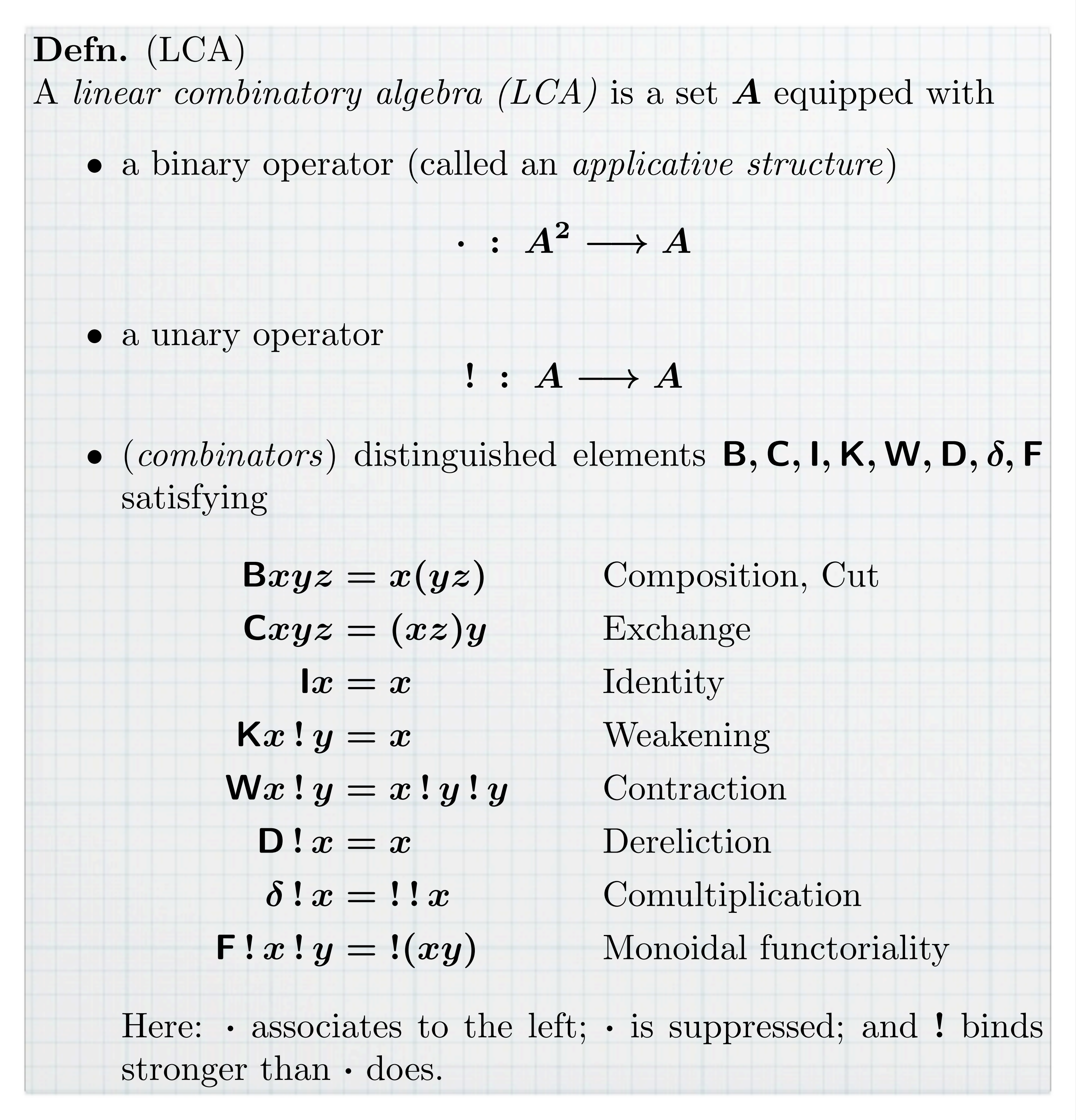
- (*combinators*) distinguished elements $\mathsf{B}, \mathsf{C}, \mathsf{I}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ satisfying

$\mathsf{B}xyz = x(yz)$	Composition, Cut
$\mathsf{C}xyz = (xz)y$	Exchange
$\mathsf{I}x = x$	Identity
$\mathsf{K}x\,!\,y = x$	Weakening
$\mathsf{W}x\,!\,y = x\,!\,y\,!\,y$	Contraction
$\mathsf{D}\,!\,x = x$	Dereliction
$\delta\,!\,x = !\,!\,x$	Comultiplication
$\mathsf{F}\,!\,x\,!\,y = !(xy)$	Monoidal functoriality

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.

Linear Combinatory Algebra (LCA)

What
we want (outcome)



Defn. (LCA)
A *linear combinatory algebra (LCA)* is a set $\boldsymbol{A}$ equipped with

- a binary operator (called an *applicative structure*)

$$\cdot \;:\; \boldsymbol{A}^2 \longrightarrow \boldsymbol{A}$$

- a unary operator

$$! \;:\; \boldsymbol{A} \longrightarrow \boldsymbol{A}$$

- (*combinators*) distinguished elements $\mathsf{B}, \mathsf{C}, \mathsf{I}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ satisfying

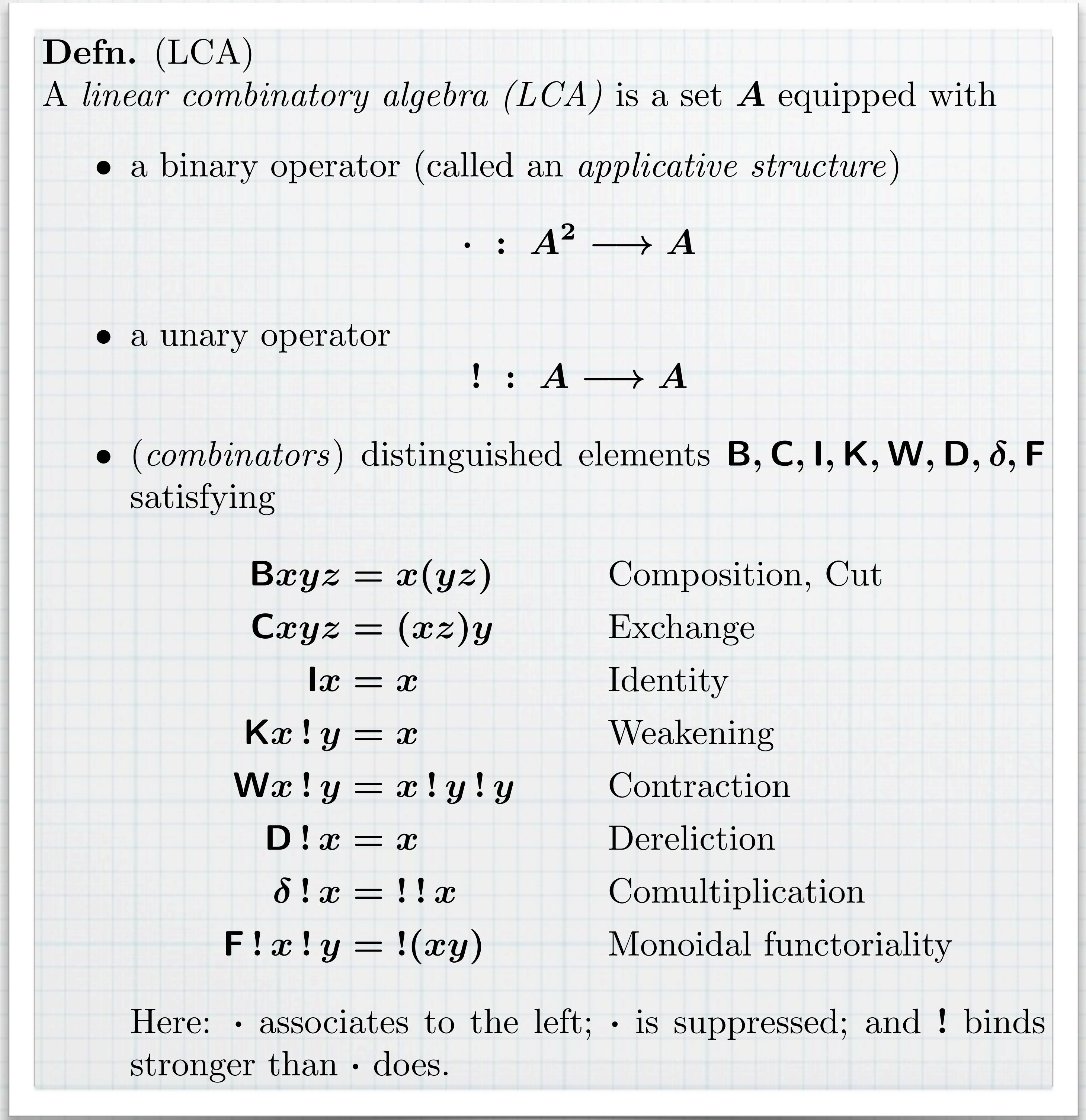
$\mathsf{B}xyz = x(yz)$	Composition, Cut
$\mathsf{C}xyz = (xz)y$	Exchange
$\mathsf{I}x = x$	Identity
$\mathsf{K}x\,!\,y = x$	Weakening
$\mathsf{W}x\,!\,y = x\,!\,y\,!\,y$	Contraction
$\mathsf{D}\,!\,x = x$	Dereliction
$\delta\,!\,x = !\,!\,x$	Comultiplication
$\mathsf{F}\,!\,x\,!\,y = !(xy)$	Monoidal functoriality

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.

Linear Combinatory Algebra (LCA)

What

we want (outcome)



Defn. (LCA)
A *linear combinatory algebra (LCA)* is a set $\boldsymbol{A}$ equipped with

- a binary operator (called an *applicative structure*)

$$\cdot \;:\; \boldsymbol{A}^2 \longrightarrow \boldsymbol{A}$$

- a unary operator

$$! \;:\; \boldsymbol{A} \longrightarrow \boldsymbol{A}$$

- (*combinators*) distinguished elements $\mathsf{B}, \mathsf{C}, \mathsf{I}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ satisfying

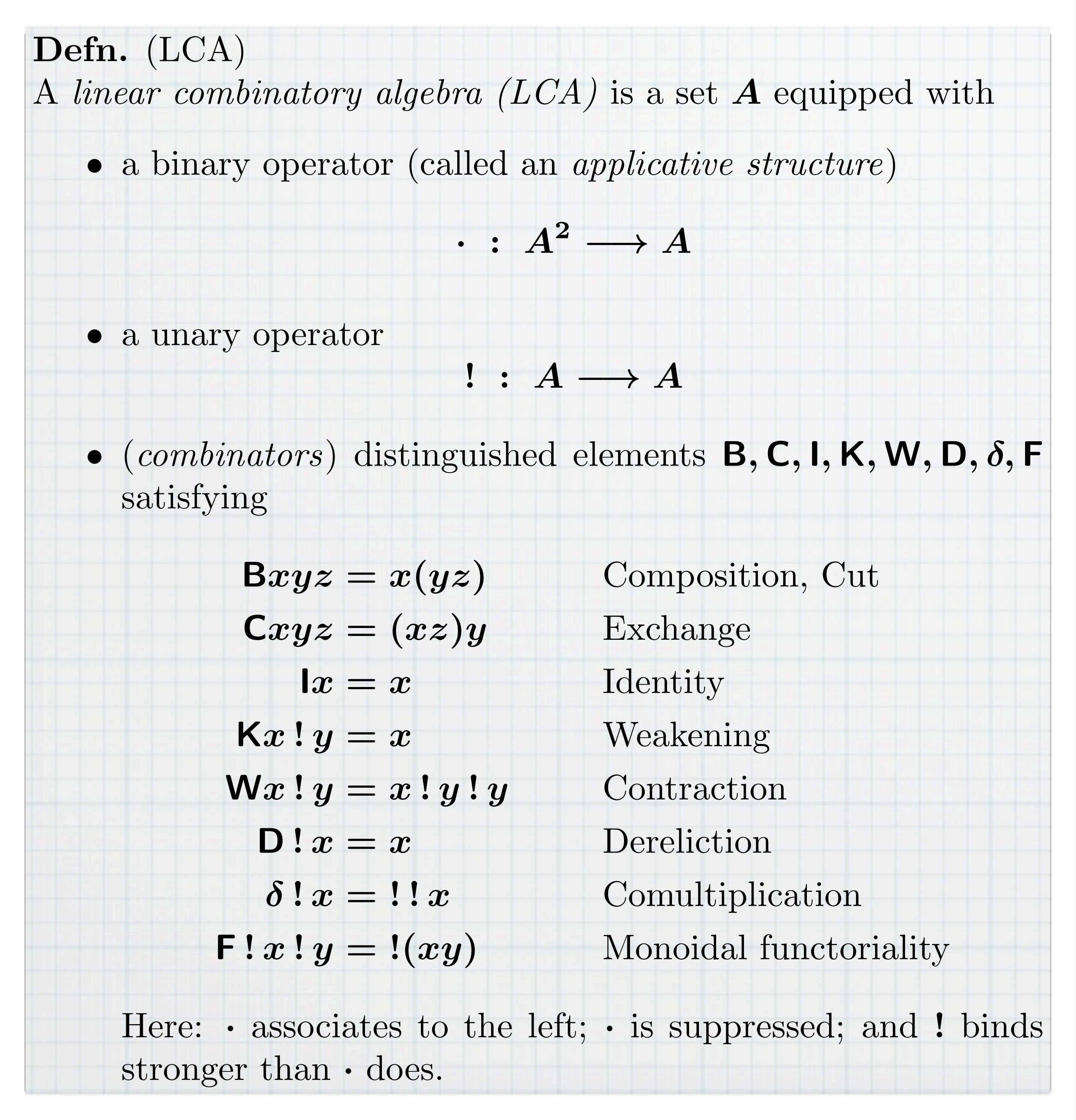
$$\begin{array}{rcll}
\mathsf{B}xyz &=& x(yz) & \text{Composition, Cut} \\
\mathsf{C}xyz &=& (xz)y & \text{Exchange} \\
\mathsf{I}x &=& x & \text{Identity} \\
\mathsf{K}x\,!\,y &=& x & \text{Weakening} \\
\mathsf{W}x\,!\,y &=& x\,!\,y\,!\,y & \text{Contraction} \\
\mathsf{D}\,!\,x &=& x & \text{Dereliction} \\
\delta\,!\,x &=& !\,!\,x & \text{Comultiplication} \\
\mathsf{F}\,!\,x\,!\,y &=& !(xy) & \text{Monoidal functoriality}
\end{array}$$

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.

- ✱ Model of untyped linear λ

Linear Combinatory Algebra (LCA)

What
we want (outcome)



Defn. (LCA)
A *linear combinatory algebra (LCA)* is a set $\boldsymbol{A}$ equipped with

- a binary operator (called an *applicative structure*)

$$\cdot\ :\ \boldsymbol{A}^2 \longrightarrow \boldsymbol{A}$$

- a unary operator

$$!\ :\ \boldsymbol{A} \longrightarrow \boldsymbol{A}$$

- (*combinators*) distinguished elements $\mathsf{B}, \mathsf{C}, \mathsf{I}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ satisfying

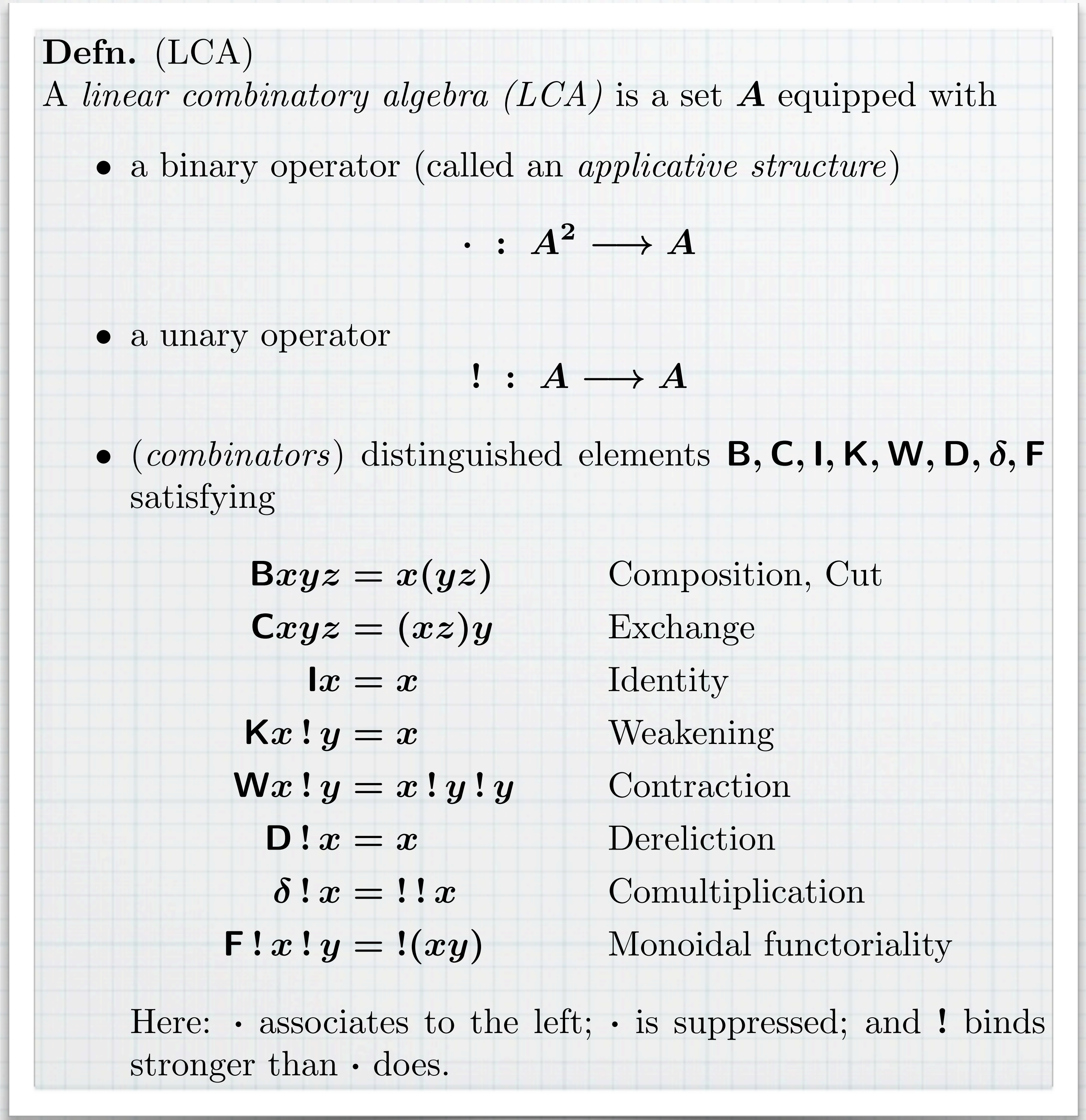
$\mathsf{B}xyz = x(yz)$	Composition, Cut
$\mathsf{C}xyz = (xz)y$	Exchange
$\mathsf{I}x = x$	Identity
$\mathsf{K}x\,!\,y = x$	Weakening
$\mathsf{W}x\,!\,y = x\,!\,y\,!\,y$	Contraction
$\mathsf{D}\,!\,x = x$	Dereliction
$\delta\,!\,x = !\,!\,x$	Comultiplication
$\mathsf{F}\,!\,x\,!\,y = !(xy)$	Monoidal functoriality

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.

- ✱ Model of untyped linear λ
- ✱ a ∈ A ≈ closed linear λ-term

Linear Combinatory Algebra (LCA)

What
we want (outcome)



Defn. (LCA)
A *linear combinatory algebra (LCA)* is a set $\boldsymbol{A}$ equipped with

- a binary operator (called an *applicative structure*)

$$\cdot \;:\; \boldsymbol{A}^2 \longrightarrow \boldsymbol{A}$$

- a unary operator

$$! \;:\; \boldsymbol{A} \longrightarrow \boldsymbol{A}$$

- (*combinators*) distinguished elements $\mathsf{B}, \mathsf{C}, \mathsf{I}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ satisfying

$\mathsf{B}xyz = x(yz)$	Composition, Cut
$\mathsf{C}xyz = (xz)y$	Exchange
$\mathsf{I}x = x$	Identity
$\mathsf{K}x\,!\,y = x$	Weakening
$\mathsf{W}x\,!\,y = x\,!\,y\,!\,y$	Contraction
$\mathsf{D}\,!\,x = x$	Dereliction
$\delta\,!\,x = !\,!\,x$	Comultiplication
$\mathsf{F}\,!\,x\,!\,y = !(xy)$	Monoidal functoriality

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.

- Model of untyped linear λ
- a ∈ A ≈ closed linear λ-term
- No **S** or **K** (linear!)
- Combinatory completeness: e.g.

$$\lambda xyz.\, zxy$$

designates an elem. of A

What we use (ingredient)

GoI situation

Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

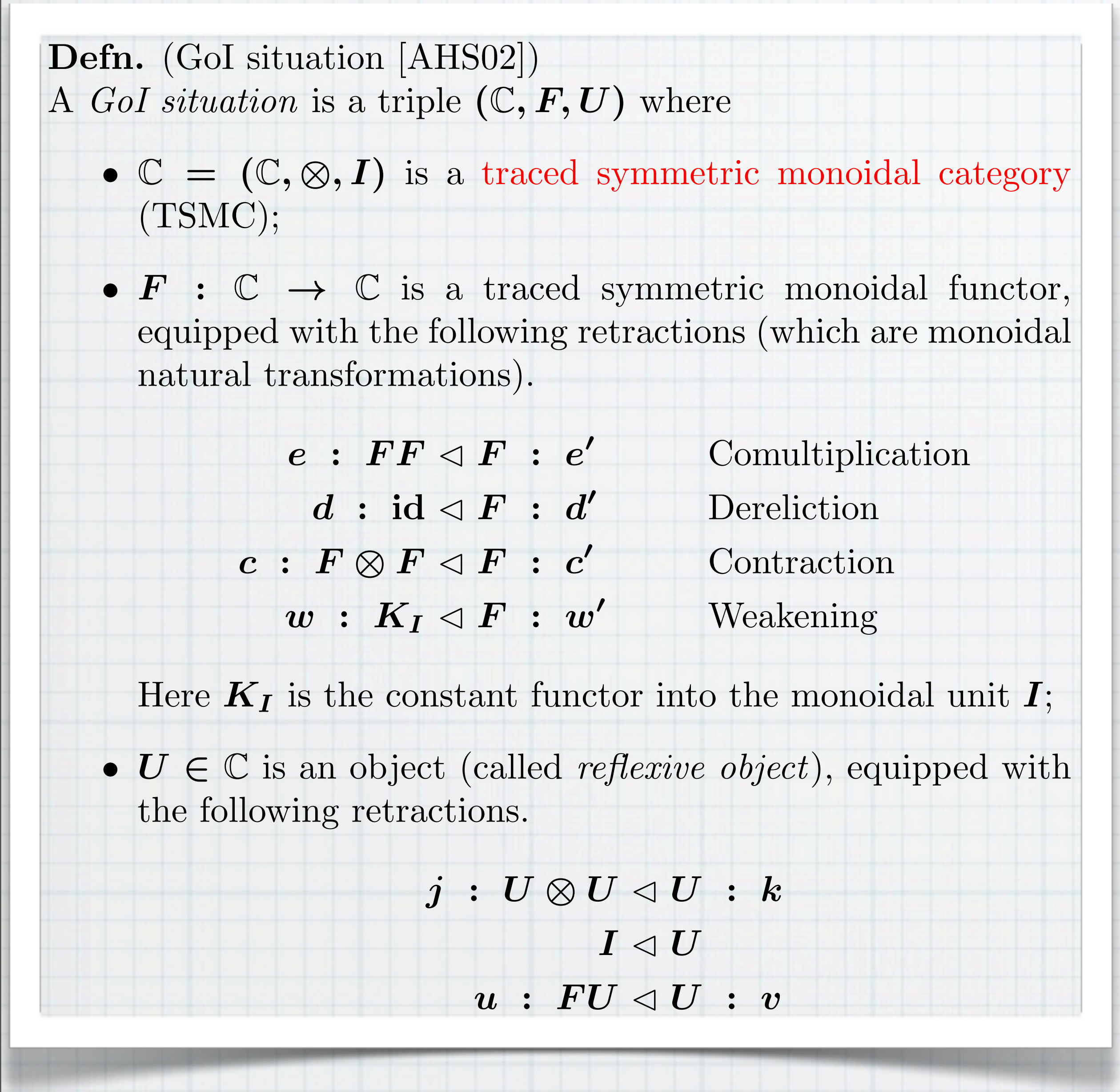
- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

  $$\begin{array}{rcrclcl} \boldsymbol{e} & : & \boldsymbol{FF} & \lhd & \boldsymbol{F} & : & \boldsymbol{e'} & \quad \text{Comultiplication} \\ \boldsymbol{d} & : & \textbf{id} & \lhd & \boldsymbol{F} & : & \boldsymbol{d'} & \quad \text{Dereliction} \\ \boldsymbol{c} & : & \boldsymbol{F} \otimes \boldsymbol{F} & \lhd & \boldsymbol{F} & : & \boldsymbol{c'} & \quad \text{Contraction} \\ \boldsymbol{w} & : & \boldsymbol{K_I} & \lhd & \boldsymbol{F} & : & \boldsymbol{w'} & \quad \text{Weakening} \end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

  $$\begin{array}{rcrclcl} \boldsymbol{j} & : & \boldsymbol{U} \otimes \boldsymbol{U} & \lhd & \boldsymbol{U} & : & \boldsymbol{k} \\ & & \boldsymbol{I} & \lhd & \boldsymbol{U} & & \\ \boldsymbol{u} & : & \boldsymbol{FU} & \lhd & \boldsymbol{U} & : & \boldsymbol{v} \end{array}$$

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

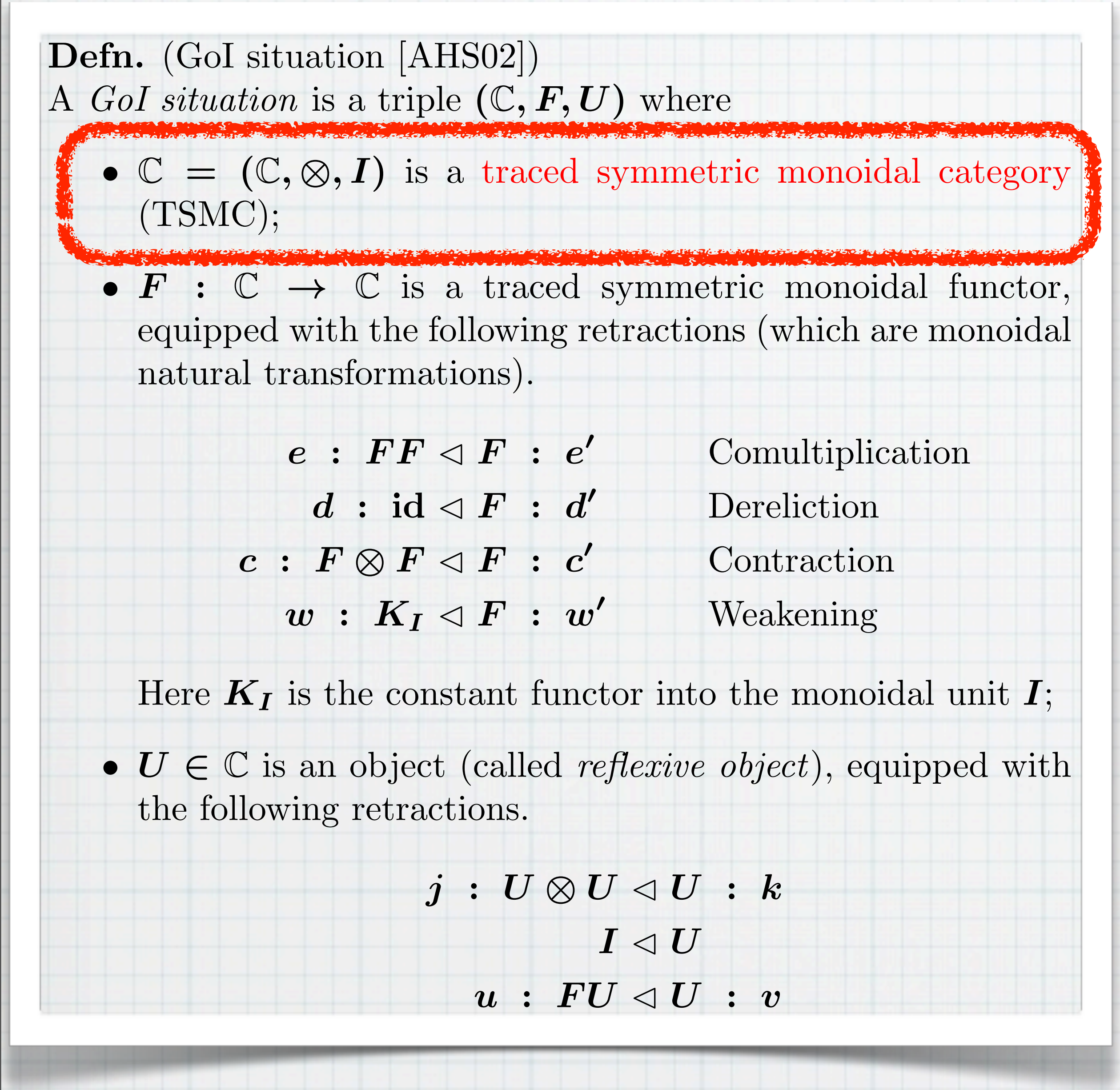
  $$\begin{array}{rcl}
  \boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{e}' & & \text{Comultiplication} \\
  \boldsymbol{d} : \mathbf{id} \lhd \boldsymbol{F} : \boldsymbol{d}' & & \text{Dereliction} \\
  \boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c}' & & \text{Contraction} \\
  \boldsymbol{w} : \boldsymbol{K_I} \lhd \boldsymbol{F} : \boldsymbol{w}' & & \text{Weakening}
  \end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

  $$\begin{array}{c}
  \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{k} \\
  \boldsymbol{I} \lhd \boldsymbol{U} \\
  \boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{v}
  \end{array}$$

- Monoidal category $(\mathbb{C}, \otimes, \boldsymbol{I})$
  - String diagrams

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

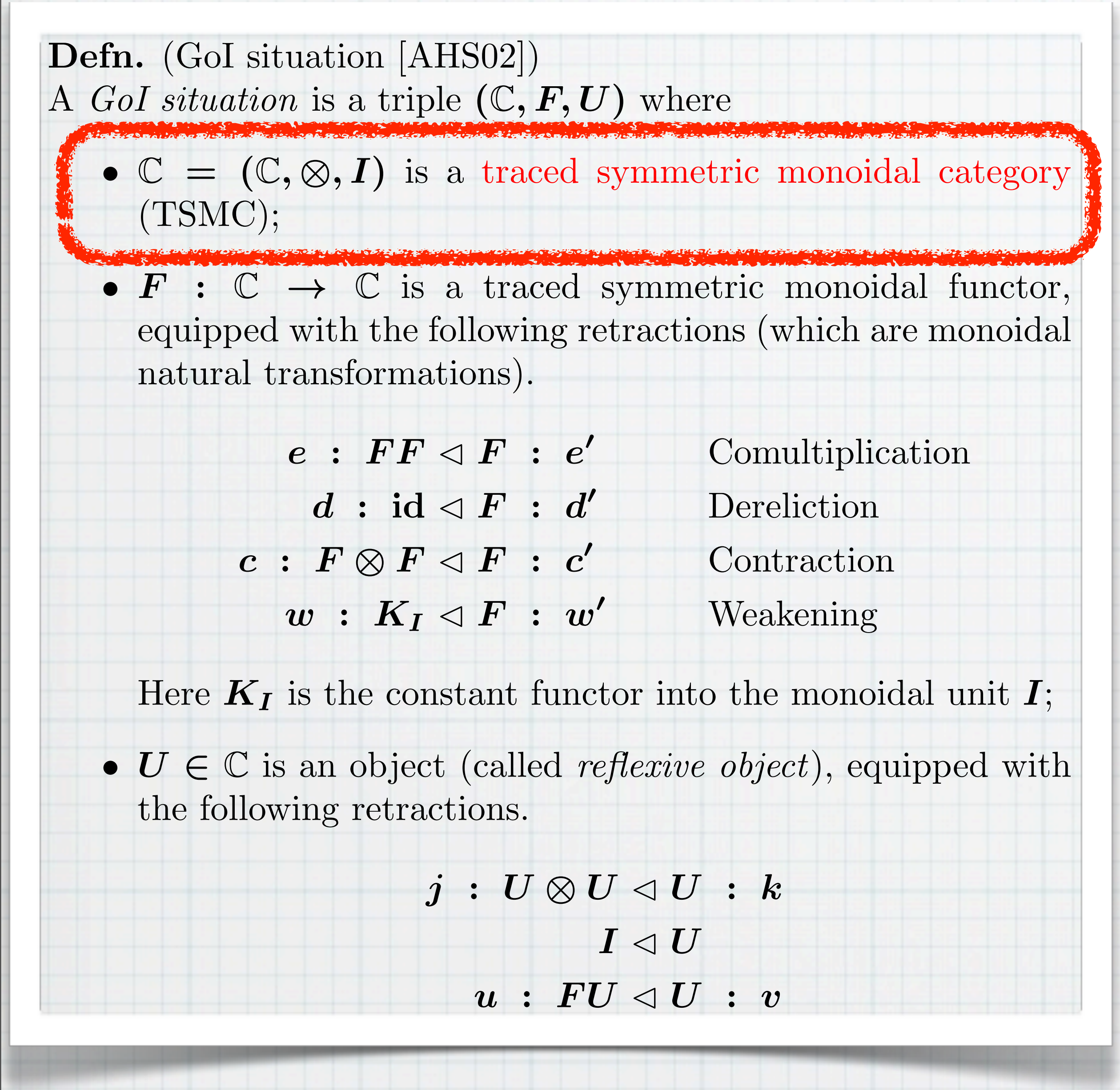
$$\begin{array}{rcl l}
\boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{e}' & & & \text{Comultiplication} \\
\boldsymbol{d} : \mathbf{id} \lhd \boldsymbol{F} : \boldsymbol{d}' & & & \text{Dereliction} \\
\boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c}' & & & \text{Contraction} \\
\boldsymbol{w} : \boldsymbol{K_I} \lhd \boldsymbol{F} : \boldsymbol{w}' & & & \text{Weakening}
\end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{c}
\boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{k} \\
\boldsymbol{I} \lhd \boldsymbol{U} \\
\boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{v}
\end{array}$$

- **Monoidal category** $(\mathbb{C}, \otimes, I)$
  - **String diagrams**

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{array}{rcl}
e : FF \lhd F : e' & & \text{Comultiplication} \\
d : \mathrm{id} \lhd F : d' & & \text{Dereliction} \\
c : F \otimes F \lhd F : c' & & \text{Contraction} \\
w : K_I \lhd F : w' & & \text{Weakening}
\end{array}$$

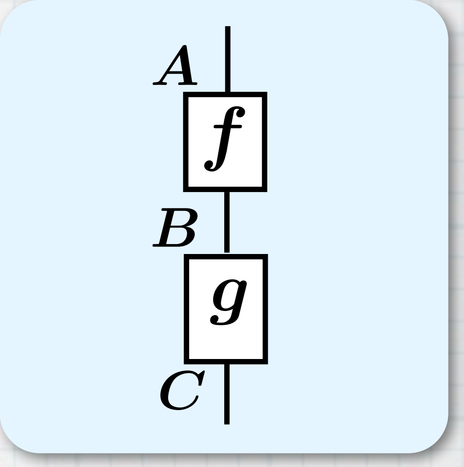
Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;

- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

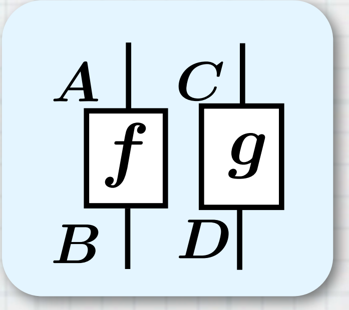
$$\begin{array}{c}
j : U \otimes U \lhd U : k \\
I \lhd U \\
u : FU \lhd U : v
\end{array}$$

- **Monoidal category** $(\mathbb{C}, \otimes, I)$
  - **String diagrams**

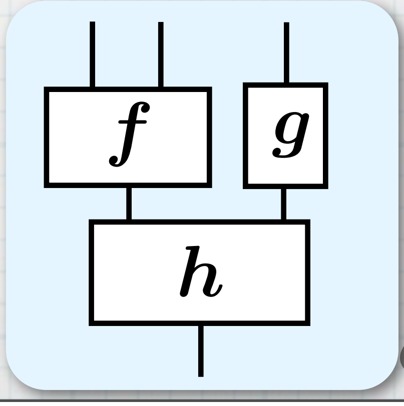
$$\frac{A \xrightarrow{f} B \qquad B \xrightarrow{g} C}{A \xrightarrow{g \circ f} C}$$



$$\frac{A \xrightarrow{f} B \qquad C \xrightarrow{g} D}{A \otimes C \xrightarrow{f \otimes g} B \otimes D}$$

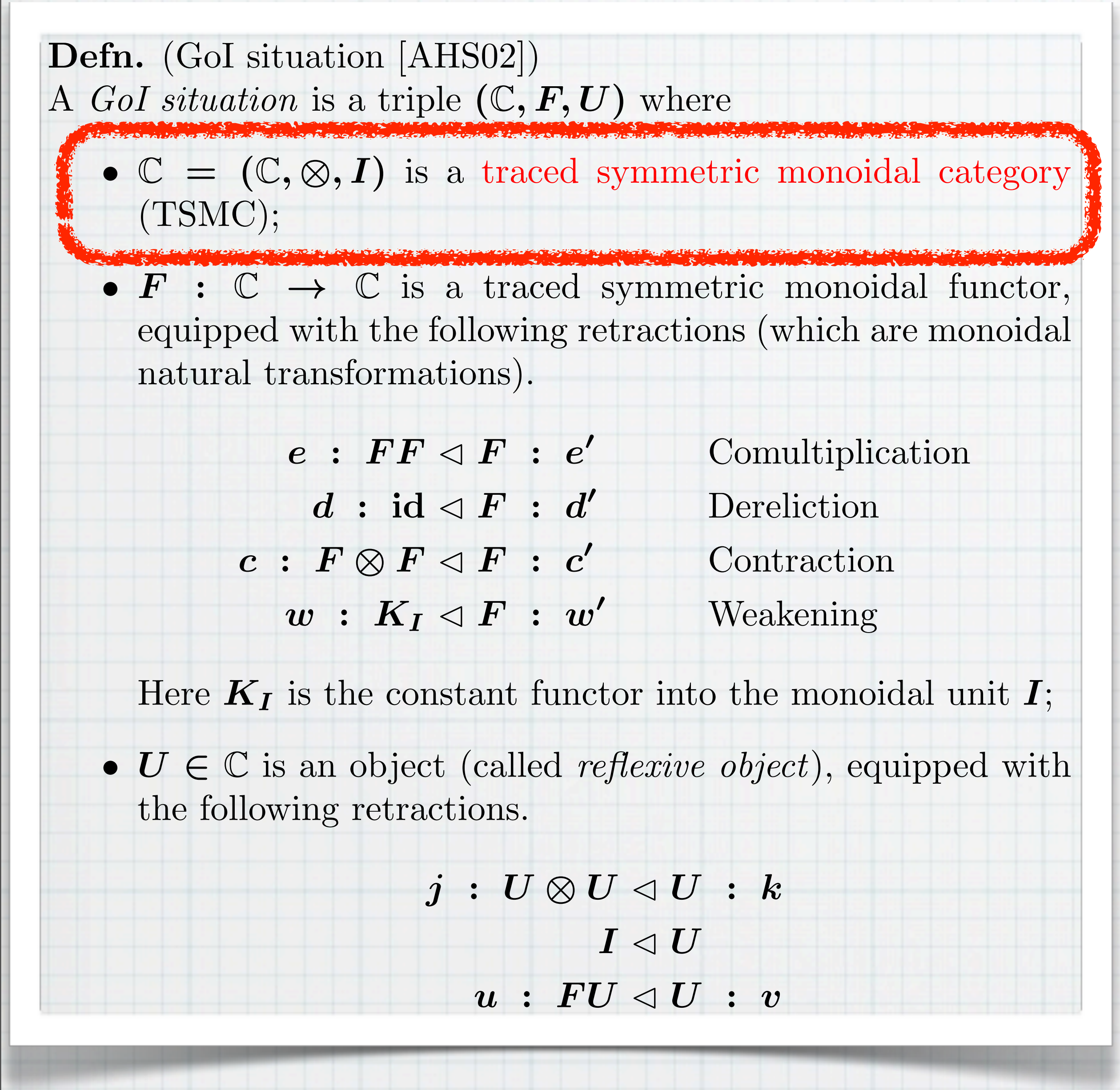


$$h \circ (f \otimes g)$$



o)

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$e : FF \lhd F : e'$	Comultiplication
$d : \mathrm{id} \lhd F : d'$	Dereliction
$c : F \otimes F \lhd F : c'$	Contraction
$w : K_I \lhd F : w'$	Weakening

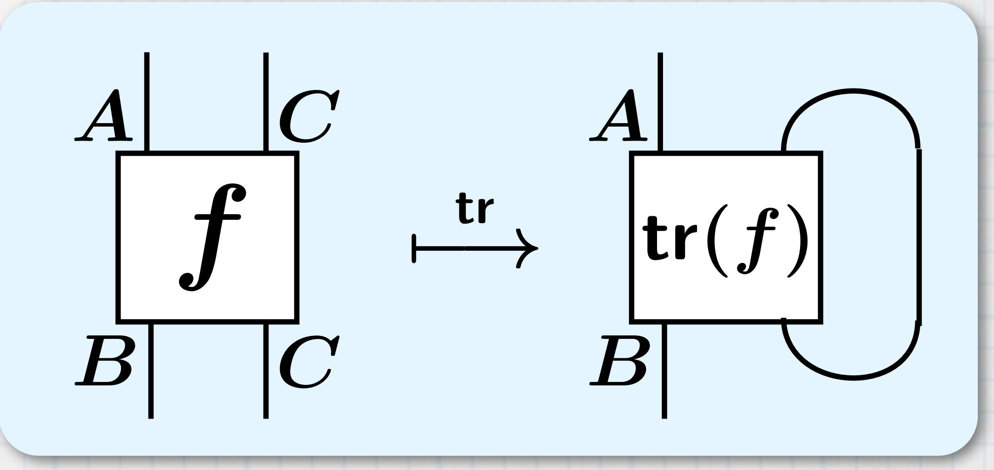
  Here K_I is the constant functor into the monoidal unit I;
- $U \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$j : U \otimes U \lhd U : k$$
$$I \lhd U$$
$$u : FU \lhd U : v$$

- **Traced** monoidal category
  - "feedback"

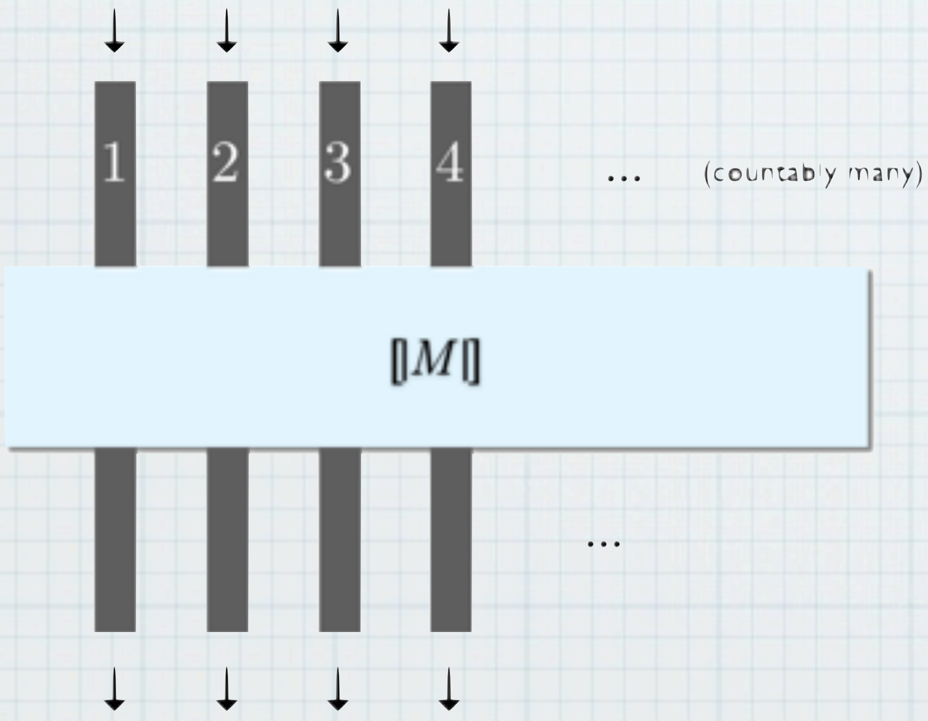
$$\frac{A \otimes C \xrightarrow{f} B \otimes C}{A \xrightarrow{\mathrm{tr}(f)} B}$$

that is

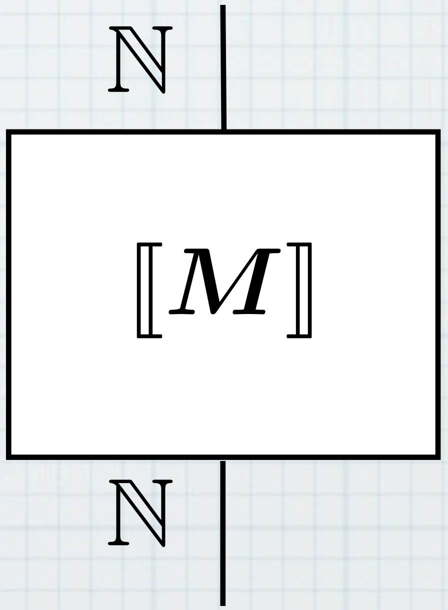


String Diagram vs. "Pipe Diagram"

- I use two ways of depicting partial functions $\mathbb{N} \rightharpoonup \mathbb{N}$



Pipe diagram

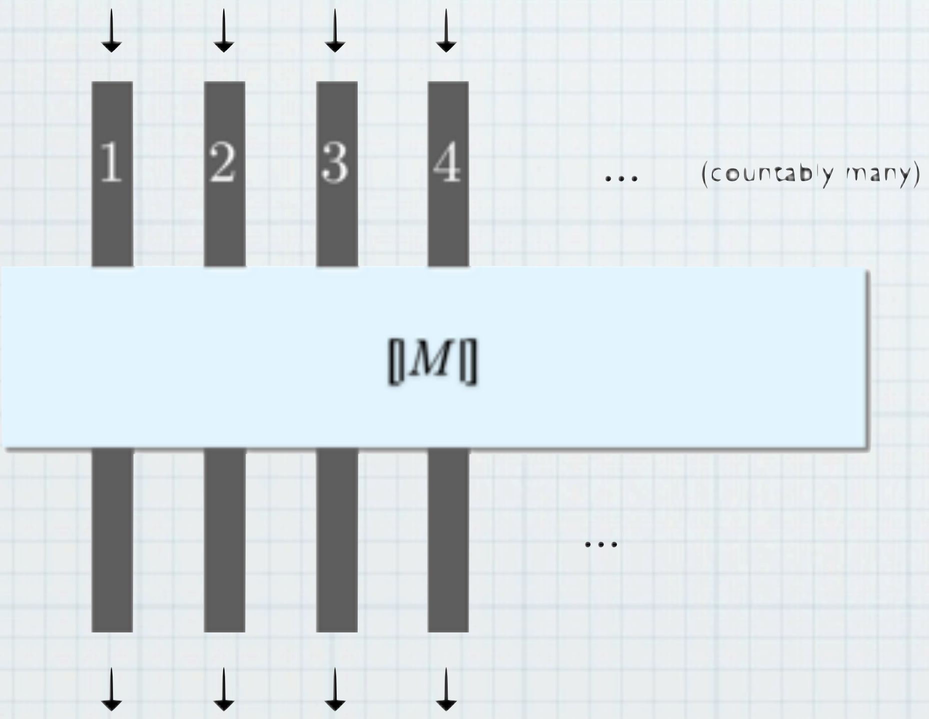


String diagram

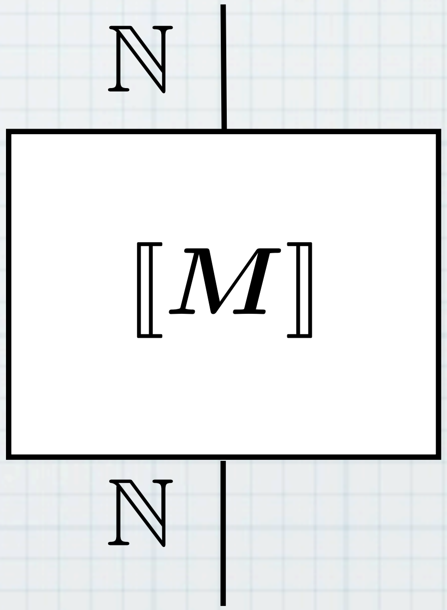
String Diagram vs. “Pipe Diagram”

- I use two ways of depicting partial functions $\mathbb{N} \rightharpoonup \mathbb{N}$

In the monoidal category $(\mathbf{Pfn}, +, 0)$



Pipe diagram



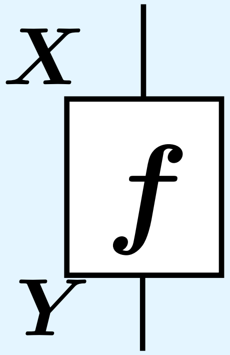
String diagram

Traced Sym. Monoidal Category

$(\mathbf{Pfn}, +, 0)$

- Category **Pfn** of partial functions
  - **Obj.** A set X
  - **Arr.** A partial function

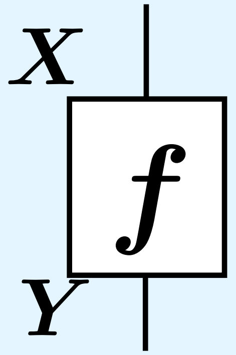
$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y, \text{ partial function} \end{array}$$



Traced Sym. Monoidal Category $(\mathbf{Pfn}, +, 0)$

- Category $\mathbf{Pfn}$ of partial functions
  - Obj. A set X
  - Arr. A partial function

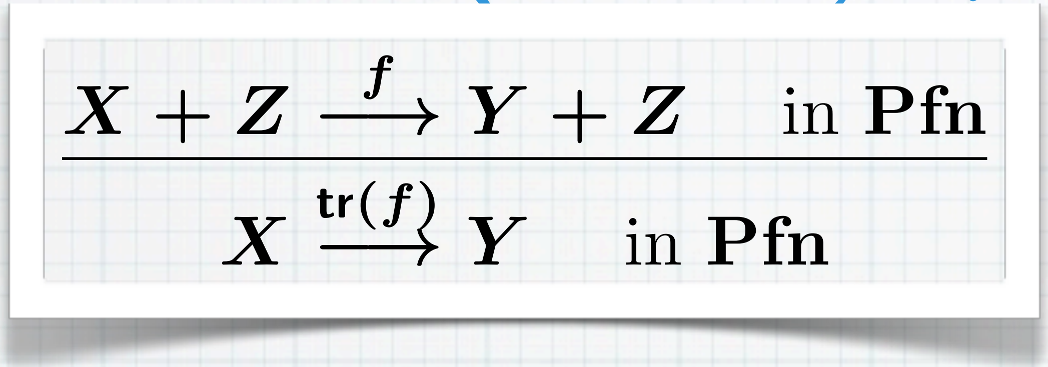
$$\frac{X \to Y \ \text{ in } \mathbf{Pfn}}{X \rightharpoonup Y, \text{ partial function}}$$



- is traced symmetric monoidal

Traced Sym. Monoidal Category

$$(\mathbf{Pfn}, +, 0)$$



$$\frac{X + Z \xrightarrow{f} Y + Z \qquad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathsf{tr}(f)} Y \qquad \text{in } \mathbf{Pfn}}$$

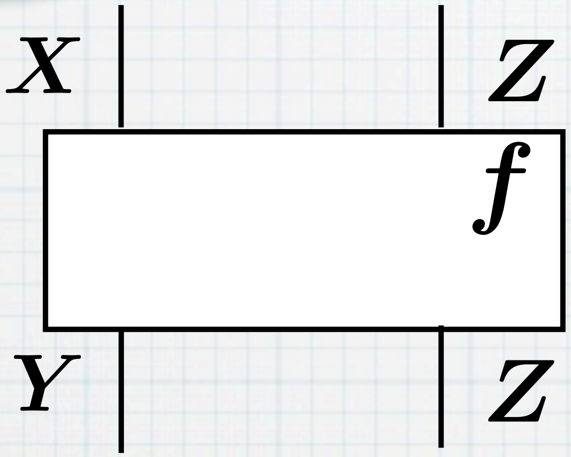
How?

