Constructing fully-abstract models of effectful λ-calculi

Ohad Kammar[†] and Shin-ya Katsumata[*] and Philip Saville[‡]

[†]School of Informatics
University of Edinburgh

*National Institute of Informatics
Tokyo

[‡]Department of Computer Science
University of Oxford

preprint at cs.ox.ac.uk/people/philip.saville/home.html

Slogan: models* go in, fully abstract models come out

Contextual equivalence [Morris, Milner,. . .]

$$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma \qquad \Longleftrightarrow \qquad \mathcal{C}[M] \Downarrow V \iff \mathcal{C}[M'] \Downarrow V$$

$\mathcal{C}[-]$ any closed ground context

Contextual equivalence [Morris, Milner,. . .]

$$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma \qquad \Longleftrightarrow \qquad \mathcal{C}[M] \Downarrow V \iff \mathcal{C}[M'] \Downarrow V$$

$\mathcal{C}[-]$ any closed ground context

swapping M and M'
doesn't affect
observable behaviour

Contextual equivalence [Morris, Milner,. . .]

$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma$

w.r.t. a semantics $[\![-]\!]$

$\Longleftrightarrow$

$[\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$

$\mathcal{C}[-]$ any closed ground context

swapping M and M'
doesn't affect
observable behaviour

Contextual equivalence [Morris, Milner,...]

$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma$ w.r.t. a semantics $[\![-]\!]$

$\Longleftrightarrow$

$[\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$ $\mathcal{C}[-]$ any closed ground context

swapping M and M' doesn't affect observable behaviour

Reasoning about $\simeq_{\mathrm{ctx}}$ is hard! Want semantic techniques round this

Contextual equivalence [Morris, Milner,...]

$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma$ $\iff$ $[\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$

w.r.t. a semantics $[\![-]\!]$

$\mathcal{C}[-]$ any closed ground context

swapping M and M'
doesn't affect
observable behaviour

How does semantic equality relate to $\simeq_{\mathrm{ctx}}$?

Adequacy: $[\![M]\!] = [\![M']\!] \implies M \simeq_{\mathrm{ctx}} M'$

Full abstraction: $M \simeq_{\mathrm{ctx}} M' \implies [\![M]\!] = [\![M']\!]$

Contextual equivalence [Morris, Milner,...]

$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma$ $\iff$ $[\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$

w.r.t. a semantics $[\![-]\!]$ — $\mathcal{C}[-]$ any closed ground context

swapping M and M'
doesn't affect
observable behaviour

How does semantic equality relate to $\simeq_{\mathrm{ctx}}$?

Adequacy: $[\![M]\!] = [\![M']\!] \implies M \simeq_{\mathrm{ctx}} M'$ ⇝ immediate!

Full abstraction: $M \simeq_{\mathrm{ctx}} M' \implies [\![M]\!] = [\![M']\!]$

Contextual equivalence [Morris, Milner,...]

$\Gamma \vdash M \simeq_{\mathrm{ctx}} M' : \sigma$ w.r.t. a semantics $[\![-]\!]$

$\iff$

$[\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$ $\mathcal{C}[-]$ any closed ground context

swapping M and M' doesn't affect observable behaviour

How does semantic equality relate to $\simeq_{\mathrm{ctx}}$?

Adequacy: $[\![M]\!] = [\![M']\!] \implies M \simeq_{\mathrm{ctx}} M' \rightsquigarrow$ immediate!

Full abstraction: $M \simeq_{\mathrm{ctx}} M' \implies [\![M]\!] = [\![M']\!]$

In an adequate, fully abstract model
semantic equality characterises contextual equivalence

The quest for full abstraction

late 1960s	roots of the definition
1975	Milner introduces full abstraction
1977	Plotkin: domains model for PCF is not fully abstract
80s & 90s	attempts to classify "sequentiality" + lots more
late 90s	games models,
	O'Hearn & Riecke's domains + logical relations model
21st C	Marz, Riecke, Ehrhard *et al.*, Matache *et al.*, ...
	$\rightsquigarrow$ mainly focussed on languages with recursion

This work: cranking the handle

signature S

= chosen base types, effect operations, & primitives

+

semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s

fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums

inspired by O'Hearn & Riecke's model

concrete over $\mathcal{M}$:
maps in $\mathrm{OHR}(\mathcal{M})$ are
maps in $\mathcal{M}$ satisfying predicates

This work: cranking the handle

signature S

= chosen base types, effect operations, & primitives

+

semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s

fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums

inspired by O'Hearn & Riecke's model

concrete over $\mathcal{M}$:
maps in $\mathrm{OHR}(\mathcal{M})$ are
maps in $\mathcal{M}$ satisfying predicates

This work: cranking the handle

signature S

= chosen base types, effect operations, & primitives

+

semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s

$\Downarrow$

determines an interpretation
$s[\![\Gamma \vdash M : \sigma]\!] : s[\![\Gamma]\!] \to Ts[\![\sigma]\!]$

fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums

inspired by O'Hearn & Riecke's model

concrete over $\mathcal{M}$:
maps in $\mathrm{OHR}(\mathcal{M})$ are
maps in $\mathcal{M}$ satisfying predicates

This work: cranking the handle

signature S

= chosen base types, effect operations, & primitives

$+$

semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s

$\Downarrow$

determines an interpretation
$s[\![\Gamma \vdash M : \sigma]\!] : s[\![\Gamma]\!] \to Ts[\![\sigma]\!]$

fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums

inspired by O'Hearn & Riecke's model

concrete over $\mathcal{M}$:
maps in $\mathrm{OHR}(\mathcal{M})$ are
maps in $\mathcal{M}$ satisfying predicates

This work: cranking the handle

signature S

= chosen base types, effect operations, & primitives

$+$

semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s

$\Downarrow$

determines an interpretation
$s[\![\Gamma \vdash M : \sigma]\!] : s[\![\Gamma]\!] \to Ts[\![\sigma]\!]$

fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums

inspired by O'Hearn & Riecke's model

concrete over $\mathcal{M}$:
maps in $\mathrm{OHR}(\mathcal{M})$ are
maps in $\mathcal{M}$ satisfying predicates

e.g. small subcategory Set_κ of Set
+ any monad
+ any interpretation
with a constant $\underline{b}$ for every $b \in s[\![\beta]\!]$

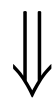
signature S

= chosen base types, effect operations, & primitives

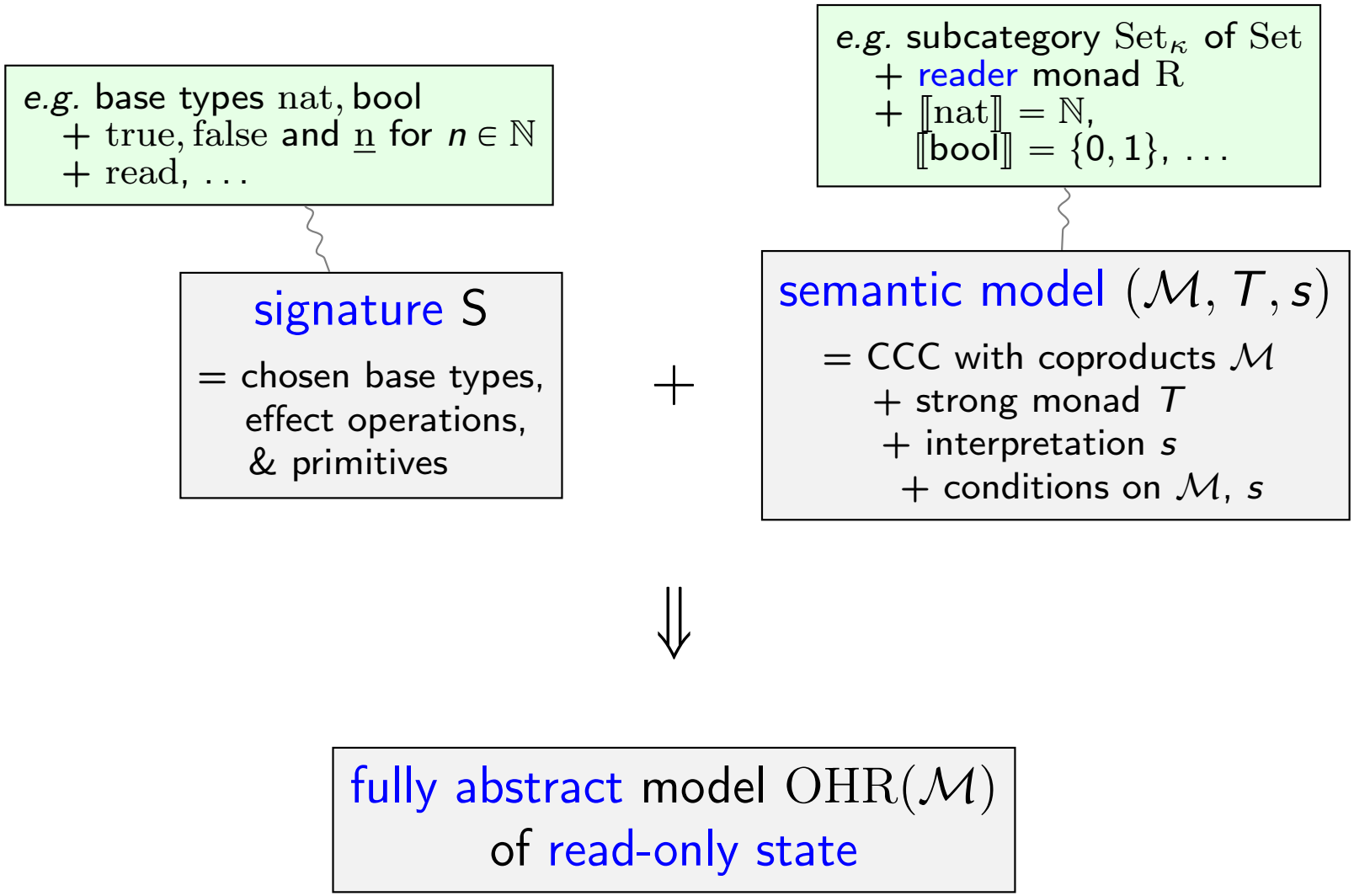
+

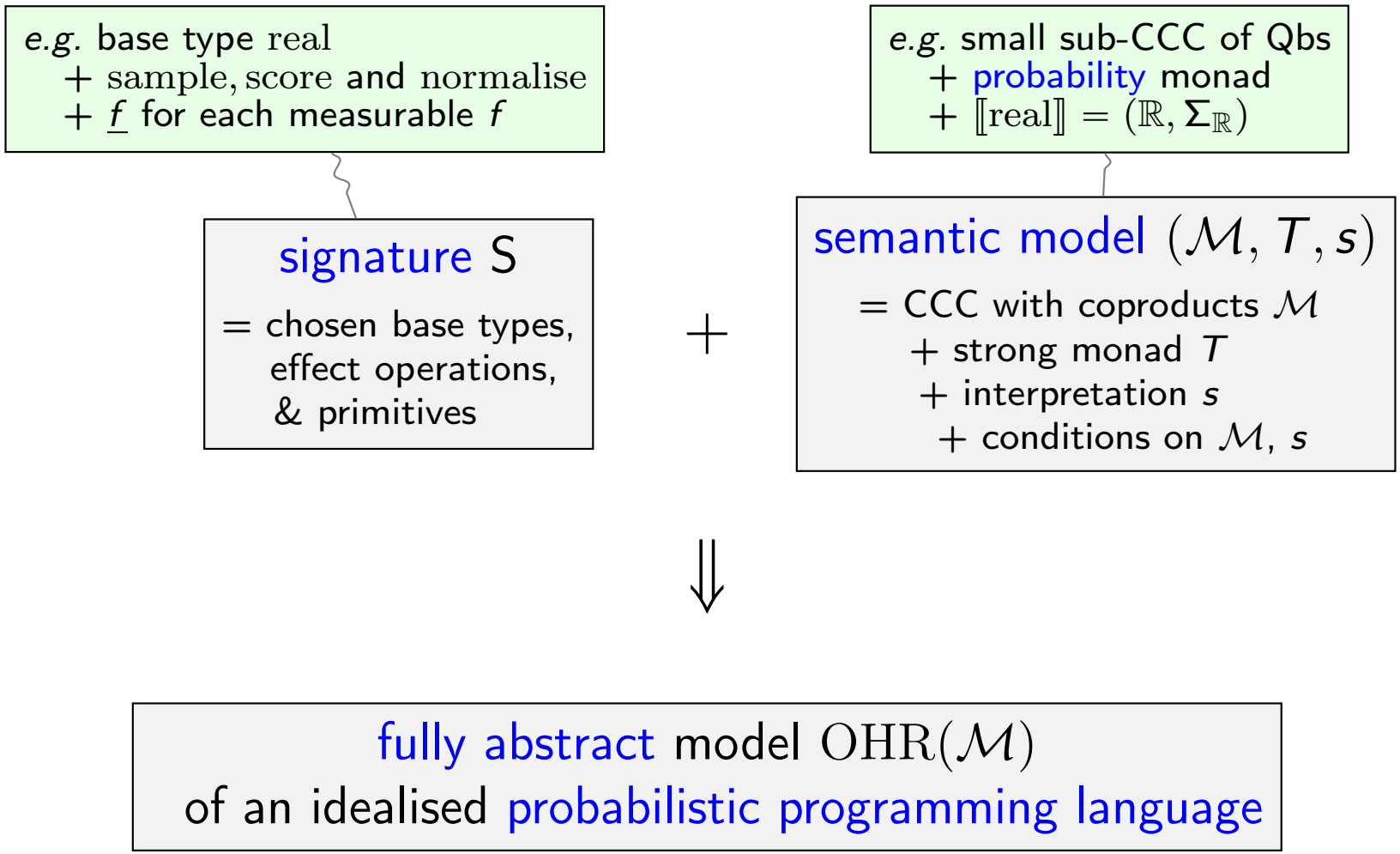
semantic model $(\mathcal{M}, T, s)$

= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s



fully abstract model $\mathrm{OHR}(\mathcal{M})$
of computational λ-calculus + constants + sums


e.g. base types nat, bool
+ $\mathrm{true}, \mathrm{false}$ and $\underline{\mathrm{n}}$ for $n \in \mathbb{N}$
+ read, ...
e.g. subcategory Set_κ of Set
+ reader monad R
+ $[\![\mathrm{nat}]\!] = \mathbb{N}$,
$[\![\mathrm{bool}]\!] = \{0, 1\}$, ...
signature S
= chosen base types,
effect operations,
& primitives
+
semantic model $(\mathcal{M}, T, s)$
= CCC with coproducts $\mathcal{M}$
+ strong monad T
+ interpretation s
+ conditions on $\mathcal{M}$, s
⇓
fully abstract model $\mathrm{OHR}(\mathcal{M})$
of read-only state


e.g. base type real
+ sample, score and normalise
+ f for each measurable f
e.g. small sub-CCC of Qbs
+ probability monad
+ [[real]] = (ℝ, Σ_ℝ)
signature S
= chosen base types,
effect operations,
& primitives
+
semantic model (M, T, s)
= CCC with coproducts M
+ strong monad T
+ interpretation s
+ conditions on M, s
⇓
fully abstract model OHR(M)
of an idealised probabilistic programming language

The big picture

The big picture

Obstruction to $(\mathcal{M}, T, s)$ being fully abstract:

$\exists$ morphisms in $\mathcal{M}$ expressing behaviour the syntax cannot

[*c.f.* parallel-or]

The big picture

Obstruction to $(\mathcal{M}, T, s)$ being fully abstract:

$\exists$ morphisms in $\mathcal{M}$ expressing behaviour the syntax cannot

[*c.f.* parallel-or]

Solution:

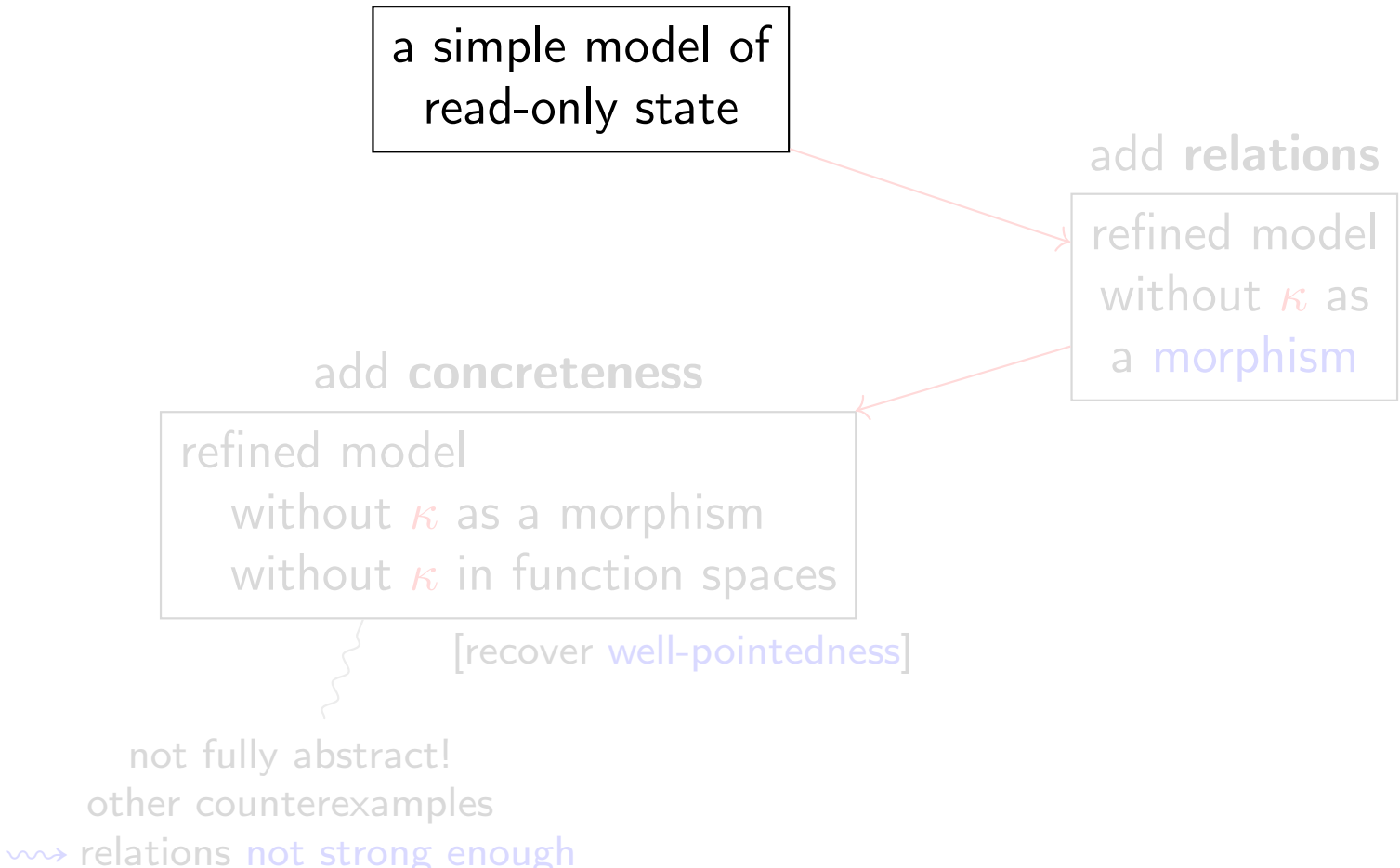
remove all such counterexamples to contextual equivalence

Up next: a recipe that doesn't quite work

. . . but is the template for our construction

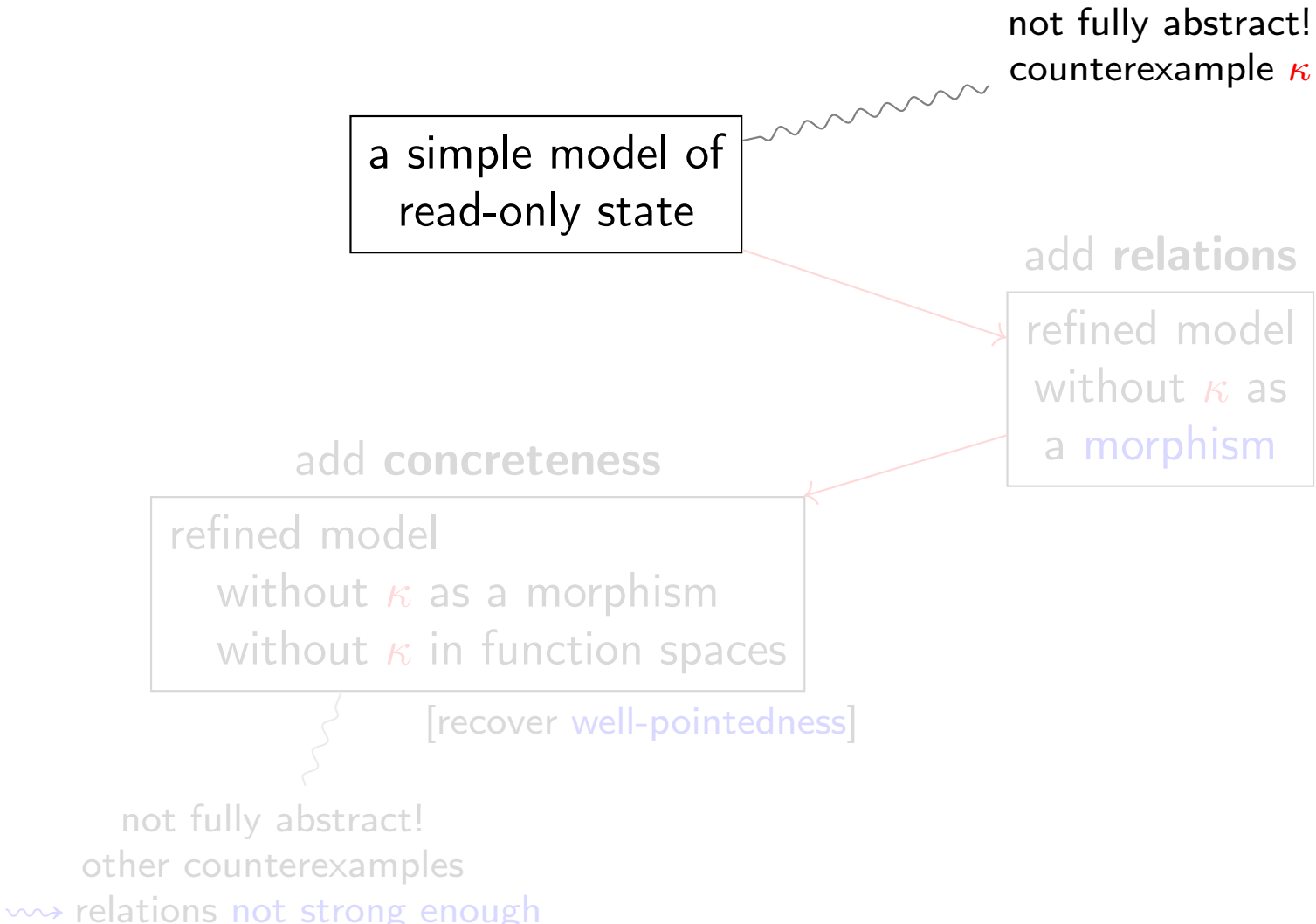
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



