How Nice is this Functor?

Two Squares and Some Homology go a Long Way

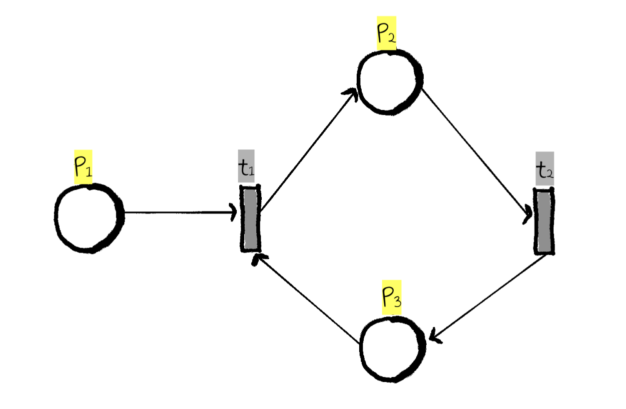
Daniel Rosiak (NIST)

Benjamin Merlin Bumpus, James Fairbanks, Fabrizio Genovese, Caterina Puca, and **Daniel Rosiak**

Motivation

Motivation via Petri nets

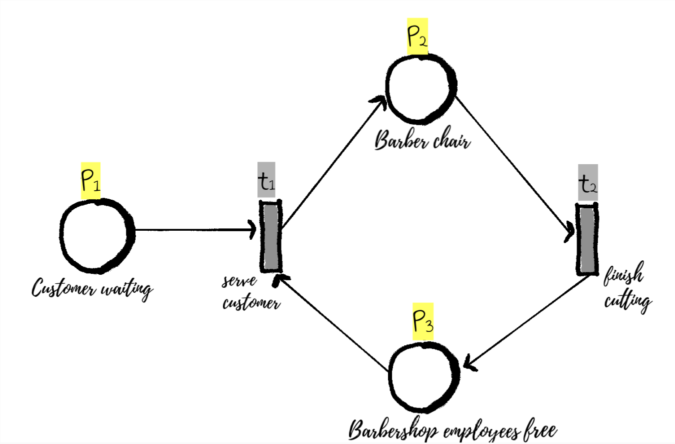
Consider the following Petri net:



Recall that a Petri net fundamentally consists of places (drawn as circles) and transitions (drawn as boxes), where directed edges go from places to transitions and from transitions to places.

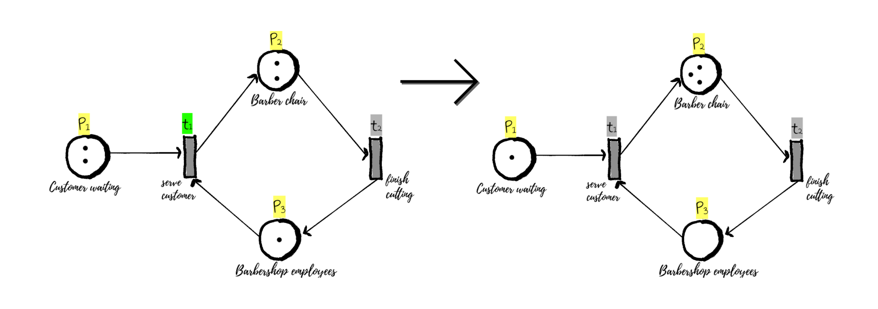
Example

It may represent the action at a barbershop



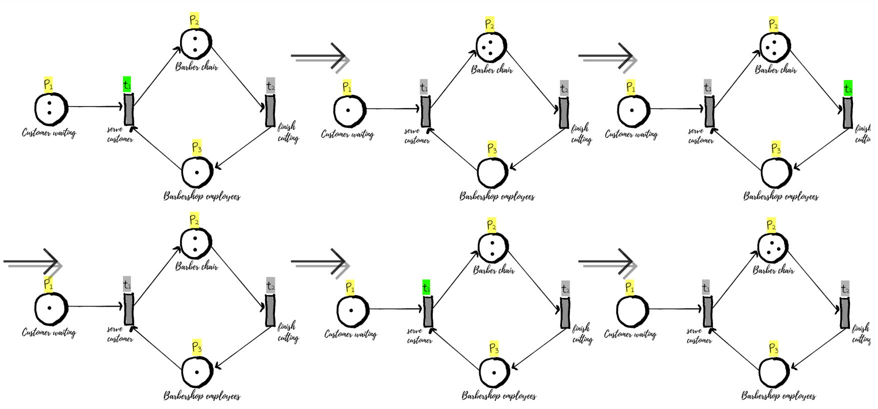
Tokens

We then place dots called tokens in the places, moving them around using the transitions. E.g., the marking on the left becomes (upon firing t_1)



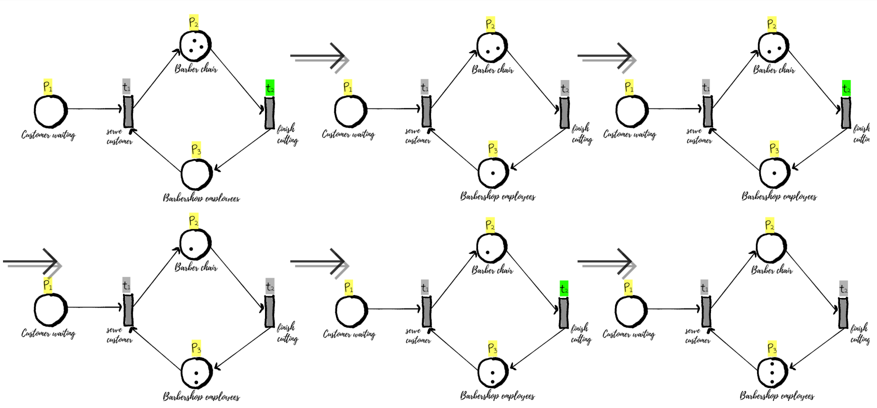
Executions

Altogether, going through certain transitions for the initial marking,



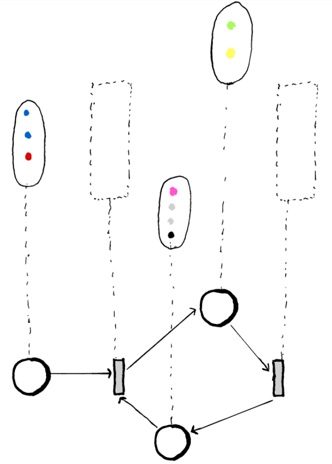
Execution (continued)

continuing...

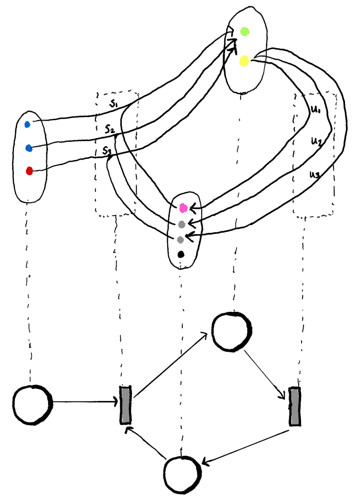


Colored nets

But sometimes we want to distinguish between individual tokens, so we consider colored nets, which you can think of as assigning sets of colors to places



and transition guards to transitions



Colored nets

In more detail, with guarded (or coloured) nets,

- tokens are endowed with colors or attribute (which depend on the place of the token),

Colored nets

In more detail, with guarded (or coloured) nets,

- tokens are endowed with colors or attribute (which depend on the place of the token),
- each arc is decorated with an expression (modifying tokens' attributes as they flow through the net),

Colored nets

In more detail, with guarded (or coloured) nets,

- tokens are endowed with colors or attribute (which depend on the place of the token),
- each arc is decorated with an expression (modifying tokens' attributes as they flow through the net),
- each transition is decorated with a predicate and only fires on tokens whose attributes satisfy the predicate.

Legend

Think

- blue customers are rich
- red customers are poor

and

- pink represents expensive barber(s)
- gray represents moderately-priced barber(s)
- black represents the boss (who doesn't cut hair)

and

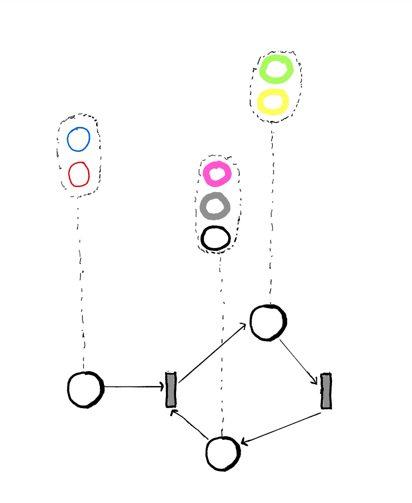
- green represents a fancy barber seat
- yellow represents a normal chair

and where, e.g.,

- s_1 amounts to (customer = one of the rich customers $\wedge$ employee = the expensive barber $\wedge$ barber seat = the fancy one)

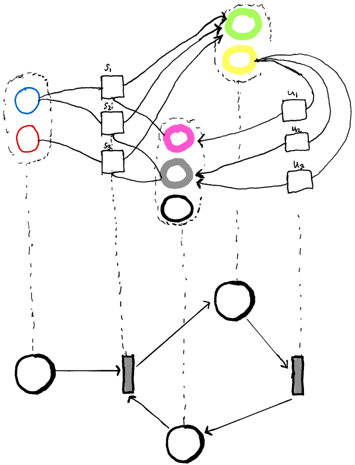
As an unguarded net

Now (via details recalled later in talk), we can convert this guarded/colored net back into a classical net, first by assigning a subplace to each color, as follows

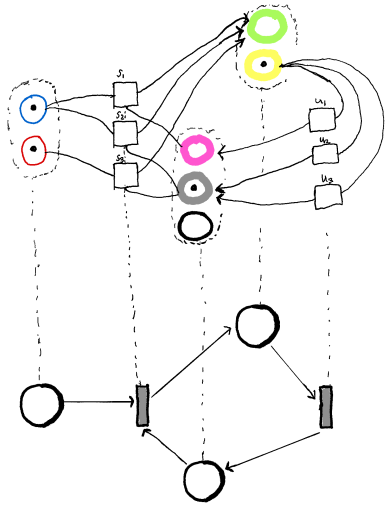


As an unguarded net

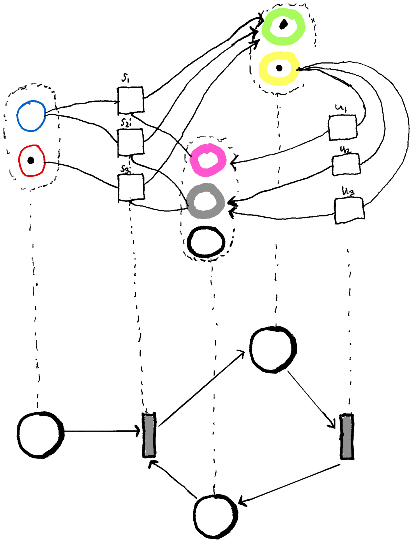
and then a distinct transition box for each of the expressions in the transition guards



