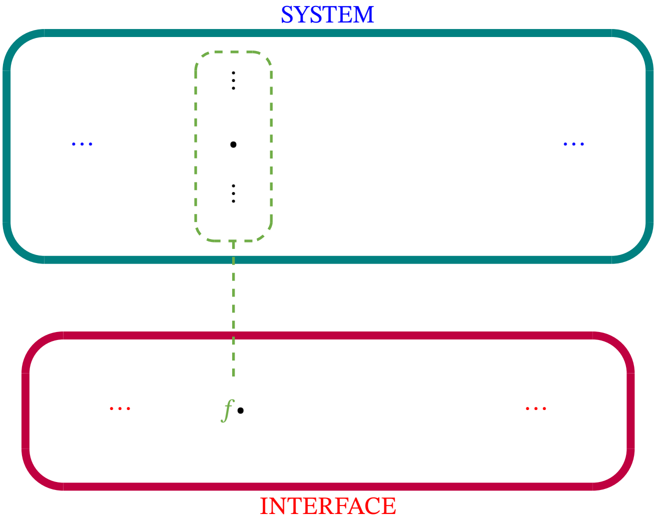
Limits and Colimits in a Category of Lenses

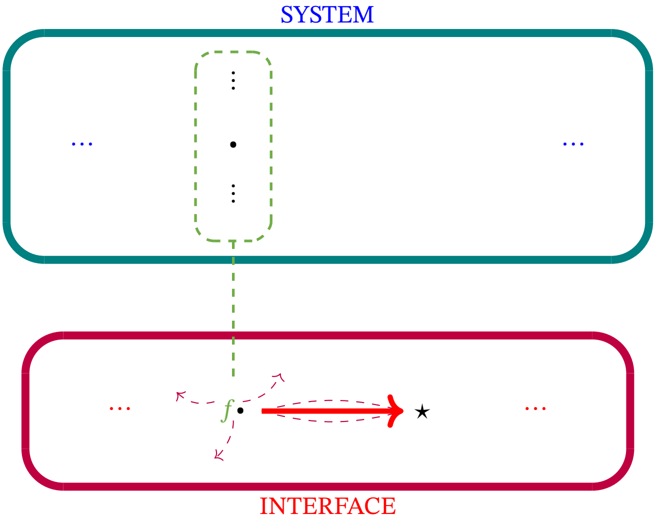
Emma Chollet, Bryce Clarke, Michael Johnson, Maurine Songa,
Vincent Wang, **Gioele Zardini**

4th International Conference on Applied Category Theory (ACT2021)
Cambridge (UK)

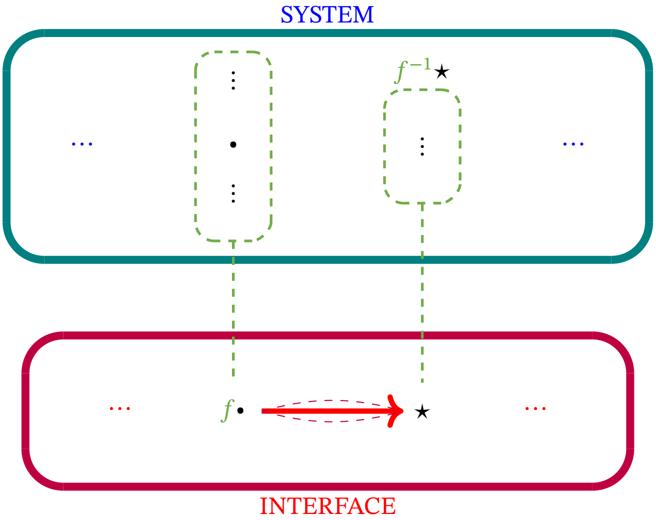
Lenses, informally



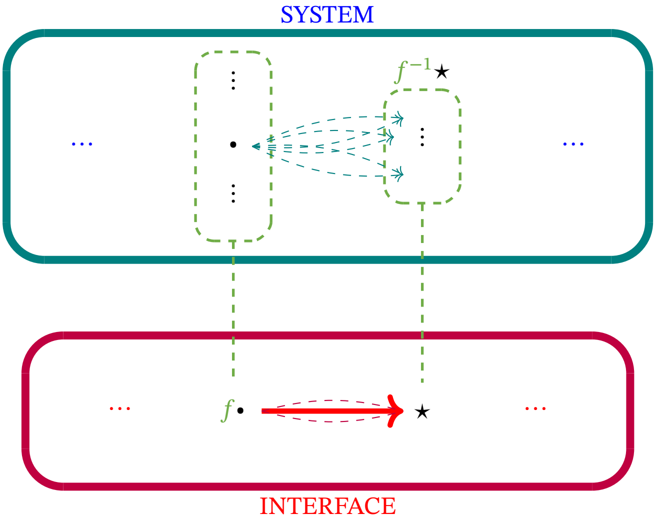
Lenses, informally



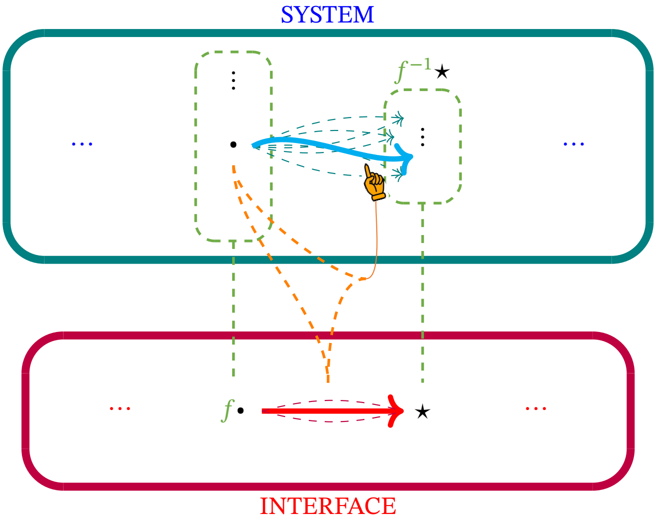
Lenses, informally



Lenses, informally



Lenses, informally



“Lenses”

