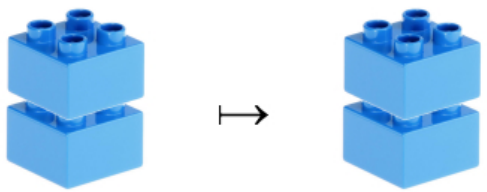
My Hobby: Self-Distributing Monads

Maaike Zwart

IT University of Copenhagen

15 November 2024

$\lambda :$  $\mapsto$

Overview

- Background: monads and distributive laws

Overview

- Background: monads and distributive laws
- Self-Distribution - easy peasy?

Overview

- Background: monads and distributive laws
- Self-Distribution - easy peasy?
- Self-Distribution - confusingly hard?

Overview

- Background: monads and distributive laws
- Self-Distribution - easy peasy?
- Self-Distribution - confusingly hard?
- Self-Distribution - really not a special case at all!

Overview

- Background: monads and distributive laws
- Self-Distribution - easy peasy?
- Self-Distribution - confusingly hard?
- Self-Distribution - really not a special case at all!
- Interesting open questions

Overview

- Background: monads and distributive laws
- Self-Distribution - easy peasy?
- Self-Distribution - confusingly hard?
- Self-Distribution - really not a special case at all!
- Interesting open questions
- Conclusion

Monads

What they are:

What they do:

Monads

What they are: Functors with structure.

$$\langle \mathcal{M},\ \eta : 1 \to \mathcal{M},\ \mu : \mathcal{M}\mathcal{M} \to \mathcal{M} \rangle$$

η : x $\mapsto$ 

μ :  $\mapsto$ 

$$\begin{array}{ccccc} & & M & & \\ & {\scriptstyle M\eta}\swarrow & \| & \searrow{\scriptstyle \eta M} & \\ MM & \xrightarrow{\mu} & M & \xleftarrow{\mu} & MM \end{array}$$

$$\begin{array}{ccc} MMM & \xrightarrow{M\mu} & MM \\ {\scriptstyle \mu M}\downarrow & & \downarrow{\scriptstyle \mu} \\ MM & \xrightarrow{\mu} & M \end{array}$$

What they do:

Monads

What they are: Functors with structure.

$$\langle \mathcal{M},\ \eta : 1 \to \mathcal{M},\ \mu : \mathcal{M}\mathcal{M} \to \mathcal{M} \rangle$$

$\eta :$ x $\mapsto$ 

$\mu :$  $\mapsto$ 

$$\begin{array}{ccccc} & & M & & \\ & {\scriptstyle M\eta}\swarrow & \| & \searrow{\scriptstyle \eta M} & \\ MM & \xrightarrow{\mu} & M & \xleftarrow{\mu} & MM \end{array}$$

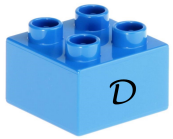
$$\begin{array}{ccc} MMM & \xrightarrow{M\mu} & MM \\ {\scriptstyle \mu M}\downarrow & & \downarrow{\scriptstyle \mu} \\ MM & \xrightarrow{\mu} & M \end{array}$$

What they do: Models of computation and datastructures.

non-determinism, probability, states, exceptions, ...

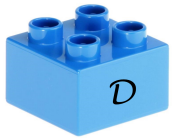
lists, trees, multisets, groups, ...

Probability Distributions Monad



$$\mathcal{D}(X) = \{\mu | \mu \text{ has finite support}\}$$
$$\eta_{\mathcal{D}}(x) = \delta_x$$
$$\mu_{\mathcal{D}}(e)(x) = \Sigma_{d \in \text{supp}(e)}\, e(d) \cdot d(x)$$

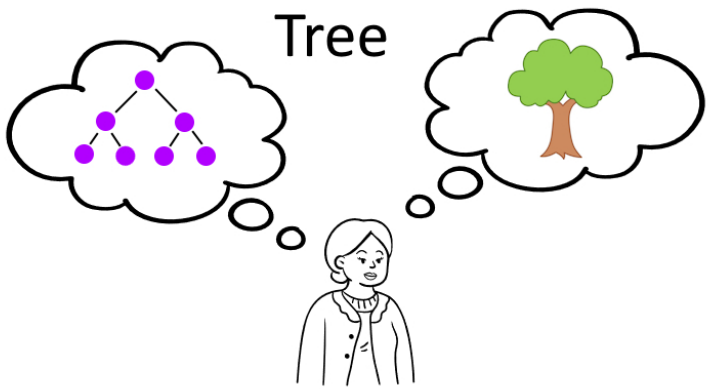
Probability Distributions Monad



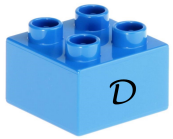
$$\mathcal{D}(X) = \{\mu | \mu \text{ has finite support}\}$$
$$\eta_{\mathcal{D}}(x) = \delta_x$$
$$\mu_{\mathcal{D}}(e)(x) = \Sigma_{d \in \text{supp}(e)}\, e(d) \cdot d(x)$$

Use probability monad to model uncertainty in NLP:

Homonyms:



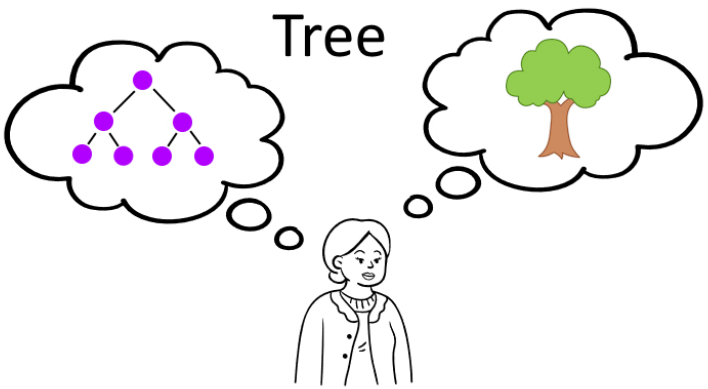
Probability Distributions Monad



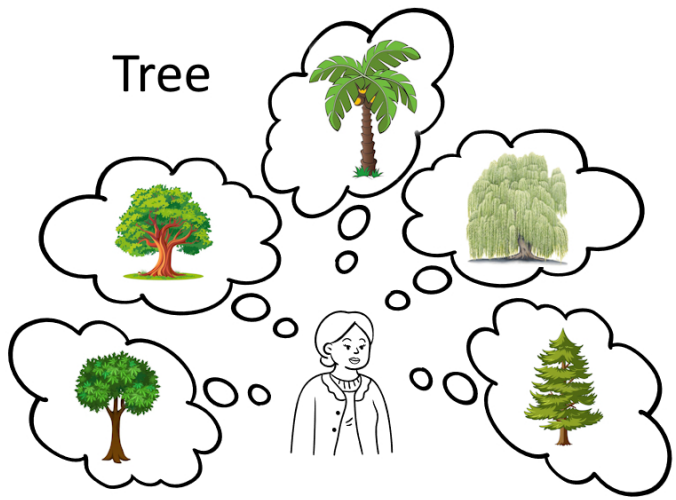
$$\mathcal{D}(X) = \{\mu | \mu \text{ has finite support}\}$$
$$\eta_{\mathcal{D}}(x) = \delta_x$$
$$\mu_{\mathcal{D}}(e)(x) = \Sigma_{d \in \text{supp}(e)}\, e(d) \cdot d(x)$$

Use probability monad to model uncertainty in NLP:

