universität freiburg

$$
\begin{tikzcd}
\mathrm{Hom}(A,A) \arrow[r, "\mathrm{Hom}(A{,}f)"] \arrow[d, "\Phi_A"'] & \mathrm{Hom}(A,X) \arrow[d, "\Phi_X"] \\
F(A) \arrow[r, "Ff"'] & F(X)
\end{tikzcd}
$$

$$
\begin{tikzcd}
\mathrm{id}_A \arrow[r, mapsto] \arrow[d, mapsto] & f \arrow[d, mapsto] \\
u \arrow[r, mapsto] & (Ff)u = \Phi_X(f)
\end{tikzcd}
$$

What you needa know about Yoneda

Emma Bach (she/her)

Seminar on Functional Programming and Logic, Summer Semester 2025

Motivation

- A common sentiment in many cultures is the idea that people are defined by how they interact with their surroundings.

[1] Quoted as a proverb in Don Quixote

Motivation

- A common sentiment in many cultures is the idea that people are defined by how they interact with their surroundings.
- "Tell me your company, and I will tell you what you are."[1]

[1] Quoted as a proverb in Don Quixote

Motivation

- A common sentiment in many cultures is the idea that people are defined by how they interact with their surroundings.
- "Tell me your company, and I will tell you what you are."[1]
- The Yoneda Lemma is the result of applying this way of thinking to mathematical objects within the extremely general setting of category theory.

[1] Quoted as a proverb in Don Quixote

Motivation

- A common sentiment in many cultures is the idea that people are defined by how they interact with their surroundings.
- "Tell me your company, and I will tell you what you are."[1]
- The Yoneda Lemma is the result of applying this way of thinking to mathematical objects within the extremely general setting of category theory.
- As a result, a category $\mathbb{C}$ is often best understood by instead studying functors from that category into $\mathbb{S}et$.

[1] Quoted as a proverb in Don Quixote

Categories

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"'] \arrow[loop above, "id_1"] & B \arrow[d, "g"] \arrow[loop above, "id_2"] \\
& C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:

Categories

$$
\begin{tikzcd}
A \arrow[loop above, "id_1"] \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[loop above, "id_2"] \arrow[d, "g"] \\
& C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;

Categories

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[loop above, "id_1"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \arrow[loop above, "id_2"] \\
 & C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;
  - for all $A, B \in |\mathbb{C}|$, a collection $\mathbb{C}(A, B)$ of morphisms from A to B;

Categories

$$
\begin{tikzcd}
A \arrow[loop above, "id_1"] \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[loop above, "id_2"] \arrow[d, "g"] \\
& C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;
  - for all $A, B \in |\mathbb{C}|$, a collection $\mathbb{C}(A, B)$ of morphisms from A to B;
  - for all $A \in |\mathbb{C}|$, an identity morphism $id_A \in \mathbb{C}(A, A)$;

Categories

$$\begin{tikzcd} A \arrow[loop above, "id_1"] \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[loop above, "id_2"] \arrow[d, "g"] \\ & C \arrow[loop below, "id_3"] \end{tikzcd}$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;
  - for all $A, B \in |\mathbb{C}|$, a collection $\mathbb{C}(A, B)$ of morphisms from A to B;
  - for all $A \in |\mathbb{C}|$, an identity morphism $id_A \in \mathbb{C}(A, A)$;
  - an associative composition morphism $f \circ g \in \mathbb{C}(A, C)$ for each pair of morphisms $f \in \mathbb{C}(A, B)$, $g \in \mathbb{C}(A, B)$.

Categories

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[loop above, "id_1"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \arrow[loop above, "id_2"] \\
 & C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;
  - for all $A, B \in |\mathbb{C}|$, a collection $\mathbb{C}(A, B)$ of morphisms from A to B;
  - for all $A \in |\mathbb{C}|$, an identity morphism $id_A \in \mathbb{C}(A, A)$;
  - an associative composition morphism $f \circ g \in \mathbb{C}(A, C)$ for each pair of morphisms $f \in \mathbb{C}(A, B)$, $g \in \mathbb{C}(A, B)$.
- If $\mathbb{C}(A, B)$ is a set, we call it the homset from A to B.

