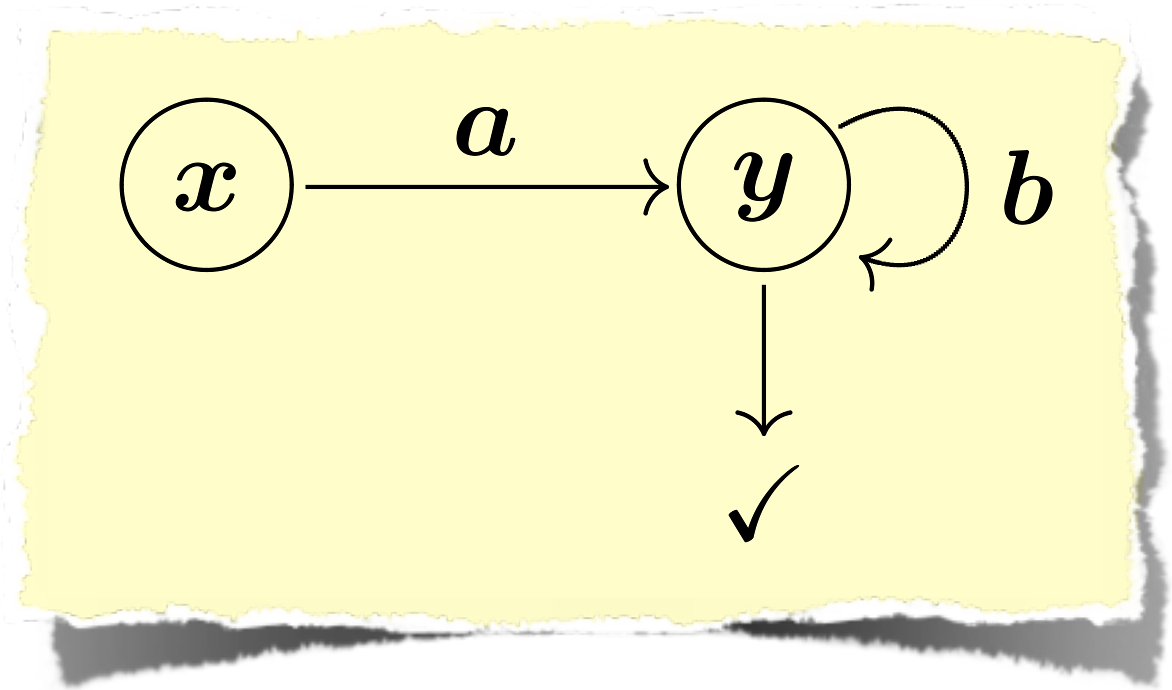
Theory of Coalgebra

Towards Mathematics of Systems

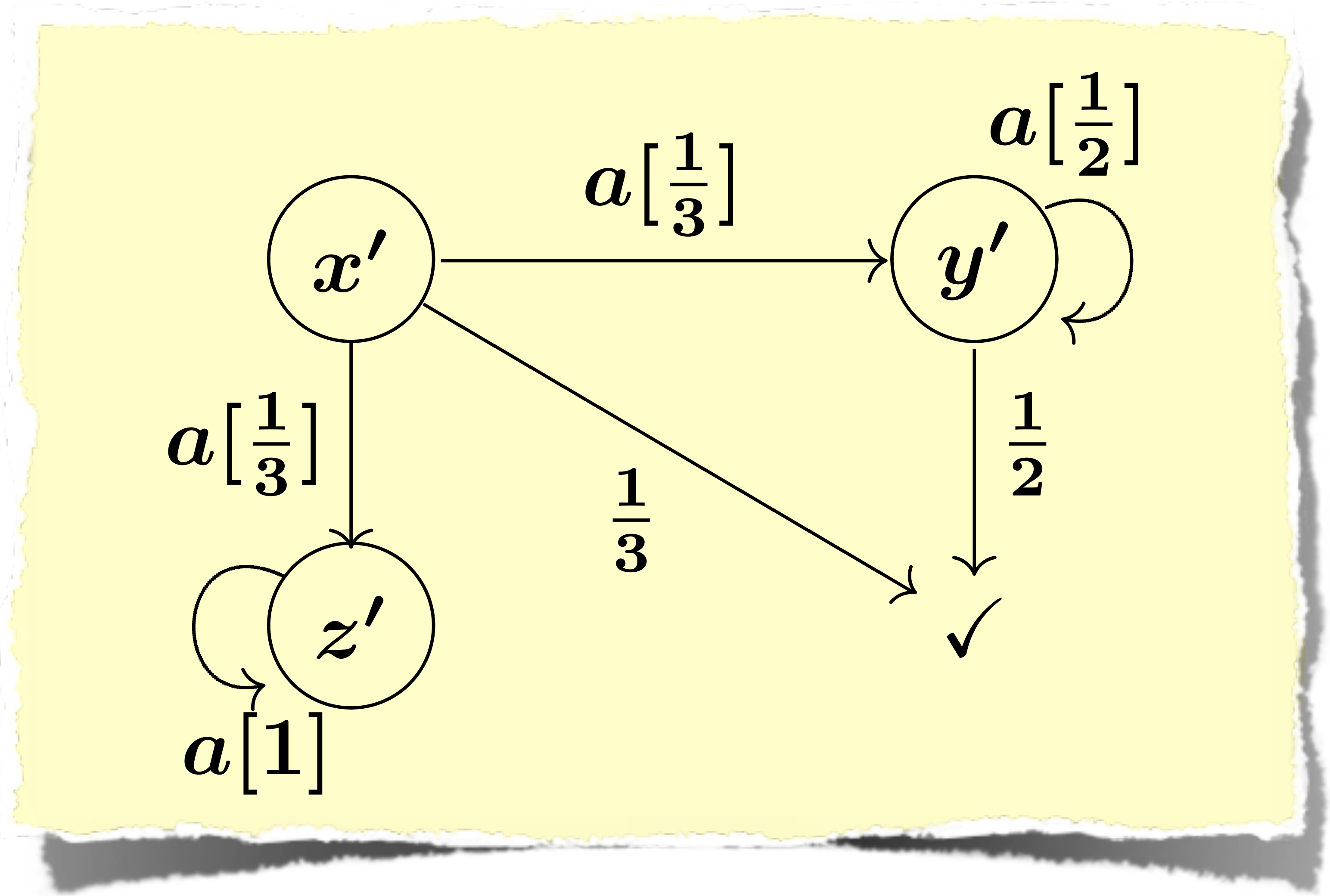
Ichiro Hasuo
RIMS, Kyoto Univ., JP
PRESTO *Sakigake* Promotion Program, JST, JP

Trace Semantics



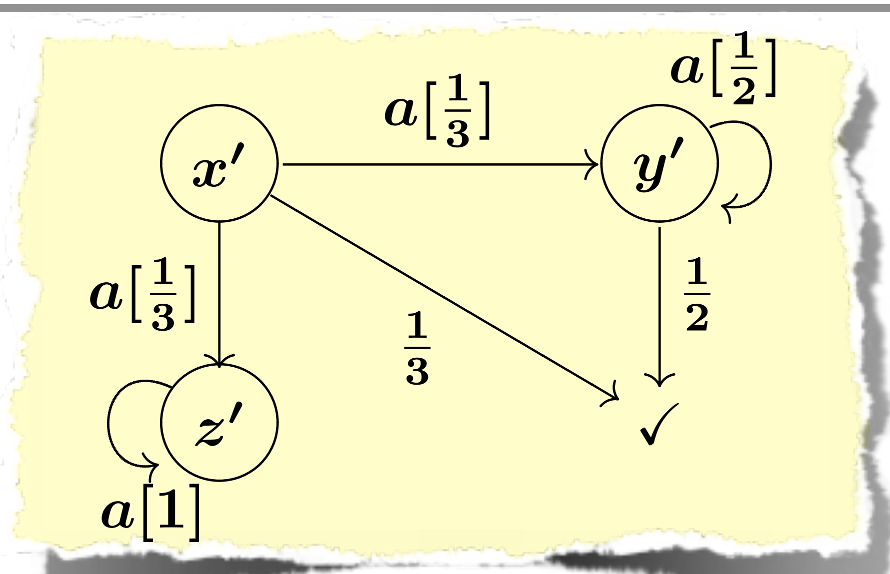
$$\mathbf{tr}(x) = \{a, ab, abb, \ldots\} = ab^*$$

Trace Semantics



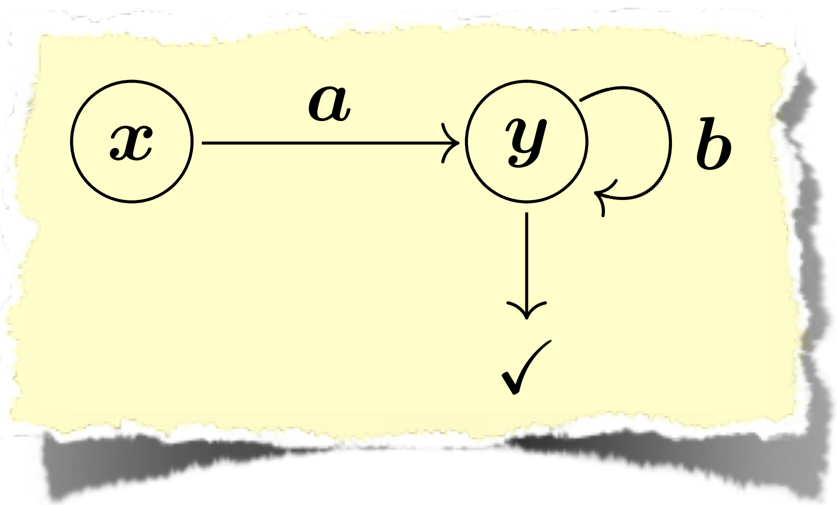
$$\mathsf{tr}(x') = \left[\begin{array}{l} \langle\rangle \mapsto \frac{1}{3}, \quad a \mapsto \frac{1}{3}\cdot\frac{1}{2}, \quad a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, \quad \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, \quad \cdots \end{array}\right]$$

Trace Semantics

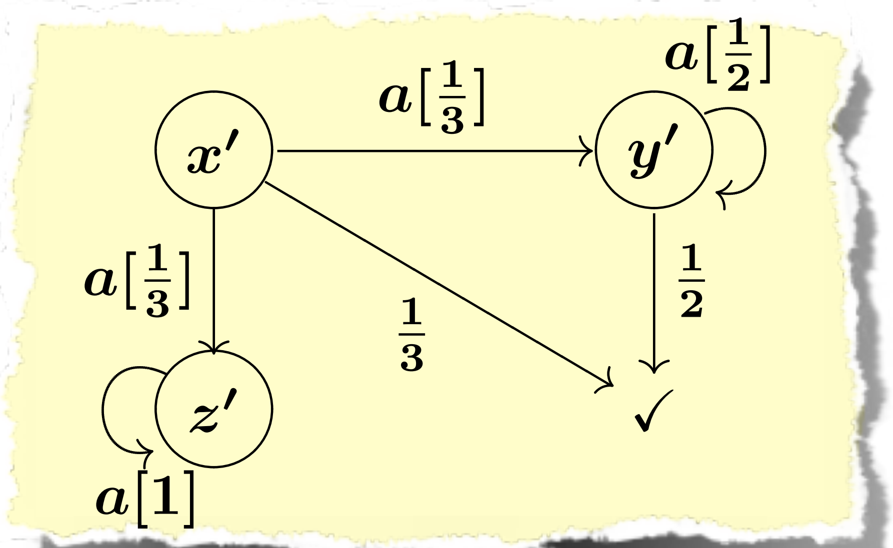


$$\mathsf{tr}(x') = \left[\begin{array}{l} \langle\rangle \mapsto \frac{1}{3}, \quad a \mapsto \frac{1}{3}\cdot\frac{1}{2}, \quad a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, \quad \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, \quad \cdots \end{array} \right]$$

Trace Semantics

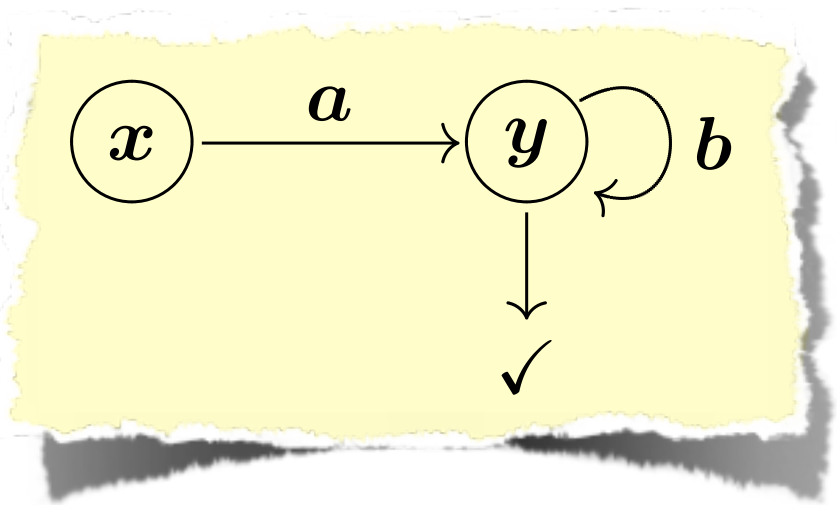


$$\mathsf{tr}(x) = \{a, ab, abb, \dots\} = ab^*$$



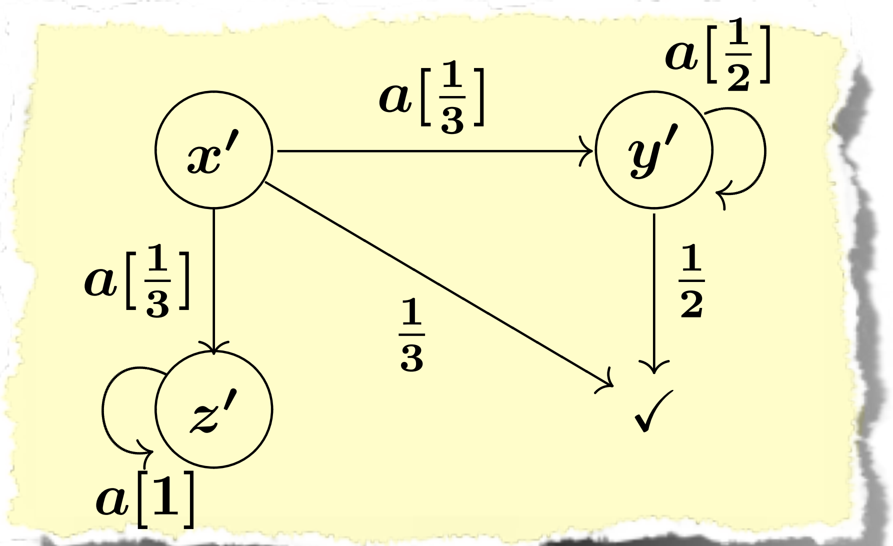
$$\mathsf{tr}(x') = \left[\begin{array}{l} \langle\rangle \mapsto \frac{1}{3}, \quad a \mapsto \frac{1}{3}\cdot\frac{1}{2}, \quad a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, \quad \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, \quad \cdots \end{array}\right]$$

Trace Semantics



non-deterministic branching
(set of options)

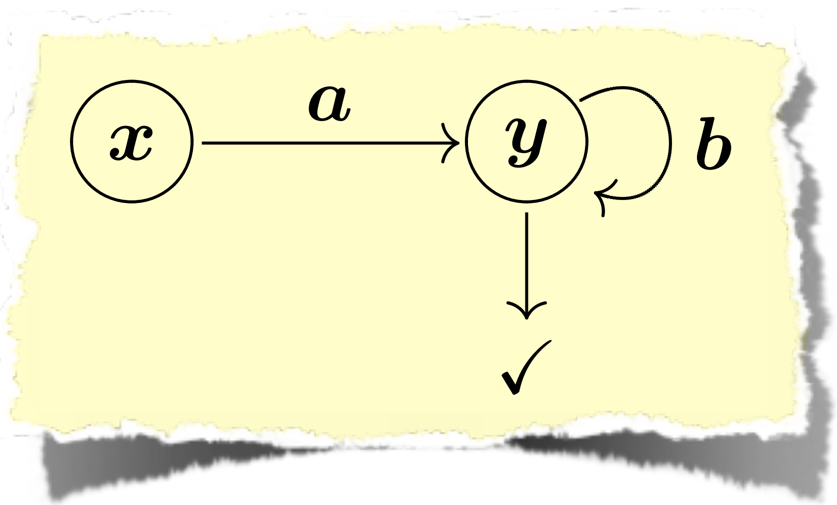
$$\mathbf{tr}(x) = \{a, ab, abb, \dots\} = ab^*$$



probabilistic branching
(prob. distribution over options)

$$\mathbf{tr}(x') = \left[\begin{array}{llll} \langle\rangle \mapsto \frac{1}{3}, & a \mapsto \frac{1}{3}\cdot\frac{1}{2}, & a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, & \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, & \cdots & & \end{array}\right]$$

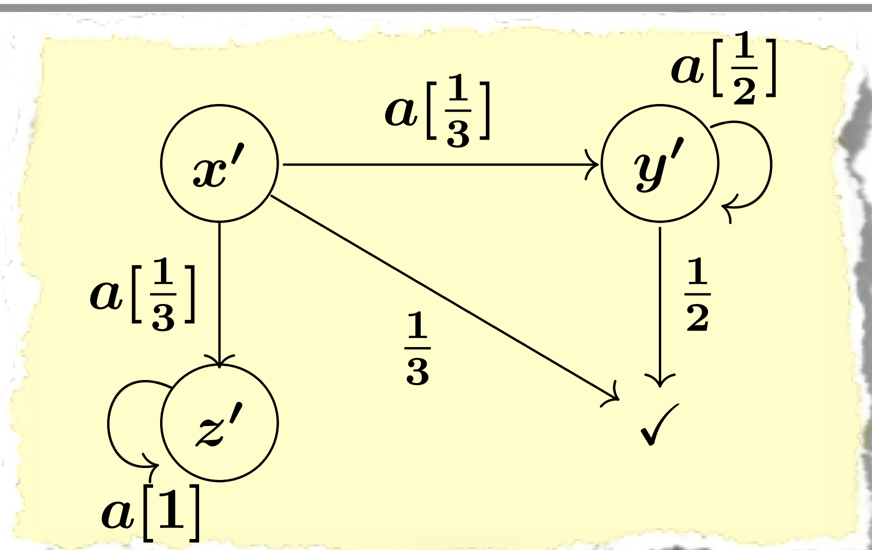
Trace Semantics



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

non-deterministic branching
(set of options)

set of execution traces

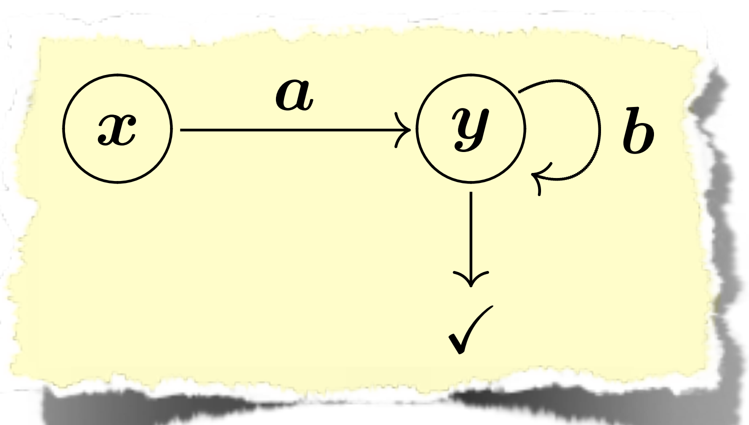


$$\mathbf{tr}(x') = \left[\begin{array}{l} \langle\rangle \mapsto \frac{1}{3}, \quad a \mapsto \frac{1}{3}\cdot\frac{1}{2}, \quad a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, \quad \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, \quad \cdots \end{array} \right]$$

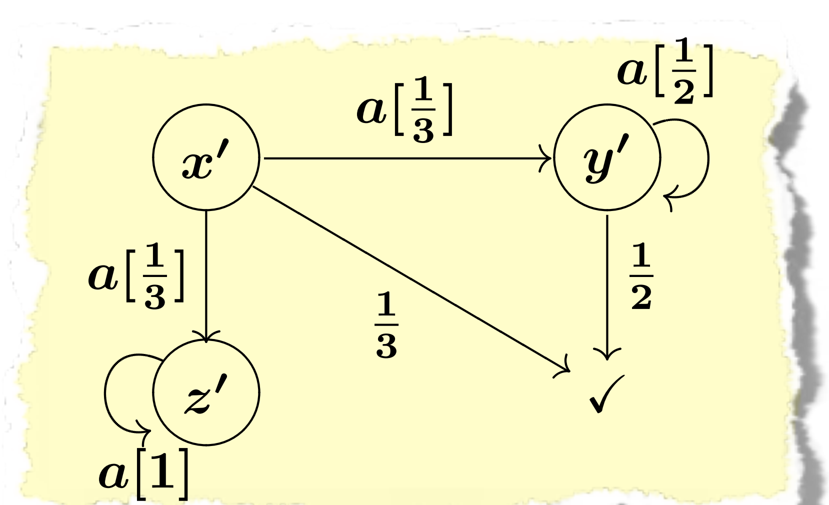
probabilistic branching
(prob. distribution over options)

prob. distribution over
execution traces

Coalgebra Offers a Uniform Understanding



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



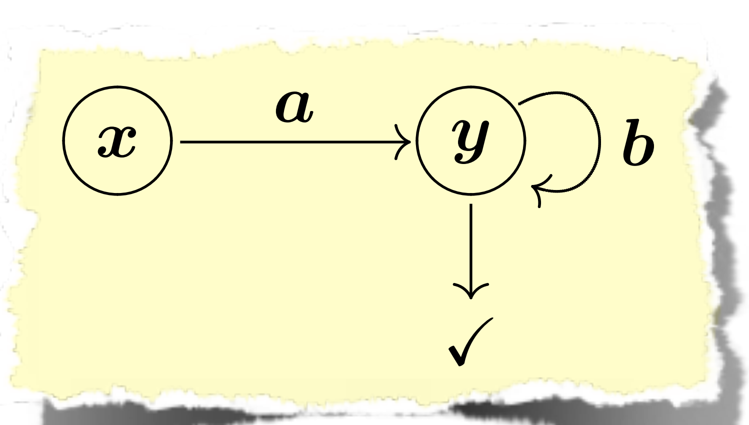
$$\mathbf{tr}(x') = \begin{bmatrix} \langle\rangle \mapsto \frac{1}{3}, & a \mapsto \frac{1}{3}\cdot\frac{1}{2}, & a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, & \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, & \cdots & & \end{bmatrix}$$

Coalgebra Offers a Uniform Understanding

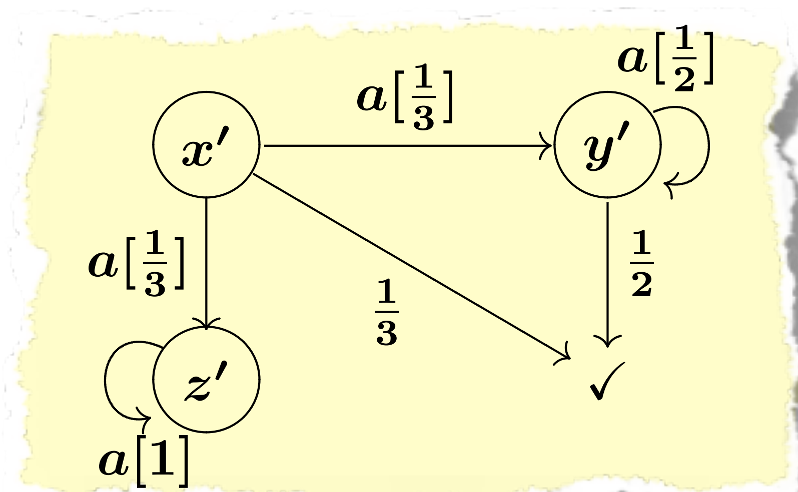
$$\begin{array}{ccc} FX & \overset{F(\mathsf{tr}(c))}{\dashrightarrow} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \underset{\mathsf{tr}(c)}{\dashrightarrow} & Z \end{array} \quad \text{in } \mathcal{K}\ell(T)$$

$T = \mathcal{P}$

$T = \mathcal{D}$



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



$\mathsf{tr}(x') = \left[\begin{array}{llll} \langle\rangle \mapsto \frac{1}{3}, & a \mapsto \frac{1}{3}\cdot\frac{1}{2}, & a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, & \dots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, & \dots & & \end{array}\right]$

Coalgebra Offers a Uniform Understanding

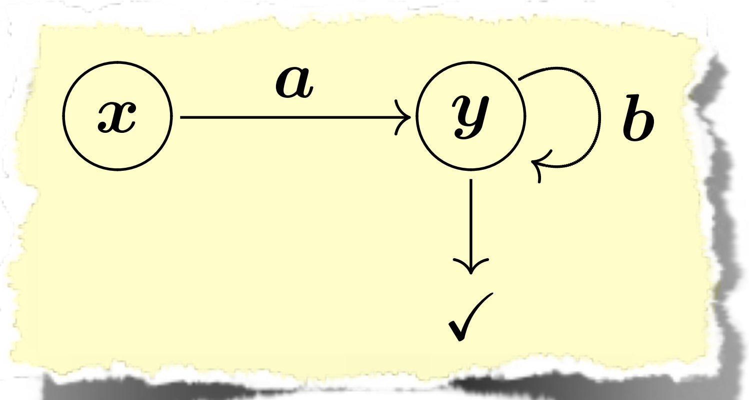
$$\begin{array}{ccc} FX & \overset{F(\mathsf{tr}(c))}{\dashrightarrow} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \underset{\mathsf{tr}(c)}{\dashrightarrow} & Z \end{array} \quad \text{in } \mathcal{K}\ell(T)$$

generic, coalgebraic

$T = \mathcal{P}$

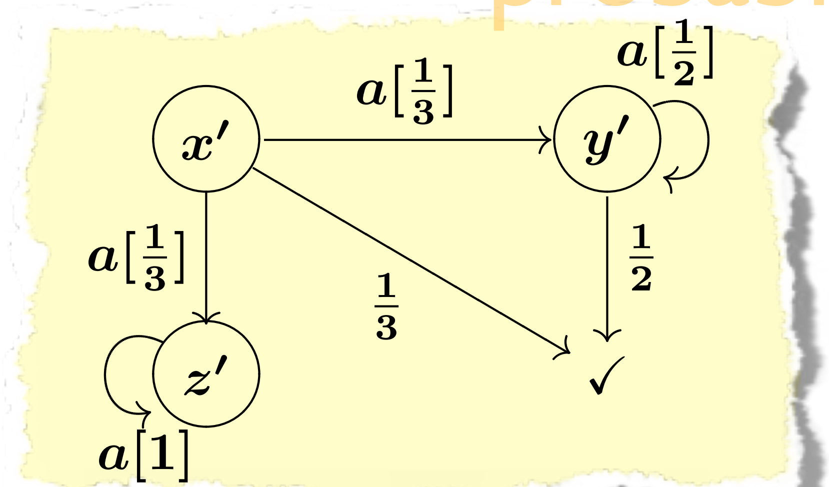
$T = \mathcal{D}$

non-deterministic



$\mathsf{tr}(x) = \{a, ab, abb, \ldots\} = ab^*$

probabilistic

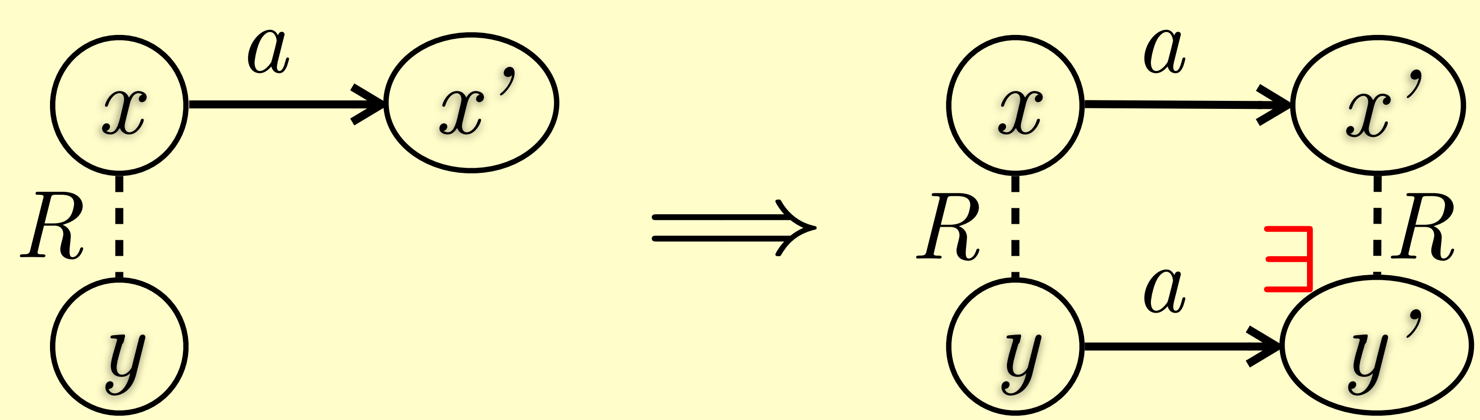


$$\mathsf{tr}(x') = \left[\begin{array}{llll} \langle\rangle \mapsto \frac{1}{3}, & a \mapsto \frac{1}{3}\cdot\frac{1}{2}, & a^2 \mapsto \frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{2}, & \cdots \\ a^n \mapsto \frac{1}{3}\cdot\left(\frac{1}{2}\right)^n, & \cdots & & \end{array} \right]$$

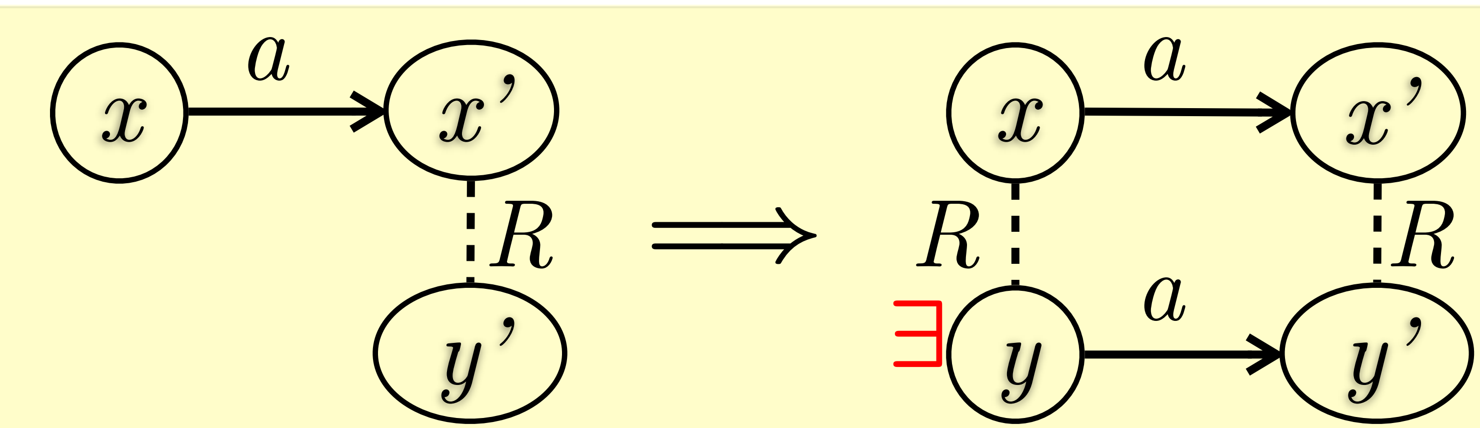
Forward/Backward Simulation

Forward simulation

A relation R between states of two systems, s.t.



Backward simulation



Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

"trace inclusion"

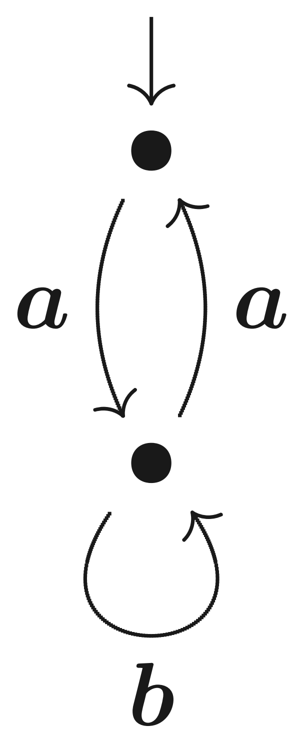
Forward/Backward Simulation

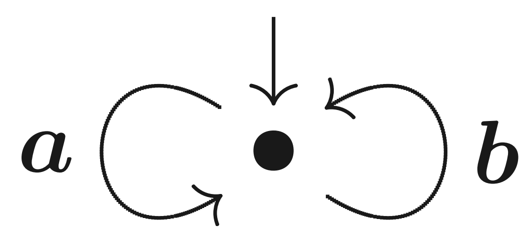
Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

"trace inclusion"





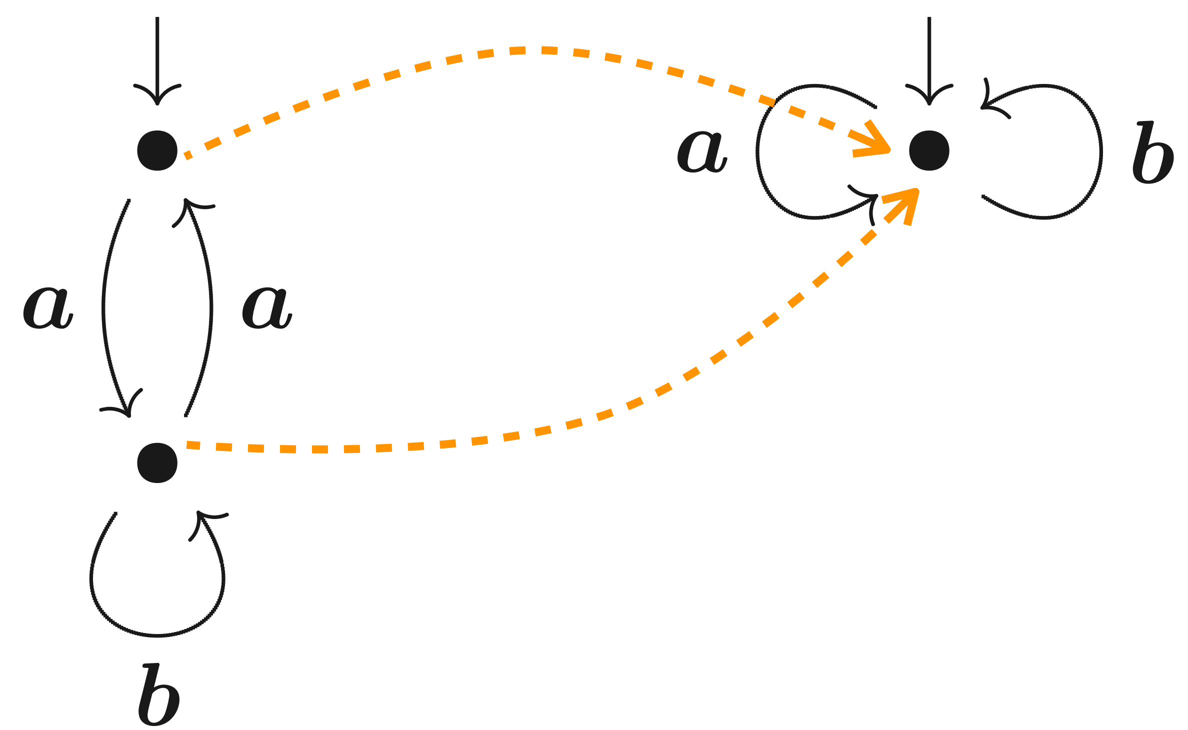
Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

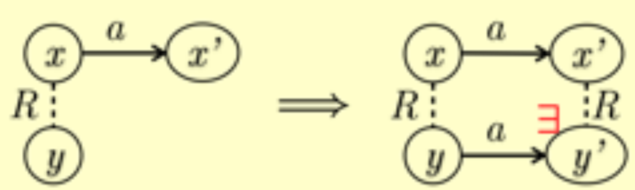
"trace inclusion"



Coalgebra Transfers Definitions & Results

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

Coalgebra Transfers Definitions & Results

In $\mathcal{K}\ell(T)$

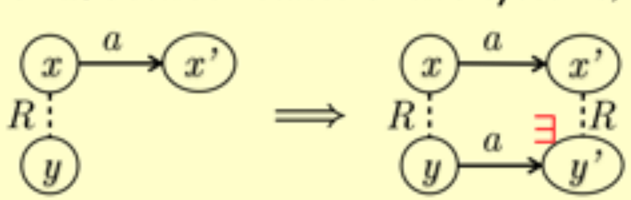
$$\begin{array}{ccc} FX & \xrightarrow{\;Ff\;} & FY \\ c\uparrow & \sqsupseteq & \uparrow d \\ X & \xrightarrow[\;f\;]{} & Y \end{array} \qquad \begin{array}{ccc} FX & \xrightarrow{\;Ff\;} & FY \\ c\uparrow & \sqsubseteq & \uparrow d \\ X & \xrightarrow[\;f\;]{} & Y \end{array}$$

forward simulation   backward simulation

$T = \mathcal{P}$

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

Coalgebra Transfers Definitions & Results

In $\mathcal{K}\ell(T)$

$$\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsupseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array} \qquad \begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsubseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$$

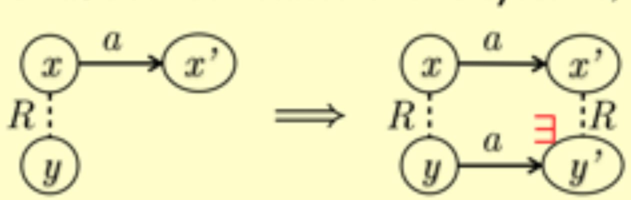
forward simulation — backward simulation

$T = \mathcal{P}$ — $T = \mathcal{D}$

$T = \mathcal{P}$

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

$T = \mathcal{D}$

Forward simulation

Definition. Let $\mathcal{X} = (X, x_0, c)$ and $\mathcal{Y} = (Y, y_0, d)$ be GPAs. A *forward (Kleisli) simulation* from $\mathcal{X}$ to $\mathcal{Y}$ is a function $f : Y \to \mathcal{D}X$ which satisfies the following (in)equalities.

$$\mathbf{Pr}[y_0 \dashrightarrow x_0] = 1 \qquad \text{(INIT)}$$

$$\textstyle\sum_{x\in X} \mathbf{Pr}[y \dashrightarrow x \to \checkmark] \le \mathbf{Pr}[y \to \checkmark] \quad \text{for each } y \in Y \qquad \text{(TERM)}$$

$$\textstyle\sum_{x\in X} \mathbf{Pr}[y \dashrightarrow x \xrightarrow{a} x'] \le \sum_{y'\in Y} \mathbf{Pr}[y \xrightarrow{a} y' \dashrightarrow x'] \quad \text{for each } y \in Y,\ a \in \mathsf{Act} \text{ and } x' \in X \qquad \text{(ACT)}$$

Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

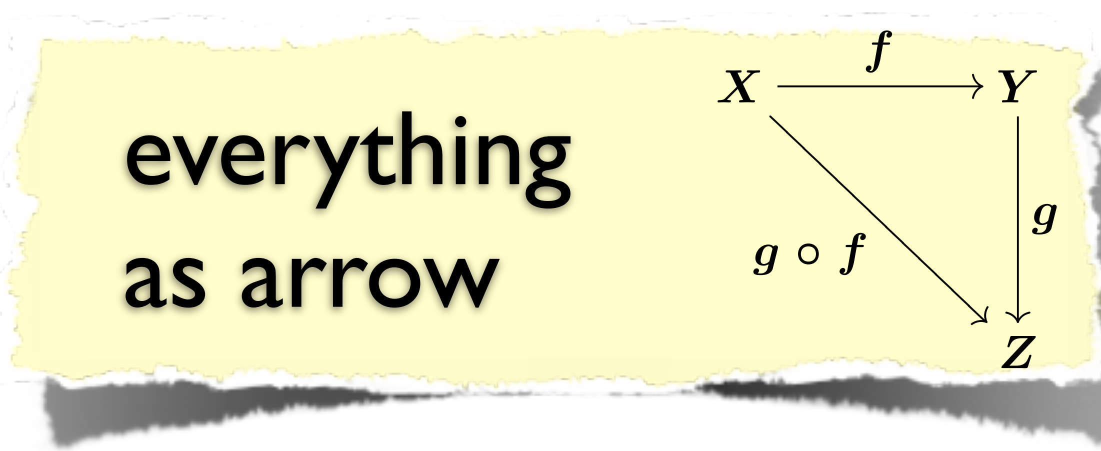
Coalgebra: Mathematical Theory of Systems

- Models state-based dynamic systems
  - Deterministic/non-deterministic automata, LTS, Mealy/Moore machines, probabilistic/weighted systems, ...

Coalgebra: Mathematical Theory of Systems

- Models state-based dynamic systems
    - Deterministic/non-deterministic automata, LTS, Mealy/Moore machines, probabilistic/weighted systems, ...

- With the language of category theory
    - Focus on the *essence*
    - Genericity, abstraction



Plan

- Introduction to coalgebra
- Recent research topics
- Coalgebraic trace semantics
- Wrapping up

I Theory of Coalgebra: Basics

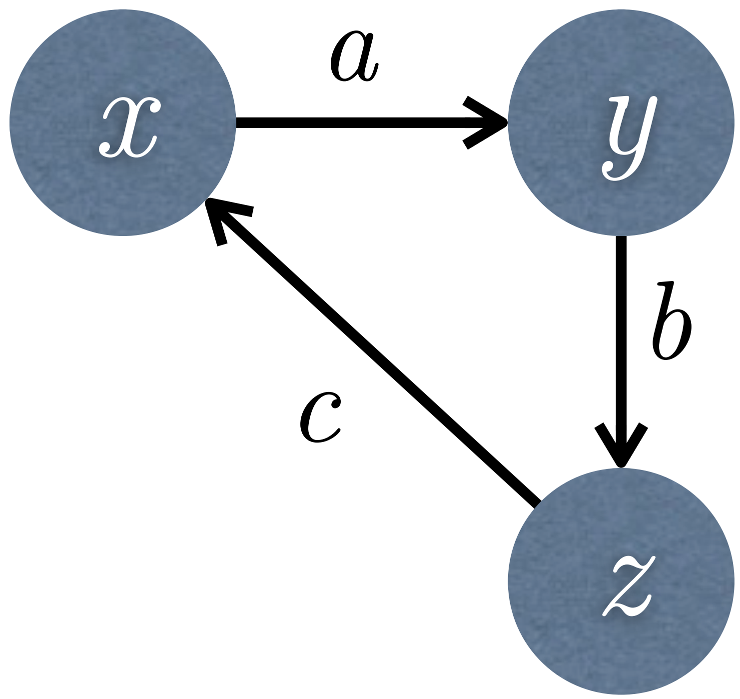
Coalgebra

Definition.
Let $\mathbb{C}$ be a category,
$\boldsymbol{F} : \mathbb{C} \to \mathbb{C}$ be a functor.

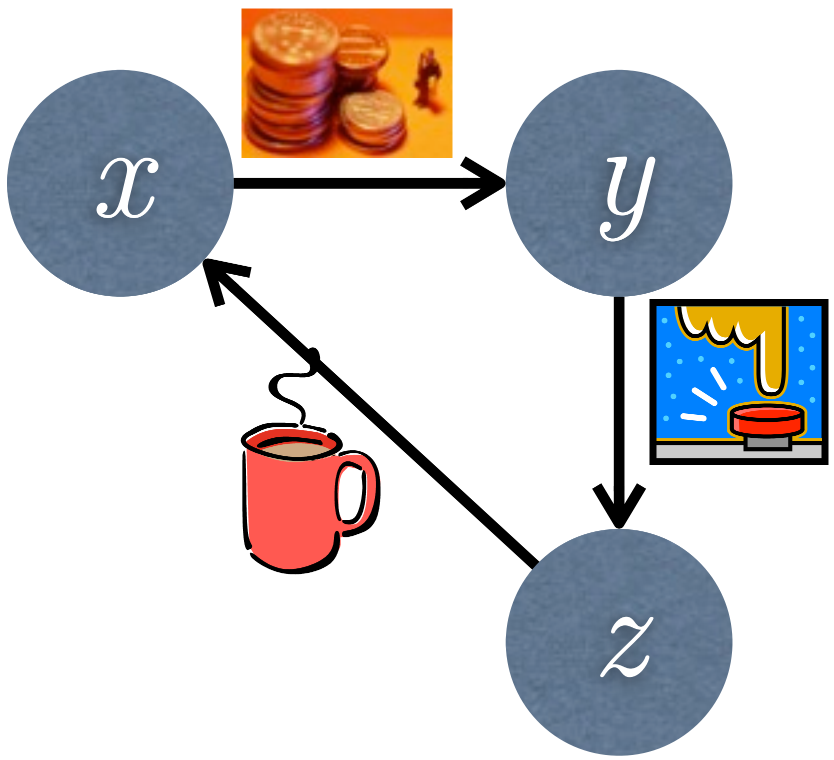
A *coalgebra* is $\begin{array}{c} \boldsymbol{FX} \\ \boldsymbol{c}\big\uparrow \\ \boldsymbol{X} \end{array}$ in $\mathbb{C}$.