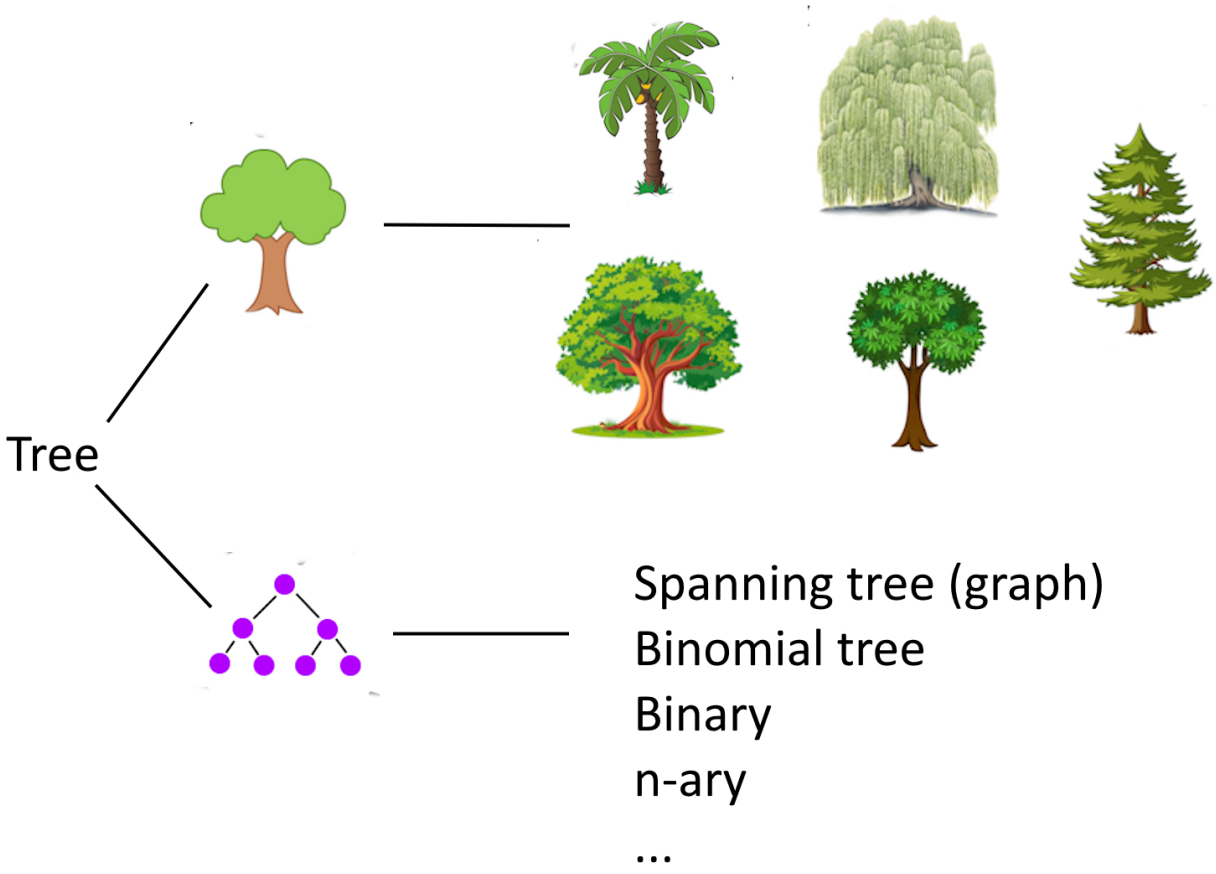
Homonyms:



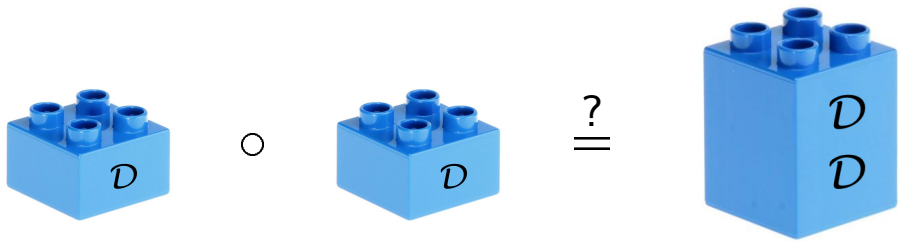
Hypernyms:



Homonyms and Hypernyms



Composing Monads



Composing Monads

Composing Monads

✓ Compose two functors TS.

Composing Monads

- ✓ Compose two functors TS.
- ✓ Compose two units $\eta^T \eta^S$.

Composing Monads

- ✓ Compose two functors TS.
- ✓ Compose two units $\eta^T \eta^S$.
- ? Compose two multiplications...

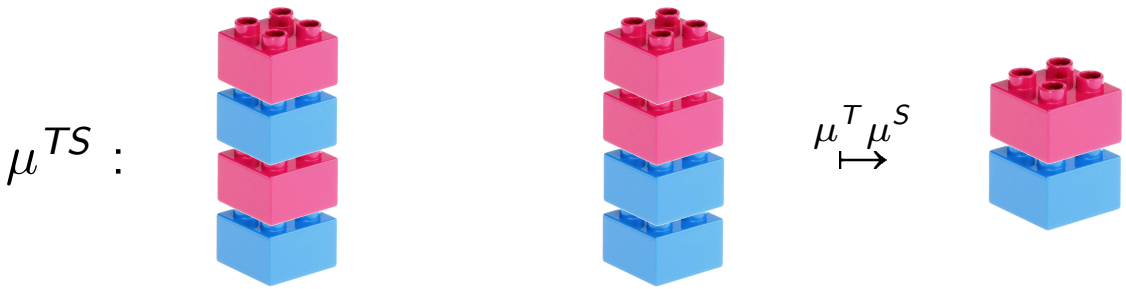
Composing Monads

- ✓ Compose two functors TS.
- ✓ Compose two units $\eta^T \eta^S$.
- ? Compose two multiplications...

μ^{TS} :

Composing Monads

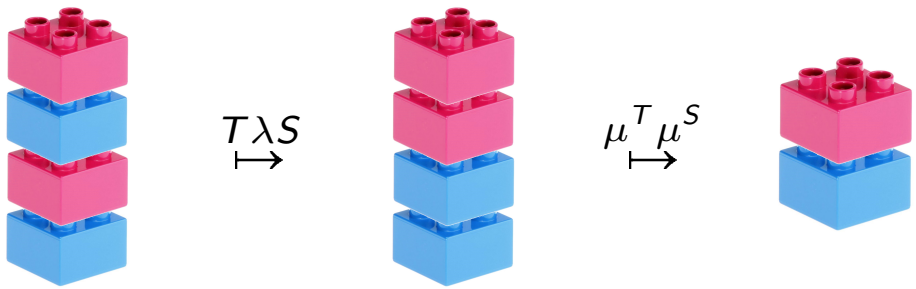
- ✓ Compose two functors TS.
- ✓ Compose two units $\eta^T \eta^S$.
- ? Compose two multiplications...



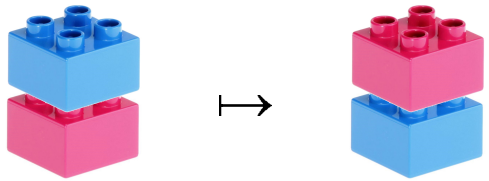
Composing Monads

- ✓ Compose two functors TS.
- ✓ Compose two units $\eta^T \eta^S$.
- ? Compose two multiplications...

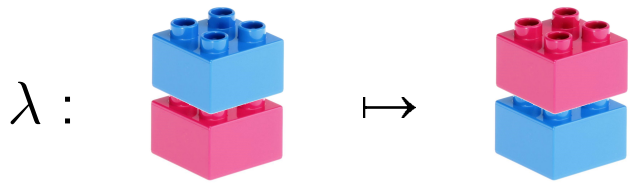
μ^{TS} :

