Operational Semantics Coalgebraically

Bartek Klin
University of Cambridge, Warsaw University

CMCS, Paphos, 27/03/10

Summary

Structural Operational Semantics

is about

Defining transition relations by induction

Summary

Structural Operational Semantics

is about

~~Defining transition relations by induction~~

Summary

Structural Operational Semantics

is about

Distributing syntax over behaviour

I.

FROM COINDUCTIVE DEFINITIONS TO BIALGEBRAS

A coinductive definition

Example: streams

$$BX = A \times X$$

A coinductive definition

Example: streams $\qquad Z = A^\omega$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

A coinductive definition

Example: streams $$Z = A^\omega$$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

Define: $\mathtt{alt} : Z^2 \to Z$

A coinductive definition

Example: streams

$$Z = A^\omega$$

$$BX = A \times X$$

$$z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

$$a_1, a_2, a_3, a_4, a_5, \ldots$$

$$b_1,\ b_2,\ b_3,\ b_4,\ b_5, \ldots$$

Define: $\mathtt{alt} : Z^2 \to Z$

A coinductive definition

Example: streams

$$Z = A^\omega$$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

$$a_1, a_2, a_3, a_4, a_5, \ldots$$

$$b_1, b_2, b_3, b_4, b_5, \ldots$$

Define: $\mathtt{alt} : Z^2 \to Z$

A coinductive definition

Example: streams

$$Z = A^\omega$$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

Define: $\mathtt{alt} : Z^2 \to Z$

$$\frac{\begin{array}{c} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_2, a_3, b_4, a_5, \ldots}\ \mathtt{alt}$$

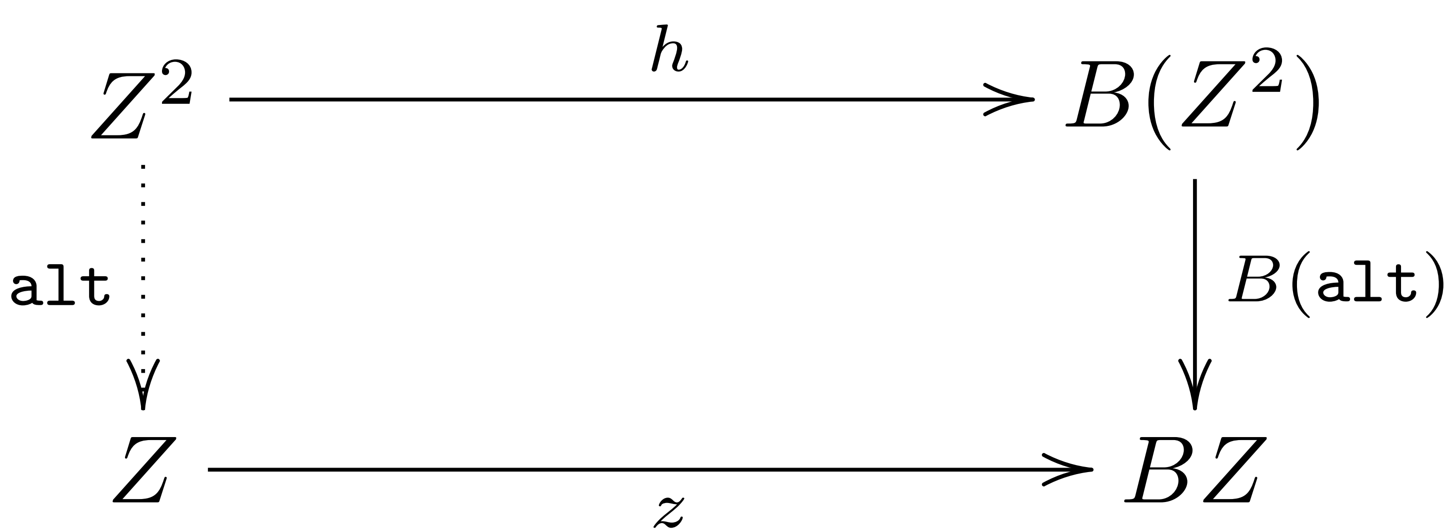
A coinductive definition

Example: streams $\qquad Z = A^\omega$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

Define: $\mathtt{alt} : Z^2 \to Z$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_2, a_3, b_4, a_5, \ldots}\ \mathtt{alt}$$



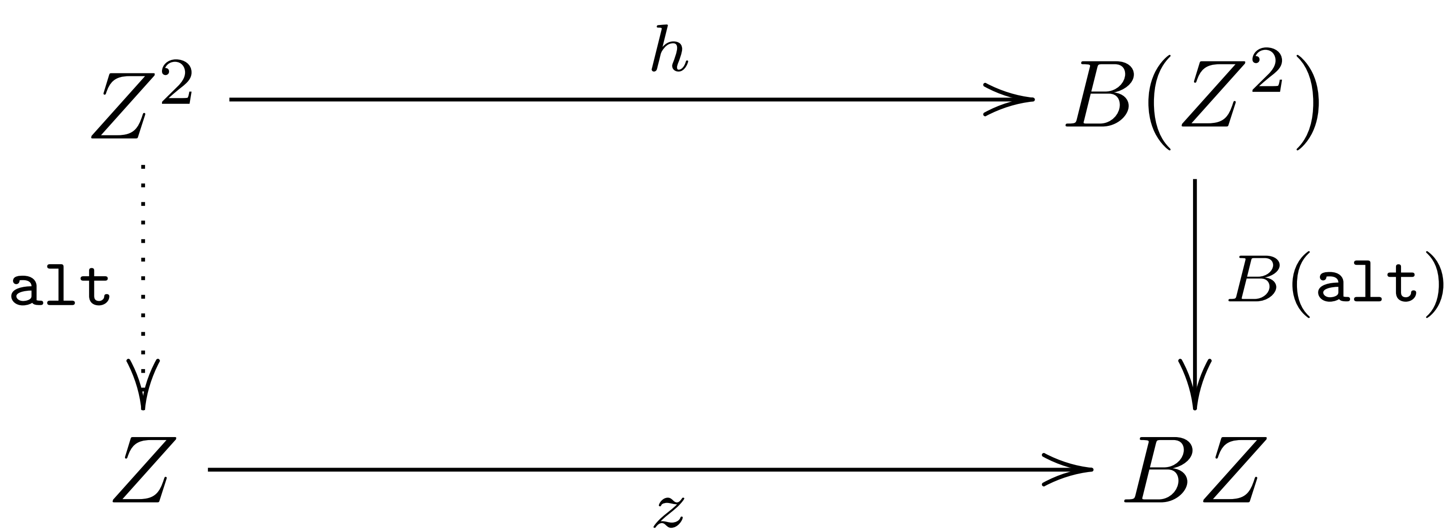
A coinductive definition

Example: streams $\qquad Z = A^\omega$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_2, a_3, b_4, a_5, \ldots}\ \mathtt{alt}$$

Define: $\mathtt{alt} : Z^2 \to Z$



$$h(\sigma, \tau) \;=\; \mathtt{hd}(\sigma), (\mathtt{tl}(\tau), \mathtt{tl}(\sigma))$$

A coinductive definition

Example: streams

$$Z = A^\omega$$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

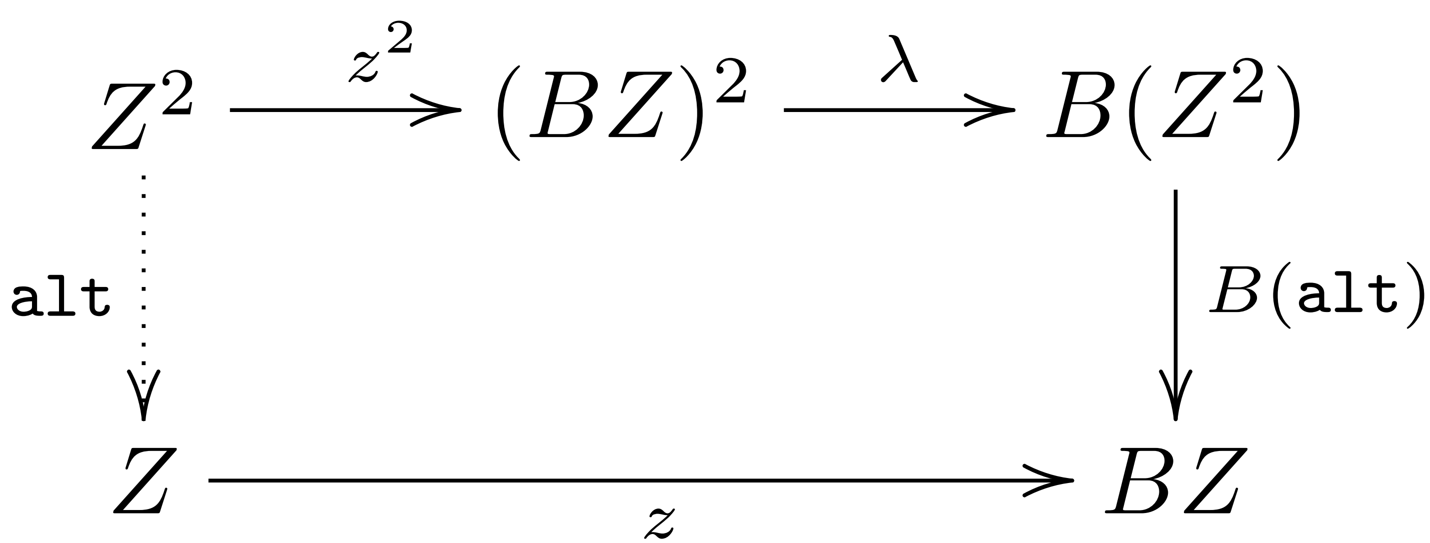
$$a_1, a_2, a_3, a_4, a_5, \ldots$$

$$b_1, b_2, b_3, b_4, b_5, \ldots$$

$$\overline{} \ \mathtt{alt}$$

$$a_1, b_2, a_3, b_4, a_5, \ldots$$

Define: $\mathtt{alt} : Z^2 \to Z$



$$h(\sigma, \tau)$$

$$\|$$

$$\mathtt{hd}(\sigma), (\mathtt{tl}(\tau), \mathtt{tl}(\sigma))$$

A coinductive definition

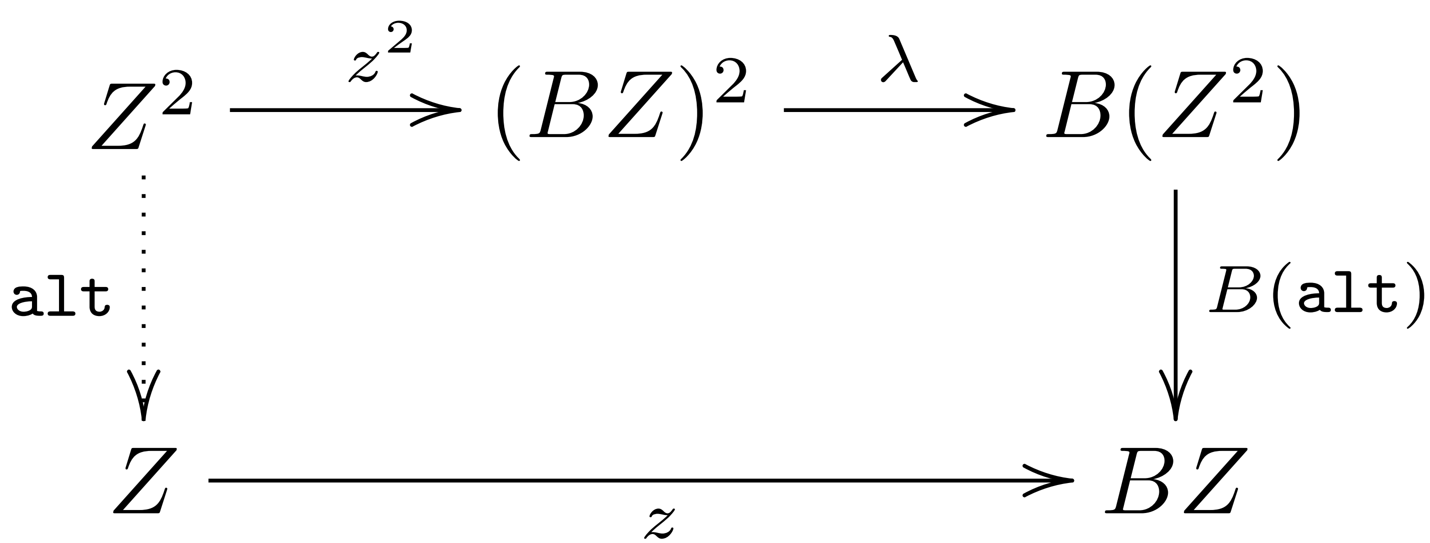
Example: streams

$$Z = A^\omega$$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_2, a_3, b_4, a_5, \ldots}\ \mathtt{alt}$$

Define: $\mathtt{alt} : Z^2 \to Z$



$$\lambda(a, \sigma, b, \tau) \;\;=\;\; a, (\tau, \sigma)$$

A coinductive definition

Example: streams $\qquad Z = A^\omega$

$$BX = A \times X \qquad z = \langle \mathtt{hd}, \mathtt{tl} \rangle : Z \to BZ$$

Define: $\mathtt{alt} : Z^2 \to Z$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_2, a_3, b_4, a_5, \ldots}\ \mathtt{alt}$$

$$\begin{array}{ccccc} Z^2 & \xrightarrow{z^2} & (BZ)^2 & \xrightarrow{\lambda} & B(Z^2) \\ \mathtt{alt} \downarrow & & & & \downarrow B(\mathtt{alt}) \\ Z & & \xrightarrow{\quad z \quad} & & BZ \end{array}$$

$$\lambda(a, \sigma, b, \tau) = a, (\tau, \sigma)$$

$\lambda : (B-)^2 \Longrightarrow B(-)^2$ is natural.

