

DIAGRAMS FOR EFFECTFUL CATEGORIES (& PREMONOIDAL)

MARIO ROMÁN

JUNE 20st, 2022
ACT 22@ Strathclyde.

TALLINN UNIVERSITY OF TECHNOLOGY

Supported by the EU Estonian IT Academy Research Measure.



PREMONOIDAL CATEGORIES

DEFINITION (Power, Robinson). A **binoidal** category $(\mathcal{C}, \otimes, I)$, is a category with an object $I \in \mathcal{C}$ and an assignment on objects that is separately functorial on each component, $(A \otimes \cdot): \mathcal{C} \rightarrow \mathcal{C}$ and $(\cdot \otimes B): \mathcal{C} \rightarrow \mathcal{C}$.

- That is, $(\otimes)$ is a sesquifunctor.
- A morphism $f: A \rightarrow B$ is central if $(f \otimes \text{id}) \circ (\text{id} \otimes g) = (\text{id} \otimes g) \circ (f \otimes \text{id})$ for any $g: A' \rightarrow B'$.

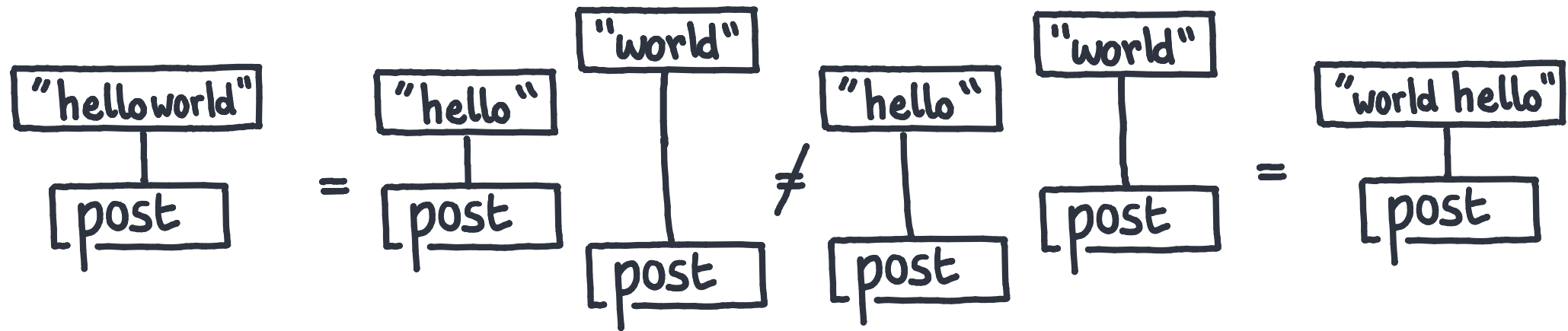
DEFINITION (Power, Robinson). A **premonoidal** category $(\mathcal{C}, \otimes, I)$ is a binoidal category with natural central isomorphisms

$$\alpha_{A,B,C}: (A \otimes B) \otimes C \rightarrow A \otimes (B \otimes C), \quad \rho_A: A \otimes I \rightarrow A, \quad \lambda_A: I \otimes A \rightarrow A,$$

satisfying pentagon and triangle equations. It is **strict** if these are identities.

PREMONOIDAL CATEGORIES

Premonoidal categories are monoidal categories without the interchange law.



PROPOSITION. The free algebras of a strong monad form a premonoidal category.

EXAMPLES. Processes with global state, e.g. $(\Sigma^* \times \cdot) : \text{SET} \rightarrow \text{SET}$.



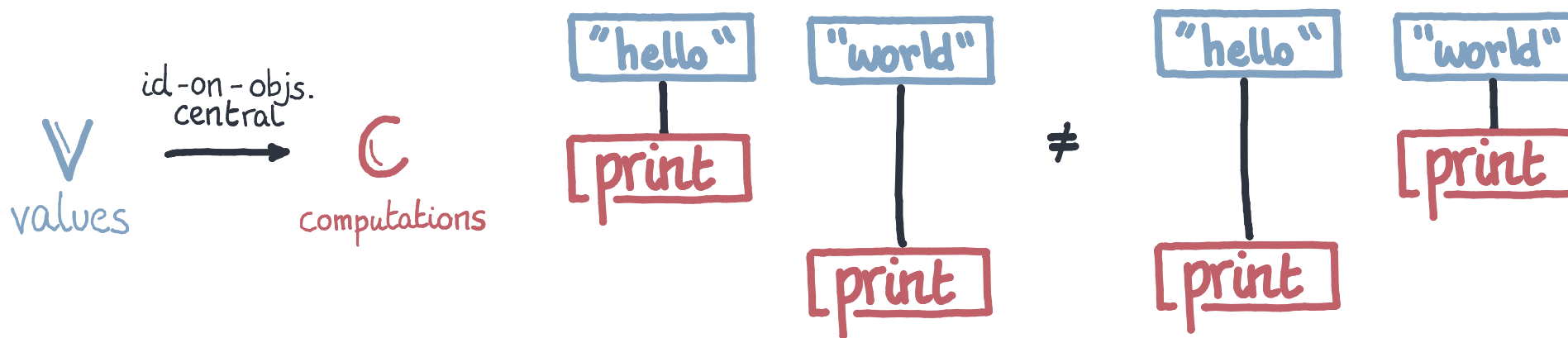
Power, Robinson, 97 ; Power, Thielecke, 99.

EFFECTFUL CATEGORIES

PROBLEM. There is not a good notion of **premonoidal** functor. We need premonoidals "with a chosen center".

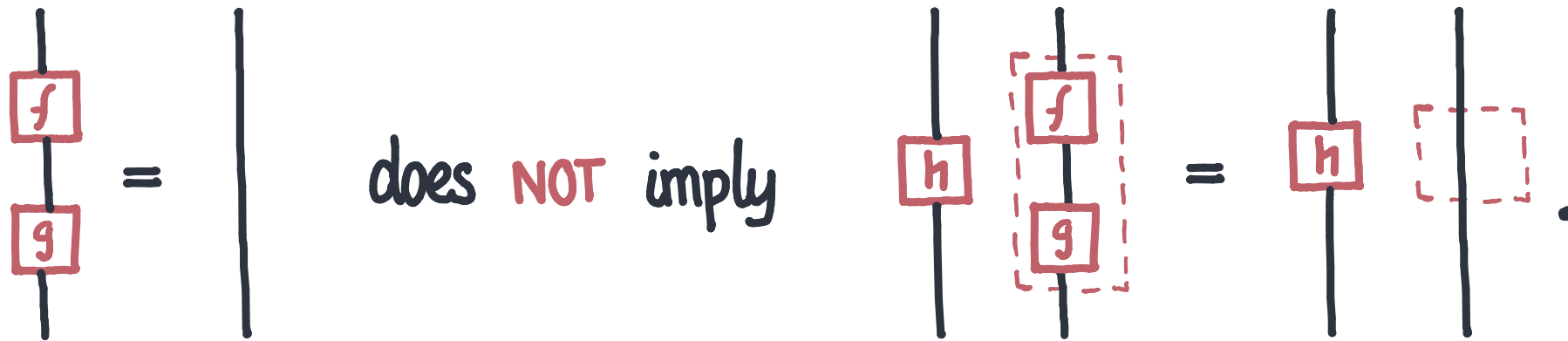
- This is the motivation for Freyd categories. 📄 Staton, Levy

DEFINITION. An **effectful category** is an identity-on-objects functor from a symmetric monoidal category $\mathcal{V}$ ("the **values**") to a symmetric premonoidal category $\mathcal{C}$ ("the **computations**"), strictly preserving the premonoidal structure and centrality.