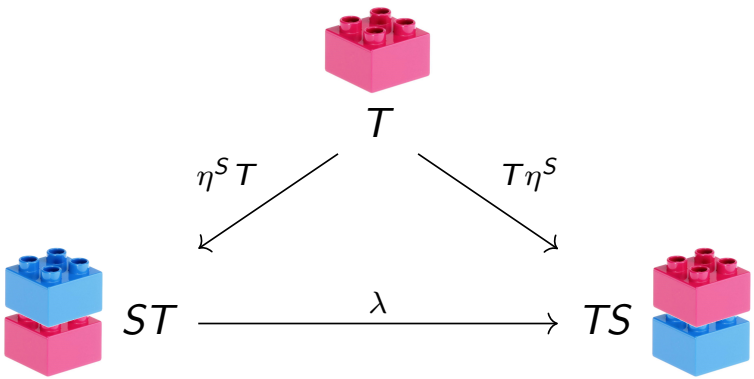


Distributive Law

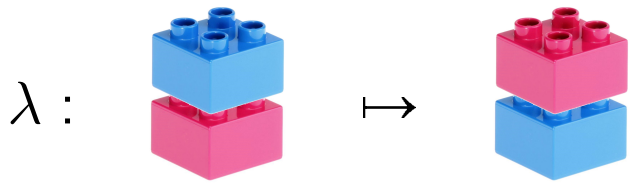
$\lambda :$  $\mapsto$

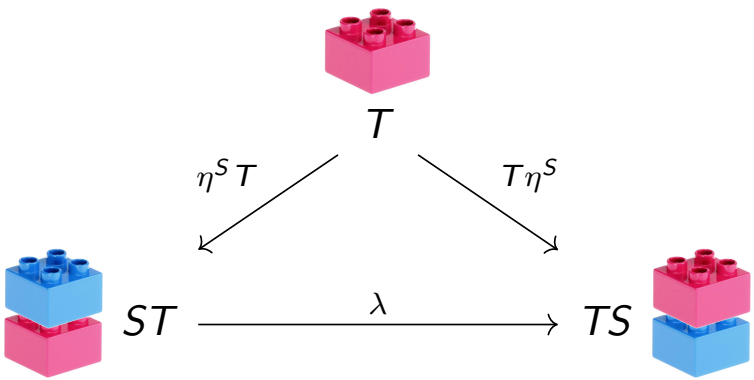
Distributive Law

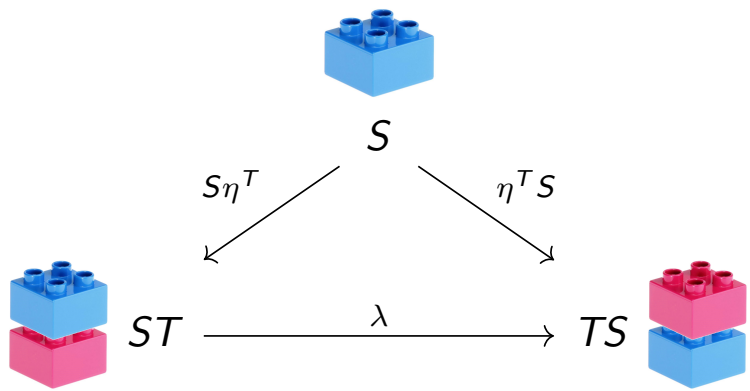




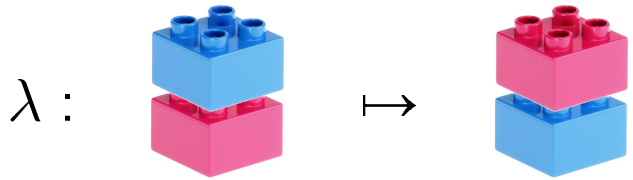
Distributive Law

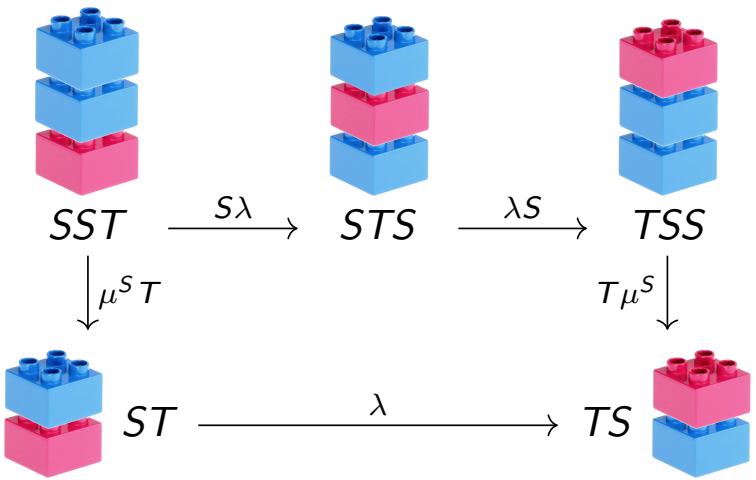




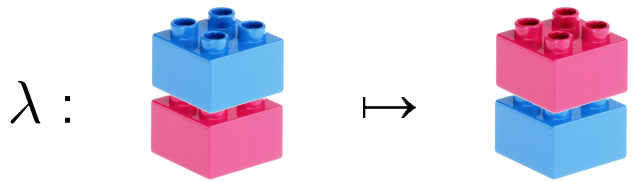


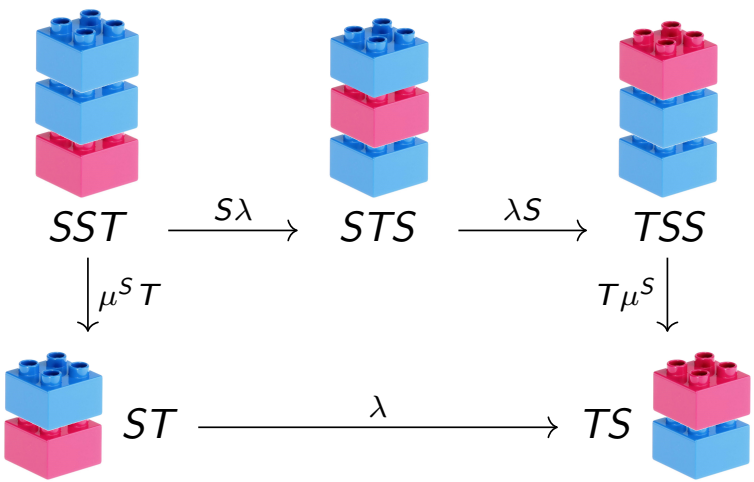
Distributive Law

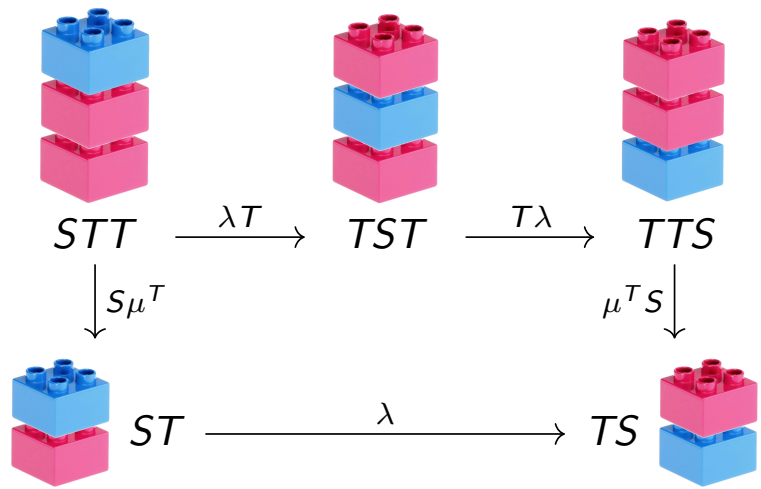




Distributive Law







Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Self-Distribution - Trivial?

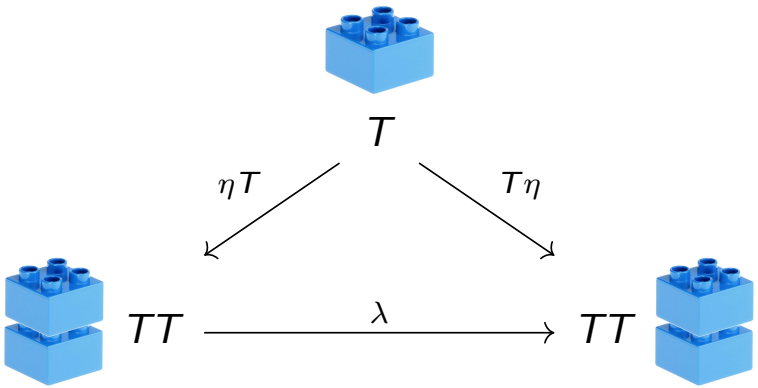
$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$?

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

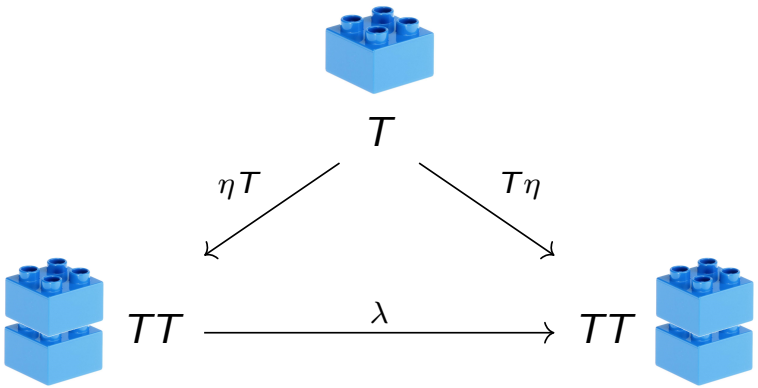
Can we take $\lambda = Id$?



Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

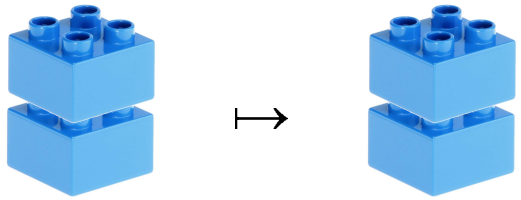
Can we take $\lambda = Id$?



$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

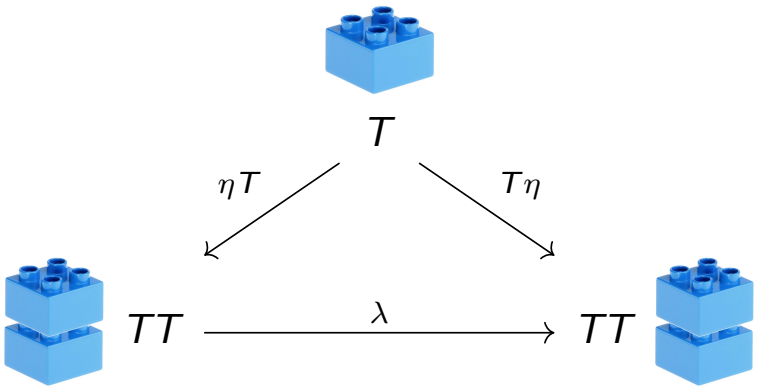
Example: completion monads, such as Groups $\rightarrow$ Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$

Can we take $\lambda = Id$?

Yes, iff T is an idempotent monad.