- Mathematical simplicity

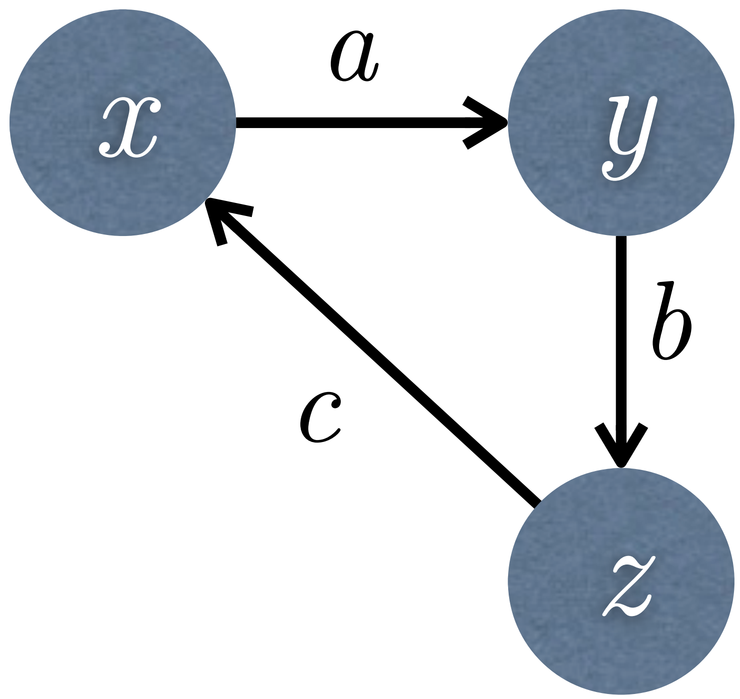
  ➛ feedback to mathematics



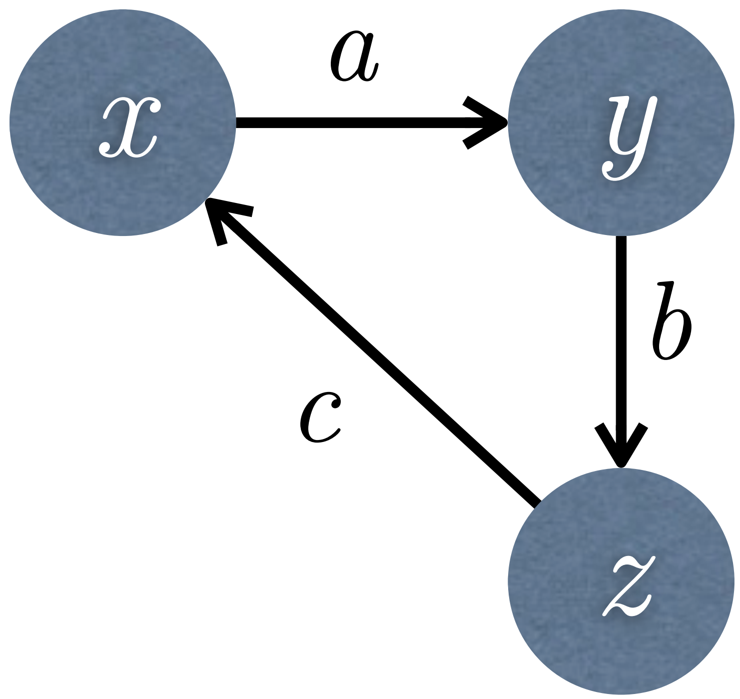
$$\text{as} \quad \begin{array}{c} \boldsymbol{F}\boldsymbol{X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$



$$\text{as} \quad \begin{array}{c} \boldsymbol{F}\boldsymbol{X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$



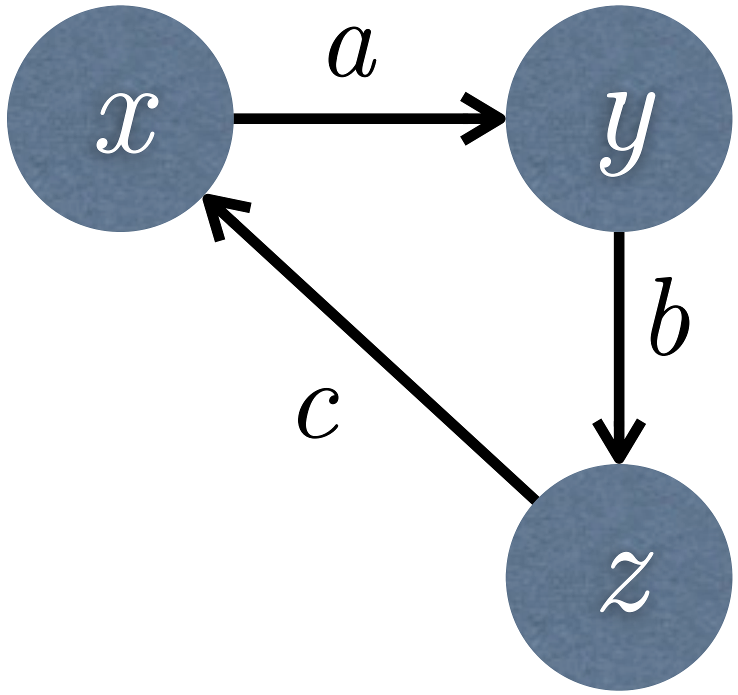
$$\text{as}\quad \begin{array}{c}\boldsymbol{F X}\\ \uparrow\\ \boldsymbol{X}\end{array}\quad \text{in } \mathbb{C}.$$



$$\text{as}\quad \begin{array}{c} \boldsymbol{F X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$

$$\begin{array}{cccc} \boldsymbol{\Sigma \times X} & \boldsymbol{(a, y)} & \boldsymbol{(b, z)} & \boldsymbol{(c, x)} \\ \uparrow & \uparrow & \uparrow & \uparrow \\ \boldsymbol{X} & \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{array}$$

$$\boldsymbol{X = \{x, y, z\}}$$
$$\boldsymbol{\Sigma = \{a, b, c\}}$$

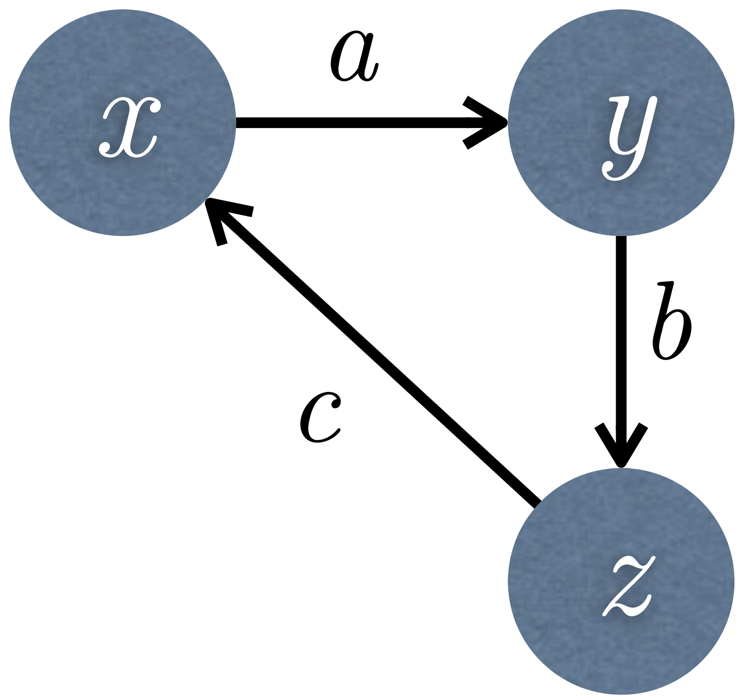


$$\mathbb{C} = \mathbf{Sets}$$
$$F = \boldsymbol{\Sigma} \times \_$$

$$\text{as} \quad \begin{array}{c} \boldsymbol{F\,X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$

$$\begin{array}{cccc} \boldsymbol{\Sigma \times X} & \boldsymbol{(a, y)} & \boldsymbol{(b, z)} & \boldsymbol{(c, x)} \\ \uparrow & \uparrow & \uparrow & \uparrow \\ \boldsymbol{X} & \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{array}$$

$$\boldsymbol{X = \{x, y, z\}}$$
$$\boldsymbol{\Sigma = \{a, b, c\}}$$



action and continue

$$\mathbb{C} = \mathbf{Sets}$$

$$F = \Sigma \times \_$$

as $\begin{array}{c} \boldsymbol{FX} \\ \uparrow \\ \boldsymbol{X} \end{array}$ in $\mathbb{C}$.

$$\begin{array}{cccc} \boldsymbol{\Sigma \times X} & (\boldsymbol{a}, \boldsymbol{y}) & (\boldsymbol{b}, \boldsymbol{z}) & (\boldsymbol{c}, \boldsymbol{x}) \\ \uparrow & \uparrow & \uparrow & \uparrow \\ \boldsymbol{X} & \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{array}$$

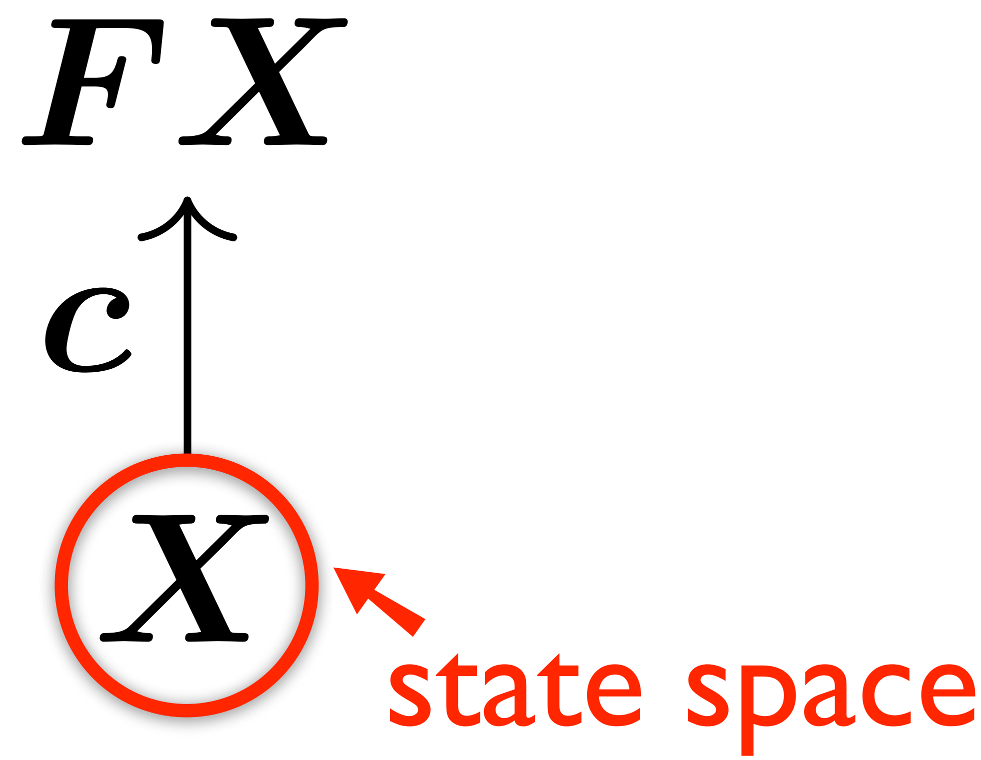
$$\boldsymbol{X} = \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}$$

$$\boldsymbol{\Sigma} = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$$

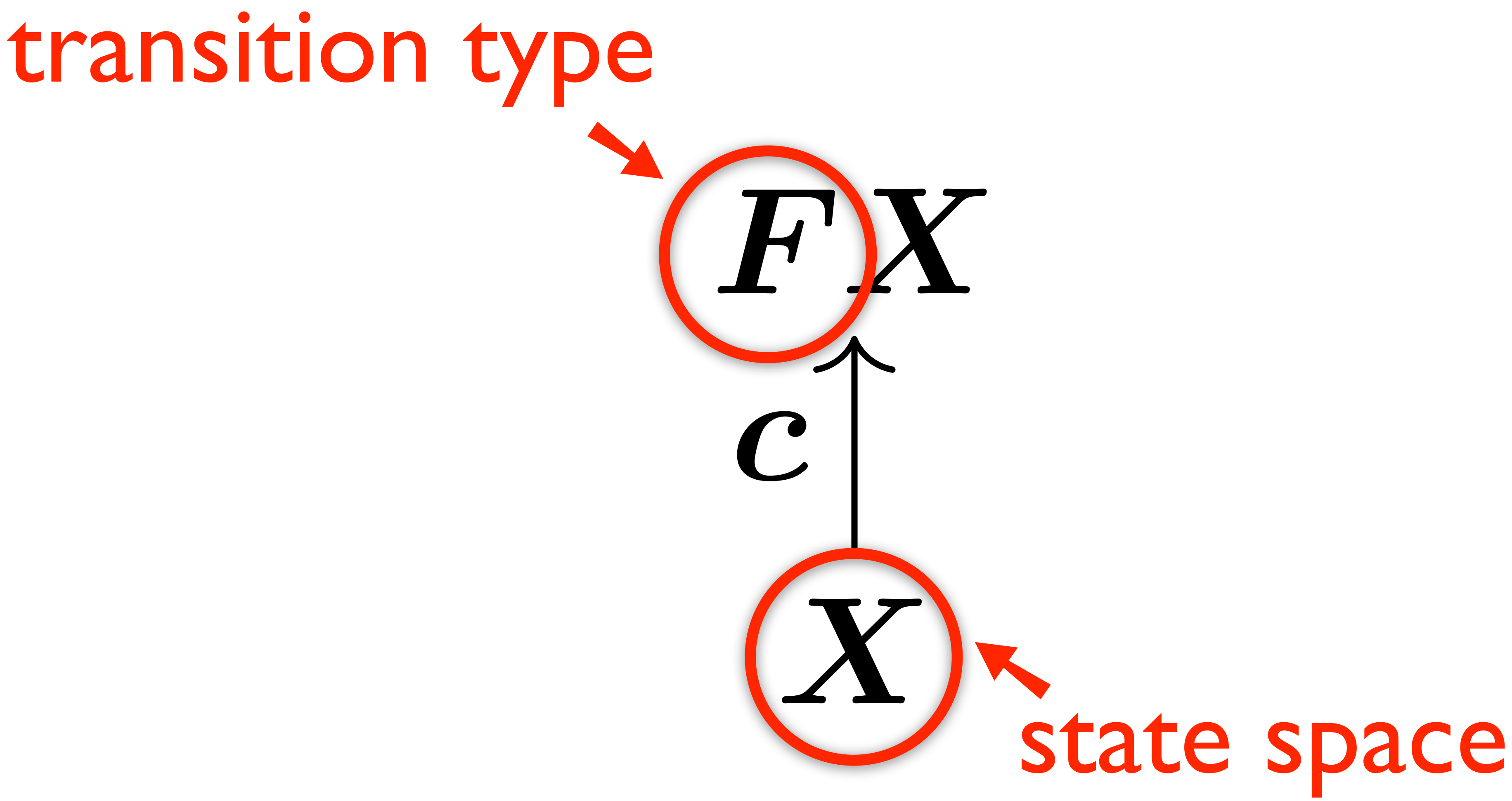
Coalgebra

$$\begin{array}{c} \boldsymbol{F\,X} \\ \boldsymbol{c}\big\uparrow \\ \boldsymbol{X} \end{array}$$

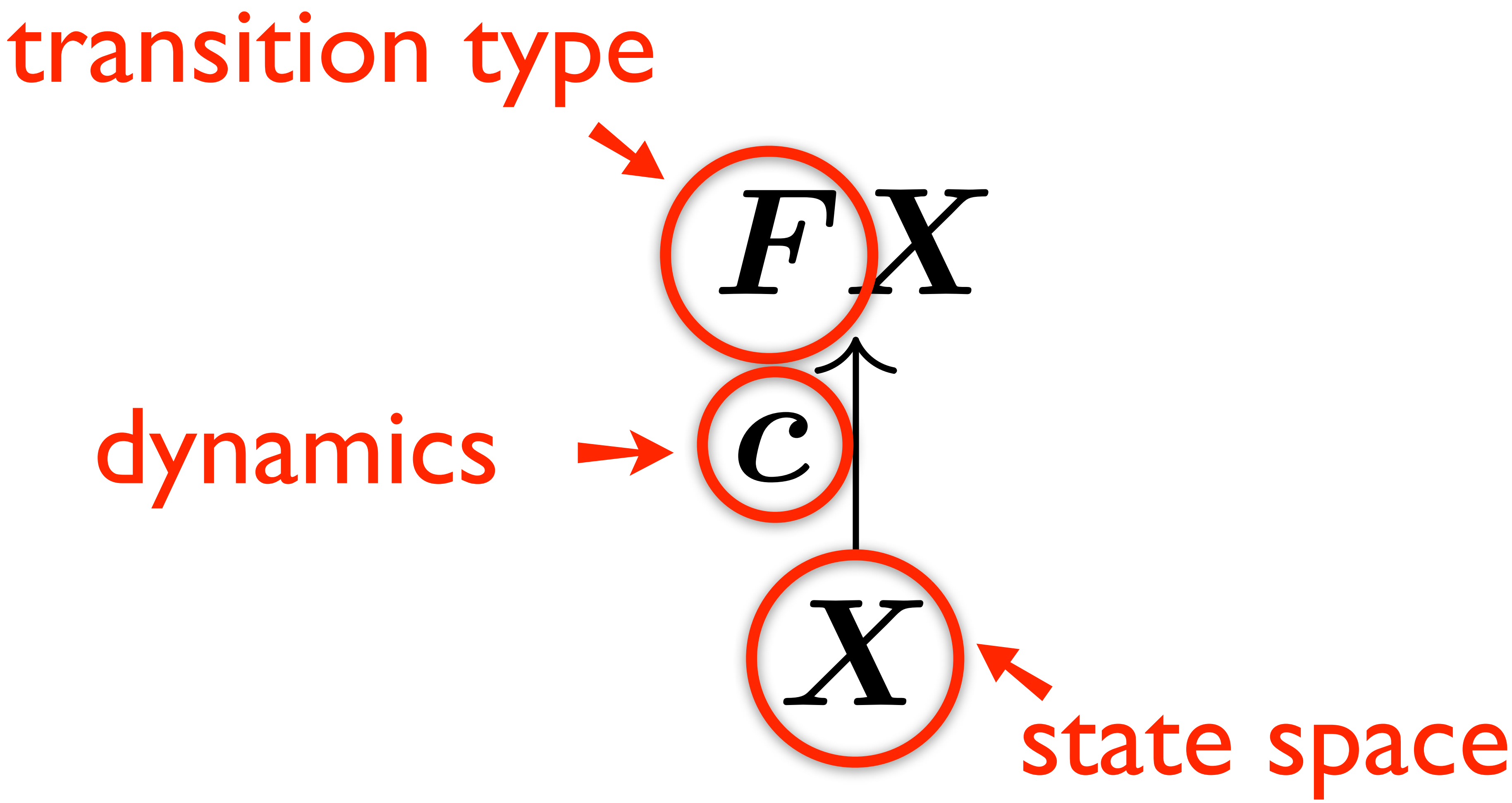
Coalgebra

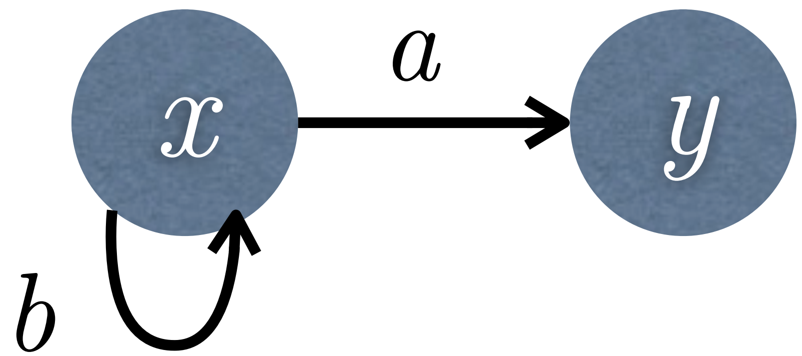


Coalgebra

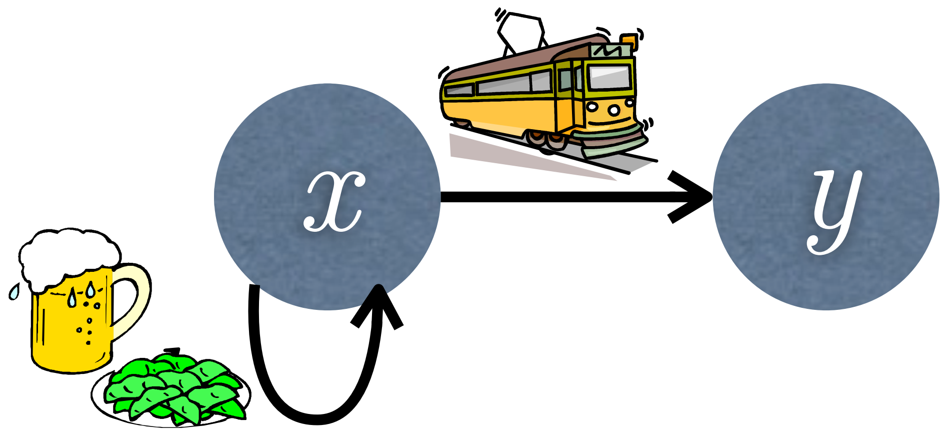


Coalgebra

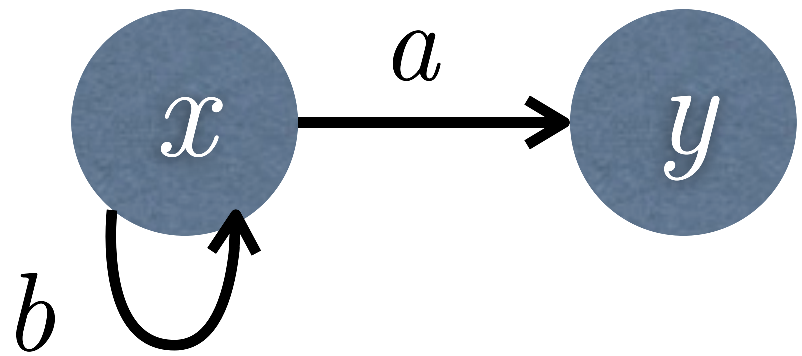




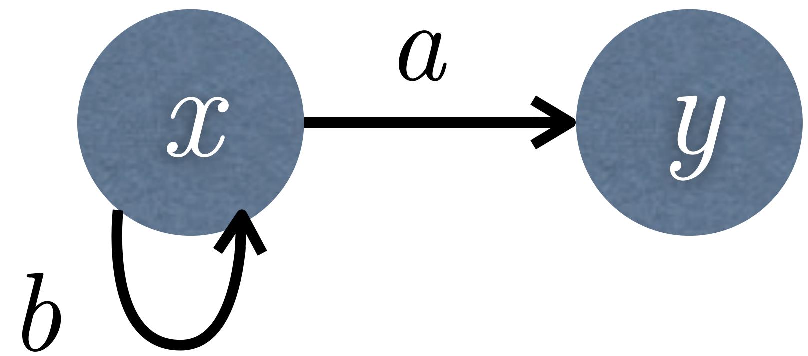
$$\text{as} \quad \begin{array}{c} \boldsymbol{F}\boldsymbol{X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$



as $\begin{array}{c} \boldsymbol{F}\boldsymbol{X} \\ \uparrow \\ \boldsymbol{X} \end{array}$ in $\mathbb{C}$.

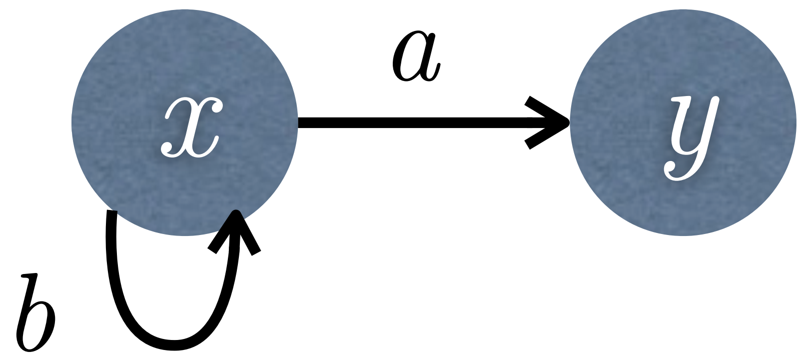


$$\text{as} \quad \begin{array}{c} \boldsymbol{FX} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$



$$\text{as}\quad \begin{array}{c} \boldsymbol{F X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$

$$\begin{array}{ccc} \mathcal{P}(\boldsymbol{\Sigma} \times \boldsymbol{X}) & \{(\boldsymbol{b}, \boldsymbol{x}), (\boldsymbol{a}, \boldsymbol{y})\} & \emptyset \\ \uparrow & \uparrow & \uparrow \\ \boldsymbol{X} & \boldsymbol{x} & \boldsymbol{y} \end{array}$$

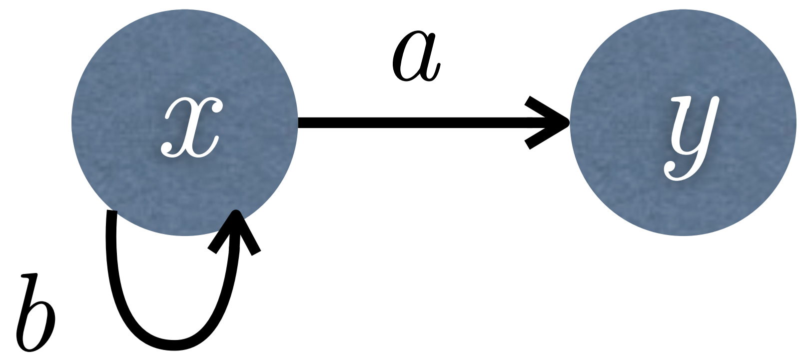


$\mathbb{C} = \mathbf{Sets}$

$F = \mathcal{P}(\Sigma \times \_\,)$

$$\text{as} \quad \begin{array}{c} \boldsymbol{FX} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$

$$\begin{array}{c} \boldsymbol{\mathcal{P}(\Sigma \times X)} \\ \uparrow \\ \boldsymbol{X} \end{array} \qquad \begin{array}{c} \boldsymbol{\{(b,x),(a,y)\}} \\ \uparrow \\ \boldsymbol{x} \end{array} \qquad \begin{array}{c} \boldsymbol{\emptyset} \\ \uparrow \\ \boldsymbol{y} \end{array}$$



non-det. choice
over (action & continue)

$$\mathbb{C} = \mathbf{Sets}$$
$$F = \mathcal{P}(\Sigma \times \_\,)$$

$$\text{as} \quad \begin{array}{c} FX \\ \uparrow \\ X \end{array} \quad \text{in } \mathbb{C}.$$

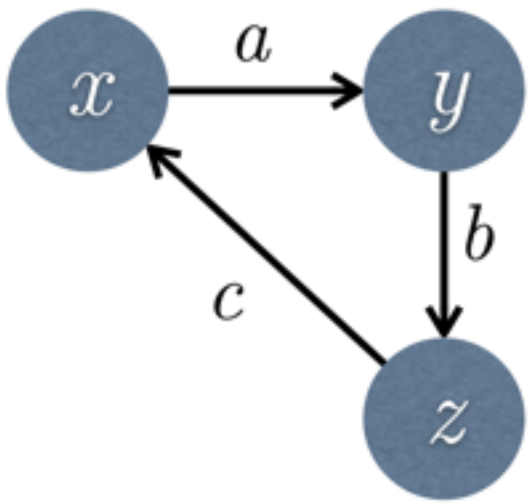
$$\begin{array}{ccc} \mathcal{P}(\Sigma \times X) & \{(b,x),(a,y)\} & \emptyset \\ \uparrow & \uparrow & \uparrow \\ X & x & y \end{array}$$

Jacobs, Rutten, Reichel, ... in 90s

Theory of Coalgebra

	coalgebraically
system	coalgebra $\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$
behavior-preserving map	coalgebra morphism $\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$
behavior	by final coalgebra ("coinduction") $\begin{array}{ccc} FX & \overset{F\mathbf{beh}(c)}{\dashrightarrow} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \underset{\mathbf{beh}(c)}{\dashrightarrow} & Z \end{array}$

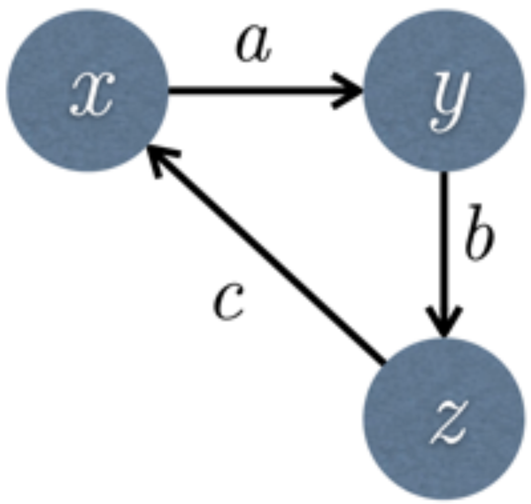
Coinduction: Behavior by Final Coalgebra



$$\text{as} \quad \begin{array}{c} \boldsymbol{F\,X} \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbb{C}.$$

$$\mathbb{C} = \mathbf{Sets}$$
$$\boldsymbol{F} = \Sigma \times \_$$

Coinduction: Behavior by Final Coalgebra



as $\begin{array}{c} FX \\ \uparrow \\ X \end{array}$ in $\mathbb{C}$.

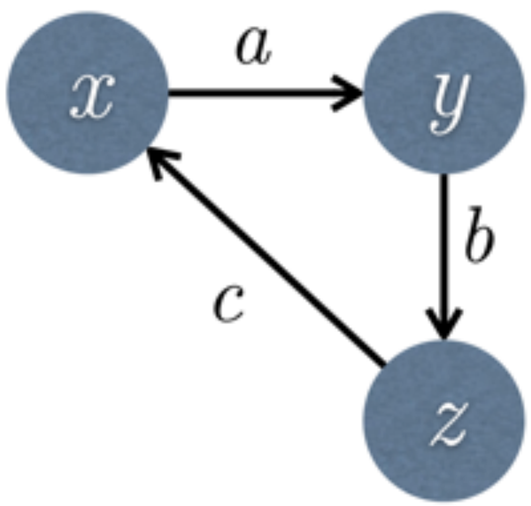
$\mathbb{C} = \mathbf{Sets}$

$F = \Sigma \times \_$

final F*-coalgebra:*

$$\begin{array}{c} \Sigma \times \Sigma^{\mathbb{N}} \\ \cong\uparrow \\ \Sigma^{\mathbb{N}} \end{array} \qquad \begin{array}{c} (a_0, a_1 a_2 \dots) \\ \uparrow \\ a_0 a_1 a_2 \dots \end{array}$$

Coinduction: Behavior by Final Coalgebra



as $\begin{array}{c} FX \\ \uparrow \\ X \end{array}$ in $\mathbb{C}$.

$\mathbb{C} = \mathbf{Sets}$

$F = \Sigma \times \_$

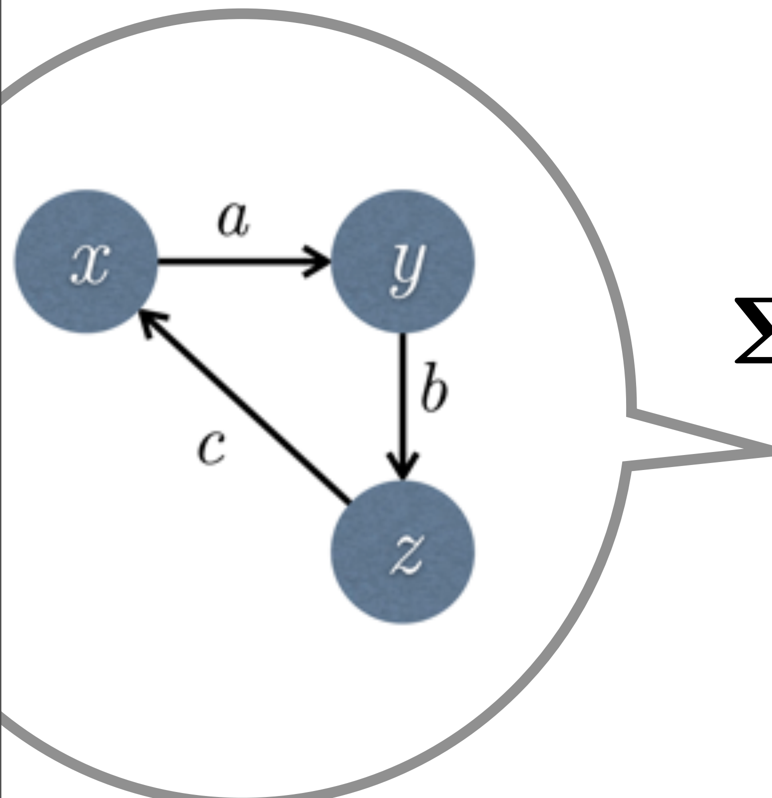
final F-coalgebra:

$$\begin{array}{ccc} \Sigma \times \Sigma^{\mathbb{N}} & & (a_0, a_1 a_2 \ldots) \\ \cong\uparrow & & \uparrow \\ \{\text{possible behaviors}\} = \Sigma^{\mathbb{N}} & & a_0 a_1 a_2 \ldots \end{array}$$

Coinduction: Behavior by Final Coalgebra

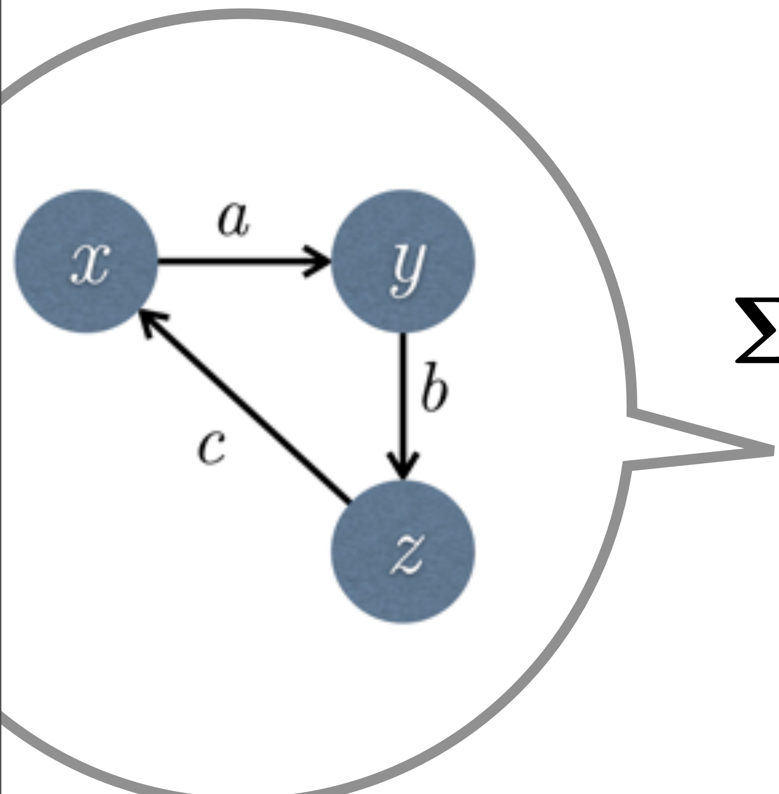
$$\begin{array}{c} \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\ \uparrow \text{final} \\ \boldsymbol{\Sigma}^{\mathbb{N}} \end{array}$$

Coinduction: Behavior by Final Coalgebra



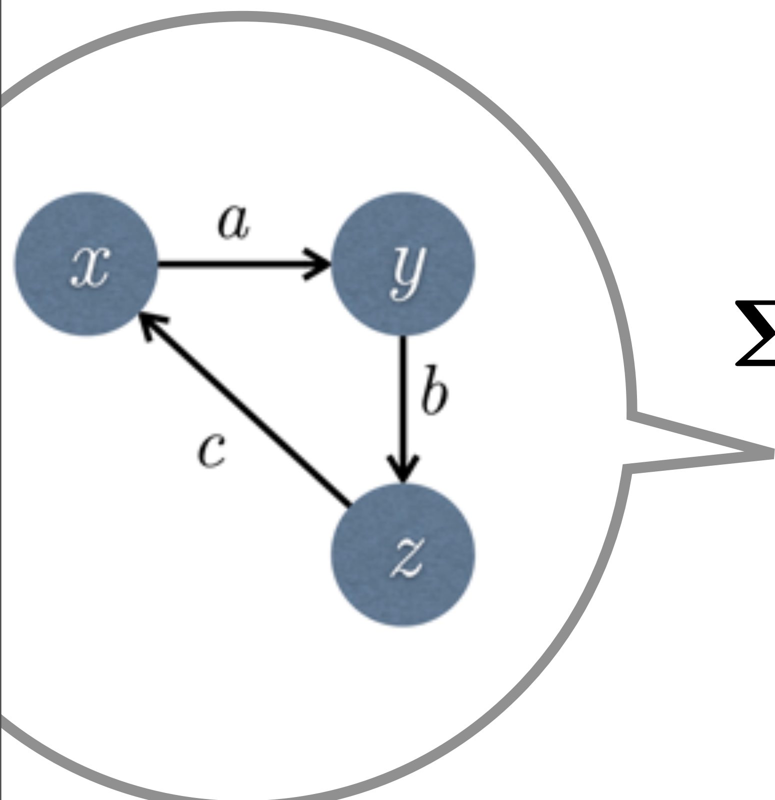
$$\begin{array}{c} \boldsymbol{\Sigma} \times \boldsymbol{X} \\ \boldsymbol{c}\uparrow \\ \boldsymbol{X} \end{array} \qquad\qquad \begin{array}{c} \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\ \uparrow \text{final} \\ \boldsymbol{\Sigma}^{\mathbb{N}} \end{array}$$

Coinduction: Behavior by Final Coalgebra



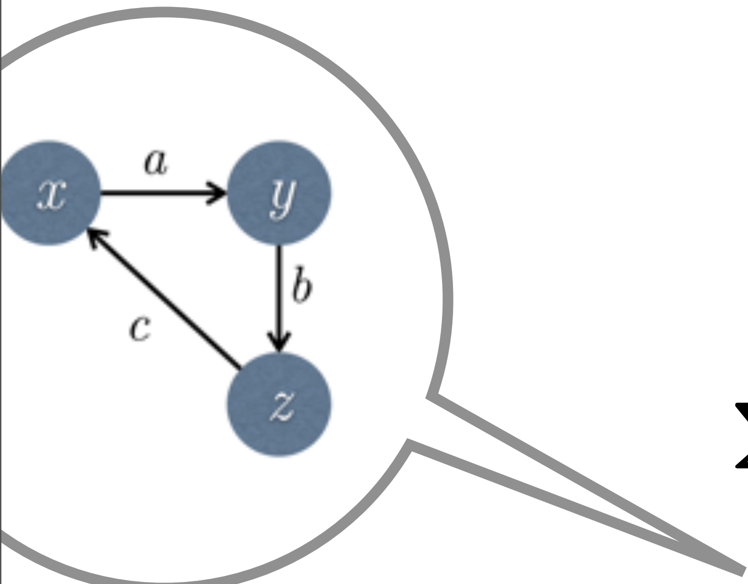
$$\begin{array}{ccc}
\boldsymbol{\Sigma} \times \boldsymbol{X} & \xrightarrow{\ \boldsymbol{\Sigma} \times \textbf{beh}(\boldsymbol{c})\ } & \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\
\boldsymbol{c}\uparrow & & \uparrow \text{final} \\
\boldsymbol{X} & \xrightarrow[\ \textbf{beh}(\boldsymbol{c})\ ]{} & \boldsymbol{\Sigma}^{\mathbb{N}}
\end{array}$$

Coinduction: Behavior by Final Coalgebra



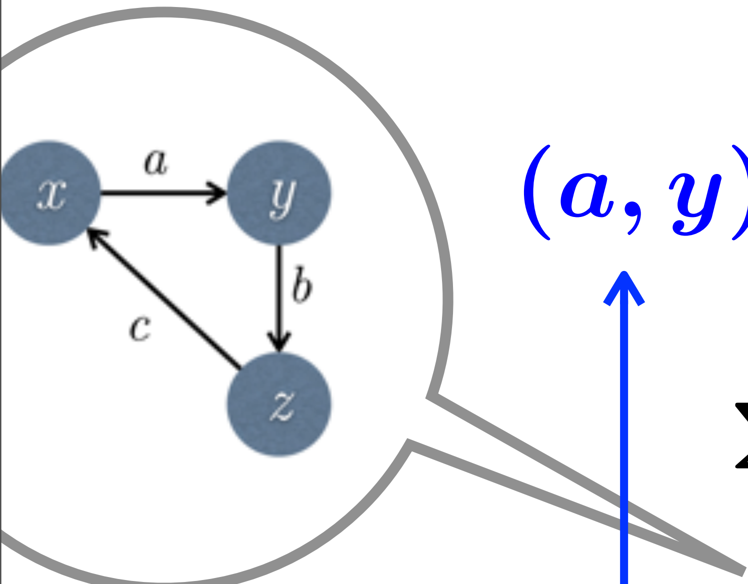
$$\begin{array}{ccc} \boldsymbol{\Sigma} \times \boldsymbol{X} & \xrightarrow{\boldsymbol{\Sigma} \times \textbf{beh}(\boldsymbol{c})} & \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\ \boldsymbol{c} \uparrow & & \uparrow \text{final} \\ \boldsymbol{X} & \xrightarrow[\textbf{beh}(\boldsymbol{c})]{} & \boldsymbol{\Sigma}^{\mathbb{N}} \\ \boldsymbol{x} & \longmapsto & \boldsymbol{abcabc\ldots} \end{array}$$

Coinduction: Behavior by Final Coalgebra



$$
\begin{array}{ccc}
\boldsymbol{\Sigma} \times \boldsymbol{X} & \xrightarrow{\boldsymbol{\Sigma} \times \mathbf{beh}(\boldsymbol{c})} & \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\
\boldsymbol{c} \uparrow & & \uparrow \text{final} \\
\boldsymbol{X} & \xrightarrow[\mathbf{beh}(\boldsymbol{c})]{} & \boldsymbol{\Sigma}^{\mathbb{N}}
\end{array}
$$

Coinduction: Behavior by Final Coalgebra

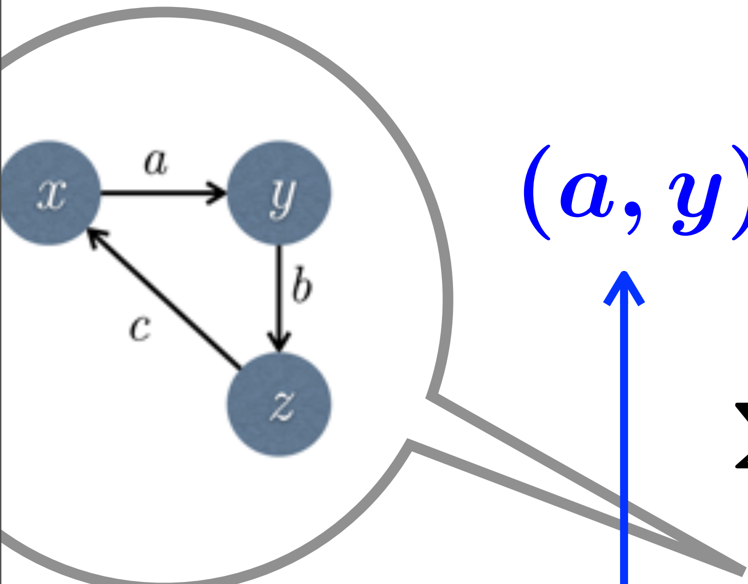


$$(a, y) \longmapsto (\, a, \mathbf{beh}(c)(y) \,)$$

$$\begin{array}{ccc} \Sigma \times X & \xrightarrow{\Sigma \times \mathbf{beh}(c)} & \Sigma \times \Sigma^{\mathbb{N}} \\ c \uparrow & & \uparrow \text{final} \\ X & \xrightarrow[\mathbf{beh}(c)]{} & \Sigma^{\mathbb{N}} \end{array}$$

$$x \longmapsto \mathbf{beh}(c)(x)$$

Coinduction: Behavior by Final Coalgebra



$$
\begin{array}{ccc}
(a, y) & \longmapsto & (\, a, \, \mathbf{beh}(c)(y) \,) \\
\uparrow & & \uparrow \\
& \Sigma \times X \xrightarrow{\;\Sigma \times \mathbf{beh}(c)\;} \Sigma \times \Sigma^{\mathbb{N}} & \\
& c \uparrow \qquad\qquad\qquad \uparrow \text{final} & \\
& X \xrightarrow[\;\mathbf{beh}(c)\;]{} \Sigma^{\mathbb{N}} & \\
\vert & & \vert \\
x & \longmapsto & \mathbf{beh}(c)(x)
\end{array}
$$

beh makes the diagram commute $\iff \mathbf{beh}(c)(x) = a \cdot \mathbf{beh}(c)(y)$

Coinduction: Behavior by Final Coalgebra

Definition. A coalgebra $\begin{matrix} \boldsymbol{FZ} \\ \boldsymbol{\zeta}\uparrow \\ \boldsymbol{Z} \end{matrix}$ is *final* if,

- given any coalgebra $\begin{matrix} \boldsymbol{FX} \\ \boldsymbol{c}\uparrow \\ \boldsymbol{X} \end{matrix}$,
- there is a unique homomorphism from $\boldsymbol{c}$ to $\boldsymbol{\zeta}$:

$$\begin{array}{ccc} \boldsymbol{FX} & \xrightarrow{\boldsymbol{F}\mathsf{beh}(\boldsymbol{c})} & \boldsymbol{FZ} \\ \boldsymbol{c}\uparrow & & \boldsymbol{\zeta}\uparrow\text{final} \\ \boldsymbol{X} & \xrightarrow[\mathsf{beh}(\boldsymbol{c})]{} & \boldsymbol{Z} \end{array}$$

Coinduction: Behavior by Final Coalgebra

Definition. A coalgebra $\begin{array}{c} FZ \\ \zeta\uparrow \\ Z \end{array}$ is *final* if,

- given any coalgebra $\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$,
- there is a unique homomorphism from c to ζ:

$$\begin{array}{ccc} FX & \xrightarrow{F\mathsf{beh}(c)} & FZ \\ c\uparrow & & \zeta\uparrow\text{final} \\ X & \underset{\mathsf{beh}(c)}{\dashrightarrow} & Z \end{array}$$

{possible behaviors}

Jacobs, Rutten, Reichel, ... in 90s

Theory of Coalgebra

	coalgebraically
system	coalgebra $\begin{array}{c} \boldsymbol{FX} \\ \boldsymbol{c}\uparrow \\ \boldsymbol{X} \end{array}$
behavior-preserving map	coalgebra morphism $\begin{array}{ccc} \boldsymbol{FX} & \xrightarrow{\boldsymbol{Ff}} & \boldsymbol{FY} \\ \boldsymbol{c}\uparrow & & \uparrow\boldsymbol{d} \\ \boldsymbol{X} & \xrightarrow[\boldsymbol{f}]{} & \boldsymbol{Y} \end{array}$
behavior	by final coalgebra ("coinduction") $\begin{array}{ccc} \boldsymbol{FX} & \overset{\boldsymbol{F}\mathbf{beh}(\boldsymbol{c})}{\dashrightarrow} & \boldsymbol{FZ} \\ \boldsymbol{c}\uparrow & & \uparrow\text{final} \\ \boldsymbol{X} & \underset{\mathbf{beh}(\boldsymbol{c})}{\dashrightarrow} & \boldsymbol{Z} \end{array}$

"Categorical Disciplines" in Computer Science

- Semantics of functional programming, λ-calculus (Hasegawa, Hasegawa, Kakutani, Katsumata, ...)
  - Terminating vs. non-terminating, reactive
- Algebraic data type, program calculation (Hu, Matsuzaki, Morihata, Takeichi, ...)
- Graph rewriting via pushouts

Category

Definition. A *category* $\mathbb{C}$ consists of

- a collection $\mathbf{obj}(\mathbb{C})$ of *objects* and
- a collection $\mathbb{C}(X, Y)$ of *arrows* from X to Y, for each $X, Y \in \mathbf{obj}(\mathbb{C})$,

equipped with

- an *identity arrow* $\mathbf{id}_X : X \to X$ for each $X \in \mathbf{obj}(\mathbb{C})$ and
- *composition* $g \circ f$ of arrows for each successive $X \xrightarrow{f} Y \xrightarrow{g} Z$.

Category

Definition. A *category* $\mathbb{C}$ consists of

- a collection $\mathbf{obj}(\mathbb{C})$ of *objects* and
- a collection $\mathbb{C}(X, Y)$ of *arrows* from X to Y, for each $X, Y \in \mathbf{obj}(\mathbb{C})$,

equipped with

- an *identity arrow* $\mathbf{id}_X : X \to X$ for each $X \in \mathbf{obj}(\mathbb{C})$ and
- *composition* $g \circ f$ of arrows for each successive $X \xrightarrow{f} Y \xrightarrow{g} Z$.

Sets

obj. X: a set

arr. $f\colon X \to Y$: a function

BA

obj. X:

a Boolean algebra

arr. $f\colon X \to Y$:

a homomorphism

Category

Definition. A *category* $\mathbb{C}$ consists of

- a collection $\mathbf{obj}(\mathbb{C})$ of *objects* and
- a collection $\mathbb{C}(\boldsymbol{X}, \boldsymbol{Y})$ of *arrows* from $\boldsymbol{X}$ to $\boldsymbol{Y}$, for each $\boldsymbol{X}, \boldsymbol{Y} \in \mathbf{obj}(\mathbb{C})$,

equipped with

- an *identity arrow* $\mathbf{id}_{\boldsymbol{X}} : \boldsymbol{X} \to \boldsymbol{X}$ for each $\boldsymbol{X} \in \mathbf{obj}(\mathbb{C})$ and
- *composition* $\boldsymbol{g} \circ \boldsymbol{f}$ of arrows for each successive $\boldsymbol{X} \xrightarrow{\boldsymbol{f}} \boldsymbol{Y} \xrightarrow{\boldsymbol{g}} \boldsymbol{Z}$.

Sets

obj. X: a set

arr. $f\colon X \to Y$: a function

BA

obj. X:

a Boolean algebra

arr. $f\colon X \to Y$:

a homomorphism

Hask

obj. X:

a Haskell type

arr. $f\colon X \to Y$:

a program

CPO

obj. X:

a compl. partial order

arr. $f\colon X \to Y$:

a continuous function

Category

Definition. A *category* $\mathbb{C}$ consists of

- a collection $\mathbf{obj}(\mathbb{C})$ of *objects* and
- a collection $\mathbb{C}(X, Y)$ of *arrows* from X to Y, for each $X, Y \in \mathbf{obj}(\mathbb{C})$,

equipped with

- an *identity arrow* $\mathbf{id}_X : X \to X$ for each $X \in \mathbf{obj}(\mathbb{C})$ and
- *composition* $g \circ f$ of arrows for each successive $X \xrightarrow{f} Y \xrightarrow{g} Z$.

Sets

obj. X: a set

arr. $f: X \to Y$: a function

BA

obj. X: a Boolean algebra

arr. $f: X \to Y$: a homomorphism

$\mathbf{N}_{\leq}$

obj. X: a natural number

arr. $f: X \to Y$: the order $\leq$

Hask

obj. X: a Haskell type

arr. $f: X \to Y$: a program

CPO

obj. X: a compl. partial order

arr. $f: X \to Y$: a continuous function

2 Recent Topics in Coalgebra

- Bisimilarity
  - esp. for probabilistic systems (Panangaden, Sokolova, ...)
- Coalgebraic modal logic
  - Transfer conventional results (Cirstea, Pattinson, Roessiger, Schroeder, ...)
  - Via the Stone duality (Bonsangue, Gehrke, Kupke, Kurz, Venema, ...)
- Process algebra, SOS
  - Bialgebraic modeling (Klin, Plotkin, Turi, ...)
  - Component calculi (Barbosa, Clarke, Silva, ...)
- Coinductive data type in functional programming (Capretta, Uustalu, Vene, ...)

LTS

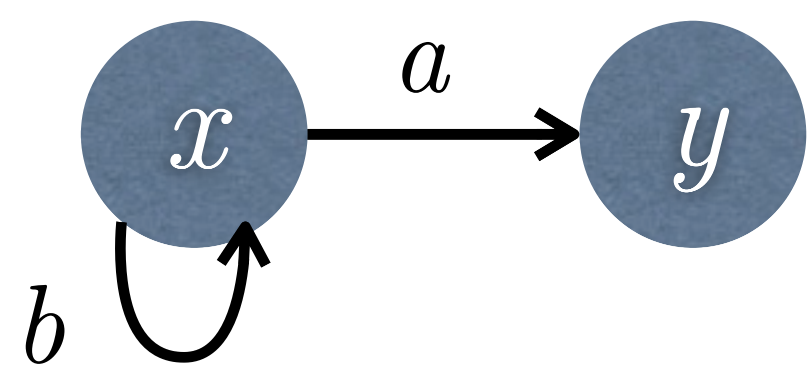
Definition. A *labeled transition system (LTS)* is a triple

$$(\boldsymbol{X}, \boldsymbol{\Sigma}, \{\xrightarrow{\boldsymbol{a}}\}_{\boldsymbol{a}\in\boldsymbol{\Sigma}})$$

where

- $\boldsymbol{X}$ is a non-empty set of *states*;
- $\boldsymbol{\Sigma}$ is a non-empty set of *labels*;
- $\xrightarrow{\boldsymbol{a}} \subseteq \boldsymbol{X} \times \boldsymbol{X}$ is a binary relation, for each $\boldsymbol{a} \in \boldsymbol{\Sigma}$.

