Distributive laws for Lawvere theories

Eugenia Cheng

University of Sheffield
CT2011
Plan

1. Introduction
Plan

1. Introduction
2. Lawvere theories
Plan

1. Introduction
2. Lawvere theories
3. Distributive laws for monads
Plan

1. Introduction
2. Lawvere theories
3. Distributive laws for monads
4. Three ways to do it
Plan

1. Introduction
2. Lawvere theories
3. Distributive laws for monads
4. Three ways to do it
5. Comparison.
Distributive laws give us a way of combining two or more types of algebraic structure expressed as monads.
1. Introduction

**Distributive laws** give us a way of combining two or more types of algebraic structure expressed as monads.

E.g. monoids and abelian groups → rings
Distributive laws give us a way of combining two or more types of algebraic structure expressed as monads.

E.g. monoids and abelian groups \rightarrow rings

Question
What’s a distributive law for Lawvere theories?
Distributive laws give us a way of combining two or more types of algebraic structure expressed as monads.

E.g. monoids and abelian groups \(\rightarrow\) rings

Question

What’s a distributive law for Lawvere theories?

- Lawvere theories correspond to finitary monads on \(\textit{Set}\).
1. Introduction

**Distributive laws** give us a way of combining two or more types of algebraic structure expressed as monads.

E.g. monoids and abelian groups \(\rightarrow\) rings

**Question**

What’s a distributive law for Lawvere theories?

- Lawvere theories correspond to finitary monads on \(\text{Set}\).
- Lawvere theories are themselves monads in a certain bicategory.
1. Introduction

**Distributive laws** give us a way of combining two or more types of algebraic structure expressed as monads.

E.g. monoids and abelian groups $\longrightarrow$ rings

**Question**

What’s a distributive law for Lawvere theories?

- Lawvere theories correspond to finitary monads on $\text{Set}$.
- Lawvere theories are themselves monads in a certain bicategory.

—So we can look for distributive laws between these monads.
1. Introduction

- A monad on \( \mathcal{V} \) only gives algebras in \( \mathcal{V} \).
- A Lawvere theory gives models in any finite-product category.
1. Introduction

- A monad on $\mathcal{V}$ only gives algebras in $\mathcal{V}$.
- A Lawvere theory gives models in any finite-product category.

![Diagram](chart.png)
1. Introduction

- A monad on \( \mathcal{V} \) only gives algebras in \( \mathcal{V} \).
- A Lawvere theory gives models in any finite-product category.
1. Introduction

- A monad on $\mathcal{V}$ only gives algebras in $\mathcal{V}$.
- A Lawvere theory gives models in any finite-product category.
1. Introduction

- A monad on $\mathcal{V}$ only gives algebras in $\mathcal{V}$.
- A Lawvere theory gives models in any finite-product category.
1. Introduction

- A monad on $\mathcal{V}$ only gives algebras in $\mathcal{V}$.
- A Lawvere theory gives models in any finite-product category.

Example

Distributive law for monoids over abelian groups

\[\text{rings internal to any finite-product category } \mathcal{V}.\]
2. Lawvere theories
2. Lawvere theories

Idea
Encapsulate an algebraic theory in a category $\mathbb{L}$. 
2. Lawvere theories

Idea
Encapsulate an algebraic theory in a category $\mathbb{L}$.

- The objects of $\mathbb{L}$ are the natural numbers, our *arities*.
- A morphism $k \rightarrow 1$ is an operation of arity $k$.
- A morphism $k \rightarrow m$ is $m$ operations of arity $k$. 
2. Lawvere theories

Idea
Encapsulate an algebraic theory in a category $\mathbb{L}$.

- The objects of $\mathbb{L}$ are the natural numbers, our *arities*.
- A morphism $k \rightarrow 1$ is an operation of arity $k$.
- A morphism $k \rightarrow m$ is $m$ operations of arity $k$.

We use $\mathbb{F}$ a skeleton of $\textbf{FinSet}$ (finite sets and functions).
2. Lawvere theories

Idea
Encapsulate an algebraic theory in a category $\mathbb{L}$.

- The objects of $\mathbb{L}$ are the natural numbers, our *arities*.
- A morphism $k \rightarrow 1$ is an operation of arity $k$.
- A morphism $k \rightarrow m$ is $m$ operations of arity $k$.

We use $\mathbb{F}$ a skeleton of $\text{FinSet}$ (finite sets and functions).

Definition
A Lawvere theory is a small category $\mathbb{L}$ with finite products, equipped with a strict identity-on-objects functor

$$\mathbb{F}^{\text{op}} \rightarrow \mathbb{L}.$$
2. Lawvere theories

Idea
Encapsulate an algebraic theory in a category $\mathbb{L}$.

- The objects of $\mathbb{L}$ are the natural numbers, our arities.
- A morphism $k \rightarrow 1$ is an operation of arity $k$.
- A morphism $k \rightarrow m$ is $m$ operations of arity $k$.

We use $\mathbb{F}$ a skeleton of $\text{FinSet}$ (finite sets and functions).

Definition
A Lawvere theory is a small category $\mathbb{L}$ with finite products, equipped with a strict identity-on-objects functor

$$\mathbb{F}^{\text{op}} \rightarrow \mathbb{L}.$$  

Note: in $\mathbb{F}^{\text{op}}$ the object $m$ is the product of $m$ copies of 1.
2. Lawvere theories

Note
We are allowed to forget and repeat variables.
2. Lawvere theories

Note
We are allowed to forget and repeat variables.

Example
2-ary operations in the theory of monoids
2. Lawvere theories

Note
We are allowed to forget and repeat variables.

Example
2-ary operations in the theory of monoids
• (non-$\Sigma$) operads: only one i.e. $ab$
2. Lawvere theories

**Note**
We are allowed to forget and repeat variables.

**Example**
2-ary operations in the theory of monoids

- (non-$\Sigma$) operads: only one i.e. $ab$

- Lawvere theory: $ab, a, a^2, b, b^2, aba, ab^3a^5, \ldots$
2. Lawvere theories

Note
We are allowed to forget and repeat variables.