Traced Sym. Monoidal Category

$$(\mathbf{Pfn}, +, 0)$$

- $$\frac{X + Z \xrightarrow{f} Y + Z \quad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathsf{tr}(f)} Y \quad \text{in } \mathbf{Pfn}}$$

How?

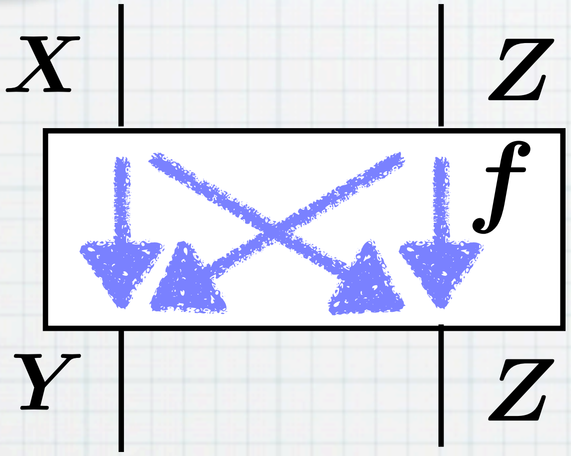
- 

Traced Sym. Monoidal Category

$(\mathbf{Pfn}, +, 0)$

- $$\frac{X + Z \xrightarrow{f} Y + Z \quad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathrm{tr}(f)} Y \quad \text{in } \mathbf{Pfn}}$$

How?

- 

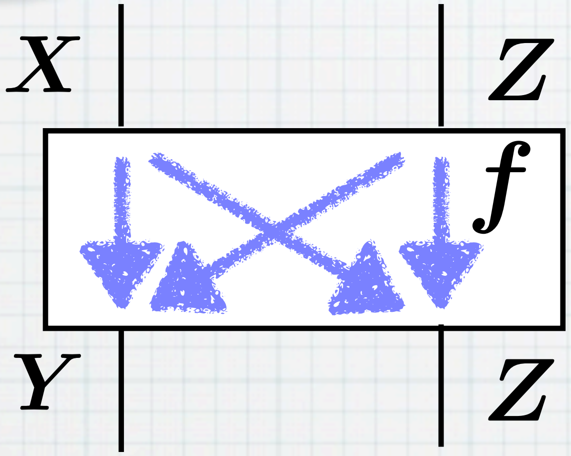
Hasuo (Tokyo)

Traced Sym. Monoidal Category

$(\mathbf{Pfn}, +, 0)$

- $$\frac{X + Z \xrightarrow{f} Y + Z \quad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathrm{tr}(f)} Y \quad \text{in } \mathbf{Pfn}}$$

How?

- 

$$f_{XY}(x) := \begin{cases} f(x) & \text{if } f(x) \in Y \\ \bot & \text{o.w.} \end{cases}$$

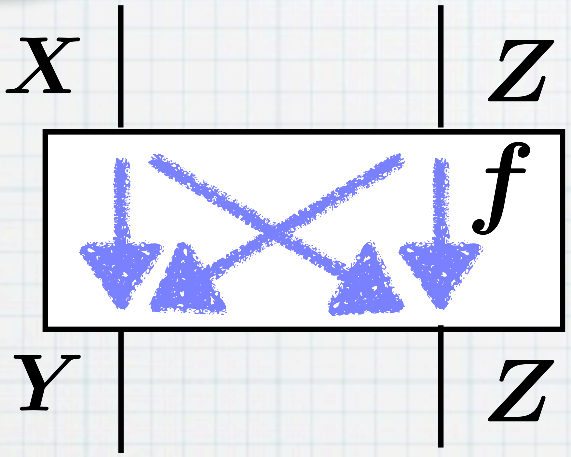
Similar for f_{XZ}, f_{ZY}, f_{ZZ}

Traced Sym. Monoidal Category

$$(\mathbf{Pfn}, +, 0)$$

- $$\frac{X + Z \xrightarrow{f} Y + Z \qquad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathsf{tr}(f)} Y \qquad \text{in } \mathbf{Pfn}}$$

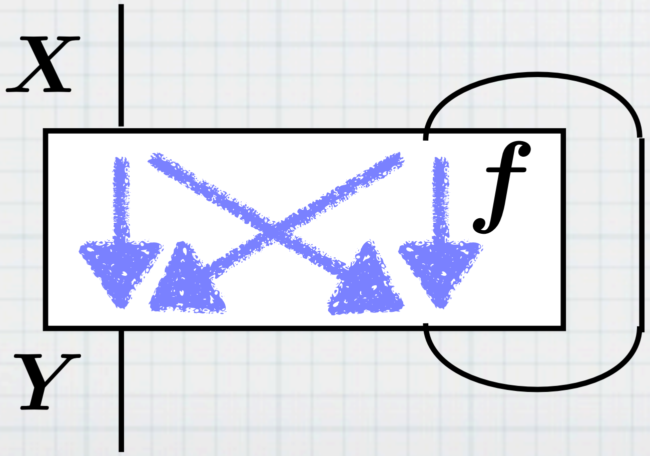
How?

- 

$$f_{XY}(x) := \begin{cases} f(x) & \text{if } f(x) \in Y \\ \bot & \text{o.w.} \end{cases}$$

Similar for f_{XZ}, f_{ZY}, f_{ZZ}

- Trace operator:

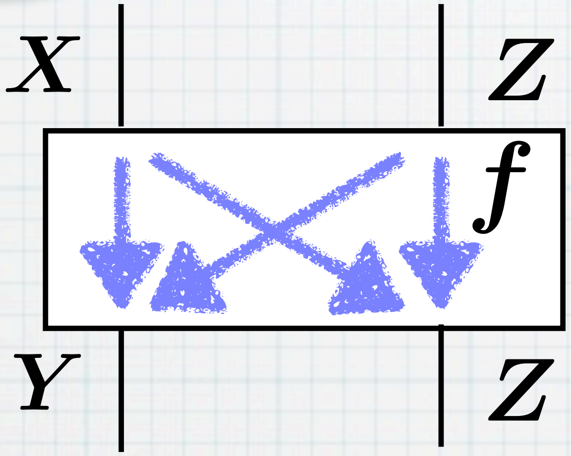


Traced Sym. Monoidal Category

$(\mathbf{Pfn}, +, 0)$

- $$\dfrac{X+Z \xrightarrow{f} Y+Z \qquad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathsf{tr}(f)} Y \qquad \text{in } \mathbf{Pfn}}$$

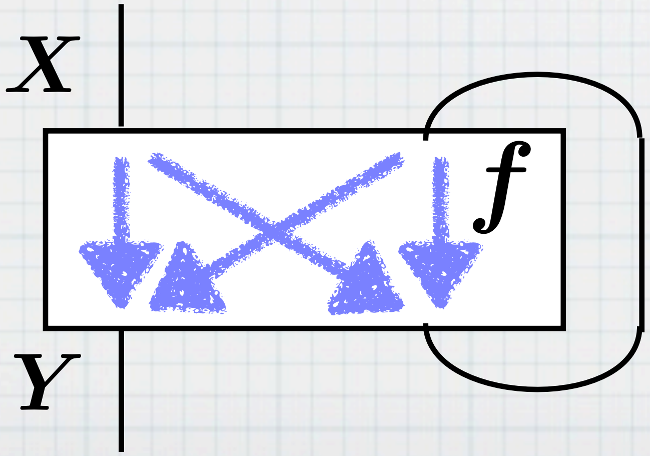
How?

- 

$$f_{XY}(x) := \begin{cases} f(x) & \text{if } f(x) \in Y \\ \bot & \text{o.w.} \end{cases}$$

Similar for f_{XZ}, f_{ZY}, f_{ZZ}

- Trace operator:



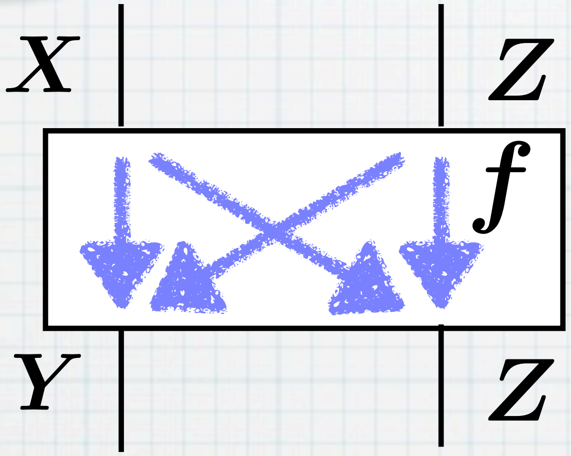
$$\mathsf{tr}(f) = f_{XY} \sqcup \left(\coprod_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

Traced Sym. Monoidal Category

$(\mathbf{Pfn}, +, 0)$

- $$\frac{X + Z \xrightarrow{f} Y + Z \qquad \text{in } \mathbf{Pfn}}{X \xrightarrow{\mathsf{tr}(f)} Y \qquad \text{in } \mathbf{Pfn}}$$

How?

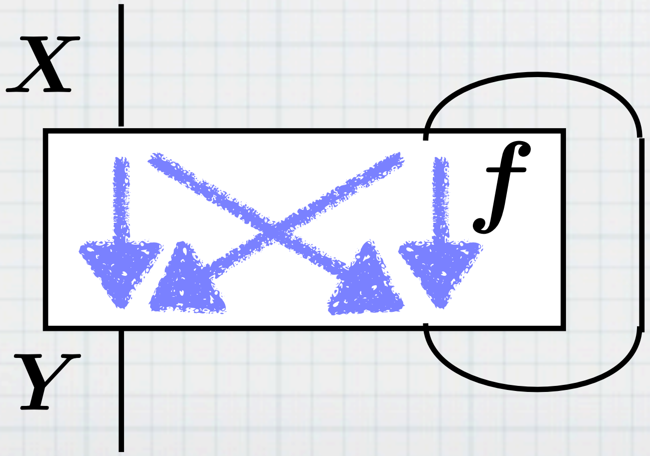
- 

$$f_{XY}(x) := \begin{cases} f(x) & \text{if } f(x) \in Y \\ \bot & \text{o.w.} \end{cases}$$

Similar for f_{XZ}, f_{ZY}, f_{ZZ}

- **Execution formula** (Girard)
- Partiality is essential (infinite loop)

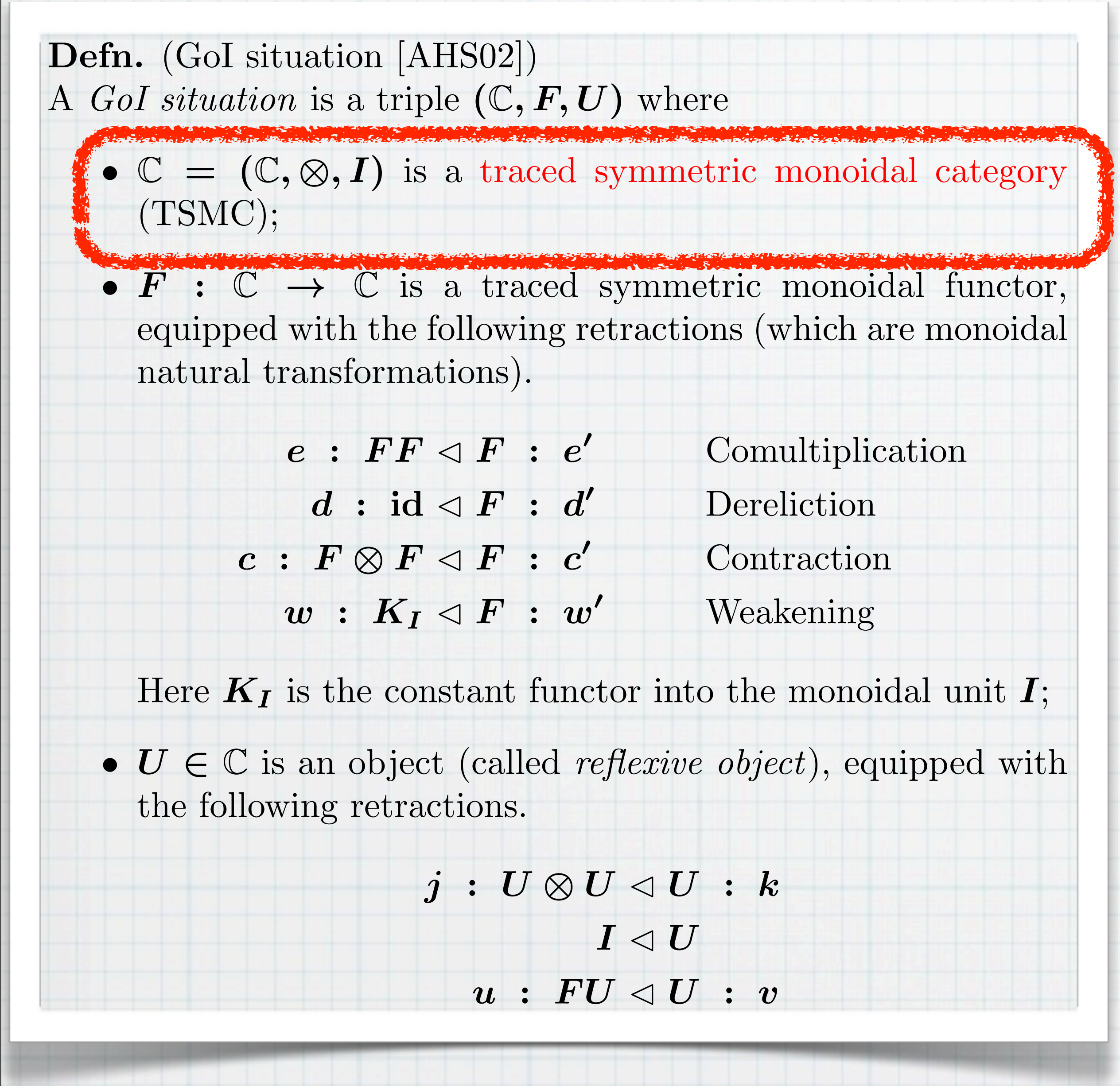
- Trace operator:



$$\mathsf{tr}(f) =$$

$$f_{XY} \sqcup \left(\coprod_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)$$

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

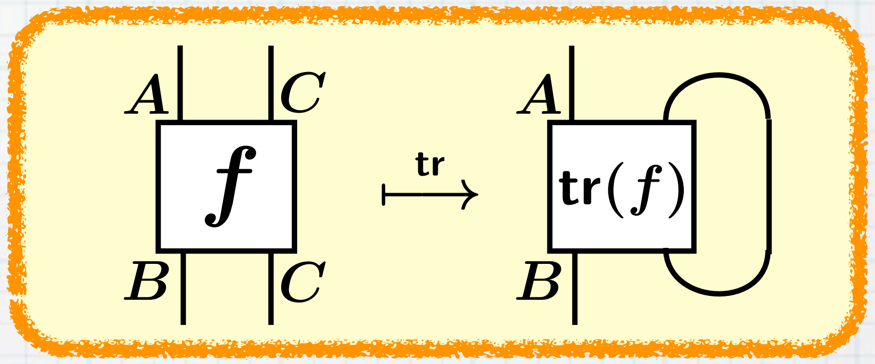
- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{array}{rcl}
e : FF \lhd F : e' & & \text{Comultiplication} \\
d : \mathrm{id} \lhd F : d' & & \text{Dereliction} \\
c : F \otimes F \lhd F : c' & & \text{Contraction} \\
w : K_I \lhd F : w' & & \text{Weakening}
\end{array}$$

  Here K_I is the constant functor into the monoidal unit I;
- $U \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

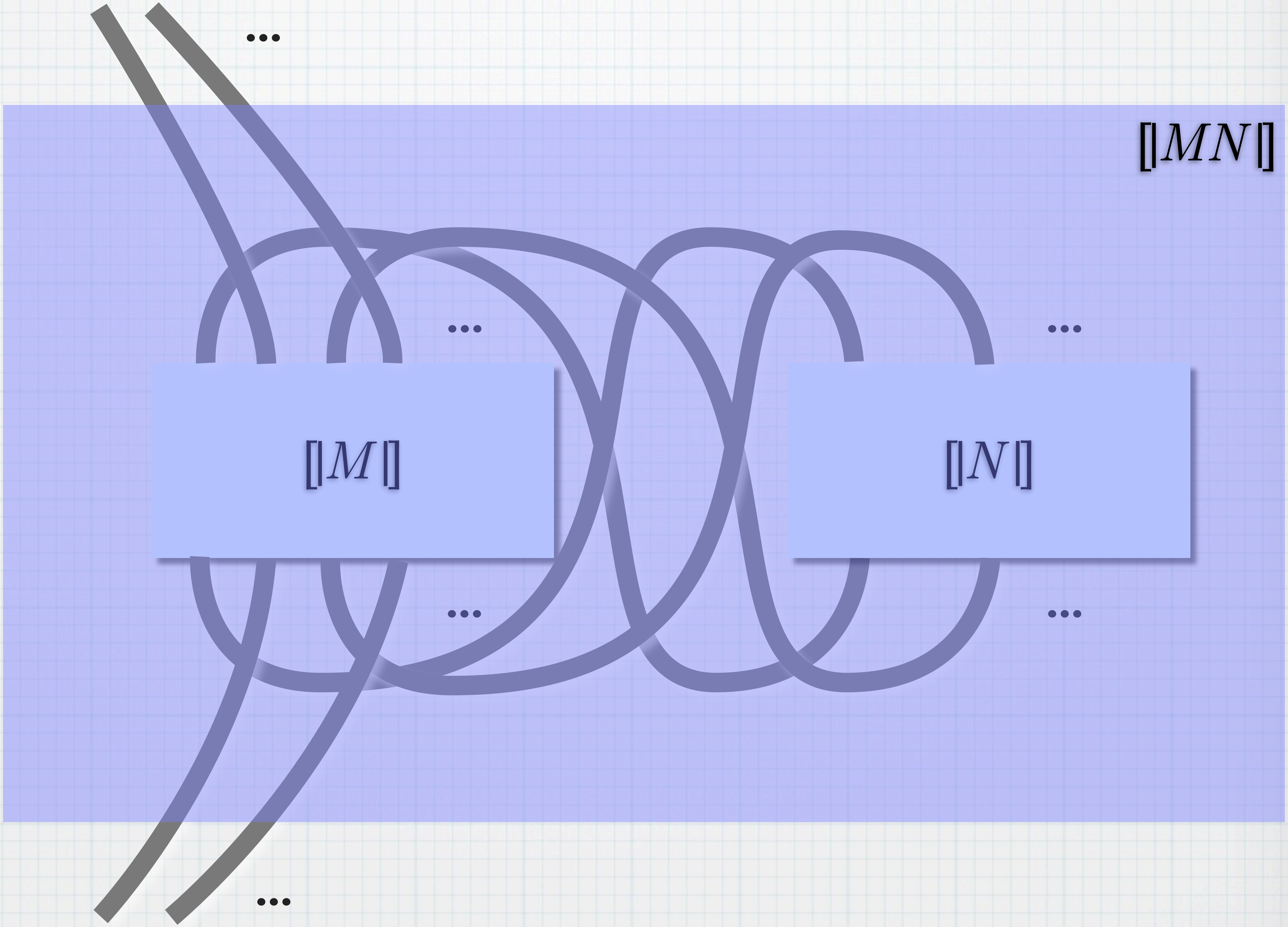
$$\begin{array}{c}
j : U \otimes U \lhd U : k \\
I \lhd U \\
u : FU \lhd U : v
\end{array}$$

- Traced sym. monoidal cat.
  - Where one can "feedback"

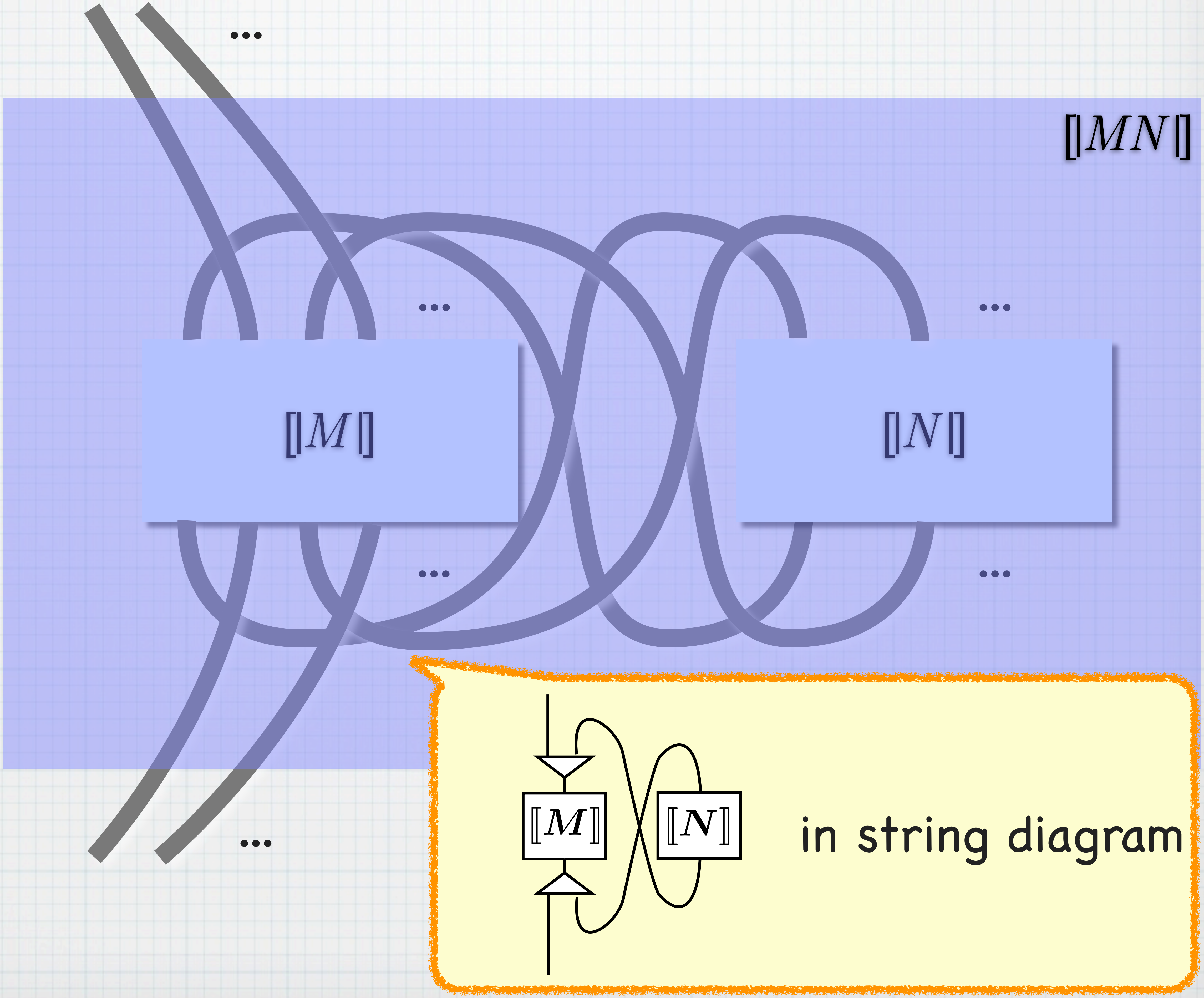


  - Why for GoI?

$[\![\boldsymbol{M}\boldsymbol{N}]\!] =$

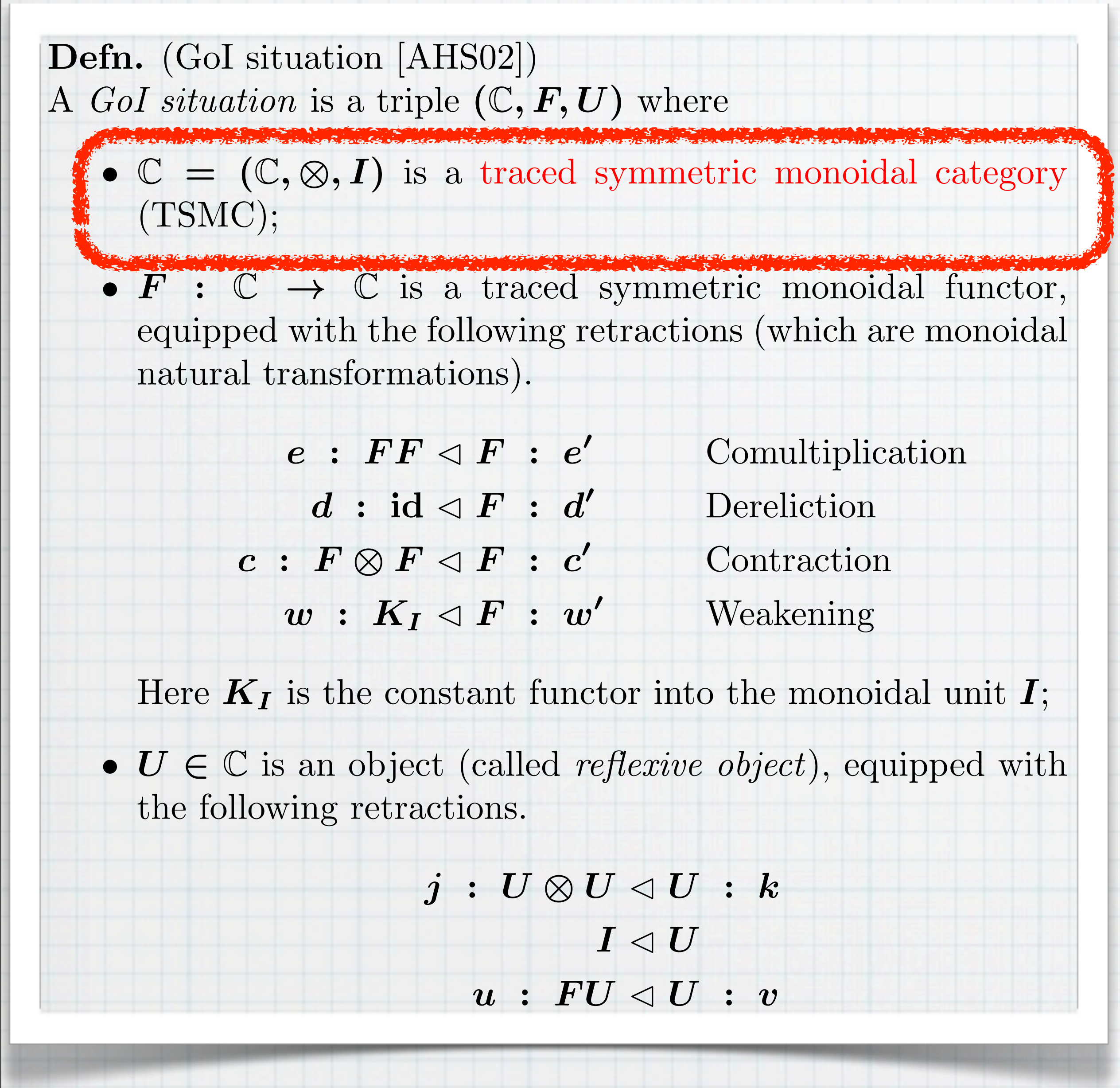


$[\![MN]\!] =$



in string diagram

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

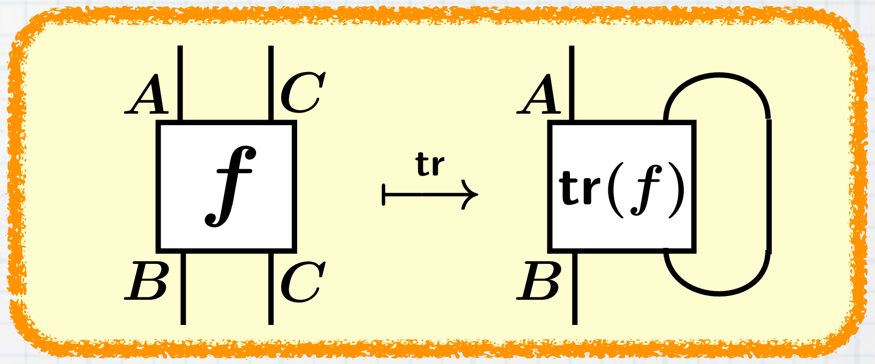
$$\begin{array}{rcl l}
\boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{e}' & & & \text{Comultiplication} \\
\boldsymbol{d} : \mathbf{id} \lhd \boldsymbol{F} : \boldsymbol{d}' & & & \text{Dereliction} \\
\boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c}' & & & \text{Contraction} \\
\boldsymbol{w} : \boldsymbol{K}_{\boldsymbol{I}} \lhd \boldsymbol{F} : \boldsymbol{w}' & & & \text{Weakening}
\end{array}$$

Here $\boldsymbol{K}_{\boldsymbol{I}}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;

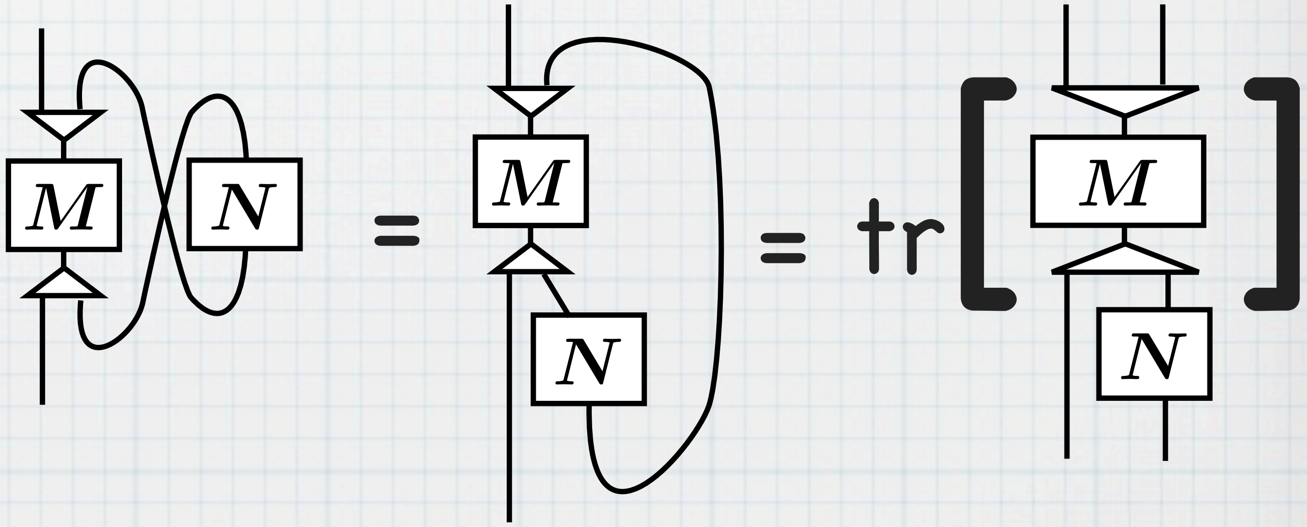
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{c}
\boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{k} \\
\boldsymbol{I} \lhd \boldsymbol{U} \\
\boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{v}
\end{array}$$

- Traced sym. monoidal cat.
  - Where one can "feedback"



  - Why for GoI?



  - Leading example: **Pfn**

GoI situation

Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$e : FF \lhd F : e'$	Comultiplication
$d : \mathrm{id} \lhd F : d'$	Dereliction
$c : F \otimes F \lhd F : c'$	Contraction
$w : K_I \lhd F : w'$	Weakening

  Here K_I is the constant functor into the monoidal unit I;
- $U \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

  $$j : U \otimes U \lhd U : k$$
  $$I \lhd U$$
  $$u : FU \lhd U : v$$

Defn. (Retraction)
A *retraction* from X to Y,

$$f : X \lhd Y : g \ ,$$

is a pair of arrows

$$\begin{tikzcd} X \arrow[loop left, "\mathrm{id}"] \arrow[r, bend left, "f"] & Y \arrow[l, bend left, "g"] \end{tikzcd}$$

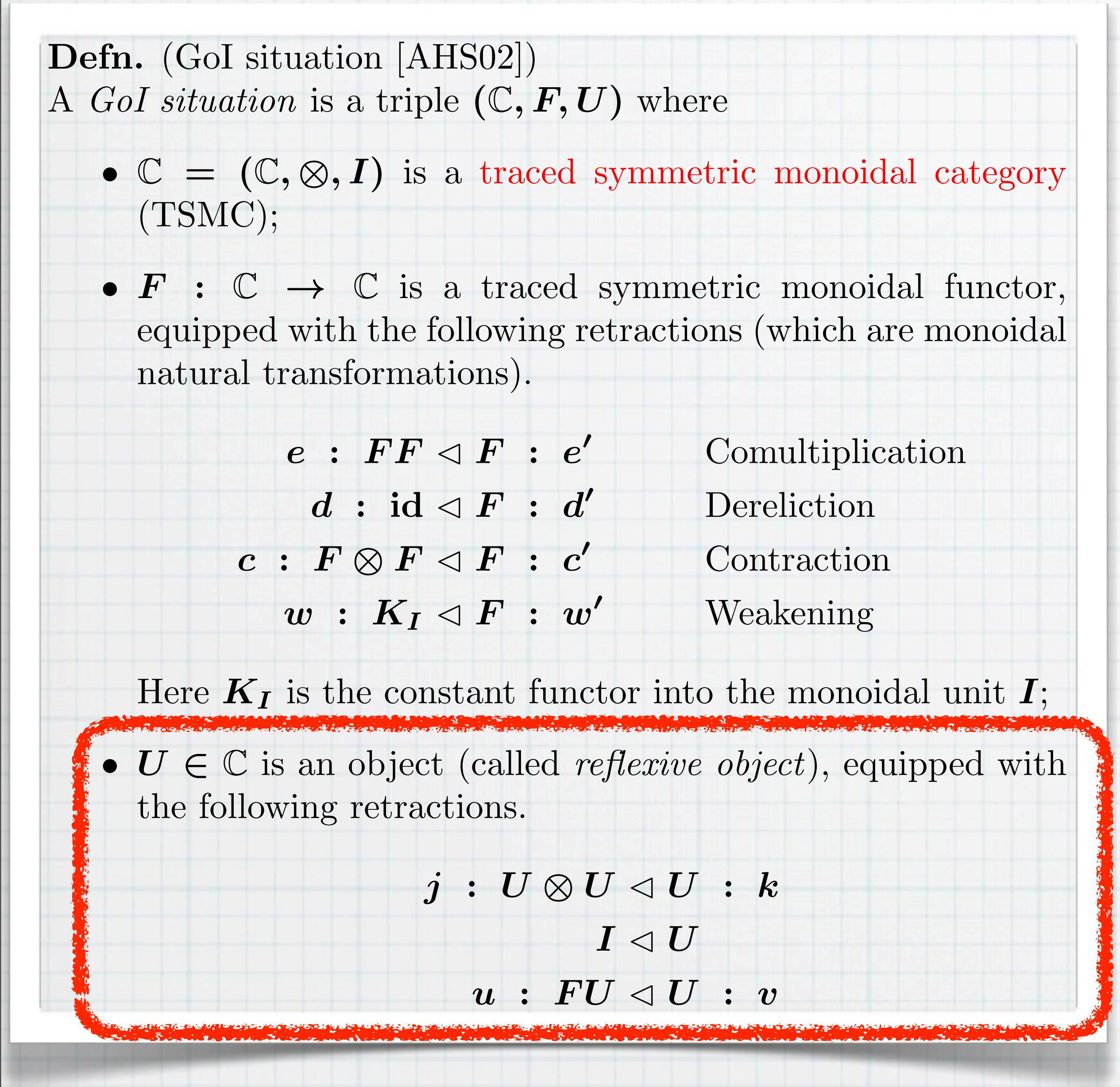
"embedding"

"projection"

such that $g \circ f = \mathrm{id}_X$.

- Functor F
  - For obtaining $! : A \to A$

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

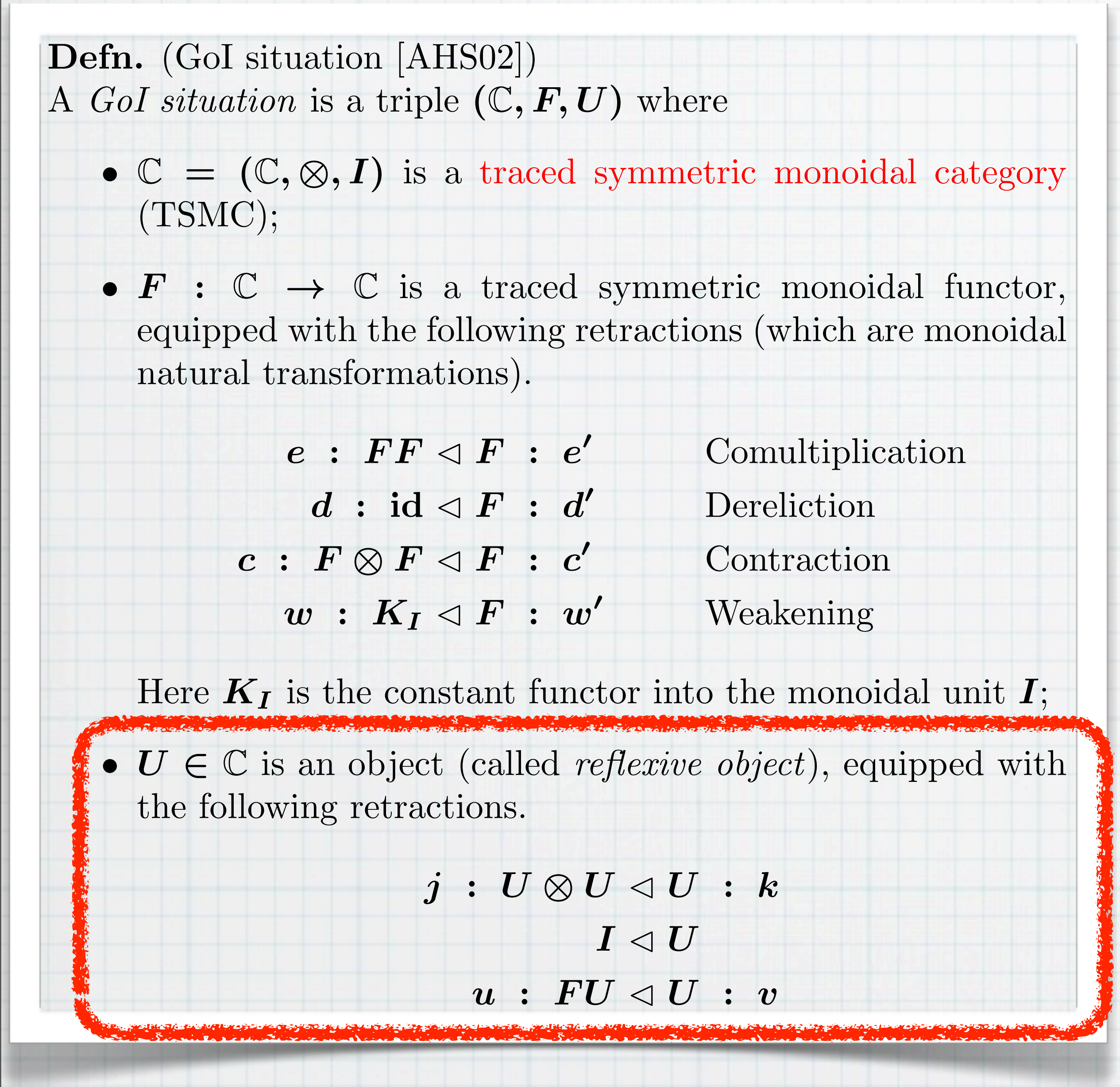
  $$\begin{array}{rcll} \boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} & \lhd & \boldsymbol{F} : \boldsymbol{e}' & \text{Comultiplication} \\ \boldsymbol{d} : \mathbf{id} & \lhd & \boldsymbol{F} : \boldsymbol{d}' & \text{Dereliction} \\ \boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} & \lhd & \boldsymbol{F} : \boldsymbol{c}' & \text{Contraction} \\ \boldsymbol{w} : \boldsymbol{K_I} & \lhd & \boldsymbol{F} : \boldsymbol{w}' & \text{Weakening} \end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

  $$\begin{array}{rcl} \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} & \lhd & \boldsymbol{U} : \boldsymbol{k} \\ \boldsymbol{I} & \lhd & \boldsymbol{U} \\ \boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} & \lhd & \boldsymbol{U} : \boldsymbol{v} \end{array}$$

- The reflexive object U
- Retr. $$\begin{tikzcd} \boldsymbol{U} \otimes \boldsymbol{U} \arrow[r, bend left, "\boldsymbol{j}"] & \boldsymbol{U} \arrow[l, bend left, "\boldsymbol{k}"] \end{tikzcd}$$

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

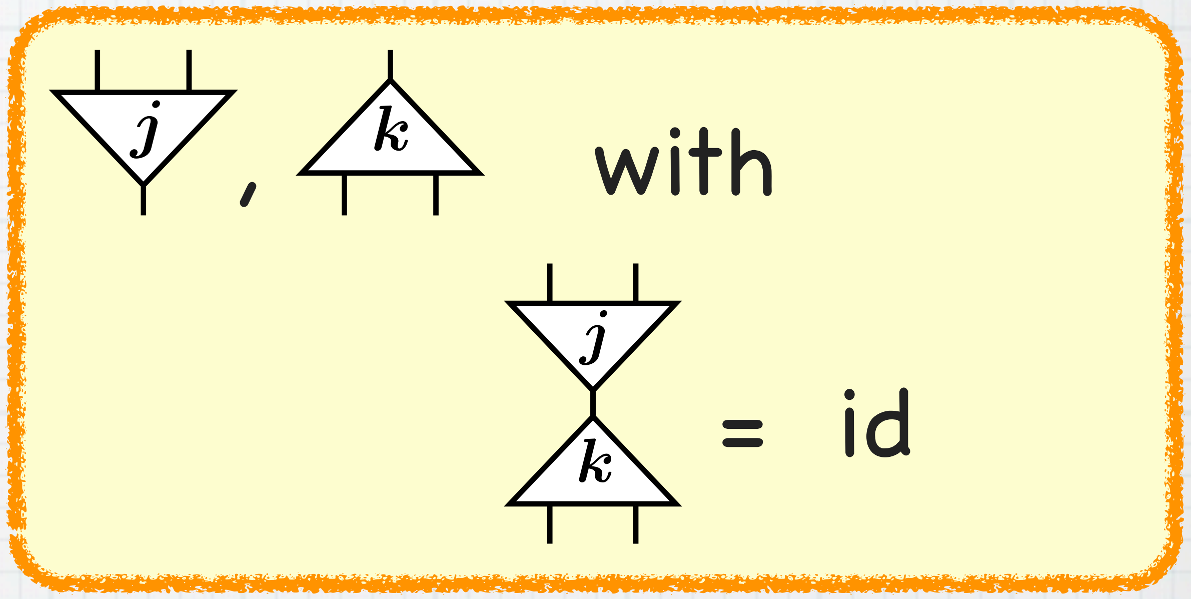
$$\begin{array}{rcrcl l}
e & : & FF \lhd F & : & e' & \text{Comultiplication} \\
d & : & \mathrm{id} \lhd F & : & d' & \text{Dereliction} \\
c & : & F \otimes F \lhd F & : & c' & \text{Contraction} \\
w & : & K_I \lhd F & : & w' & \text{Weakening}
\end{array}$$

Here K_I is the constant functor into the monoidal unit I;

- $U \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{rcrcl}
j & : & U \otimes U \lhd U & : & k \\
 & & I \lhd U & & \\
u & : & FU \lhd U & : & v
\end{array}$$

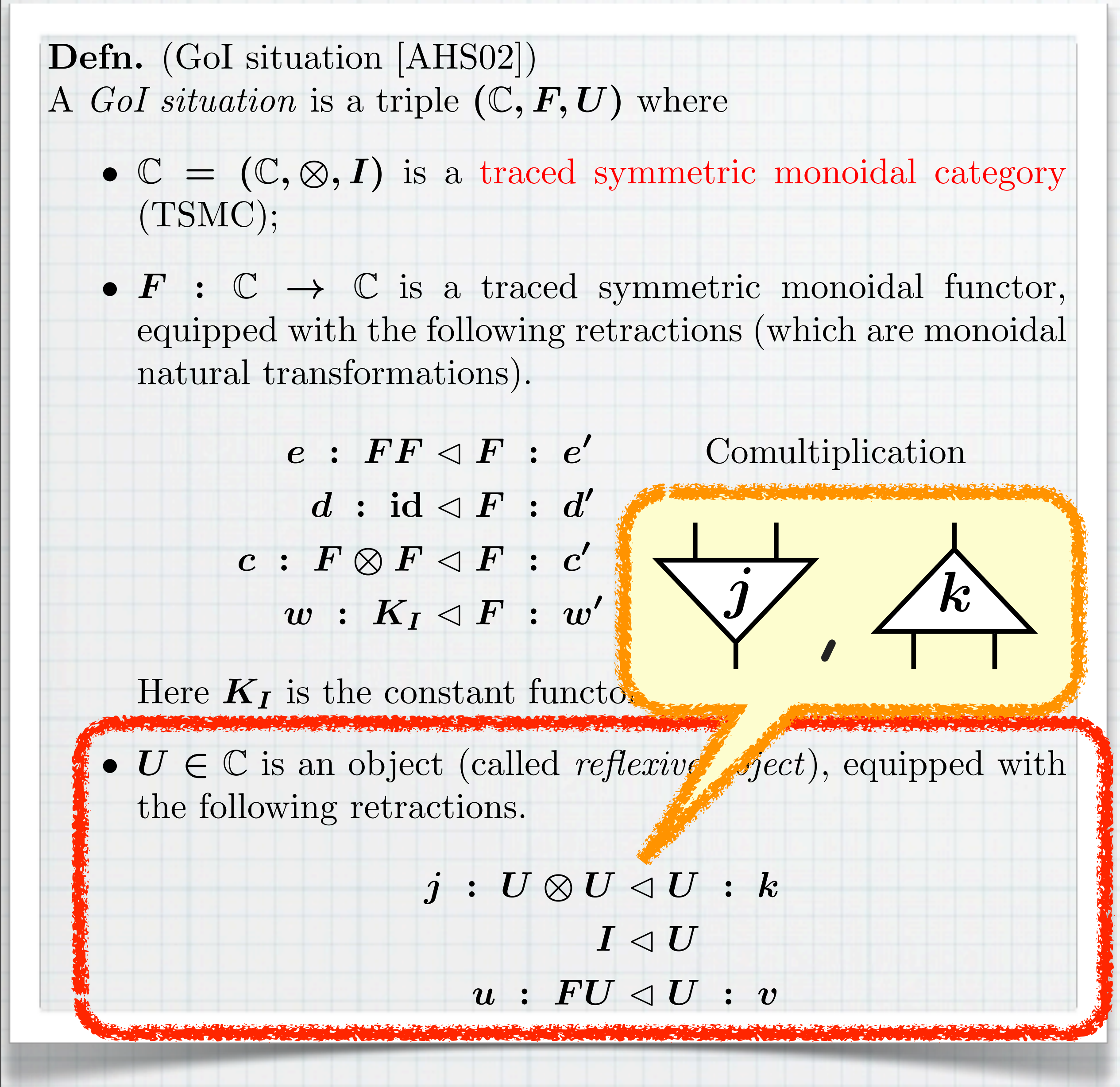
- The reflexive object U
- Retr. $$\begin{tikzcd} U \otimes U \arrow[r, bend left, "j"] & U \arrow[l, bend left, "k"] \end{tikzcd}$$



, with

= id

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{aligned} \boldsymbol{e} &: \boldsymbol{FF} \lhd \boldsymbol{F} : \boldsymbol{e}' \qquad \text{Comultiplication} \\ \boldsymbol{d} &: \mathbf{id} \lhd \boldsymbol{F} : \boldsymbol{d}' \\ \boldsymbol{c} &: \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c}' \\ \boldsymbol{w} &: \boldsymbol{K_I} \lhd \boldsymbol{F} : \boldsymbol{w}' \end{aligned}$$

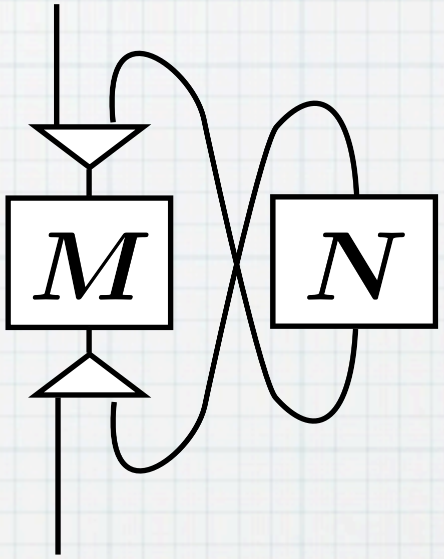
Here $\boldsymbol{K_I}$ is the constant functo

- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{aligned} \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} &\lhd \boldsymbol{U} : \boldsymbol{k} \\ \boldsymbol{I} &\lhd \boldsymbol{U} \\ \boldsymbol{u} : \boldsymbol{FU} &\lhd \boldsymbol{U} : \boldsymbol{v} \end{aligned}$$

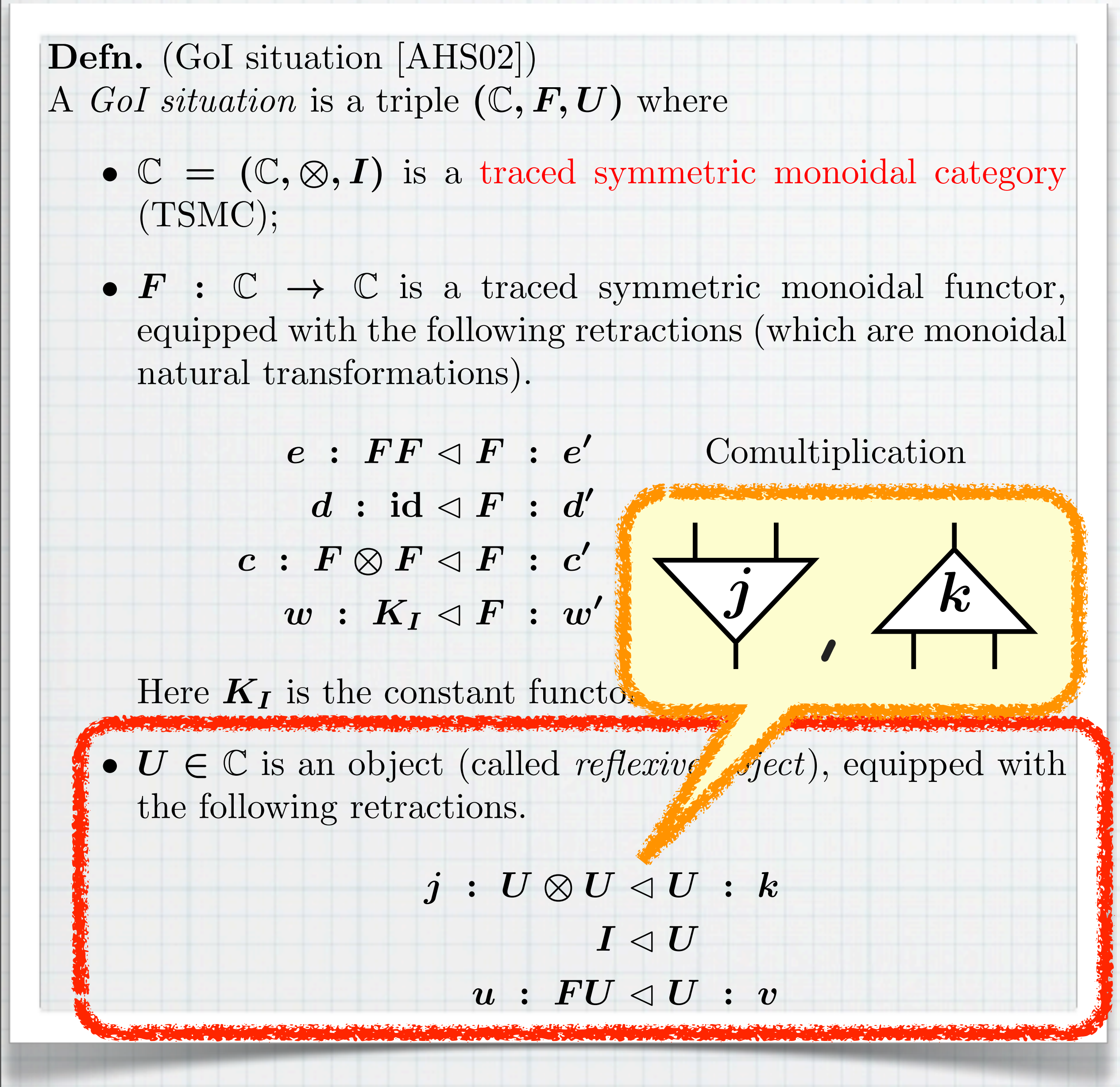
- The reflexive object U

- Why for GoI?