- Non-determinism
- Well-accepted model of systems/processes (cf. Milner)



LTS

Definition. A *labeled transition system (LTS)* is a triple

$$(X, \Sigma, \{\xrightarrow{a}\}_{a \in \Sigma})$$

where

- X is a non-empty set of *states*;
- Σ is a non-empty set of *labels*;
- $\xrightarrow{a} \subseteq X \times X$ is a binary relation, for each $a \in \Sigma$.

$$\mathcal{P}(\Sigma \times X) \qquad \{\, (a, x') \mid a \in \Sigma, x \xrightarrow{a} x' \,\}$$

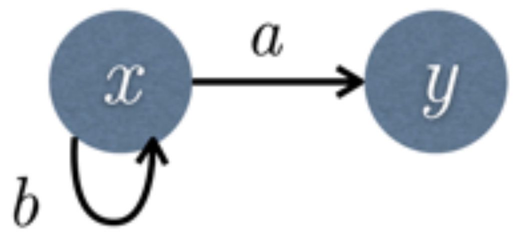
$$\uparrow \qquad\qquad\qquad\qquad \uparrow$$

$$X \qquad\qquad\qquad\qquad x$$

LTS = $\mathcal{P}(\Sigma \times \_\,)$-coalgebra

Coalgebraic Modal Logic

LTS



- (Model of) system

Modal logic

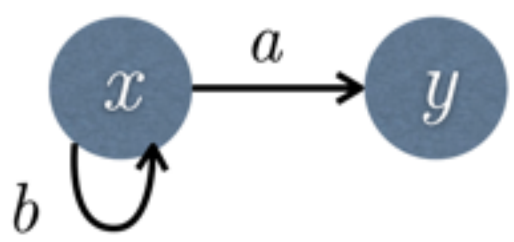
$\Diamond_{\boldsymbol{a}} \top$ a-transition is possible

$\mathbf{Req} \Rightarrow \mathbf{G}(\top \mathbf{U} \mathbf{Res})$
request is eventually responded

- LTL, CTL, μ-calculus,...
- Specification language

Coalgebraic Modal Logic

LTS



- (Model of) system

Semantics

$x \models \Diamond_a \varphi \overset{\text{def.}}{\iff}$
$\exists x'. \, (\, x \xrightarrow{a} x' \;\&\; x' \models \varphi \,)$

- Model checking, satisfiability check, ...

Modal logic

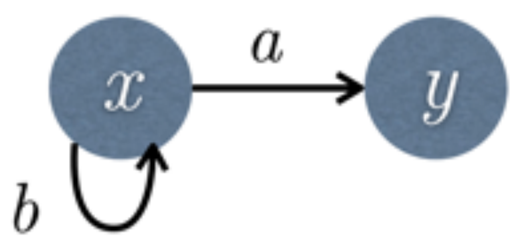
$\Diamond_{\boldsymbol{a}} \top$ a-transition is possible

$\mathbf{Req} \Rightarrow \mathbf{G}(\top \mathbf{U} \mathbf{Res})$
request is eventually responded

- LTL, CTL, μ-calculus,...
- Specification language

Coalgebraic Modal Logic

LTS



- (Model of) system

Semantics

$x \models \Diamond_a \varphi \overset{\text{def.}}{\iff}$
$\exists x'.\,(\,x \xrightarrow{a} x' \;\&\; x' \models \varphi\,)$

- Model checking, satisfiability check, ...

Modal logic

$\Diamond_{\boldsymbol{a}} \top$ a-transition is possible

$\mathbf{Req} \Rightarrow \mathbf{G}(\top \mathbf{U} \mathbf{Res})$
request is eventually responded

- LTL, CTL, µ-calculus,...
- Specification language

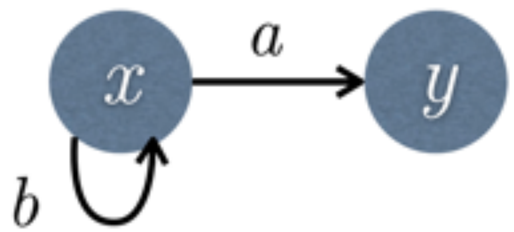
Coalgebra

$$\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$$

- Non-deterministic, probabilistic, ...

Coalgebraic Modal Logic

LTS



- (Model of) system

Semantics

$$x \models \Diamond_a \varphi \overset{\text{def.}}{\Longleftrightarrow} \exists x'.\,(\, x \overset{a}{\to} x' \;\&\; x' \models \varphi \,)$$

- Model checking, satisfiability check, ...

Modal logic

$\Diamond_a \top$ a-transition is possible

$\mathbf{Req} \Rightarrow \mathbf{G}(\top \mathbf{U} \mathbf{Res})$
request is eventually responded

- LTL, CTL, μ-calculus,...
- Specification language

Coalgebra

$$\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$$

- Non-deterministic, probabilistic, ...

??

Modal logic

??

Coalgebraic Modal Logic I: via Predicate Liftings

Coalgebra

$$\begin{array}{c} FX \\ c \uparrow \\ X \end{array}$$

- Non-deterministic, probabilistic, ...

??

Modal logic

??

Cirstea, Pattinson, Roessiger, Schroeder, ...

Coalgebraic Modal Logic I: via Predicate Liftings

Coalgebra

$$\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$$

- Non-deterministic, probabilistic, ...

??

Modal logic

Given/assumed set of modalities Λ

Coalgebraic Modal Logic I: via Predicate Liftings

Coalgebra

- Non-deterministic, probabilistic, ...

$$\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$$

Modal logic

Given/assumed set of modalities Λ

- Assumed: for each $L \in \Lambda$, a *predicate lifting*

$$2^X \xrightarrow{\lambda_L} 2^{FX} \qquad \text{(natural in } X\text{)}$$

- $x \models L\varphi \stackrel{\text{def.}}{\iff} c(x) \in \lambda_L(\,[\![\varphi]\!]\,)$

Coalgebraic Modal Logic I: via Predicate Liftings

Coalgebra

$$\begin{array}{c} FX \\ c \uparrow \\ X \end{array}$$

- Non-deterministic, probabilistic, ...

Modal logic

Given/assumed set of modalities Λ

- Assumed: for each $L \in \Lambda$, a *predicate lifting*

$$2^X \xrightarrow{\lambda_L} 2^{FX} \qquad \text{(natural in } X\text{)}$$

- $$x \models L\varphi \overset{\text{def.}}{\iff} c(x) \in \underbrace{\lambda_L\big(\underbrace{[\![\varphi]\!]}_{\in 2^X}\big)}_{\in 2^{FX}}$$

Coalgebraic Modal Logic I: via Predicate Liftings

- Assumed: for each $\boldsymbol{L} \in \boldsymbol{\Lambda}$, a *predicate lifting*

$$\mathbf{2}^{\boldsymbol{X}} \xrightarrow{\boldsymbol{\lambda_L}} \mathbf{2}^{\boldsymbol{FX}} \quad \text{(natural in } \boldsymbol{X}\text{)}$$

- $\boldsymbol{x} \models \boldsymbol{L\varphi} \overset{\text{def.}}{\iff} \boldsymbol{c(x)} \in \boldsymbol{\lambda_L}\big(\,[\![\boldsymbol{\varphi}]\!]\,\big)$

Coalgebraic Modal Logic I: via Predicate Liftings

- Assumed: for each $\boldsymbol{L} \in \boldsymbol{\Lambda}$, a *predicate lifting*
  $$\mathbf{2}^{\boldsymbol{X}} \xrightarrow{\boldsymbol{\lambda_L}} \mathbf{2}^{\boldsymbol{FX}} \quad (\text{natural in } \boldsymbol{X})$$
- $\boldsymbol{x} \models \boldsymbol{L\varphi} \overset{\text{def.}}{\iff} \boldsymbol{c(x)} \in \boldsymbol{\lambda_L}(\,[\![\boldsymbol{\varphi}]\!]\,)$

- Kripke frame ($\Diamond$,$\Box$), LTS ($\Diamond_a$), probabilistic system ($\Diamond_p$), neighborhood structure, coalition logic, ...

Coalgebraic Modal Logic I: via Predicate Liftings

- Assumed: for each $L \in \Lambda$, a *predicate lifting*
  $$2^X \xrightarrow{\lambda_L} 2^{FX} \quad \text{(natural in } X\text{)}$$
- $x \models L\varphi \overset{\text{def.}}{\iff} c(x) \in \lambda_L(\,[\![\varphi]\!]\,)$

- Kripke frame ($\Diamond$,$\Box$), LTS ($\Diamond_a$), probabilistic system ($\Diamond_p$), neighborhood structure, coalition logic, ...
- <u>One-Step Paradigm</u>
  Many conventional techniques can be transferred!
  - Axiomatization, soundness/completeness, finite model property, complexity results, cut-elimination, fixed point operators, ...

Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

Coalgebraic Modal Logic II: via Stone-Like Dualities

Stone duality

$$\mathbf{Stone}^{\mathrm{op}} \underset{S^{\mathrm{op}}}{\overset{P}{\rightleftarrows}} \mathbf{BA} \qquad (\simeq)$$

(state) spaces

propositional logic

$$\mathbf{Sets}^{\mathrm{op}} \underset{S^{\mathrm{op}}}{\overset{P}{\rightleftarrows}} \mathbf{BA} \qquad (\top)$$

Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

Coalgebraic Modal Logic II: via Stone-Like Dualities

Stone duality

$$\mathrm{Stone}^{\mathrm{op}} \underset{S^{\mathrm{op}}}{\overset{P}{\rightleftarrows}} \mathrm{BA} \qquad (\simeq)$$

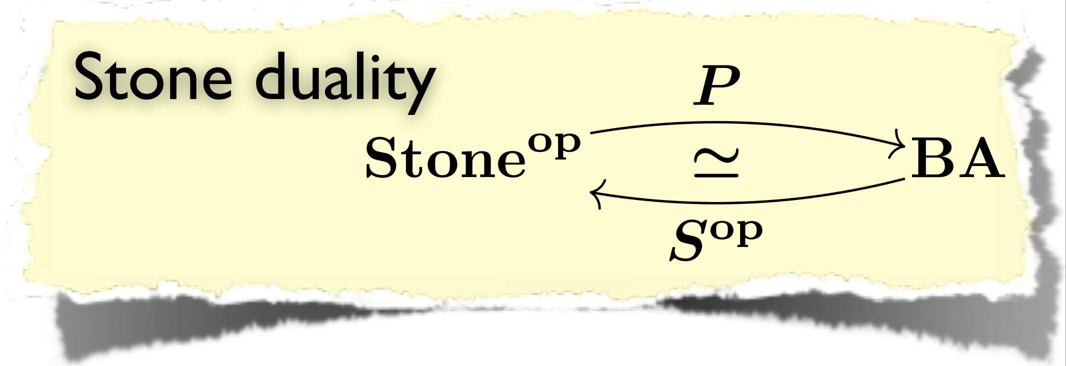
(state) spaces → $\mathbf{Sets}^{\mathrm{op}}$

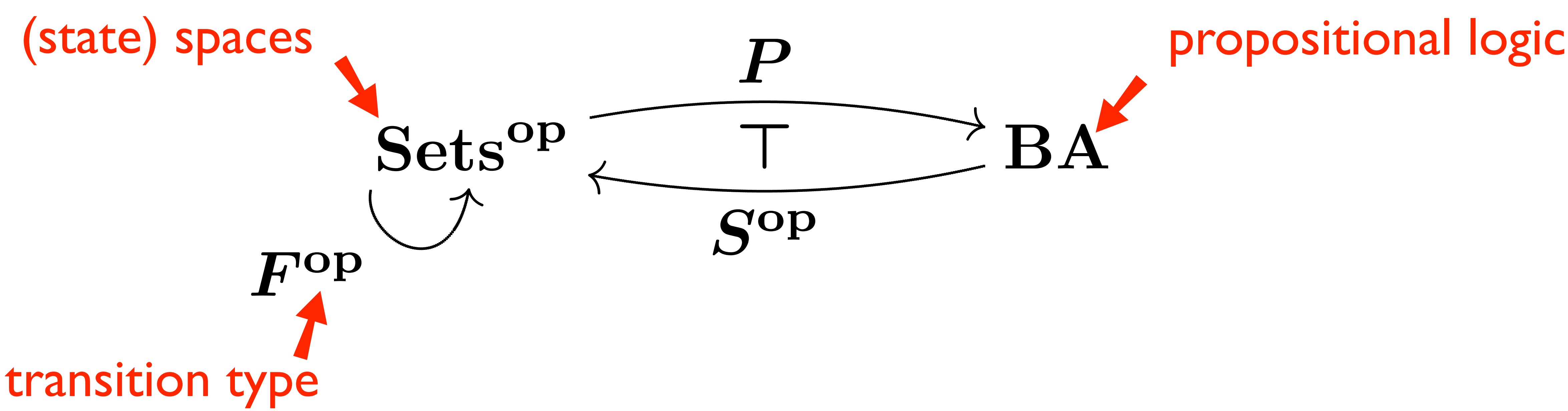
propositional logic → $\mathbf{BA}$

$$F^{\mathrm{op}} \circlearrowright \mathbf{Sets}^{\mathrm{op}} \underset{S^{\mathrm{op}}}{\overset{P}{\rightleftarrows}} \mathbf{BA} \qquad (\top)$$

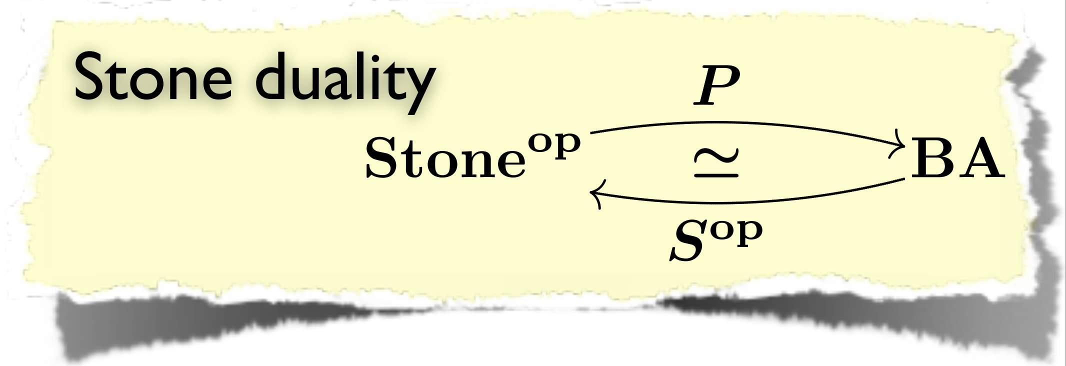
Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

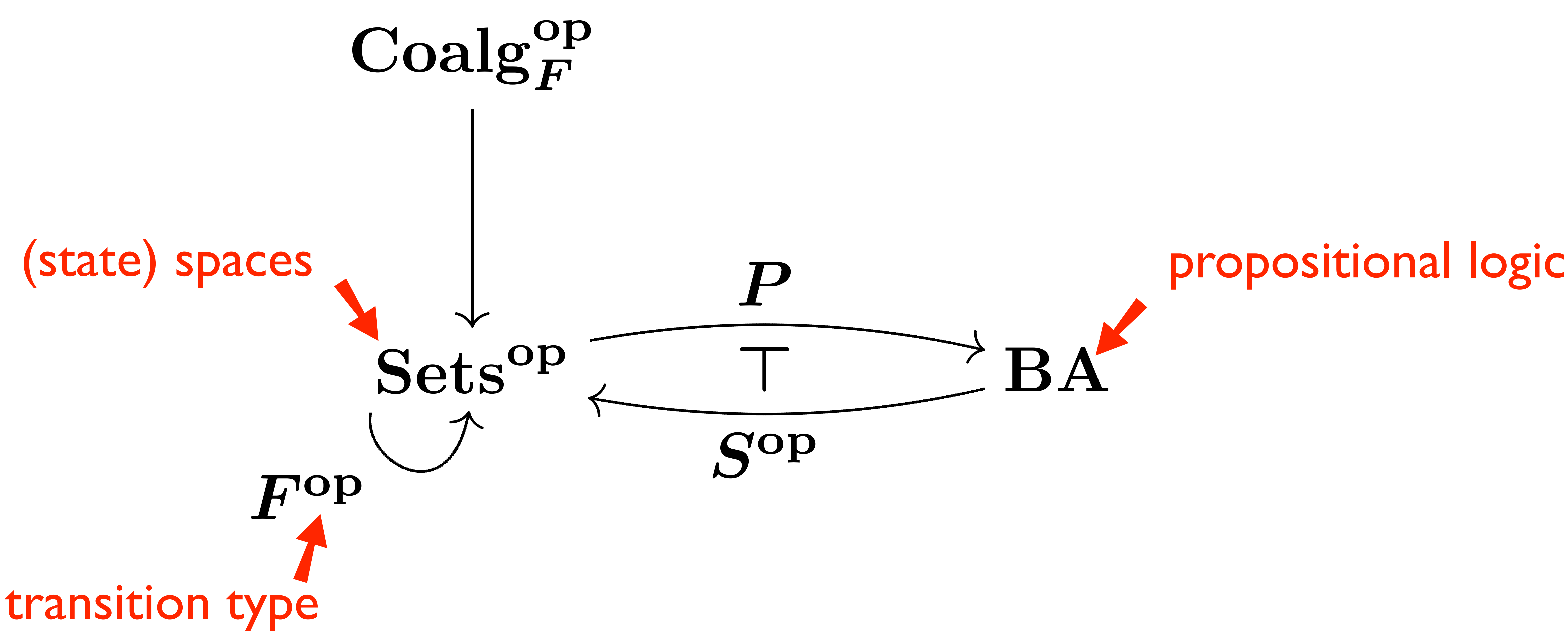
Coalgebraic Modal Logic II: via Stone-Like Dualities





Coalgebraic Modal Logic II: via Stone-Like Dualities



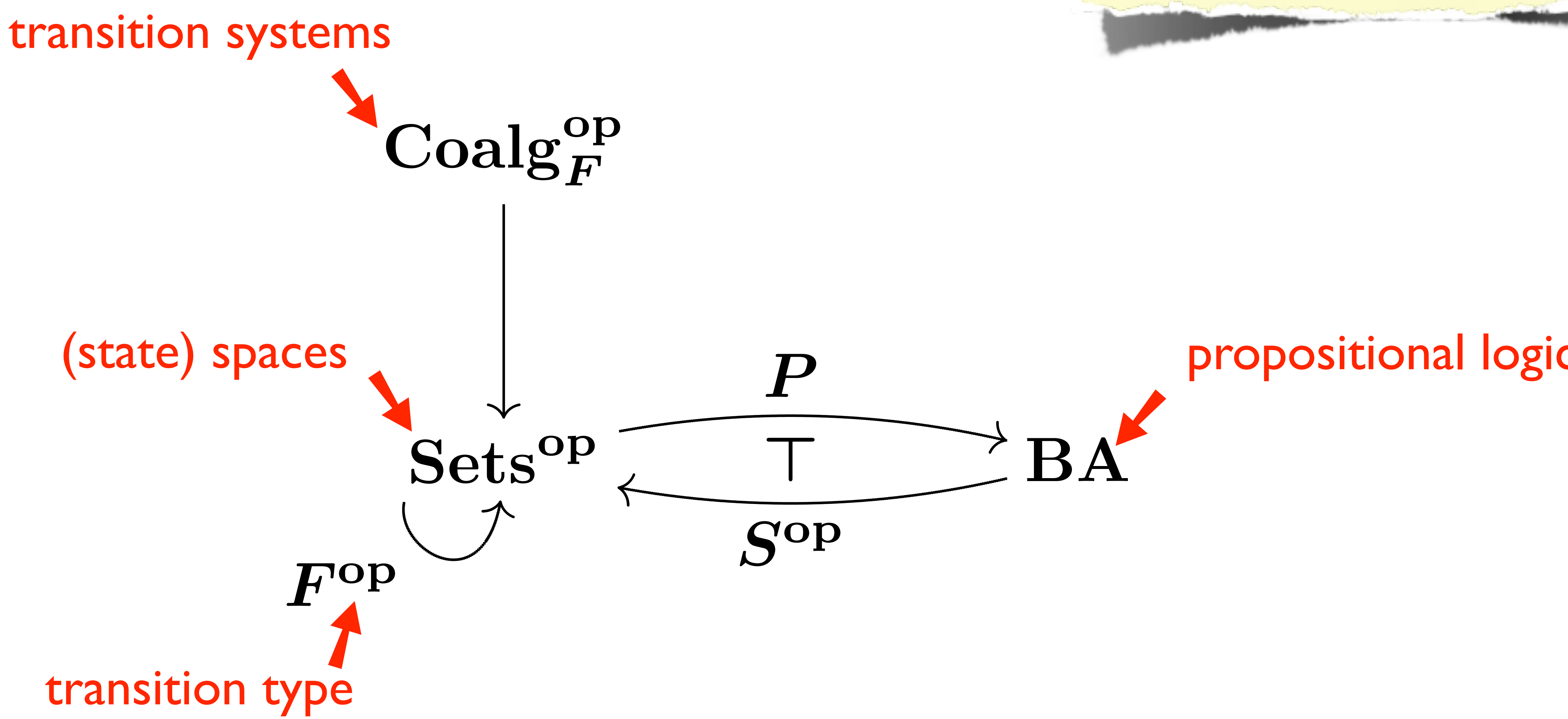


Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

Coalgebraic Modal Logic II: via Stone-Like Dualities

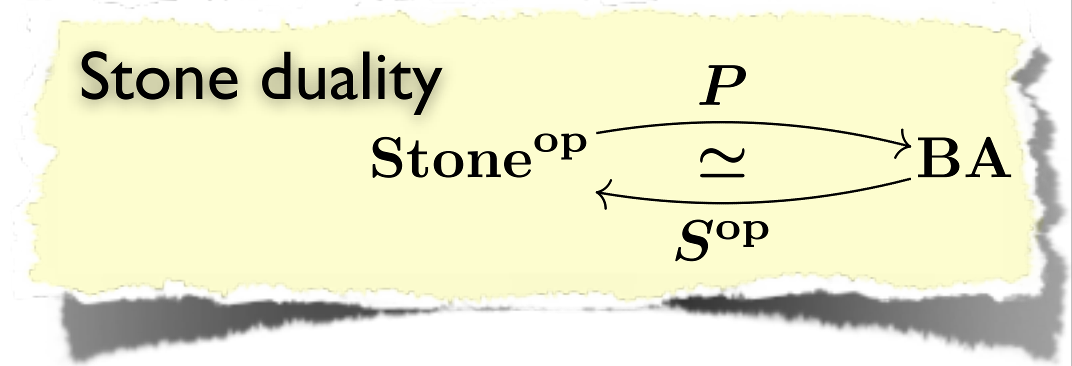
Stone duality

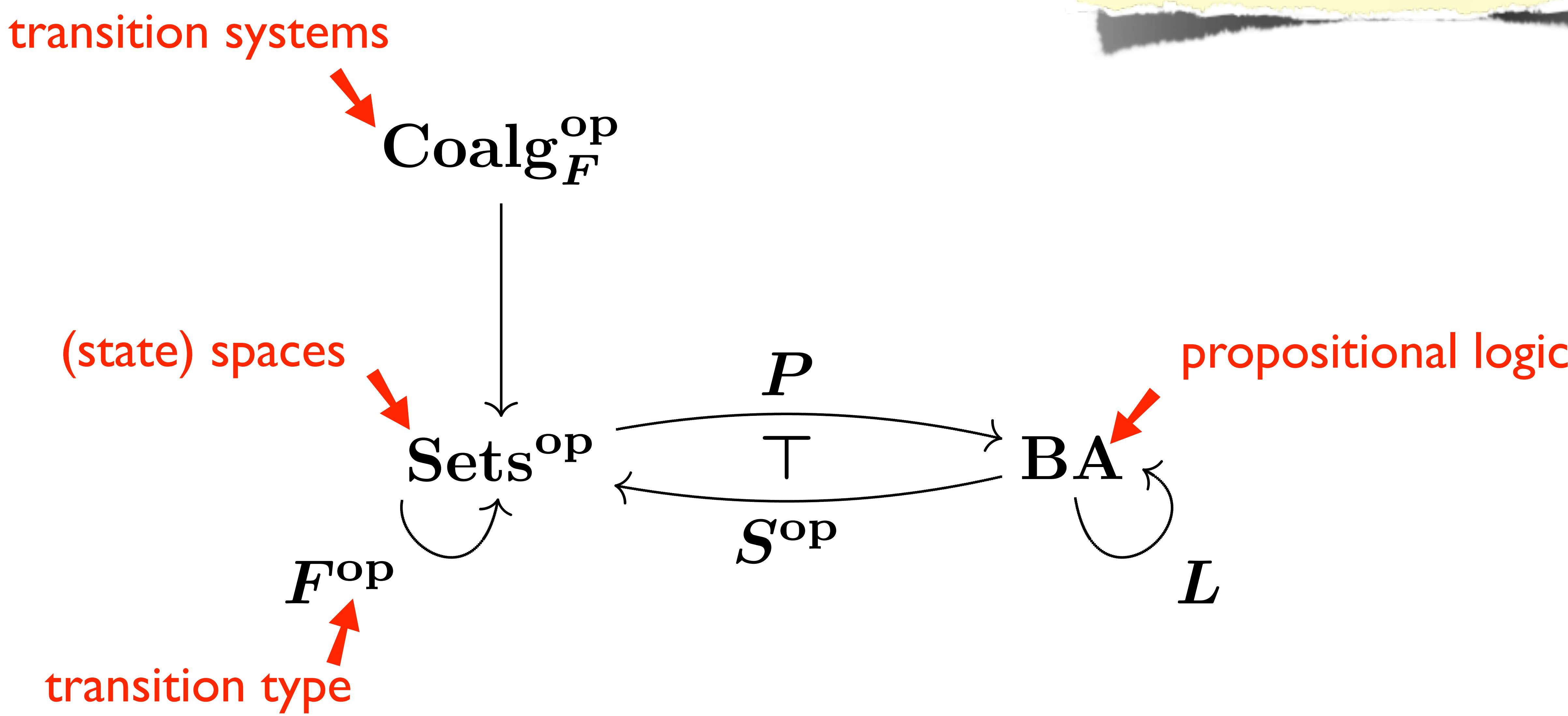
$$\mathbf{Stone}^{op} \underset{S^{op}}{\overset{P}{\rightleftarrows}} \mathbf{BA} \quad (\simeq)$$



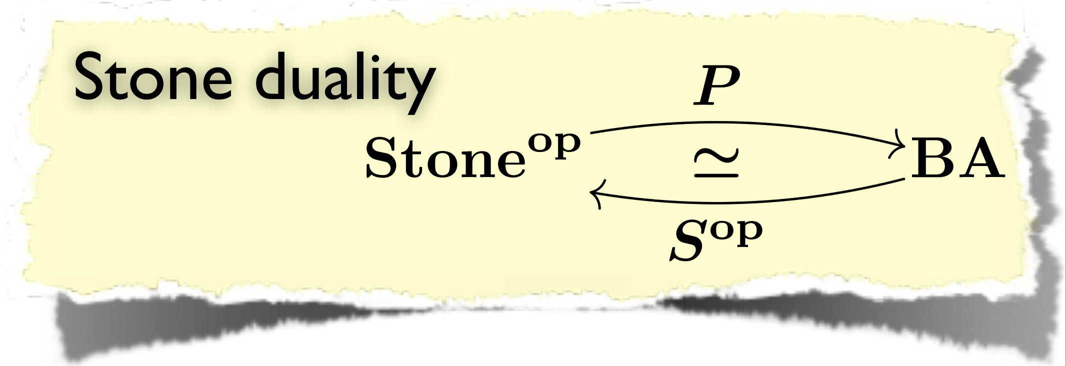
Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

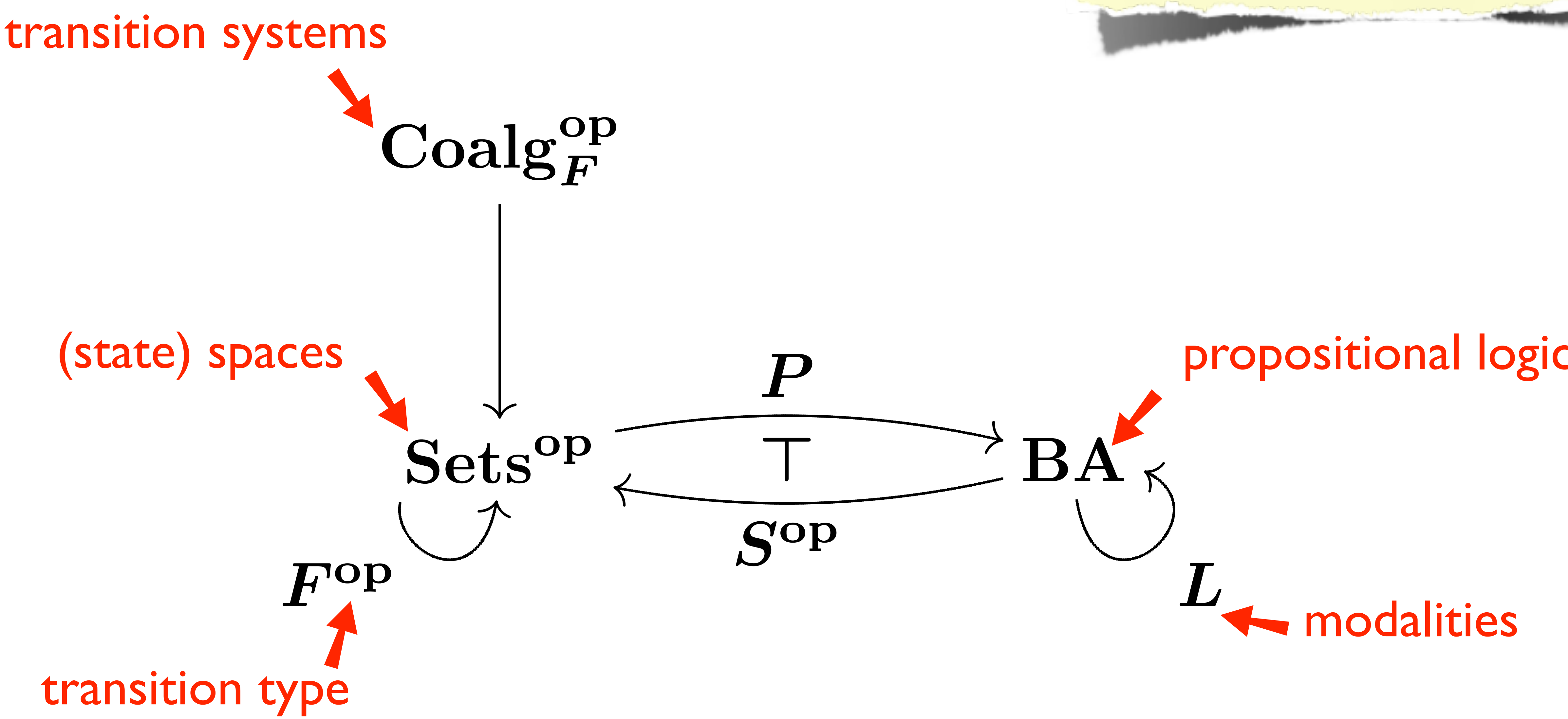
Coalgebraic Modal Logic II: via Stone-Like Dualities





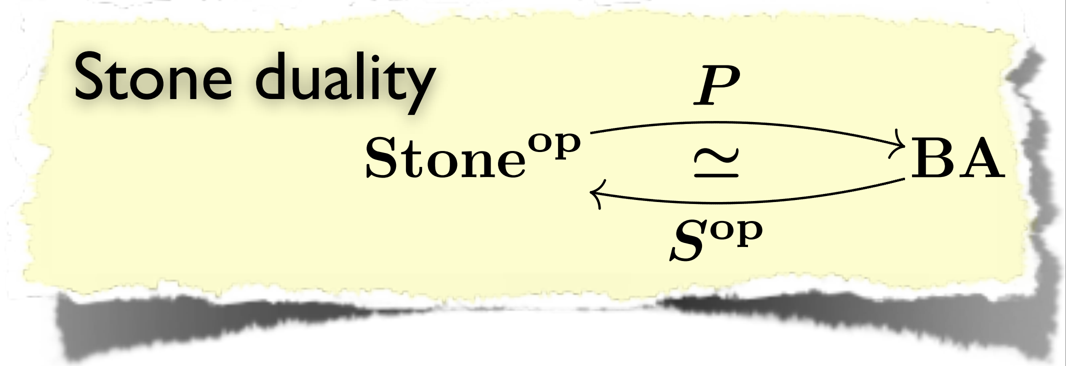
Coalgebraic Modal Logic II: via Stone-Like Dualities

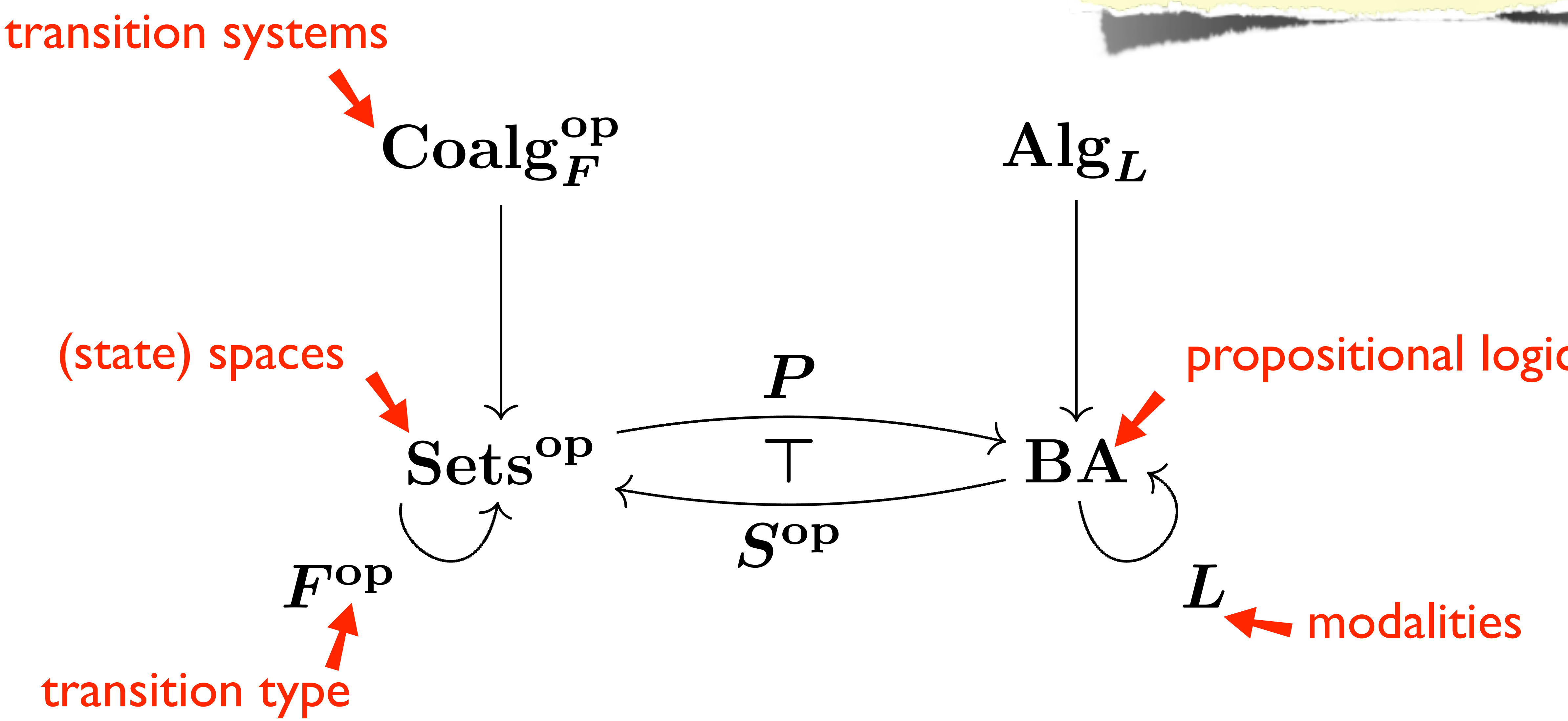




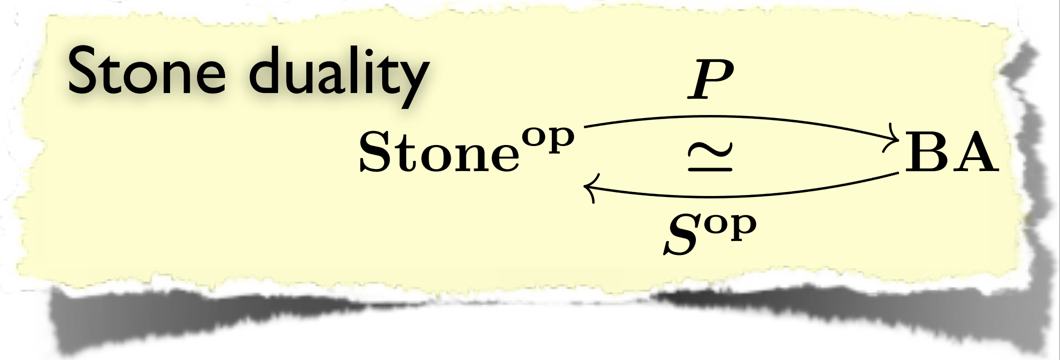
Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

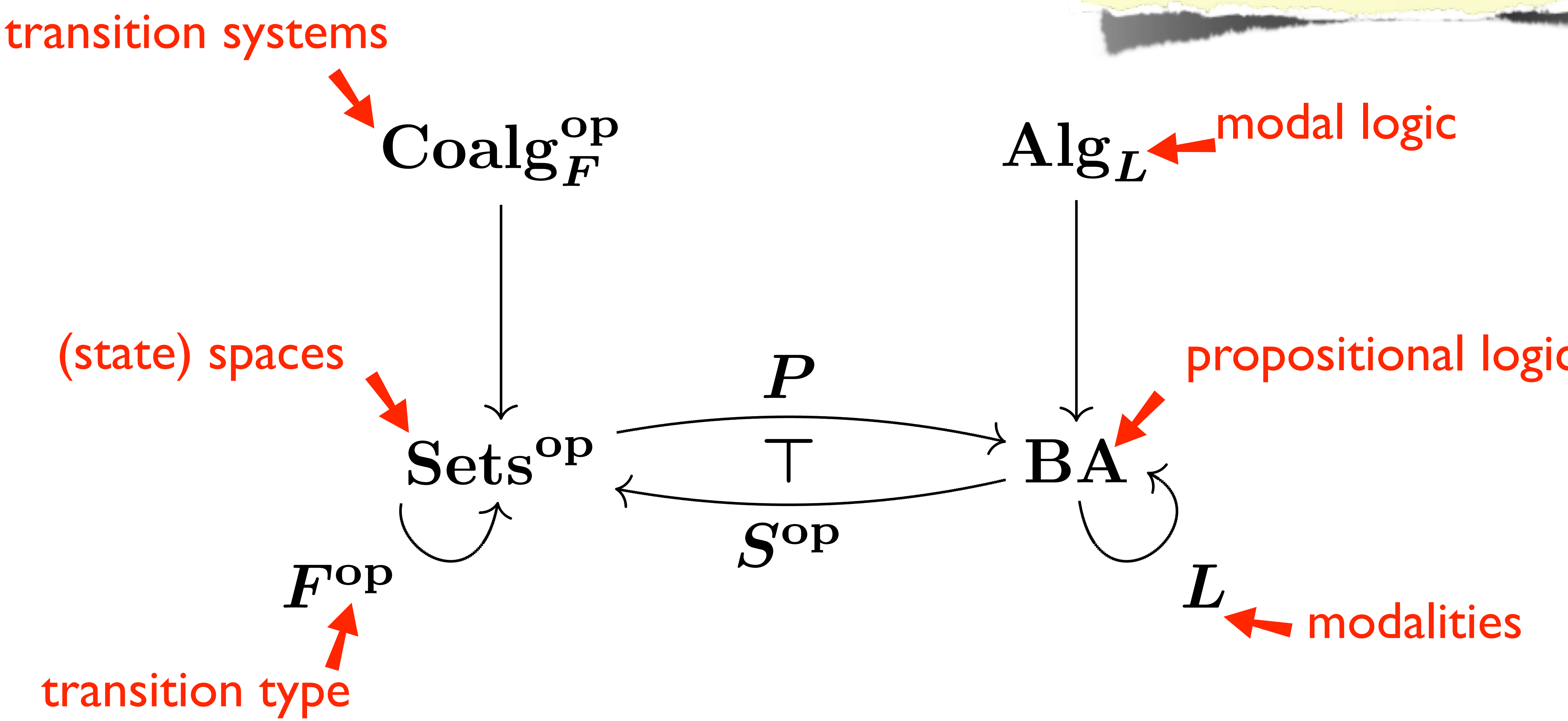
Coalgebraic Modal Logic II: via Stone-Like Dualities



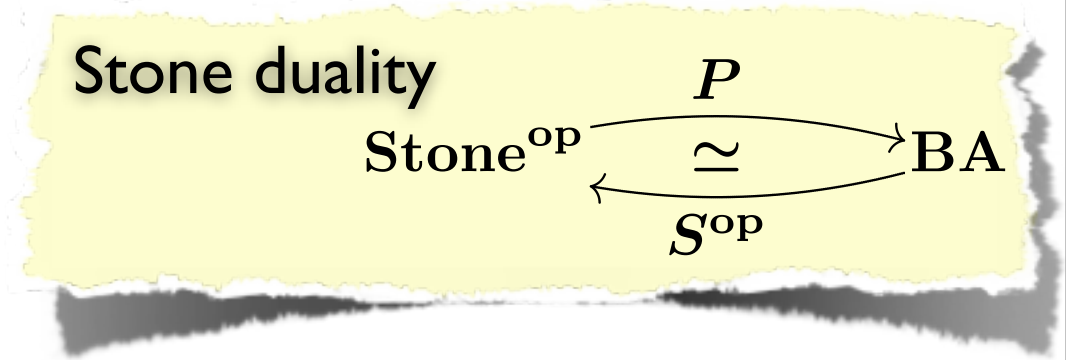


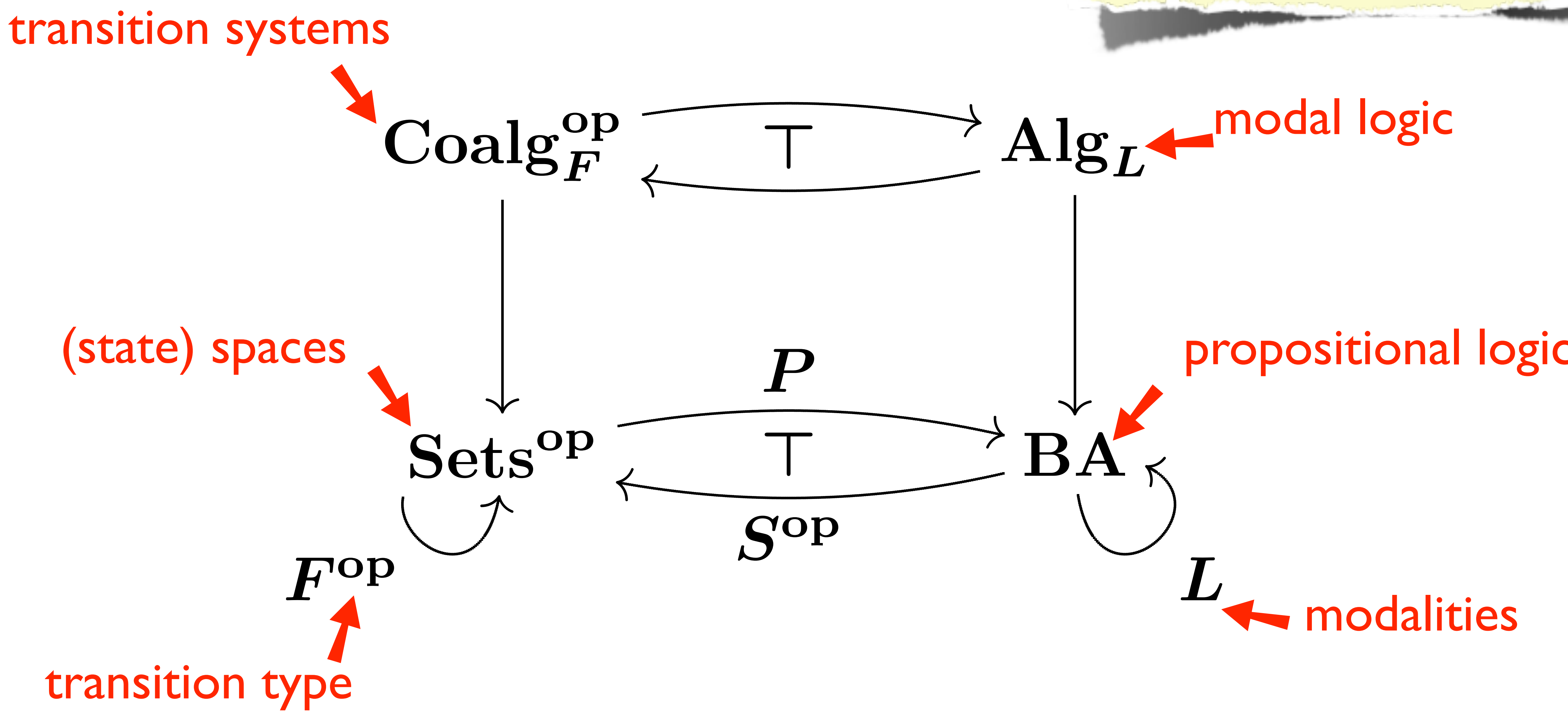
Coalgebraic Modal Logic II: via Stone-Like Dualities





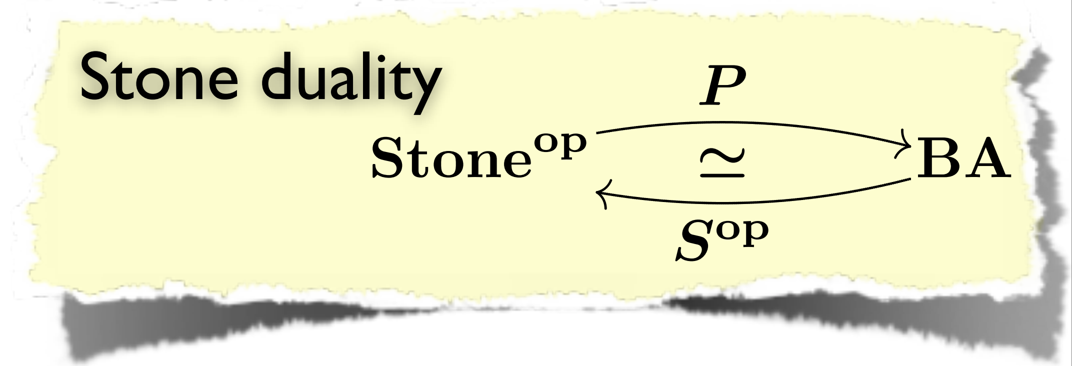
Coalgebraic Modal Logic II: via Stone-Like Dualities

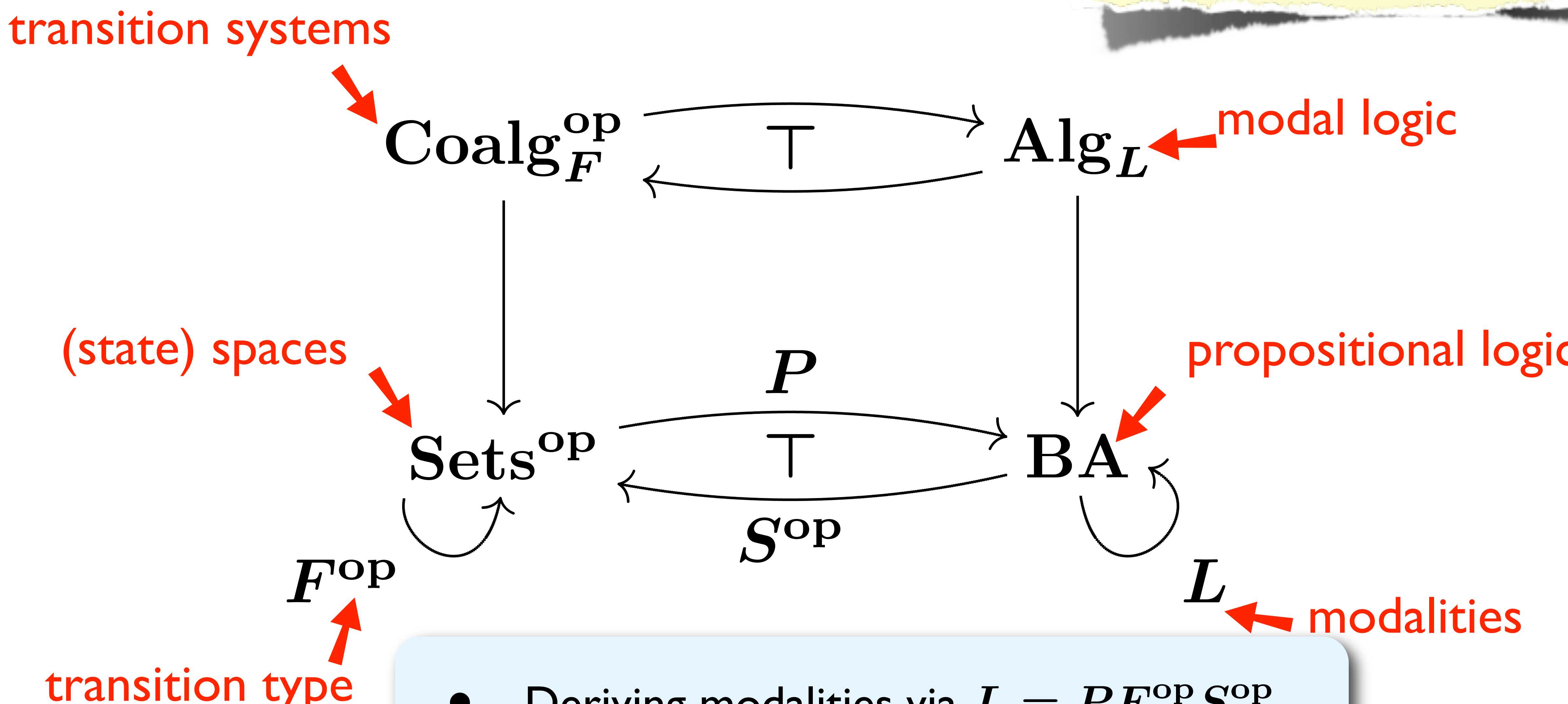




Bonsangue, Gehrke, Kupke, Kurz, Venema, ...

