

Logical relations for C(P)U

SCHOOL OF ARTIFICIAL INTELLIGENCE
UNIVERSITY OF EDINBURGH

Memorandum:
Date:-

SAI-RM-4
October, 1973

Subject: Lambda-definability and logical relations
Author: G.D. Plotkin

J.W. Pedro H. Azevedo de Amorim
(Oxford)

Objectives :

- 1) What is a logical relation?
- 2) How do we understand log. rels. denotationally?
- 3) What is a logical relation in the presence
of side effects?
- 4) How does this picture extend to CBRV?

① What is a logical relation?

(a) The high-level idea

A tool for proving (meta)theoretic
properties of logics / programming languages

A tool for proving (meta)theoretic properties of logics / programming languages

eg

- definability — Plotkin (1973), Jung-Tiuwyn (1993),...
 - ↳ what parts of a model are the interpretations of terms?
- effect simulation — Milne (1974), ..., Katsunata (2013),...
 - ↳ do two models model effect(s) the same way?
- adequacy — Sieber (1992),...
 - ↳ do the denotational and operational semantics agree at base types?

Idea : logical relations are those relations
which are "invariant" under all the
term-formation operations.

→ builds on the fact every A-term's denotation is invariant
under permutations But that there can be
uncountably infinitely many such elements in a model.

"Definition": fix a model. A logical relation R is
a family of (n -ary) relations $\{R_\sigma \subseteq [\sigma]^n \mid \sigma \in \text{Type}\}$
such that

"Definition": fix a model. A logical relation R is
a family of (n -ary) relations $\{R_\sigma \subseteq [\sigma]^n \mid \sigma \in \text{Type}\}$
such that if

$$\frac{(t_i : \tau_i)_{i \in I}}{\text{op}(t_1, \dots, t_n) : \tau}$$

is a term formation rule, then:

"Definition": fix a model. A logical relation R is
 a family of (n -ary) relations $\{R_\sigma \subseteq [\sigma]^n \mid \sigma \in \text{Type}\}$
 such that if

$$\frac{(t_i : \tau_i)_{i \in I}}{\text{op}(t_1, \dots, t_n) : \tau}$$

is a term formation rule, then:

$$(t_i : \tau_i)_{i \in I} \in R_{\tau} \quad \Rightarrow \quad ([\varphi(t_1, \dots, t_n)], \dots, [\varphi(t_1, \dots, t_n)]) \in R_{\tau}$$

for all i

"Definition": fix a model. A logical relation R is
 a family of (n -ary) relations $\{R_\sigma \subseteq [\sigma]^n \mid \sigma \in \text{Type}\}$
 such that if

$$\frac{(t_i : \tau_i)_{i \in I}}{\text{op}(t_1, \dots, t_n) : \tau}$$

is a term formation rule, then:

$$(t_i : \tau_i)_{i \in I} \in R_{\tau} \quad \Rightarrow \quad ([\varphi(t_1, \dots, t_n)], \dots, [\varphi(t_1, \dots, t_n)]) \in R_{\tau}$$

for all i

BASIC LEMMA: if $t : \sigma$ then $(t, \dots, t) \in R_\sigma$.
 (so long as this is true at base types)

How do we use logical relations?

DEFINABILITY : let M be a model and $x \in M$.

Is x the interpretation of some term?

DEFINABILITY: let M be a model and $x \in M$.

Is x the interpretation of some term?

STS: there is some logical relation R and type σ
st. $(x_1, \dots, x_n) \in R_\sigma$.

DEFINABILITY: let M be a model and $x \in M$.

Is x the interpretation of some term?

STS: there is some logical relation R and type σ
st. $(x_1, \dots, x_n) \in R_\sigma$.

e.g. [Plotkin, Sieber]

- definability of elements in Scott's D_∞ model
- parallel-or is not definable in PCF
 - ↳ sparked a huge literature on full abstraction!

[Milner '74]

EFFECT SIMULATION

: let M and N be two models

for the same effect. Do they capture "the same" behavior?

e.g.

powerset and list for non-determinism. Is it the case that

$$\llbracket E \rrbracket^P = \{x_1, \dots, x_n\} \Leftrightarrow \llbracket E \rrbracket^{\text{list}} = (\text{some permutation of } [x_1, \dots, x_n]) ?$$

EFFECT SIMULATION

: let M and N be two models

for the same effect. Do they capture "the same" behaviour?

eg

powerset and list for non-determinism. Is it the case that

$$\llbracket E \rrbracket^P = \{x_1, \dots, x_n\} \Leftrightarrow \llbracket E \rrbracket^{\text{list}} = (\text{some permutation of } [x_1, \dots, x_n]) ?$$

STRATEGY : define a logical relation R on $M \times N$ saying they have the same behaviour at base types. Then the interpretation of $\vdash t : \sigma$ is a pair $(\llbracket E \rrbracket^M, \llbracket E \rrbracket^N) \in R$ relating the interpretations.

EFFECT SIMULATION

: let M and N be two models
for the same effect. Do they capture "the same" behaviour?

eg

powerset and list for non-determinism. Is it the case that

$$\llbracket E \rrbracket^P = \{x_1, \dots, x_n\} \Leftrightarrow \llbracket E \rrbracket^{\text{list}} = (\text{some permutation of } [x_1, \dots, x_n]) ?$$

STRATEGY : define a logical relation R on $M \times N$ saying they have the same behaviour at base types. Then the interpretation of $\vdash t : o$ is a pair $(\llbracket E \rrbracket^M, \llbracket E \rrbracket^N) \in R_o$ relating the interpretations.

Cf. also: Friedman's completeness proof for STLC, adequacy proofs, ...

① What is a logical relation ?

(b) for STLC

STLC = simply-typed λ -calculus

types $\tau ::= \beta \in \text{Base} \mid \perp \mid \tau_1 \times \tau_2 \mid \sigma \rightarrow \tau$

terms $t ::= x \mid () \mid \pi_i(t) \mid \langle t_1, t_2 \rangle \mid t u \mid \lambda x. t$

eqns $\pi_i(\langle t_1, t_2 \rangle) \equiv_p t_i , \quad (\lambda x. t) u \equiv_p t[u/x]$

$t \equiv_1 \langle \pi_1 t, \pi_2 t \rangle , \quad t \equiv_1 \lambda x. t^x x.$

$() \equiv_H t \quad (\text{for } t : 1)$

STLC = simply-typed λ -calculus

types $\tau ::= \beta \in \text{Base} \mid \perp \mid \tau_1 \times \tau_2 \mid \sigma \rightarrow \tau$

terms $t ::= x \mid () \mid \pi_i(t) \mid \langle t_1, t_2 \rangle \mid t u \mid \lambda x. t$

eqns $\pi_i(\langle t_1, t_2 \rangle) \equiv_p t_i \quad , \quad (\lambda x. t) u \equiv_p t[u/x]$

$t \equiv_1 \langle \pi_1 t, \pi_2 t \rangle \quad , \quad t \equiv_1 \lambda x. t^x x$

$() \equiv_H t \quad (\text{for } t : 1)$

SEMANTIC MODEL :

ccc \mathfrak{C} + $s : \text{Base} \longrightarrow \mathfrak{C}$ interpreting base types

... for a set-like model:

DEFN: an ^{unary} (STLC) logical relation R is a family of relations $\{R_\sigma \subseteq [\sigma] \mid \sigma \in \text{Type}\}$ s.t.

... for a set-like model:

DEFN: an ^{unary} (STLC) logical relation R is a family of relations $\{R_\sigma \subseteq [\sigma] \mid \sigma \in \text{Type}\}$ s.t.

$$\textcircled{1} \quad R_1 = \{\kappa\}$$

$$\textcircled{2} \quad (x_1, x_2) \in R_{\sigma_1 \times \sigma_2} \subseteq [\sigma_1] \times [\sigma_2]$$

$$\iff x_i \in R_{\sigma_i} \text{ for } i=1,2$$

$$\text{i.e. } p \in R_{\sigma_1 \times \sigma_2} \Leftrightarrow \pi_i(p) \in R_{\sigma_i} \text{ for } i=1,2$$

... for a set-like model:

DEFN: an ^{unary} (STLC) logical relation R is a family of relations $\{R_\sigma \subseteq [\sigma] \mid \sigma \in \text{Type}\}$ s.t.

$$\textcircled{1} \quad R_1 = \{\kappa\}$$

$$\textcircled{2} \quad (x_1, x_2) \in R_{\sigma_1 \times \sigma_2} \subseteq [\sigma_1] \times [\sigma_2]$$

$$\iff x_i \in R_{\sigma_i} \text{ for } i=1,2$$

$$\text{ie. } p \in R_{\sigma_1 \times \sigma_2} \Leftrightarrow \pi_i(p) \in R_{\sigma_i} \text{ for } i=1,2$$

$$\textcircled{3} \quad f \in R_{\sigma \rightarrow \tau} \subseteq [\sigma] \Rightarrow [\tau]$$

$$\iff \forall x \in R_\sigma \subseteq [\sigma]. \quad f x \in R_\tau \subseteq [\tau]$$

//

... for a CCC \mathcal{C} :

DEFN: an ^{unary} (STLC) logical relation R is a family of relations $\{R_\sigma \subseteq \mathcal{C}(1, [\sigma]) \mid \sigma \in \text{Type}\}$ s.t.

$$\textcircled{1} \quad R_1 = \{\kappa\}$$

$$\textcircled{2} \quad \langle x_1, x_2 \rangle \in R_{\sigma_1 \times \sigma_2} \subseteq \mathcal{C}(1, [\sigma_1] \times [\sigma_2])$$

$$\iff x_i \in R_{\sigma_i} \text{ for } i=1,2$$

$$\text{i.e. } p \in R_{\sigma_1 \times \sigma_2} \Leftrightarrow \pi_i(p) \in R_{\sigma_i} \text{ for } i=1,2$$

$$\textcircled{3} \quad f \in R_{\sigma \rightarrow \tau} \subseteq \mathcal{C}(1, [\sigma] \rightarrow [\tau])$$

$$\iff \forall x \in R_\sigma \subseteq \mathcal{C}(1, [\sigma]),$$

$$1 \xrightarrow{\langle f, x \rangle} ([\sigma] \rightarrow [\tau]) \times [\sigma] \xrightarrow{\text{eval}} [\tau] \in R_\tau$$

DEFⁿ: an ^{unary} (STLC) logical relation R is a family of relations $\{R_\sigma \subseteq C(1, [\sigma]) \mid \sigma \in \text{Type}\}$ st.

$$\textcircled{1} \quad R_1 = \{\star\}$$

$$\textcircled{2} \quad \langle x_1, x_2 \rangle \in R_{\sigma_1 \times \sigma_2} \subseteq C(1, [\sigma_1] \times [\sigma_2])$$

$$\iff x_i \in R_{\sigma_i} \text{ for } i=1,2$$

$$\text{ie. } p \in R_{\sigma_1 \times \sigma_2} \Leftrightarrow \pi_i(p) \in R_{\sigma_i} \text{ for } i=1,2$$

$$\textcircled{3} \quad f \in R_{\sigma \rightarrow \tau} \subseteq C(1, [\sigma] \Rightarrow [\tau])$$

$$\iff \forall x \in R_\sigma \subseteq C(1, [\sigma]).$$

$$1 \xrightarrow{\langle f, x \rangle} ([\sigma] \Rightarrow [\tau]) \times [\sigma] \xrightarrow{\text{eval}} [\tau] \in R_\tau$$

Basic lemma: if R is a logical relation and $t : \sigma$
then $[t] \in R_\sigma$.

Now you have many variants:

- n-ary relations vs unary relations ...
- varying arity = taking account of contexts ...
- adding sum types or constants ...

{
}

Now you have many variants:

- n-ary relations vs unary relations ...
 - varying arity = taking account of contexts ...
 - adding sum types or constants ...
- ⋮

There are lots of 'tweaked' versions of logical relations for SLC et al, which solve slightly different problems.

(eg. adding sums
or other types)

② How do we understand
logical relations denotationally?

BACK TO STLC IN SET

given $s : \text{Base} \rightarrow \text{Set}$ so we get $s[\ell] : [\Gamma] \rightarrow [\sigma]$
for every term $\Gamma \vdash t : \sigma$. How does R come in?

BACK TO STLC IN SET

given $s : \text{Base} \rightarrow \text{Set}$ so we get $S[\ell] : [\Gamma] \rightarrow [\sigma]$
for every term $\Gamma \vdash t : \sigma$. How does R come in?

DEFⁿ: Pred is the category with

- objects: sets X with a subset $\bar{X} \subseteq X$
- maps $(X, \bar{X}) \rightarrow (Y, \bar{Y})$: functions $f : X \rightarrow Y$
which preserve the relation: $x \in \bar{X} \Rightarrow f(x) \in \bar{Y}$

Pred is a ccc

$$1 := (1, 1)$$

$$(X_1, \bar{X}_1) \times (X_2, \bar{X}_2) := (X_1 \times X_2, \bar{X}_1 \times \bar{X}_2)$$

$$(X_1, \bar{X}) \Rightarrow (Y, \bar{Y}) := (X \Rightarrow Y, \bar{X} \Rightarrow \bar{Y})$$

where $f \in \bar{X} \Rightarrow \bar{Y} \Leftrightarrow \forall x \in \bar{X}. f x \in \bar{Y}$

"functions that preserve the relation"

Pred is a ccc

$$1 := (1, 1)$$

$$(X_1, \bar{X}_1) \times (X_2, \bar{X}_2) := (X_1 \times X_2, \bar{X}_1 \times \bar{X}_2)$$

$$(X_1, \bar{X}) \Rightarrow (Y, \bar{Y}) := (X \Rightarrow Y, \bar{X} \Rightarrow \bar{Y})$$

where $f \in \bar{X} \Rightarrow \bar{Y} \Leftrightarrow \forall x \in \bar{X}. f x \in \bar{Y}$

... and the forgetful functor $p : \text{Pred} \rightarrow \text{Set}$
strictly preserves this structure

Pred is a ccc

$$1 := (1, 1)$$

$$(X_1, \bar{X}_1) \times (X_2, \bar{X}_2) := (X_1 \times X_2, \bar{X}_1 \times \bar{X}_2)$$

$$(X_1, \bar{X}) \Rightarrow (Y, \bar{Y}) := (X \Rightarrow Y, \bar{X} \Rightarrow \bar{Y})$$

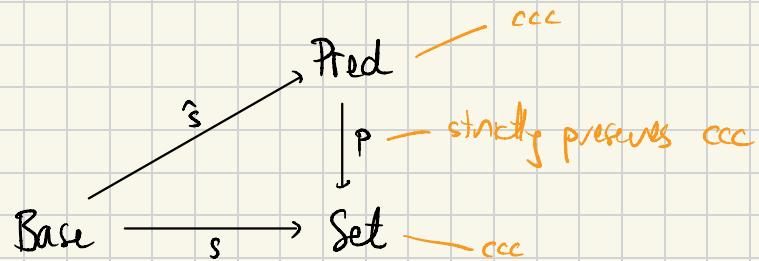
where $f \in \bar{X} \Rightarrow \bar{Y} \Leftrightarrow \forall x \in \bar{X}. f x \in \bar{Y}$

... and the forgetful functor $p : \text{Pred} \rightarrow \text{Set}$
strictly preserves this structure

NOW WE CAN SEE WHERE LOG. RELS. COME FROM...

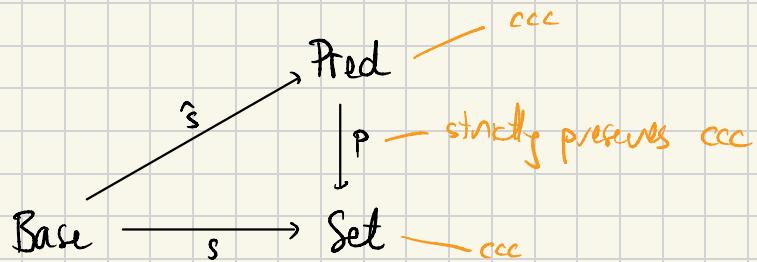
Suppose we pick for each $\beta \in \text{Base}$ a relation $R_\beta \subseteq (\mathbb{I}_\beta \times \mathbb{I})$

This amounts to :

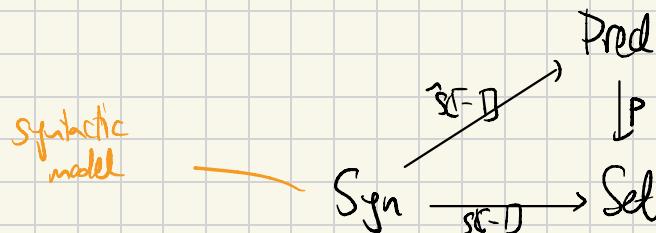


Suppose we pick for each $\beta \in \text{Base}$ a relation $R_\beta \subseteq \mathbb{S}^{\mathbb{S}-D}$.