- **Lawful**
- **Category-Based**
- **Asymmetric**

“Lenses”

- **Lawful**
  - (*vs.* “Lawless”): Wild-West of Machine Learning, Game Theory, Economics...
- **Category-Based**
- **Asymmetric**

“Lenses”

- **Lawful**
  - (*vs.* “Lawless”): Wild-West of Machine Learning, Game Theory, Economics...
- **Category-Based**
  - Strictly generalises Set-Based Lenses
  - Also known as “Delta Lenses”
- **Asymmetric**

“Lenses”

- **Lawful**
  - (*vs.* “Lawless”): Wild-West of Machine Learning, Game Theory, Economics...
- **Category-Based**
  - Strictly generalises Set-Based Lenses
  - Also known as “Delta Lenses”
- **Asymmetric**
  - Asymmetric: One system knows everything the other does
  - Symmetric: Either system may know something the other doesn’t
  - All Symmetric Lenses can be constructed from Asymmetric ones

“Lenses”

- **Lawful (Specification for *niceness*)**
  - (*vs.* “Lawless”): Wild-West of Machine Learning, Game Theory, Economics...
- **Category-Based (Generalisation)**
  - Strictly generalises Set-Based Lenses
  - Also known as “Delta Lenses”
- **Asymmetric (Sufficiency)**
  - Asymmetric: One system knows everything the other does
  - Symmetric: Either system may know something the other doesn’t
  - All Symmetric Lenses can be constructed from Asymmetric ones

Nice but *general* Lenses

Lenses, formally (Nuts-and-Bolts)

Definition

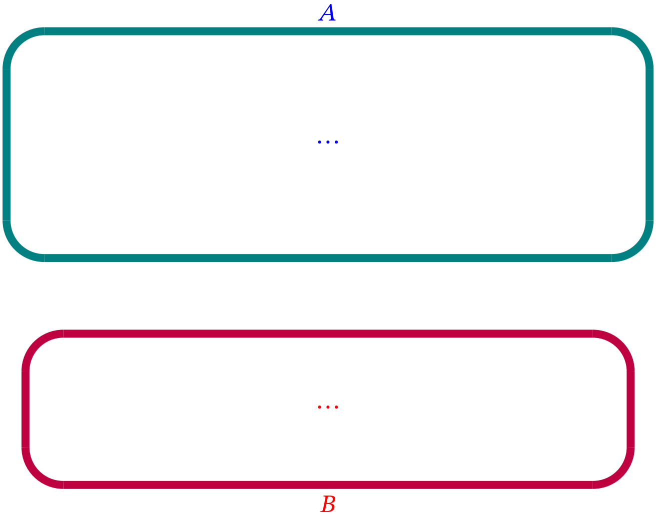
Let A and B be categories. A *lens* $\langle f, \varphi \rangle : A \rightleftharpoons B$ consists of a functor $f \colon A \to B$ and a lifting operation,

$$(a \in A, u \colon fa \to b \in B) \quad \longmapsto \quad \varphi(a, u) \colon a \to a' \in A$$

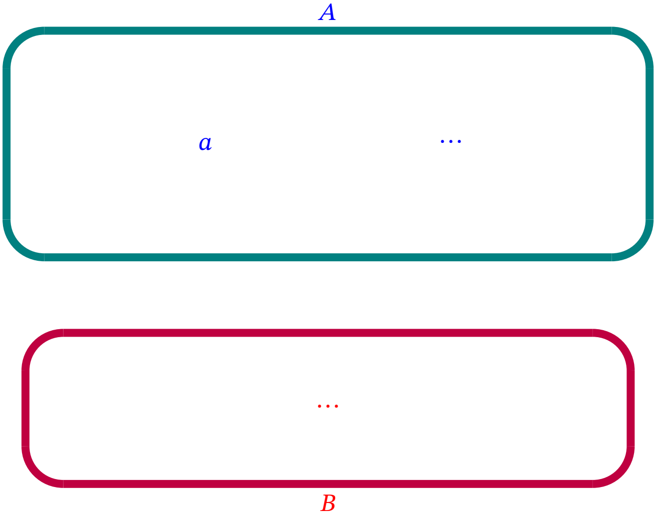
which satisfies the following axioms:

1. $f\varphi(a, u) = u$;
2. $\varphi(a, 1_{fa}) = 1_a$;
3. $\varphi(a, v \circ u) = \varphi(a', v) \circ \varphi(a, u)$.