Coalgebraic Modal Logic II: via Stone-Like Dualities





- Deriving modalities via $L = PF^{\text{op}}S^{\text{op}}$
- Semantics by $\lambda : LP \Rightarrow PF^{\text{op}}$

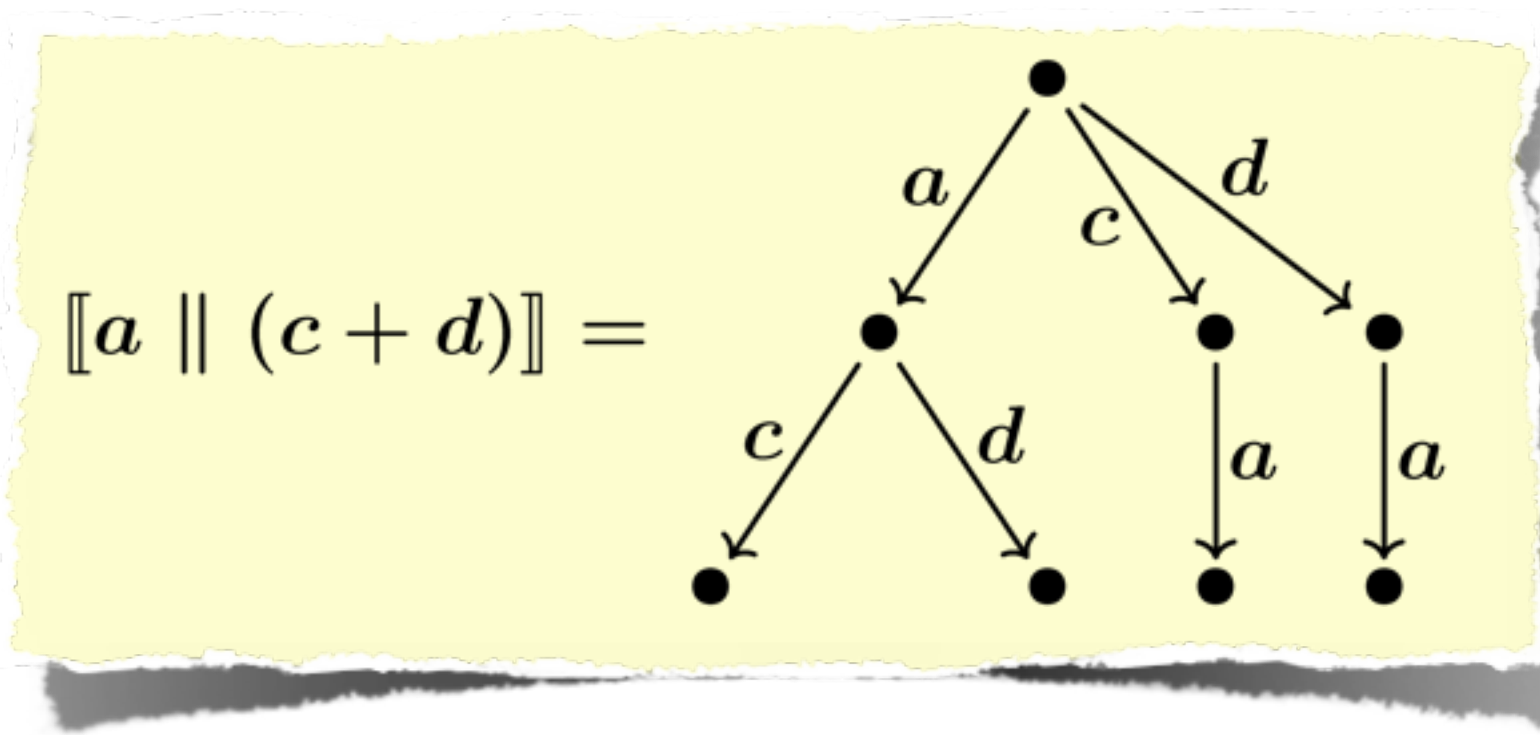
Process Algebra

- Simple "programming language" for describing systems
  (CCS, π-cal. [Milner], CSP [Hoare], ACP [Bergstra-Klop], ...)

$P \parallel Q$	parallel/concurrent composition	*P and Q at the same time*
$P;Q$	sequential composition	*first P, and then Q*
$P+Q$	non-deterministic choice	*do either P or Q*
$!P$	replication	*infinitely many copies of P in parallel*

- E.g. **coin; (boilWater $\parallel$ grindBeans)**

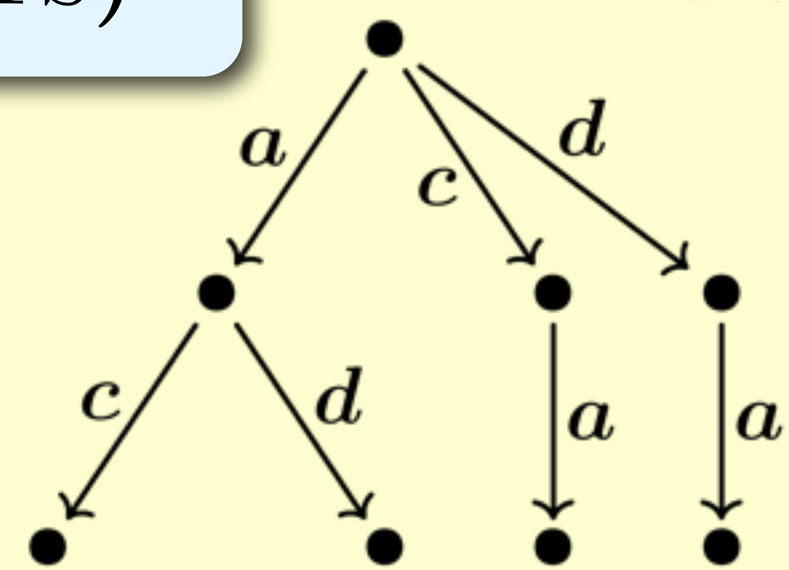
Operational Semantics



Operational Semantics

$$[\![\_]\!] : (\text{process term}) \longmapsto (\text{LTS})$$

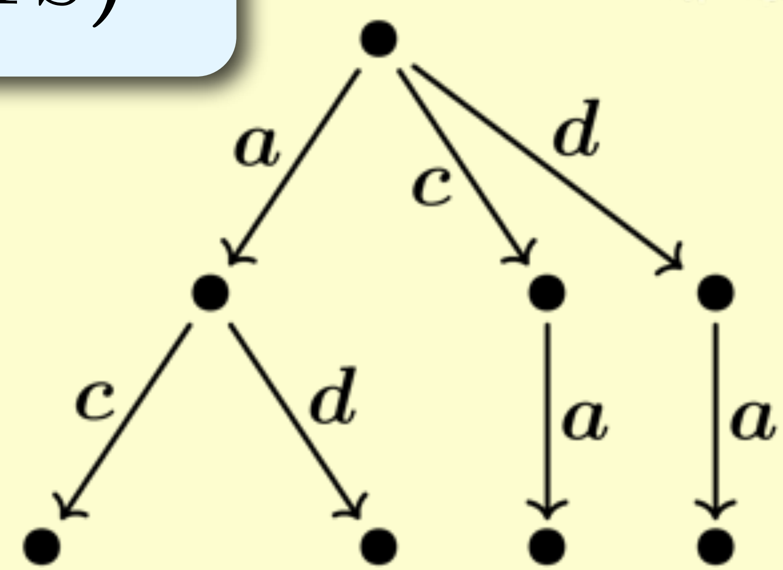
$$[\![a \parallel (c + d)]\!] =$$



Operational Semantics

$$[\![\_]\!] : (\text{process term}) \longmapsto (\text{LTS})$$

$$[\![a \parallel (c + d)]\!] =$$



- Mathematically rigorous definition?

SOS = Structural Operational Semantics

- First introduce SOS rules...

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \; (\|\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'} \; (\|\mathrm{R})$$

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \; (+\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'} \; (+\mathrm{R})$$

$$\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x' \parallel !x} \; (!) \qquad \frac{}{a \xrightarrow{a} \checkmark} \; (\textsc{AtomAct})$$

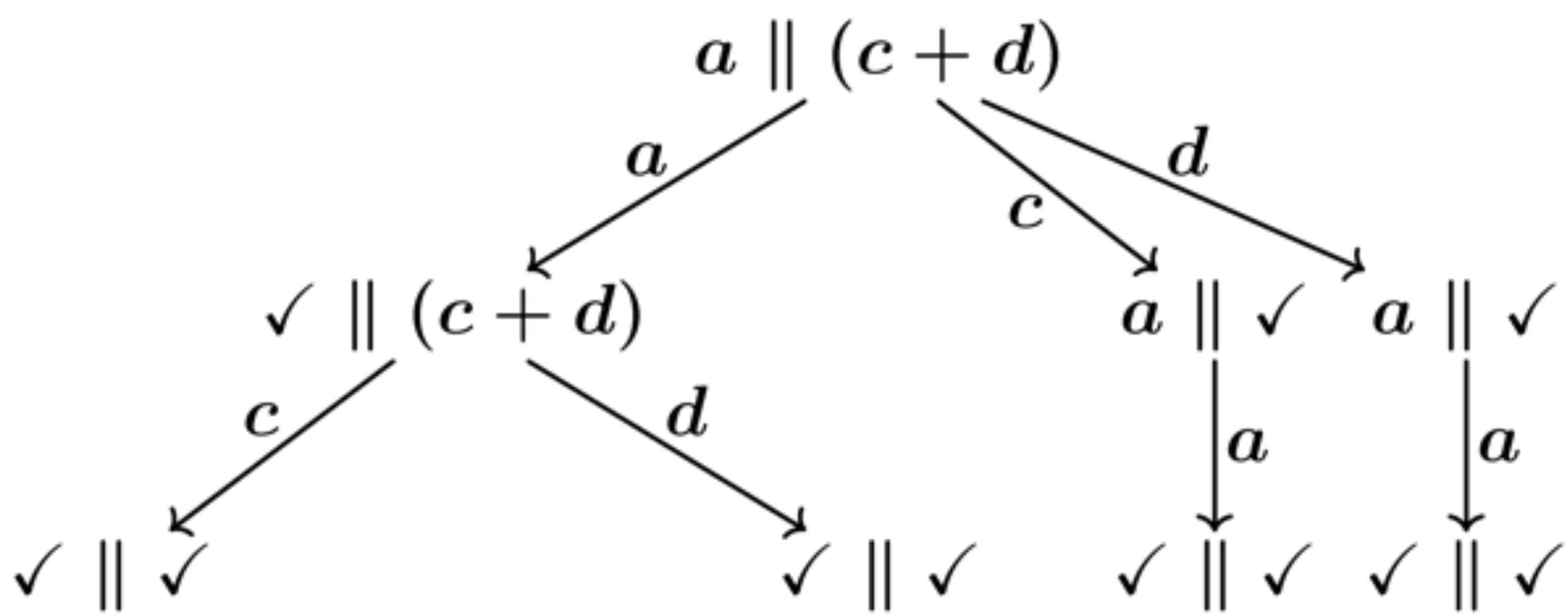
SOS = Structural Operational Semantics

- ... from which we derive transitions

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y}\ (\parallel\text{L}) \qquad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'}\ (\parallel\text{R})$$

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'}\ (+\text{L}) \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}\ (+\text{R})$$

$$\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x' \parallel !x}\ (!) \qquad \frac{}{a \xrightarrow{a} \checkmark}\ (\text{AtomAct})$$



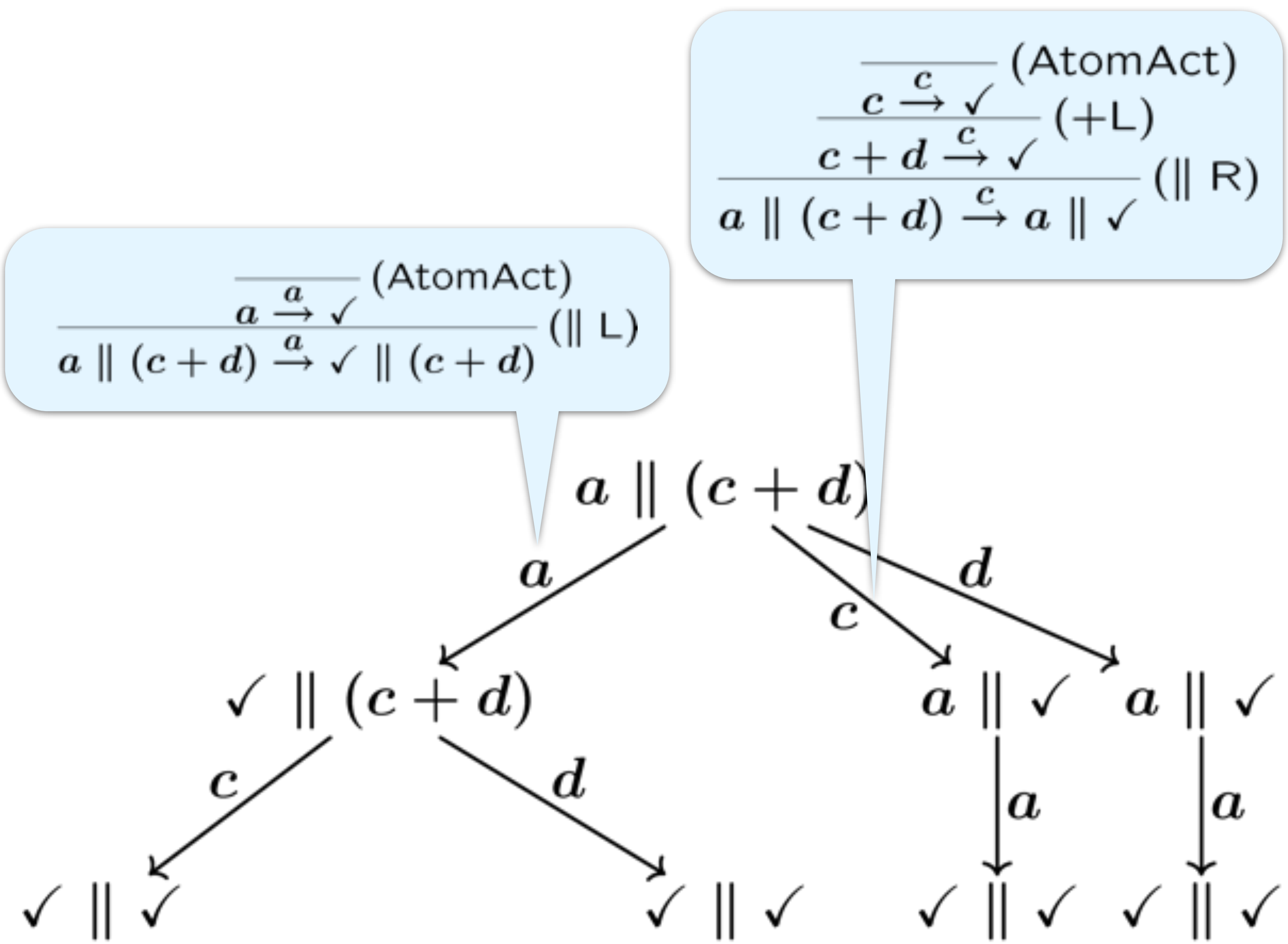
SOS = Structural Operational Semantics

- ... from which we derive transitions

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y}\ (\|\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'}\ (\|\mathrm{R})$$

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'}\ (+\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}\ (+\mathrm{R})$$

$$\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x' \parallel !x}\ (!) \qquad \frac{}{a \xrightarrow{a} \checkmark}\ (\textsc{AtomAct})$$



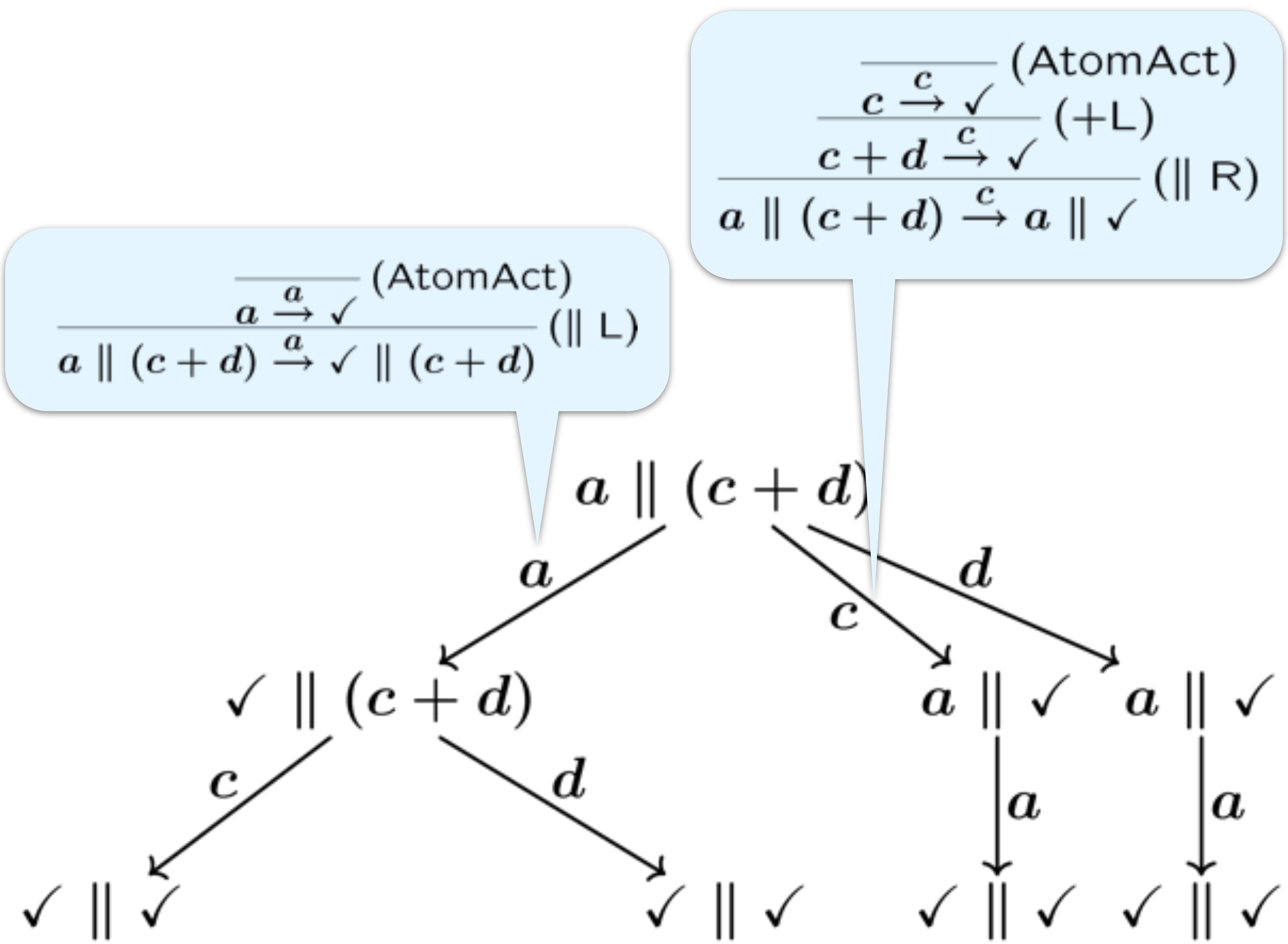
SOS = Structural Operational Semantics

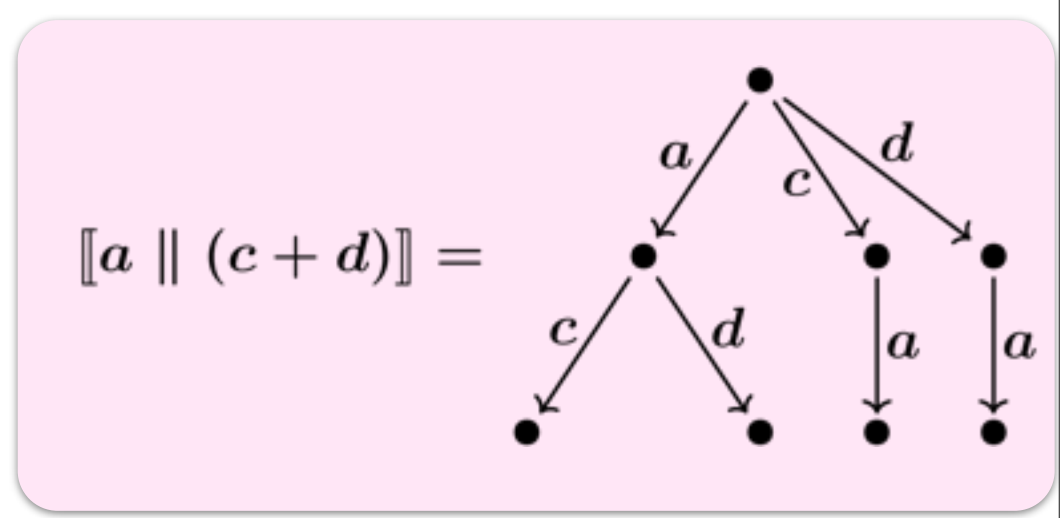
- ... from which we derive transitions

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y}\ (\|\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'}\ (\|\mathrm{R})$$

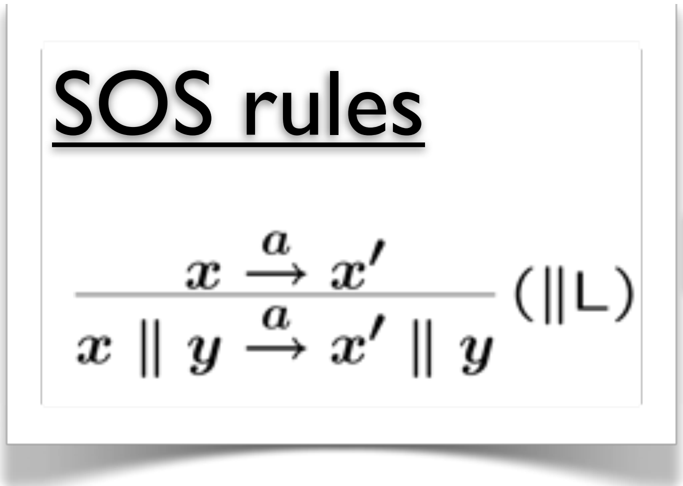
$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'}\ (+\mathrm{L}) \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}\ (+\mathrm{R})$$

$$\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x' \parallel !x}\ (!) \qquad \frac{}{a \xrightarrow{a} \checkmark}\ (\textsc{AtomAct})$$

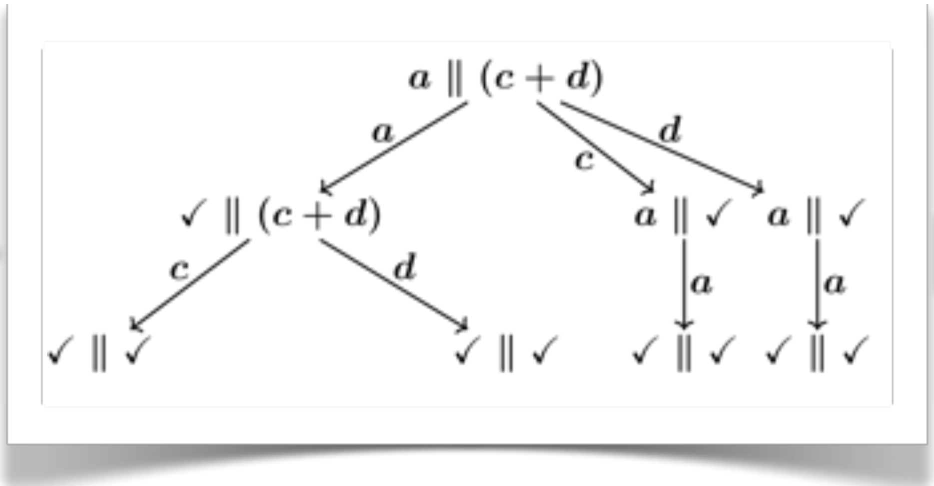




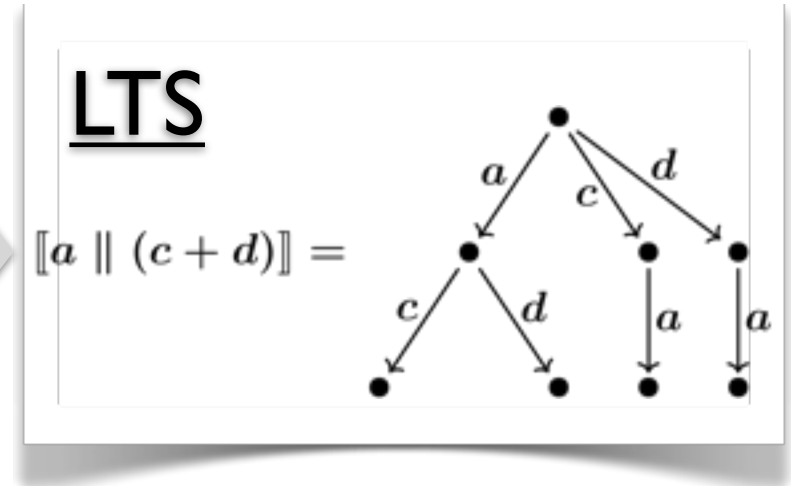
SOS, Categorically



derives



abstracted

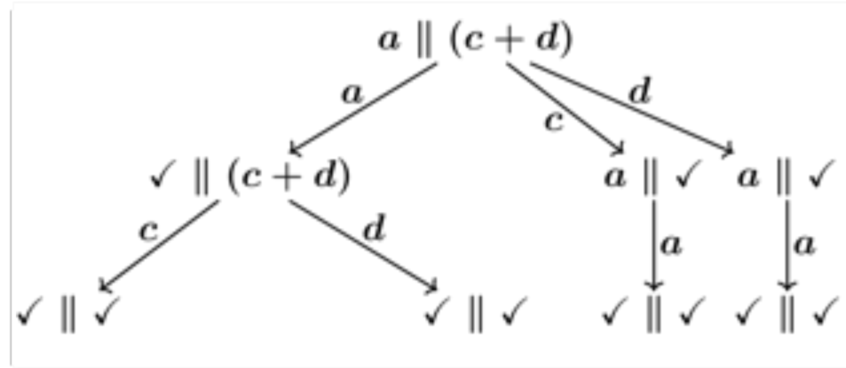


SOS, Categorically

SOS rules

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y}\ (\parallel\text{L})$$

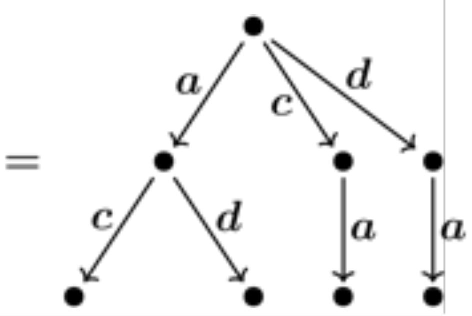
derives



abstracted

LTS

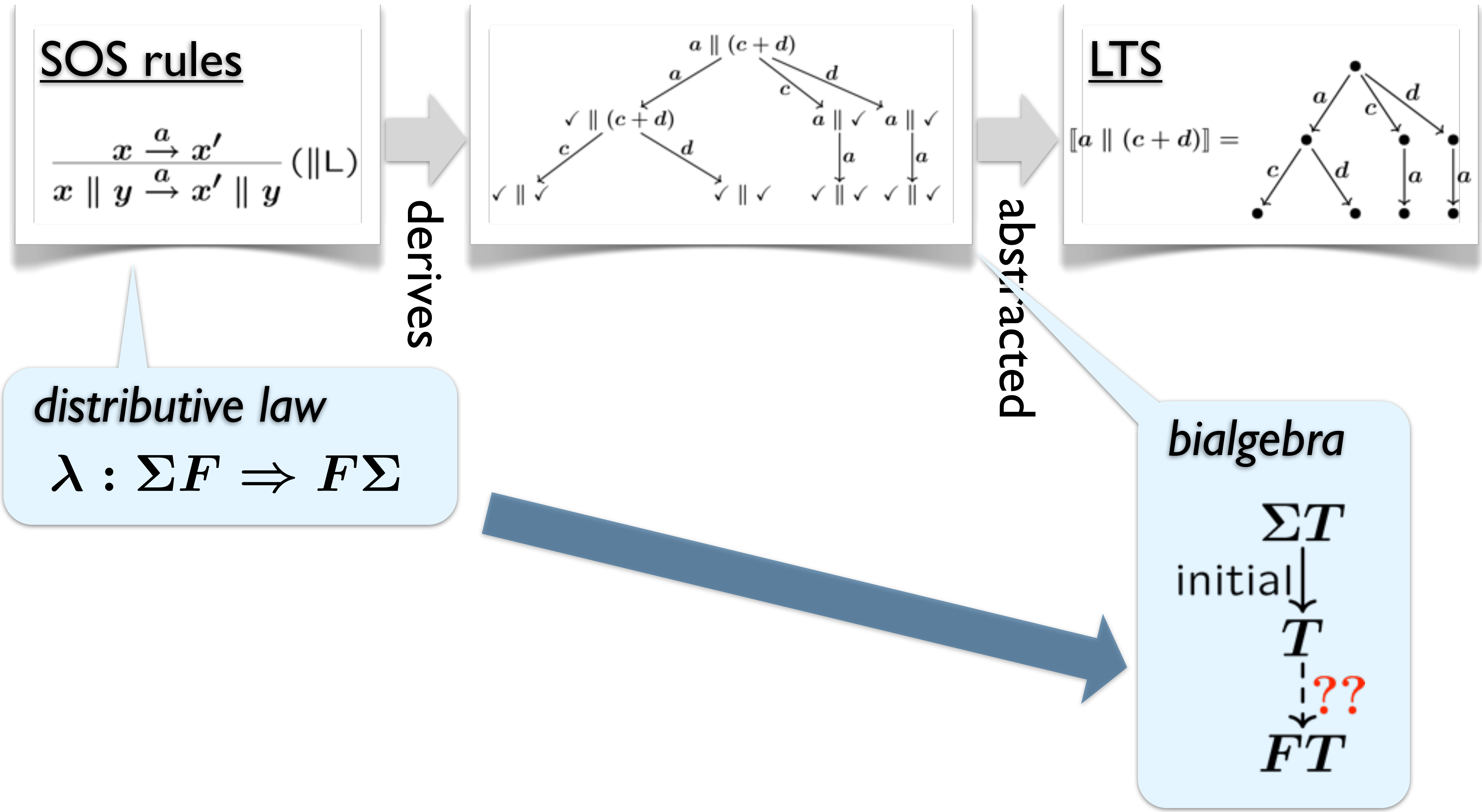
$[\![a \parallel (c+d)]\!] =$



distributive law

$$\lambda : \Sigma F \Rightarrow F\Sigma$$

SOS, Categorically

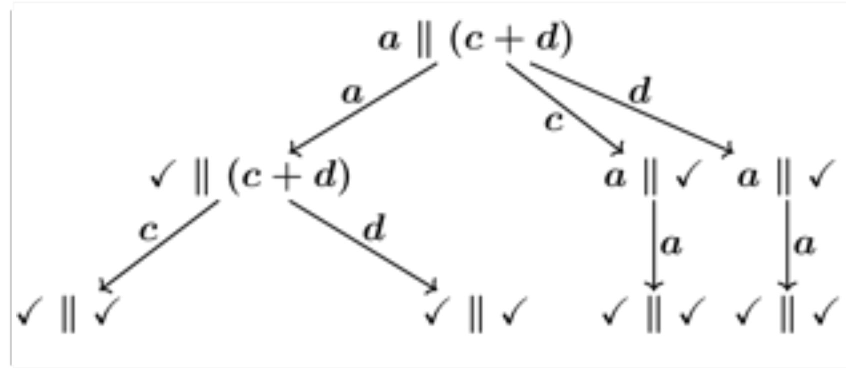


SOS, Categorically

SOS rules

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \ (\parallel L)$$

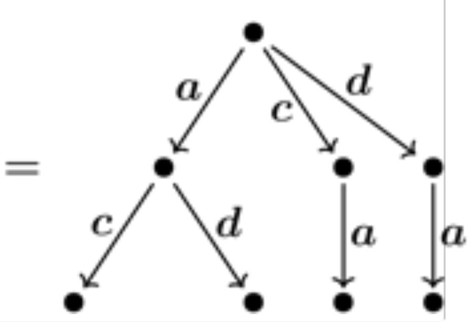
derives



abstracted

LTS

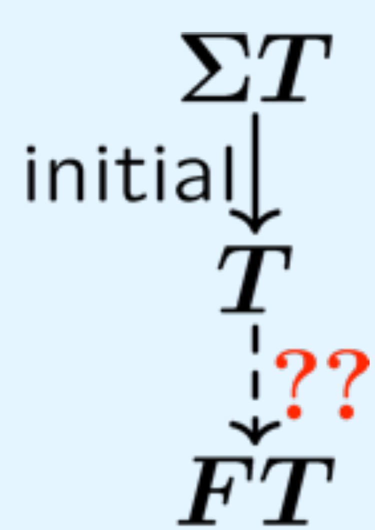
$[\![a \parallel (c+d)]\!] =$



distributive law

$\lambda : \Sigma F \Rightarrow F\Sigma$

bialgebra



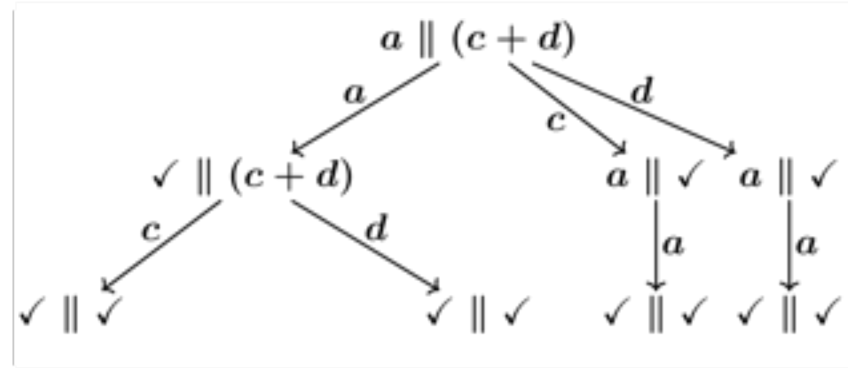
$T \dashrightarrow FT$

SOS, Categorically

SOS rules

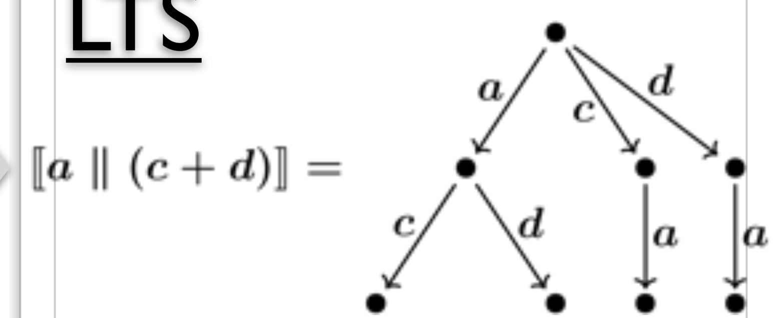
$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \;(\parallel\text{L})$$

derives



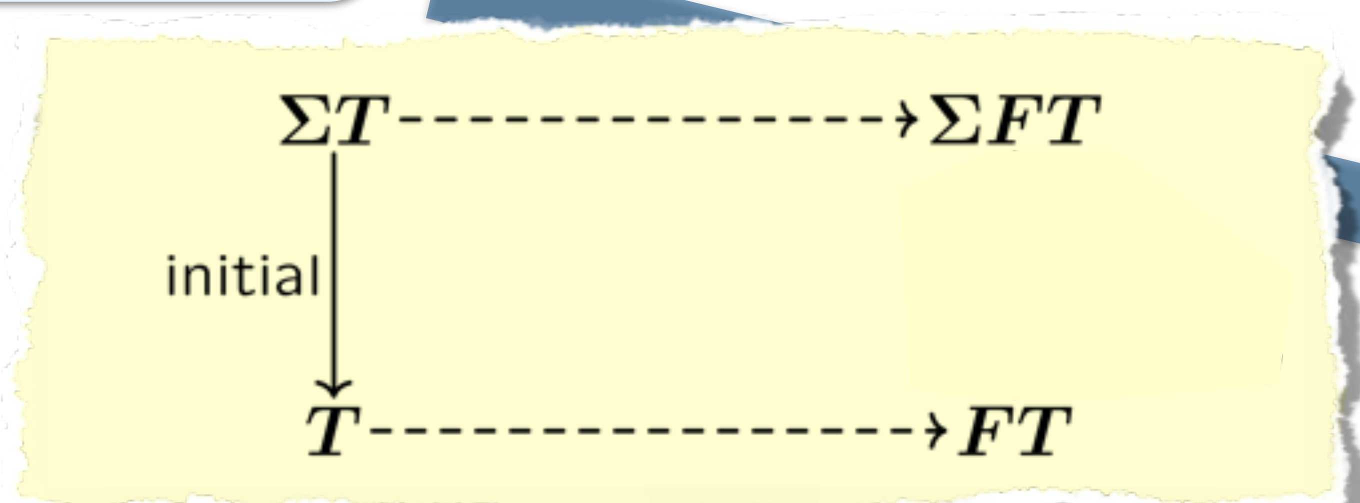
abstracted

LTS

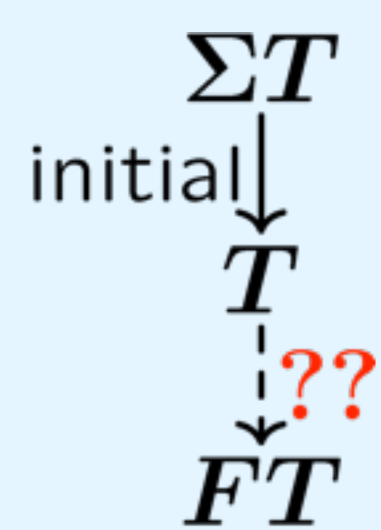


distributive law

$\lambda : \Sigma F \Rightarrow F\Sigma$



bialgebra

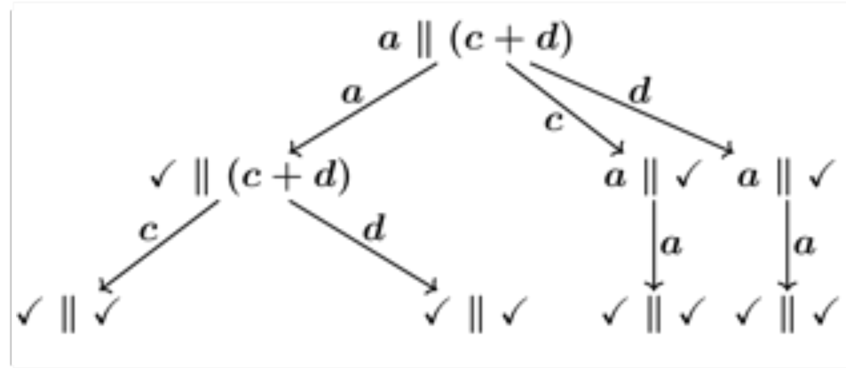


SOS, Categorically

SOS rules

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \; (\| \mathsf{L})$$

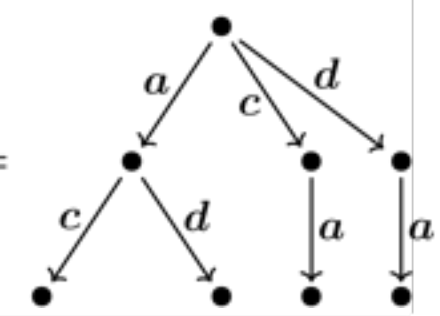
derives



abstracted

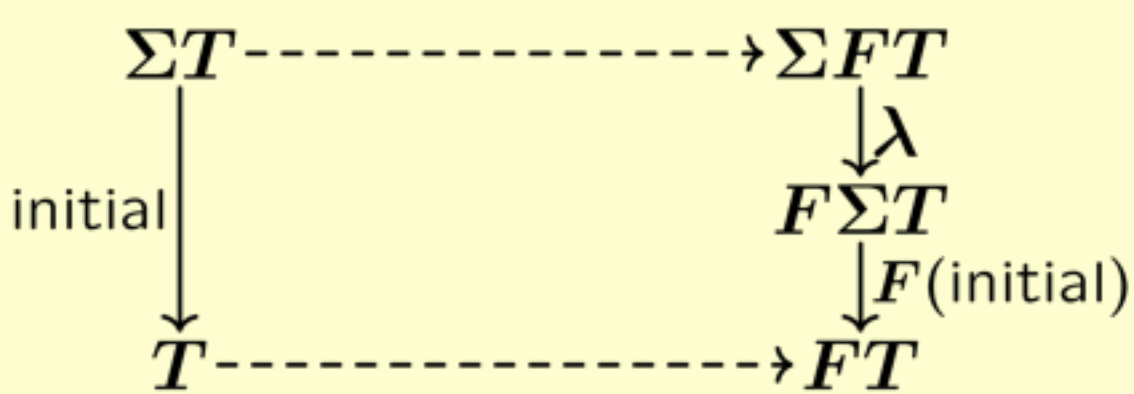
LTS

$[\![a \parallel (c+d)]\!] =$

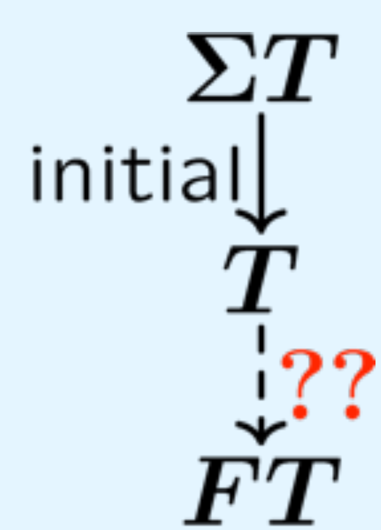


distributive law

$$\lambda : \Sigma F \Rightarrow F\Sigma$$



bialgebra



SOS, Categorically

program = algebra

system = coalgebra

SOS, Categorically

program = algebra

SOS

Structural Operational Semantics

system = coalgebra

Plotkin, Rutten, Turi, ...

SOS, Categorically

program = algebra

SOS

Structural Operational Semantics

bialgebra

system = coalgebra

Plotkin, Rutten, Turi, ...

SOS, Categorically

program = algebra

SOS
Structural Operational Semantics

bialgebra

system = coalgebra

- Probabilistic systems (Bartels, Kick-Power-Simpson, ...)
- Combined with modal logic (Klin)
- π-calculus and name-passing calculir (Fiore-Staton, ...)
- *Microcosm* extension, component calculus (Hasuo-Heunen-Jacobs-Sokolova, ...)

Time to Wake Up!!

3 Coalgebraic Trace Semantics

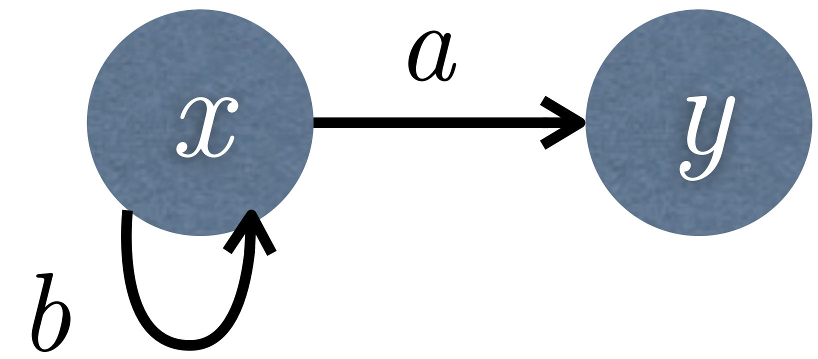
LTS

Definition. A *labeled transition system (LTS)* is a triple

$$(X, \Sigma, \{\xrightarrow{a}\}_{a \in \Sigma})$$

where

- X is a non-empty set of *states*;
- Σ is a non-empty set of *labels*;
- $\xrightarrow{a} \subseteq X \times X$ is a binary relation, for each $a \in \Sigma$.



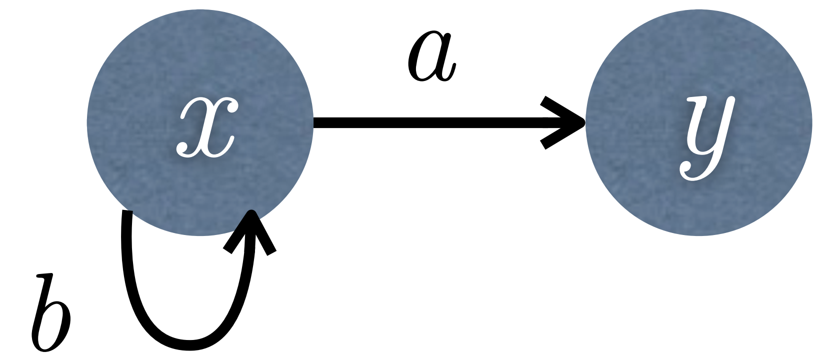
LTS

Definition. A *labeled transition system (LTS)* is a triple

$$(X, \Sigma, \{\xrightarrow{a}\}_{a\in\Sigma})$$

where

- X is a non-empty set of *states*;
- Σ is a non-empty set of *labels*;
- $\xrightarrow{a} \subseteq X \times X$ is a binary relation, for each $a \in \Sigma$.