$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

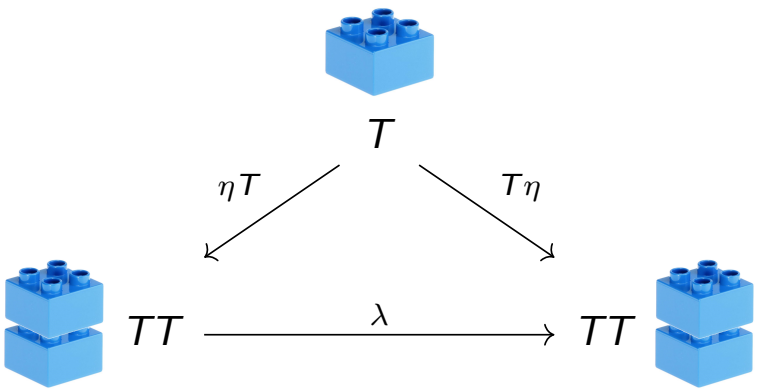
Example: completion monads, such as Groups → Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$?
Can we take $\lambda = \eta T \circ \mu$?

Yes, iff T is an idempotent monad.



$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

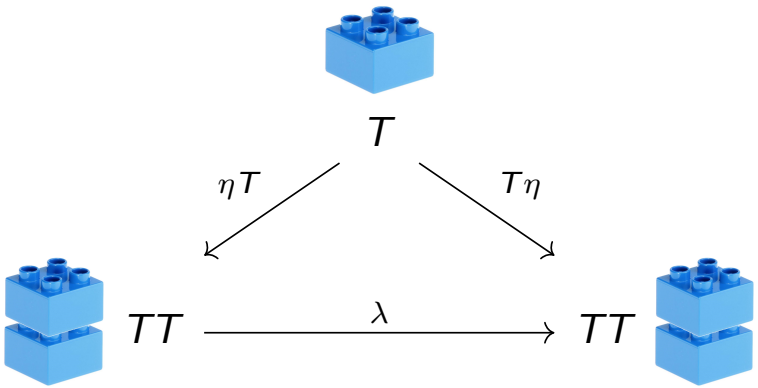
Example: completion monads, such as Groups $\rightarrow$ Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$? Yes, iff T is an idempotent monad.

Can we take $\lambda = \eta T \circ \mu$? Yes, iff T is an idempotent monad.



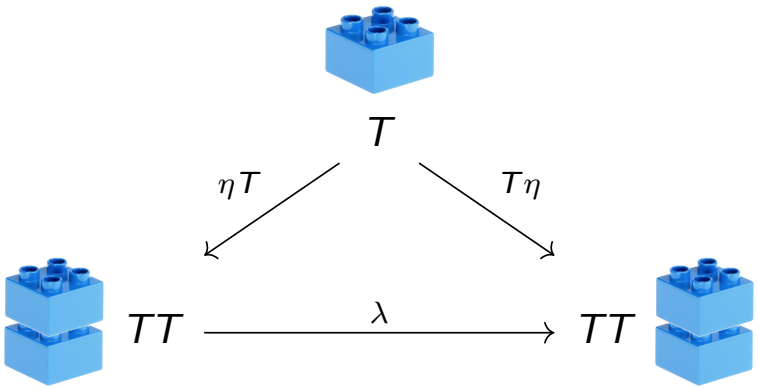
$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

Example: completion monads, such as Groups $\rightarrow$ Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$? Yes, iff T is an idempotent monad.
Can we take $\lambda = \eta T \circ \mu$? Yes, iff T is an idempotent monad.
Can we take $\lambda = T\eta \circ \mu$?



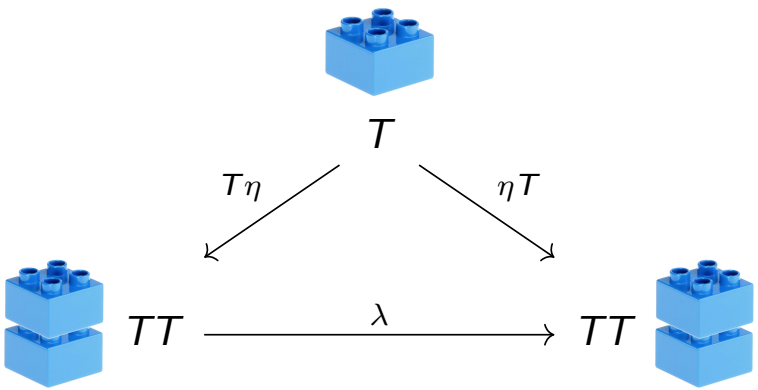
$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

Example: completion monads, such as Groups → Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$? Yes, iff T is an idempotent monad.
Can we take $\lambda = \eta T \circ \mu$? Yes, iff T is an idempotent monad.
Can we take $\lambda = T\eta \circ \mu$?



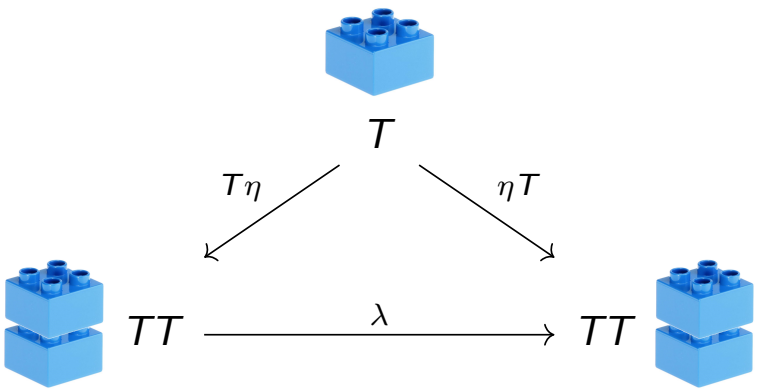
$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

Example: completion monads, such as Groups → Abelian groups

Self-Distribution - Trivial?

$\lambda :$  $\mapsto$ 

Can we take $\lambda = Id$? Yes, iff T is an idempotent monad.
Can we take $\lambda = \eta T \circ \mu$? Yes, iff T is an idempotent monad.
Can we take $\lambda = T\eta \circ \mu$? Yes, iff T is an idempotent monad.



$\eta^T T = T\eta^T$ is a defining property for idempotent monads.

Example: completion monads, such as Groups → Abelian groups

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\wr\wr a\int, \wr b, c\int\int \mapsto \wr\wr a, b\int, \wr a, c\int\int$$

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\wr\, \wr\, a \int, \wr b, c \int \int \mapsto \wr\, \wr\, a, b \int, \wr a, c \int \int$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{ \{\!\{ a \}\!\}, \{\!\{ b, c \}\!\} \}\!\} \mapsto \{\!\{ \{\!\{ a, b \}\!\}, \{\!\{ a, c \}\!\} \}\!\}$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

- ✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{\{\!\{a\}\!\}, \{\!\{b, c\}\!\}\}\!\} \mapsto \{\!\{\{\!\{a, b\}\!\}, \{\!\{a, c\}\!\}\}\!\}$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

- ✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

- ? List

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{ \{\!\{ a \}\!\}, \{\!\{ b, c \}\!\} \}\!\} \mapsto \{\!\{ \{\!\{ a, b \}\!\}, \{\!\{ a, c \}\!\} \}\!\}$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

- ✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

- ? List
- ? Powerset

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{\{\!\{a\}\!\}, \{\!\{b, c\}\!\}\}\!\} \mapsto \{\!\{\{\!\{a, b\}\!\}, \{\!\{a, c\}\!\}\}\!\}$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

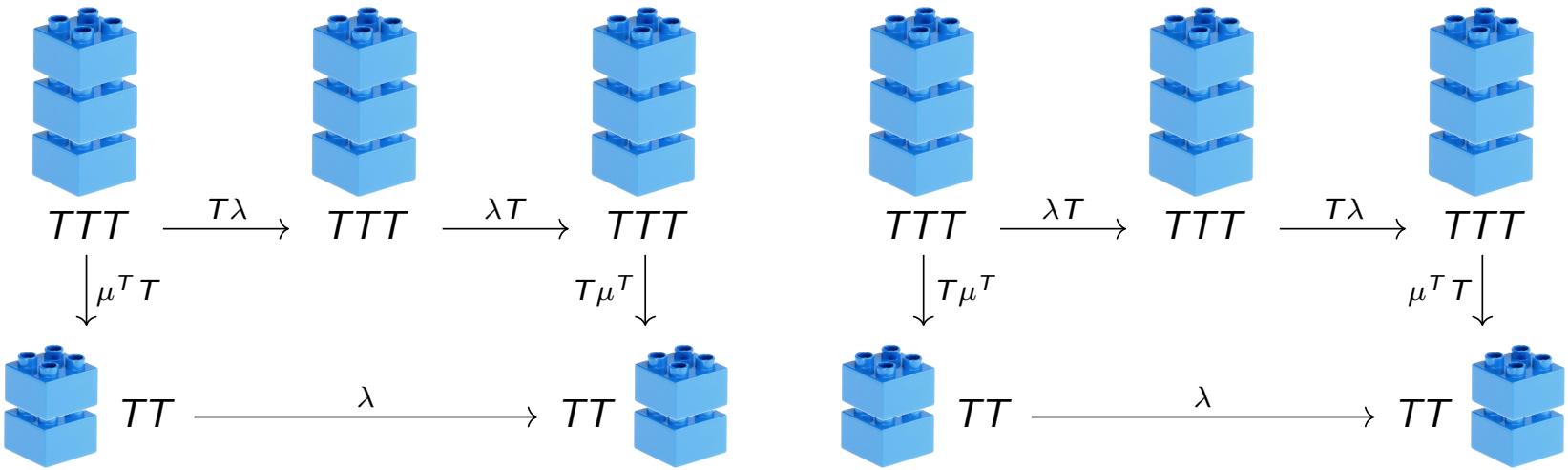
$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

- ✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

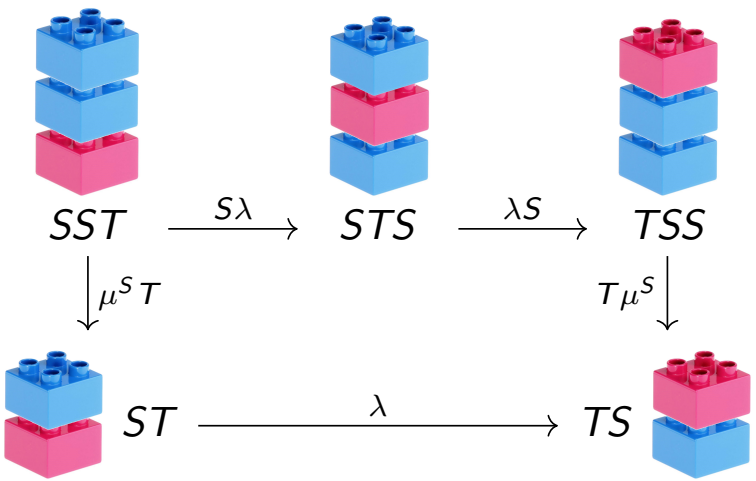
$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

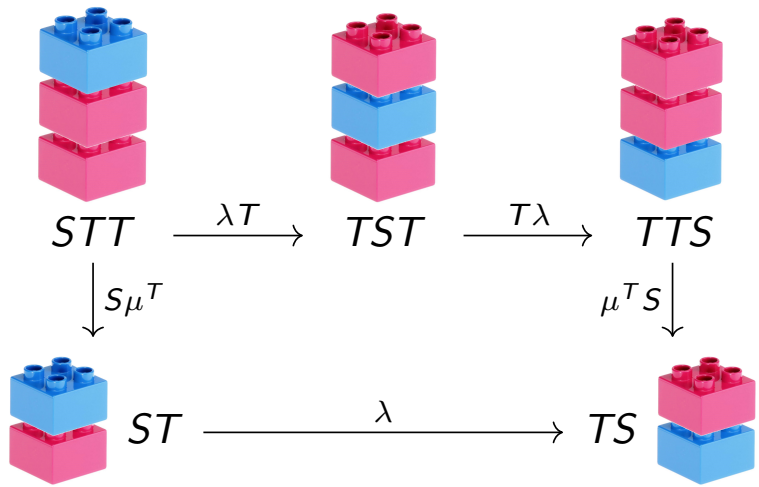
- ? List
- ? Powerset
- ? Probability Distributions

Self-Distribution - Fiddly!

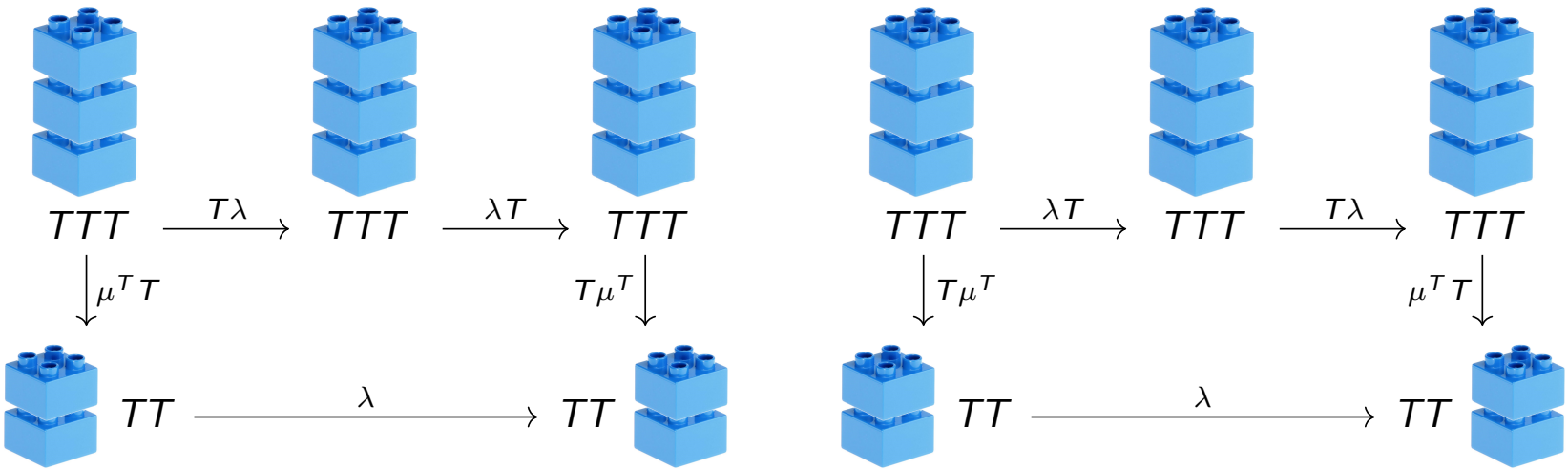


Self-Distribution - Fiddly!

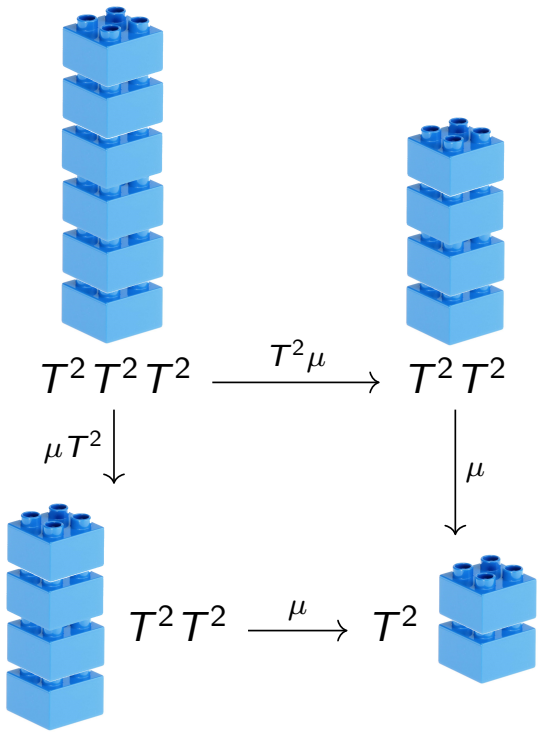




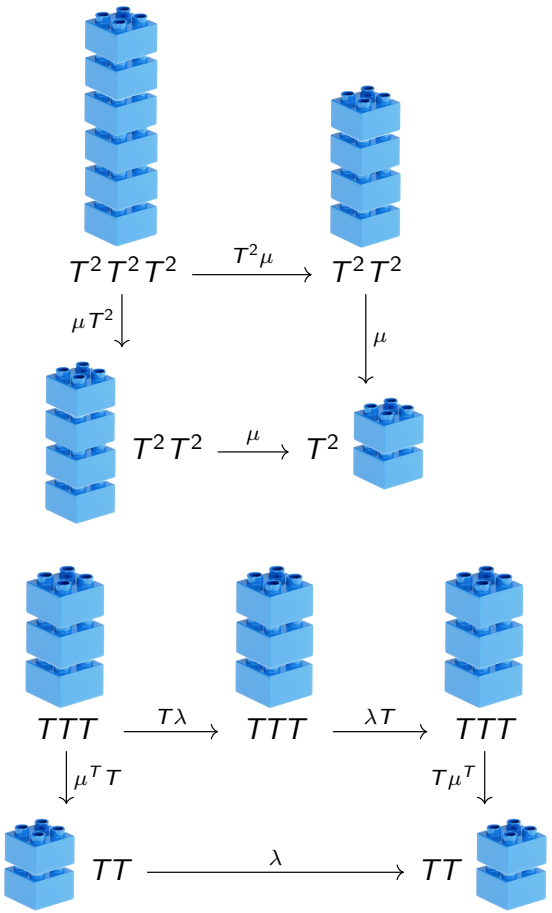
Self-Distribution - Fiddly!



Self-Distribution - Fiddly!



Self-Distribution - Fiddly!