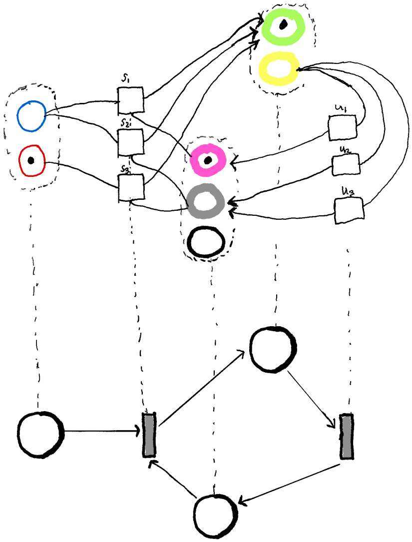
So, then, given a marking of this net, an execution might look like:



An execution

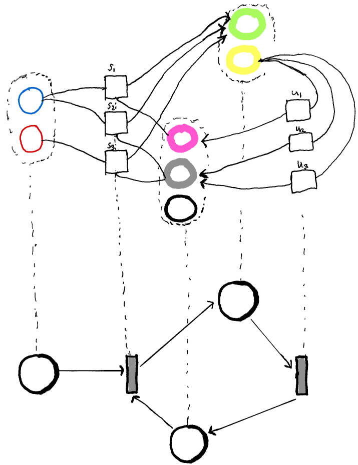


An execution



Observation 1

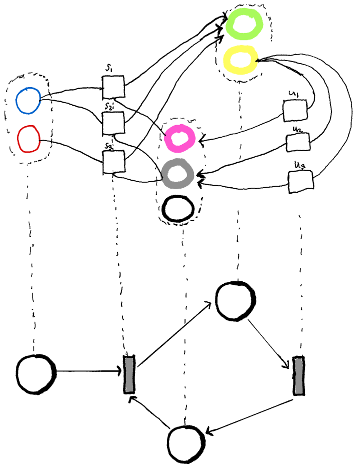
Now let's consider two things. First, consider the green place:



- Essentially, there's an obstruction of sorts, where the same fancy chair may have sent to it multiple customer-barber pairs.

Observation 1

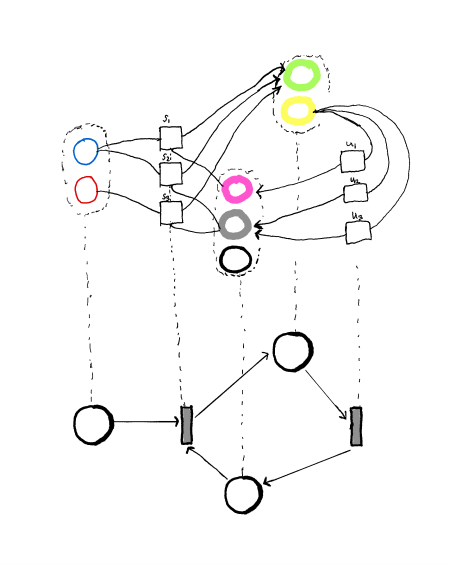
Now let's consider two things. First, consider the green place:



- Essentially, there's an obstruction of sorts, where the same fancy chair may have sent to it multiple customer-barber pairs.
- Looking ahead: we can frame this as a violation of (or obstruction to) the uniqueness condition required for being a discrete fibration.

Observation 2

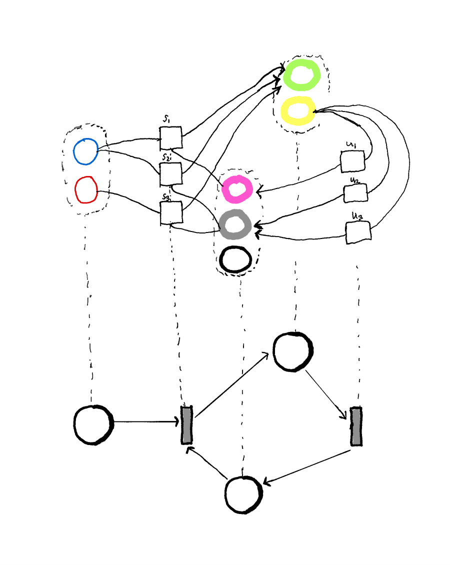
Second, consider the yellow place...



- Essentially, there's another obstruction of sorts, where there's a yellow chair type from which we release an employee back into the pool of available barbershop employees, even when there's no customer whose hair they could have been cutting at that chair.

Observation 2

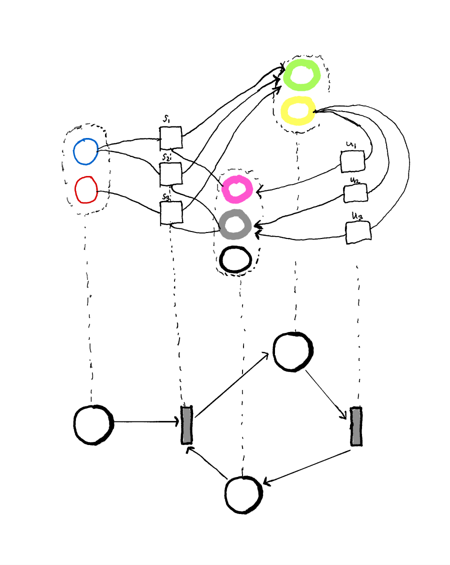
Second, consider the yellow place...



- Essentially, there’s another obstruction of sorts, where there’s a yellow chair type from which we release an employee back into the pool of available barbershop employees, even when there’s no customer whose hair they could have been cutting at that chair.
- This suggests the barber was released from their station at that chair without ever having served any customer. This is something to flag, as maybe barbers get paid whenever they get sent back from a chair into the pool of available employees. But we only want to pay barbers that actually cut a customer’s hair!

Observation 2

Second, consider the yellow place...



- Essentially, there's another obstruction of sorts, where there's a yellow chair type from which we release an employee back into the pool of available barbershop employees, even when there's no customer whose hair they could have been cutting at that chair.
- This suggests the barber was released from their station at that chair without ever having served any customer. This is something to flag, as maybe barbers get paid whenever they get sent back from a chair into the pool of available employees. But we only want to pay barbers that actually cut a customer's hair!
- Looking ahead: we can frame this as a violation of the existence condition of being a discrete fibration.

Bounded Nets

- Another question that arises for Petri nets brings us to bounded nets, for which we can isolate similar sorts of ‘obstructions’.

Bounded Nets

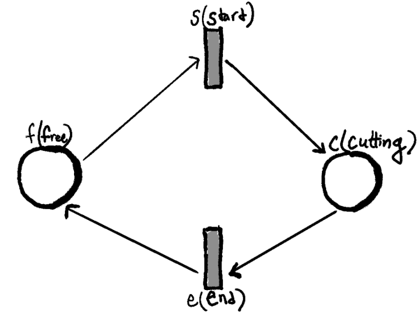
- Another question that arises for Petri nets brings us to bounded nets, for which we can isolate similar sorts of 'obstructions'.
- In such settings, we want to establish when a given Petri net is bounded, meaning that starting from a given marking, no place will hold more than a predetermined number of tokens throughout any possible firing.

Bounded Nets

- Another question that arises for Petri nets brings us to bounded nets, for which we can isolate similar sorts of 'obstructions'.
- In such settings, we want to establish when a given Petri net is bounded, meaning that starting from a given marking, no place will hold more than a predetermined number of tokens throughout any possible firing.
- Classically, we can turn any net into a bounded one by
  - doubling-up the places adding what we call anti-places (keeping track of how many tokens we can still add to the place), and
  - editing transitions so that each input (output) from (to) a place is now paired with a corresponding output (input) to (from) the corresponding anti-place.

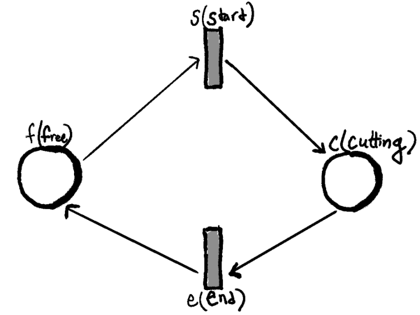
Example

Suppose we start with a simple net as follows:



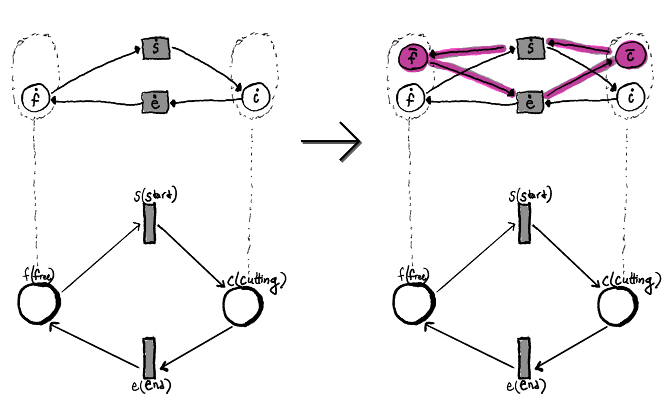
Example

Suppose we start with a simple net as follows:



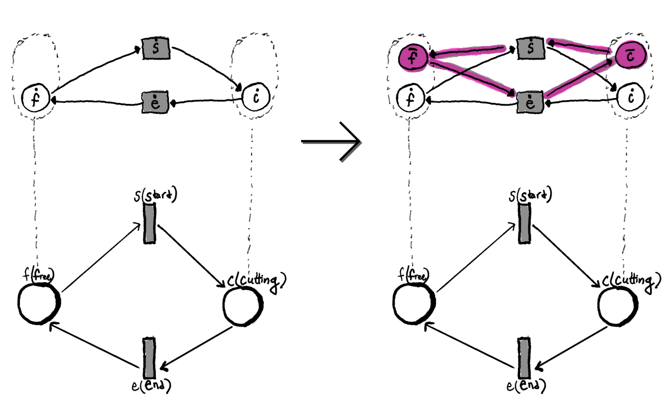
We then construct a bounded net by taking the places and doubling-up with anti-places (depicted in purple) and new arcs to and from transitions turning things around...

Example bounded net



- E.g., three tokens in the antiplace $\bar{f}$ means '3 more tokens can be added to f' (i.e., there are only 3 more barbers that can cut hair).

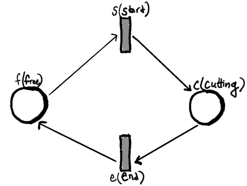
Example bounded net



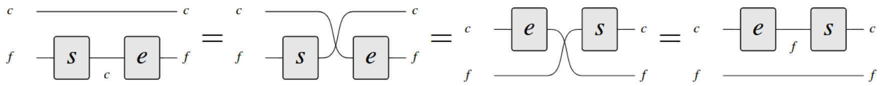
- E.g., three tokens in the antiplace $\bar{f}$ means '3 more tokens can be added to f' (i.e., there are only 3 more barbers that can cut hair).
- Transitions consuming tokens from a place add tokens to the corresponding antiplace; transitions outputting tokens into a place need consume equivalent amount from antiplace.