EFFECTFUL CATEGORIES

Even when every effectful can be strictified, and even if we keep track of central and non-central morphisms,



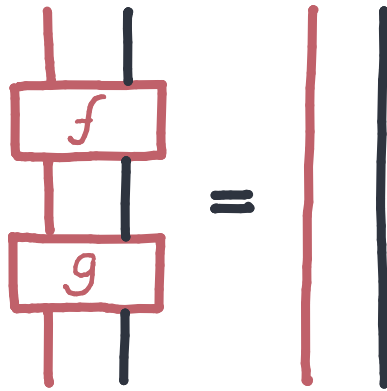
we lose 'locality' of transformations.

- Because of this discrepancy, the categories in ACT are seldom premonoidal. Diagrams are bad. What if there were a solution?

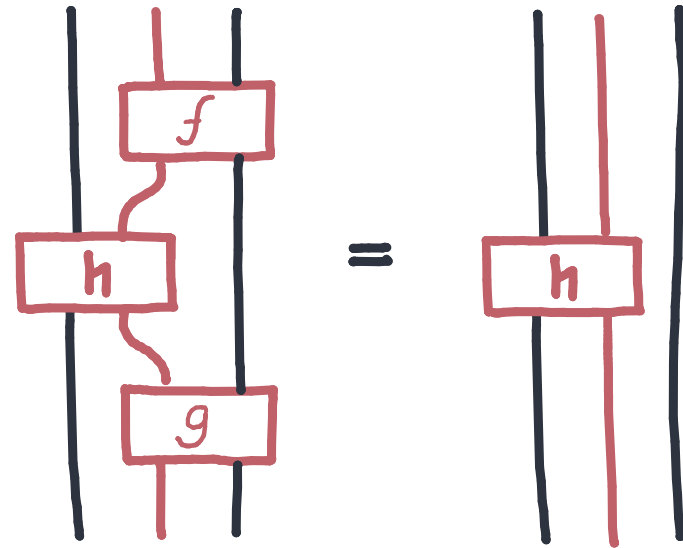
EFFECTFUL CATEGORIES

A solution was proposed in the 90s: *add an extra wire.*

 Jeffrey, 97.



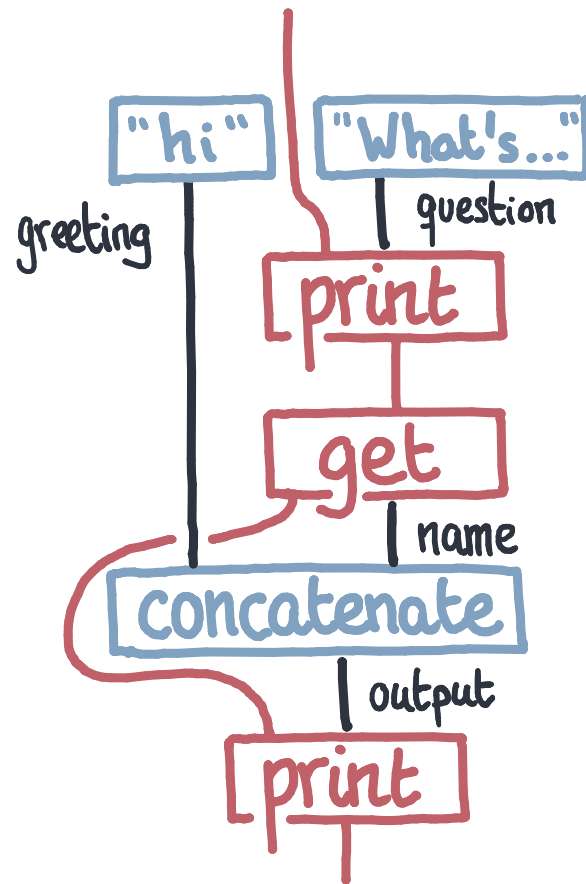
does NOT imply



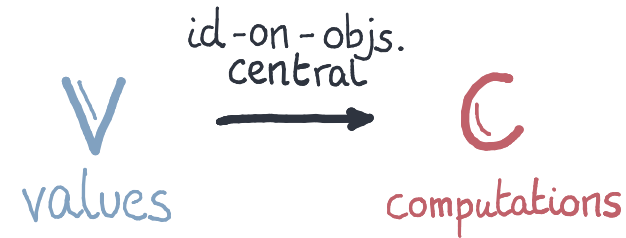
Interpretation: *Runtime*, needed for *computation*, is a resource of your resource theory.

RUNTIME

 Jeffrey, 97
 Power, Thielecke, 99
 Staton, Levy, 13



- An extra wire represents **runtime**: global state of the system and control over it.
- JEFFREY'S **runtime** is flexible: we can choose a class of **pure** morphisms. It works for effectful categories.



- We can always choose $\mathbb{Z}\mathbb{C} \rightarrow \mathbb{C}$ for premonoidals.

From now on, without loss of generality, we work with effectful categories.

WHY EFFECTFUL CATEGORIES (GET/PUT EXAMPLE)

Many structures are clear once the runtime is present: consider the state monad $T_S X = S \rightarrow S \times X$ with $\text{get } \eta : 1 \rightarrow S$ and $\text{put } \flat : S \rightarrow 1$.

$$\eta \mid = \eta \cup ; \quad \boxed{\eta} = \eta ;$$

get-get get-discard

$$\flat \mid = \flat \cup ;$$

put-put

$$\flat \mid = \boxed{\eta} ;$$

get-put

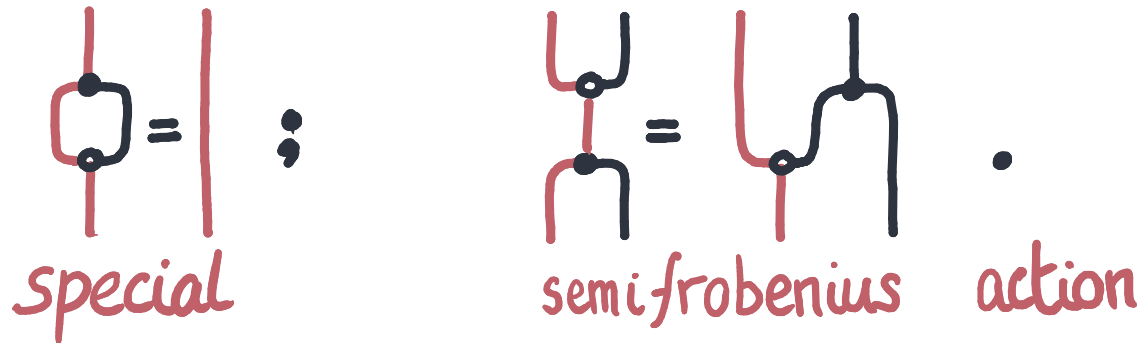
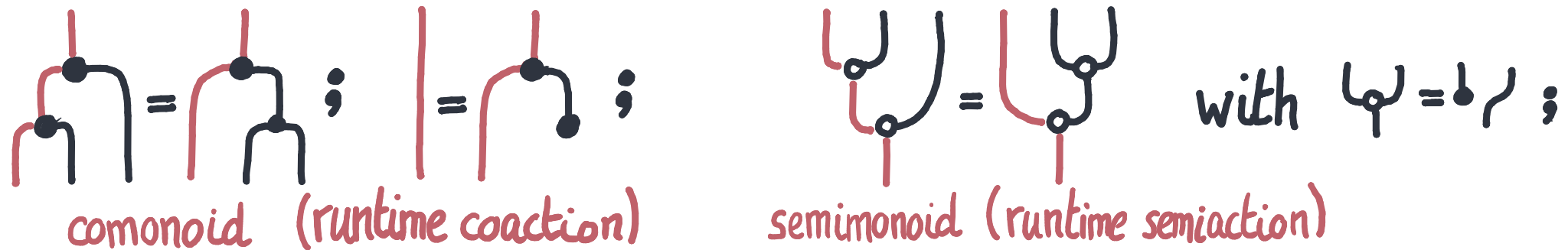
$$\eta \mid = \eta \cup \flat$$

put-get

Consider some lens laws, drawn naively.