This amounts to:

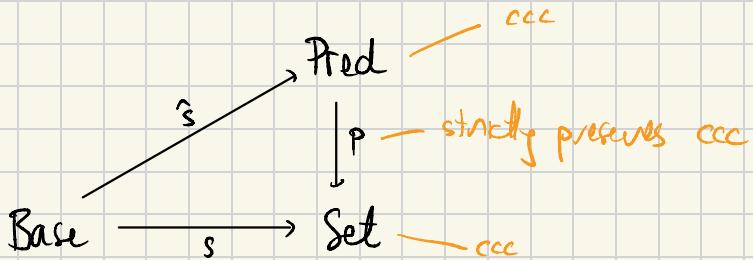


Hence, by induction / initiality, we get

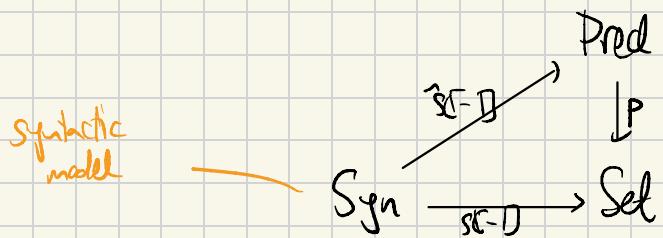


Suppose we pick for each $\beta \in \text{Base}$ a relation $R_\beta \subseteq \hat{s}[\sigma]^\beta$.

This amounts to:

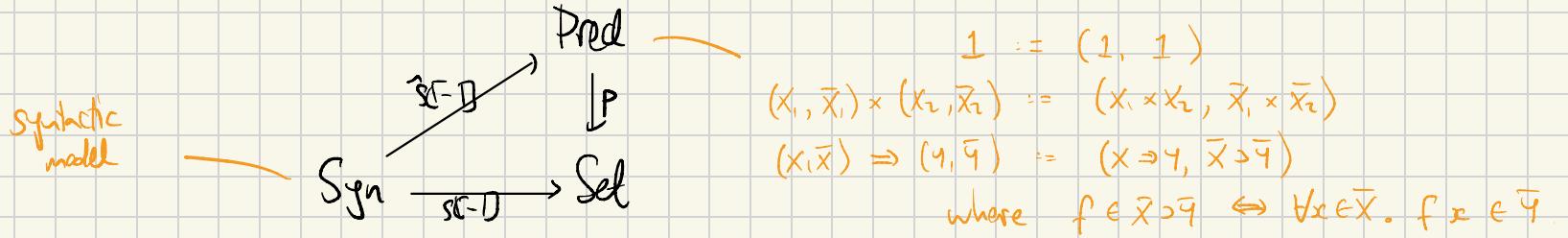


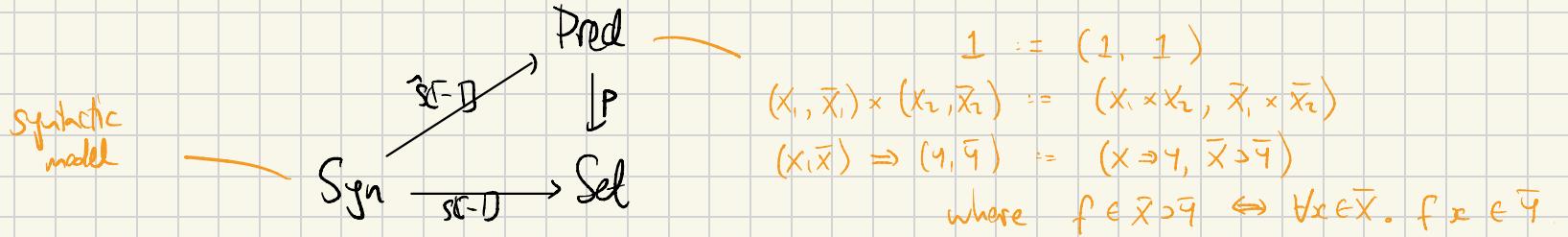
Hence, by induction / initially, we get



syntactic model

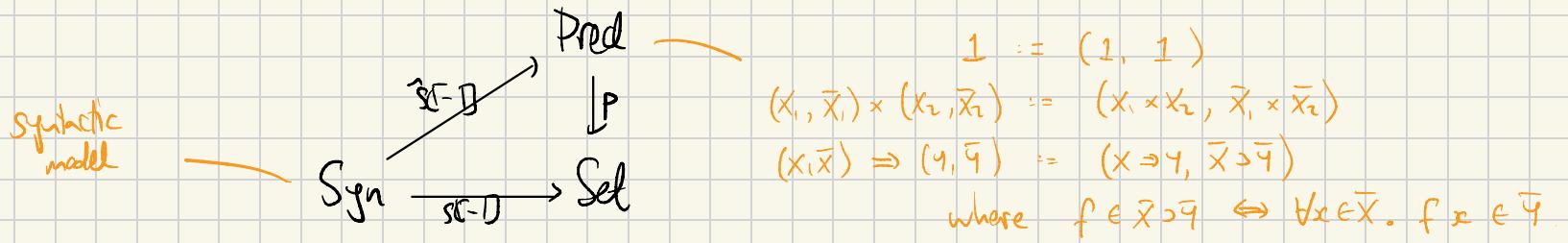
$\hat{s}[\sigma]$ encodes
a logical relation:
 $\hat{s}[\sigma] = (s[\sigma], R_0)$





$$\hat{s}[f] = (s[f], \underline{R_f})$$

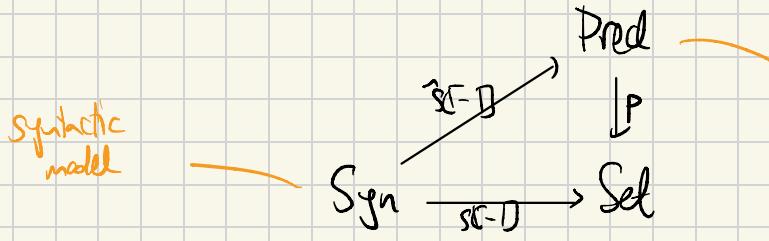
$$\hat{s}[1] = (1, \{*\}) = (1, \underline{R_1})$$



$$\hat{s}[f] = (s[f], \underline{R_f})$$

$$\hat{s}[1] = (1, \{*\}) = (1, \underline{R_1})$$

$$\hat{s}[\sigma_1 \times \sigma_2] = \hat{s}[\sigma_1] \times \hat{s}[\sigma_2] = (s[\sigma_1], R_{\sigma_1}) \times (s[\sigma_2], R_{\sigma_2}) := (s[\sigma_1] \times s[\sigma_2], \underline{R_{\sigma_1} \times R_{\sigma_2}})$$



$$\begin{aligned}
 1 &:= (1, 1) \\
 (x_1, \bar{x}_1) \times (x_2, \bar{x}_2) &:= (x_1 \times x_2, \bar{x}_1 \times \bar{x}_2) \\
 (x, \bar{x}) \Rightarrow (y, \bar{y}) &:= (x \Rightarrow y, \bar{x} \Rightarrow \bar{y}) \\
 \text{where } f \in \bar{x} \Rightarrow \bar{y} &\Leftrightarrow \forall x \in \bar{x}. f x \in \bar{y}
 \end{aligned}$$

$$\hat{s}[\beta] = (s[\beta], \underline{R_\beta})$$

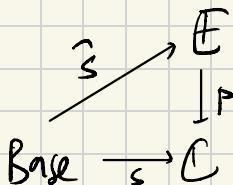
$$\hat{s}[1] = (1, \{*\}) = (1, \underline{R_1})$$

$$\hat{s}[\sigma_1 \times \sigma_2] = \hat{s}[\sigma_1] \times \hat{s}[\sigma_2] = (s[\sigma_1], R_{\sigma_1}) \times (s[\sigma_2], R_{\sigma_2}) = (s[\sigma_1] \times s[\sigma_2], \underline{R_{\sigma_1} \times R_{\sigma_2}})$$

$$\hat{s}[\sigma \rightarrow \tau] = \hat{s}[\sigma] \Rightarrow \hat{s}[\tau] = (s[\sigma], R_\sigma) \Rightarrow (s[\tau], R_\tau) = (s[\sigma] \Rightarrow s[\tau], \underline{R_\sigma \Rightarrow R_\tau})$$

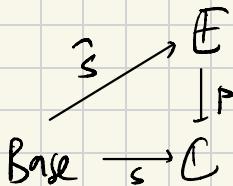
SLOGAN: to give an STLC logical relation for a
[Reynolds, Ma] Model (\mathcal{C}, s) is to give

- ① a a ccc \mathbb{E} of "relations"
- ② a functor $p: \mathbb{E} \longrightarrow \mathcal{C}$ strictly preserving the
ccc structure
- ③ an interpretation \hat{s} in \mathbb{E} st.



SLOGAN: to give an STLC logical relation for a
[Reynolds, Ma] Model (\mathcal{C}, s) is to give

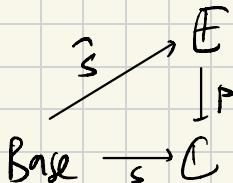
- ① a a ccc \mathbb{E} of "relations"
- ② a functor $p: \mathbb{E} \longrightarrow \mathcal{C}$ strictly preserving the
ccc structure
- ③ an interpretation \hat{s} in \mathbb{E} st.



... then R_σ arises via $\hat{s}[\sigma]$ in \mathbb{E} .

SLOGAN: to give an STLC logical relation for a
[Reynolds, Ma] Model (\mathcal{C}, s) is to give

- what is this?
- [① a a ccc \mathbb{E} of "relations"
 - ② a functor $p: \mathbb{E} \rightarrow \mathcal{C}$ strictly preserving the ccc structure
 - ③ an interpretation \hat{s} in \mathbb{E} st.



... then R_o arises via $\hat{s}[o]$ in \mathbb{E} .

from this perspective,

logical relations is about
building new, "relational", models
over existing models

RELATIONS MODELS

- how can we spot them?
- how can we reason about them?
- what facts do we know hold for any such model?

RELATIONS MODELS

- how can we spot them?
- how can we reason about them?
- what facts do we know hold for any such model?

L as "fibrations for logical relations"

P : Pred \rightarrow Set has a special property :

given $\bar{Y} \subseteq Y$ and $f : X \rightarrow Y$, there is a
canonical way to "pull back" the predicate \bar{Y}
to a predicate on X , st. f lifts to a map in Pred:

$P : \text{Pred} \rightarrow \text{Set}$ has a special property:

given $\bar{Y} \subseteq Y$ and $f : X \rightarrow Y$, there is a canonical way to "pull back" the predicate \bar{Y} to a predicate on X , st. f lifts to a map in Pred :

$$\begin{array}{ccc} (Y, \bar{Y}) & & \text{Pred} \\ & \downarrow p & \\ X \xrightarrow{f} Y = p(Y, \bar{Y}) & & \text{Set} \end{array}$$

P : Pred \rightarrow Set has a special property :

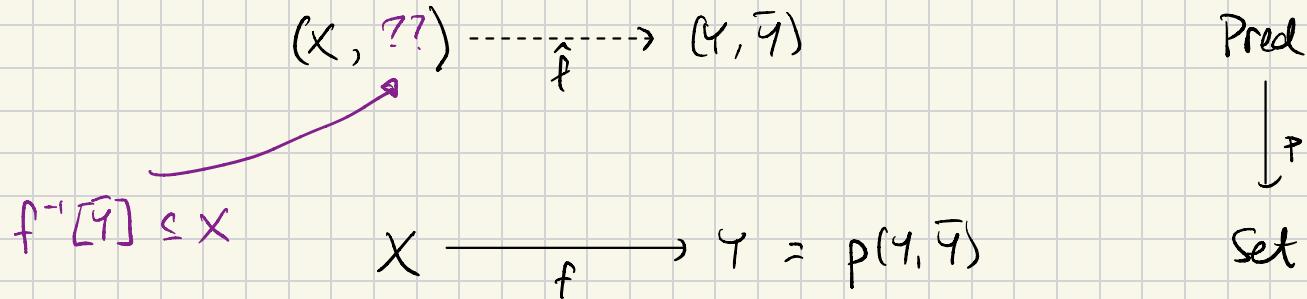
given $\bar{Y} \subseteq Y$ and $f : X \rightarrow Y$, there is a canonical way to "pull back" the predicate \bar{Y} to a predicate on X , st. f lifts to a map in Pred:

$$\begin{array}{ccc} (X, ??) & \xrightarrow[\hat{f}]{} & (Y, \bar{Y}) \\ & & \downarrow p \\ X & \xrightarrow[f]{} & Y = p(Y, \bar{Y}) \end{array}$$

Pred
↓
Set

P : Pred \rightarrow Set has a special property :

given $\bar{Y} \subseteq Y$ and $f : X \rightarrow Y$, there is a canonical way to "pull back" the predicate \bar{Y} to a predicate on X , st. f lifts to a map in Pred:



(Grothendieck)

DEFⁿ: a ^Afibration is a functor $p : E \rightarrow B$ such that for every $f : A \rightarrow Y$ in B there is an object $\hat{A} \in E$ and a map $\hat{f} : \hat{A} \rightarrow Y$ st.

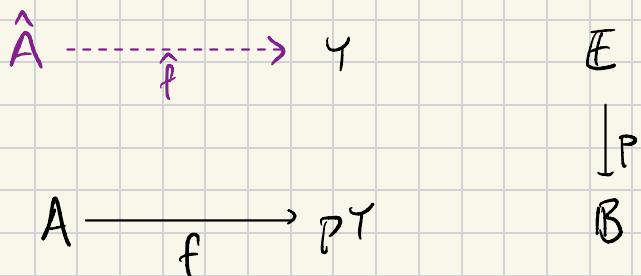
- $p(\hat{A}) = A$
- $p(\hat{f}) = f$
- a universal property holds

(Grothendieck)

DEFⁿ: a ^Afibration is a functor $p : E \rightarrow B$ such that for every $f : A \rightarrow pY$ in B there is an object $\hat{A} \in E$ and a map $\hat{f} : \hat{A} \rightarrow Y$ st.