Example
2-ary operations in the theory of monoids

- (non-$\Sigma$) operads: only one i.e. $ab$
- Lawvere theory: $ab, a, a^2, b, b^2, aba, ab^3 a^5, \ldots$
  i.e. everything in the free monad on $\{a, b\}$.
2. Lawvere theories

Note
We are allowed to forget and repeat variables.

Example
2-ary operations in the theory of monoids

- (non-$\Sigma$) operads: only one i.e. $ab$
- Lawvere theory: $ab, a, a^2, b, b^2, aba, ab^3a^5, \ldots$
  
  i.e. everything in the free monad on $\{a, b\}$.

A morphism $3 \rightarrow 2$ is two 3-ary operations e.g.

$$(ab, a^3), (a^2b, abc), \ldots$$
2. Lawvere theories

Note

We are allowed to forget and repeat variables.

Example

2-ary operations in the theory of monoids

- (non-$\Sigma$) operads: only one i.e. $ab$
- Lawvere theory: $ab, a, a^2, b, b^2, aba, ab^3a^5, \ldots$
  i.e. everything in the free monad on $\{a, b\}$.

A morphism $3 \rightarrow 2$ is two 3-ary operations e.g.

$$(ab, a^3), (a^2b, abc), \ldots$$

Composition: $3 \xrightarrow{\{ab, a^3\}} 2 \xrightarrow{x^2y} 1$
2. Lawvere theories

Note
We are allowed to forget and repeat variables.

Example
2-ary operations in the theory of monoids

- (non-$\Sigma$) operads: only one i.e. $ab$
- Lawvere theory: $ab, a, a^2, b, b^2, aba, ab^3a^5, \ldots$
  i.e. everything in the free monad on $\{a, b\}$.

A morphism $3 \rightarrow 2$ is two 3-ary operations e.g.

$$(ab, a^3), (a^2b, abc), \ldots$$

Composition:

$$\begin{array}{ccc}
3 & \xrightarrow{\{ab, a^3\}} & 2 & \xrightarrow{x^2y} & 1 \\
& ab.ab.a^3 & & & \\
\end{array}$$
2. Lawvere theories

We have many arities for the “same” operation.

<table>
<thead>
<tr>
<th>arity</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$a, b, c$</td>
</tr>
<tr>
<td>4</td>
<td>$a, b, c, d$</td>
</tr>
<tr>
<td>5</td>
<td>$a, b, c, d, e$</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>
## 2. Lawvere theories

We have many arities for the “same” operation.

<table>
<thead>
<tr>
<th>arity</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$a, b, c$</td>
</tr>
<tr>
<td>4</td>
<td>$a, b, c, d$</td>
</tr>
<tr>
<td>5</td>
<td>$a, b, c, d, e$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

These are all related by forgetting variables i.e. via projections in $\mathbb{F}^{\text{op}}$. 
2. Lawvere theories

Generalisations
2. Lawvere theories

Generalisations

- use $F = \text{FinSet}$ instead of a skeleton
2. Lawvere theories

Generalisations

- use $F = \text{FinSet}$ instead of a skeleton
- put $\mathcal{P} = \text{“free finite product category” 2-monad}$
  note that $\text{FinSet}^{\text{op}}$ is $\mathcal{P}1$
  —could use $\mathcal{P}\Delta$ to get “typed” theory
Generalisations

- use $F = \text{FinSet}$ instead of a skeleton

- put $P = \text{“free finite product category” 2-monad}$
  note that $\text{FinSet}^{\text{op}}$ is $P1$
  —could use $PA$ to get “typed” theory

- could just say a Lawvere theory is any finite product category $C$
2. Lawvere theories

Generalisations

• use $F = \text{FinSet}$ instead of a skeleton

• put $P = \text{“free finite product category”}$ 2-monad
  note that $\text{FinSet}^{\text{op}}$ is $P1$
  —could use $\mathcal{P}A$ to get “typed” theory

• could just say a Lawvere theory is $\text{any}$ finite product
  category $\mathcal{C}$

• could do finite limits instead of just products.
Models \equiv \text{algebras}

A model for \mathbb{L} in a finite-product category \mathcal{C} is a finite-product preserving functor

\begin{align*}
\mathbb{L} & \rightarrow \mathcal{C}
\end{align*}
2. Lawvere theories

Models ≡ algebras

A model for \( \mathbb{L} \) in a finite-product category \( \mathcal{C} \) is a finite-product preserving functor

\[
\mathbb{L} \longrightarrow \mathcal{C}
\]

Idea

\[
1 \longrightarrow A \in \mathcal{C} \quad \text{underlying data}
\]
2. Lawvere theories

Models $\equiv$ algebras

A model for $\mathbb{L}$ in a finite-product category $\mathcal{C}$ is a finite-product preserving functor

$$\mathbb{L} \longrightarrow \mathcal{C}$$

Idea

$$1 \quad \longrightarrow \quad A \in \mathcal{C} \quad \text{underlying data}$$

$$k \quad \quad \quad \quad \longrightarrow \quad A^k$$
2. Lawvere theories

Models $\equiv$ algebras

A model for $\mathbb{L}$ in a finite-product category $\mathcal{C}$ is a finite-product preserving functor

$$\mathbb{L} \longrightarrow \mathcal{C}$$

Idea

$$1 \overset{}{\longrightarrow} A \in \mathcal{C} \quad \text{underlying data}$$

$$k \overset{}{\longrightarrow} A^k \quad \text{operation of arity } k$$

\[\begin{array}{ccc}
1 & \overset{}{\longrightarrow} & \downarrow \\
& & A \\
k & \overset{}{\longrightarrow} & A^k \\
\end{array}\]
2. Lawvere theories

Lawvere theories vs monads
2. Lawvere theories

Lawvere theories vs monads

Lawvere theory                      monad
2. Lawvere theories

Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism</td>
<td>“$k$-ary operation”</td>
</tr>
<tr>
<td>$k \rightarrow 1$</td>
<td></td>
</tr>
</tbody>
</table>
2. Lawvere theories

Lawvere theories vs monads

Lawvere theory

- morphism
- $k \rightarrow 1$
- “$k$-ary operation”

