STEALING FROM DUSKO

Ross Duncan

University of Strathclyde Glasgow

Computing Science Group

Geometry of abstraction in quantum computation

Dusko Pavlovic
Oxford University and Kestrel Institute

CS-RR-09-13

Computing Science Group

Geometry of abstraction in String Diagrams

Dusko Pavlovic
Oxford University and Kestrel Institute

CS-RR-09-13

Computing Science Group

Geometry of abstraction in String Diagrams

Du

ROSS

Oxford U y and Kestrel Institute

CS-RR-09-13

TOWARDS PATTERN-MATCHING ~~BINDING~~ & SUBSTITUTION in STRING DIAGRAMS

ROSS DUNCAN — UNIVERSITY OF STRATHCLYDE

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

```
f (1 :: (2 :: 3 :: []))
```

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

```
f (1 :: (2 :: 3 :: []))
```

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

```
f (1 :: (2 :: 3 :: []))
```

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

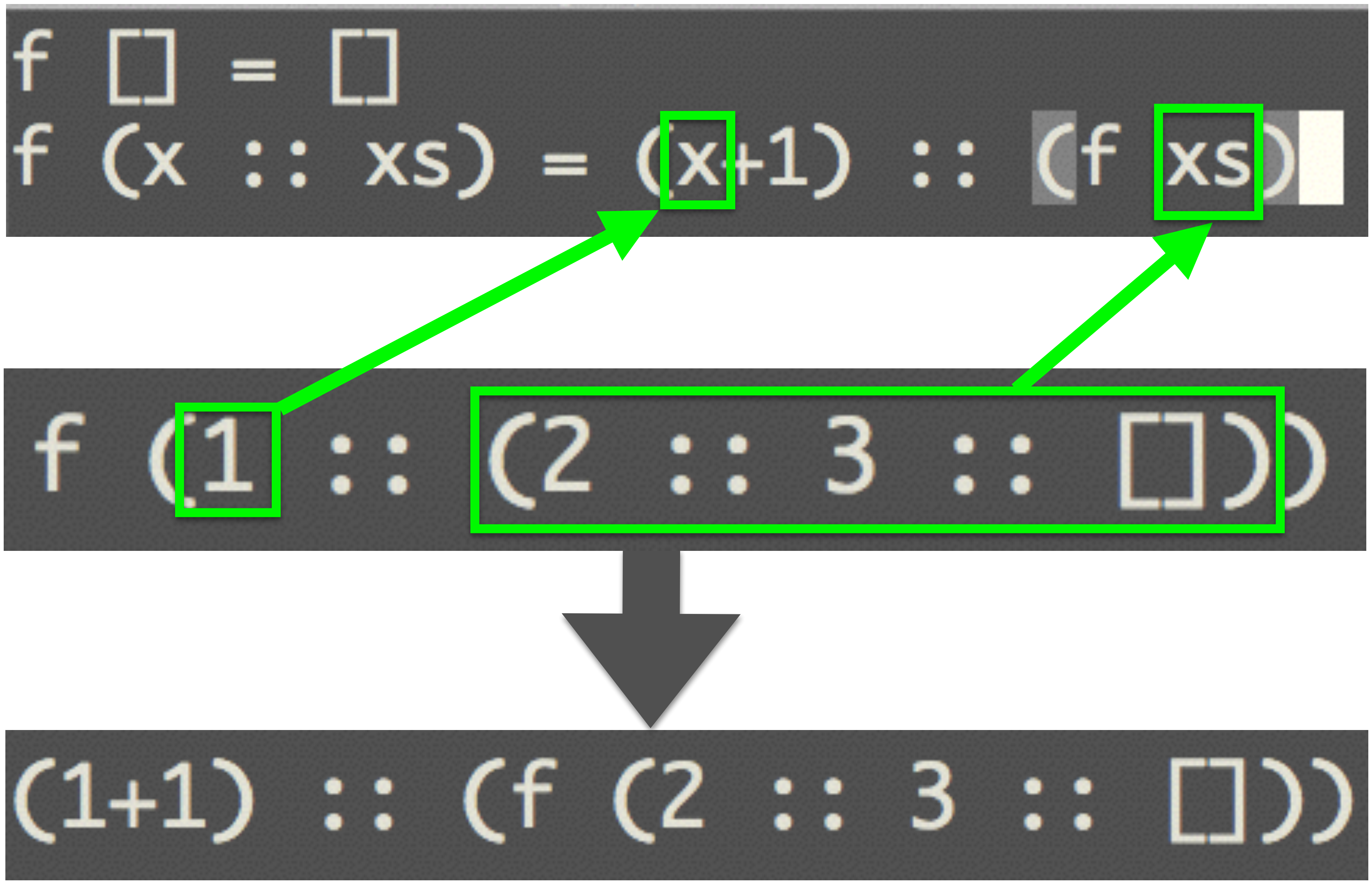
```
f (1 :: (2 :: 3 :: []))
```

Pattern matching, binding and substitution

```
f [] = []
f (x :: xs) = (x+1) :: (f xs)
```

```
f (1 :: (2 :: 3 :: []))
```

Pattern matching, binding and substitution



Computing Science Group

Geometry of abstraction in quantum computation

Dusko Pavlovic
Oxford University and Kestrel Institute

CS-RR-09-13

Geometry of abstraction in quantum computation

Dusko (2009,2012)

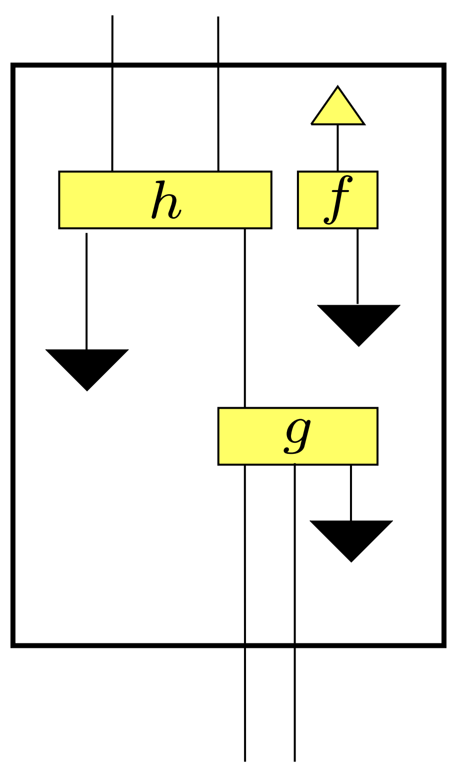
Quantum algorithms are sequences of abstract operations, performed on non-existent computers. They are in obvious need of categorical semantics.

Geometry of abstraction in quantum computation

Dusko (2009,2012)

monoidal category $\mathcal{C}$

polynomial monoidal category $\mathcal{C}[x : X]$

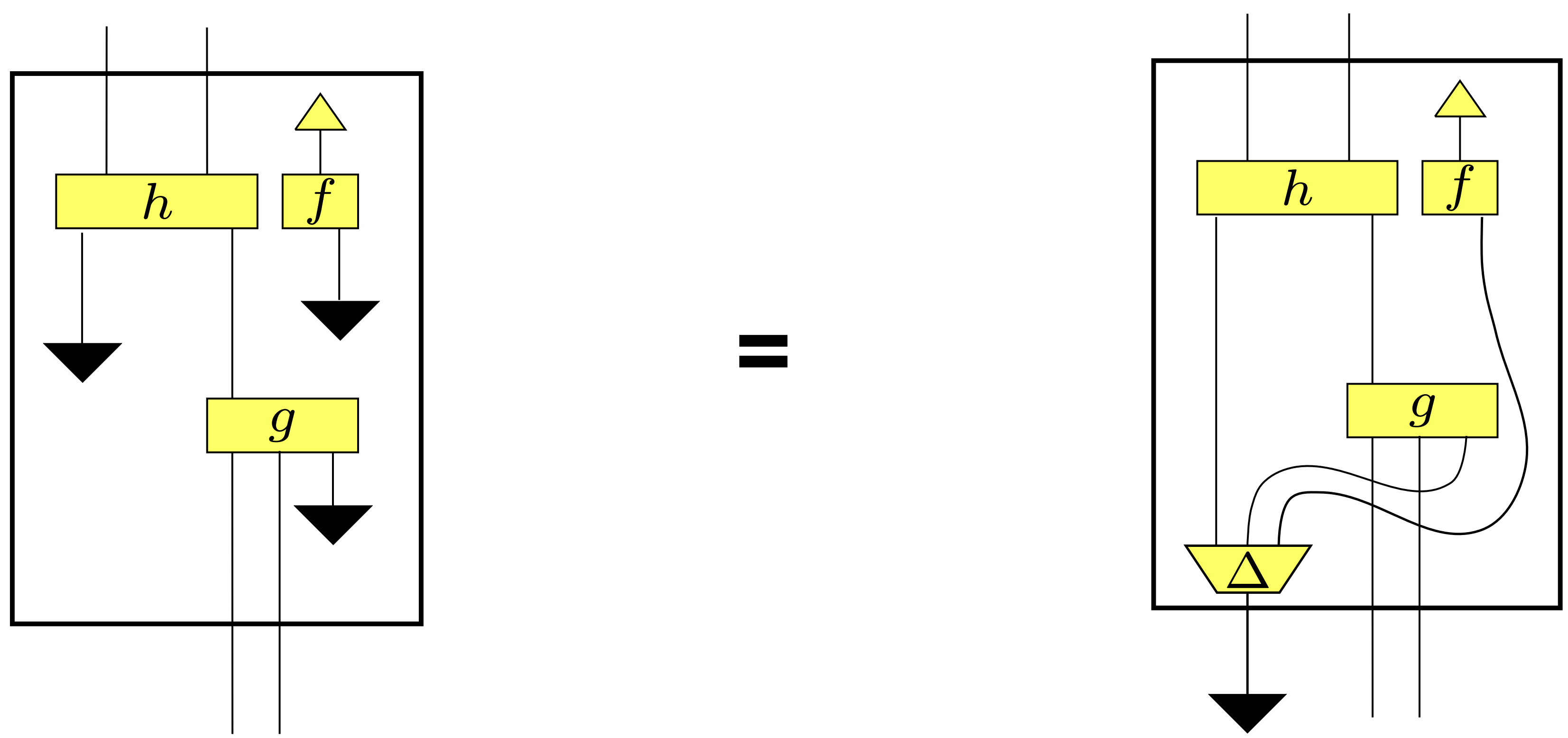


Geometry of abstraction in quantum computation

Dusko (2009,2012)

monoidal category $\mathcal{C}$

polynomial monoidal category $\mathcal{C}[x : X]$



Geometry of abstraction in quantum computation

Dusko (2009,2012)

monoidal category $\mathcal{C}$

polynomial monoidal category $\mathcal{C}[x : X]$

Theorem 3.4 *The category* $\mathrm{Abs}_{\mathrm{C}}$ *of monoidal abstractions is equivalent with the category* $\mathrm{C}_{\times}$ *of commutative comonoids in* C.

Geometry of abstraction in quantum computation

Dusko (2009,2012)

monoidal category $\mathcal{C}$

polynomial monoidal category $\mathcal{C}[x : X]$

Theorem 3.4 *The category* $\mathrm{Abs}_{\mathrm{C}}$ *of monoidal abstractions is equivalent with the category* $\mathrm{C}_{\times}$ *of commutative comonoids in* C.

Corollary 4.5 *The category of dagger-monoidal abstractions* $\ddagger$-$\mathrm{Abs}_{\mathrm{C}}$ *is equivalent with the category* C_{Δ} *of commutative dagger-Frobenius algebras and comonoid homomorphisms in* C

TOWARDS PATTERN-MATCHING ~~BINDING~~ & SUBSTITUTION in STRING DIAGRAMS

ROSS DUNCAN — UNIVERSITY OF STRATHCLYDE

1. OPERADs

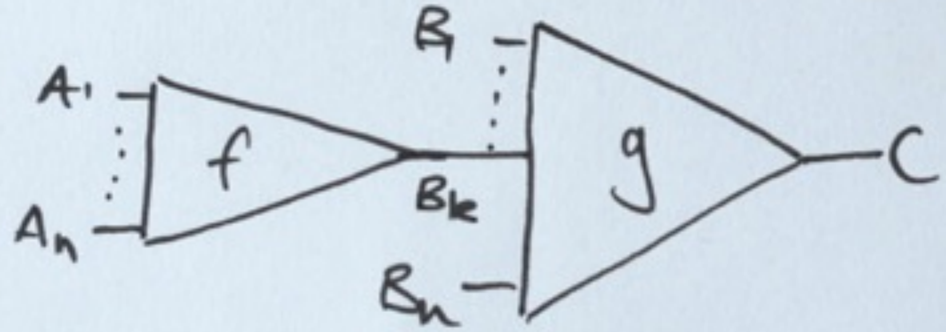
$$A \xrightarrow{f} B \xrightarrow{g} C$$

(arrows in a category)

1. OPERADs

$$A \xrightarrow{f} B \xrightarrow{g} C$$

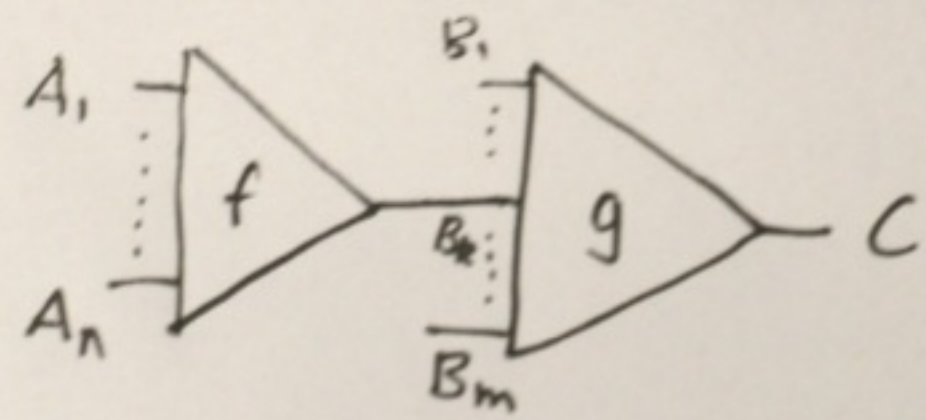
(arrows in a category)



(arrows in an operad)

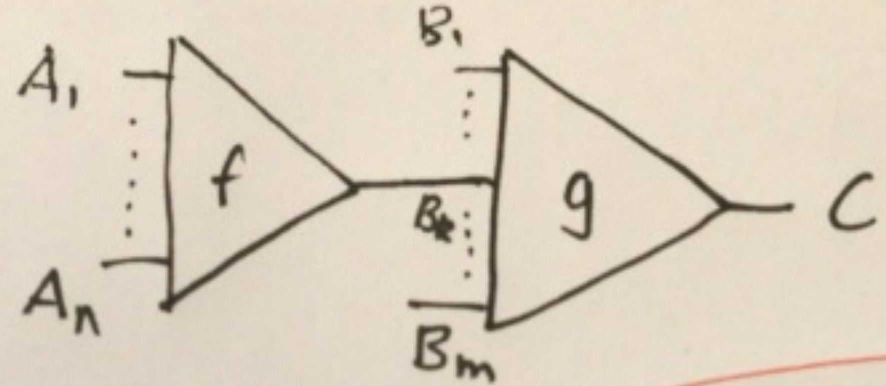
aka. multicategory.

1. OPERADS



$$\frac{x_1 : A_1, \dots, x_n : A_n \vdash f : B_k \qquad y_1 : B_1, \dots, y_k : B_k, \dots, y_m : B_n \vdash g : C}{y_1 : B_1, \dots, x_1 : A_1, \dots, x_n : A_n, \dots, y_m : B_m \vdash g[f/x] : C}\,\text{CUT}$$

1. OPERADS



$$\frac{x_1 : A_1, \dots, x_n : A_n \vdash f : B_k \qquad y_1 : B_1, \dots, \cancel{y_k : B_k}, \dots, y_m : B_m \vdash g : C}{y_1 : B_1, \dots, \boxed{x_1 : A_1, \dots, x_n : A_n}, \dots, y_m : B_m \vdash g[f/y_k] : C}\ \text{CUT}$$

2. MAKING AN OPERAD FROM A PRO

- Let (Σ, E) be a presentation of a PRO.

variables.