- $p(\hat{A}) = A$
- $p(\hat{f}) = f$
- a universal property holds

PICTORIALLY:



EXAMPLES

- $p: \text{Pred}_n \longrightarrow \text{Set}$ $n\text{-ary}$ predicates over Set
L more generally $\text{Sub}_n(\mathcal{C}) \longrightarrow \mathcal{C}$ $n\text{-ary}$ subjects over \mathcal{C} for nice enough \mathcal{C}

- $\text{cod}: \mathcal{C}^{\rightarrow} \longrightarrow \mathcal{C}$ the codomain fibration

/

def: $f: X \rightarrow Y$ in \mathcal{C}

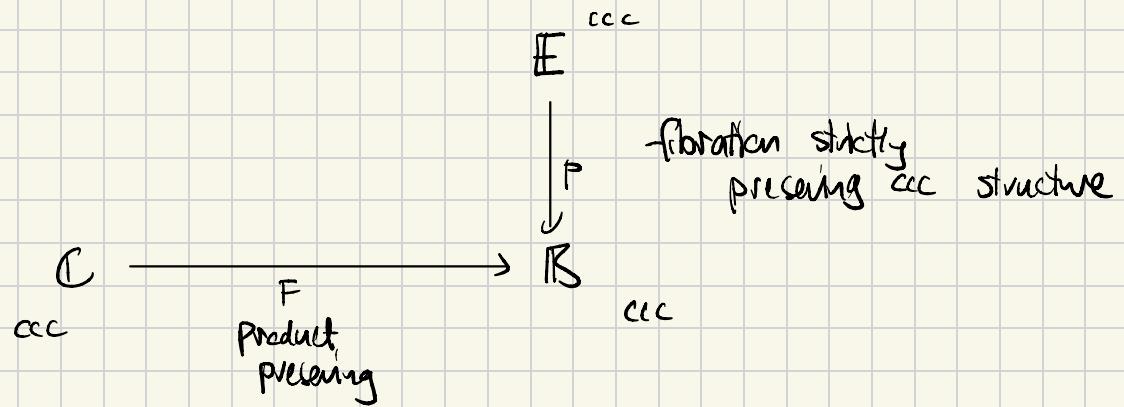
maps: squares $\begin{array}{ccc} X & \xrightarrow{\quad} & X' \\ f \downarrow & \approx & \downarrow f' \\ Y & \xrightarrow{\quad} & Y' \end{array}$

A RECIPE :

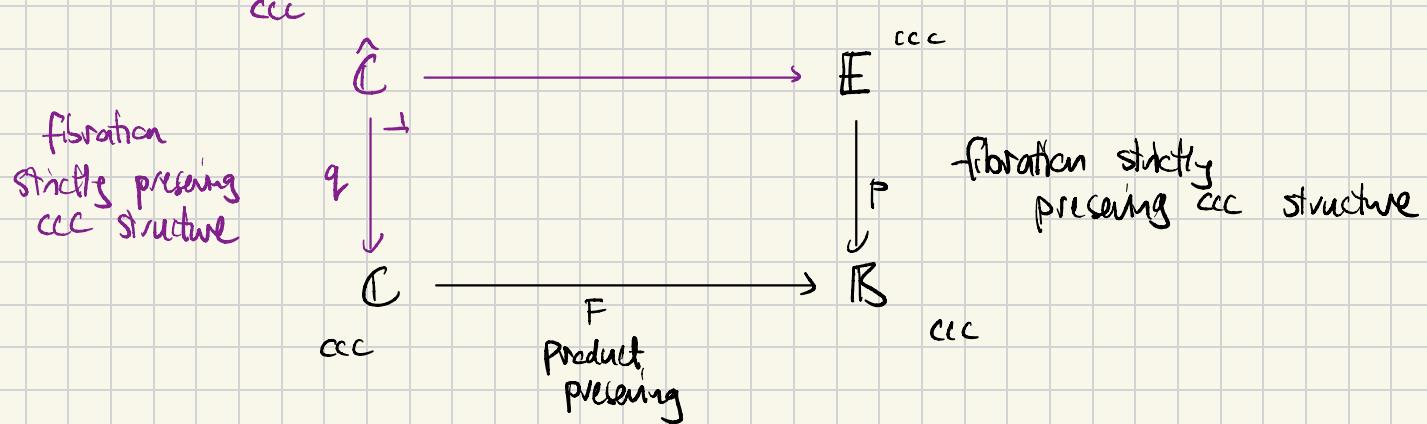
C

cc

A RECIPE :

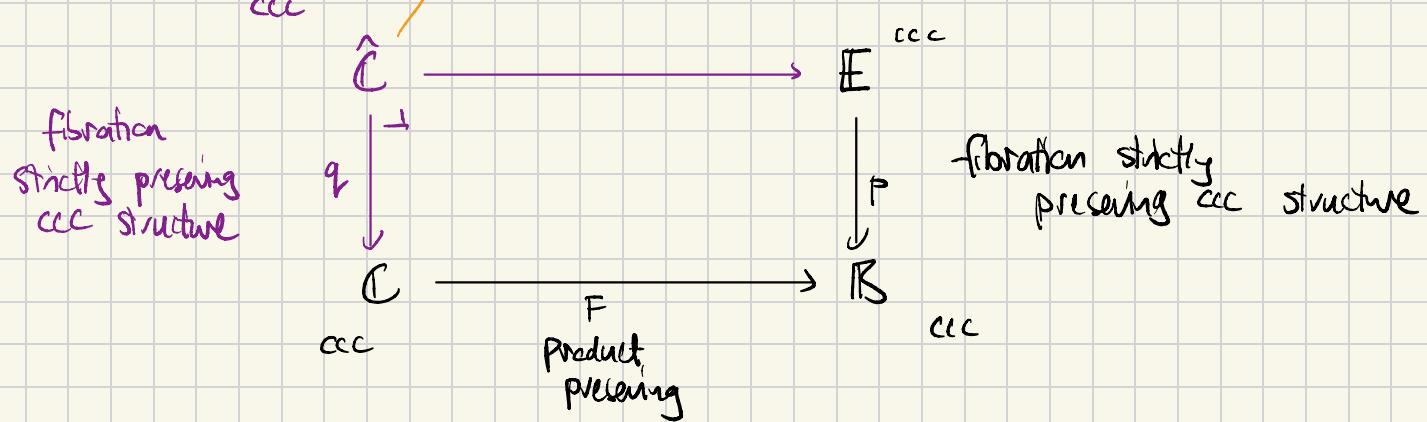


A RECIPE :



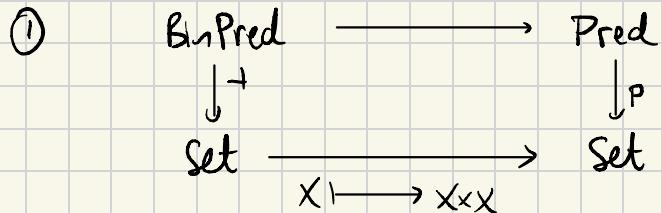
A RECIPE :

obj: $(C \in \mathcal{C}, X \in E)$ st. $FC = pX$,
ie. a C-object and a 'relation' on it

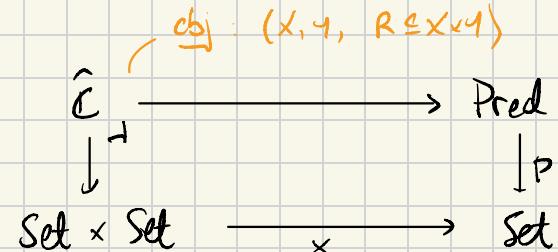


EXAMPLES

obj: $(X, R \subseteq X \times X)$

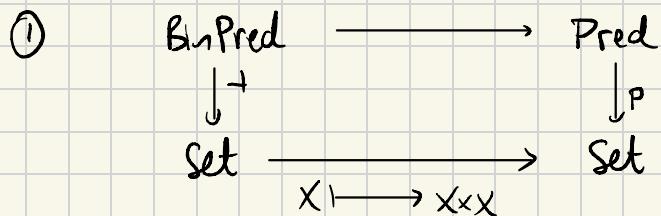


②

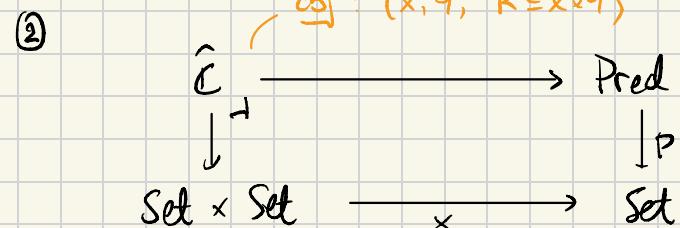


can start to
relate models

EXAMPLES

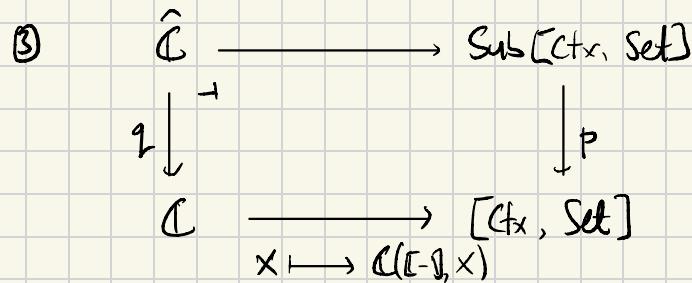


obj: $(X, R \subseteq \text{XXX})$



obj: $(X, Y, R \subseteq X \times Y)$

can start to
relate models



obj: $(X \in C, R \supseteq C(\Gamma, X))$

ie. for every context Γ a subset

$$R(\Gamma) \subseteq C(\Gamma, X)$$

compatible with renamings

Kripke relations of varying arity

SUMMING UP

: we have a nice denotational account of
STLC logical relations

SUMMING UP

: we have a nice denotational account of
STLC logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones

SUMMING UP

: we have a nice denotational account of
STLC logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the ccc structure — i.e. the model

SUMMING UP

: we have a nice denotational account of
STLC logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the ccc structure — i.e. the model
- ③ the general theory gives us a useful framework for finding such fibrations (= relations models)

SUMMING UP

: we have a nice denotational account of
STLC logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the ccc structure — i.e. the model
- ③ the general theory gives us a useful framework for finding such fibrations (= relations models)

DEFN : an STLC fibration is a fibration that strictly preserves ccc structure \approx a logical relation

③ What is a logical relation
in the presence of side effects?

↳ spoiler : it will be a fibration which
strictly preserves the model structure

SIDE EFFECTS

STLC programs are simple: they always terminate,
never interact with the world in interesting ways

SIDE EFFECTS

STLC programs are simple: they always terminate,
never interact with the world in interesting ways

actual programs have things like

- exceptions
- non-determinism
- state
- user input
- output / printing
- probabilistic behaviours
- ; ;

SIDE EFFECTS

STLC programs are simple: they always terminate,
never interact with the world in interesting ways

actual programs have things like

- exceptions
- non-determinism
- state
- user input
- output / printing
- probabilistic behaviours
- ; ;

structure
of these
programs
captured by
monads

MONADIC METALANGUAGE

Extend STLC with

types

$\tau ::= \dots | T_C$

$t \gg x$

)

terms

$t ::= \dots | \text{return } t | \text{let } x = u \text{ in } t$

a pure program
is trivially effectful

equations

$\dots \text{ let } x = (\text{return } u) \text{ in } t \dots$

$\equiv_B t[u/x]$

run u , bind it to
 x , then run t .
Haskell notation:

EXAMPLE MONADS

- List or powerset for non-determinism
- Maybe = $(-) + 1$ for non-termination
- Exception = $(-) + E$ for a set E of exceptions
- Writer = $(-) \times C^*$ for C a set of characters
- $K_R := (- \Rightarrow R) \Rightarrow R$ for continuations

SEMANTICS : ccc \mathcal{L} + a (strong) monad T

$$[\![T\sigma]\!] := T[\![\sigma]\!]$$

$$[\![\text{return } t]\!] := [\![\Gamma]\!] \xrightarrow{(\lambda t)} [\![\sigma]\!] \xrightarrow{\text{return}} T[\![\sigma]\!]$$

SEMANTICS : CCC \mathcal{C} + a (strong) monad T

$$[\![T\sigma]\!] := T[\![\sigma]\!]$$

$$[\![\text{return } t]\!] := [\![\Gamma]\!] \xrightarrow{(\text{return})} [\![\sigma]\!] \xrightarrow{!} T[\![\sigma]\!]$$

eg

for powerset : $[\![\text{return } t]\!](\gamma) = \{\text{EDN}\}$

for Maybe : $[\![\text{return } t]\!](\gamma) = \text{inl } (\text{ED } \gamma) \in [\![\sigma]\!] + 1$

for Writer : $[\![\text{return } t]\!](\gamma) = (\text{ED } \gamma, \varepsilon)$

SEMANTICS : $\text{acc } \mathcal{C} + \alpha$ (strong) Monad T

$$[\bar{T}\sigma] := T[\sigma]$$

$$[\text{return } t] := [\Gamma] \xrightarrow{\text{RET}} [\sigma] \xrightarrow{1} T[\sigma]$$

$$[\text{let } x = u \text{ in } t] := [\Gamma] \xrightarrow{\langle \text{id}, \text{RET} \rangle} [\Gamma] \times T[\sigma] \xrightarrow{\text{strength}} T([\Gamma] \times [\sigma]) \xrightarrow{T\text{RET}} T^2[\sigma] \xrightarrow{\mu} T[\sigma]$$

SEMANTICS : $\text{acc } C + \alpha$ (strong) Monad T

$$[\bar{T}\sigma] := T[\sigma]$$

$$[\text{return } t] := [\Gamma] \xrightarrow{\text{RET}} [\sigma] \longrightarrow T[\sigma]$$

$$[\text{let } x = u \text{ in } t] := [\Gamma] \xrightarrow{\text{id}, \text{RET}} [\Gamma] \times T[\sigma] \xrightarrow{\text{strength}} T([\Gamma] \times [\sigma]) \xrightarrow{\text{RET}} T^2[\sigma] \xrightarrow{\sim} T[\sigma]$$

eg for powerset : $[\sqcup](\gamma) \in P[\sigma]$ and $[\sqcup] : [\Gamma] \times [\sigma] \longrightarrow P[\sigma]$, so

$$[\text{let } x = u \text{ in } t](\gamma) = \text{let } [\sqcup](\gamma) = S \subseteq P[\sigma] \text{ in } \bigcup_{s \in S} [\sqcup](\gamma, s)$$

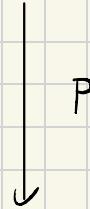
for writer:

$$[\text{let } x = u \text{ in } t](\gamma) = \text{let } [\sqcup](\gamma) = (x, w) \in [\sigma] \times C^\star \text{ in } \begin{aligned} & \text{let } [\sqcup](\gamma, x) = (y, w') \in [\sigma] \times C^\star \text{ in } \\ & \quad (y, ww') \end{aligned}$$

strict

MAPS OF λ_m MODELS

ccc +
strong monad (\mathbb{E}, \dagger)



ccc +
strong monad (\mathbb{B}, T)

strict

MAPS OF λ_m MODELS

ccc +
strong monad (\mathbb{E}, \dagger)

$$\downarrow P$$

ccc +
strong monad (\mathbb{B}, T)

P preserves ccc structure
+ monad structure on the nose:

$$P(\hat{T}x) = T(Px)$$

$$P(\hat{\mu}_x) = \mu_{Px}$$

$$P(\hat{s}_{x,y}) = s_{Px,Py}$$

$$P(\hat{\eta}_x) = \eta_{Px}$$

EXAMPLE :

1) if \mathbb{E}, \mathbb{B} are cccs and $p: \mathbb{E} \rightarrow \mathbb{B}$
preserves ccc-structure, $(\mathbb{E}, (\neg \Rightarrow R) \Rightarrow R) \xrightarrow{p} (\mathbb{B}, (\neg \Rightarrow pR) \Rightarrow pR)$

EXAMPLE:

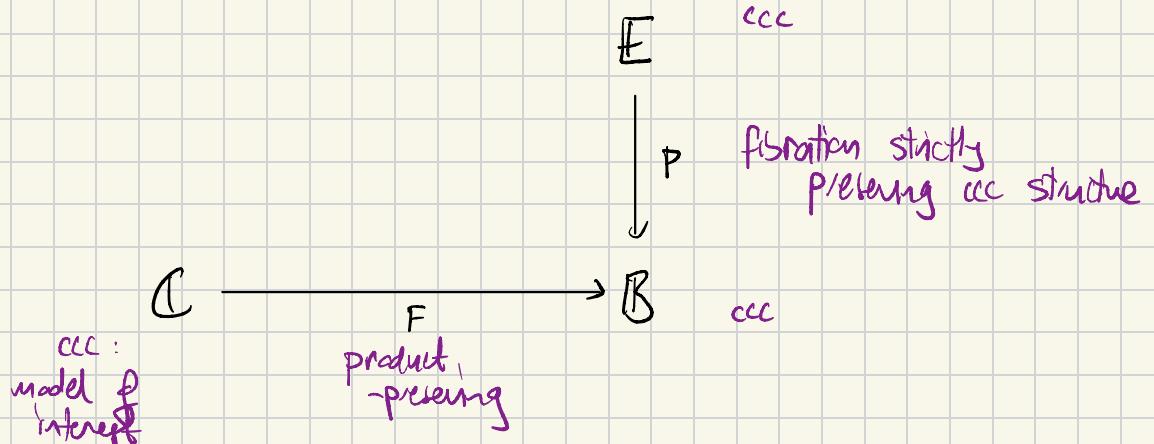
1) if \mathbb{E}, \mathbb{B} are cccs and $p: \mathbb{E} \rightarrow \mathbb{B}$
preserves ccc-structure, $(\mathbb{E}, (\neg \Rightarrow R) \Rightarrow R) \xrightarrow{p} (\mathbb{B}, (\neg \Rightarrow pR) \Rightarrow pR)$

2) if $C \in \text{Set}$ is a set of characters and $\bar{C} \subseteq C$
then $\bar{C}^* \xrightarrow{\text{submonoid}} C^*$ so $(-) \times (C^*, \bar{C}^*) : \text{Pred} \rightarrow \text{Pred}$
 $(x, \bar{x}) \mapsto (x \times C^*, \bar{x} \times \bar{C}^*)$

and $(\text{Pred}, (-) \times (C^*, \bar{C}^*)) \longrightarrow (\text{Set}, (-) \times C^*)$

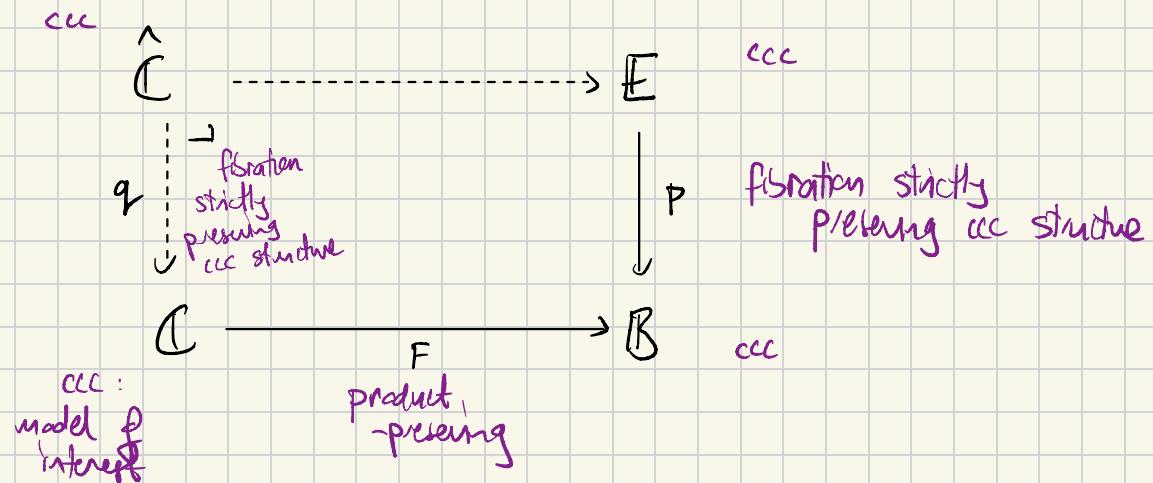
LOGICAL RELATIONS FOR $\lambda\text{-ml}$

Take the STLC picture...