Observation

Consider the original net

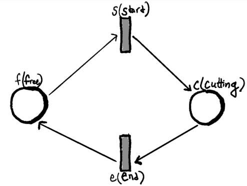


for which we have a particular execution, which can be written down in two equivalent ways (because our semantics is commutative)

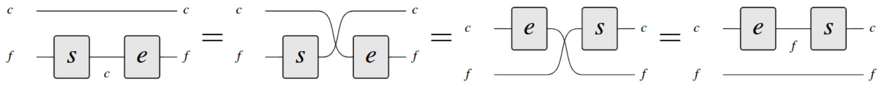


Observation

Consider the original net



for which we have a particular execution, which can be written down in two equivalent ways (because our semantics is commutative)



Think: a barber only gets paid when they end a hair cutting; the owner/balance sheet doesn't see the difference between the different ways of writing the execution (money is debited once).

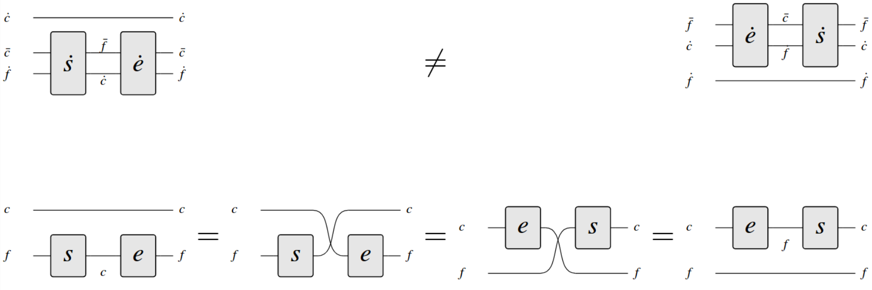
Unpacking

But this equality is not lifted to the bounded net up top, on account of the doubled places.

Unpacking

But this equality is not lifted to the bounded net up top, on account of the doubled places.

Whereas we could exchange any place c with itself down below (using commutativity), we cannot do the same up above with $\dot{c}$ and $\bar{c}$, the place and antiplace corresponding to c, are considered as different generators and cannot be swapped one for the other.



Moral

- Such ‘obstructions’ witness histories that should ‘morally be identified’ in the category of executions of the bounded net, but are not.

Moral

- Such ‘obstructions’ witness histories that should ‘morally be identified’ in the category of executions of the bounded net, but are not.
- In our example: keeping track of bounds on resources (like free barbers and barbers-on-the-job), the owner must now regard as distinct histories that really ‘ought to be identified’ (from a payment perspective, say).

Moral

- Such ‘obstructions’ witness histories that should ‘morally be identified’ in the category of executions of the bounded net, but are not.
- In our example: keeping track of bounds on resources (like free barbers and barbers-on-the-job), the owner must now regard as distinct histories that really ‘ought to be identified’ (from a payment perspective, say).
- These can be considered imperfections of the bounding technique.

Purpose

- These are just a few of the applications where we're interested in **qualifying obstructions**.

Purpose

- These are just a few of the applications where we're interested in **qualifying obstructions**.
- Our paper focused on leveraging some straightforward theory (combining established facts) to satisfy the goal of qualifying obstructions to a variety of properties a functor may satisfy, **applying homology** directly to the categorical setting.

Purpose

- These are just a few of the applications where we're interested in **qualifying obstructions**.
- Our paper focused on leveraging some straightforward theory (combining established facts) to satisfy the goal of qualifying obstructions to a variety of properties a functor may satisfy, **applying homology** directly to the categorical setting.
- Using tools borrowed from homology, we will be able to measure how much a functor fails to be 'something' — a discrete (op-)fibration, a pseudofunctor, …— depending on context.

Context

This refines the intuition that topological holes and obstructions to category-theoretic compositionality should be regarded as two sides of the same coin.

Context

This refines the intuition that topological holes and obstructions to category-theoretic compositionality should be regarded as two sides of the same coin.

This work falls within the line of thought that

Motto

compositionality, far from being a universal notion, should be understood as involving a spectrum of distinct, context-dependent nuances.

Basic Set-up: first observations

Observe

Given a(ny flavor of) category C, we can always consider the following diagram:

$$\cdots \;\; \begin{matrix}\longrightarrow\\ \longrightarrow\\ \longrightarrow\end{matrix} \;\; C_3 \;\; \begin{matrix}\xrightarrow{\;\fatsemi^{l}\;}\\ \xrightarrow{\;\fatsemi^{r}\;}\end{matrix} \;\; C_2 \xrightarrow{\;\fatsemi\;} C_1 \;\; \begin{matrix}\xrightarrow{\;s\;}\\ \xrightarrow{\;t\;}\end{matrix} \;\; C_0$$

- C_0 represents the objects of C,

Observe

Given a(ny flavor of) category C, we can always consider the following diagram:

$$\cdots \; \substack{\longrightarrow \\ \longrightarrow \\ \longrightarrow} \; C_3 \; \underset{\fatsemi^r}{\overset{\fatsemi^l}{\rightrightarrows}} \; C_2 \xrightarrow{\;\fatsemi\;} C_1 \; \underset{t}{\overset{s}{\rightrightarrows}} \; C_0$$

- C_0 represents the objects of C,
- C_1 represents its morphisms,

Observe

Given a(ny flavor of) category C, we can always consider the following diagram:

$$\cdots \; \substack{\longrightarrow \\ \longrightarrow \\ \longrightarrow} \; C_3 \; \substack{\xrightarrow{\fatsemi^l} \\ \xrightarrow{\fatsemi^r}} \; C_2 \xrightarrow{\fatsemi} C_1 \; \substack{\xrightarrow{s} \\ \xrightarrow{t}} \; C_0$$

- C_0 represents the objects of C,
- C_1 represents its morphisms,
- C_n represents n-tuples of morphisms with matching codomain and domain; e.g., $(A \xrightarrow{f} B, B \xrightarrow{g} C)$ could be an element of C_2.

Unpacking the pieces

- $C_1 \underset{t}{\overset{s}{\rightrightarrows}} C_0$

Unpacking the pieces

The arrows s, t represent the usual source and target assignments, s taking a generic $f : A \to B$ to A, t taking f to B.

Unpacking the pieces

- $C_1 \underset{t}{\overset{s}{\rightrightarrows}} C_0$
- $C_2 \xrightarrow{\fatsemi} C_1$

Unpacking the pieces

$\fatsemi$ represents composition, sending the generic pair of compatible morphisms $f : A \to B, g : B \to C$ to the single morphism $f \fatsemi g : A \to C$

Unpacking the pieces

- $C_1 \overset{s}{\underset{t}{\rightrightarrows}} C_0$
- $C_2 \xrightarrow{\fatsemi} C_1$
- $C_3 \overset{\fatsemi^l}{\underset{\fatsemi^r}{\rightrightarrows}} C_2$

Unpacking the pieces

C_3 consists of triples (f, g, h) of composable morphisms and the two maps $\fatsemi^l$ ("compose on the left") and $\fatsemi^r$ ("compose on the right") take such triples of composable morphisms to pairs of morphisms in C_2, i.e., to $(f \fatsemi g, h)$ or to $(f, g \fatsemi h)$.

Unpacking the pieces

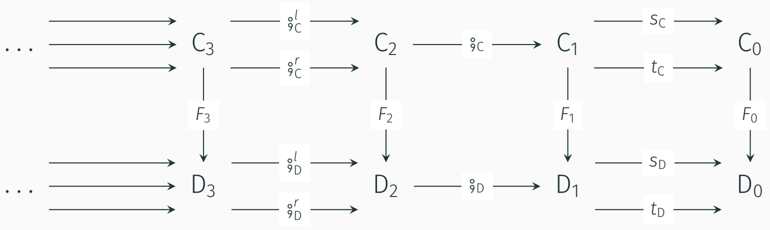
- $C_1 \underset{t}{\overset{s}{\rightrightarrows}} C_0$
- $C_2 \xrightarrow{\fatsemi} C_1$
- $C_3 \underset{\fatsemi^r}{\overset{\fatsemi^l}{\rightrightarrows}} C_2$
- $\ldots \; \overset{\longrightarrow}{\underset{\longrightarrow}{\longrightarrow}} \; C_3$

Unpacking the pieces

More generally, there are n different composition mappings from C_{n+1} to C_n, as we can choose to compose any two adjacent morphisms in a $n+1$-tuple to obtain a n-tuple.

Main Diagram

Staring at this diagram, one finds a few obvious facts, but nothing of great interest. Cool things happen when we put two of these diagrams together. Given a functor $F : \mathsf{C} \to \mathsf{D}$, we can consider:

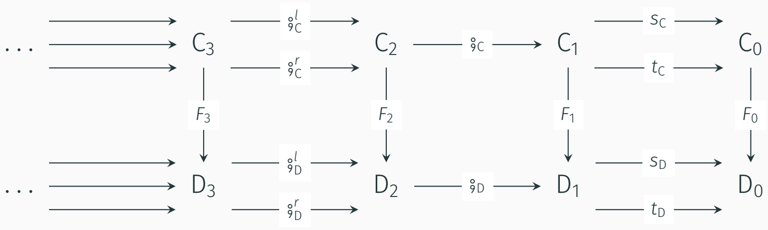


Notice

- F_0 is F defined on objects,

Main Diagram

Staring at this diagram, one finds a few obvious facts, but nothing of great interest. Cool things happen when we put two of these diagrams together. Given a functor $F : \mathsf{C} \to \mathsf{D}$, we can consider:

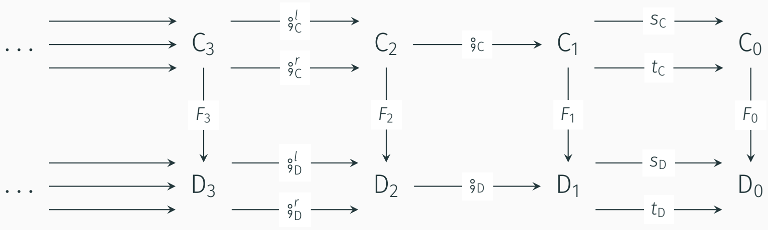


Notice

- F_0 is F defined on objects,
- F_1 is F defined on morphisms,

Main Diagram

Staring at this diagram, one finds a few obvious facts, but nothing of great interest. Cool things happen when we put two of these diagrams together. Given a functor $F : \mathsf{C} \to \mathsf{D}$, we can consider:

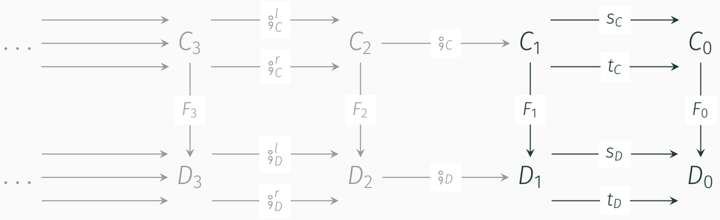


Notice

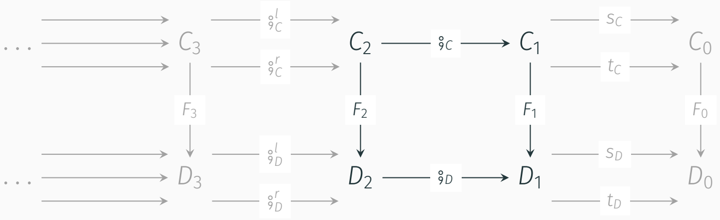
- F_0 is F defined on objects,
- F_1 is F defined on morphisms,
- any other F_n acts on n-tuples by applying F_1 component-wise.