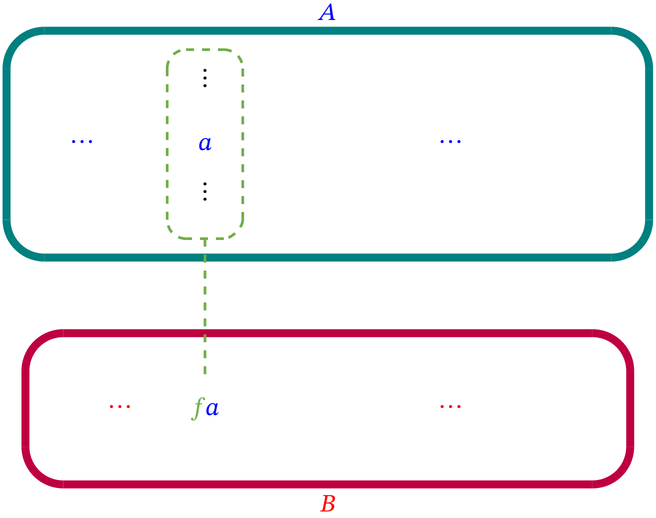
Lenses, formally (Nuts-and-Bolts)



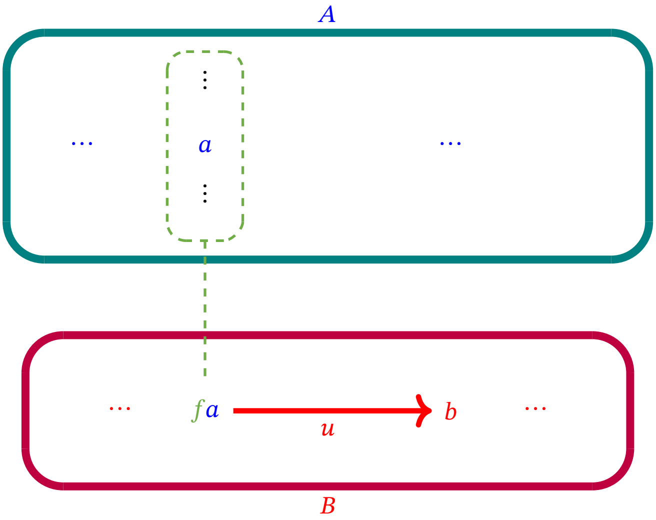
Lenses, formally (Nuts-and-Bolts)



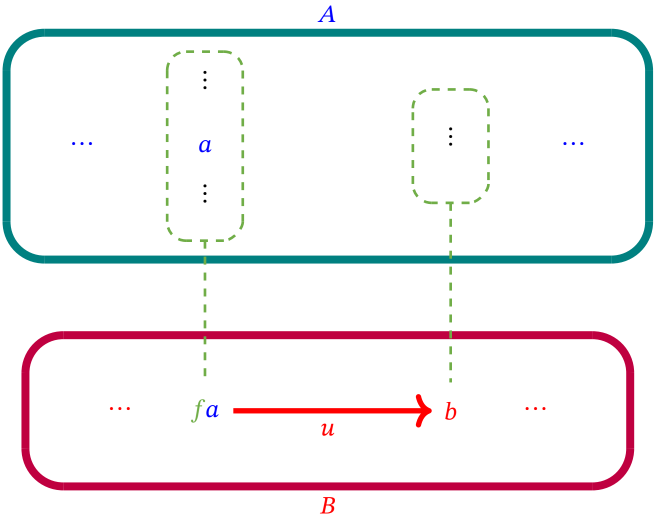
Lenses, formally (Nuts-and-Bolts)



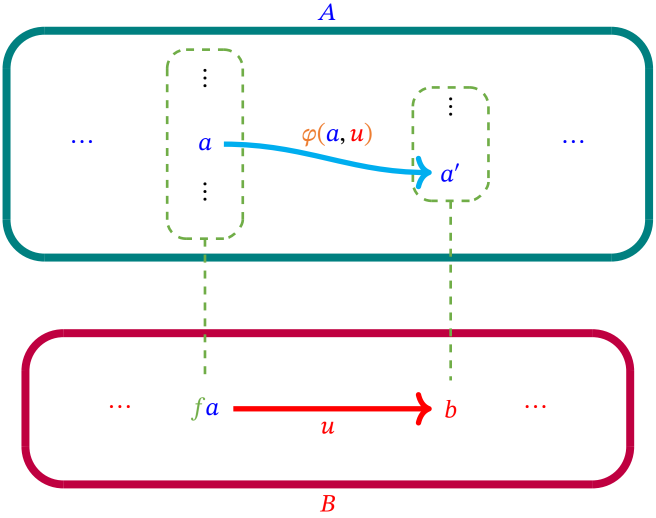
Lenses, formally (Nuts-and-Bolts)



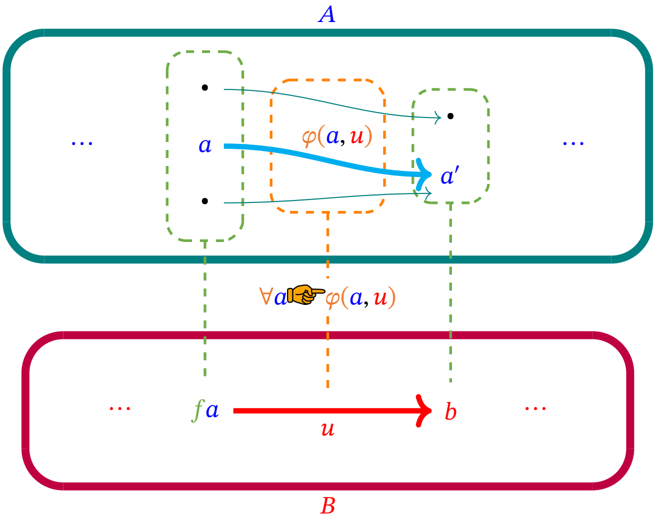
Lenses, formally (Nuts-and-Bolts)