WHY EFFECTFUL CATEGORIES (GET/PUT EXAMPLE)

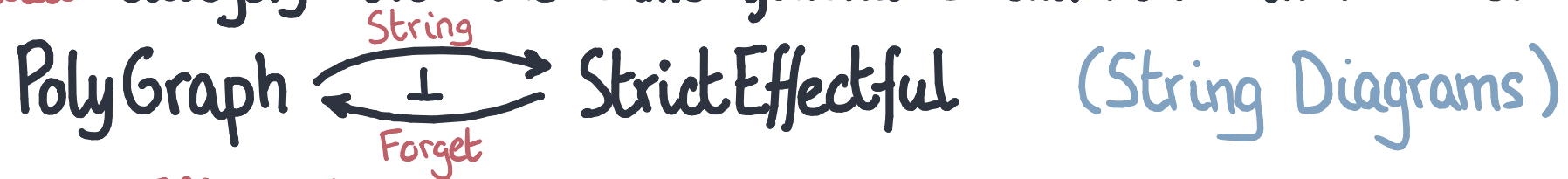
Many structures are clear once the runtime is present: consider the state monad $T_S X = S \rightarrow S * X$ with $\text{get} \dashv : 1 \rightarrow S$ and $\text{put} \dashv : S \rightarrow 1$.



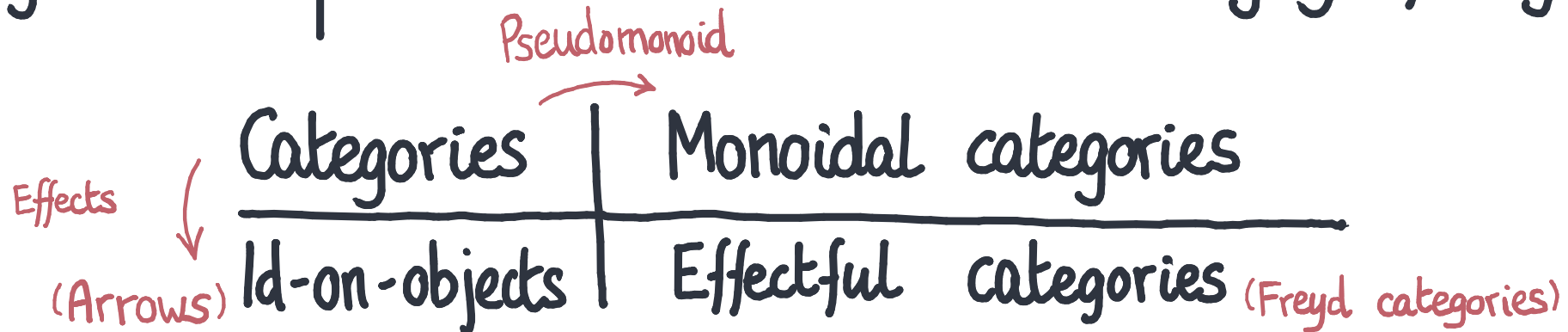
We can study a two-colour PROP instead.

CONTRIBUTIONS

1. THEOREM. The free **effectful** category in some generators is the free **monoidal** category over the same generators endowed with runtime.



2. THEOREM. **Effectful categories** are pseudomonoids in a monoidal bicategory of promonads; in the same way that monoidal categories are pseudomonoids in a monoidal bicategory of categories.



PART 1: RUNTIME

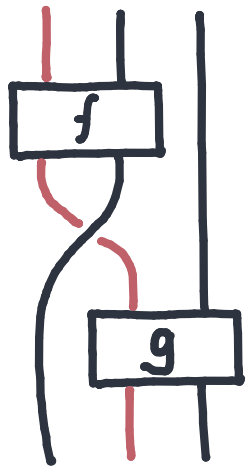
ADDING RUNTIME

JEFFREY'S NOTATION. Avoid interchange by adding an extra wire: the **runtime**.

THEOREM. The morphisms $A_0 \otimes \dots \otimes A_n \rightarrow B_0 \otimes \dots \otimes B_m$ of the free effectful category over a 'signature' $\mathcal{H} \rightarrow \mathcal{G}$ are morphisms

$$R \otimes A_0 \otimes \dots \otimes A_n \rightarrow R \otimes B_0 \otimes \dots \otimes B_m$$

of a monoidal category with generators



and the expected axioms.

POLYGRAPHS

DEFINITION. A **polygraph** $\mathcal{G}$ is given by a set of objects, $\mathcal{G}_{\text{obj}}$, and a set of arrows $\mathcal{G}(A_0, \dots, A_n; B_0, \dots, B_n)$ for sequences $A_0, \dots, A_n$ and $B_0, \dots, B_n$.

EXAMPLE.

$$\mathcal{G} = \left\{ A, B, C \mid \begin{array}{c} A \quad B \\ \boxed{f} \\ C \end{array}, \begin{array}{c} C \\ \boxed{g} \\ C \end{array} \right\}$$

Polygraphs give signatures for monoidal categories: string diagrams over a polygraph form the free strict monoidal category on it.

THEOREM. String diagrams construct an adjunction

$$\text{POLYGRAPH} \begin{array}{c} \xrightarrow{\text{Strings}} \\ \perp \\ \xleftarrow{\text{Forget}} \end{array} \text{STRICTMONCAT.}$$

POLYGRAPHS

DEFINITION. A **polygraph couple** $(\mathcal{H}, \mathcal{G})$ is a pair of polygraphs sharing the same objects, $\mathcal{H}_{\text{obj}} = \mathcal{G}_{\text{obj}}$.

EXAMPLE.

$$(\mathcal{H}, \mathcal{G}) = \left\{ A, B, C \mid \begin{array}{c} \text{A} \quad \text{B} \\ | \quad | \\ \boxed{f} \\ | \\ \text{C} \end{array}, \begin{array}{c} \text{C} \\ | \\ \boxed{g} \\ | \\ \text{C} \end{array}, \begin{array}{c} \text{A} \quad \text{A} \\ | \quad | \\ \boxed{h} \\ | \\ \text{B} \end{array} \right\}$$

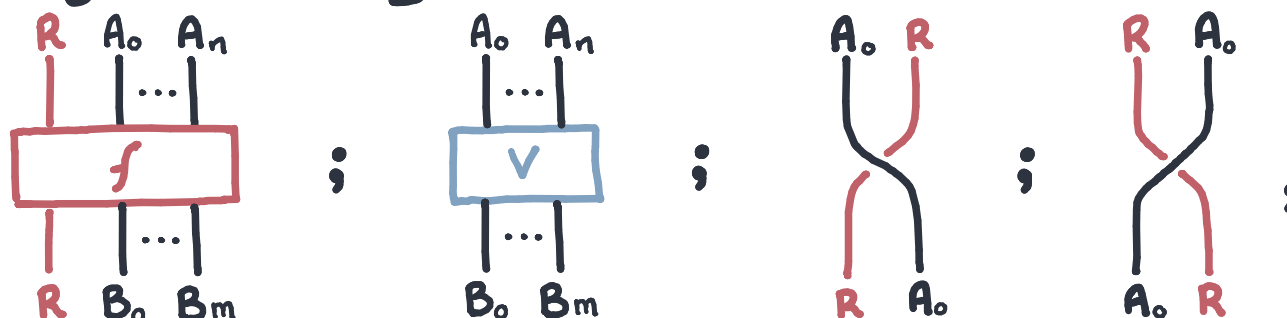
Polygraph couples give signatures for effectful categories.