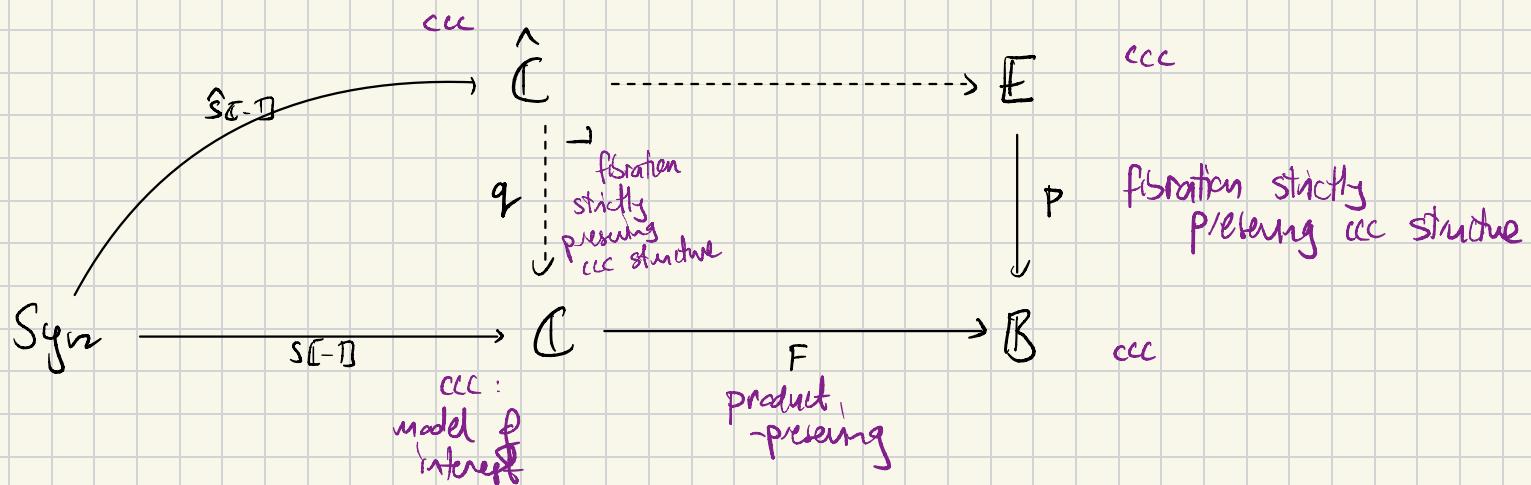
LOGICAL RELATIONS FOR LML

Take the STLC picture...



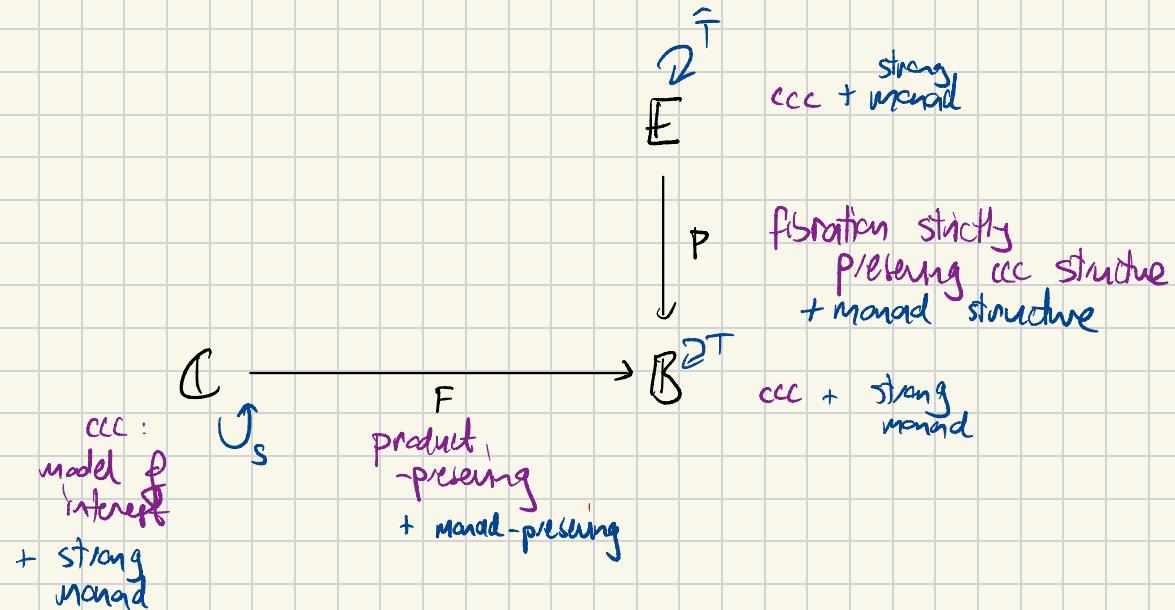
LOGICAL RELATIONS FOR LML

Take the STLC picture...



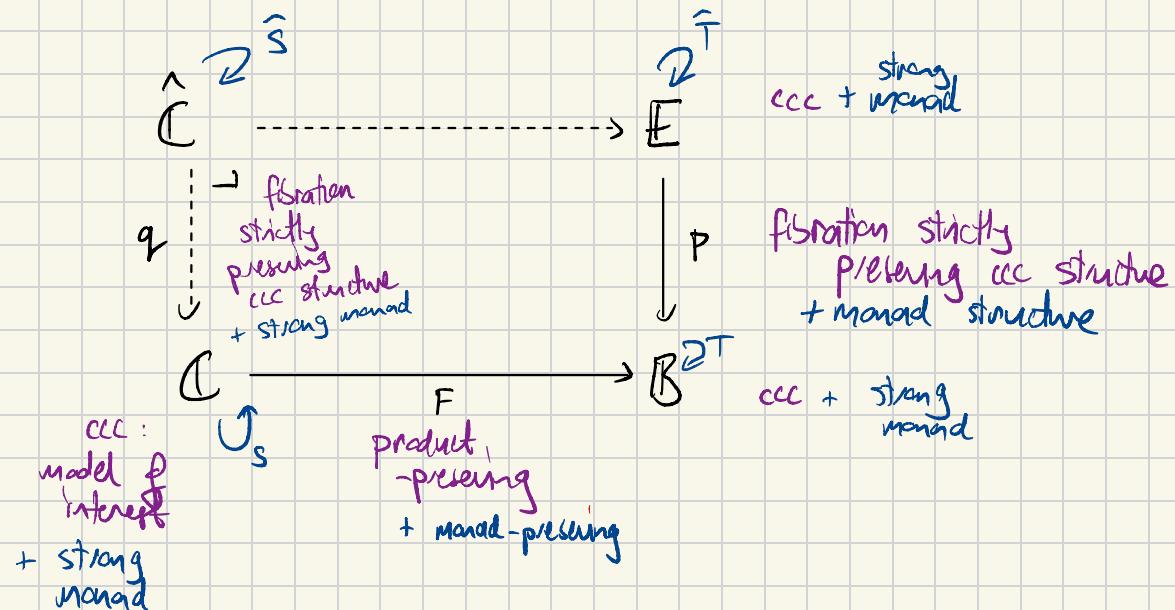
LOGICAL RELATIONS FOR $\lambda\text{-ml}$

Take the STLC picture... and add monads



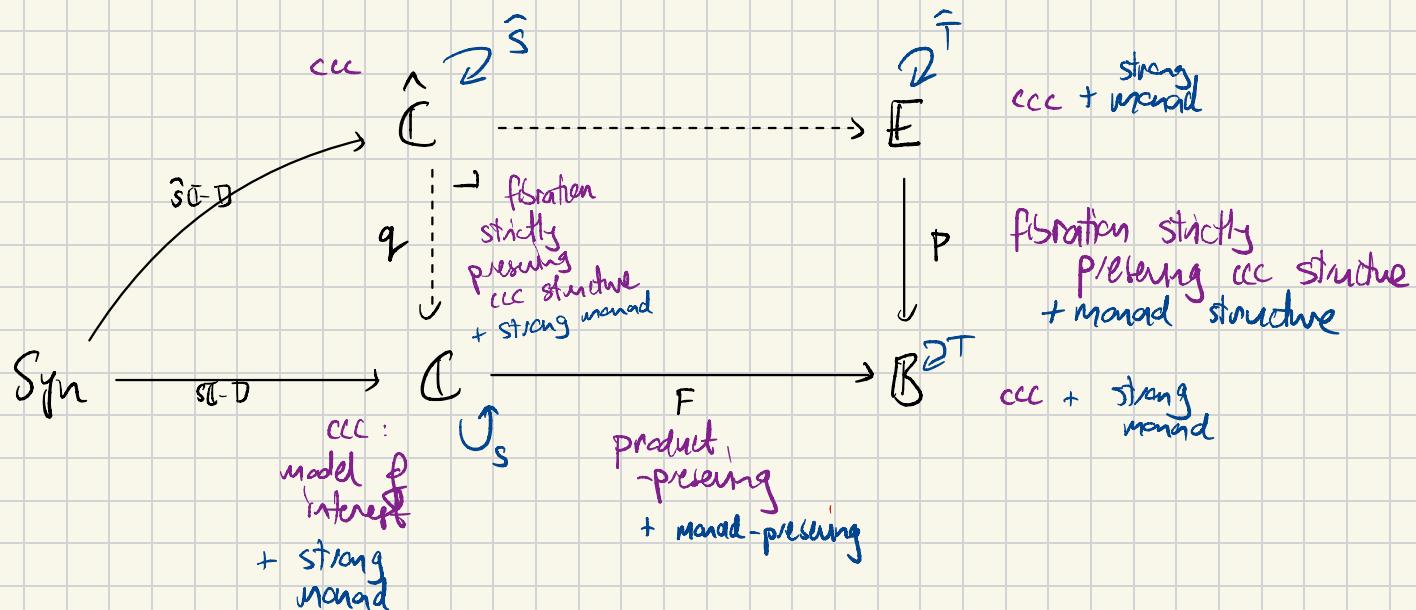
LOGICAL RELATIONS FOR $\lambda\text{-ml}$

Take the STLC picture... and add monads



LOGICAL RELATIONS FOR $\lambda\text{-ml}$

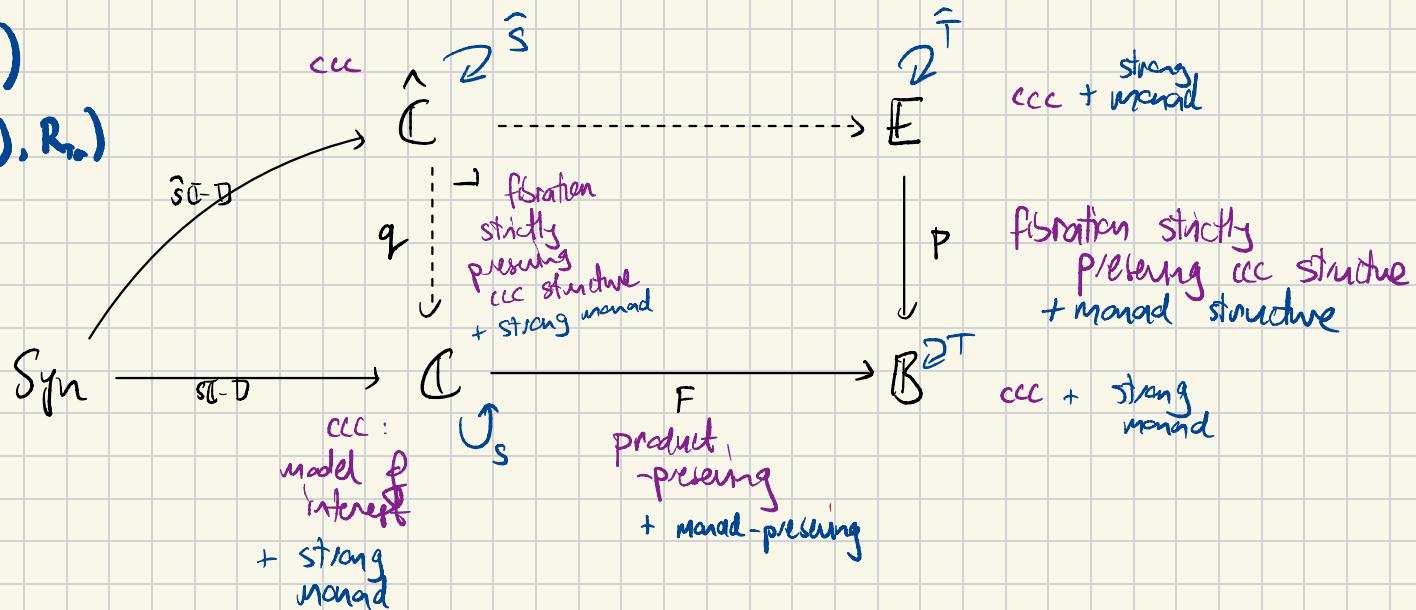
Take the STLC picture... and add monads



LOGICAL RELATIONS FOR $\lambda\text{-ml}$

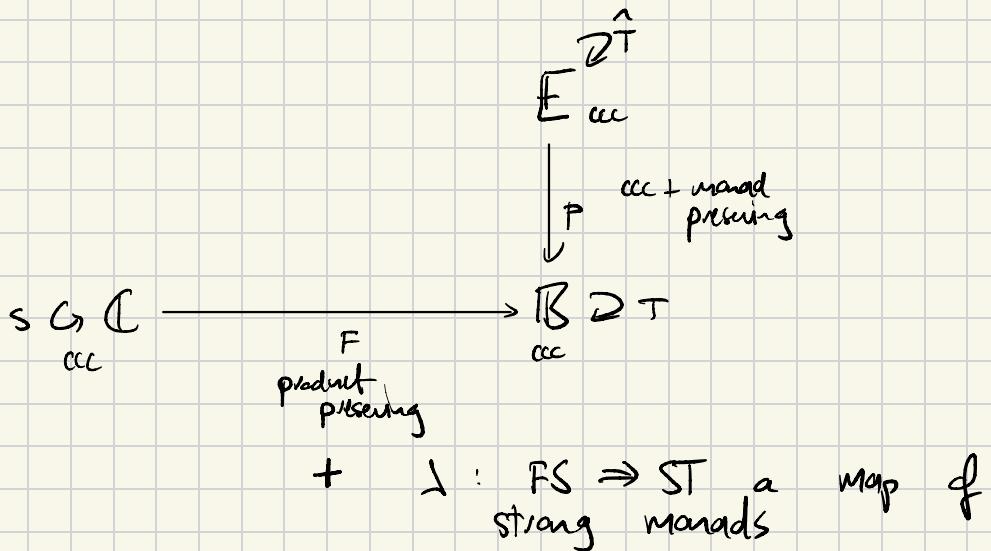
Take the STLC picture... and add monads

$$\hat{S}(\text{SLC}, R_0) \\ = (S(\text{SLC}), R_0)$$



Propw. [Katsunata, Katsunata - Kammar - S.]