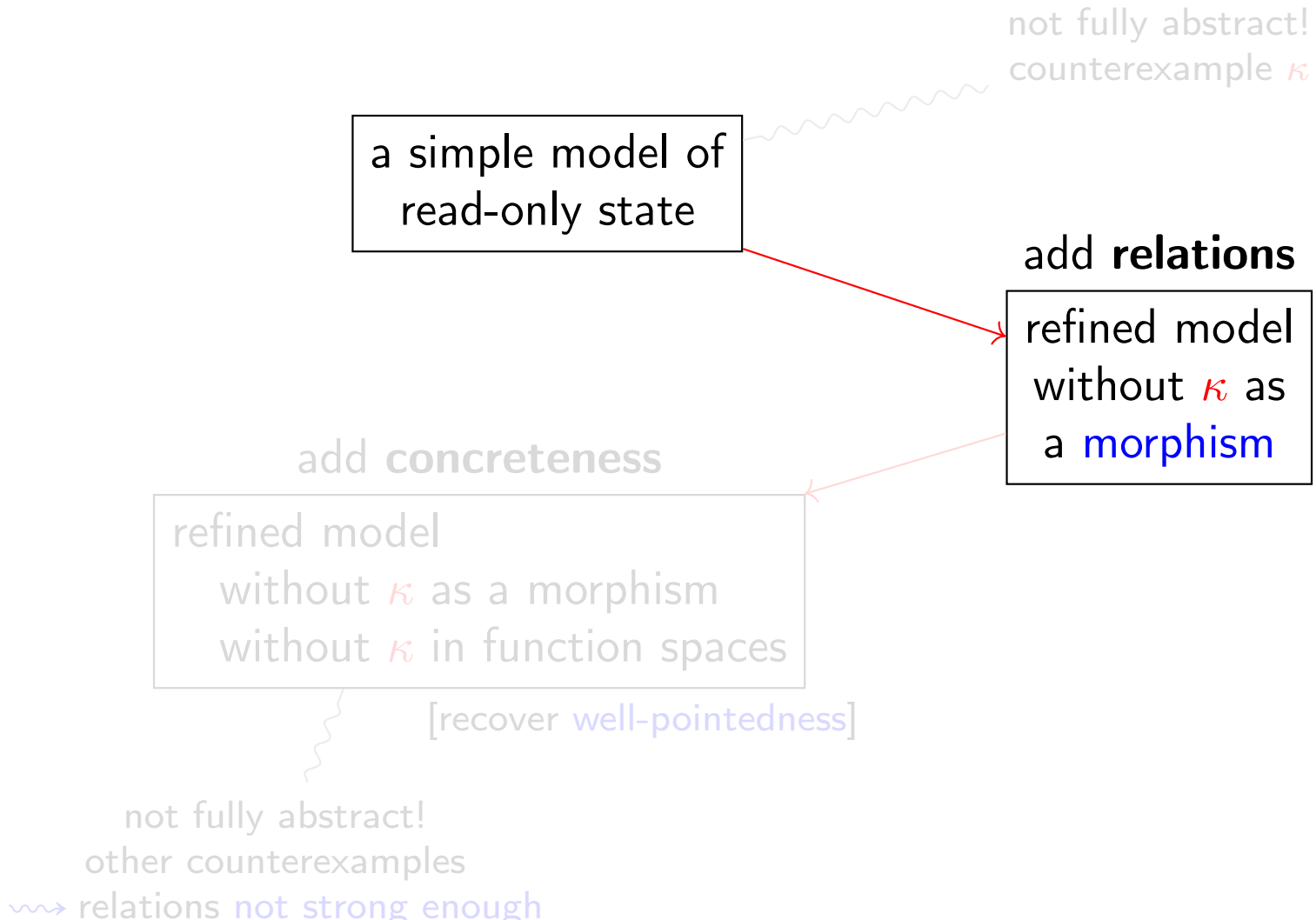
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



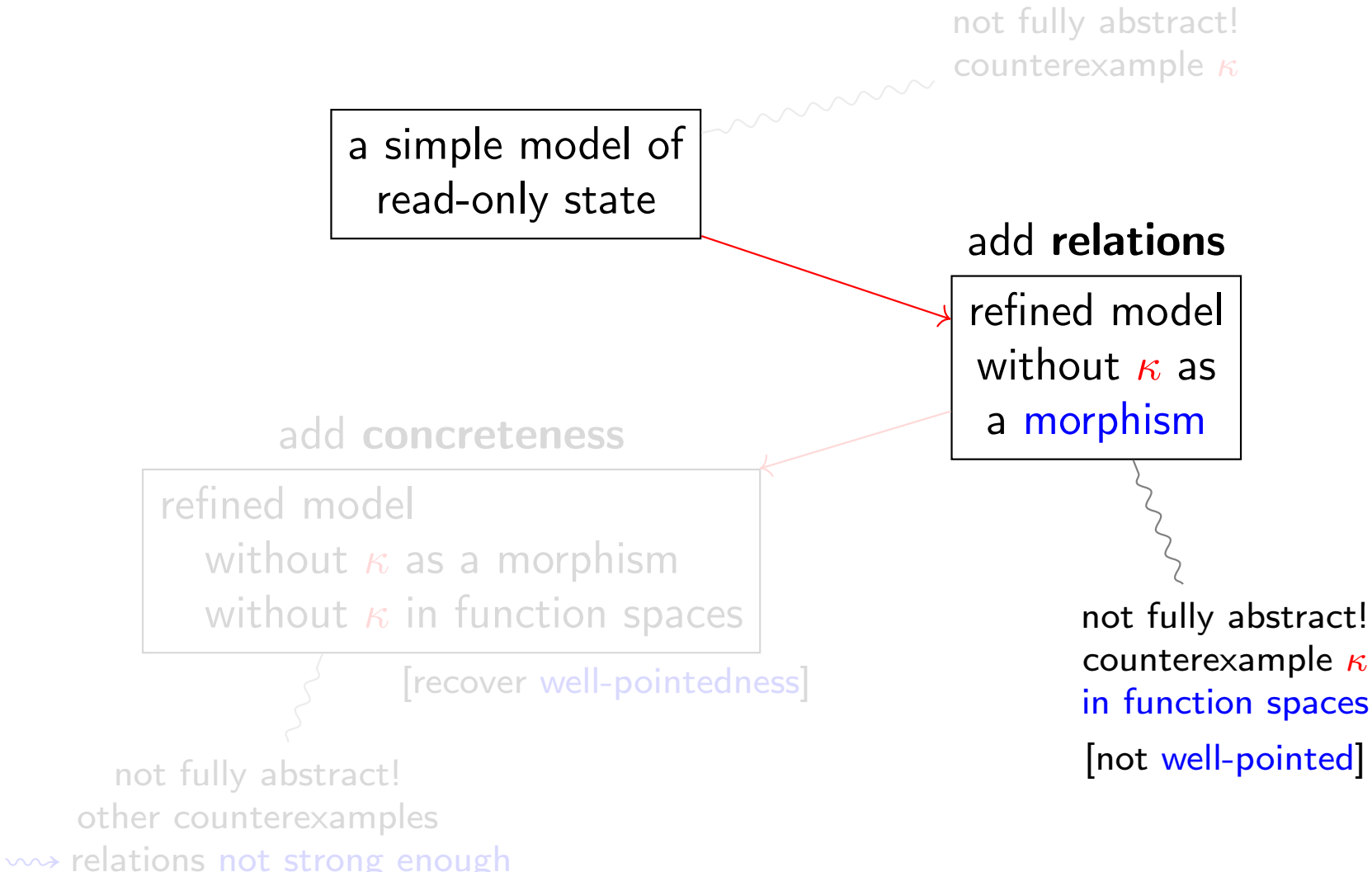
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



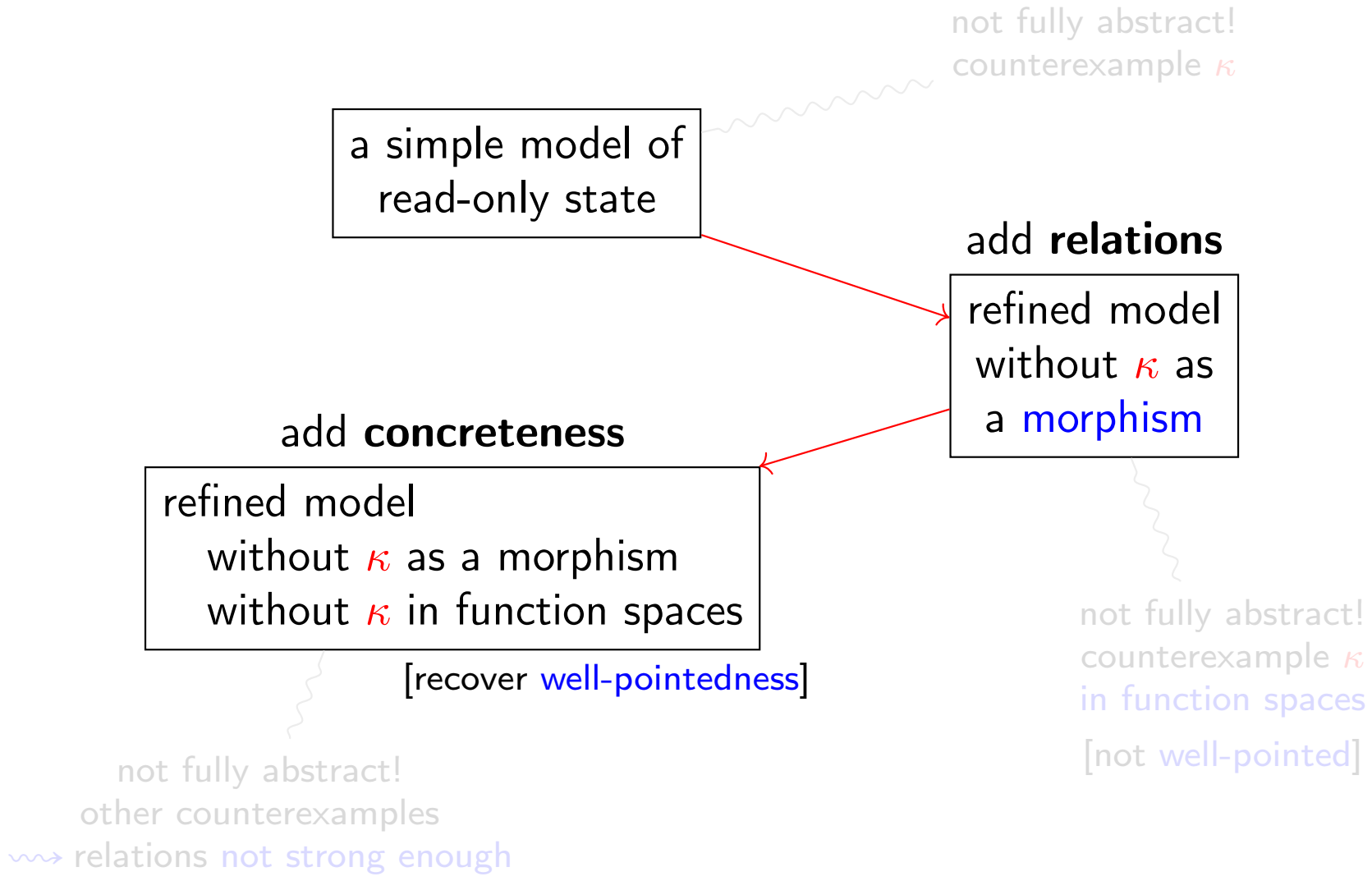
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



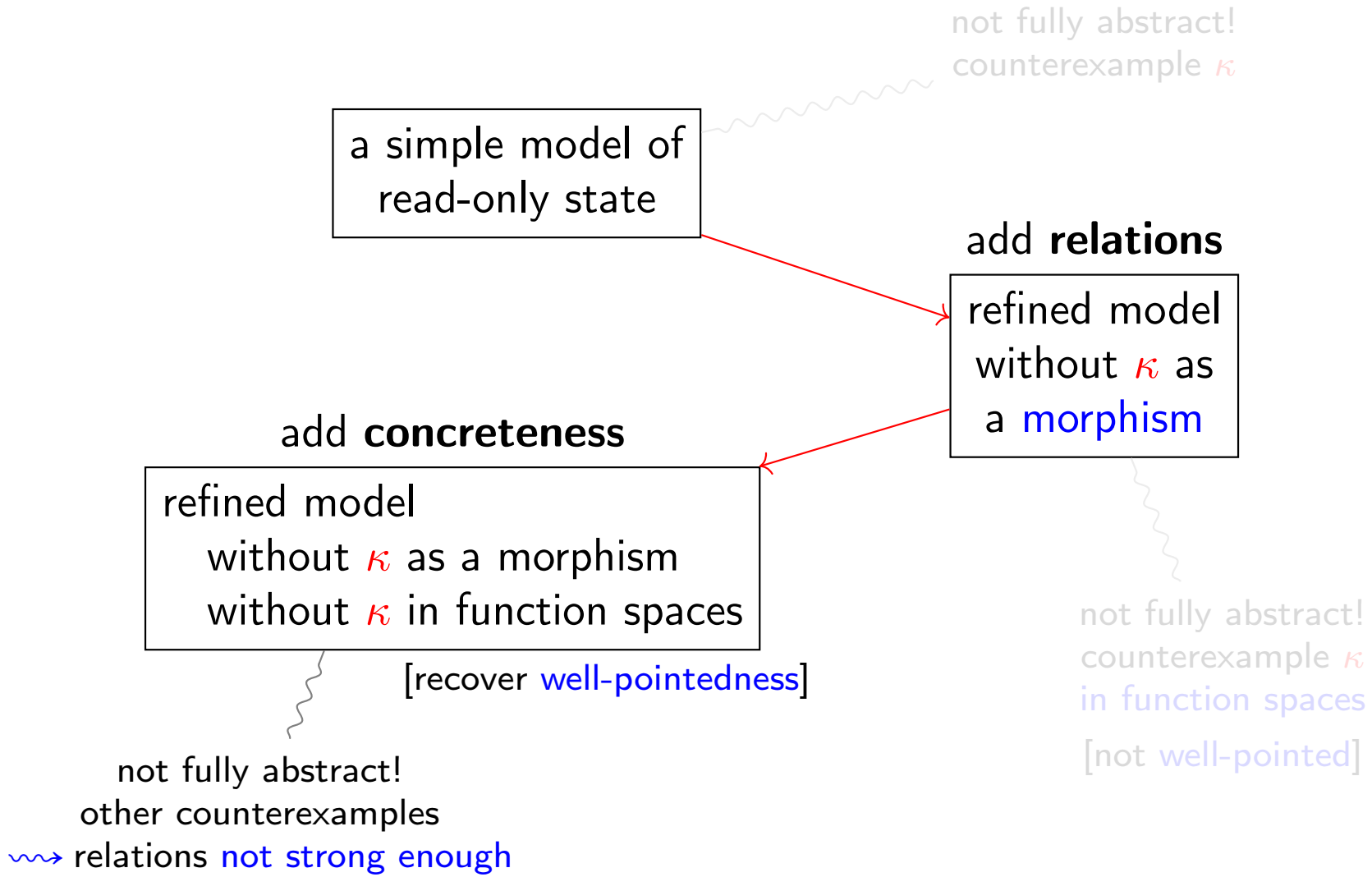
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



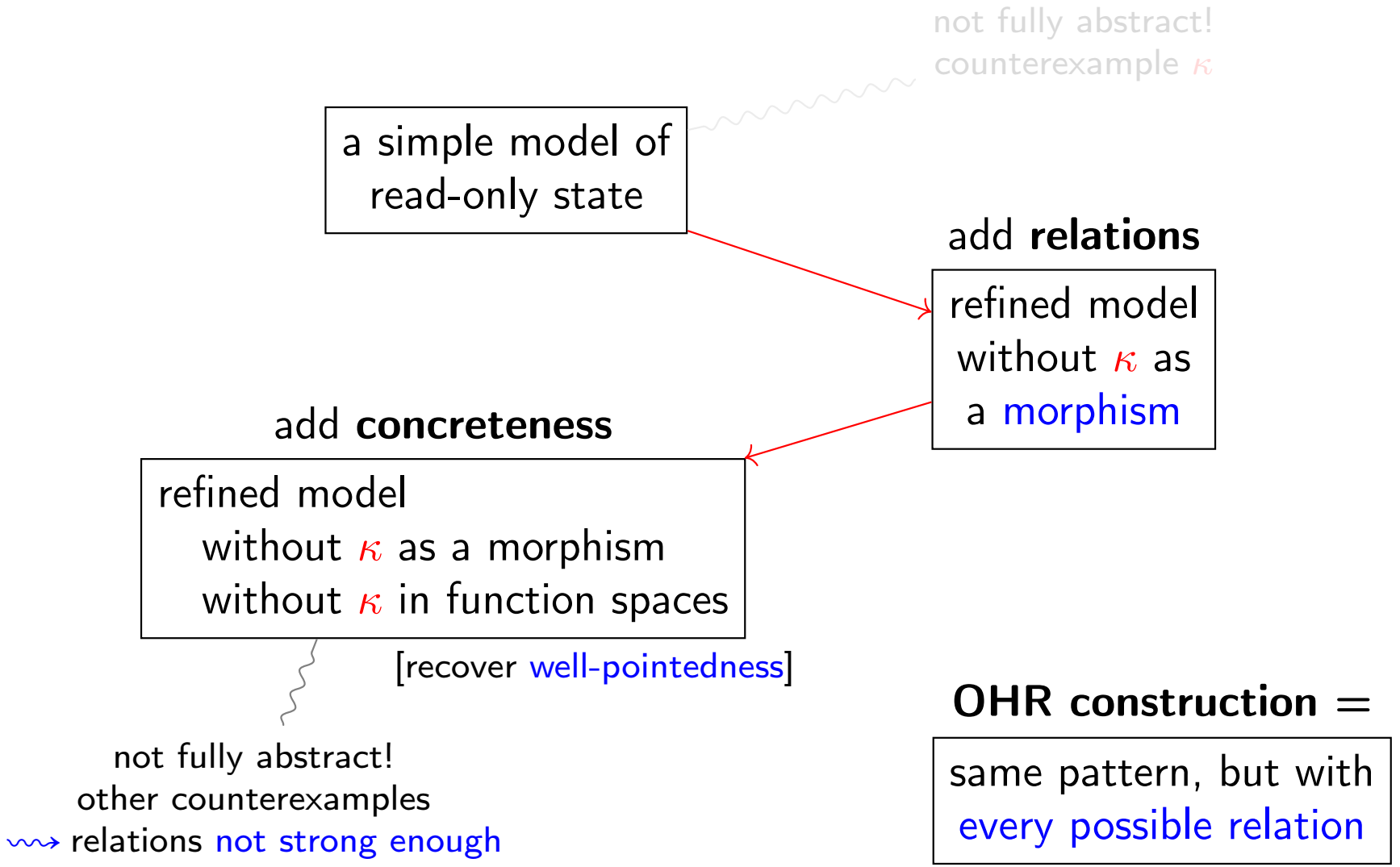
Up next: a recipe that doesn't quite work

. . . but is the template for our construction



Up next: a recipe that doesn't quite work

. . . but is the template for our construction



A motivating example: read-only state [Matache & Staton]

Idea [omitting sums for now]