Purpose

The purpose of our paper was to focus on these squares, in particular on the two rightmost ones,



as well as the middle square

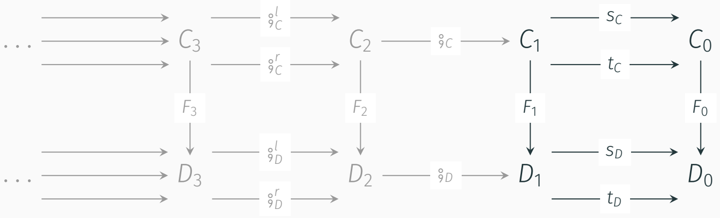


to say things about F.

Table of contents

The Rightmost Squares

We start by focusing on the rightmost square:



Decoupling things, this actually consists of two different squares:

$$\begin{array}{ccc} C_1 & \xrightarrow{s_C} & C_0 \\ \downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\ D_1 & \xrightarrow{s_D} & D_0 \end{array} \qquad \begin{array}{ccc} C_1 & \xrightarrow{t_C} & C_0 \\ \downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\ D_1 & \xrightarrow{t_D} & D_0 \end{array} \tag{1}$$

Simple Observation

$$
\begin{array}{ccc}
\mathsf{C}_1 & \xrightarrow{s_{\mathsf{C}}} & \mathsf{C}_0 \\
\downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\
\mathsf{D}_1 & \xrightarrow{s_{\mathsf{D}}} & \mathsf{D}_0
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathsf{C}_1 & \xrightarrow{t_{\mathsf{C}}} & \mathsf{C}_0 \\
\downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\
\mathsf{D}_1 & \xrightarrow{t_{\mathsf{D}}} & \mathsf{D}_0
\end{array}
$$

- Since functors preserve source and target of morphisms, it follows that if $F : \mathsf{C} \to \mathsf{D}$ is a functor, then the two squares above commute.

Simple Observation

$$
\begin{array}{ccc}
C_1 & \xrightarrow{s_C} & C_0 \\
\downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\
D_1 & \xrightarrow{s_D} & D_0
\end{array}
\qquad\qquad
\begin{array}{ccc}
C_1 & \xrightarrow{t_C} & C_0 \\
\downarrow{\scriptstyle F_1} & & \downarrow{\scriptstyle F_0} \\
D_1 & \xrightarrow{t_D} & D_0
\end{array}
$$

- Since functors preserve source and target of morphisms, it follows that if $F : C \to D$ is a functor, then the two squares above commute.
- But there's more...

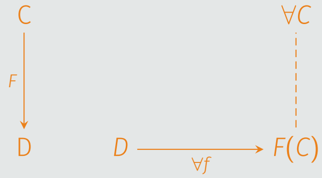
Definition

- A functor $F : \mathsf{C} \to \mathsf{D}$ is a **discrete fibration** if for each $C \in \mathsf{C}$ and $f_{\mathsf{D}} : D \to F_0(C)$, there exists a unique $f_{\mathsf{C}} : C' \to C$ such that $F_1(f_{\mathsf{C}}) = f_{\mathsf{D}}$ (and thus also $F_0(C') = D$).

Definition

Definition

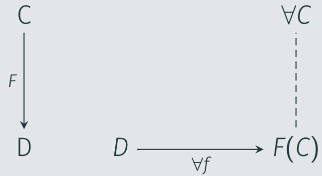
- A functor $F : \mathsf{C} \to \mathsf{D}$ is a **discrete fibration** if for each $C \in \mathsf{C}$ and $f_{\mathsf{D}} : D \to F_0(C)$, there exists a unique $f_{\mathsf{C}} : C' \to C$ such that $F_1(f_{\mathsf{C}}) = f_{\mathsf{D}}$ (and thus also $F_0(C') = D$).
- In a picture,



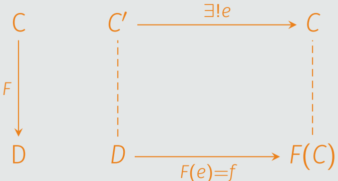
Definition

Definition

- A functor $F : \mathsf{C} \to \mathsf{D}$ is a **discrete fibration** if for each $C \in \mathsf{C}$ and $f_{\mathsf{D}} : D \to F_0(C)$, there exists a unique $f_{\mathsf{C}} : C' \to C$ such that $F_1(f_{\mathsf{C}}) = f_{\mathsf{D}}$ (and thus also $F_0(C') = D$).
- In a picture,



- then there's a unique lifting of f to a morphism above



Definition

Similarly,

Definition

- A functor $F : \mathsf{C} \to \mathsf{D}$ is a **discrete opfibration** if for each $C \in \mathsf{C}_0$ and $f_{\mathsf{D}} : F_0(C) \to D'$, there exists a unique $f_{\mathsf{C}} : C \to C'$ such that $F_1(f_{\mathsf{C}}) = f_{\mathsf{D}}$.

Proposition (giving an "internal" reformulation of the definition)

Proposition 1

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. The right square in Equation (1)

$$\begin{array}{ccc} \mathsf{C}_1 & \xrightarrow{s_{\mathsf{C}}} & \mathsf{C}_0 \\ \downarrow F_1 & & \downarrow F_0 \\ \mathsf{D}_1 & \xrightarrow{s_{\mathsf{D}}} & \mathsf{D}_0 \end{array} \qquad \begin{array}{ccc} \mathsf{C}_1 & \xrightarrow{t_{\mathsf{C}}} & \mathsf{C}_0 \\ \downarrow F_1 & & \downarrow F_0 \\ \mathsf{D}_1 & \xrightarrow{t_{\mathsf{D}}} & \mathsf{D}_0 \end{array}$$

is a pullback square if and only if F is a discrete fibration.

Similarly, the left square is a pullback if and only if F is a discrete opfibration.

Proposition (giving an "internal" reformulation of the definition)

Proposition 1

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. The right square in Equation (1)

$$
\begin{array}{ccc}
\mathsf{C}_1 & \xrightarrow{s_{\mathsf{C}}} & \mathsf{C}_0 \\
{\scriptstyle F_1}\downarrow & & \downarrow{\scriptstyle F_0} \\
\mathsf{D}_1 & \xrightarrow{s_{\mathsf{D}}} & \mathsf{D}_0
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathsf{C}_1 & \xrightarrow{t_{\mathsf{C}}} & \mathsf{C}_0 \\
{\scriptstyle F_1}\downarrow & & \downarrow{\scriptstyle F_0} \\
\mathsf{D}_1 & \xrightarrow{t_{\mathsf{D}}} & \mathsf{D}_0
\end{array}
$$

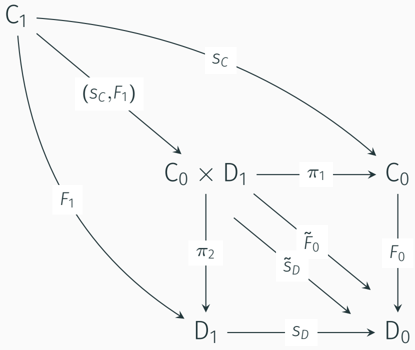
is a pullback square if and only if F is a discrete fibration.

Similarly, the left square is a pullback if and only if F is a discrete opfibration.

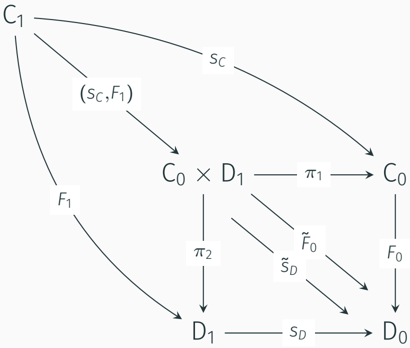
The content of Proposition 1 can be refined. Indeed, one can qualify *how much* F fails to be a discrete (op-)fibration.

Idea

Let's first rewrite the diagrams in Equation (1); focus on leftmost:



Let's first rewrite the diagrams in Equation (1); focus on leftmost:

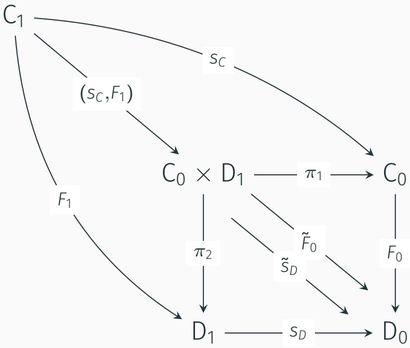


We can thus come to rewrite each of the diagrams in Equation (1) as:

$$\mathrm{C}_1 \xrightarrow{(s_C,F_1)} \mathrm{C}_0 \times \mathrm{D}_1 \underset{\tilde{s}_D}{\overset{\tilde{F}_0}{\rightrightarrows}} \mathrm{D}_0 \qquad \mathrm{C}_1 \xrightarrow{(t_C,F_1)} \mathrm{C}_0 \times \mathrm{D}_1 \underset{\tilde{t}_D}{\overset{\tilde{F}_0}{\rightrightarrows}} \mathrm{D}_0$$

Idea

Let's first rewrite the diagrams in Equation (1); focus on leftmost:



We can thus come to rewrite each of the diagrams in Equation (1) as:

$$C_1 \xrightarrow{(s_C, F_1)} C_0 \times D_1 \underset{\tilde{s}_D}{\overset{\tilde{F}_0}{\rightrightarrows}} D_0 \qquad C_1 \xrightarrow{(t_C, F_1)} C_0 \times D_1 \underset{\tilde{t}_D}{\overset{\tilde{F}_0}{\rightrightarrows}} D_0$$

If F is a functor, these diagrams commute.

Homology Groups

- Now we apply the left adjoint Set $\to$ AbGrp to them.

Homology Groups

- Now we apply the left adjoint Set $\to$ AbGrp to them.
- This means replacing all sets with the abelian groups freely generated by them, and all functions with the corresponding homomorphisms.

Homology Groups

- Now we apply the left adjoint $\mathrm{Set} \to \mathrm{AbGrp}$ to them.
- This means replacing all sets with the abelian groups freely generated by them, and all functions with the corresponding homomorphisms.
- Now homomorphisms can be summed and inverted pointwise, and we write:

$$0 \longrightarrow C_1 \xrightarrow{(s_C, F_1)} C_0 \times D_1 \xrightarrow{\tilde{F}_0 - \tilde{s}_D} D_0 \qquad 0 \longrightarrow C_1 \xrightarrow{(t_C, F_1)} C_0 \times D_1 \xrightarrow{\tilde{F}_0 - \tilde{t}_D} D_0$$

Homology Groups

- Now we apply the left adjoint $\mathrm{Set} \to \mathrm{AbGrp}$ to them.
- This means replacing all sets with the abelian groups freely generated by them, and all functions with the corresponding homomorphisms.
- Now homomorphisms can be summed and inverted pointwise, and we write:

$$0 \longrightarrow \mathrm{C}_1 \xrightarrow{(s_C, F_1)} \mathrm{C}_0 \times \mathrm{D}_1 \xrightarrow{\tilde{F}_0 - \tilde{s}_D} \mathrm{D}_0 \qquad 0 \longrightarrow \mathrm{C}_1 \xrightarrow{(t_C, F_1)} \mathrm{C}_0 \times \mathrm{D}_1 \xrightarrow{\tilde{F}_0 - \tilde{t}_D} \mathrm{D}_0$$

- Since the original diagram commutes, (s_{C}, F_1) equalizes $\tilde{F}_0$ and $\tilde{s}_{\mathrm{D}}$, and so the composition above evaluates to 0, implying $\mathrm{Im}\,(s_{\mathrm{C}}, F_1) \subseteq \ker(\tilde{F}_0 - \tilde{s}_{\mathrm{D}})$.