Monad

- element of $T([k])$
2. Lawvere theories

Lawvere theories vs monads

Lawvere theory

morphism $k \rightarrow 1$

“$k$-ary operation”

monad

element of $T([k])$

i.e. $1 \rightarrow T([k]) \in \text{Set}$

set of $k$ elements
2. Lawvere theories

Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism $k \rightarrow 1$</td>
<td>“$k$-ary operation”</td>
</tr>
<tr>
<td>morphism $k \rightarrow m$</td>
<td>“$m$ operations of arity $k$”</td>
</tr>
</tbody>
</table>
## 2. Lawvere theories

### Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism</td>
<td>element of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow 1$</td>
<td>i.e. $1 \rightarrow T([k]) \in \textbf{Set}$</td>
</tr>
<tr>
<td>“$k$-ary operation”</td>
<td>$m$ elements of $T([k])$</td>
</tr>
<tr>
<td>morphism</td>
<td>$m$ elements of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow m$</td>
<td>$m$ elements of $T([k])$</td>
</tr>
<tr>
<td>“$m$ operations of arity $k$”</td>
<td></td>
</tr>
</tbody>
</table>

Set of $k$ elements
2. Lawvere theories

Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism</td>
<td>element of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow 1$</td>
<td>i.e. $1 \rightarrow T([k]) \in \text{Set}$</td>
</tr>
<tr>
<td>morphism</td>
<td>$m$ elements of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow m$</td>
<td>i.e. $[m] \rightarrow T([k]) \in \text{Set}$</td>
</tr>
</tbody>
</table>

$k$-ary operation

set of $k$ elements
## 2. Lawvere theories

### Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism</td>
<td>element of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow 1$</td>
<td>i.e. $1 \rightarrow T([k]) \in \text{Set}$</td>
</tr>
<tr>
<td>morphism</td>
<td>$m$ elements of $T([k])$</td>
</tr>
<tr>
<td>$k \rightarrow m$</td>
<td>i.e. $[m] \rightarrow T([k]) \in \text{Set}$</td>
</tr>
<tr>
<td></td>
<td>i.e. $[m] \rightarrow [k] \in \text{Kl} T$</td>
</tr>
</tbody>
</table>
## 2. Lawvere theories

### Lawvere theories vs monads

<table>
<thead>
<tr>
<th>Lawvere theory</th>
<th>monad</th>
</tr>
</thead>
<tbody>
<tr>
<td>morphism ( k \to 1 ) “( k )-ary operation”</td>
<td>element of ( T([k]) ) i.e. ( 1 \to T([k]) \in \text{Set} )</td>
</tr>
<tr>
<td>morphism ( k \to m ) “( m ) operations of arity ( k )”</td>
<td>( m ) elements of ( T([k]) ) i.e. ( [m] \to T([k]) \in \text{Set} ), i.e. ( [m] \to [k] \in \text{Kl}T )</td>
</tr>
</tbody>
</table>

**Idea**

Lawvere theories are related to monads via the Kleisli category.
2. Lawvere theories

Definition

Monad $T$ on $\textbf{Set}$ \quad $\longrightarrow$ \quad Lawvere theory $\mathbb{L}_T$

$$\mathbb{L}_T = \text{full subcategory of } (\textbf{Kl}T)^{\text{op}}$$

whose objects are finite sets.
2. Lawvere theories

Definition

Monad $T$ on $\textbf{Set}$ \quad $\longrightarrow$ \quad Lawvere theory $\mathbb{L}_T$

$\mathbb{L}_T = \text{full subcategory of } (\mathbf{Kl}T)^{\text{op}}$

whose objects are \textit{finite} sets.

Lawvere theory $\mathbb{L}$ \quad $\longrightarrow$ \quad monad $T_\mathbb{L}$ on $\textbf{Set}$

$$T_\mathbb{L}X = \int_{n \in \mathbb{F}^{\text{op}}} \mathbb{L}(n, 1) \times X^n.$$
2. Lawvere theories

Definition

Monad $T$ on $\textbf{Set}$ $\xrightarrow{}$ Lawvere theory $\mathbb{L}_T$

\[ \mathbb{L}_T = \text{full subcategory of } (\textbf{Kl}T)^{\text{op}} \]

whose objects are finite sets.

Lawvere theory $\mathbb{L}$ $\xrightarrow{}$ monad $T_{\mathbb{L}}$ on $\textbf{Set}$

\[ T_{\mathbb{L}}X = \int_{n \in \mathbb{F}^{\text{op}}} \mathbb{L}(n, 1) \times X^n. \]

Theorem

This gives a correspondence between Lawvere theories and finitary monads on $\textbf{Set}$. 
3. Distributive laws for monads
3. Distributive laws for monads

Idea
Given monads $S$ and $T$ on $\mathcal{C}$, can we make $TS$ into a monad?
3. Distributive laws for monads

Idea
Given monads $S$ and $T$ on $\mathcal{C}$, can we make $TS$ into a monad?

$$TSTS \overset{?}{\longrightarrow} TTSS \overset{\mu^T \mu^S}{\longrightarrow} TS$$
3. Distributive laws for monads

Idea
Given monads $S$ and $T$ on $\mathcal{C}$, can we make $TS$ into a monad?

\[
TSTS \xrightarrow{?} TTSS \xrightarrow{\mu^T \mu^S} TS
\]

Definition (Beck)
A distributive law of monads $S$ over $T$ consists of a natural transformation

\[
\lambda : ST \Rightarrow TS
\]

satisfying some axioms.
3. Distributive laws for monads

Idea
Given monads $S$ and $T$ on $\mathcal{C}$, can we make $TS$ into a monad?

$$TSTS \xrightarrow{?} TTSS \xrightarrow{\mu^T \mu^S} TS$$

Definition (Beck)
A distributive law of monads $S$ over $T$ consists of a natural transformation

$$\lambda : ST \Rightarrow TS$$

satisfying some axioms.

- **Formal theory of monads (Street)**
  Do this inside any bicategory, not just $\text{Cat}$. 
3. Distributive laws for monads

Idea
Given monads $S$ and $T$ on $\mathbf{C}$, can we make $TS$ into a monad?

$$TSTS \xrightarrow{?} TTSS \xrightarrow{\mu^T \mu^S} TS$$

Definition (Beck)
A distributive law of monads $S$ over $T$ consists of a natural transformation

$$\lambda : ST \Rightarrow TS$$

satisfying some axioms.