1. A global, one-bit memory cell
2. You can read, but not write

The signature

types $\tau ::= \texttt{bool} \mid \tau * \tau \mid 1 \mid \tau \to \tau$

types τ ::= bool | τ * τ | 1 | $\tau \to \tau$

terms M ::=

STLC

x	# variables
| (M,M)	# product types
| π_i(M)	
| ()	
| λx . M	# function types
| M M	

types τ ::= bool | τ * τ | 1 | $\tau \to \tau$

terms M ::=

```
# STLC
      x                          # variables
    | (M,M)                      # product types
    | π_i(M)
    | ()
    | λx . M                     # function types
    | M M
# primitives
    | tt                         # boolean values
    | ff
    | ∧                          # boolean operations
    | ∨
    | ¬
    | if M then M else M         # branching
```

types τ ::= bool | τ * τ | 1 | $\tau \to \tau$

terms M ::=

```
# STLC
      x                          # variables
   | (M,M)                       # product types
   | π_i(M)
   | ()
   | λx . M                      # function types
   | M M
# primitives
   | tt                          # boolean values
   | ff
   | ∧                           # boolean operations
   | ∨
   | ¬
   | if M then M else M          # branching
# effect operations
   | read : 1 → bool             # read from the cell
```

The semantic model

A model $(\mathrm{Fin}, \mathrm{R}, \boldsymbol{s})$

Idea:

1. Interpret programs as functions;
2. Parametrise by what's in the cell.

A model $(\mathrm{Fin}, \mathrm{R}, \boldsymbol{s})$

Idea:

1. Interpret programs as functions;
2. Parametrise by what's in the cell.

- Use the category Fin of finite sets;

A model $(\mathrm{Fin}, \mathrm{R}, s)$

Idea:

1. Interpret programs as functions;
2. Parametrise by what's in the cell.

- Use the category Fin of finite sets;
- Use the natural interpretation:

$$s[\![\mathsf{bool}]\!] := 2 := \{0, 1\}$$

$$s[\![\diamond \vdash \mathtt{tt} : \mathsf{bool}]\!] = \mathrm{const}_1$$

$$s[\![\diamond \vdash \mathtt{ff} : \mathsf{bool}]\!] = \mathrm{const}_0$$

$$s[\![\Gamma \vdash \neg M : \mathsf{bool}]\!] = \lambda\gamma\,.\,\lambda i\,.\,\neg\big(s[\![M]\!](\gamma)(i)\big)$$

$$s[\![\Gamma \vdash \mathtt{read}() : \mathsf{bool}]\!] = \lambda\gamma\,.\,\lambda i\,.\,i$$

A model $(\mathrm{Fin}, \mathrm{R}, s)$

Idea:

1. Interpret programs as functions;
2. Parametrise by what's in the cell.

- Use the category Fin of finite sets;
- Use the natural interpretation:

$$s[\![\mathsf{bool}]\!] := 2 := \{0, 1\}$$

$$s[\![\diamond \vdash \mathtt{tt} : \mathsf{bool}]\!] = \mathrm{const}_1$$

$$s[\![\diamond \vdash \mathtt{ff} : \mathsf{bool}]\!] = \mathrm{const}_0$$

$$s[\![\Gamma \vdash \neg M : \mathsf{bool}]\!] = \lambda\gamma\,.\,\lambda i\,.\,\neg\big(s[\![M]\!](\gamma)(i)\big)$$

$$s[\![\Gamma \vdash \mathtt{read}() : \mathsf{bool}]\!] = \lambda\gamma\,.\,\lambda i\,.\,i$$

- Use the reader monad: $\mathrm{R}\underline{X} := (2 \Rightarrow \underline{X})$:

$$s[\![\diamond \vdash M : \tau]\!] \in \mathrm{R}(s[\![\tau]\!])$$

$$s[\![\Gamma \vdash M : \tau]\!] : s[\![\Gamma]\!] \to \mathrm{R}s[\![\tau]\!]$$

$s[\![M]\!](i)$ = value M returns when i in the cell

$(\mathrm{Fin}, \mathrm{R}, s)$ **is not fully abstract** [Matache & Staton]

$(\mathrm{Fin}, \mathrm{R}, s)$ is not fully abstract [Matache & Staton]

M, M' : ((1 → bool) → bool) → bool

apply f : (1 → bool) → bool to λx . tt
then to λx . ff
then take the disjunction
M := λf . (f (λx . tt)) ∨ (f (λx . ff))

$(\mathrm{Fin}, \mathrm{R}, s)$ is not fully abstract [Matache & Staton]

M, M' : ((1 → bool) → bool) → bool

```
# apply f : (1 → bool) → bool to λx . tt
#  then to λx . ff
#  then take the disjunction
M  := λf . (f (λx . tt)) ∨ (f (λx . ff))

# apply f to read then to
#  the function negating the read value
#  then take the disjunction
M' := λf . (f read) ∨ (f (λx . ¬(read x)))
```

$(\mathrm{Fin}, \mathrm{R}, s)$ is not fully abstract [Matache & Staton]

M, M' : ((1 → bool) → bool) → bool

```
# apply f : (1 → bool) → bool to λx . tt
#  then to λx . ff
#  then take the disjunction
M  := λf . (f (λx . tt)) ∨ (f (λx . ff))

# apply f to read then to
#  the function negating the read value
#  then take the disjunction
M' := λf . (f read) ∨ (f (λx . ¬(read x)))
```

Intuitively, $M \simeq_{\mathrm{ctx}} M'$. But...

$(\mathrm{Fin}, \mathrm{R}, s)$ **is not fully abstract** [Matache & Staton]

Have:

$$s[\![M]\!], s[\![M']\!] \in \mathrm{R}\Big(\big((1 \Rightarrow \mathrm{R2}) \Rightarrow \mathrm{R2}\big) \Rightarrow \mathrm{R2}\Big)$$

$(\mathrm{Fin}, \mathrm{R}, s)$ is not fully abstract [Matache & Staton]

Have:

$$s[\![M]\!], s[\![M']\!] \in \mathrm{R}\big(((1 \Rightarrow \mathrm{R2}) \Rightarrow \mathrm{R2}) \Rightarrow \mathrm{R2}\big)$$

Take $\kappa : (1 \Rightarrow \mathrm{R2}) \to \mathrm{R2}$:

$$\kappa(g) := \begin{cases} \mathrm{const}_1 & \text{if } g(*) = \mathrm{const}_1 \\ \mathrm{const}_0 & \text{else} \end{cases}$$

Then

$$s[\![M]\!](i)(\kappa)(j) = 1 \neq 0 = s[\![M']\!](i)(\kappa)(j)$$

What goes wrong?

The model describes behaviours Λ_{ROS} cannot express

What goes wrong?

The model describes behaviours Λ_{ROS} cannot express

$$\kappa(g) := \begin{cases} \mathrm{const}_1 & \text{if } g(*) = \mathrm{const}_1 \\ \mathrm{const}_0 & \text{else} \end{cases}$$

κ knows how g behaves
both when the cell contains 0
and when it contains 1

State is read-only
— programs can't do this!

What goes wrong?

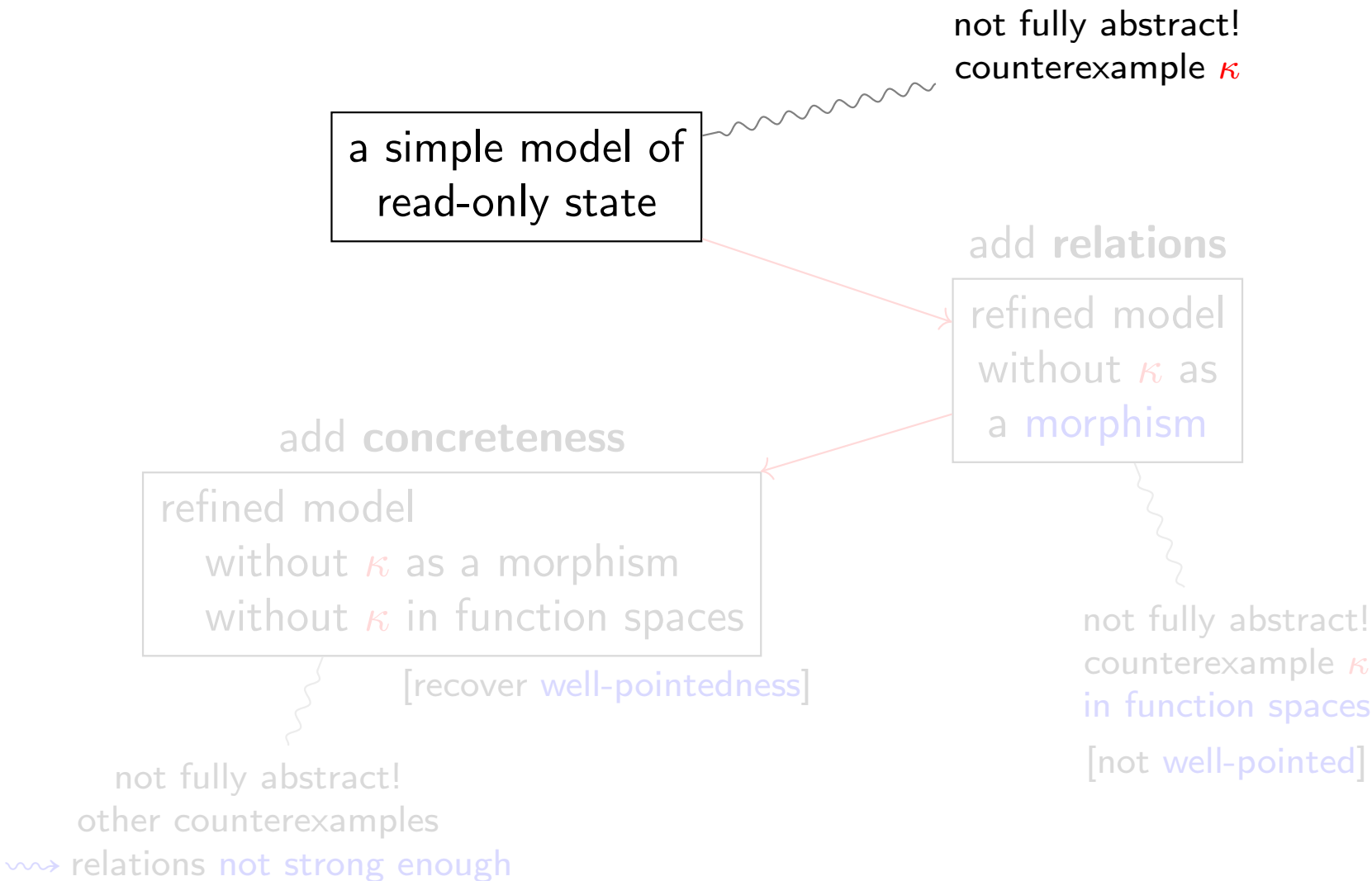
The model describes behaviours $\Lambda_{\mathbf{ROS}}$ cannot express

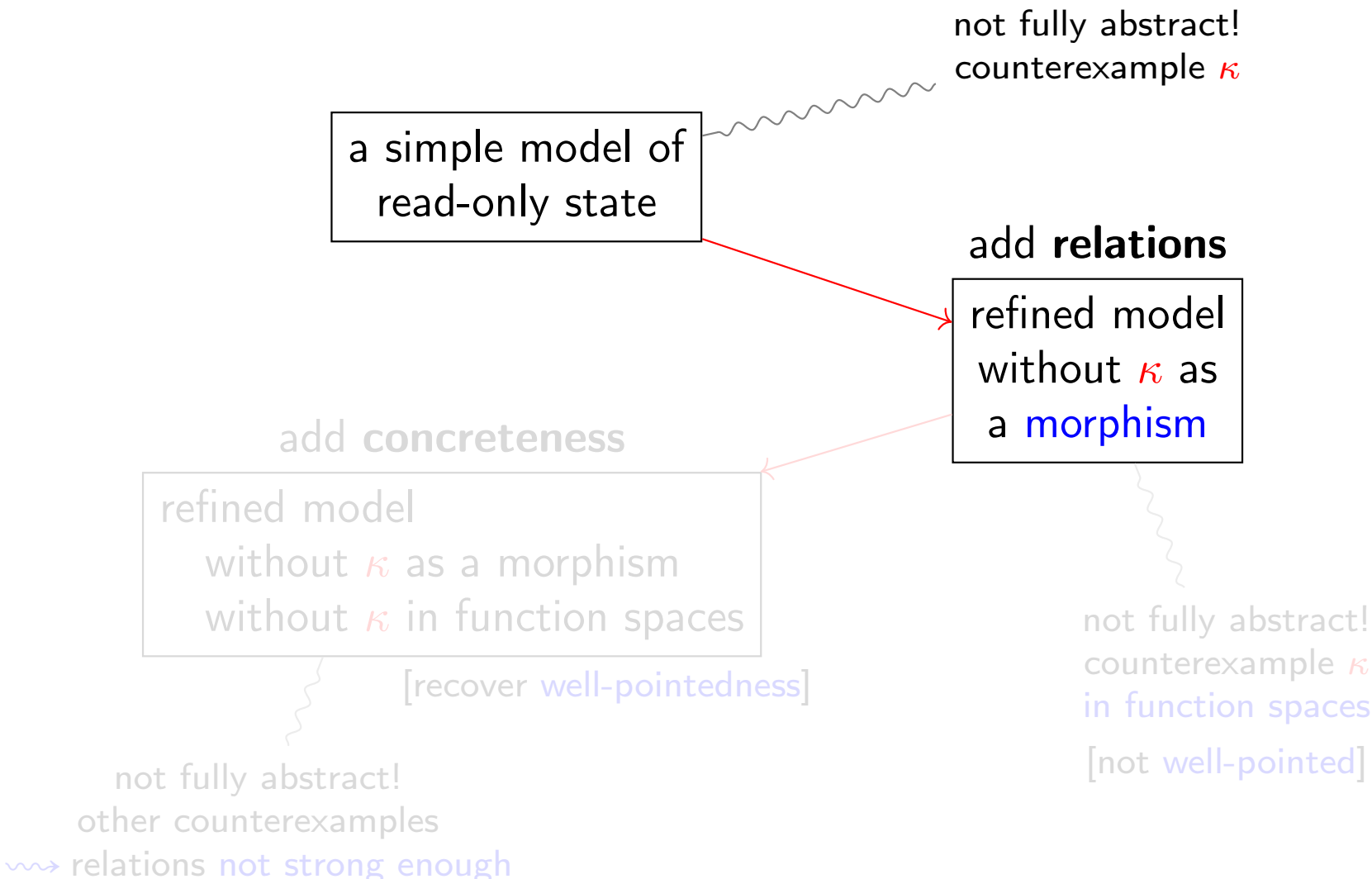
$$\kappa(g) := \begin{cases} \mathrm{const}_1 & \text{if } g(*) = \mathrm{const}_1 \\ \mathrm{const}_0 & \text{else} \end{cases}$$

κ knows how g behaves
both when the cell contains 0
and when it contains 1

State is read-only
— programs can't do this!

κ is a counterexample to contextual equivalence


not fully abstract!
counterexample κ
a simple model of
read-only state
add **relations**
refined model
without κ as
a morphism
add **concreteness**
refined model
without κ as a morphism
without κ in function spaces
[recover well-pointedness]
not fully abstract!
other counterexamples
relations not strong enough
not fully abstract!
counterexample κ
in function spaces
[not well-pointed]


not fully abstract!
counterexample κ
a simple model of
read-only state
add **relations**
refined model
without κ as
a morphism
add **concreteness**
refined model
without κ as a morphism
without κ in function spaces
[recover well-pointedness]
not fully abstract!
other counterexamples
relations not strong enough
not fully abstract!
counterexample κ
in function spaces
[not well-pointed]

A refined model $(\mathbb{L}, \hat{\mathrm{R}}, t)$

Idea:

- pair each set with relations R_0 and R_1
- restrict to functions preserving these relations

preserving $R_i \rightsquigarrow$ respecting behaviour when cell contains i

A refined model $(\mathbb{L}, \hat{\mathrm{R}}, t)$

Idea:

- pair each set with relations R_0 and R_1
- restrict to functions preserving these relations

preserving R_i $\rightsquigarrow$ respecting behaviour when cell contains i

The cartesian closed category $\mathbb{L}$

objects: triples $(\underline{X}, R_0, R_1)$ $\rightsquigarrow$ $\underline{X} \in \mathsf{Fin}$, $R_i \subseteq \underline{X}^2$

A refined model $(\mathbb{L}, \hat{\mathrm{R}}, t)$

Idea:

- pair each set with relations R_0 and R_1
- restrict to functions preserving these relations

preserving R_i $\rightsquigarrow$ respecting behaviour when cell contains i

The cartesian closed category $\mathbb{L}$

objects: triples $(\underline{X}, R_0, R_1)$ $\rightsquigarrow$ $\underline{X} \in \mathsf{Fin}$, $R_i \subseteq \underline{X}^2$

maps $(\underline{X}, R_0, R_1) \to (\underline{Y}, S_0, S_1)$:

maps $f : \underline{X} \to \underline{Y}$ preserving the relations

$\rightsquigarrow$ $(x, x') \in R_i \implies (f\,x, f\,x') \in S_i$ for $i = 1, 2$

The cartesian closed category $\mathbb{L}$

objects: triples $(\underline{X}, R_0, R_1)$

maps $(\underline{X}, R_0, R_1) \to (\underline{Y}, S_0, S_1)$:

maps $f : \underline{X} \to \underline{Y}$ preserving the relations

The cartesian closed category $\mathbb{L}$

objects: triples $(\underline{X}, R_0, R_1)$

maps $(\underline{X}, R_0, R_1) \to (\underline{Y}, S_0, S_1)$:

maps $f : \underline{X} \to \underline{Y}$ preserving the relations

The monad $\hat{\mathrm{R}}$ $\rightsquigarrow$ defined by $\top\top$-lifting

$$\hat{\mathrm{R}}(\underline{Y}, S_0, S_1) = \big(2 \Rightarrow \underline{Y}, \hat{\mathrm{R}}(S_0), \hat{\mathrm{R}}(S_1)\big)$$

$(h, h') \in \hat{\mathrm{R}}(S_i)$
$\iff (h\,i, h'\,i) \in S_i$

The cartesian closed category $\mathbb{L}$

objects: triples $(\underline{X}, R_0, R_1)$

maps $(\underline{X}, R_0, R_1) \to (\underline{Y}, S_0, S_1)$:
maps $f : \underline{X} \to \underline{Y}$ preserving the relations

The monad $\hat{\mathrm{R}}$ ⇝ defined by $\top\top$-lifting

$$\hat{\mathrm{R}}(\underline{Y}, S_0, S_1) = \big(2 \Rightarrow \underline{Y}, \hat{\mathrm{R}}(S_0), \hat{\mathrm{R}}(S_1)\big)$$

$(h, h') \in \hat{\mathrm{R}}(S_i)$
$\iff (h\, i, h'\, i) \in S_i$

The interpretation t

$t(\mathsf{bool}) = \big(2, \{(0,0),(1,1)\}, \{(0,0),(1,1)\}\big)$
$t(\mathrm{read}) = s[\![\mathrm{read}]\!]$ ⇝ already preserves the relations
...

One step closer to full abstraction?

κ **is not a morphism in** $\mathbb{L}$**!**

One step closer to full abstraction?

κ **is not a morphism in** $\mathbb{L}$**!**

$\Downarrow$

we've removed a
counterexample
to contextual equivalence

One step closer to full abstraction?

κ is not a morphism in $\mathbb{L}$!

$\Downarrow$

we've removed a
counterexample
to contextual equivalence

$(\mathbb{L}, \hat{\mathrm{R}}, t)$ — κ removed

$\downarrow U$ forgetful

$(\mathrm{Fin}, \mathrm{R}, s)$ — κ lives here

One step closer to full abstraction?

All the structure is preserved:

$(\mathbb{L}, \hat{\mathrm{R}}, t)$ κ removed

$\downarrow U$ forgetful

$(\mathsf{Fin}, \mathrm{R}, s)$ κ lives here

One step closer to full abstraction?

All the structure is preserved

$$\begin{array}{ll} (\mathbb{L}, \hat{\mathrm{R}}, t) & \kappa \text{ removed} \\ \Big\downarrow U \text{ forgetful} & \\ (\mathsf{Fin}, \mathrm{R}, s) & \kappa \text{ lives here} \end{array}$$

$\Longrightarrow$ . . . so the semantics is preserved:

$$U(t[\![K]\!]) = s[\![K]\!] \text{ for all } K$$

One step closer to full abstraction?

All the structure is preserved

$$(\mathbb{L}, \hat{\mathrm{R}}, t) \quad \kappa \text{ removed}$$
$$\Big\downarrow U \text{ forgetful}$$
$$(\mathsf{Fin}, \mathrm{R}, s) \quad \kappa \text{ lives here}$$

. . . so the semantics is preserved:

$$\Longrightarrow \qquad U(t[\![K]\!]) = s[\![K]\!] \text{ for all } K$$

This can **never** be enough

One step closer to full abstraction?

All the structure is preserved

$$(\mathbb{L}, \hat{\mathrm{R}}, t) \quad \kappa \text{ removed}$$
$$\Big\downarrow U \text{ forgetful}$$
$$(\mathsf{Fin}, \mathrm{R}, s) \quad \kappa \text{ lives here}$$

$\Longrightarrow$ . . . so the semantics is preserved:

$$U(t[\![K]\!]) = s[\![K]\!] \text{ for all } K$$

This can **never** be enough

1. Suppose $(\mathbb{L}, \hat{\mathrm{R}}, t)$ is fully abstract

One step closer to full abstraction?

All the structure is preserved ... so the semantics is preserved:

$(\mathbb{L}, \hat{\mathrm{R}}, t)$ — κ removed

$\Big\downarrow U$ forgetful $\qquad \Longrightarrow \qquad U(t[\![K]\!]) = s[\![K]\!]$ for all K

$(\mathsf{Fin}, \mathrm{R}, s)$ — κ lives here

This can **never** be enough

1. Suppose $(\mathbb{L}, \hat{\mathrm{R}}, t)$ is fully abstract
2. ... so $t[\![M]\!] = t[\![M']\!]$

One step closer to full abstraction?