Homology Groups

- Now we apply the left adjoint Set $\to$ AbGrp to them.
- This means replacing all sets with the abelian groups freely generated by them, and all functions with the corresponding homomorphisms.
- Now homomorphisms can be summed and inverted pointwise, and we write:

$$0 \longrightarrow C_1 \xrightarrow{(s_C, F_1)} C_0 \times D_1 \xrightarrow{\tilde{F}_0 - \tilde{s}_D} D_0 \qquad 0 \longrightarrow C_1 \xrightarrow{(t_C, F_1)} C_0 \times D_1 \xrightarrow{\tilde{F}_0 - \tilde{t}_D} D_0$$

- Since the original diagram commutes, (s_C, F_1) equalizes $\tilde{F}_0$ and $\tilde{s}_D$, and so the composition above evaluates to 0, implying $\mathrm{Im}\,(s_C, F_1) \subseteq \ker(\tilde{F}_0 - \tilde{s}_D)$.
- This places us into homology land, and we define:

Definition (Homology DFib Groups)

$$H^{-1}_{\mathrm{opfib}} := \ker(s_C, F_1) \qquad H^{0}_{\mathrm{opfib}} := \ker(\tilde{F}_0 - \tilde{s}_D)/\,\mathrm{Im}\,(s_C, F_1)$$

$$H^{-1}_{\mathrm{fib}} := \ker(t_C, F_1) \qquad H^{0}_{\mathrm{fib}} := \ker(\tilde{F}_0 - \tilde{t}_D)/\,\mathrm{Im}\,(t_C, F_1).$$

Moral

- These groups provide a qualitative description of the *obstructions* for the diagrams in Equation (1) being a pullback.

Moral

- These groups provide a qualitative description of the *obstructions* for the diagrams in Equation (1) being a pullback.
- When they are trivial there are no obstructions. With Prop 1, get

Proposition 2

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. $H^{-1}_{\mathrm{fib}}, H^{0}_{\mathrm{fib}}$ are trivial if and only if F is a discrete fibration. Similarly, $H^{-1}_{\mathrm{opfib}}, H^{0}_{\mathrm{opfib}}$ are trivial if and only if F is a discrete opfibration.

Moral

- These groups provide a qualitative description of the *obstructions* for the diagrams in Equation (1) being a pullback.
- When they are trivial there are no obstructions. With Prop 1, get

Proposition 2

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. $H^{-1}_{\text{fib}}, H^{0}_{\text{fib}}$ are trivial if and only if F is a discrete fibration. Similarly, $H^{-1}_{\text{opfib}}, H^{0}_{\text{opfib}}$ are trivial if and only if F is a discrete opfibration.

- Proof (take d.o.f. case) is by def of equalizer,

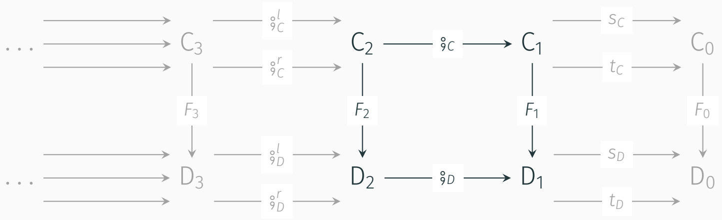
$$
\begin{array}{ccccccc}
0 & \longrightarrow & \mathsf{C}_1 & \xrightarrow{(s_{\mathsf{C}}, F_1)} & \mathsf{C}_0 \times \mathsf{D}_1 & \xrightarrow{\tilde{F}_0 - \tilde{s}_{\mathsf{D}}} & \mathsf{D}_0 \\
 & & & {\scriptstyle k} \searrow & \nearrow & & \\
 & & & Eq(\tilde{F}_0 - \tilde{s}_{\mathsf{D}}) & & &
\end{array}
$$

H^0_{opfib} is trivial iff k surjective; H^{-1}_{opfib} trivial iff k injective. So $H^{-1}_{\text{opfib}}, H^0_{\text{opfib}}$ are trivial iff $\mathsf{C}_1 \simeq \mathrm{Eq}(\tilde{F}_0, \tilde{s}_{\mathsf{D}})$, which by def holds iff the corresponding square in Prop 1 is a pullback iff F is d.o.f.

The Middle Square

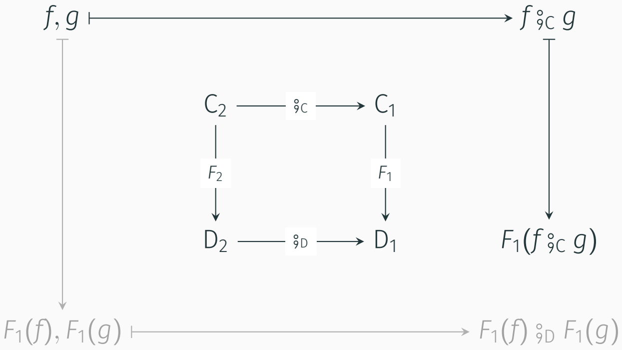
Next Step

The middle square is probably the most interesting one in our endeavor.



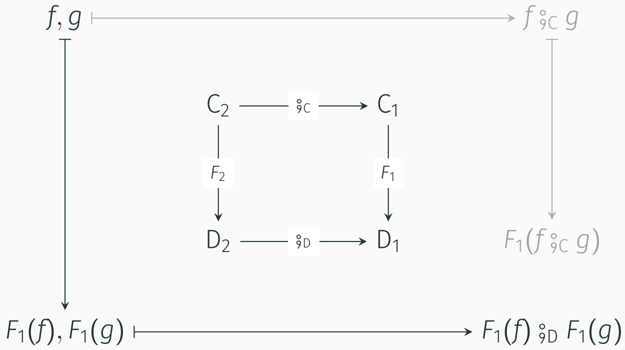
Next Step

- It takes a pair of morphisms $f : A \to B$ and $g : B \to C$ in C_2, and maps them to $F_1(f \mathbin{⨾}_{\mathsf{C}} g)$ (right-down),



Next Step

- It takes a pair of morphisms $f : A \to B$ and $g : B \to C$ in C_2, and maps them to $F_1(f \fatsemi_{\mathsf{C}} g)$ (right-down),
- and to $F_1(f) \fatsemi_{\mathsf{D}} F_1(g)$ (down-right).



Next Step

- Square commutativity states the familiar condition $F(f \fatsemi_{\mathsf{C}} g) = F(f) \fatsemi_{\mathsf{D}} F(g)$, i.e., the preservation of composition by a functor $F : \mathsf{C} \to \mathsf{D}$.

Next Step

- Square commutativity states the familiar condition $F(f \fatsemi_{\mathsf{C}} g) = F(f) \fatsemi_{\mathsf{D}} F(g)$, i.e., the preservation of composition by a functor $F : \mathsf{C} \to \mathsf{D}$.
- More to the point, can also prove

Proposition 3

If either of the squares in Equation (1) is a pullback, then the following is also a pullback:

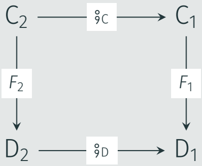
$$
\begin{array}{ccc}
\mathsf{C}_2 & \xrightarrow{\ \fatsemi_{\mathsf{C}}\ } & \mathsf{C}_1 \\
\downarrow{\scriptstyle F_2} & & \downarrow{\scriptstyle F_1} \\
\mathsf{D}_2 & \xrightarrow{\ \fatsemi_{\mathsf{D}}\ } & \mathsf{D}_1
\end{array}
$$

Next Step

- Square commutativity states the familiar condition $F(f \fatsemi_{\mathsf{C}} g) = F(f) \fatsemi_{\mathsf{D}} F(g)$, i.e., the preservation of composition by a functor $F : \mathsf{C} \to \mathsf{D}$.
- More to the point, can also prove

Proposition 3

If either of the squares in Equation (1) is a pullback, then the following is also a pullback:



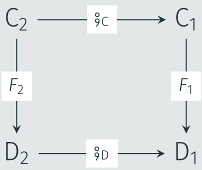
- But what does it mean for the middle square to be a pullback?

Next Step

- Square commutativity states the familiar condition $F(f \fatsemi_{\mathsf{C}} g) = F(f) \fatsemi_{\mathsf{D}} F(g)$, i.e., the preservation of composition by a functor $F : \mathsf{C} \to \mathsf{D}$.
- More to the point, can also prove

Proposition 3

If either of the squares in Equation (1) is a pullback, then the following is also a pullback:



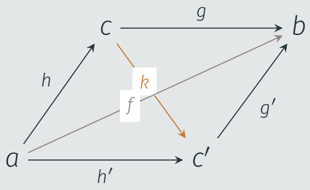
- But what does it mean for the middle square to be a pullback?
- As with discrete (op-)fibrations, this corresponds to a well-known concept.

Conduché Fibration

Definition

A functor $F : \mathsf{C} \to \mathsf{D}$ is a **Conduché functor** if for $f : a \to b$ in C_1 and any factorization $F_0(a) \xrightarrow{v} d \xrightarrow{u} F_0(b)$ of $F_1(f) : F_0(a) \to F_0(b)$ in D:

- There exists a factorization $a \xrightarrow{h} c \xrightarrow{g} b$ of $f : a \to b$ in C such that $F_1(g) = u$ and $F_1(h) = v$.
- Such a factorization is unique up to equalivalence, i.e., any two such factorizations $g \circ h = g' \circ h'$ in C are equivalent if there exists a $k : \mathrm{cod}\ h \to \mathrm{dom}\ g'$ such that $k \circ h = h'$ and $g' \circ k = g$



and $F(k)$ is an identity morphism.

Proposition connecting middle square and DCFs

- Moreover, a Conduché functor is **discrete** if each factorisation is unique, i.e., the lifting of a factorization is unique ‘on the nose’ (not just up to equivalence).

Proposition connecting middle square and DCFs

- Moreover, a Conduché functor is **discrete** if each factorisation is unique, i.e., the lifting of a factorization is unique ‘on the nose’ (not just up to equivalence).
- In a way very similar to Prop 1, we can prove the following:

Proposition 4

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. The middle square is a pullback square if and only if F is a discrete Conduché functor.

Proposition connecting middle square and DCFs

- Moreover, a Conduché functor is **discrete** if each factorisation is unique, i.e., the lifting of a factorization is unique 'on the nose' (not just up to equivalence).
- In a way very similar to Prop 1, we can prove the following:

Proposition 4

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. The middle square is a pullback square if and only if F is a discrete Conduché functor.

- Putting together Prop 1, Prop 3, and Prop 4, we recover that

Discrete Conduché fibrations include all discrete (op-)fibrations

every discrete (op-)fibration is also a discrete Conduché functor.

Homology

- As earlier with (op)fibrations, when the middle square is not a pullback we want to investigate *how far F* is from being a d.c.f.