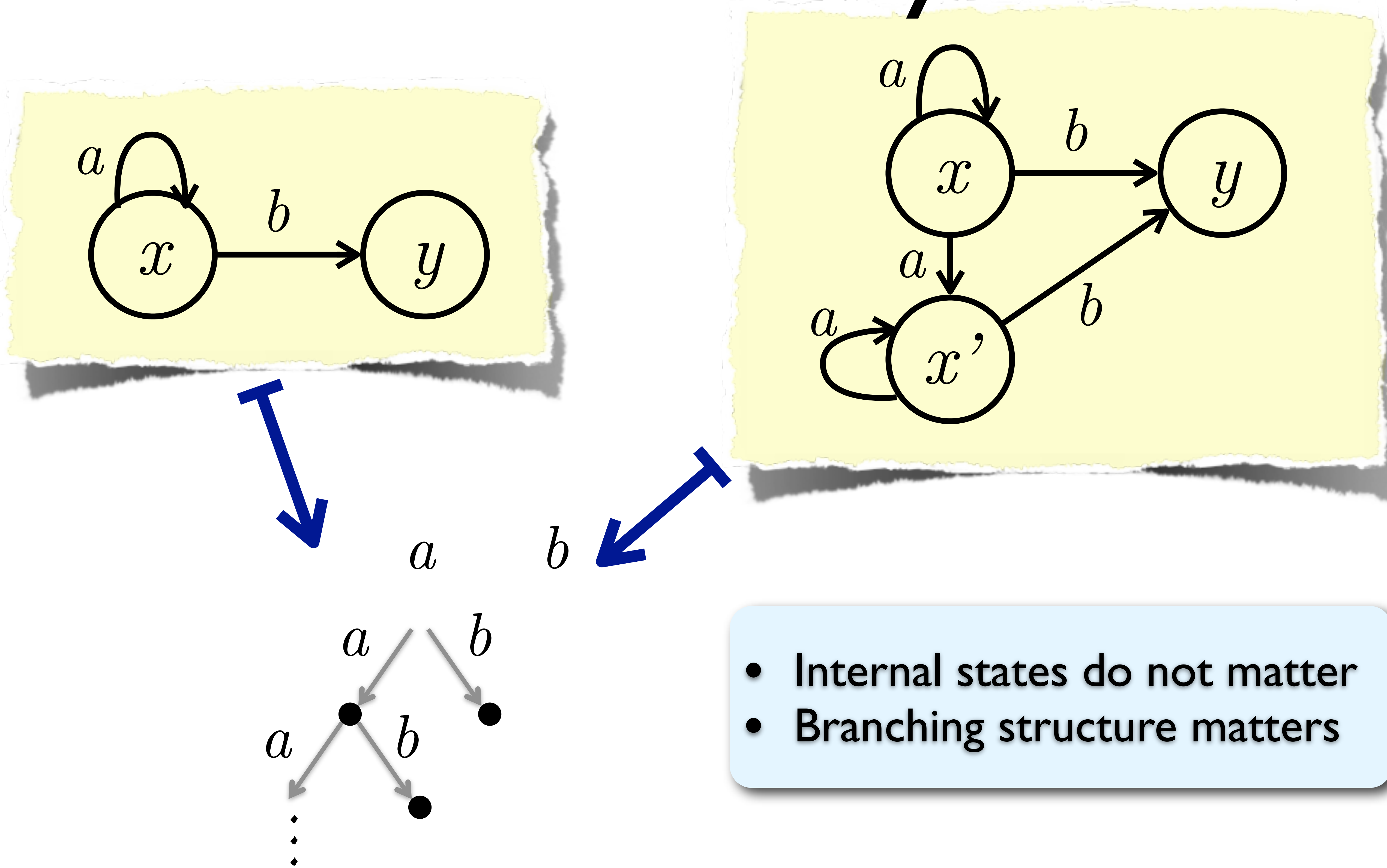


$$\mathcal{P}(\Sigma \times X) \qquad \{\, (a, x') \mid a \in \Sigma, x \xrightarrow{a} x' \,\}$$

$$\uparrow \qquad\qquad\qquad\qquad \uparrow$$

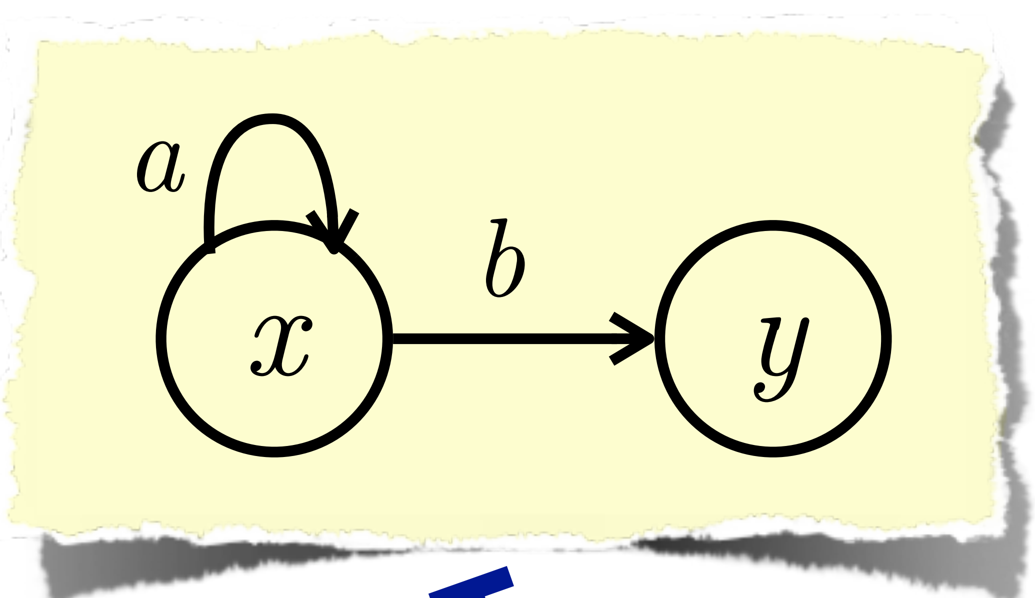
$$X \qquad\qquad\qquad\qquad x$$

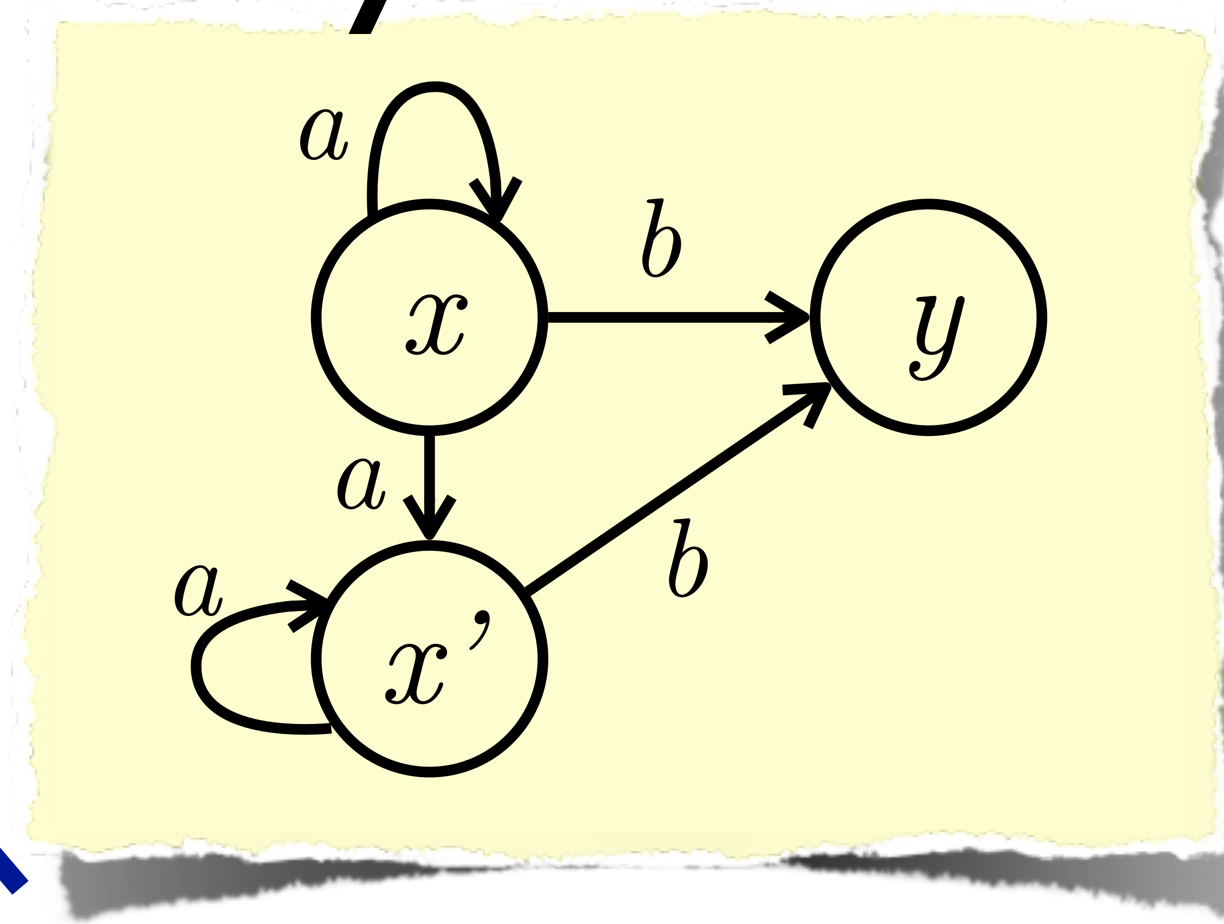
Bisimilarity

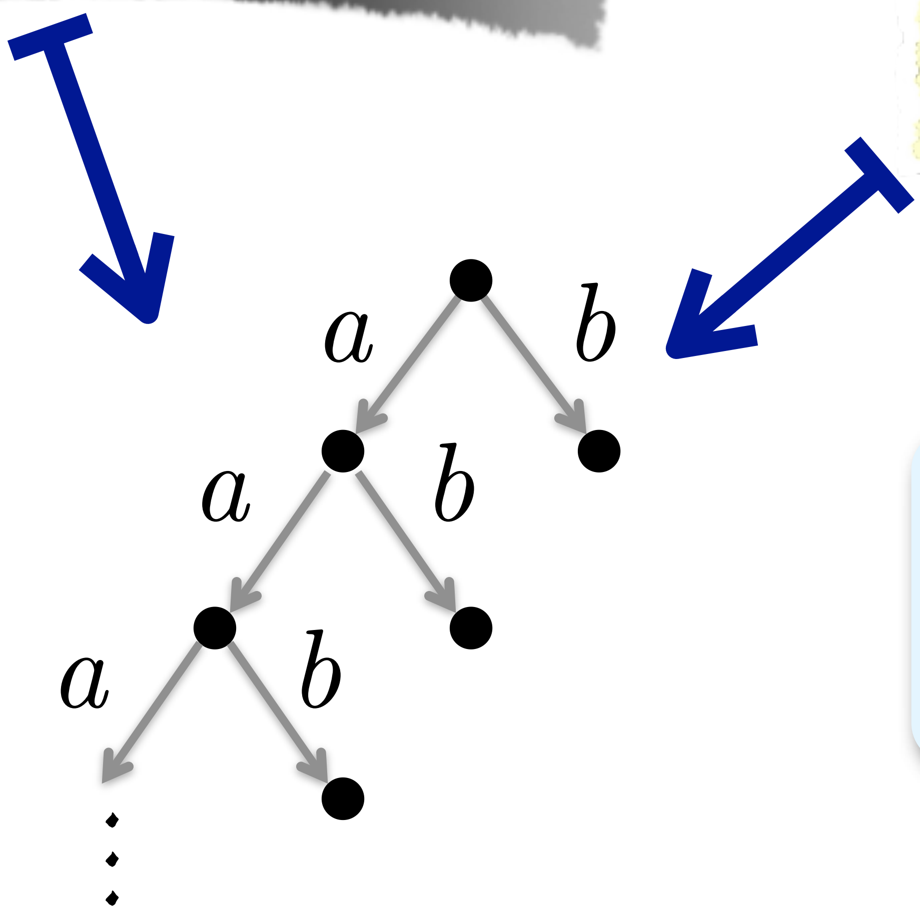


- Internal states do not matter
- Branching structure matters

Bisimilarity

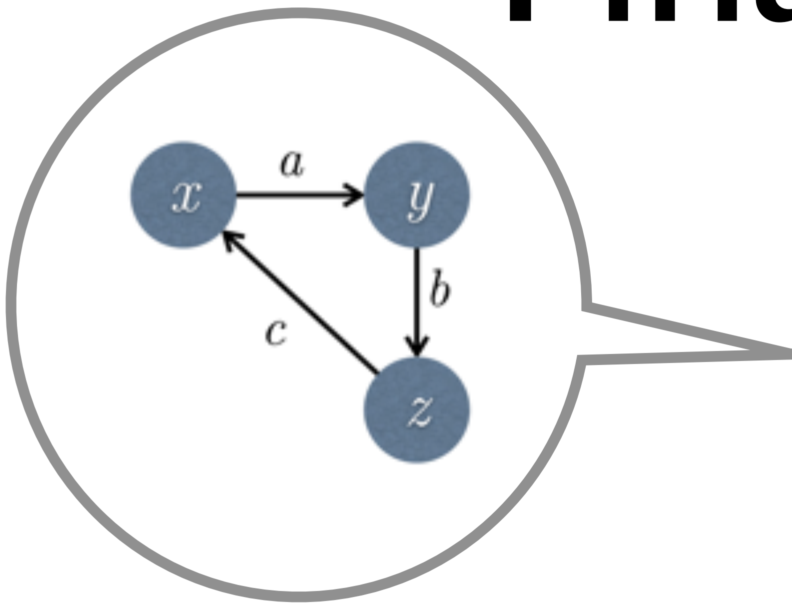






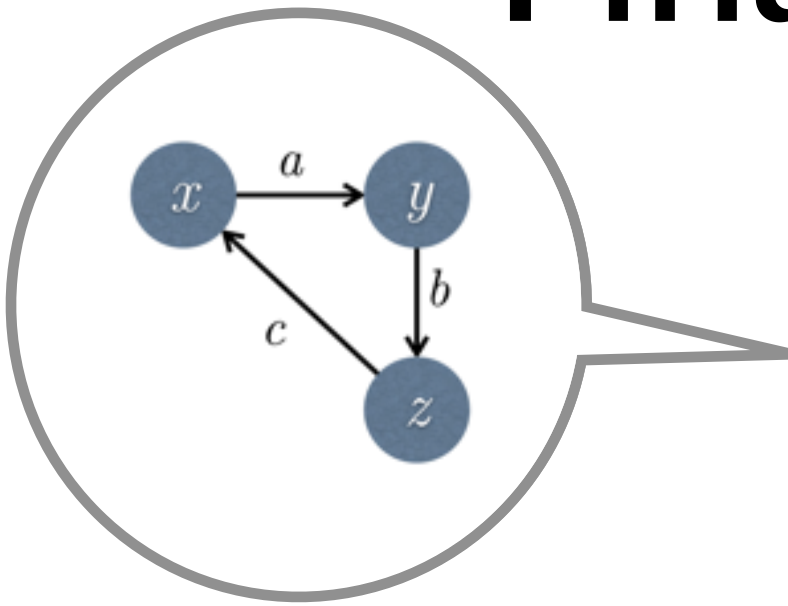
- Internal states do not matter
- Branching structure matters

Bisimilarity by Final Coalgebra

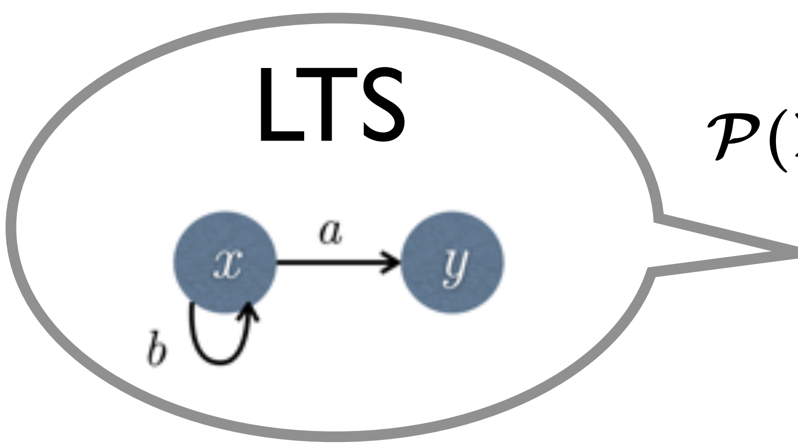


$$
\begin{array}{ccc}
\boldsymbol{\Sigma} \times \boldsymbol{X} & \xrightarrow{\;\boldsymbol{\Sigma} \times \mathbf{beh}(\boldsymbol{c})\;} & \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^{\mathbb{N}} \\
\boldsymbol{c} \uparrow & & \uparrow \text{final} \\
\boldsymbol{X} & \xrightarrow[\mathbf{beh}(\boldsymbol{c})]{} & \boldsymbol{\Sigma}^{\mathbb{N}} \\
\boldsymbol{x} & \longmapsto & \boldsymbol{abcabc\ldots}
\end{array}
$$

Bisimilarity by Final Coalgebra

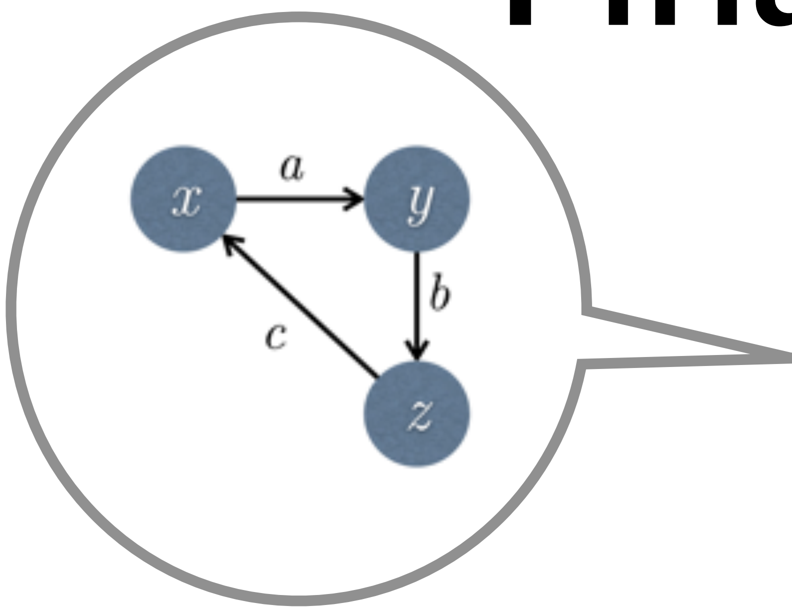


$$
\begin{array}{ccc}
\Sigma \times X & \xrightarrow{\ \Sigma \times \mathbf{beh}(c)\ } & \Sigma \times \Sigma^{\mathbb{N}} \\
c \uparrow & & \uparrow \text{final} \\
X & \xrightarrow[\ \mathbf{beh}(c)\ ]{} & \Sigma^{\mathbb{N}} \\
x & \longmapsto & abcabc\ldots
\end{array}
$$

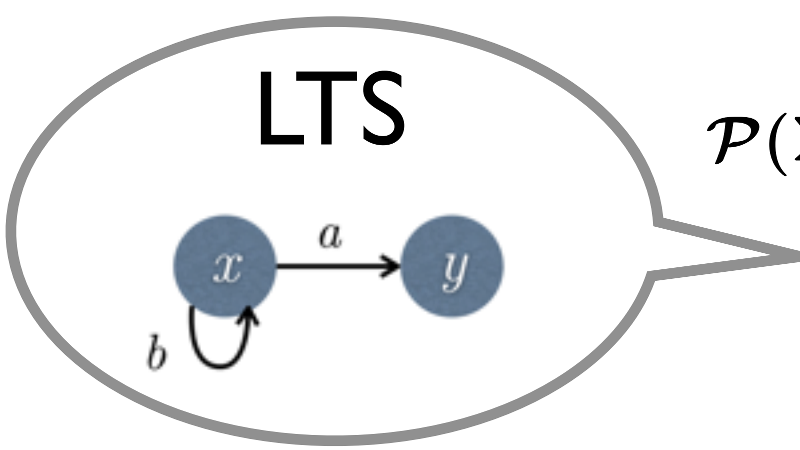


$$
\begin{array}{ccc}
\mathcal{P}(\Sigma \times X) & \xrightarrow{\ \mathcal{P}(\Sigma \times \mathbf{beh}(c))\ } & \mathcal{P}(\Sigma \times Z) \\
c \uparrow & & \uparrow \text{final} \\
X & \xrightarrow[\ \mathbf{beh}(c)\ ]{} & Z
\end{array}
$$

Bisimilarity by Final Coalgebra



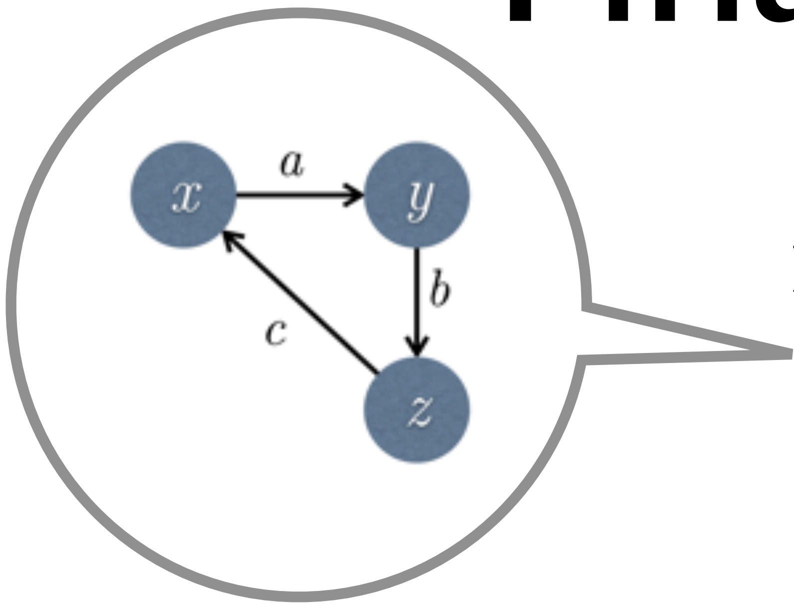
$$\begin{array}{ccc}
\Sigma \times X & \xrightarrow{\Sigma \times \mathbf{beh}(c)} & \Sigma \times \Sigma^{\mathbb{N}} \\
c \uparrow & & \uparrow \text{final} \\
X & \xrightarrow[\mathbf{beh}(c)]{} & \Sigma^{\mathbb{N}} \\
x & \longmapsto & abcabc\ldots
\end{array}$$



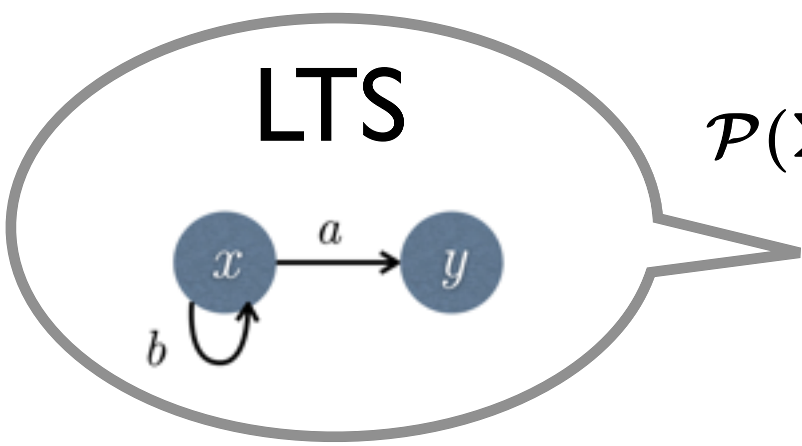
$$\begin{array}{ccc}
\mathcal{P}(\Sigma \times X) & \xrightarrow{\mathcal{P}(\Sigma \times \mathbf{beh}(c))} & \mathcal{P}(\Sigma \times Z) \\
c \uparrow & & \uparrow \text{final} \\
X & \xrightarrow[\mathbf{beh}(c)]{} & Z
\end{array}$$

= {bisimilarity classes}
= {synchronization trees}

Bisimilarity by Final Coalgebra



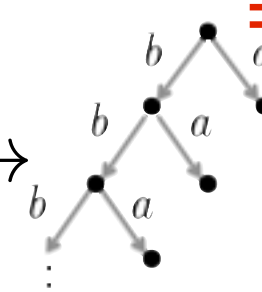
$$\begin{array}{ccc} \Sigma \times X & \xrightarrow{\Sigma \times \mathsf{beh}(c)} & \Sigma \times \Sigma^{\mathbb{N}} \\ c \uparrow & & \uparrow \text{final} \\ X & \xrightarrow[\mathsf{beh}(c)]{} & \Sigma^{\mathbb{N}} \\ x & \longmapsto & abcabc\ldots \end{array}$$



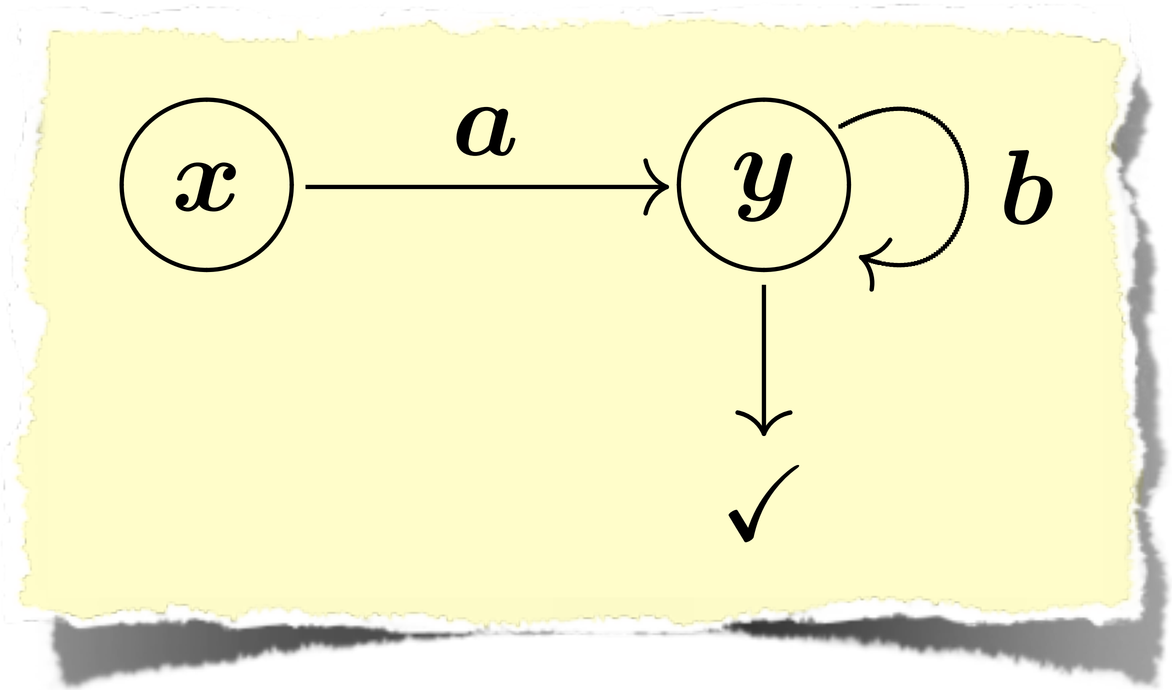
$$\begin{array}{ccc} \mathcal{P}(\Sigma \times X) & \xrightarrow{\mathcal{P}(\Sigma \times \mathsf{beh}(c))} & \mathcal{P}(\Sigma \times Z) \\ c \uparrow & & \uparrow \text{final} \\ X & \xrightarrow[\mathsf{beh}(c)]{} & Z \\ x & \longmapsto & \end{array}$$

Z = {bisimilarity classes}

= {synchronization trees}

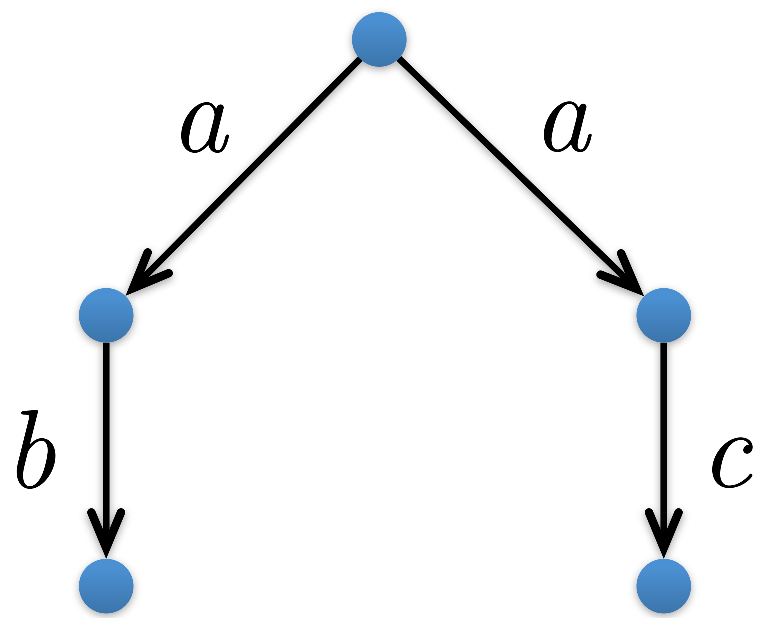


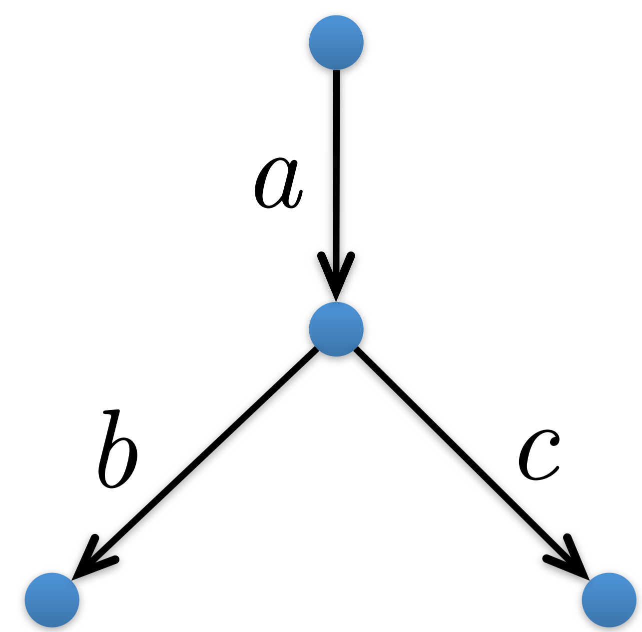
Trace Semantics



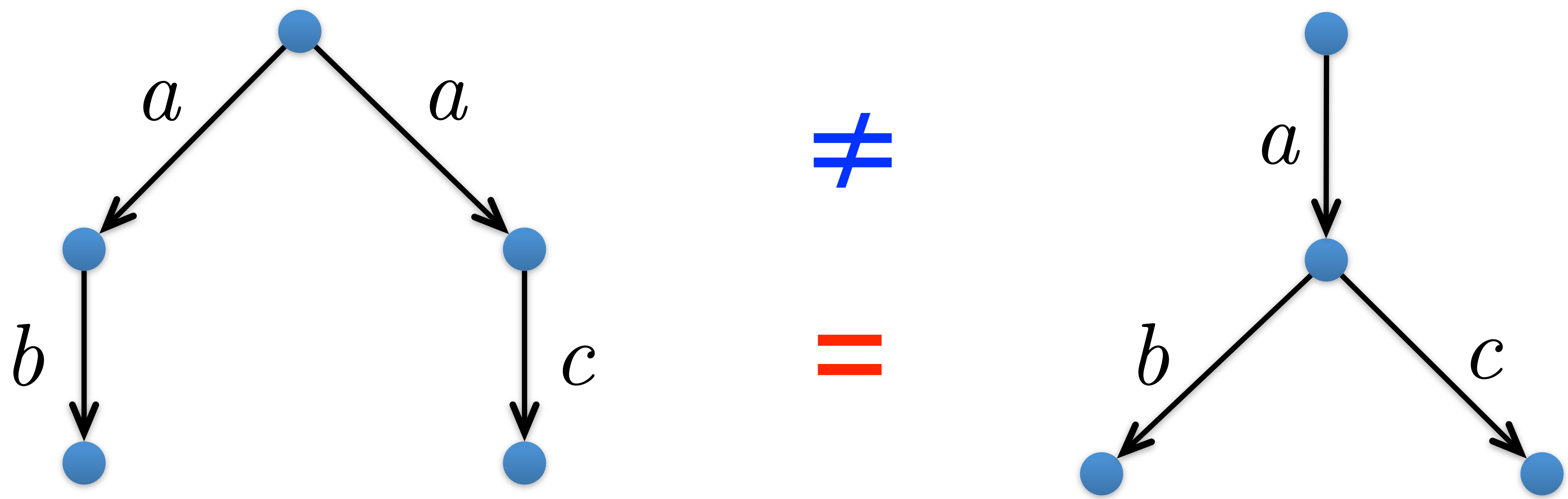
$$\mathbf{tr}(x) = \{a, ab, abb, \ldots\} = ab^*$$

Bisimilarity vs. Trace Semantics

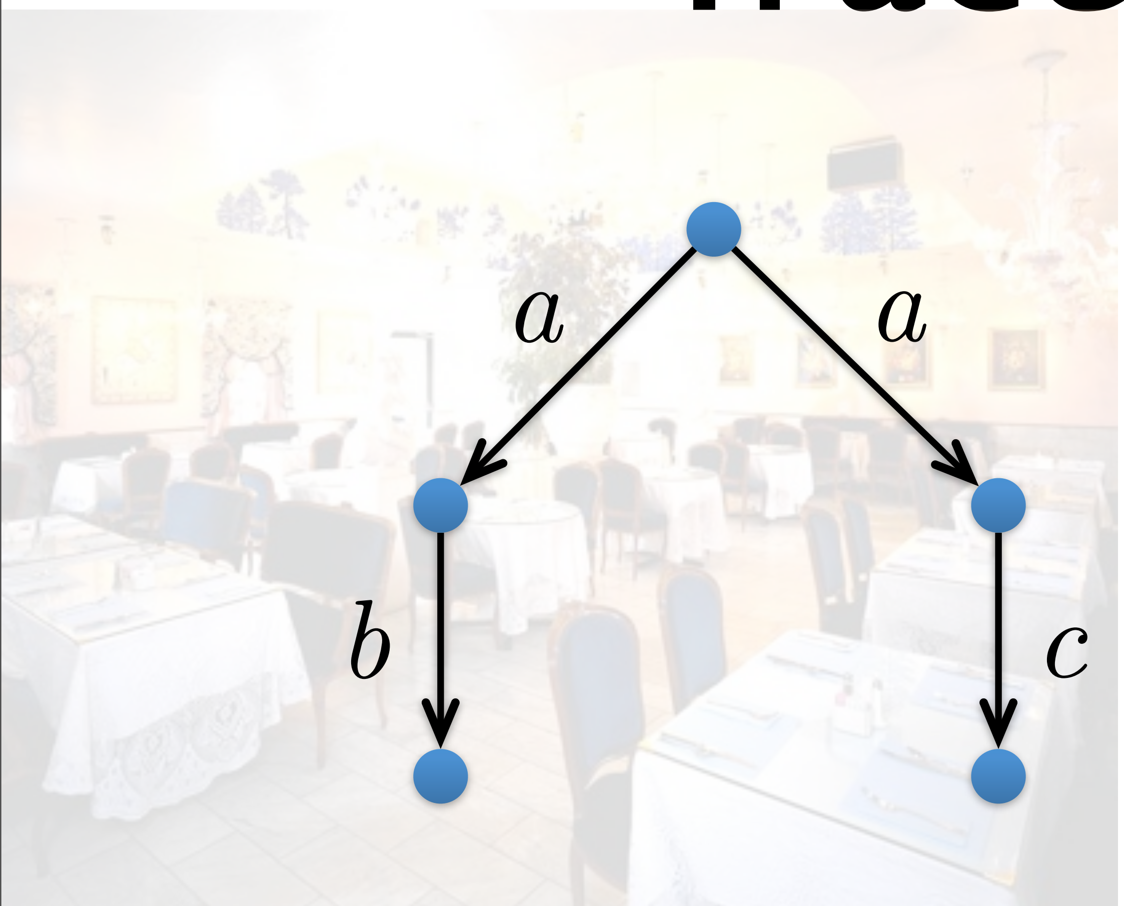




Bisimilarity vs. Trace Semantics

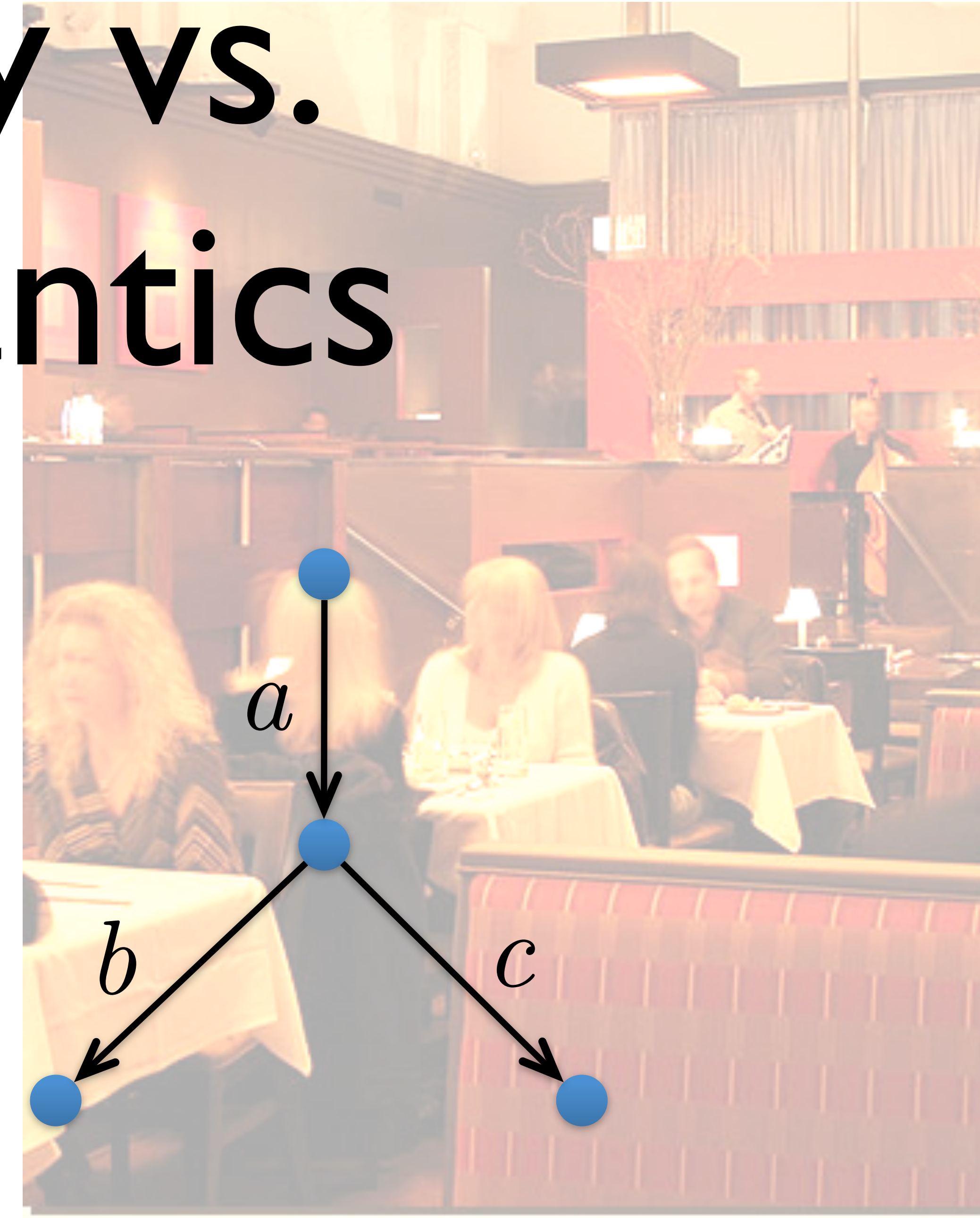


Bisimilarity vs. Trace Semantics

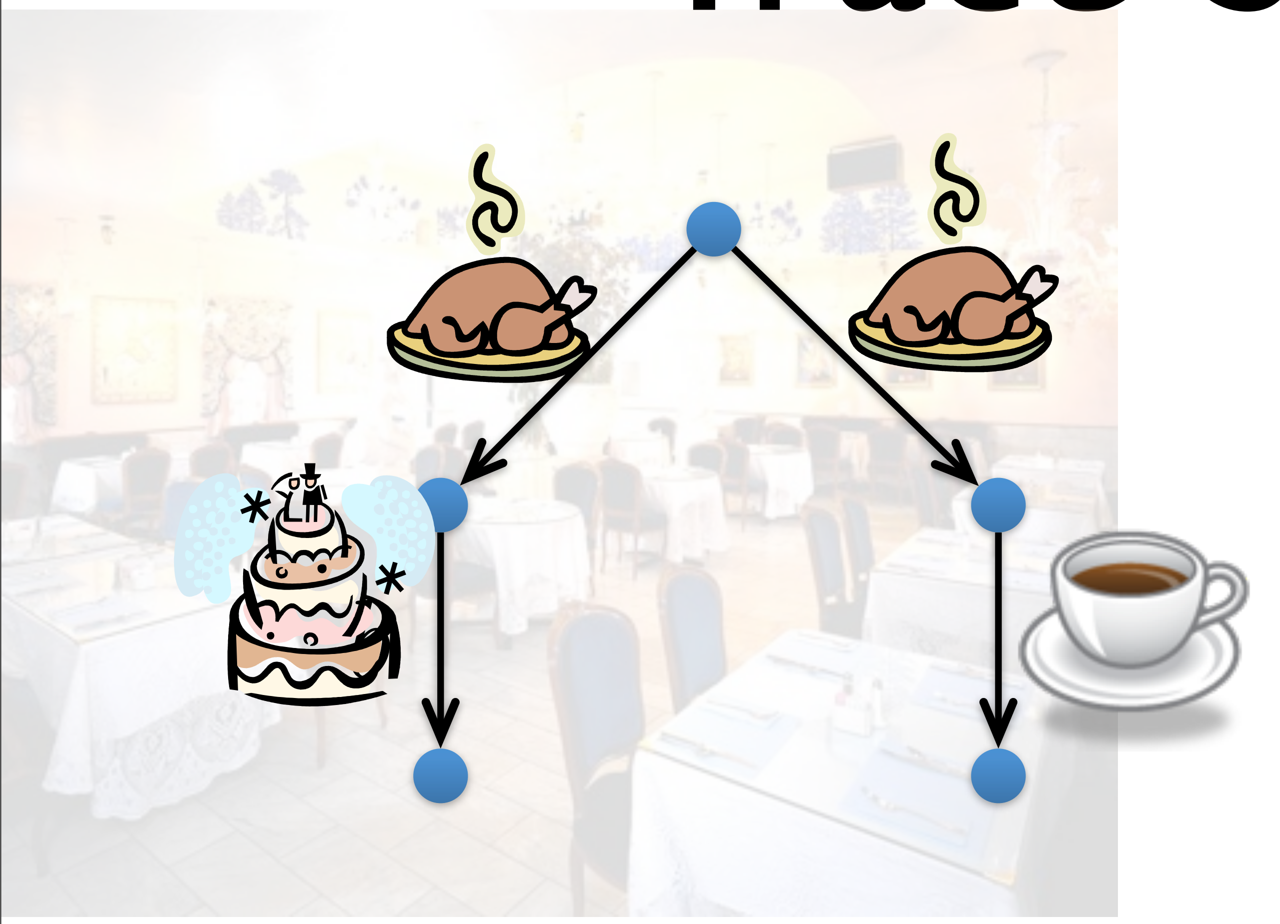


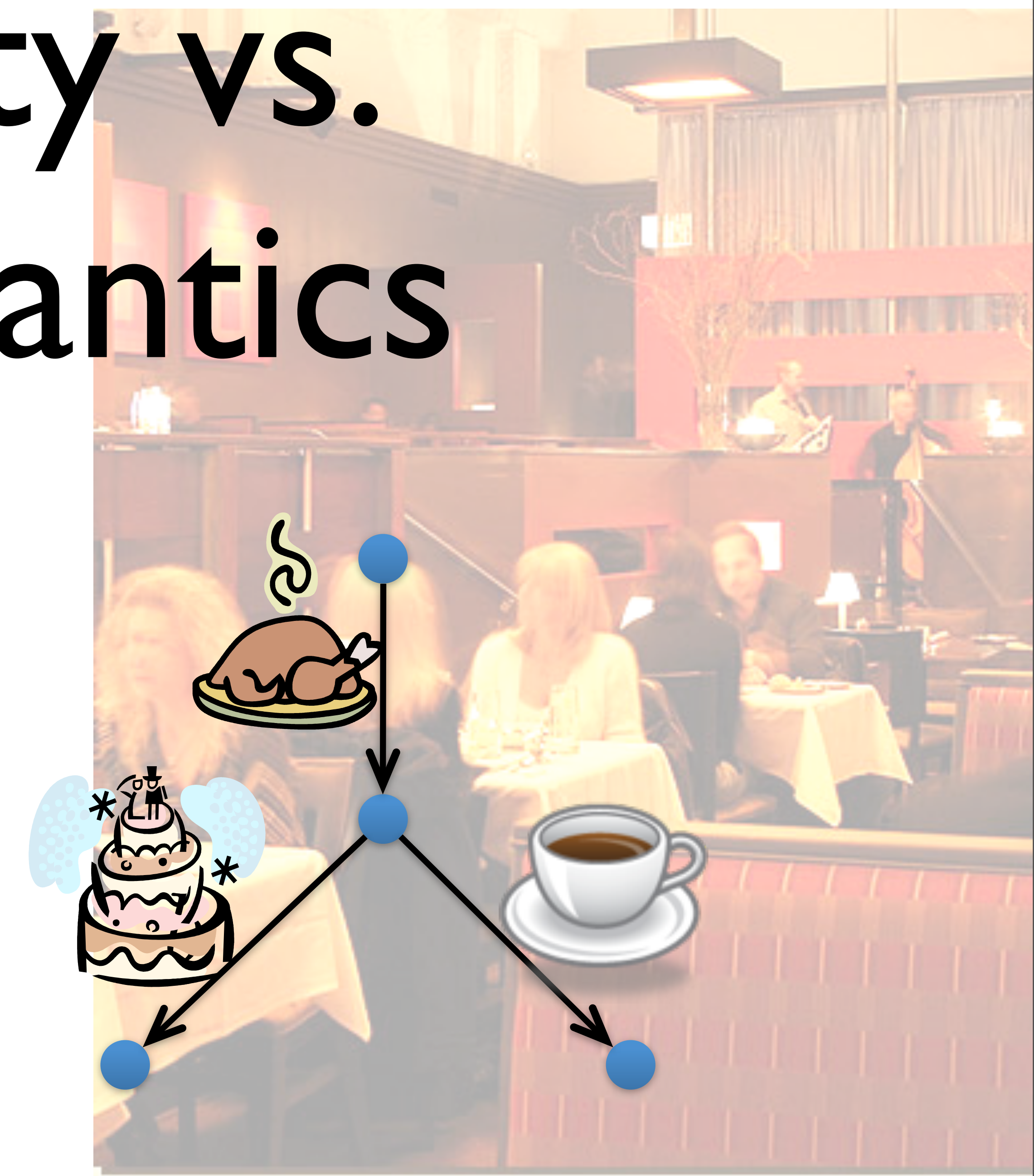
$\neq$

$=$



Bisimilarity vs. Trace Semantics





Bisimilarity

Branching structure matters.

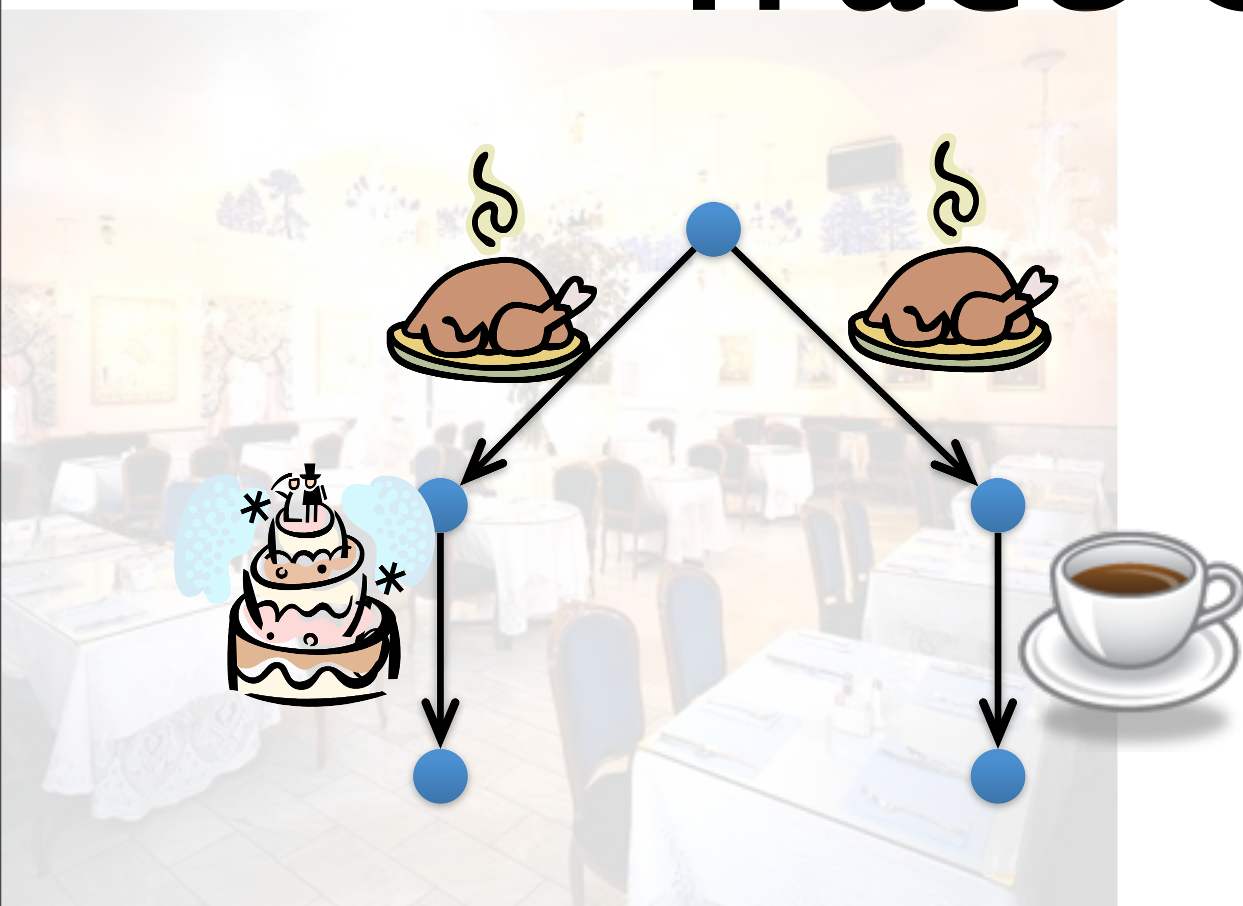
Can I choose later?