All the structure is preserved

$$(\mathbb{L}, \hat{\mathrm{R}}, t) \quad \kappa \text{ removed}$$
$$\Big\downarrow U \text{ forgetful}$$
$$(\mathsf{Fin}, \mathrm{R}, s) \quad \kappa \text{ lives here}$$

. . . so the semantics is preserved:

$$\Longrightarrow \qquad U(t[\![K]\!]) = s[\![K]\!] \text{ for all } K$$

This can **never** be enough

---

1. Suppose $(\mathbb{L}, \hat{\mathrm{R}}, t)$ is fully abstract
2. . . . so $t[\![M]\!] = t[\![M']\!]$
3. Then $s[\![M]\!] = U(t[\![M]\!]) = U(t[\![M']\!]) = s[\![M']\!]$

One step closer to full abstraction?

All the structure is preserved

$$\begin{array}{l}(\mathbb{L}, \hat{\mathrm{R}}, t) \qquad \kappa \text{ removed} \\ \quad \Big\downarrow U \text{ forgetful} \\ (\mathsf{Fin}, \mathrm{R}, s) \qquad \kappa \text{ lives here}\end{array}$$

$\Longrightarrow$ . . . so the semantics is preserved:

$$U(t[\![K]\!]) = s[\![K]\!] \text{ for all } K$$

This can **never** be enough $\rightsquigarrow$ relations are never sufficient

---

1. Suppose $(\mathbb{L}, \hat{\mathrm{R}}, t)$ is fully abstract
2. . . . so $t[\![M]\!] = t[\![M']\!]$
3. Then $s[\![M]\!] = U(t[\![M]\!]) = U(t[\![M']\!]) = s[\![M']\!]$

Diagnosing the problem

$$\kappa \in s[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] = U(t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!])$$

Diagnosing the problem

$$\kappa \in s[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] = U(t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!])$$

$$\Downarrow$$

$$t[\![M]\!], t[\![M']\!] : t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] \to \hat{\mathrm{R}}(t[\![\mathsf{bool}]\!])$$

can still disagree on κ!

Diagnosing the problem

$$\kappa \in s[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] = U(t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!])$$

$$\Downarrow$$

$$t[\![M]\!], t[\![M']\!] : t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] \to \hat{\mathrm{R}}(t[\![\mathsf{bool}]\!])$$

can still disagree on κ!

κ is not in the hom-sets in $\mathbb{L}$

Diagnosing the problem

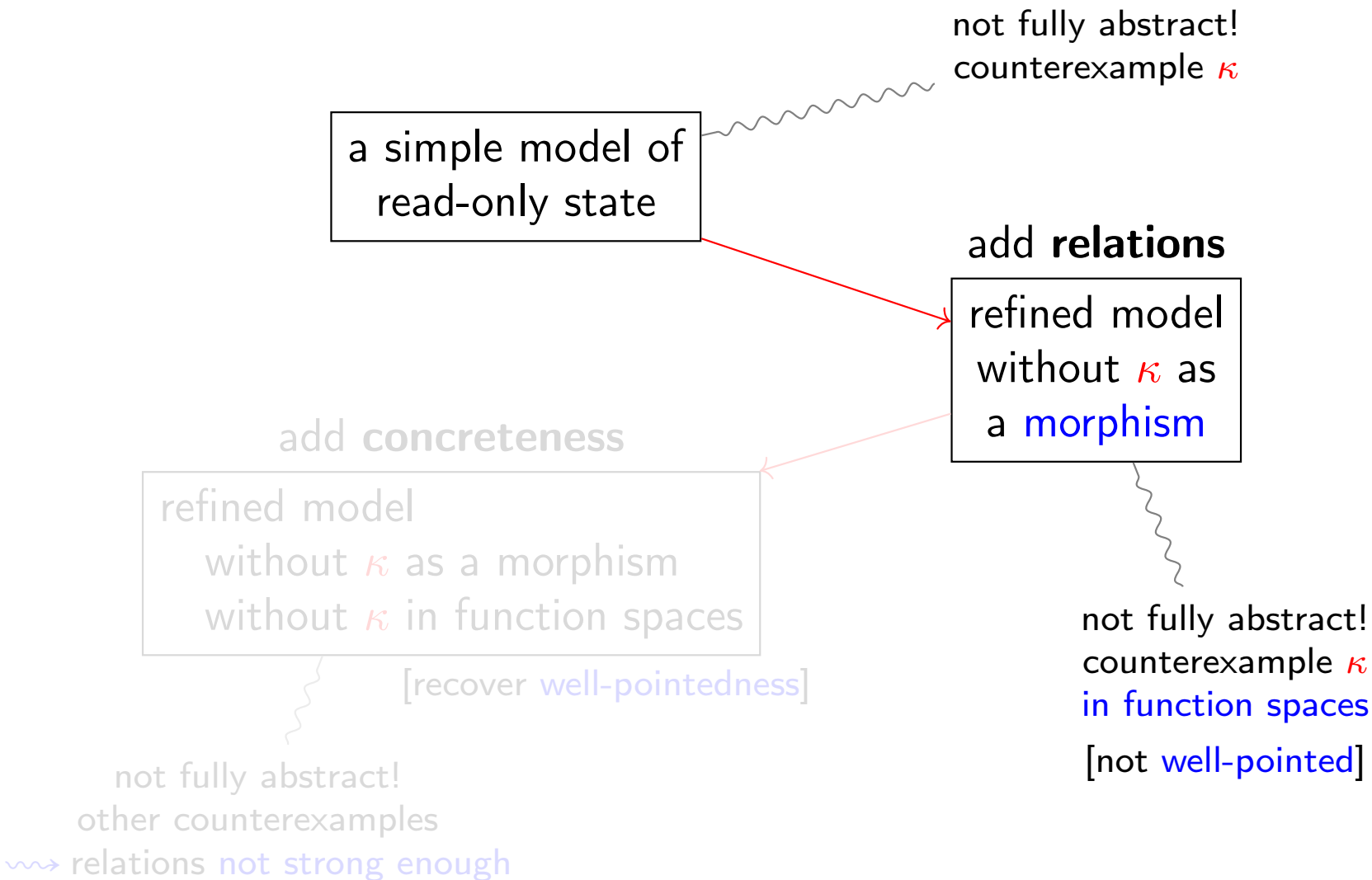
$$\kappa \in s[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] = U(t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!])$$

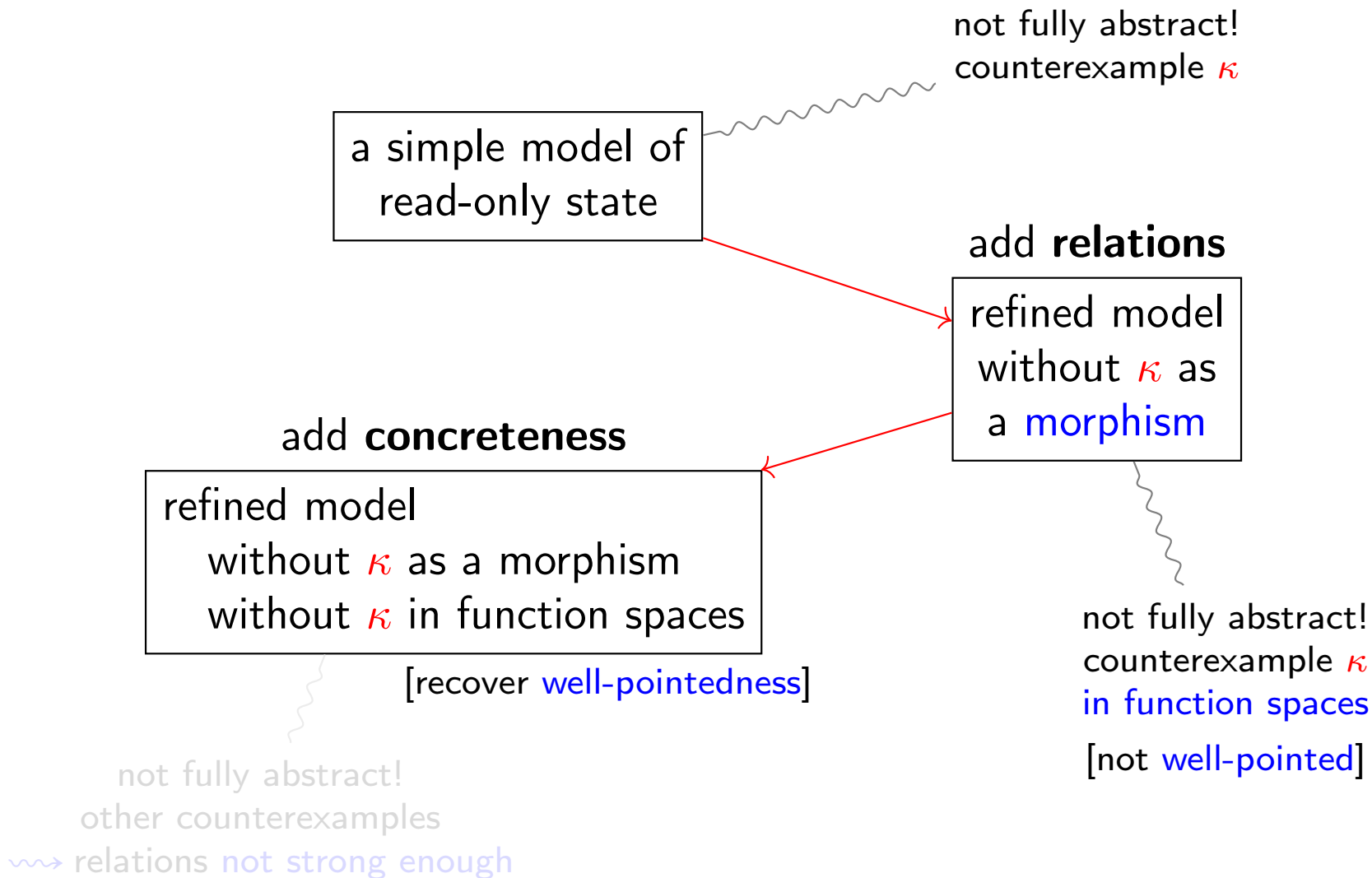
$$\Downarrow$$

$$t[\![M]\!], t[\![M']\!] : t[\![(1 \to \mathsf{bool}) \to \mathsf{bool}]\!] \to \hat{\mathrm{R}}(t[\![\mathsf{bool}]\!])$$

can still disagree on κ!

κ is not in the hom-sets in $\mathbb{L}$ **but** κ is in the function spaces in $\mathbb{L}$


not fully abstract!
counterexample κ
a simple model of
read-only state
add **relations**
refined model
without κ as
a morphism
add **concreteness**
refined model
without κ as a morphism
without κ in function spaces
[recover well-pointedness]
not fully abstract!
other counterexamples
relations not strong enough
not fully abstract!
counterexample κ
in function spaces
[not well-pointed]


not fully abstract!
counterexample κ
a simple model of
read-only state
add **relations**
refined model
without κ as
a morphism
add **concreteness**
refined model
without κ as a morphism
without κ in function spaces
[recover well-pointedness]
not fully abstract!
counterexample κ
in function spaces
[not well-pointed]
not fully abstract!
other counterexamples
$\rightsquigarrow$ relations not strong enough

Removing κ from the function space

Removing κ from the function space

Observation

We have

$$\kappa \in U(t[\![(1 \to \mathrm{bool}) \to \mathrm{bool}]\!])$$

but there is no global element in $\mathbb{L}$

$$\gamma : 1 \to t[\![(1 \to \mathrm{bool}) \to \mathrm{bool}]\!]$$

such that $\gamma(*) = \kappa$.

$\Rightarrow \mathbb{L}$ is not well-pointed

[$f = g$ iff $f \circ \gamma = g \circ \gamma$
for all global elements γ]

Removing κ from the function space

Observation $\rightsquigarrow$ κ appears as a shadow!

We have

$$\kappa \in U(t[\![(1 \to \mathrm{bool}) \to \mathrm{bool}]\!])$$

but there is no global element in $\mathbb{L}$

$$\gamma : 1 \to t[\![(1 \to \mathrm{bool}) \to \mathrm{bool}]\!]$$

such that $\gamma(*) = \kappa$.

$\Rightarrow \mathbb{L}$ is not well-pointed

$[f = g$ iff $f \circ \gamma = g \circ \gamma$
for all global elements $\gamma]$

Solution: restrict to things named by a global element

Solution: restrict to things named by a global element

---

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

Solution: restrict to things named by a global element

---

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

Solution: restrict to things named by a global element

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

Explicitly: for every $x \in \underline{X}$, the pair $(x, x) \in R_0$ and $(x, x) \in R_1$.

Solution: restrict to things named by a global element

---

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

Solution: restrict to things named by a global element

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

$$\mathbb{C} \hookrightarrow j \longrightarrow \mathbb{L} \circlearrowright \hat{\mathrm{R}}$$

Solution: restrict to things named by a global element

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

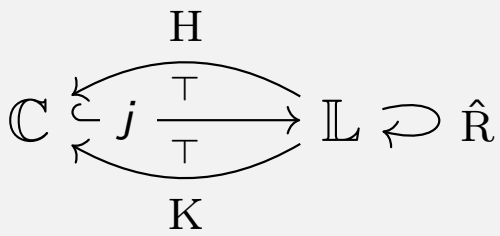
$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

restrict to subset
of concrete elements



union relations
with diagonal

Solution: restrict to things named by a global element

1. for $(\underline{X}, R_0, R_1) \in \mathbb{L}$, $x \in \underline{X}$ is concrete if

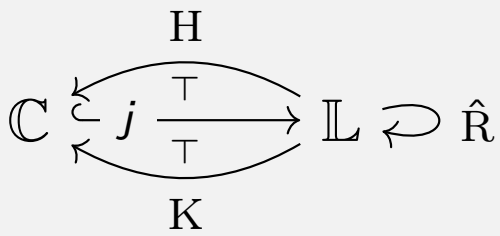
$$\ulcorner x \urcorner : * \mapsto x : 1 \to (\underline{X}, R_0, R_1)$$

   is a map in $\mathbb{L}$;

2. $(\underline{X}, R_0, R_1) \in \mathbb{L}$ is concrete if every $x \in \underline{X}$ is concrete.

$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

restrict to subset
of concrete elements



union relations
with diagonal

$\Rightarrow$

$\mathbb{C}$ a CCC
with monad $\mathrm{H}\hat{\mathrm{R}}j$

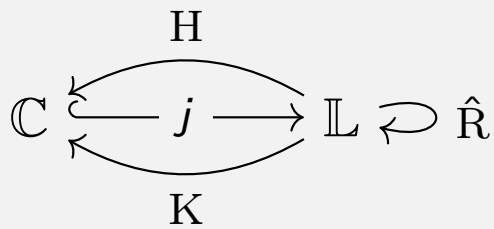
$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

objects: $(\underline{X}, R_0, R_1)$ such that $x \in \underline{X} \implies (x, x) \in R_i$

maps: set-maps preserving the relations

---

restrict to subset
of concrete elements



$\Rightarrow$

$\mathbb{C}$ a CCC
with monad $\mathrm{H}\hat{\mathrm{R}}j$

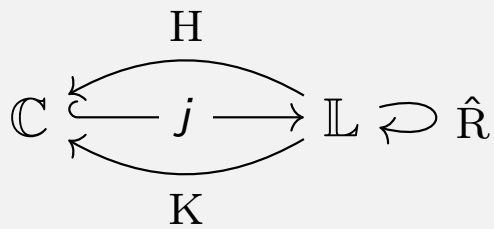
$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

objects: $(\underline{X}, R_0, R_1)$ such that $x \in \underline{X} \implies (x, x) \in R_i$

maps: set-maps preserving the relations

---

restrict to subset
of concrete elements



$\Rightarrow$

$\mathbb{C}$ a CCC
with monad $\mathrm{H}\hat{\mathrm{R}}j$

Exponentials internalise preserving relations

$$(X \Rightarrow_{\mathbb{C}} Y) = \mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big)$$

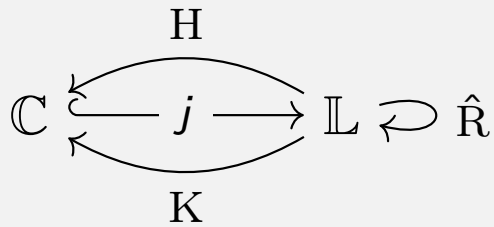
$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

objects: $(\underline{X}, R_0, R_1)$ such that $x \in \underline{X} \implies (x, x) \in R_i$

maps: set-maps preserving the relations

---

restrict to subset
of concrete elements



$\Rightarrow$

$\mathbb{C}$ a CCC
with monad $\mathrm{H}\hat{\mathrm{R}}j$

Exponentials internalise preserving relations

$$(X \Rightarrow_{\mathbb{C}} Y) = \mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big)$$

$f \in U\mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big)$

iff $f \in U\big(X \Rightarrow_{\mathbb{L}} Y\big)$

and $\exists$ a global element in $\mathbb{L}$ corresponding to f

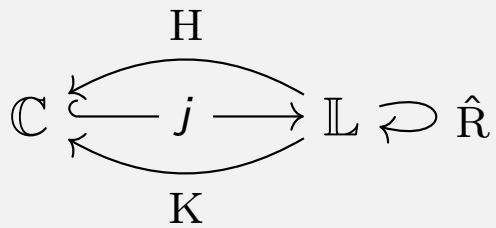
$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

objects: $(\underline{X}, R_0, R_1)$ such that $x \in \underline{X} \implies (x, x) \in R_i$

maps: set-maps preserving the relations

---

restrict to subset of concrete elements



$\Rightarrow$

$\mathbb{C}$ a CCC with monad $\mathrm{H}\hat{\mathrm{R}}j$

Exponentials internalise preserving relations

$$(X \Rightarrow_{\mathbb{C}} Y) = \mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big)$$

$$U\big(X \Rightarrow_{\mathbb{C}} Y\big) = U\mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big) \cong \mathbb{L}(X, Y)$$

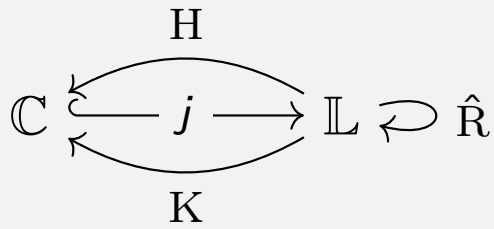
$\mathbb{C}$ = full subcategory of $\mathbb{L}$ of concrete objects

objects: $(\underline{X}, R_0, R_1)$ such that $x \in \underline{X} \implies (x, x) \in R_i$

maps: set-maps preserving the relations

---

restrict to subset
of concrete elements



$\Rightarrow$

$\mathbb{C}$ a CCC
with monad $\mathrm{H}\hat{\mathrm{R}}j$

Exponentials internalise preserving relations

$$(X \Rightarrow_{\mathbb{C}} Y) = \mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big)$$

$$U\big(X \Rightarrow_{\mathbb{C}} Y\big) = U\mathrm{H}\big(X \Rightarrow_{\mathbb{L}} Y\big) \cong \mathbb{L}(X, Y)$$

$\rightsquigarrow$ concreteness removes κ from the function space

Success!

Our new semantic model:

Success!

Our new semantic model:

1. CCC $\mathbb{C}$

Success!

Our new semantic model:

1. CCC $\mathbb{C}$
2. monad $\mathrm{H}\hat{\mathrm{R}}j$

Success!

Our new semantic model:

1. CCC $\mathbb{C}$
2. monad $\mathrm{H}\hat{\mathrm{R}}j$
3. interpretation t by restriction

Success!

Our new semantic model:

1. CCC $\mathbb{C}$
2. monad $\mathrm{H}\hat{\mathrm{R}}j$
3. interpretation t by restriction

Aim: remove bad morphism κ

1. from hom-sets $\rightsquigarrow$ [logical] relations
2. from function spaces $\rightsquigarrow$ concreteness

Success?

Our new semantic model:

1. CCC $\mathbb{C}$
2. monad $\mathrm{H}\hat{\mathrm{R}}j$
3. interpretation t by restriction

Aim: remove bad morphism κ

1. from hom-sets $\rightsquigarrow$ [logical] relations
2. from function spaces $\rightsquigarrow$ concreteness

But $(\mathbb{C}, \mathrm{H}\hat{\mathrm{R}}j, t)$ is not fully abstract $\rightsquigarrow$ need stronger relations!

We know how to remove a single bad morphism...

We know how to remove a single bad morphism...

Question:

how can we soup up $\mathbb{C}$ to remove every bad morphism?

We know how to remove a single bad morphism...

Question:

how can we soup up $\mathbb{C}$ to remove every bad morphism?

Want to identify a class of relations such that

f preserves those relations $\implies$ f is not a bad morphism

We know how to remove a single bad morphism...

Question:

how can we soup up $\mathbb{C}$ to remove every bad morphism?

Want to identify a class of relations such that

f preserves those relations $\implies$ f is not a bad morphism

A sufficient condition:

$f = s[\![K]\!]$ for some K

We know how to remove a single bad morphism...

Question:

how can we soup up $\mathbb{C}$ to remove every bad morphism?