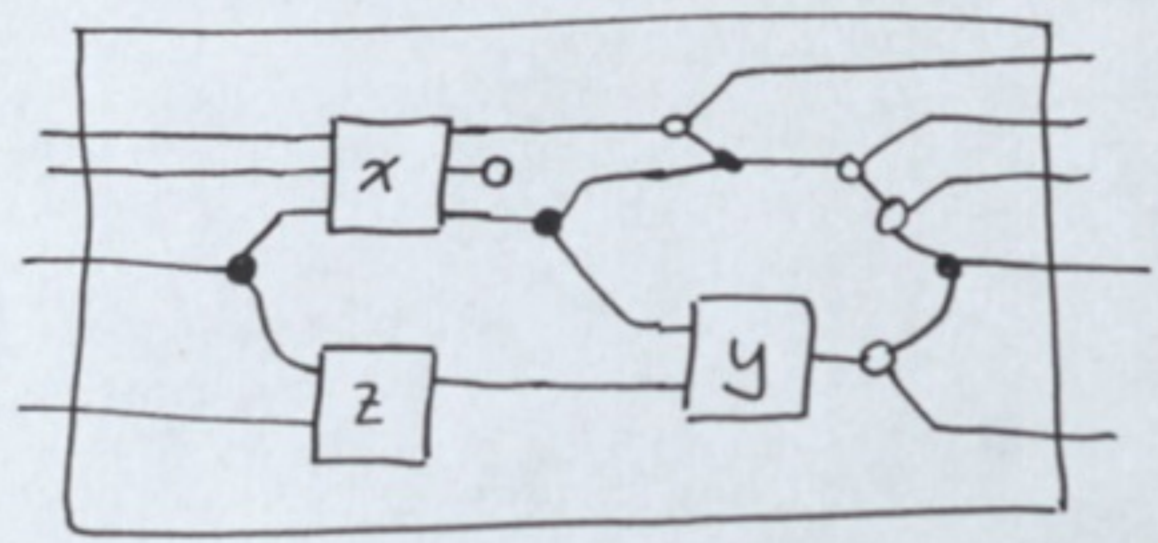
- Adjoin "enough" new generators $x : m \to n$ for every $m, n \in \mathbb{N}$.
- Then $(\Sigma + Var, E)$ is again a PRO with (term) variables.

2. Making an Operad from a PRO

- Let (Σ, E) be a presentation of a PRO.

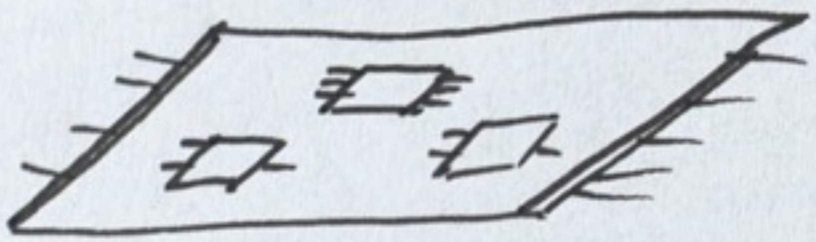
variables.

- Adjoin "enough" new generators $x : m \to n$ for every $m, n \in \mathbb{N}$.
- Then $(\Sigma + \mathrm{Var}, E)$ is again a PRO with (term) variables.

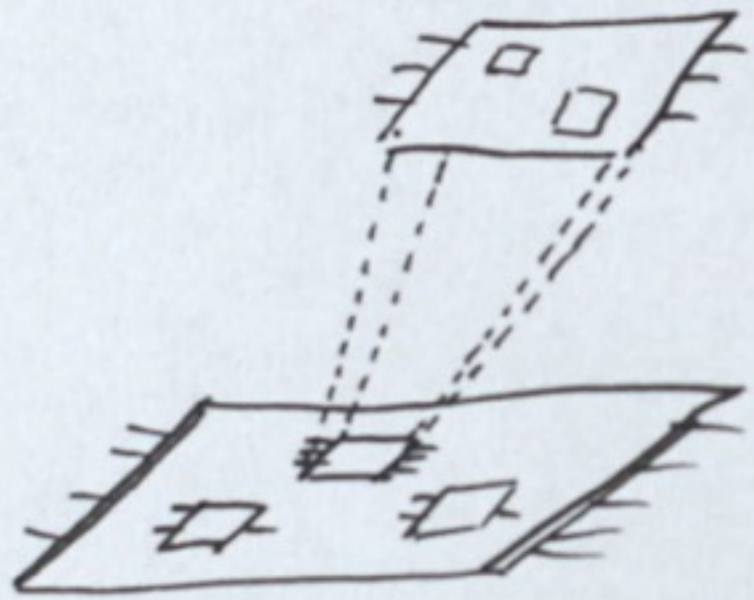


Assume variables only occur once (for now)

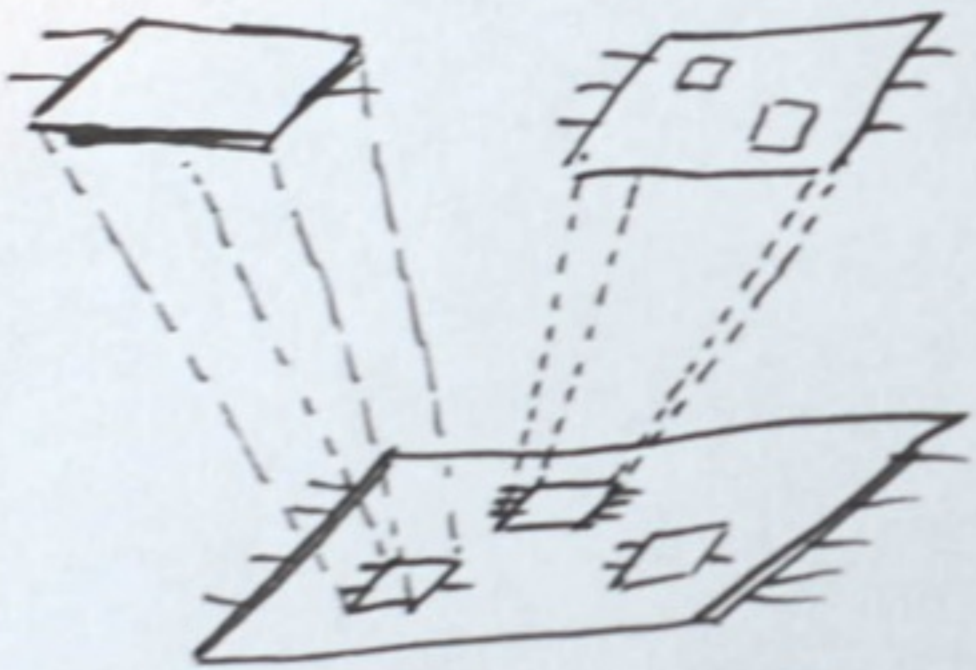
$$x:(3,3),\ y:(2,1),\ z:(2,1) \vdash f:(4,5)$$



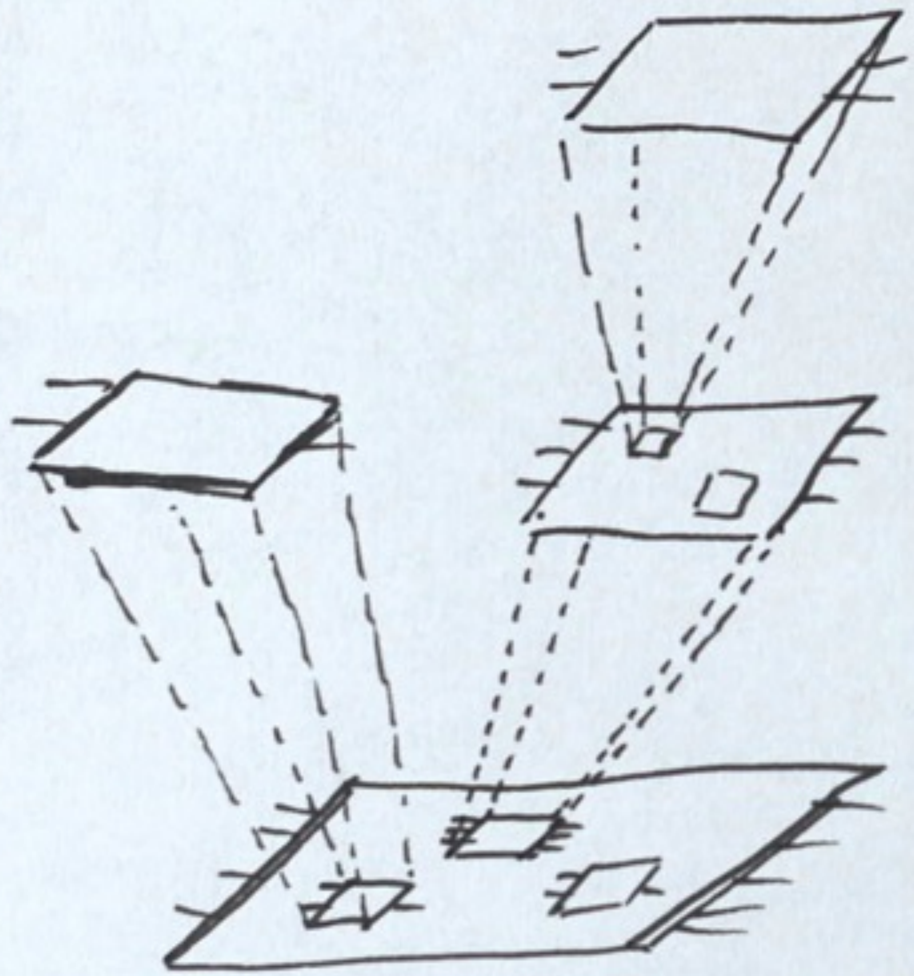
f



$f[g/x]$



$f[g/x][h/z]$



$f[g/x][h/z][k/w]$

etc.

DOUBLE PUSH-OUT REWRITING

$$L = R$$

DOUBLE PUSH-OUT REWRITING

$$L \Rightarrow R$$

Double Push-Out Rewriting

$$L \Rightarrow R$$

$$L \xleftarrow{i_1} \partial \xrightarrow{i_2} R$$

Double Push-Out Rewriting

$$L \Rightarrow R$$

$$\begin{tikzcd} L \arrow[d, hook, "m"'] & \partial \arrow[l, hook', "i_1"'] \arrow[r, hook, "i_2"] & R \\ G & & \end{tikzcd}$$

DOUBLE PUSH-OUT REWRITING

$$L \Rightarrow R$$

$$\begin{tikzcd}
L \arrow[d, hook, "m"'] & \partial \arrow[l, hook', "i_1"'] \arrow[r, hook, "i_2"] \arrow[d, hook] & R \\
G \arrow[ur, phantom, very near start, "\ulcorner"] & G \setminus L \arrow[l, hook']
\end{tikzcd}$$

compute
push-out complement.

DOUBLE PUSH-OUT REWRITING

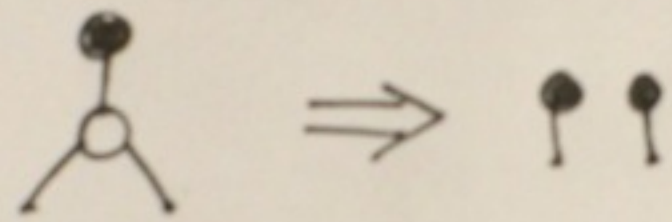
$$L \Rightarrow R$$

$$\begin{tikzcd}
L \arrow[d, "m"'] & \partial \arrow[l, hook', "i_1"'] \arrow[r, hook, "i_2"] \arrow[d, hook, "j"] & R \arrow[d, hook] \\
G \arrow[ur, phantom, "\lrcorner", very near start] & G \setminus L \arrow[l, hook'] \arrow[r, hook] & G[R/L] \arrow[ul, phantom, "\ulcorner", very near start]
\end{tikzcd}$$

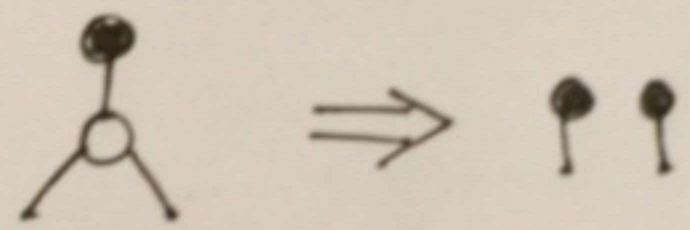
compute
push-out complement.

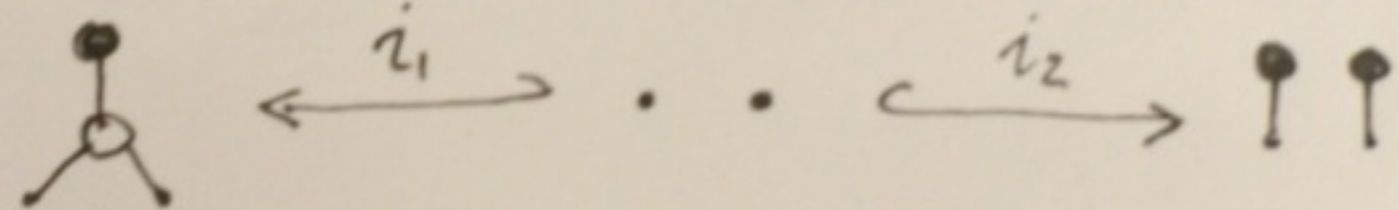
compute ~~po~~ pushout.