Categories

$$
\begin{tikzcd}
A \arrow[loop above, "id_1"] \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[loop above, "id_2"] \arrow[d, "g"] \\
& C \arrow[loop below, "id_3"]
\end{tikzcd}
$$

- A category $\mathbb{C}$ consists of:
  - a collection $|\mathbb{C}|$ of objects;
  - for all $A, B \in |\mathbb{C}|$, a collection $\mathbb{C}(A, B)$ of morphisms from A to B;
  - for all $A \in |\mathbb{C}|$, an identity morphism $id_A \in \mathbb{C}(A, A)$;
  - an associative composition morphism $f \circ g \in \mathbb{C}(A, C)$ for each pair of morphisms $f \in \mathbb{C}(A, B)$, $g \in \mathbb{C}(A, B)$.
- If $\mathbb{C}(A, B)$ is a set, we call it the homset from A to B.
- For every category $\mathbb{C}$, there exists an opposite category $\mathbb{C}^{op}$, in which all morphisms are reversed.

Homfunctors

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"] & B \arrow[d, "g"] \\
 & C
\end{tikzcd}
$$

$$
\begin{tikzcd}
\{id_A\} \arrow[r, "f \circ"] \arrow[dr, "f \circ g \circ"] & \{f\} \arrow[d, "g \circ"] \\
 & \{f \circ g\}
\end{tikzcd}
$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.

Homfunctors

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\ & C \end{tikzcd}$$

$$\begin{tikzcd} \{id_A\} \arrow[r, "f \circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g \circ"] \\ & \{f \circ g\} \end{tikzcd}$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:

Homfunctors

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\
& C
\end{tikzcd}
$$

$$
\begin{tikzcd}
\{id_A\} \arrow[r, "f \circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g \circ"] \\
& \{f \circ g\}
\end{tikzcd}
$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:
  - $\mathbb{C}(A, -)$ maps an Object B to the Homset $\mathbb{C}(A, B)$

Homfunctors

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\
 & C
\end{tikzcd}
$$

$$
\begin{tikzcd}
\{id_A\} \arrow[r, "f\circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g\circ"] \\
 & \{f \circ g\}
\end{tikzcd}
$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:
  - $\mathbb{C}(A, -)$ maps an Object B to the Homset $\mathbb{C}(A, B)$
  - A morphism $f : \mathbb{C}(B, C)$ is mapped to the morphism $f\circ : \mathbb{C}(A, B) \to \mathbb{C}(A, C)$

Homfunctors

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\
 & C
\end{tikzcd}
$$

$$
\begin{tikzcd}
\{id_A\} \arrow[r, "f\circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g\circ"] \\
 & \{f \circ g\}
\end{tikzcd}
$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:
  - $\mathbb{C}(A, -)$ maps an Object B to the Homset $\mathbb{C}(A, B)$
  - A morphism $f : \mathbb{C}(B, C)$ is mapped to the morphism $f\circ : \mathbb{C}(A, B) \to \mathbb{C}(A, C)$
- Similarly, $\mathbb{C}(-, B)$ is a functor $\mathbb{C}^{op} \to \mathbb{S}et$:

Homfunctors

$$
\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\
& C
\end{tikzcd}
$$

$$
\begin{tikzcd}
\{id_A\} \arrow[r, "f\circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g\circ"] \\
& \{f \circ g\}
\end{tikzcd}
$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:
  - $\mathbb{C}(A, -)$ maps an Object B to the Homset $\mathbb{C}(A, B)$
  - A morphism $f : \mathbb{C}(B, C)$ is mapped to the morphism $f\circ : \mathbb{C}(A, B) \to \mathbb{C}(A, C)$
- Similarly, $\mathbb{C}(-, B)$ is a functor $\mathbb{C}^{op} \to \mathbb{S}et$:
  - $\mathbb{C}(-, B)$ maps an Object A to the Homset $\mathbb{C}(A, B)$