Want to identify a class of relations such that

f preserves those relations $\implies$ f is not a bad morphism

A sufficient condition:

$f = s[\![K]\!]$ for some K $\rightsquigarrow$ f is definable

The route from here

The route from here

1. the model we've just seen, abstractly

The route from here

1. the model we've just seen, abstractly
2. restricting to definable morphisms

The route from here

1. the model we've just seen, abstractly
2. restricting to definable morphisms
3. the abstract OHR construction $\rightsquigarrow$ follows pattern just seen

The route from here

1. the model we've just seen, abstractly
2. restricting to definable morphisms
3. the abstract OHR construction $\rightsquigarrow$ follows pattern just seen
4. getting full abstraction

Constructing $(\mathbb{C}, \mathrm{H}\hat{\mathrm{R}}j, t)$: a recipe

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

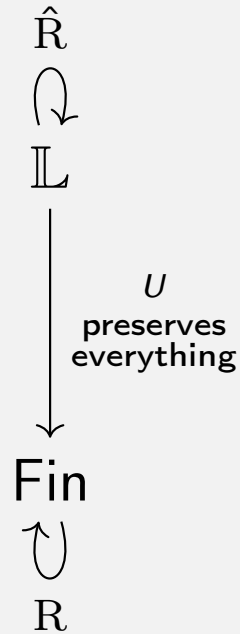
---

$$\begin{array}{c} \mathrm{Fin} \\ \circlearrowleft \\ \mathrm{R} \end{array}$$

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

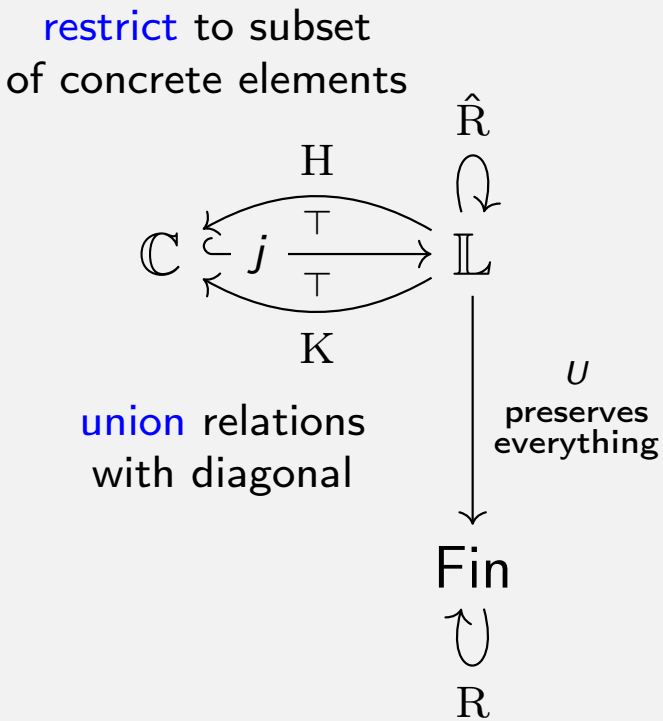
---



A recipe for removing bad morphisms κ

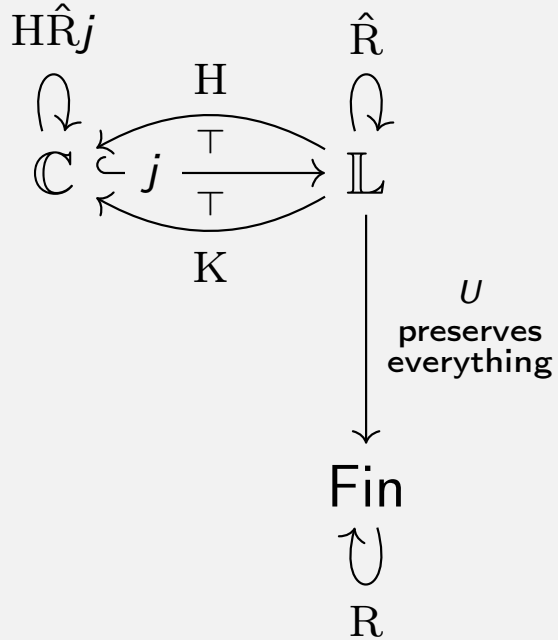
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

---



A recipe for removing bad morphisms κ

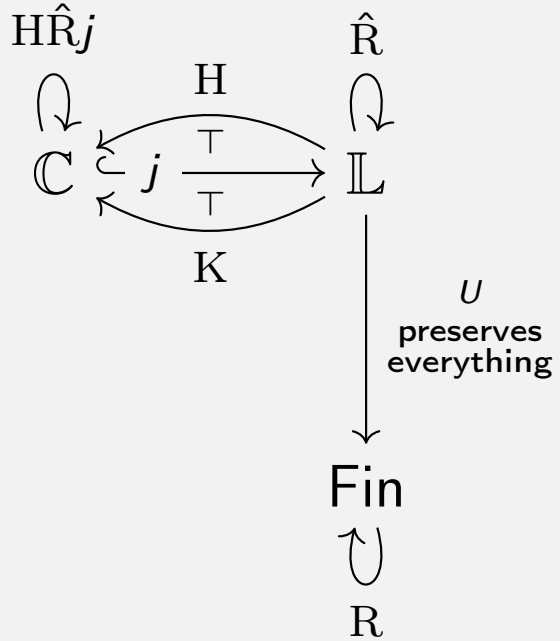
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

---



(X, A) such that $A \subseteq X$;
maps preserve the predicate

$$\mathrm{Sub}(\mathsf{Fin}) \times \mathrm{Sub}(\mathsf{Fin})$$

$$\downarrow \; \mathrm{cod} \times \mathrm{cod}$$

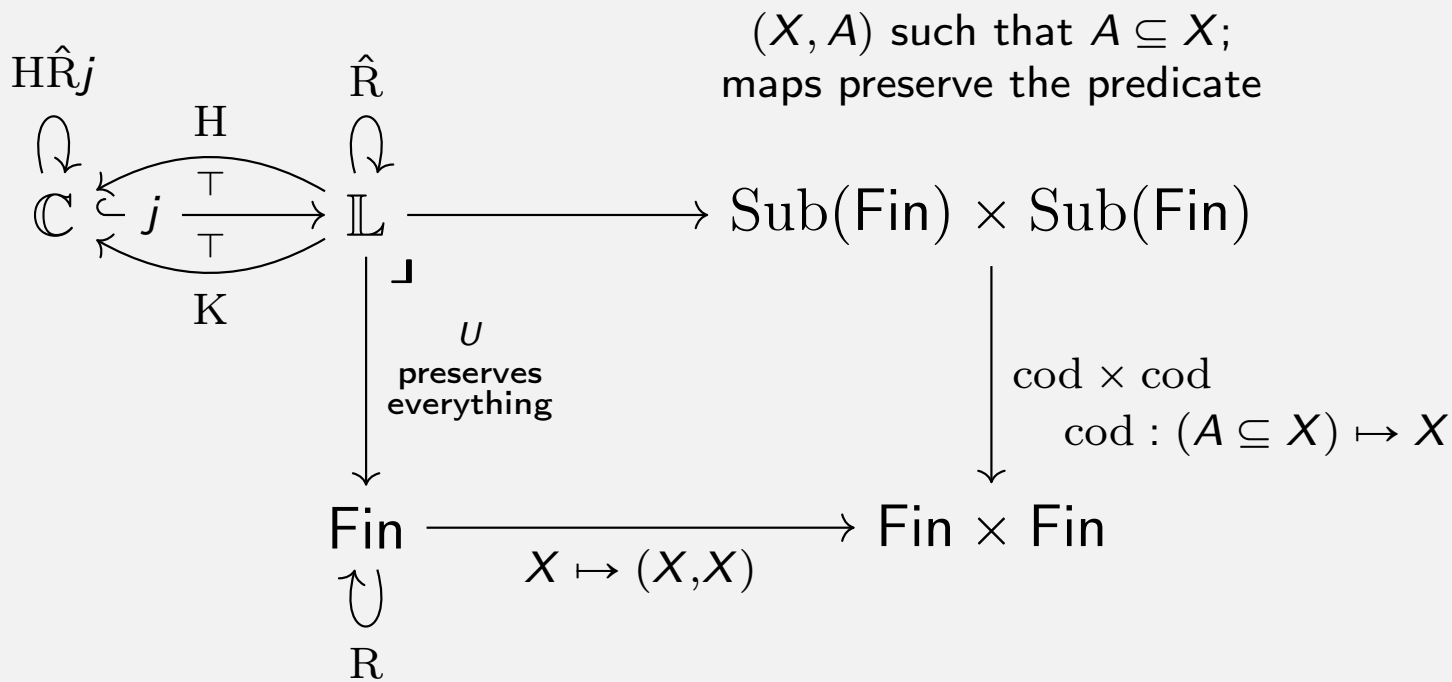
$$\mathrm{cod} : (A \subseteq X) \mapsto X$$

$$\mathsf{Fin} \times \mathsf{Fin}$$

A recipe for removing bad morphisms κ

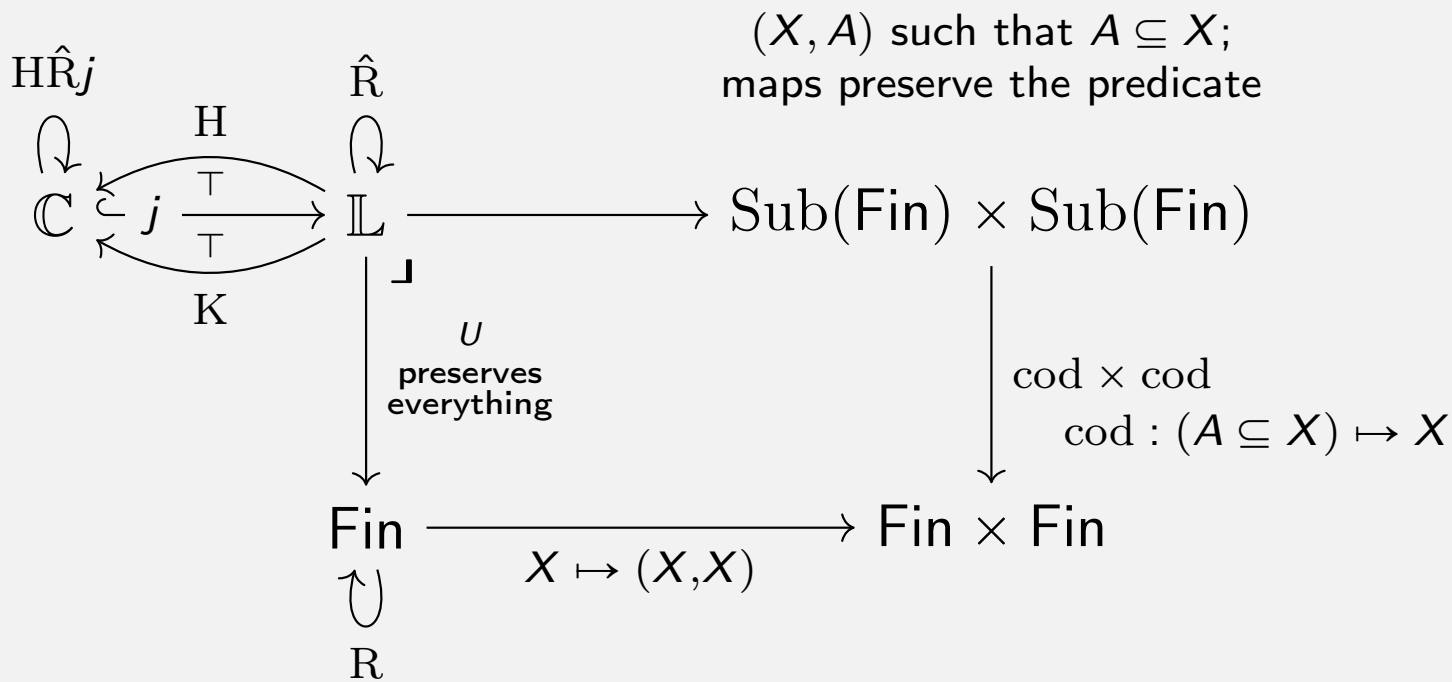
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

---



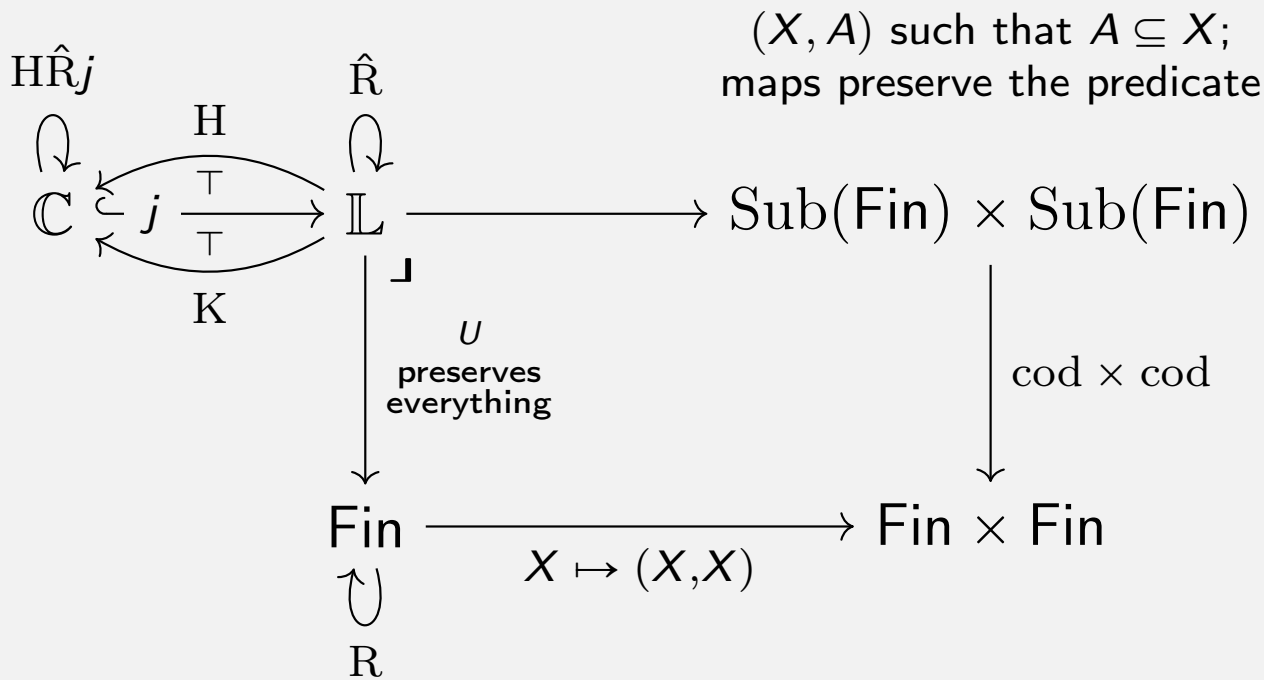
A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