Trace semantics

Branching structure does not matter.

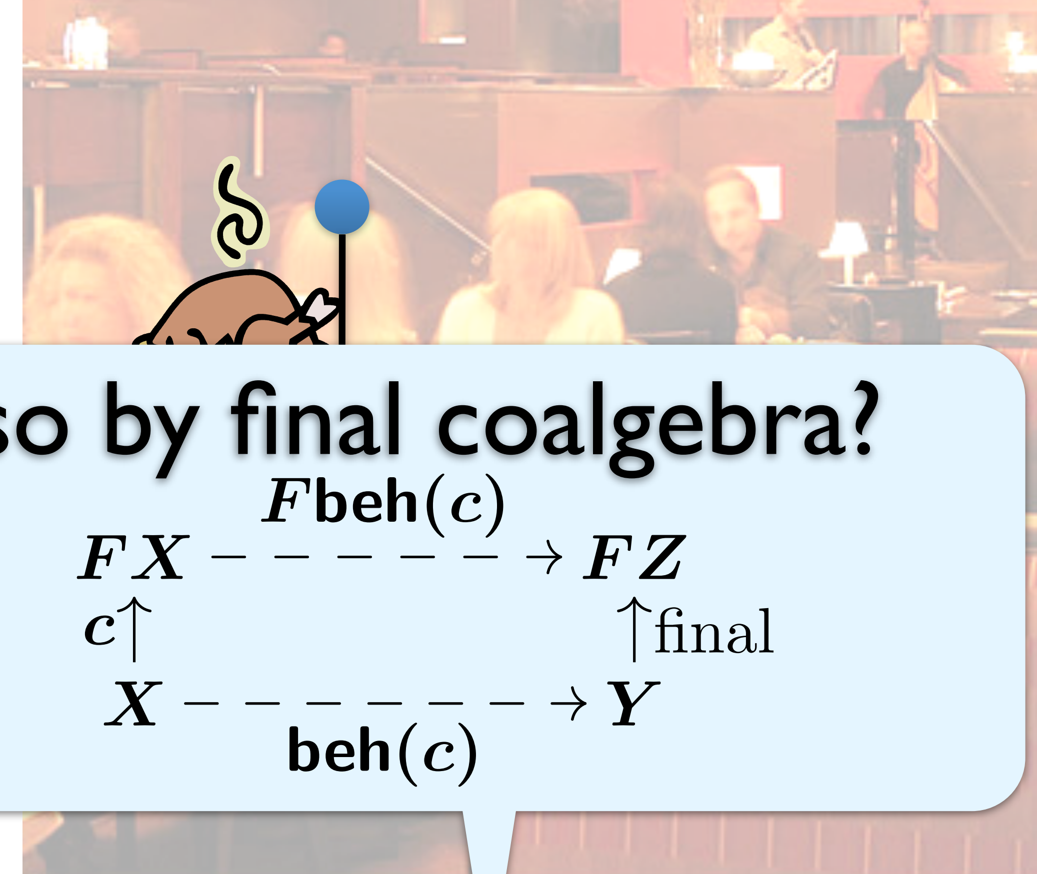
Anyway we'll get the same food.

Bisimilarity vs. Trace Semantics



$\neq$

$=$



Also by final coalgebra?

$$\begin{array}{ccc} FX & \overset{F\mathsf{beh}(c)}{\dashrightarrow} & FZ \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \mathrm{final}} \\ X & \underset{\mathsf{beh}(c)}{\dashrightarrow} & Y \end{array}$$

Bisimilarity

Branching structure matters.

Can I choose later?

Trace semantics

Branching structure does not matter.

Anyway we'll get the same food.

Coalgebraic Trace Semantics

- Yes!   By moving to the *Kleisli category*

$$\begin{array}{c} \mathcal{P}(\Sigma \times \boldsymbol{X}) \\ \uparrow \\ \boldsymbol{X} \end{array} \qquad \text{in } \mathbf{Sets}$$

Coalgebraic Trace Semantics

- Yes!   By moving to the *Kleisli category*

branching

$$\begin{array}{c} \mathcal{P}(\Sigma \times \boldsymbol{X}) \\ \uparrow \\ \boldsymbol{X} \end{array} \quad \text{in } \mathbf{Sets}$$

Coalgebraic Trace Semantics

- Yes! By moving to the *Kleisli category*

branching

$$\begin{array}{c} \mathcal{P}(\Sigma \times X) \\ \uparrow \\ X \end{array} \quad \text{in } \mathbf{Sets}$$

=

$$\begin{array}{c} \Sigma \times X \\ \nrightarrow\!\!\!\!\!\uparrow \\ X \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

Coalgebraic Trace Semantics

- Yes! By moving to the *Kleisli category*

branching

$$\begin{array}{c} \mathcal{P}(\Sigma \times X) \\ \uparrow \\ X \end{array} \quad \text{in } \mathbf{Sets}$$

//

- non-det. branching is built-in.
  Throw under the rug

$$\begin{array}{c} \Sigma \times X \\ \uparrow \\ X \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in } \mathbf{Sets}}$$

"non-deterministic function"

Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

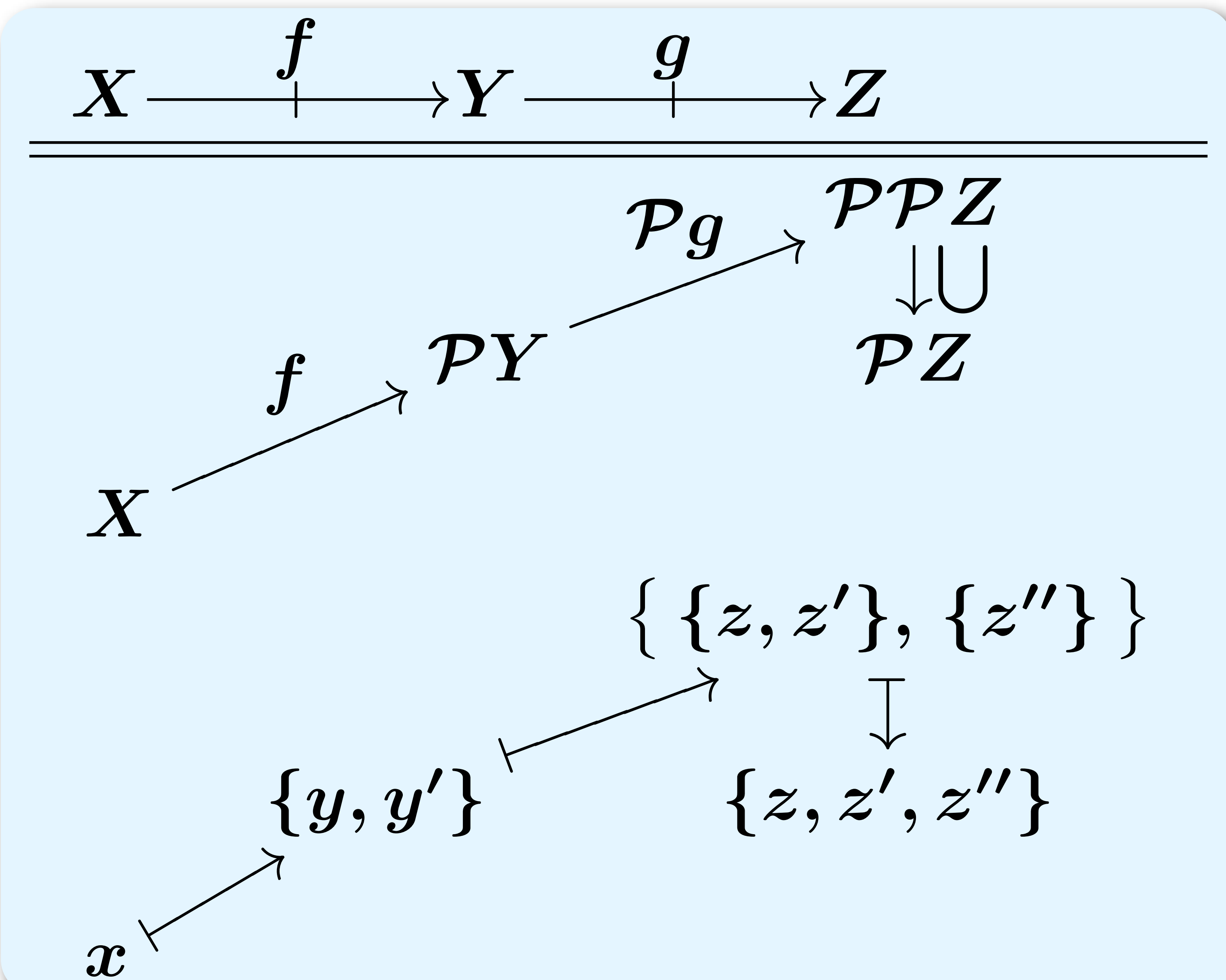
$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?

Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!\mid\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in } \mathbf{Sets}}$$

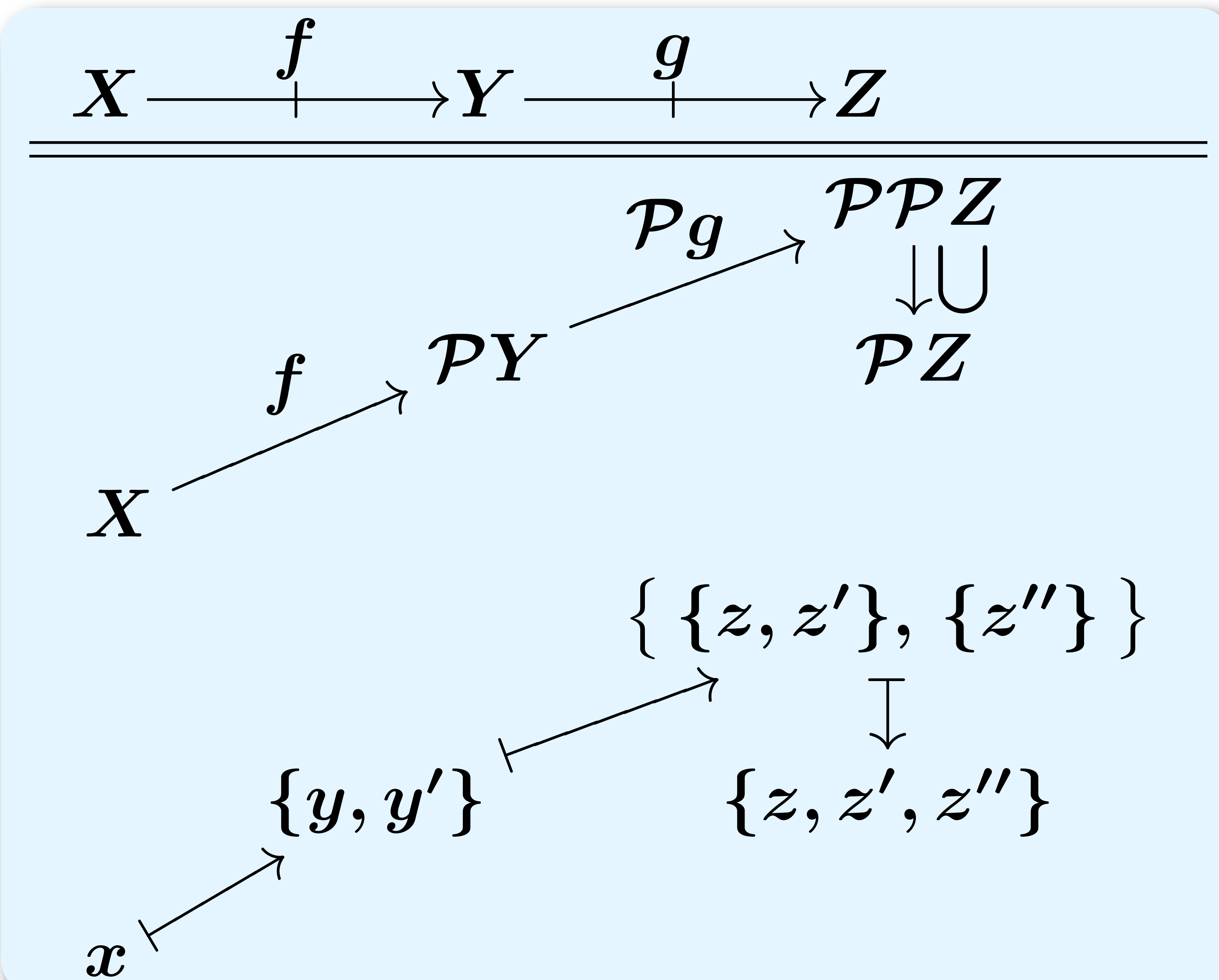
- Composition of arrows?



Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\dfrac{X \longrightarrow\!\!\!\!\!\!\!\!\!\!+\;\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in Sets}}$$

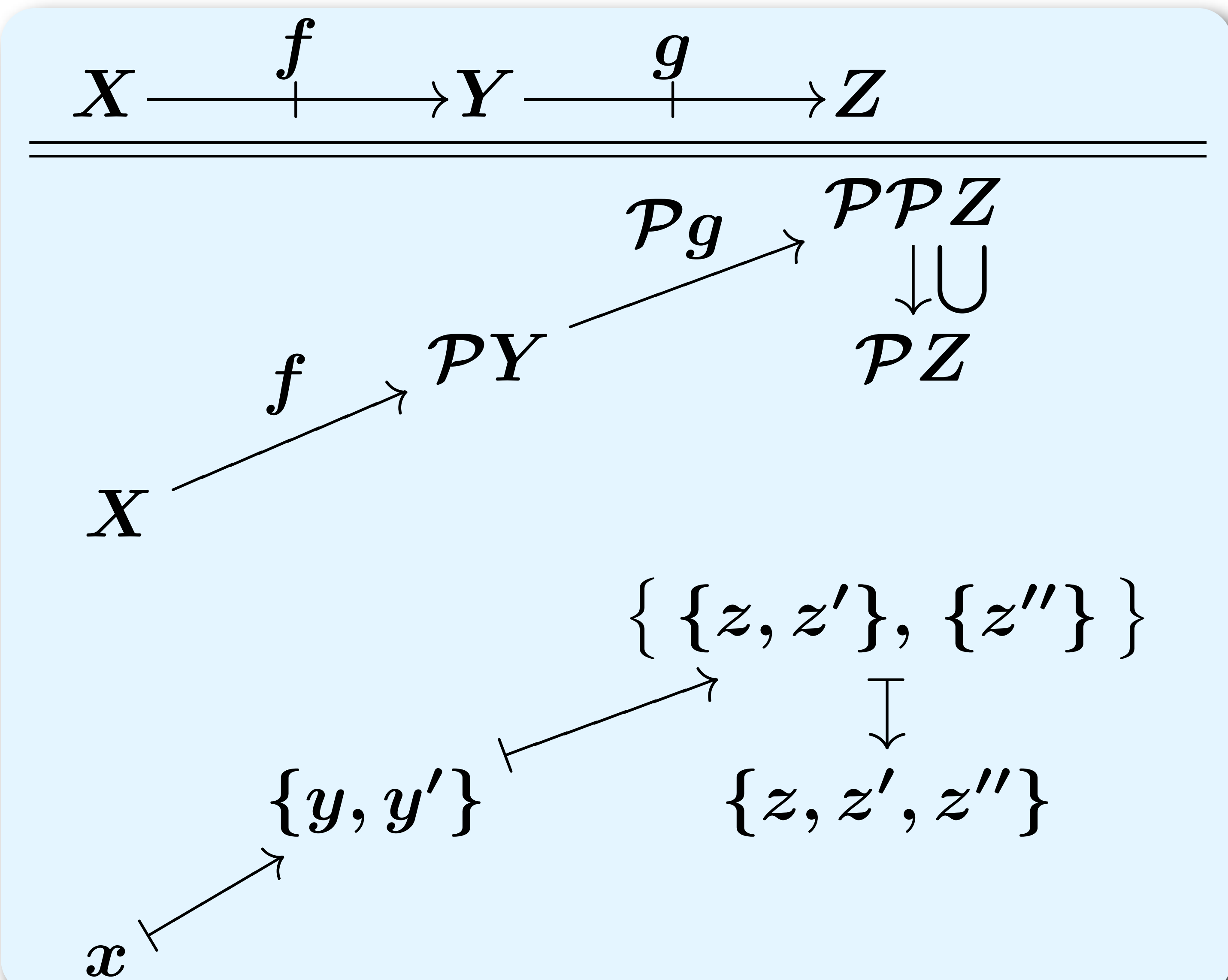
- Composition of arrows?

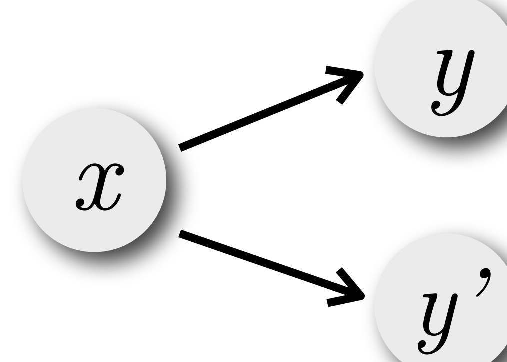


Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in Sets}}$$

- Composition of arrows?

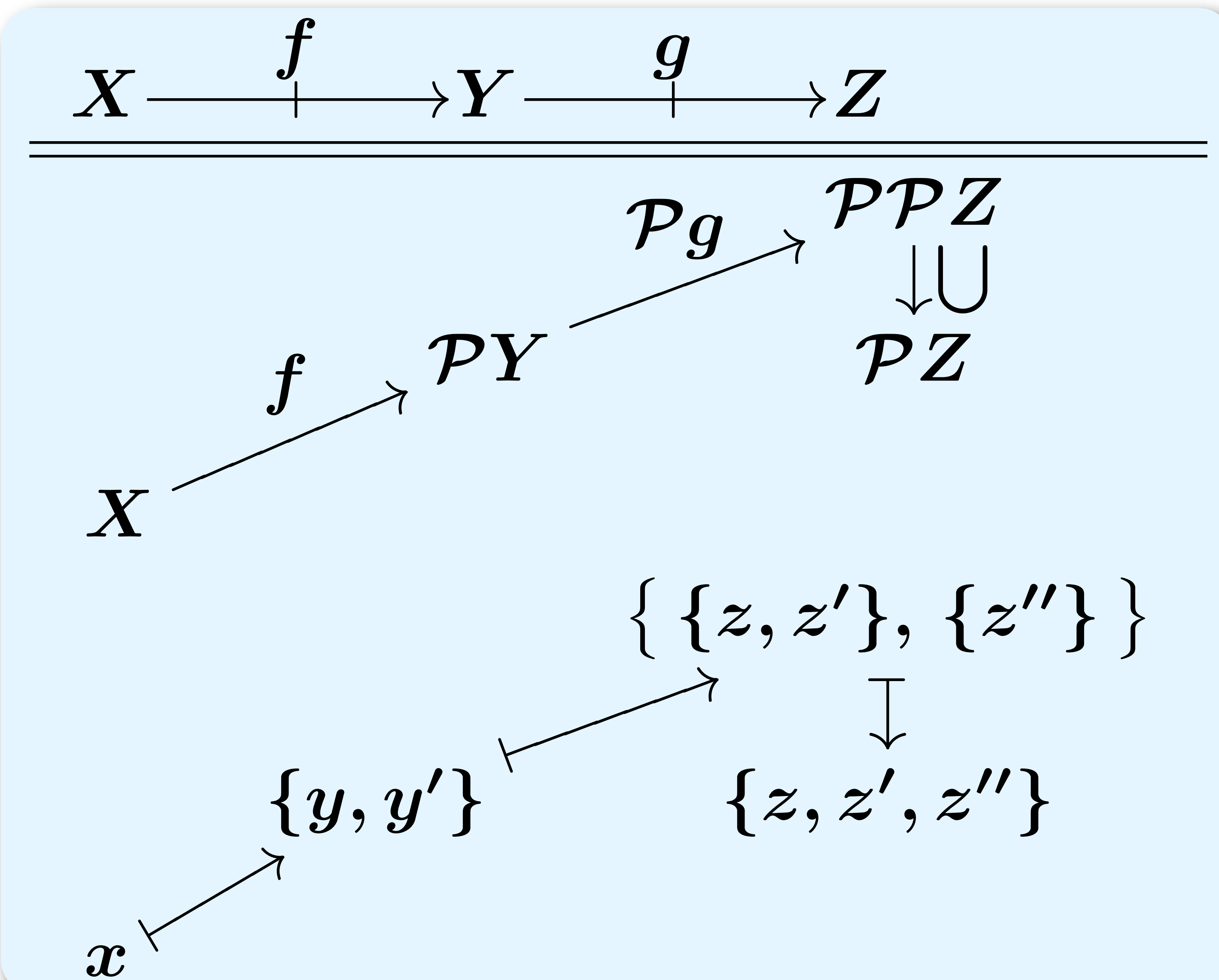


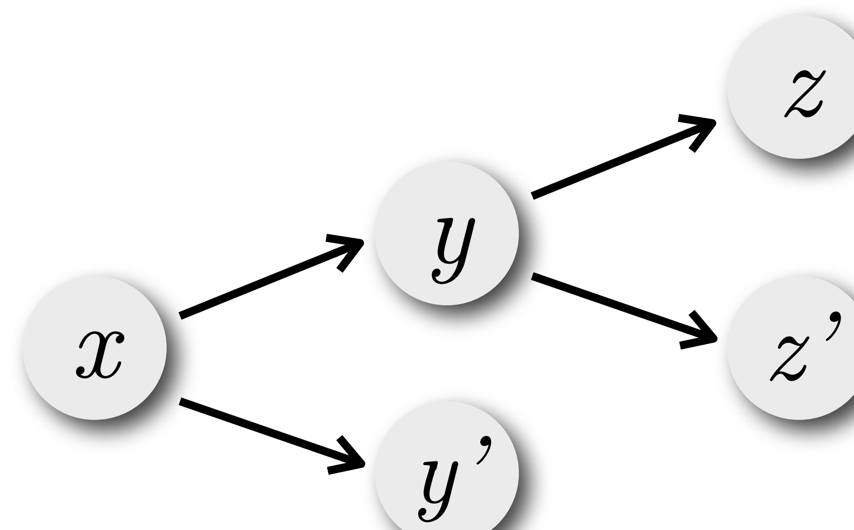


Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in Sets}}$$

- Composition of arrows?

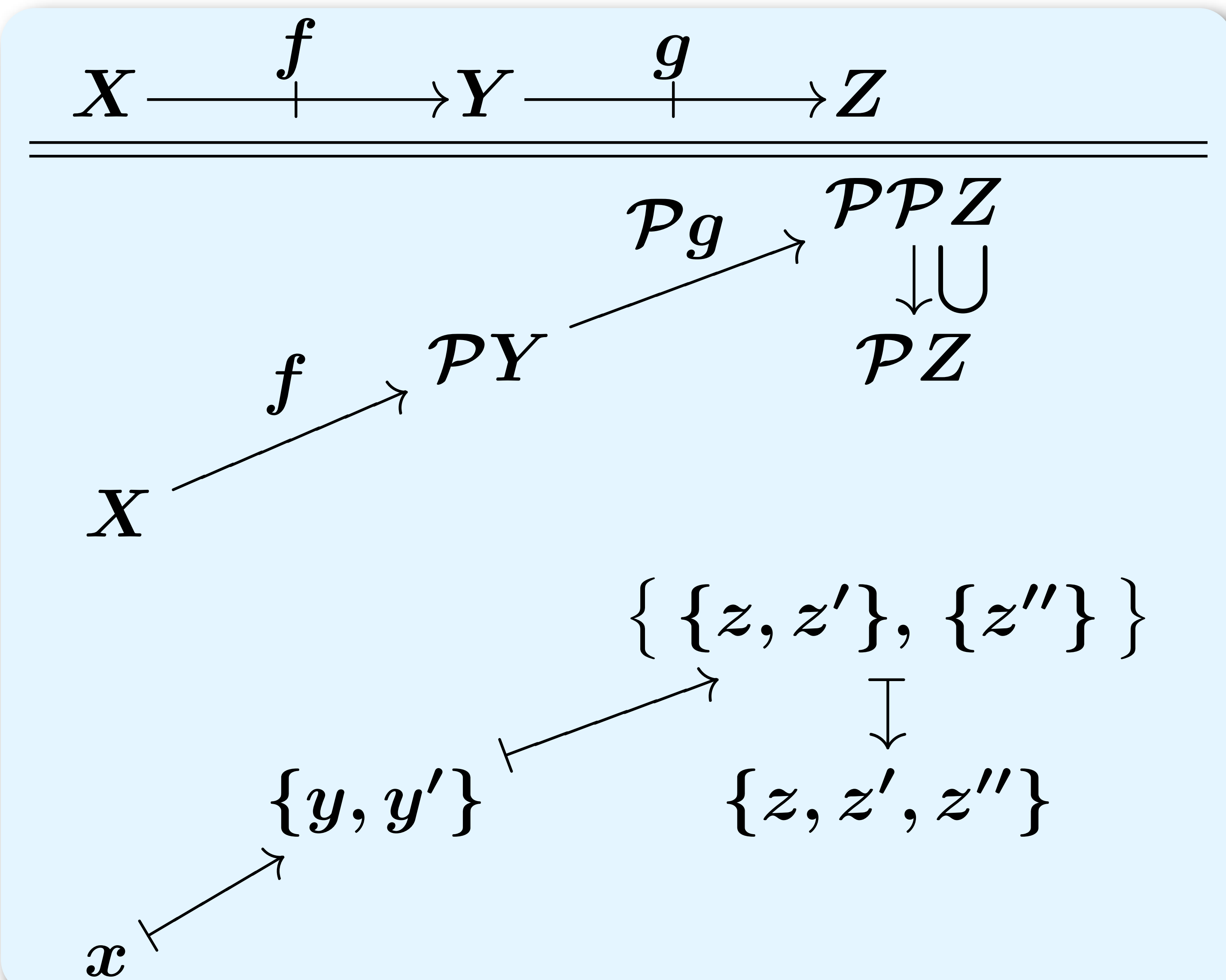


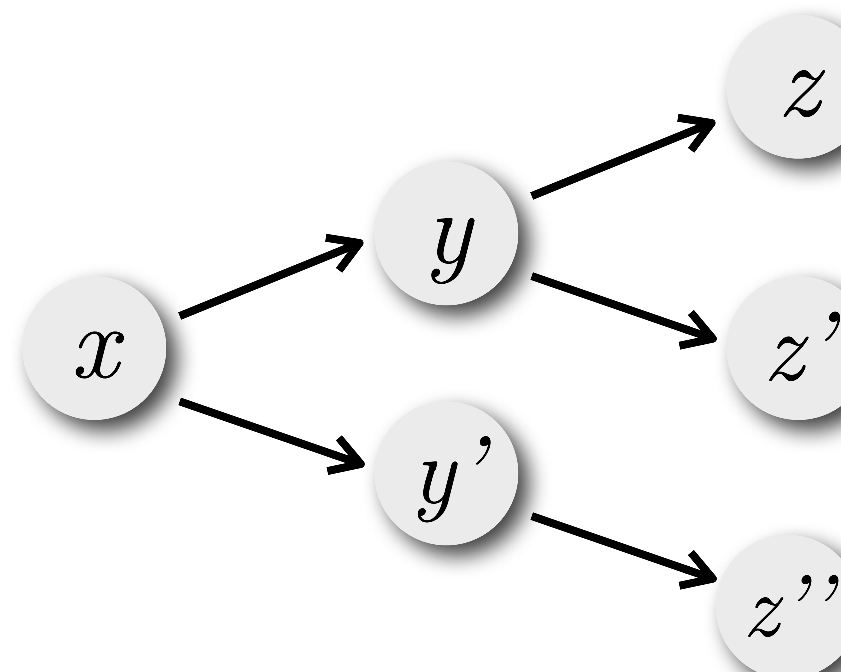


Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!\!\mid\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?

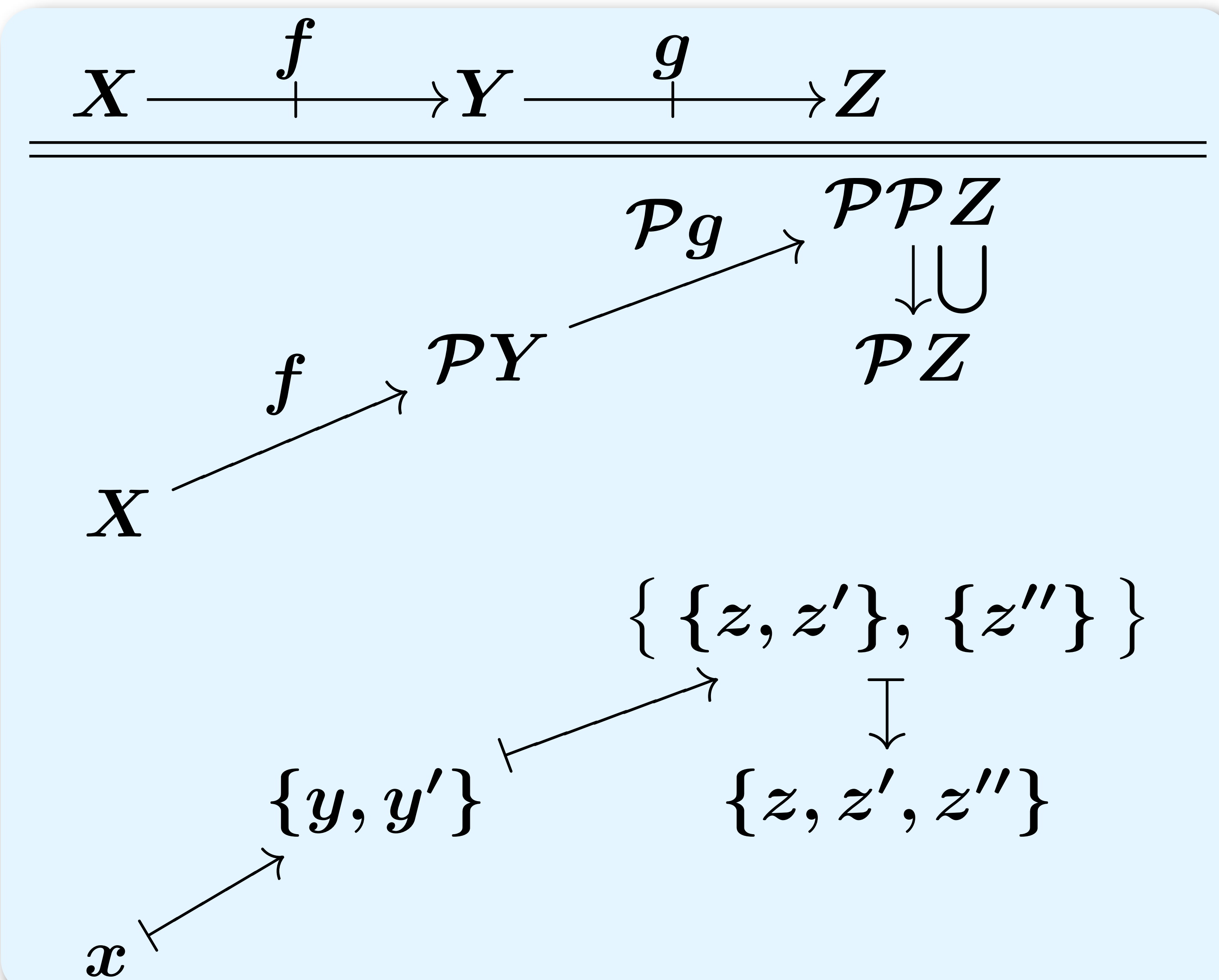


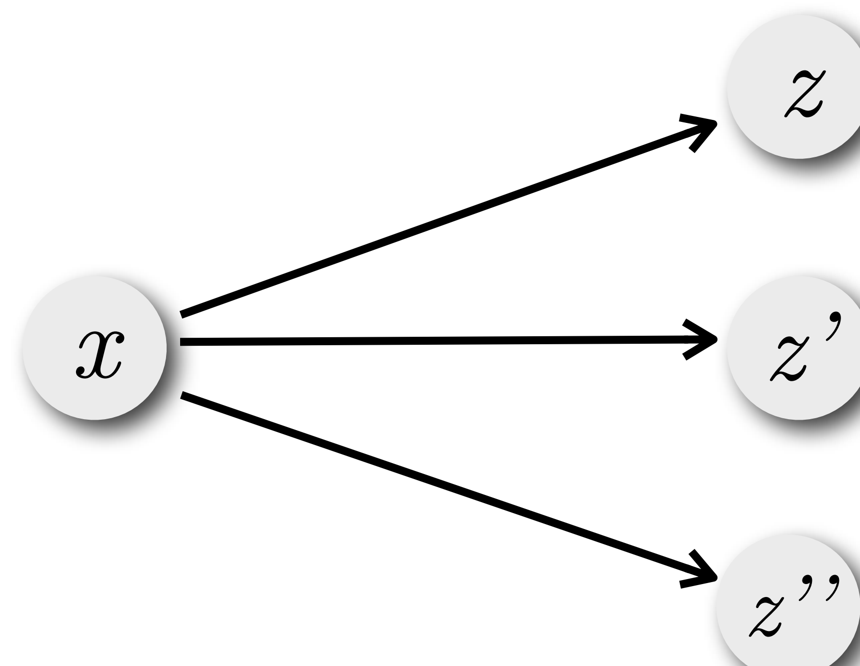


Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in Sets}}$$

- Composition of arrows?



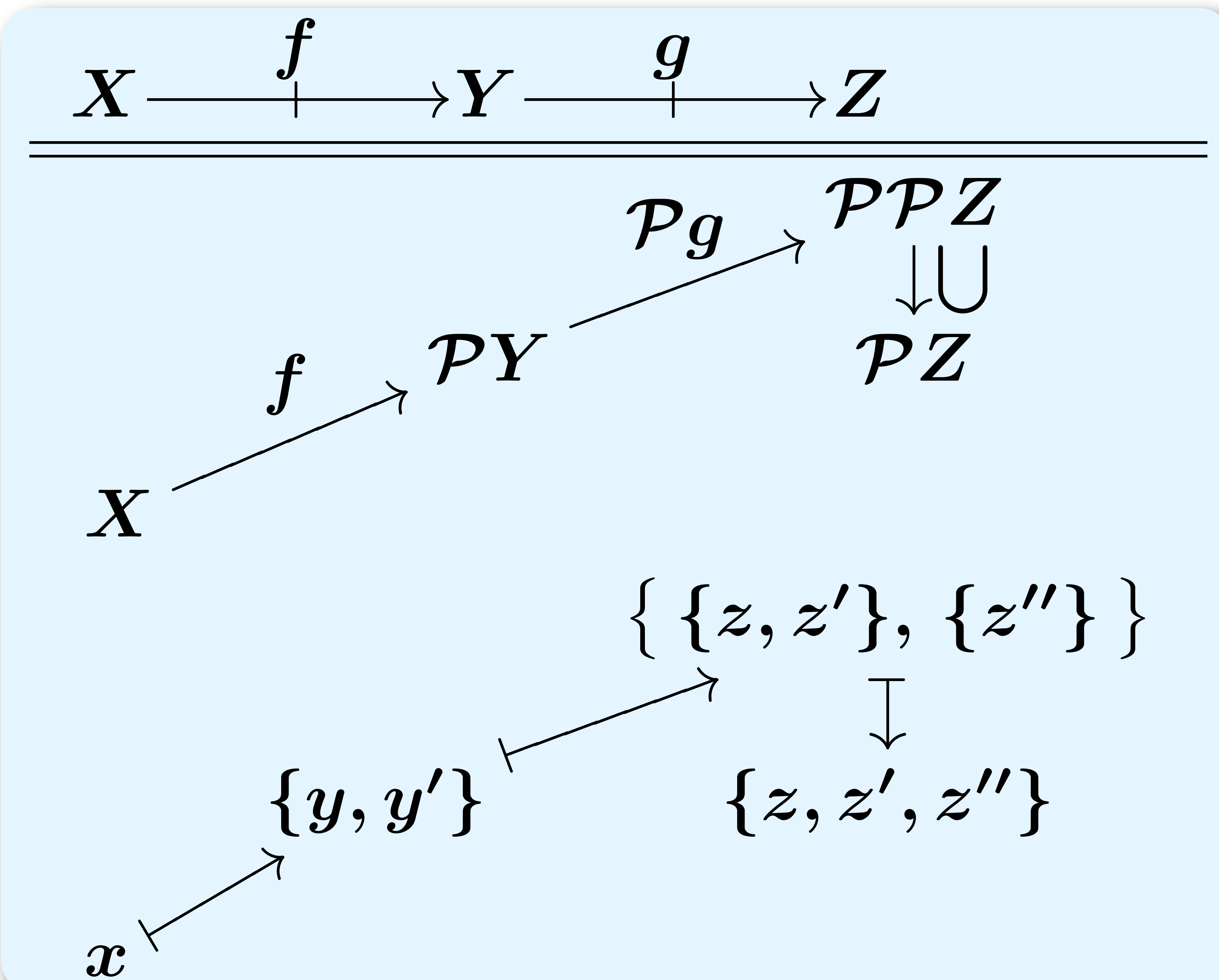


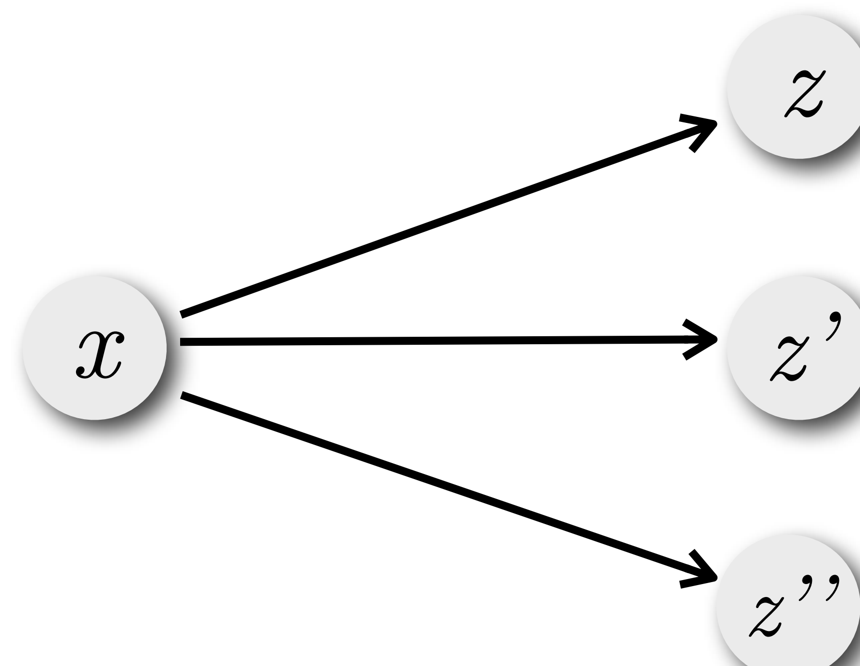
Kleisli Category $\mathcal{K}\ell(\mathcal{P})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \longrightarrow \mathcal{P}Y \quad \text{in Sets}}$$

- Composition of arrows?

unfolding
internal branching





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

$$
\begin{array}{ccc}
FX & \overset{F(\mathsf{tr}(c))}{- - \not\to -\to} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\
X & \underset{\mathsf{tr}(c)}{- - - \not\to - - \to} & Z
\end{array}
\quad \text{in } \mathcal{K}\ell(\mathcal{P})
$$

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

$$\begin{array}{c} \mathcal{P}FX \\ c \uparrow \\ X \end{array} \quad \text{in } \mathbf{Sets}$$

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ c \uparrow & & \uparrow \text{final} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

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

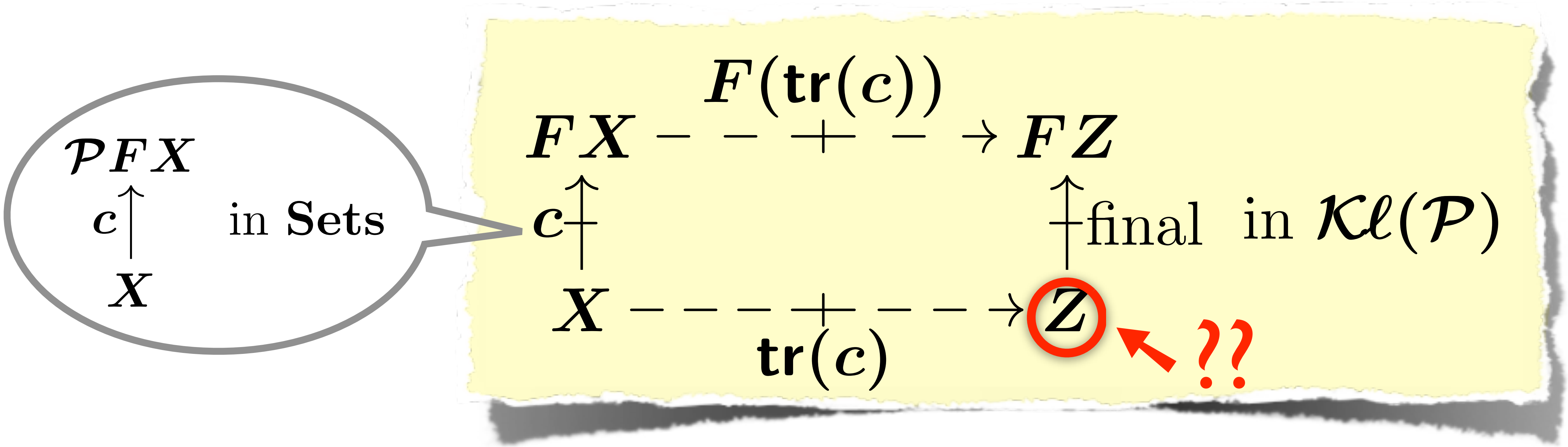
$$\begin{array}{c} \mathcal{P}FX \\ c \uparrow \\ X \end{array} \quad \text{in } \mathbf{Sets}$$

$$\begin{array}{ccc} FX & \overset{F(\mathsf{tr}(c))}{- - \dashrightarrow} & FZ \\ c \uparrow & & \uparrow \text{final} \\ X & \underset{\mathsf{tr}(c)}{- - \dashrightarrow} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

- Final coalgebra *captures* trace semantics:

$$\mathsf{tr}(c)(x) = \mathsf{tr}(d)(y) \iff \begin{array}{c} x \text{ and } y \text{ have the same} \\ \text{trace semantics} \end{array}$$

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



- Final coalgebra *captures* trace semantics:

$$\mathsf{tr}(c)(x) = \mathsf{tr}(d)(y) \quad \Longleftrightarrow$$

x and y have the same trace semantics

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

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

$$\begin{array}{c} FA \\ {\scriptstyle \text{init}}\downarrow\cong \\ A \end{array} \quad // \quad \begin{array}{c} FA \\ {\scriptstyle \text{init}^{-1}}\uparrow\cong \\ A \end{array} \quad / \quad \begin{array}{c} \mathcal{P}FA \\ \uparrow{\scriptstyle \eta_{FA}} \\ FA \\ {\scriptstyle \text{init}^{-1}}\uparrow\cong \\ A \end{array} \quad // \quad \begin{array}{c} FA \\ \uparrow{\scriptstyle \text{final}} \\ A \end{array}$$

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

$$
\begin{array}{ccc}
FX & \xrightarrow{F(\mathbf{tr}(c))} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\
X & \xrightarrow[\mathbf{tr}(c)]{} & Z
\end{array}
\quad \text{in } \mathcal{K}\ell(\mathcal{P})
$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

$$
\begin{array}{c}
FA \\
{\scriptstyle \text{init}}\downarrow\cong \\
A
\end{array}
\;\Big/\!\!\Big/\;
\begin{array}{c}
FA \\
{\scriptstyle \text{init}^{-1}}\uparrow\cong \\
A
\end{array}
\;\Big/\;
\begin{array}{c}
\mathcal{P}FA \\
\uparrow{\scriptstyle \eta_{FA}} \\
FA \\
{\scriptstyle \text{init}^{-1}}\uparrow\cong \\
A
\end{array}
\;\Big/\!\!\Big/\;
\begin{array}{c}
FA \\
\uparrow{\scriptstyle \text{final}} \\
A
\end{array}
$$

Lambek's Lemma

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

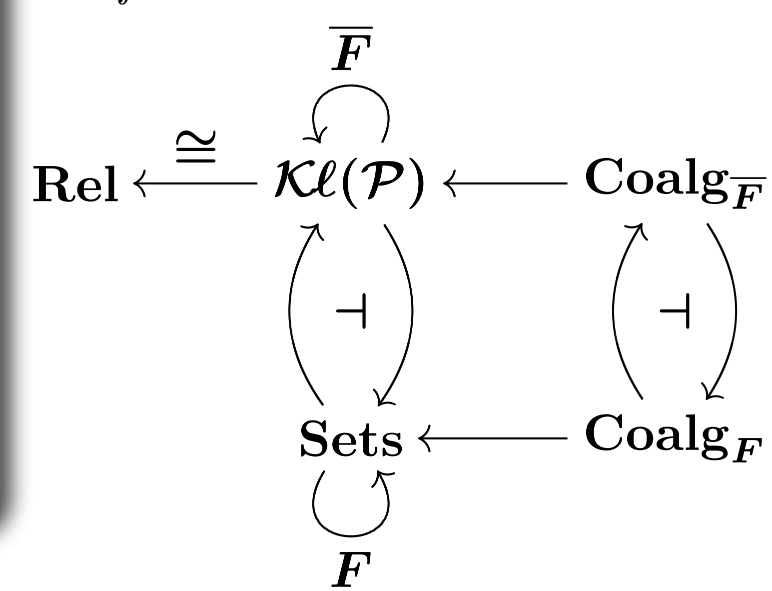
$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ c \uparrow & & \uparrow \text{final} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

$$\begin{array}{c} FA \\ \text{init} \downarrow \cong \\ A \end{array} \;\big/\!\!\big/\; \begin{array}{c} FA \\ \text{init}^{-1} \uparrow \cong \\ A \end{array} \;\big/\; \begin{array}{c} \mathcal{P}FA \\ \uparrow \eta_{FA} \\ FA \\ \text{init}^{-1} \uparrow \cong \\ A \end{array} \;\big/\!\!\big/\; \begin{array}{c} FA \\ \uparrow \text{final} \\ A \end{array}$$

Lambek's Lemma

Proof.



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

$$\begin{array}{ccc} FX & \overset{F(\mathsf{tr}(c))}{\dashrightarrow} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \underset{\mathsf{tr}(c)}{\dashrightarrow} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

??