- **Formal theory of monads (Street)**
  Do this inside any bicategory, not just $\mathbf{Cat}$.

- **Iterated distributive laws (Cheng)**
  Combine $n$ monads with distributive laws and Yang-Baxter condition.
3. Distributive laws for monads

Examples

monoid + abelian group → ring

horizontal composition + vertical composition → 2-category
3. Distributive laws for monads

**Examples**

monoid + abelian group → ring

horizontal composition + vertical composition → 2-category

Or combining more structures:

\[
\begin{aligned}
\text{0-composition} &+ \\
\text{1-composition} &+ \\
&\vdots \\
\text{}(n-1)\text{-composition} &+ \\
\end{aligned}
\]
→ \( n \)-category
3. Distributive laws for monads

A point of view

- The monad $TS$ says we can express all structure as “$S$-structure followed by $T$-structure”.

3. Distributive laws for monads

A point of view

- The monad $TS$ says we can express all structure as “$S$-structure followed by $T$-structure”.

- The distributive law $ST \rightarrow TS$ says “if we had it the other way round we could switch it over”.

3. Distributive laws for monads

A point of view

• The monad $TS$ says we can express all structure as “$S$-structure followed by $T$-structure”.

• The distributive law $ST \to TS$ says “if we had it the other way round we could switch it over”.

For Lawvere theories

We want a way of combining $A$ and $B$ to give $BA$ corresponding to a distributive law of monads

$$T_A T_B \to T_B T_A$$
3. Distributive laws for monads

A point of view

- The monad $TS$ says we can express all structure as “$S$-structure followed by $T$-structure”.

- The distributive law $ST \rightarrow TS$ says “if we had it the other way round we could switch it over”.

For Lawvere theories

We want a way of combining $A$ and $B$ to give $BA$ corresponding to a distributive law of monads

\[ T_A T_B \rightarrow T_B T_A \]

with

\[ T_B T_A = T_{BA} \]
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$.
   
   —Rosebrugh and Wood, Distributive laws and factorization (JPAA 2002)
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$.
   —Rosebrugh and Wood, Distributive laws and factorization (JPAA 2002)

2. Profunctors internal to $\text{Mon}$.  
   —Lack, Composing props (TAC 2004)  
   —Akhvlediani, Composing Lawvere theories (CT2010)
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$.
   —Rosebrugh and Wood, Distributive laws and factorization (JPAA 2002)

2. Profunctors internal to $\text{Mon}$.
   —Lack, Composing props (TAC 2004)
   —Akhvlediani, Composing Lawvere theories (CT2010)

3. Kleisli bicategory of $\mathcal{P}$ on profunctors
   —Hyland, Distributive laws (CLP 2010)
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$
4. Three ways to do it

1. Factorisation systems over $F^{\text{op}}$

In the composite theory $\mathcal{B}\mathcal{A}$ every morphism can be expressed as a composite

$$\epsilon_{\mathcal{A}} \rightarrow \epsilon_{\mathcal{B}}$$
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$

In the composite theory $\mathbb{BA}$ every morphism can be expressed as a composite

$$
\xrightarrow{\in A} \quad \xrightarrow{\in B}
$$

For example: $\times$ and $+$

The composite 3-ary operation $a(b + c)$ can be expressed as
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$

In the composite theory $\mathbb{BA}$ every morphism can be expressed as a composite

$$\epsilon_A \quad \epsilon_B$$

For example: $\times$ and $+$

The composite 3-ary operation $a(b + c)$ can be expressed as

$$3 \quad 2 \quad 1$$

$ab, ac \quad x + y$
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{op}$

In the composite theory $\mathbb{BA}$ every morphism can be expressed as a composite

$$\in \mathbb{A} \quad \in \mathbb{B}$$

For example: $\times$ and $+$

The composite 3-ary operation $a(b + c)$ can be expressed as

$$ab, ac, ab^2c \quad 3 \quad x + y$$

$$ab, ac \quad 2 \quad x + y$$
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{\text{op}}$

In the composite theory $\mathcal{BA}$ every morphism can be expressed as a composite.

For example: $\times$ and $+$

The composite 3-ary operation $a(b + c)$ can be expressed as:

![Diagram](image.png)
4. Three ways to do it

1. Factorisation systems over $\mathbb{F}^{op}$

In the composite theory $\mathbb{B} \mathbb{A}$ every morphism can be expressed as a composite

For example: $\times$ and $+$

The composite 3-ary operation $a(b + c)$ can be expressed as

---factorisations are only unique up to morphisms in $\mathbb{F}^{op}$. 

4. Three ways to do it

**Appealing fact** (Rosebrugh and Wood)
Strict factorisation systems are distributive laws in **Span**.
4. Three ways to do it

Appealing fact (Rosebrugh and Wood)

Strict factorisation systems are distributive laws in $\text{Span}$.

- $\mathcal{A}$ and $\mathcal{B}$ are categories i.e. monads in $\text{Span}$.
- $\mathcal{A}\mathcal{B} \xrightarrow{\text{}} \mathcal{B}\mathcal{A}$ makes $\mathcal{B}\mathcal{A}$ into a monad in $\text{Span}$ i.e. a category.
4. Three ways to do it

Appealing fact (Rosebrugh and Wood)
Strict factorisation systems are distributive laws in $\text{Span}$.

- $A$ and $B$ are categories i.e. monads in $\text{Span}$.
- $A \mathrel{\rightarrow} B$ makes $BA$ into a monad in $\text{Span}$ i.e. a category.

It is the pullback

\[
\begin{array}{c}
A \\
\text{ob} F \\
\text{ob} F
\end{array}
\]
4. Three ways to do it

**Appealing fact** (Rosebrugh and Wood)
Strict factorisation systems are distributive laws in $\text{Span}$.  

- $A$ and $B$ are categories i.e. monads in $\text{Span}$.  
- $\xymatrix{AB & BA\ar[r] & BA}$ makes $BA$ into a monad in $\text{Span}$ i.e. a category.