Suppose you have



Propw. [Katsunata, Katsunata - Kammar - S.]

Suppose you have

$$\begin{array}{ccc} \mathcal{S} \otimes \mathcal{C} & \xrightarrow{\quad F \quad} & \mathcal{B} \hat{\otimes} \mathcal{T} \\ \text{cc} & & \text{cc} \\ & \downarrow \text{product preserving} & \\ & \mathcal{E}_{\text{cc}} & \end{array}$$

\vdash ccc + monad preserving

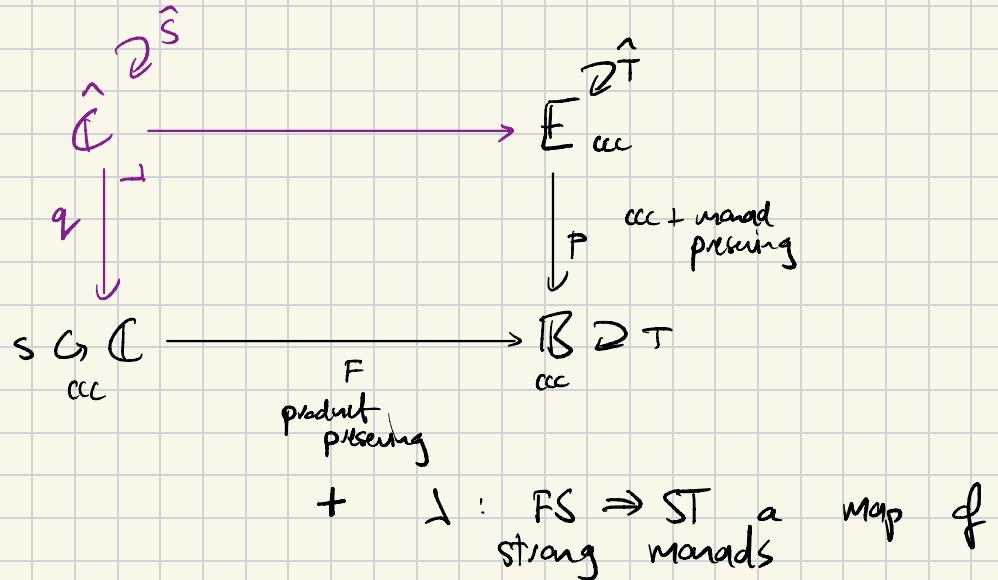
+

$\dashv : FS \Rightarrow ST$ a map of
strong monads

Then you get a universal model $(\hat{\mathcal{C}}, \hat{\mathcal{S}})$

Propw. [Katsunata, Katsunata - Kammar - S.]

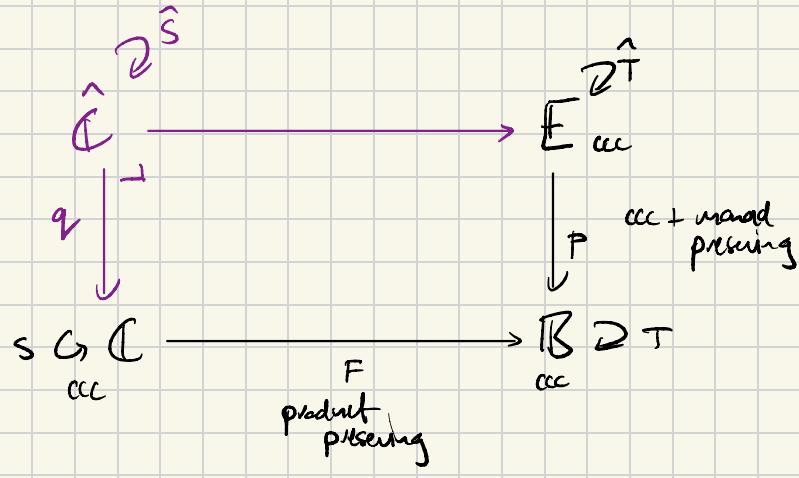
Suppose you have



Then you get a universal model (\hat{C}, \hat{S}) as shown.

Propw. [Katsunata, Katsunata - Kammar - S.]

Suppose you have

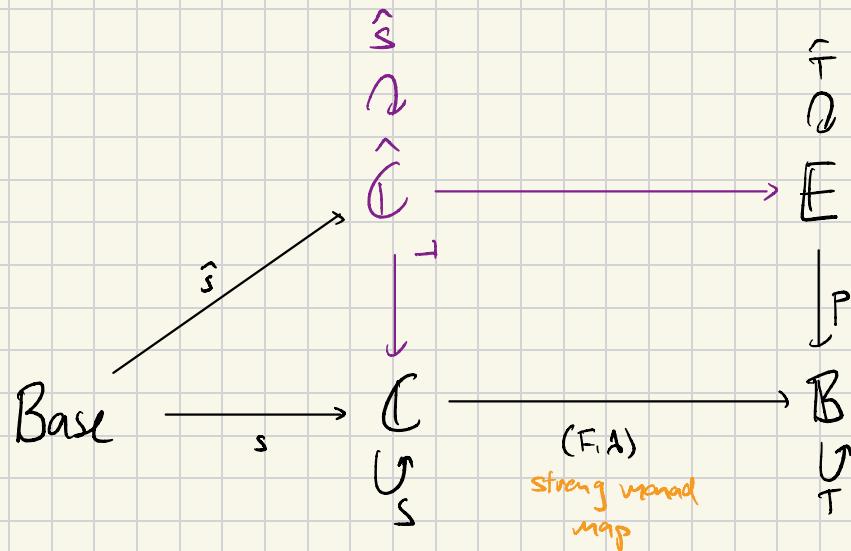


this is a
fibration!

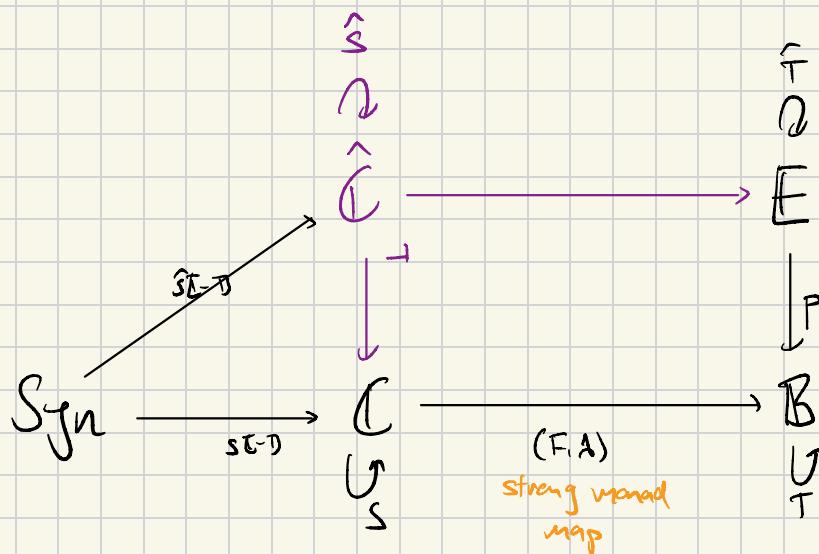
+ $\perp : FS \Rightarrow ST$ a map of
strong monads

Then you get a universal Model (\hat{C}, \hat{S}) as shown.

DEFINING A LOGICAL RELATION



DEFINING A LOGICAL RELATION



i.e. we have

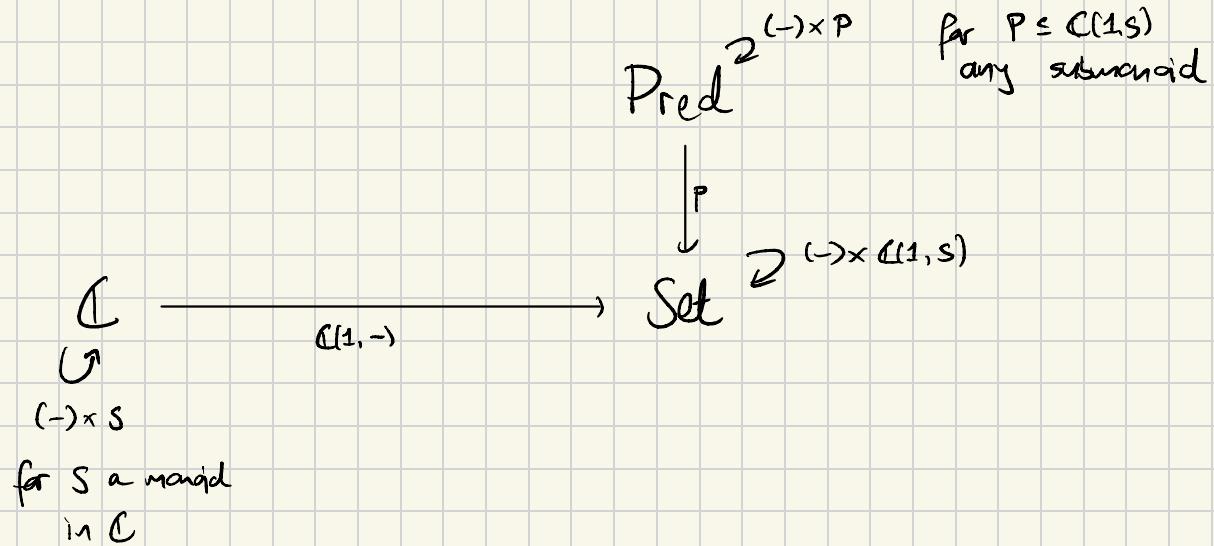
$$\hat{S}[\sigma] = (S[\sigma], R_\sigma)$$

so we get

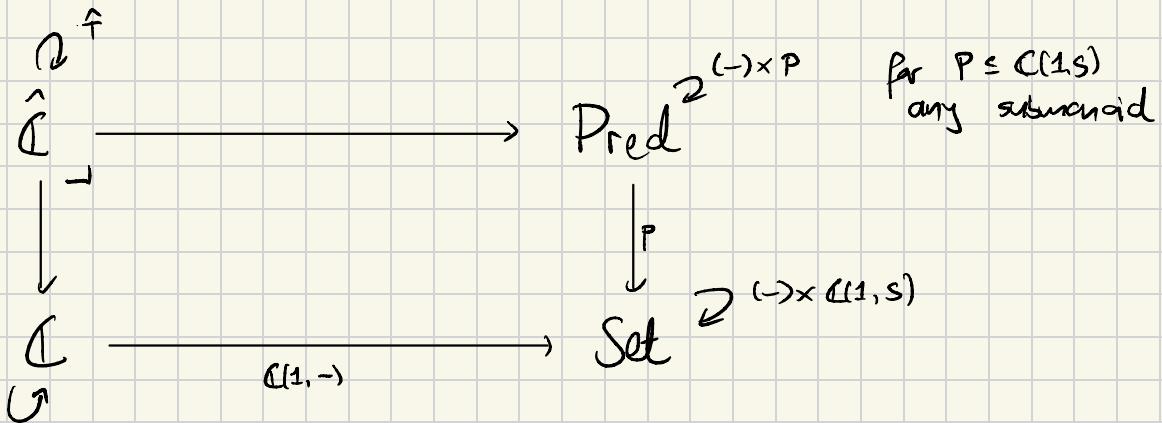
$$R := \{R_\sigma \mid \sigma \in \text{Type}\}$$

including \top !

EXAMPLE : GLOBAL STATE

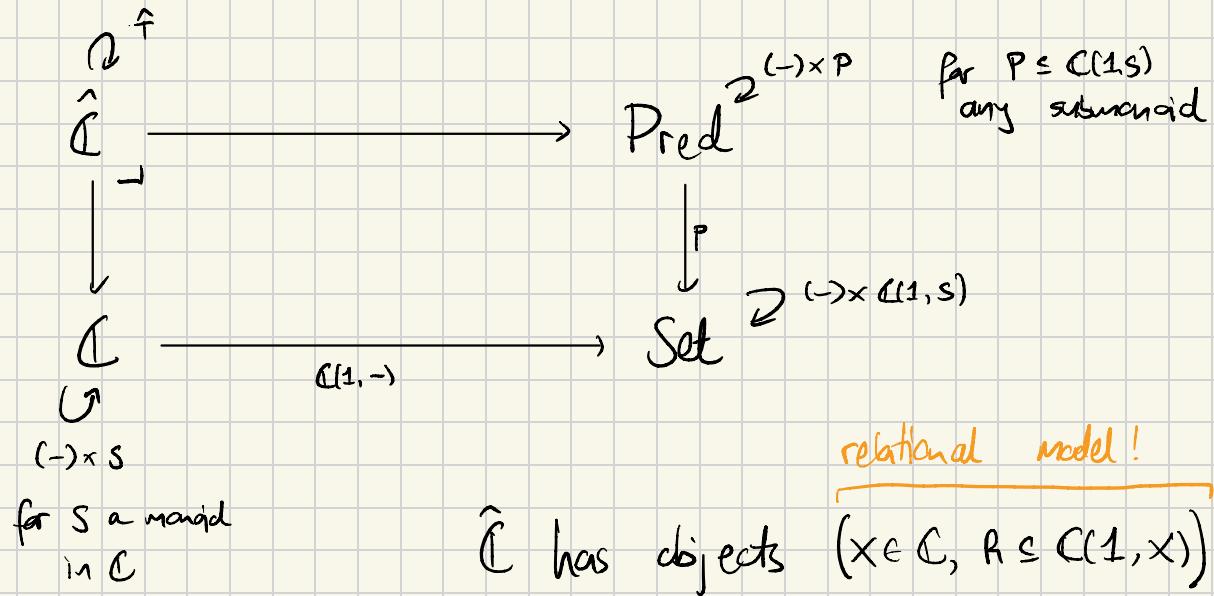


EXAMPLE : GLOBAL STATE



for S a monad
in C

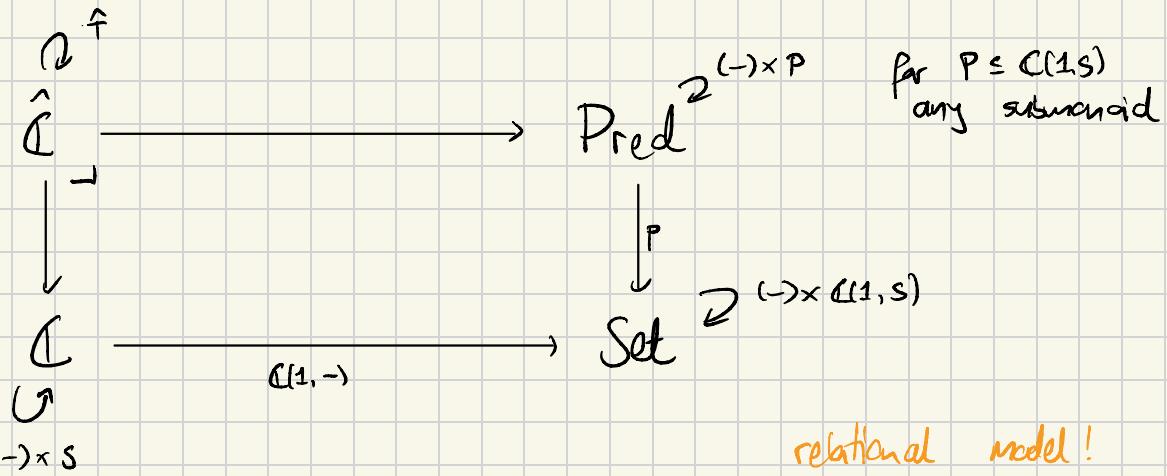
EXAMPLE : GLOBAL STATE



$$\hat{T}(X, R) = (X \times S, \hat{T}R)$$

$(x, s) \in \hat{T}R \Leftrightarrow x \in R$ and $s \in P \subseteq C(1, S)$

EXAMPLE : GLOBAL STATE



for S a monoid
in C

\hat{C} has objects $(x \in C, R \subseteq C(1, x))$

relational model!

for a logical relation:

$$\hat{T}(X, R) = (X \times S, \hat{T}_R)$$

$$R_{T_0} := \{(x, s) \mid x \in R_0 \text{ and } s \in P\}$$

$$(x, s) \in \hat{T}_R \Leftrightarrow x \in R \text{ and } s \in P \subseteq C(1, s)$$