Another example

$BX = A \times X$ $\qquad$ $Z = A^\omega$

Another example

$BX = A \times X$ $\qquad$ $Z = A^{\omega}$

Assume: $+ : A^2 \to A$

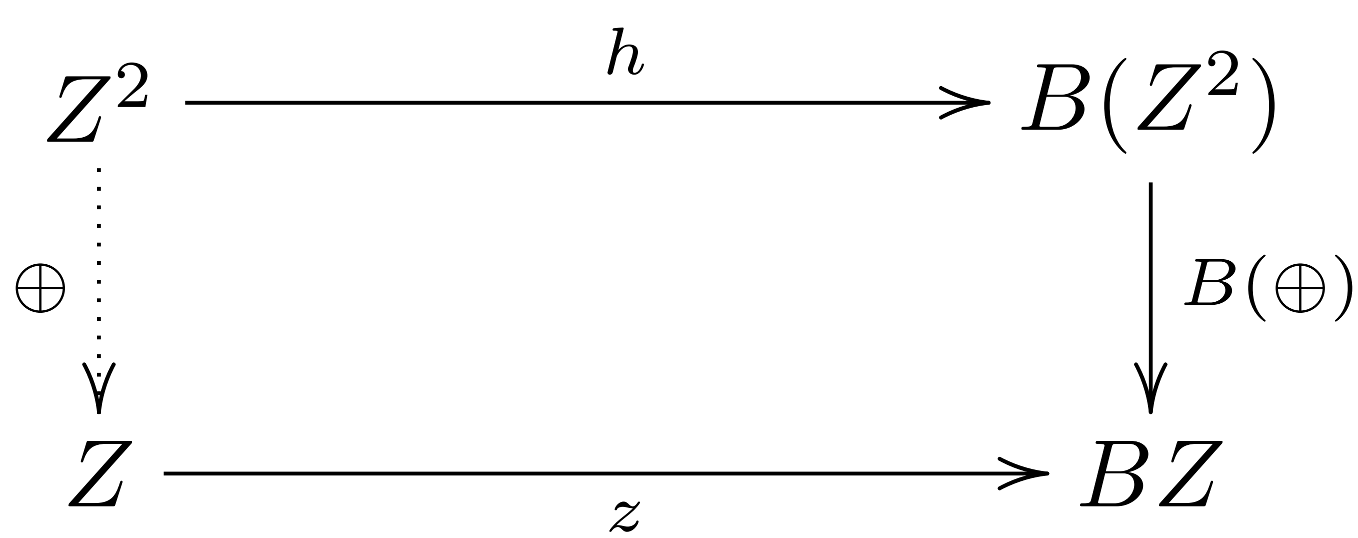
Define: $\oplus : Z^2 \to Z$ $\qquad$ as $+$ pointwise

Another example

$BX = A \times X$ $\qquad\qquad Z = A^\omega$

Assume: $+ : A^2 \to A$

Define: $\oplus : Z^2 \to Z$ $\qquad$ as $+$ pointwise

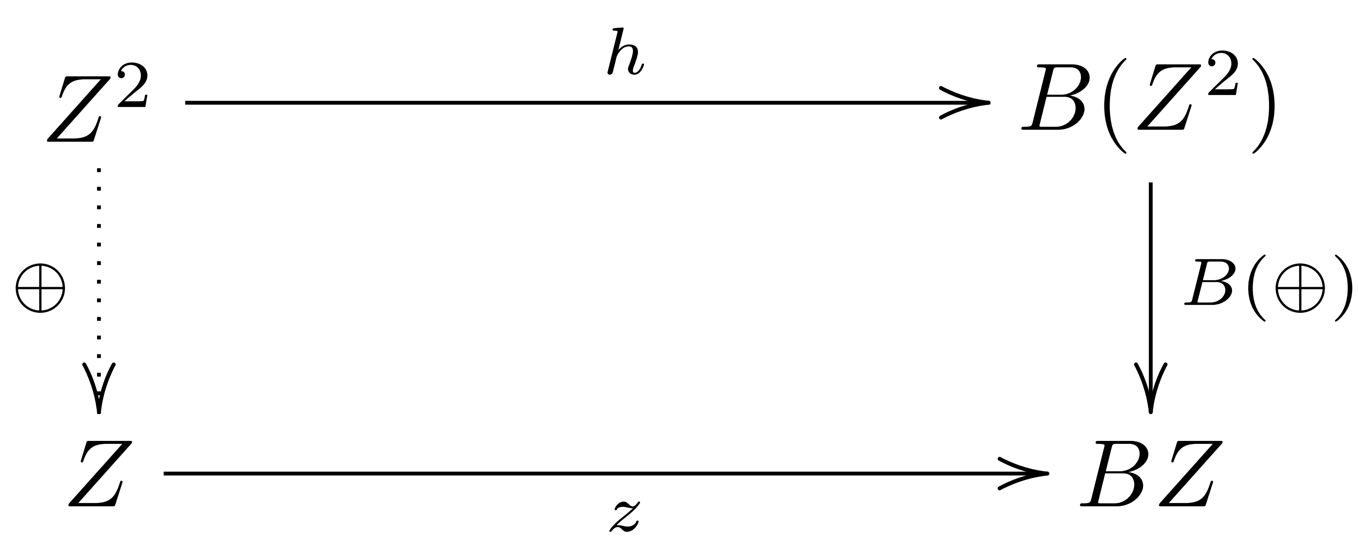


Another example

$BX = A \times X$ $\qquad\qquad$ $Z = A^\omega$

Assume: $+ : A^2 \to A$

Define: $\oplus : Z^2 \to Z$ $\qquad$ as $+$ pointwise



$$h(\sigma, \tau)$$

$$\|$$

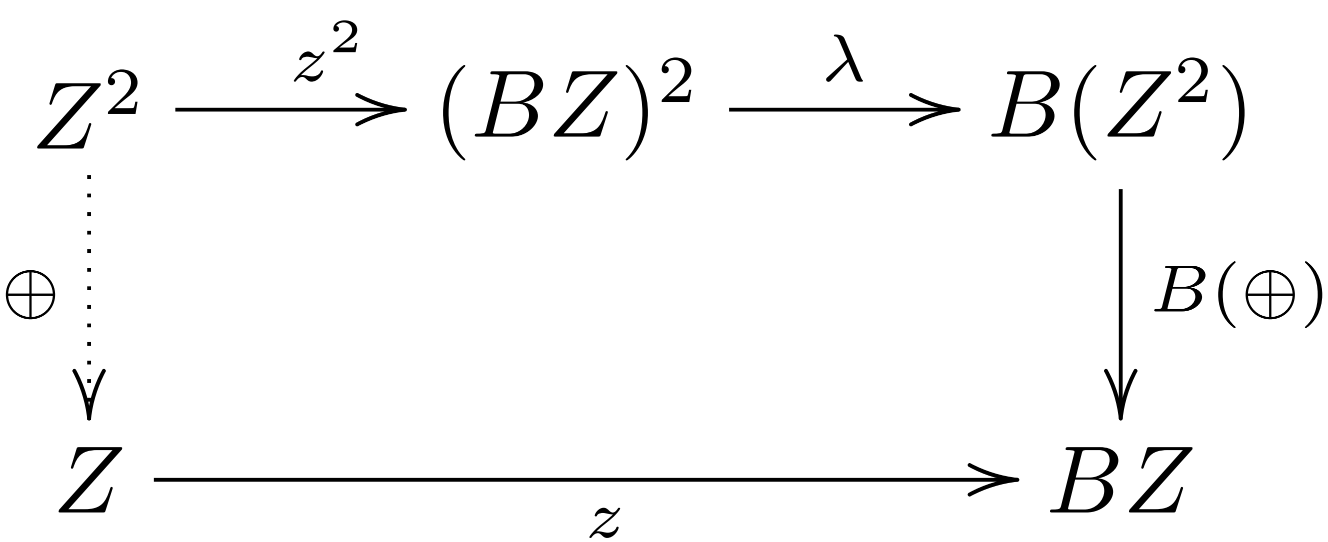
$$\mathtt{hd}(\sigma) + \mathtt{hd}(\tau), (\mathtt{tl}(\sigma), \mathtt{tl}(\tau))$$

Another example

$BX = A \times X$ $\qquad\qquad$ $Z = A^\omega$

Assume: $+ : A^2 \to A$

Define: $\oplus : Z^2 \to Z$ $\qquad$ as $+$ pointwise



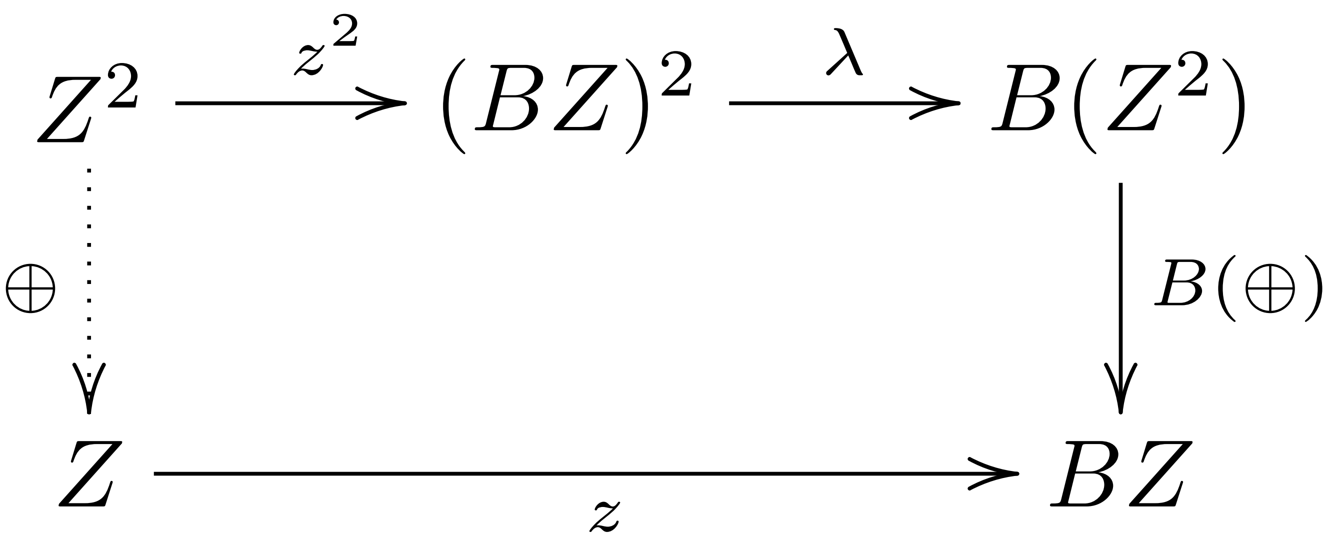
$$h(\sigma, \tau)$$
$$||$$
$$\mathtt{hd}(\sigma) + \mathtt{hd}(\tau), (\mathtt{tl}(\sigma), \mathtt{tl}(\tau))$$

Another example

$BX = A \times X$ $\qquad$ $Z = A^\omega$

Assume: $+ : A^2 \to A$

Define: $\oplus : Z^2 \to Z$ $\qquad$ as $+$ pointwise



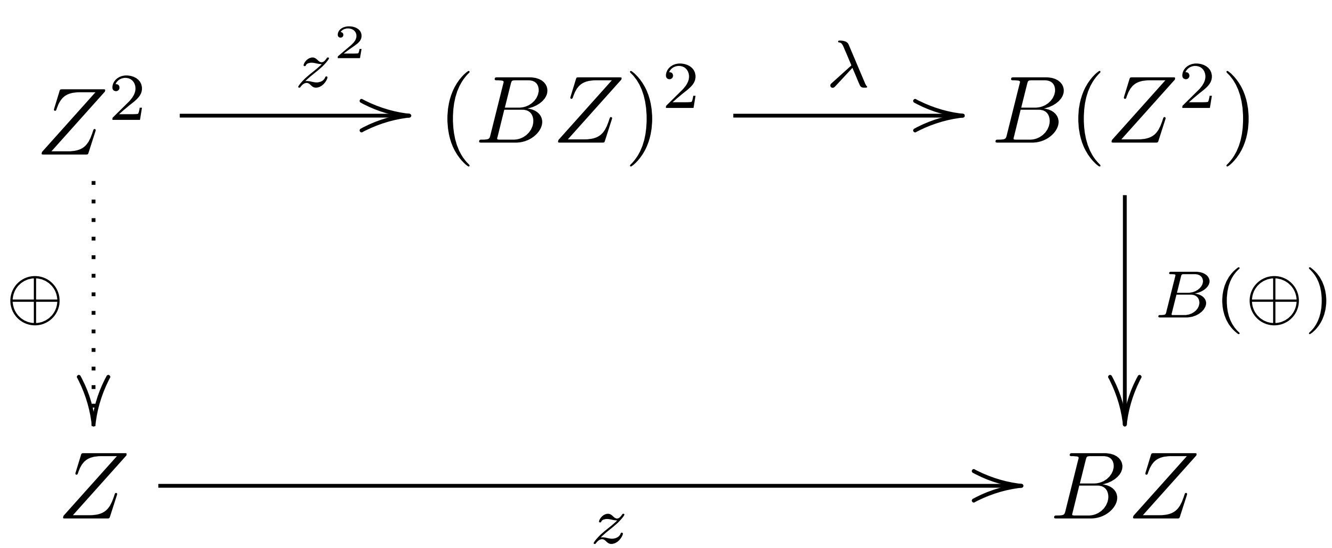
$$\lambda(a, \sigma, b, \tau)$$
$$\|$$
$$a + b, (\sigma, \tau)$$

Another example

$BX = A \times X$ $\qquad Z = A^\omega$

Assume: $+ : A^2 \to A$

Define: $\oplus : Z^2 \to Z$ as $+$ pointwise



$$\lambda(a, \sigma, b, \tau) \;\|\; a + b, (\sigma, \tau)$$

Again, $\lambda : (B-)^2 \Longrightarrow B(-)^2$ natural.

Distributive laws

To define $g : \Sigma Z \to Z$ provide $\lambda : \Sigma B \Longrightarrow B\Sigma$

Distributive laws

To define $g : \Sigma Z \to Z$ provide $\lambda : \Sigma B \Longrightarrow B\Sigma$

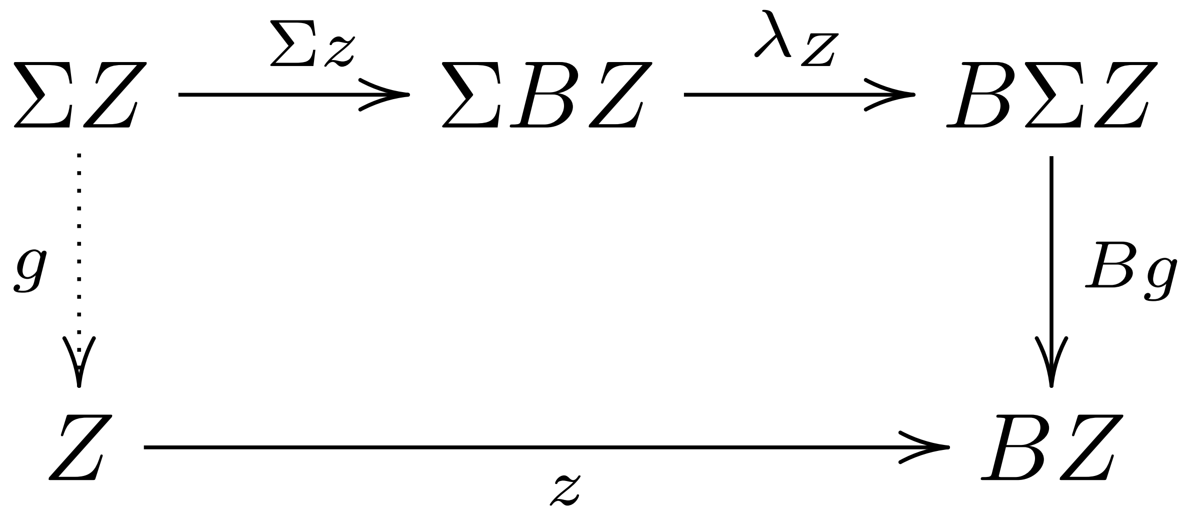
a distributive law

Distributive laws

To define $g : \Sigma Z \to Z$ provide $\lambda : \Sigma B \Longrightarrow B\Sigma$

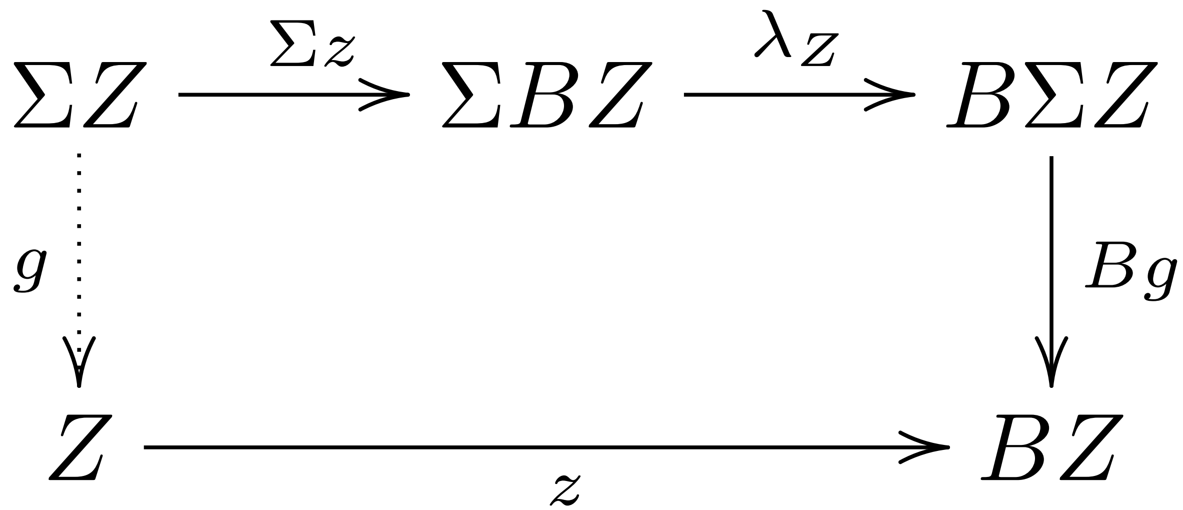
Distributive laws

To define $g : \Sigma Z \to Z$ provide $\lambda : \Sigma B \Longrightarrow B\Sigma$