It is the pullback

\[
\begin{array}{ccc}
A & \xymatrix{\ar[r] & B} & \ar[l] \text{ob}\mathcal{F} \\
\text{ob}\mathcal{F} & \text{ob}\mathcal{F} & \text{ob}\mathcal{F}
\end{array}
\]
4. Three ways to do it

**Appealing fact** (Rosebrugh and Wood)

Strict factorisation systems are distributive laws in \( \text{Span} \).

- \( A \) and \( B \) are categories i.e. monads in \( \text{Span} \).
- \( A \rightarrow B \rightarrow A \) makes \( BA \) into a monad in \( \text{Span} \) i.e. a category.

It is the pullback

\[
\begin{array}{ccc}
\text{ob}_F & \rightarrow & \text{ob}_F \\
\downarrow \quad \quad \quad \downarrow \\
A & \rightarrow & B \\
\quad \quad \quad \downarrow \\
\text{ob}_F & \rightarrow & \text{ob}_F
\end{array}
\]
4. Three ways to do it

**Appealing fact** (Rosebrugh and Wood)
Strict factorisation systems are distributive laws in $\text{Span}$.

- $A$ and $B$ are categories i.e. monads in $\text{Span}$.
- $AB \rightarrow BA$ makes $BA$ into a monad in $\text{Span}$ i.e. a category.

It is the pullback

$$k \in A \quad l \in B \quad m$$

$$
\begin{array}{c}
A \\
\downarrow \\
\text{ob } F
\end{array} 
\quad 
\begin{array}{c}
B \\
\downarrow \\
\text{ob } F
\end{array} 
\quad 
\begin{array}{c}
\text{ob } F
\end{array}
$$
4. Three ways to do it

**Appealing fact** (Rosebrugh and Wood)
Strict factorisation systems are distributive laws in $\text{Span}$.

- $A$ and $B$ are categories i.e. monads in $\text{Span}$.
- $AB \xrightarrow{BA} BA$ makes $BA$ into a monad in $\text{Span}$ i.e. a category.

It is the pullback

\[\begin{array}{ccc}
A & \xrightarrow{k} & B \\
\downarrow & & \downarrow \\
\text{ob}F & & \text{ob}F \\
\end{array}\]

The distributive law tells us how to re-express a pair

\[\begin{array}{ccc}
k & \in A & l \in B & m \\
\in A & \in A & \in B \\
\end{array}\]

as

\[\begin{array}{ccc}
k & \in A & l' \in B & m \\
\in A & \in A & \in B \\
\end{array}\]
4. Three ways to do it

RW define *distributive laws over I* for I a groupoid—ensures equivalence relation on composable pairs.
4. Three ways to do it

RW define *distributive laws over* $I$ for $I$ a groupoid—ensures equivalence relation on composable pairs. However instead we can *generate* an equivalence relation.
4. Three ways to do it

RW define *distributive laws over J* for J a groupoid —ensures equivalence relation on composable pairs. However instead we can *generate* an equivalence relation.

Idea
4. Three ways to do it

RW define *distributive laws over* \( J \) for \( J \) a groupoid —ensures equivalence relation on composable pairs.

However instead we can *generate* an equivalence relation.

**Idea**

Our original pullback

\[
\begin{array}{ccc}
\text{ob}_F & \xrightarrow{\text{ob}_F} & \text{ob}_F \\
B \otimes A & \xrightarrow{A} & B \\
A & \xrightarrow{B} & B
\end{array}
\]

ignored the fact that \( \mathcal{F}^{\text{op}} \) is in both \( A \) and \( B \).
4. Three ways to do it

RW define *distributive laws over* $I$ for $I$ a groupoid —ensures equivalence relation on composable pairs. However instead we can *generate* an equivalence relation.

**Idea**

Our original pullback

\[
\begin{array}{ccc}
B \otimes A & \xrightarrow{B} & B \\
\downarrow & & \downarrow \\
A & \xleftarrow{A} & B
\end{array}
\]

\[
\begin{array}{ccc}
& \text{ob}F & \\
\downarrow & & \downarrow \\
A & \xleftarrow{ob}F & B
\end{array}
\]

ignored the fact that $F^{op}$ is in both $A$ and $B$.

So we want a coequaliser

\[
\begin{array}{ccc}
B \otimes F^{op} \otimes A & \xrightarrow{\text{absorb } F^{op} \text{ into } A} & B \otimes A \\
& \xrightarrow{\text{absorb } F^{op} \text{ into } B} & \end{array}
\]

—looks like bimodules.
4. Three ways to do it

Definition 1

A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $F^{\text{op}}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$. 
4. Three ways to do it

Definition 1

A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $F^{\text{op}}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$.

$\equiv$ a distributive law of $\mathbb{A}$ over $\mathbb{B}$ expressed as monads in $\text{Bim}(\text{Span})$
4. Three ways to do it

Definition 1
A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $F^{op}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$.

$\equiv$ a distributive law of $\mathbb{A}$ over $\mathbb{B}$ expressed as monads in

$$\text{Bim}(\text{Span}) \simeq \text{Prof}.$$
4. Three ways to do it

Definition 1

A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $F^{op}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$.

$\equiv$ a distributive law of $\mathbb{A}$ over $\mathbb{B}$ expressed as monads in

$$\text{Bim}(\text{Span}) \simeq \text{Prof}.$$ 

Aside on profunctors
4. Three ways to do it

Definition 1
A distributive law of Lawvere theories $\mathcal{A}$ over $\mathcal{B}$ is a factorisation system over $F^{\text{op}}$ on the composite $\mathcal{B} \otimes \mathcal{A}$ in $\text{Span}$.

≡ a distributive law of $\mathcal{A}$ over $\mathcal{B}$ expressed as monads in

$$\text{Bim(Span)} \simeq \text{Prof}.$$

Aside on profunctors
Given categories $\mathcal{C}$ and $\mathcal{D}$, a profunctor $\mathcal{C} \xrightarrow{F} \mathcal{D}$ is a functor

$$\mathcal{D}^{\text{op}} \times \mathcal{C} \xrightarrow{F} \text{Set}.$$
4. Three ways to do it

Definition 1

A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $F^{\mathrm{op}}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$. 