EXAMPLE : SIMULATING EFFECTS

à la Katsurata

STRANGER MAP AND MAP

$$\gamma : L \longrightarrow S_{fin}$$

$$[x_1 \dots x_n] \longmapsto \{x_1, \dots, x_n\}$$

EXAMPLE : COMPARING EFFECTS

à la Katsurada

$$\gamma : L \longrightarrow S_{fin}$$

$$[x_1 \dots x_n] \longmapsto \{x_1, \dots, x_n\}$$

$$\begin{matrix} \text{Set} \times \text{Set} \\ \uparrow \\ L \times S_{fin} \end{matrix}$$

EXAMPLE : COMPARING EFFECTS

à la Katsurada

$$\gamma : L \longrightarrow S_{fin}$$

$$[x_1 \dots x_n] \longmapsto \{x_1, \dots, x_n\}$$

\hat{T}

\hat{D}

BinPred

$\downarrow P$

$$\underline{\text{d}}\underline{\text{g}} : (X, Y, R \rightarrow X \times Y)$$

$$\hat{T}(X, Y, R) = (S_{fin} X, S_{fin} Y, S_{fin} R)$$

$$S_{fin} R \rightarrow S_{fin} X \times S_{fin} Y$$

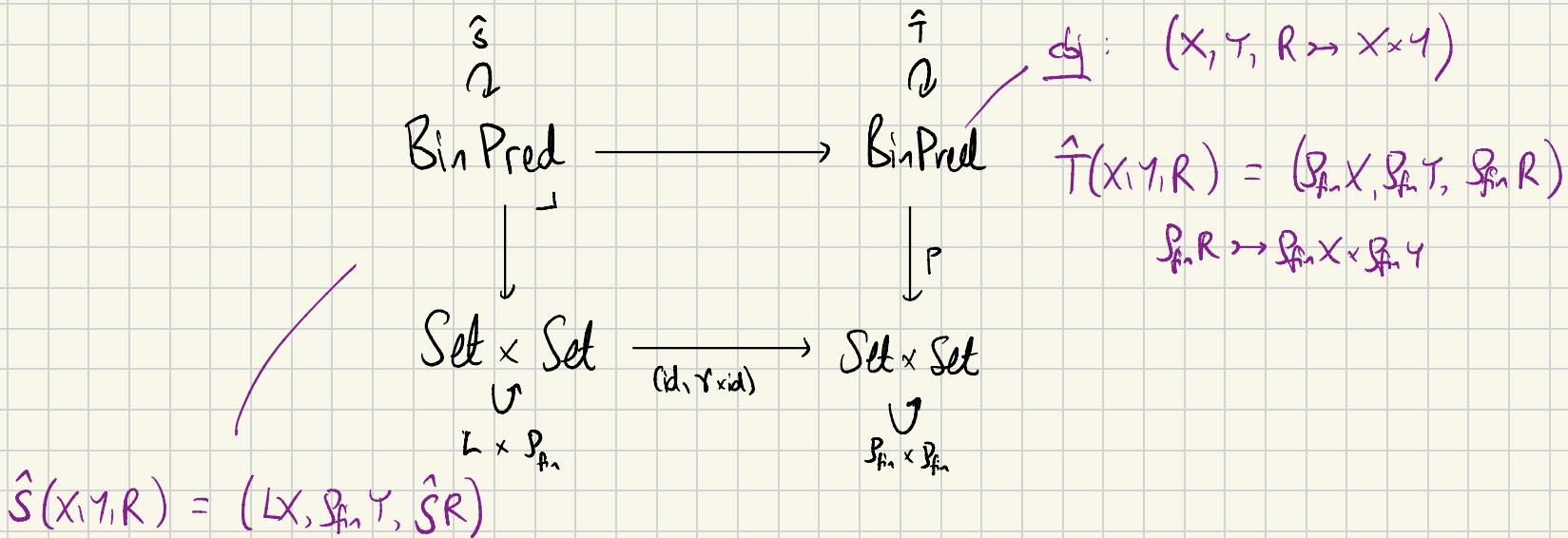
$$\begin{array}{ccc} \text{Set} \times \text{Set} & \xrightarrow{\quad (\text{id}, \gamma \times \text{id}) \quad} & \text{Set} \times \text{Set} \\ \cup \\ L \times S_{fin} & & \cup \\ & & S_{fin} \times S_{fin} \end{array}$$

EXAMPLE : COMPARING EFFECTS

à la Katsurata

$$\gamma : L \longrightarrow S_{fin}$$

$$[x_1, \dots, x_n] \longmapsto \{x_1, \dots, x_n\}$$



$$\hat{S}(X, Y, R) = (LX, S_{fin} Y, \hat{S}R)$$

$$([x_1, \dots, x_n], \{y_1, \dots, y_m\}) \in \hat{S}R$$

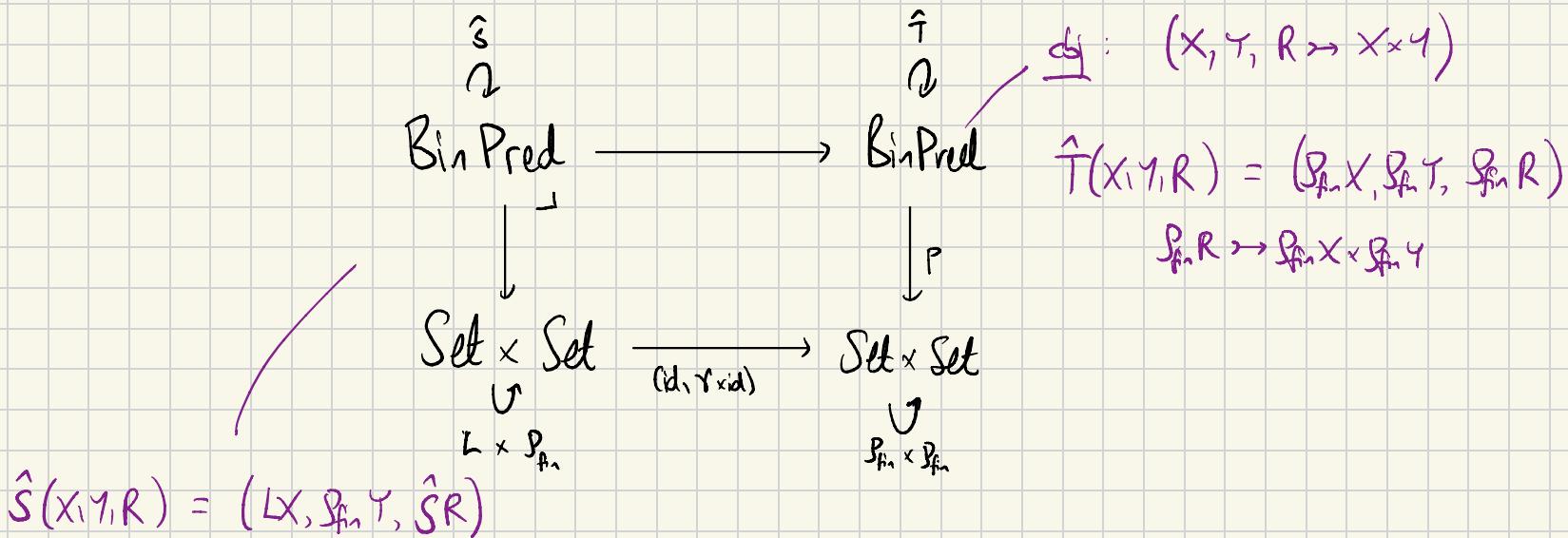
$$\Leftrightarrow n=m \text{ and } (x_i, y_i) \in R \text{ for } i=1, \dots, n$$

EXAMPLE : COMPARING EFFECTS

à la Katsurata

$$\gamma : L \longrightarrow S_{fin}$$

$$[x_1, \dots, x_n] \longmapsto \{x_1, \dots, x_n\}$$



$$\hat{S}(X, Y, R) = (LX, S_{fin}Y, \hat{S}R)$$

$$([x_1, \dots, x_n], \{y_1, \dots, y_m\}) \in \hat{S}R$$

$$\Leftrightarrow n=m \text{ and } (x_i, y_i) \in R$$

$$\text{for } i=1, \dots, n$$

$$\underline{\text{def}} : (X, Y, R \Rightarrow X \times Y)$$

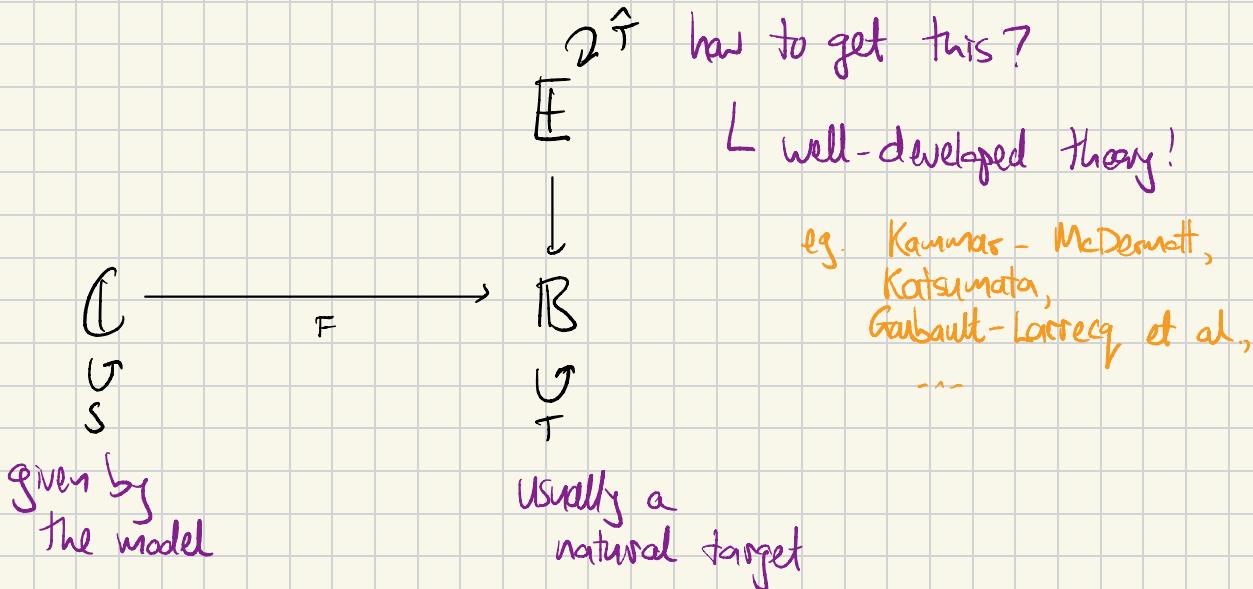
$$\hat{T}(X, Y, R) = (S_{fin}X, S_{fin}Y, S_{fin}R)$$

$$S_{fin}R \Rightarrow S_{fin}X \times S_{fin}Y$$

LOGICAL RELATION M :

$$([x_1, \dots, x_n], \{y_1, \dots, y_m\}) \in M_{\text{to}} \Leftrightarrow m=n \text{ and each } (x_i, y_i) \in M_0$$

DIFFICULT BIT : defining the target lifting



EXAMPLE : TT-LIFTING (Katsunata)

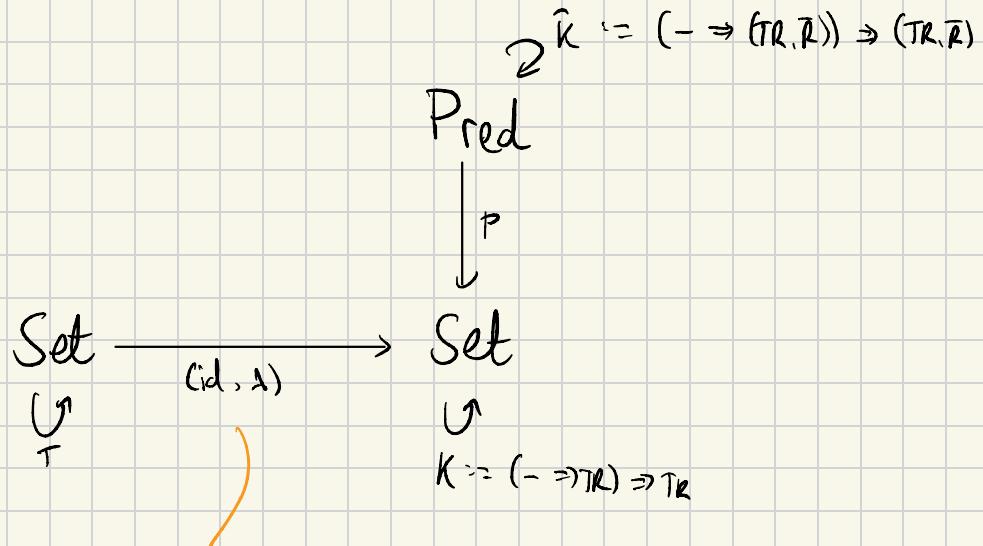
for $R \in \text{Set}$ and $\bar{R} \subseteq TR$

Set

\uparrow_T

EXAMPLE : TT-LIFTING (Katsunata)

for $R \in \text{Set}$ and $\bar{R} \subseteq \text{TR}$

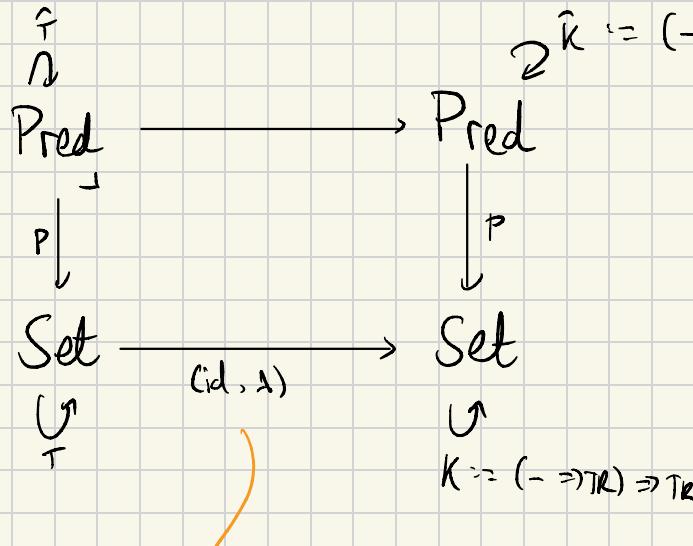


$$\lambda_x : TX \longrightarrow (X \Rightarrow \text{TR}) \Rightarrow \text{TR}$$

$$t \longmapsto \lambda f : X \rightarrow \text{TR}. \text{ let } x = t \text{ in } fx$$

EXAMPLE : TT-LIFTING (Katsunuma)

for $R \subseteq \text{Set}$ and $\bar{R} \subseteq \text{TR}$



$$\hat{T}(x, \bar{x}) = (Tx, \hat{\bar{x}})$$

$$t \in \hat{\bar{x}} \Leftrightarrow \forall f \in (\bar{x} \Rightarrow \bar{R}).$$

let $x = t$ in $f x \in \bar{R}$

"for all nice continuations f ,
 $f \approx t$ is nice"

$$\lambda_x : Tx \longrightarrow (x \Rightarrow \text{TR}) \Rightarrow \text{TR}$$

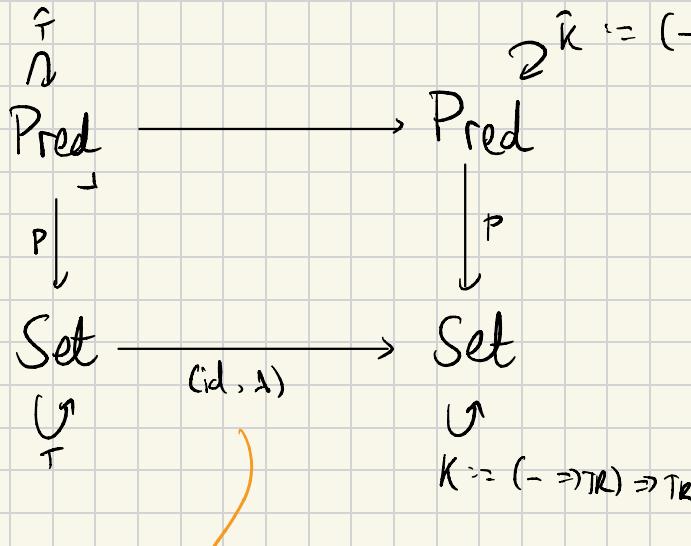
cf. Kripke realizability

$$t \longmapsto \lambda f : x \rightarrow \text{TR}. \text{ let } x = t \text{ in } f x$$

EXAMPLE : TT-LIFTING

(Katsunuma)

for $R \subseteq \text{Set}$ and $\bar{R} \subseteq \text{TR}$



$$\lambda_x : Tx \longrightarrow (x \Rightarrow \text{TR}) \Rightarrow \text{TR}$$

$$t \longmapsto \lambda f : x \rightarrow \text{TR}. \text{ let } x = t \text{ in } fx$$

LOGICAL RELATION S :

$$t \in S_{\text{Tr}} \Leftrightarrow \forall f \in (S_o \triangleright \bar{R}).$$

let $x = t$ in $fx \in \bar{R}$

"for all nice continuations f ,
 $f \approx t$ IS nice"

cf. Kripke realizability

Just as we identified
STLC logical relations with
Model-preserving fibrations,
So we can do for CBV models

SUMMING UP

: we have a nice denotational account of
CBV logical relations

SUMMING UP

: we have a nice denotational account of
CBV logical relations

① denotationally, logical relations is about
constructing refined models over existing ones

SUMMING UP

: we have a nice denotational account of
CBV logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the CCC + monad structure — i.e. the model

SUMMING UP

: we have a nice denotational account of
CBV logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the CCC + monad structure — i.e. the model
- ③ the general theory gives us a useful framework for finding such fibrations (= relations models)

SUMMING UP

: we have a nice denotational account of CBU logical relations

- ① denotationally, logical relations is about constructing refined models over existing ones
- ② such models can be identified as fibrations which strictly preserve the CCC + monad structure — i.e. the model
- ③ the general theory gives us a useful framework for finding such fibrations (= relations models)

DEFN : a CBU fibration is a fibration that strictly preserves CCC + the monad \approx a logical relation

④ How does this picture
extend to CBPV?