- Example in **Pfn**:

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

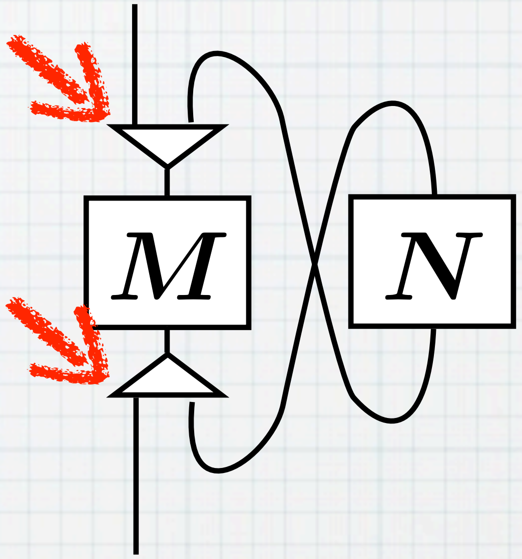
$$\begin{aligned} e &: FF \lhd F : e' && \text{Comultiplication} \\ d &: \mathrm{id} \lhd F : d' \\ c &: F \otimes F \lhd F : c' \\ w &: K_I \lhd F : w' \end{aligned}$$

Here K_I is the constant functo

- $U \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

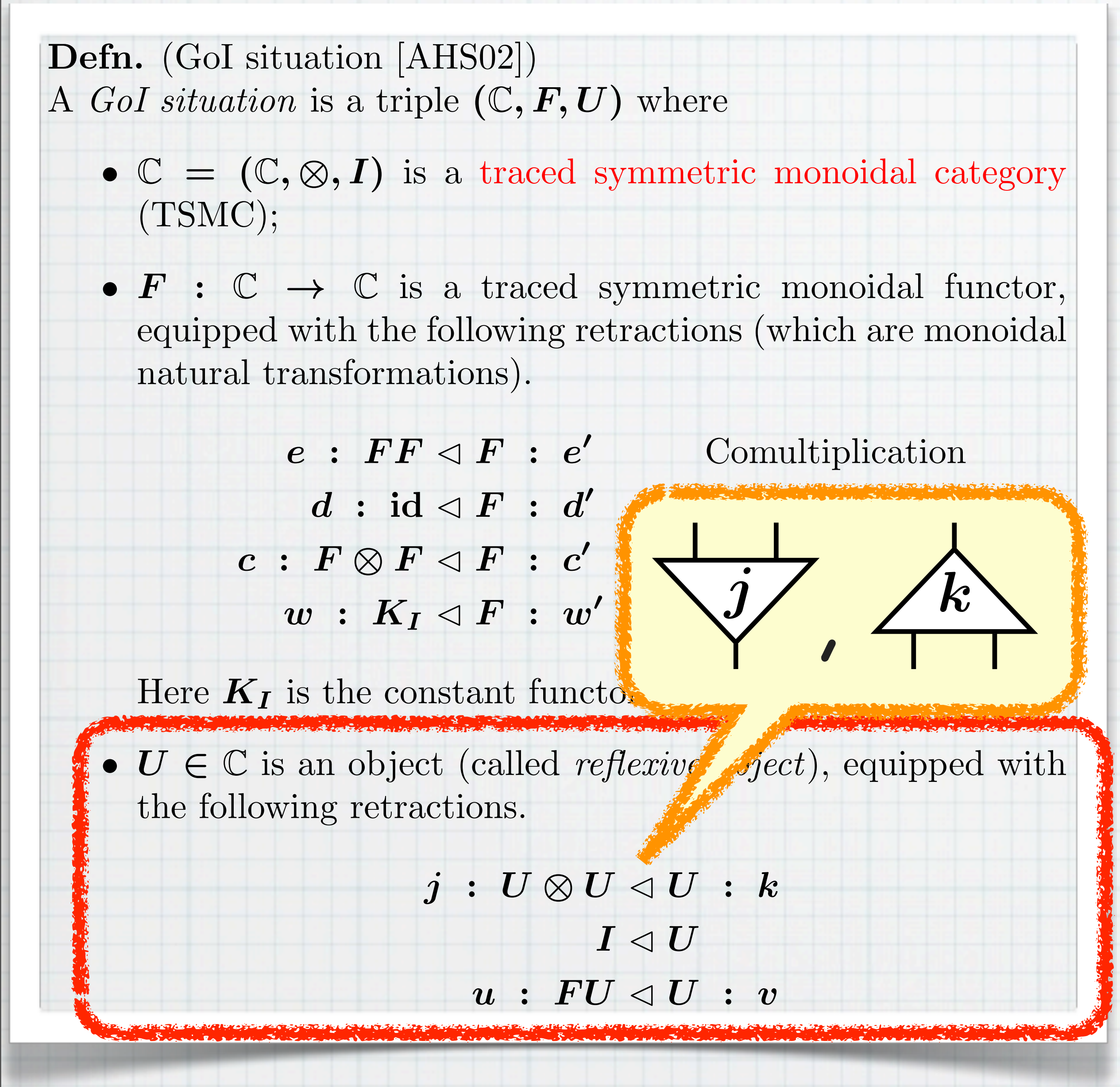
$$\begin{aligned} j : U \otimes U &\lhd U : k \\ I &\lhd U \\ u : FU &\lhd U : v \end{aligned}$$

- The reflexive object U
- Why for GoI?



- Example in **Pfn**:

GoI situation



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

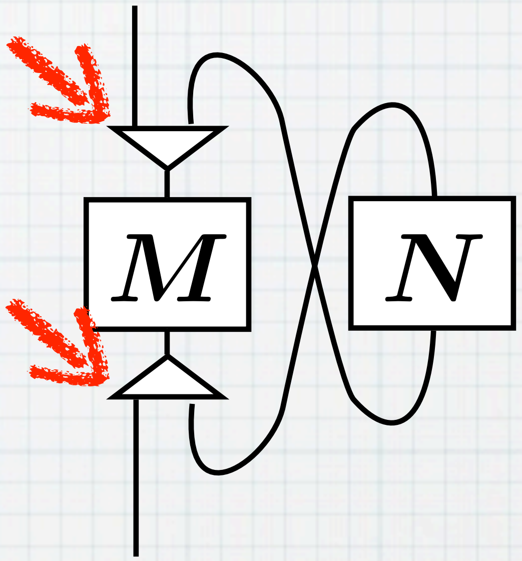
$$\begin{array}{rcrcl} \boldsymbol{e} & : & \boldsymbol{FF} \lhd \boldsymbol{F} & : & \boldsymbol{e'} \qquad \text{Comultiplication} \\ \boldsymbol{d} & : & \textbf{id} \lhd \boldsymbol{F} & : & \boldsymbol{d'} \\ \boldsymbol{c} & : & \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} & : & \boldsymbol{c'} \\ \boldsymbol{w} & : & \boldsymbol{K_I} \lhd \boldsymbol{F} & : & \boldsymbol{w'} \end{array}$$

Here $\boldsymbol{K_I}$ is the constant functo

- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{rcrcl} \boldsymbol{j} & : & \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} & : & \boldsymbol{k} \\ & & \boldsymbol{I} \lhd \boldsymbol{U} & & \\ \boldsymbol{u} & : & \boldsymbol{FU} \lhd \boldsymbol{U} & : & \boldsymbol{v} \end{array}$$

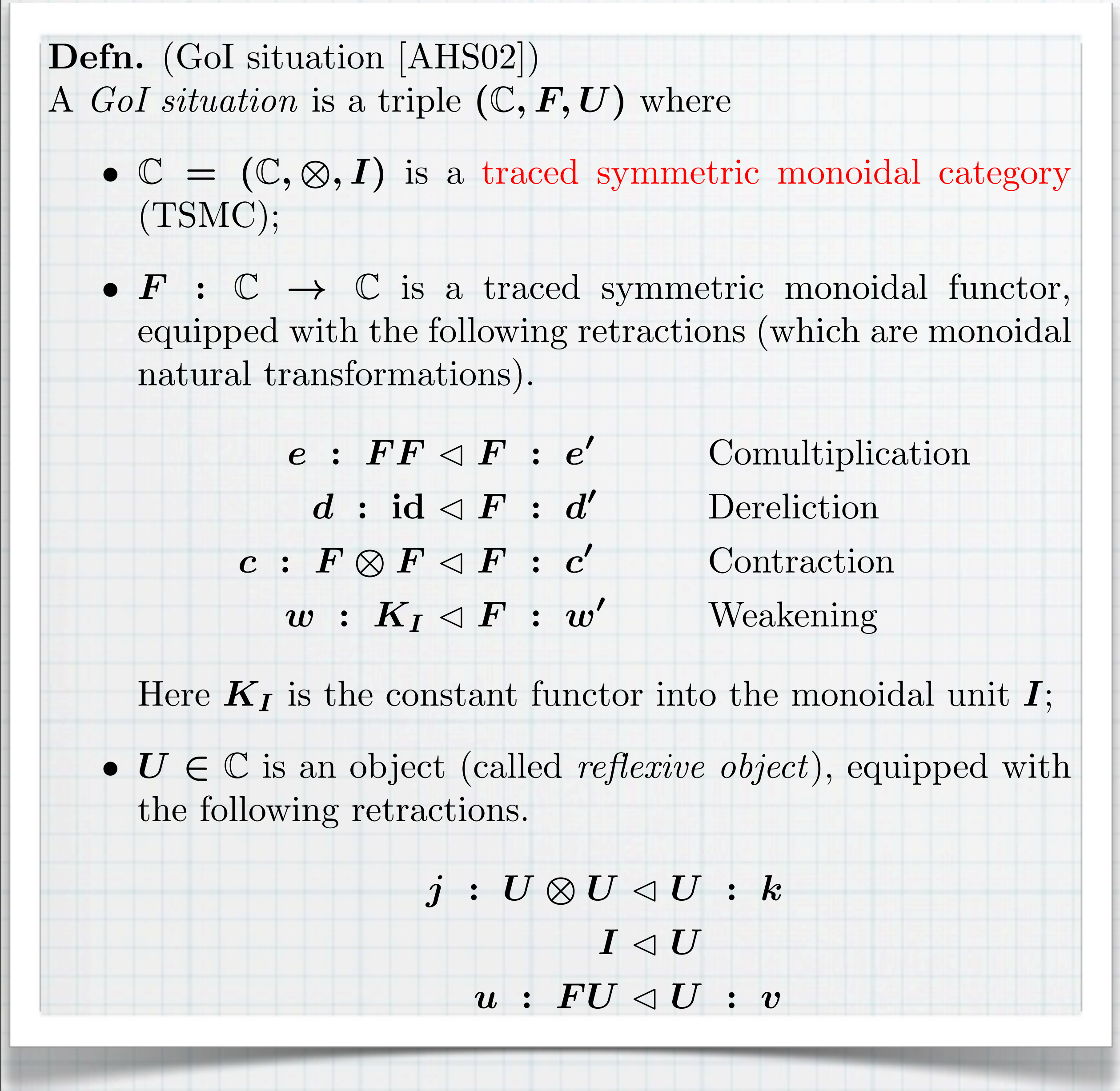
- The reflexive object U
  - Why for GoI?



  - Example in **Pfn**:

$$\mathbb{N} \in \mathbf{Pfn}, \quad \text{with}$$
$$\mathbb{N} + \mathbb{N} \cong \mathbb{N},$$
$$\mathbb{N} \cdot \mathbb{N} \cong \mathbb{N}$$

GoI Situation: Summary



Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

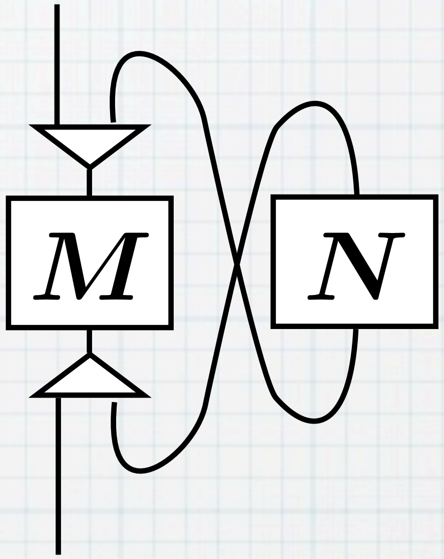
- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{array}{rcll} \boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{e}' & & & \text{Comultiplication} \\ \boldsymbol{d} : \textbf{id} \lhd \boldsymbol{F} : \boldsymbol{d}' & & & \text{Dereliction} \\ \boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c}' & & & \text{Contraction} \\ \boldsymbol{w} : \boldsymbol{K_I} \lhd \boldsymbol{F} : \boldsymbol{w}' & & & \text{Weakening} \end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{c} \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{k} \\ \boldsymbol{I} \lhd \boldsymbol{U} \\ \boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{v} \end{array}$$

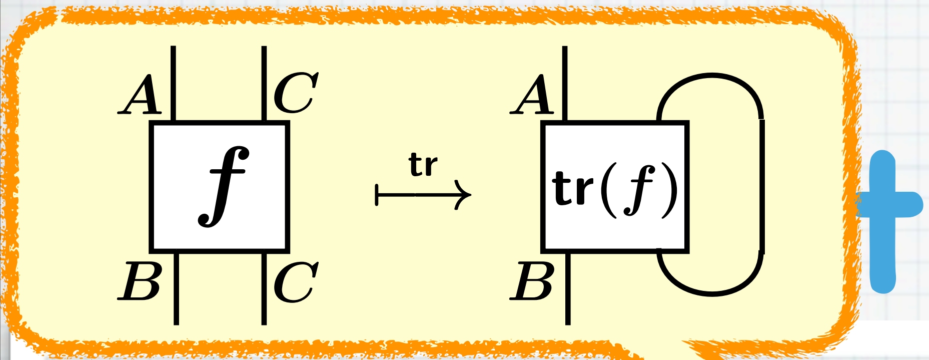
- Categorical axiomatics of the "GoI animation"

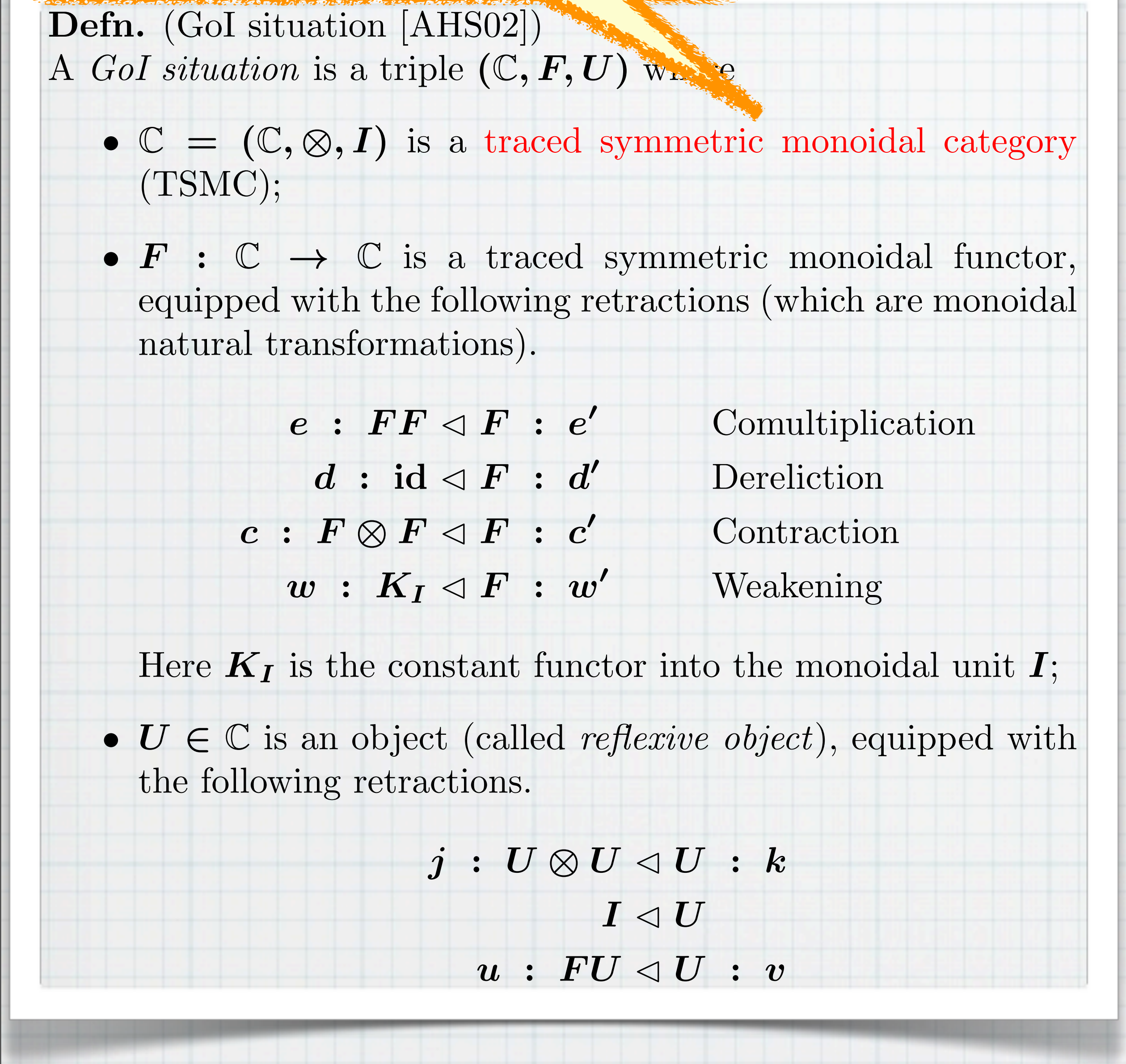


- Example:

$$(\mathbf{Pfn}, \mathbb{N} \cdot \_\,, \mathbb{N})$$

tuation: Summary





Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

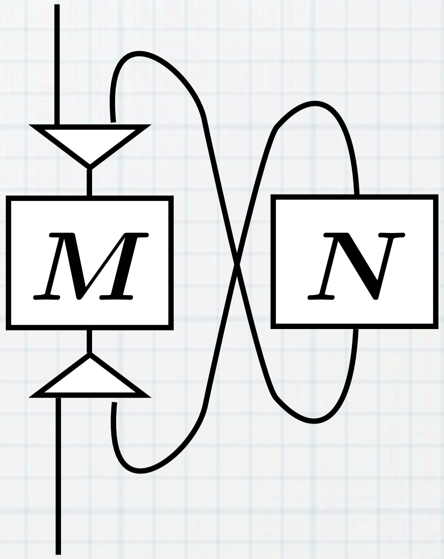
$$\begin{array}{rll} \boldsymbol{e} : \boldsymbol{F}\boldsymbol{F} \triangleleft \boldsymbol{F} : \boldsymbol{e}' & & \text{Comultiplication} \\ \boldsymbol{d} : \mathrm{id} \triangleleft \boldsymbol{F} : \boldsymbol{d}' & & \text{Dereliction} \\ \boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \triangleleft \boldsymbol{F} : \boldsymbol{c}' & & \text{Contraction} \\ \boldsymbol{w} : \boldsymbol{K_I} \triangleleft \boldsymbol{F} : \boldsymbol{w}' & & \text{Weakening} \end{array}$$

Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;

- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

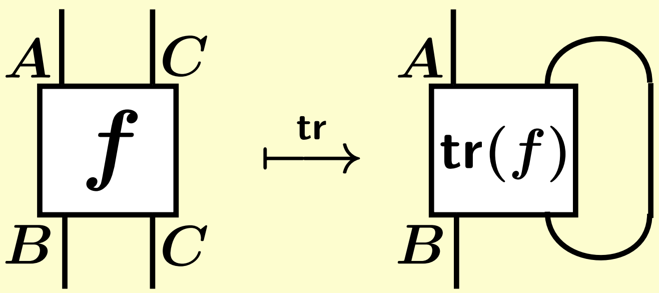
$$\begin{array}{c} \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \triangleleft \boldsymbol{U} : \boldsymbol{k} \\ \boldsymbol{I} \triangleleft \boldsymbol{U} \\ \boldsymbol{u} : \boldsymbol{F}\boldsymbol{U} \triangleleft \boldsymbol{U} : \boldsymbol{v} \end{array}$$

- Categorical axiomatics of the "GoI animation"



- Example:

$$(\mathbf{Pfn}, \mathbb{N} \cdot \_\,, \mathbb{N})$$



tuation: Summary

Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{aligned} e &: FF \lhd F : e' \\ d &: \mathrm{id} \lhd F : d' \\ c &: F \otimes F \lhd F : c' \\ w &: K_I \lhd F : w' \end{aligned}$$

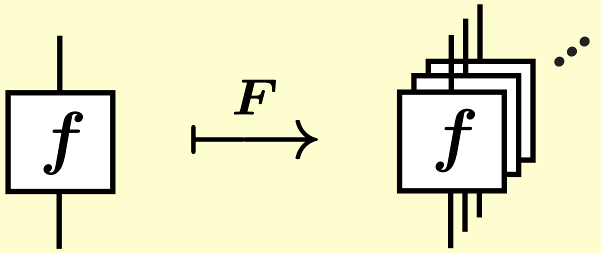
De
Co
We

Here K_I is the constant functor into th

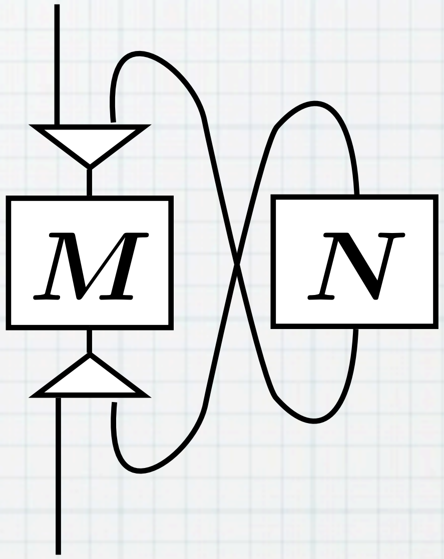
- $U \in \mathbb{C}$ is an object (called *reflexive obj* the following retractions.

$$\begin{aligned} j : U \otimes U &\lhd U : k \\ I &\lhd U \\ u : FU &\lhd U : v \end{aligned}$$

For ! , via



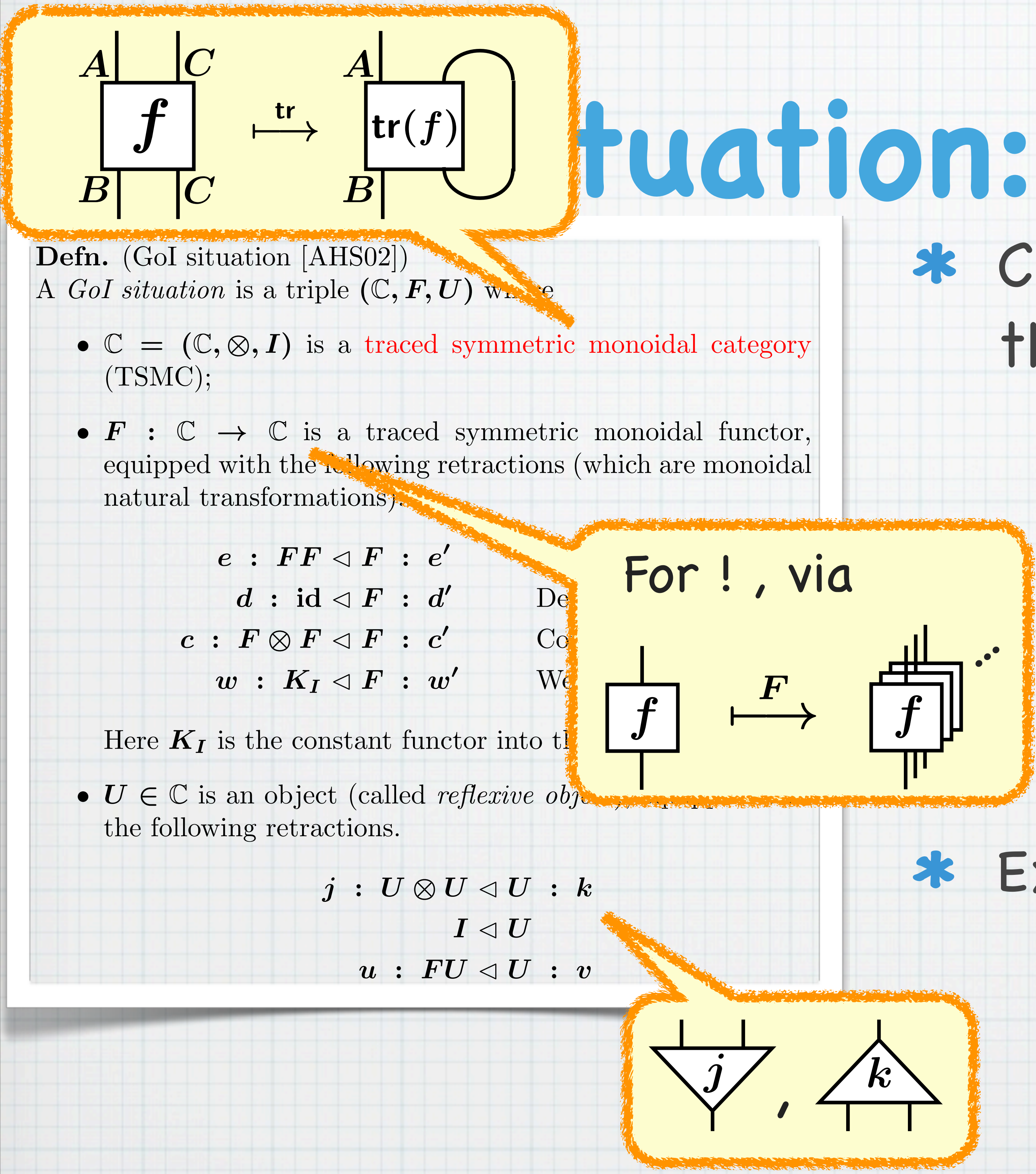
- Categorical axiomatics of the "GoI animation"



- Example:

$$(\mathbf{Pfn}, \mathbb{N} \cdot \_\,, \mathbb{N})$$

tuation: Summary



A
C
f
B
C
tr
A
tr(f)
B

Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations):

$$\begin{aligned} e &: FF \lhd F : e' \\ d &: \mathrm{id} \lhd F : d' \qquad \text{De} \\ c &: F \otimes F \lhd F : c' \qquad \text{Co} \\ w &: K_I \lhd F : w' \qquad \text{We} \end{aligned}$$

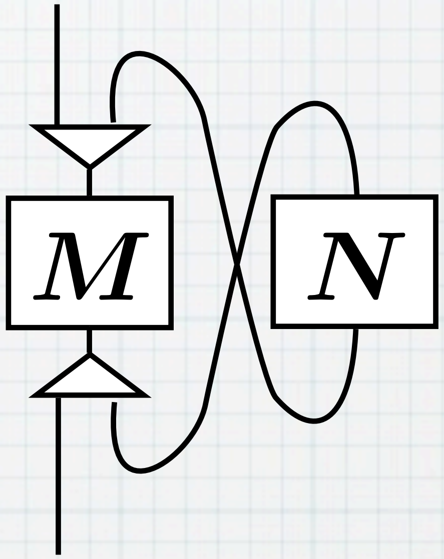
Here K_I is the constant functor into th

- $U \in \mathbb{C}$ is an object (called *reflexive obj* the following retractions.

$$\begin{aligned} j : U \otimes U &\lhd U : k \\ I &\lhd U \\ u : FU &\lhd U : v \end{aligned}$$

For ! , via

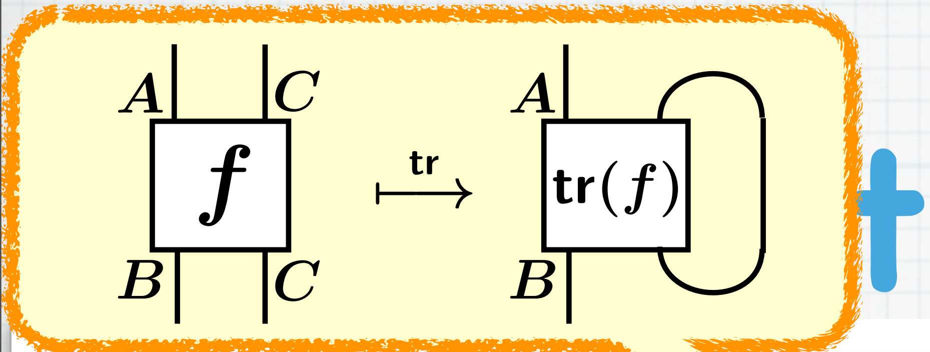
- Categorical axiomatics of the "GoI animation"

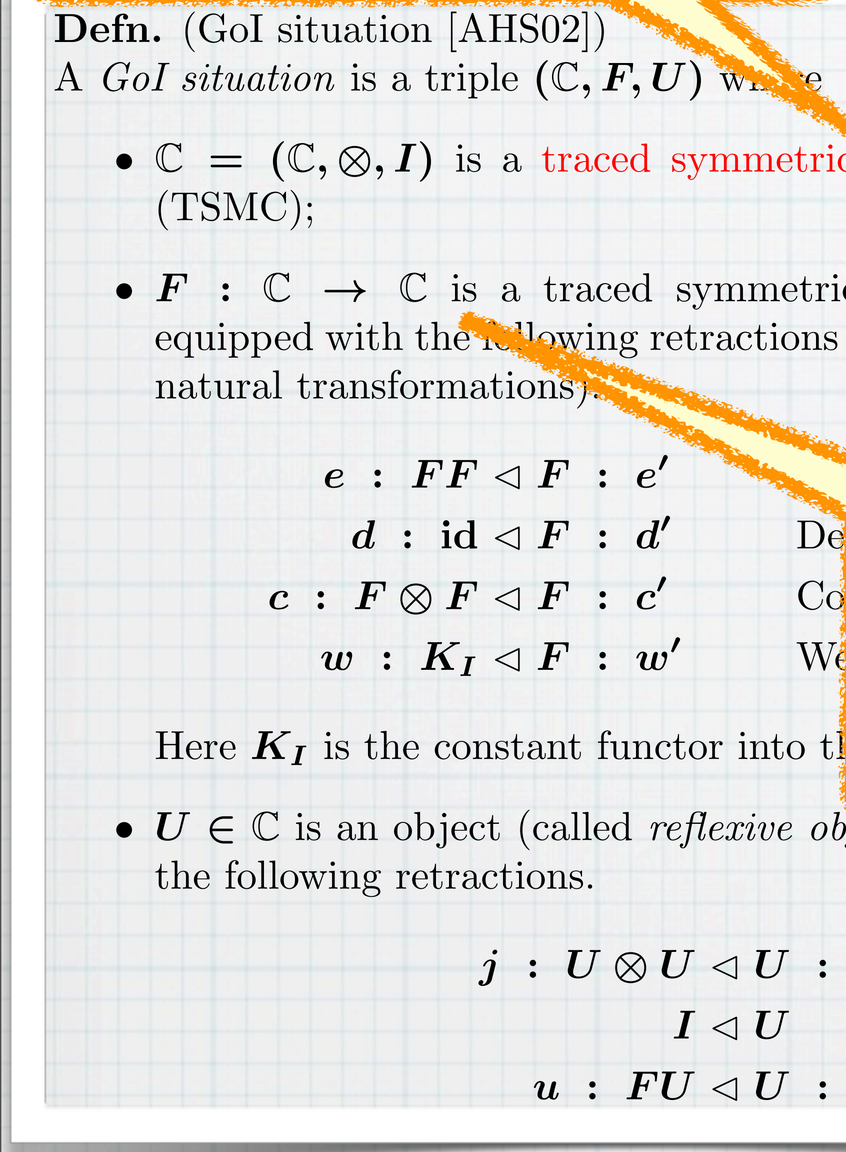


- Example:

$$(\mathbf{Pfn}, \mathbb{N} \cdot \_, \mathbb{N})$$

tuation: Summary





Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, F, U)$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations):

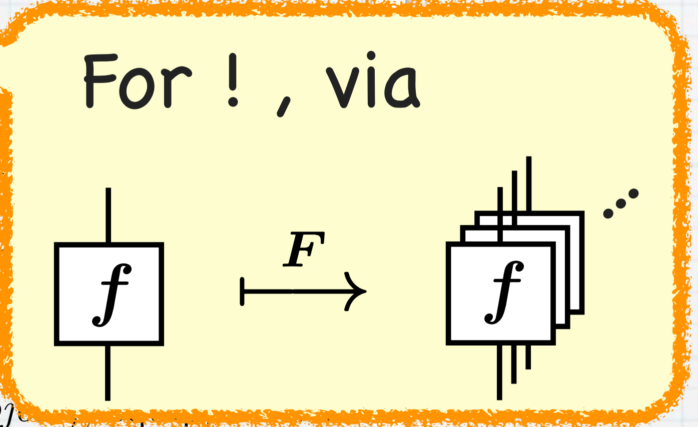
$$e : FF \lhd F : e'$$
$$d : \mathrm{id} \lhd F : d' \qquad \text{De}$$
$$c : F \otimes F \lhd F : c' \qquad \text{Co}$$
$$w : K_I \lhd F : w' \qquad \text{We}$$

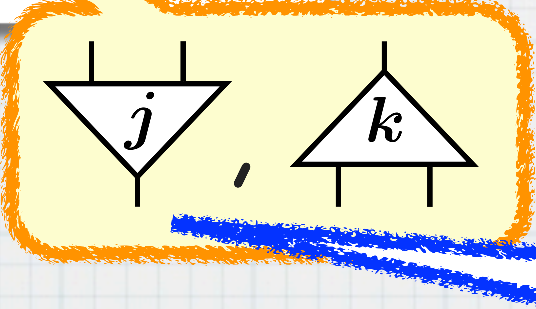
Here K_I is the constant functor into t

- $U \in \mathbb{C}$ is an object (called *reflexive obj* the following retractions.

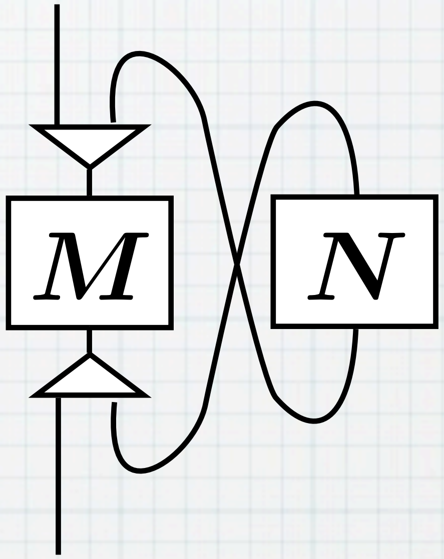
$$j : U \otimes U \lhd U : k$$
$$I \lhd U$$
$$u : FU \lhd U : v$$

For ! , via



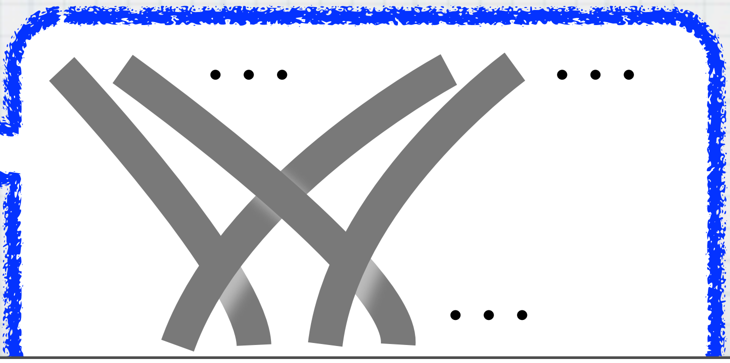


- Categorical axiomatics of the "GoI animation"



- Example:

$$(\mathbf{Pfn}, \mathbb{N} \cdot \_\, , \mathbb{N})$$



Categorical GoI: Constr. of an LCA

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

Categorical GoI: Constr. of an LCA

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

- Applicative str. $\cdot$
- ! operator
- Combinators B, C, I, ...

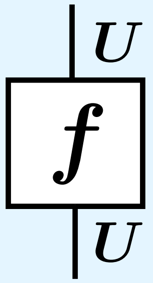
Categorical GoI: Constr. of an LCA

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

- Applicative str. $\cdot$
- ! operator
- Combinators B, C, I, ...



$\in \mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$

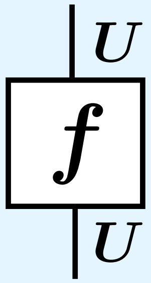
Categorical GoI: Constr. of an LCA

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

- Applicative str. $\cdot$
- ! operator
- Combinators B, C, I, ...

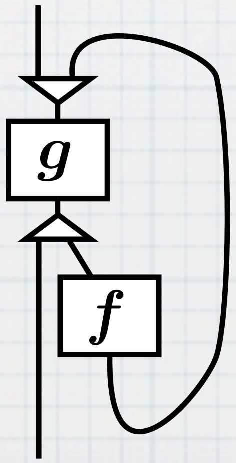


$\in \mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$

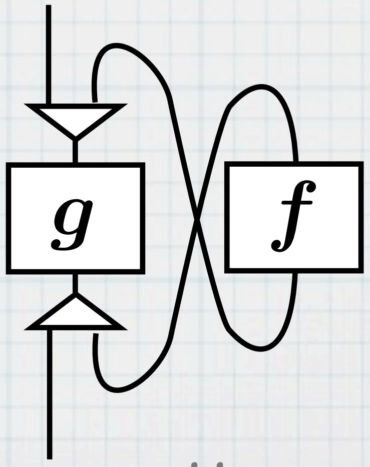
- $\boldsymbol{g} \cdot \boldsymbol{f}$

$$:= \mathbf{tr}\Big((\boldsymbol{U} \otimes \boldsymbol{f}) \circ \boldsymbol{k} \circ \boldsymbol{g} \circ \boldsymbol{j} \Big)$$

=



=



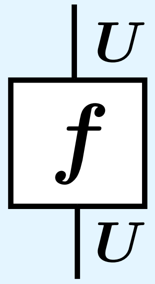
Categorical GoI: Constr. of an LCA

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

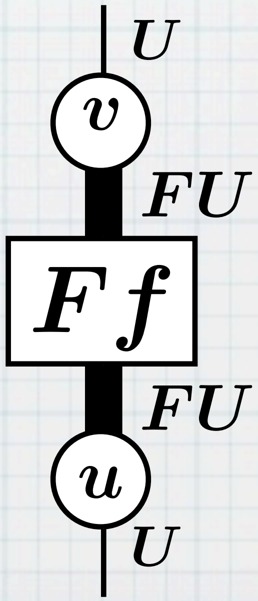
- Applicative str. ·
- ! operator
- Combinators B, C, I, ...



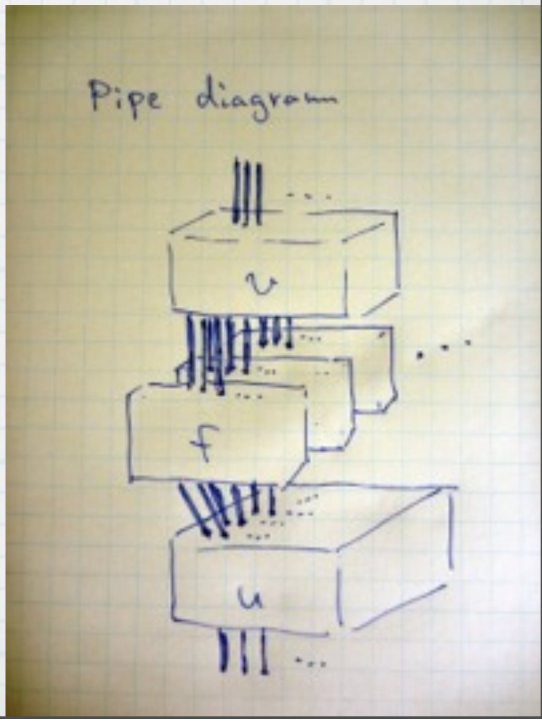
$\in \mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$

- $!\boldsymbol{f} \;:=\; \boldsymbol{u} \circ \boldsymbol{F}\boldsymbol{f} \circ \boldsymbol{v}$

=



=



Categorical GoI: Constr. of an LCA

- Combinator $Bxyz = x(yz)$

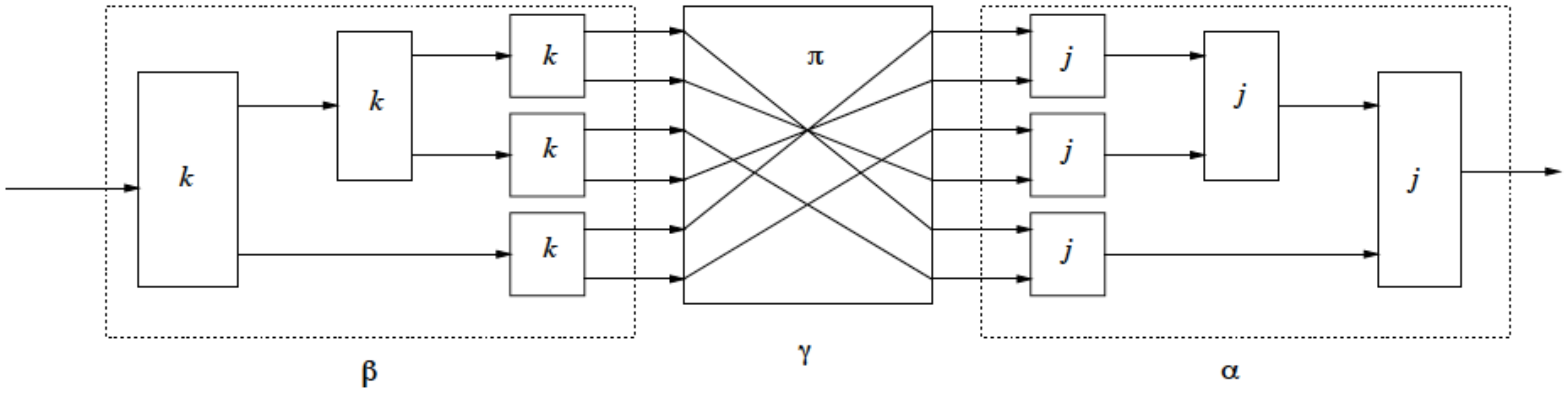
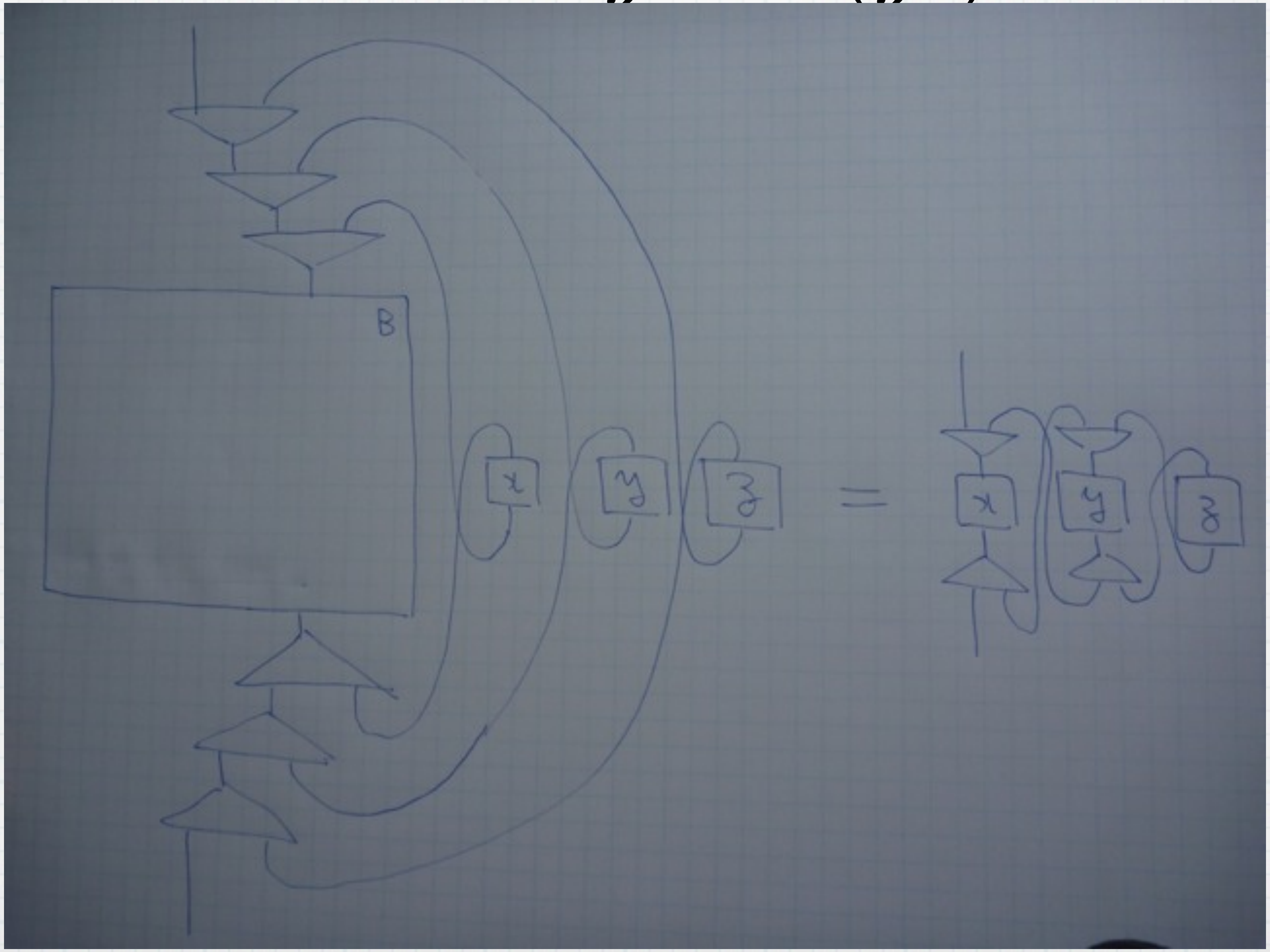


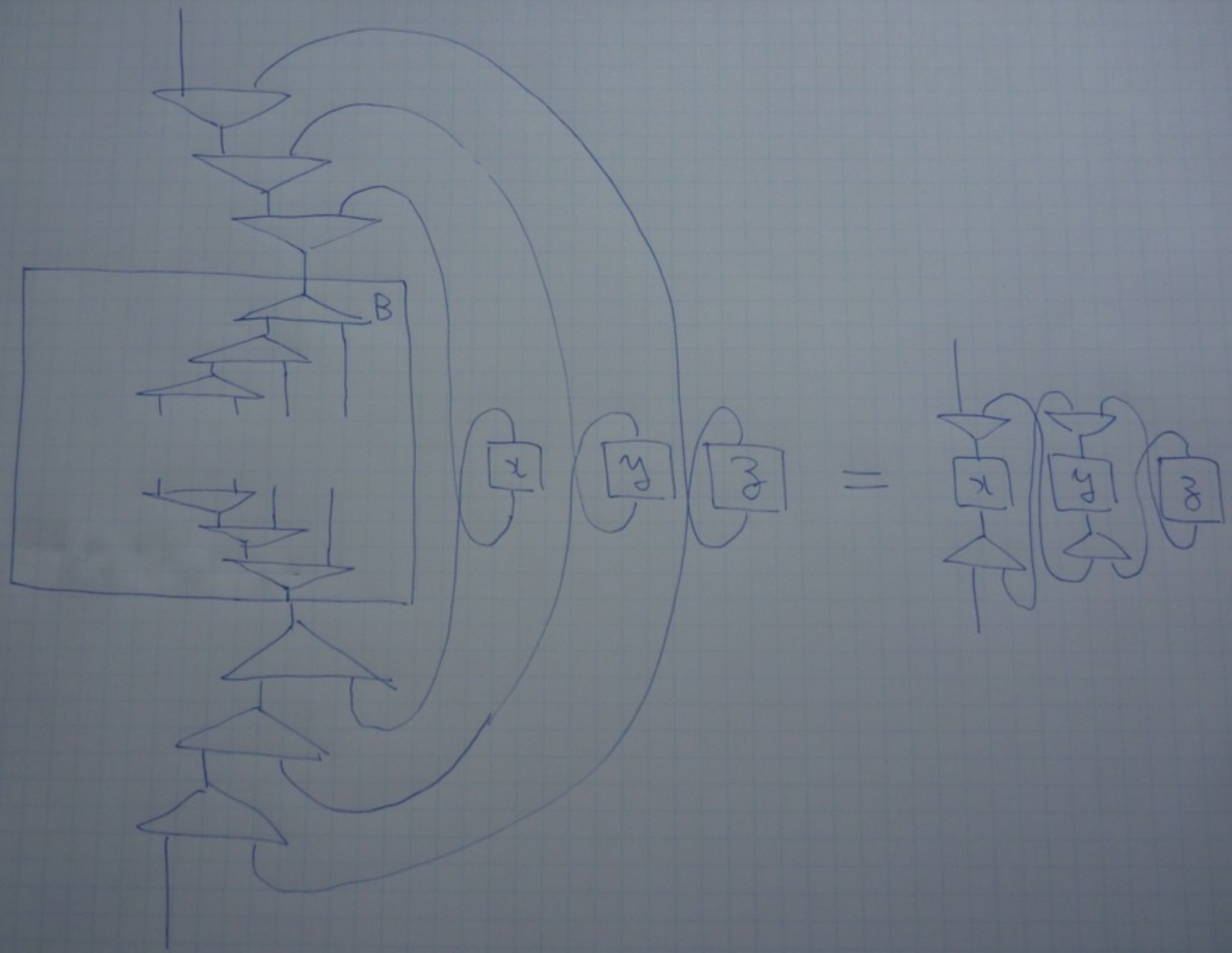
Figure 7: Composition Combinator B

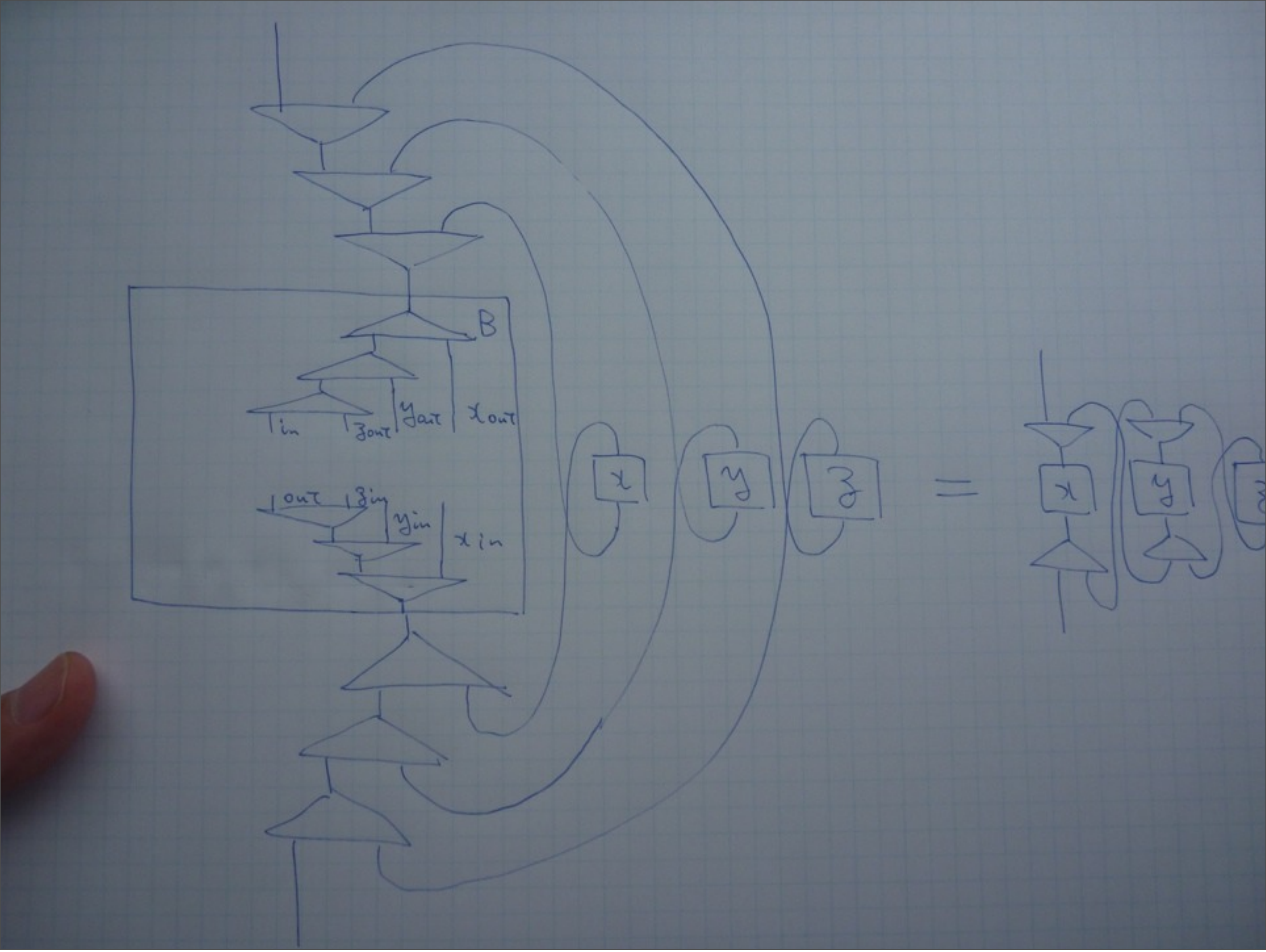
from [AHS02]