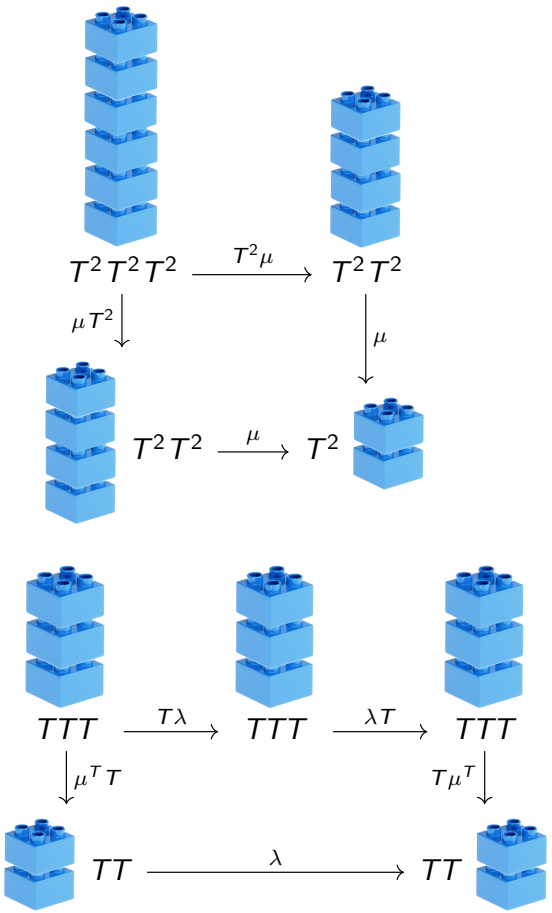
EXAMPLE 2.12. Let $P : \mathcal{S} \to \mathcal{S}$ be the *covariant* power set functor mapping X to 2^X but for $f : X \to Y$, $Pf(A)$ is the direct image $\{f(x) : x \in A\} \subset Y$. That this is a functor is routine. The *families monad* is (P^2, η, μ) where $\eta_X(x) = \{\{x\}\}$ and

$$\mu_X(\overline{\mathcal{A}}) = \left\{ \bigcup_{\mathcal{A} \in \Gamma} \Theta_{\mathcal{A}} : \Gamma \in \overline{\mathcal{A}}, (\Theta_{\mathcal{A}}) \in \prod_{\mathcal{A} \in \Gamma} \mathcal{A} \right\}$$

The reader is *not* invited to verify the axioms at this stage. We will first offer an equivalent definition of monad which has only three axioms and in which T is never iterated. This version of monad was first written down in [76, Exercise 1.3.12, p. 32].

- Manes 2003

Self-Distribution - Fiddly!



EXAMPLE 2.12. Let $P : \mathcal{S} \to \mathcal{S}$ be the *covariant* power set functor mapping X to 2^X but for $f : X \to Y$, $Pf(A)$ is the direct image $\{f(x) : x \in A\} \subset Y$. That this is a functor is routine. The *families monad* is (P^2, η, μ) where $\eta_X(x) = \{\{x\}\}$ and

$$\mu_X(\overline{\mathcal{A}}) = \left\{ \bigcup_{\mathcal{A} \in \Gamma} \Theta_{\mathcal{A}} : \Gamma \in \overline{\mathcal{A}}, (\Theta_{\mathcal{A}}) \in \prod_{\mathcal{A} \in \Gamma} \mathcal{A} \right\}$$

The reader is *not* invited to verify the axioms at this stage. We will first offer an equivalent definition of monad which has only three axioms and in which T is never iterated. This version of monad was first written down in [76, Exercise 1.3.12, p. 32].

- Manes 2003

5 Mistakes

We shall now summarise different sources in which it has been mistakenly concluded that $\mathcal{PP}$ is a monad.

- Klin and Salamanca 2018

Self-Distributions

- ✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{\{\!\{a\}\!\}, \{\!\{b, c\}\!\}\}\!\} \mapsto \{\!\{\{\!\{a, b\}\!\}, \{\!\{a, c\}\!\}\}\!\}$$

- ✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

- ✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

- ? List
- ? Powerset
- ? Probability Distributions

Self-Distributions

✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{\{\!\{a\}\!\}, \{\!\{b, c\}\!\}\}\!\} \mapsto \{\!\{\{\!\{a, b\}\!\}, \{\!\{a, c\}\!\}\}\!\}$$

✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

✗ List - Zwart and Marsden 2019

✗ Powerset - Klin and Salamanca 2018

✗ Probability Distributions - Zwart and Marsden 2019

Self-Distributions

✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{ \{\!\{ a \}\!\}, \{\!\{ b, c \}\!\} \}\!\} \mapsto \{\!\{ \{\!\{ a, b \}\!\}, \{\!\{ a, c \}\!\} \}\!\}$$

✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

x List - Zwart and Marsden 2019

x Powerset - Klin and Salamanca 2018

x Probability Distributions - Zwart and Marsden 2019

? Delay Monad / Guarded Delay Monad

Intermezzo - Delay Monad

A model for computation steps / computation time

Guarded recursive delay monad: $D^\kappa X \simeq X + \triangleright^\kappa DX$

- lives in guarded cubical type theory - constructive mathematics

Intermezzo - Delay Monad

A model for computation steps / computation time

Guarded recursive delay monad: $D^\kappa X \simeq X + \triangleright^\kappa D X$

- lives in guarded cubical type theory - constructive mathematics

Coinductive delay monad: $DX \simeq X + DX$

- lives in cubical type theory

Intermezzo - Delay Monad

A model for computation steps / computation time

Guarded recursive delay monad: $D^\kappa X \simeq X + \triangleright^\kappa D X$

- lives in guarded cubical type theory - constructive mathematics

Coinductive delay monad: $DX \simeq X + DX$

- lives in cubical type theory
- but also in **Set** - classical mathematics

Intermezzo - Delay Monad

A model for computation steps / computation time

Guarded recursive delay monad: $D^\kappa X \simeq X + \triangleright^\kappa DX$

- lives in guarded cubical type theory - constructive mathematics

Coinductive delay monad: $DX \simeq X + DX$

- lives in cubical type theory
- but also in **Set** - classical mathematics

So do these monads distribute over themselves?

Algebraic Perspective

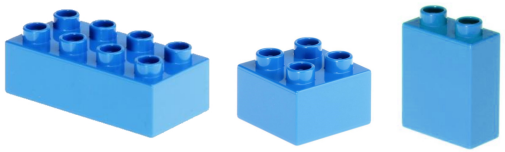
Monads $\Leftrightarrow$ Algebraic Theories

Distributive Laws $\Leftrightarrow$ Composite Theories

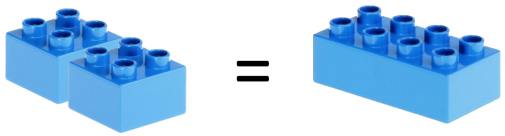
Recap: Algebraic Theories

Algebraic Theory:

A set Σ of operation symbols with arity.



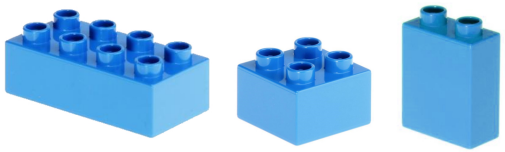
A set E of equations.



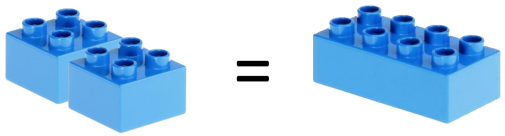
Recap: Algebraic Theories

Algebraic Theory:

A set Σ of operation symbols with arity.



A set E of equations.



Monoids:

$$\Sigma^M = \{c^{(0)}, *^{(2)}\}$$
$$E^M = \{x * c = x,$$
$$c * x = x,$$
$$(x * y) * z = x * (y * z)\ \}$$

Delay:

$$\Sigma^D = \{\bot^{(0)}, \text{step}^{(1)}\}$$
$$E^D = \{\text{step}(\bot) = \bot\}$$

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

For example:

$$\Sigma^T = \{*^2\} \text{  }, \; E^T = \varnothing \qquad \Sigma^S = \{\star^2\} \text{  }, \; E^S = \varnothing$$

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

For example:

$$\Sigma^T = \{*^2\}\ \text{  },\ E^T = \varnothing \qquad \Sigma^S = \{\star^2\}\ \text{  },\ E^S = \varnothing$$

Want: $t[s_x/x]$, so $(x \star y) * z$  but not $(x * y) \star z$  .

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

For example:

$$\Sigma^T = \{*^2\}\ \text{  },\ E^T = \varnothing \qquad \Sigma^S = \{\star^2\}\ \text{  },\ E^S = \varnothing$$

Want: $t[s_x/x]$, so $(x \star y) * z$  but not $(x * y) \star z$  .

Not possible to capture in Σ.

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

For example:

$$\Sigma^T = \{*^2\} \text{  }, \; E^T = \varnothing \qquad \Sigma^S = \{\star^2\} \text{  }, \; E^S = \varnothing$$

Want: $t[s_x/x]$, so $(x \star y) * z$  but not $(x * y) \star z$  .

Not possible to capture in Σ.