THEOREM. String diagrams with runtime construct an adjunction

$$\text{POLYGRAPH COUPLE} \begin{array}{c} \xrightarrow{\text{Strings}} \\ \perp \\ \xleftarrow{\text{Forget}} \end{array} \text{STRICT EFFECTFUL.}$$

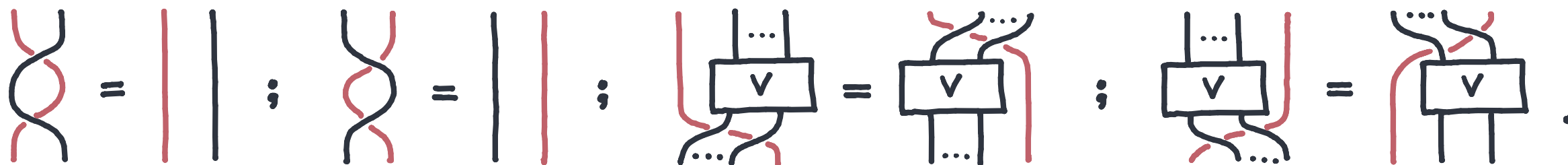
STRING DIAGRAMS WITH RUNTIME

String diagrams generated by



for $f \in \mathcal{G}(A_0, \dots, A_n; B_0, \dots, B_m)$, and $v \in \mathcal{H}(A_0, \dots, A_n; B_0, \dots, B_m)$.

and quotiented by braiding axioms, asking R to be on the Drinfeld centre:



We prove that diagrams $R \otimes A_0 \otimes \dots \otimes A_n \rightarrow R \otimes B_0 \otimes \dots \otimes B_m$ form an effectful category.

CLIQUE

Technical problem: $R \otimes A_0 \otimes \dots \otimes A_n \rightarrow R \otimes B_0 \otimes \dots \otimes B_m$ always assumes that the runtime is on the left.

DEFINITION. A **clique** is a collection of objects, $\{A_i\}_{i \in I}$, together with an isomorphism $\gamma_{ij} : A_i \rightarrow A_j$, such that $\gamma_{ii} = \text{id}_i$, and $\gamma_{ij} \circ \gamma_{jk} = \gamma_{ik}$.

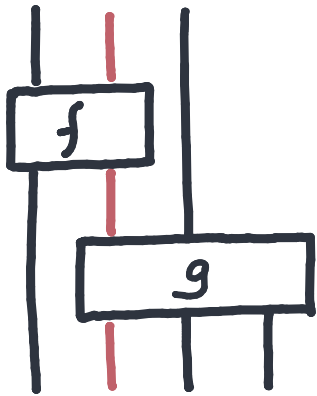
CONSTRUCTION. The **braid clique** on a list of objects $[A_0, \dots, A_n]$ has R inserted at each position.

$$\begin{array}{ccc} & \nearrow \sigma & A_0 \otimes R \otimes \dots \otimes A_n \\ R \otimes A_0 \otimes \dots \otimes A_n & \parallel & \searrow \sigma \\ & \nwarrow \sigma & A_0 \otimes \dots \otimes R \otimes A_n \end{array}$$

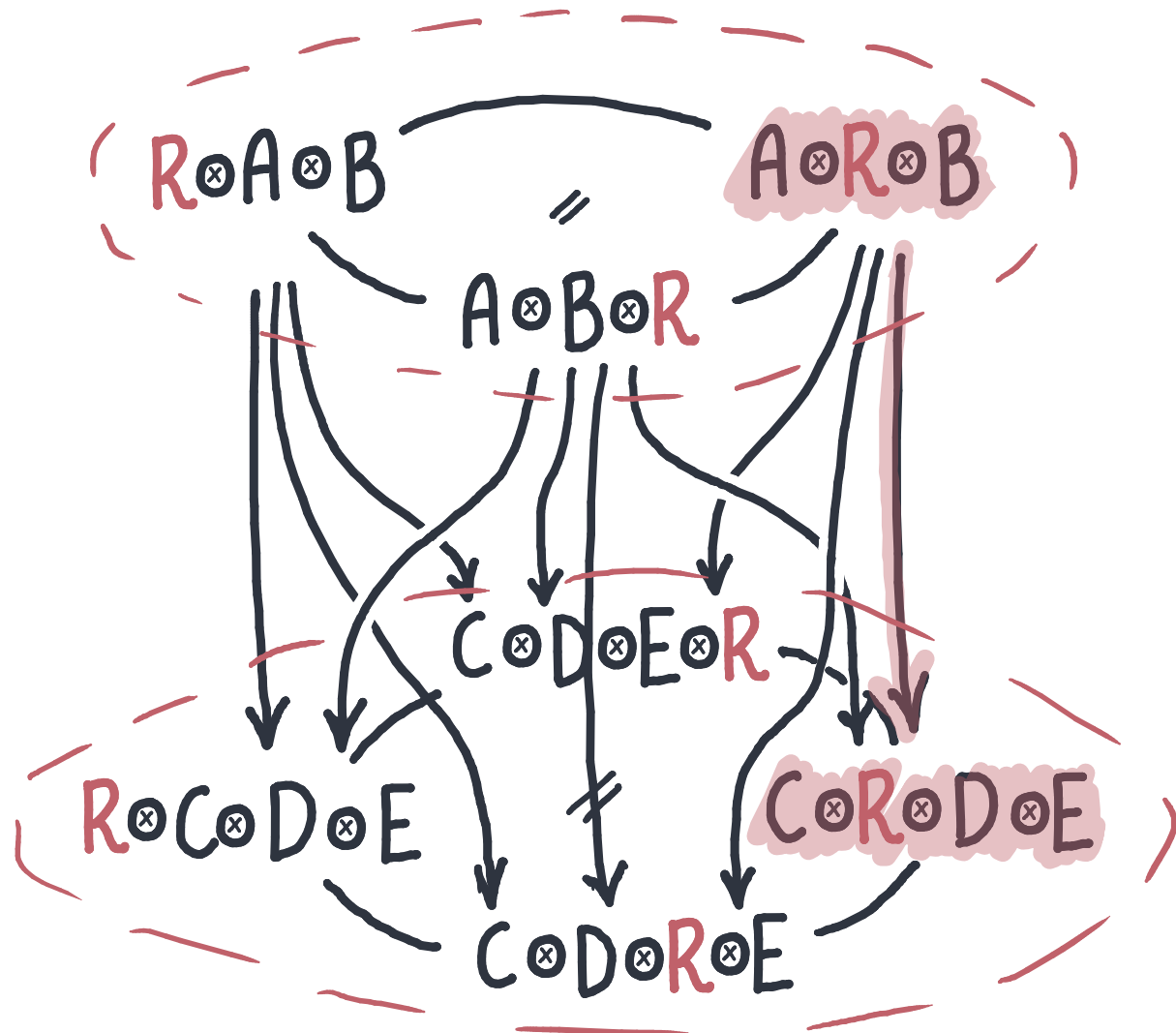
~ See Trimble's intuition of monoidal coherence, on the nLab.