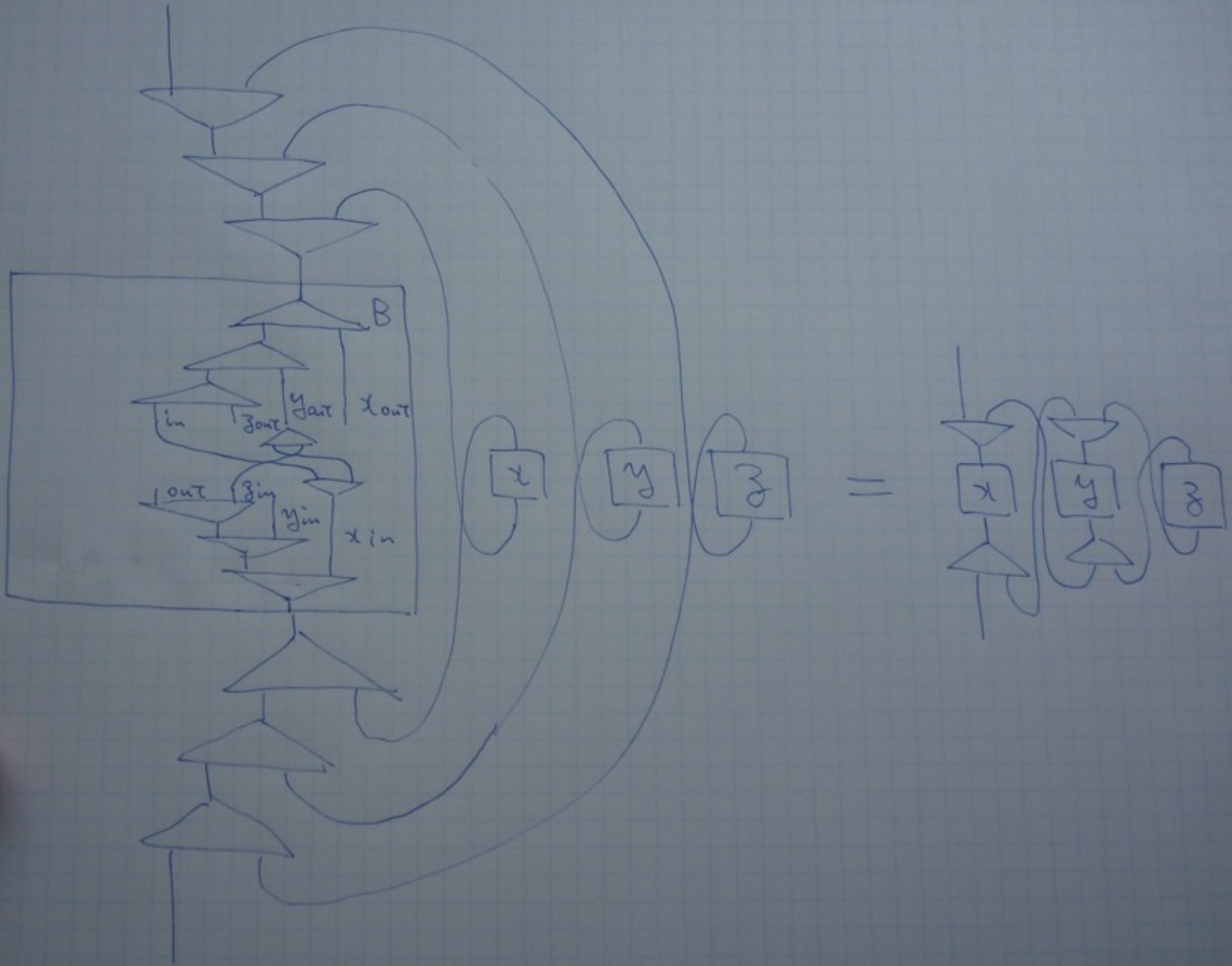
Categorical GoI: Constr. of an LCA

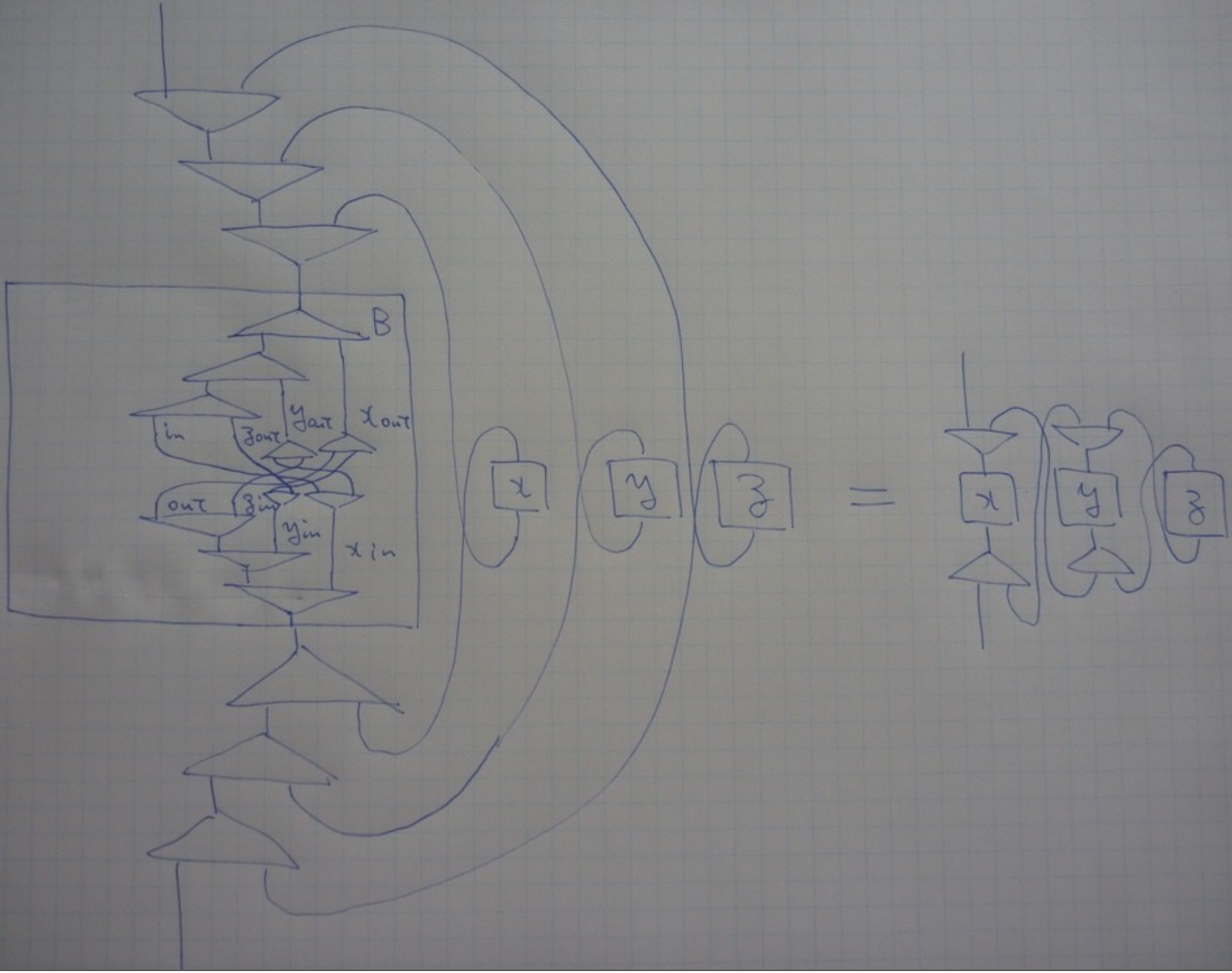
- Combinator $Bxyz = x(yz)$




B
x
y
z
=
x
y
z


B
in
zout
yout
xout
out
zin
yin
xin
x
y
z
=
x
y
z


B
in
zout
yout
xout
out
zin
yin
xin
x
y
z
=
x
y
z


B
in
z out
y out
x out
out
z in
y in
x in
x
y
z
=
x
y
z

Categorical GoI: Constr. of an LCA

- Combinator $Bxyz = x(yz)$

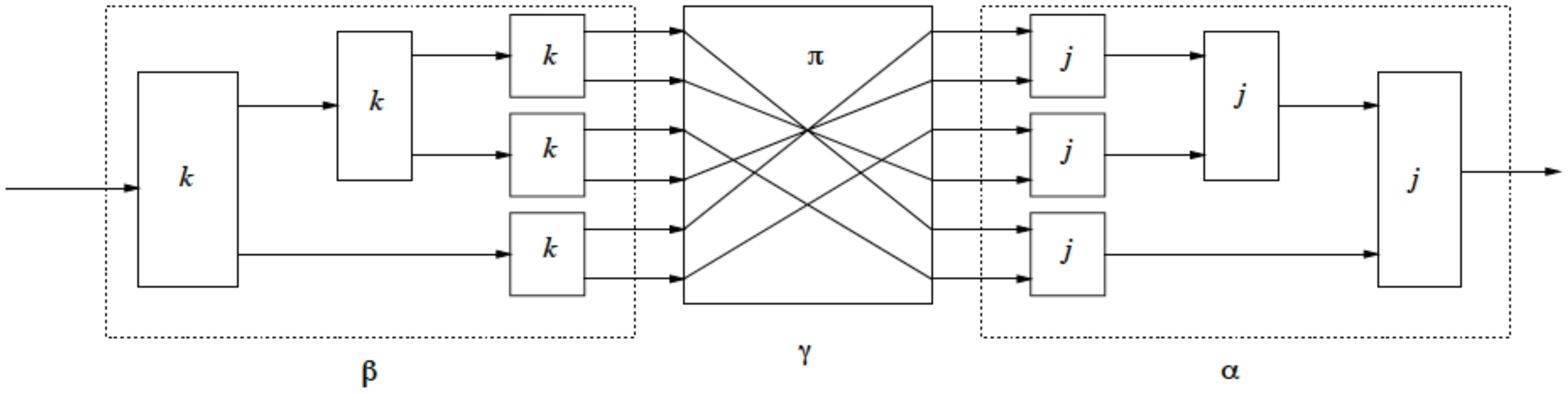


Figure 7: Composition Combinator B

from [AHS02]

Categorical GoI: Constr. of an LCA

- Combinator $Bxyz = x(yz)$

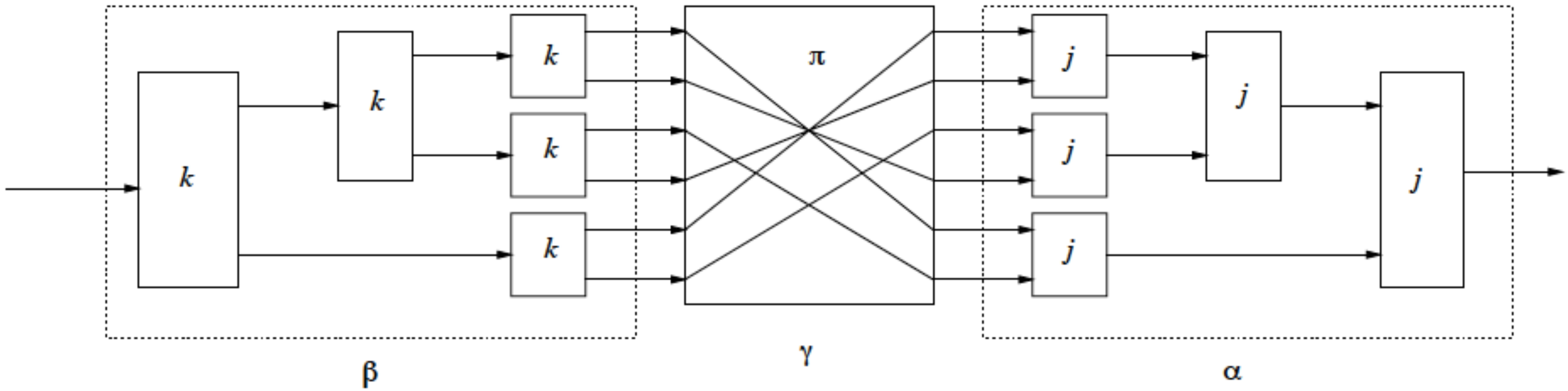


Figure 7: Composition Combinator B

from [AHS02]

Nice dynamic interpretation of (linear) computation!!

Summary: Categorical GoI

Defn. (GoI situation [AHS02])
A *GoI situation* is a triple $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$ where

- $\mathbb{C} = (\mathbb{C}, \otimes, \boldsymbol{I})$ is a traced symmetric monoidal category (TSMC);
- $\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).

$$\begin{array}{rcll} \boldsymbol{e} : \boldsymbol{FF} \lhd \boldsymbol{F} : \boldsymbol{e'} & & & \text{Comultiplication} \\ \boldsymbol{d} : \mathbf{id} \lhd \boldsymbol{F} : \boldsymbol{d'} & & & \text{Dereliction} \\ \boldsymbol{c} : \boldsymbol{F} \otimes \boldsymbol{F} \lhd \boldsymbol{F} : \boldsymbol{c'} & & & \text{Contraction} \\ \boldsymbol{w} : \boldsymbol{K_I} \lhd \boldsymbol{F} : \boldsymbol{w'} & & & \text{Weakening} \end{array}$$

  Here $\boldsymbol{K_I}$ is the constant functor into the monoidal unit $\boldsymbol{I}$;
- $\boldsymbol{U} \in \mathbb{C}$ is an object (called *reflexive object*), equipped with the following retractions.

$$\begin{array}{c} \boldsymbol{j} : \boldsymbol{U} \otimes \boldsymbol{U} \lhd \boldsymbol{U} : \boldsymbol{k} \\ \boldsymbol{I} \lhd \boldsymbol{U} \\ \boldsymbol{u} : \boldsymbol{FU} \lhd \boldsymbol{U} : \boldsymbol{v} \end{array}$$

Thm. ([AHS02])
Given a GoI situation $(\mathbb{C}, \boldsymbol{F}, \boldsymbol{U})$, the homset

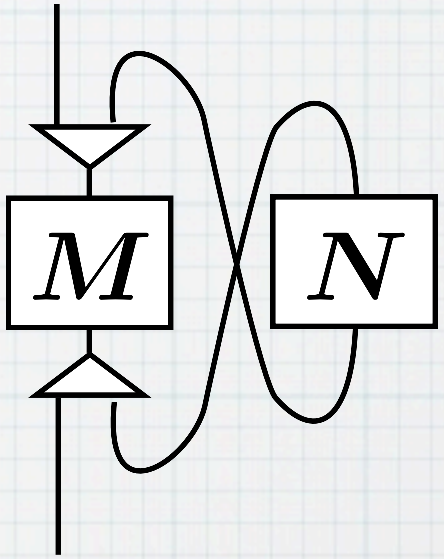
$$\mathbb{C}(\boldsymbol{U}, \boldsymbol{U})$$

carries a canonical LCA structure.

Hasuo (Tokyo)

Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

- Strategy: find a TSMC!



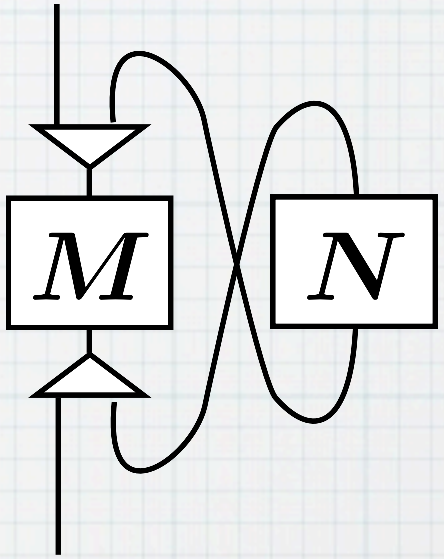
- "Wave-style" examples
  - ⊗ is Cartesian product(-like)
    - in which case,

      trace ≈ fixed point operator [Hasegawa/Hyland]
  - An example: $\big(\,(\omega\text{-}\mathbf{Cpo}, \times, 1),\ (\_)^{\mathbb{N}},\ A^{\mathbb{N}}\,\big)$
  - (... less of a dynamic flavor)

Hasuo (Tokyo)

Why Categorical Generalization?: Examples Other Than $\mathbf{Pfn}$ [AHS02]

- "Particle-style" examples
  - Obj. X∈C is set-like; ⊗ is coproduct-like
    - The GoI animation is valid
  - Examples:
    - Partial functions $\big(\,(\mathbf{Pfn},+,\mathbf{0}),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$
    - Binary relations $\big(\,(\mathbf{Rel},+,\mathbf{0}),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$
    - "Discrete stochastic relations" $\big(\,(\mathbf{DSRel},+,\mathbf{0}),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$



Hasuo (Tokyo)

Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

- **Pfn** (partial functions)

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y, \text{ partial function} \\ \hline\hline X \to \mathcal{L}Y \ \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y$$

- **Rel** (relations)

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y, \text{ relation} \\ \hline\hline X \to \mathcal{P}Y \ \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

- **DSRel**

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \ \text{ in } \mathbf{Sets} \end{array}$$

$$\text{where } \mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_{y} d(y) \leq 1\}$$

Why Catego

Examples Other Than Pfn [AHS02]

Categories of sets and
(functions with different branching/partiality)

- **Pfn** (partial functions)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y \text{, partial function} \\ \hline\hline X \to \mathcal{L}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\perp\} + Y$$

- **Rel** (relations)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y \text{, relation} \\ \hline\hline X \to \mathcal{P}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

- **DSRel**

$$\begin{array}{c} X \to Y \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \text{ in } \mathbf{Sets} \end{array}$$

where $\mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_y d(y) \leq 1\}$

Why Catego
Examples Other Than **Fun** [AHS02]

Categories of sets and
(functions with different branching/partiality)

- **Pfn** (partial functions)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y\text{, partial function} \\ \hline\hline X \to \mathcal{L}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y$$

(Potential) non-termination

- **Rel** (relations)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y\text{, relation} \\ \hline\hline X \to \mathcal{P}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

Non-determinism

- **DSRel**

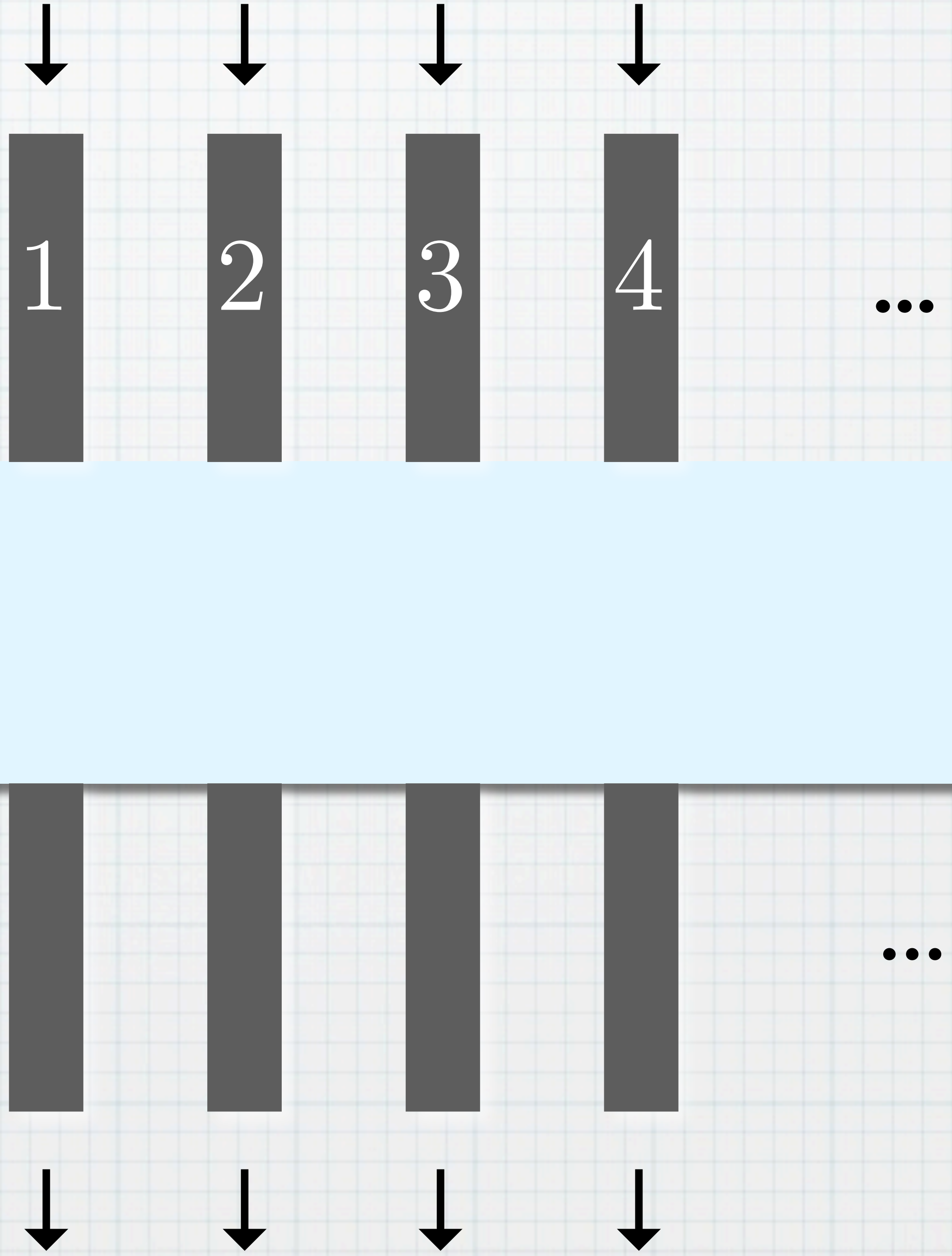
$$\begin{array}{c} X \to Y \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \text{ in } \mathbf{Sets} \end{array}$$

$$\text{where } \mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_{y} d(y) \le 1\}$$

Probabilistic branching

Different Branching in The GoI Animation

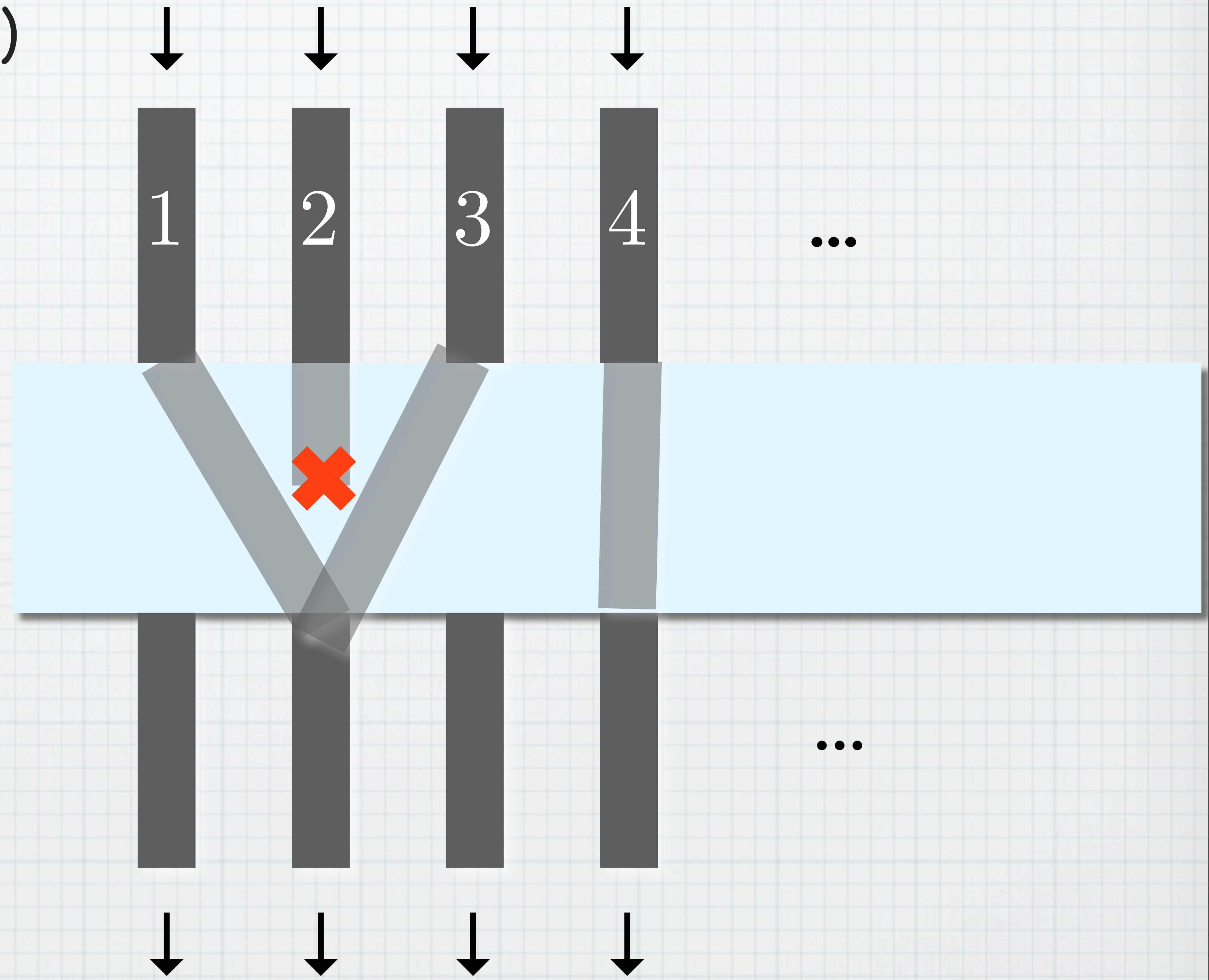
- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

Different Branching in The GoI Animation

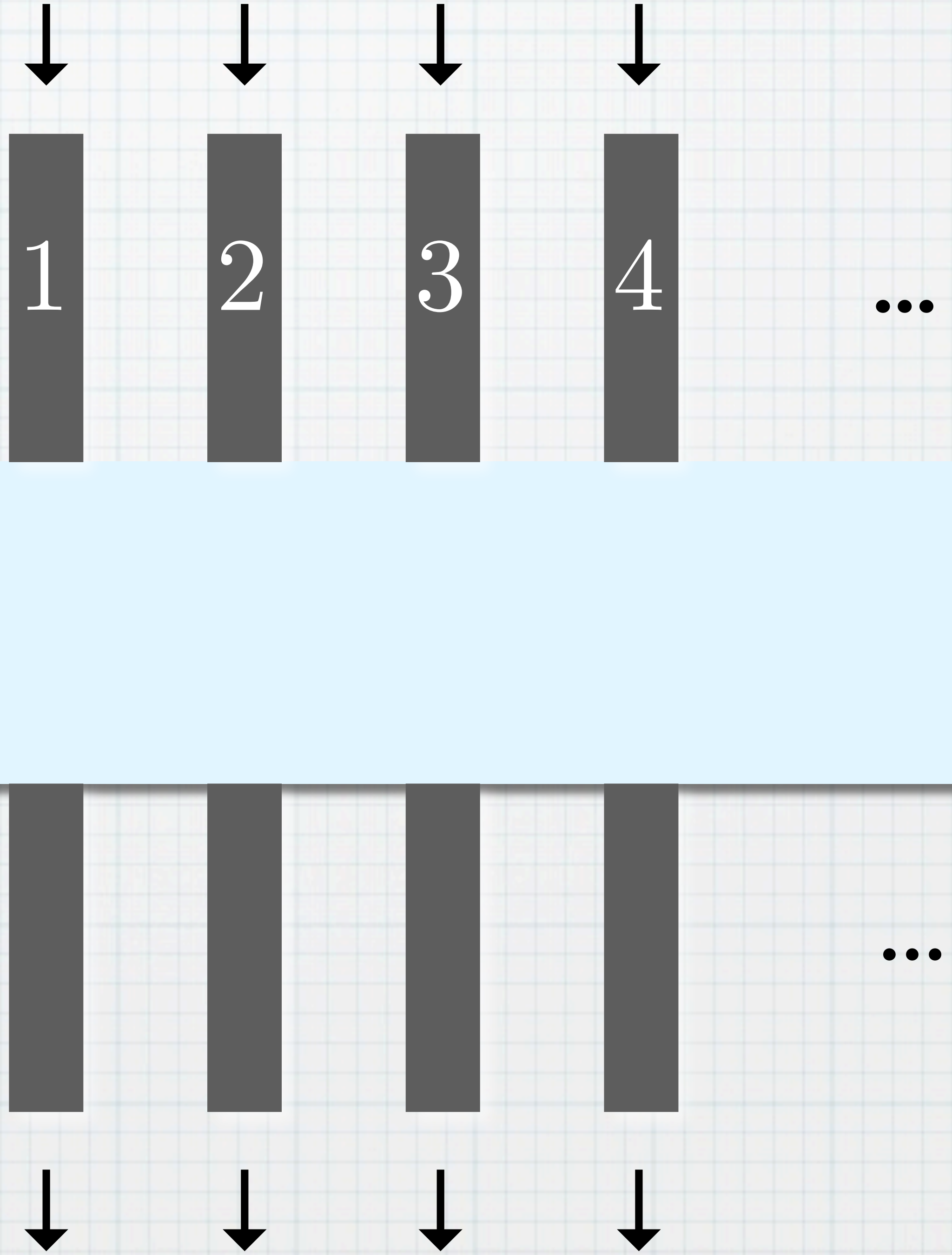
- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

Different Branching in The GoI Animation

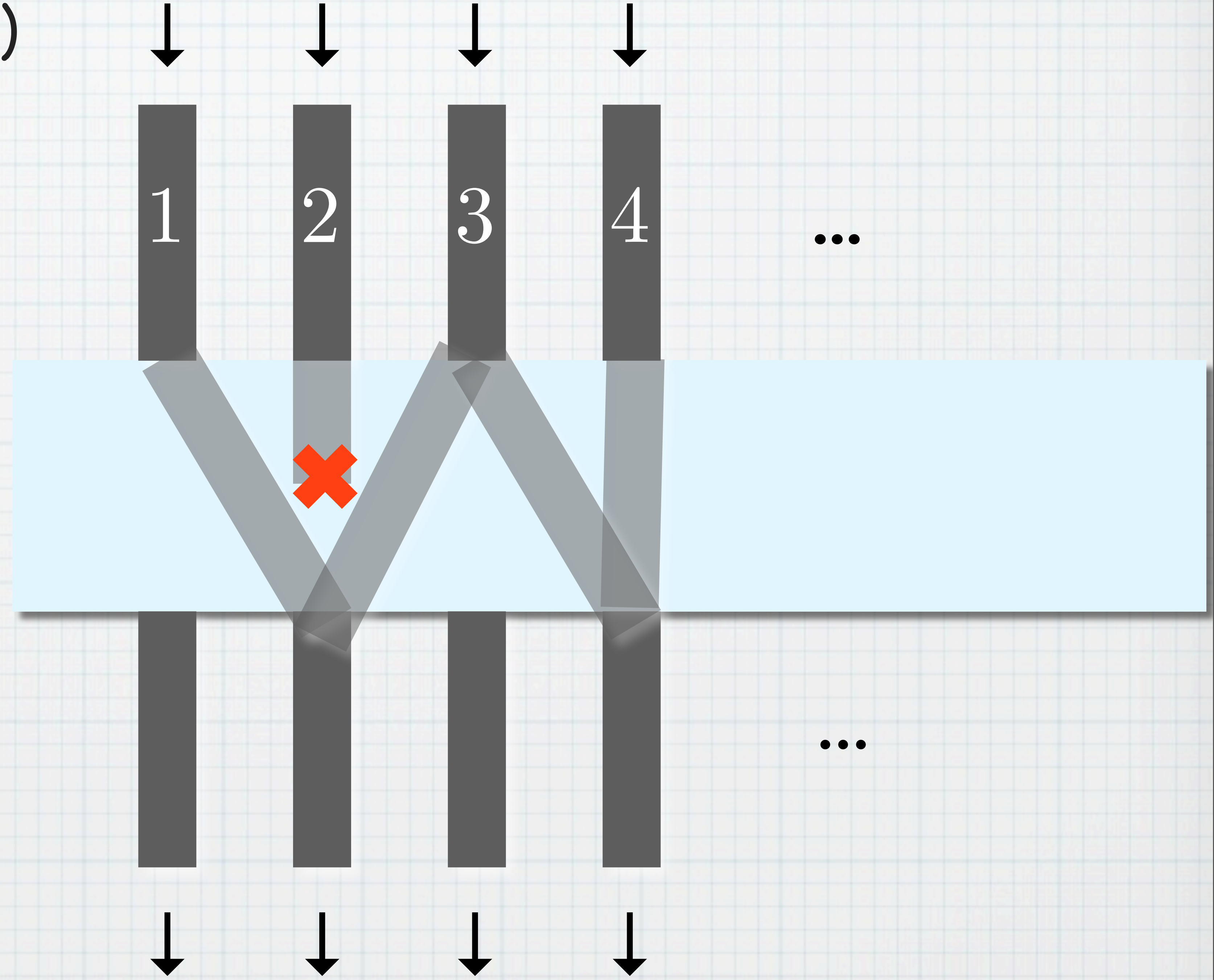
- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

Different Branching in The GoI Animation

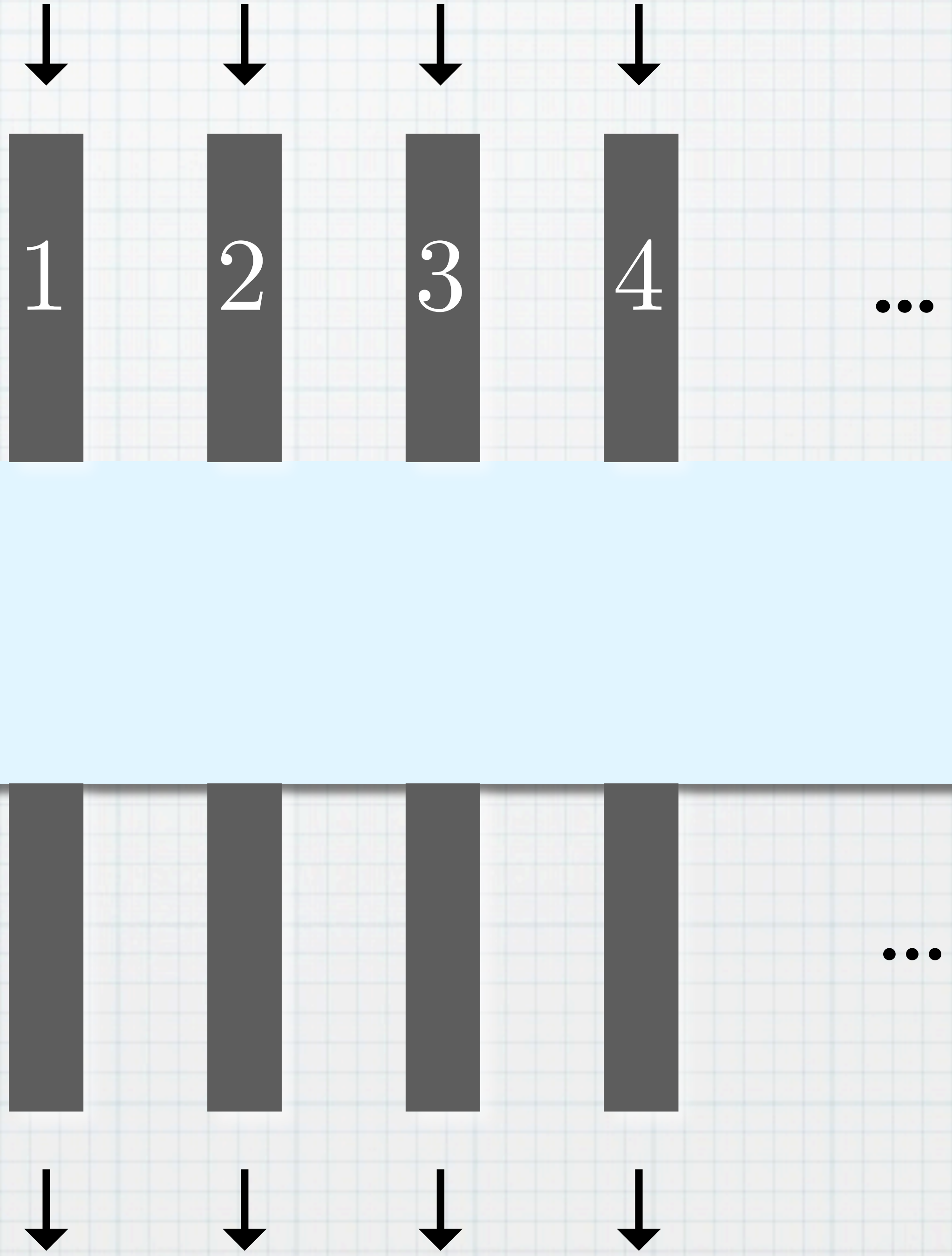
- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

Different Branching in The GoI Animation

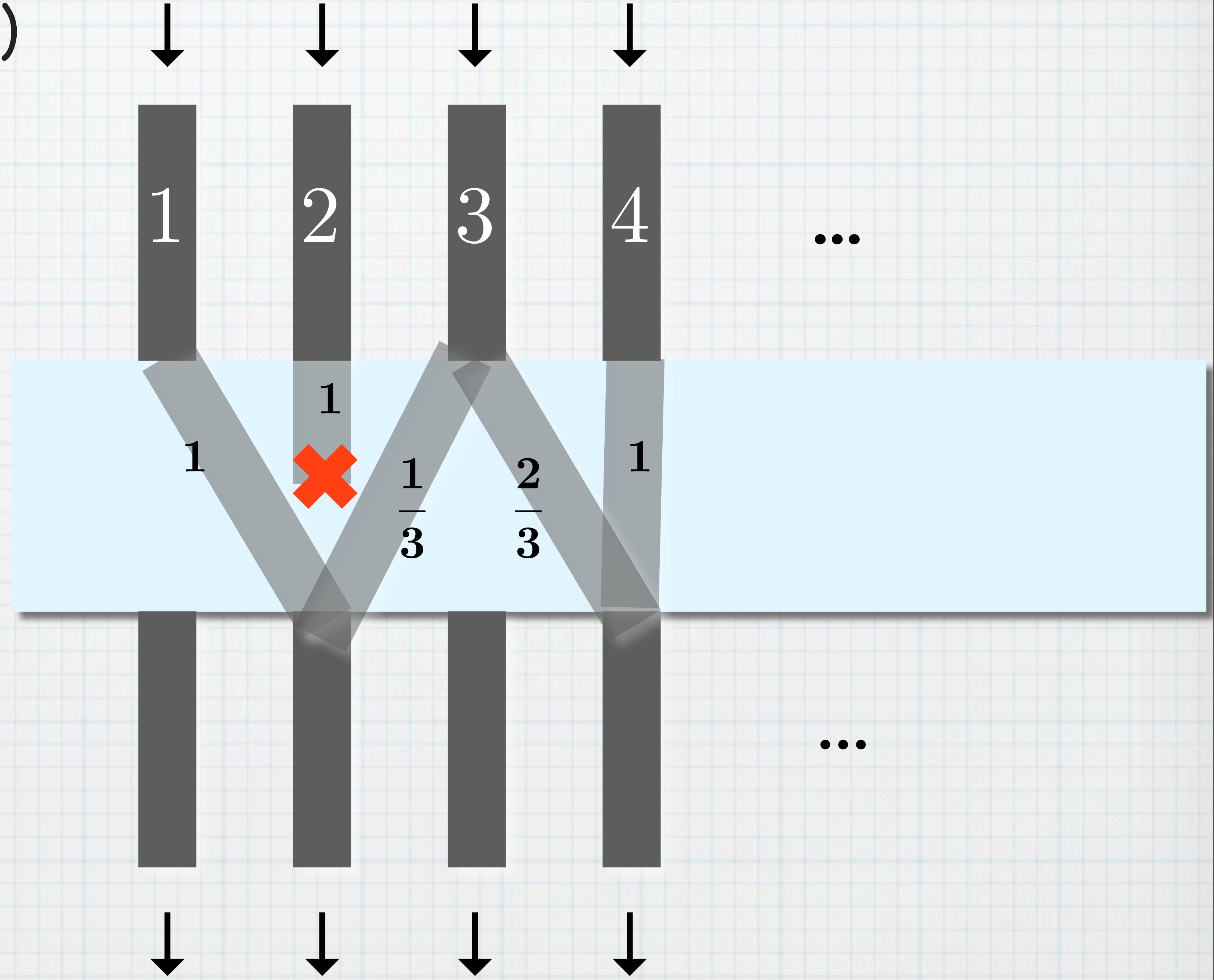
- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

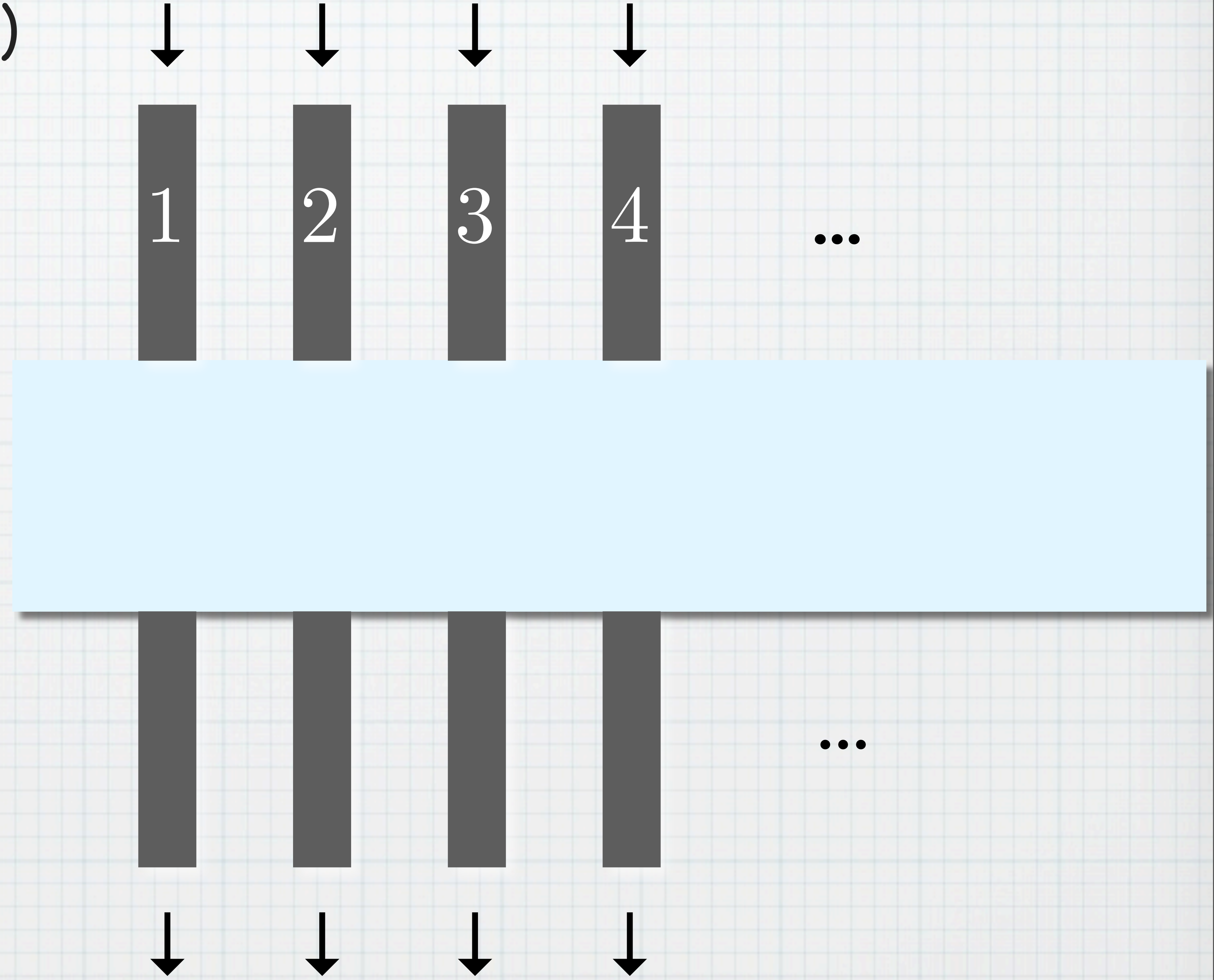
Different Branching in The GoI Animation

- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- → **DSRel**
  - Pipes can branch probabilistically



Different Branching in The GoI Animation

- **Pfn** (partial functions)
  - Pipes can be stuck
- **Rel** (relations)
  - Pipes can branch
- **DSRel**
  - Pipes can branch probabilistically



Hasuo (Tokyo)

Why Catego
Examples Other Than Pfn

Categories of sets and (functions with different branching/partiality)

- **Pfn** (partial functions)

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y, \text{ partial function} \\ \hline\hline X \to \mathcal{L}Y \ \text{ in } \mathsf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y$$

(Potential) non-termination

- **Rel** (relations)

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y, \text{ relation} \\ \hline\hline X \to \mathcal{P}Y \ \text{ in } \mathsf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

Non-determinism

- **DSRel**

$$\begin{array}{c} X \to Y \ \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \ \text{ in } \mathsf{Sets} \end{array}$$

$$\text{where } \mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_{y} d(y) \leq 1\}$$

Probabilistic branching

Why Catego
Examples

Kl(*B*) for different branching monads *B*

- **Pfn** (partial functions)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y, \text{ partial function} \\ \hline\hline X \to \mathcal{L}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y$$

(Potential) non-termination

- **Rel** (relations)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y, \text{ relation} \\ \hline\hline X \to \mathcal{P}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

Non-determinism

- **DSRel**

$$\begin{array}{c} X \to Y \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \text{ in } \mathbf{Sets} \end{array}$$

$$\text{where } \mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_y d(y) \le 1\}$$

Probabilistic branching

The Coauthor

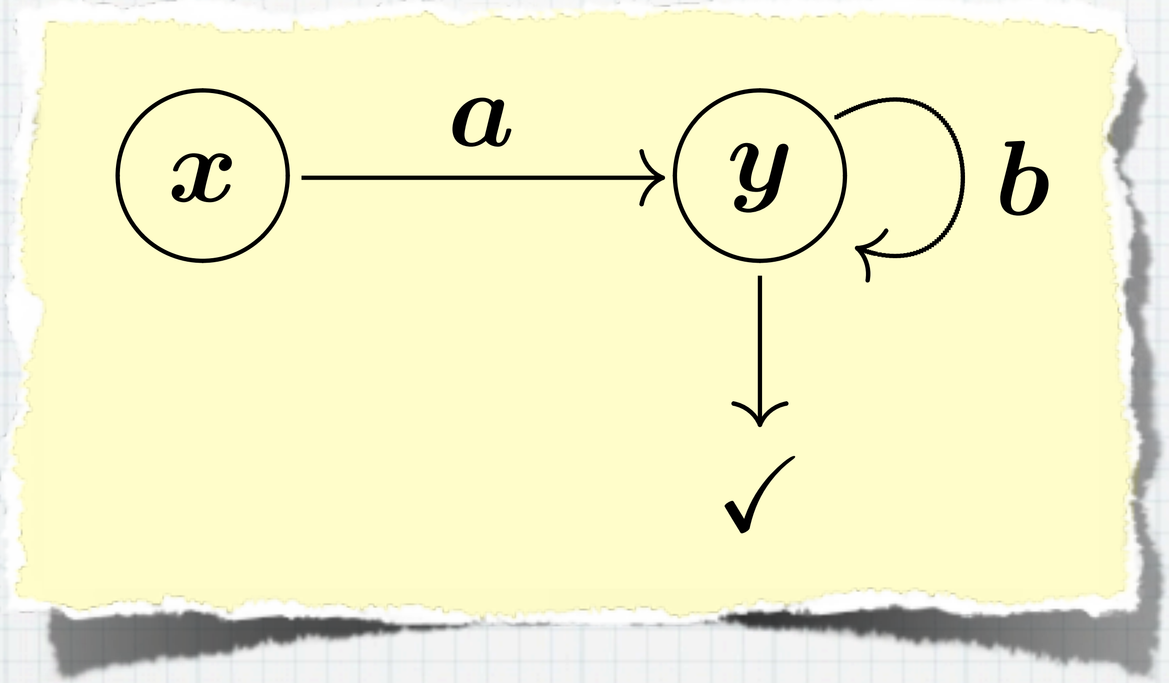
- Naohiko Hoshino
  - DSc (Kyoto, 2011)
    - Supervisor: Masahito "Hassei" Hasegawa
  - Currently at RIMS, Kyoto U.
    - http://www.kurims.kyoto-u.ac.jp/~naophiko/

Intermission

(If time allows)

Coalgebraic Trace Semantics

Trace Semantics of Systems

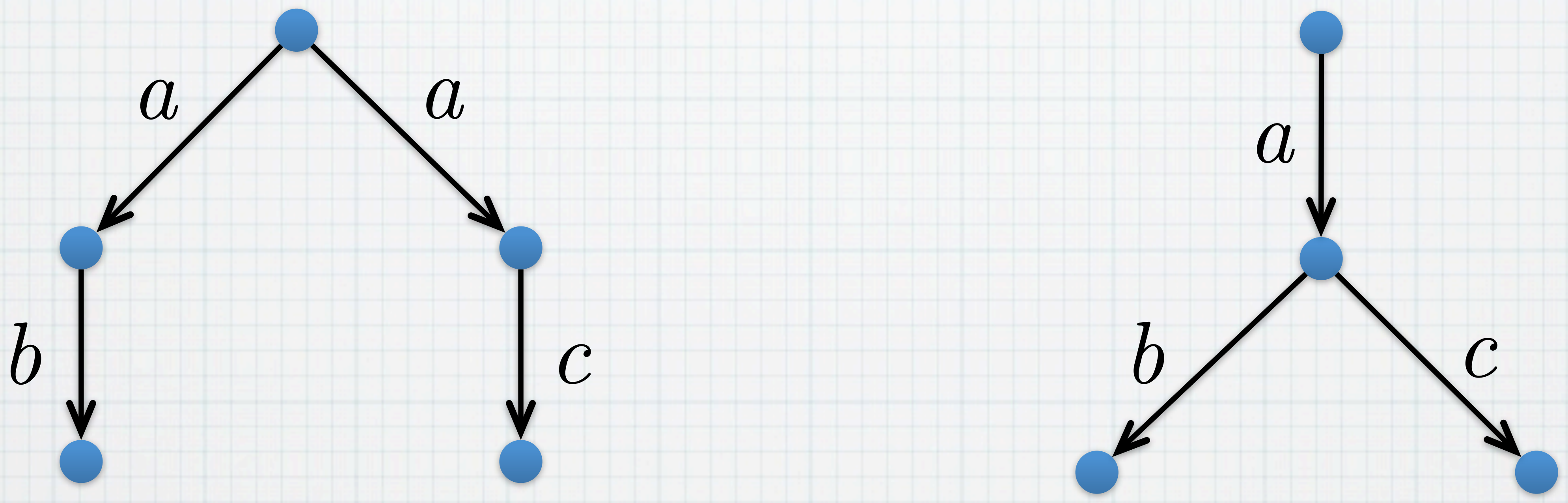


$$\mathbf{tr}(x) = \{a, ab, abb, \dots\} = ab^*$$

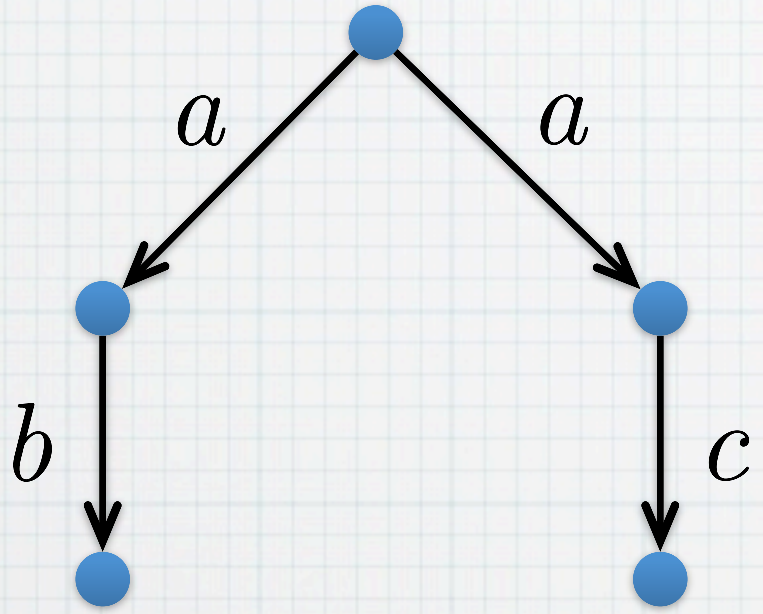
- **Non-deterministic** branching: sign. functor is $\mathcal{P}(1 + \Sigma \times \_)$

Hasuo (Tokyo)

Bisimilarity vs. Trace Sem.