Lenses, formally (Nuts-and-Bolts)



Lenses, formally (Nuts-and-Bolts)



Lenses, formally (Nuts-and-Bolts)

Definition

Let A and B be categories. A *lens* $\langle f, \varphi \rangle : A \rightleftharpoons B$ consists of a functor $f : A \to B$ and a lifting operation,

$$(a \in A, u : fa \to b \in B) \quad \longmapsto \quad \varphi(a, u) : a \to a' \in A$$

which satisfies the following axioms:

1. $f\varphi(a, u) = u$
2. $\varphi(a, 1_{fa}) = 1_a$
3. $\varphi(a, v \circ u) = \varphi(a', v) \circ \varphi(a, u)$

Lenses, formally (Nuts-and-Bolts)

Definition

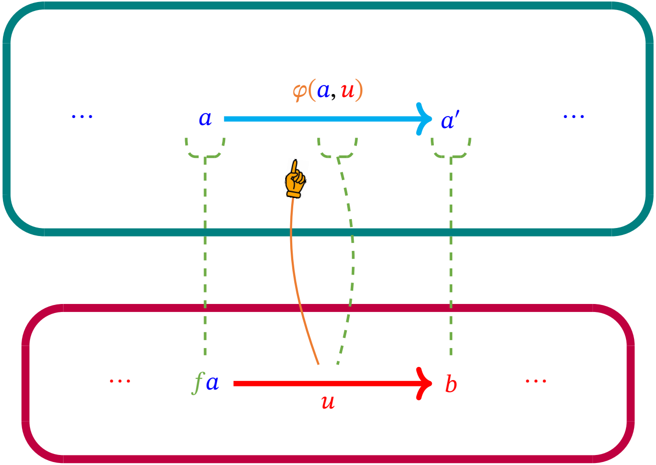
Let A and B be categories. A *lens* $\langle f, \varphi \rangle : A \rightleftharpoons B$ consists of a functor $f : A \to B$ and a lifting operation,

$$(a \in A, u : fa \to b \in B) \quad \longmapsto \quad \varphi(a, u) : a \to a' \in A$$

which satisfies the following axioms:

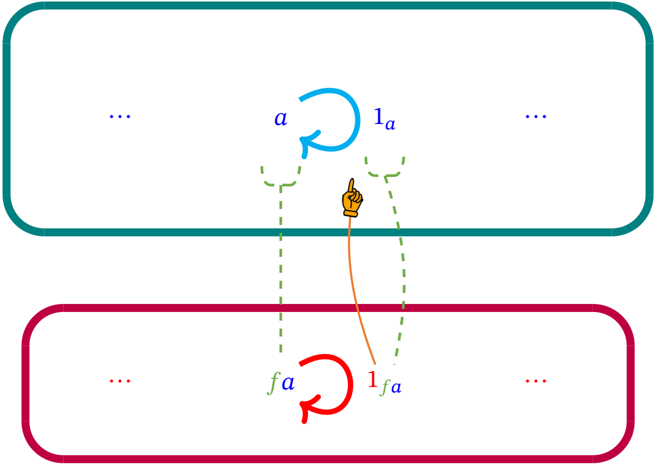
1. $f\varphi(a, u) = u$ **Put followed by Get is trivial for morphisms**
2. $\varphi(a, 1_{fa}) = 1_a$ **Get followed by Put preserves identities**
3. $\varphi(a, v \circ u) = \varphi(a', v) \circ \varphi(a, u)$ **The Put of composites is the composite of Puts**

Lenses, formally (Nuts-and-Bolts)



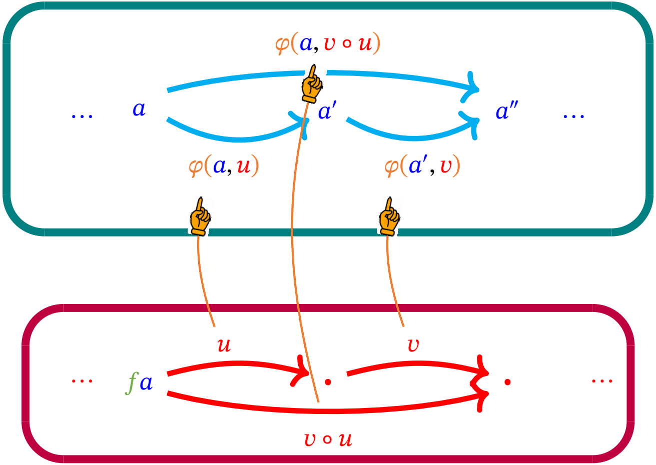
$$f\varphi(a, u) = u$$

Lenses, Formally (Nuts-and-Bolts)



$$\varphi(a, 1_{fa}) = 1_a$$

Lenses, formally (Nuts-and-Bolts)



$$\varphi(a, v \circ u) = \varphi(a', v) \circ \varphi(a, u)$$

Lenses, formally (slick)