Homfunctors

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[dr, "f \circ g"'] & B \arrow[d, "g"] \\ & C \end{tikzcd}$$

$$\begin{tikzcd} \{id_A\} \arrow[r, "f \circ"] \arrow[dr, "f \circ g \circ"'] & \{f\} \arrow[d, "g \circ"] \\ & \{f \circ g\} \end{tikzcd}$$

- For any category $\mathbb{C}$, a homset $\mathbb{C}(A, B)$ is a set of morphisms.
- We define a functor $\mathbb{C}(A, -) : \mathbb{C} \to \mathbb{S}et$:
  - $\mathbb{C}(A, -)$ maps an Object B to the Homset $\mathbb{C}(A, B)$
  - A morphism $f : \mathbb{C}(B, C)$ is mapped to the morphism $f \circ : \mathbb{C}(A, B) \to \mathbb{C}(A, C)$
- Similarly, $\mathbb{C}(-, B)$ is a functor $\mathbb{C}^{op} \to \mathbb{S}et$:
  - $\mathbb{C}(-, B)$ maps an Object A to the Homset $\mathbb{C}(A, B)$
  - A morphism $f : \mathbb{C}(A, B)$ is mapped to the morphism $\circ f : \mathbb{C}(B, C) \to \mathbb{C}(A, C)$

Natural Transformations

- A structure-preserving map between functors.
  - Let $F, G : \mathbb{C} \to \mathbb{D}$ be functors.
  - A natural transformation ϕ is an indexed family of morphisms $\phi_A \in \mathbb{D}(F(A), G(A))$ from $F(A)$ to $G(A)$
  - These morphisms satisfy the following naturality condition:

$$\forall f \in \mathbb{C}(A, B) : \phi_B \circ F(f) = G(f) \circ \phi_A$$

- Given two functors F and G, we write the collection of all natural transformation between them as $\mathrm{Nat}(F, G)$.

Naturality from Polymorphism

- The naturality condition resembles an equality we saw a few weeks ago:

$$r_B \circ \mathtt{map}(a) = \mathtt{map}(a) \circ r_A$$

Naturality from Polymorphism

- The naturality condition resembles an equality we saw a few weeks ago:

$$r_B \circ \mathtt{map}(a) = \mathtt{map}(a) \circ r_A$$

- This is the free theorem we got for a parametrically polymorphic function `r :: [X] -> [X]` and an arbitrary function `a : A -> B`.

Naturality from Polymorphism

- The naturality condition resembles an equality we saw a few weeks ago:

$$r_B \circ \mathtt{map}(a) = \mathtt{map}(a) \circ r_A$$

- This is the free theorem we got for a parametrically polymorphic function `r :: [X] -> [X]` and an arbitrary function `a :  A -> B`.
- This free theorem is equivalent to the statement that r is a natural transformation.

Naturality from Polymorphism

- In general, assume we have:

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :  F x -> G x`,

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :  F x -> G x`,
  - an arbitrary function `f :  A -> B`.

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :: F x -> G x`,
  - an arbitrary function `f :: A -> B`.
- Then we get the following free theorem:

```
r . fmap f = fmap f . r
```

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :  F x -> G x`,
  - an arbitrary function `f :  A -> B`.
- Then we get the following free theorem:

```
r .  fmap f = fmap f .  r
```

- In categorical notation:

$$r_B \circ F(f) = G(f) \circ r_A$$

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :: F x -> G x`,
  - an arbitrary function `f :: A -> B`.
- Then we get the following free theorem:

```
r . fmap f = fmap f . r
```

- In categorical notation:

$$r_B \circ F(f) = G(f) \circ r_A$$

- So our free theorem is a proof that any parametrically polymorphic function `r` is a natural transformation!

Naturality from Polymorphism

- In general, assume we have:
  - two functors `F` and `G`,
  - a parametrically polymorphic function `r :  F x -> G x`,
  - an arbitrary function `f :  A -> B`.
- Then we get the following free theorem:

```
r .  fmap f = fmap f .  r
```

- In categorical notation:

$$r_B \circ F(f) = G(f) \circ r_A$$

- So our free theorem is a proof that any parametrically polymorphic function `r` is a natural transformation!
- It turns out that parametrically polymorphic functions correspond exactly to natural transformations between endofunctors $\mathbb{S}et \to \mathbb{S}et$.