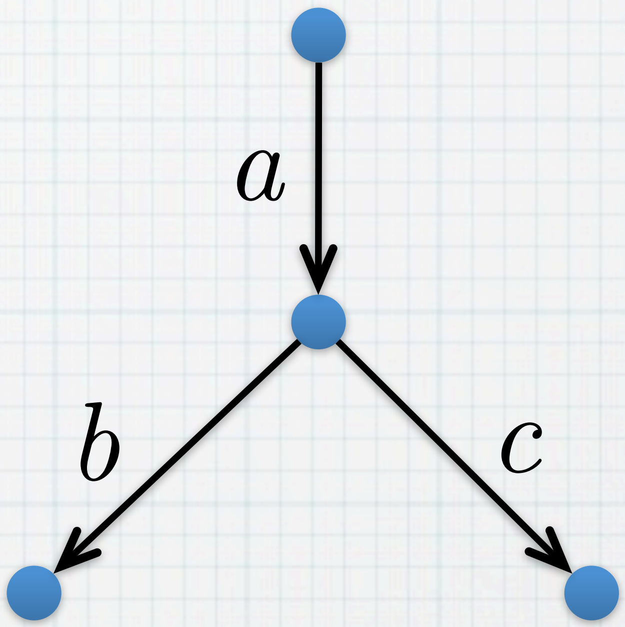


Bisimilarity vs. Trace Sem.

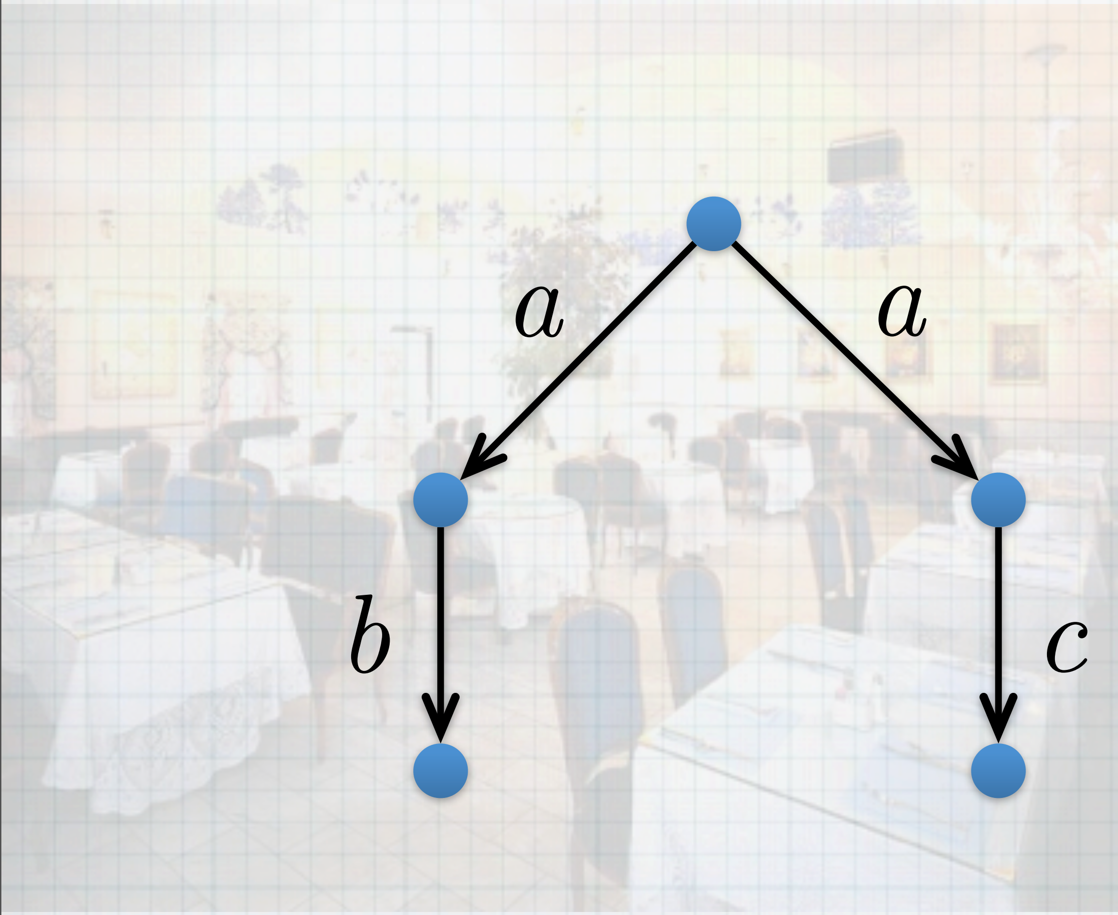


$\neq$

$=$

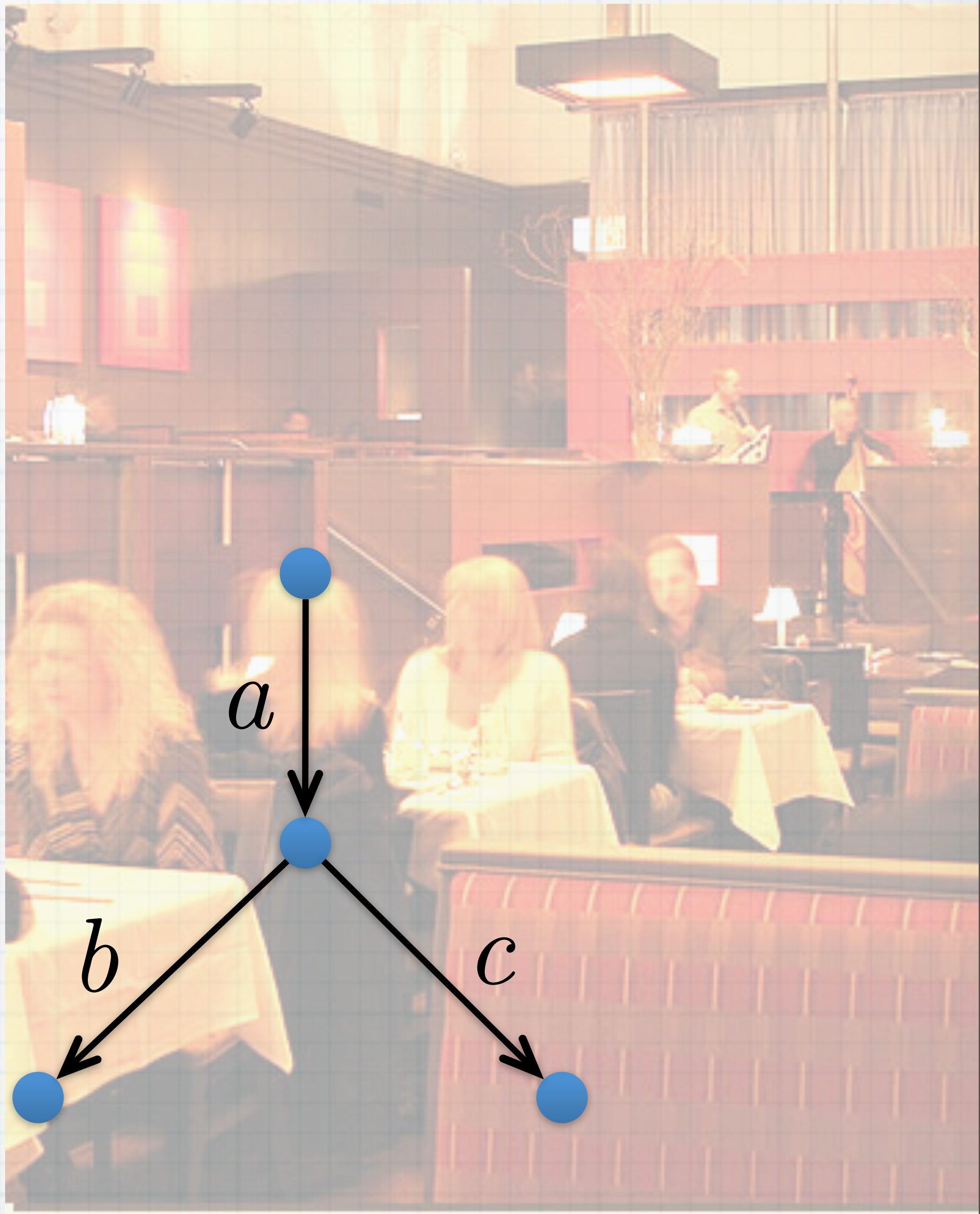


Bisimilarity vs. Trace Sem.

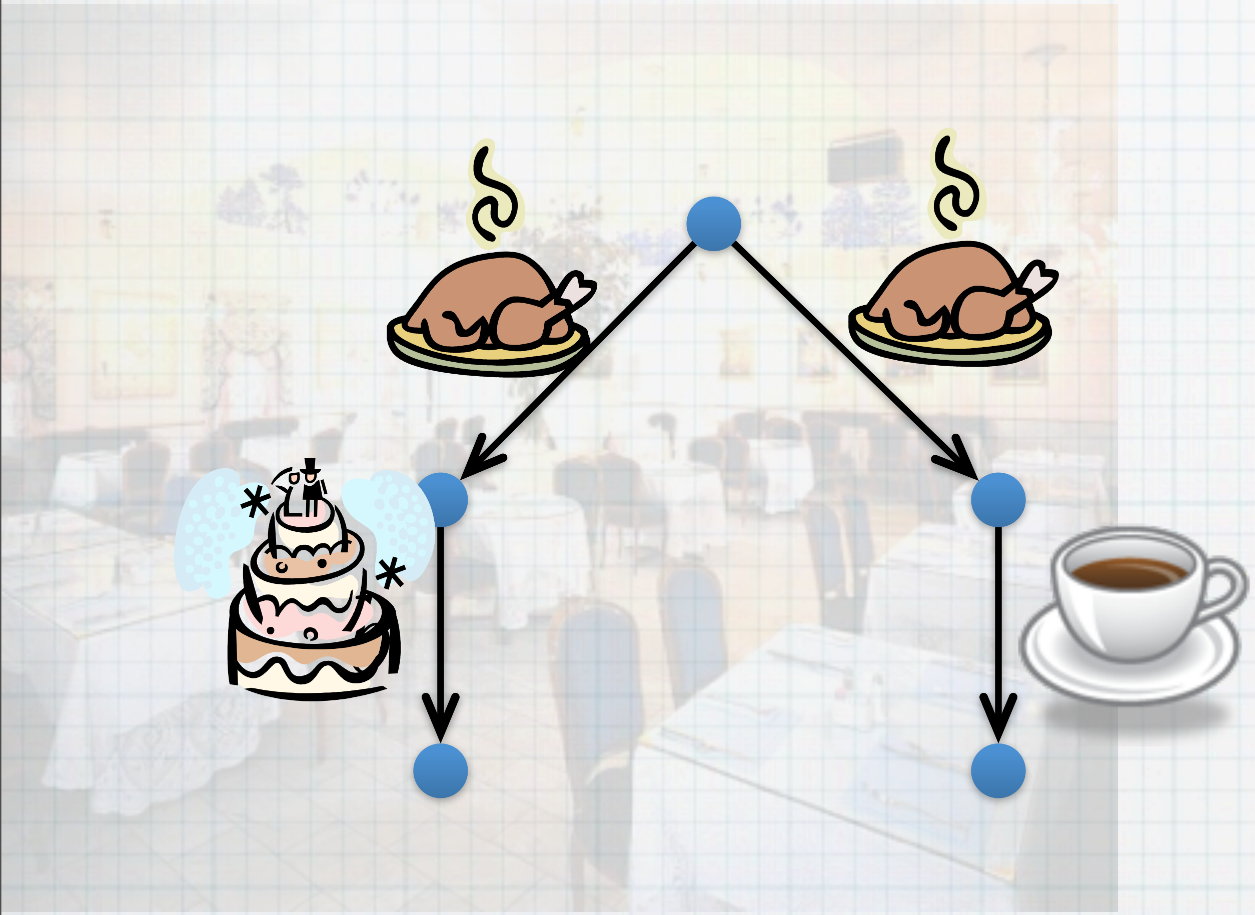


$\neq$

$=$

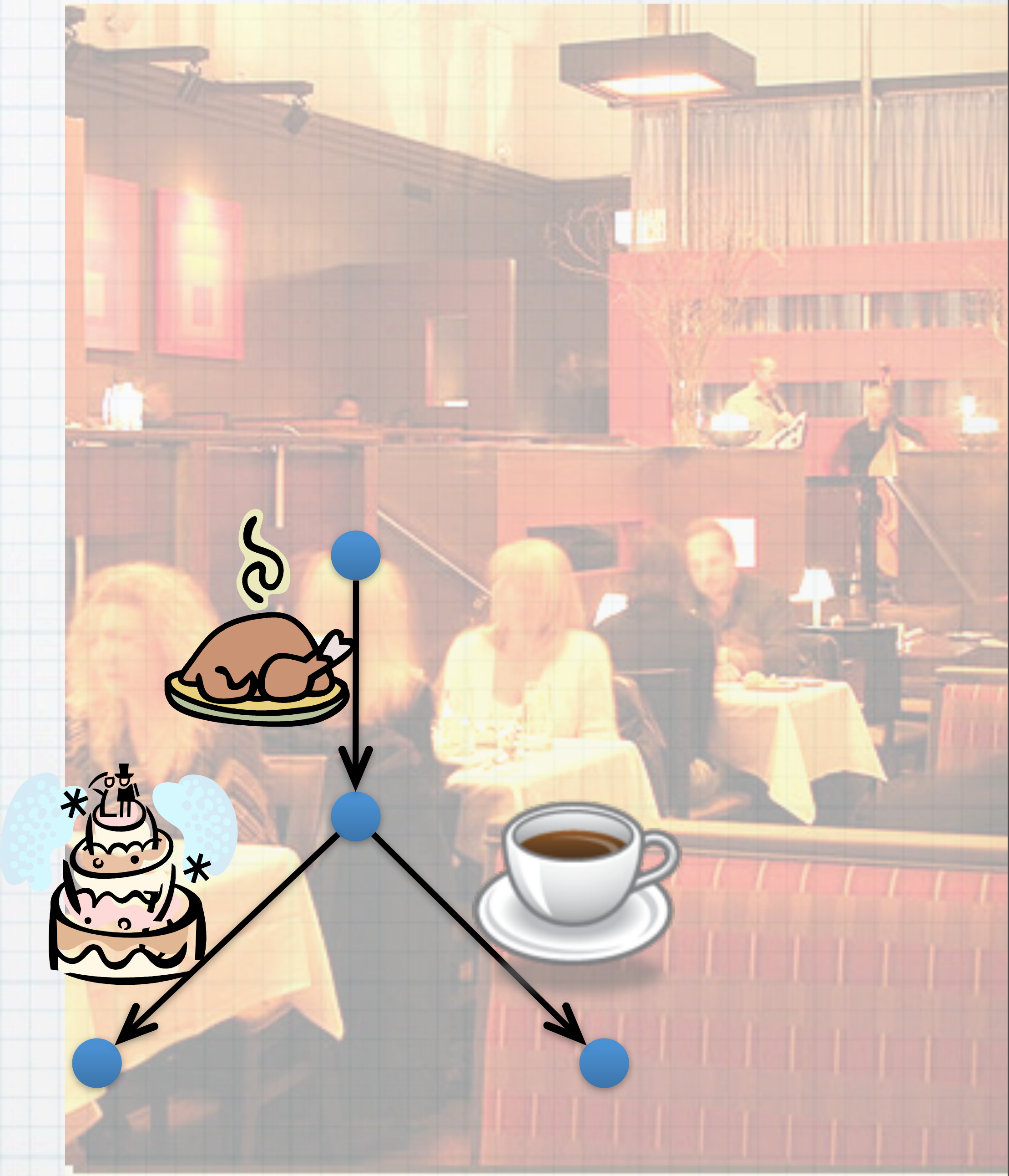


Bisimilarity vs. Trace Sem.



≠

=



Bisimilarity

Branching structure matters.

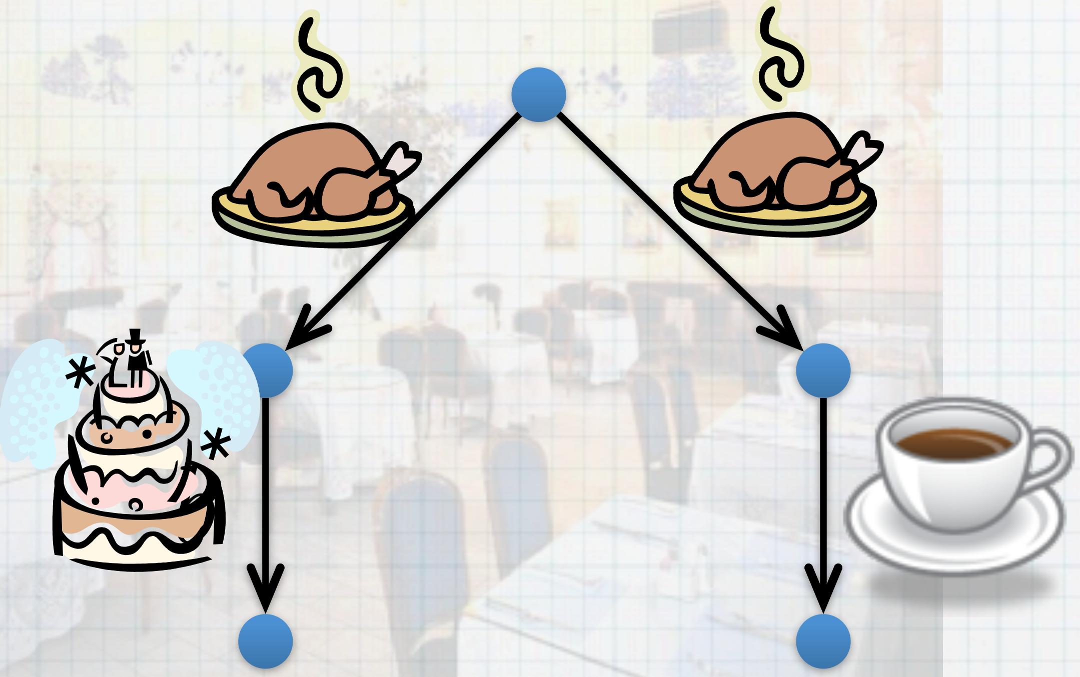
Can I choose later?

Trace semantics

Branching structure does not matter.

Anyway we'll get the same sets of food.

Bisimilarity vs. Trace Sem.



$\neq$

$=$

Also by final coalgebra?

$$\begin{tikzcd} FX \arrow[r, dashed, "F\mathbf{beh}(c)"] & FZ \\ X \arrow[u, "c"] \arrow[r, dashed, "\mathbf{beh}(c)"'] & Y \arrow[u, "\text{final}"'] \end{tikzcd}$$

Bisimilarity

Branching structure matters.

Can I choose later?

Trace semantics

Branching structure does not matter.

Anyway we'll get the same sets of food.

Coinduction in a Kleisli Category

[IH, Jacobs, Sokolova, '07]

$$\begin{array}{c} X \longrightarrow\!\!\!\!\!\!\!\!\!\!+\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(B) \\ \hline\hline X \longrightarrow BY \quad \text{in } \mathbf{Sets} \end{array}$$

Thm. Let F be an endofunctor, and B be a monad, both on **Sets**. Assume:

1. We have a distributive law $\lambda : FB \Rightarrow BF$.
2. The functor F preserves ω-colimits, yielding an initial algebra $\begin{array}{c} FA \\ \cong\downarrow\alpha \\ A \end{array}$.
3. The Kleisli category $\mathcal{K}\ell(B)$ is $\mathbf{Cpo}_\perp$-enriched and composition in $\mathcal{K}\ell(B)$ is left-strict.

Then:

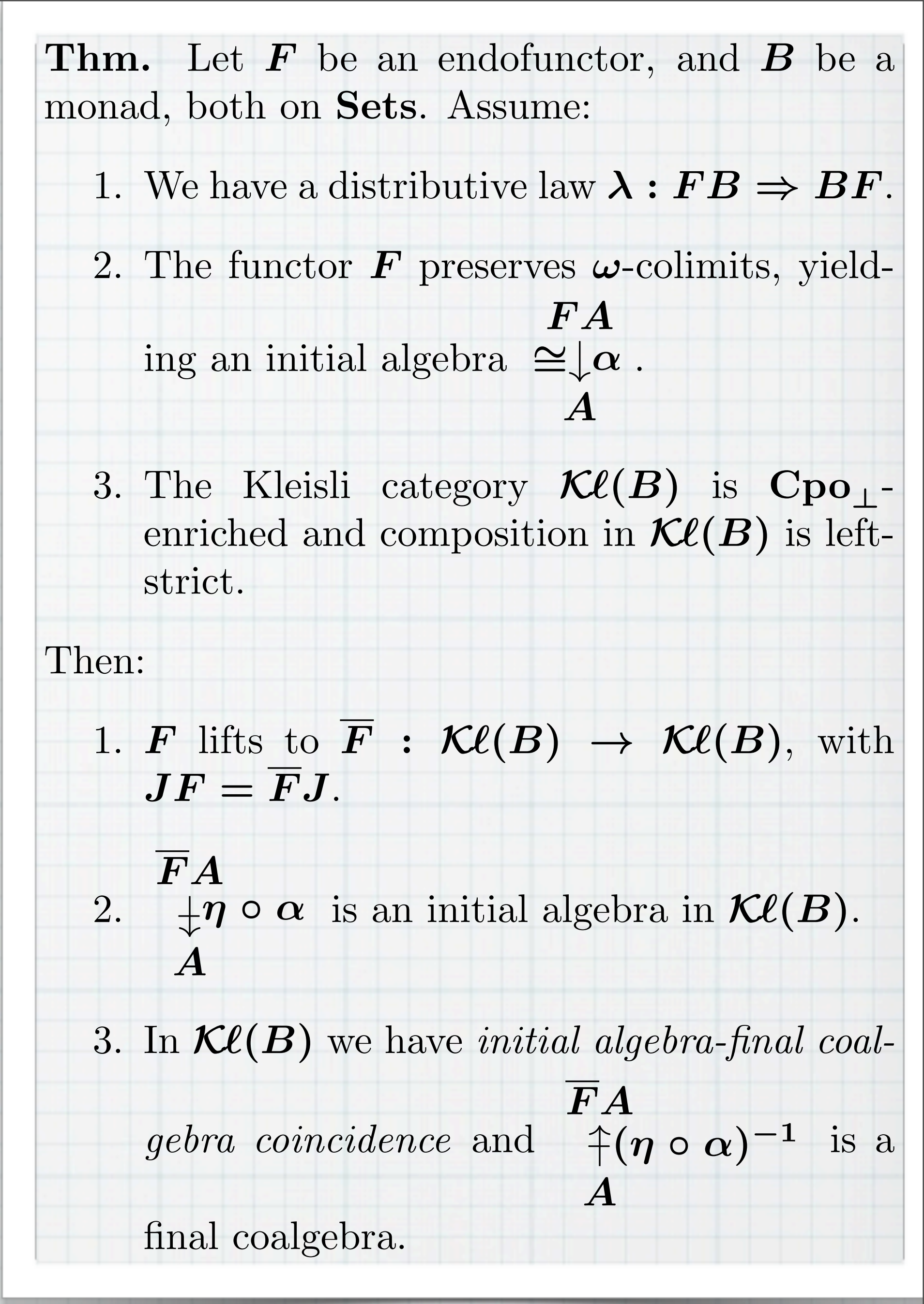
1. F lifts to $\overline{F} : \mathcal{K}\ell(B) \to \mathcal{K}\ell(B)$, with $JF = \overline{F}J$.
2. $\begin{array}{c} \overline{F}A \\ \downarrow\!\!\!\!+\,\eta\circ\alpha \\ A \end{array}$ is an initial algebra in $\mathcal{K}\ell(B)$.
3. In $\mathcal{K}\ell(B)$ we have *initial algebra-final coalgebra coincidence* and $\begin{array}{c} \overline{F}A \\ \uparrow\!\!\!\!+\,(\eta\circ\alpha)^{-1} \\ A \end{array}$ is a final coalgebra.

Coinduction in a Kleisli Category

[IH, Jacobs, Sokolova, '07]

$$\frac{X \relbar\joinrel\mapstochar\joinrel\rightarrow Y \quad \text{in } \mathcal{K}\ell(B)}{X \longrightarrow BY \quad \text{in } \mathbf{Sets}}$$

- Initial algebra lifts from Sets to $Kl(B)$
  - diagram chasing [Johnstone]



Thm. Let F be an endofunctor, and B be a monad, both on **Sets**. Assume:

1. We have a distributive law $\lambda : FB \Rightarrow BF$.
2. The functor F preserves ω-colimits, yielding an initial algebra $\begin{array}{c} FA \\ \cong\downarrow\alpha \\ A \end{array}$.
3. The Kleisli category $\mathcal{K}\ell(B)$ is $\mathbf{Cpo}_\perp$-enriched and composition in $\mathcal{K}\ell(B)$ is left-strict.

Then:

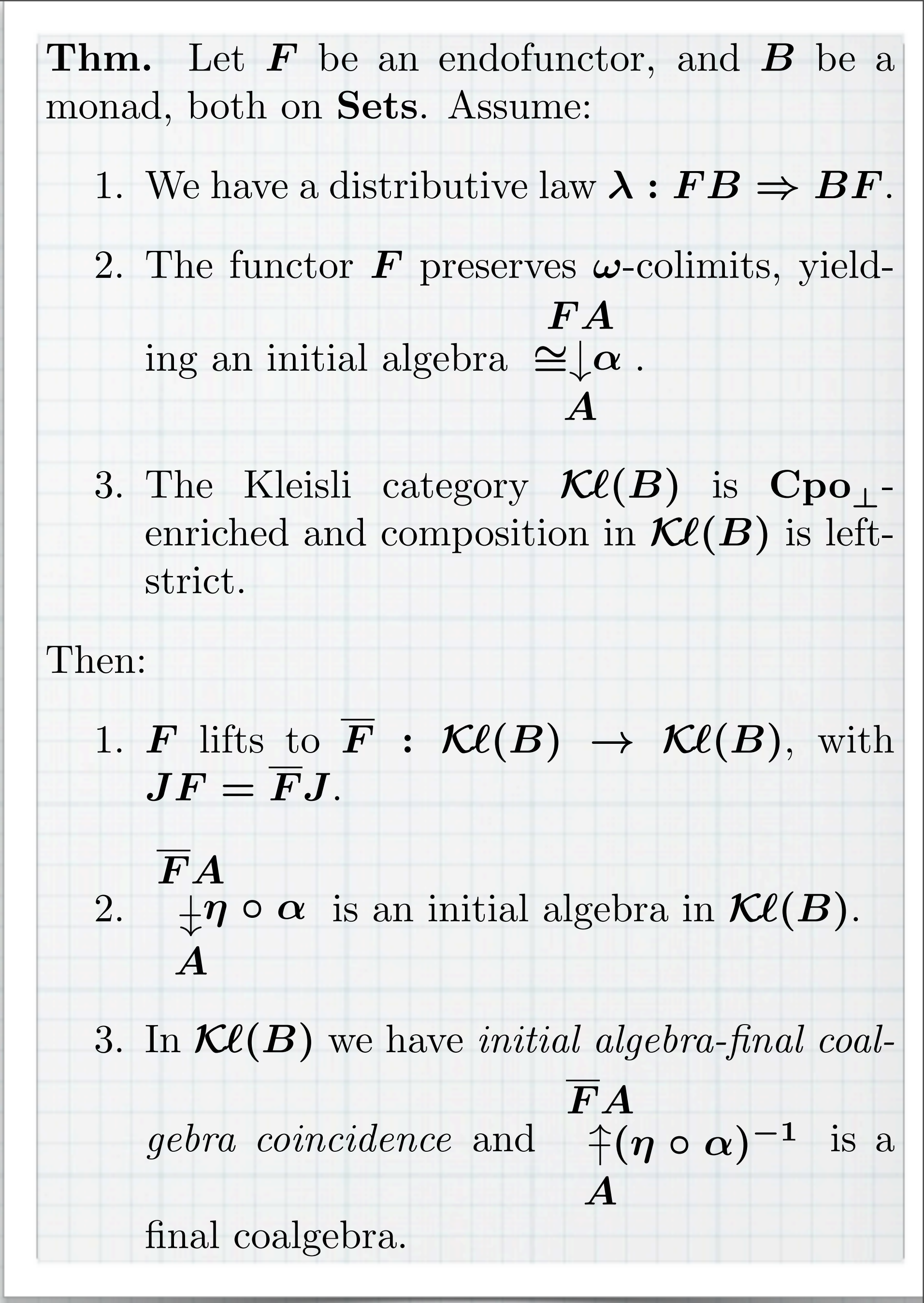
1. F lifts to $\overline{F} : \mathcal{K}\ell(B) \to \mathcal{K}\ell(B)$, with $JF = \overline{F}J$.
2. $\begin{array}{c} \overline{F}A \\ \downarrow\!\!\!\!+\ \eta\circ\alpha \\ A \end{array}$ is an initial algebra in $\mathcal{K}\ell(B)$.
3. In $\mathcal{K}\ell(B)$ we have *initial algebra-final coalgebra coincidence* and $\begin{array}{c} \overline{F}A \\ \uparrow\!\!\!\!+\ (\eta\circ\alpha)^{-1} \\ A \end{array}$ is a final coalgebra.

Hasuo (Tokyo)

Coinduction in a Kleisli Category

[IH, Jacobs, Sokolova, '07]

$$\begin{array}{ll} X \relbar\joinrel\mapstochar\joinrel\rightarrow Y & \text{in } \mathcal{K}\ell(B) \\ \hline\hline X \longrightarrow BY & \text{in } \mathbf{Sets} \end{array}$$



Thm. Let F be an endofunctor, and B be a monad, both on $\mathbf{Sets}$. Assume:

1. We have a distributive law $\lambda : FB \Rightarrow BF$.
2. The functor F preserves ω-colimits, yielding an initial algebra $\begin{array}{c} FA \\ \cong\downarrow\alpha \\ A \end{array}$.
3. The Kleisli category $\mathcal{K}\ell(B)$ is $\mathbf{Cpo}_{\perp}$-enriched and composition in $\mathcal{K}\ell(B)$ is left-strict.

Then:

1. F lifts to $\overline{F} : \mathcal{K}\ell(B) \to \mathcal{K}\ell(B)$, with $JF = \overline{F}J$.
2. $\begin{array}{c} \overline{F}A \\ \downarrow\!\!\!\mapstochar\, \eta\circ\alpha \\ A \end{array}$ is an initial algebra in $\mathcal{K}\ell(B)$.
3. In $\mathcal{K}\ell(B)$ we have *initial algebra-final coalgebra coincidence* and $\begin{array}{c} \overline{F}A \\ \uparrow\!\!\!\mapstochar\,(\eta\circ\alpha)^{-1} \\ A \end{array}$ is a final coalgebra.

- Initial algebra lifts from **Sets** to $Kl(B)$
  - diagram chasing [Johnstone]
- In $Kl(B)$ we have IA-FC coincidence
  - typical of "domain-theoretic" categories
  - "Algebraically compact" [Freyd]

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P},\ F = 1 + \Sigma \times (\_\,)$

$$\begin{tikzcd}
1 + \Sigma \times X \arrow[r, dashed, "\shortmid" marking, "1 + \Sigma \times \mathbf{tr}(c)"] & 1 + \Sigma \times \Sigma^* & \text{in } \mathcal{K}\ell(\mathcal{P}) \\
X \arrow[u, "\shortmid" marking, "c"] \arrow[r, dashed, "\shortmid" marking, "\mathbf{tr}(c)"'] & \Sigma^* \arrow[u, "\shortmid" marking, "\text{final}"'] &
\end{tikzcd}$$

- Separation between B and F

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_\,)$

$$\begin{tikzcd}
1 + \Sigma \times X \arrow[r, dashed, "1 + \Sigma \times \mathbf{tr}(c)", "\shortmid" marking] & 1 + \Sigma \times \Sigma^* \\
X \arrow[u, "c", "\shortmid" marking] \arrow[r, dashed, "\mathbf{tr}(c)"', "\shortmid" marking] & \Sigma^* \arrow[u, "\text{final}"', "\shortmid" marking]
\end{tikzcd}
\quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

$$\begin{tikzcd}
\mathcal{P}(1 + \Sigma \times X) \\
X \arrow[u, "c"]
\end{tikzcd}
\quad \text{in } \mathbf{Sets}$$

- Separation between B and F

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_)$

$$\begin{tikzcd}
1+\Sigma\times X \arrow[r, dashed, "|" marking, "1+\Sigma\times \mathsf{tr}(c)"] & 1+\Sigma\times\Sigma^* \\
X \arrow[u, "|" marking, "c"] \arrow[r, dashed, "|" marking, "\mathsf{tr}(c)"'] & \Sigma^* \arrow[u, "|" marking, "\text{final}"']
\end{tikzcd}$$

in $\mathcal{K}\ell(\mathcal{P})$

$$\begin{tikzcd}
\mathcal{P}(1+\Sigma\times X) \\
X \arrow[u, "c"]
\end{tikzcd}$$

in **Sets**

induced by

$$\begin{tikzcd}
1+\Sigma\times\Sigma^* \arrow[d, "\text{initial}"] \\
\Sigma^*
\end{tikzcd}$$

in **Sets**

- Separation between B and F

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P},\ F = 1 + \Sigma \times (\_)$

$$\begin{tikzcd}
1 + \Sigma \times X \arrow[r, dashed, "1 + \Sigma \times \mathbf{tr}(c)"] & 1 + \Sigma \times \Sigma^* \\
X \arrow[u, "c"] \arrow[r, dashed, "\mathbf{tr}(c)"'] & \Sigma^* \arrow[u, "\text{final}"']
\end{tikzcd}$$

in $\mathcal{K}\ell(\mathcal{P})$

induced by

$$\begin{tikzcd}
1 + \Sigma \times \Sigma^* \arrow[d, "\text{initial}"] \\
\Sigma^*
\end{tikzcd}$$

in **Sets**

$$\begin{tikzcd}
\mathcal{P}(1 + \Sigma \times X) \\
X \arrow[u, "c"]
\end{tikzcd}$$

in **Sets**

$$X \xrightarrow{\mathbf{tr}(c)} \mathcal{P}(\Sigma^*)$$

- Separation between B and F

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_)$

$$\begin{tikzcd}
1 + \Sigma \times X \arrow[r, dashed, "1 + \Sigma \times \mathbf{tr}(c)"] & 1 + \Sigma \times \Sigma^* \\
X \arrow[u, "c"] \arrow[r, dashed, "\mathbf{tr}(c)"'] & \Sigma^* \arrow[u, "\text{final}"']
\end{tikzcd}$$

in $\mathcal{K}\ell(\mathcal{P})$

$$\begin{tikzcd}
\mathcal{P}(1 + \Sigma \times X) \\
X \arrow[u, "c"]
\end{tikzcd}$$

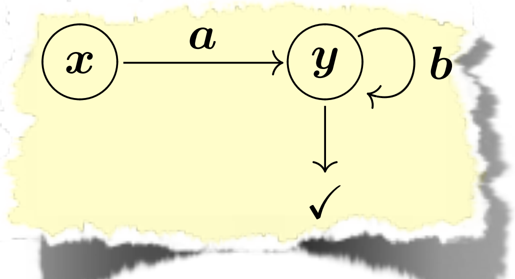
in **Sets**

induced by

$$\begin{tikzcd}
1 + \Sigma \times \Sigma^* \arrow[d, "\text{initial}"] \\
\Sigma^*
\end{tikzcd}$$

in **Sets**

$$X \xrightarrow{\mathbf{tr}(c)} \mathcal{P}(\Sigma^*)$$



$\mathbf{tr}(x) = \{a, ab, abb, \dots\} = ab^*$

- Separation between B and F

Coinduction in a Kleisli Category

- E.g. $B = \mathcal{P},\ F = 1 + \Sigma \times (\_)$

$$\begin{tikzcd}
1 + \Sigma \times X \arrow[r, dashed, "1 + \Sigma \times \mathbf{tr}(c)"] & 1 + \Sigma \times \Sigma^* \\
X \arrow[u, "c"] \arrow[r, dashed, "\mathbf{tr}(c)"'] & \Sigma^* \arrow[u, "\text{final}"']
\end{tikzcd}$$

in $\mathcal{K}\ell(\mathcal{P})$

induced by

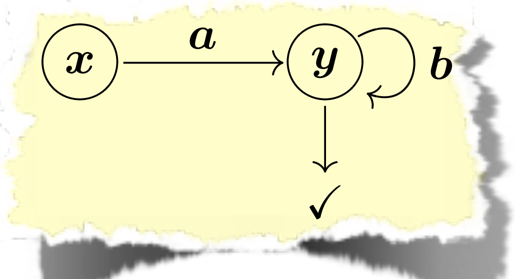
$$\begin{tikzcd}
1 + \Sigma \times \Sigma^* \arrow[d, "\text{initial}"] \\
\Sigma^*
\end{tikzcd}$$

in **Sets**

$$\begin{tikzcd}
\mathcal{P}(1 + \Sigma \times X) \\
X \arrow[u, "c"]
\end{tikzcd}$$

in **Sets**

$$\begin{tikzcd}
X \arrow[r, "\mathbf{tr}(c)"] & \mathcal{P}(\Sigma^*)
\end{tikzcd}$$



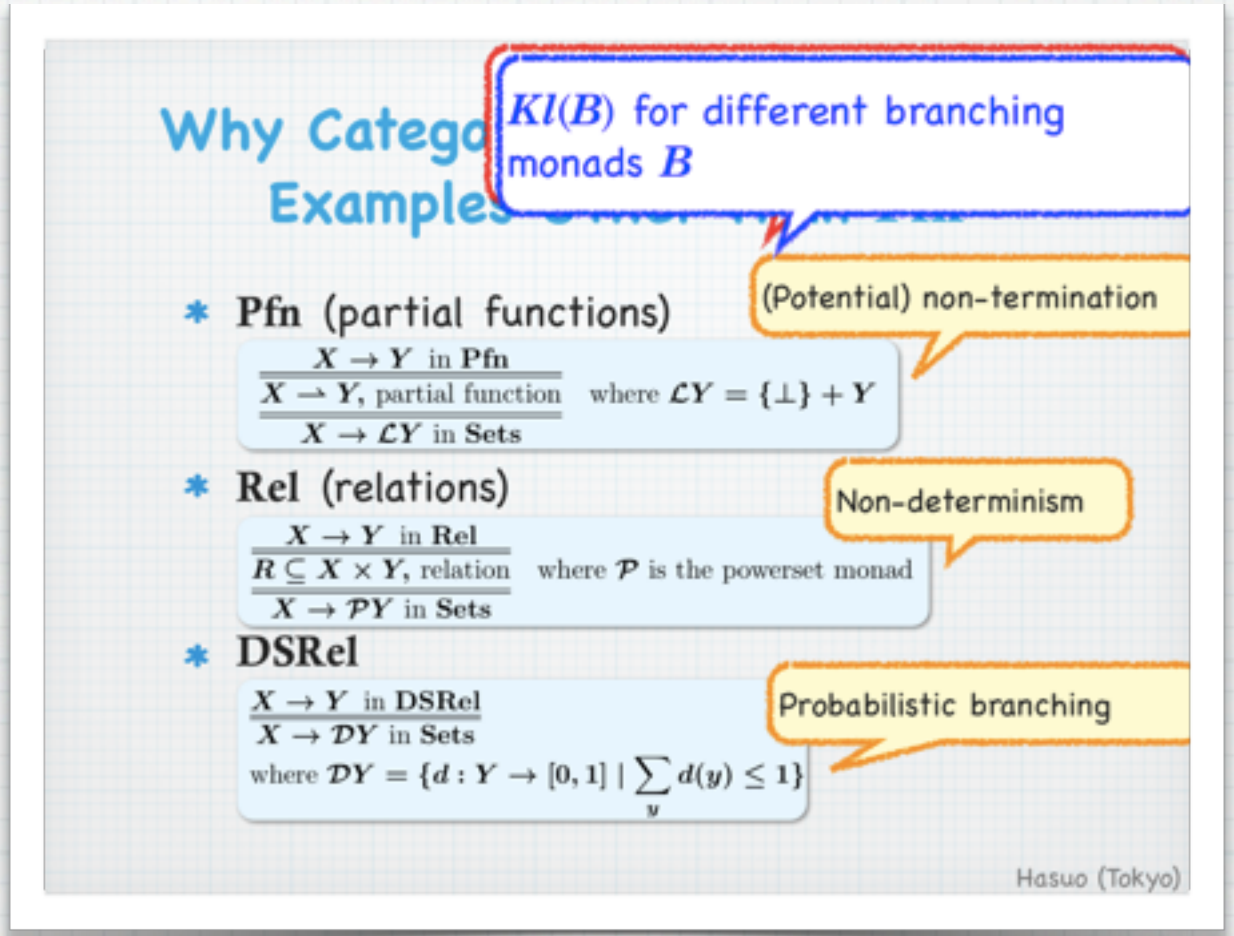
$\mathbf{tr}(x) = \{a, ab, abb, \dots\} = ab^*$

- Separation between B and F

Hasuo (Tokyo)

Examples

- A branching monad B:
  - Lift monad $\mathcal{L} = 1 + (\_)$, powerset monad $\mathcal{P}$, subdistribution monad $\mathcal{D}$
  - Precisely those in



Why Catego

Example

$Kl(B)$ for different branching monads B

- Pfn (partial functions)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Pfn} \\ \hline\hline X \rightharpoonup Y\text{, partial function} \\ \hline\hline X \to \mathcal{L}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y$$

(Potential) non-termination

- Rel (relations)

$$\begin{array}{c} X \to Y \text{ in } \mathbf{Rel} \\ \hline\hline R \subseteq X \times Y\text{, relation} \\ \hline\hline X \to \mathcal{P}Y \text{ in } \mathbf{Sets} \end{array} \quad \text{where } \mathcal{P} \text{ is the powerset monad}$$

Non-determinism

- DSRel

$$\begin{array}{c} X \to Y \text{ in } \mathbf{DSRel} \\ \hline\hline X \to \mathcal{D}Y \text{ in } \mathbf{Sets} \end{array}$$

where $\mathcal{D}Y = \{d : Y \to [0,1] \mid \sum_y d(y) \leq 1\}$

Probabilistic branching

Hasuo (Tokyo)

- A functor F: polynomial functors

From Coalgebraic Trace to Monoidal Trace

Thm. ([Jacobs,CMCS10])
Given a "branching monad" $\boldsymbol{B}$ on **Sets**, the monoidal category

$$(\mathcal{K}\ell(B), +, \mathbf{0})$$

is a traced symmetric monoidal category.

Cor.
$\big(\, (\mathcal{K}\ell(B), +, \mathbf{0}),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$ is a GoI situation.

From Coalgebraic Trace to Monoidal Trace

Thm. ([Jacobs,CMCS10])
Given a "branching monad" B on **Sets**, the monoidal category

$$(\mathcal{K}\ell(B), +, \mathbf{0})$$

is a traced symmetric monoidal category.

Cor.
$\big(\,(\mathcal{K}\ell(B), +, \mathbf{0}),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$ is a GoI situation.

From Coalgebraic Trace to Monoidal Trace

Thm. ([Jacobs,CMCS10])
Given a "branching monad" B on **Sets**, the monoidal category

$$(\mathcal{K}\ell(B), +, 0)$$

is a traced symmetric monoidal category.

Cor.
$\big(\, (\mathcal{K}\ell(B), +, 0),\ \mathbb{N}\cdot\_\,,\ \mathbb{N}\,\big)$ is a GoI situation.

Proof. We need

$$\frac{X + Z \xrightarrow{\ f\ } Y + Z \quad \text{in } \mathcal{K}\ell(T)}{X \xrightarrow{\ \mathrm{tr}(f)\ } Y \quad \text{in } \mathcal{K}\ell(T)}$$

- $X + Z \xrightarrow{\ f\ } Y + Z \xrightarrow{\ \kappa\ } Y + (X + Z)$ is a $Y + (\_)$-coalgebra

- $$\begin{tikzcd} Y + \mathbb{N}\cdot Y \arrow[d, "\cong"', "\alpha"] \\ \mathbb{N}\cdot Y \end{tikzcd}$$ is an initial algebra in **Sets**

- Therefore in $\mathcal{K}\ell(T)$:

$$\begin{tikzcd}
Y + (X + Z) \arrow[r, dashed] & Y + \mathbb{N}\cdot Y \\
X + Z \arrow[u, "\kappa \circ f", "|" marking] \arrow[r, dashed, "\mathrm{tr}(c)"', "|" marking] & \mathbb{N}\cdot Y \arrow[u, "\text{final}"', "|" marking] \arrow[d, "\nabla", "|" marking] \\
X \arrow[u, "\kappa_1", "|" marking] & Y
\end{tikzcd}$$

From Coalgebraic Trace to Monoidal Trace

Thm. ([Jacobs,CMCS10])
Given a "branching monad" B on **Sets**, the monoidal category

$$(\mathcal{K}\ell(B), +, 0)$$

is a traced symmetric monoidal category.

Cor.
$\big((\mathcal{K}\ell(B), +, 0),\ \mathbb{N}\cdot\_\,,\ \mathbb{N} \big)$ is a GoI situation.