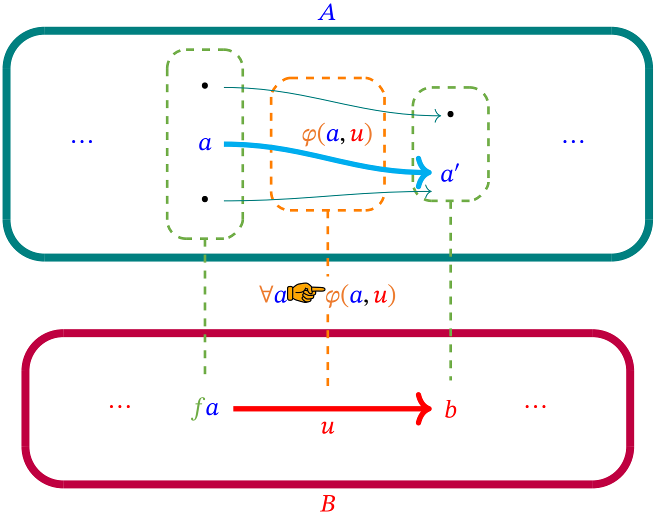
Proposition (Lenses as Functors and Cofunctors)

Every lens $\langle f, \varphi \rangle : A \rightleftharpoons B$ may be represented as a commutative diagram of functors,

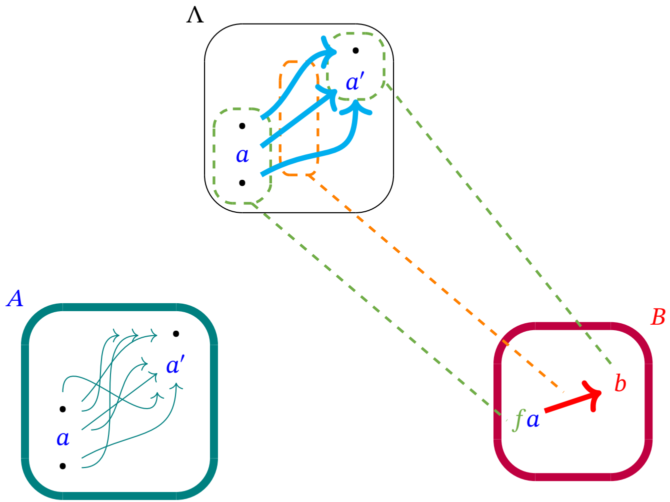
$$\begin{tikzcd} & \Lambda \arrow[dl, "\varphi"'] \arrow[dr, "\bar{\varphi}"] & \\ A \arrow[rr, "f"'] & & B \end{tikzcd}$$

where φ is a faithful, identity-on-objects functor and $\bar{\varphi}$ is a discrete opfibration.

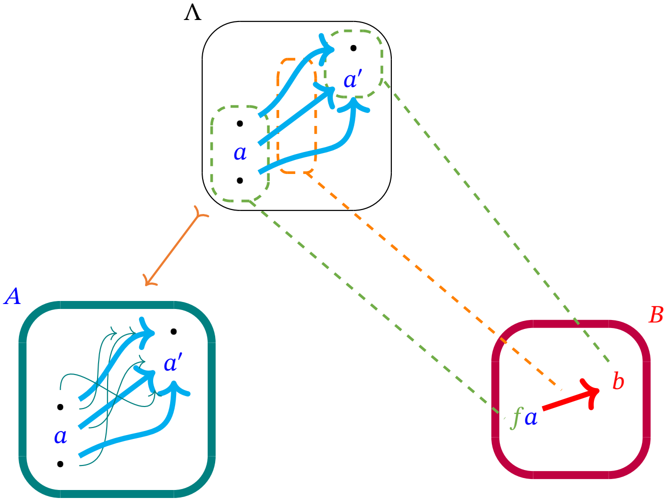
Lenses, formally (slick)



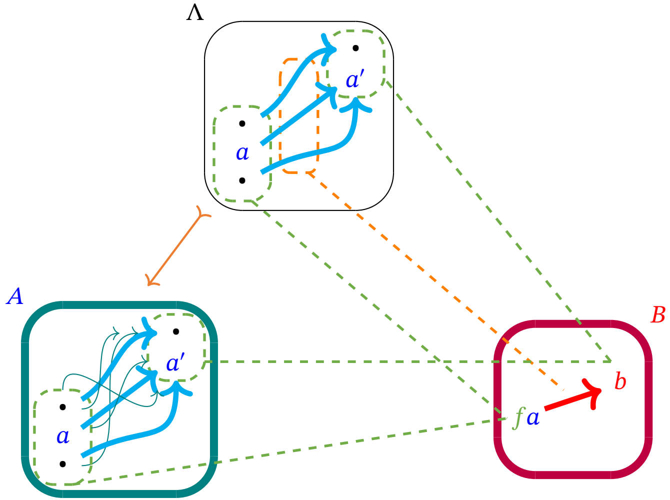
Lenses, formally (slick)



Lenses, formally (slick)



Lenses, formally (slick)



Lenses, formally (slick)

$$\begin{tikzcd}
 & \Lambda \arrow[dl, "\text{Id-on-Obj.}"'] \arrow[dr, "\text{D.Opf.}"] & \\
A \arrow[rr, "f"'] & & B
\end{tikzcd}$$

The category $\mathcal{L}$ens

Definition

Let $\mathcal{L}$ens denote the category whose objects are categories and whose morphisms are lenses. Given a pair of lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$ and $\langle g, \gamma \rangle : B \rightleftharpoons C$, their composite is given by the functor $g \circ f : A \to C$ together with the lifting operation:

$$\langle a \in A, u \colon gfa \to c \in C \rangle \quad \longmapsto \quad \varphi(a, \gamma(fa, u)).$$

Actually, $\mathcal{L}$ens is a pretty place

Actually, $\mathcal{L}$ens is a pretty place

- $\mathcal{L}$ens has **initial** and **terminal** objects;

Actually, $\mathcal{L}$ens is a pretty place

- $\mathcal{L}$ens has **initial** and **terminal** objects;
- $\mathcal{L}$ens has **small coproducts**;