DPO REWRITING

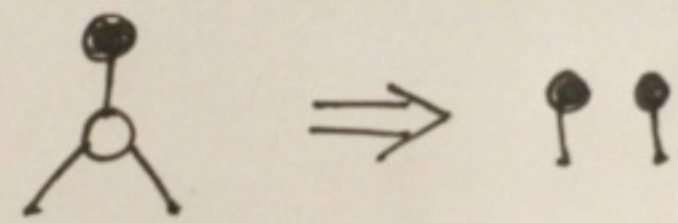


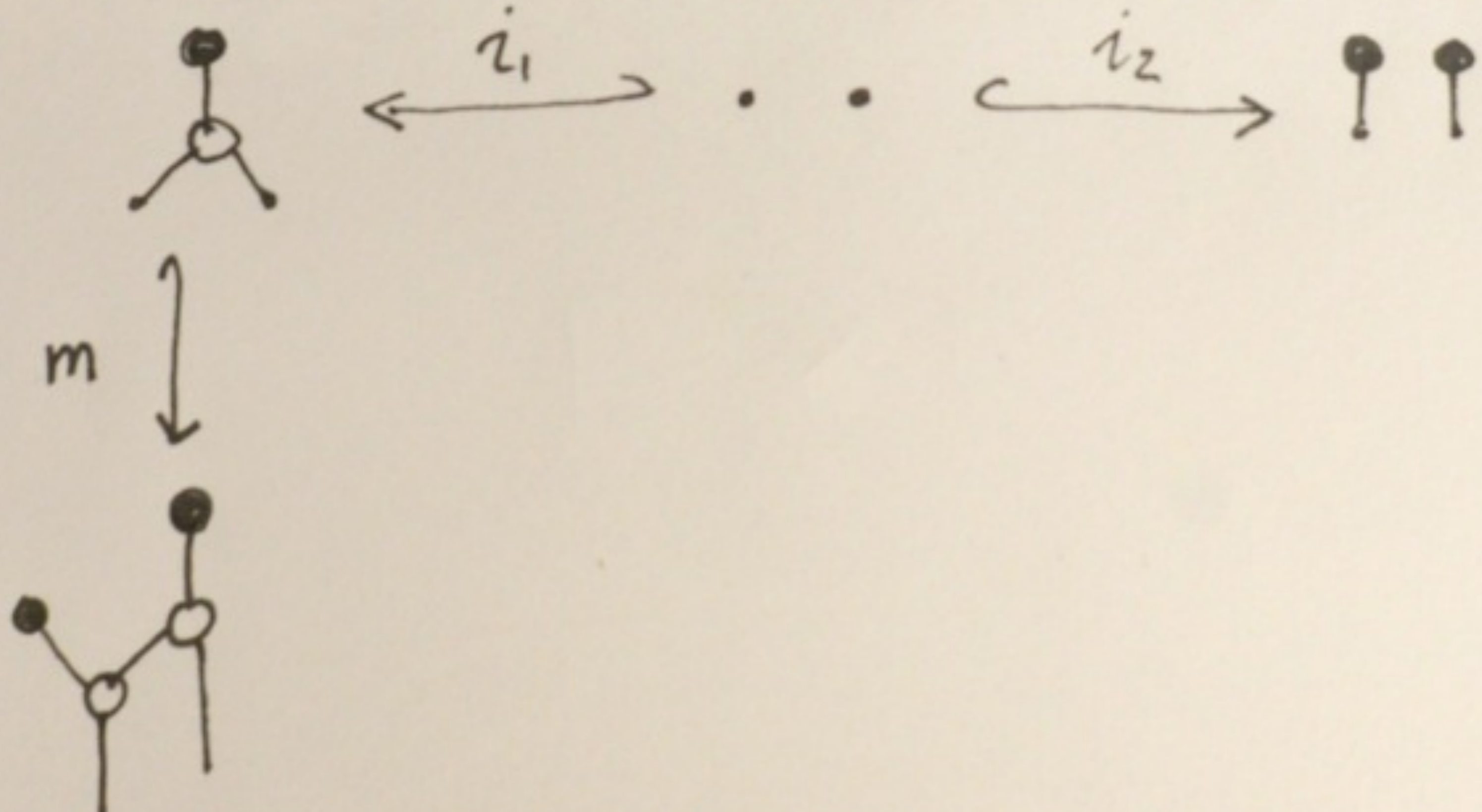
DPO REWRITING



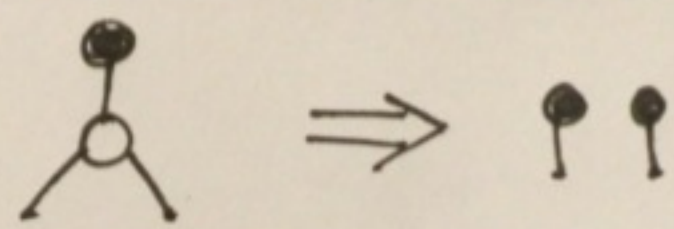


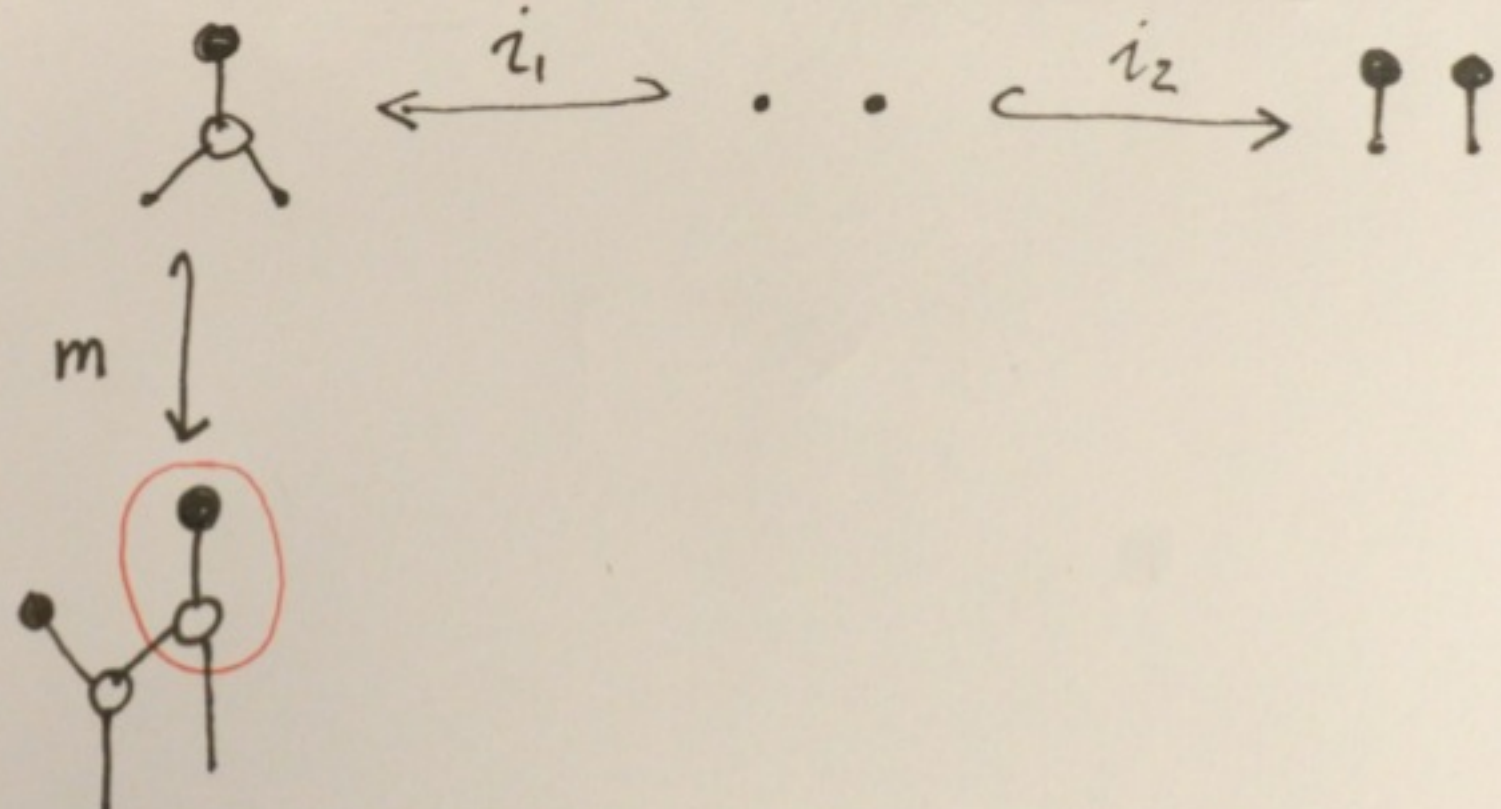
DPO REWRITING



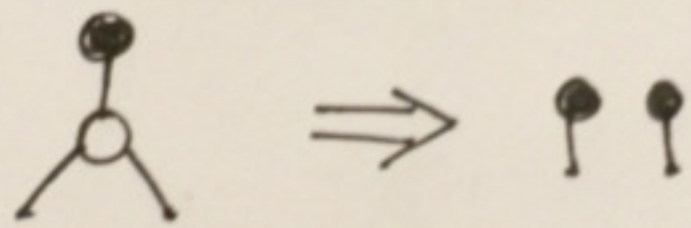


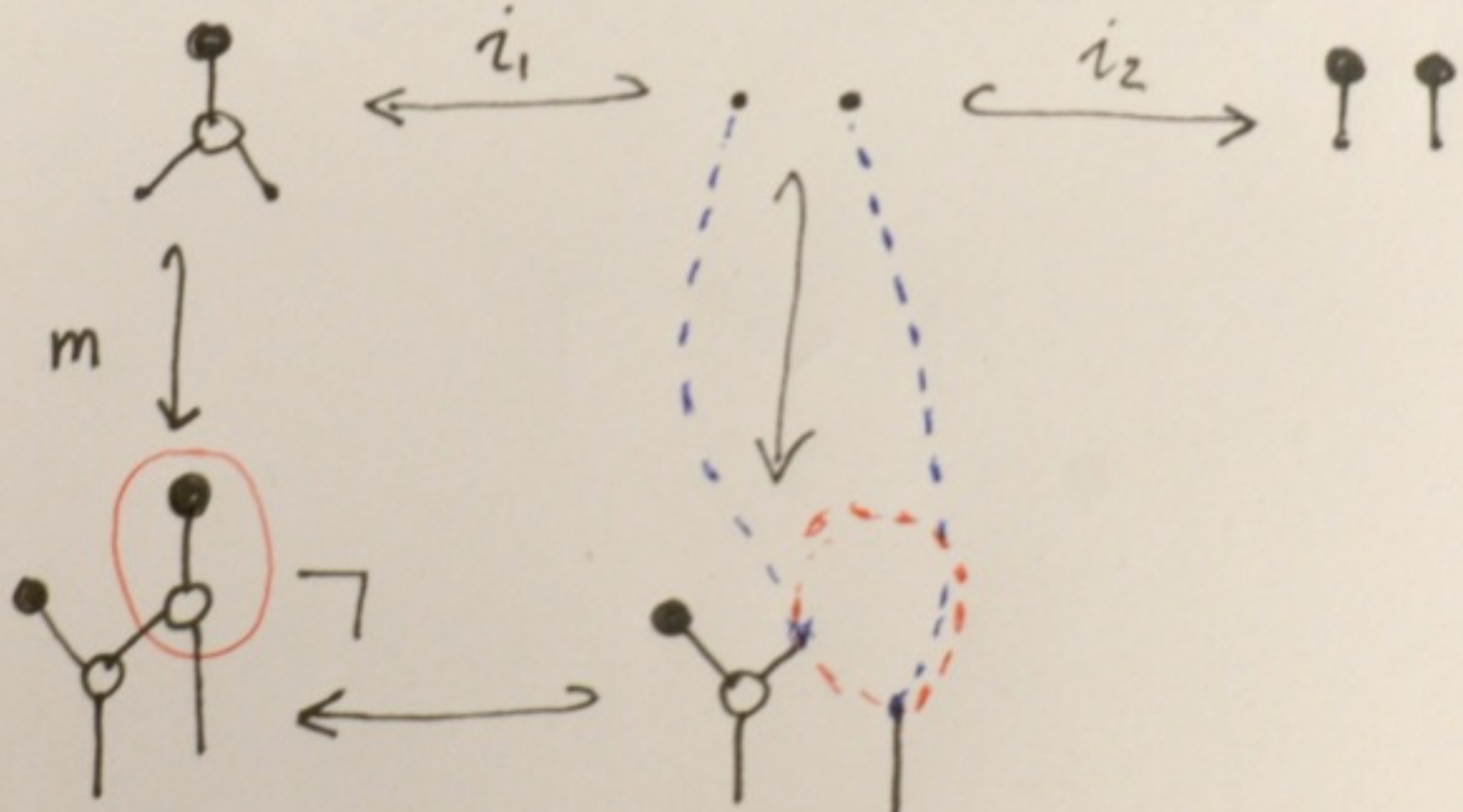
DPO REWRITING



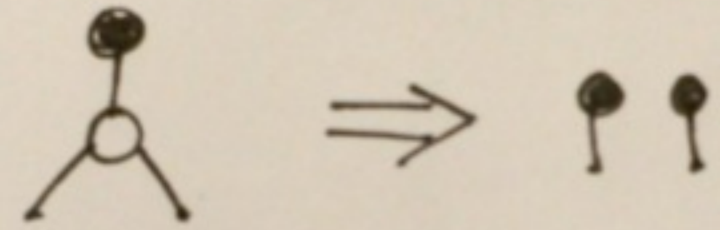


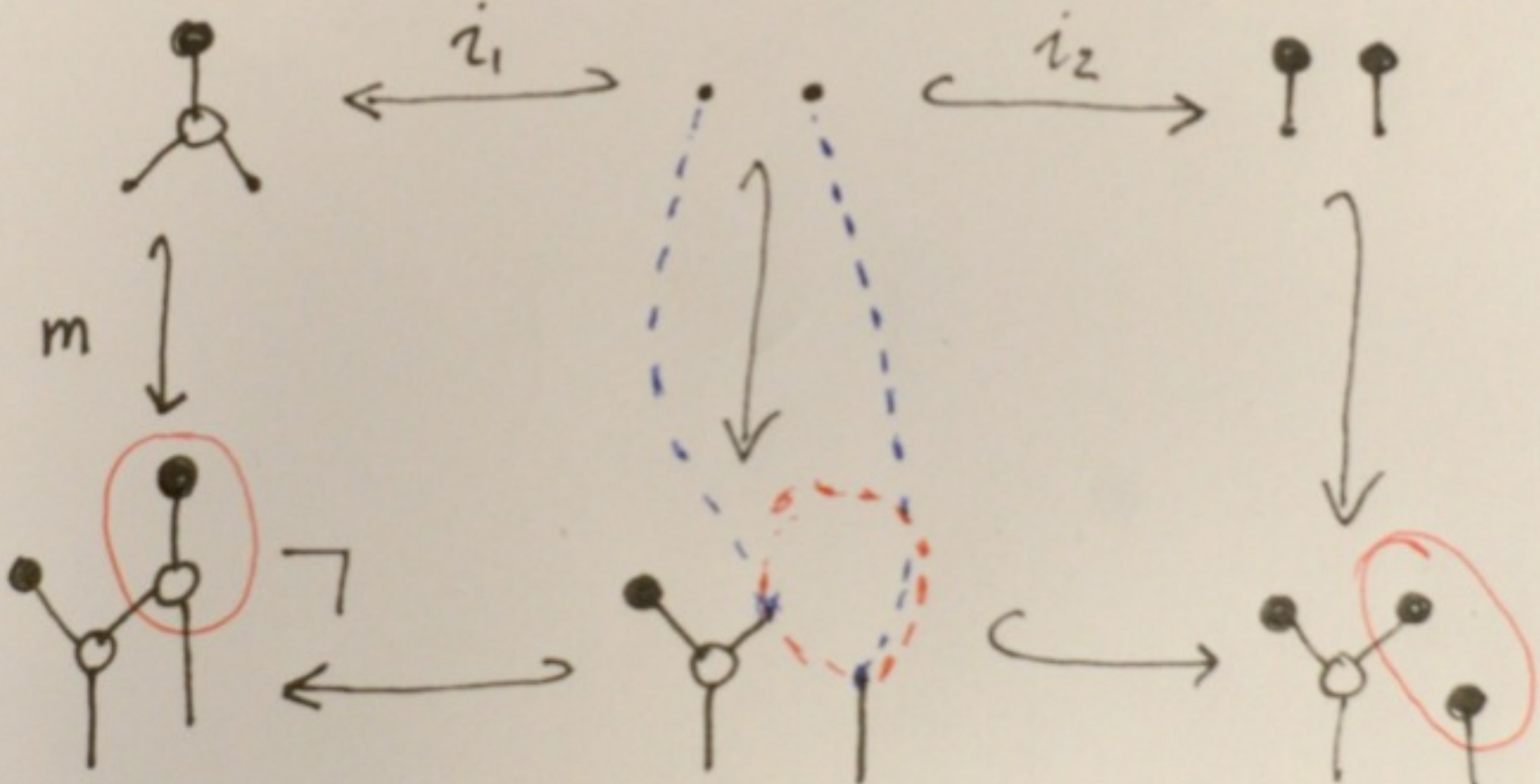
DPO REWRITING





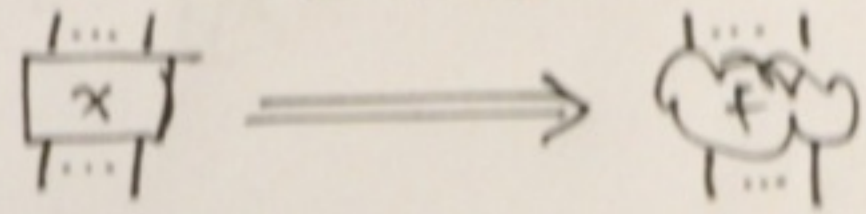
DPO Rewriting



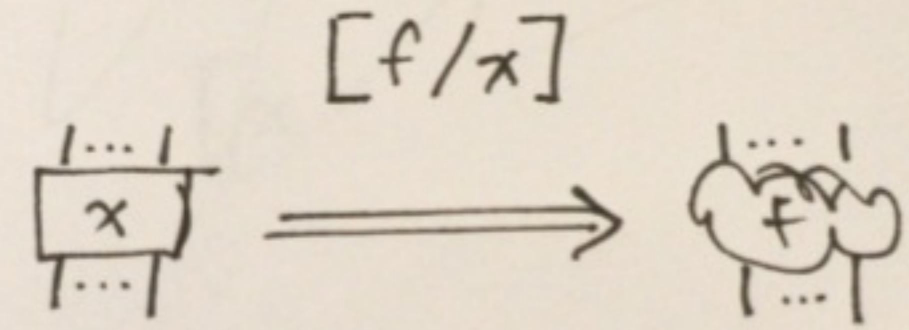


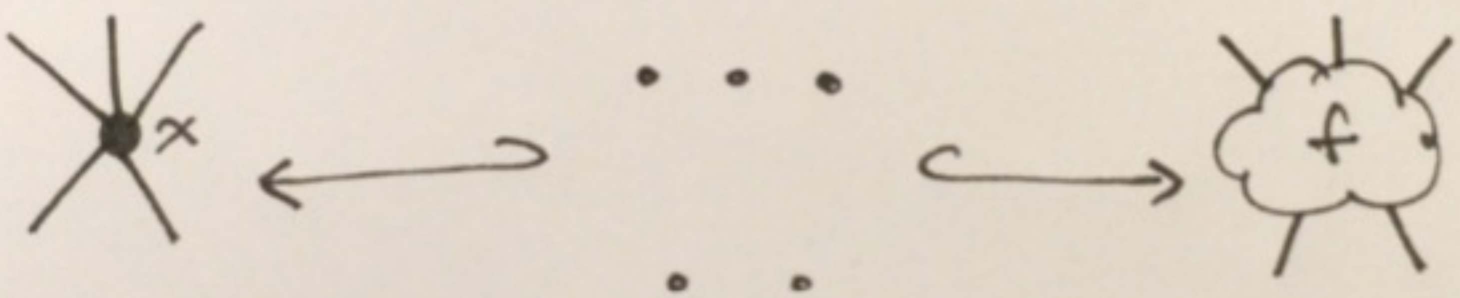
SUBSTITUTION VIA DPO

$[f/x]$



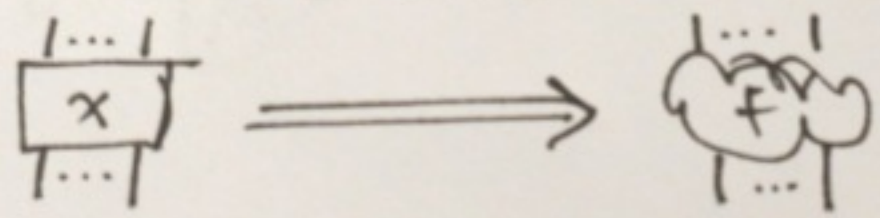
PLANE SUBSTITUTION VIA DPO

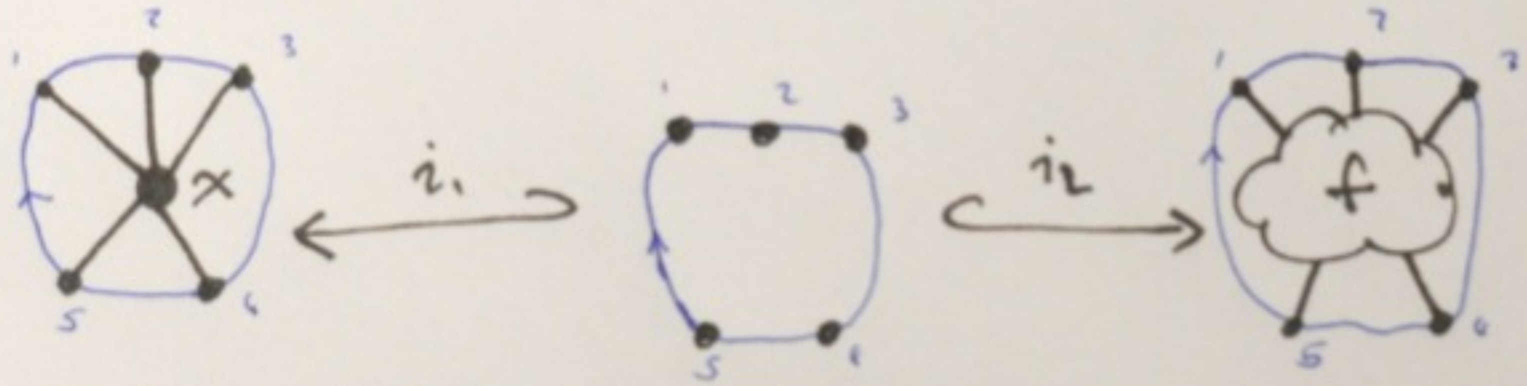