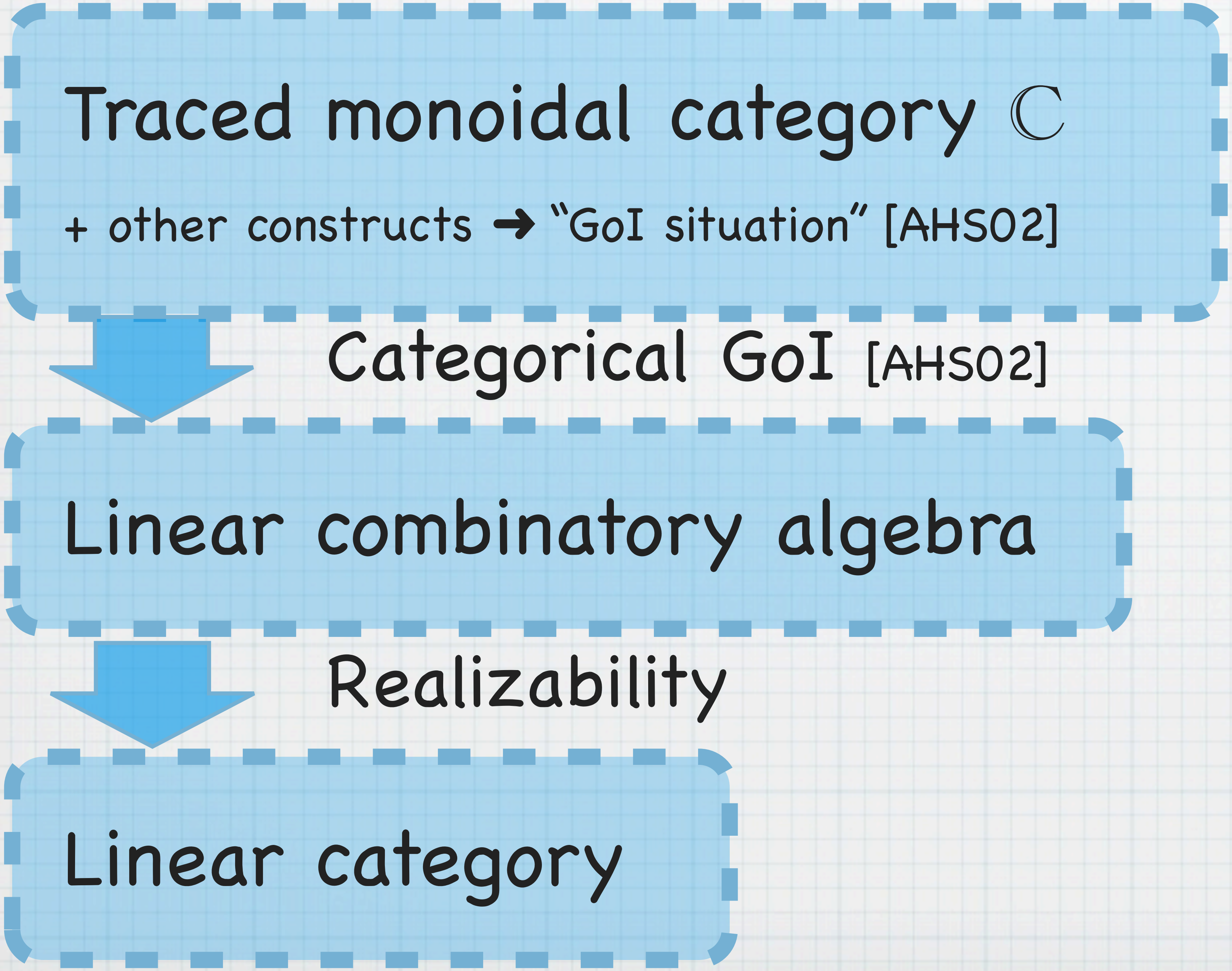
Proof. We need

$$\frac{X + Z \xrightarrow{f} Y + Z \quad \text{in } \mathcal{K}\ell(T)}{X \xrightarrow{\mathsf{tr}(f)} Y \quad \text{in } \mathcal{K}\ell(T)}$$

- $X + Z \xrightarrow{f} Y + Z \xrightarrow{\kappa} Y + (X + Z)$ is a $Y + (\_)$-coalgebra
- $\begin{array}{c} Y + \mathbb{N}\cdot Y \\ \cong\downarrow\alpha \\ \mathbb{N}\cdot Y \end{array}$ is an initial algebra in **Sets**
- Therefore in $\mathcal{K}\ell(T)$:

$$\begin{tikzcd}
Y + (X + Z) \arrow[r, dashed] & Y + \mathbb{N}\cdot Y \\
X + Z \arrow[u, "\kappa \circ f"] \arrow[r, dashed, "\mathsf{tr}(c)"'] & \mathbb{N}\cdot Y \arrow[u, "\text{final}"'] \arrow[d, "\nabla"] \\
X \arrow[u, "\kappa_1"] & Y
\end{tikzcd}$$

The Categorical GoI Workflow



Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

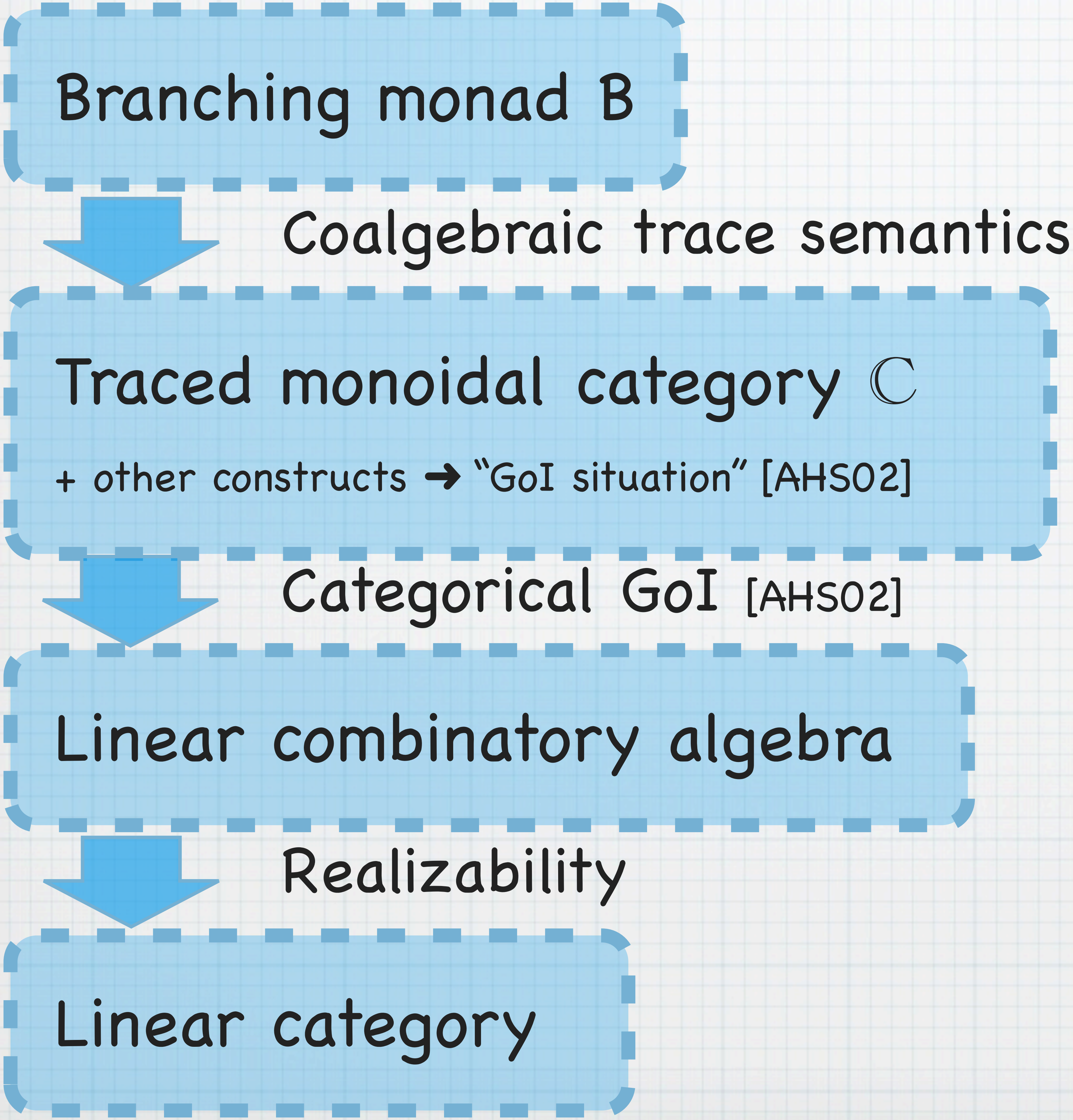
Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

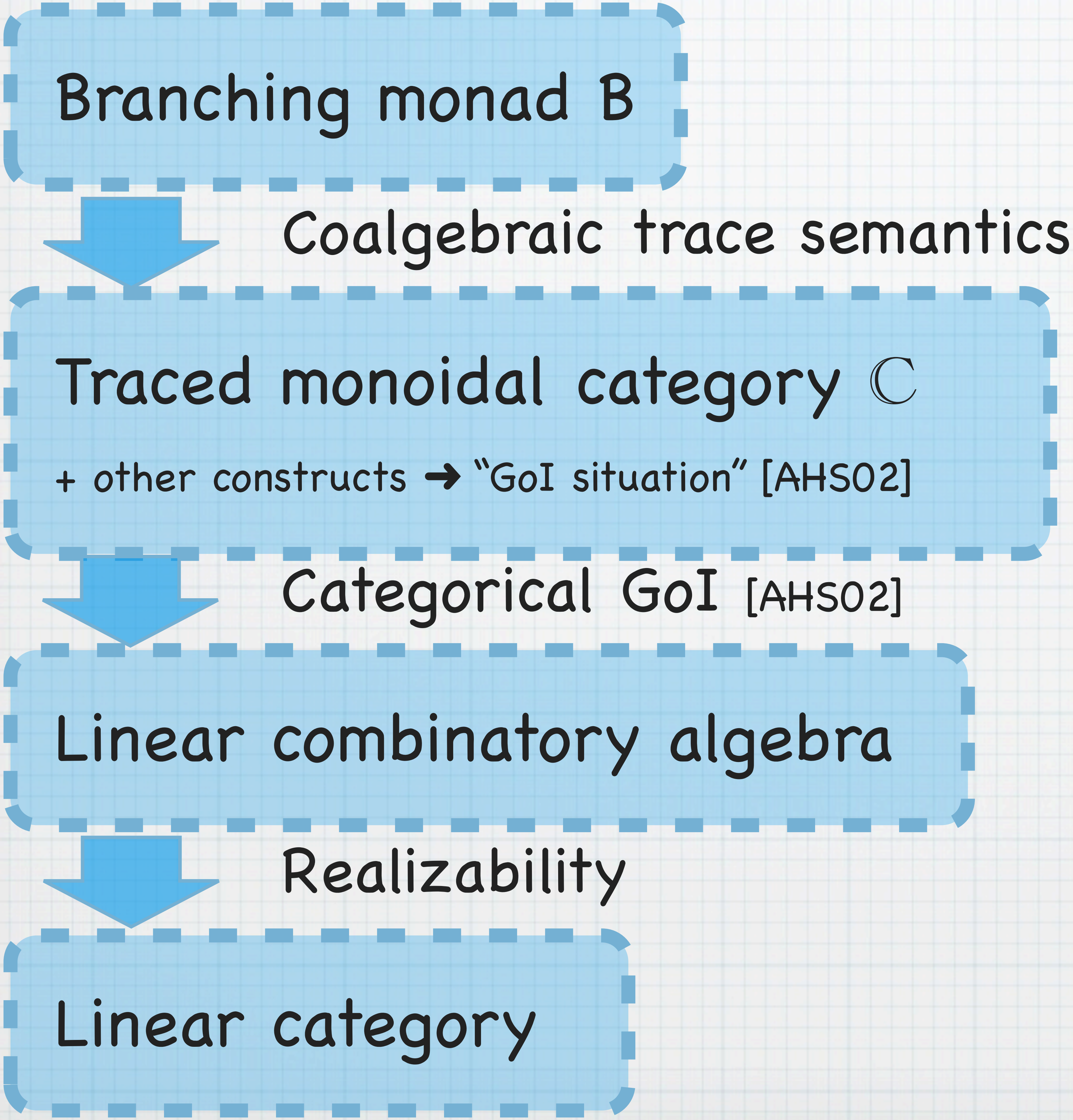
Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ “GoI situation” [AHS02]

Categorical GoI [AHS02]

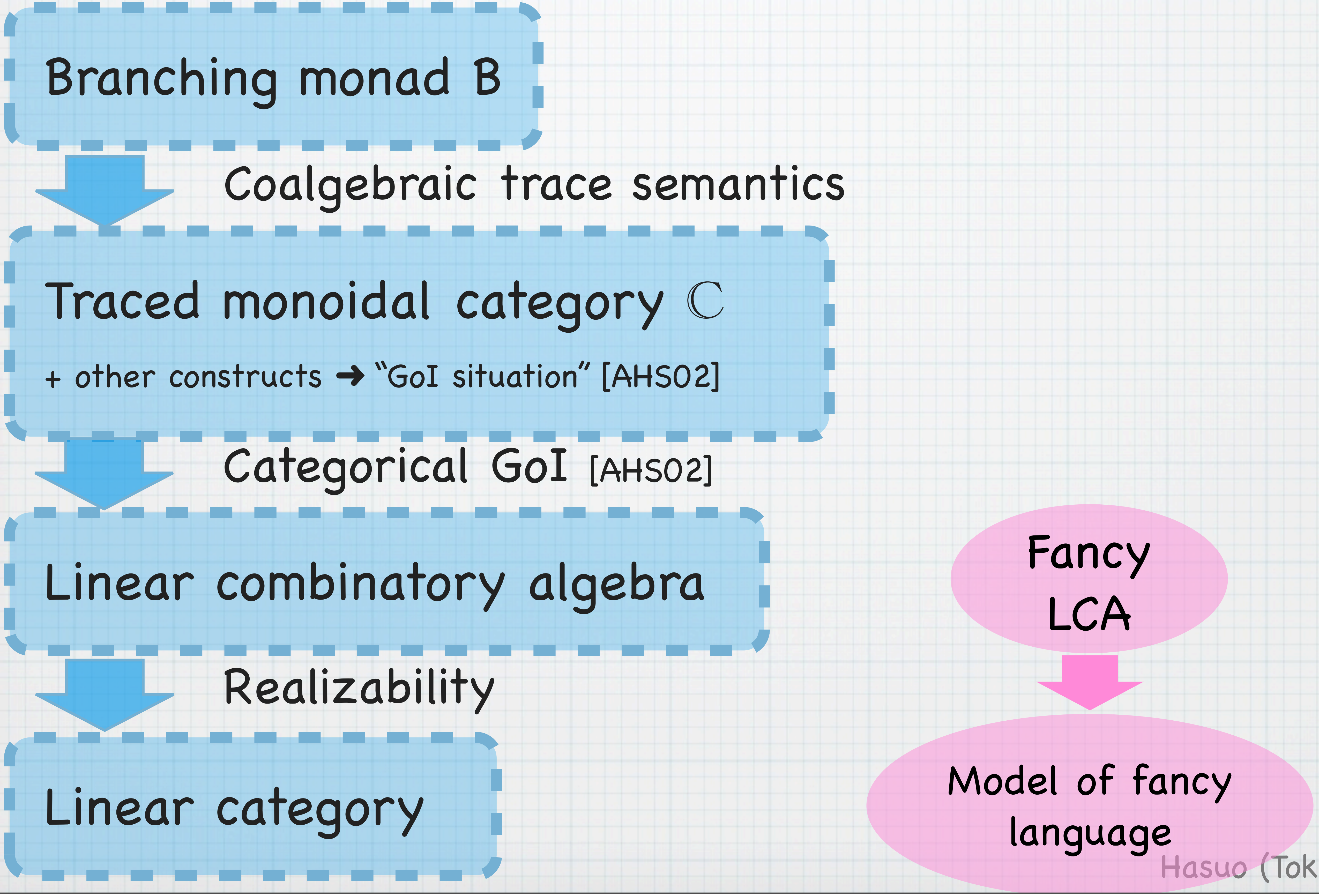
Linear combinatory algebra

Realizability

Linear category

Model of fancy language

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

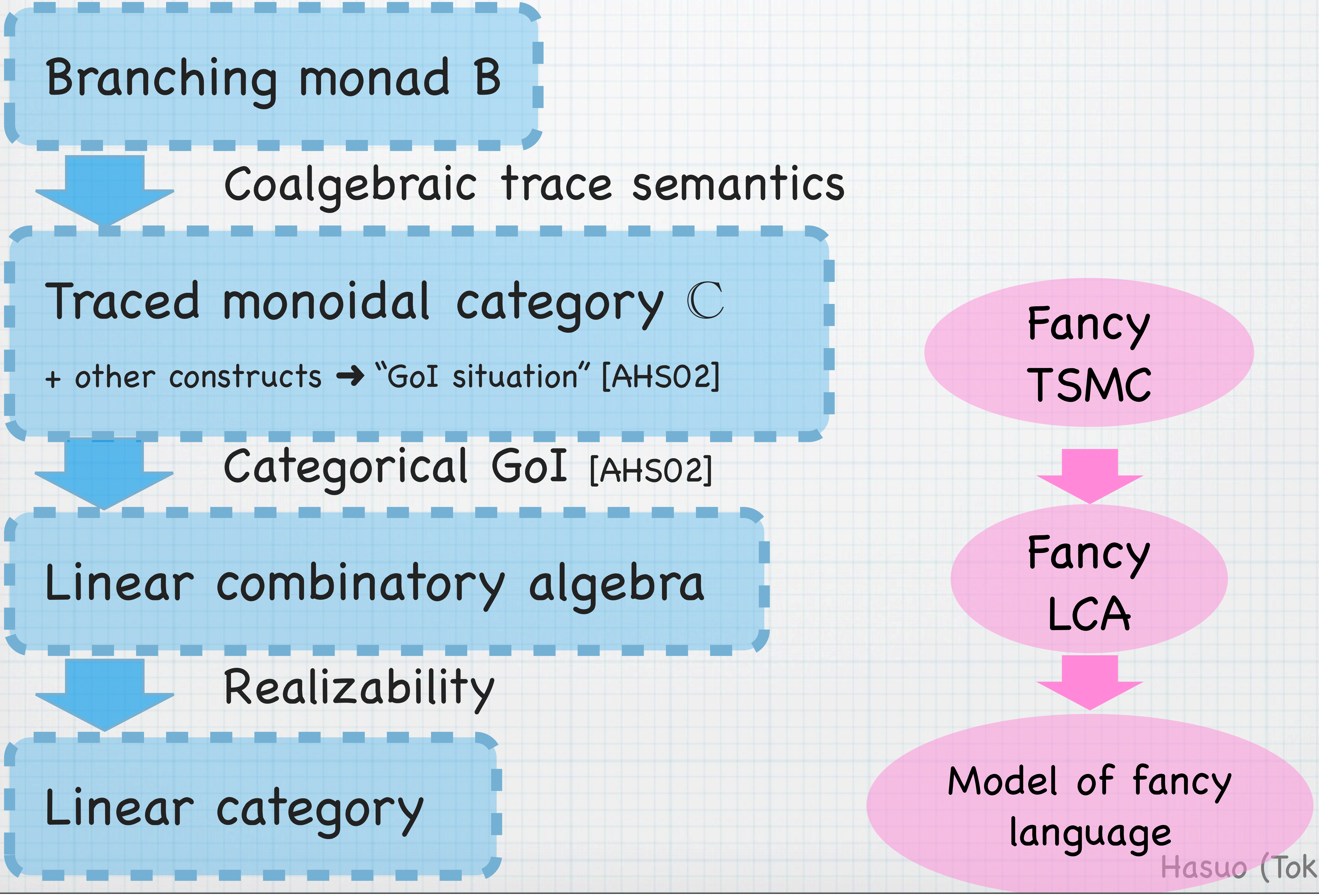
Realizability

Linear category

Fancy LCA

Model of fancy language

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➔ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

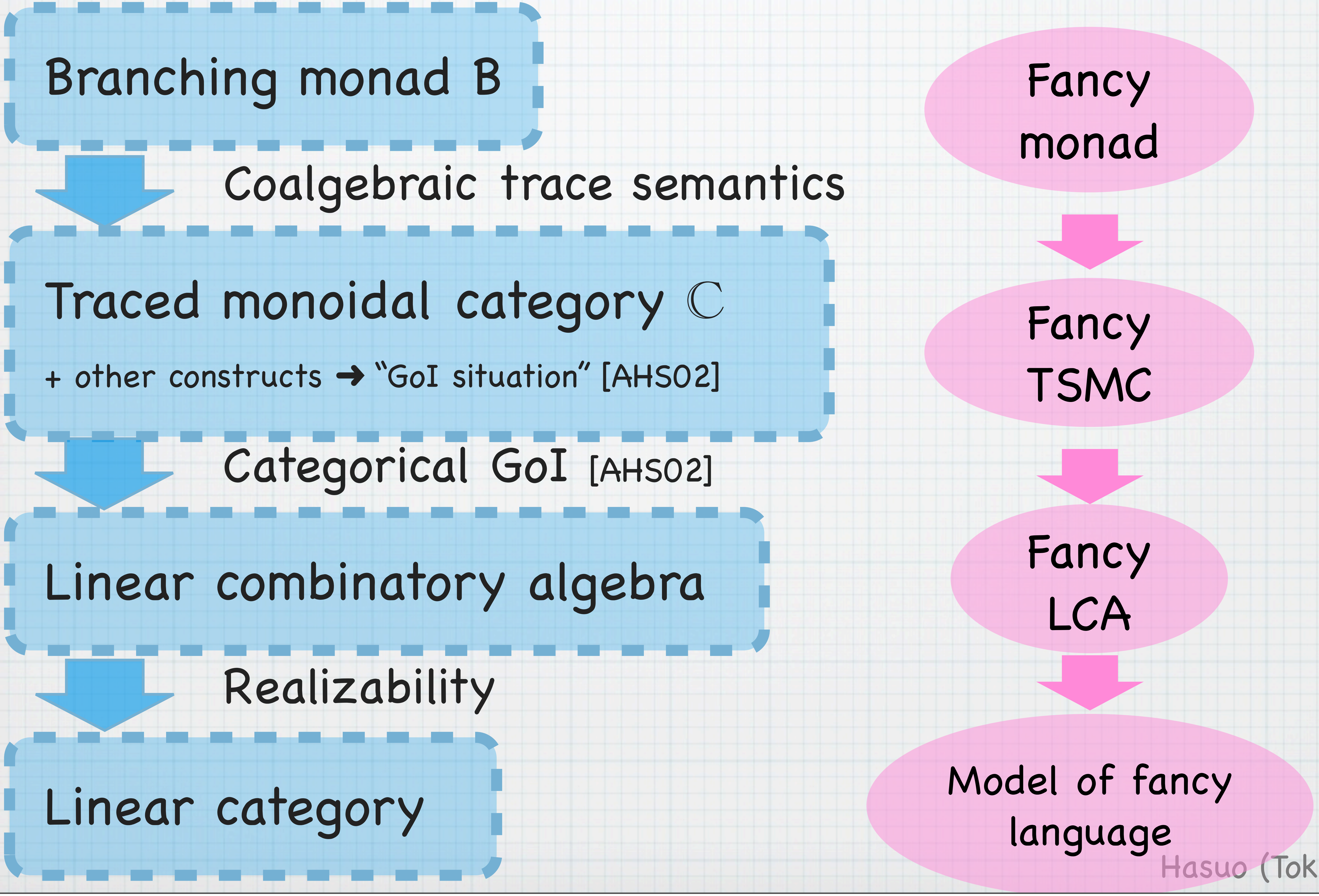
Linear category

Fancy TSMC

Fancy LCA

Model of fancy language

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➔ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Fancy monad

Fancy TSMC

Fancy LCA

Model of fancy language

What is Fancy, Nowadays?

What is Fancy, Nowadays?

- Biology?
- Hybrid systems?
    - Both discrete and continuous data, typically in **cyber-physical systems** (CPS)
    - ➜ Our approach via **non-standard analysis** [Suenaga, IH, ICALP'11] [IH, Suenaga, CAV'12] [Suenaga, Sekine, IH, POPL'13]
- Quantum?
    - Yes this worked!

Part 3

(Optional)

Realizability:

from Untyped to Typed

Realizability

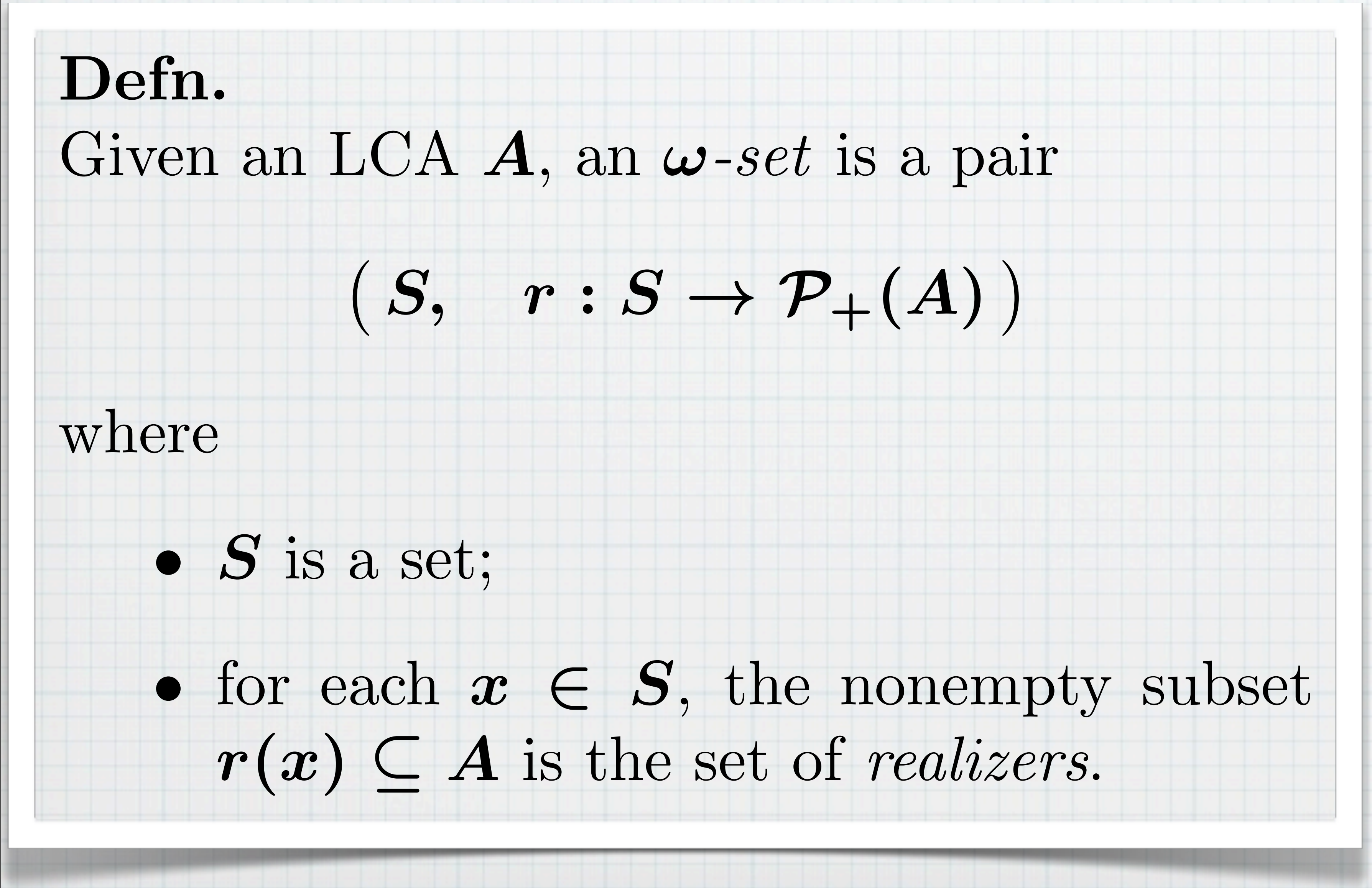
- Dates back to Kleene
- Cf. the Brouwer-Heyting-Kolmogorov (BHK) interpretation
  - A p'f of $A \wedge B$ is a pair: (p'f of A, p'f of B)
  - A p'f of $A \rightarrow B$ is a function carrying (p'f of A) to (p'f of B)
  - Proof = "realizer"

Realizability

- Our technical view on realizability: a construction
  - from a combinatory algebra,
  - of a categorical model of a typed calculus
- Here: construct a linear category from an LCA
- References:
  - [AL05] S. Abramsky and M. Lenisa, "Linear realizability and full completeness for typed lambda-calculi," APAA 2005.
  - [Hos07] N. Hoshino, "Linear realizability," CSL 2007.

Realizability

- Either by **ω-sets** (intuitive) or by **PERs** (tech. convenient)



Defn.
Given an LCA $\boldsymbol{A}$, an $\boldsymbol{\omega}$-*set* is a pair

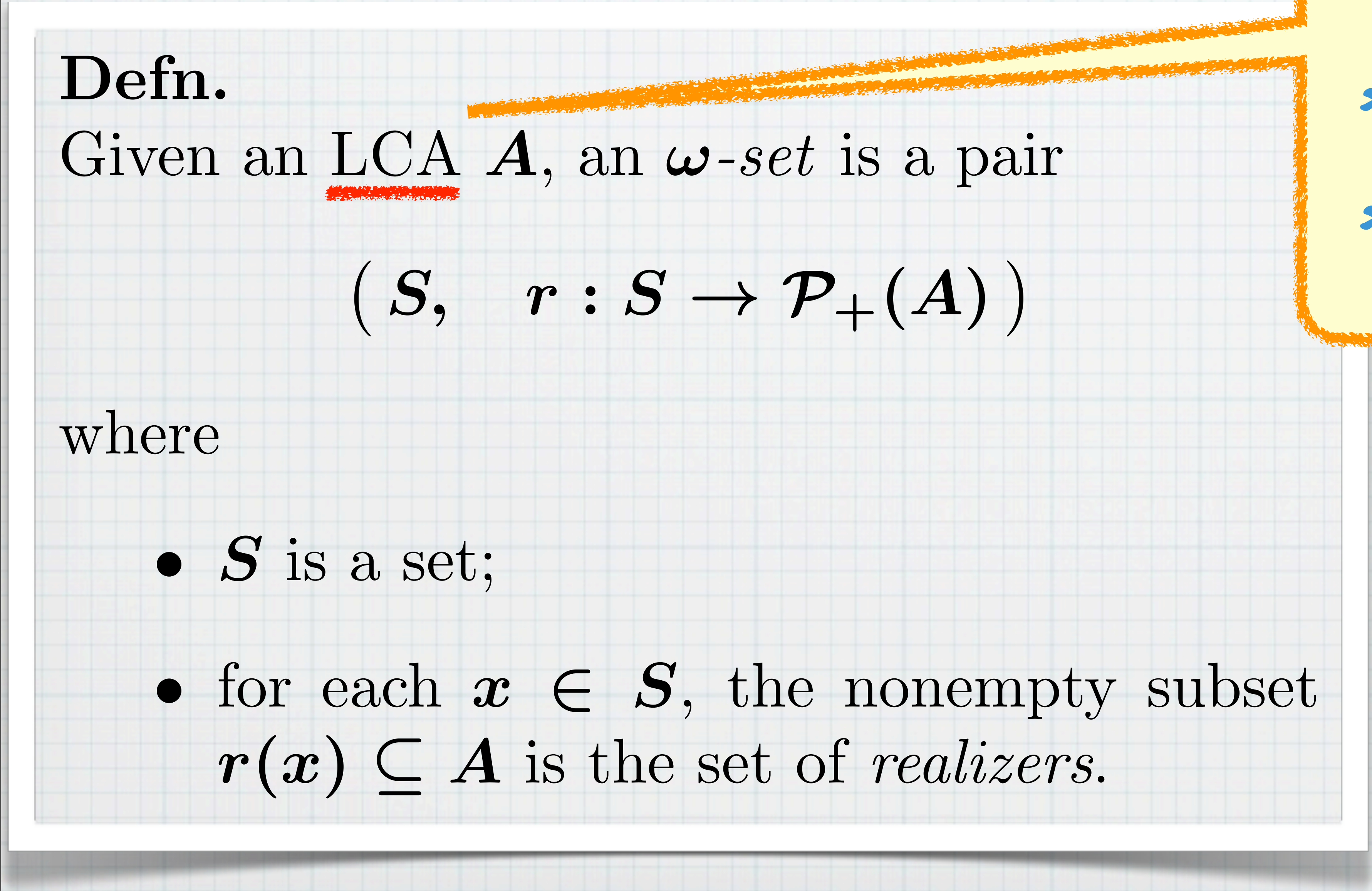
$$(\, \boldsymbol{S}, \quad \boldsymbol{r} : \boldsymbol{S} \to \boldsymbol{\mathcal{P}}_{+}(\boldsymbol{A}) \,)$$

where

- $\boldsymbol{S}$ is a set;
- for each $\boldsymbol{x} \in \boldsymbol{S}$, the nonempty subset $\boldsymbol{r}(\boldsymbol{x}) \subseteq \boldsymbol{A}$ is the set of *realizers*.

Realizability

- Either by **ω-sets** (intuitive) or by **PERs** (tech. convenient)



Defn.
Given an LCA $\boldsymbol{A}$, an $\boldsymbol{\omega}$-*set* is a pair

$$(\, \boldsymbol{S}, \quad \boldsymbol{r} : \boldsymbol{S} \to \mathcal{P}_+(\boldsymbol{A})\,)$$

where

- $\boldsymbol{S}$ is a set;
- for each $\boldsymbol{x} \in \boldsymbol{S}$, the nonempty subset $\boldsymbol{r}(\boldsymbol{x}) \subseteq \boldsymbol{A}$ is the set of *realizers*.

Could as well be a **partial combinatory algebra**. Its examples:

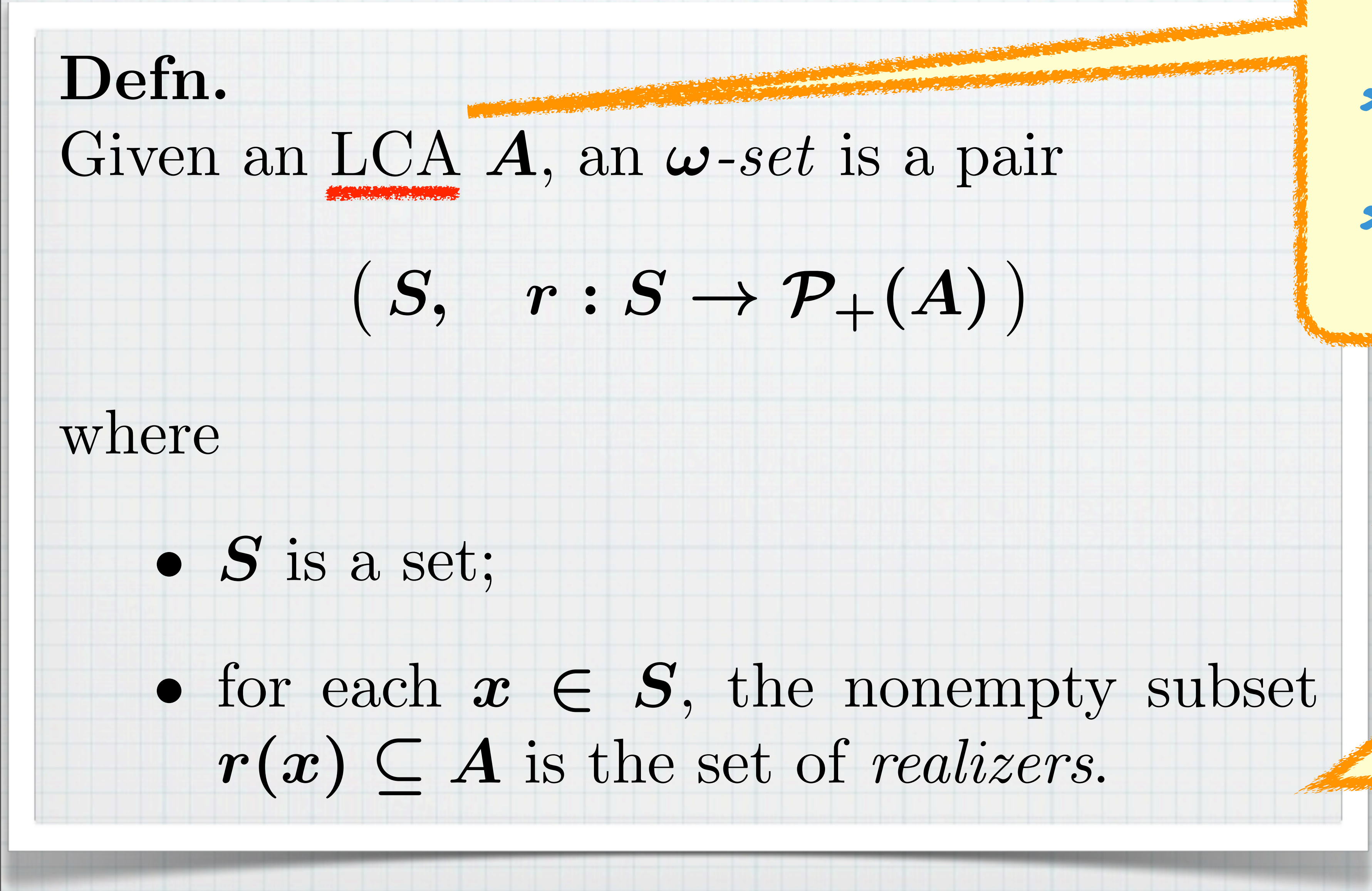
- $\mathbb{N}$ with n·m = comp(n,m)
- { closed λ-terms }

Realizability

- Either by **ω-sets** (intuitive) or by **PERs** (tech. convenient)

Could as well be a **partial combinatory algebra**. Its examples:

- $\mathbb{N}$ with n·m = comp(n,m)
- { closed λ-terms }



Defn.
Given an LCA $\boldsymbol{A}$, an $\boldsymbol{\omega}$-*set* is a pair

$$(\,\boldsymbol{S}, \quad \boldsymbol{r} : \boldsymbol{S} \to \mathcal{P}_{+}(\boldsymbol{A})\,)$$

where

- $\boldsymbol{S}$ is a set;
- for each $\boldsymbol{x} \in \boldsymbol{S}$, the nonempty subset $\boldsymbol{r}(\boldsymbol{x}) \subseteq \boldsymbol{A}$ is the set of *realizers*.

a ∈ r(x) :

- "realizes" x, or
- "witnesses existence of" x

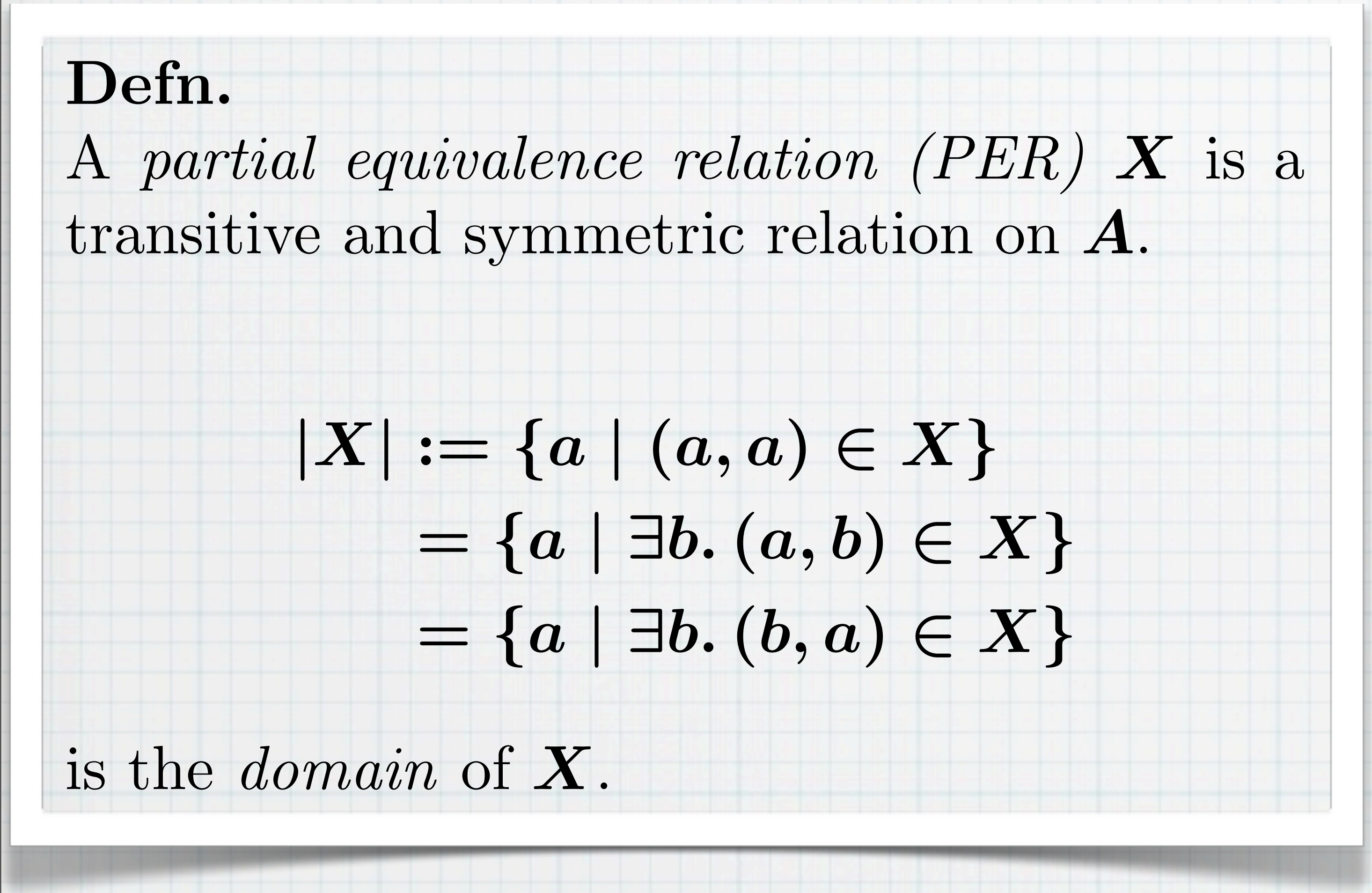
Realizability

Defn.
A *partial equivalence relation (PER)* $\boldsymbol{X}$ is a transitive and symmetric relation on $\boldsymbol{A}$.

$$\begin{aligned} |\boldsymbol{X}| &:= \{\boldsymbol{a} \mid (\boldsymbol{a}, \boldsymbol{a}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{b}, \boldsymbol{a}) \in \boldsymbol{X}\} \end{aligned}$$

is the *domain* of $\boldsymbol{X}$.

Realizability



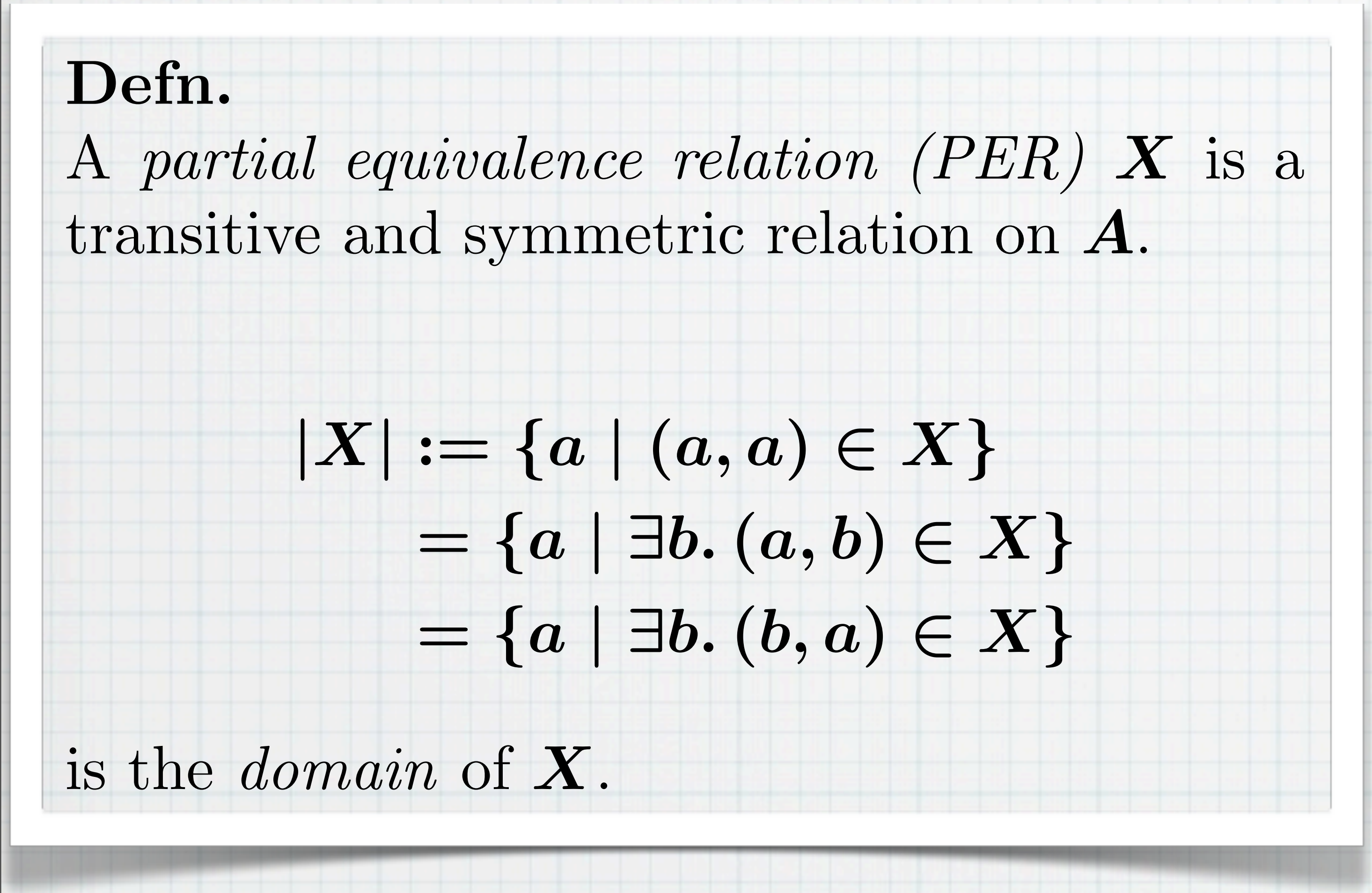
Defn.
A *partial equivalence relation (PER)* $\boldsymbol{X}$ is a transitive and symmetric relation on $\boldsymbol{A}$.

$$\begin{aligned} |\boldsymbol{X}| &:= \{\boldsymbol{a} \mid (\boldsymbol{a}, \boldsymbol{a}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{b}, \boldsymbol{a}) \in \boldsymbol{X}\} \end{aligned}$$

is the *domain* of $\boldsymbol{X}$.

- PER = eq. rel. - refl.

Realizability



Defn.
A *partial equivalence relation (PER)* $\boldsymbol{X}$ is a transitive and symmetric relation on $\boldsymbol{A}$.

$$\begin{aligned}|\boldsymbol{X}| &:= \{\boldsymbol{a} \mid (\boldsymbol{a},\boldsymbol{a}) \in \boldsymbol{X}\}\\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\,(\boldsymbol{a},\boldsymbol{b}) \in \boldsymbol{X}\}\\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\,(\boldsymbol{b},\boldsymbol{a}) \in \boldsymbol{X}\}\end{aligned}$$

is the *domain* of $\boldsymbol{X}$.

- PER = eq. rel. - refl.
- An eq. rel. when restricted to $|X|$

Realizability

Defn.
A *partial equivalence relation (PER)* $\boldsymbol{X}$ is a transitive and symmetric relation on $\boldsymbol{A}$.

$$\begin{aligned} |\boldsymbol{X}| &:= \{\boldsymbol{a} \mid (\boldsymbol{a},\boldsymbol{a}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{a},\boldsymbol{b}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{b},\boldsymbol{a}) \in \boldsymbol{X}\} \end{aligned}$$

is the *domain* of $\boldsymbol{X}$.

- PER = eq. rel. - refl.
- An eq. rel. when restricted to $|X|$
- PER to ω-set:

$$\Big(\, |\boldsymbol{X}|/\boldsymbol{X}, \quad |\boldsymbol{X}|/\boldsymbol{X} \xrightarrow{r} \mathcal{P}_{+}(\boldsymbol{A}) \,\Big)$$

with $\quad [\boldsymbol{a}] \overset{r}{\longmapsto} \{\boldsymbol{b} \mid (\boldsymbol{a},\boldsymbol{b}) \in \boldsymbol{X}\}$

Hasuo (Tokyo)

Realizability

Defn.
A *partial equivalence relation (PER)* $\boldsymbol{X}$ is a transitive and symmetric relation on $\boldsymbol{A}$.

$$\begin{aligned} |\boldsymbol{X}| &:= \{\boldsymbol{a} \mid (\boldsymbol{a},\boldsymbol{a}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{a},\boldsymbol{b}) \in \boldsymbol{X}\} \\ &= \{\boldsymbol{a} \mid \exists \boldsymbol{b}.\, (\boldsymbol{b},\boldsymbol{a}) \in \boldsymbol{X}\} \end{aligned}$$

is the *domain* of $\boldsymbol{X}$.

- PER = eq. rel. - refl.
- An eq. rel. when restricted to $|X|$
- PER to ω-set:

$$\Big(\ |\boldsymbol{X}|/\boldsymbol{X},\quad |\boldsymbol{X}|/\boldsymbol{X} \xrightarrow{r} \mathcal{P}_{+}(\boldsymbol{A})\ \Big)$$

with $\quad [\boldsymbol{a}] \overset{r}{\longmapsto} \{\boldsymbol{b} \mid (\boldsymbol{a},\boldsymbol{b}) \in \boldsymbol{X}\}$

- Also: ω-set to PER

$\mathbf{PER}_A$:
The Category of PERs

- **Obj.** A PER X on A

- **Arr.** The homset is

$$\mathbf{PER}_A(X, Y)$$

$$= \left\{ c \in A \;\middle|\; (x, x') \in X \Longrightarrow (cx, cx') \in Y \right\}$$

$$\left\{ (c, c') \;\middle|\; \forall x \in |X|.\ (cx, c'x) \in Y \right\}$$

- Thus:

- Often put:

PER$_A$:
The Category of PERs

- **Obj.** A PER X on A
- **Arr.** The homset is

All the valid **codes** c (well-dfd?)

$$\mathrm{PER}_A(X,Y)$$
$$= \Big\{ c \in A \,\Big|\, (x,x') \in X \Longrightarrow (cx, cx') \in Y \Big\} \Big/ \{(c,c') \mid \forall x \in |X|.\ (cx, c'x) \in Y\}$$

- Thus:
- Often put:

PER$_A$:

The Category of PERs

- **Obj.** A PER X on A
- **Arr.** The homset is

$$\mathbf{PER}_A(X,Y) = \left\{ c \in A \,\middle|\, (x,x') \in X \implies (cx, cx') \in Y \right\}$$

All the valid **codes** c (well-dfd?)

$$\left\{(c,c') \,\middle|\, \forall x \in |X|.\ (cx, c'x) \in Y\right\}$$

Modulo "the same function"

- Thus:
- Often put:

$\mathbf{PER}_A$: The Category of PERs

- **Obj.** A PER X on A
- **Arr.** The homset is

All the valid **codes** c (well-dfd?)

Modulo “the same function”

$$\mathbf{PER}_A(X,Y)$$
$$= \left\{ c \in A \,\middle|\, (x,x') \in X \Longrightarrow (cx, cx') \in Y \right\}$$

$$\left\{ (c,c') \,\middle|\, \forall x \in |X|.\ (cx, c'x) \in Y \right\}$$

- Thus: $[c] : X \longrightarrow Y$ (with $c \in A$)

Hasuo (Tokyo)

PER_A: The Category of PERs

- **Obj.** A PER X on A
- **Arr.** The homset is

Modulo "the same function"

All the valid **codes** c (well-dfd?)

$$\mathrm{PER}_A(X,Y)$$
$$= \left\{ c \in A \,\middle|\, (x,x') \in X \Longrightarrow (cx, cx') \in Y \right\} \Big/ \left\{ (c,c') \,\middle|\, \forall x \in |X|.\ (cx, c'x) \in Y \right\}$$

- Thus: $[c] : X \longrightarrow Y$ (with $c \in A$)
- Often put: $\mathrm{PER}_A(X,Y) = \left\{ (c,c') \,\middle|\, (x,x') \in X \Longrightarrow (cx, c'x') \in Y \right\}$

Type Constructors in $\mathbf{PER}_A$

Thm. ([AL05])
If $\boldsymbol{A}$ is an affine LCA, then $\mathbf{PER}_{\boldsymbol{A}}$ is a linear category.
Furthermore, $\mathbf{PER}_{\boldsymbol{A}}$ has finite products and coproducts.

- Linear category [Benton&Wadler,LICS'96][Bierman,TLCA'95]
  - Categorical model of linear logic/linear λ, with
    - Monoidal closed with $\boxtimes, \mathbf{I}, \multimap$
    - Linear exponential comonad $\mathbf{!}$

Type Constructors in $\mathbf{PER}_A$

with full K: Kxy=x

Thm. ([AL05])
If $\boldsymbol{A}$ is an affine LCA, then $\mathbf{PER}_{\boldsymbol{A}}$ is a linear category.
Furthermore, $\mathbf{PER}_{\boldsymbol{A}}$ has finite products and coproducts.

- Linear category [Benton&Wadler,LICS'96][Bierman,TLCA'95]
  - Categorical model of linear logic/linear λ, with
    - Monoidal closed with $\boxtimes, \mathbf{I}, \multimap$
    - Linear exponential comonad $\mathbf{!}$

Type Constructors in $\mathbf{PER}_A$

with full K: Kxy=x

Thm. ([AL05])
If $\boldsymbol{A}$ is an affine LCA, then $\mathbf{PER}_{\boldsymbol{A}}$ is a linear category.
Furthermore, $\mathbf{PER}_{\boldsymbol{A}}$ has finite products and coproducts.

- Linear category [Benton&Wadler,LICS'96][Bierman,TLCA'95]
  - Categorical model of linear logic/linear λ, with
    - Monoidal closed with $\boxtimes, \mathbf{I}, \multimap$
    - Linear exponential comonad !

Not ⊗,
for distinction

Hasuo (Tokyo)

Type Constructors in $\mathbf{PER}_A$

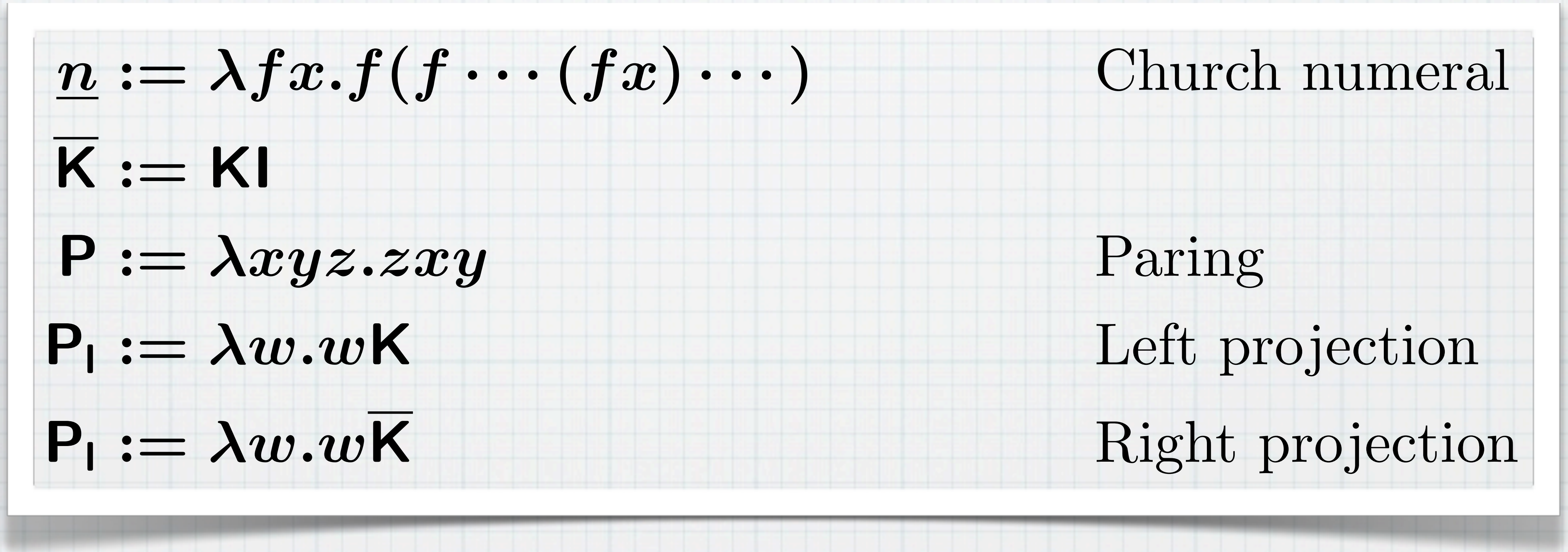
- How to get operators $\boxtimes, \times, +, \ldots$
  - Like “programming in untyped λ”!

Type Constructors in $\mathbf{PER}_A$

- How to get operators $\boxtimes, \times, +, \ldots$
  - Like "programming in untyped λ"!

Type Constructors in $\mathbf{PER}_A$

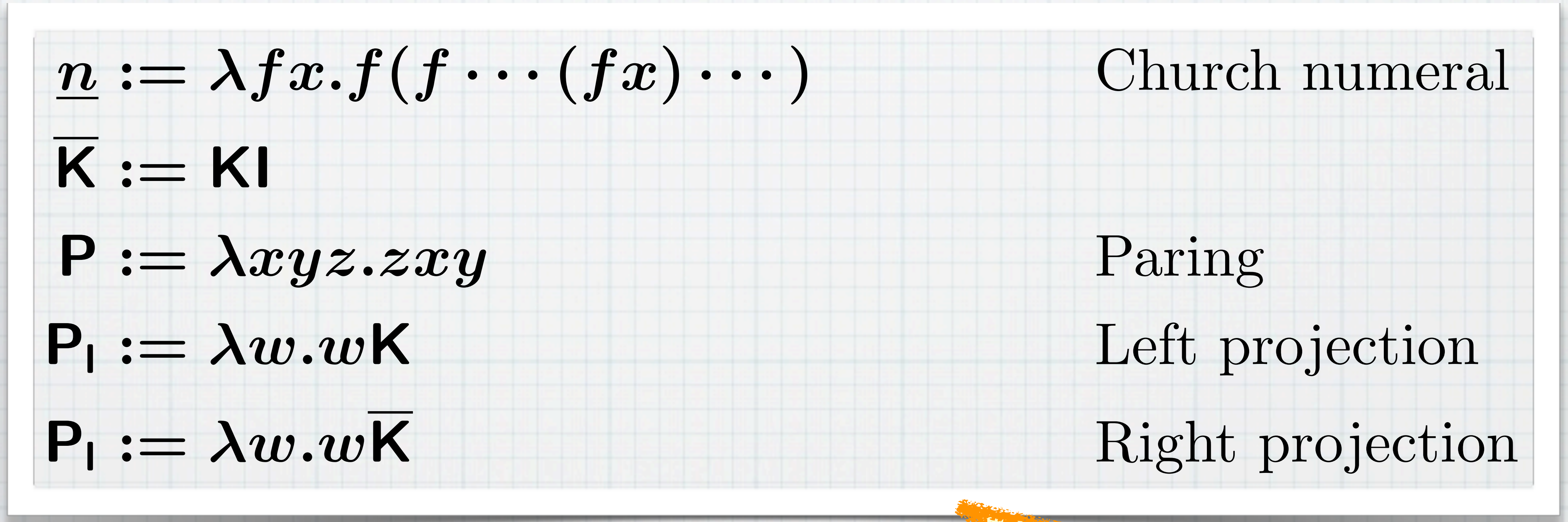
- How to get operators $\boxtimes, \times, +, \ldots$
  - Like "programming in untyped λ"!