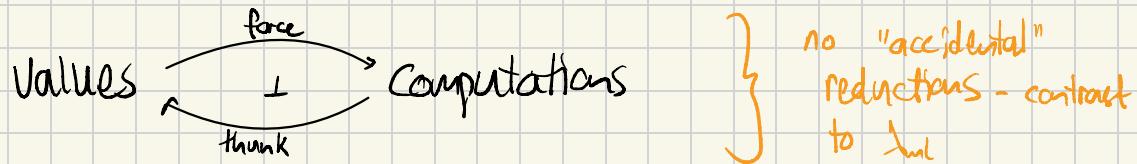
Why CBPV?

Subsumes both CBV and CBN

-- by giving fine control over
when effects happen

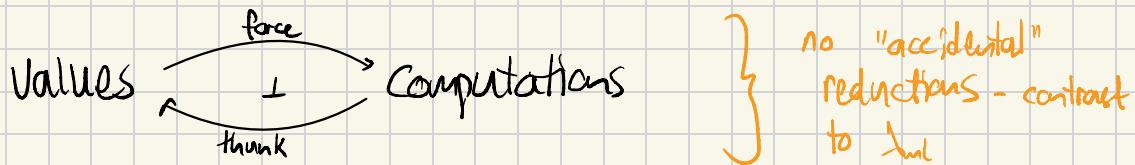
FEATURES OF CBPV

- Computations and values separate, but related:



FEATURES OF CBU

- Computations and values separate, but related:



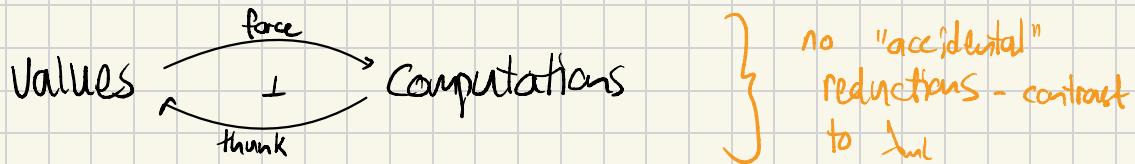
best of
both worlds

- "SUMS are nice in CBU" + "ARROWS are nice in CBU"
 - i.e. the sum type consists of things like $\text{inj } V$ for V a value

in CBU, effects commute with λ
so the η -law holds

FEATURES OF CBU

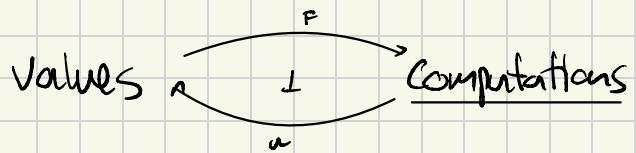
- Computations and values separate, but related:



best of
both worlds

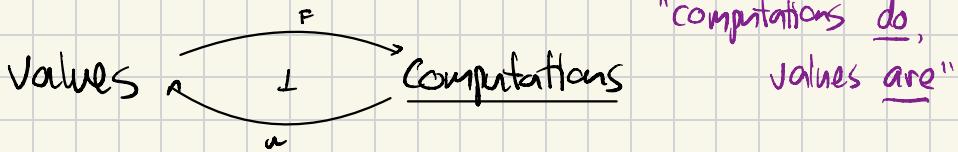
- "sums are nice in CBU" + "arrows are nice in CBU"
 - i.e. the sum type consists of things like $\text{inj } V$ for V a value in CBU, effects commute with λ so the η -law holds
- let-binding into any computation type, not just free ones
 - important for getting good embeddings of e.g. sum types from λ

Some syntax



"computations do,
values are"

Some syntax



"computations do,
values are"

$$\Gamma \vdash v : A$$

$$\Gamma \vdash^c \text{return } v : FA$$

values are trial
computations

contexts are values

$$\frac{\Gamma \vdash^c M : \underline{B}}{\Gamma \vdash \text{thunk } M : \underline{UB}}$$

$$\Gamma \vdash \text{thunk } M : \underline{UB}$$

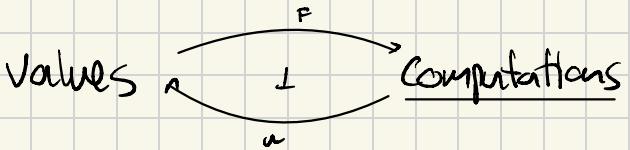
prevent a computation
running

$$\Gamma \vdash v : \underline{UB}$$

$$\Gamma \vdash^c \text{force } v : \underline{B}$$

allow the
computation to
run

Some syntax



"computations do,
values are"

$$\Gamma \vdash v : A$$

$$\frac{}{\Gamma \vdash^c \text{return } v : FA}$$

values are trial
computations

contexts are values

$$\frac{}{\Gamma \vdash^c M : \underline{B}}$$

$$\Gamma \vdash \text{thunk } M : \underline{UB}$$

prevent a computation
running

$$\frac{}{\Gamma \vdash v : \underline{UB}}$$

$$\Gamma \vdash^c \text{force } v : \underline{B}$$

allow the
computation to
run

CBV function type becomes

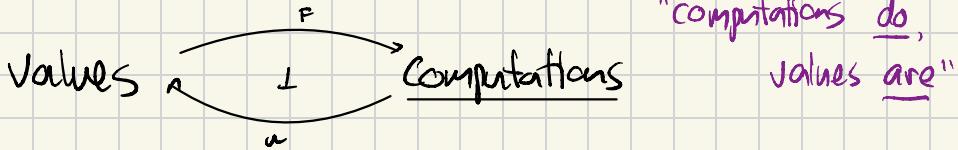
$$[A \rightarrow_{\text{CBV}} B] = U(A \rightarrow FB)$$

function types
are computations

$$[\lambda. M] = \text{thunk } \lambda. [M]$$

value = CBV
makes it a value

Some syntax



$$\frac{T \vdash V : A}{T \vdash^c \text{return } V : FA}$$

Values are trial
computations

$$\frac{}{T \vdash^c M : \underline{B}}$$

$$T \vdash \text{thunk } M : \underline{UB}$$

prevent a computation
running

$$\frac{}{T \vdash V : \underline{UB}}$$

$$T \vdash^c \text{force } V : \underline{B}$$

allow the
computation to
run

CBV function type becomes

$$[A \rightarrow_{\text{CBV}} B] = U(A \rightarrow FB)$$

function types
are computations

$$[\lambda x. M] = \text{thunk } \lambda x. [M]$$

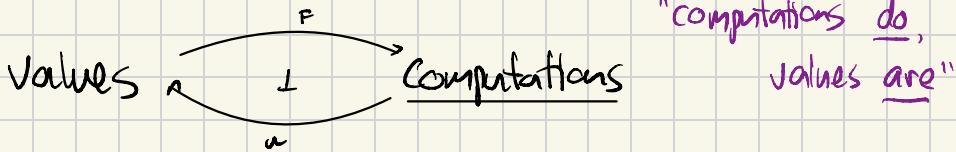
value = CBV
makes it a value

CBN function type becomes

$$[A \rightarrow_{\text{CBN}} B] = (U A) \rightarrow B$$

in CBN variables
are bound to unevaluated
terms, so we regard these
as thunks

Some syntax



"computations do,
values are"

$$\frac{\Gamma \vdash v : A}{\Gamma \vdash^c \text{return } v : FA}$$

values are trial
computations

contexts are values

$$\frac{\Gamma \vdash^c M : \underline{B}}{\Gamma \vdash \text{thunk } M : U\underline{B}}$$

$\Gamma \vdash \text{thunk } M : U\underline{B}$

prevent a computation
running

$$\frac{\Gamma \vdash v : U\underline{B}}{\Gamma \vdash^c \text{force } v : \underline{B}}$$

allow the
computation to
run

the monad
is still
there

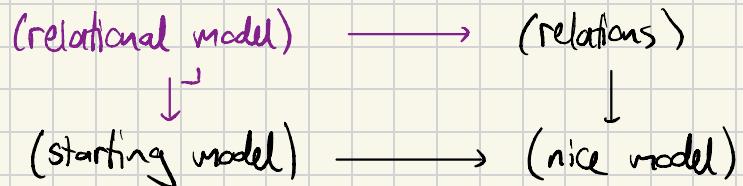
$$\frac{\Gamma \vdash v : A}{\frac{\Gamma \vdash \text{return } v : FA}{\Gamma \vdash \text{thunk}(\text{return } v) : U(FA)}}$$

What we want: a denotational account
of log. rel. for CBPV

What we want: a denotational account
of log. rel. for CBPV

1) CBPV fibration = fibration s.t. model
structure preserved

2) Universal construction of relations models



3) recover sensible syntactic defⁿ

$$\begin{aligned} &\{ R_A \mid A \in \text{ValType} \} \\ &\{ R_B \mid B \in \text{ComType} \} \end{aligned}$$

Semantics of C_{BNV} [Levy]

Need to interpret

$$\Gamma \vdash v : A$$

values

$$\Gamma \vdash^c M : \underline{B}$$

computations

$$\Gamma | B \vdash^k k : C$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$

Semantics of C_{BNV} [Levy]

Need to interpret

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B} \in \text{Cmp}$

$$\Gamma \vdash v : A$$

values

$$\Gamma \vdash^c M : \underline{B}$$

computations

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B}, \underline{C} \in \text{Cmp}$

$$\Gamma \mid \underline{B} \vdash^k k : \underline{C}$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$

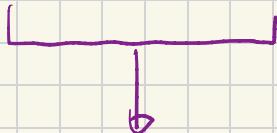
Semantics of CFPV [Levy]

Need to interpret

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B} \in \text{Cmp}$

$$\Gamma \vdash^c V : A$$

values



In a cartesian
category \mathbb{V} , ie.
 $\mathbb{V}(\underline{\Gamma}, \underline{A})$

$$\Gamma \vdash^c M : \underline{B}$$

computations



$$\begin{aligned} &\text{in } \mathbb{V}(\underline{\Gamma}, \underline{B}) \\ &\equiv \mathcal{C}_{\underline{\Gamma}}(F_1, \underline{B}) \end{aligned}$$

Need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B}, \underline{C} \in \text{Cmp}$

$$\Gamma \mid \underline{B} \vdash^k k : \underline{C}$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$



in a locally \mathbb{V} -indexed
category \mathcal{C} , ie. in
 $\mathcal{C}(\underline{B}, \underline{C})$

Semantics of CRPV [Levy]

DEFN: let \mathbb{V} be cartesian. A locally \mathbb{V} -indexed category \mathcal{G} has

- objects A, B, \dots
- for every $T \in \mathbb{V}, A, B \in \mathcal{G}$ a hom set $C_T(A, B)$ of arrows $f: A \xrightarrow[T]{} B$ over T

...

Semantics of CRPV [Levy]

DEFN: let \mathbb{V} be cartesian. A locally \mathbb{V} -indexed category \mathcal{C} has

- objects A, B, \dots
- for every $T \in \mathbb{V}, A, B \in \mathcal{C}$ a hom set $C_T(A, B)$ of arrows $f: A \xrightarrow[T]{} B$ over T

st. each $C_T(B)$ a category with the same objects,
compatible with maps in \mathbb{V} //

Semantics of CRPV

[Lévy]

a category enriched
in $[\mathbb{V}^*, \text{Set}]$

DEFN: let \mathbb{V} be cartesian. A locally \mathbb{V} -indexed category \mathcal{C} has

- objects A, B, \dots
- for every $T \in \mathbb{V}, A, B \in \mathcal{C}$ a hom set $C_T(A, B)$ of arrows $f: A \xrightarrow{T} B$ over T

st. each $C_T(B)$ a category with the same objects,
compatible with maps in \mathbb{V} //

EXAMPLES for \mathbb{W} cartesian

- self \mathbb{W} has objects as in \mathbb{W} and
$$(\text{self } \mathbb{W})_r(A, B) := \mathbb{W}(r \times A, B)$$

EXAMPLES for \mathbb{V} cartesian

- self \mathbb{V} has objects as in \mathbb{V} and
$$(\text{self } \mathbb{V})_r(A, B) := \mathbb{V}(r \times A, B)$$
- if T is a strong monad on \mathbb{V} , get $E(T)$ with objects T -algebras and maps $f: A \Rightarrow B$ Maps $f: r \times A \rightarrow B$ in \mathbb{V} that are right-linear

SEMANTICS

(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

SEMANTICS

(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

DEFⁿ: A CBPV model consists of

- a cartesian category \mathbb{V} for the values

SEMANTICS

(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

DEFⁿ: A CBPV model consists of

- a cartesian category \mathbb{V} for the values
- a locally \mathbb{V} -indexed category \mathcal{C} for stacks / computations

SEMANTICS

(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

DEFN: A CBPV model consists of

- a cartesian category \mathbb{V} for the values
- a locally \mathbb{V} -indexed category \mathcal{C} for stacks/
computations
- a locally \mathbb{V} -indexed adjunction i.e. an adjunction
in $[\mathbb{V}, \text{Set}]$ -Cat

$$\text{Self } \mathbb{V} \begin{array}{c} \xleftarrow{\quad \perp \quad} \\[-1ex] \xrightarrow{\quad F \quad} \end{array} \mathcal{C}$$



$$\mathbb{V}(F \times \underline{B}) \cong \mathcal{C}_F(F \times \underline{B})$$

SEMANTICS

(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

DEFN: A CBPV model consists of

- a cartesian category \mathbb{V} for the values
- a locally \mathbb{V} -indexed category \mathcal{C} for stacks / computations
- a locally \mathbb{V} -indexed adjunction i.e. an adjunction
in $[\mathbb{V}, \text{Set}]$ -Cat $\mathbb{V} \dashv \mathcal{C}$

$$\text{Self } \mathbb{V} \begin{array}{c} \xrightarrow{\quad F \quad} \\ \Downarrow \perp \\ \xleftarrow{\quad u \quad} \end{array} \mathcal{C}$$

$$\mathbb{V}(F \times \underline{u} \mathcal{B}) \cong \mathcal{C}_F(F \times \underline{B})$$

+ structure for $\rightarrow, +, \times$ etc.