PLANE SUBSTITUTION VIA DPO

$[f/x]$

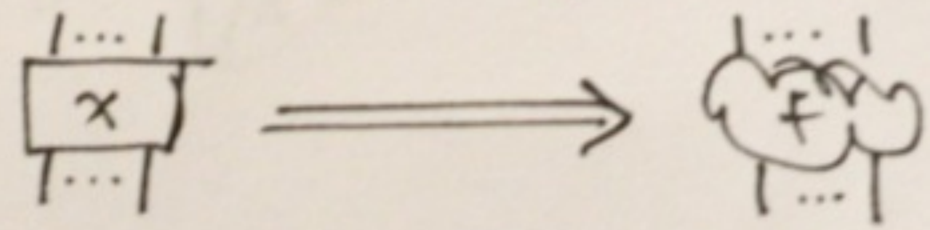


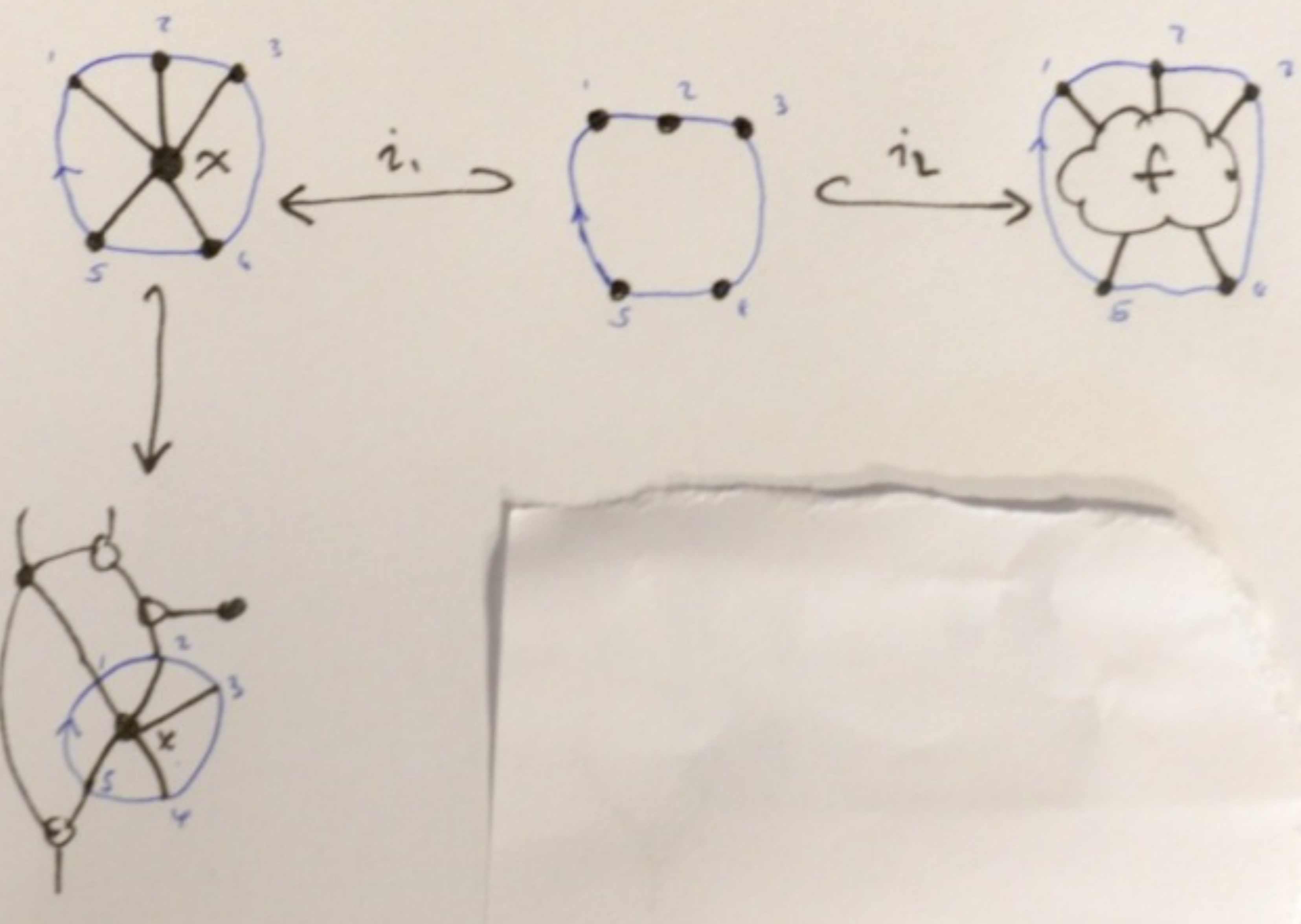


EMBEDDINGS BIJECTIVE ON THE BOUNDARY

PLANE SUBSTITUTION VIA DPO

$[f/x]$

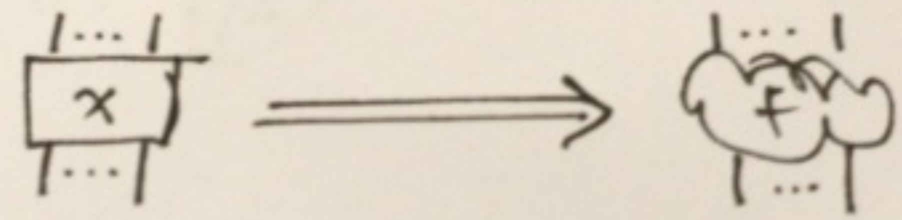


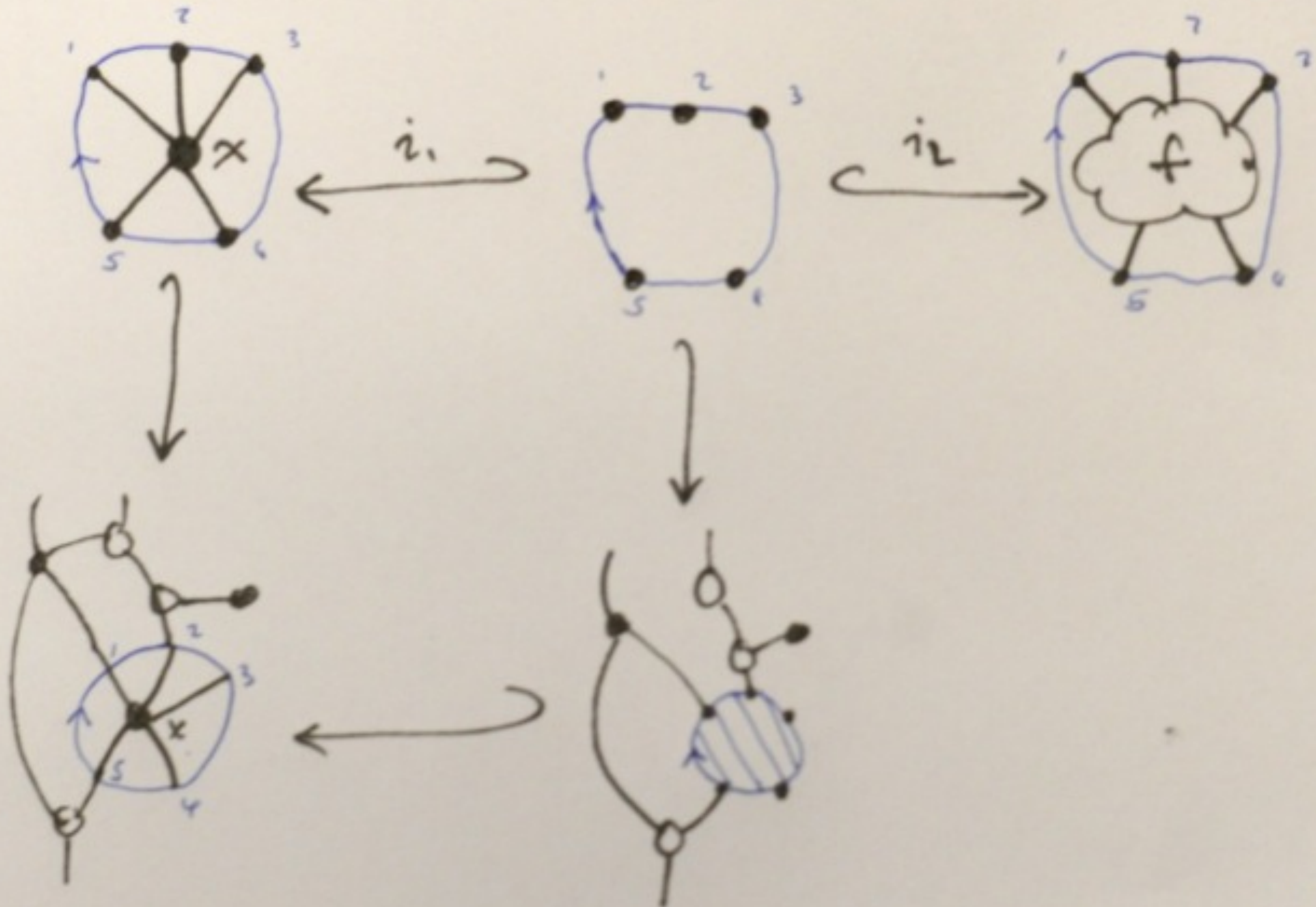


EMBEDDINGS
BIJECTIVE
ON THE
BOUNDARY

PLANE SUBSTITUTION VIA DPO

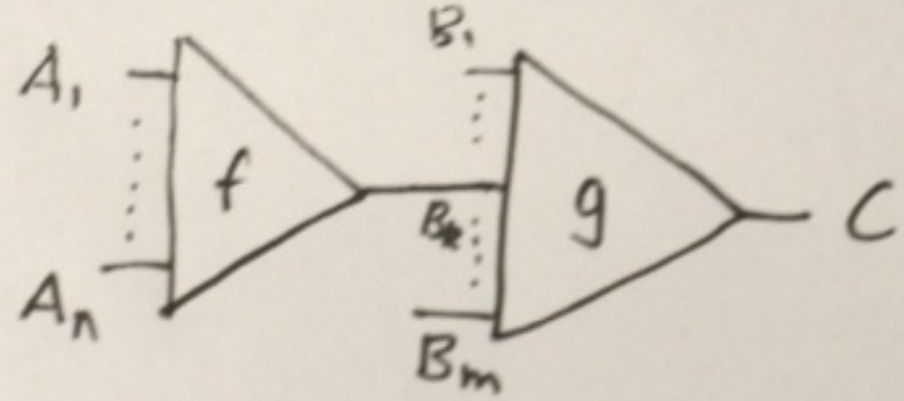
$[f/x]$





EMBEDDINGS
BIJECTIVE
ON THE
BOUNDARY

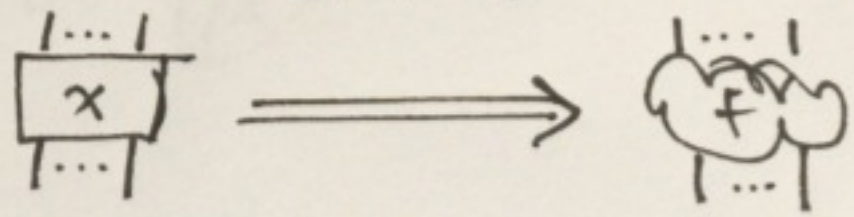
1. OPERADS



$$\frac{x_1 : A_1, \ldots, x_n : A_n \vdash f : B_k \qquad y_1 : B_1, \ldots, y_k : B_k, \ldots, y_m : B_m \vdash g : C}{y_1 : B_1, \ldots, x_1 : A_1, \ldots, x_n : A_n, \ldots, y_m : B_m \vdash g[f/x] : C} \text{CUT}$$