$\underline{n} := \lambda fx.f(f\cdots(fx)\cdots)$	Church numeral
$\overline{\mathsf{K}} := \mathsf{KI}$	
$\mathsf{P} := \lambda xyz.zxy$	Paring
$\mathsf{P}_\mathsf{l} := \lambda w.w\mathsf{K}$	Left projection
$\mathsf{P}_\mathsf{l} := \lambda w.w\overline{\mathsf{K}}$	Right projection

Type Constructors in $\mathbf{PER}_A$

- How to get operators $\boxtimes, \times, +, \ldots$
  - Like "programming in untyped λ"!



$\underline{n}$	$:= \lambda f x . f(f \cdots (f x) \cdots)$	Church numeral
$\overline{\mathsf{K}}$	$:= \mathsf{KI}$	
P	$:= \lambda xyz.zxy$	Paring
P_l	$:= \lambda w.w\mathsf{K}$	Left projection
P_l	$:= \lambda w.w\overline{\mathsf{K}}$	Right projection

$$\mathsf{P}_\mathsf{l}(\mathsf{P}xy) = x$$

$$\mathsf{P}_\mathsf{r}(\mathsf{P}xy) = y$$

Type Constructors in $\mathbf{PER}_A$

- How to get operators $\boxtimes, \times, +, \ldots$
  - Like "programming in untyped λ"!

$\underline{n} := \lambda fx.f(f \cdots (fx) \cdots)$	Church numeral
$\overline{\mathsf{K}} := \mathsf{KI}$	
$\mathsf{P} := \lambda xyz.zxy$	Paring
$\mathsf{P}_\mathsf{l} := \lambda w.w\mathsf{K}$	Left projection
$\mathsf{P}_\mathsf{l} := \lambda w.w\overline{\mathsf{K}}$	Right projection

$$\mathsf{P}_\mathsf{l}(\mathsf{P}xy) = x$$

$$\mathsf{P}_\mathsf{r}(\mathsf{P}xy) = y$$

  - Cf. Combinaroty completeness

Type Constructors in $\mathbf{PER}_A$

$$\frac{\boldsymbol{X} \in \mathbf{PER}_{\boldsymbol{A}}}{\boldsymbol{X} \subseteq \boldsymbol{A} \times \boldsymbol{A}, \text{ sym., trans.}}$$

Type Constructors in $\mathbf{PER}_A$

$$\frac{X \in \mathrm{PER}_A}{X \subseteq A \times A,\ \text{sym., trans.}}$$

$$
\begin{aligned}
X \boxtimes Y &:= \Big\{ (\mathsf{P}xy, \mathsf{P}x'y') \,\Big|\, (x,x') \in X \wedge (y,y') \in Y \Big\} \\
X \times Y &:= \Big\{ \big(\mathsf{P}k_1(\mathsf{P}k_2 u),\ \mathsf{P}k'_1(\mathsf{P}k'_2 u')\big) \,\Big| \\
&\qquad\qquad (k_1 u, k'_1 u') \in X \wedge (k_2 u, k'_2 u') \in Y \Big\} \\
!\,X &:= \Big\{ (!\,x, !\,x') \,\Big|\, (x,x') \in X \Big\} \\
X + Y &:= \Big\{ (\mathsf{PK}x, \mathsf{PK}x') \,\Big|\, (x,x') \in X \Big\} \\
&\quad \cup \Big\{ (\mathsf{PK}y, \mathsf{PK}y') \,\Big|\, (y,y') \in Y \Big\} \\
X \multimap Y &:= \Big\{ (c,c') \,\Big|\, (x,x') \in X \Longrightarrow (cx, c'x') \in Y \Big\}
\end{aligned}
$$

Type Constructors in $\mathbf{PER}_A$

$$\frac{\boldsymbol{X} \in \mathbf{PER}_{\boldsymbol{A}}}{\boldsymbol{X} \subseteq \boldsymbol{A} \times \boldsymbol{A}, \text{ sym., trans.}}$$

multiplicative and

$$\boldsymbol{X} \boxtimes \boldsymbol{Y} \;:=\; \left\{ (\mathsf{P}xy, \mathsf{P}x'y') \;\middle|\; (x, x') \in \boldsymbol{X} \wedge (y, y') \in \boldsymbol{Y} \right\}$$

$$\boldsymbol{X} \times \boldsymbol{Y} \;:=\; \left\{ \big(\mathsf{P}k_1(\mathsf{P}k_2u), \mathsf{P}k_1'(\mathsf{P}k_2'u')\big) \;\middle|\; (k_1u, k_1'u') \in \boldsymbol{X} \wedge (k_2u, k_2'u') \in \boldsymbol{Y} \right\}$$

$$!\,\boldsymbol{X} \;:=\; \left\{ (!\,x, !\,x') \;\middle|\; (x, x') \in \boldsymbol{X} \right\}$$

$$\boldsymbol{X} + \boldsymbol{Y} \;:=\; \left\{ (\mathsf{PK}x, \mathsf{PK}x') \;\middle|\; (x, x') \in \boldsymbol{X} \right\} \cup \left\{ (\mathsf{PK}y, \mathsf{PK}y') \;\middle|\; (y, y') \in \boldsymbol{Y} \right\}$$

$$\boldsymbol{X} \multimap \boldsymbol{Y} \;:=\; \left\{ (c, c') \;\middle|\; (x, x') \in \boldsymbol{X} \implies (cx, c'x') \in \boldsymbol{Y} \right\}$$

Type Constructors in $\mathbf{PER}_A$

$$\frac{X \in \mathbf{PER}_A}{X \subseteq A \times A,\ \text{sym., trans.}}$$

multiplicative
and

additive
and

$$
\begin{aligned}
X \boxtimes Y &:= \left\{ (\mathsf{P}xy, \mathsf{P}x'y') \;\middle|\; (x,x') \in X \wedge (y,y') \in Y \right\} \\
X \times Y &:= \Big\{ \big(\mathsf{P}k_1(\mathsf{P}k_2u),\ \mathsf{P}k_1'(\mathsf{P}k_2'u') \big) \;\Big| \\
&\qquad\qquad (k_1u, k_1'u') \in X \wedge (k_2u, k_2'u') \in Y \Big\} \\
!\,X &:= \left\{ (!\,x, !\,x') \;\middle|\; (x,x') \in X \right\} \\
X + Y &:= \left\{ (\mathsf{PK}x, \mathsf{PK}x') \;\middle|\; (x,x') \in X \right\} \\
&\quad \cup \left\{ (\mathsf{PK}y, \mathsf{PK}y') \;\middle|\; (y,y') \in Y \right\} \\
X \multimap Y &:= \left\{ (c,c') \;\middle|\; (x,x') \in X \Longrightarrow (cx, c'x') \in Y \right\}
\end{aligned}
$$

Hasuo (Tokyo)

Type Constructors in $\mathbf{PER}_A$

$$\frac{\boldsymbol{X} \in \mathbf{PER}_{\boldsymbol{A}}}{\boldsymbol{X} \subseteq \boldsymbol{A} \times \boldsymbol{A}, \text{ sym., trans.}}$$

multiplicative and

$$\boldsymbol{X} \boxtimes \boldsymbol{Y} := \left\{ (\mathsf{P}xy, \mathsf{P}x'y') \;\middle|\; (x, x') \in \boldsymbol{X} \wedge (y, y') \in \boldsymbol{Y} \right\}$$

$$\boldsymbol{X} \times \boldsymbol{Y} := \left\{ \left(\mathsf{P}k_1(\mathsf{P}k_2 u),\ \mathsf{P}k'_1(\mathsf{P}k'_2 u') \right) \;\middle|\; (k_1 u, k'_1 u') \in \boldsymbol{X} \wedge (k_2 u, k'_2 u') \in \boldsymbol{Y} \right\}$$

additive and

CPS-style. k_1, k_2: "access methods"

$$!\,\boldsymbol{X} := \left\{ (!\,x, !\,x') \;\middle|\; (x, x') \in \boldsymbol{X} \right\}$$

$$\boldsymbol{X} + \boldsymbol{Y} := \left\{ (\mathsf{PK}x, \mathsf{PK}x') \;\middle|\; (x, x') \in \boldsymbol{X} \right\} \cup \left\{ (\mathsf{PK}y, \mathsf{PK}y') \;\middle|\; (y, y') \in \boldsymbol{Y} \right\}$$

$$\boldsymbol{X} \multimap \boldsymbol{Y} := \left\{ (c, c') \;\middle|\; (x, x') \in \boldsymbol{X} \Longrightarrow (cx, c'x') \in \boldsymbol{Y} \right\}$$

Summary: Realizability

Affine LCA A

$a \cdot b, \quad !\,a, \quad \mathsf{B}, \mathsf{C}, \mathsf{I}, \ldots$

$\longmapsto$

Linear category $\mathbf{PER}_A$

- $$\begin{tikzcd} \boldsymbol{X} \arrow[r, "{[\boldsymbol{c}]}"] & \boldsymbol{Y} \\ {[\boldsymbol{a}]} \arrow[r, mapsto] & {[\boldsymbol{c} \cdot \boldsymbol{a}]} \end{tikzcd}$$ $(a,c \in A)$
- Type constructors via "programming in untyped λ"
  - Symmetric monoidal closed $\boxtimes, \mathbf{I}, \multimap$
  - Finite product, coproduct

Summary: Realizability

Affine LCA A

$a \cdot b$, $!\,a$, $\mathsf{B}, \mathsf{C}, \mathsf{I}, \ldots$

$\longmapsto$

Linear category $\mathbf{PER}_A$

- $$\begin{tikzcd} X \arrow[r, "{[c]}"] & Y \\ {[a]} \arrow[r, mapsto] & {[c \cdot a]} \end{tikzcd}$$ $(a, c \in A)$

- Type constructors via “programming in untyped λ”
  - Symmetric monoidal closed $\boxtimes, \mathbf{I}, \multimap$
  - Finite product, coproduct

Not ⊗,
for distinction

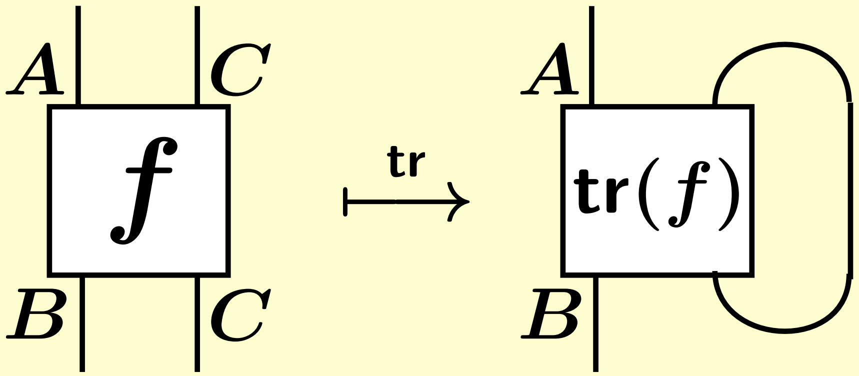
The Categorical GoI Workflow

Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



Categorical GoI [AHS02]

Linear combinatory algebra

- ✱ Applicative str. + combinators
- ✱ Model of untyped calculus

Realizability

Linear category

Model of typed calculus

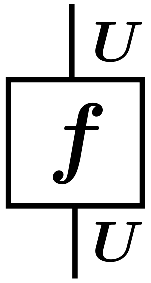
The Categorical GoI Workflow

Branching monad B

Coalgebraic trace semantics

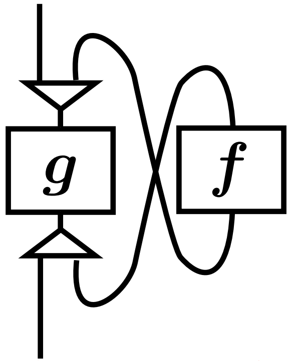
Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]



$\in \mathbb{C}(U,U)$

$g \cdot f =$



Categorical GoI [AHS02]

Linear combinatory algebra

- Applicative str. + combinators
- Model of untyped calculus

Realizability

Linear category

Model of typed calculus

Summary: Realizability

Affine LCA A

$a \cdot b, \quad !\,a, \quad \mathsf{B}, \mathsf{C}, \mathsf{I}, \ldots$

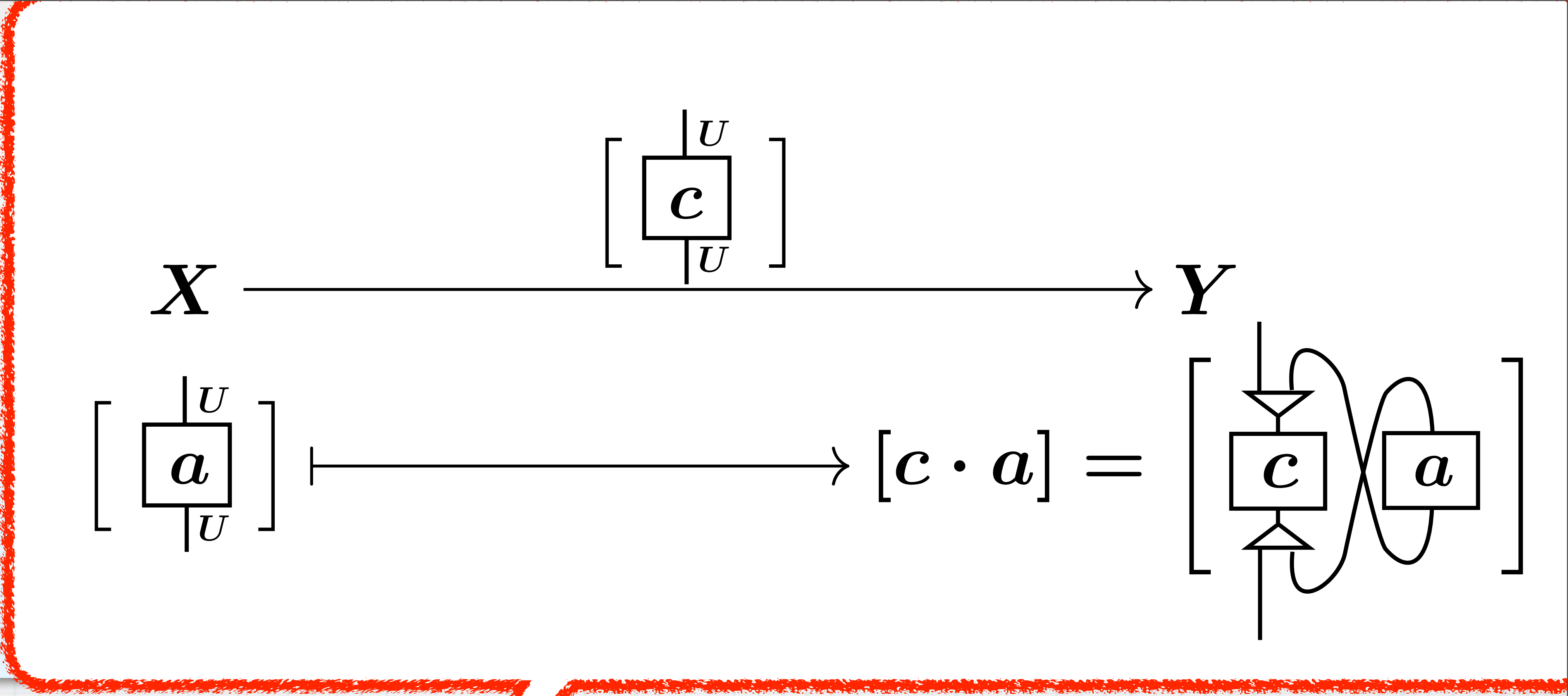
Linear category $\mathbf{PER}_A$

- $$\begin{tikzcd} X \arrow[r, "{[c]}"] & Y \\ {[a]} \arrow[r, mapsto] & {[c \cdot a]} \end{tikzcd}$$ $(a{,}c{\in}A)$
- Type constructors via "programming in untyped λ"
  - Symmetric monoidal closed $\boxtimes, \mathbf{I}, \multimap$
  - Finite product, coproduct

Not ⊗,
for distinction

Affine LCA

$a \cdot b, \quad !\,a, \quad$ B, C



Linear category $\mathbf{PER}_A$

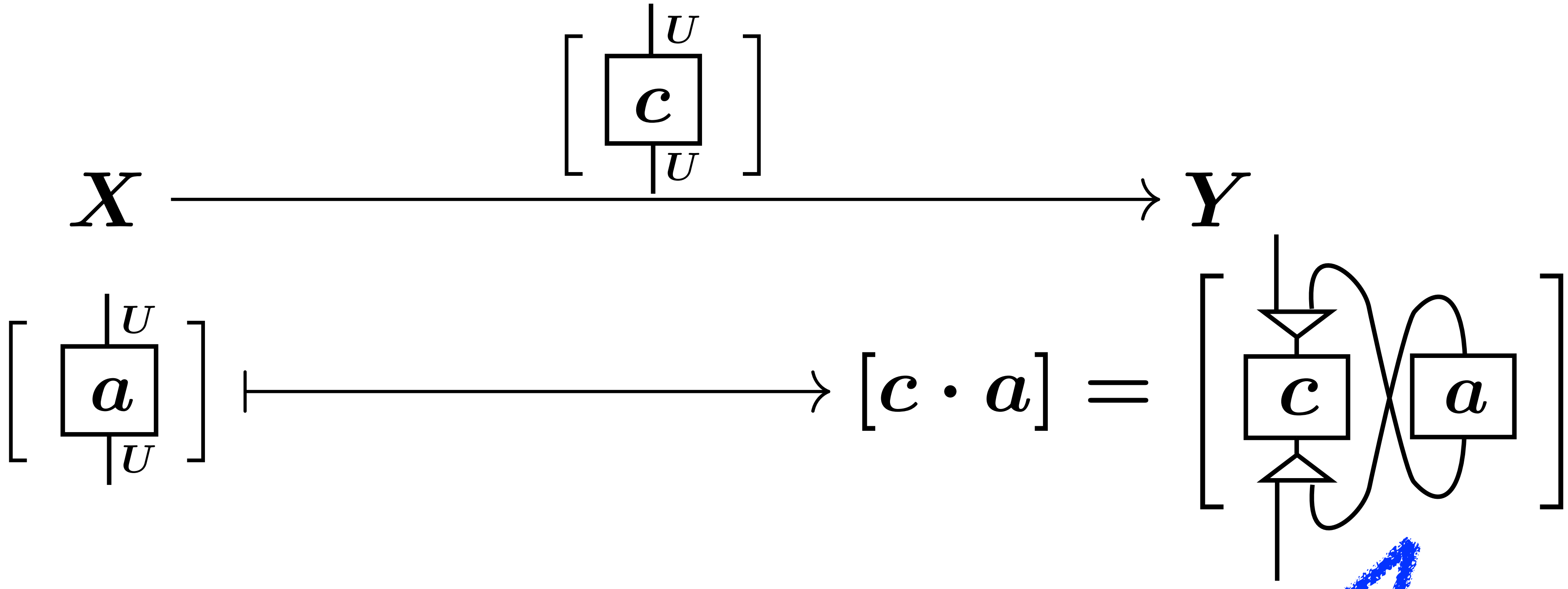
- $$\begin{tikzcd} X \arrow[r, "{[c]}"] & Y \\ {[a]} \arrow[r, mapsto] & {[c \cdot a]} \end{tikzcd}$$ $(a, c \in A)$

- Type constructors via “programming in untyped λ”
  - Symmetric monoidal closed $\boxtimes, \mathbf{I}, \multimap$
  - Finite product, coproduct

Not ⊗, for distinction

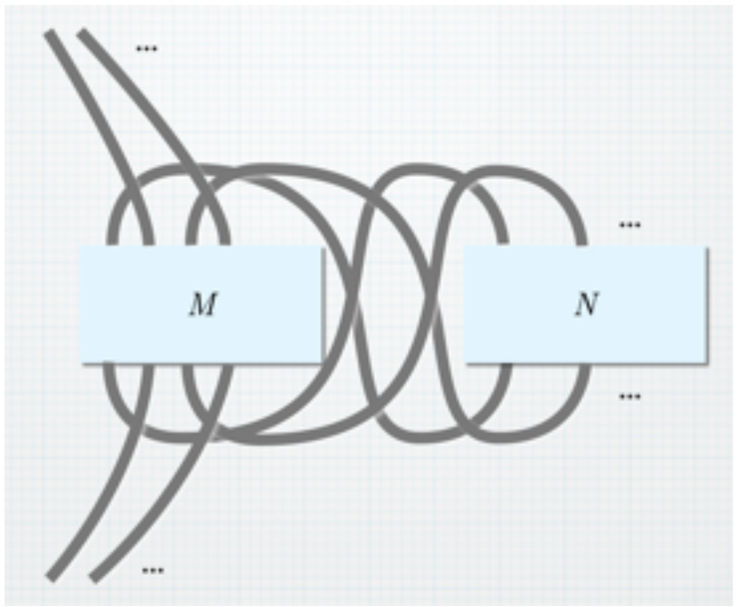
Affine LCA

$a \cdot b, \quad !\,a, \quad$ B, C



Linear category $\mathbf{PER}_A$

- $$\begin{tikzcd} X \arrow[r, "{[c]}"] & Y \\ {[a]} \arrow[r, mapsto] & {[c \cdot a]} \end{tikzcd}$$ $(a,c \in A)$

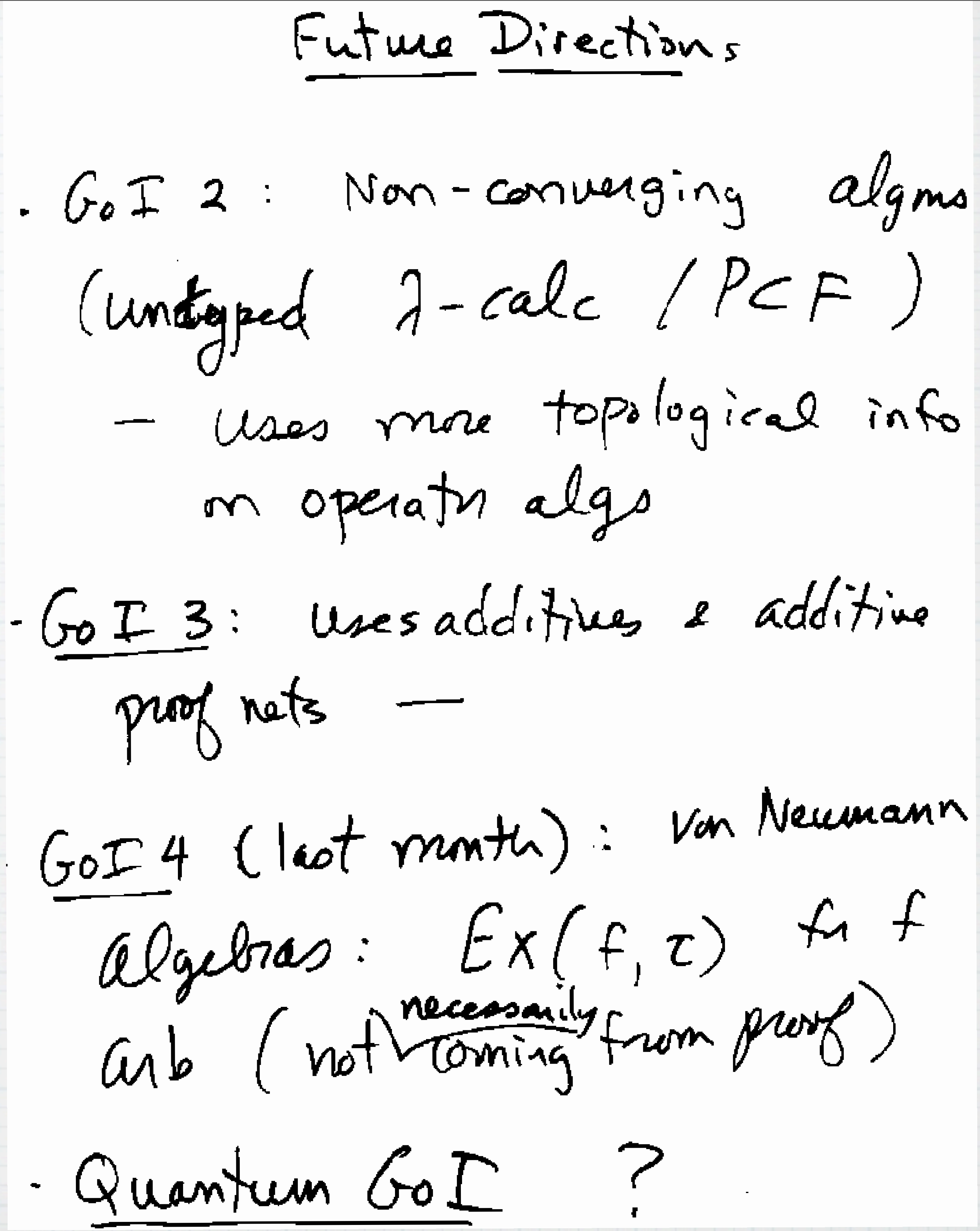


- Type constructors via "programming in untyped λ"
  - Symmetric monoidal closed $\boxtimes, \mathbf{I}, \multimap$
  - Finite product, coproduct

Not ⊗,
for distinction

Part 4

Phil Scott.
Tutorial on Geometry of Interaction, FMCS 2004.
Page 47/47



Future Directions

- GoI 2: Non-converging algms (untyped λ-calc / PCF)
  - uses more topological info on operatr algs
- GoI 3: uses additives & additive proof nets —
- GoI 4 (last month): von Neumann algebras: $Ex(f, \tau)$ for f arb (not necessarily coming from proof)
- Quantum GoI ?

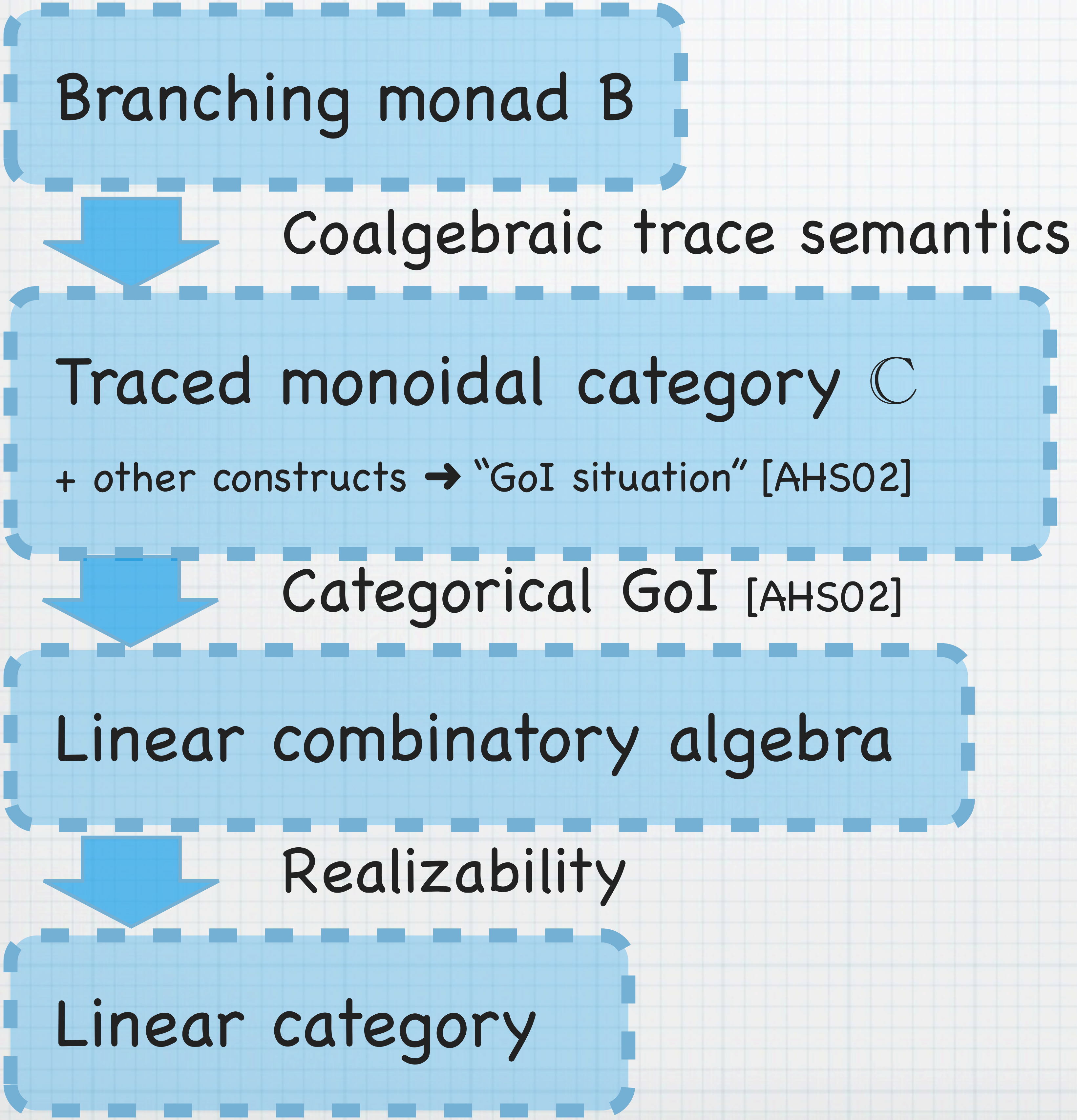
Part 4

Phil Scott.
Tutorial on Geometry of
Interaction, FMCS 2004.
Page 47/47

Future Directions

- GoI 2: Non-converging algms (untyped λ-calc / PCF)
  - uses more topological info on operatr algs
- GoI 3: uses additives & additive proof nets —
- GoI 4 (last month): von Neumann algebras: $Ex(f, \tau)$ for f arb (not necessarily coming from proof)
- $\boxed{\text{Quantum GoI}}$?

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

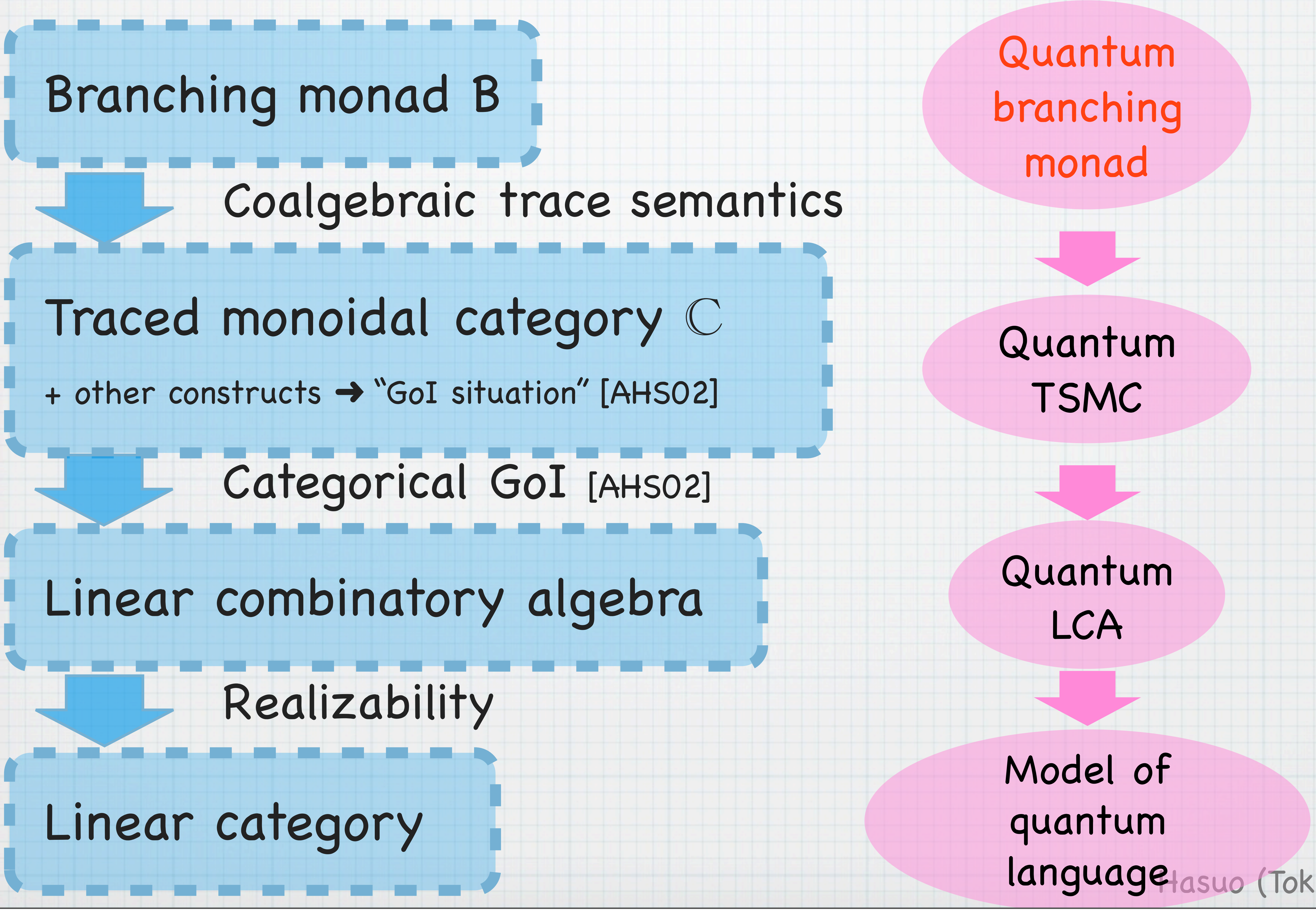
Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

The Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Quantum branching monad

Quantum TSMC

Quantum LCA

Model of quantum language

The Quantum Branching Monad

$$\boldsymbol{\mathcal{Q}Y} = \left\{ \boldsymbol{c : Y \to} \prod_{\boldsymbol{m,n} \in \mathbb{N}} \mathbf{QO}_{\boldsymbol{m,n}} \;\middle|\; \text{the trace condition} \right\}$$

The Quantum Branching M

$$\mathbf{QO}_{m,n} := \left\{ \begin{array}{l} \text{quantum operations,} \\ \text{from dim. } m \text{ to dim. } n \end{array} \right\}$$

$$QY = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} \underline{\mathbf{QO}_{m,n}} \;\middle|\; \text{the trace condition} \right\}$$

The Quantum Branching M

$$\mathbf{QO}_{m,n} := \left\{ \begin{array}{l} \text{quantum operations,} \\ \text{from dim. } \boldsymbol{m} \text{ to dim. } \boldsymbol{n} \end{array} \right\}$$

$$\boldsymbol{QY} = \Bigl\{ \boldsymbol{c} : \boldsymbol{Y} \to \prod_{\boldsymbol{m},\boldsymbol{n}\in\mathbb{N}} \underline{\mathbf{QO}_{\boldsymbol{m},\boldsymbol{n}}} \;\Big|\; \underline{\text{the trace condition}} \Bigr\}$$

$$\sum_{\boldsymbol{y}\in\boldsymbol{Y}} \sum_{\boldsymbol{n}\in\mathbb{N}} \mathbf{tr}\bigl[(\boldsymbol{c}(\boldsymbol{y}))_{\boldsymbol{m},\boldsymbol{n}}(\boldsymbol{\rho})\bigr] \leq \mathbf{1} \ ,$$

$$\forall \boldsymbol{m} \in \mathbb{N},\ \forall \boldsymbol{\rho} \in \boldsymbol{D}_{\boldsymbol{m}}.$$

The Quantum Branching M

$$\mathbf{QO}_{m,n} := \left\{ \begin{array}{l} \text{quantum operations,} \\ \text{from dim. } m \text{ to dim. } n \end{array} \right\}$$

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \underline{\mathbf{QO}_{m,n}} \;\middle|\; \underline{\text{the trace condition}} \right\}$$

$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \leq 1 \;,$$

$$\forall m \in \mathbb{N},\ \forall \rho \in D_m .$$

- Compare with

$$\mathcal{P}Y = \{ c : Y \to 2 \}$$

$$\mathcal{D}Y = \left\{ c : Y \to [0,1] \;\middle|\; \sum_{y\in Y} c(y) \leq 1 \right\}$$

The Quantum Branching M

$$\mathbf{QO}_{m,n} := \left\{ \begin{array}{l} \text{quantum operations,} \\ \text{from dim. } m \text{ to dim. } n \end{array} \right\}$$

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} \mathbf{QO}_{m,n} \;\middle|\; \text{the trace condition} \right\}$$

$$\sum_{y \in Y} \sum_{n \in \mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \leq 1 \;,$$

$$\forall m \in \mathbb{N},\ \forall \rho \in D_m .$$

- Compare with

$$\mathcal{P}Y = \left\{ c : Y \to 2 \right\}$$

$$\mathcal{D}Y = \left\{ c : Y \to [0,1] \;\middle|\; \sum_{y \in Y} c(y) \leq 1 \right\}$$

The Quantum Branching M

$$\mathbf{QO}_{m,n} := \left\{ \begin{array}{l} \text{quantum operations,} \\ \text{from dim. } m \text{ to dim. } n \end{array} \right\}$$

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \underline{\mathbf{QO}_{m,n}} \;\middle|\; \underline{\text{the trace condition}} \right\}$$

$$\sum_{y\in Y} \sum_{n\in\mathbb{N}} \mathbf{tr}\left[\left(c(y)\right)_{m,n}(\rho)\right] \leq 1 \ ,$$

$$\forall m \in \mathbb{N},\ \forall \rho \in D_m .$$

- Compare with

$$\mathcal{P}Y = \left\{ c : Y \to 2 \right\}$$

$$\mathcal{D}Y = \left\{ c : Y \to [0,1] \;\middle|\; \underline{\sum_{y\in Y} c(y) \leq 1} \right\}$$

The Quantum Branching Monad

$$\mathcal{Q}Y = \Big\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\Big|\; \underline{\text{the trace condition}} \Big\}$$

$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[(c(y))_{m,n}(\rho)\big] \le 1 \;, \qquad \forall m\in\mathbb{N},\ \forall\rho\in D_m .$$

$$\begin{array}{c} X \xrightarrow{f} Y \;\text{ in } \mathcal{K}\ell(\mathcal{Q}) \\ \hline\hline X \xrightarrow{f} \mathcal{Q}Y \;\text{ in } \mathbf{Sets} \end{array}$$

- Given $x \in X$, $y \in Y$, $m \in \mathbb{N}$, $n \in \mathbb{N}$ determines a quantum operation

$$\Big(f(x)(y) \Big)_{m,n} \;:\; D_m \to D_n$$

- Subject to the **trace condition**

The Quantum Branching Monad

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\middle|\; \text{the trace condition} \right\}$$

$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[(c(y))_{m,n}(\rho)\big] \le 1 \ , \qquad \forall m\in\mathbb{N},\ \forall\rho\in D_m .$$

$$\begin{array}{c} X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q}) \\ \hline\hline X \xrightarrow{f} \mathcal{Q}Y \ \text{ in } \mathbf{Sets} \end{array}$$

- Given $x\in X$, $y\in Y$, $m\in\mathbb{N}$, $n\in\mathbb{N}$ determines a quantum operation

$$\Big(\, f(x)(y) \,\Big)_{m,n} \ : \ D_m \to D_n$$

- Subject to the **trace condition**

Any opr. on quantum data:

combination of

- preparation
- unitary transf.
- measurement

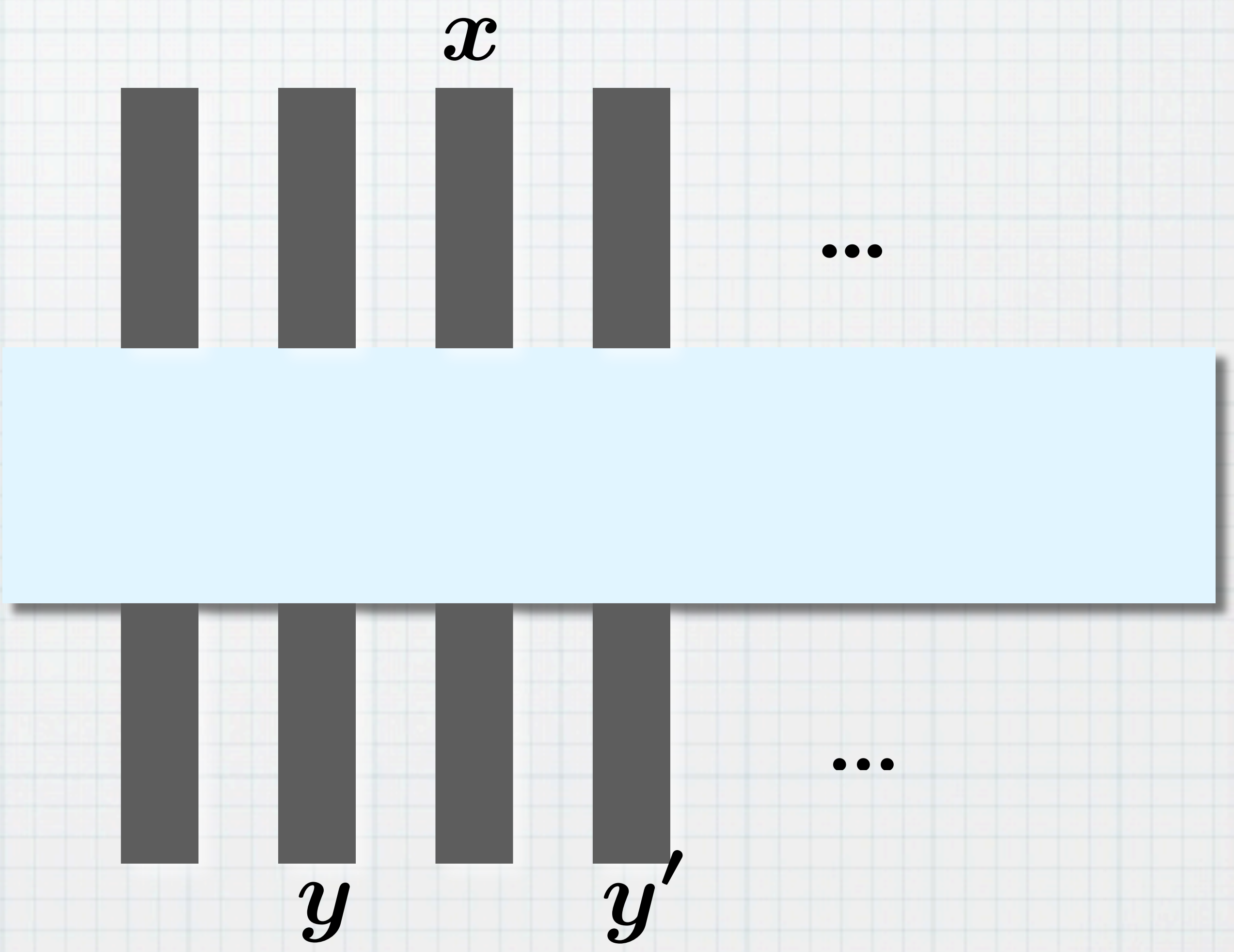
The Quantum Branching Monad

$$\mathcal{Q}Y = \Big\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\Big|\; \text{the trace condition} \Big\}$$

$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \le 1\ , \quad \forall m \in \mathbb{N},\ \forall \rho \in D_m.$$

$$\frac{X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q})}{X \xrightarrow{f} \mathcal{Q}Y \ \text{ in Sets}}$$

- Given $x \in X$, $y \in Y$, $m \in \mathbb{N}$, $n \in \mathbb{N}$ determines a quantum operation $\Big(f(x)(y)\Big)_{m,n}$
- **trace cond.:**



The Quantum Branching Monad

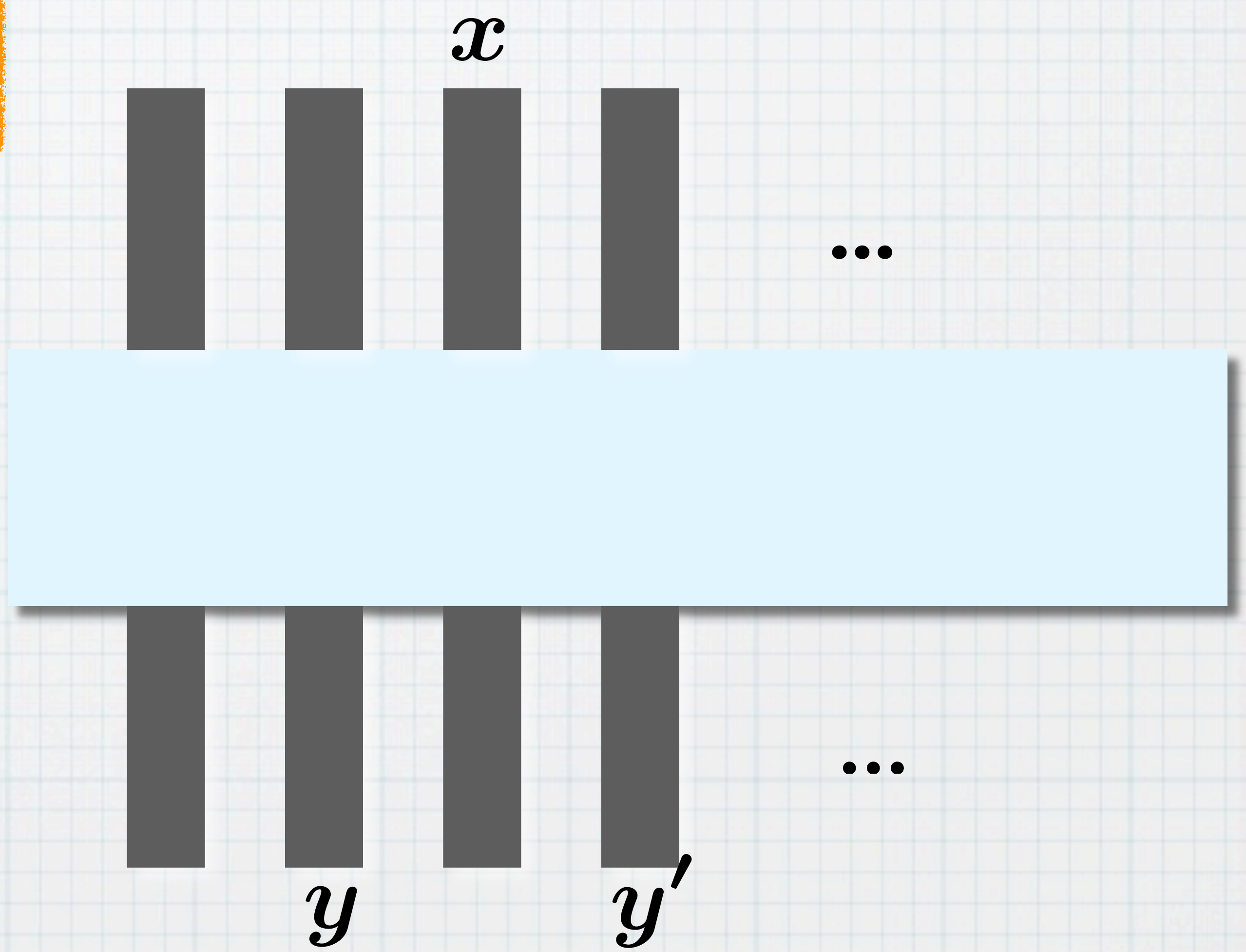
$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\middle|\; \text{the trace condition} \right\}$$

$$\sum_{y\in Y} \sum_{n\in\mathbb{N}} \mathbf{tr}\Big[\big(c(y)\big)_{m,n}(\rho)\Big] \le 1 \;, \qquad \forall m\in\mathbb{N},\ \forall \rho \in D_m .$$

$$\begin{array}{c} X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q}) \\ \hline\hline X \xrightarrow{f} \mathcal{Q}Y \ \text{ in Sets} \end{array}$$

entrance exit in dim. out dim.

- Given $x\in X,\ y\in Y,\ m\in\mathbb{N},\ n\in\mathbb{N}$ determines a quantum operation $\Big(f(x)(y)\Big)_{m,n}$
- **trace cond.:**



The Quantum Branching Monad

$$\frac{X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q})}{X \xrightarrow{f} \mathcal{Q}Y \ \text{ in Sets}}$$

entrance

exit

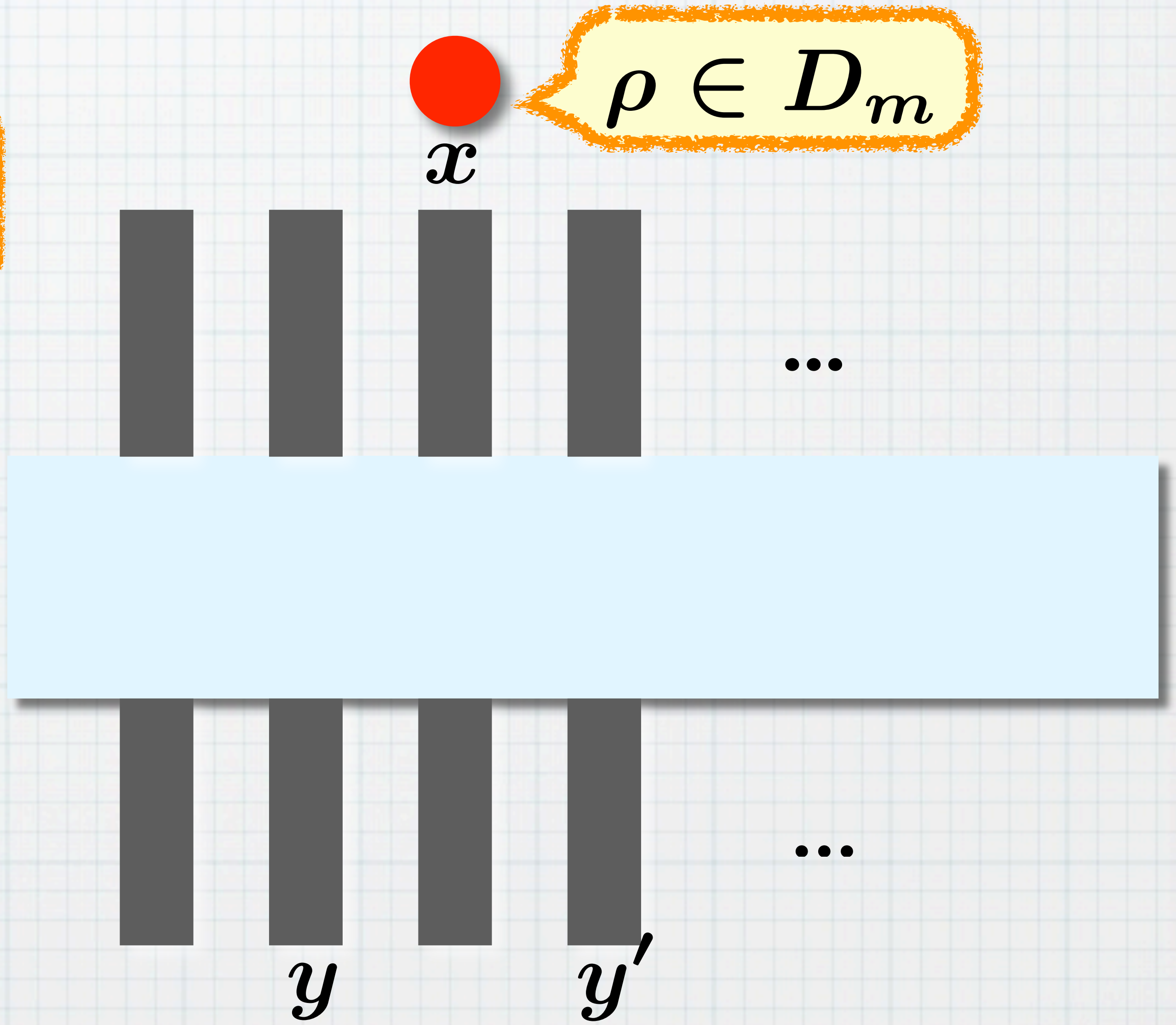
in dim.

out dim.