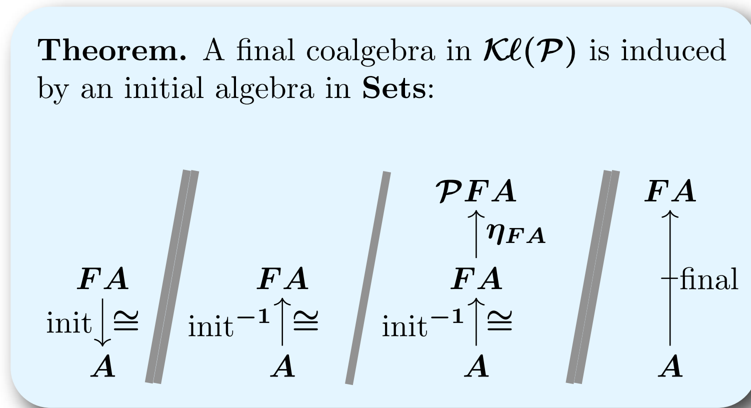
$$\begin{array}{c} FA \\ \text{init}\downarrow\cong \\ A \end{array} \;\Big/\!\!\Big/\; \begin{array}{c} FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \;\Big/\; \begin{array}{c} \mathcal{P}FA \\ \uparrow\eta_{FA} \\ FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \;\Big/\!\!\Big/\; \begin{array}{c} FA \\ \uparrow\text{final} \\ A \end{array}$$

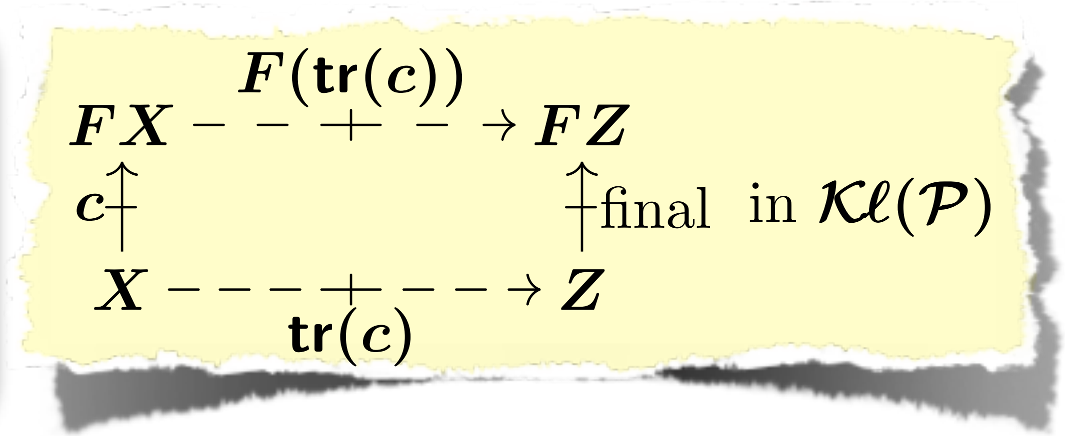
Lambek's Lemma

Proof.

$$\begin{array}{ccccc} & & \overline{F}\circlearrowleft & & \\ \mathbf{Rel} & \overset{\cong}{\longleftarrow} & \mathcal{K}\ell(\mathcal{P}) & \longleftarrow & \mathbf{Coalg}_{\overline{F}} \\ & & \uparrow\dashv\downarrow & & \uparrow\dashv\downarrow \\ & & \mathbf{Sets} & \longleftarrow & \mathbf{Coalg}_F \\ & & F\circlearrowleft & & \end{array}$$

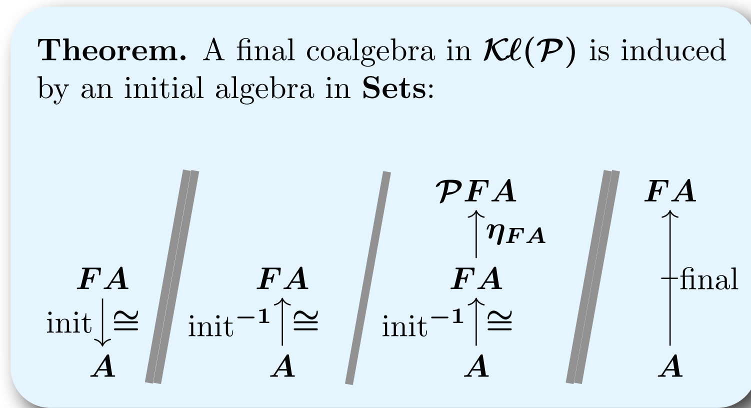
Initial algebra in Sets

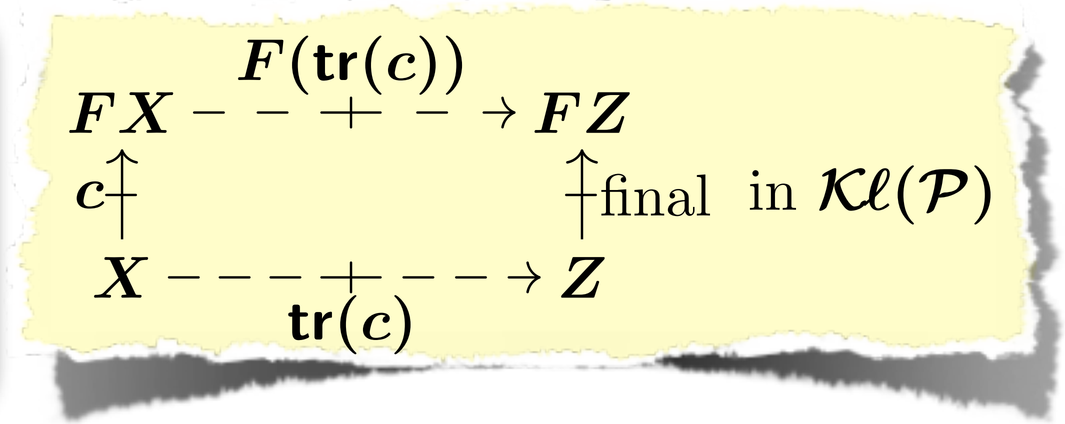




functor F	datatype constructor
initial/free algebra $\begin{array}{c} FA \\ \text{init}\downarrow\cong \\ A \end{array}$	algebraic datatype

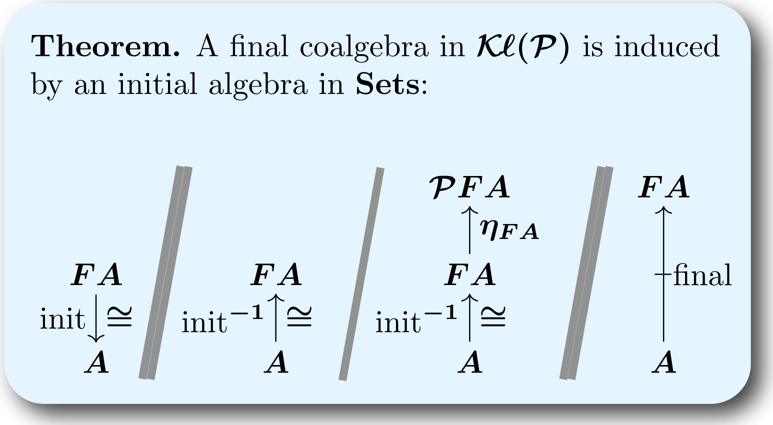
Initial algebra in Sets

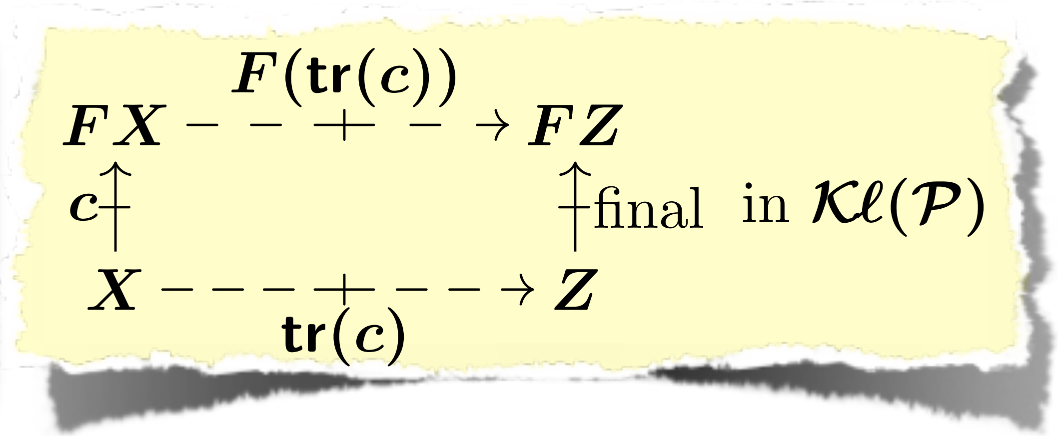




functor F	datatype constructor
$F = 1 + \Sigma \times \_$	$\mathtt{nil}$, $\mathtt{cons}\ a\ \_$ (for $a \in \Sigma$)
initial/free algebra $\begin{array}{c} FA \\ \text{init} \downarrow \cong \\ A \end{array}$	algebraic datatype

Initial algebra in Sets





functor F	datatype constructor
$F = 1 + \Sigma \times \_$	$\texttt{nil}$, $\texttt{cons}\ a\ \_$ (for $a \in \Sigma$)
initial/free algebra $\begin{array}{c} FA \\ \mathrm{init}\downarrow\cong \\ A \end{array}$	algebraic datatype
$\begin{array}{c} 1 + \Sigma \times \Sigma^* \\ [\texttt{nil}, \texttt{cons}]\downarrow\cong \\ \Sigma^* \end{array}$	lists over Σ

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

$$F = 1 + \Sigma \times \_$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

$$\begin{array}{c} FA \\ \text{init}\downarrow\cong \\ A \end{array} \quad \Big/ \quad \begin{array}{c} FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \quad \Big/ \quad \begin{array}{c} \mathcal{P}FA \\ \uparrow\eta_{FA} \\ FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \quad \Big/ \quad \begin{array}{c} FA \\ \uparrow\text{final} \\ A \end{array}$$

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathbf{tr}(c))} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \xrightarrow[\mathbf{tr}(c)]{} & Z \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

A

$\begin{array}{c} 1 + \Sigma \times \Sigma^* \\ [\mathtt{nil}, \mathtt{cons}]\downarrow\cong \\ \Sigma^* \end{array}$	lists over Σ

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

$$F = 1 + \Sigma \times \_$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

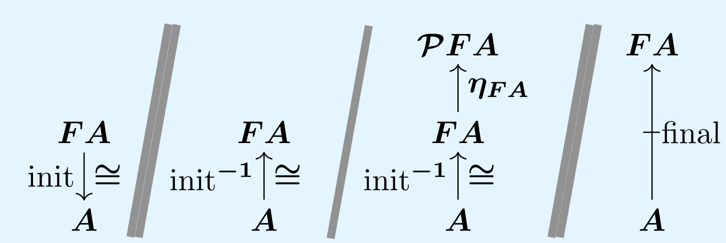
$$\begin{array}{c} FA \\ \text{init}\downarrow\cong \\ A \end{array} \Bigg/ \begin{array}{c} FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \Bigg/ \begin{array}{c} \mathcal{P}FA \\ \uparrow\eta_{FA} \\ FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array} \Bigg/ \begin{array}{c} FA \\ \uparrow\text{final} \\ A \end{array}$$

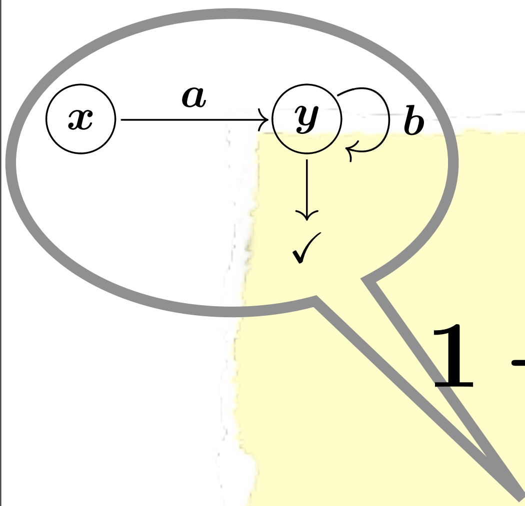
$$\begin{array}{ccc} 1 + \Sigma \times X & \xrightarrow{F(\mathsf{tr}(c))} & 1 + \Sigma \times \Sigma^* \\ c\uparrow & & \uparrow\text{final} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & \Sigma^* \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

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

$$F = 1 + \Sigma \times \_$$

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:





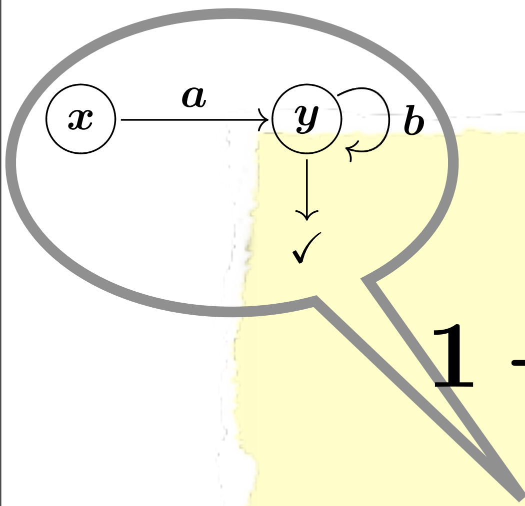
$$\begin{array}{ccc}
1 + \Sigma \times X & \xrightarrow{F(\mathbf{tr}(c))} & 1 + \Sigma \times \Sigma^* \\
c \uparrow & & \uparrow \text{final} \\
X & \xrightarrow[\mathbf{tr}(c)]{} & \Sigma^*
\end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

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

Theorem. A final coalgebra in $\mathcal{K}\ell(\mathcal{P})$ is induced by an initial algebra in **Sets**:

$$\frac{\begin{array}{c} FA \\ \text{init}\downarrow\cong \\ A \end{array}}{\begin{array}{c} FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array}} \Bigg/ \frac{\begin{array}{c} \mathcal{P}FA \\ \uparrow\eta_{FA} \\ FA \\ \text{init}^{-1}\uparrow\cong \\ A \end{array}}{\begin{array}{c} FA \\ \uparrow\!\!\!+\text{final} \\ A \end{array}}$$

$$F = 1 + \Sigma \times \_$$



$$\begin{array}{ccc} 1+\Sigma\times X & \overset{F(\mathbf{tr}(c))}{- - \!\!+\!\! - \rightarrow} & 1+\Sigma\times\Sigma^* \\ c\uparrow\!\!\!+ & & \uparrow\!\!\!+\text{final} \\ X & \underset{\mathbf{tr}(c)}{- - - - \!\!+\!\! - - - \rightarrow} & \Sigma^* \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{P})$$

$$\frac{X \xrightarrow{\mathbf{tr}(c)} \Sigma^* \quad \text{in } \mathcal{K}\ell(\mathcal{P})}{X \xrightarrow{\mathbf{tr}(c)} \mathcal{P}\Sigma^* \quad \text{in } \mathbf{Sets}}$$

$$x \longmapsto \{a, ab, abb, \dots\}$$

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

$$\begin{array}{ccc}
FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\
X & \xrightarrow[\mathsf{tr}(c)]{} & Z
\end{array}$$

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

$$
\begin{array}{ccc}
FX & \xrightarrow{F(\mathbf{tr}(c))} & FZ \\
c \uparrow & & \uparrow \text{final} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\
X & \xrightarrow[\mathbf{tr}(c)]{} & Z
\end{array}
$$

branching is implicit,
unfolded

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

$$
\begin{array}{ccc}
FX & \xrightarrow{\;F(\mathsf{tr}(c))\;} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\
X & \xrightarrow[\;\mathsf{tr}(c)\;]{} & Z
\end{array}
$$

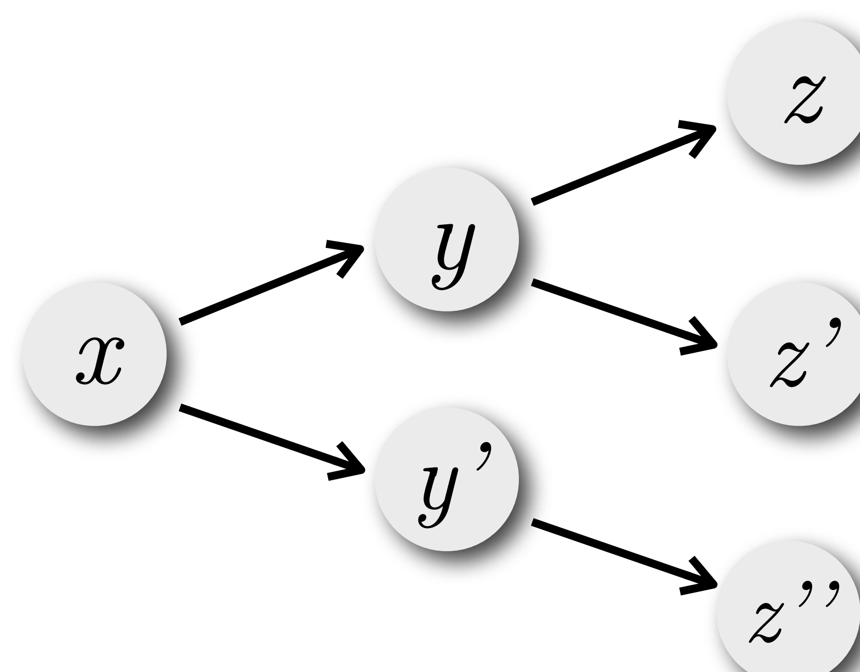
branching is implicit,
unfolded

x

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

$$\begin{array}{ccc} FX & \overset{F(\mathbf{tr}(c))}{-\,-\,\dashrightarrow} & FZ \\ c\uparrow & & \uparrow \text{final in } \mathcal{K}\ell(\mathcal{P}) \\ X & \underset{\mathbf{tr}(c)}{-\,-\,-\,\dashrightarrow} & Z \end{array}$$

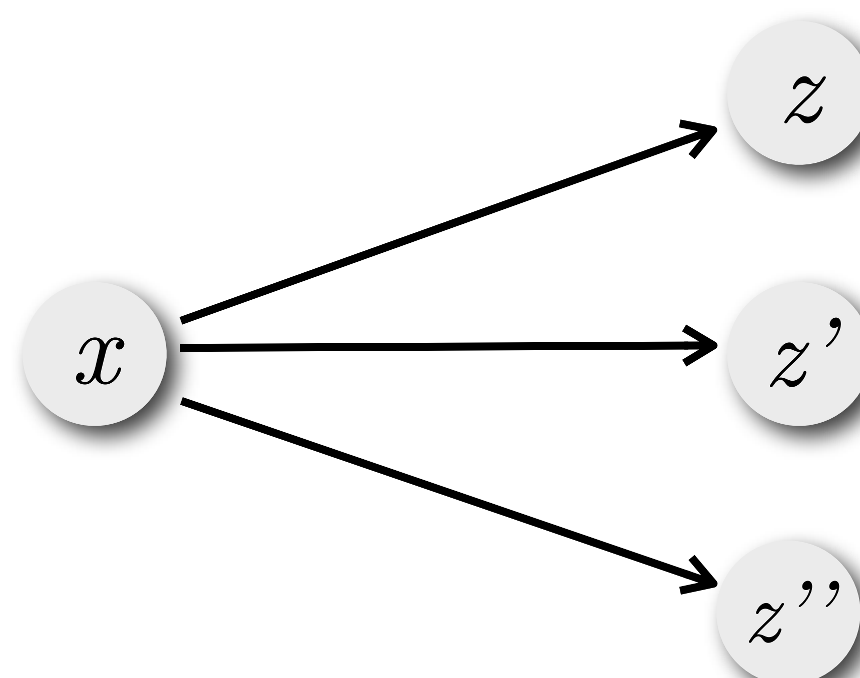
branching is implicit, unfolded



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

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array}$$

branching is implicit,
unfolded

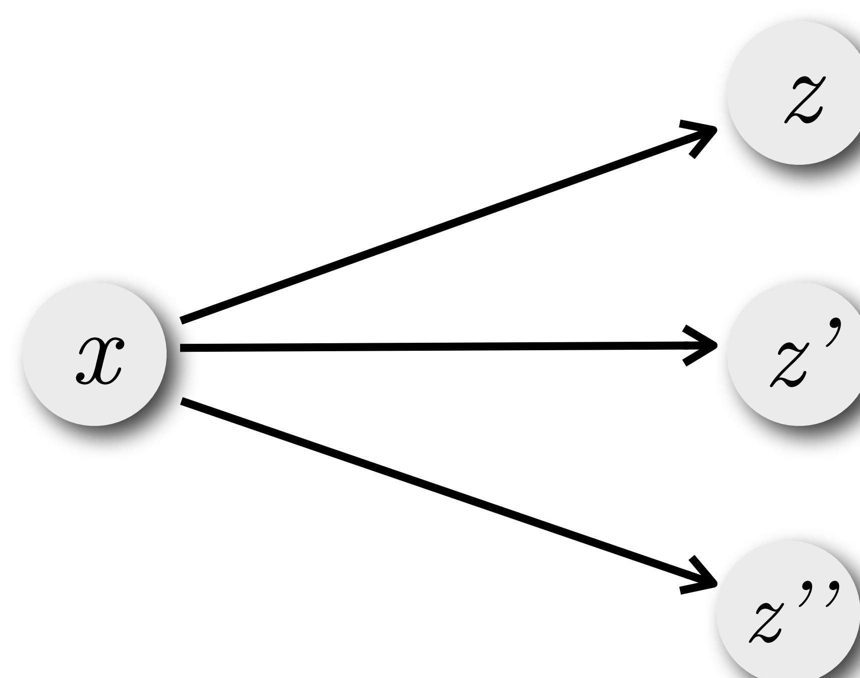


Final Coalgebra in $\mathcal{K}\ell(\mathcal{P})$

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathbf{tr}(c))} & FZ \\ c \uparrow & & \uparrow \text{final in } \mathcal{K}\ell(\mathcal{P}) \\ X & \xrightarrow[\mathbf{tr}(c)]{} & Z \end{array}$$

branching is implicit, unfolded

"trace semantics"



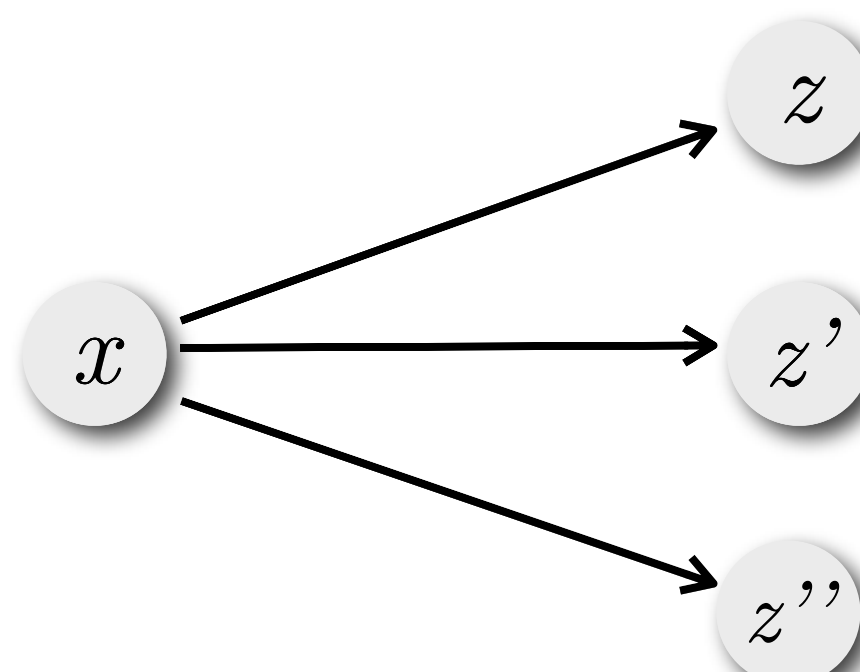
Final Coalgebra in $\mathcal{K}\ell(\mathcal{P})$

$$\begin{array}{ccc} FX & \overset{F(\mathbf{tr}(c))}{- - \!\!\!+\!\!\! - \to} & FZ \\ c\Big\uparrow & & \Big\uparrow \text{final} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\ X & \underset{\mathbf{tr}(c)}{- - - \!\!\!+\!\!\! - - \to} & Z \end{array}$$

branching is implicit, unfolded

arises from initial algebra

"trace semantics"



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

$$
\begin{array}{ccc}
FX & \xrightarrow{\;F(\mathsf{tr}(c))\;} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \quad \text{in } \mathcal{K}\ell(\mathcal{P}) \\
X & \xrightarrow[\;\mathsf{tr}(c)\;]{} & Z
\end{array}
$$

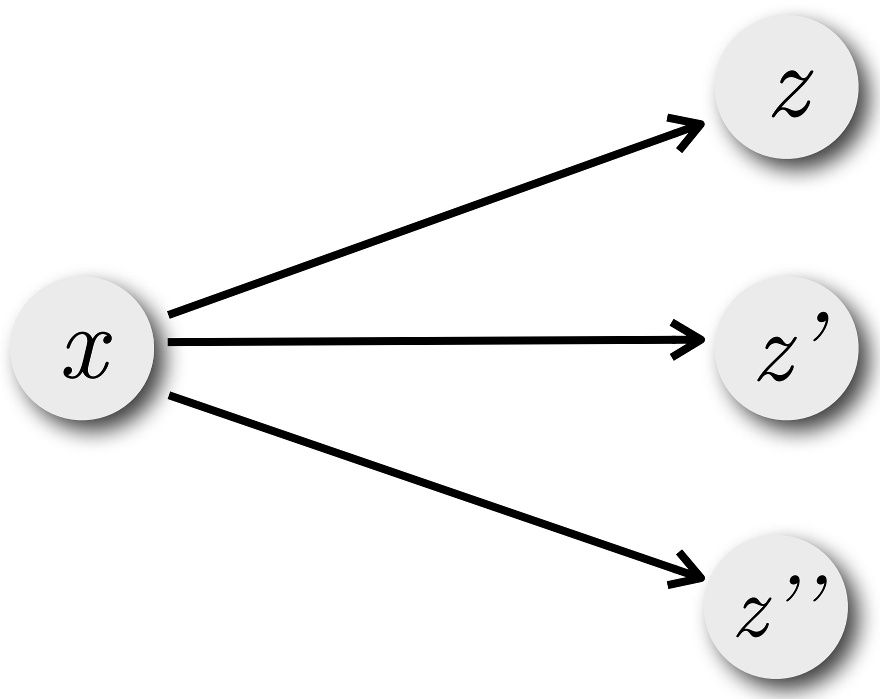
branching is implicit, unfolded

arises from initial algebra

"trace semantics"

The diagram commutes

$$\iff \quad \mathsf{tr}(c)(x) = \bigcup_{x \xrightarrow{a} y} \{a \cdot \sigma \mid \sigma \in \mathsf{tr}(c)(y)\}$$



Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ c \uparrow & & \uparrow \text{final in } \mathcal{K}\ell(\mathcal{D}) \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array}$$

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

probabilistic branching is implicit, unfolded

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FZ \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final in } \mathcal{K}\ell(\mathcal{D})} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array}$$

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$$
\begin{array}{ccc}
FX & \xrightarrow{\;F(\mathbf{tr}(c))\;} & FZ \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final in } \mathcal{K}\ell(\mathcal{D})} \\
X & \xrightarrow[\;\mathbf{tr}(c)\;]{} & Z
\end{array}
$$

<u>probabilistic</u> branching is implicit, unfolded

"trace semantics"

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$$\begin{array}{ccc} \boldsymbol{FX} & \overset{\boldsymbol{F}(\mathbf{tr}(\boldsymbol{c}))}{\dashrightarrow} & \boldsymbol{FZ} \\ \boldsymbol{c}\uparrow & & \uparrow \text{final in } \mathcal{K}\ell(\mathcal{D}) \\ \boldsymbol{X} & \underset{\mathbf{tr}(\boldsymbol{c})}{\dashrightarrow} & \boldsymbol{Z} \end{array}$$

probabilistic branching is implicit, unfolded

arises from initial algebra

"trace semantics"

Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

"probabilistic function"

- The set of *subdistributions*

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

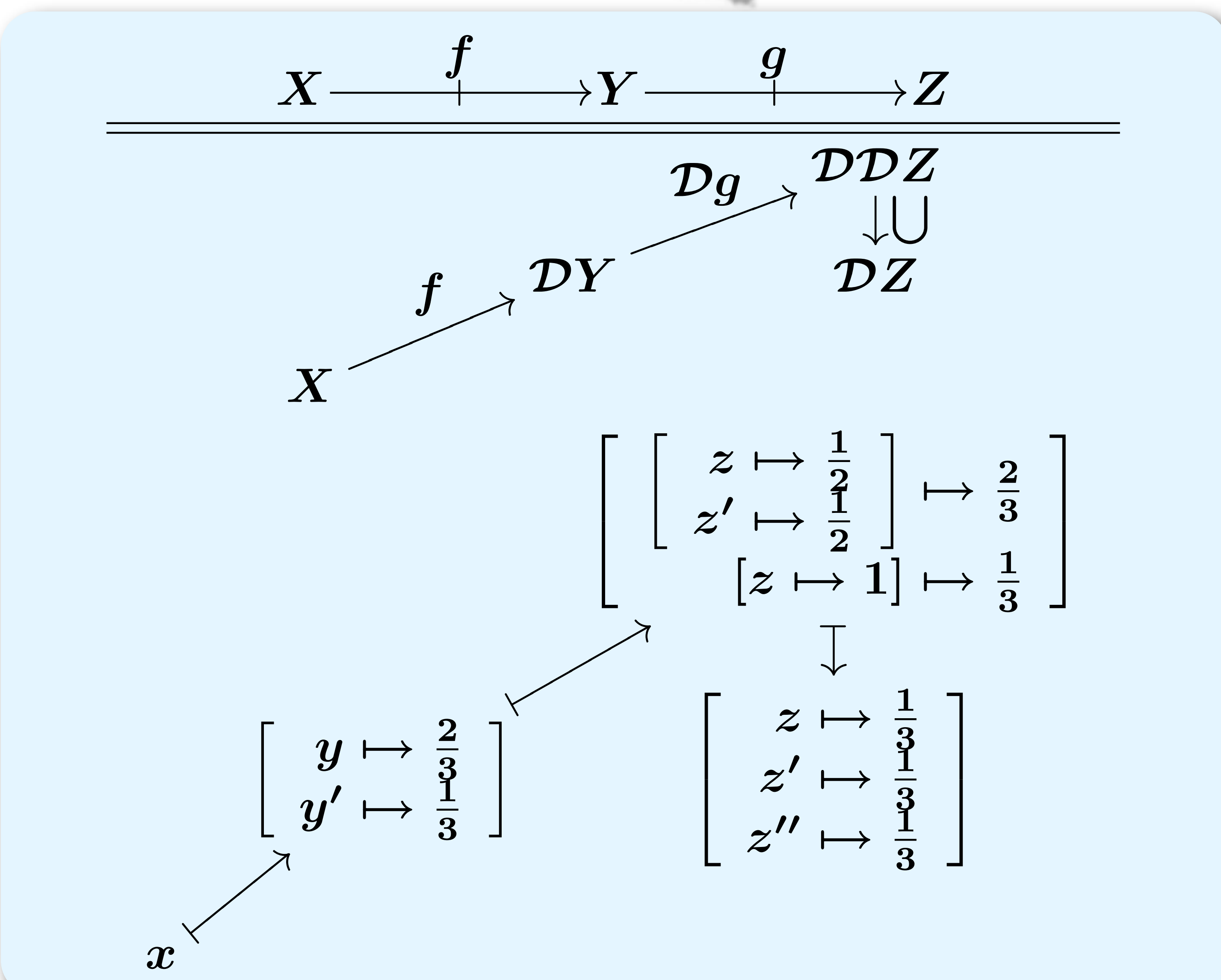
- $X \xrightarrow{\quad f \quad} \mathcal{D}Y$

$$x \longmapsto \begin{bmatrix} y \mapsto \frac{2}{3} \\ y' \mapsto \frac{1}{3} \end{bmatrix}$$

Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

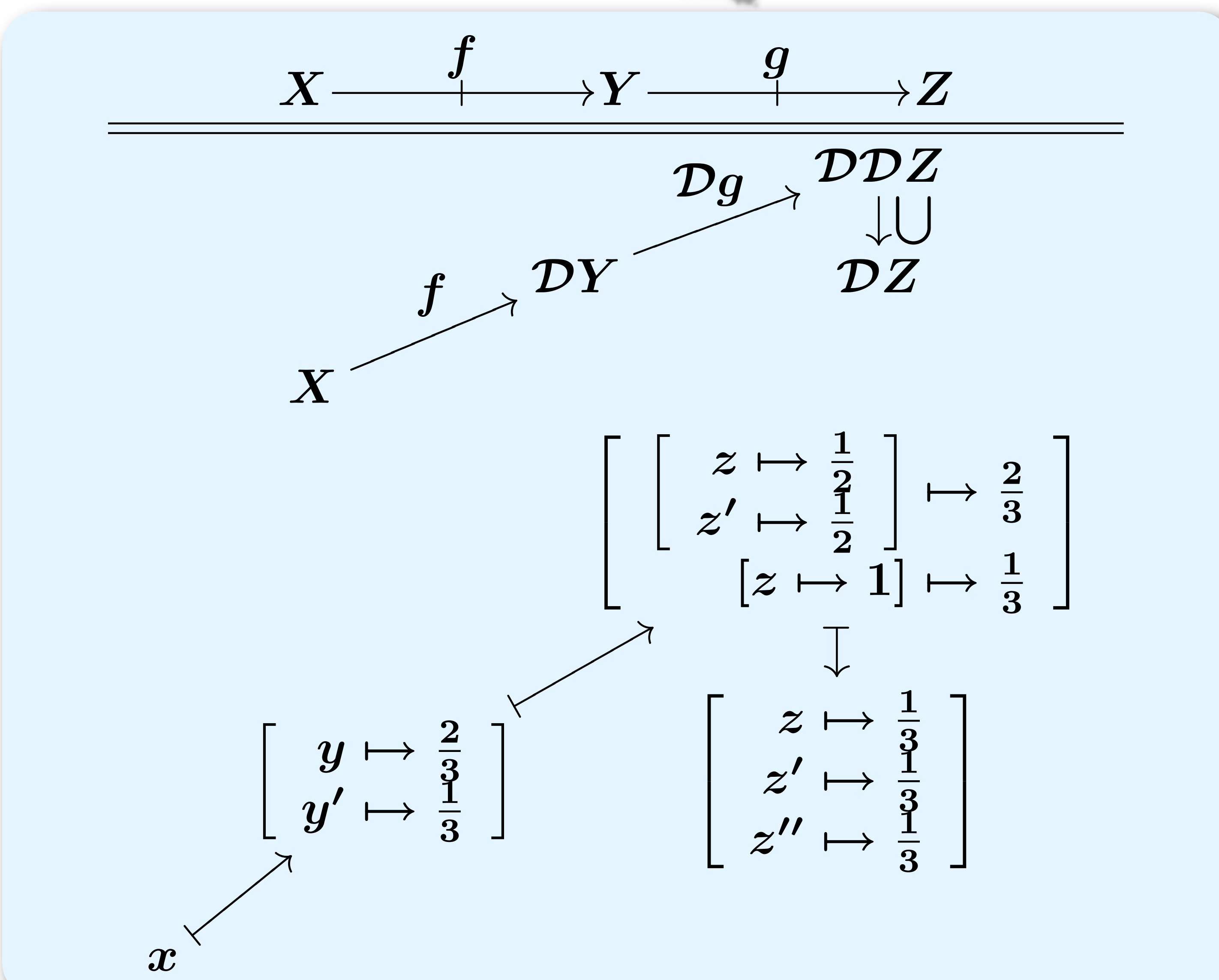
- Composition of arrows?



Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\ \ Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?

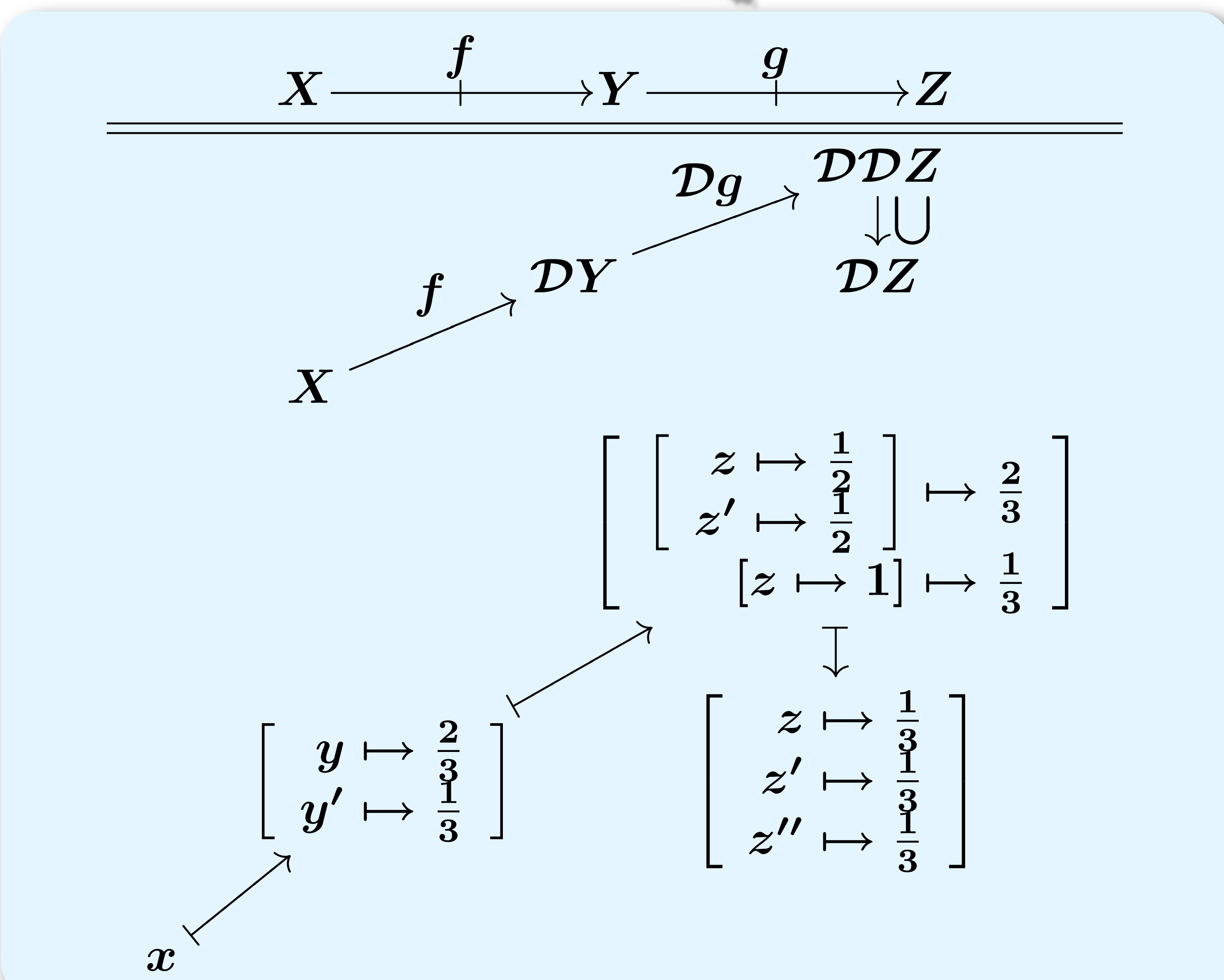


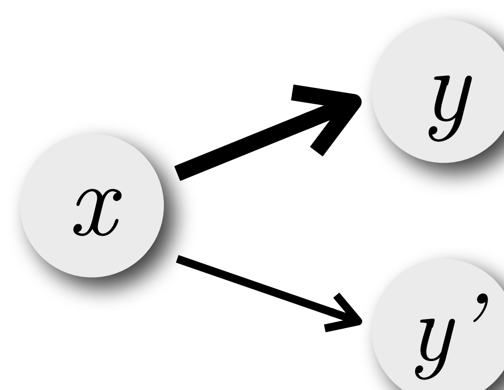
x

Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\;\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?

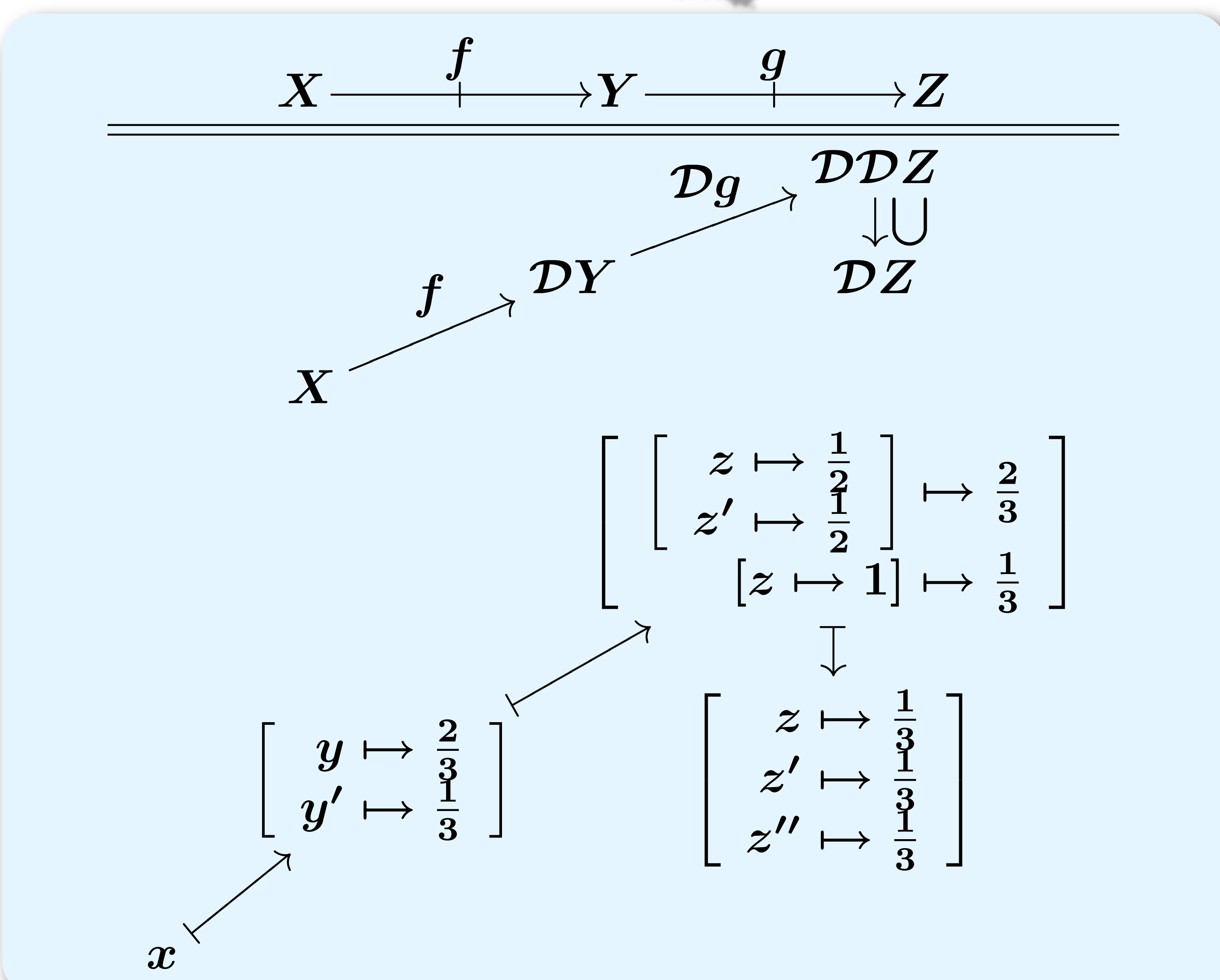


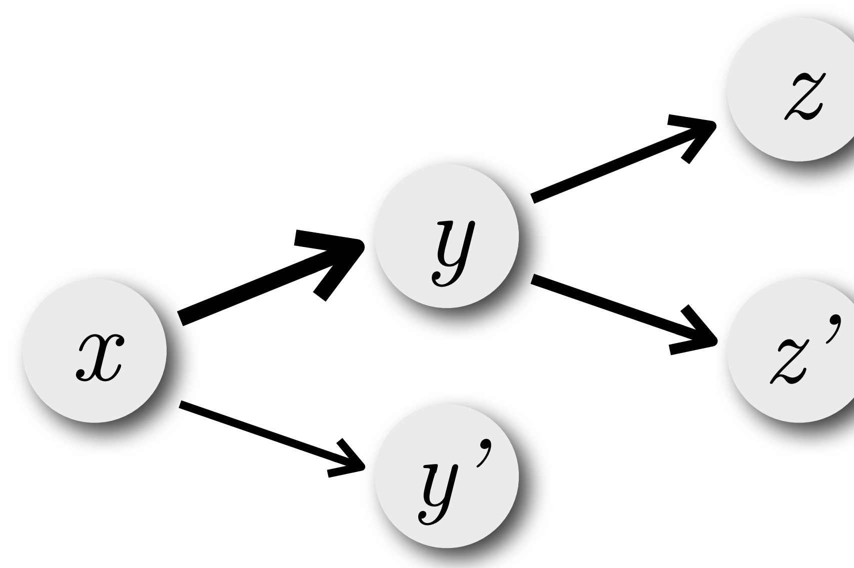


Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?

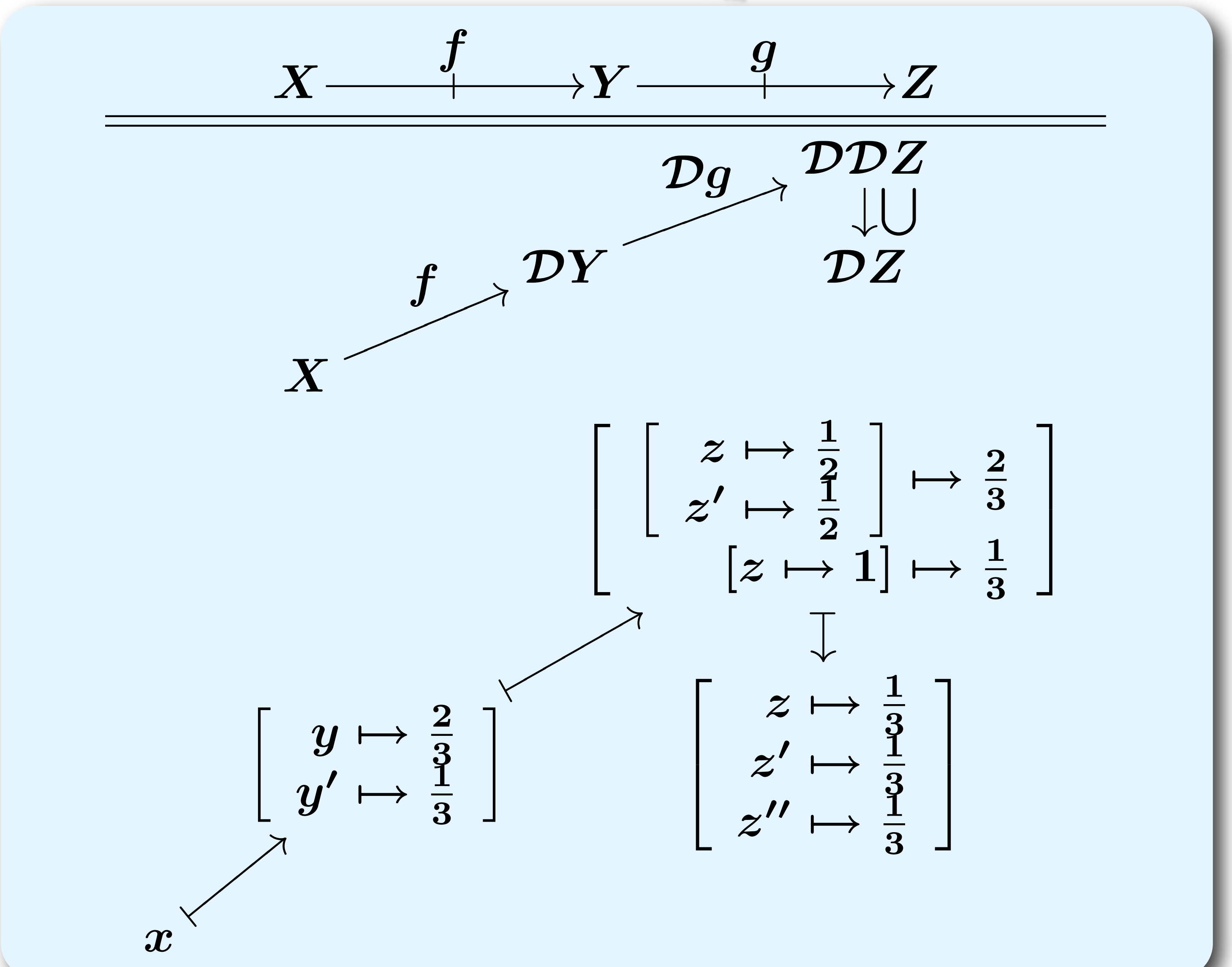


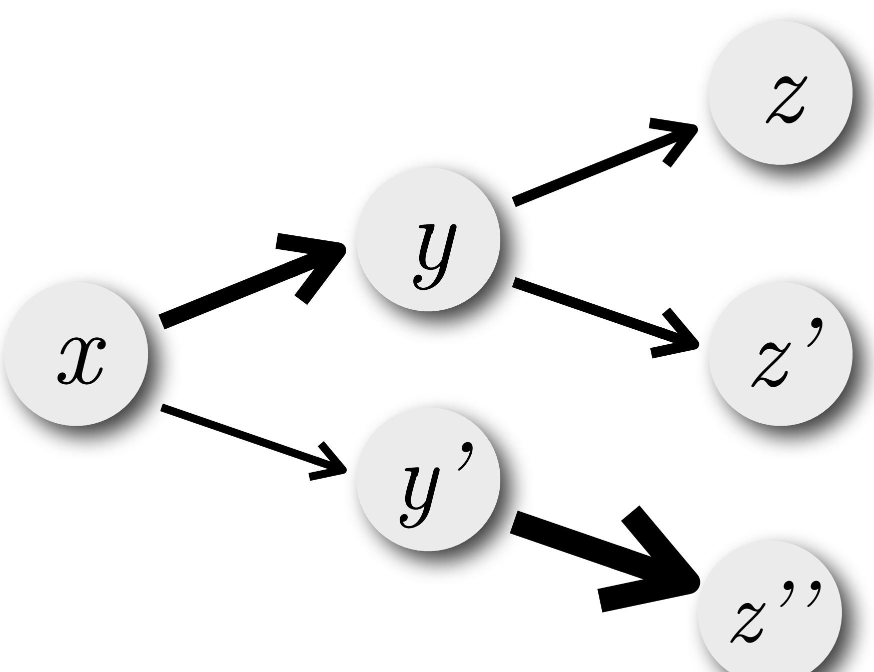


Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!+\;\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in Sets}}$$

- Composition of arrows?