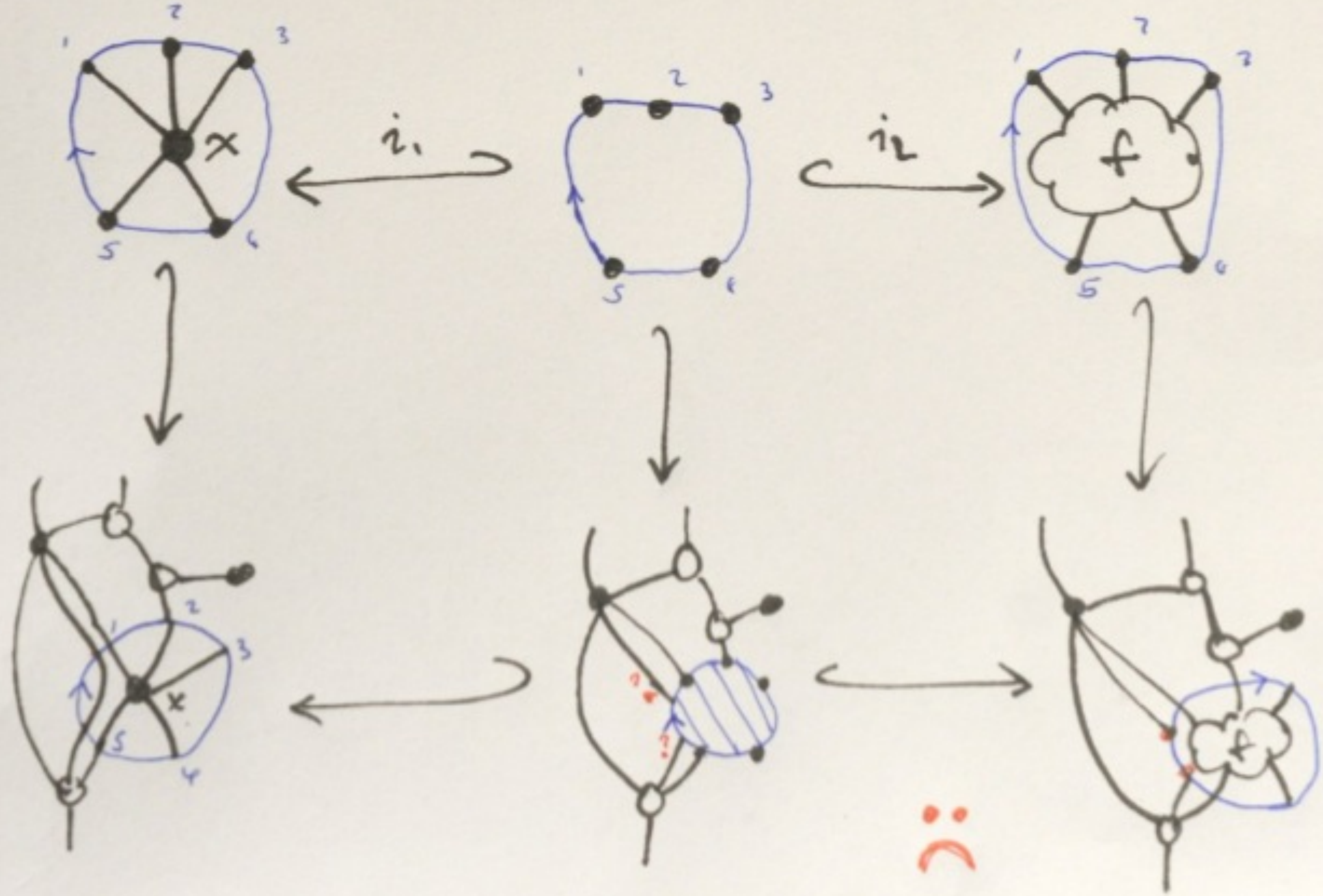
PLANE SUBSTITUTION VIA DPO

$[f/x]$



EMBEDDINGS BIJECTIVE ON THE BOUNDARY

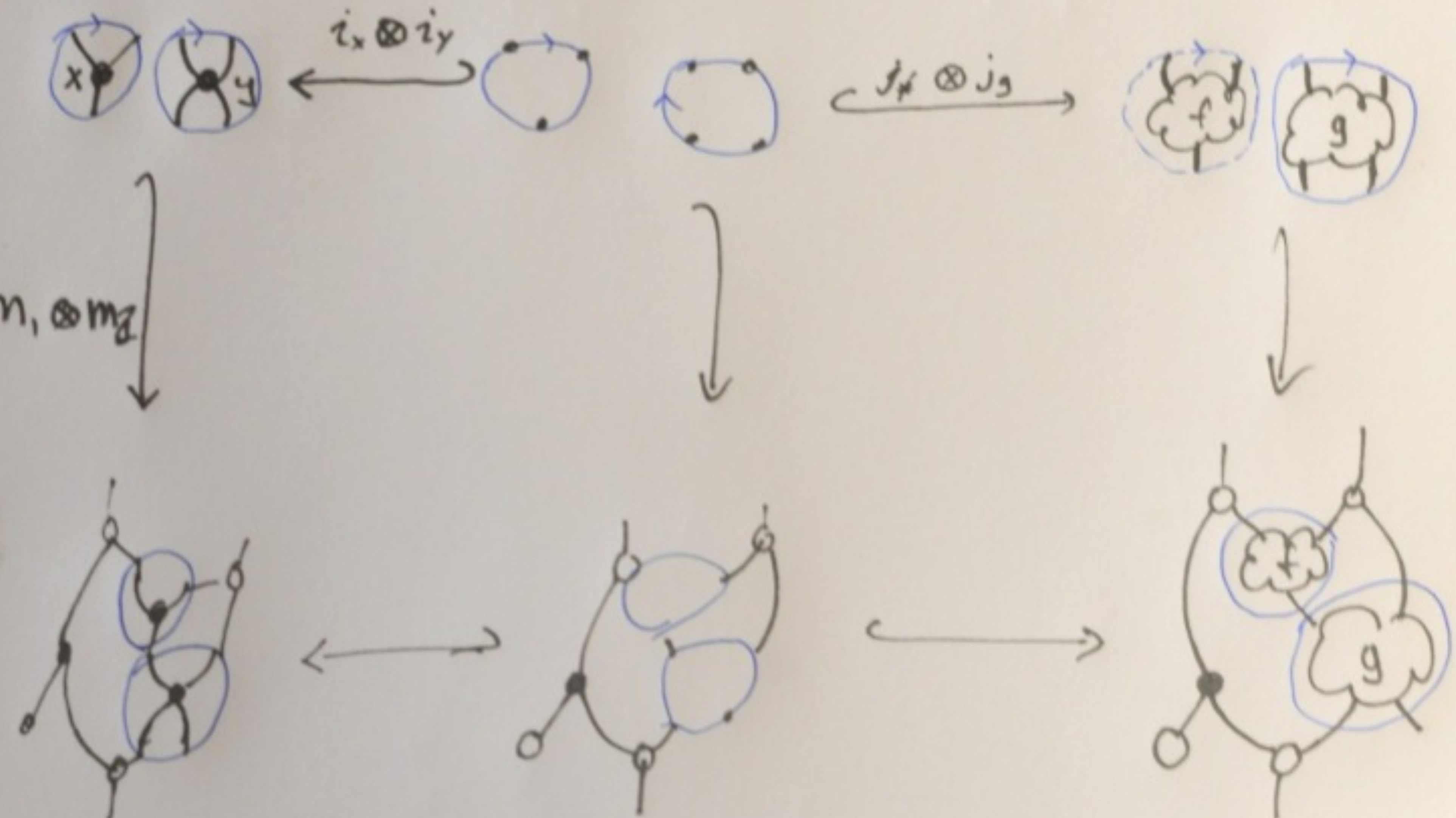
BAD!



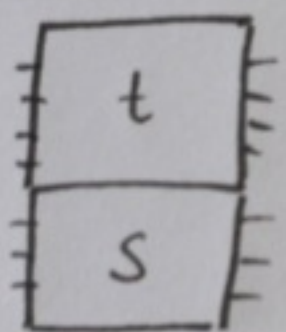
See Slofstra

Plane Substitution

$$[f/x,\ g/y]$$



LOGICAL RULES

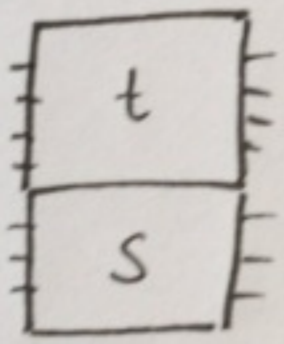


"tensor"

$$\frac{\bar{x} : \Delta \vdash t : A \qquad \bar{y} : \Gamma \vdash s : B}{\bar{x} : \Delta, \bar{y} : \Gamma \vdash t \otimes s : A \otimes B}$$

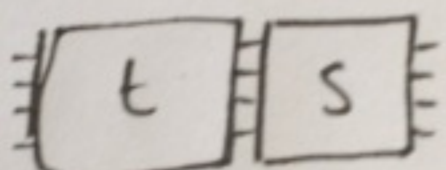
LOGICAL RULES

"tensor"



$$\frac{\bar{x} : \Delta \vdash t : A \qquad \bar{y} : \Gamma \vdash s : B}{\bar{x} : \Delta, \bar{y} : \Gamma \vdash t \otimes s : A \otimes B}$$

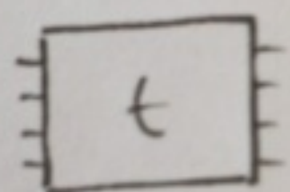
"composite"



$$\frac{\bar{x} : \Delta \vdash t : (n,m) \qquad \bar{y} : \Gamma \vdash s : (m,k)}{\bar{x} : \Delta, \bar{y} : \Gamma \vdash s \circ t : (n,k)}$$

Logical Rules

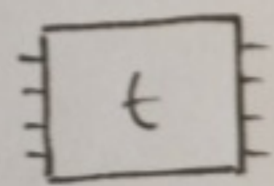
weakening



$$\frac{\bar{x} : \Delta \vdash t : A}{y : B , \bar{x} : \Delta \vdash t : A}$$

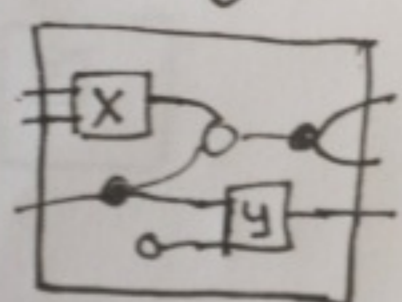
LOGICAL RULES

weakening



$$\frac{\bar{x} : \Delta \vdash t : A}{y : B, \bar{x} : \Delta \vdash t : A}$$

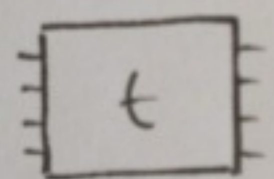
t



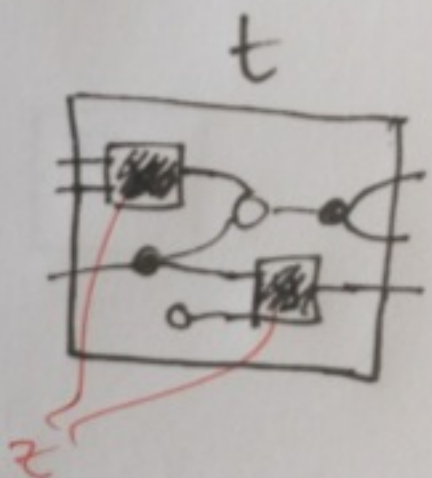
$$\frac{x : A, y : A \vdash t : B}{z : A \vdash t[z/x, z/y]}$$

LOGICAL RULES

weakening



$$\frac{\bar{x} : \Delta \vdash t : A}{y : B, \bar{x} : \Delta \vdash t : A}$$



$$\frac{x : A, y : A \vdash t : B}{z : A \vdash t[z/x, z/y]}$$

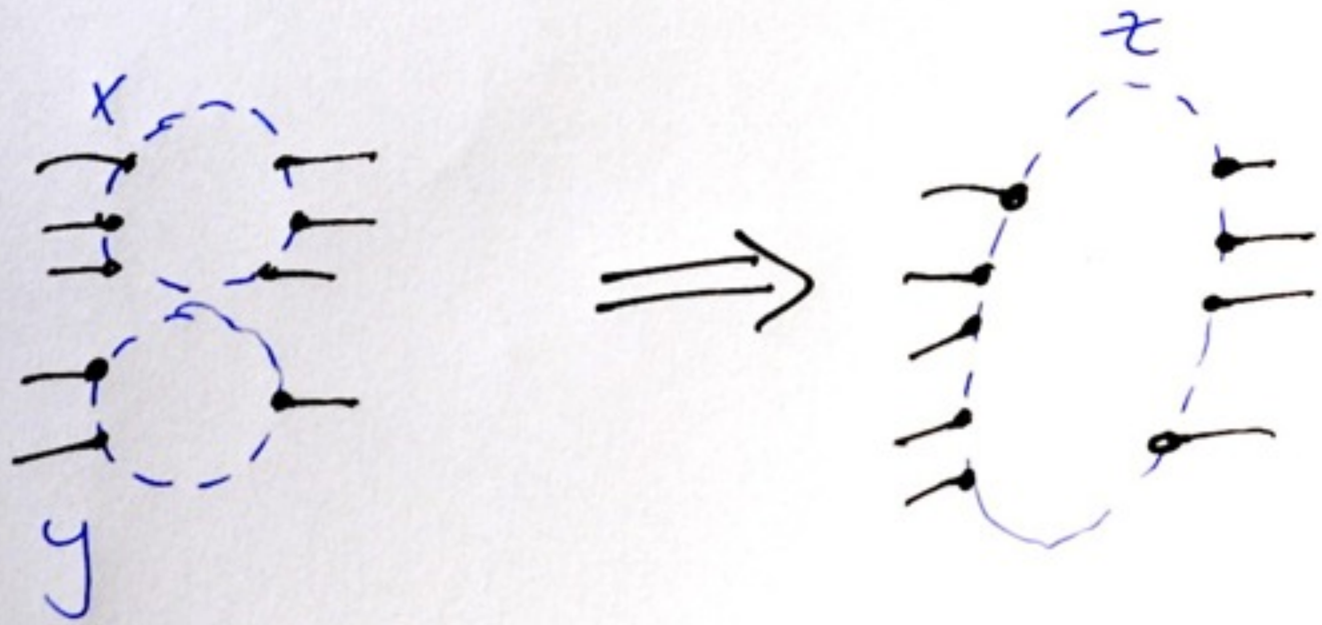
"CONTRACTION"

NOT ALLOWED

$$\frac{x : A,\ y : B \vdash t : C}{z : A \otimes B \vdash \text{let } z = x \otimes y \text{ in } t : C}$$

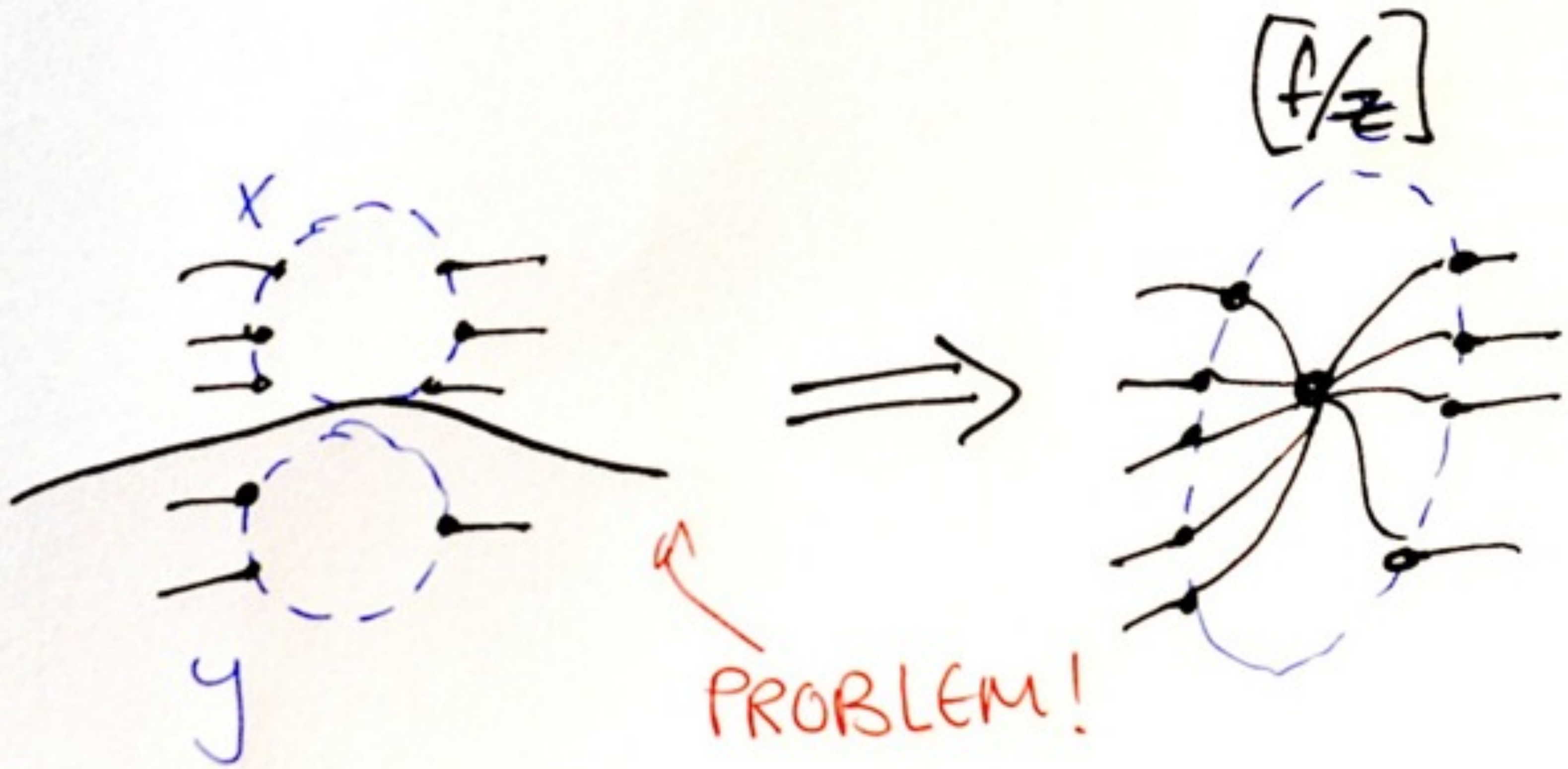
Not Allowed

$$\frac{x : A,\ y : B \ \vdash\ t : C}{z : A \otimes B \ \vdash\ \text{let } z = x \otimes y \text{ in } t \ : C}$$