$\equiv$ a distributive law of $\mathbb{A}$ over $\mathbb{B}$ expressed as monads in $\text{Bim}(\text{Span}) \simeq \text{Prof}$. 

Aside on profunctors

Given categories $\mathcal{C}$ and $\mathcal{D}$, a profunctor $\mathcal{C} \xrightarrow{F} \mathcal{D}$ is a functor $\mathcal{D}^{\mathrm{op}} \times \mathcal{C} \xrightarrow{F} \text{Set}$. 

A monad $\mathcal{C} \xrightarrow{A} \mathcal{C} \in \text{Prof}$ corresponds to a category $\mathbb{A}$ equipped with an identity-on-objects functor $\mathcal{C} \longrightarrow \mathbb{A}$. 

4. Three ways to do it

Definition 1
A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a factorisation system over $\mathbb{F}^{\text{op}}$ on the composite $\mathbb{B} \otimes \mathbb{A}$ in $\text{Span}$.

$\equiv$ a distributive law of $\mathbb{A}$ over $\mathbb{B}$ expressed as monads in $\text{Bim} (\text{Span}) \simeq \text{Prof}$.

Aside on profunctors
Given categories $\mathbb{C}$ and $\mathbb{D}$, a profunctor $\mathbb{C} \xrightarrow{F} \mathbb{D}$ is a functor $\mathbb{D}^{\text{op}} \times \mathbb{C} \xrightarrow{F} \text{Set}$.

A monad $\mathbb{C} \xrightarrow{A} \mathbb{C} \in \text{Prof}$ corresponds to a category $\mathbb{A}$ equipped with an identity-on-objects functor $\mathbb{C} \rightarrow \mathbb{A}$.

So Lawvere theories arise as particular monads on $\mathbb{F}^{\text{op}}$. 
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids
2. Prof(Mon) — **internal profunctors in monoids**

A monad $\mathcal{C} \hookrightarrow \mathcal{C}$ is now a *monoidal* category $\mathbb{A}$ equipped with an identity-on-objects *monoidal* functor $\mathcal{C} \longrightarrow \mathbb{A}$.
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $\mathcal{C} \to \mathcal{C}$ is now a *monoidal* category $\mathcal{A}$ equipped with an identity-on-objects *monoidal* functor $\mathcal{C} \to \mathcal{A}$.

So again Lawvere theories arise as particular monads on $\mathbb{F}^{\text{op}}$. 
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $C \to C$ is now a monoidal category $A$ equipped with an identity-on-objects monoidal functor $C \to A$.

So again Lawvere theories arise as particular monads on $F^{op}$.

Definition 2

A distributive law of Lawvere theories $A$ over $B$ is a distributive law in the bicategory $Prof(Mon)$. 
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $\mathcal{C} \longrightarrow \mathcal{C}$ is now a *monoidal* category $\mathcal{A}$ equipped with an identity-on-objects *monoidal* functor $\mathcal{C} \longrightarrow \mathcal{A}$.

So again Lawvere theories arise as particular monads on $\mathbb{F}^{\text{op}}$.

**Definition 2**

A distributive law of Lawvere theories $\mathcal{A}$ over $\mathcal{B}$ is a distributive law in the bicategory $\text{Prof(Mon)}$.

**Theorem**

Such a distributive law makes $\mathcal{B} \otimes_{\mathbb{F}^{\text{op}}} \mathcal{A}$ into a Lawvere theory.

i.e. if $\mathcal{A}$ and $\mathcal{B}$ are finite-product categories, so is $\mathcal{B} \otimes_{\mathbb{F}^{\text{op}}} \mathcal{A}$. 

4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $C \xrightarrow{\rightarrow} C$ is now a monoidal category $\mathcal{A}$ equipped with an identity-on-objects monoidal functor $C \xrightarrow{\rightarrow} \mathcal{A}$.

So again Lawvere theories arise as particular monads on $\mathcal{F}^{\text{op}}$.

Definition 2

A distributive law of Lawvere theories $\mathcal{A}$ over $\mathcal{B}$ is a distributive law in the bicategory $\text{Prof}(\text{Mon})$.

Theorem

Such a distributive law makes $\mathcal{B} \otimes_{\mathcal{F}^{\text{op}}} \mathcal{A}$ into a Lawvere theory.

i.e. if $\mathcal{A}$ and $\mathcal{B}$ are finite-product categories, so is $\mathcal{B} \otimes_{\mathcal{F}^{\text{op}}} \mathcal{A}$.
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $C \xrightarrow{\lambda} C$ is now a *monoidal* category $A$ equipped with an identity-on-objects *monoidal* functor $C \xrightarrow{\lambda} A$.

So again Lawvere theories arise as particular monads on $F^{op}$.

**Definition 2**

A distributive law of Lawvere theories $A$ over $B$ is a distributive law in the bicategory $\text{Prof(Mon)}$.

**Theorem**

Such a distributive law makes $B \otimes_{F^{op}} A$ into a Lawvere theory. i.e. if $A$ and $B$ are finite-product categories, so is $B \otimes_{F^{op}} A$.

**Proof**  •  Bare hands, or
4. Three ways to do it

2. Prof(Mon) — internal profunctors in monoids

A monad $\mathcal{C} \xlongrightarrow{} \mathcal{C}$ is now a *monoidal* category $\mathbb{A}$ equipped with an identity-on-objects *monoidal* functor $\mathcal{C} \xlongrightarrow{} \mathbb{A}$.

So again Lawvere theories arise as particular monads on $\mathbb{F}^{\text{op}}$.

**Definition 2**

A distributive law of Lawvere theories $\mathbb{A}$ over $\mathbb{B}$ is a distributive law in the bicategory $\text{Prof}(\text{Mon})$.

**Theorem**

Such a distributive law makes $\mathbb{B} \mathcal{\otimes}_{\mathbb{F}^{\text{op}}} \mathbb{A}$ into a Lawvere theory. i.e. if $\mathbb{A}$ and $\mathbb{B}$ are finite-product categories, so is $\mathbb{B} \mathcal{\otimes}_{\mathbb{F}^{\text{op}}} \mathbb{A}$.