CLIQUEES

$$A \otimes R \otimes B$$

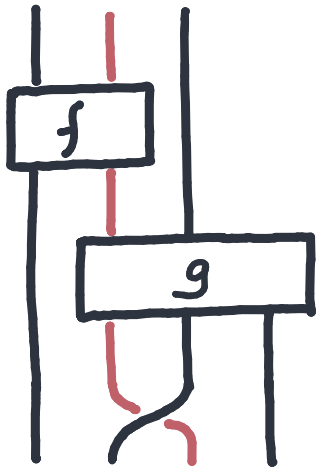


$$C \otimes R \otimes D \otimes E$$

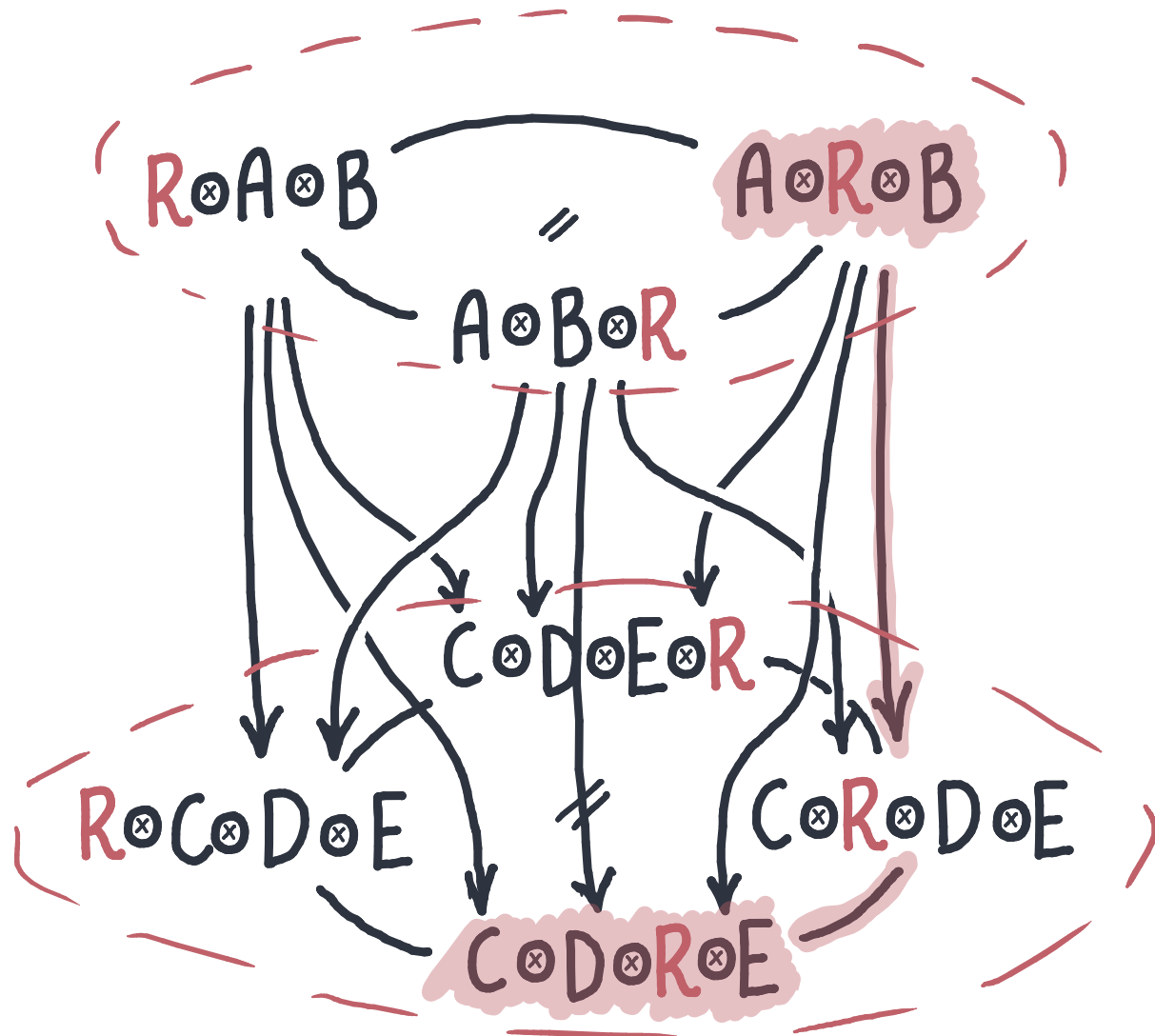


CLIQUEES

$A \otimes R \otimes B$

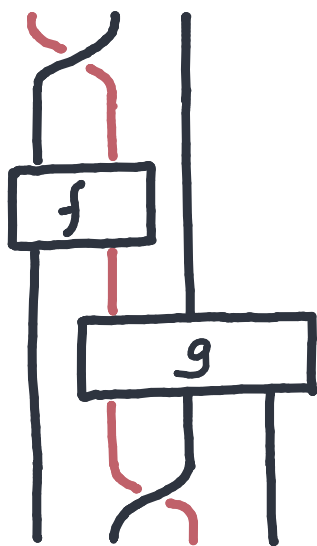


$C \otimes D \otimes R \otimes E$

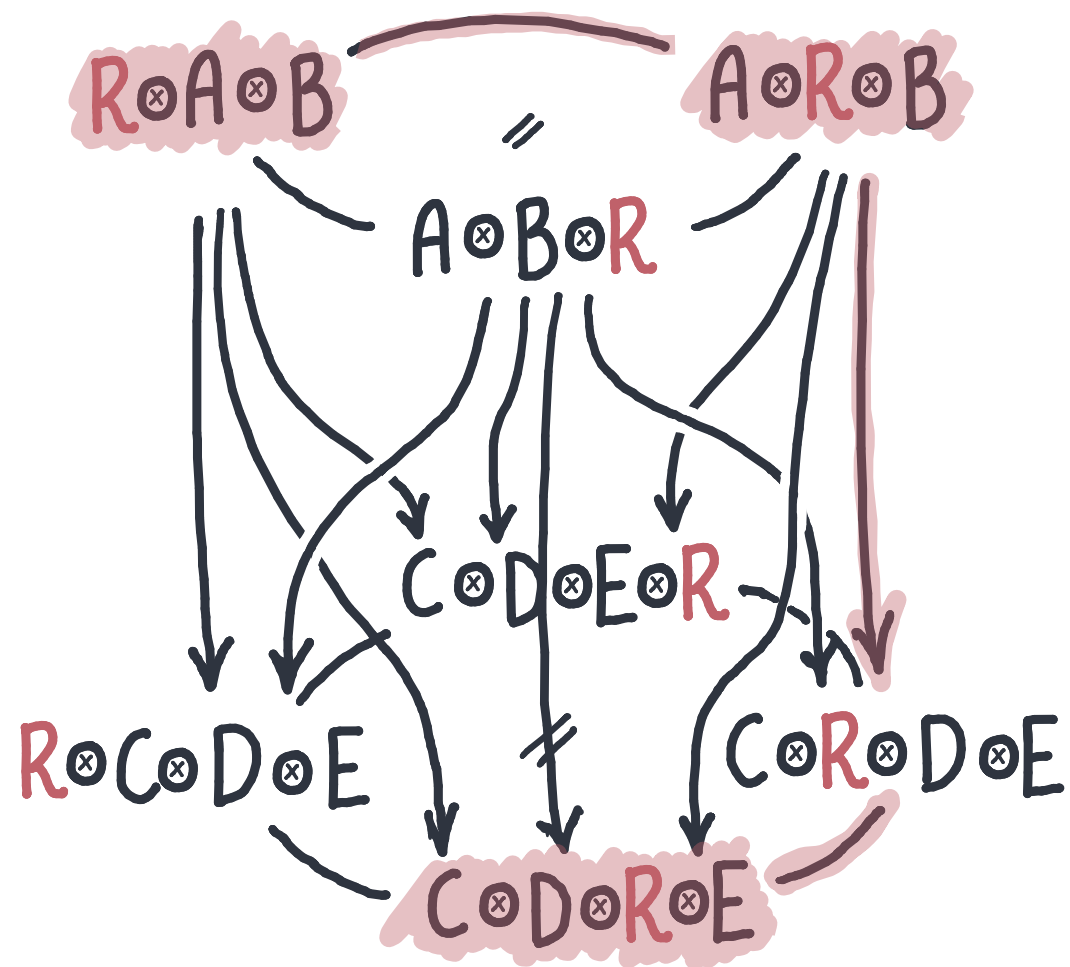


CLIQUEES

$A \otimes R \otimes B$



$C \otimes D \otimes R \otimes E$



STRING DIAGRAMS WITH RUNTIME

Finally, the assignment determining the universal effectful functor is

$$H\left(\begin{array}{c} \vdots \\ \boxed{} \\ \vdots \end{array}\right) = \text{id} \ ; \ H\left(\begin{array}{c} \text{ } \\ \diagup \quad \diagdown \\ \text{ } \end{array}\right) = H\left(\begin{array}{c} \text{ } \\ \diagdown \quad \diagup \\ \text{ } \end{array}\right) = \text{id} \ ; \ H\left(\begin{array}{c} \vdots \\ \boxed{f} \\ \vdots \end{array}\right) = F(f) \ ;$$