- Given $x \in X$, $y \in Y$, $m \in \mathbb{N}$, $n \in \mathbb{N}$ determines a quantum operation $\big(f(x)(y)\big)_{m,n}$
- **trace cond.:**

$$\mathcal{Q}Y = \Big\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \ \Big| \ \text{the trace condition} \Big\}$$

$$\sum_{y\in Y} \sum_{n\in\mathbb{N}} \mathrm{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \le 1 \ , \qquad \forall m \in \mathbb{N},\ \forall \rho \in D_m .$$



The Quantum Branching Monad

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\middle|\; \text{the trace condition} \right\}$$

$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \le 1 \;,$$

$$\forall m \in \mathbb{N},\ \forall \rho \in D_m .$$

$$\frac{X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q})}{X \xrightarrow{f} \mathcal{Q}Y \ \text{ in } \mathbf{Sets}}$$

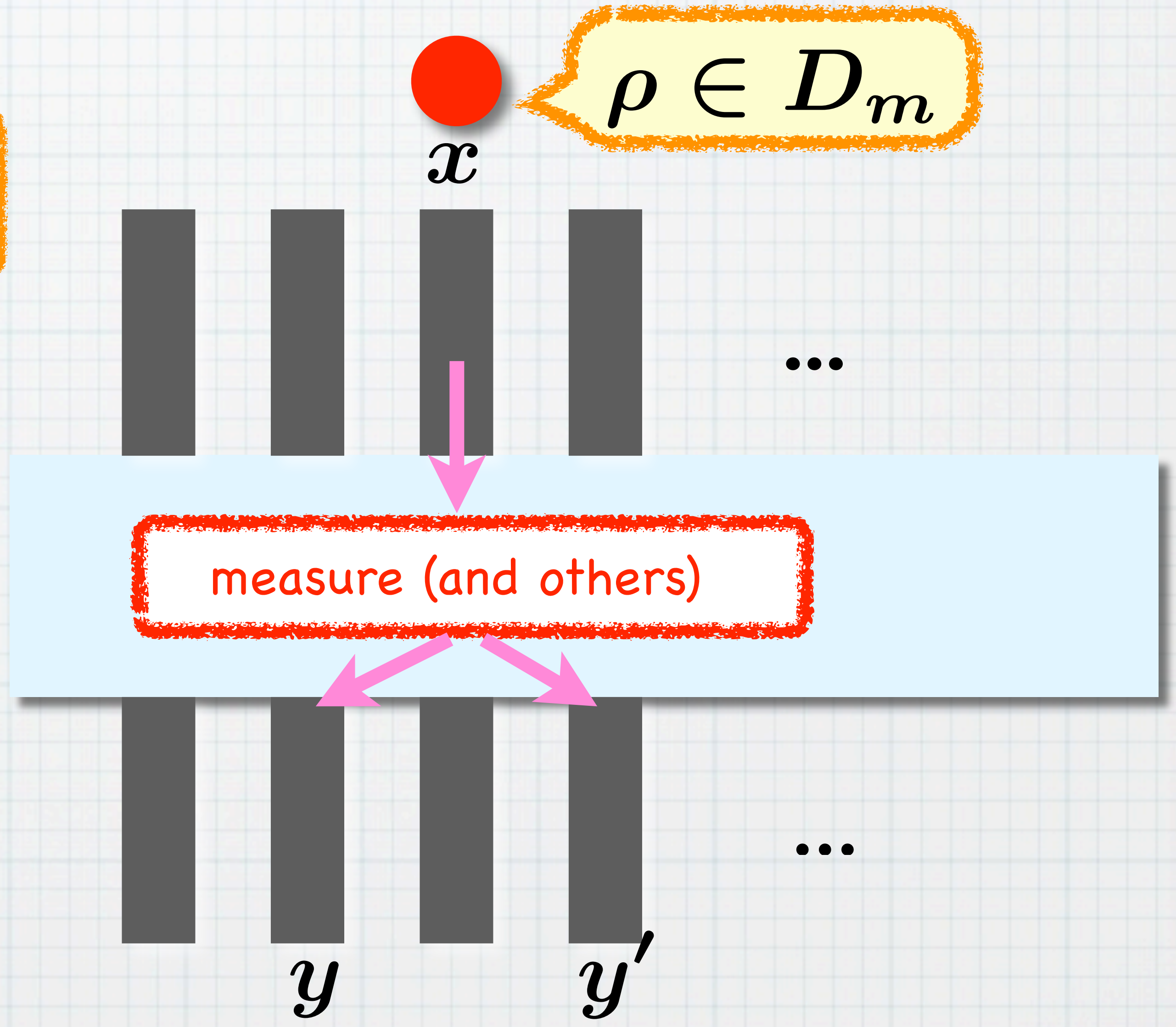
entrance

exit

in dim.

out dim.

- * Given $x \in X$, $y \in Y$, $m \in \mathbb{N}$, $n \in \mathbb{N}$ determines a quantum operation $\big(f(x)(y)\big)_{m,n}$
- * **trace cond.:**



$$\mathcal{Q}Y = \Big\{ c : Y \to \prod_{m,n \in \mathbb{N}} \mathbf{QO}_{m,n} \;\Big|\; \text{the trace condition} \Big\}$$

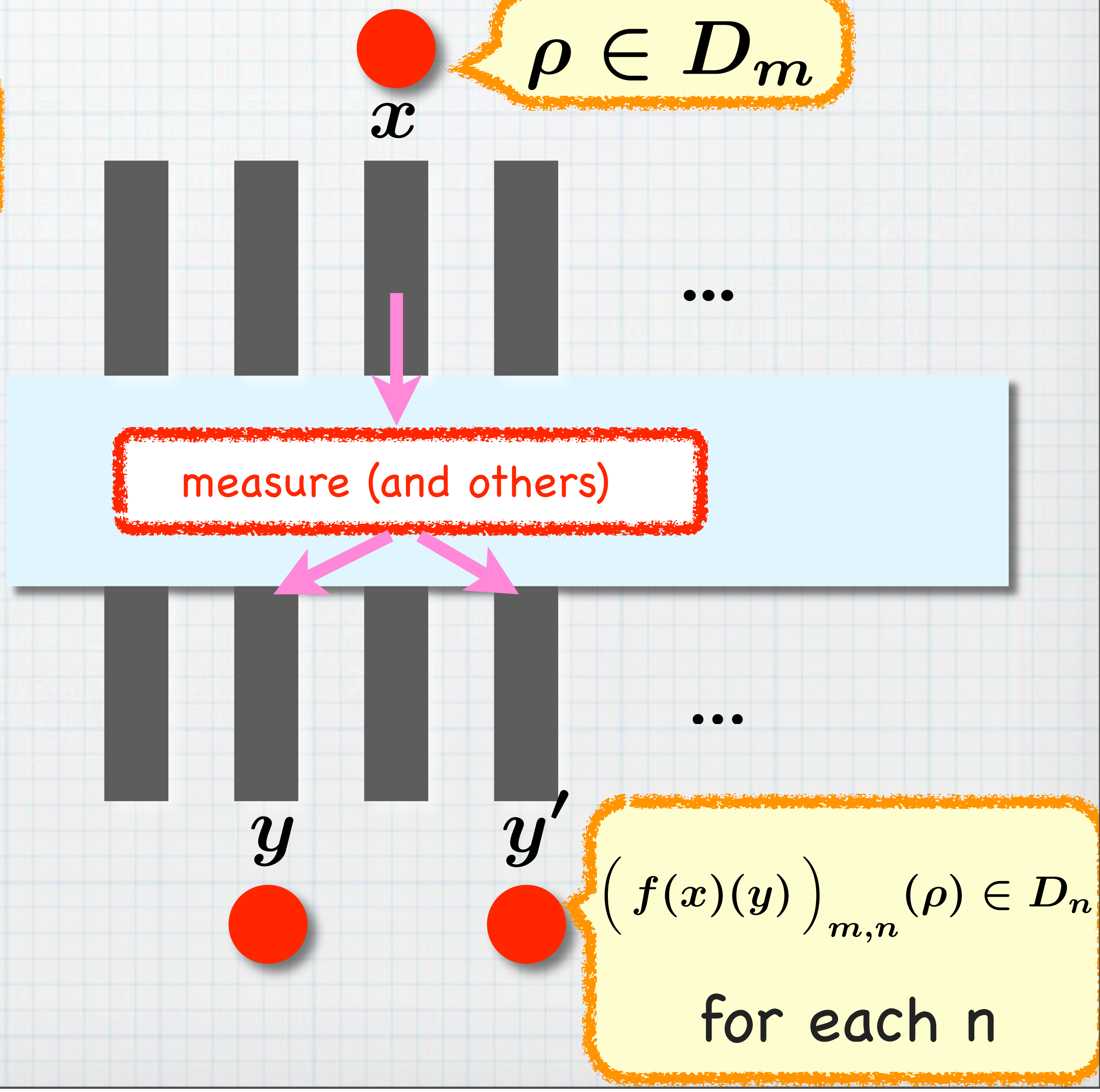
$$\sum_{y \in Y} \sum_{n \in \mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho) \big] \leq 1 \;, \quad \forall m \in \mathbb{N},\ \forall \rho \in D_m .$$

The Quantum Branching Monad

$$\frac{X \xrightarrow{f} Y \;\text{ in } \mathcal{K}\ell(\mathcal{Q})}{X \xrightarrow{f} \mathcal{Q}Y \;\text{ in } \mathbf{Sets}}$$

entrance exit in dim. out dim.

- Given $x \in X$, $y \in Y$, $m \in \mathbb{N}$, $n \in \mathbb{N}$ determines a quantum operation $\big(f(x)(y) \big)_{m,n}$
- **trace cond.:**



$\rho \in D_m$

$\big(f(x)(y) \big)_{m,n}(\rho) \in D_n$

for each n

The Quantum Branching Monad

$$\mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n\in\mathbb{N}} \mathbf{QO}_{m,n} \;\middle|\; \text{the trace condition} \right\}$$

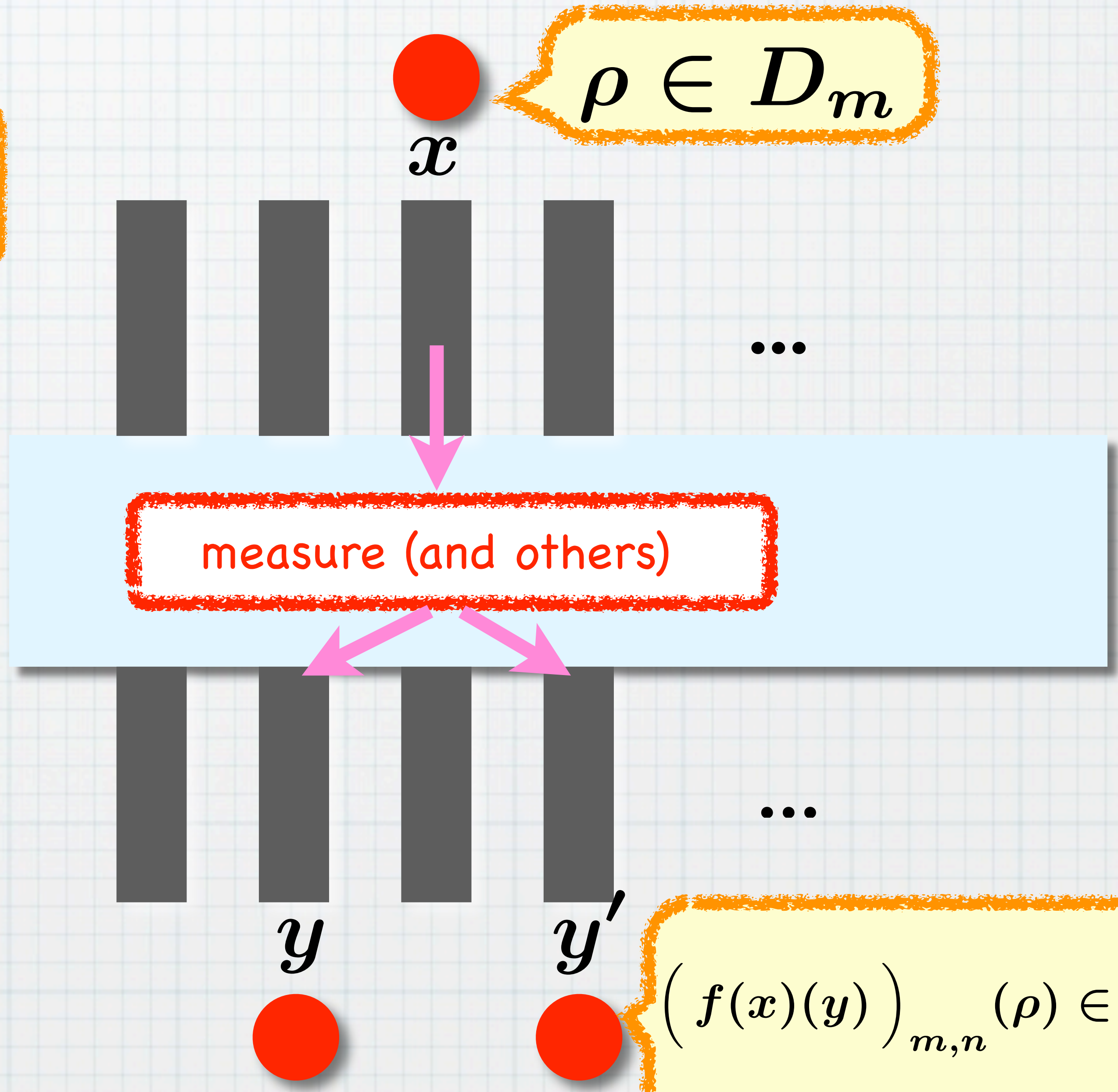
$$\sum_{y\in Y}\sum_{n\in\mathbb{N}} \mathbf{tr}\big[\big(c(y)\big)_{m,n}(\rho)\big] \le 1 \;,\quad \forall m\in\mathbb{N},\ \forall \rho\in D_m .$$

$$\frac{X \xrightarrow{f} Y \ \text{ in } \mathcal{K}\ell(\mathcal{Q})}{X \xrightarrow{f} \mathcal{Q}Y \ \text{ in } \mathbf{Sets}}$$

entrance exit in dim. out dim.

- Given $x\in X$, $y\in Y$, $m\in\mathbb{N}$, $n\in\mathbb{N}$ determines a quantum operation $\big(f(x)(y)\big)_{m,n}$
- **trace cond.:**

$$\sum_{y,n} \Pr\left(\begin{array}{c}\text{Token led}\\ \text{to } y\\ \text{with dim. } n\end{array}\right) \le 1$$



$\rho\in D_m$

$\big(f(x)(y)\big)_{m,n}(\rho) \in D_n$

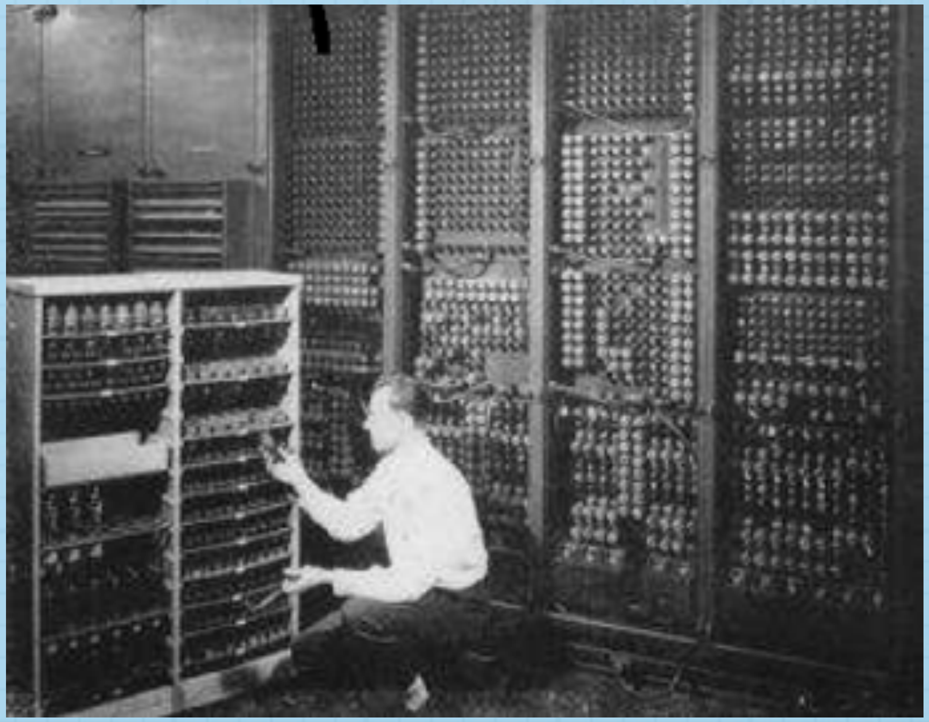
for each n

“Quantum Data, Classical Control”

Illustration by N. Hoshino

Quantum data

Classical control



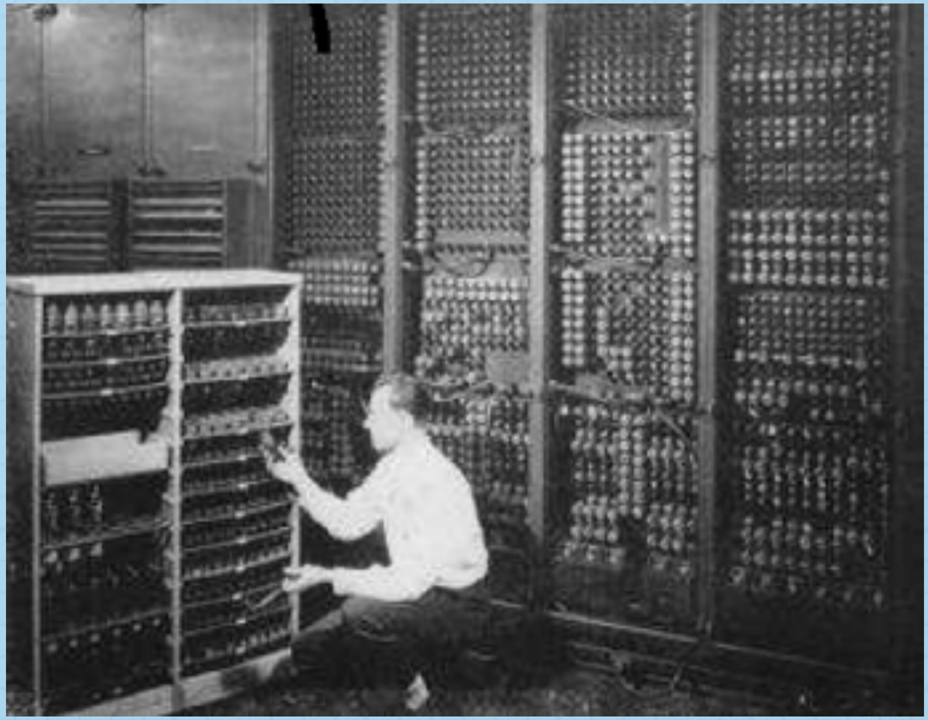
“Quantum Data, Classical Control”

Illustration by N. Hoshino

Quantum data

$$\frac{1}{\sqrt{2}}$$

Classical control



“Quantum Data, Classical Control”

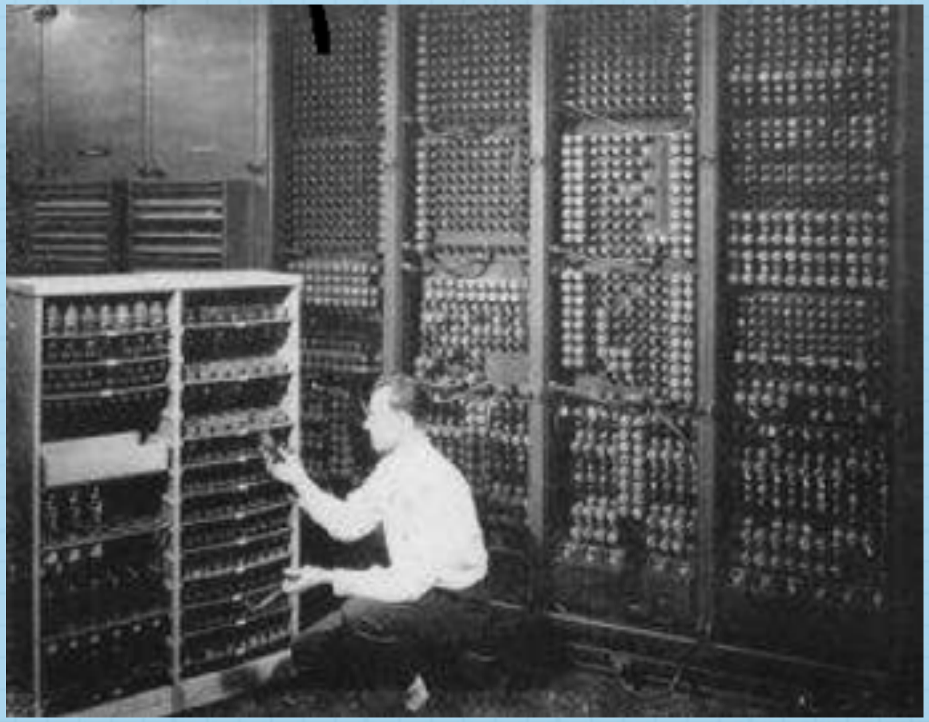
Illustration by N. Hoshino

Quantum data

$$\frac{1}{\sqrt{2}}$$

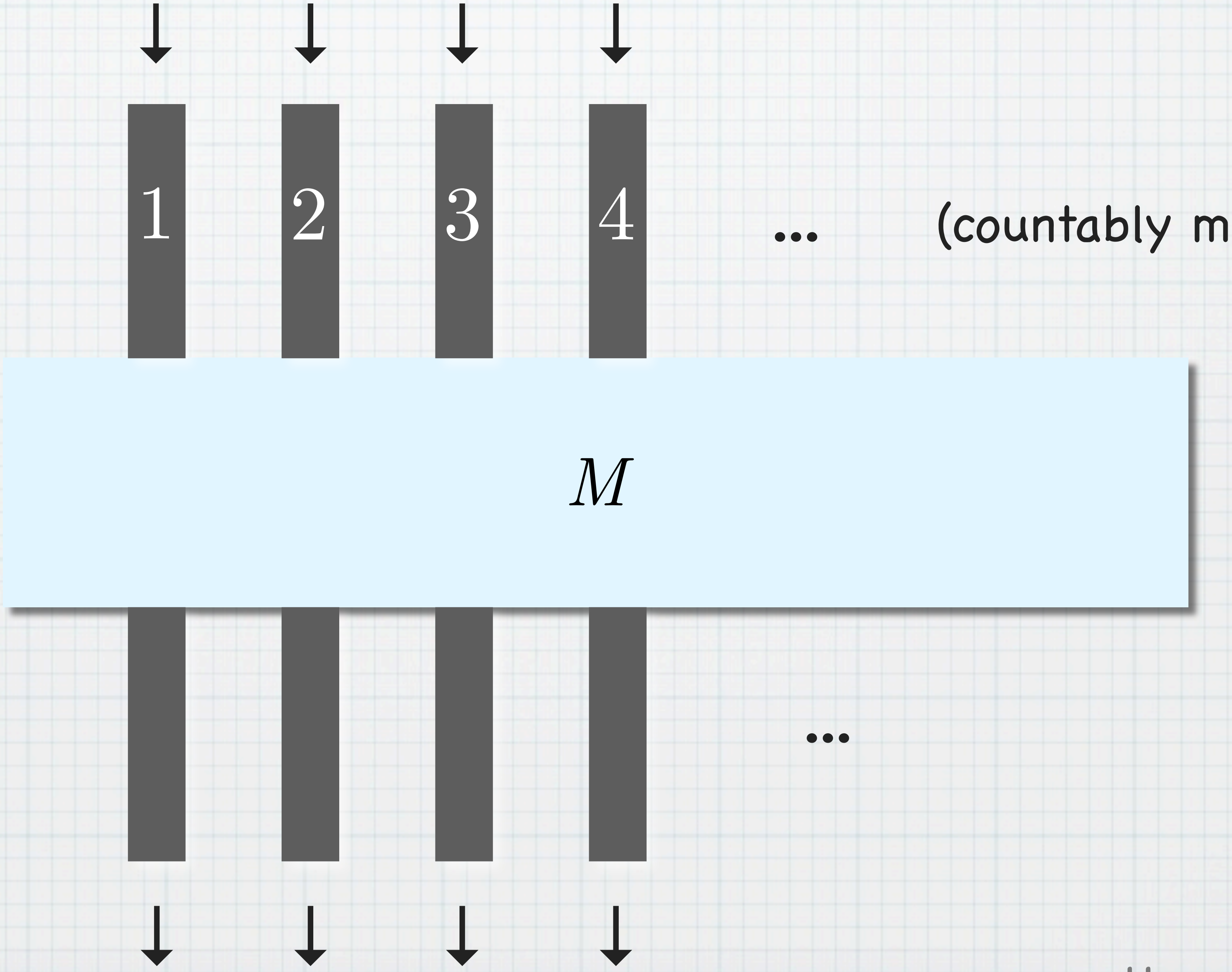
$$+\frac{1}{\sqrt{2}}$$

Classical control



Quantum
Geometry of Interaction

$[\![M]\!] =$



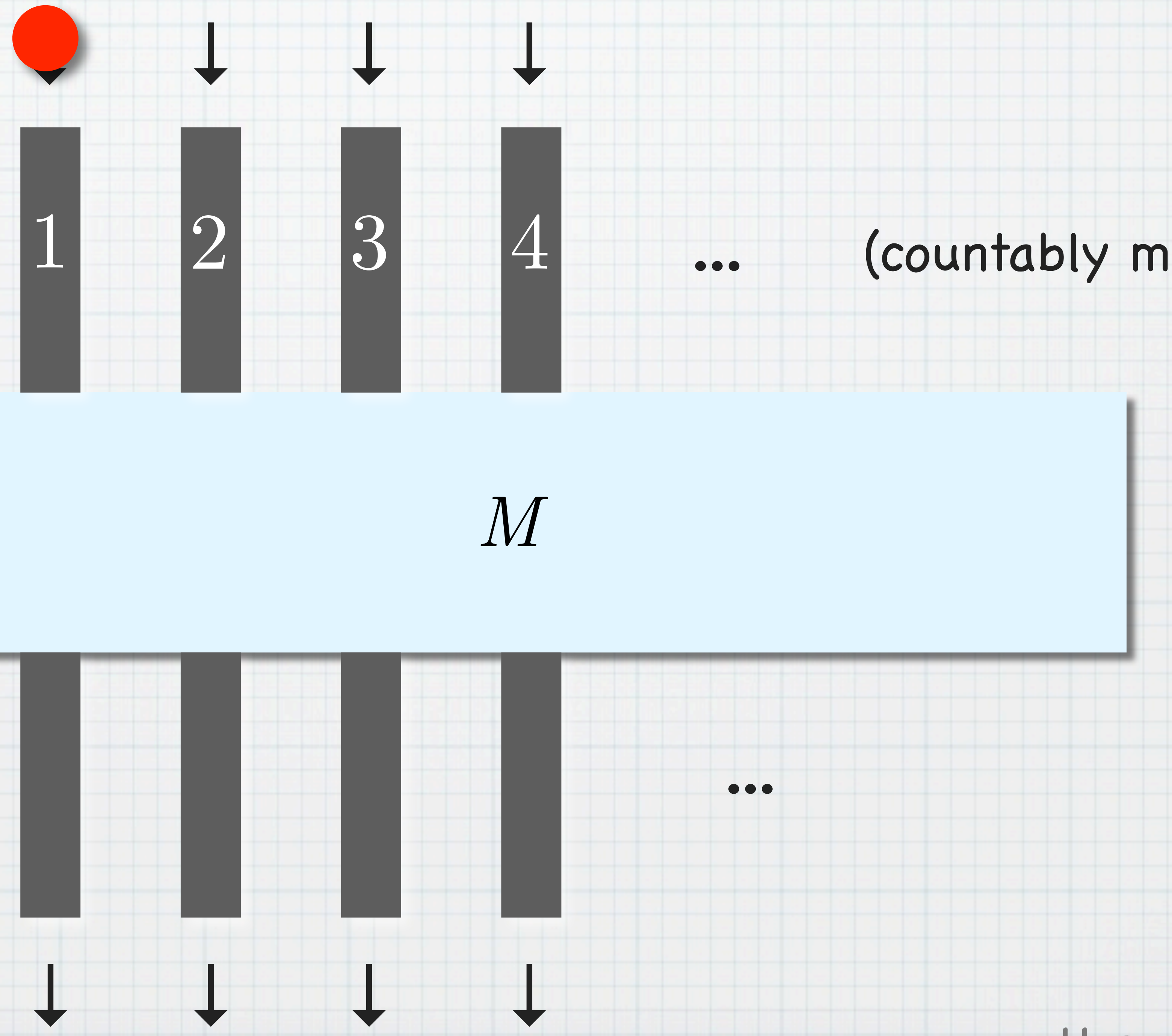
(countably many)

Hasuo (Tokyo)

Quantum
Geometry of Interaction

Not just a token/ particle, but quantum state!

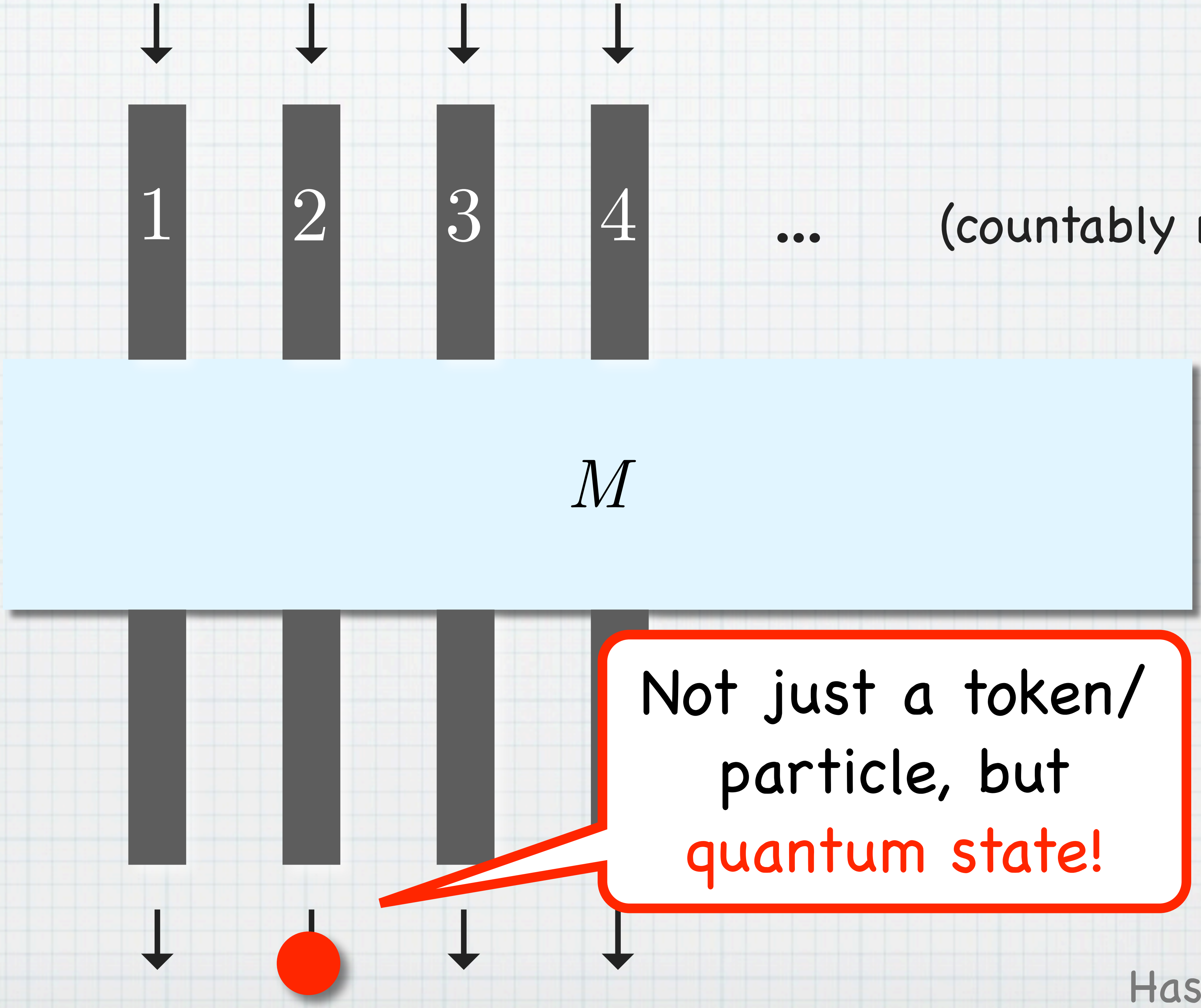
$$[\![M]\!] =$$



(countably many)

Quantum
Geometry of Interaction

$[\![M]\!] =$



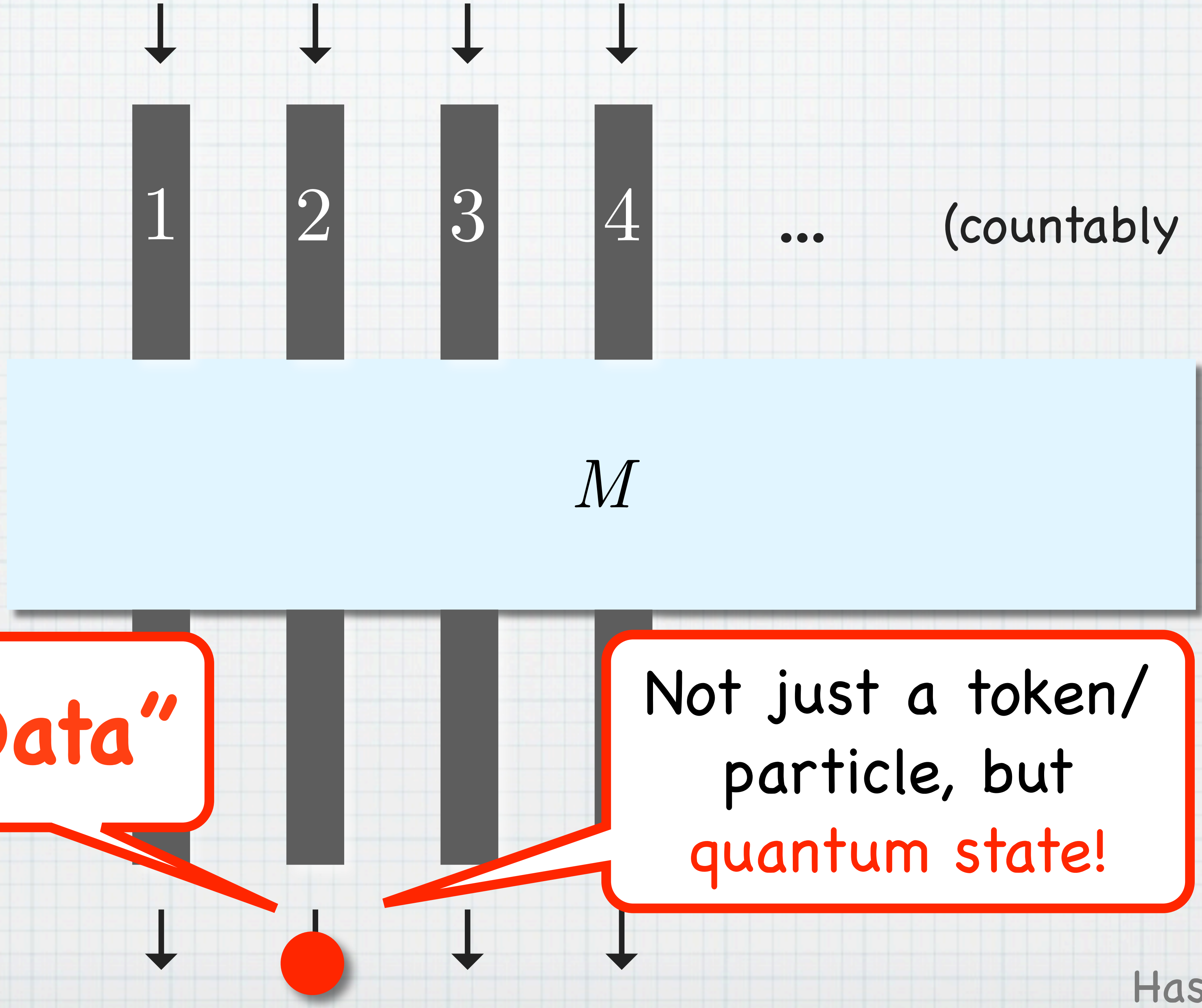
(countably many)

Not just a token/ particle, but quantum state!

Hasuo (Tokyo)

Quantum
Geometry of Interaction

$$[\![M]\!] =$$



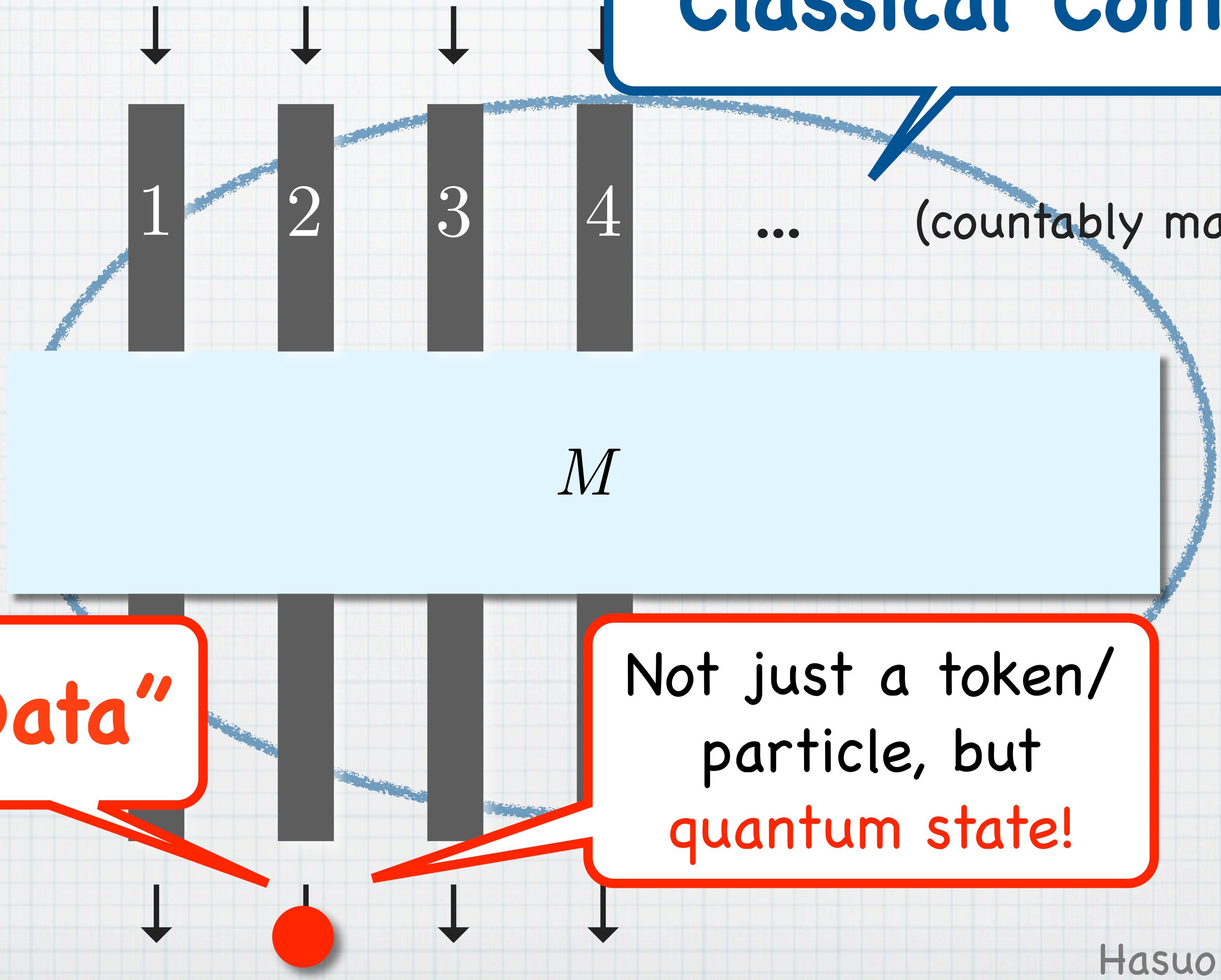
(countably many)

“Quantum Data”

Not just a token/ particle, but quantum state!

Quantum
Geometry of Interaction

$$[\![M]\!] =$$



“Classical Control”

“Quantum Data”

Not just a token/ particle, but quantum state!

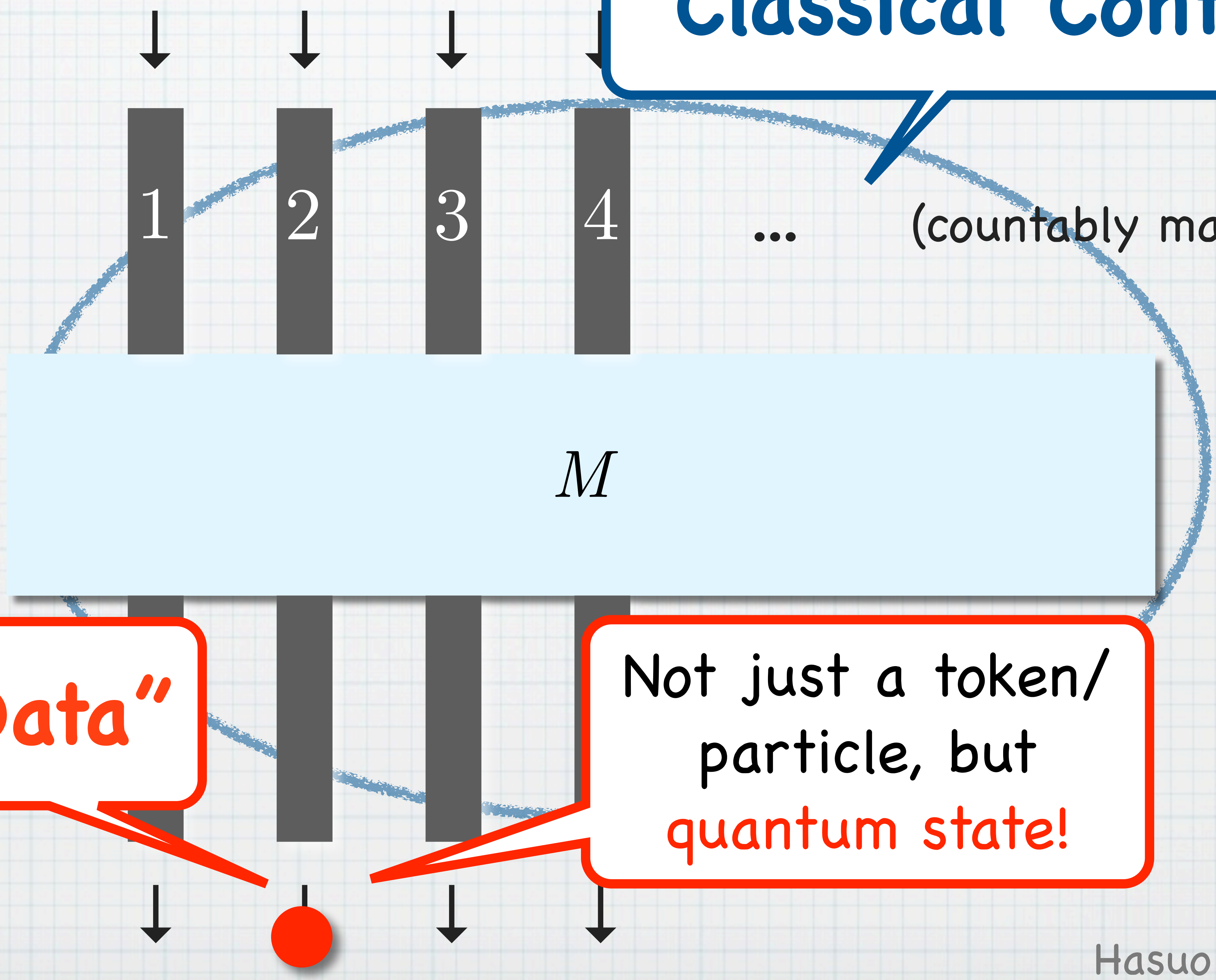
Qua

Geometry o

- "in which pipe"
- (measurement ➜ case-distinction) leads a token to different pipes

"Classical Control"

$$[\![M]\!] =$$



"Quantum Data"

Not just a token/ particle, but quantum state!

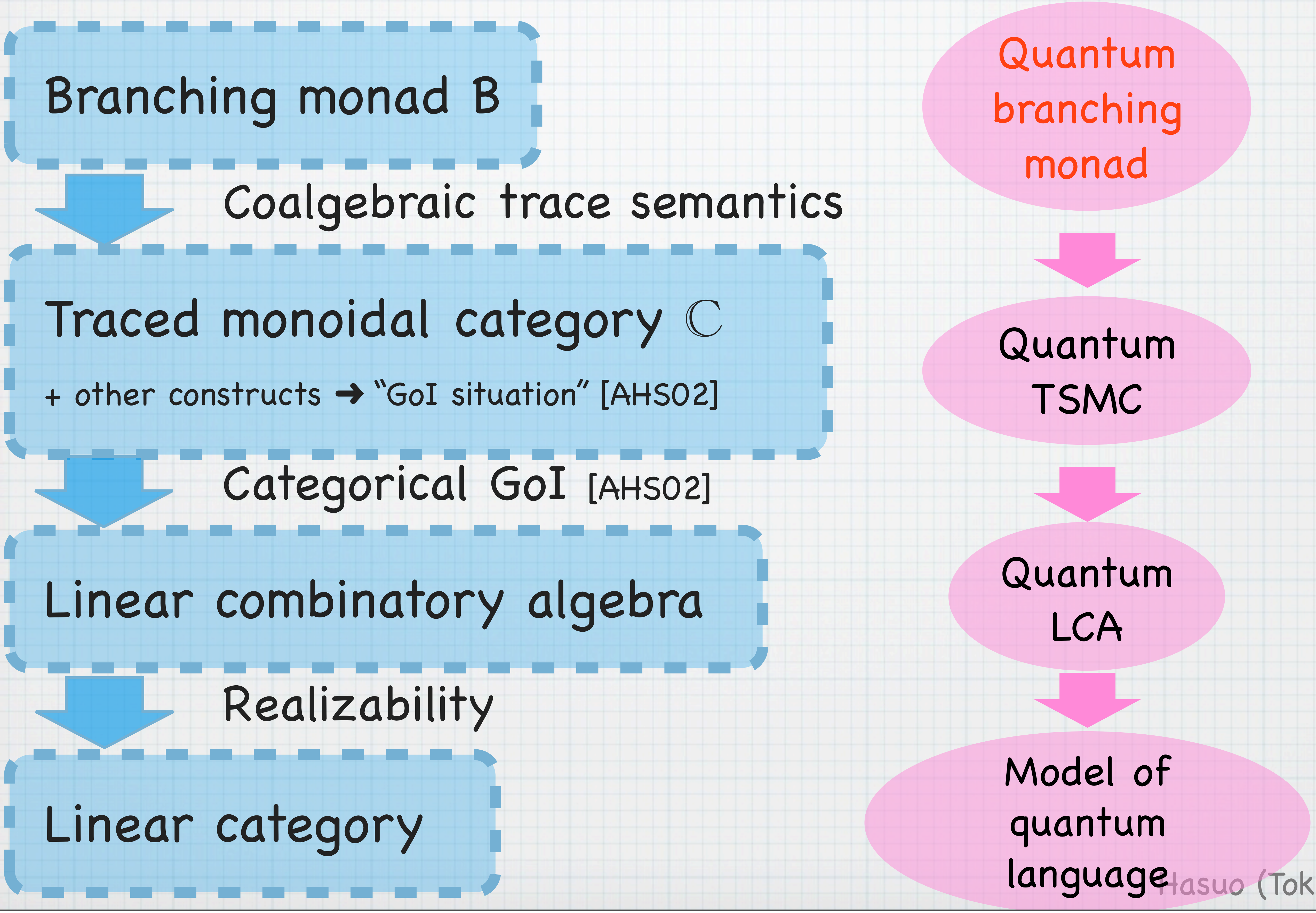
End of the Story?

- No! All the technicalities are yet to come:
  - CPS interpretation (for partial measurement)
  - Result type: a final coalgebra in $\mathbf{PER}_Q$
  - **Admissible PERs** for recursion
  - ...
- On the next occasion :-)

Results

- The monad $\mathcal{Q}$ qualifies as a "branching monad"
- The quantum GoI workflow leads to a linear category $\mathbf{PER}_{\mathcal{Q}}$
- From which we construct an adequate denotational model for a quantum λ-calculus (a variant of Selinger & Valiron's)

Conclusion: the Categorical GoI Workflow



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs → "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Quantum branching monad

Quantum TSMC

Quantum LCA

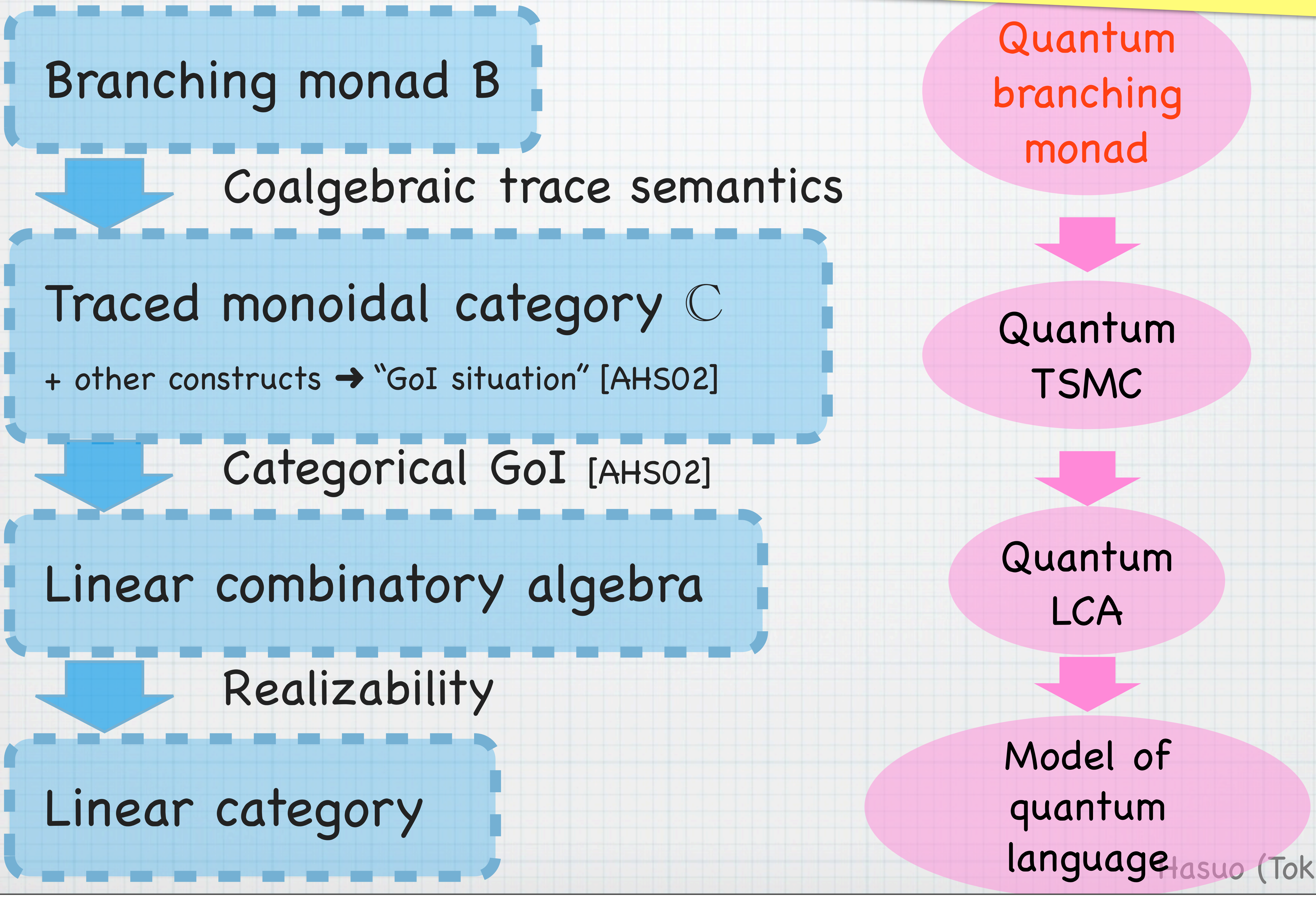
Model of quantum language

Conclusion: the Cat

Thank you for your attention!

Ichiro Hasuo (Dept. CS, U Tokyo)

http://www-mmm.is.s.u-tokyo.ac.jp/~ichiro/



Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$

+ other constructs ➜ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Quantum branching monad

Quantum TSMC

Quantum LCA

Model of quantum language