NOT ALLOWED

$$\frac{x:A,\, y:B \vdash t:C}{z:A\otimes B \vdash \text{let } z = x\otimes y \text{ in } t \;:\; C}$$

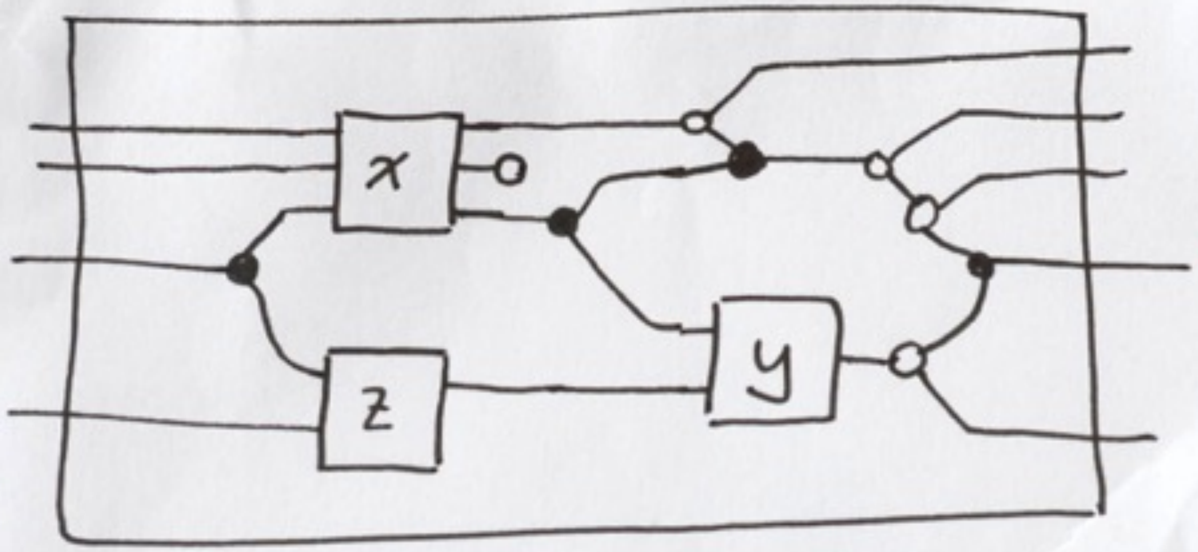


Summary Pt. 1

1. Substitution and operations in underlying PRO form a "monoidal++" operad.

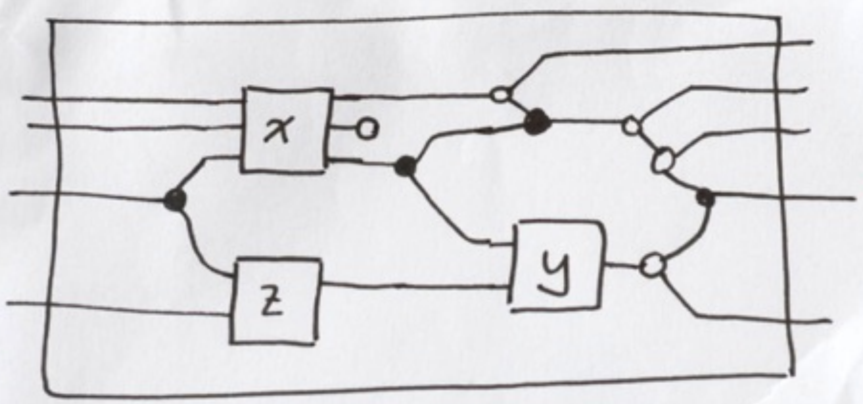
2. Variable manipulations give a cocommutative comonoid.
But don't allow $\otimes$.

3. PATTERN-MATCHING.

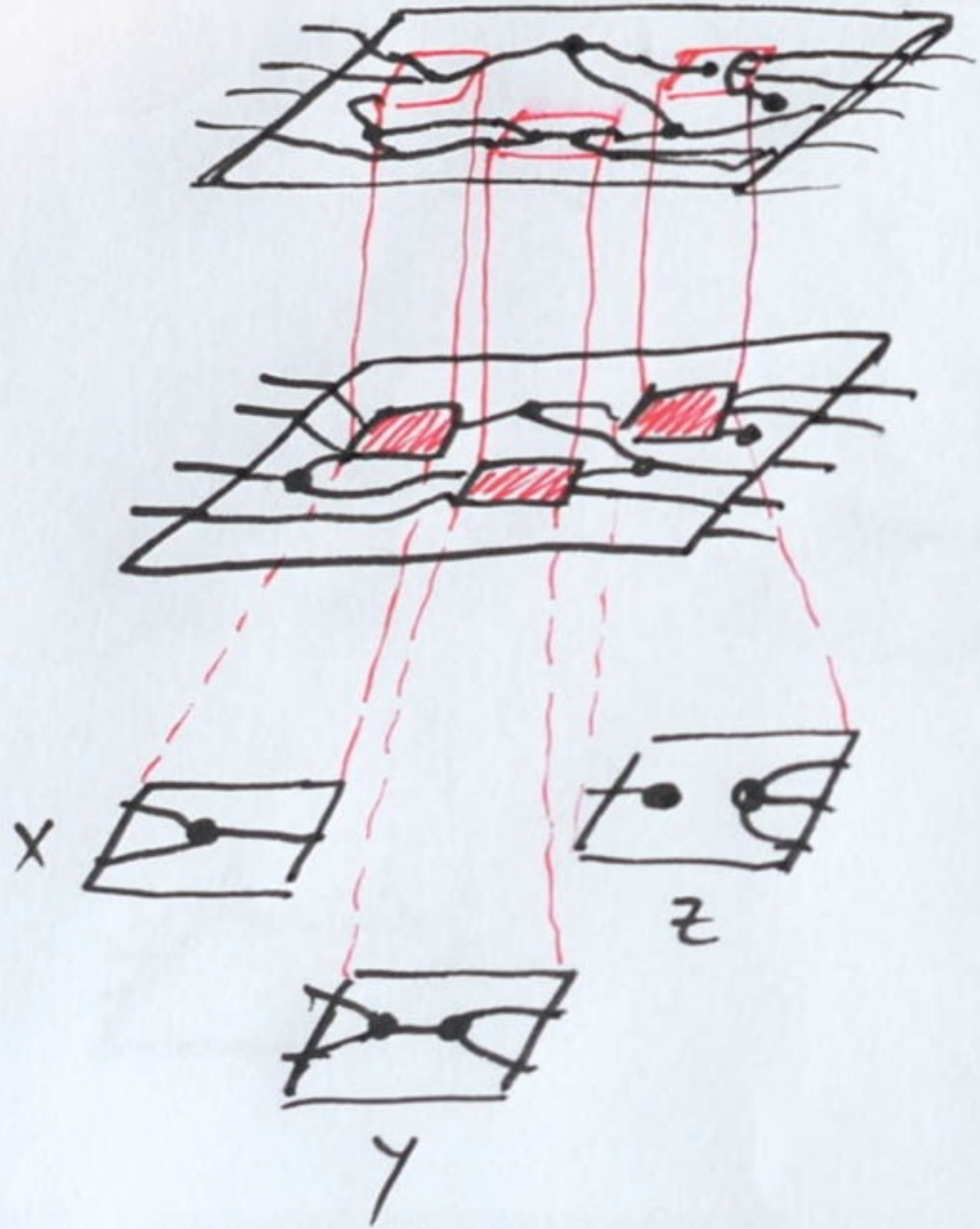


$x:(3,3),\ y:(2,1),\ z:(2,1) \vdash f:(4,5)$

3. PATTERN-MATCHING



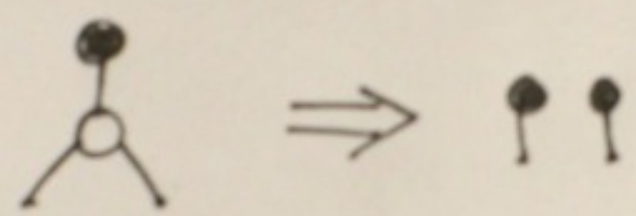
$x:(3,3),\ y:(2,1),\ z:(2,1) \vdash f:(4,5)$



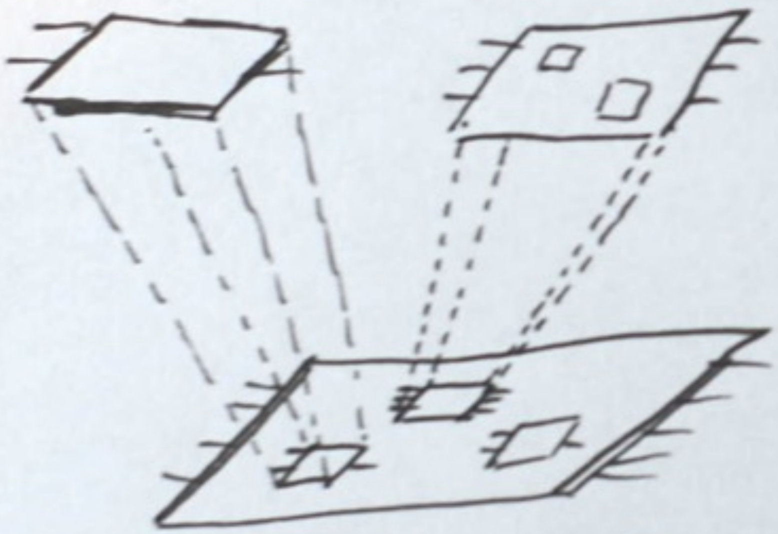
3. Pattern Matching

$$\begin{tikzcd} V \arrow[d, hook, "m"'] & \partial \arrow[l, hook', "i"'] \\ G & \end{tikzcd}$$

DPO REWRITING



2. OPERADS FROM PRO.S.

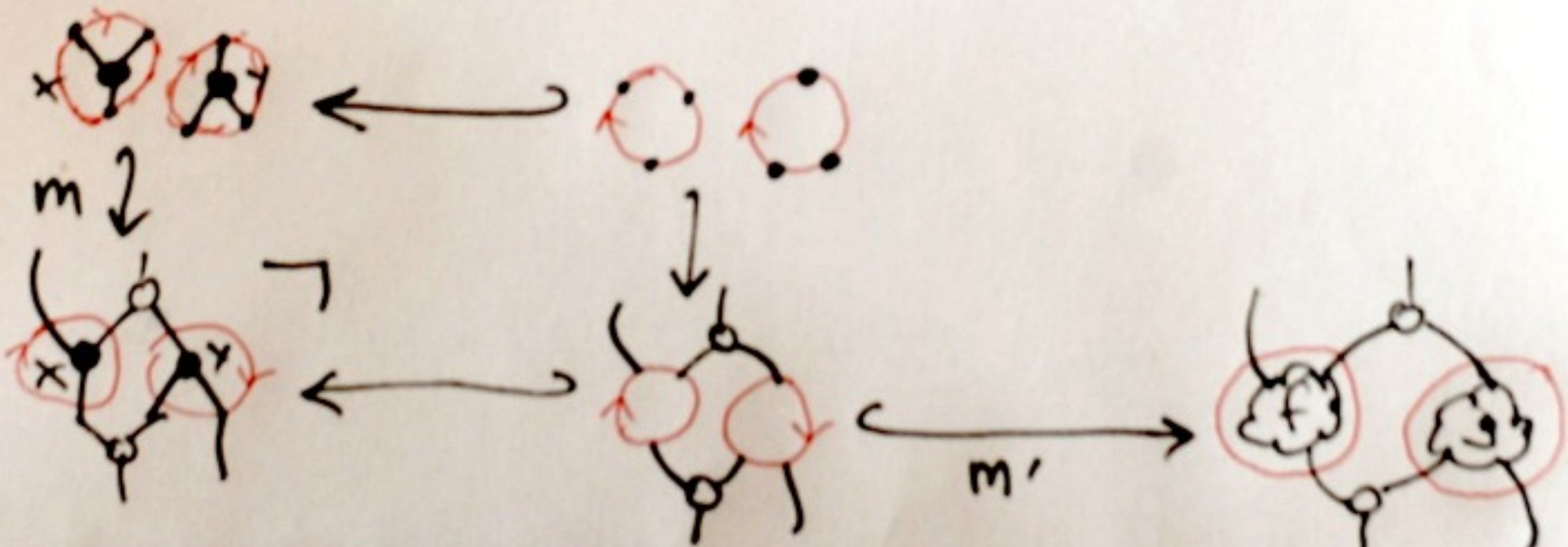


$f[g/x][h/z]$

3. Pattern Matching

$$
\begin{tikzcd}
V \arrow[d, hook, "m"'] & \partial \arrow[l, hook', "i"'] \arrow[d, hook] & \\
G \arrow[ur, phantom, very near start, "\urcorner"] & G\setminus V \arrow[l, hook'] \arrow[r, hook, "m'"'] & H
\end{tikzcd}
$$

suppose this exists.

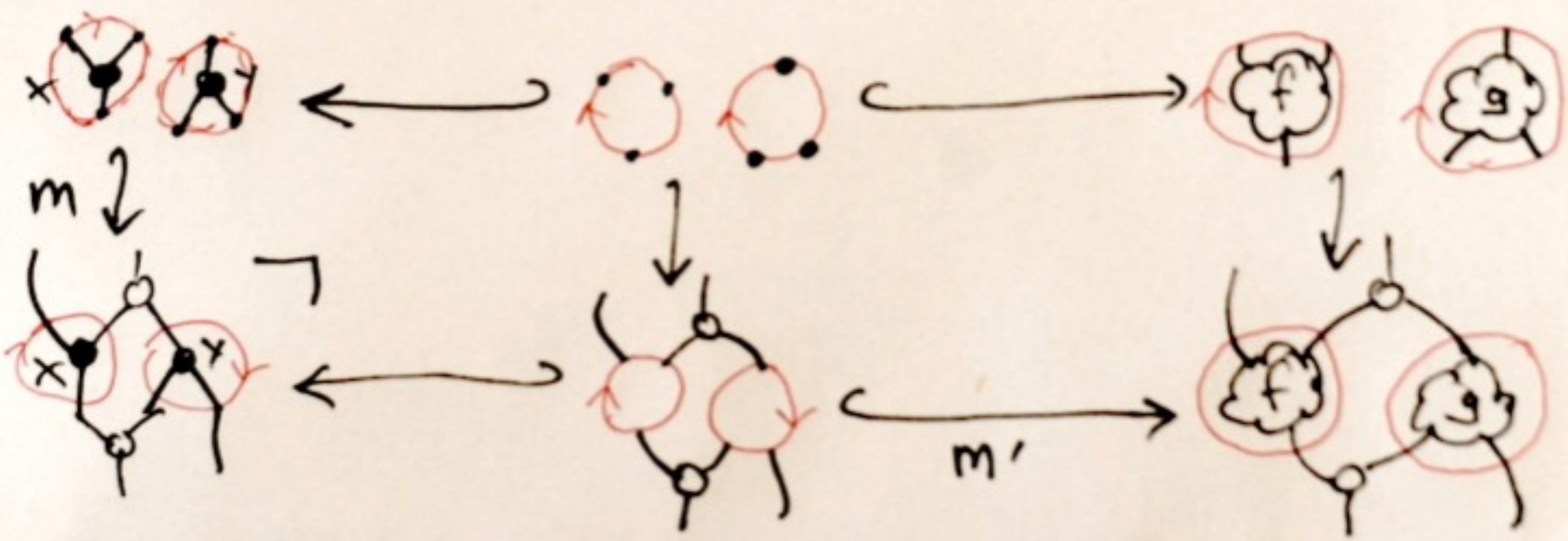


3. Pattern Matching

compute this pushout complement.