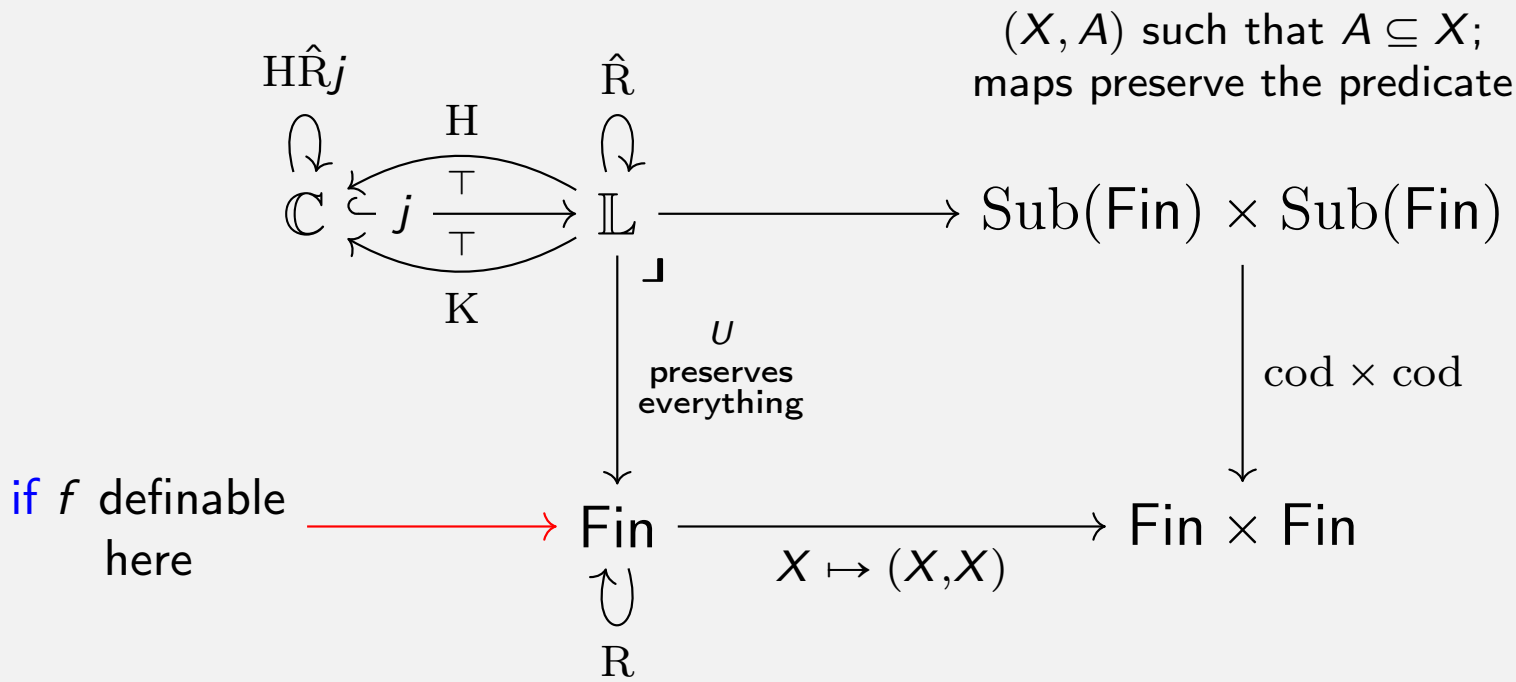


NB: two relations $\rightsquigarrow$ two categories on RHS

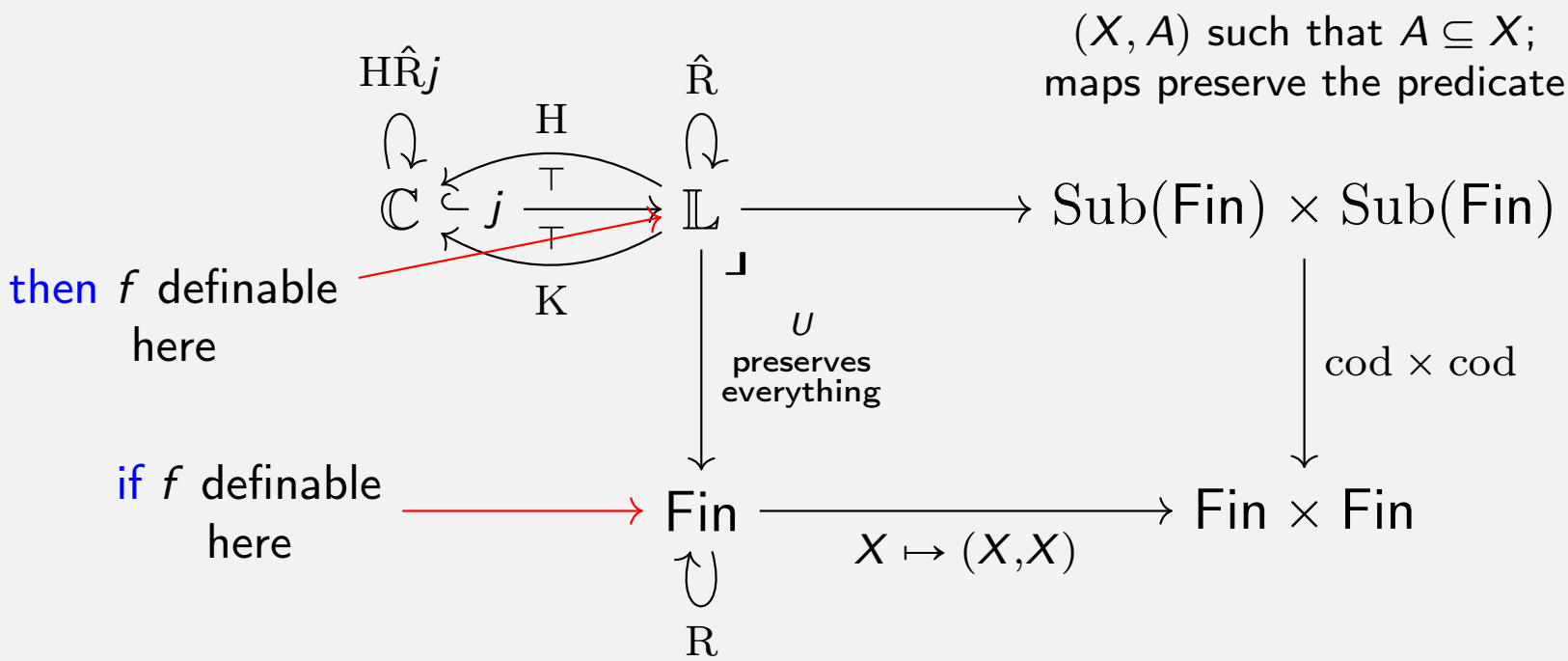
A recipe for removing bad morphisms κ



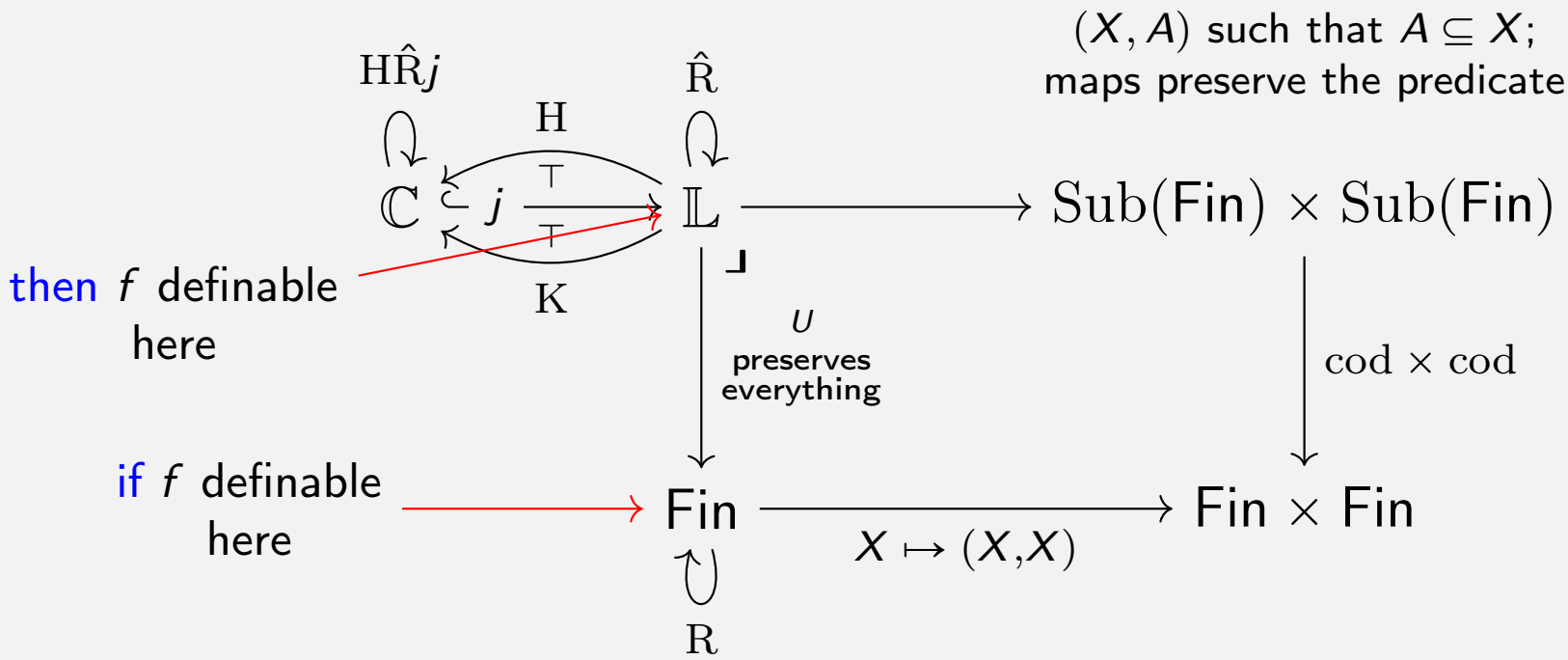
A recipe for removing bad morphisms κ



A recipe for removing bad morphisms κ



A recipe for removing bad morphisms κ



Every definable $f : s[\![\Gamma]\!] \to T(s[\![\sigma]\!])$ lifts to $f : t[\![\Gamma]\!] \to \hat{T}(t[\![\sigma]\!])$

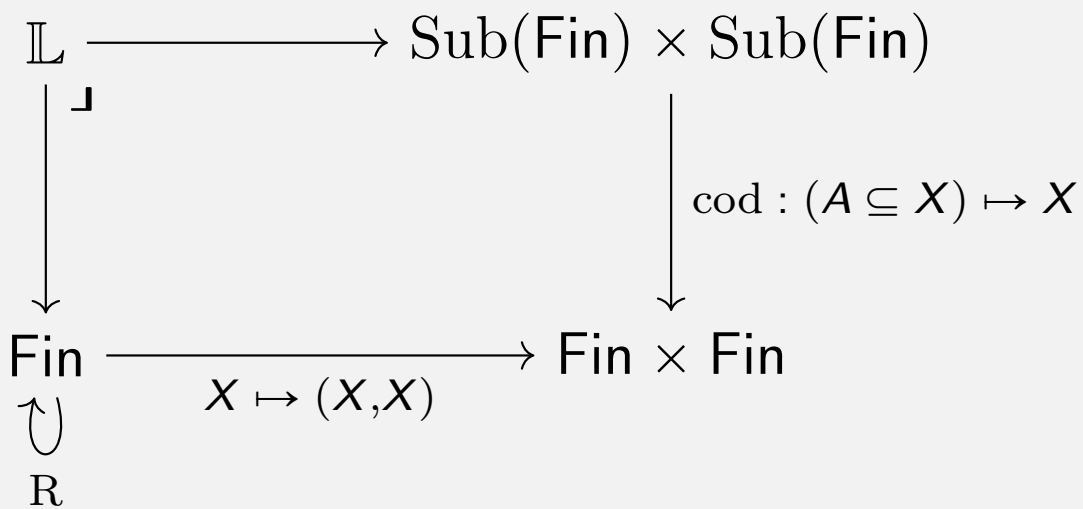
How can we change this construction so only definable maps lift?

$\mathbb{K}_{\mathcal{M},F}$, abstractly

Idea:

replace relations on sets . . .

(X, A) such that $A \subseteq X$;
maps preserve the predicate

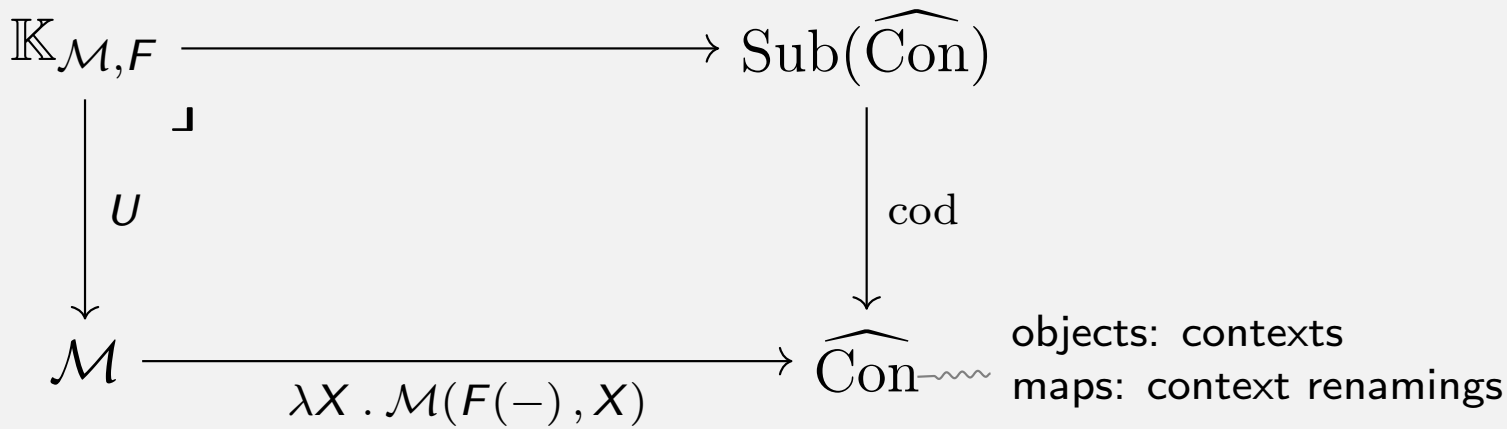


$\mathbb{K}_{\mathcal{M},F}$, abstractly

Idea:

replace relations on sets . . .

with families of relations indexed by contexts



The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

category of contexts and renamings

"semantic interpretation"

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

"semantic interpretation"

Con: category of contexts and renamings

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

"semantic interpretation"

Con: category of contexts and renamings

object $\in \mathcal{M}$

predicate = Kripke relation

$\{R(\Gamma) \subseteq \mathcal{M}(F\Gamma, W)\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

"semantic interpretation"

Con: category of contexts and renamings

object $\in \mathcal{M}$

predicate = Kripke relation

$\{R(\Gamma) \subseteq \mathcal{M}(F\Gamma, W)\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

over Set with $|F\Gamma| = n \rightsquigarrow R$ an n-ary relation

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

"semantic interpretation"

Con: category of contexts and renamings

object $\in \mathcal{M}$

predicate = Kripke relation

$\{R(\Gamma) \subseteq \mathcal{M}(F\Gamma, W)\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

$f : W \to W'$

preserves the predicate

$h \in R(\Gamma) \Rightarrow (f \circ h) \in R'(\Gamma)$

over Set with $|F\Gamma| = n \rightsquigarrow R$ an n-ary relation

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

predicate = Kripke relation

$\{R(\Gamma) \subseteq \mathcal{M}(F\Gamma, W)\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

object $\in \mathcal{M}$

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

Fact: $\mathbb{K}_{\mathcal{M},F}$ is a CCC

Notation: $(W, R) \Rightarrow (X, S) := (W \Rightarrow X, R \supset S)$

The category $\mathbb{K}_{\mathcal{M},F}$ of Kripke relations [Jung & Tiuryn, Alimohamed]

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

predicate = Kripke relation

$\{R(\Gamma) \subseteq \mathcal{M}(F\Gamma, W)\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

object $\in \mathcal{M}$

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

Fact: $\mathbb{K}_{\mathcal{M},F}$ is a CCC

Notation: $(W, R) \Rightarrow (X, S) := (W \Rightarrow X, R \supset S)$

$(f_1, \ldots, f_n) \in (R \supset S)(\Gamma)$ iff, for any $\rho : \Gamma \to \Delta$,

$(x_1, \ldots, x_m) \in R(\Delta) \subseteq W^{F\Gamma} \Rightarrow (f_{\rho(1)}x_1, \ldots, f_{\rho(m)}x_m) \in S(\Delta) \subseteq X^{F\Delta}$

Kripke relations for STLC definability [Jung & Tiuryn, Alimohamed, . . .]

Kripke relations for STLC definability [Jung & Tiuryn, Alimohamed, ...]

$$\mathrm{DEF}_\sigma(\Gamma) := \left\{ s[\![\Gamma \vdash M : \sigma]\!] \;\middle|\; M \text{ is derivable} \right\}$$

Kripke relations for STLC definability [Jung & Tiuryn, Alimohamed, . . .]

$$\mathrm{DEF}_\sigma(\Gamma) := \left\{ s[\![\Gamma \vdash M : \sigma]\!] \;\middle|\; M \text{ is derivable} \right\}$$

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

Kripke relations for STLC definability [Jung & Tiuryn, Alimohamed, . . .]

$$\mathrm{DEF}_\sigma(\Gamma) := \left\{ s[\![\Gamma \vdash M : \sigma]\!] \;\middle|\; M \text{ is derivable} \right\}$$

data:
- CCC $\mathcal{M}$
- $F : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

$s[\![\sigma]\!] \in \mathcal{M}$

definability predicate

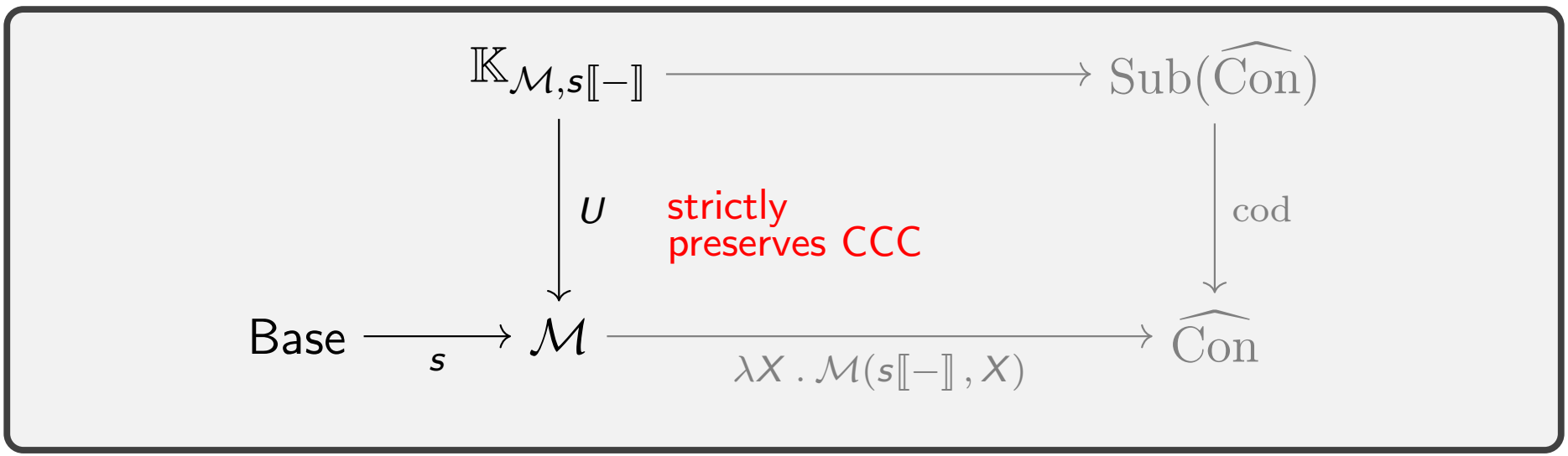
$\{\mathrm{DEF}_\sigma(\Gamma) \subseteq \mathcal{M}(s[\![\Gamma]\!], s[\![\sigma]\!])\}_{\Gamma \in \mathrm{Con}}$

compatible with renaming

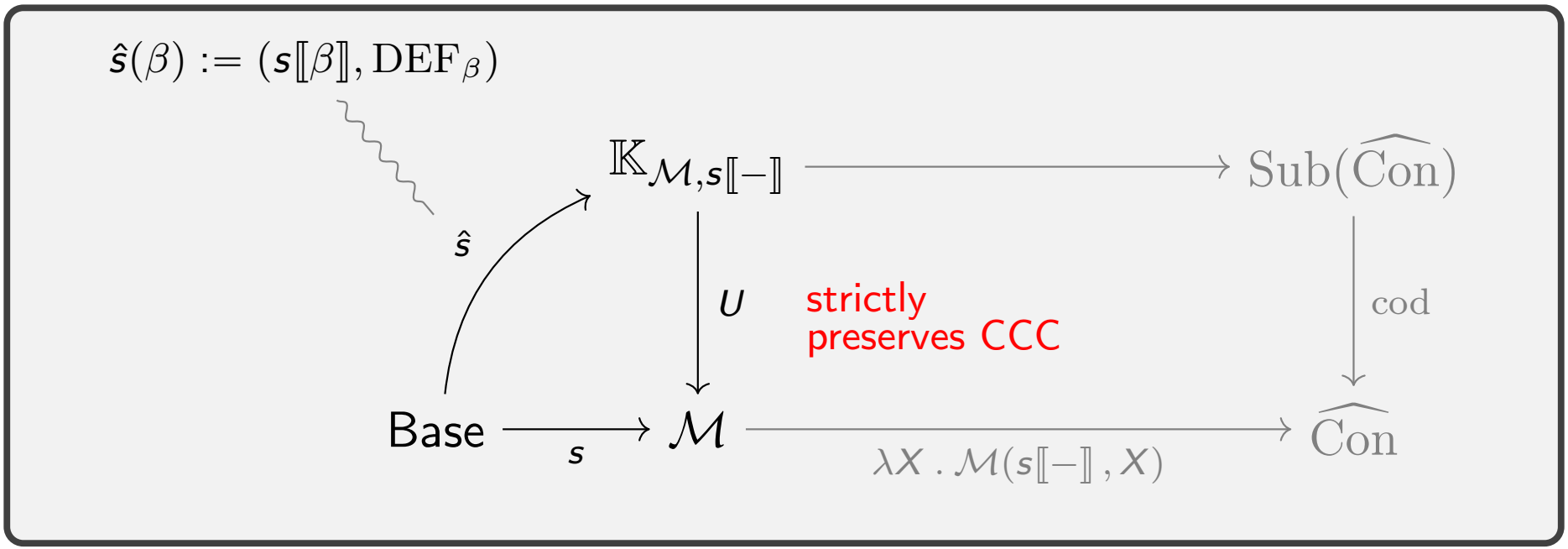
objects: (W, R)

maps: $(W, R) \xrightarrow{f} (W', R')$

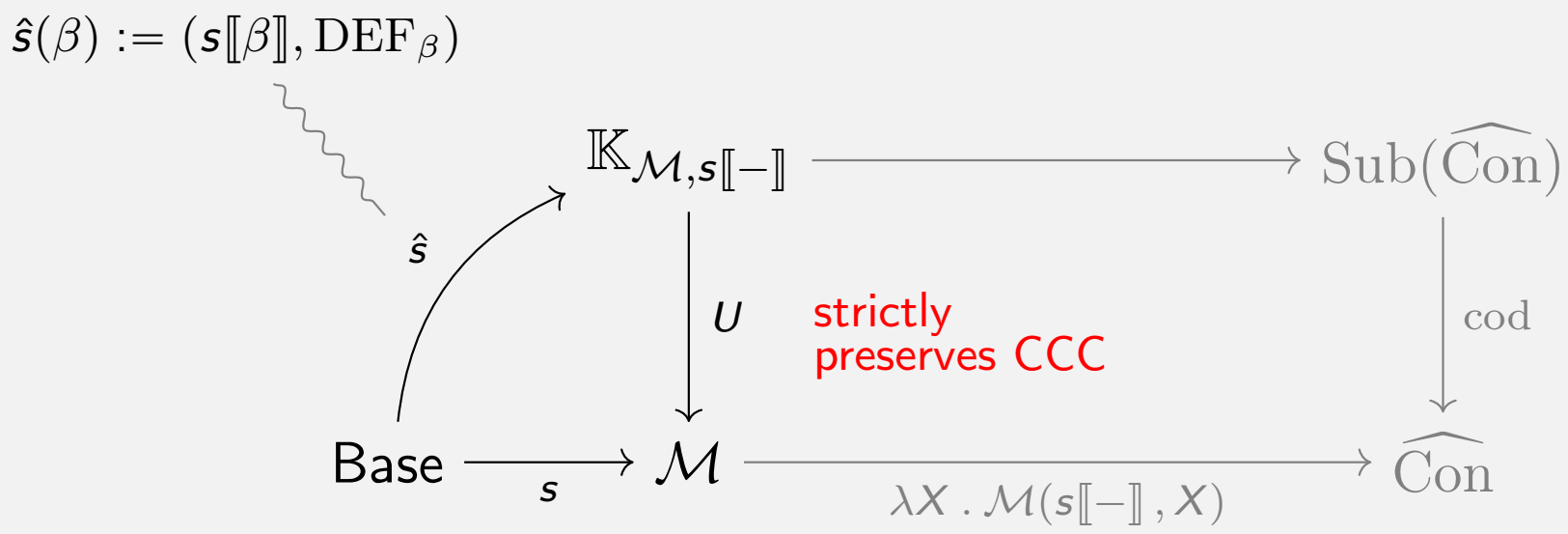
Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]



Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]



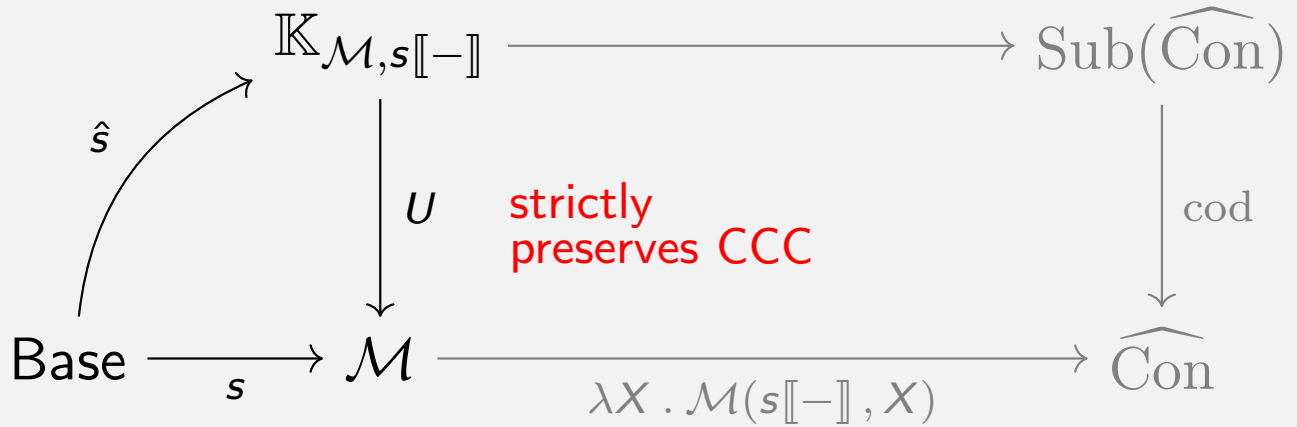
Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]



$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

i.e. $(\mathrm{DEF}_\sigma \supset \mathrm{DEF}_\tau) = \mathrm{DEF}_{\sigma \Rightarrow \tau}$ *etc.*