Functor Categories

- For any $\mathbb{C}$, $\mathbb{D}$, the collection of functors $\mathbb{C} \to \mathbb{D}$ form a category.

Functor Categories

- For any $\mathbb{C}$, $\mathbb{D}$, the collection of functors $\mathbb{C} \to \mathbb{D}$ form a category.
- This category is known as a functor category and denoted $\mathbb{D}^{\mathbb{C}}$.

Functor Categories

- For any $\mathbb{C}$, $\mathbb{D}$, the collection of functors $\mathbb{C} \to \mathbb{D}$ form a category.
- This category is known as a functor category and denoted $\mathbb{D}^{\mathbb{C}}$.
- A morphism $\mathbb{D}^{\mathbb{C}}(F, G)$ is a natural transformation $F \to G$.

The Yoneda Embedding

- Remember that the goal is finding out everything about an object A through its relations to other objects.

$$\begin{tikzcd}
A \arrow[r, "f \,\in\, \mathbb{C}(A{,}B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\
\mathbb{C}(A, -) \arrow[r, "\mathcal{Y}(f)"'] & C(B, -)
\end{tikzcd}$$

The Yoneda Embedding

- Remember that the goal is finding out everything about an object A through its relations to other objects.
- So we want a correspondence between objects and their homsets.

$$\begin{tikzcd} A \arrow[r, "f \,\in\, \mathbb{C}(A{,}B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\ \mathbb{C}(A,-) \arrow[r, "\mathcal{Y}(f)"'] & C(B,-) \end{tikzcd}$$

The Yoneda Embedding

$$
\begin{tikzcd}
A \arrow[rr, "f \,\in\, \mathbb{C}(A{,}B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\
\mathbb{C}(A,-) \arrow[rr, "\mathcal{Y}(f)"'] & & C(B,-)
\end{tikzcd}
$$

- Remember that the goal is finding out everything about an object A through its relations to other objects.
- So we want a correspondence between objects and their homsets.
- Formally, we want a bijective functor

$$
\begin{aligned}
\mathcal{Y} : \mathbb{C} &\to \mathbb{S}et^{\mathbb{C}} \\
A &\mapsto \mathbb{C}(A,-)
\end{aligned}
$$

The Yoneda Embedding

$$\begin{tikzcd}
A \arrow[r, "f \,\in\, \mathbb{C}(A,B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\
\mathbb{C}(A,-) \arrow[r, "\mathcal{Y}(f)"'] & C(B,-)
\end{tikzcd}$$

- Remember that the goal is finding out everything about an object A through its relations to other objects.
- So we want a correspondence between objects and their homsets.
- Formally, we want a bijective functor

$$\begin{aligned} \mathcal{Y} : \mathbb{C} &\to \mathbb{S}et^{\mathbb{C}} \\ A &\mapsto \mathbb{C}(A,-) \end{aligned}$$

- We call $\mathcal{Y}$ the Yoneda Embedding.

The Yoneda Embedding

$$\begin{tikzcd} A \arrow[rr, "f \in \mathbb{C}(A{,}B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\ \mathbb{C}(A, -) \arrow[rr, "\mathcal{Y}(f)"'] & & C(B, -) \end{tikzcd}$$

- Remember that the goal is finding out everything about an object A through its relations to other objects.
- So we want a correspondence between objects and their homsets.
- Formally, we want a bijective functor

$$\begin{aligned} \mathcal{Y} : \mathbb{C} &\to \mathbb{S}et^{\mathbb{C}} \\ A &\mapsto \mathbb{C}(A, -) \end{aligned}$$

- We call $\mathcal{Y}$ the Yoneda Embedding.
- Given $f \in \mathbb{C}(A, B)$, $\mathcal{Y}(f)$ is a morphism between $\mathbb{C}(A, -)$ and $\mathbb{C}(B, -)$ in the functor category $\mathbb{S}et^{\mathbb{C}}$.