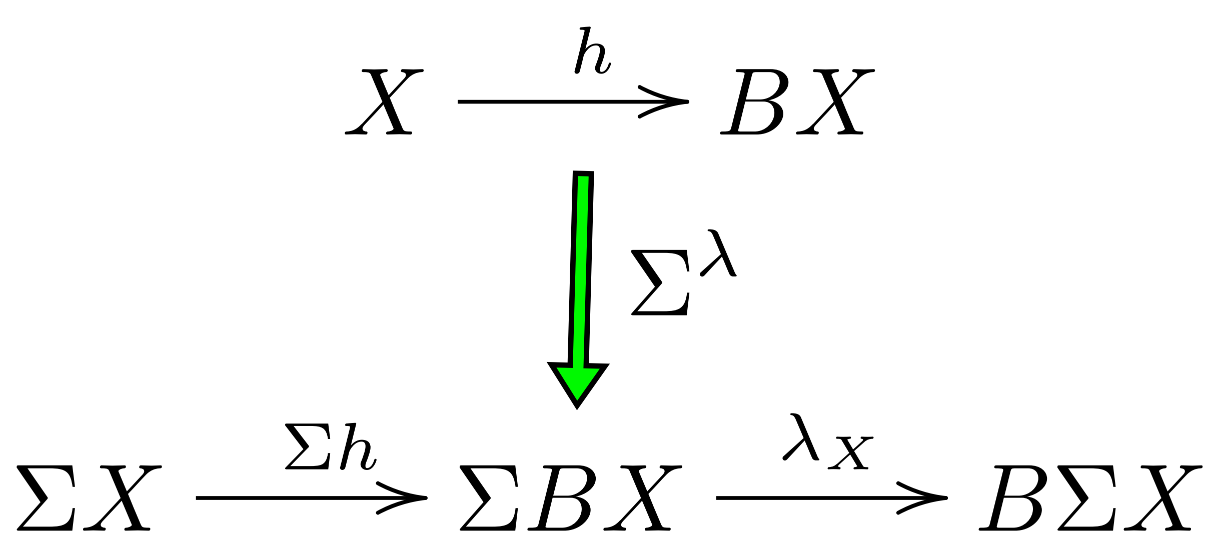


Distributive laws

To define $g : \Sigma Z \to Z$ provide $\lambda : \Sigma B \Longrightarrow B\Sigma$

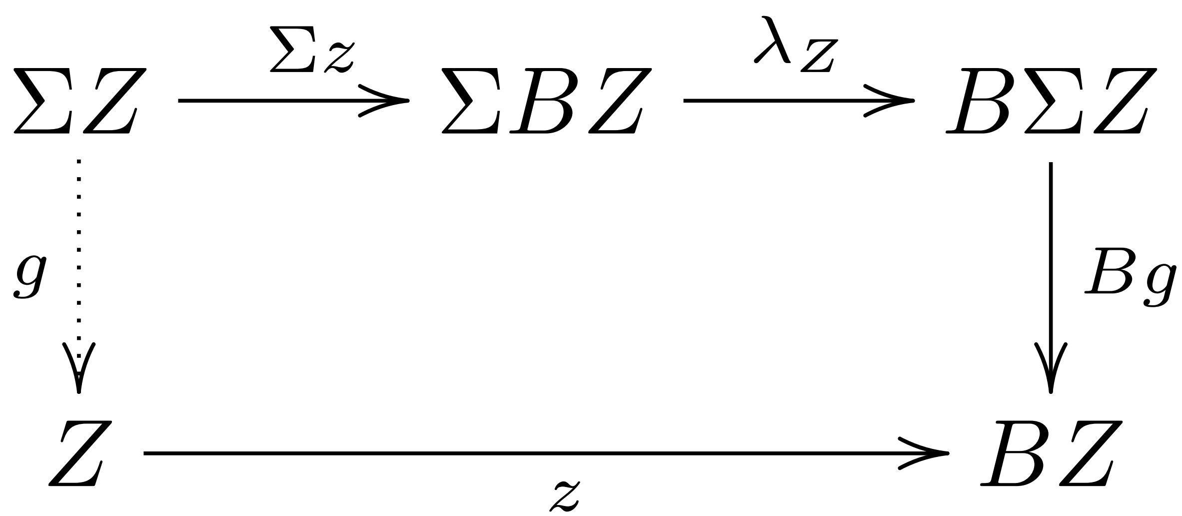


Benefits of naturality:



More benefits

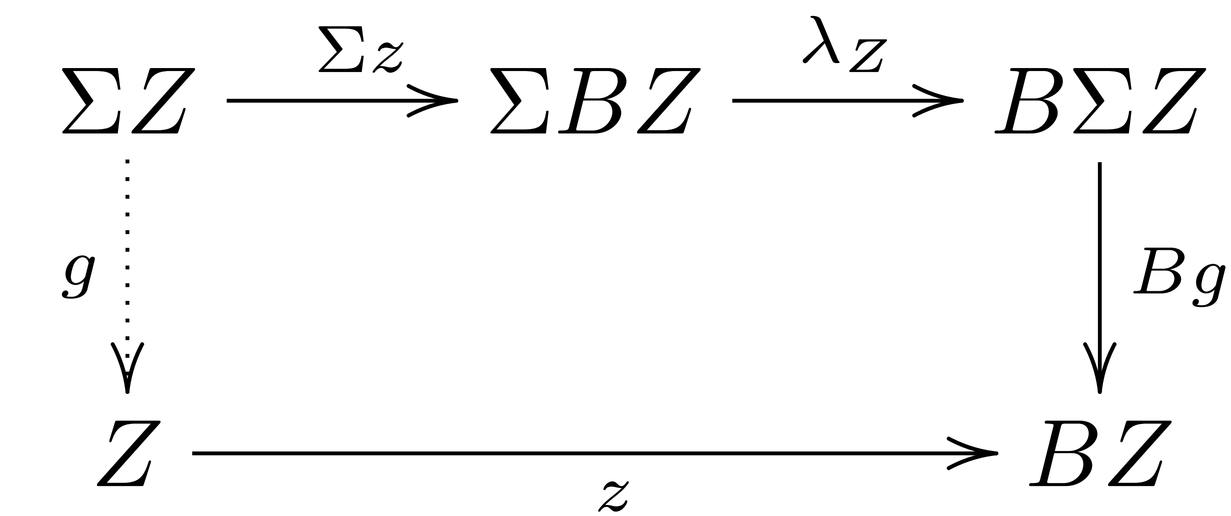
$$\lambda : \Sigma B \Longrightarrow B\Sigma$$



operations
on final B-coalgebras

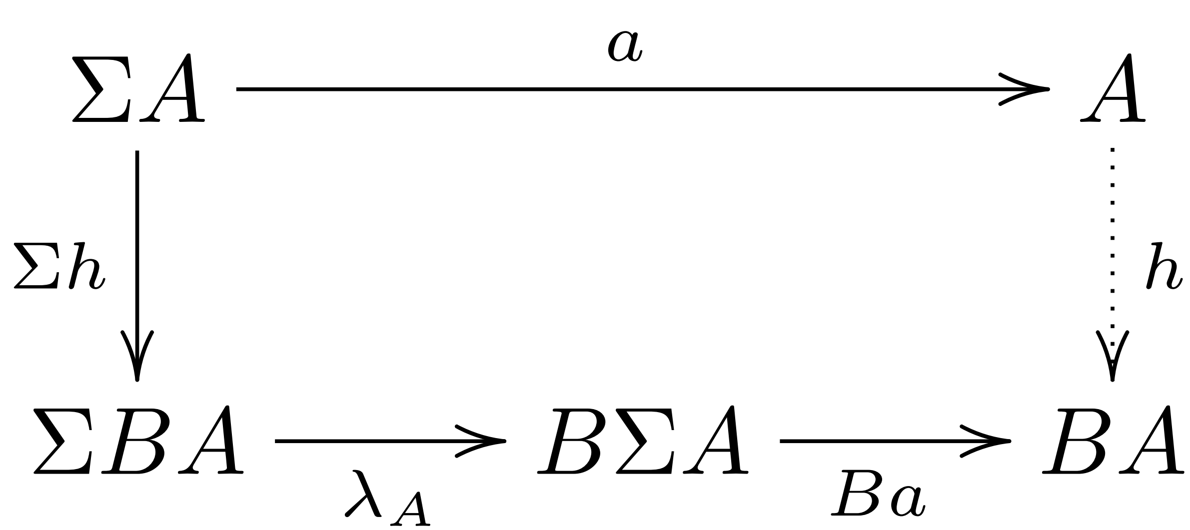
More benefits

$$\lambda : \Sigma B \Longrightarrow B\Sigma$$



operations
on final B-coalgebras

but also:



behaviour
of initial Σ-algebras

Behaviour of terms

$\Sigma X = A + X^2$ $\qquad$ $BX = A \times X$

`alt` binary

`a` constant, for $a \in A$

Behaviour of terms

$\Sigma X = A + X^2$ $\qquad$ $BX = A \times X$

$\texttt{alt}$ binary

$\texttt{a}$ constant, for $a \in A$

$$\lambda_X : \Sigma B X \to B \Sigma X$$

$$\lambda_X^{\texttt{alt}} : (BX)^2 \to B\Sigma X \qquad \lambda_X^{\texttt{a}} : 1 \to B\Sigma X$$

$$(a, x', b, y') \mapsto a, (y', x') \qquad () \mapsto a, ()$$

Behaviour of terms

$\Sigma X = A + X^2$ $BX = A \times X$

$\mathtt{alt}$ binary

$\mathtt{a}$ constant, for $a \in A$

$$\lambda_X : \Sigma B X \to B \Sigma X$$

$$\lambda_X^{\mathtt{alt}} : (BX)^2 \to B\Sigma X \qquad \lambda_X^{\mathtt{a}} : 1 \to B\Sigma X$$

$$(a, x', b, y') \mapsto a, (y', x') \qquad () \mapsto a, ()$$

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{\mathtt{alt}(x, y) \xrightarrow{a} \mathtt{alt}(y', x')} \qquad \frac{}{\mathtt{a} \xrightarrow{a} \mathtt{a}}$$

Behaviour of terms

$\Sigma X = A + X^2$ $\qquad BX = A \times X$

$\mathtt{alt}$ binary

$\mathtt{a}$ constant, for $a \in A$

$$\lambda_X : \Sigma B X \to B \Sigma X$$

$$\lambda_X^{\mathtt{alt}} : (BX)^2 \to B\Sigma X \qquad \lambda_X^{\mathtt{a}} : 1 \to B\Sigma X$$

$$(a, x', b, y') \mapsto a, (y', x') \qquad () \mapsto a, ()$$

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{\mathtt{alt}(x, y) \xrightarrow{a} \mathtt{alt}(y', x')} \qquad \frac{}{\mathtt{a} \xrightarrow{a} \mathtt{a}}$$

Then, e.g., $\mathtt{alt}(\mathtt{a}, \mathtt{b}) \xrightarrow{a} \mathtt{alt}(\mathtt{b}, \mathtt{a})$ etc.

More benefits

$$\lambda : \Sigma B \Longrightarrow B\Sigma$$

$$\begin{array}{ccccc}
\Sigma Z & \xrightarrow{\Sigma z} & \Sigma B Z & \xrightarrow{\lambda_Z} & B\Sigma Z \\
{\scriptstyle g}\downarrow & & & & \downarrow{\scriptstyle Bg} \\
Z & & \xrightarrow[z]{\qquad\qquad} & & BZ
\end{array}$$

operations
on final B-coalgebras

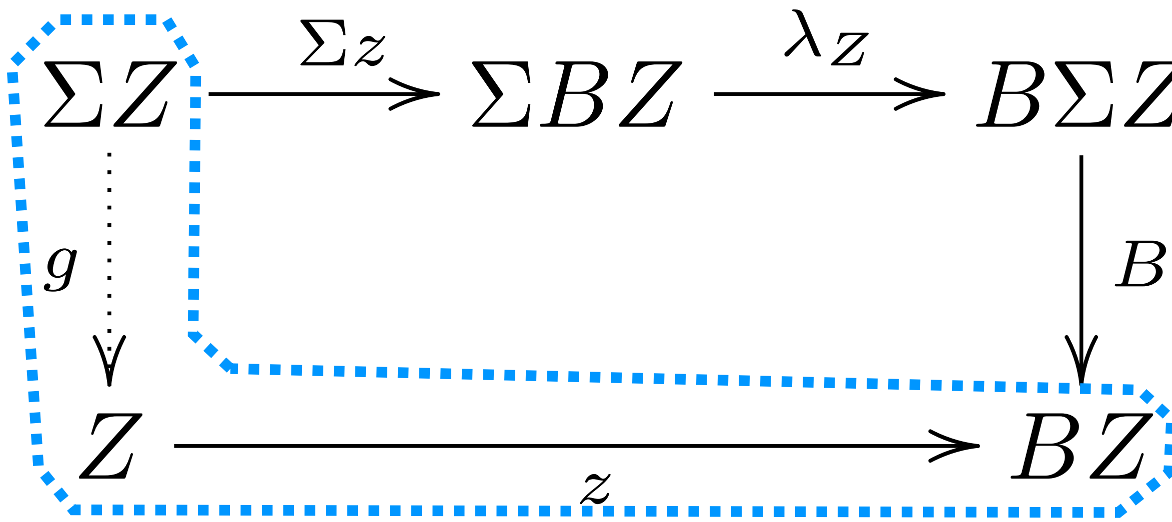
but also:

$$\begin{array}{ccccc}
\Sigma A & & \xrightarrow{a} & & A \\
{\scriptstyle \Sigma h}\downarrow & & & & \downarrow{\scriptstyle h} \\
\Sigma B A & \xrightarrow[\lambda_A]{} & B\Sigma A & \xrightarrow[Ba]{} & BA
\end{array}$$