So make E such that all terms *are equal to* terms of form $t[s_x/x]$,

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

For example:

$$\Sigma^T = \{*^2\} \text{  }, \; E^T = \varnothing \qquad \Sigma^S = \{\star^2\} \text{  }, \; E^S = \varnothing$$

Want: $t[s_x/x]$, so $(x \star y) * z$  but not $(x * y) \star z$  .

Not possible to capture in Σ.

So make E such that all terms *are equal to* terms of form $t[s_x/x]$, without introducing any 'extra' equalities.

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

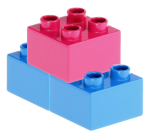
For example:

$$\Sigma^T = \{*^2\} \text{  }, \; E^T = \varnothing \qquad \Sigma^S = \{\star^2\} \text{  }, \; E^S = \varnothing$$

Want: $t[s_x/x]$, so $(x \star y) * z$  but not $(x * y) \star z$  .

Not possible to capture in Σ.

So make E such that all terms *are equal to* terms of form $t[s_x/x]$, without introducing any 'extra' equalities.

So: $(x * y) \star z = (x \star z) * (y \star z)$  $=$  ,

but not $(x * y) * z = x * y$  $=$  .

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

- $\Sigma_{T \circ_\lambda S} = \Sigma_T \uplus \Sigma_S$
- $E_{T \circ_\lambda S} = E_T \cup E_S \cup E_\lambda$

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

- $\Sigma_{T \circ_\lambda S} = \Sigma_T \uplus \Sigma_S$
- $E_{T \circ_\lambda S} = E_T \cup E_S \cup E_\lambda$

Such that:

- Every term of $T \circ_\lambda S$ can be written as $t[s_x/x]$,

Composite Theories

How to compose $T = (\Sigma_T, E_T)$ and $S = (\Sigma_S, E_S)$?

- $\Sigma_{T \circ_\lambda S} = \Sigma_T \uplus \Sigma_S$
- $E_{T \circ_\lambda S} = E_T \cup E_S \cup E_\lambda$

Such that:

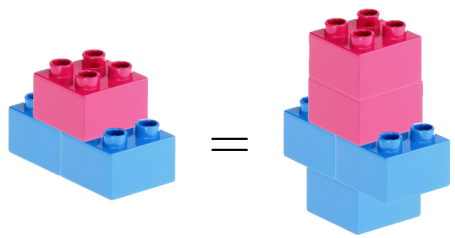
- Every term of $T \circ_\lambda S$ can be written as $t[s_x/x]$,
- in an essentially unique way, i.e. if $t[s_x/x] = t'[s_y/y]$, then there are variable substitutions f and g, such that:

$t[f(x)/x] =_T t'[g(y)/y]$.

$f(x_1) = f(x_2) \Leftrightarrow s_{x_1} =_S s_{x_2}$.

$g(y_1) = g(y_2) \Leftrightarrow s'_{y_1} =_S s'_{y_2}$.

$f(x) = g(y) \Leftrightarrow s_x =_S s'_y$.



Composite Theory - Self Composition

How to compose $T = (\Sigma_T, E_T)$ with itself?

Composite Theory - Self Composition

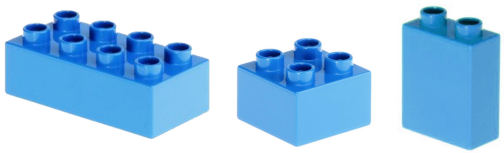
How to compose $T = (\Sigma_T, E_T)$ with itself?

- $\Sigma_{T \circ_\lambda T} = \Sigma_T \uplus \Sigma_T$

Composite Theory - Self Composition

How to compose $T = (\Sigma_T, E_T)$ with itself?

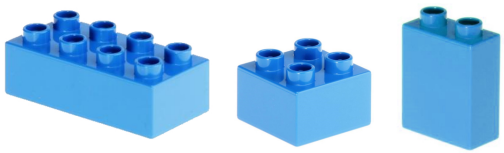
- $\Sigma_{T \circ_\lambda T} = \Sigma_T \uplus \Sigma_T$

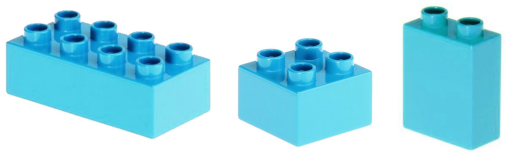


Composite Theory - Self Composition

How to compose $T = (\Sigma_T, E_T)$ with itself?

- $\Sigma_{T \circ_\lambda T} = \Sigma_T \uplus \Sigma_T$

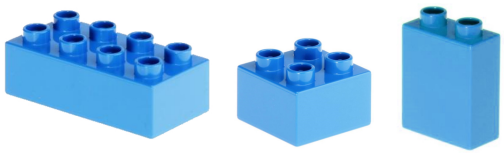


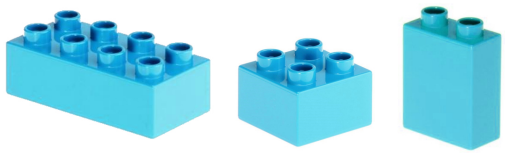


Composite Theory - Self Composition

How to compose $T = (\Sigma_T, E_T)$ with itself?

- $\Sigma_{T \circ_\lambda T} = \Sigma_T \uplus \Sigma_T$



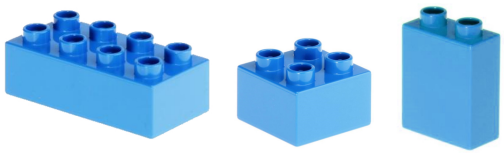


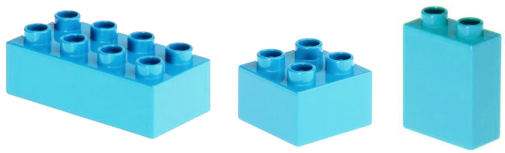
- $E_{T \circ_\lambda T} = E_T \cup E_T \cup E_\lambda$

Composite Theory - Self Composition

How to compose $T = (\Sigma_T, E_T)$ with itself?

- $\Sigma_{T \circ_\lambda T} = \Sigma_T \uplus \Sigma_T$





- $E_{T \circ_\lambda T} = E_T \cup E_T \cup E_\lambda$

Really no different from general composition!

Self Composition - Example

Example: Delay $\circ$ Delay

- $\Sigma_{D \circ_\lambda M} = \{\bot^{(0)}, \mathsf{step}^{(1)}, \bot^{(0)}, \mathsf{step}^{(1)}\}$
- $E_{D \circ_\lambda M} = \{\mathsf{step}(\bot) = \bot\} \cup \{\mathsf{step}(\bot) = \bot\} \cup E_\lambda$

Self Composition - Example

Example: Delay $\circ$ Delay

- $\Sigma_{D \circ_\lambda M} = \{\bot^{(0)}, \mathsf{step}^{(1)}, \bot^{(0)}, \mathsf{step}^{(1)}\}$
- $E_{D \circ_\lambda M} = \{\mathsf{step}(\bot) = \bot\} \cup \{\mathsf{step}(\bot) = \bot\} \cup E_\lambda$

$$E_\lambda = \{\, \mathsf{step}(\bot) = \bot \\ \mathsf{step}(\mathsf{step}(x)) = \mathsf{step}(\mathsf{step}(x)) \,\}$$

Self Composition - Example

Example: Delay $\circ$ Delay

- $\Sigma_{D\circ_\lambda M} = \{\bot^{(0)}, \mathsf{step}^{(1)}, \bot^{(0)}, \mathsf{step}^{(1)}\}$
- $E_{D\circ_\lambda M} = \{\mathsf{step}(\bot) = \bot\} \cup \{\mathsf{step}(\bot) = \bot\} \cup E_\lambda$

$$E_\lambda = \{\, \mathsf{step}(\bot) = \bot \qquad \mathsf{step}(\mathsf{step}(x)) = \mathsf{step}(\mathsf{step}(x)) \,\}$$

? $\quad \mathsf{step}(\bot) = \bot, \text{ or } \mathsf{step}(\bot) = \bot, \text{ or } \bot = \bot \quad$?

Self Composition - Example

Example: Delay $\circ$ Delay

- $\Sigma_{D \circ_\lambda M} = \{\bot^{(0)}, \mathsf{step}^{(1)}, \bot^{(0)}, \mathsf{step}^{(1)}\}$
- $E_{D \circ_\lambda M} = \{\mathsf{step}(\bot) = \bot\} \cup \{\mathsf{step}(\bot) = \bot\} \cup E_\lambda$

$$E_\lambda = \{\, \mathsf{step}(\bot) = \bot \qquad \mathsf{step}(\mathsf{step}(x)) = \mathsf{step}(\mathsf{step}(x)) \,\}$$

? $\quad \mathsf{step}(\bot) = \bot, \text{ or } \mathsf{step}(\bot) = \bot, \text{ or } \bot = \bot \quad$?

Would violate essential uniqueness!