$$H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \\ \boxed{y} \\ \vdots \end{array}\right) = H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) \circ H\left(\begin{array}{c} \vdots \\ \boxed{y} \\ \vdots \end{array}\right) \ ;$$

$$H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array} \begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array}\right) = H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) \otimes \text{id} \circ \text{id} \otimes H_0\left(\begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array}\right) = \text{id} \otimes H_0\left(\begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array}\right) \circ H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) \otimes \text{id} \ ;$$

$$H\left(\begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array} \begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) = H_0\left(\begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array}\right) \otimes \text{id} \circ \text{id} \otimes H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) = \text{id} \otimes H\left(\begin{array}{c} \vdots \\ \boxed{x} \\ \vdots \end{array}\right) \circ H_0\left(\begin{array}{c} \vdots \\ \boxed{u} \\ \vdots \end{array}\right) \otimes \text{id} \ ;$$

THEOREM. String diagrams with runtime construct an adjunction

$$\text{POLYGRAPH COUPLE} \begin{array}{c} \xrightarrow{\text{Strings}} \\ \perp \\ \xleftarrow{\text{Forget}} \end{array} \text{STRICT EFFECTFUL.}$$

PART 2: PROMONADS

PROFUNCTORS

A **profunctor** $P: \mathcal{C}^{\text{op}} \rightarrow \mathcal{D}$ is the same thing as a functor $P: \mathcal{C} \times \mathcal{D} \rightarrow \text{SET}$.

Alternatively, a family of sets, $P(A, B)$, with actions

$(>): \text{hom}(A', A) \times P(A, B) \rightarrow P(A', B)$ $(<): P(A, B) \times \text{hom}(B, B') \rightarrow P(A, B')$ such that

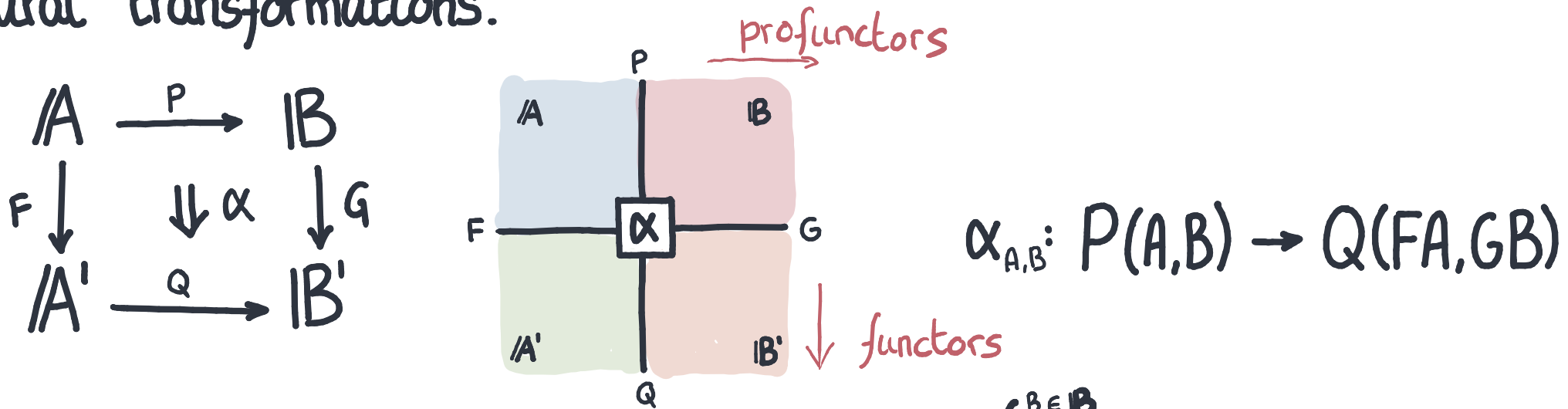
$$\begin{aligned} f_1 \circ f_2 > p &= f_1 > f_2 > p & p < g_1 \circ g_2 &= p < g_1 < g_2 & (f > p) < g &= f > (p < g) \\ \text{id} > p &= p & p < \text{id} &= p \end{aligned}$$

Profunctors compose as bimodules: $P \diamond Q(A, C)$ has elements (p, q) for $p \in P(A, B)$ and $q \in Q(B, C)$, quotiented by $(p < g, q) \sim (p, g > q)$. This is a coend

$$(P \diamond Q)(A, C) = \int^{B \in \mathcal{D}} P(A, B) \times Q(B, C).$$

DOUBLE CATEGORY CAT

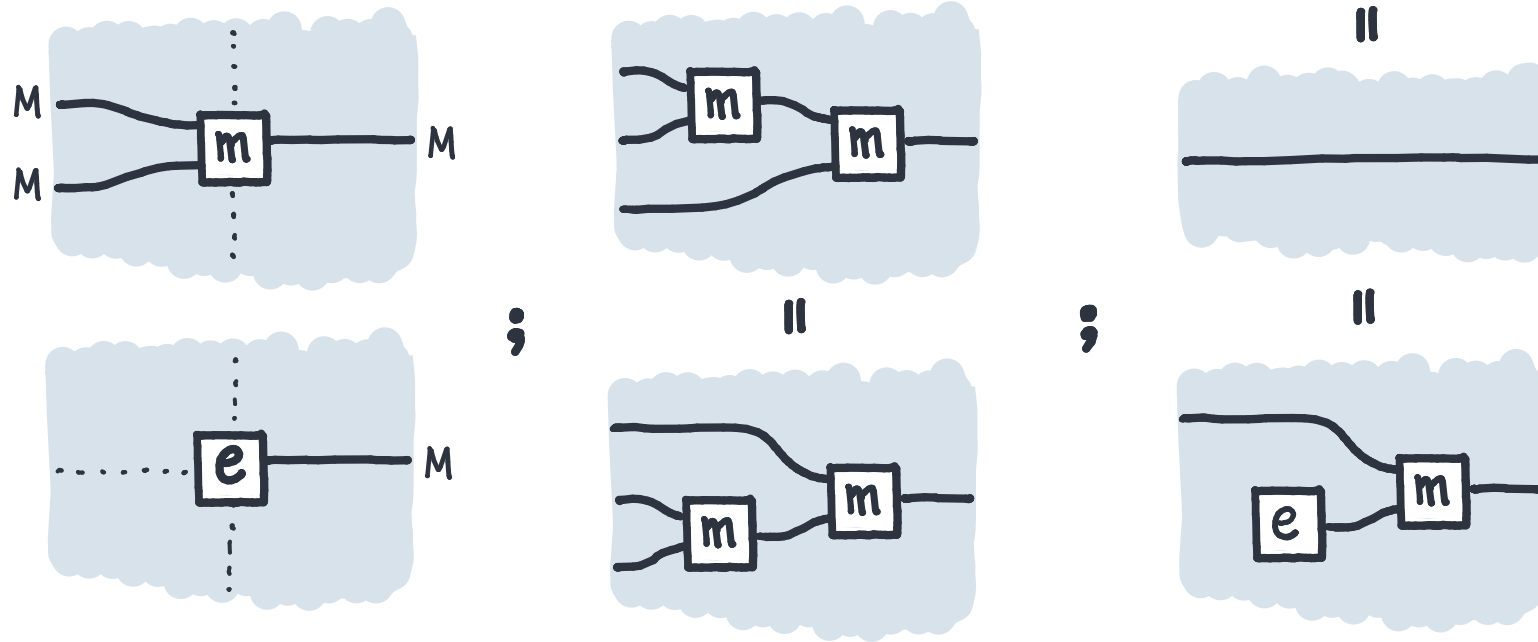
Consider the double category of categories, functors, profunctors and natural transformations.