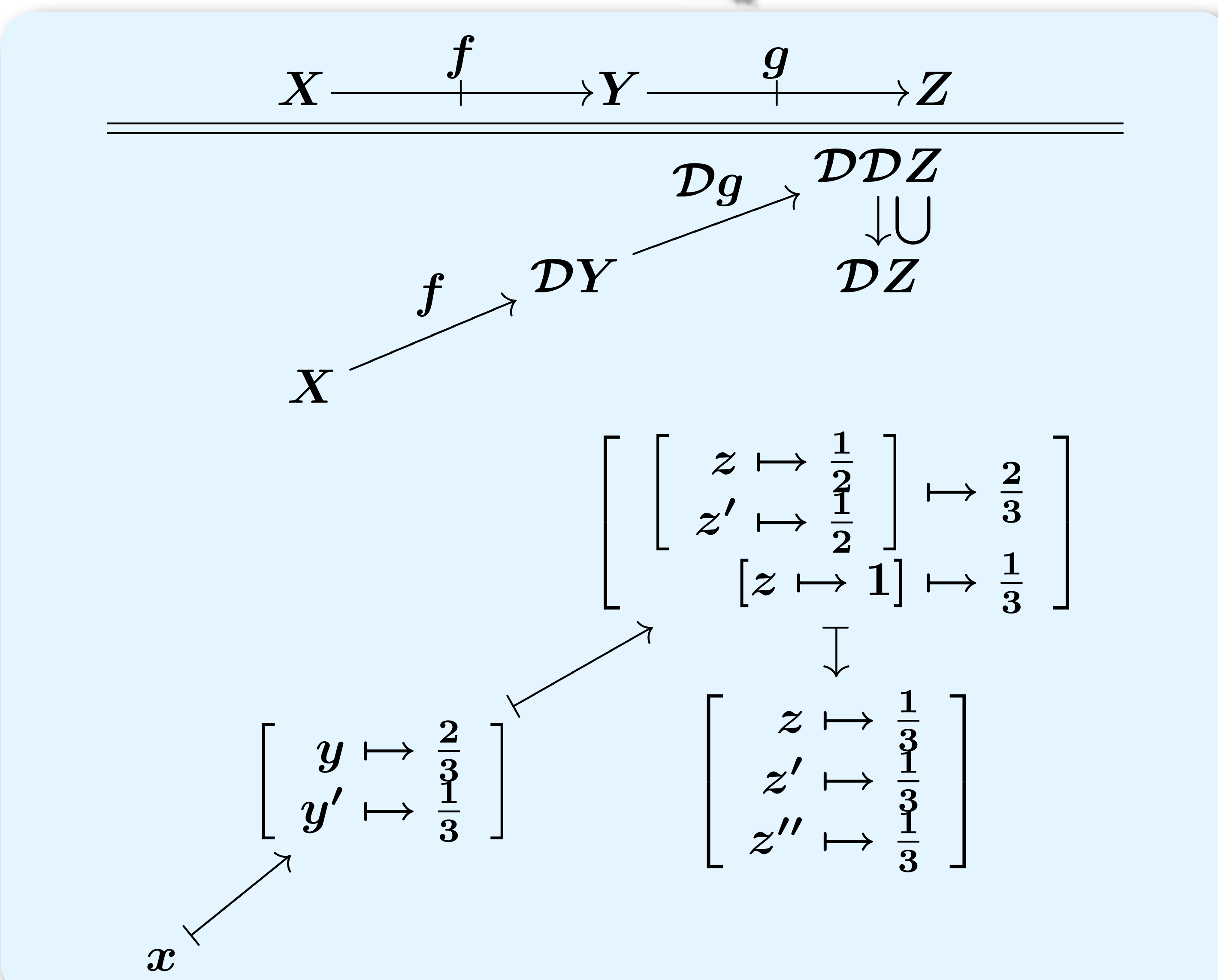


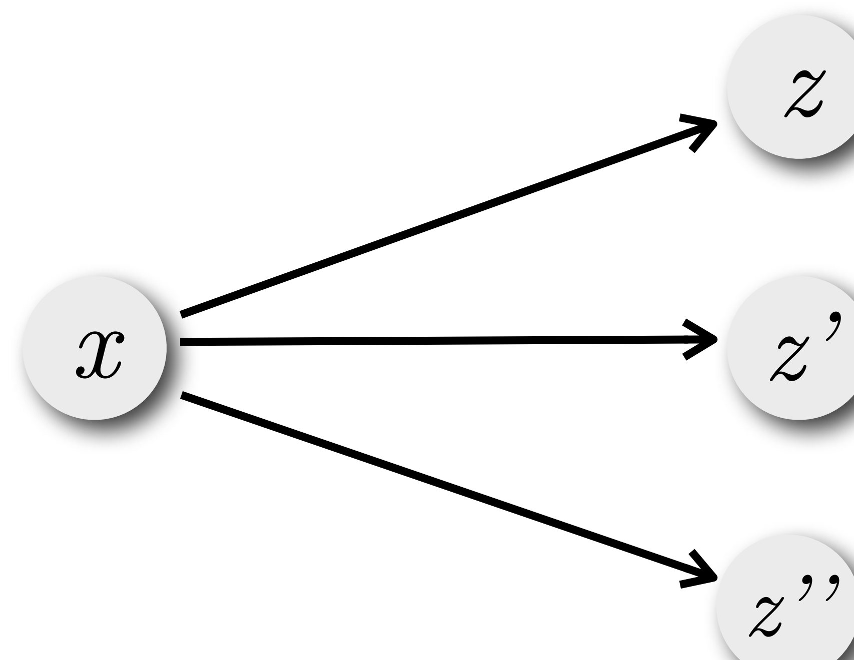


Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!\!\shortmid\quad Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in } \mathbf{Sets}}$$

- Composition of arrows?



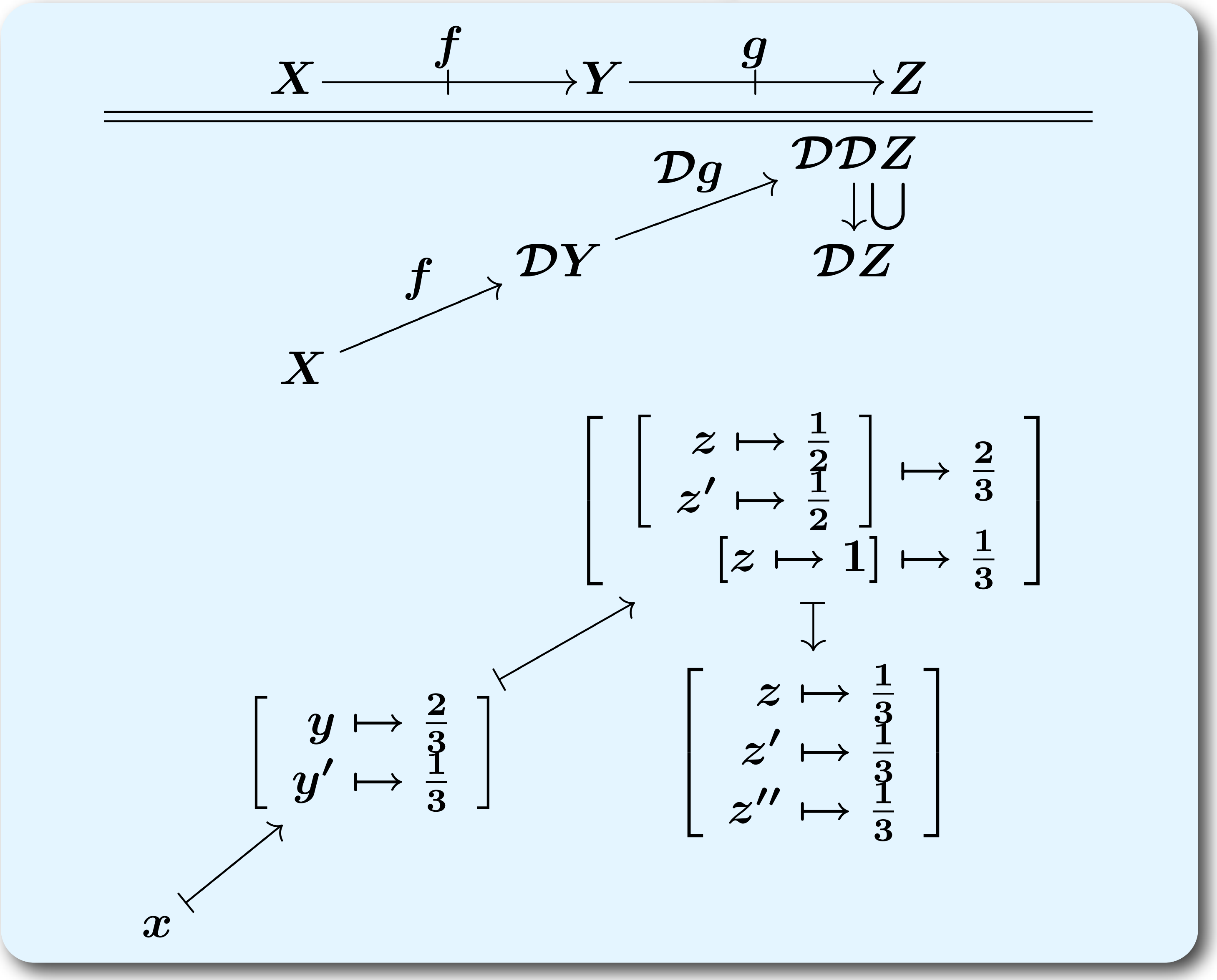


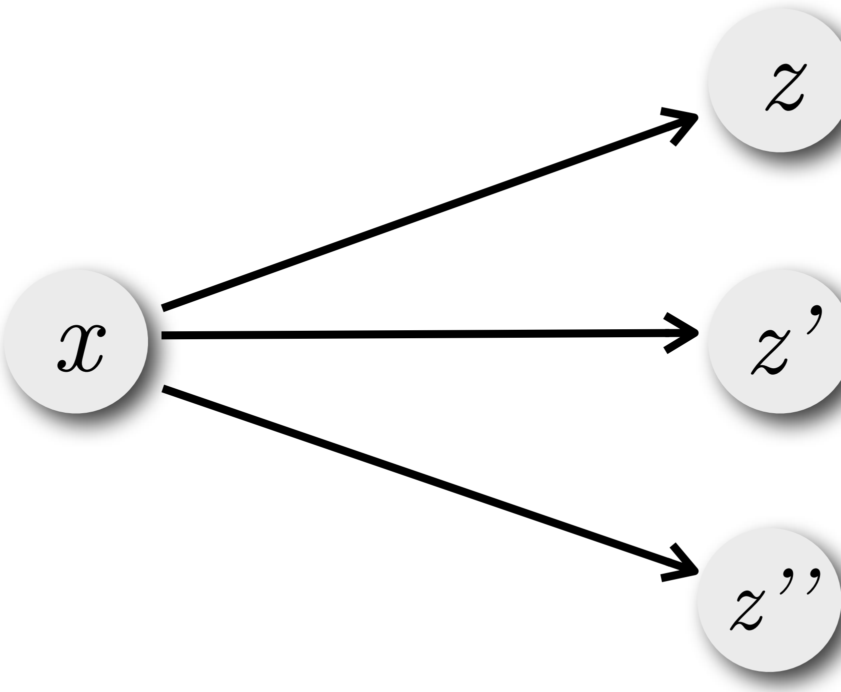
Kleisli Category $\mathcal{K}\ell(\mathcal{D})$

$$\frac{X \longrightarrow\!\!\!\!\!\!\!+\;\; Y \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \longrightarrow \mathcal{D}Y \quad \text{in Sets}}$$

- Composition of arrows?

unfolding internal branching





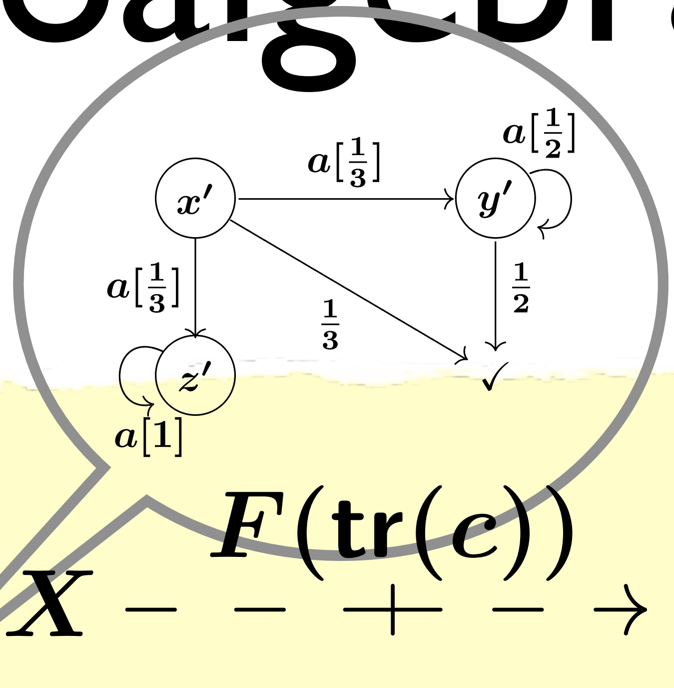
Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$\boldsymbol{F} = 1 + \boldsymbol{\Sigma} \times \_$

$$\begin{array}{ccc} 1 + \boldsymbol{\Sigma} \times \boldsymbol{X} & \xrightarrow{\boldsymbol{F}(\mathbf{tr}(\boldsymbol{c}))} & 1 + \boldsymbol{\Sigma} \times \boldsymbol{\Sigma}^* \\ \boldsymbol{c} \uparrow & & \uparrow \text{final} \\ \boldsymbol{X} & \xrightarrow[\mathbf{tr}(\boldsymbol{c})]{} & \boldsymbol{\Sigma}^* \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{D})$$

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

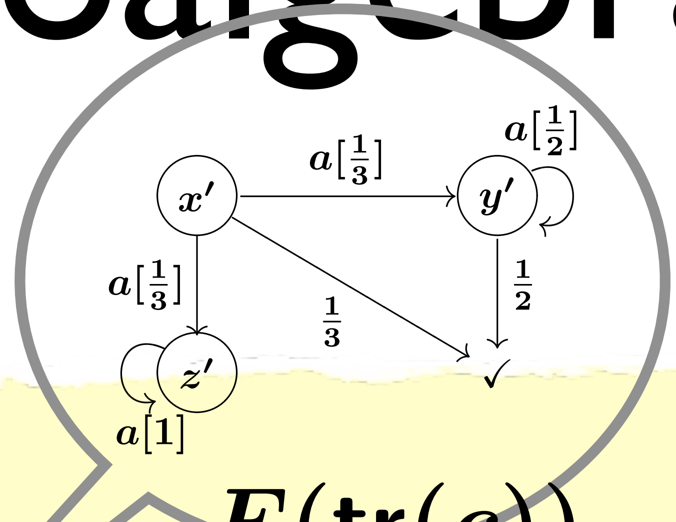
$$F = 1 + \Sigma \times \_$$



$$\begin{array}{ccc}
1 + \Sigma \times X & \xrightarrow{F(\mathsf{tr}(c))} & 1 + \Sigma \times \Sigma^* \\
{\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\
X & \xrightarrow[\mathsf{tr}(c)]{} & \Sigma^*
\end{array} \qquad \text{in } \mathcal{K}\ell(\mathcal{D})$$

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$$F = 1 + \Sigma \times \_$$

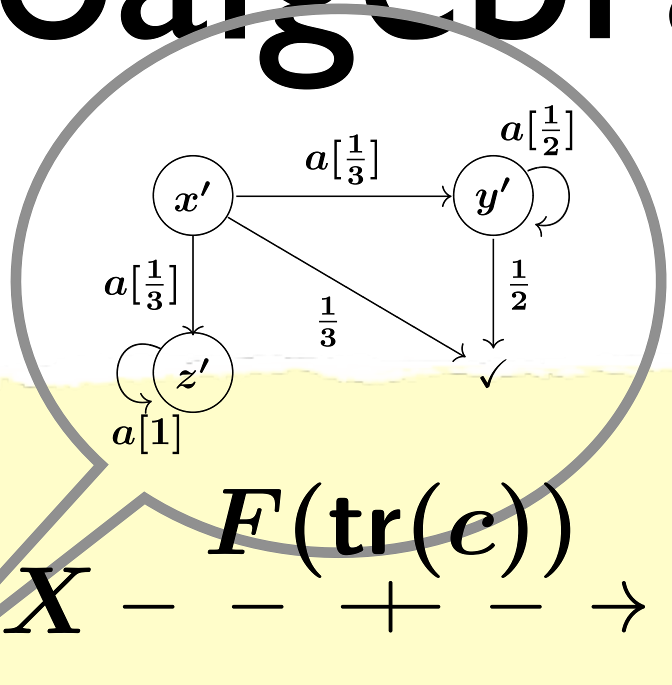


$$\begin{array}{ccc} 1 + \Sigma \times X & \xrightarrow{F(\mathsf{tr}(c))} & 1 + \Sigma \times \Sigma^* \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & \Sigma^* \end{array} \quad \text{in } \mathcal{K}\ell(\mathcal{D})$$

$$\dfrac{X \xrightarrow{\mathsf{tr}(c)} \Sigma^* \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \xrightarrow{\mathsf{tr}(c)} \mathcal{D}(\Sigma^*) \quad \text{in Sets}}$$

Final Coalgebra in $\mathcal{K}\ell(\mathcal{D})$

$$F = 1 + \Sigma \times \_$$



$$\begin{array}{ccc} 1+\Sigma\times X & \xrightarrow{F(\mathsf{tr}(c))} & 1+\Sigma\times\Sigma^* \\ c \uparrow & & \uparrow \text{final} \quad \text{in } \mathcal{K}\ell(\mathcal{D}) \\ X & \xrightarrow[\mathsf{tr}(c)]{} & \Sigma^* \end{array}$$

$$\frac{X \xrightarrow{\mathsf{tr}(c)} \Sigma^* \quad \text{in } \mathcal{K}\ell(\mathcal{D})}{X \xrightarrow{\mathsf{tr}(c)} \mathcal{D}(\Sigma^*) \quad \text{in Sets}}$$

$$x \longmapsto \left[\langle\rangle \mapsto \tfrac{1}{3},\ a \mapsto \tfrac{1}{6},\ aa \mapsto \tfrac{1}{12},\ \ldots \right]$$

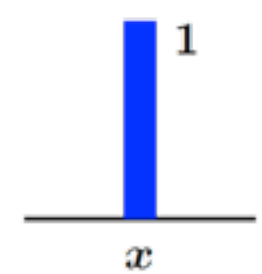
Monads for Branching

A *monad* is a functor T equipped with	$\mathcal{P}$ powerset monad	$\mathcal{D}$ powerset monad
unit $X \xrightarrow{\eta} TX$	singleton $X \longrightarrow \mathcal{P}X$ $x \longmapsto \{x\}$	point-mass distr. $X \longrightarrow \mathcal{D}X$ $x \longmapsto$
multiplication $TTX \xrightarrow{\mu} TX$	union $\mathcal{PP}X \longrightarrow \mathcal{P}X$ $\{\{x,y\},\{z\}\} \longmapsto \{x,y,z\}$	$\mathcal{DD}X \longrightarrow \mathcal{D}X$ $\begin{bmatrix} \begin{bmatrix} x \mapsto 1/2 \\ y \mapsto 1/2 \end{bmatrix} & \mapsto & 1/3 \\ [z \mapsto 1] & \mapsto & 2/3 \end{bmatrix} \overset{\mu}{\longmapsto} \begin{bmatrix} x \mapsto 1/6 \\ y \mapsto 1/6 \\ z \mapsto 2/3 \end{bmatrix}$

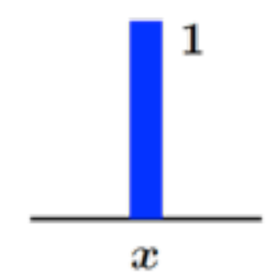
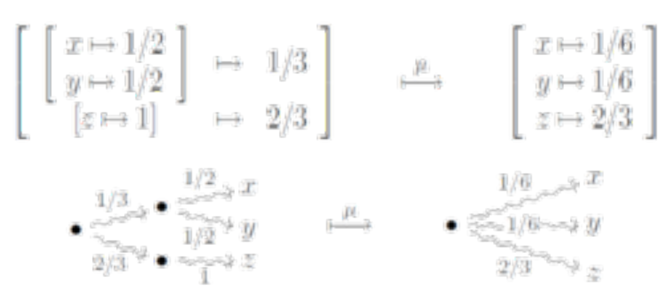
Monads for Branching

A *monad* is a functor T equipped with	$\mathcal{P}$ powerset monad	$\mathcal{D}$ powerset monad
unit $X \xrightarrow{\eta} TX$	singleton $X \longrightarrow \mathcal{P}X$ $x \longmapsto \{x\}$	point-mass distr. $X \longrightarrow \mathcal{D}X$ $x \longmapsto$
multiplication $TTX \xrightarrow{\mu} TX$	union $\mathcal{P}\mathcal{P}X \longrightarrow \mathcal{P}X$ $\{\{x, y\}, \{z\}\} \longmapsto \{x, y, z\}$ 	$\mathcal{D}\mathcal{D}X \longrightarrow \mathcal{D}X$

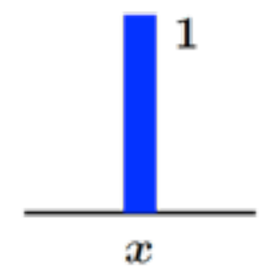
Monads for Branching

A *monad* is a functor T equipped with	$\mathcal{P}$ powerset monad	$\mathcal{D}$ powerset monad
unit $X \xrightarrow{\eta} TX$	singleton $X \longrightarrow \mathcal{P}X$ $x \longmapsto \{x\}$	point-mass distr. $X \longrightarrow \mathcal{D}X$ $x \longmapsto$ 
multiplication $TTX \xrightarrow{\mu} TX$	union $\mathcal{P}\mathcal{P}X \longrightarrow \mathcal{P}X$ $\{\{x,y\},\{z\}\} \longmapsto \{x,y,z\}$ 	$\mathcal{D}\mathcal{D}X \longrightarrow \mathcal{D}X$ $\left[\begin{smallmatrix}\left[\begin{smallmatrix}x\mapsto 1/2\\ y\mapsto 1/2\end{smallmatrix}\right] & \mapsto 1/3\\ [z\mapsto 1] & \mapsto 2/3\end{smallmatrix}\right] \overset{\mu}{\longmapsto} \left[\begin{smallmatrix}x\mapsto 1/6\\ y\mapsto 1/6\\ z\mapsto 2/3\end{smallmatrix}\right]$ 

Monads for Branching

A *monad* is a functor T equipped with	$\mathcal{P}$ powerset monad	$\mathcal{D}$ powerset monad	
unit $X \xrightarrow{\eta} TX$	singleton $X \longrightarrow \mathcal{P}X$ $x \longmapsto \{x\}$	point-mass distr. $X \longrightarrow \mathcal{D}X$ $x \longmapsto$ 	*trivial branching*
multiplication $TTX \xrightarrow{\mu} TX$	union $\mathcal{PP}X \longrightarrow \mathcal{P}X$ $\{\{x,y\},\{z\}\} \longmapsto \{x,y,z\}$ 	$\mathcal{DD}X \longrightarrow \mathcal{D}X$ 	

Monads for Branching

A *monad* is a functor T equipped with	$\mathcal{P}$ powerset monad	$\mathcal{D}$ powerset monad	
unit $X \xrightarrow{\eta} TX$	singleton $X \longrightarrow \mathcal{P}X$ $x \longmapsto \{x\}$	point-mass distr. $X \longrightarrow \mathcal{D}X$ $x \longmapsto$ 	*trivial branching*
multiplication $TTX \xrightarrow{\mu} TX$	union $\mathcal{PP}X \longrightarrow \mathcal{P}X$ $\{\{x,y\},\{z\}\} \longmapsto \{x,y,z\}$ 	$\mathcal{DD}X \longrightarrow \mathcal{D}X$ $\left[\begin{smallmatrix}\left[\begin{smallmatrix}x\mapsto 1/2\\ y\mapsto 1/2\end{smallmatrix}\right] & \mapsto 1/3\\ [z\mapsto 1] & \mapsto 2/3\end{smallmatrix}\right] \overset{\mu}{\mapsto} \left[\begin{smallmatrix}x\mapsto 1/6\\ y\mapsto 1/6\\ z\mapsto 2/3\end{smallmatrix}\right]$ 	*unfold internal branching*

Coalgebraic Trace Semantics

Theorem. Let T be a commutative monad s.t. $\mathcal{K}\ell(T)$ is **Cppo**-enriched.
A final coalgebra in $\mathcal{K}\ell(T)$ is induced by an initial algebra in **Sets**.

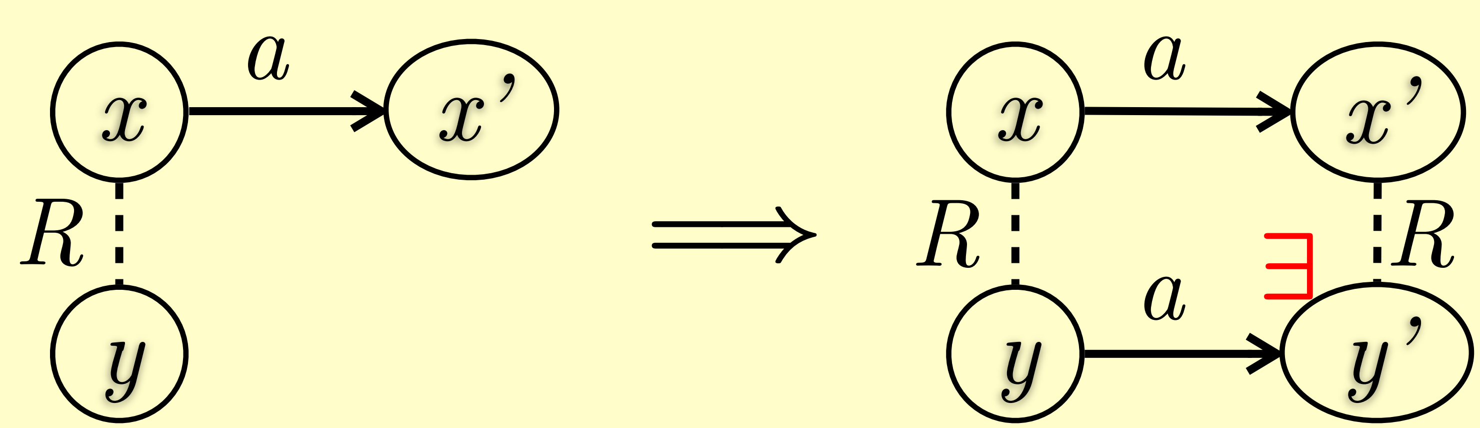
Observation. Such a final coalgebra in $\mathcal{K}\ell(T)$ captures "trace semantics."

$$\begin{array}{ccc} FX & \xrightarrow{F(\mathsf{tr}(c))} & FA \\ {\scriptstyle c}\uparrow & & \uparrow{\scriptstyle \text{final}} \\ X & \xrightarrow[\mathsf{tr}(c)]{} & Z \end{array} \quad \text{in } \mathcal{K}\ell(T)$$

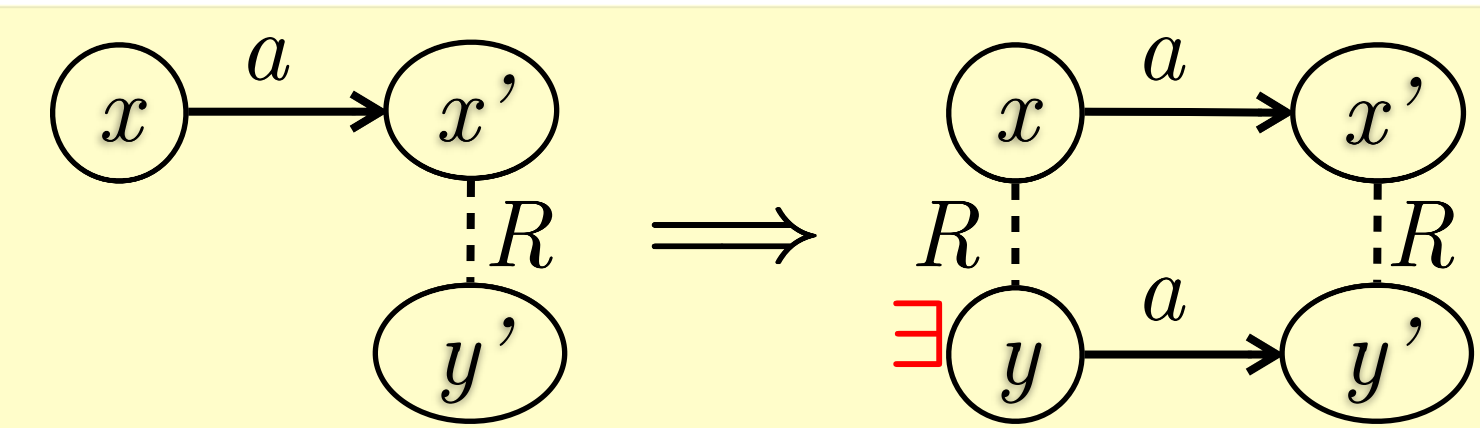
Forward/Backward Simulation

Forward simulation

A relation R between states of two systems, s.t.



Backward simulation



Lynch-Vaandrager, Inf. Comp. 1995

Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

"trace inclusion"

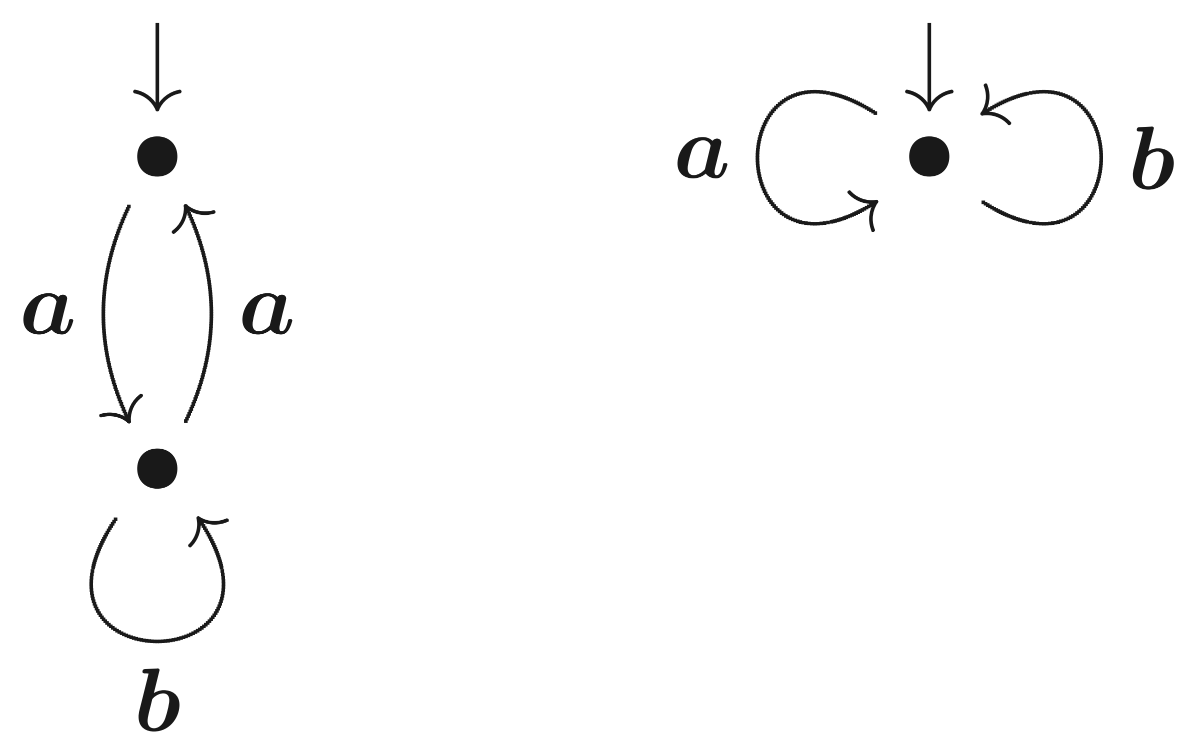
Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

"trace inclusion"



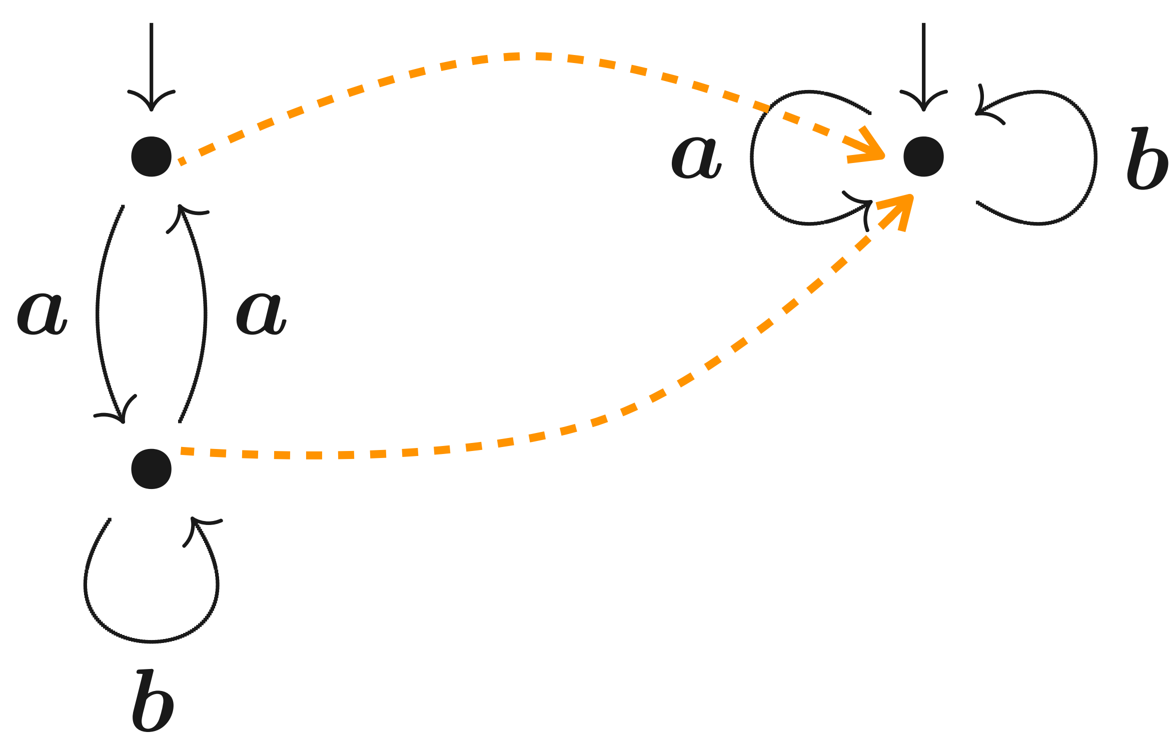
Forward/Backward Simulation

Soundness theorem

If there is a fwd. or bwd. simulation from $\mathcal{S}$ to $\mathcal{T}$, then

$$\mathbf{tr}(\mathcal{S}) \subseteq \mathbf{tr}(\mathcal{T})$$

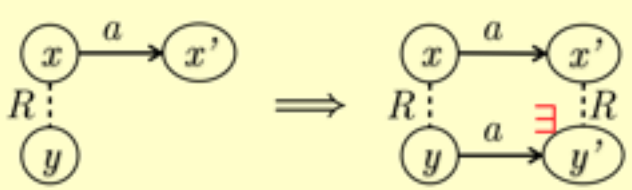
"trace inclusion"



Coalgebra Transfers Definitions & Results

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

Coalgebra Transfers Definitions & Results

In $\mathcal{K}\ell(T)$

$$\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsupseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$$

forward simulation

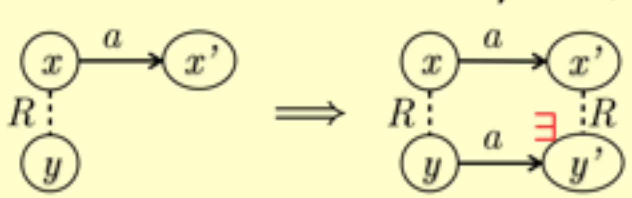
$$\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsubseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$$

backward simulation

$T = \mathcal{P}$

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

Coalgebra Transfers Definitions & Results

In $\mathcal{K}\ell(T)$

$$\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsupseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$$

forward simulation

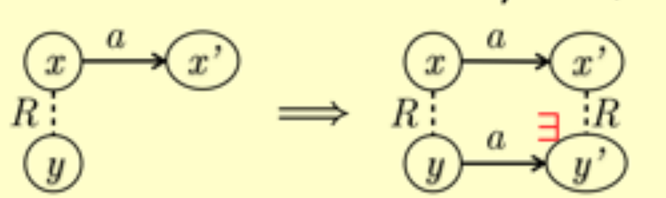
$$\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & \sqsubseteq & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$$

backward simulation

$T = \mathcal{P}$

Forward simulation

A relation R between states of two systems, s.t.



Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

$T = \mathcal{D}$

Forward simulation

Definition. Let $\mathcal{X} = (X, x_0, c)$ and $\mathcal{Y} = (Y, y_0, d)$ be GPAs. A *forward (Kleisli) simulation* from $\mathcal{X}$ to $\mathcal{Y}$ is a function $f : Y \to \mathcal{D}X$ which satisfies the following (in)equalities.

$$\Pr[y_0 \dashrightarrow x_0] = 1 \qquad \text{(Init)}$$

$$\textstyle\sum_{x \in X} \Pr[y \dashrightarrow x \to \checkmark] \le \Pr[y \to \checkmark] \quad \text{for each } y \in Y \qquad \text{(Term)}$$

$$\textstyle\sum_{x \in X} \Pr[y \dashrightarrow x \xrightarrow{a} x'] \le \sum_{y' \in Y} \Pr[y \xrightarrow{a} y' \dashrightarrow x'] \quad \text{for each } y \in Y,\ a \in \mathbf{Act} \text{ and } x' \in X \qquad \text{(Act)}$$

Soundness theorem

Existence of fwd./bwd. simulation
⇒ trace incl.

Case Study: Probabilistic Anonymity

Simulation-based verif. method for non-deterministic anonymity

Kawabe-Mano-Sakurada-Tsukada, *Inf. Proc. Let.* 2007

Case Study: Probabilistic Anonymity

Simulation-based verif. method for non-deterministic anonymity

Kawabe-Mano-Sakurada-Tsukada, *Inf. Proc. Let.* 2007

↓

Simulation-based verif. method for probabilistic anonymity

Hasuo-Kawabe-Sakurada, *Theor. Comp. Sci.* 2010

Case Study: Probabilistic Anonymity

Simulation-based verif. method for non-deterministic anonymity

Kawabe-Mano-Sakurada-Tsukada, *Inf. Proc. Let.* 2007

$T = \mathcal{P}$

Simulation-based verif. method for probabilistic anonymity

Hasuo-Kawabe-Sakurada, *Theor. Comp. Sci.* 2010

Case Study: Probabilistic Anonymity

$T = \mathcal{P}$

Simulation-based verif. method for
non-deterministic anonymity

Kawabe-Mano-Sakurada-Tsukada, *Inf. Proc. Let.* 2007

↓

$T = \mathcal{D}$

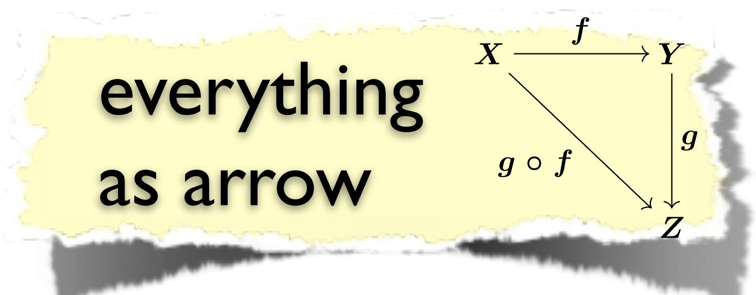
Simulation-based verif. method for
probabilistic anonymity

Hasuo-Kawabe-Sakurada, *Theor. Comp. Sci.* 2010

4 Wrapping Up

Conclusions

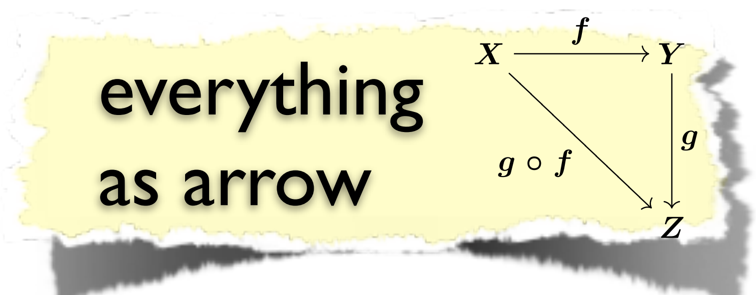
- *Mathematics for systems* via coalgebras
  - The language of *category theory*

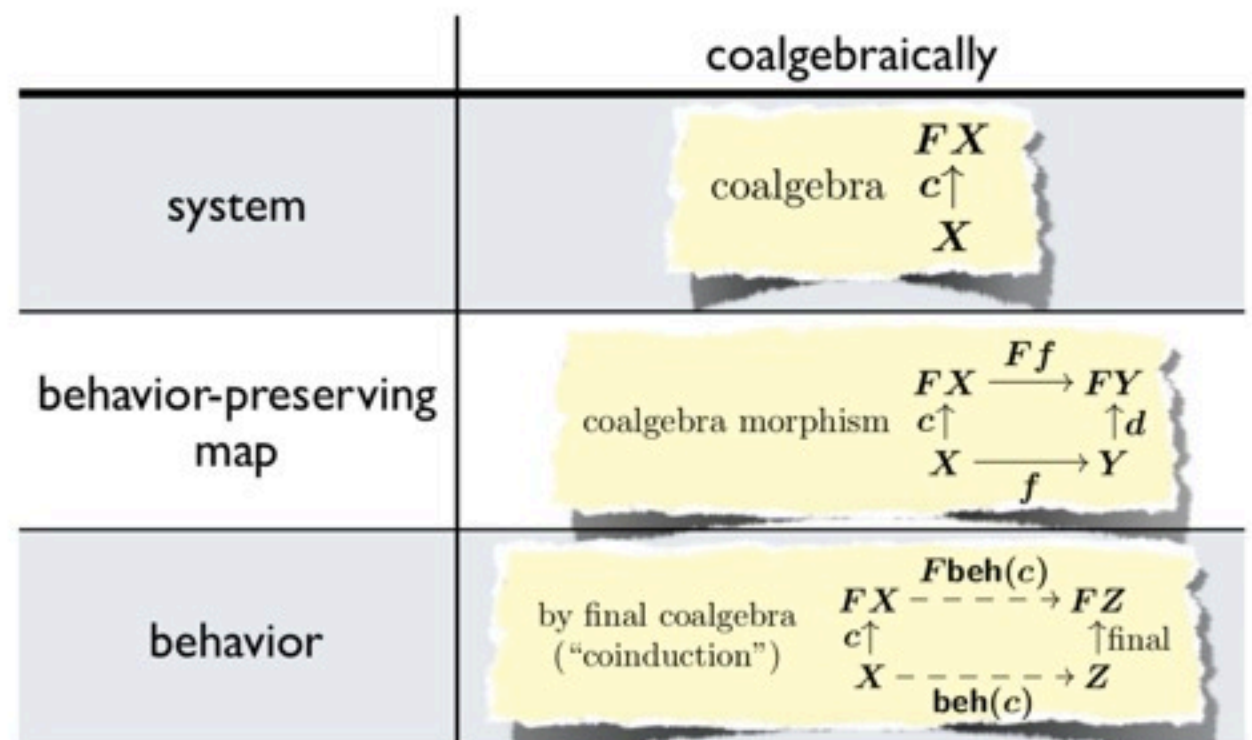


  - "System as coalgebra": robust under change of base categories

Conclusions

- *Mathematics for systems* via coalgebras
  - The language of *category theory*
  - "System as coalgebra": robust under change of base categories

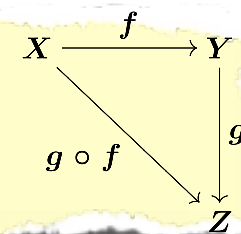




Conclusions

- *Mathematics for systems* via coalgebras
  - The language of *category theory*
  - "System as coalgebra": robust under change of base categories

everything as arrow



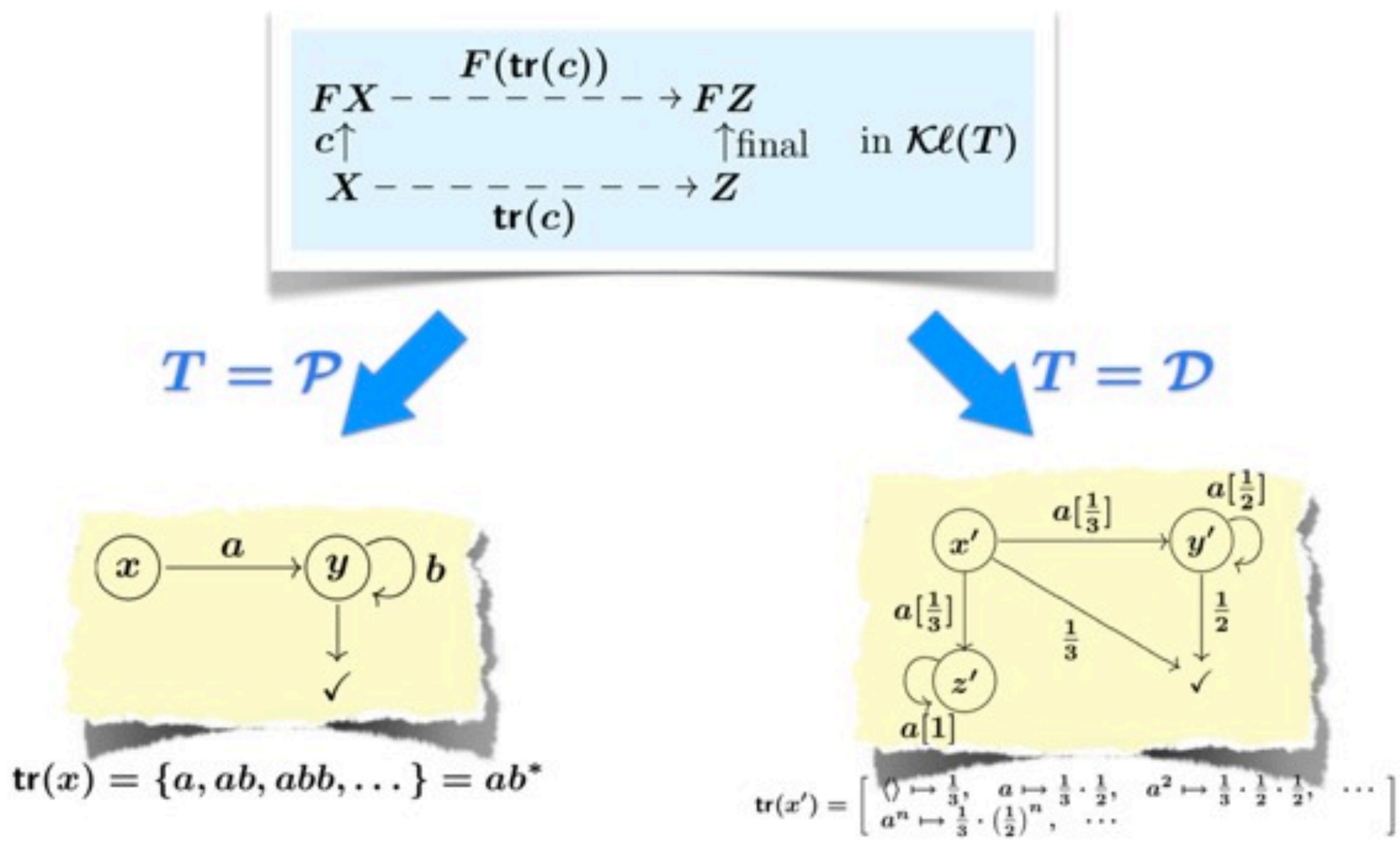
	coalgebraically
system	coalgebra $\begin{array}{c} FX \\ c\uparrow \\ X \end{array}$
behavior-preserving map	coalgebra morphism $\begin{array}{ccc} FX & \xrightarrow{Ff} & FY \\ c\uparrow & & \uparrow d \\ X & \xrightarrow[f]{} & Y \end{array}$
behavior	by final coalgebra ("coinduction") $\begin{array}{ccc} FX & \xrightarrow{F\mathsf{beh}(c)} & FZ \\ c\uparrow & & \uparrow\text{final} \\ X & \xrightarrow[\mathsf{beh}(c)]{} & Z \end{array}$

bisimilarity in Sets

trace semantics in $Kl(T)$

Conclusions

- Abstraction & genericity (& joy)
  - Generic theory, transfer of results



Conclusions

- Young field with exciting topics and vibrant community. Join us!

References

- General introduction
  - Bart Jacobs, Introduction to Coalgebra. Towards Mathematics of States and Observations. Two-thirds of a book in preparation, available online.
  - Jan Rutten: Universal coalgebra: a theory of systems. Theor. Comput. Sci. 249(1): 3-80 (2000)
  - Dirk Pattinson. An introduction to the theory of coalgebras, 2003. Lecture Notes, Second North American Summer School on Logic, Language and Information.
- Our work
  - Ichiro Hasuo, Bart Jacobs and Ana Sokolova. Generic Trace Semantics via Coinduction. Logical Methods in Computer Science, 3(4:11), 2007.
  - Ichiro Hasuo. Generic Forward and Backward Simulations. International Conference on Concurrency Theory (CONCUR 2006). LNCS 4137, p.406-420. Springer-Verlag.
  - Ichiro Hasuo. Generic Forward and Backward Simulations II: Probabilistic Simulation. To appear in Proc. CONCUR 2010, LNCS.
  - Ichiro Hasuo, Yoshinobu Kawabe and Hideki Sakurada. Probabilistic Anonymity via Coalgebraic Simulations. Theoretical Computer Science, 411(22-24):2239-2259, 2010.
- Ichiro Hasuo. Tracing Anonymity via Coalgebras. PhD Thesis, Radboud University Nijmegen, 2008. Hardcopies available on request.