**Proof**  
• Bare hands, or  
• The free finite-product category 2-monad on $\text{Prof}$.
4. Three ways to do it

4. Three ways to do it


- Let $P$ be the free finite-product category 2-monad on $\text{Cat}$. 
- $P$ extends to $\text{Prof}$ via a distributive law. 
- Let $\text{Prof}_P$ be the Kleisli bicategory for the extended $P$. 
4. Three ways to do it


- Let $P$ be the free finite-product category 2-monad on $\text{Cat}$.
- $P$ extends to $\text{Prof}$ via a distributive law.
- Let $\text{Prof}_P$ be the Kleisli bicategory for the extended $P$.

Then monads on 1 in $\text{Prof}_P$ are precisely Lawvere theories.
4. Three ways to do it


- Let $\mathcal{P}$ be the free finite-product category 2-monad on $\text{Cat}$.
- $\mathcal{P}$ extends to $\text{Prof}$ via a distributive law.
- Let $\text{Prof}_\mathcal{P}$ be the Kleisli bicategory for the extended $\mathcal{P}$.

Then monads on 1 in $\text{Prof}_\mathcal{P}$ are precisely Lawvere theories.

- A monad in $\text{Prof}_\mathcal{P}$ is a profunctor $1 \rightarrow \mathcal{P}1$

- Let \( \mathcal{P} \) be the free finite-product category 2-monad on \( \text{Cat} \).
- \( \mathcal{P} \) extends to \( \text{Prof} \) via a distributive law.
- Let \( \text{Prof}_\mathcal{P} \) be the Kleisli bicategory for the extended \( \mathcal{P} \).

Then monads on 1 in \( \text{Prof}_\mathcal{P} \) are precisely Lawvere theories.

- A monad in \( \text{Prof}_\mathcal{P} \) is a profunctor \( 1 \rightarrow \mathcal{P}1 \)
  i.e. \( \mathcal{P}1^{\text{op}} \times 1 \rightarrow \text{Set} \)
4. Three ways to do it


- Let $\mathcal{P}$ be the free finite-product category 2-monad on $\textbf{Cat}$.
- $\mathcal{P}$ extends to $\textbf{Prof}$ via a distributive law.
- Let $\textbf{Prof}_\mathcal{P}$ be the Kleisli bicategory for the extended $\mathcal{P}$.

Then monads on $1$ in $\textbf{Prof}_\mathcal{P}$ are precisely Lawvere theories.

- A monad in $\textbf{Prof}_\mathcal{P}$ is a profunctor $1 \rightarrow \mathcal{P}1$
  i.e. $\mathcal{P}1^{\text{op}} \times 1 \rightarrow \textbf{Set}$
  i.e. $\text{FinSet} \rightarrow \textbf{Set}$ a finitary monad.
4. Three ways to do it


- Let \( \mathcal{P} \) be the free finite-product category 2-monad on \( \text{Cat} \).
- \( \mathcal{P} \) extends to \( \text{Prof} \) via a distributive law.
- Let \( \text{Prof}_\mathcal{P} \) be the Kleisli bicategory for the extended \( \mathcal{P} \).

Then monads on 1 in \( \text{Prof}_\mathcal{P} \) are precisely Lawvere theories.

- A monad in \( \text{Prof}_\mathcal{P} \) is a profunctor \( 1 \to \mathcal{P}1 \)
  i.e. \( \mathcal{P}1^{\text{op}} \times 1 \to \text{Set} \)
  i.e. \( \text{FinSet} \to \text{Set} \) a finitary monad.

Definition 3
4. Three ways to do it


- Let $\mathcal{P}$ be the free finite-product category 2-monad on $\text{Cat}$.
- $\mathcal{P}$ extends to $\text{Prof}$ via a distributive law.
- Let $\text{Prof}_\mathcal{P}$ be the Kleisli bicategory for the extended $\mathcal{P}$.

Then monads on 1 in $\text{Prof}_\mathcal{P}$ are precisely Lawvere theories.

- A monad in $\text{Prof}_\mathcal{P}$ is a profunctor $1 \longrightarrow \mathcal{P}1$
  i.e. $\mathcal{P}1^{\text{op}} \times 1 \longrightarrow \text{Set}$
  i.e. $\text{FinSet} \longrightarrow \text{Set}$ a finitary monad.

Definition 3

A distributive law of Lawvere theories $\mathcal{A}$ over $\mathcal{B}$ is a distributive law in the bicategory $\text{Prof}_\mathcal{P}$. 
5. Comparison
Claim
These three methods all give the same answer as a distributive law between the associated monads.
5. Comparison

Claim
These three methods all give the same answer as a distributive law between the associated monads.

Idea
5. Comparison

Claim
These three methods all give the same answer as a distributive law between the associated monads.

Idea
Compare

- Finitary monads $\text{Set} \rightarrow \text{Set}$ in $\text{CAT}$
5. Comparison

Claim
These three methods all give the same answer as a distributive law between the associated monads.

Idea
Compare

- Finitary monads \textbf{Set} \rightarrow \textbf{Set} in \textbf{CAT}

- Lawvere theories as
  
  1. monads \( \mathbb{F} \rightarrow \mathbb{F} \) in \textbf{Prof}
  2. monads \( \mathbb{F} \rightarrow \mathbb{F} \) in \textbf{Prof}(\textbf{Mon})
  3. monads \( 1 \rightarrow 1 \) in \textbf{Prof}_P.
5. Comparison

Claim
These three methods all give the same answer as a distributive law between the associated monads.

Idea

Compare

- Finitary monads \( \text{Set} \to \text{Set} \) in \( \text{CAT} \)
  \[ \text{Set} \xrightarrow{T} \text{Set} \]

- Lawvere theories as
  1. monads \( \mathcal{F} \to \mathcal{F} \) in \( \text{Prof} \)
  2. monads \( \mathcal{F} \to \mathcal{F} \) in \( \text{Prof}(\text{Mon}) \)
  3. monads \( 1 \to 1 \) in \( \text{Prof}_p \).
5. Comparison

Claim

These three methods all give the same answer as a distributive law between the associated monads.