behaviour
of initial Σ-algebras

More benefits

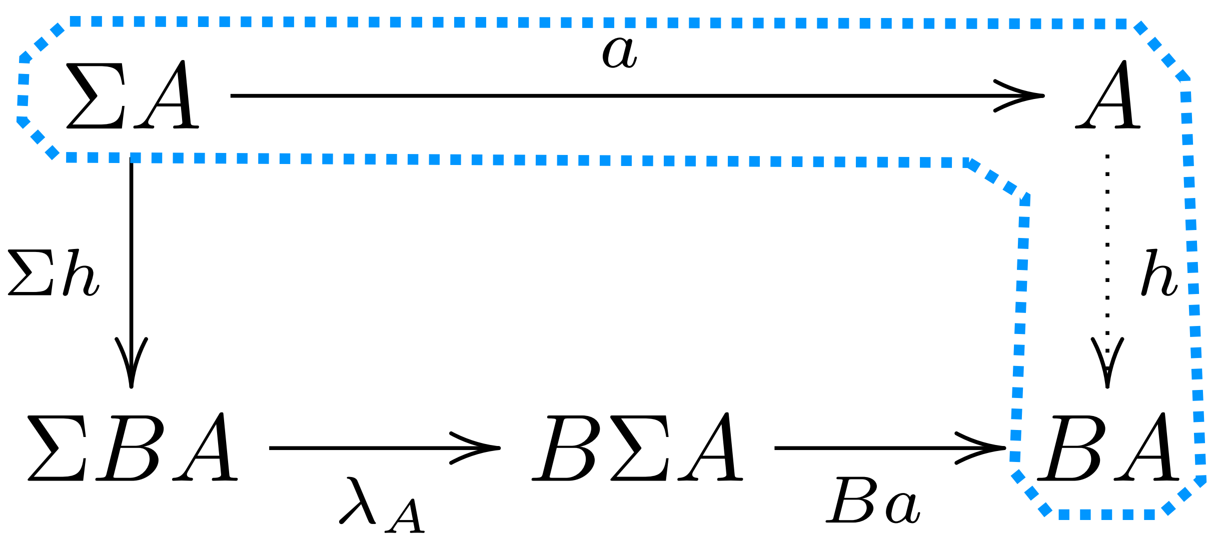
$$\lambda : \Sigma B \Longrightarrow B\Sigma$$



operations

on final B-coalgebras

but also:



behaviour

of initial Σ-algebras

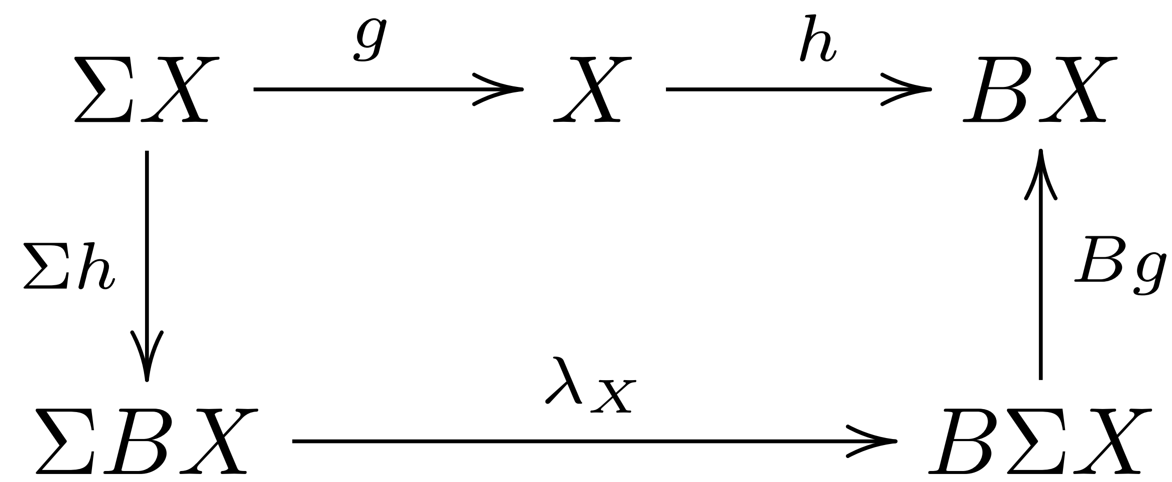
Bialgebras

λ - bialgebra:

$$\begin{CD} \Sigma X @>{g}>> X @>{h}>> BX \\ @V{\Sigma h}VV @. @AA{Bg}A \\ \Sigma BX @>>{\lambda_X}> @. B\Sigma X \end{CD}$$

Bialgebras

λ - bialgebra:



morphisms:

Bialgebras

λ - bialgebra:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ \downarrow{\scriptstyle \Sigma h} & & & & \uparrow{\scriptstyle Bg} \\ \Sigma BX & & \xrightarrow{\lambda_X} & & B\Sigma X \end{array}$$

morphisms:

$$\Sigma X \xrightarrow{g} X \xrightarrow{h} BX$$

$$\Sigma Y \xrightarrow{k} Y \xrightarrow{l} BY$$

Bialgebras

λ - bialgebra:

$$\begin{array}{ccccc}
\Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\
\Sigma h \downarrow & & & & \uparrow Bg \\
\Sigma BX & & \xrightarrow{\lambda_X} & & B\Sigma X
\end{array}$$

morphisms:

$$\begin{array}{ccccc}
\Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\
 & & \downarrow f & & \\
\Sigma Y & \xrightarrow{k} & Y & \xrightarrow{l} & BY
\end{array}$$

Bialgebras

λ - bialgebra:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ {\scriptstyle \Sigma h}\downarrow & & & & \uparrow{\scriptstyle Bg} \\ \Sigma BX & & \xrightarrow{\lambda_X} & & B\Sigma X \end{array}$$

morphisms:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ {\scriptstyle \Sigma f}\downarrow & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle Bf} \\ \Sigma Y & \xrightarrow{k} & Y & \xrightarrow{l} & BY \end{array}$$

Bialgebras

λ - bialgebra:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ \downarrow{\scriptstyle \Sigma h} & & & & \uparrow{\scriptstyle Bg} \\ \Sigma B X & & \xrightarrow{\lambda_X} & & B\Sigma X \end{array}$$

morphisms:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ \downarrow{\scriptstyle \Sigma f} & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle Bf} \\ \Sigma Y & \xrightarrow{k} & Y & \xrightarrow{l} & BY \end{array}$$

Bialgebras are (co)algebras:

$$\lambda\textbf{-bialg} \cong \Sigma^{\lambda}\textbf{-alg}$$

Bialgebras

λ - bialgebra:

$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ \downarrow{\scriptstyle \Sigma h} & & & & \uparrow{\scriptstyle Bg} \\ \Sigma BX & & \xrightarrow{\lambda_X} & & B\Sigma X \end{array}$$

morphisms:

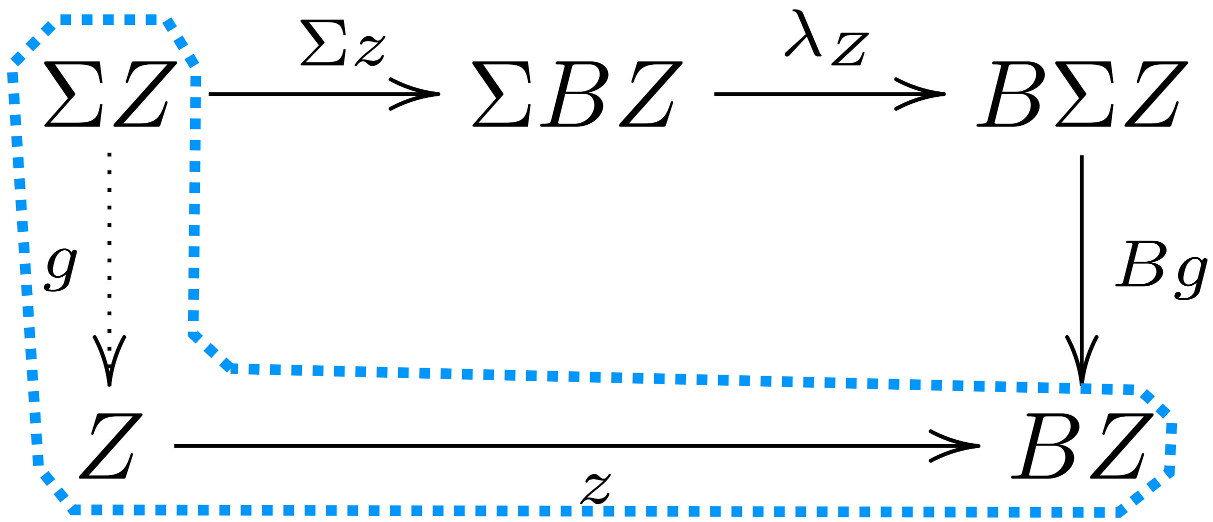
$$\begin{array}{ccccc} \Sigma X & \xrightarrow{g} & X & \xrightarrow{h} & BX \\ \downarrow{\scriptstyle \Sigma f} & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle Bf} \\ \Sigma Y & \xrightarrow{k} & Y & \xrightarrow{l} & BY \end{array}$$

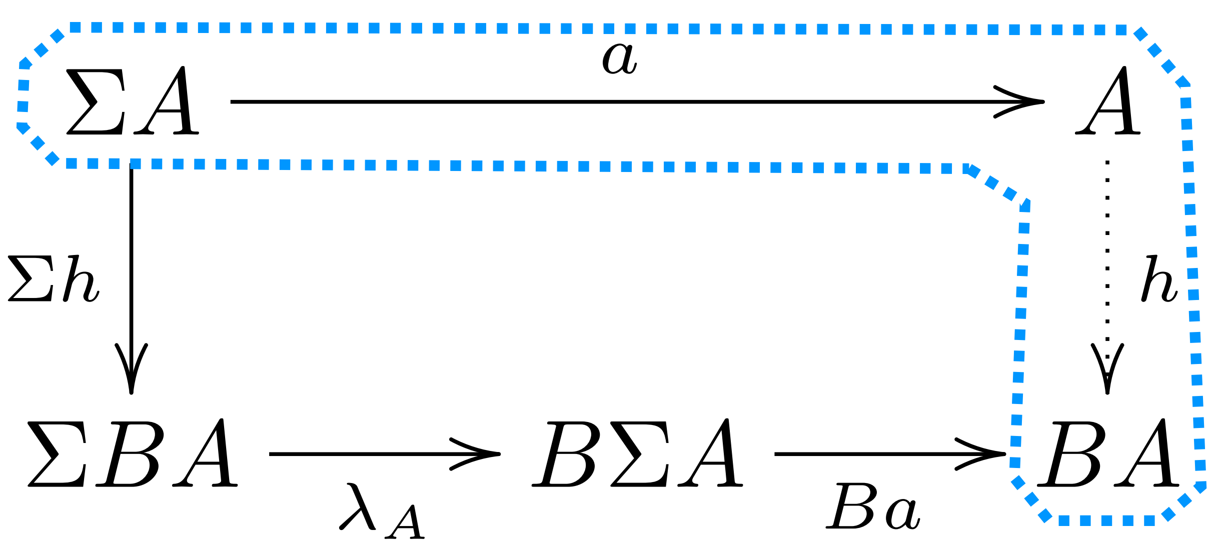
Bialgebras are (co)algebras:

$$\lambda\textbf{-bialg} \cong \Sigma^{\lambda}\textbf{-alg} \cong B_{\lambda}\textbf{-coalg}$$

More benefits

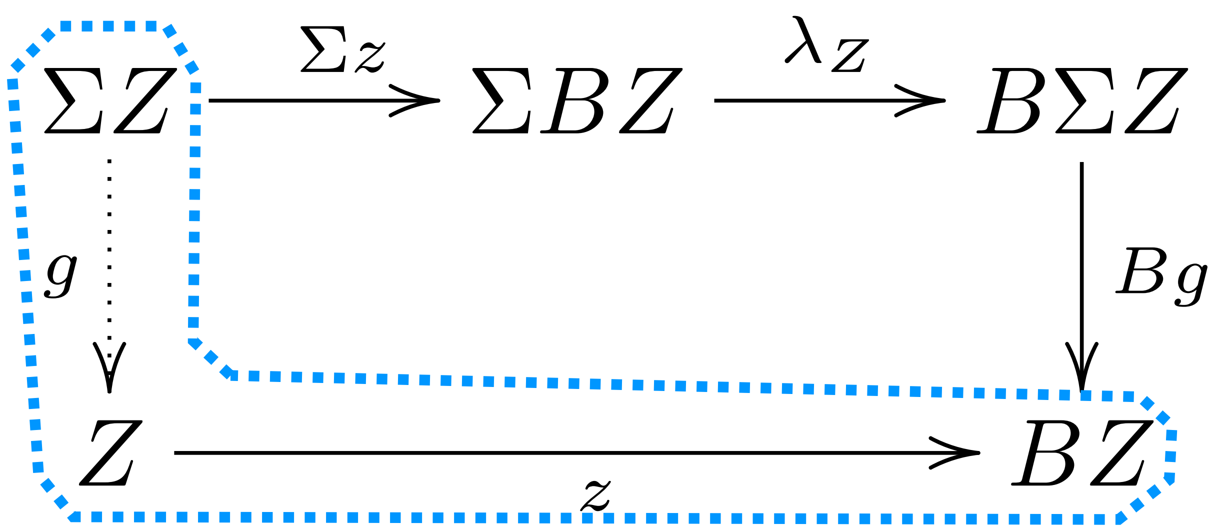
$$\lambda : \Sigma B \Longrightarrow B\Sigma$$



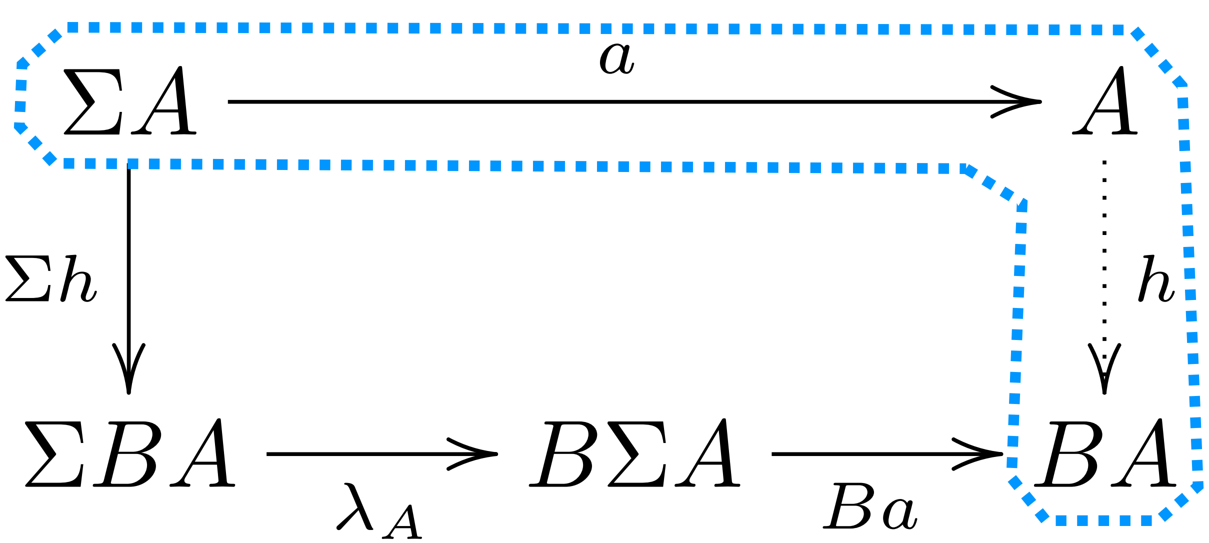


More benefits

$$\lambda : \Sigma B \Longrightarrow B\Sigma$$



the final bialgebra



the initial bialgebra

Bisimilarity as a congruence

$$\begin{array}{ccccc}
\Sigma A & \xrightarrow[\cong]{a} & A & \xrightarrow{h} & BA \\
\downarrow{\scriptstyle \Sigma f} & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle Bf} \\
\Sigma Z & \xrightarrow[g]{} & Z & \xrightarrow[z]{\cong} & BZ
\end{array}$$

The kernel relation of f is:

- bisimilarity on h,
- a congruence on a.