Self Composition - Example

Example: Delay $\circ$ Delay

- $\Sigma_{D \circ_\lambda M} = \{\bot^{(0)}, \mathsf{step}^{(1)}, \bot^{(0)}, \mathsf{step}^{(1)}\}$
- $E_{D \circ_\lambda M} = \{\mathsf{step}(\bot) = \bot\} \cup \{\mathsf{step}(\bot) = \bot\} \cup E_\lambda$

$$E_\lambda = \{\, \mathsf{step}(\bot) = \bot \qquad \mathsf{step}(\mathsf{step}(x)) = \mathsf{step}(\mathsf{step}(x)) \,\}$$

? $\quad \mathsf{step}(\bot) = \bot, \text{ or } \mathsf{step}(\bot) = \bot, \text{ or } \bot = \bot \quad$?

Would violate essential uniqueness!

$$\mathsf{step}(x)[\bot/x] = x[\bot/x] \Rightarrow \mathsf{step}(x) = x$$

Composite Theory → Distributive Law

Given composite theory, then define a distributive law via:

$$\lambda : ST \to TS,\ \overline{s[\overline{t_x}^{T}/x]}^{S} \mapsto \overline{t'[\overline{s'_y}^{S}/y]}^{T}$$

Composite Theory → Distributive Law

Given composite theory, then define a distributive law via:

$$\lambda : ST \to TS,\ \overline{s[\overline{t_x}^{T}/x]}^{S} \mapsto \overline{t'[\overline{s'_y}^{S}/y]}^{T}$$

⇒ Self-distributive law for Delay monad $DX \simeq X + DX$ (coinductive version, in the classical setting):

$$\lambda(\mathrm{step}^n(\bot)) = \bot$$
$$\lambda(\mathrm{step}^n(\mathrm{step}^m(x))) = \mathrm{step}^m(\mathrm{step}^n(x))$$

Composite Theory → Distributive Law

Given composite theory, then define a distributive law via:

$$\lambda : ST \to TS,\ \overline{s[\overline{t_x}^{T}/x]}^{S} \mapsto \overline{t'[\overline{s'_y}^{S}/y]}^{T}$$

⇒ Self-distributive law for Delay monad $DX \simeq X + DX$ (coinductive version, in the classical setting):

$$\lambda(\text{step}^n(\bot)) = \bot$$
$$\lambda(\text{step}^n(\text{step}^m(x))) = \text{step}^m(\text{step}^n(x))$$

Self-distribution for guarded delay monad remains an open question.

Self-Distributions

✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{ \{\!\{ a \}\!\}, \{\!\{ b, c \}\!\} \}\!\} \mapsto \{\!\{ \{\!\{ a, b \}\!\}, \{\!\{ a, c \}\!\} \}\!\}$$

✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

x List - Zwart and Marsden 2019

x Powerset - Klin and Salamanca 2018

x Probability Distributions - Zwart and Marsden 2019

? Delay Monad / Guarded Delay Monad

Self-Distributions

✓ Multiset via "times over plus" - Manes and Mulry 2007

$$\{\!\{ \{\!\{ a \}\!\}, \{\!\{ b, c \}\!\} \}\!\} \mapsto \{\!\{ \{\!\{ a, b \}\!\}, \{\!\{ a, c \}\!\} \}\!\}$$

✓ List$^+$ via "rebracketing" - Manes and Mulry 2007

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b], [c], [d, e], [f]]$$

✓ List$^+$ via "firsts" or "lasts" - Manes and Mulry 2008

$$[[a], [b, c, d], [e, f]] \mapsto [[a, b, e]] \text{ or } [[a, d, f]]$$

x List - Zwart and Marsden 2019

x Powerset - Klin and Salamanca 2018

x Probability Distributions - Zwart and Marsden 2019

? Delay Monad / Guarded Delay Monad

? (Abelian) Groups

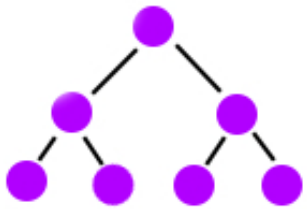
? Commutative Trees

Searching for structure

Boom Hierarchy: a tree of trees

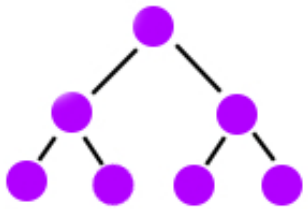
Searching for structure

Boom Hierarchy: a tree of trees



Searching for structure

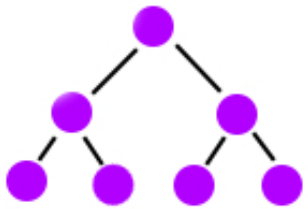
Boom Hierarchy: a tree of trees



- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

Searching for structure

Boom Hierarchy: a tree of trees

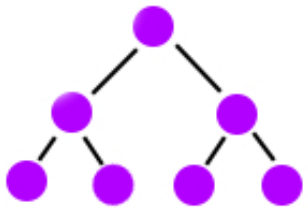


- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

(U) Unit

Searching for structure

Boom Hierarchy: a tree of trees



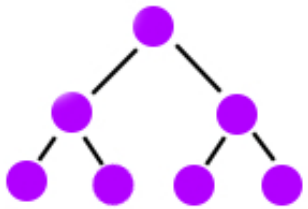
- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

(U) Unit

(A) Associativity

Searching for structure

Boom Hierarchy: a tree of trees



- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

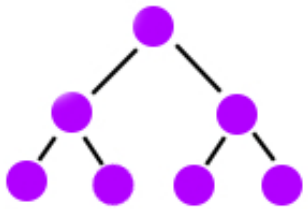
(U) Unit

(A) Associativity

(C) Commutativity

Searching for structure

Boom Hierarchy: a tree of trees



- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

(U) Unit

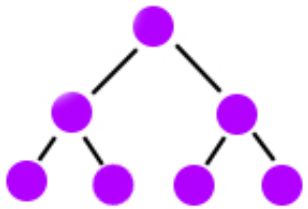
(A) Associativity

(C) Commutativity

(I) Idempotence

Searching for structure

Boom Hierarchy: a tree of trees



- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset

(U) Unit

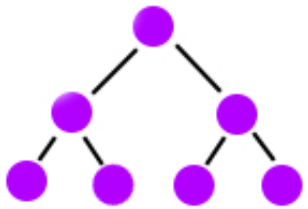
(A) Associativity

(C) Commutativity

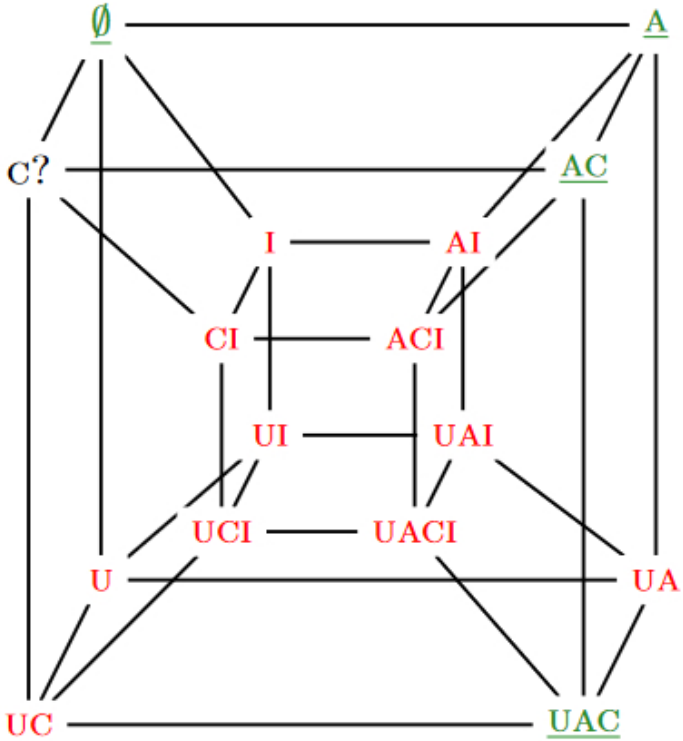
(I) Idempotence

Searching for structure

Boom Hierarchy: a tree of trees



- Tree
  - Unital Tree
    - List
      - Multiset
        - Powerset



(U) Unit

(A) Associativity

(C) Commutativity

(I) Idempotence

Conclusion

- Self distributions come up a lot in practice.
  - powerset, list, multiset, probability distributions, delay, and all kinds of trees

Conclusion

- Self distributions come up a lot in practice.
  - powerset, list, multiset, probability distributions, delay, and all kinds of trees
- Categorically, it is easy to make mistakes when working on self-distribution.

Conclusion

- Self distributions come up a lot in practice.
  - powerset, list, multiset, probability distributions, delay, and all kinds of trees
- Categorically, it is easy to make mistakes when working on self-distribution.
- Algebraic perspective treats self-distribution as any other distribution.

Conclusion

- Self distributions come up a lot in practice.
  - powerset, list, multiset, probability distributions, delay, and all kinds of trees
- Categorically, it is easy to make mistakes when working on self-distribution.
- Algebraic perspective treats self-distribution as any other distribution.
- Still fun to study self-distribution, as it yields interesting research questions and can guide intuition towards more general behaviour.