The Yoneda Embedding

$$
\begin{tikzcd}
A \arrow[r, "f \in \mathbb{C}(A{,}B)"] \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"'] & B \arrow[d, dotted, leftrightarrow, "\mathcal{Y}"] \\
\mathbb{C}(A, -) \arrow[r, "\mathcal{Y}(f)"'] & C(B, -)
\end{tikzcd}
$$

- Remember that the goal is finding out everything about an object A through its relations to other objects.
- So we want a correspondence between objects and their homsets.
- Formally, we want a bijective functor

$$
\begin{aligned}
\mathcal{Y} : \mathbb{C} &\to \mathbb{S}et^{\mathbb{C}} \\
A &\mapsto \mathbb{C}(A, -)
\end{aligned}
$$

- We call $\mathcal{Y}$ the Yoneda Embedding.
- Given $f \in \mathbb{C}(A, B)$, $\mathcal{Y}(f)$ is a morphism between $\mathbb{C}(A, -)$ and $\mathbb{C}(B, -)$ in the functor category $\mathbb{S}et^{\mathbb{C}}$.
- Therefore, $\mathcal{Y}(f)$ is a natural transformation between $\mathbb{C}(A, -)$ and $\mathbb{C}(B, -)$

The Yoneda Lemma

- It turns out we can do even better!

The Yoneda Lemma

- It turns out we can do even better!
- We can construct the set of all natural transformations between $\mathbb{C}(A, -)$ and <u>any</u> Functor $F : \mathbb{C} \to \mathbb{S}et$.

The Yoneda Lemma

- It turns out we can do even better!
- We can construct the set of all natural transformations between $\mathbb{C}(A, -)$ and <u>any</u> Functor $F : \mathbb{C} \to \mathbb{S}et$.
- Specifically, the Yoneda Lemma states that:

$$\mathrm{Nat}(\mathbb{C}(A, -), F) \simeq F(A))$$

The Yoneda Lemma

- It turns out we can do even better!
- We can construct the set of all natural transformations between $\mathbb{C}(A, -)$ and <u>any</u> Functor $F : \mathbb{C} \to \mathbb{S}et$.
- Specifically, the Yoneda Lemma states that:

$$\mathrm{Nat}(\mathbb{C}(A, -), F) \simeq F(A))$$

  - Furthermore, this isomorphism is a natural transformation.

The Yoneda Lemma

- It turns out we can do even better!
- We can construct the set of all natural transformations between $\mathbb{C}(A, -)$ and <u>any</u> Functor $F : \mathbb{C} \rightarrow \mathbb{S}et$.
- Specifically, the Yoneda Lemma states that:

$$\mathrm{Nat}(\mathbb{C}(A, -), F) \simeq F(A))$$

  - Furthermore, this isomorphism is a natural transformation.
  - So we can construct a unique $\mathcal{Y}$ with our desired properties from any element $F(A)$.

The Yoneda Lemma

- It turns out we can do even better!
- We can construct the set of all natural transformations between $\mathbb{C}(A, -)$ and <u>any</u> Functor $F : \mathbb{C} \to \mathbb{S}et$.
- Specifically, the Yoneda Lemma states that:

$$\mathrm{Nat}(\mathbb{C}(A, -), F) \simeq F(A))$$

  - Furthermore, this isomorphism is a natural transformation.
  - So we can construct a unique $\mathcal{Y}$ with our desired properties from any element $F(A)$.
  - Vice versa, if we know all natural transformations $\mathrm{Nat}(\mathbb{C}(A, -), F)$, we can construct the set $F(A)$.

Instances of the Yoneda Lemma

- Cayley's theorem in group theory
- Countless theorems in algebra, particulary in algebraic topology
- Proofs by indirect inequality: $b \preceq a$ iff. $\forall c : (a \preceq c) \implies (b \preceq c)$
- Profunctor optics in functional programming

Profunctor Optics