Profunctors compose by *coends*, $P \diamond Q (A,C) = \int^{B \in \mathcal{B}} P(A,B) \times Q(B,C)$;
the identity is the *hom*-profunctor.

MONADS

Monads are the **monoids** of CAT.



$$I \Rightarrow M$$

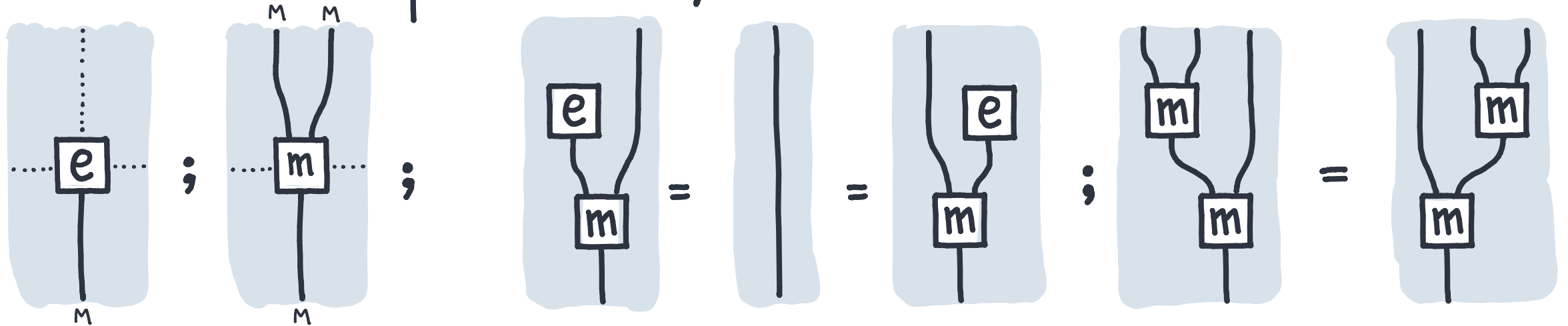
$$M \circ M \Rightarrow M$$

$$\eta: X \rightarrow MX$$

$$\mu: MMX \rightarrow MX$$

PROMONADS

Promonads are the promonoids of CAT.



$$\text{hom} \Rightarrow M$$

$$M \circ M \Rightarrow M$$

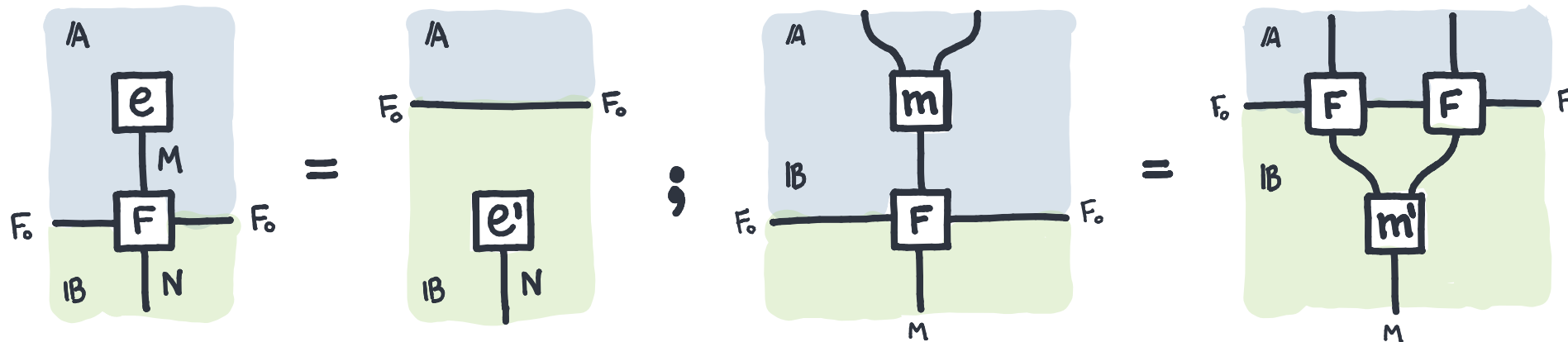
$$(\cdot)^\circ: \text{hom}(A, B) \rightarrow M(A, B)$$

$$\star: \int^B M(A, B) \times M(B, C) \rightarrow M(A, C)$$

PROPOSITION. A promonad $P: \mathcal{C} \nrightarrow \mathcal{C}$ is exactly an **id-on-objects functor** $i: \mathcal{C} \rightarrow \tilde{P}$.

PROMONAD MORPHISMS

Promonad homomorphisms are promonoid homomorphisms in the double category CAT.



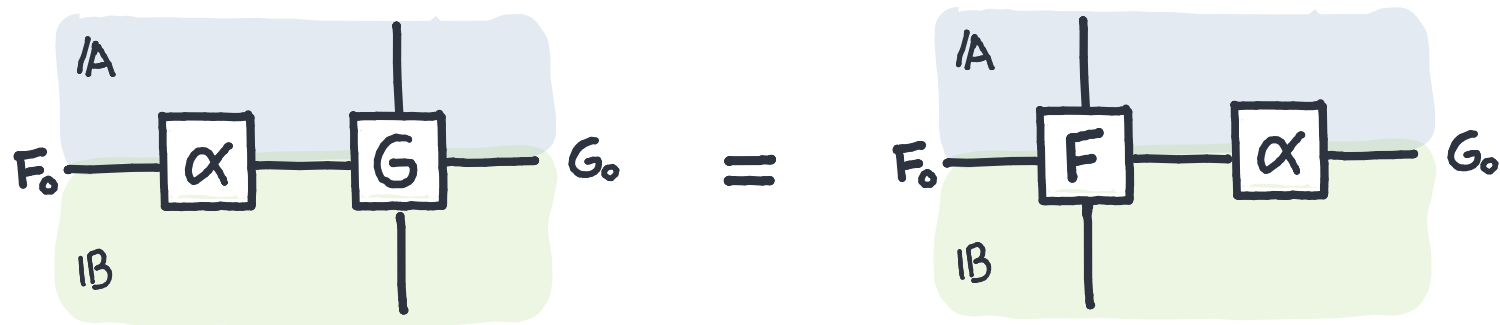
PROPOSITION. A **promonad homomorphism** is exactly a commuting square.

$$\begin{array}{ccc}
 \mathbb{C} & \xrightarrow{(\circ)} & \mathbb{P} \\
 F_0 \downarrow & \parallel & \downarrow F \\
 \mathbb{D} & \xrightarrow{(\circ)} & \mathbb{Q}
 \end{array}$$

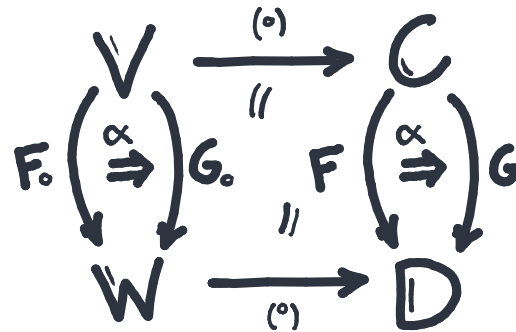
$$\begin{aligned}
 F(f \star g) &= Ff \star Fg, \\
 F(f^\circ) &= F_0(f)^\circ.
 \end{aligned}$$

PROMONAD TRANSFORMATION

Promonad transformations are promonoid transformations in the double category CAT.