Bisimilarity as a congruence

$$
\begin{array}{ccccc}
\Sigma A & \xrightarrow[\cong]{a} & A & \xrightarrow{h} & BA \\
\Big\downarrow{\scriptstyle \Sigma f} & & \Big\downarrow{\scriptstyle f} & & \Big\downarrow{\scriptstyle Bf} \\
\Sigma Z & \xrightarrow[g]{} & Z & \xrightarrow[z]{\cong} & BZ
\end{array}
$$

The kernel relation of f is:

- bisimilarity on h,
- a congruence on a.

Bisimilarity is a congruence

Bisimilarity as a congruence

$$\begin{array}{ccccc} \Sigma A & \xrightarrow[\cong]{a} & A & \xrightarrow{h} & BA \\ \Big\downarrow{\scriptstyle \Sigma f} & & \Big\downarrow{\scriptstyle f} & & \Big\downarrow{\scriptstyle Bf} \\ \Sigma Z & \xrightarrow[g]{} & Z & \xrightarrow[z]{\cong} & BZ \end{array}$$

The kernel relation of f is:

- bisimilarity on h,
- a congruence on a.

Bisimilarity is a congruence

For example,

$$\texttt{alt}(\texttt{a},\texttt{a}) \approx \texttt{a} \quad \text{hence} \quad C[\texttt{alt}(\texttt{a},\texttt{a})] \approx C[\texttt{a}]$$

II.

MORE DISTRIBUTIVE LAWS

Another coinductive definition

$BX = A \times X$ $\qquad\qquad$ $Z = A^\omega$

Another coinductive definition

$BX = A \times X$ $\qquad\qquad$ $Z = A^\omega$

Define: $\texttt{zip} : Z^2 \to Z$

Another coinductive definition

$BX = A \times X$

$Z = A^\omega$

$a_1, a_2, a_3, a_4, a_5, \ldots$

$b_1,\ b_2,\ b_3,\ b_4,\ b_5, \ldots$

Define: $\texttt{zip} : Z^2 \to Z$

Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

$$a_1, a_2, a_3, a_4, a_5, \ldots$$

$$b_1, b_2, b_3, b_4, b_5, \ldots$$

Define: $\texttt{zip} : Z^2 \to Z$

Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

Define: $\texttt{zip} : Z^2 \to Z$

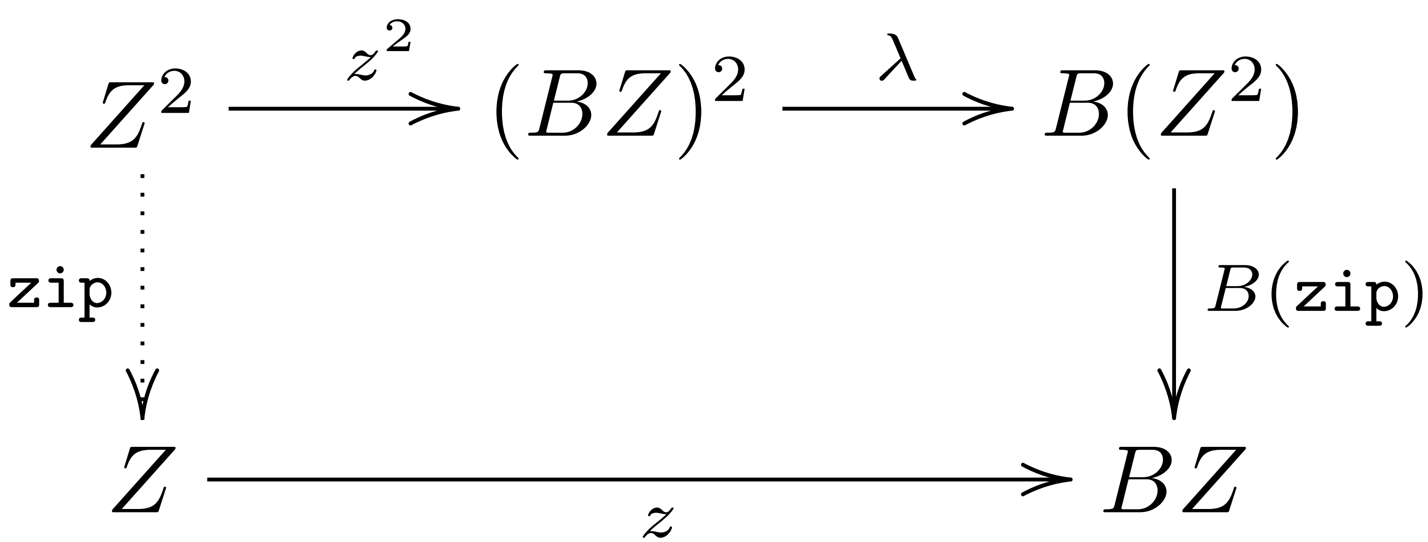
$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots \qquad b_1,\ b_2,\ b_3,\ b_4,\ b_5, \ldots}{a_1, b_1,\ a_2, b_2,\ a_3, \ldots}\ \texttt{zip}$$

Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

$$\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1,\ b_2,\ b_3,\ b_4,\ b_5, \ldots \\ \hline a_1, b_1,\ a_2, b_2,\ a_3, \ldots \end{array} \ \texttt{zip}$$

Define: $\texttt{zip} : Z^2 \to Z$

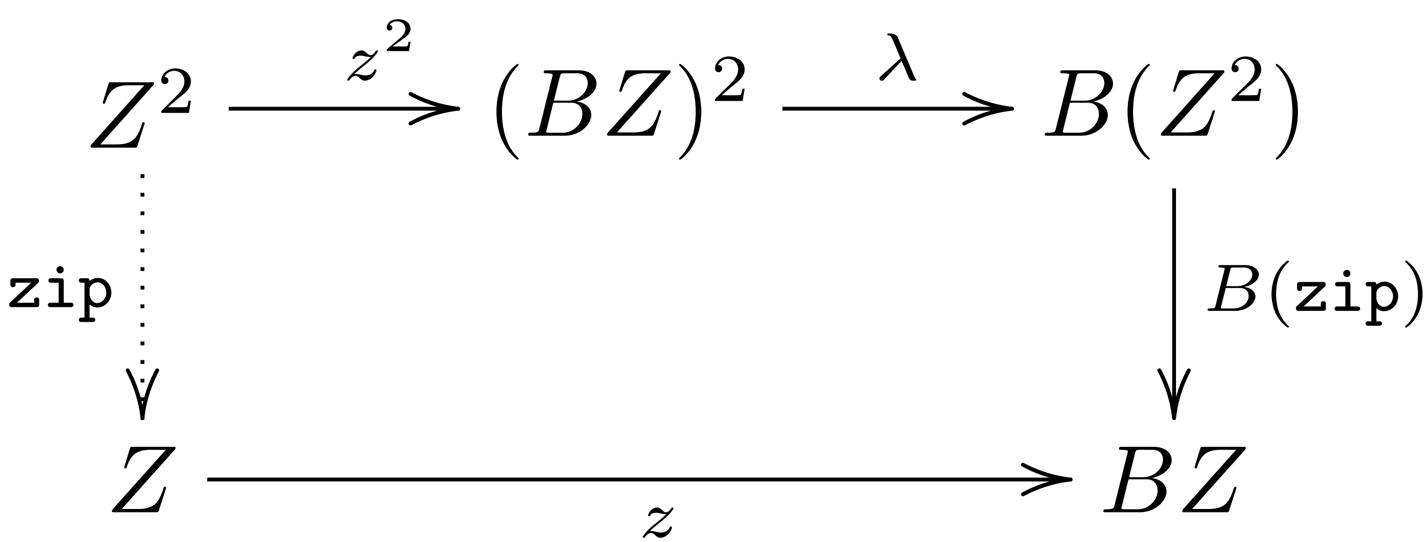


Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

Define: $\texttt{zip} : Z^2 \to Z$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1,\ b_2,\ b_3,\ b_4,\ b_5, \ldots \end{array}}{a_1, b_1,\ a_2, b_2,\ a_3, \ldots}\ \texttt{zip}$$



$$\lambda(a, \sigma, b, \tau) = a, (b :: \tau, \sigma)$$

Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

Define: $\mathtt{zip} : Z^2 \to Z$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_1, a_2, b_2, a_3, \ldots}\ \mathtt{zip}$$

$$\begin{array}{ccccc} Z^2 & \xrightarrow{z^2} & (BZ)^2 & \xrightarrow{\lambda} & B(Z^2) \\ {\scriptstyle \mathtt{zip}}\downarrow & & & & \downarrow{\scriptstyle B(\mathtt{zip})} \\ Z & & \xrightarrow[z]{} & & BZ \end{array}$$

$$\lambda(a, \sigma, b, \tau) = a, (b :: \tau, \sigma)$$

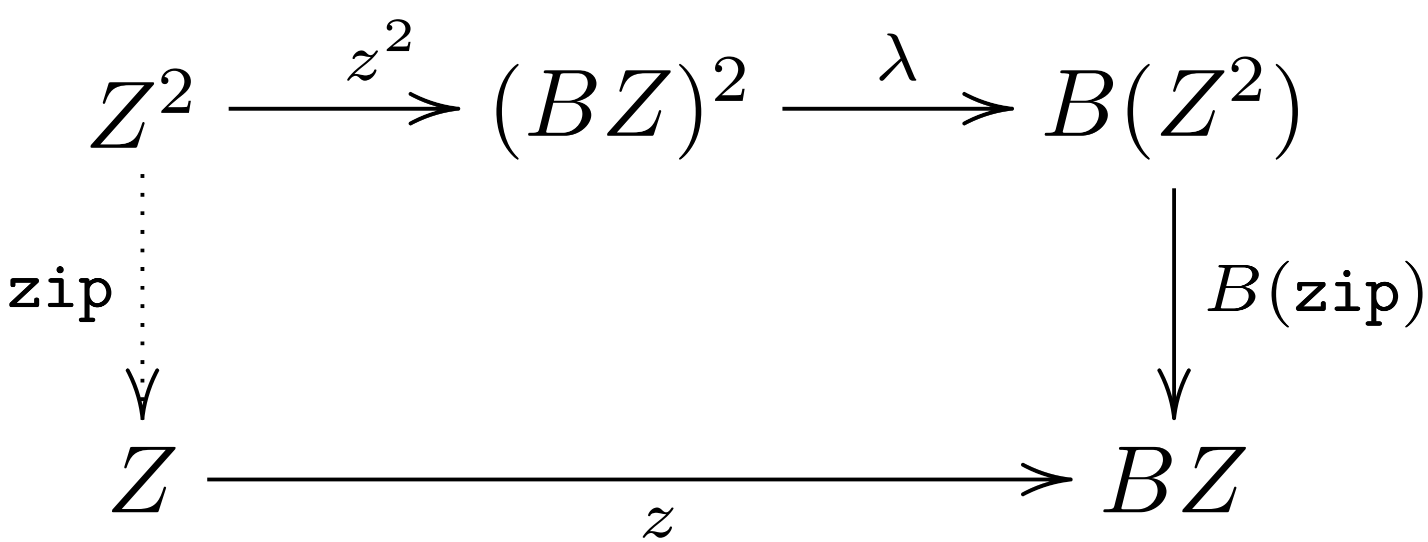
$$\frac{x \xrightarrow{a} x'}{\mathtt{zip}(x, y) \xrightarrow{a} \mathtt{zip}(y, x')}$$

Another coinductive definition

$$BX = A \times X \qquad\qquad Z = A^\omega$$

Define: $\mathtt{zip} : Z^2 \to Z$

$$\frac{\begin{array}{l} a_1, a_2, a_3, a_4, a_5, \ldots \\ b_1, b_2, b_3, b_4, b_5, \ldots \end{array}}{a_1, b_1, a_2, b_2, a_3, \ldots}\ \mathtt{zip}$$



$$\lambda(a, \sigma, b, \tau) = a, (b :: \tau, \sigma)$$

$$\frac{x \xrightarrow{a} x'}{\mathtt{zip}(x, y) \xrightarrow{a} \mathtt{zip}(y, x')}$$

But this is not natural!

Copointed coalgebras

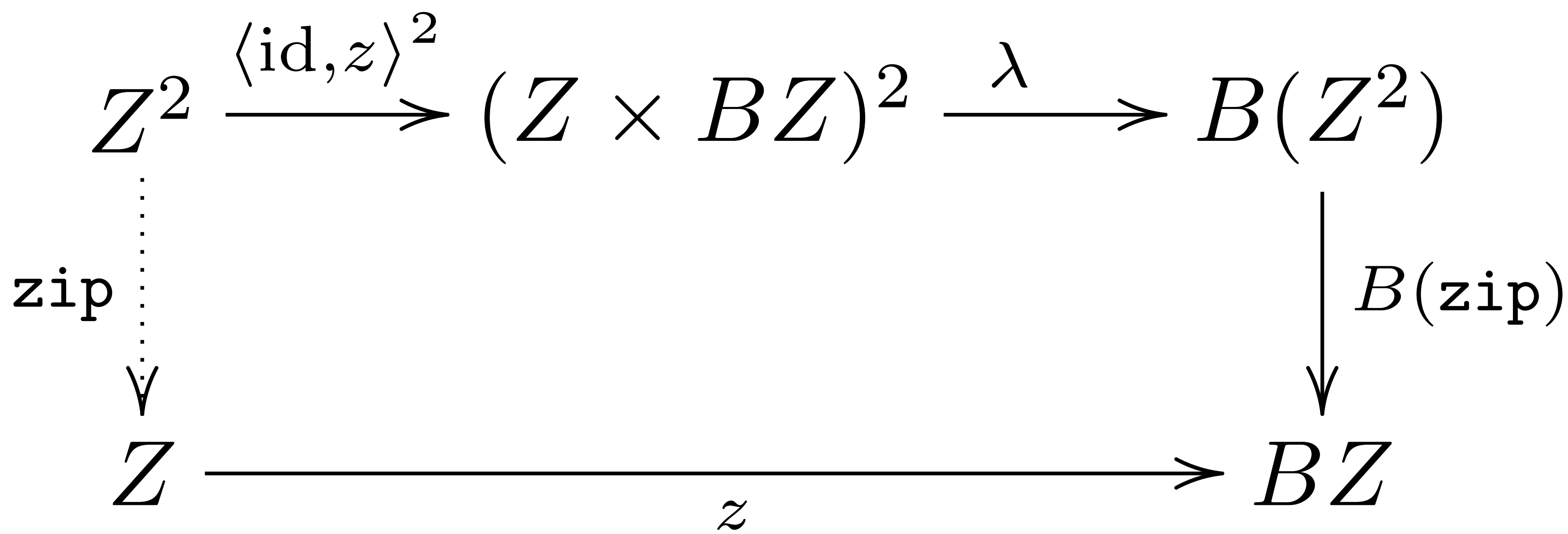
A distributive law definition of `zip` :

$$\begin{array}{ccccc} Z^2 & \xrightarrow{\langle \mathrm{id}, z\rangle^2} & (Z \times BZ)^2 & \xrightarrow{\lambda} & B(Z^2) \\ {\scriptstyle \mathtt{zip}}\,\vdots & & & & \downarrow {\scriptstyle B(\mathtt{zip})} \\ Z & & \xrightarrow[z]{\qquad\qquad} & & BZ \end{array}$$

$$(x, a, x', y, b, y') \mapsto a, (y, x') \qquad \frac{x \xrightarrow{a} x'}{\mathtt{zip}(x, y) \xrightarrow{a} \mathtt{zip}(y, x')}$$

Copointed coalgebras

A distributive law definition of `zip` :