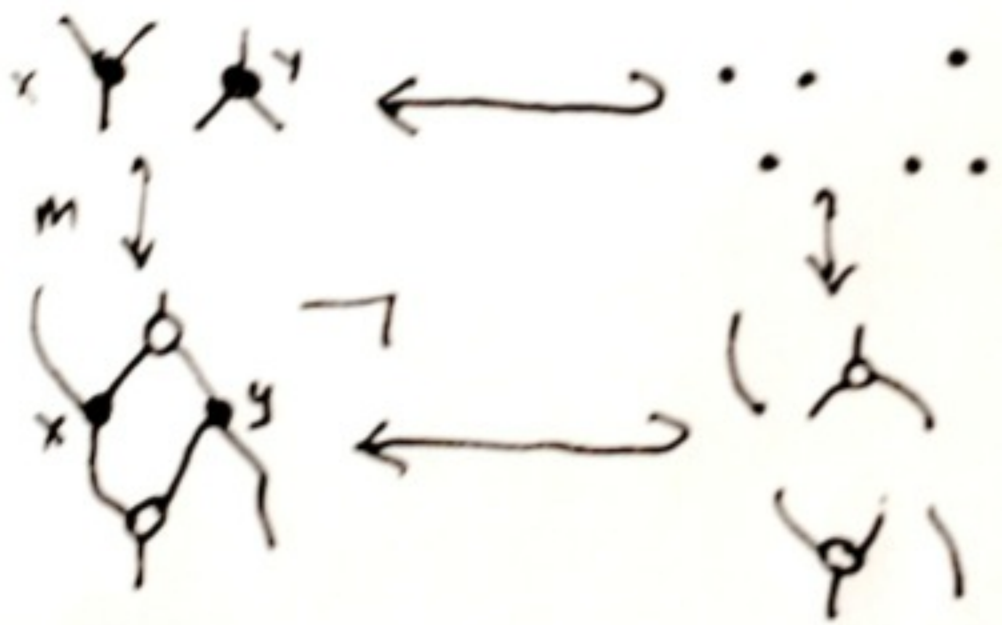
$$\begin{tikzcd}
V \arrow[d, hook, "m"'] & \partial \arrow[l, hook', "i"'] \arrow[r, hook, "i'"] \arrow[d, hook] & H \setminus (G \setminus V) \arrow[d, hook] \\
G \arrow[ur, phantom, very near start, "\urcorner"] & G \setminus V \arrow[l, hook'] \arrow[r, hook, "m'"'] & H \arrow[ul, phantom, very near start, "\ulcorner"]
\end{tikzcd}$$

suppose this exists.



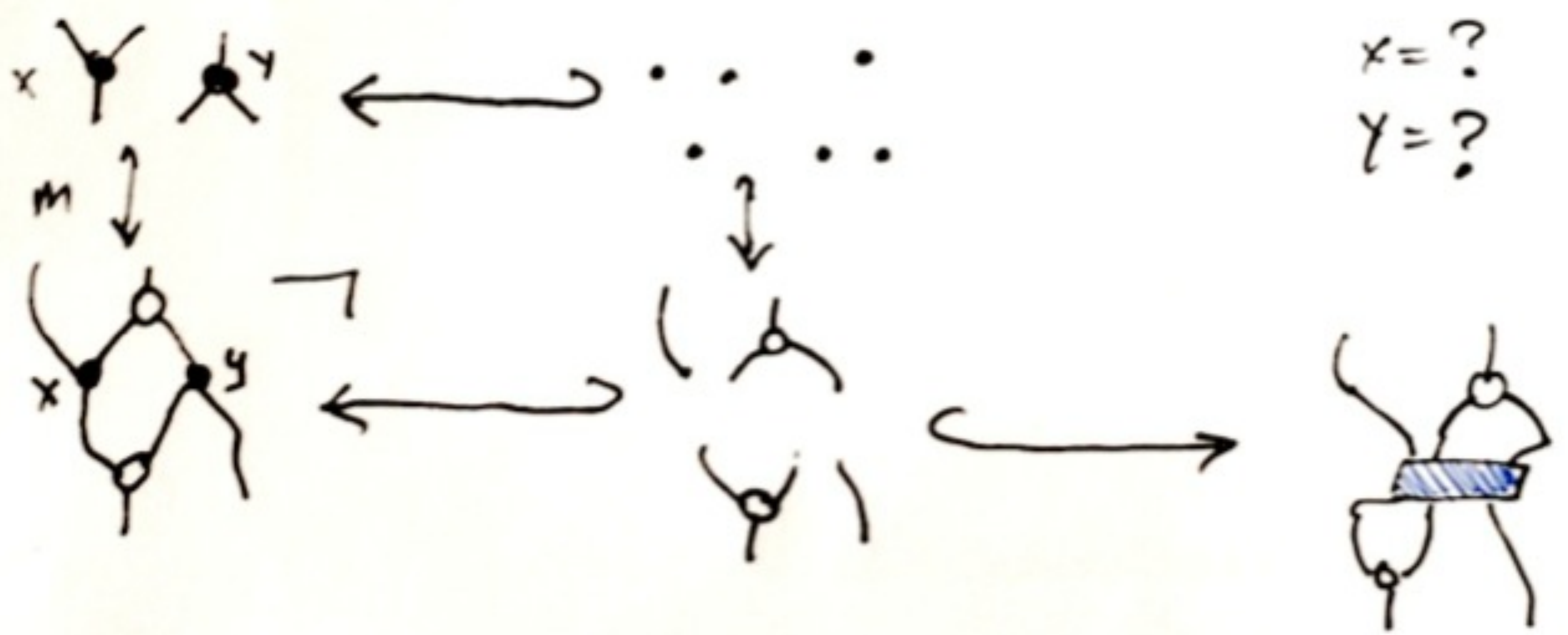
3. PATTERN MATCHING

NOTE PRESERVATION OF BOUNDARY CURVE IS ESSENTIAL:



3. PATTERN MATCHING

NOTE PRESERVATION OF BOUNDARY CURVE IS ESSENTIAL:

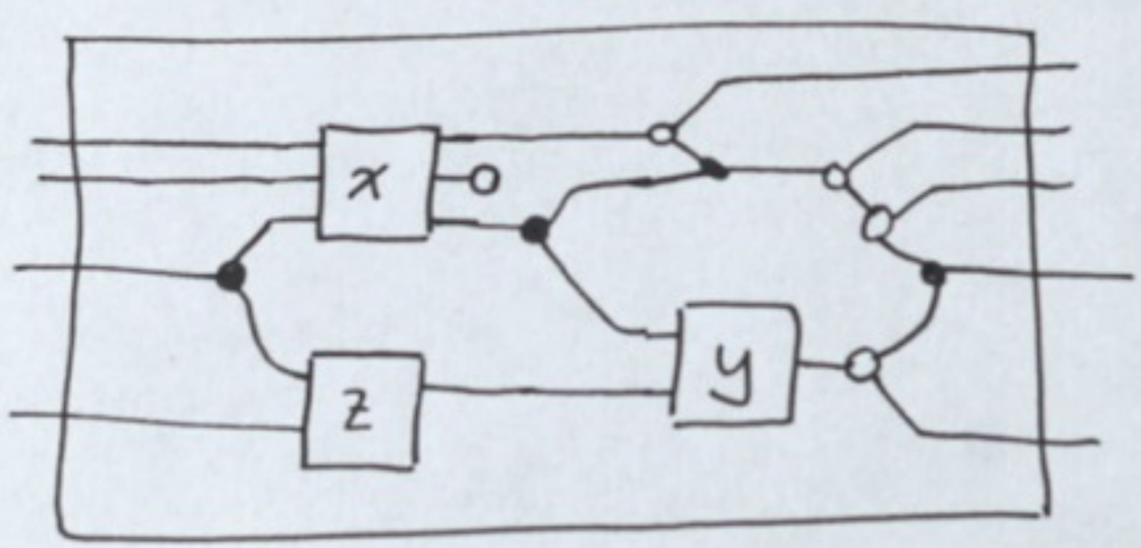


2. MAKING AN OPERAD FROM A PRO

- Let (Σ, E) be a presentation of a PRO.
- Adjoin "enough" new generators $x : m \to n$ for every $m, n \in \mathbb{N}$.

Variables.

- Then $(\Sigma + Var, E)$ is again a PRO with (term) variables.



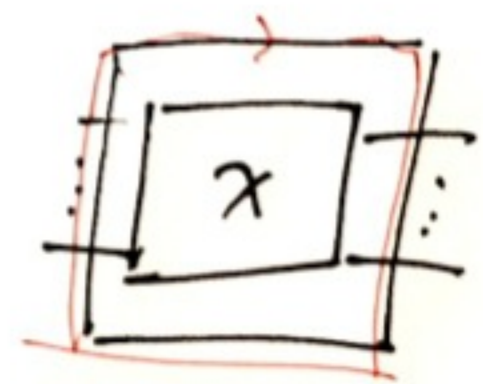
Assume variables only occur once (for now)

$$x:(3,3),\ y:(2,1),\ z:(2,1) \vdash f:(4,5)$$

Putting it together

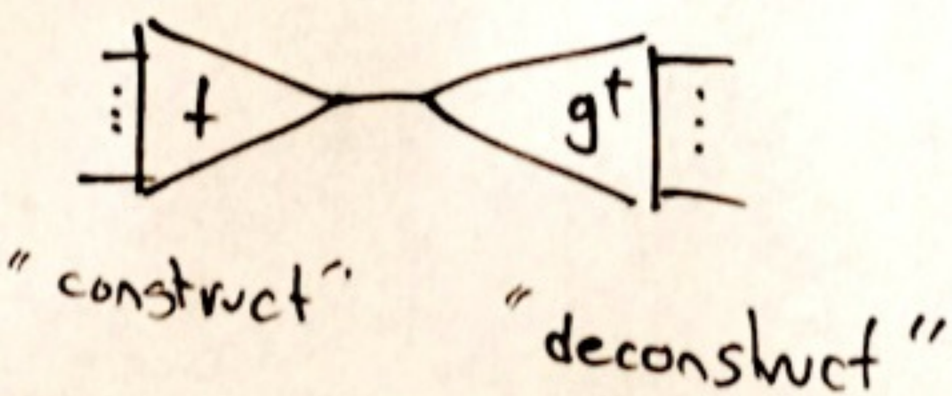
Identities are same in operad and cooperad

$$\overline{x : A \vdash x : A}$$



PUTTING IT TOGETHER

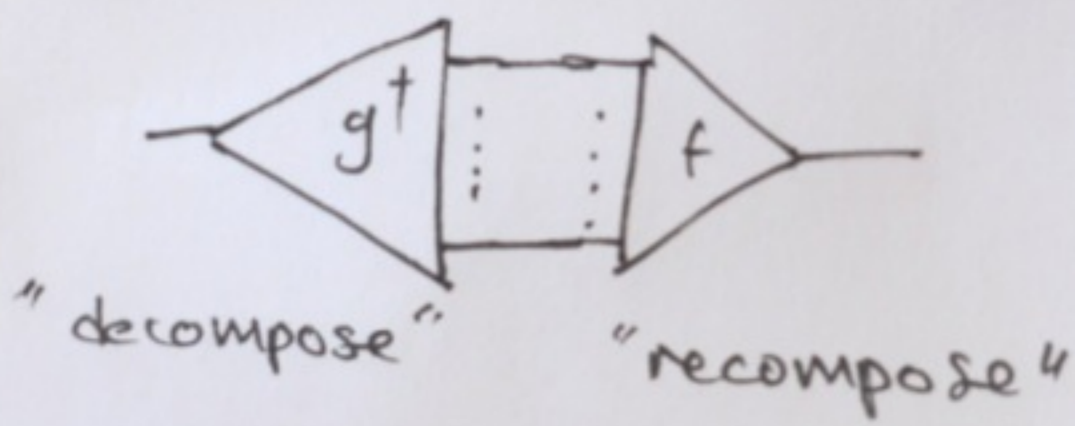
Composing like this



makes sense.

PUTTING IT TOGETHER

Composing like this



makes sense.

PUTTING IT TOGETHER

STRING DIAGRAMS WITH VARIABLES
FORM A $\begin{cases} \underline{\text{COMPUTAD}} \\ \underline{\text{POLYCATEGORY}} \end{cases}$

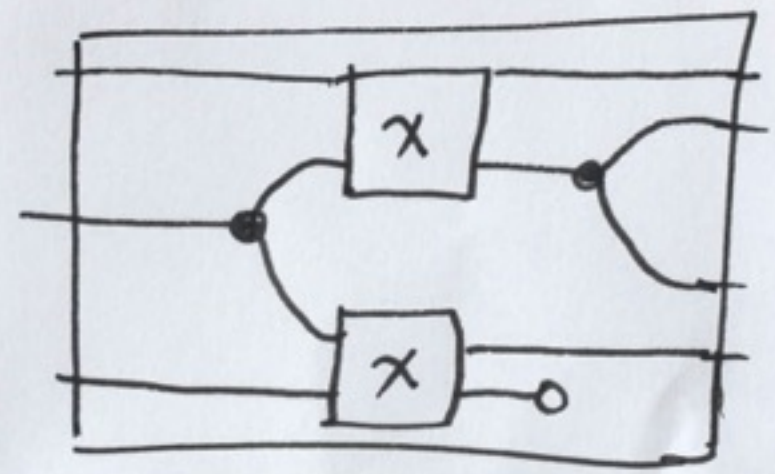
OF MANY-TO-MANY, DIAGRAM TRANSFORMATIONS
PARTIAL.