Homology

- As earlier with (op)fibrations, when the middle square is not a pullback we want to investigate *how far F* is from being a d.c.f.
- $$0 \longrightarrow C_2 \xrightarrow{(\fatsemi_C, F_2)} C_1 \times D_2 \xrightarrow{\tilde{F}_1 - \tilde{\fatsemi}_D} D_1$$

  again commutes, implying that $\mathrm{Im}(\fatsemi_C, F_2) \subseteq \ker(\tilde{F}_1 - \tilde{\fatsemi}_D)$, and so

Definition (Discrete Cond Fib Homology Groups)

$$H^{-1}_{\mathrm{Cond}} := \ker(\fatsemi_C, F_2) \qquad H^{0}_{\mathrm{Cond}} := \ker(\tilde{F}_1 - \tilde{\fatsemi}_D) / \mathrm{Im}(\fatsemi_C, F_2)$$

Homology

- As earlier with (op)fibrations, when the middle square is not a pullback we want to investigate *how far* F is from being a d.c.f.
-
$$0 \longrightarrow \mathsf{C}_2 \xrightarrow{(\fatsemi_{\mathsf{C}}, F_2)} \mathsf{C}_1 \times \mathsf{D}_2 \xrightarrow{\tilde{F}_1 - \tilde{\fatsemi}_{\mathsf{D}}} \mathsf{D}_1$$
again commutes, implying that $\operatorname{Im}(\fatsemi_{\mathsf{C}}, F_2) \subseteq \ker(\tilde{F}_1 - \tilde{\fatsemi}_{\mathsf{D}})$, and so

Definition (Discrete Cond Fib Homology Groups)

$$H^{-1}_{\mathrm{Cond}} := \ker(\fatsemi_{\mathsf{C}}, F_2) \qquad H^{0}_{\mathrm{Cond}} := \ker(\tilde{F}_1 - \tilde{\fatsemi}_{\mathsf{D}}) / \operatorname{Im}(\fatsemi_{\mathsf{C}}, F_2)$$

- In a way similar to Proposition 2, can prove the following:

Proposition 5

Let $F : \mathsf{C} \to \mathsf{D}$ be a functor. $H^{-1}_{\mathrm{Cond}}, H^{0}_{\mathrm{Cond}}$ are trivial if and only if F is a discrete Conduché functor.

Homology

- As earlier with (op)fibrations, when the middle square is not a pullback we want to investigate *how far F* is from being a d.c.f.
-
$$0 \longrightarrow C_2 \xrightarrow{(\fatsemi_C, F_2)} C_1 \times D_2 \xrightarrow{\tilde{F}_1 - \tilde{\fatsemi}_D} D_1$$
again commutes, implying that $\mathrm{Im}(\fatsemi_C, F_2) \subseteq \ker(\tilde{F}_1 - \tilde{\fatsemi}_D)$, and so

Definition (Discrete Cond Fib Homology Groups)

$$H^{-1}_{\mathrm{Cond}} := \ker(\fatsemi_C, F_2) \qquad H^{0}_{\mathrm{Cond}} := \ker(\tilde{F}_1 - \tilde{\fatsemi}_D) / \mathrm{Im}(\fatsemi_C, F_2)$$

- In a way similar to Proposition 2, can prove the following:

Proposition 5

Let $F : C \to D$ be a functor. $H^{-1}_{\mathrm{Cond}}, H^{0}_{\mathrm{Cond}}$ are trivial if and only if F is a discrete Conduché functor.

- Such groups thus essentially give us a way of measuring obstructions to being a discrete Conduché functor.

Applications: Decorated Petri Nets, Delta Lenses

First Application

- As suggested by the opening slides, one application comes in the context of categorical semantics for Petri nets.

First Application

- As suggested by the opening slides, one application comes in the context of categorical semantics for Petri nets.
- To apply our ideas, let's be a bit more precise about how to think about Petri nets and their semantics categorically.

First Application

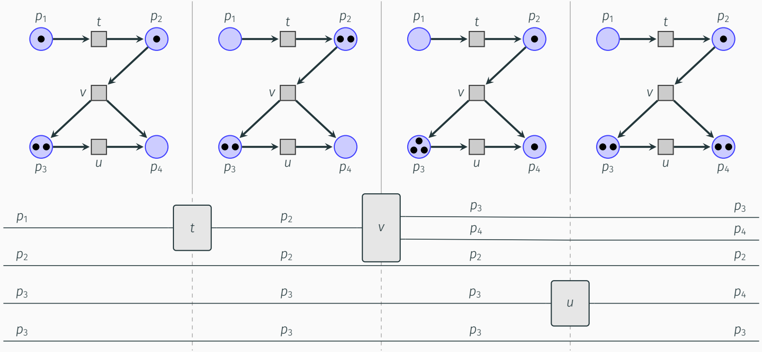
- As suggested by the opening slides, one application comes in the context of categorical semantics for Petri nets.
- To apply our ideas, let's be a bit more precise about how to think about Petri nets and their semantics categorically.
- It is well-known that the (different kinds of) Petri net freely generates a monoidal category (of an appropriate sort), by
  - using its places to generate a monoid of objects,
  - using each transition as a generating morphism, with domain and codomain the monoidal product of its input/output places.

First Application

- As suggested by the opening slides, one application comes in the context of categorical semantics for Petri nets.
- To apply our ideas, let's be a bit more precise about how to think about Petri nets and their semantics categorically.
- It is well-known that the (different kinds of) Petri net freely generates a monoidal category (of an appropriate sort), by
  - using its places to generate a monoid of objects,
  - using each transition as a generating morphism, with domain and codomain the monoidal product of its input/output places.
- A marking of the net (a placement of tokens throughout the net) then corresponds to an object in the monoidal category, and an execution or firing sequence (a sequence of transitions carrying markings to other markings) corresponds to a morphism.

In a Picture

Here's a picture helping to visualize this, where we make use of wiring diagrams to display morphisms of the monoidal category (corresponding to executions of the net):



Categorical Approach in Detail

Given a Petri net N, we can generate a *free symmetric strict monoidal category*, $\mathfrak{F}(N)$, the category of executions of N:

- The free monoid of objects is $P^{\otimes}$, the set of strings generated by P (places of the net), with unit the empty string, monoidal product, denoted $p \otimes p'$, given by string concatenation.

Categorical Approach in Detail

Given a Petri net N, we can generate a *free symmetric strict monoidal category*, $\mathfrak{F}(N)$, the category of executions of N:

- The free monoid of objects is $P^{\otimes}$, the set of strings generated by P (places of the net), with unit the empty string, monoidal product, denoted $p \otimes p'$, given by string concatenation.
- Morphisms are generated by the set of transitions T: each $u \in T$ corresponds to a morphism generator $s(u) \xrightarrow{u} t(u)$, where $s(u), t(u)$ are obtained by choosing some ordering on their multisets; morphisms obtained by all formal horizontal and vertical compositions of generators, identities and symmetries.

Categorical Approach in Detail

Given a Petri net N, we can generate a *free symmetric strict monoidal category*, $\mathfrak{F}(N)$, the category of executions of N:

- The free monoid of objects is $P^{\otimes}$, the set of strings generated by P (places of the net), with unit the empty string, monoidal product, denoted $p \otimes p'$, given by string concatenation.
- Morphisms are generated by the set of transitions T: each $u \in T$ corresponds to a morphism generator $s(u) \xrightarrow{u} t(u)$, where $s(u), t(u)$ are obtained by choosing some ordering on their multisets; morphisms obtained by all formal horizontal and vertical compositions of generators, identities and symmetries.

Similarly, given a Petri net N, we can generate a *free commutative strict monoidal category* $\mathfrak{C}(N)$ by considering the set of multisets $P^{\oplus}$ as the free commutative monoid of objects, empty multiset as unit and multiset sum as the multiplication.

Petri Net

- The correspondence between Petri nets and free strict symmetric monoidal categories — mapping a net to its possible executions — supplies a *process semantics* for a net. The semantics $\mathfrak{F}(N)$ distinguishes between tokens living in the same place, $\mathfrak{C}(N)$ doesn't.

Petri Net

- The correspondence between Petri nets and free strict symmetric monoidal categories — mapping a net to its possible executions — supplies a *process semantics* for a net. The semantics $\mathfrak{F}(N)$ distinguishes between tokens living in the same place, $\mathfrak{C}(N)$ doesn't.
- Part of the utility of the execution semantics is that it allows us to interface with other structures using monoidal functors.

Petri Net

- The correspondence between Petri nets and free strict symmetric monoidal categories — mapping a net to its possible executions — supplies a *process semantics* for a net. The semantics $\mathfrak{F}(N)$ distinguishes between tokens living in the same place, $\mathfrak{C}(N)$ doesn't.
- Part of the utility of the execution semantics is that it allows us to interface with other structures using monoidal functors.
- Categorically, extensions (as found with guarded/colored nets and bounded nets) are described by endowing a net N with some sort of functor $\mathfrak{F}(N) \to \mathsf{D}$ or, depending on the choice of token philosophy, $\mathfrak{C}(N) \to \mathsf{D}$. (In practice, D is often taken to be Span, the functor usually denoted $N^{\sharp}$.)

Petri Net

- The correspondence between Petri nets and free strict symmetric monoidal categories — mapping a net to its possible executions — supplies a *process semantics* for a net. The semantics $\mathfrak{F}(N)$ distinguishes between tokens living in the same place, $\mathfrak{C}(N)$ doesn't.
- Part of the utility of the execution semantics is that it allows us to interface with other structures using monoidal functors.
- Categorically, extensions (as found with guarded/colored nets and bounded nets) are described by endowing a net N with some sort of functor $\mathfrak{F}(N) \to \mathsf{D}$ or, depending on the choice of token philosophy, $\mathfrak{C}(N) \to \mathsf{D}$. (In practice, D is often taken to be Span, the functor usually denoted $N^\sharp$.)
- **Guarded nets** (with side effects) can be described as nets endowed with strict monoidal functors $\mathfrak{F}(N) \to \mathsf{Span}$; **bounded nets** as lax-monoidal-lax functors $\mathfrak{C}(N) \to \mathsf{Span}$.

Applying our ideas

And now we start to arrive at how we can make our earlier results bear on such things...

Applying our ideas

And now we start to arrive at how we can make our earlier results bear on such things...

To apply our results to Petri nets, we need to use the following fact:

Lemma

[Due to Bénabou] Fix B. Any lax double functor $\mathbb{B} \xrightarrow{F} \mathsf{Span}(\mathsf{Set})$ is a pseudofunctor if and only if the projection $\int F \xrightarrow{\pi_F} \mathsf{B}$ of its Grothendieck construction is a discrete Conduché functor.

Applying our ideas

And now we start to arrive at how we can make our earlier results bear on such things...

To apply our results to Petri nets, we need to use the following fact:

Lemma

[Due to Bénabou] Fix B. Any lax double functor $\mathbb{B} \xrightarrow{F} \mathrm{Span}(\mathrm{Set})$ is a pseudofunctor if and only if the projection $\int F \xrightarrow{\pi_F} \mathrm{B}$ of its Grothendieck construction is a discrete Conduché functor.