PROPOSITION. A **promonad transformation** is exactly a cylinder transformation of squares.



PURE TENSOR

We already have a bicategory: promonads, morphisms and transformations.
There exists a notion of free product with commuting subgroups.

DEFINITION. The **pure tensor** of promonads, $C: V \nrightarrow V$ and $D: W \nrightarrow W$, is a promonad, $C * D: V * W \nrightarrow V * W$, whose elements are generated by

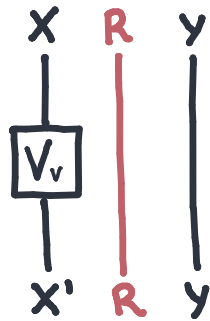
$$p_C \in C * D(X, Y; X', Y) \text{ for } p \in C(X, X'), \text{ or} \\ q_D \in C * D(X, Y; X, Y') \text{ for } q \in D(Y, Y').$$

quotiented by $f_C \circ w_D^\circ = w_D^\circ \circ f_C$ and $g_D \circ v_C^\circ = v_C^\circ \circ g_D$, plus other axioms.

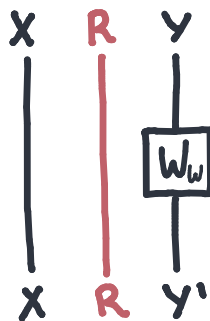
THEOREM. There exist a pair of promonad homomorphisms
 $L: C \times W \rightarrow C * D$ and $R: V \times D \rightarrow C * D$
and the **pure tensor** is universal with respect to them.

PURE TENSOR

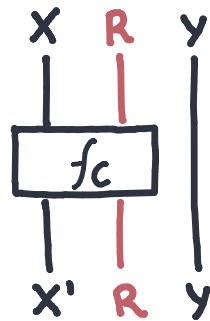
We already have a bicategory: promonads, morphisms and transformations.
There exists a notion of free product with commuting subgroups.



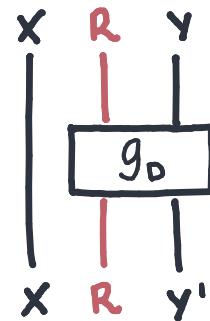
for $v \in V(x, x')$



for $w \in W(y, y')$



for $f \in C(x, x')$



for $g \in D(x, x')$

All these commutativity restrictions are better encoded by string diagrams.

- Thanks to an extra wire.

EFFECTFULS AS PSEUDOMONONIDS

Pseudomonoids are 2-dimensional monoids.

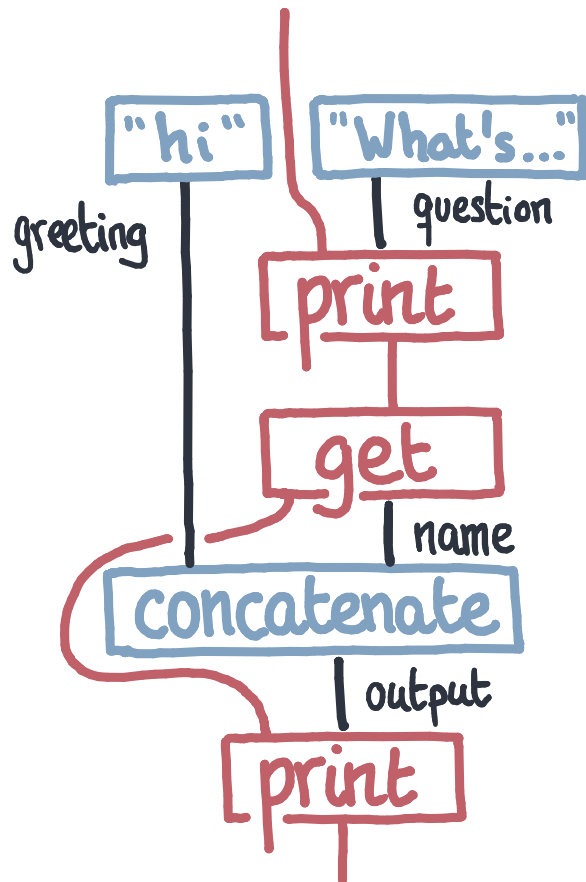
THEOREM ( Street, Day). Monoidal categories are pseudomonoids in the monoidal bicategory of categories.

THEOREM. Effectful categories (and thus premonoidal categories with their centre) are pseudomonoids in the monoidal bicategory of promonads with the pure tensor.

PART 3: CONCLUSION

CONCLUSION

Premonoidal categories are already widely employed in programming languages. This is an invitation to use them in ACT, without leaving the comfort of monoidal categories and string diagrams.



ACT
Monoidal
Categories

Processes
Effectful
Categories

PLang
Premonoidal
Categories

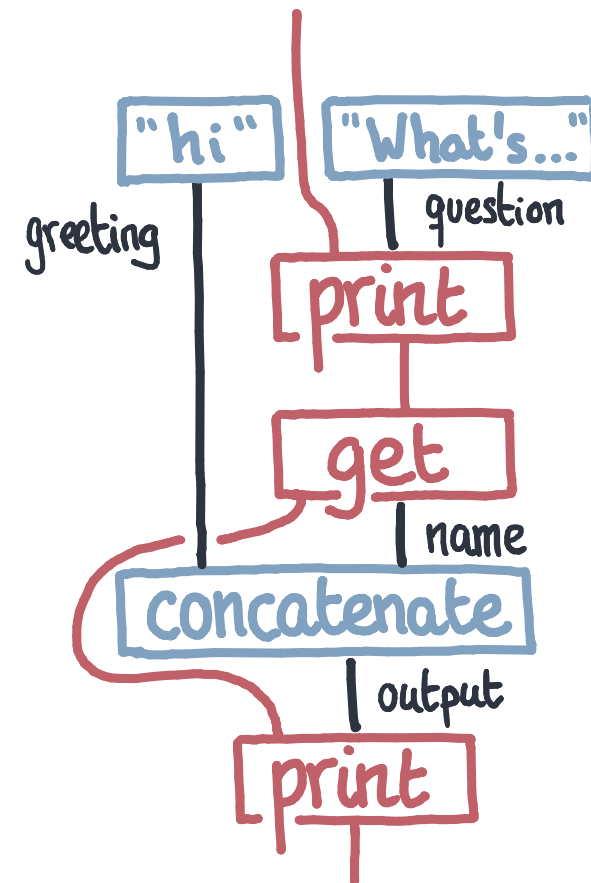
1. `print w;`
2. `n = get();`
3. `print concat("hi", n);`
4. `return ();`

ARROW-DO NOTATION

Correspondence between string diagrams and the DO-notation.

EXAMPLE. HelloProgram

```
proc () → do
  question ← "What's your name?"
  () ← print ← question
  name ← get ← ()
  () ← print ← concatenate("hi, ", name)
return ()
```





PROMONADS AND STRING DIAGRAMS FOR EFFECTFUL CATEGORIES (ROMÁN, 2022)

ArXiv: 2205.07664