Actually, $\mathcal{L}$ens is a pretty place

- $\mathcal{L}$ens has **initial** and **terminal** objects;
- $\mathcal{L}$ens has **small coproducts**;
- $\mathcal{L}$ens has **equalisers**;

Actually, $\mathcal{L}$ens is a pretty place

- $\mathcal{L}$ens has **initial** and **terminal** objects;
- $\mathcal{L}$ens has **small coproducts**;
- $\mathcal{L}$ens has **equalisers**;
- $\mathcal{L}$ens has an **orthogonal factorisation system**, which factors every lens into a surjective-on-objects lens (epic) followed by a injective-on-objects discrete opfibration (monic);

Actually, $\mathcal{L}$ens is a pretty place

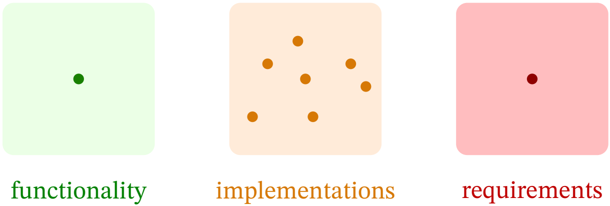
- $\mathcal{L}$ens has **initial** and **terminal** objects;
- $\mathcal{L}$ens has **small coproducts**;
- $\mathcal{L}$ens has **equalisers**;
- $\mathcal{L}$ens has an **orthogonal factorisation system**, which factors every lens into a surjective-on-objects lens (epic) followed by a injective-on-objects discrete opfibration (monic);
- Limit constructions imported from $\mathcal{C}$at behave well, even if they are missing universal property in $\mathcal{L}$ens: we have **distributivity** of *imported* products over coproducts, and **extensivity**.

Actually, $\mathcal{L}$ens is a pretty place

- $\mathcal{L}$ens has **initial** and **terminal** objects;
- $\mathcal{L}$ens has **small coproducts**;
- $\mathcal{L}$ens has **equalisers**;
- $\mathcal{L}$ens has an **orthogonal factorisation system**, which factors every lens into a surjective-on-objects lens (epic) followed by a injective-on-objects discrete opfibration (monic);
- Limit constructions imported from $\mathcal{C}$at behave well, even if they are missing universal property in $\mathcal{L}$ens: we have **distributivity** of *imported* products over coproducts, and **extensivity**.
- (Next Talk): $\mathcal{L}$ens has **(certain) coequalisers**

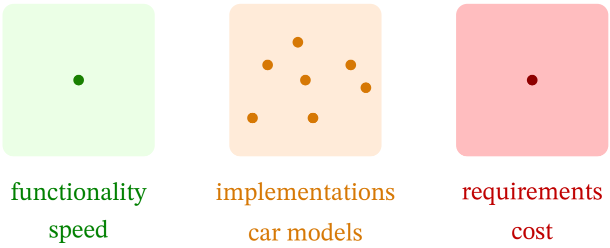
Engineering co-design as a guiding example

- Design is characterised by three spaces:
  - **implementation space**: the options we can choose from;
  - **functionality space**: what we need to provide/achieve;
  - **requirements/costs space**: resources we need to have available;



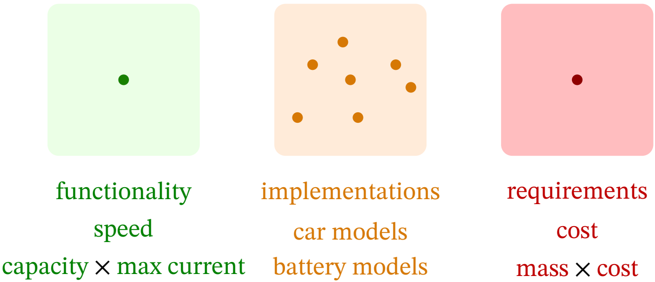
Engineering co-design as a guiding example

- Design is characterised by three spaces:
  - **implementation space**: the options we can choose from;
  - **functionality space**: what we need to provide/achieve;
  - **requirements/costs space**: resources we need to have available;



Engineering co-design as a guiding example

- Design is characterised by three spaces:
  - **implementation space**: the options we can choose from;
  - **functionality space**: what we need to provide/achieve;
  - **requirements/costs space**: resources we need to have available;

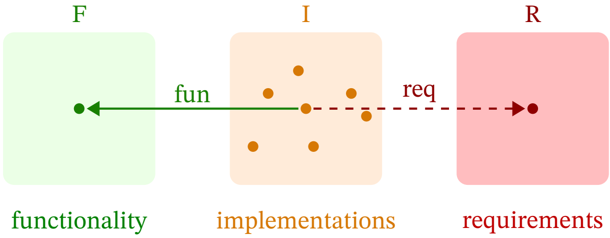


Design problems, formally

Definition

A *design problem with implementation* (DPI) is a tuple $\langle \mathrm{F}, \mathrm{R}, \mathrm{I}, \mathrm{fun}, \mathrm{req} \rangle$, where:

- F is a poset, called *functionality space*;
- R is a poset, called *requirements space*;
- I is a set, called *implementation space*;
- the map $\mathrm{fun} : \mathrm{I} \to \mathrm{F}$ maps an implementation to the functionality it provides;
- the map $\mathrm{req} : \mathrm{I} \to \mathrm{R}$ maps an implementation to the resources it requires.