(WITH THE MIX RULE)

4. Ditching Linearity

$$\frac{\Delta \vdash t : A,\ t' : A}{\Delta \vdash t'' : A}\ \text{Contraction}$$

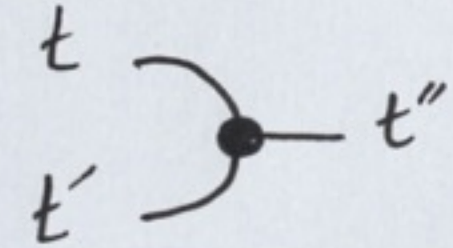
$$\frac{\Delta \vdash t : A}{\Delta \vdash t : A,\ t'' : B}\ \text{Weakening}$$



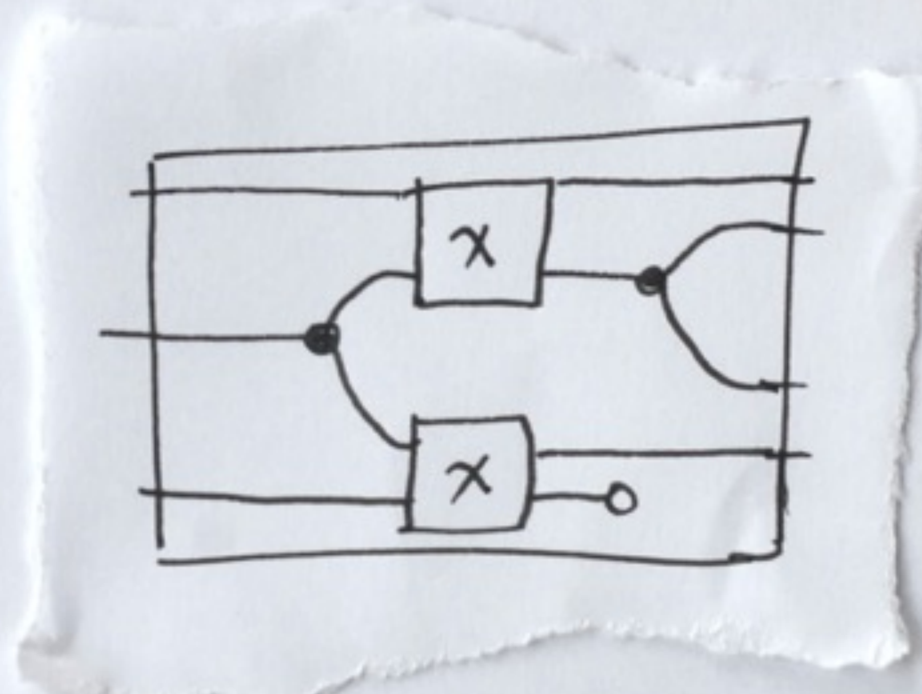
4. Ditching Linearity

$$\frac{\Delta \vdash t : A,\ t' : A}{\Delta \vdash t'' : A}\ \text{Contraction}$$

with $t = t' = t''$



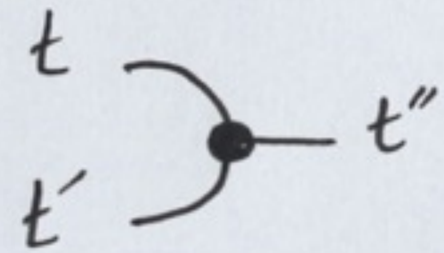
$$\frac{\Delta \vdash t : A}{\Delta \vdash t : A,\ t'' : B}\ \text{Weakening}$$



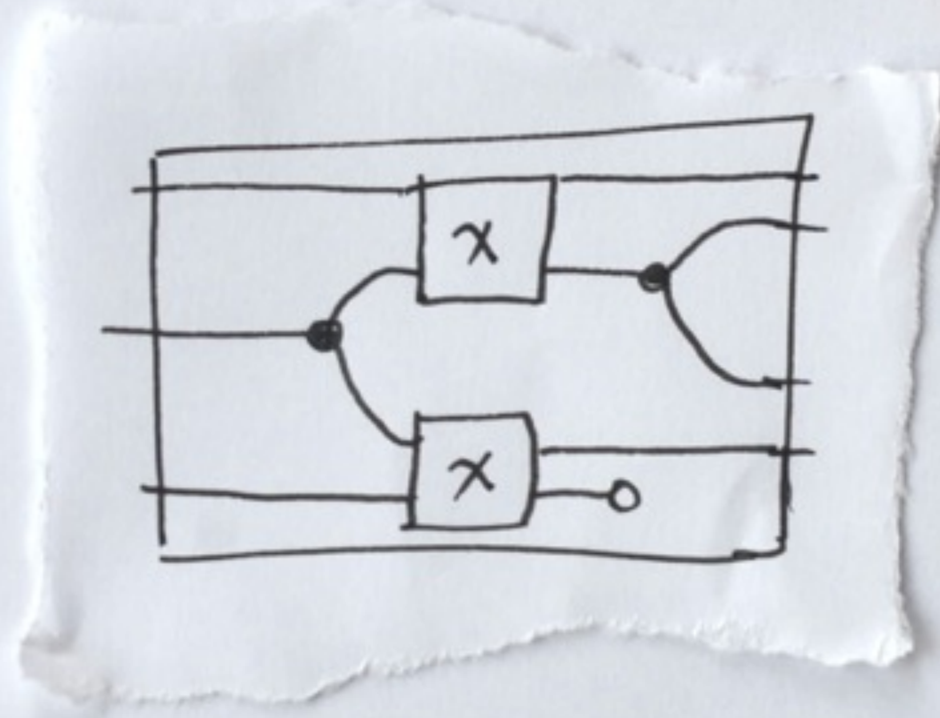
4. Ditching Linearity

$$\frac{\Delta \vdash t:A,\ t':A}{\Delta \vdash t'':A}\ \text{Contraction}$$

with $t = t' = t''$



$$\frac{\Delta \vdash t:A}{\Delta \vdash t:A,\ t'':B}\ \text{Weakening}$$



4. Ditching Linearity

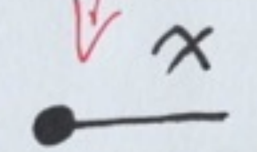
$$\frac{\Delta \vdash t : A,\ t' : A}{\Delta \vdash t'' : A}\ \text{Contraction}$$

with $\cancel{t = t' = t''}$

$t'' = \text{M.C.U.}(t, t')$

Commutative Monoid!*

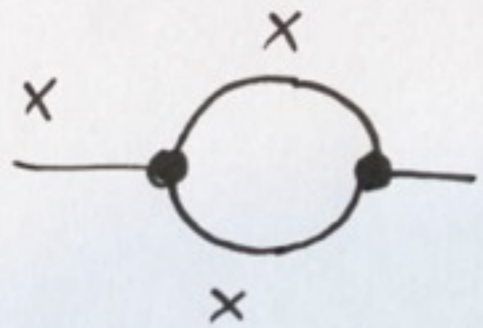
$$\frac{\Delta \vdash t : A}{\Delta \vdash t : A,\ t'' : B}\ \text{Weakening}$$



where x is a fresh variable.

4. DITCHING LINEARITY.

SPECIAL!

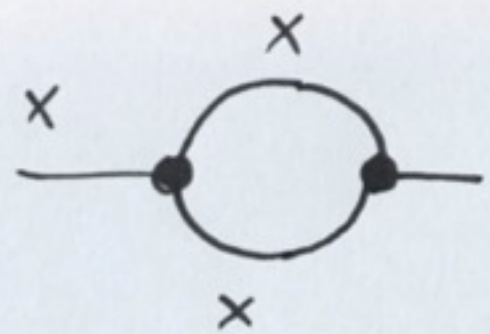


MGU(x, x) = x

=

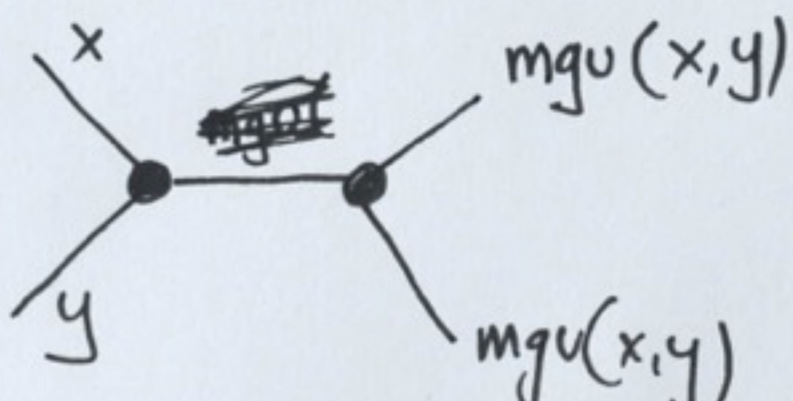
4. DITCHING LINEARITY.

SPECIAL!

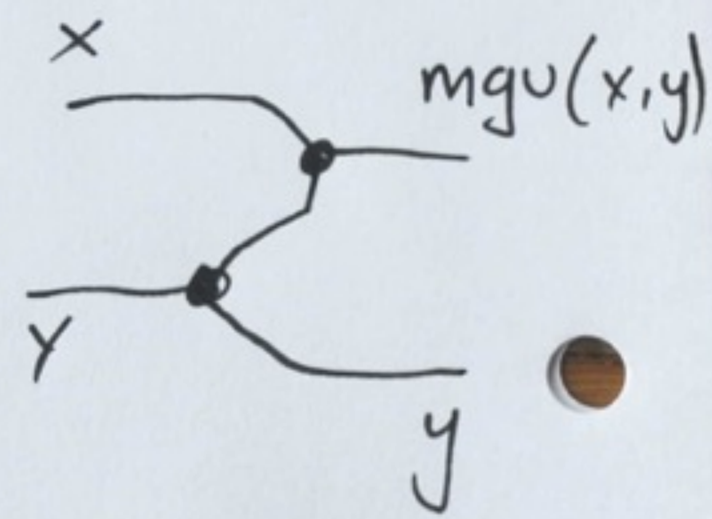


$MGU(x,x) = x$

$=$



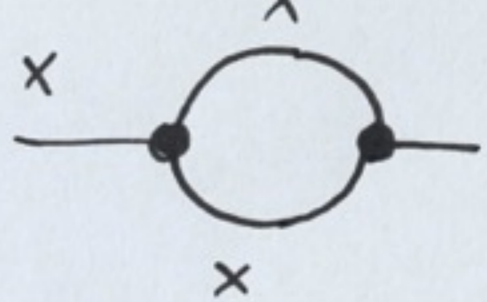
$\neq$



NOT FROB :(

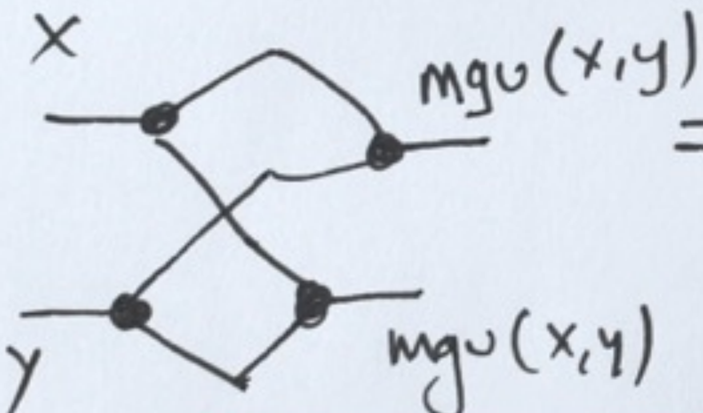
4. DITCHING LINEARITY.

SPECIAL!

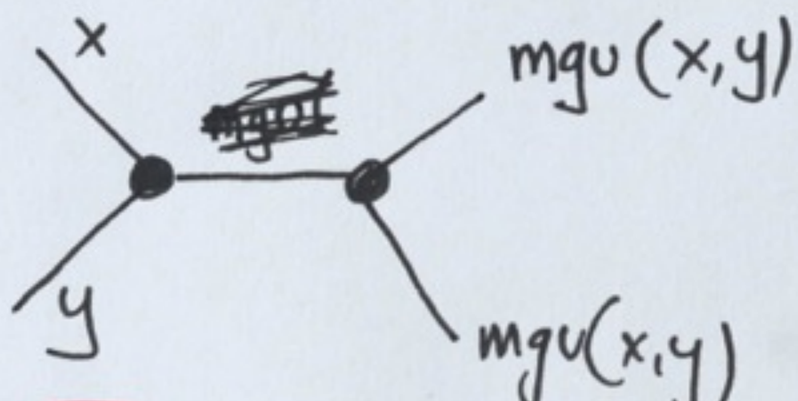


$MGU(x,x) = x$

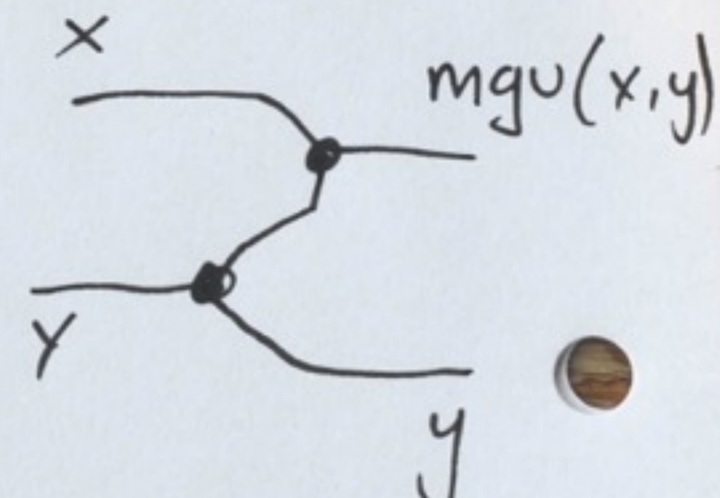
$=$



$=$



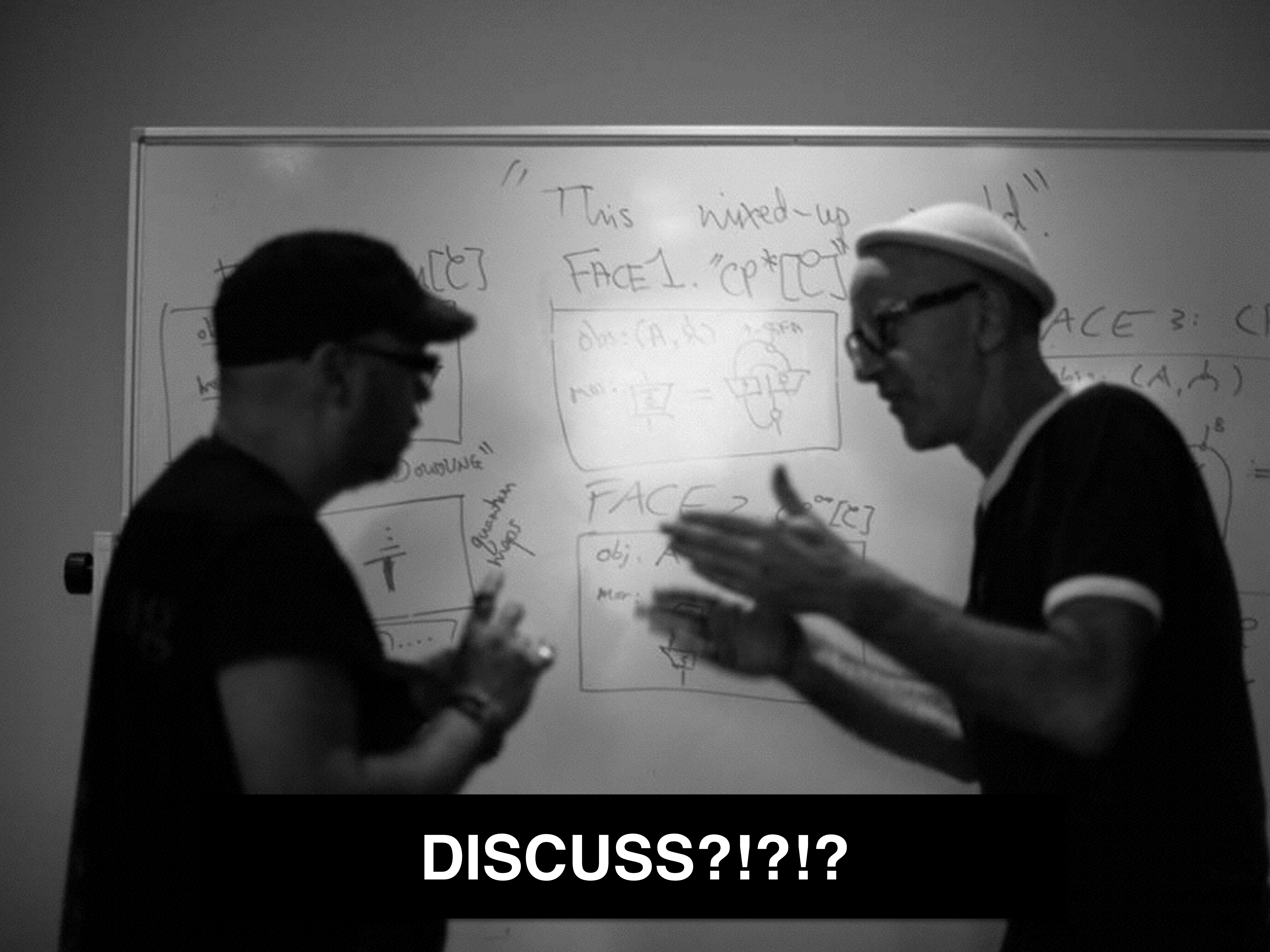
$\neq$



IS BIALGEBRA! :)

NOT FROB :(

HAPPY BIRTHDAY DUSKO!



DISCUSS?!?!?