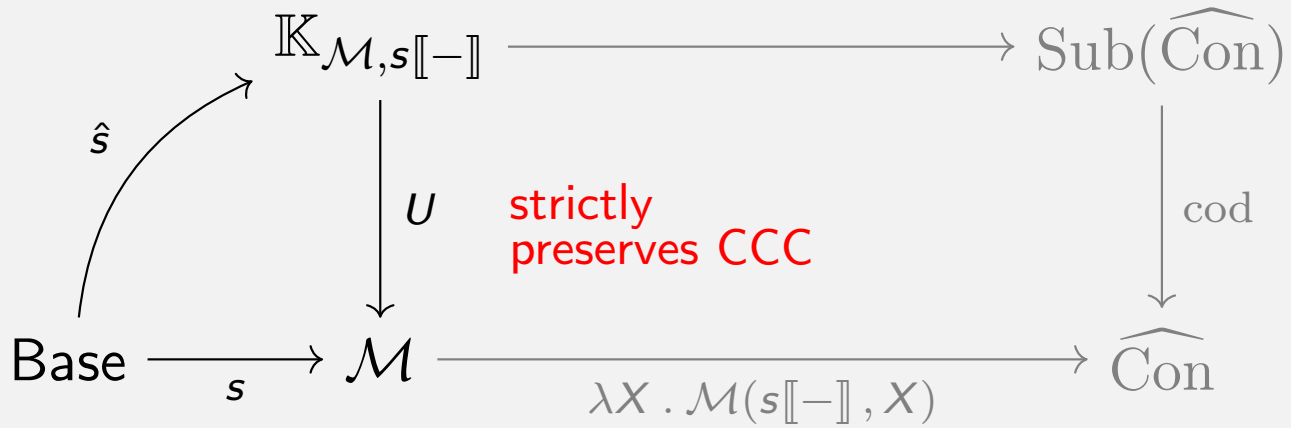
Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]



$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

$f : s[\![\Gamma]\!] \to s[\![\sigma]\!]$ is definable $\iff f : \hat{s}[\![\Gamma]\!] \to \hat{s}[\![\sigma]\!]$

Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]

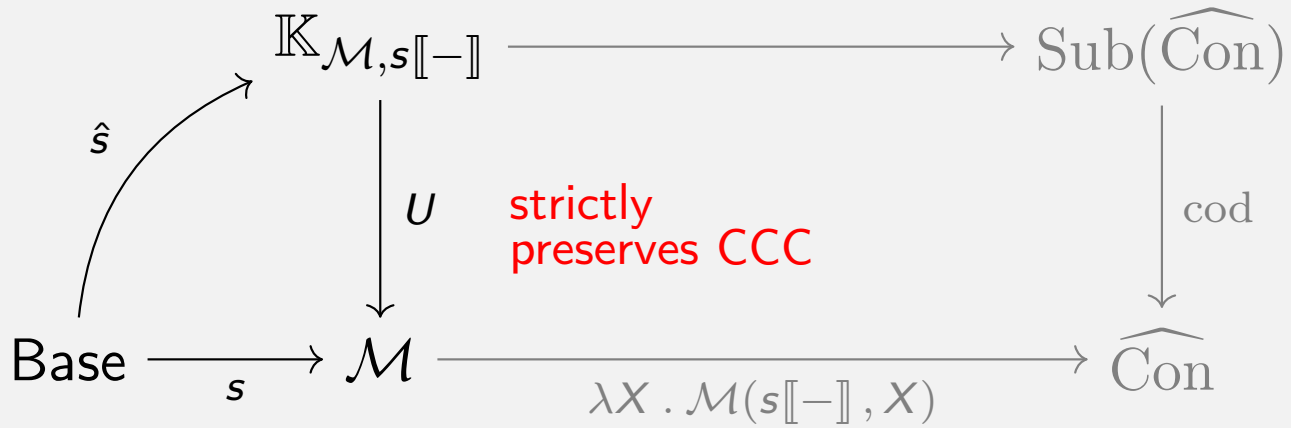


$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

$f : s[\![\Gamma]\!] \to s[\![\sigma]\!]$ is definable $\iff f : \hat{s}[\![\Gamma]\!] \to \hat{s}[\![\sigma]\!]$

$$\hat{s}[\![\Gamma]\!] \xrightarrow{\quad f \quad} \hat{s}[\![\sigma]\!]$$

Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]

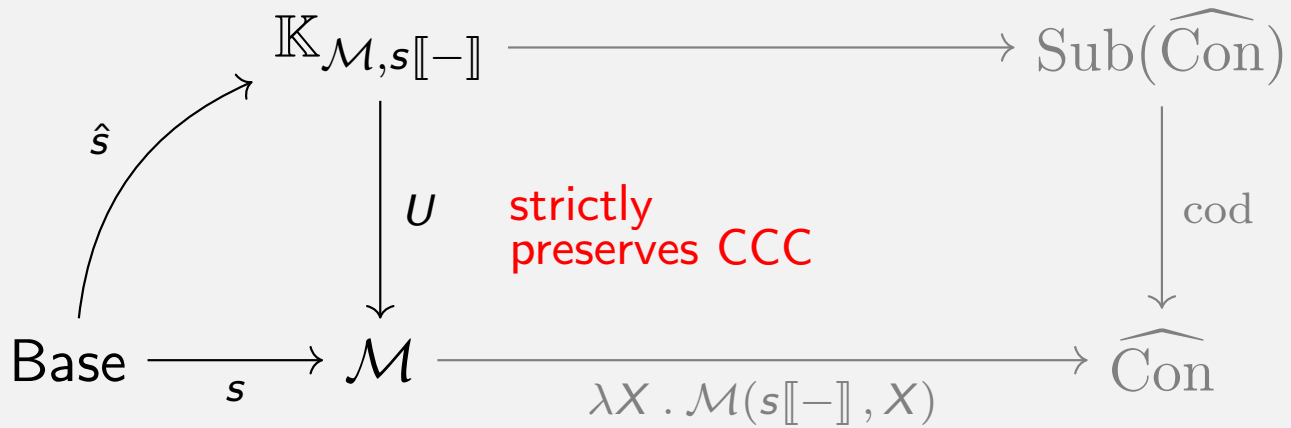


$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

$f : s[\![\Gamma]\!] \to s[\![\sigma]\!]$ is definable $\iff f : \hat{s}[\![\Gamma]\!] \to \hat{s}[\![\sigma]\!]$

$$(s[\![\Gamma]\!], \mathrm{DEF}_\Gamma) \xrightarrow{\ f\ } (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$$

Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]



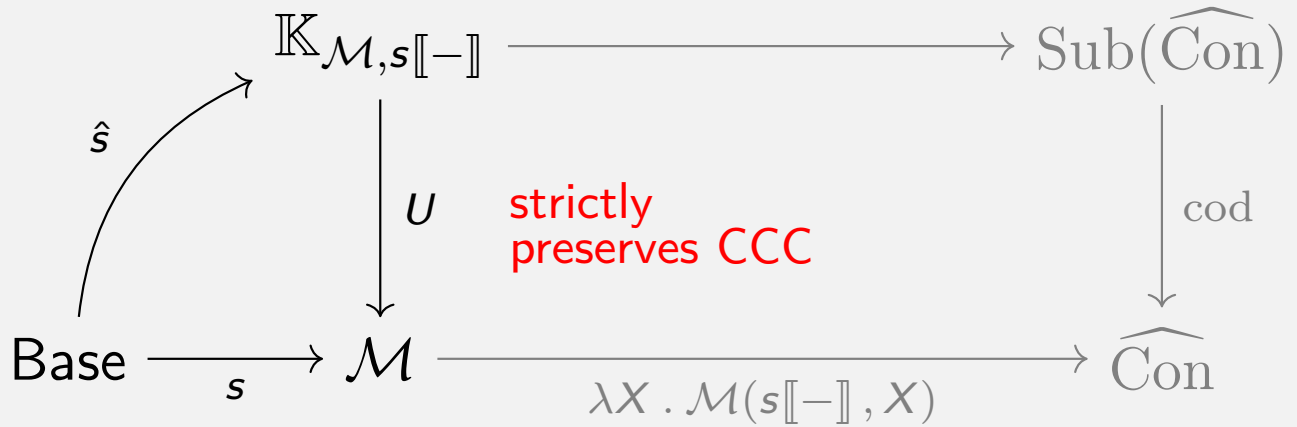
$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

$f : s[\![\Gamma]\!] \to s[\![\sigma]\!]$ is definable $\iff f : \hat{s}[\![\Gamma]\!] \to \hat{s}[\![\sigma]\!]$

$$(s[\![\Gamma]\!], \mathrm{DEF}_\Gamma) \xrightarrow{\ f\ } (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$$

$$\underset{\text{is definable}}{\mathrm{id}_{[\![\Gamma]\!]}} \longmapsto f \circ \mathrm{id}_{[\![\Gamma]\!]} \in \mathrm{DEF}_\sigma(\Gamma)$$

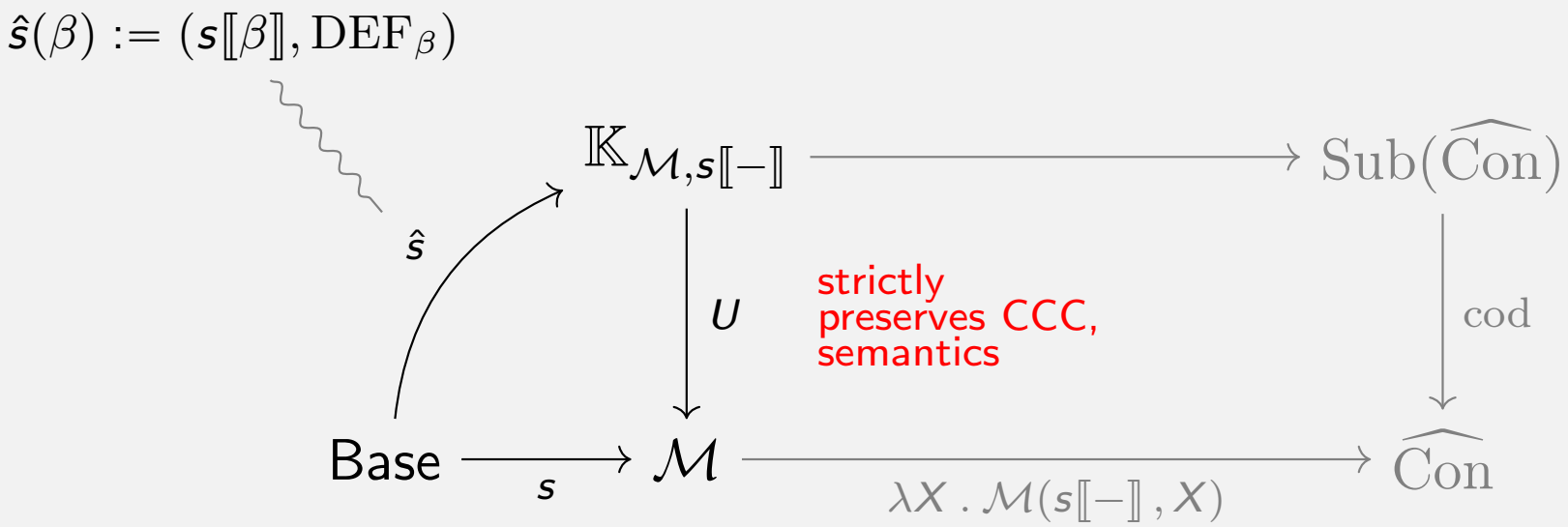
Characterising STLC-definable maps [Jung & Tiuryn, Alimohamed]

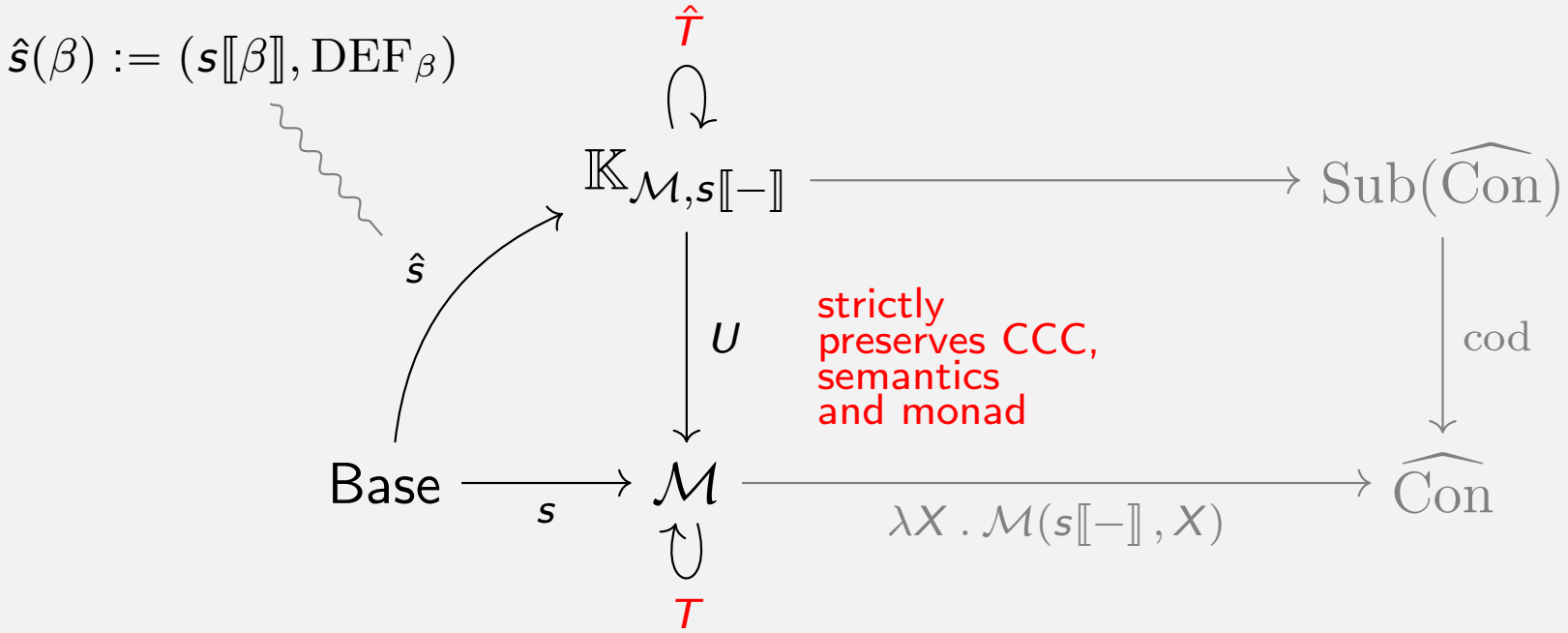


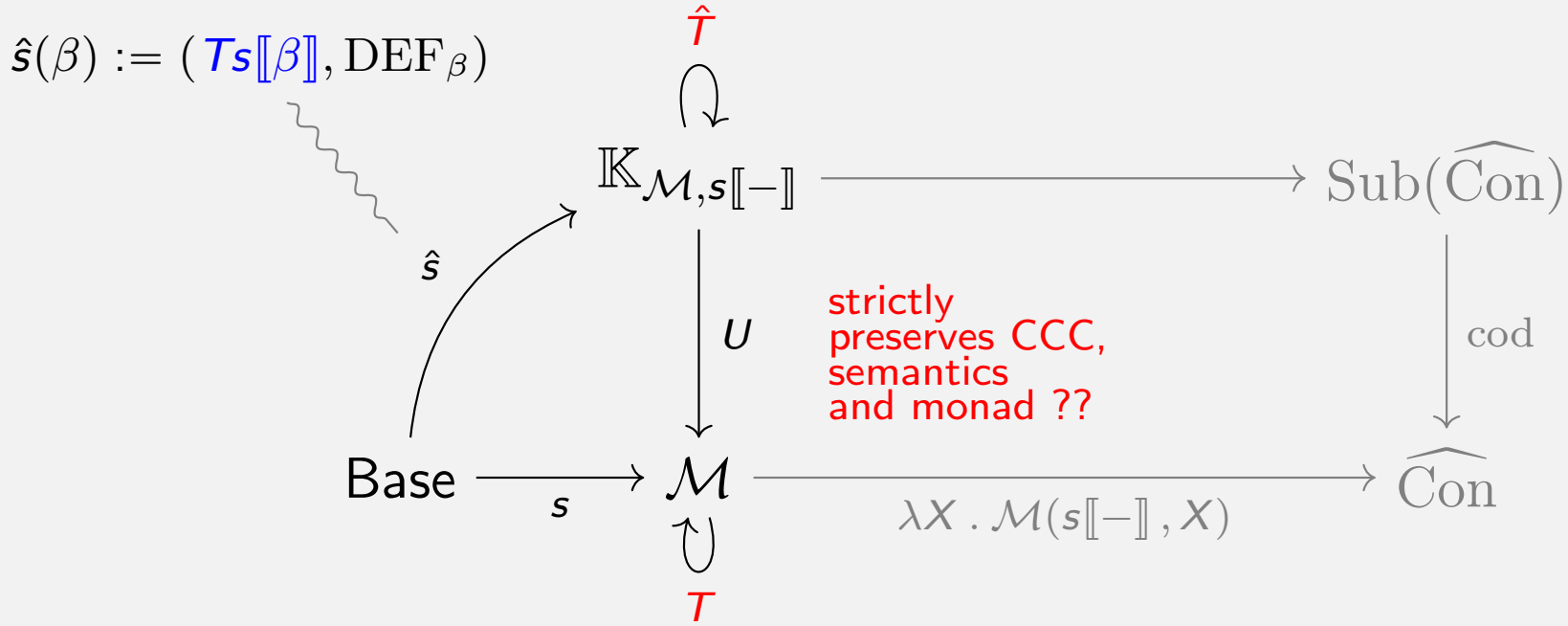
$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}_\sigma)$ for every STLC-type σ.

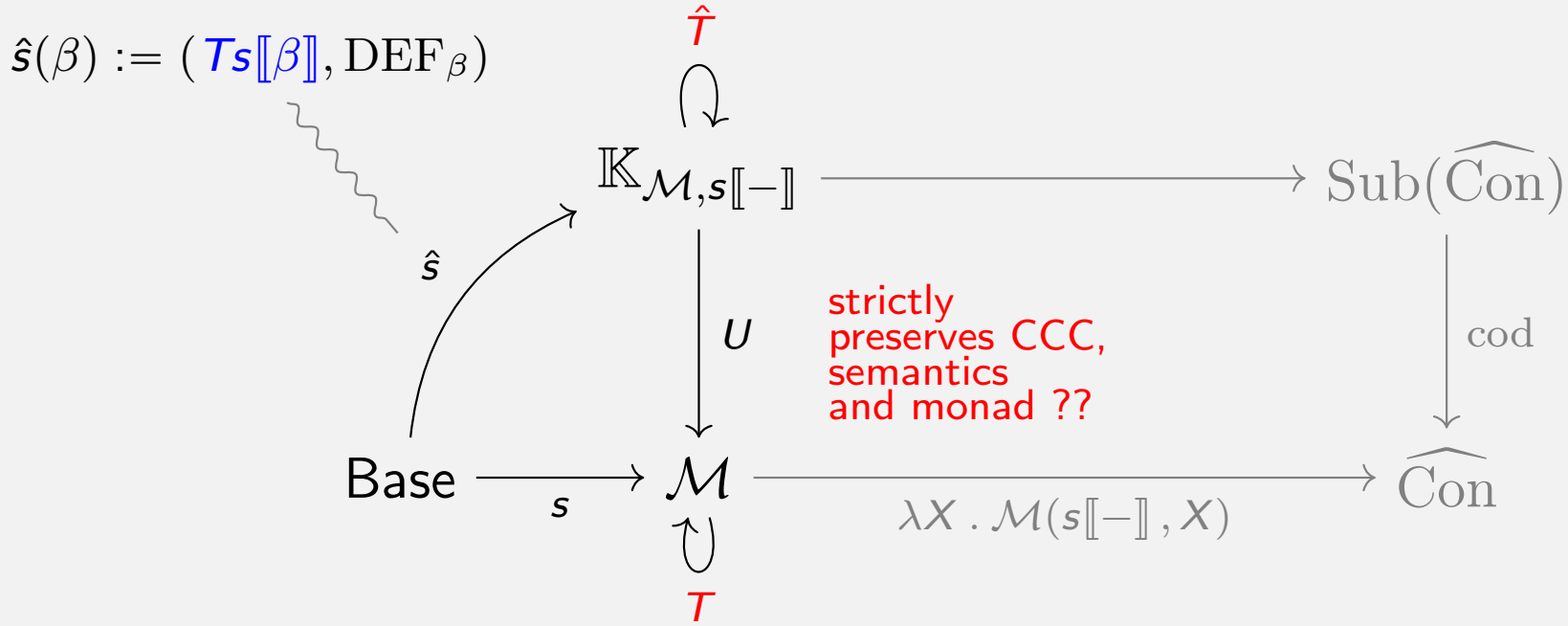
the definable maps are exactly those
that lift to $(\mathbb{K}_{\mathcal{M},s[\![-]\!]}, \hat{s})$

What about the monad?