Practically, design problems can be understood as feasibility relations

- For design purposes, we need to know **how** something is done: we need the implementations
- For the algorithmic solution of co-design problems, we consider **feasibility relations** directly;
- A *design problem* is a **boolean profunctor**:

$$
\begin{aligned}
d : \; & \mathrm{F}^{\mathrm{op}} \times \mathrm{R} \to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool} \\
& \langle f^*, r \rangle \mapsto \exists i \in \mathrm{I} : \; (f \preceq_{\mathrm{F}} \mathrm{fun}(i)) \wedge (\mathrm{req}(i) \preceq_{\mathrm{R}} r).
\end{aligned}
$$

Practically, design problems can be understood as feasibility relations

- For design purposes, we need to know **how** something is done: we need the implementations
- For the algorithmic solution of co-design problems, we consider **feasibility relations** directly;
- A *design problem* is a **boolean profunctor**:

$$
\begin{aligned}
d : \mathrm{F}^{\mathrm{op}} \times \mathrm{R} &\to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool} \\
\langle f^*, r \rangle &\mapsto \exists i \in \mathrm{I} : (f \preceq_{\mathrm{F}} \mathrm{fun}(i)) \wedge (\mathrm{req}(i) \preceq_{\mathrm{R}} r).
\end{aligned}
$$

- This is a **monotone** map (morphism in $\mathcal{P}$os):
  - **Lower** functionalities do not require **more** requirements;
  - **Higher** requirements do not provide **less** functionalities
- Design problems form the category **DP**:
  - Objects are posets, morphisms are design problems;
  - Covered in detail in ACT4E (https://applied-compositional-thinking.engineering)

Realizing design problems as lenses

- Consider the design problem related to buying a car based on its speed:

Realizing design problems as lenses

- Consider the design problem related to buying a car based on its speed:



- We consider $d : \mathrm{F}^{\mathrm{op}} \times \mathrm{R} \to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool}$ with posets

$$\begin{array}{ccc} \text{fast} & & \\ | & & \\ \text{average} & & \text{expensive} \\ | & & | \\ \text{slow} & & \text{cheap} \\ \mathrm{F} & & \mathrm{R} \end{array}$$

- *Slow* vehicles are the only *cheap* ones, the rest are *expensive*.

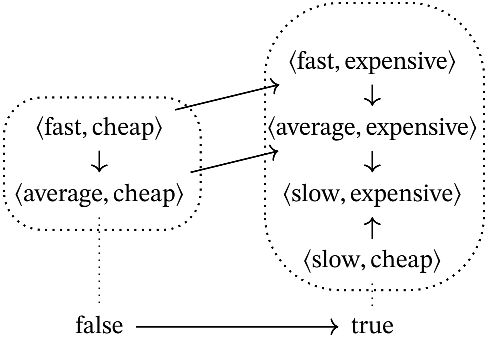
Realising design problems as lenses

- We can represent the functor $\color{green}{\mathrm{F}}^{\mathrm{op}} \times \color{red}{\mathrm{R}} \to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool}$ *fibrewise*.

$$
\begin{tikzcd}
 & \langle \mathrm{fast}, \mathrm{expensive} \rangle \arrow[d] \\
\langle \mathrm{fast}, \mathrm{cheap} \rangle \arrow[d] & \langle \mathrm{average}, \mathrm{expensive} \rangle \arrow[d] \\
\langle \mathrm{average}, \mathrm{cheap} \rangle \arrow[dd, dotted, no head] & \langle \mathrm{slow}, \mathrm{expensive} \rangle \\
 & \langle \mathrm{slow}, \mathrm{cheap} \rangle \arrow[u] \arrow[d, dotted, no head] \\
\mathrm{false} \arrow[r] & \mathrm{true}
\end{tikzcd}
$$

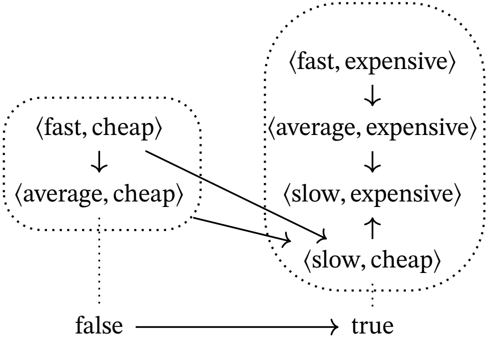
Realising design problems as lenses

- We can represent the functor $\mathrm{F}^{\mathrm{op}} \times \mathrm{R} \to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool}$ *fibrewise*.
- A lens over the functor provides a *unique, reachable* pair in $\mathrm{F}^{\mathrm{op}} \times \mathrm{R}$ from each infeasible pair.
- A lens models **feasibility** and informs **compromises** to make the unfeasible feasible.



Realising design problems as lenses

- We can represent the functor $\mathrm{F}^{\mathrm{op}} \times \mathrm{R} \to_{\mathcal{P}\mathrm{os}} \mathcal{B}\mathrm{ool}$ *fibrewise*.
- A lens over the functor provides a *unique, reachable* pair in $\mathrm{F}^{\mathrm{op}} \times \mathrm{R}$ from each infeasible pair.
- A lens models **feasibility** and informs **compromises** to make the unfeasible feasible.