Corollary

For any lax double functor $\mathbb{B} \xrightarrow{F} \mathrm{Span}(\mathrm{Set})$, the groups

$$H^{-1}_{cond}\left(\int F \xrightarrow{\pi_F} \mathrm{B}\right) \qquad H^{0}_{cond}\left(\int F \xrightarrow{\pi_F} \mathrm{B}\right)$$

measure obstructions to pseudofunctoriality of F.

Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathrm{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathrm{Span}$, for a Petri net N is from being pseudo.

Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathsf{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathsf{Span}$, for a Petri net N is from being pseudo.
- We can do this by examining the homology groups of the associated functor, e.g., $\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)$.

Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathrm{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathrm{Span}$, for a Petri net N is from being pseudo.
- We can do this by examining the homology groups of the associated functor, e.g., $\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)$.
- What meaning does this have from the Petri net point of view?

Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathsf{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathsf{Span}$, for a Petri net N is from being pseudo.
- We can do this by examining the homology groups of the associated functor, e.g., $\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)$.
- What meaning does this have from the Petri net point of view?
- To see this, we'll want to internalize the guard semantics in the free category $\mathfrak{F}(N)$ associated to a net N, where this means we associate a traditional unguarded Petri net M such that it has the same executions.

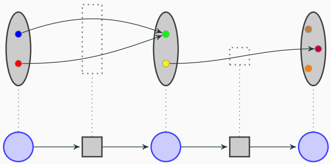
Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathsf{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathsf{Span}$, for a Petri net N is from being pseudo.
- We can do this by examining the homology groups of the associated functor, e.g., $\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)$.
- What meaning does this have from the Petri net point of view?
- To see this, we'll want to internalize the guard semantics in the free category $\mathfrak{F}(N)$ associated to a net N, where this means we associate a traditional unguarded Petri net M such that it has the same executions.
- Given a net N and a functor $N^\sharp$, the process of internalization is described by taking its Grothendieck construction $\int N^\sharp$.

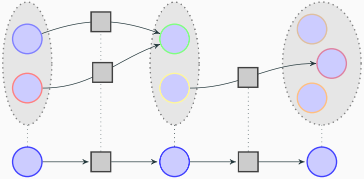
Applied

- Thanks to this lemma and corollary, we can measure *how far* a categorical decoration $N^\sharp : \mathfrak{F}(N) \to \mathsf{Span}$, or $N^\sharp : \mathfrak{C}(N) \to \mathsf{Span}$, for a Petri net N is from being pseudo.
- We can do this by examining the homology groups of the associated functor, e.g., $\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)$.
- What meaning does this have from the Petri net point of view?
- To see this, we'll want to internalize the guard semantics in the free category $\mathfrak{F}(N)$ associated to a net N, where this means we associate a traditional unguarded Petri net M such that it has the same executions.
- Given a net N and a functor $N^\sharp$, the process of internalization is described by taking its Grothendieck construction $\int N^\sharp$.
- The Grothendieck construction gives us a way to internalize span semantics to nets. In practice, it means promoting token colors to places and arcs between token colors to transitions.

Take the following guarded net (simplifying our one from the beginning):



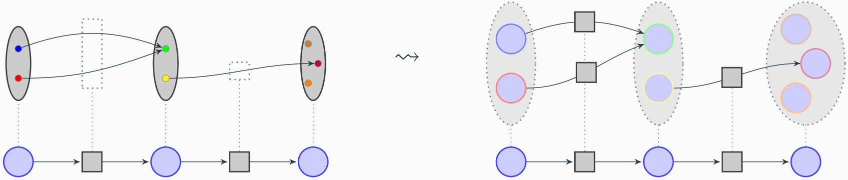
$\rightsquigarrow$



- **Picture on the left:** We decorate a base net N with token colors and transition guards by defining a strict monoidal functor $N^\sharp : \mathfrak{F}(N) \to \mathbf{Span}$.

Example summing things up

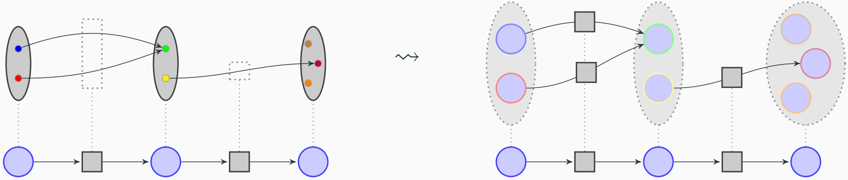
Take the following guarded net (simplifying our one from the beginning):



- **Picture on the left:** We decorate a base net N with token colors and transition guards by defining a strict monoidal functor $N^\sharp : \mathfrak{F}(N) \to \text{Span}$.
- Since $\mathfrak{F}(N)$ is freely generated, this amounts to mapping each place to a set of colors and each transition to a span, representing how colors are correlated by transitions.

Example summing things up

Take the following guarded net (simplifying our one from the beginning):



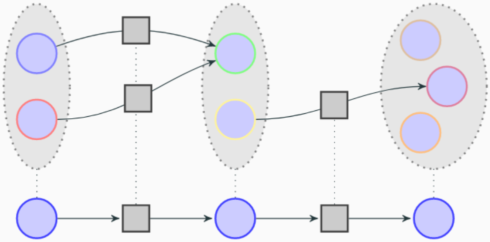
- **Picture on the left:** We decorate a base net N with token colors and transition guards by defining a strict monoidal functor $N^\sharp : \mathfrak{F}(N) \to \text{Span}$.
- Since $\mathfrak{F}(N)$ is freely generated, this amounts to mapping each place to a set of colors and each transition to a span, representing how colors are correlated by transitions.
- **Picture on the right:** Internalizing, we obtain a net M such that $\int N^\sharp = \mathfrak{F}(M)$ (see net on top). In practice, we promote token colors and arcs in the left picture to places and transitions.

Take-Away

- The functor $N^\sharp : \mathfrak{F}(N) \to \mathsf{Span}$ for guarded nets is always strict, and so the associated functor $\pi_{N^\sharp} : \int N^\sharp \to \mathfrak{F}(N)$ is always discrete Conduché.

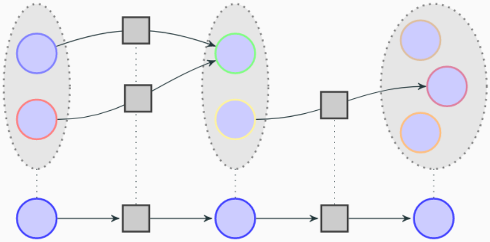
Take-Away

- The functor $N^\sharp : \mathfrak{F}(N) \to \mathbf{Span}$ for guarded nets is always strict, and so the associated functor $\pi_{N^\sharp} : \int N^\sharp \to \mathfrak{F}(N)$ is always discrete Conduché.
- However, $\pi_{N^\sharp} : \int N^\sharp \to \mathfrak{F}(N)$ is *not* always a discrete fibration.



Take-Away

- The functor $N^\sharp : \mathfrak{F}(N) \to \mathbf{Span}$ for guarded nets is always strict, and so the associated functor $\pi_{N^\sharp} : \int N^\sharp \to \mathfrak{F}(N)$ is always discrete Conduché.
- However, $\pi_{N^\sharp} : \int N^\sharp \to \mathfrak{F}(N)$ is *not* always a discrete fibration.



- Take for instance the leftmost transition in the underlying net and focus on the yellow place in $\int N^\sharp$ above: The discrete fibration condition requires a transition leading into it, but we have none (failure of existence). In the case of the green place, we have more than one (failure of uniqueness).

Take-Away

- The green-circled place would be picked up as a non-trivial element of the homology group $H^{-1}_{fib}\left(\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)\right)$.

Take-Away

- The green-circled place would be picked up as a non-trivial element of the homology group $H^{-1}_{fib}\left(\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)\right)$.
- The yellow-circled place would be picked up as a non-trivial element of the homology group $H^{0}_{fib}\left(\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{F}(N)\right)$.

Take-Away

- The green-circled place would be picked up as a non-trivial element of the homology group $H^{-1}_{fib}\left(\int N^{\sharp} \xrightarrow{\pi_{N^{\sharp}}} \mathfrak{F}(N)\right)$.
- The yellow-circled place would be picked up as a non-trivial element of the homology group $H^{0}_{fib}\left(\int N^{\sharp} \xrightarrow{\pi_{N^{\sharp}}} \mathfrak{F}(N)\right)$.
- Altogether, the homology groups give us qualitative information about which token colors are ‘problematic’ with respect to the fibration condition failing.

Bounded Nets

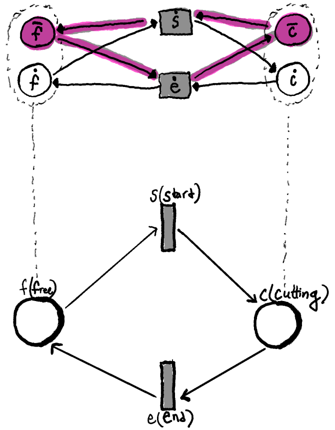
Another interesting case is that of bounded nets.

- Our semantics here is a *lax-monoidal-lax* functor $N^\sharp : \mathfrak{C}(N) \to \text{Span}$.

Bounded Nets

Another interesting case is that of bounded nets.

- Our semantics here is a *lax-monoidal-lax* functor $N^\sharp : \mathfrak{C}(N) \to \mathbf{Span}$.
- This semantics is again internalizable, and so $\int N^\sharp = \mathfrak{F}(M)$, as in:

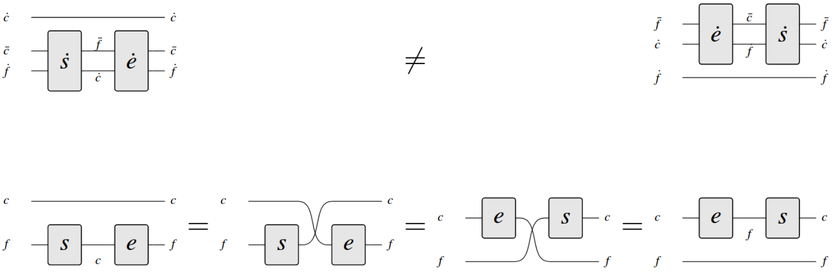


Application

Since our functor $N^\sharp$ is only lax, the Lemma tells us that the functor $\int N^\sharp \to \mathfrak{C}(N)$ is not always Conduché.

Application

Since our functor $N^\sharp$ is only lax, the Lemma tells us that the functor $\int N^\sharp \to \mathfrak{C}(N)$ is not always Conduché.



For the base net N, we have a particular execution, which can be written down in two equivalent ways because we are using the commutative semantics. This execution, seen as a morphism of $\mathfrak{C}(N)$, corresponds to two different executions of M, which are in turn morphisms in $\int N^\sharp = \mathfrak{C}(M)$.

Observations

- The reason why this equality is not lifted to $\int N^\sharp (= \mathfrak{C}(M))$ (see the picture up top) is that in M all the places of N are doubled, so whereas we could exchange any place c with itself in $\mathfrak{C}(N)$, we cannot do the same in $\mathfrak{C}(M)$, as $\dot{c}$ and $\tilde{c}$, the place and antiplace corresponding to c, are considered as different generators and cannot be swapped.