Idea

Compare

- Finitary monads Set → Set in CAT
- Lawvere theories as
  1. monads F → F in Prof
  2. monads F → F in Prof(Mon)
  3. monads 1 → 1 in Prof_p.
5. Comparison
5. Comparison

\[ \text{Prof}_P(1, 1) \]

\[ \text{Prof}(\mathbb{F}, \mathbb{F}) \]

\[ \text{Prof}(\text{Mon})(\mathbb{F}, \mathbb{F}) \]
5. Comparison

\[ \text{Prof}_P(1, 1) \]

\[ \text{Prof}(F, F) \]

\[ \text{Prof}(\text{Mon})(F, F) \]
5. Comparison

Monads

\[ \text{Prof}_P(1, 1) \]

Lawvere theories

\[ \text{Prof}(F, F) \]

\[ \text{Prof}(\text{Mon})(F, F) \]
5. Comparison

Monads

\[ \text{Prof}_P(1, 1) \quad \text{Lawvere theories} \]

\[ \text{Prof}(\mathcal{F}, \mathcal{F}) \quad \text{id-on-objects functors} \]

\[ \mathcal{F} \rightarrow \mathcal{A} \]

\[ \text{Prof}(\text{Mon})(\mathcal{F}, \mathcal{F}) \]
5. Comparison

### Monads

\[
\text{Monads}
\]

\[
\text{Prof}_P(1, 1)
\]

Lawvere theories

\[
\text{Prof}(F, F)
\]

id-on-objects functors

\[
F \rightarrow A
\]

\[
\text{Prof}(\text{Mon})(F, F)
\]

id-on-objects

monoidal functors

\[
F \rightarrow A
\]
5. Comparison

**Monads**

\[ \text{Prof}_P(1, 1) \]

Lawvere theories

\[ \text{CAT}_f(\text{Set}, \text{Set}) \]

\[ \text{Prof}(F, F) \]

\[ F \rightarrow A \]

\[ \text{id-on-objects functors} \]

\[ \text{Prof}(\text{Mon})(F, F) \]

\[ F \rightarrow A \]

\[ \text{id-on-objects} \]

\[ \text{monoidal functors} \]
5. Comparison

Monads

\[ \text{Prof}_P(1, 1) \quad \text{Lawvere theories} \]

\[ \text{CAT}_f(\text{Set}, \text{Set}) \xrightarrow{f+f} \text{Prof}(F, F) \quad \text{id-on-objects functors} \]

\[ \text{Prof}(\text{Mon})(F, F) \quad \text{id-on-objects functors} \]

\[ F \rightarrow A \quad \text{monoidal functors} \]

\[ F \rightarrow A \]
5. Comparison

Monads

Lawvere theories

\text{CAT}_f(\text{Set}, \text{Set}) \xrightarrow{f+f} \text{Prof}(\mathcal{F}, \mathcal{F})

\text{Prof}_P(1, 1)

\text{Prof} \mathcal{F}(\text{Mon}) \mathcal{F}(\mathcal{F}, \mathcal{F})

\text{id-on-objects functors}

\mathcal{F} \rightarrow \mathcal{A}

\text{id-on-objects monoidal functors}

\mathcal{F} \rightarrow \mathcal{A}
5. Comparison

**Monads**

**Lawvere theories**

\[ \text{Prof}(\mathcal{P}, \mathcal{P}) \]

\[ \text{id-on-objects functors} \]

\[ \begin{array}{c}
\mathcal{F} \\ \rightarrow \\
\mathcal{A}
\end{array} \]

\[ \text{monoidal functors} \]

\[ \begin{array}{c}
\mathcal{F} \\ \rightarrow \\
\mathcal{A}
\end{array} \]

\[ \text{Prof}(\text{Mon})(\mathcal{F}, \mathcal{F}) \]

**Comparison**

\[ \text{Prof}_{\mathcal{P}}(1, 1) \]

\[ \text{equivalece} \]

\[ \text{forgetful} \]

\[ \begin{array}{c}
\text{CAT}_{\mathcal{F}}(\text{Set}, \text{Set}) \\
\text{Prof}(\mathcal{P}1, \mathcal{P}1) \\
\text{Prof}(\mathcal{F}, \mathcal{F})
\end{array} \]

\[ \begin{array}{c}
\mathcal{F} \\
\rightarrow \\
\mathcal{A}
\end{array} \]
5. Comparison

Monads

Lawvere theories

id-on-objects functors
\[ F \to A \]

id-on-objects
monoidal functors
\[ F \to A \]
5. Comparison

Key points
5. Comparison

Key points

• The functors send $T$ to $\mathbb{L}_T$. 
5. Comparison

Key points

- The functors send $T$ to $\mathbb{L}_T$.

- By pseudo-functoriality distributive laws map to distributive laws, and

$$\mathbb{L}_T \circ \mathbb{L}_S \cong \mathbb{L}_{TS}$$
5. Comparison

Key points

- The functors send \( T \) to \( \mathbb{L}_T \).

- By pseudo-functoriality distributive laws map to distributive laws, and

\[
\mathbb{L}_T \circ \mathbb{L}_S \cong \mathbb{L}_{TS}
\]

- Moreover the functors are full and faithful, so given Lawvere theories on the right, \( \text{any} \) distributive law between them corresponds to one on the left.
5. Comparison

Monads

Lawvere theories

id-on-objects functors

\( F \to A \)

id-on-objects monoidal functors

\( F \to A \)
5. Comparison

Key points

• The functors are monoidal, and send \( T \) to \( \mathbb{L}_T \).

• By pseudo-functoriality distributive laws map to distributive laws, and

\[
\mathbb{L}_T \circ \mathbb{L}_S \cong \mathbb{L}_{TS}
\]

• Moreover the functors are full and faithful, so given Lawvere theories on the right, \textit{any} distributive law between them corresponds to one on the left.
5. Comparison

Key points

- The functors are monoidal, and send $T$ to $\mathbb{L}_T$.

- By pseudo-functoriality distributive laws map to distributive laws, and

  \[ \mathbb{L}_T \circ \mathbb{L}_S \cong \mathbb{L}_{TS} \]

- Moreover the functors are full and faithful, so given Lawvere theories on the right, any distributive law between them corresponds to one on the left.

So we have three equivalent notions of distributive laws for Lawvere theories, which correspond to distributive laws between the associated monads.