$\mathcal{L}$ens has small coproducts

- Given lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$, $\langle g, \gamma \rangle : C \rightleftharpoons B$, take the coproduct in $\mathcal{C}$at: $A + C$;
- In $\mathcal{C}$at, coproduct injection functors are **injective-on-objects discrete opfibrations**
- Given lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$, $\langle g, \gamma \rangle : C \rightleftharpoons B$, we have a **unique** lens $A + C \rightleftharpoons B$ with:

$$\begin{tikzcd} & \Lambda + \Omega \arrow[dl, "\varphi+\gamma"'] \arrow[dr, "{[\bar{\varphi},\bar{\gamma}]}"] & \\ A + C \arrow[rr, "{[f,g]}"'] & & B \end{tikzcd}$$

$\mathcal{L}$ens has small coproducts

- Given lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$, $\langle g, \gamma \rangle : C \rightleftharpoons B$, take the coproduct in $\mathcal{C}$at: $A + C$;
- In $\mathcal{C}$at, coproduct injection functors are **injective-on-objects discrete opfibrations**
- Given lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$, $\langle g, \gamma \rangle : C \rightleftharpoons B$, we have a **unique** lens $A + C \rightleftharpoons B$ with:

$$\begin{tikzcd} & \Lambda + \Omega \arrow[dl, "\varphi+\gamma"'] \arrow[dr, "{[\bar{\varphi},\bar{\gamma}]}"] & \\ A + C \arrow[rr, "{[f,g]}"'] & & B \end{tikzcd}$$

Example

- From speed$^{\mathrm{op}}$ $\times$ cost $\rightleftharpoons$ $\mathcal{B}$ool and seats$^{\mathrm{op}}$ $\times$ weight $\rightleftharpoons$ $\mathcal{B}$ool you get

$$\text{speed}^{\mathrm{op}} \times \text{cost} + \text{seats}^{\mathrm{op}} \times \text{weight} \rightleftharpoons \mathcal{B}\text{ool}$$

$\mathcal{L}$ens has equalizers

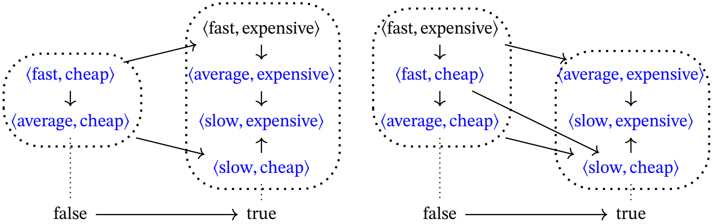
- Consider lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$ and $\langle g, \gamma \rangle : A \rightleftharpoons B$
- One can construct the equaliser $e : E \to A$ of the **underlying functors** in $\mathcal{C}$at.
- Then the equaliser is the largest subobject $m : M \rightarrowtail E$ such that $e \circ m : M \rightleftharpoons E$ is a discrete opfibration which forms a cone over the parallel pair in $\mathcal{L}$ens.

$\mathcal{L}$ens has equalizers

- Consider lenses $\langle f, \varphi \rangle : A \rightleftharpoons B$ and $\langle g, \gamma \rangle : A \rightleftharpoons B$
- One can construct the equaliser $e : E \to A$ of the **underlying functors** in $\mathcal{C}$at.
- Then the equaliser is the largest subobject $m : M \rightarrowtail E$ such that $e \circ m : M \rightleftharpoons E$ is a discrete opfibration which forms a cone over the parallel pair in $\mathcal{L}$ens.

Example

- Consider two design problems (two experts) $\langle f, \varphi \rangle : \mathrm{F}^{\mathrm{op}} \times \mathrm{R} \rightleftharpoons \mathcal{B}\mathrm{ool}$, $\langle g, \gamma \rangle : \mathrm{F}^{\mathrm{op}} \times \mathrm{R} \rightleftharpoons \mathcal{B}\mathrm{ool}$
- Their equalizer $E \rightleftharpoons \mathrm{F}^{\mathrm{op}} \times \mathrm{R}$:
  - **embeds** E into $\mathrm{F}^{\mathrm{op}} \times \mathrm{R}$, and selects pairs in $\mathrm{F}^{\mathrm{op}} \times \mathrm{R}$ for which experts **agree**
  - In the worst case, **total disagreement**, i.e. $E = 0$.



$\mathcal{L}$ens has an orthogonal factorisation system

- Johnson & Rosebrugh showed that $\mathcal{L}$ens admits a **proper orthogonal factorisation system**
- This is actually an **(epi, mono)-factorisation system**, factoring every lens into:
  - A surjective-on-object lens (epimorphism), and
  - A cosieve (monomorphism).

Example

- Consider a lens $\langle f, \varphi \rangle : \text{speed}^{\text{op}} \times \text{cost} \rightleftharpoons \mathcal{B}\text{ool}$ with just *true* values

$$
\begin{tikzcd}
\text{speed}^{\text{op}} \times \text{cost} \arrow[rr, two heads, "\text{epimorphism}"] \arrow[dr, two heads] & & \{\text{true}\} \arrow[dl, tail, "\text{monomorphism}"] \\
& \{\text{false} \to \text{true}\} &
\end{tikzcd}
$$

Conclusion and Outlook

- We considered *nice* but *general* Lenses *sufficiently rich* to model problems of:
  - synchronisation
  - coordination
  - interoperation

Conclusion and Outlook

- We considered *nice* but *general* Lenses *sufficiently rich* to model problems of:
  - synchronisation
  - coordination
  - interoperation
- We studied the category $\mathcal{L}$ens to look for canonical constructions...

Conclusion and Outlook

- We considered *nice* but *general* Lenses *sufficiently rich* to model problems of:
  - synchronisation
  - coordination
  - interoperation
- We studied the category $\mathcal{L}$ens to look for canonical constructions...
- ...and we found some.