Observations

- The reason why this equality is not lifted to $\int N^\sharp (= \mathfrak{C}(M))$ (see the picture up top) is that in M all the places of N are doubled, so whereas we could exchange any place c with itself in $\mathfrak{C}(N)$, we cannot do the same in $\mathfrak{C}(M)$, as $\dot{c}$ and $\tilde{c}$, the place and antiplace corresponding to c, are considered as different generators and cannot be swapped.
- In this case, the obstructions to being discrete Conduché witness histories that should ‘morally be identified’ in the category $\mathfrak{C}(M)$ of executions of the bounded net M, but are not.

Observations

- The reason why this equality is not lifted to $\int N^\sharp (= \mathfrak{C}(M))$ (see the picture up top) is that in M all the places of N are doubled, so whereas we could exchange any place c with itself in $\mathfrak{C}(N)$, we cannot do the same in $\mathfrak{C}(M)$, as $\dot{c}$ and $\tilde{c}$, the place and antiplace corresponding to c, are considered as different generators and cannot be swapped.
- In this case, the obstructions to being discrete Conduché witness histories that should 'morally be identified' in the category $\mathfrak{C}(M)$ of executions of the bounded net M, but are not.
- The homology groups give us qualitative information about these obstructions — again to existence and uniqueness.

Observations

- The reason why this equality is not lifted to $\int N^\sharp (= \mathfrak{C}(M))$ (see the picture up top) is that in M all the places of N are doubled, so whereas we could exchange any place c with itself in $\mathfrak{C}(N)$, we cannot do the same in $\mathfrak{C}(M)$, as $\dot{c}$ and $\tilde{c}$, the place and antiplace corresponding to c, are considered as different generators and cannot be swapped.
- In this case, the obstructions to being discrete Conduché witness histories that should 'morally be identified' in the category $\mathfrak{C}(M)$ of executions of the bounded net M, but are not.
- The homology groups give us qualitative information about these obstructions — again to existence and uniqueness.
- In particular, the execution described for the example figure above would be picked up as a non-trivial element of the homology group $H^{-1}_{cond}\left(\int N^\sharp \xrightarrow{\pi_{N^\sharp}} \mathfrak{C}(N)\right)$.

Delta Lenses

- We (briefly) consider one last application: to lenses.

Delta Lenses

- We (briefly) consider one last application: to lenses.
- Recall that a lens is made of two parts:
    - One is responsible for accessing a given part from an object constituting the whole. We call this part `get`.
    - The other, is responsible of pushing an update of the accessed part to the whole, and we call it `put`.

Delta Lenses

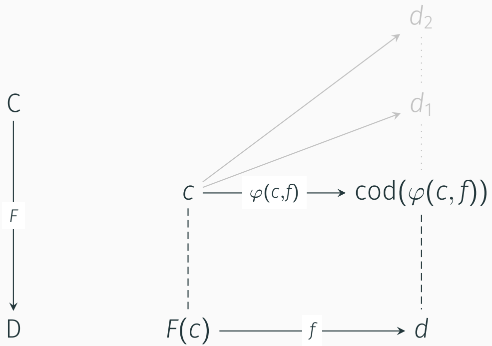
- We (briefly) consider one last application: to lenses.
- Recall that a lens is made of two parts:
    - One is responsible for accessing a given part from an object constituting the whole. We call this part *get*.
    - The other, is responsible of pushing an update of the accessed part to the whole, and we call it *put*.
- This concept has been formalized in a broad variety of ways; the formalization we are most interested in is called delta lenses.

Delta Lenses

- We (briefly) consider one last application: to lenses.
- Recall that a lens is made of two parts:
  - One is responsible for accessing a given part from an object constituting the whole. We call this part ***get***.
  - The other, is responsible of pushing an update of the accessed part to the whole, and we call it ***put***.
- This concept has been formalized in a broad variety of ways; the formalization we are most interested in is called delta lenses.
- Instead of defining a delta lens, let's just note that in [0], it's pointed out how delta lenses generalize the concept of opfibration: φ guarantees that, for each object $c \in \mathsf{C}$ and morphism $F(c) \xrightarrow{f} d \in \mathsf{D}$, we get a corresponding lift in C.

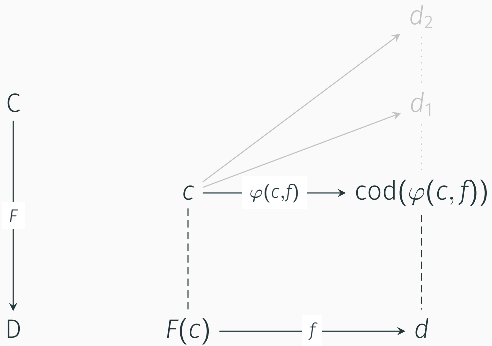
More

- Yet, this lift may not be guaranteed to be unique. Indeed, we can precisely think of a delta lens as a functor $F : \mathsf{C} \to \mathsf{D}$ with a *chosen lift* for each morphism of D.



More

- Yet, this lift may not be guaranteed to be unique. Indeed, we can precisely think of a delta lens as a functor $F : \mathsf{C} \to \mathsf{D}$ with a *chosen lift* for each morphism of D.



- Delta lenses have been heavily studied; in [0], we find a refinement of our Lemma, identifying delta lenses over D with lax double functors $\mathbb{D} \to \mathsf{Span}$ that factorise in a particular way.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.
- A delta lens $(F, \varphi) : \mathsf{C} \to \mathsf{D}$ always provides lifts for all morphisms in D and objects of C. As such, $H^0_{\text{opfib}}(F, \varphi)$ is always trivial.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.
- A delta lens $(F, \varphi) : \mathsf{C} \to \mathsf{D}$ always provides lifts for all morphisms in D and objects of C. As such, $H^0_{\text{opfib}}(F, \varphi)$ is always trivial.
- On the other hand, $H^{-1}_{\text{opfib}}(F, \varphi)$ is generally not-trivial, and measures how far we are from having a unique lift operation $\varphi(-, -)$.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.
- A delta lens $(F, \varphi) : \mathsf{C} \to \mathsf{D}$ always provides lifts for all morphisms in D and objects of C. As such, $H^0_{\text{opfib}}(F, \varphi)$ is always trivial.
- On the other hand, $H^{-1}_{\text{opfib}}(F, \varphi)$ is generally not-trivial, and measures how far we are from having a unique lift operation $\varphi(-, -)$.
- From an applicative point of view, a delta lens that is also a discrete opfibration is a very rigid structure: it tells us that there is only one way to push an update of the part to the whole.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.
- A delta lens $(F, \varphi) : \mathsf{C} \to \mathsf{D}$ always provides lifts for all morphisms in D and objects of C. As such, $H^0_{\text{opfib}}(F, \varphi)$ is always trivial.
- On the other hand, $H^{-1}_{\text{opfib}}(F, \varphi)$ is generally not-trivial, and measures how far we are from having a unique lift operation $\varphi(-, -)$.
- From an applicative point of view, a delta lens that is also a discrete opfibration is a very rigid structure: it tells us that there is only one way to push an update of the part to the whole.
- In other words: also being a discrete opfibration means the structure describing the way a part is transformed canonically induces a transformation structure for the whole.

More

- By applying the techniques developed in our work, we can make a modest contribution, measuring *how much a given delta lens fails to be a discrete opfibration*.
- A delta lens $(F, \varphi) : \mathsf{C} \to \mathsf{D}$ always provides lifts for all morphisms in D and objects of C. As such, $H^0_{\text{opfib}}(F, \varphi)$ is always trivial.
- On the other hand, $H^{-1}_{\text{opfib}}(F, \varphi)$ is generally not-trivial, and measures how far we are from having a unique lift operation $\varphi(-, -)$.
- From an applicative point of view, a delta lens that is also a discrete opfibration is a very rigid structure: it tells us that there is only one way to push an update of the part to the whole.
- In other words: also being a discrete opfibration means the structure describing the way a part is transformed canonically induces a transformation structure for the whole.
- It is thus a very sensible thing to want to consider obstructions to delta lenses being a discrete opfibration.

Discussion and Future Work

Discussion

- Investigating failures of compositionality in category theory transcends mere theoretical relevance and has the potential of impacting several real-world applications.

Discussion

- Investigating failures of compositionality in category theory transcends mere theoretical relevance and has the potential of impacting several real-world applications.
- With many modern ML systems being inherently compositional and a growing scientific community focusing on their categorical formalization, one is predictably interested in looking at where such compositionality fails.

Discussion

- Investigating failures of compositionality in category theory transcends mere theoretical relevance and has the potential of impacting several real-world applications.
- With many modern ML systems being inherently compositional and a growing scientific community focusing on their categorical formalization, one is predictably interested in looking at where such compositionality fails.
- Formal techniques to qualify such obstructions may hold promise in improving understanding of ML's inner workings.

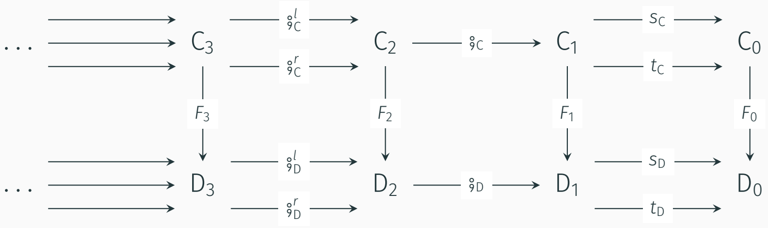
Discussion

- Investigating failures of compositionality in category theory transcends mere theoretical relevance and has the potential of impacting several real-world applications.
- With many modern ML systems being inherently compositional and a growing scientific community focusing on their categorical formalization, one is predictably interested in looking at where such compositionality fails.
- Formal techniques to qualify such obstructions may hold promise in improving understanding of ML's inner workings.
- Applying the techniques heretofore presented to the fields of categorical machine learning and cybernetics constitutes one of the main directions for future work.

Another direction

- Another important — and perhaps more obvious — direction of future work consists in asking the obvious question: ‘what about the other squares to the left?’

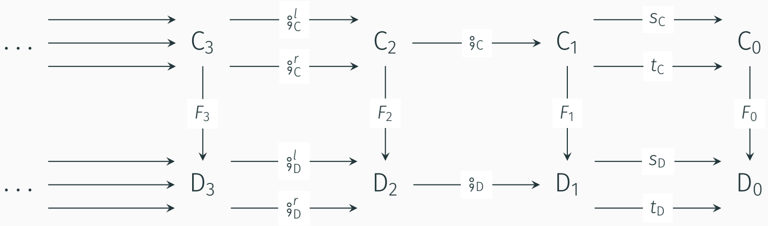
- Another important — and perhaps more obvious — direction of future work consists in asking the obvious question: ‘what about the other squares to the left?’
- We conjecture that, by looking at the diagram



the next square proceeding to the left can be used to measure the laxity of a given functor without necessarily going through the Corollary from earlier.

Another direction

- Another important — and perhaps more obvious — direction of future work consists in asking the obvious question: ‘what about the other squares to the left?’
- We conjecture that, by looking at the diagram



  the next square proceeding to the left can be used to measure the laxity of a given functor without necessarily going through the Corollary from earlier.
- There should be a way to produce some homology group measuring laxness of F directly, but this has proven to be an elusive task so far.

Thanks for listening! Questions?