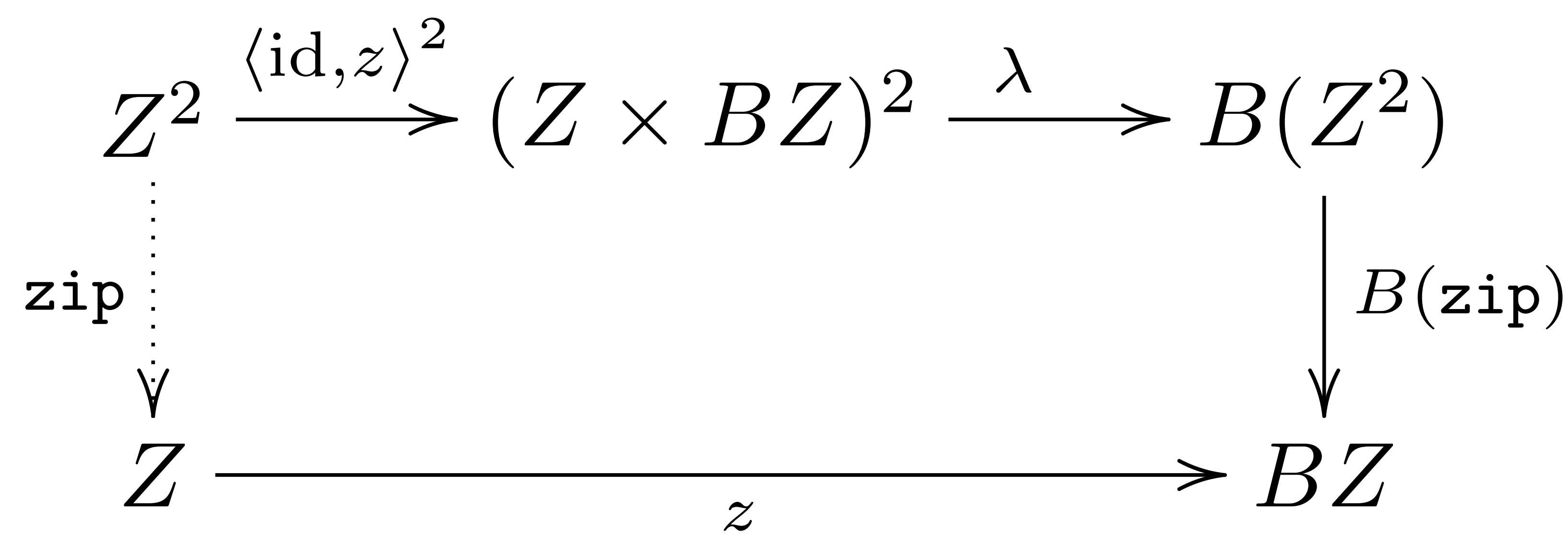
$$(x, a, x', y, b, y') \mapsto a, (y, x')$$

$$\frac{x \xrightarrow{a} x'}{\mathtt{zip}(x, y) \xrightarrow{a} \mathtt{zip}(y, x')}$$

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow B\Sigma$$

Copointed coalgebras

A distributive law definition of `zip` :



$$(x, a, x', y, b, y') \mapsto \ a, (y, x')$$

$$\frac{x \xrightarrow{a} x'}{\mathtt{zip}(x, y) \xrightarrow{a} \mathtt{zip}(y, x')}$$

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow B\Sigma$$

To gain the benefits, work with copointed coalgebras.

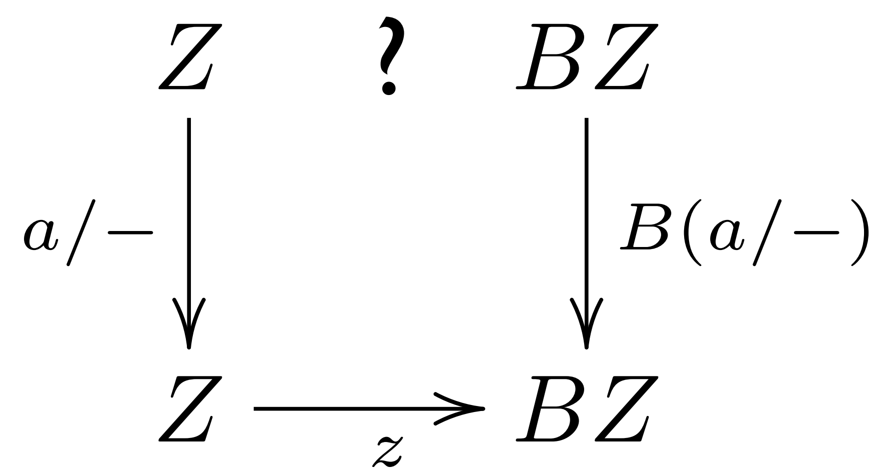
Pointed algebras

$$a/- : Z \to Z$$

$$\frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\; a/-$$

Pointed algebras

$$a/- : Z \to Z \qquad\qquad \frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\; a/-$$



Pointed algebras

$$a/- : Z \to Z \qquad \frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\; a/-$$

Pointed algebras

$$a/- : Z \to Z \qquad \frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\; a/-$$

$$\begin{array}{ccccc}
\Sigma Z & \xrightarrow{\Sigma z} & \Sigma B Z & \xrightarrow{\lambda} & B(Z + \Sigma Z) \\
{\scriptstyle g}\,\vdots & & & & \downarrow {\scriptstyle B[\mathrm{id}, g]} \\
Z & & \xrightarrow[z]{} & & BZ
\end{array}$$

Pointed algebras

$$a/- : Z \to Z \qquad\qquad \frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\; a/-$$

$$\begin{array}{ccccc}
\Sigma Z & \xrightarrow{\Sigma z} & \Sigma B Z & \xrightarrow{\lambda} & B(Z + \Sigma Z) \\
{\scriptstyle g}\,\vdots & & & & \downarrow {\scriptstyle B[\mathrm{id}, g]} \\
Z & & \xrightarrow[z]{\hspace{6em}} & & BZ
\end{array}$$

$$\lambda : BZ \to B(Z + Z)$$

$$b, x' \mapsto a, \iota_1(x') \qquad\qquad \frac{x \xrightarrow{b} x'}{a/x \xrightarrow{a} x'}$$

Pointed algebras

$$a/- : Z \to Z \qquad\qquad \frac{b_1,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}{a,\, b_2,\, b_3,\, b_4,\, b_5, \ldots}\ a/-$$

$$\begin{array}{ccccc}
\Sigma Z & \xrightarrow{\Sigma z} & \Sigma B Z & \xrightarrow{\lambda} & B(Z + \Sigma Z) \\
{\scriptstyle g}\,\vdots & & & & \downarrow {\scriptstyle B[\mathrm{id}, g]} \\
Z & & \xrightarrow[z]{\qquad\qquad} & & BZ
\end{array}$$

$$\lambda : BZ \to B(Z + Z)$$

$$b, x' \mapsto a, \iota_1(x') \qquad\qquad \frac{x \xrightarrow{b} x'}{a/x \xrightarrow{a} x'}$$

To gain the benefits, work with pointed algebras.

Complex successors

$$\frac{x \xrightarrow{a} x'}{\mathtt{f}(x) \xrightarrow{a} \mathtt{zip}(\mathtt{a}, \mathtt{f}(x'))}$$

Complex successors

$$\frac{x \xrightarrow{a} x'}{\mathtt{f}(x) \xrightarrow{a} \mathtt{zip}(\mathtt{a}, \mathtt{f}(x'))}$$

$$\lambda : \Sigma B \Longrightarrow BT_{\Sigma}$$

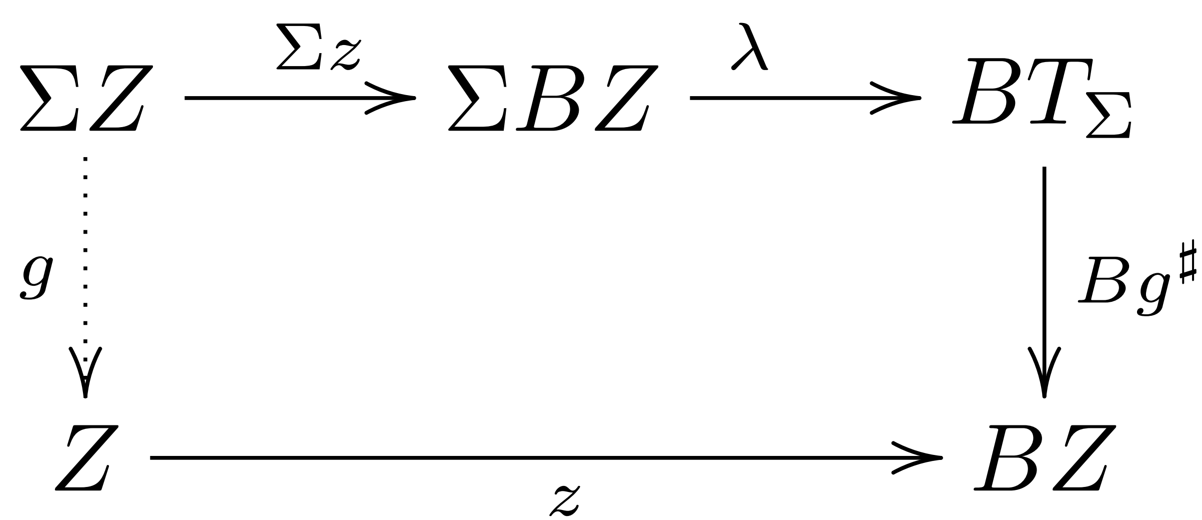
free monad over Σ

Complex successors

$$\frac{x \xrightarrow{a} x'}{\mathtt{f}(x) \xrightarrow{a} \mathtt{zip}(\mathtt{a}, \mathtt{f}(x'))}$$

$$\lambda : \Sigma B \Longrightarrow BT_\Sigma$$

free monad over Σ

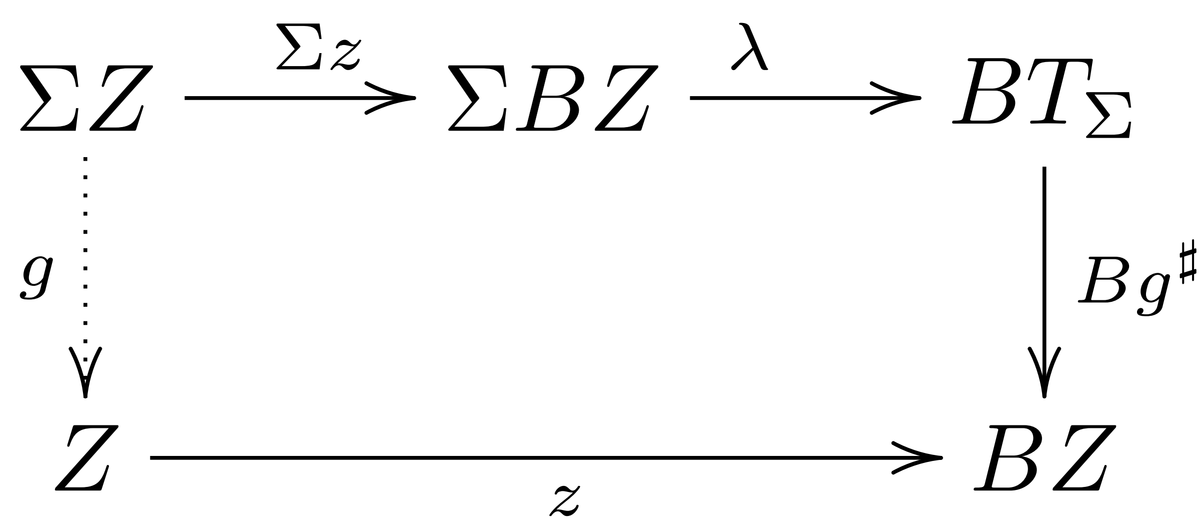


Complex successors

$$\frac{x \xrightarrow{a} x'}{\mathtt{f}(x) \xrightarrow{a} \mathtt{zip}(\mathtt{a}, \mathtt{f}(x'))}$$

$$\lambda : \Sigma B \Longrightarrow BT_\Sigma$$

free monad over Σ



Here, work with E-M algebras for the monad T_Σ

Complex successors

$$\frac{x \xrightarrow{a} x'}{\mathtt{f}(x) \xrightarrow{a} \mathtt{zip}(\mathtt{a}, \mathtt{f}(x'))}$$

$$\lambda : \Sigma B \Longrightarrow BT_\Sigma$$

free monad over Σ

$$\begin{array}{ccccc}
\Sigma Z & \xrightarrow{\Sigma z} & \Sigma B Z & \xrightarrow{\lambda} & BT_\Sigma \\
{\scriptstyle g}\,\downarrow & & & & \downarrow\,{\scriptstyle Bg^\sharp} \\
Z & & \xrightarrow[z]{\qquad\qquad} & & BZ
\end{array}$$

Here, work with E-M algebras for the monad T_Σ

Together with `zip`: $\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$

Look-ahead

Look-ahead

$\texttt{odd} : Z \to Z$

$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots}{a_1, a_3, a_5, a_7, \ldots}\ \texttt{odd}$$

Look-ahead

$\texttt{odd} : Z \to Z$

$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots}{a_1, a_3, a_5, a_7, \ldots}\ \texttt{odd}$$

$$\frac{x \xrightarrow{a} y \xrightarrow{b} z}{\texttt{odd}(x) \xrightarrow{a} \texttt{odd}(z)}$$

Look-ahead

$$\texttt{odd} : Z \to Z$$

$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots}{a_1, a_3, a_5, a_7, \ldots}\ \texttt{odd}$$

$$\frac{x \xrightarrow{a} y \xrightarrow{b} z}{\texttt{odd}(x) \xrightarrow{a} \texttt{odd}(z)}$$

$$\lambda : \Sigma D_B \Longrightarrow B\Sigma$$

Look-ahead

$\mathtt{odd} : Z \to Z$

$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots}{a_1, a_3, a_5, a_7, \ldots}\ \mathtt{odd}$$

$$\frac{x \xrightarrow{a} y \xrightarrow{b} z}{\mathtt{odd}(x) \xrightarrow{a} \mathtt{odd}(z)}$$

$$\lambda : \Sigma D_B \Longrightarrow B\Sigma$$

cofree comonad over B

Look-ahead

$\texttt{odd} : Z \to Z$

$$\frac{a_1, a_2, a_3, a_4, a_5, \ldots}{a_1, a_3, a_5, a_7, \ldots}\ \texttt{odd}$$

$$\frac{x \xrightarrow{a} y \xrightarrow{b} z}{\texttt{odd}(x) \xrightarrow{a} \texttt{odd}(z)}$$

$$\lambda : \Sigma D_B \Longrightarrow B\Sigma$$

cofree comonad over B

$$\begin{array}{ccccc} \Sigma Z & \xrightarrow{\Sigma z^\flat} & \Sigma D_B Z & \xrightarrow{\lambda_Z} & B\Sigma Z \\ {\scriptstyle g}\,\vdots & & & & \downarrow{\scriptstyle Bg} \\ Z & & \xrightarrow[z]{\qquad\qquad} & & BZ \end{array}$$