SEMANTICS

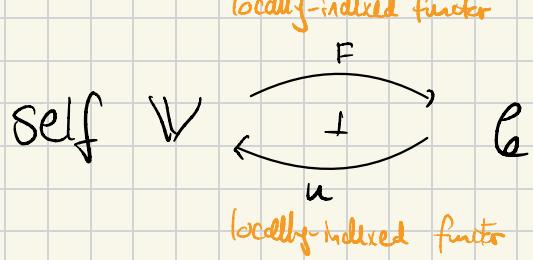
(values trivially
indexed over themselves) \hookrightarrow (computations indexed
over values)

VALUE TYPES

$$[A] \in \mathbb{V}$$

COMPUTATION TYPES

$$[B] \in \mathcal{C}$$



unit + counit give
rise to thunk / force

Semantics of C_{BNV} [Levy]

Need to interpret

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B} \in \text{Cmp}$

$$\Gamma \vdash v : A$$

values

$$\Gamma \vdash^c M : \underline{B}$$

computations

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B}, \underline{C} \in \text{Cmp}$

$$\Gamma \mid \underline{B} \vdash^k k : \underline{C}$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$

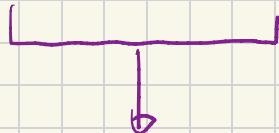
Semantics of CFPV [Levy]

Need to interpret

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B} \in \text{Cmp}$

$$\Gamma \vdash^{\sim} V : A$$

values



In a cartesian
category \mathbb{V} , ie.
 $\mathbb{V}(\mathbb{C}^{\text{F}}, \mathbb{C}^{\text{D}})$

$$\Gamma \vdash^c M : \underline{B}$$

computations

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B}, \underline{C} \in \text{Cmp}$

$$\Gamma \mid \underline{B} \vdash^k k : \underline{C}$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$



in a locally \mathbb{V} -indexed
category \mathbb{E} , ie. in
 $\mathbb{E}(\mathbb{C}^{\text{F}}, \mathbb{C}^{\text{D}})$

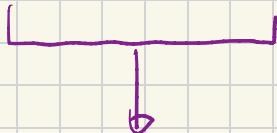
Semantics of CFPV [Levy]

Need to interpret

need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B} \in \text{Cmp}$

$$\Gamma \vdash^c V : A$$

values



In a cartesian
category \mathbb{V} , ie.
 $\mathbb{V}(\underline{\Gamma}, \underline{A})$

$$\Gamma \vdash^c M : \underline{B}$$

computations



$$\text{in } \mathbb{V}(\underline{\Gamma}, \underline{B}) \\ \cong \mathcal{C}_{\underline{\Gamma}}(F1, \underline{B})$$

Need hands depending
on $\Gamma \in \text{Val}$ and $\underline{B}, \underline{C} \in \text{Cmp}$

$$\Gamma \mid \underline{B} \vdash^k k : \underline{C}$$

stacks / contexts

cf. $C[m]$ for a uni term m
and context $C[-]$



in a locally \mathbb{V} -indexed
category \mathcal{C} , ie. in
 $\mathcal{C}_{\underline{\Gamma}}(\underline{B}, \underline{C})$

EXAMPLES:

Many more in the
CBPU book!

1) ALGEBRA MODELS

for T a strong monad
on \mathbb{V}

$$\text{Self } \mathbb{V} \xrightarrow{\perp} E(T)$$

free forget

Many more in the
CBPU book!

EXAMPLES:

1) ALGEBRA MODELS

for T a strong monad
on \mathbb{V}

$$\text{Self } \mathbb{V} \xleftrightarrow{\perp} E(T)$$

free
forget

2) STATE MODELS

for a ccc \mathcal{C} and $S\in\mathcal{C}$

$$\text{self } \mathcal{C} \xleftrightarrow{\perp} \text{self } \mathcal{C}$$

$S \Rightarrow (-)$
 $(-) \times S$

Many more in the
CBPU book!

EXAMPLES:

1) ALGEBRA MODELS

for T a strong monad
on \mathbb{V}

$$\text{Self } \mathbb{V} \xleftrightarrow{\perp} E(T)$$

free
forget

2) STATE MODELS

for a ccc C and $S \in C$

$$\text{Self } C \xleftrightarrow{\perp} \text{Self } C$$

$S \Rightarrow (-)$
 $(-) \times S$

3) CONTINUATION MODELS

for a ccc C and $R \in C$

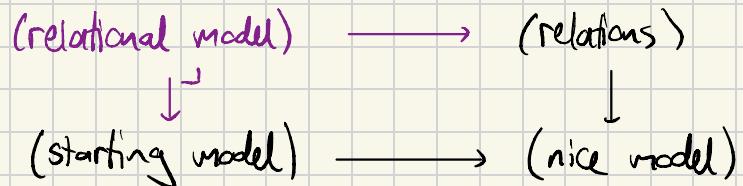
$$\text{Self } C \xleftrightarrow{\perp} \text{Self } C^P$$

$- \Rightarrow R$
 $- \rightarrow R$

What we want: a denotational account
of log. rel. for CBPV

1) CBPV fibration = fibration s.t. model
structure preserved

2) Universal construction of relations models



3) recover sensible syntactic defⁿ

$$\begin{aligned} &\{ R_A \mid A \in \text{ValType} \} \\ &\{ R_B \mid B \in \text{ComType} \} \end{aligned}$$

Qn: what is a CBPV fibration?

Prior work in the
algebras cafe, see
eg. Kommar, McDowell.

Qn: what is a CBPV fibration?

Prior work in the
algebras case, see
eg. Kommar, McDowell.

It can't be a fibration that
preserves the structure!

... at least, not immediately. Why not?

Qn: what is a CBPV fibration?

Prior work in the
algebras case, see
eg. Kommar, McDowell.

It can't be a fibration that
preserves the structure!

... at least, not immediately. Why not?

What is a fibration of locally indexed categories?

Qn: what is a CBPV fibration?

A recipe:

Qn: what is a CBPV fibration?

A recipe: 1) Define a 2-category of
locally-indexed categories

Qn: what is a CBPV fibration?

A recipe:

- 1) Define a 2-category of
locally-indexed categories
- 2) Use this to define locally-indexed fibrations
as fibrations internal to this 2-category

Qn: what is a CBPV fibration?

- A recipe:
- 1) Define a 2-category of locally-indexed categories
 - 2) Use this to define locally-indexed fibrations
 - 3) A CBPV fibration is a locally-indexed fibration that preserves the structure

CBPV fibrations

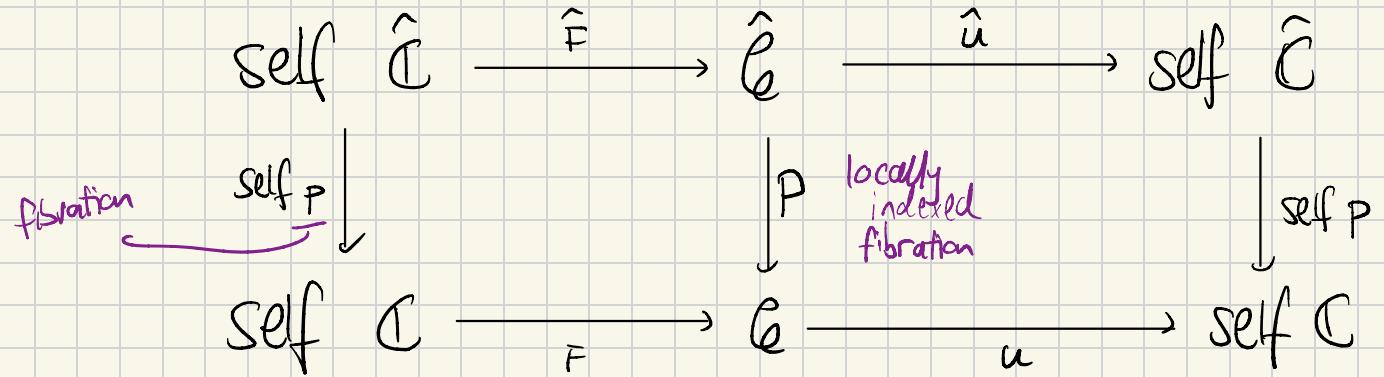
$$\text{self } \mathbb{C} \xrightarrow{F} \mathcal{G} \xrightarrow{u} \text{self } \mathbb{C}$$

CBPV fibrations

$$\text{self } \hat{\mathbb{C}} \xrightarrow{\hat{F}} \hat{\mathcal{C}} \xrightarrow{\hat{u}} \text{self } \hat{\mathbb{C}}$$

$$\text{self } \mathbb{C} \xrightarrow{F} \mathcal{C} \xrightarrow{u} \text{self } \mathbb{C}$$

CBPV fibrations



in the 2-category of
locally-indexed categories

CBPV fibrations

$$\begin{array}{ccccc} \text{self } \hat{\mathcal{C}} & \xrightarrow{\hat{F}} & \hat{\mathcal{G}} & \xrightarrow{\hat{u}} & \text{self } \hat{\mathcal{C}} \\ \text{fibration} \curvearrowright \text{self } P & \downarrow & P & \text{locally indexed} & \downarrow \text{self } P \\ \text{self } \mathcal{C} & \xrightarrow{F} & \mathcal{G} & \xrightarrow{u} & \text{self } \mathcal{C} \end{array}$$

... st the adjunction structure is preserved

from the general theory, we get :

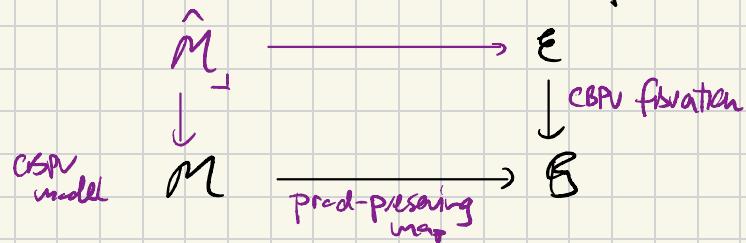
= some nice 2-category theory

from the general theory, we get :

- 1) nice closure properties

from the general theory, we get :

- 1) nice closure properties
- 2) universal constructions of relational models:



from the general theory, we get:

- 1) nice closure properties
- 2) universal constructions of relational models:

$$\begin{array}{ccc} \hat{M} & \xrightarrow{\quad} & E \\ \downarrow & & \downarrow \text{CBPV fibration} \\ M & \xrightarrow{\quad \text{prod-preserving map} \quad} & B \end{array}$$

- 3) locally-indexed versions of key fibrations
for new-from-old (e.g. subobject, codomain)
+ recovers the syntactic defⁿ of McDermott

from the general theory, we get:

- 1) nice closure properties
- 2) universal constructions of relational models:

$$\begin{array}{ccc} \hat{M} & \xrightarrow{\quad} & E \\ \downarrow & & \downarrow \text{CBV fibration} \\ M & \xrightarrow{\quad \text{prod-preserving map} \quad} & B \end{array}$$

* if really is the same
if you do enough
abstract nonsense

- 3) locally-indexed versions of key fibrations
for new-from-old (e.g. subobject, codomain)
+ recovers the syntactic defⁿ of McDermott

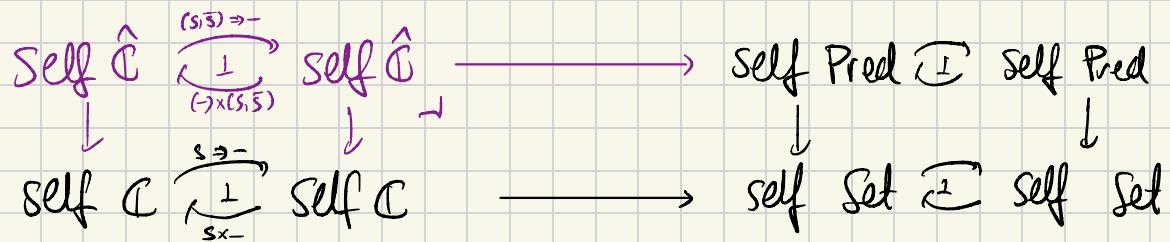
~ so essentially the theory we had for STCC / CBV *

EXAMPLES.

1) algebra models : coincides with monad lifting

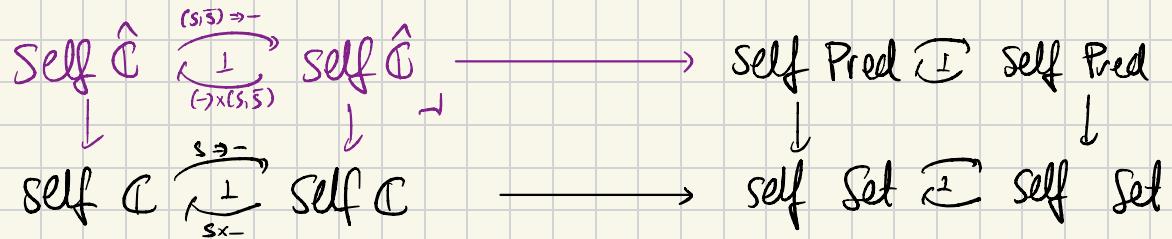
EXAMPLES.

- 1) algebra models : coincides with monad lifting
- 2) state / continuation models as expected :



EXAMPLES.

- 1) algebra models : coincides with monad lifting
- 2) state / continuation models as expected :



get logical relations as families of relations

$$\{R_A \mid A \in \text{ValType}\} , \{R_B \mid B \in \text{ComType}\}$$

EFFECT SIMULATION

for algebra models for L and $SFin$

on Set

algebras
= monoids

algebras
= sup-semilattices

EFFECT SIMULATION

for algebra models for L and $Sfin$
on Set

\downarrow
 \downarrow
algebras
= monoids
algebras
= sup-semilattices

$$\begin{array}{ccc} \text{Set} \times \text{Set} \subseteq \text{Slat} \times \text{Mon} & \xrightarrow{H} & \text{Set} \times \text{Set} \subseteq \text{Mon} \times \text{Mon} \\ & & \downarrow \\ \text{BinPud} \hookrightarrow \text{BinPudMon} & & \text{obj: } (X, Y, R \subseteq X \times Y) \\ & & \text{but } X, Y \text{ are monoids} \\ & & \text{and } R \hookrightarrow X \times Y \end{array}$$

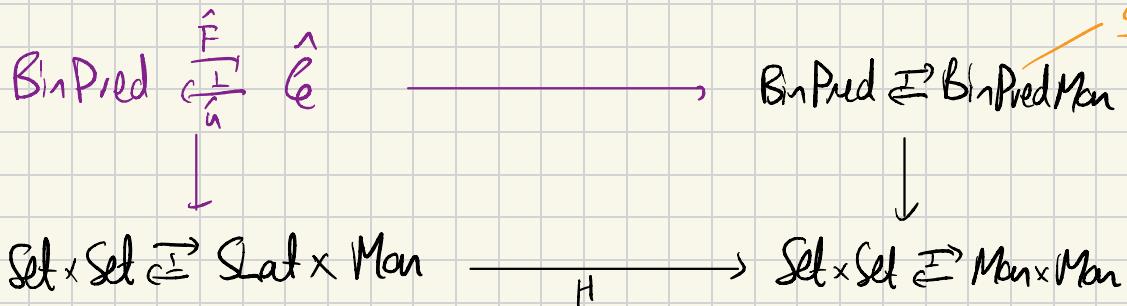
L takes the effect into account!

EFFECT SIMULATION

for algebra models for L and Spin
on Set

\uparrow \downarrow
algebras
= sub-semilattices

def: $(X, M, R \subseteq U(X \times M))$
 struktur
 submanifold



j : $(X, Y, R \subseteq X \times Y)$
 but $X \times Y$ are monoids
 and $R \supseteq X \times Y$

L takes the effect into account!

EFFECT SIMULATION

for algebra models for L and S_{fin}
 on Set



algebras
= monoids



S_{fin}
algebras
= sup-semilattices

obj: $(X, M, R \subseteq X \times M)$
 monoid
 sup-semilattice
 submonoid

BinPred $\xleftarrow[\frac{F}{G}]$ $\hat{\ell}$

$Set \times Set \subseteq Slat \times Mon$



BinPred \hookrightarrow BinPredMon



$Set \times Set \hookrightarrow Mon \times Mon$

obj: $(X, Y, R \subseteq X \times Y)$

but X, Y are monoids
 and $R \supseteq X \times Y$

L takes the effect into account!

$$\hat{F}(A, B, R) = (S_{fin}A, LB, \hat{F}R)$$

where $(P, e) \in \hat{F}R$ iff a
 "bisimulation" property holds

WE NOW HAVE

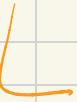
- framework for building lots of CBRV models
- denotational view on CBRV logical relations
- mathematical account encompassing both
CBV and CBRV

monad
models

adjunction
models

Summing up:

- logical relations, denotationally \approx fibrations that preserve the model

 gives an elegant framework for relational models; justifies semantically the "logical relations conditions"

Summing up:

- logical relations, denotationally \approx fibrations that preserve the model
 - ↳ gives an elegant framework for relational models; justifies semantically the "logical relations conditions"
- for CBU this is harder, because the models are not plain categories

Summing up:

- logical relations, denotationally \approx fibrations that preserve the model
 - ↳ gives an elegant framework for relational models; justifies semantically the "logical relations conditions"
- for CBU this is harder, because the models are not plain categories
- ... but by using some 2-category theory we can get back to a framework as nice as that for STLC / CBV.