Monad over comonad

The general case: $\lambda : T_\Sigma D_B \Longrightarrow D_B T_\Sigma$

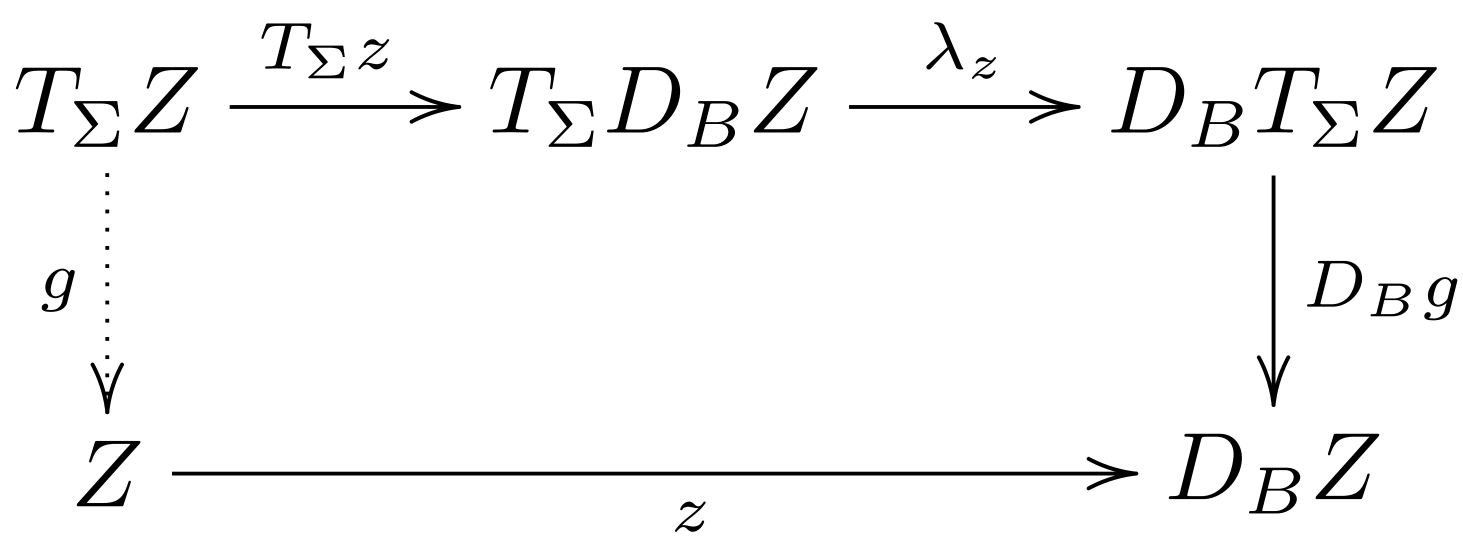
subject to laws

$$\begin{array}{ccccc}
T_\Sigma Z & \xrightarrow{T_\Sigma z} & T_\Sigma D_B Z & \xrightarrow{\lambda_z} & D_B T_\Sigma Z \\
{\scriptstyle g}\, \vdots\downarrow & & & & \downarrow {\scriptstyle D_B g} \\
Z & & \xrightarrow[z]{\hspace{6em}} & & D_B Z
\end{array}$$

Monad over comonad

The general case: $\lambda : T_\Sigma D_B \Longrightarrow D_B T_\Sigma$

subject to laws



This subsumes all previous cases.

III.

STRUCTURAL OPERATIONAL SEMANTICS

III.

STRUCTURAL OPERATIONAL SEMANTICS

$$BX = (\mathcal{P}_\omega X)^A$$

Toy SOS example

$$\Sigma X = 1 + A + X^2 \qquad BX = (\mathcal{P}_\omega X)^A$$

$$\frac{}{a \xrightarrow{a} \mathtt{nil}} \qquad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y'}{x \otimes y \xrightarrow{a} x' \otimes y'}$$

This defines: $\lambda : \Sigma B \Longrightarrow B\Sigma$

Toy SOS example

$$\Sigma X = 1 + A + X^2 \qquad BX = (\mathcal{P}_\omega X)^A$$

$$\frac{}{a \xrightarrow{a} \texttt{nil}} \qquad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y'}{x \otimes y \xrightarrow{a} x' \otimes y'}$$

This defines: $\lambda : \Sigma B \Longrightarrow B\Sigma$

Fact: the LTS induced by the rules is the initial λ-bialgebra.

Hence, bisimilarity on it is a congruence.

GSOS

Laws $\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ correspond to rules:

$$\frac{\{x_i \xrightarrow{a_{ij}} y_{ij}\}^{1\leq i\leq n}_{1\leq j\leq m_i} \quad \{x_i \stackrel{b_{ik}}{\not\longrightarrow}\}^{1\leq i\leq n}_{1\leq k\leq l_i}}{\mathtt{f}(x_1,\ldots,x_n) \xrightarrow{c} t}$$

where

- x_i , y_{ij} are all distinct
- no other variables occur in t
- there are finitely many rules for every $\mathtt{f}$, c

GSOS

Laws $\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ correspond to rules:

$$\frac{\{x_i \xrightarrow{a_{ij}} y_{ij}\}^{1\le i\le n}_{1\le j\le m_i} \qquad \{x_i \stackrel{b_{ik}}{\not\longrightarrow}\}^{1\le i\le n}_{1\le k\le l_i}}{\mathtt{f}(x_1,\ldots,x_n) \xrightarrow{c} t}$$

where

- x_i , y_{ij} are all distinct
- no other variables occur in t
- there are finitely many rules for every $\mathtt{f}$, c

no lookahead

GSOS

Laws $\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ correspond to rules:

$$\frac{\{x_i \xrightarrow{a_{ij}} y_{ij}\}^{1\leq i\leq n}_{1\leq j\leq m_i} \quad \{x_i \stackrel{b_{ik}}{\not\longrightarrow}\}^{1\leq i\leq n}_{1\leq k\leq l_i}}{\mathtt{f}(x_1,\ldots,x_n) \xrightarrow{c} t}$$

where

- x_i , y_{ij} are all distinct
- no other variables occur in t
- there are finitely many rules for every $\mathtt{f}$, c

GSOS

Laws $\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ correspond to rules:

$$\frac{\{x_i \xrightarrow{a_{ij}} y_{ij}\}^{1 \le i \le n}_{1 \le j \le m_i} \quad \{x_i \stackrel{b_{ik}}{\not\longrightarrow}\}^{1 \le i \le n}_{1 \le k \le l_i}}{\mathtt{f}(x_1, \ldots, x_n) \xrightarrow{c} t}$$

where

- x_i , y_{ij} are all distinct
- no other variables occur in t
- there are finitely many rules for every $\mathtt{f}$, c

This is GSOS, where bisimilarity is a congruence.

Safe ntree

Laws $\lambda : \Sigma D_B \Longrightarrow B(\mathrm{Id} + \Sigma)$ correspond to rules:

$$\frac{\{z_i \xrightarrow{a_i} y_i\}_{i \in I} \quad \{w_j \stackrel{b_j}{\not\longrightarrow}\}_{j \in J}}{\mathtt{f}(x_1, \ldots, x_n) \xrightarrow{c} t}$$

where

- x_i , y_i are all distinct
- no other variables occur in the rule
- the graph of premises is well-founded
- t has depth at most 1
- there are finitely many rules for every $\mathtt{f}$, c

Safe ntree

Laws $\lambda : \Sigma D_B \Longrightarrow B(\mathrm{Id} + \Sigma)$ correspond to rules:

$$\frac{\{z_i \xrightarrow{a_i} y_i\}_{i \in I} \quad \{w_j \stackrel{b_j}{\not\longrightarrow}\}_{j \in J}}{\mathtt{f}(x_1, \ldots, x_n) \xrightarrow{c} t}$$

where

- x_i , y_i are all distinct
- no other variables occur in the rule
- the graph of premises is well-founded
- t has depth at most 1
- there are finitely many rules for every $\mathtt{f}$, c

This is the "safe" ntree format.

Monad over comonad?

$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ complex successors

$\lambda : \Sigma D_B \Longrightarrow B(\mathrm{Id} + \Sigma)$ look-ahead

$\lambda : T_\Sigma D_B \Longrightarrow D_B T_\Sigma$ both?

Monad over comonad?

$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$ complex successors

$\lambda : \Sigma D_B \Longrightarrow B(\mathrm{Id} + \Sigma)$ look-ahead

$\lambda : T_\Sigma D_B \Longrightarrow D_B T_\Sigma$ both?

But:

$$\frac{}{\mathtt{k} \xrightarrow{a} \mathtt{f}(\mathtt{k})} \qquad \frac{x \xrightarrow{a} y \quad y \not\xrightarrow{b}}{\mathtt{f}(x) \xrightarrow{b} \mathtt{k}}$$

does not induce an LTS!

IV.

FURTHER DEVELOPMENTS

Probabilistic (reactive) GSOS

$BX = (1 + \mathcal{D}_\omega X)^A$

$\mathcal{D}_\omega X = \{\phi : X \to \mathbb{R}_0^+ \mid \sharp\mathrm{supp}(\phi) < \omega,\ \sum_x \phi(x) = 1\}$

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$$

$$\Updownarrow$$

$$\frac{\left\{\mathtt{x}_i \xrightarrow{a}\ \right\}_{a \in D_i} \quad \left\{\mathtt{x}_i \not\xrightarrow{a}\ \right\}_{a \in B_i} \quad \left\{\mathtt{x}_{i_j} \xrightarrow{b_j} \mathtt{y}_j\right\}_{1 \leq j \leq k}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c,w} \mathtt{t}}$$

Probabilistic bisimilarity is a congruence.

Stochastic GSOS

$BX = (\mathcal{F}X)^A$

$\mathcal{F}X = \{\phi : X \to \mathbb{R}_0^+ \mid \sharp\mathrm{supp}(\phi) < \omega\}$

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$$

$$\Updownarrow$$

$$\frac{\left\{\mathtt{x}_i \xrightarrow{a@w_{ai}}\right\}_{a \in D_i, 1 \le i \le n} \quad \left\{\mathtt{x}_{i_j} \xrightarrow{b_j} \mathtt{y}_j\right\}_{1 \le j \le k}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c,w} \mathtt{t}}$$

Stochastic bisimilarity is a congruence.

Weighted GSOS

$BX = (\mathcal{F}X)^A$

$\mathcal{F}X = \{\phi : X \to \mathbb{W} \mid \sharp\mathrm{supp}(\phi) < \omega\}$

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$$

$$\Uparrow$$

$$\frac{\left\{\mathtt{x}_i \xrightarrow{a} w_{a,i}\right\}_{a \in D_i, 1 \leq i \leq n} \quad \left\langle \mathtt{x}_{i_j} \xrightarrow{b_j} \mathtt{y}_j \right\rangle_{1 \leq j \leq k}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c,\beta} \mathtt{t}}$$

Weighted bisimilarity is a congruence.

Weighted GSOS

$$BX = (\mathcal{F}X)^A$$

$$\mathcal{F}X = \{\phi : X \to \mathbb{W} \mid \sharp\mathrm{supp}(\phi) < \omega\}$$

any comm. monoid

$$\lambda : \Sigma(\mathrm{Id} \times B) \Longrightarrow BT_\Sigma$$

$$\Uparrow$$

$$\frac{\left\{\mathtt{x}_i \xrightarrow{a} w_{a,i}\right\}_{a \in D_i, 1 \leq i \leq n} \quad \left\langle \mathtt{x}_{i_j} \xrightarrow{b_j} \mathtt{y}_j \right\rangle_{1 \leq j \leq k}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c,\beta} \mathtt{t}}$$

Weighted bisimilarity is a congruence.

Timed SOS

$$BX = \{\mathcal{T} \rightharpoonup X \mid \ldots\}$$

$\mathcal{T}$ a time domain

(a comm. monoid with cancellation and linear specialization order)

$$\lambda : T_\Sigma D_B \Longrightarrow D_B T_\Sigma$$

$$\Updownarrow$$

$$\frac{\{x_i(0) \xrightarrow{t_i} x_i(t_i) \mid t_i \in \mathcal{T} \wedge x_i(t_i){\downarrow}\}_{1 \leq i \leq n}}{\mathtt{f}(x_1(0), \ldots, x_n(0)) \xrightarrow{t} \theta_t}$$

Name-passing GSOS

Another underlying category: $\mathbf{Nom}$

(actually a bit more complicated...)

$$\lambda : \Sigma(|-| \times B|-|) \Longrightarrow BT_\Sigma|-|$$

$$\Uparrow$$

$$\frac{x \xrightarrow{c!d} x' \quad y \xrightarrow{c?(a)} y'}{x||y \xrightarrow{\tau} y||([d/a]y')}$$

"Wide open" bisimilarity a congruence.

Other work

- The "microcosm" interpretation

GSOS rules → operations on coalgebras

Other work

- The "microcosm" interpretation

GSOS rules ⟶ operations on coalgebras

Parallel composition of states

vs.

Parallel composition of coalgebras

Other work

- The “microcosm” interpretation

GSOS rules ⟶ operations on coalgebras

Parallel composition of states

vs.

Parallel composition of coalgebras

- tyft/tyxt categorically

$$\frac{t_1 \xrightarrow{a_1} u_1 \quad t_2 \xrightarrow{a_2} u_2}{\mathtt{f}(x_1, \ldots, x_n) \xrightarrow{a} t}$$

V.

SELECTED OPEN PROBLEMS

Rules/premises/conclusions

$$\begin{aligned} \text{collection of rules} &\approx \text{distributive law} \\ \text{rule} &\approx ? \end{aligned}$$

Rules/premises/conclusions

collection of rules $\approx$ distributive law

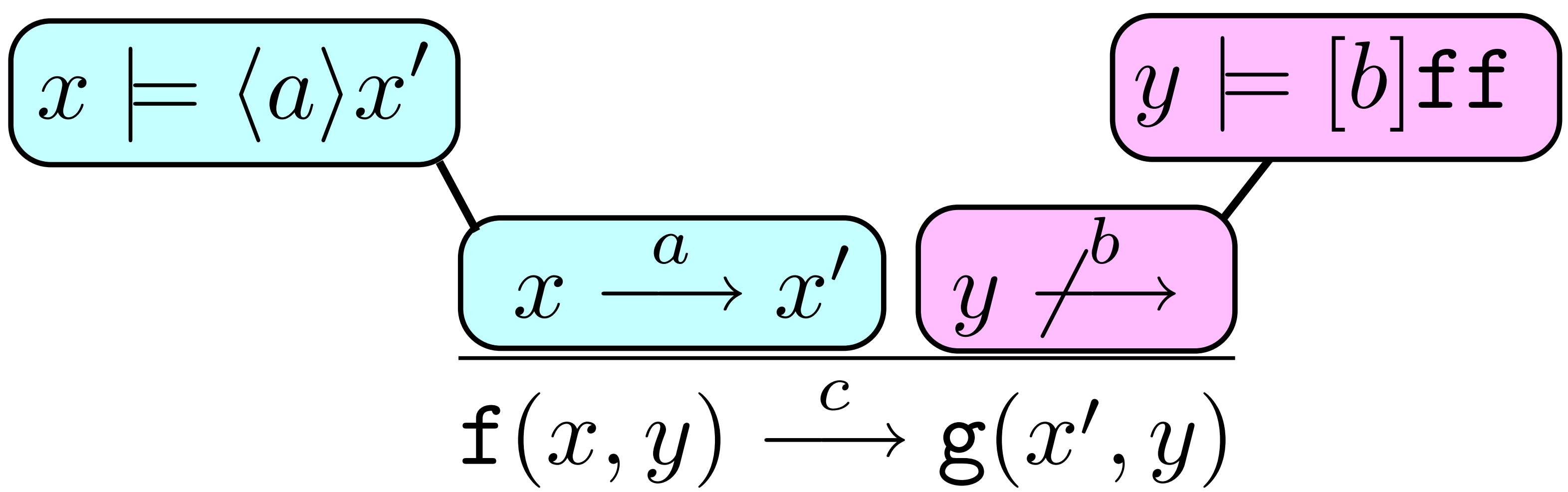
rule $\approx$?

$$\frac{x \xrightarrow{a} x' \quad y \not\xrightarrow{b}}{\mathtt{f}(x,y) \xrightarrow{c} \mathtt{g}(x',y)}$$

Rules/premises/conclusions

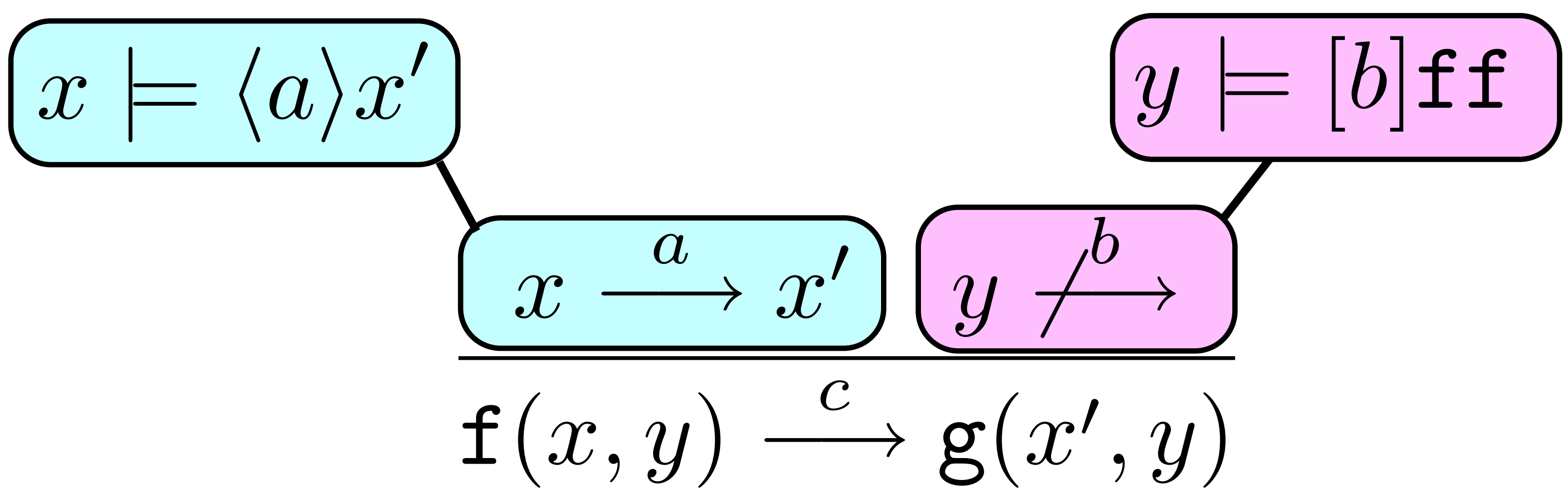
collection of rules $\approx$ distributive law

rule $\approx$?



Rules/premises/conclusions

collection of rules	$\approx$	distributive law
rule	$\approx$	?



premise	$\approx$	modal formula
conclusion	$\approx$	?

Connections to modal logic

Distributive law $\implies$ bisimilarity a congruence.

What about other equivalences?

Connections to modal logic

Distributive law $\Longrightarrow$ bisimilarity a congruence.

What about other equivalences?

Idea: see them as logical equivalences, use the logic.

$$\begin{aligned}
\top &= \mathbb{T} \\
\langle a \rangle \mathbb{T} &= \mathbf{a} \vee [\otimes]\langle a \rangle \top \\
\langle a \rangle \top &= \mathbf{a} \vee [\otimes]\langle a \rangle \top \\
\langle a \rangle (\mathtt{b} \vee [\otimes] x) &= [\otimes]\langle a \rangle x \\
\langle a \rangle [\otimes] x &= [\otimes]\langle a \rangle x
\end{aligned}$$

Translations

Morphisms of ditributive laws:
- well understood abstractly
- very few examples worked out

Translations

Morphisms of ditributive laws:
- well understood abstractly
- very few examples worked out

What is a translation of GSOS specifications?

Translations

Morphisms of ditributive laws:
- well understood abstractly
- very few examples worked out

What is a translation of GSOS specifications?

How to translate CCS into timed CCS?

Or into π-calculus?

Translations

Morphisms of ditributive laws:
- well understood abstractly
- very few examples worked out

What is a translation of GSOS specifications?

How to translate CCS into timed CCS?

Or into π-calculus?

Logics can be translated too!

Parametricity

Various form of parametricity:

Parametricity

Various form of parametricity:

- `loop` parametrized by `;` : $$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

Parametricity

Various form of parametricity:

- $\texttt{loop}$ parametrized by $\texttt{;}$: $\dfrac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$

- CCS parametrized by a set of actions

Parametricity

Various form of parametricity:

- `loop` parametrized by `;` : $$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

Parametricity

Various form of parametricity:

- `loop` parametrized by `;` : $$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:

Parametricity

Various form of parametricity:

- `loop` parametrized by ; :
$$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:

$$\begin{array}{c} \lambda_p \\ \big\downarrow \\ \lambda_b \end{array}$$

Parametricity

Various form of parametricity:

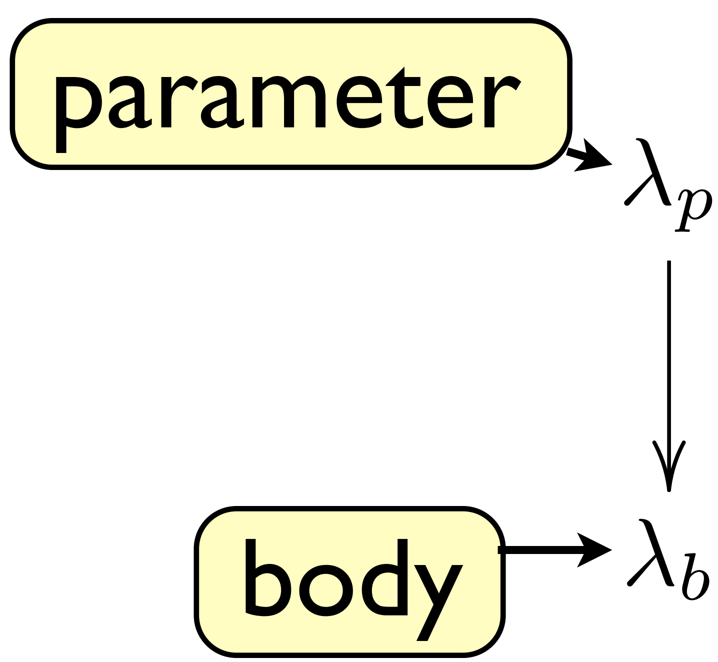
- `loop` parametrized by ; :

$$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:



Parametricity

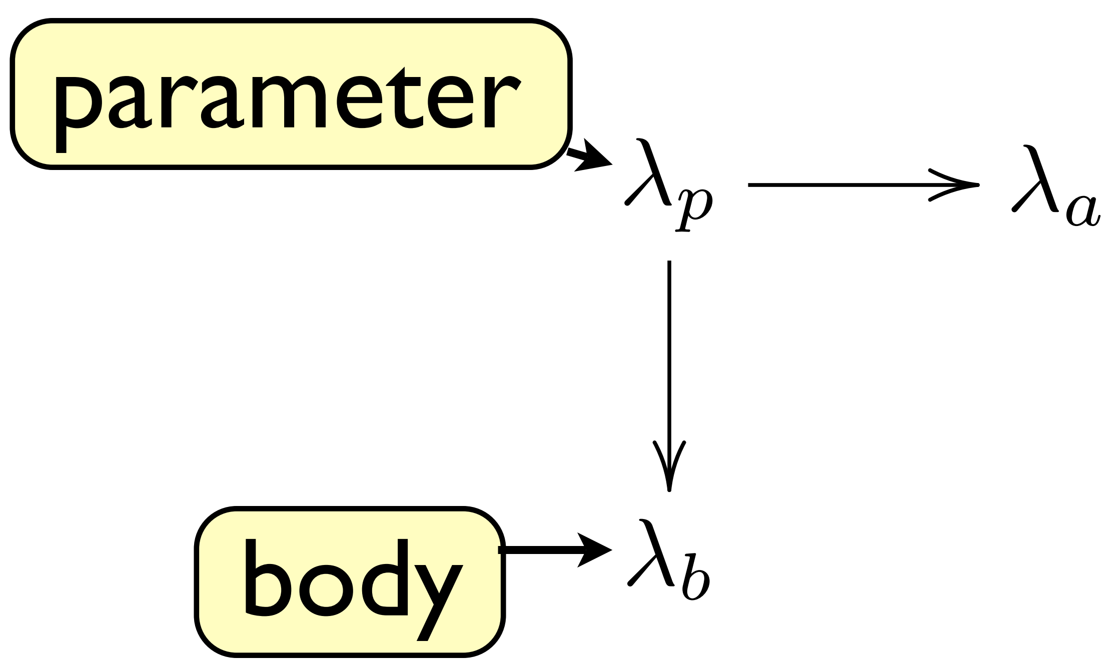
Various form of parametricity:

- loop parametrized by `;` : $\dfrac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x';\texttt{loop}(x)}$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:



Parametricity

Various form of parametricity:

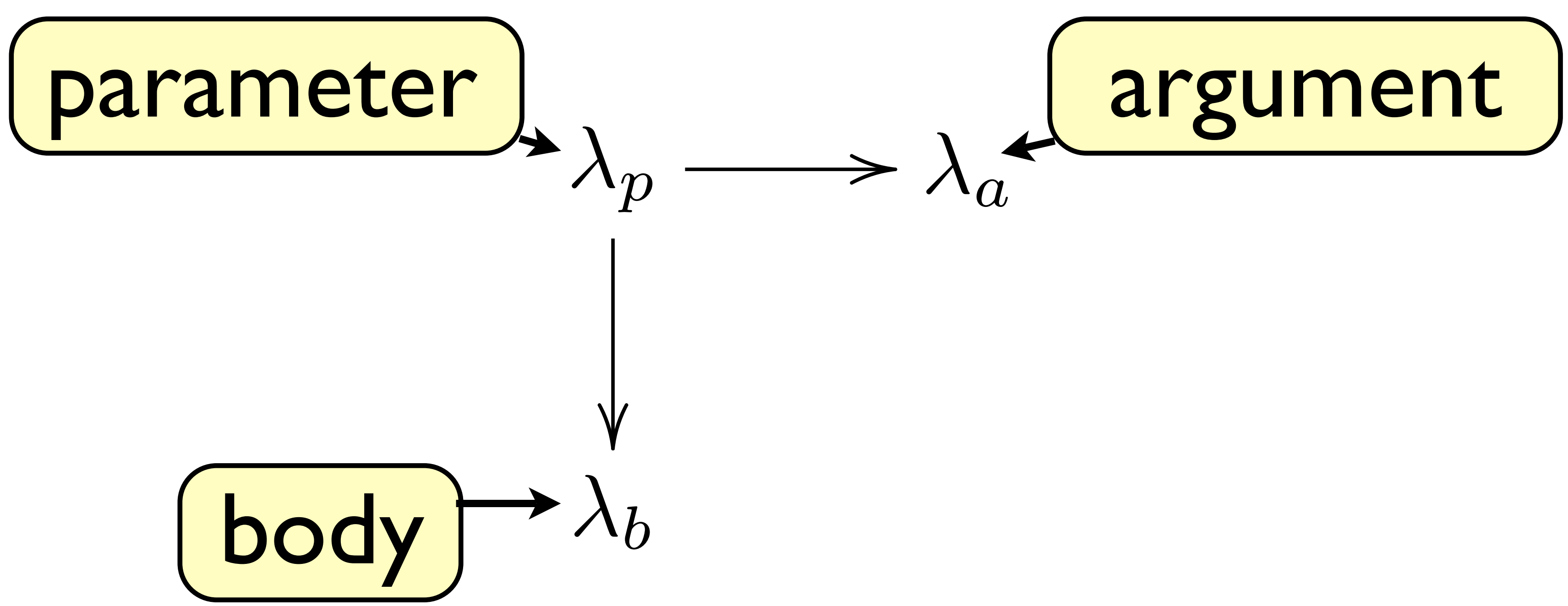
- `loop` parametrized by `;` :

$$\frac{x \xrightarrow{a} x'}{\mathtt{loop}(x) \xrightarrow{a} x'; \mathtt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:



Parametricity

Various form of parametricity:

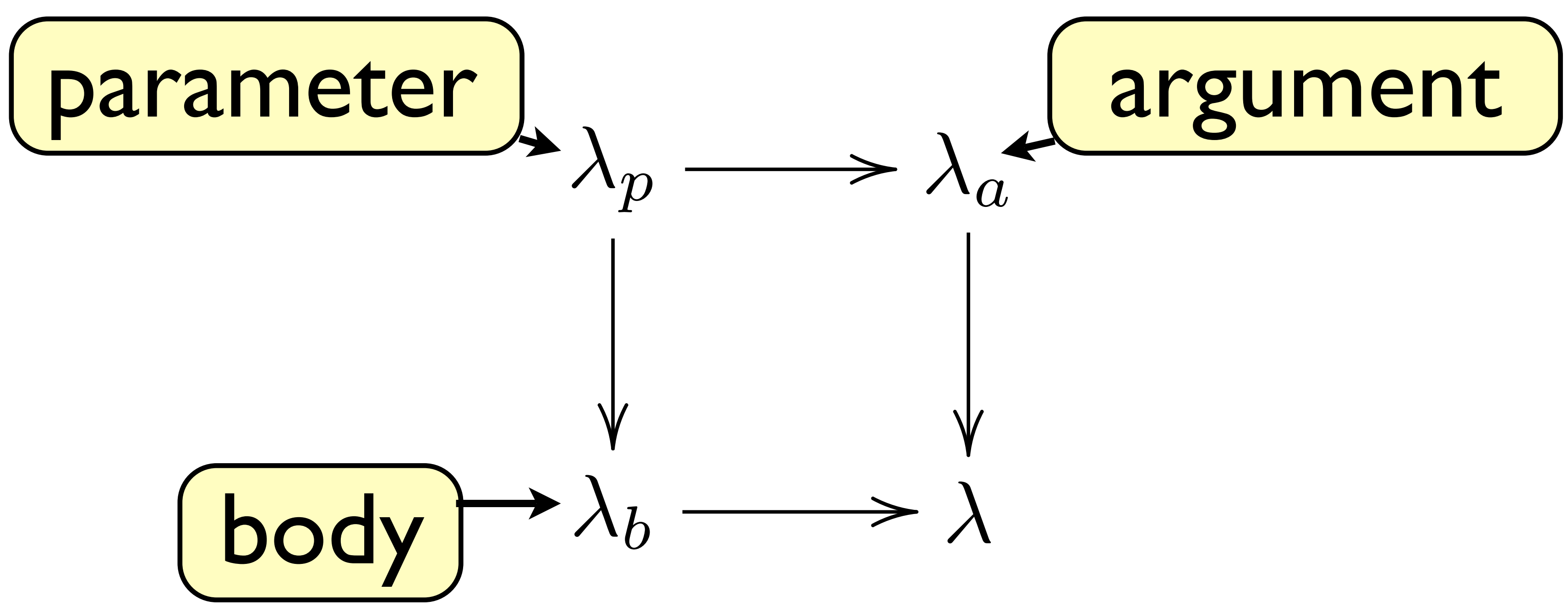
- `loop` parametrized by $;$:

$$\frac{x \xrightarrow{a} x'}{\texttt{loop}(x) \xrightarrow{a} x'; \texttt{loop}(x)}$$

- CCS parametrized by a set of actions

What is a parametric specification?

E.g. pushout parametricity:



Modular SOS formalism

Language for defining SOS specifications

- building blocks: distributive laws
  = ways of writing them down (rules?)

- ways of combining them:
  = translation
  = parametricity / instantiation