$\hat{s}(\beta) := (s[\![\beta]\!], \mathrm{DEF}_\beta)$
$\hat{s}$
$\mathbb{K}_{\mathcal{M},s[\![-]\!]}$
$\widehat{\mathrm{Sub}(\mathrm{Con})}$
U
strictly
preserves CCC,
semantics
cod
Base
s
$\mathcal{M}$
$\lambda X \,.\, \mathcal{M}(s[\![-]\!], X)$
$\widehat{\mathrm{Con}}$


$\hat{s}(\beta) := (s[\![\beta]\!], \mathrm{DEF}_\beta)$
$\hat{T}$
$\mathbb{K}_{\mathcal{M},s[\![-]\!]}$
$\widehat{\mathrm{Sub}(\mathrm{Con})}$
$\hat{s}$
U
strictly
preserves CCC,
semantics
and monad
cod
Base
s
$\mathcal{M}$
$\widehat{\mathrm{Con}}$
$\lambda X\,.\,\mathcal{M}(s[\![-]\!], X)$
T


$\hat{s}(\beta) := (Ts[\![\beta]\!], \mathrm{DEF}_\beta)$
$\hat{T}$
$\mathbb{K}_{\mathcal{M},s[\![-]\!]}$
$\widehat{\mathrm{Sub}(\mathrm{Con})}$
$\hat{s}$
U
strictly
preserves CCC,
semantics
and monad ??
cod
Base
s
$\mathcal{M}$
$\widehat{\mathrm{Con}}$
$\lambda X\,.\,\mathcal{M}(s[\![-]\!], X)$
T

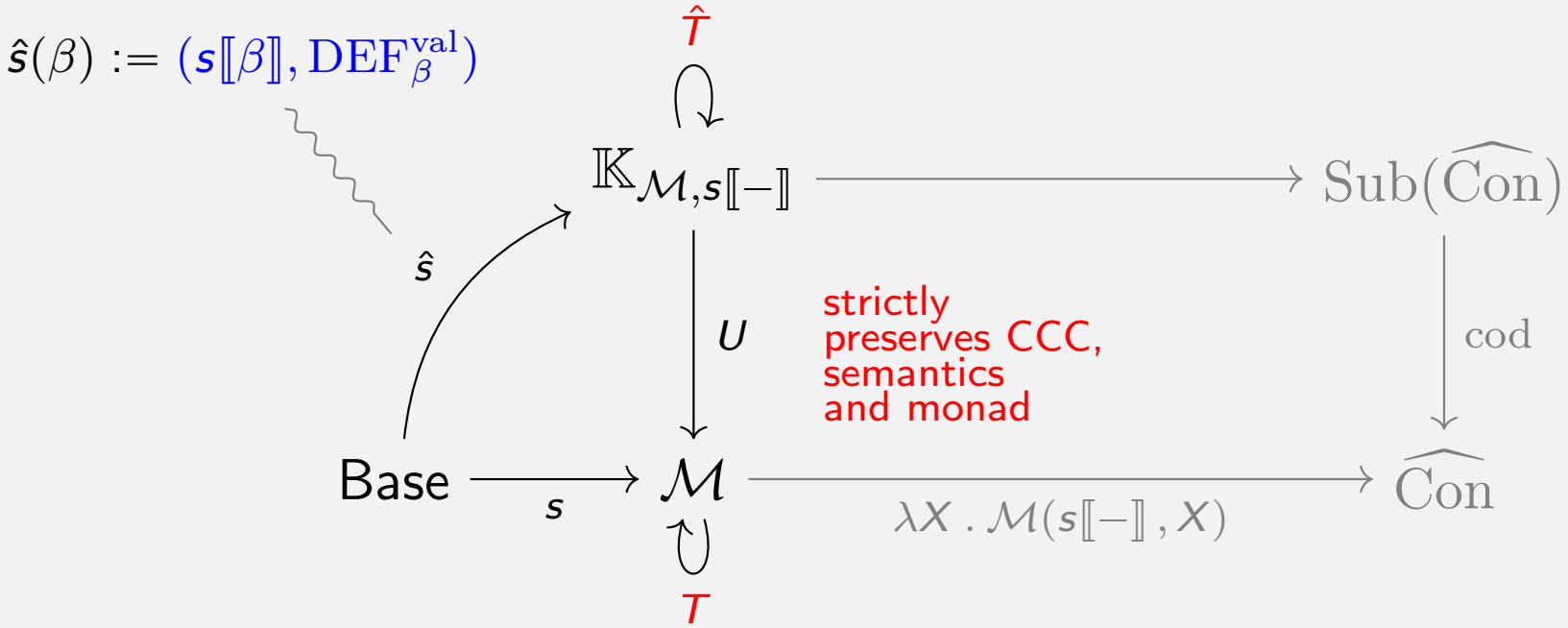


Restricting to values

for $(TX, R) \in \mathbb{K}_{\mathcal{M},F}$

$$R^{\mathrm{val}}(\Gamma) := \left\{ f \;\middle|\; \eta_X \circ f \in R(\Gamma) \right\}$$

so $(X, R^{\mathrm{val}}) \in \mathbb{K}_{\mathcal{M},F}$

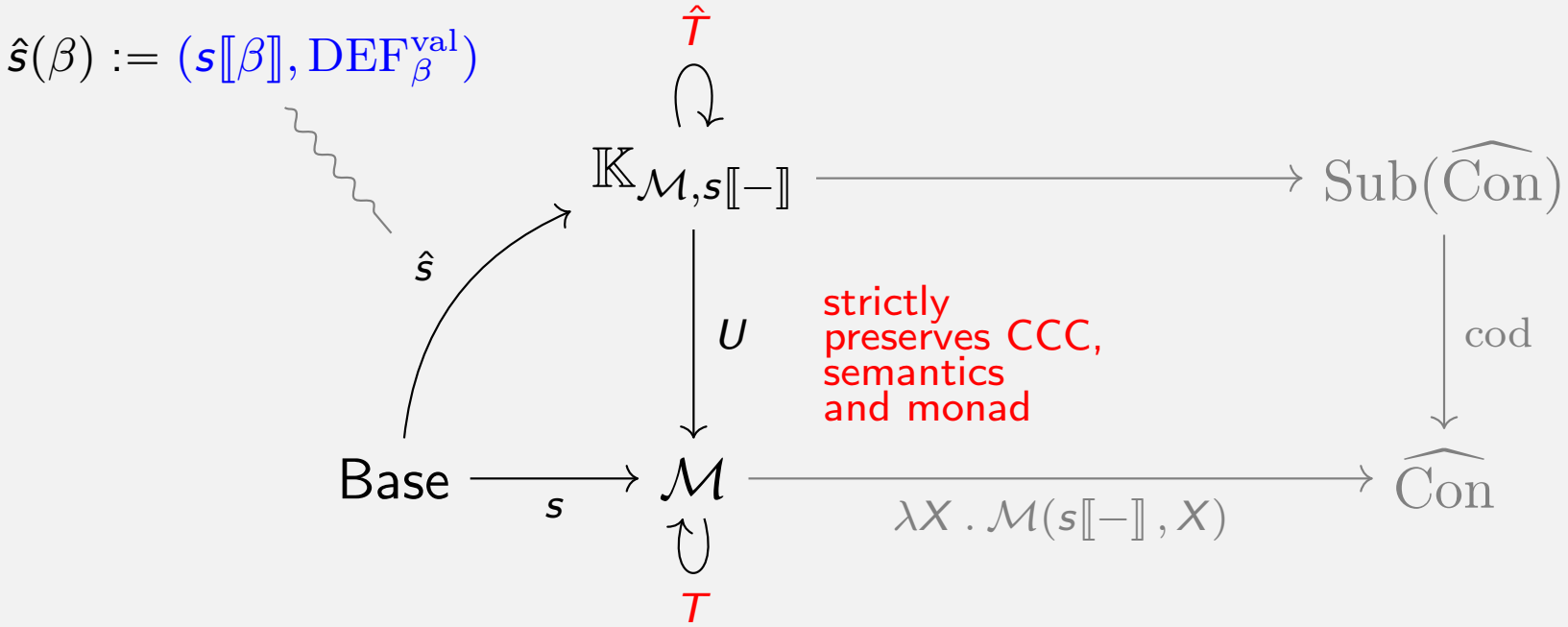


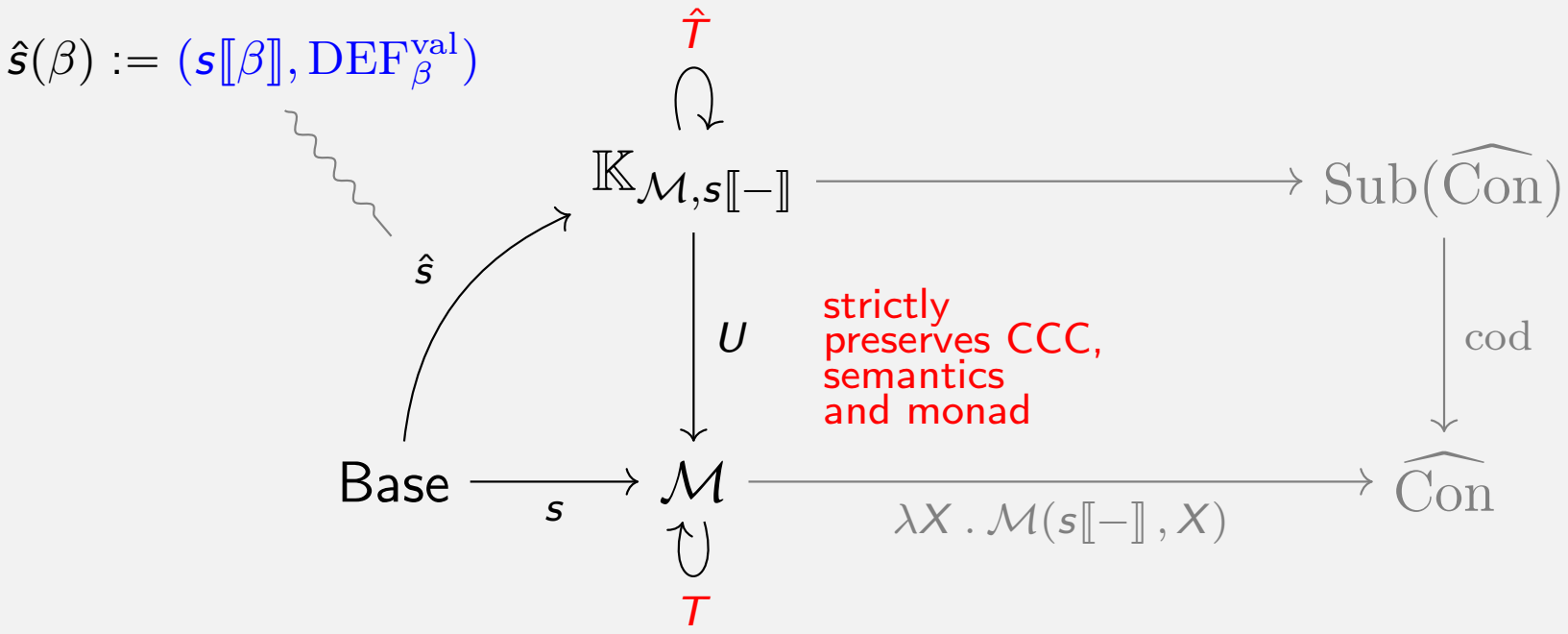
Restricting to values

for $(TX, R) \in \mathbb{K}_{\mathcal{M},F}$

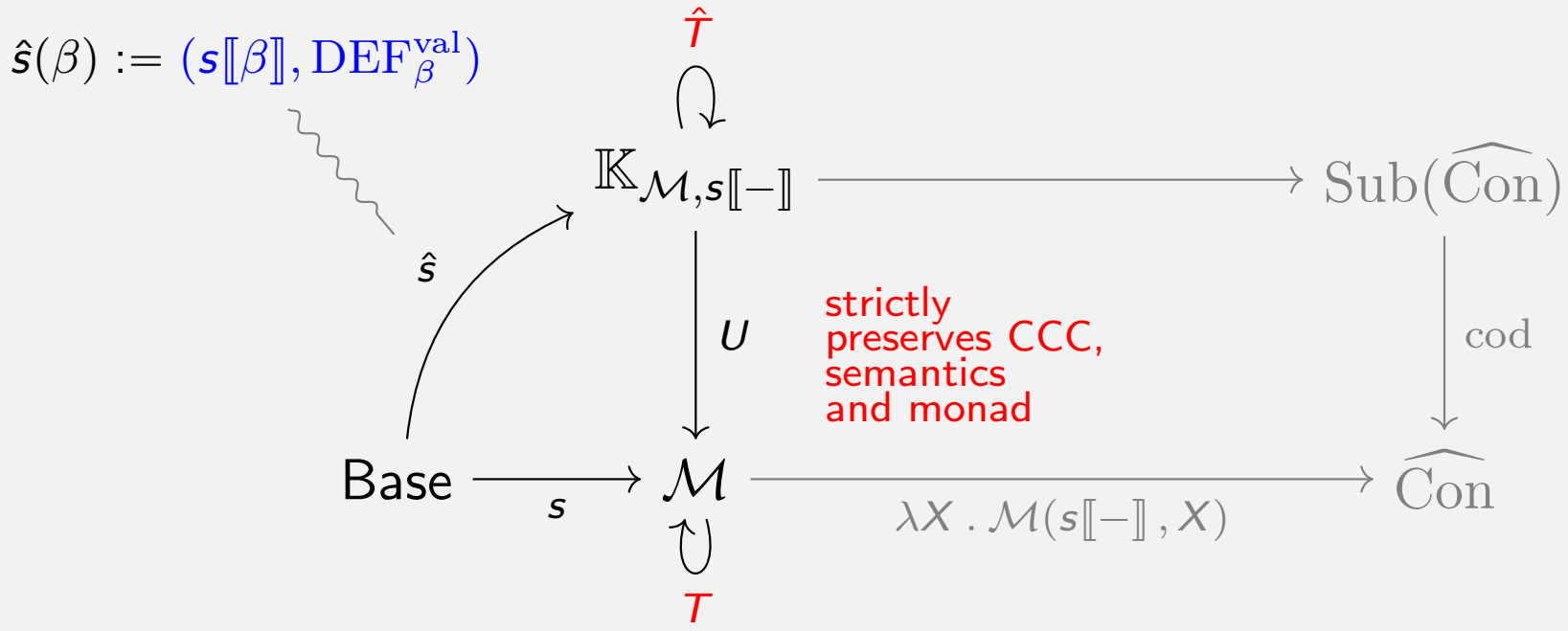
$$R^{\mathrm{val}}(\Gamma) := \left\{ f \;\middle|\; \eta_X \circ f \in R(\Gamma) \right\}$$

so $(X, R^{\mathrm{val}}) \in \mathbb{K}_{\mathcal{M},F}$


$\hat{s}(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$
$\hat{T}$
$\mathbb{K}_{\mathcal{M},s[\![-]\!]}$
$\widehat{\mathrm{Sub}(\mathrm{Con})}$
$\hat{s}$
U
strictly
preserves CCC,
semantics
and monad
cod
Base
s
$\mathcal{M}$
$\widehat{\mathrm{Con}}$
$\lambda X\,.\,\mathcal{M}(s[\![-]\!]\,, X)$
T

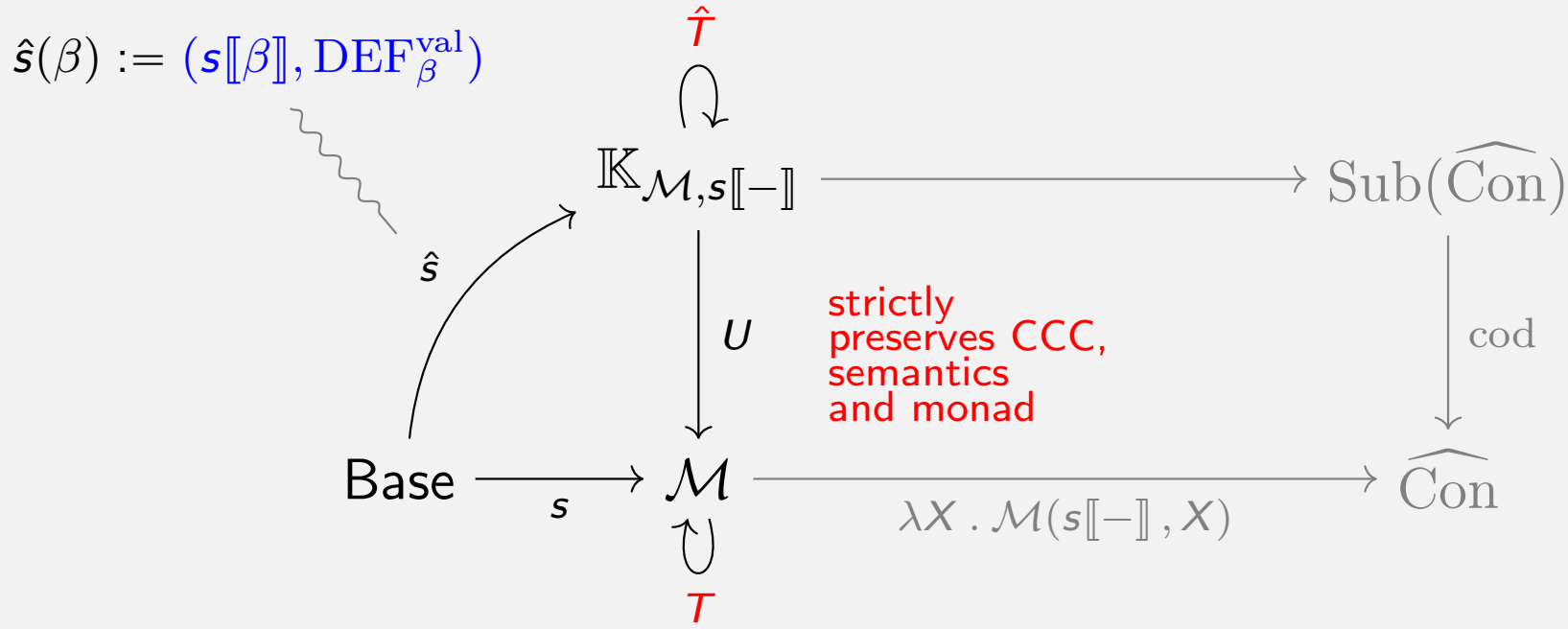


choosing $\hat{T}$ by $\top\top$-lifting



choosing $\hat{T}$ by $\top\top$-lifting ... for every type σ:

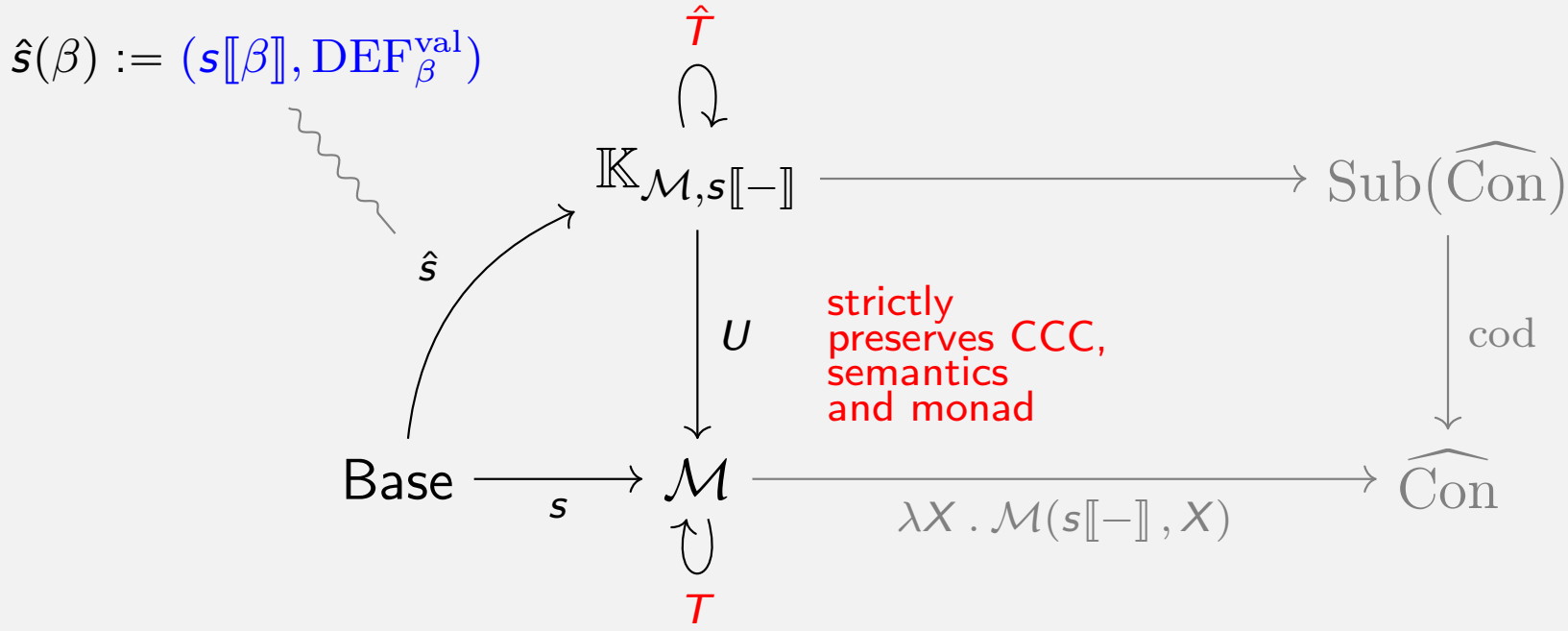
$$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}^{\mathrm{val}}_{\sigma}) \quad \text{and} \quad \hat{T}\hat{s}[\![\sigma]\!] = (Ts[\![\sigma]\!], \mathrm{DEF}_{\sigma})$$



choosing $\hat{T}$ by $\top\top$-lifting ... for every type σ:

$$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}^{\mathrm{val}}_{\sigma}) \quad \text{and} \quad \hat{T}\hat{s}[\![\sigma]\!] = (Ts[\![\sigma]\!], \mathrm{DEF}_{\sigma})$$

$f : s[\![\Gamma]\!] \to s[\![\sigma]\!]$ is definable $\iff f : \hat{s}[\![\Gamma]\!] \to \hat{s}[\![\sigma]\!]$



choosing $\hat{T}$ by $\top\top$-lifting ... for every type σ:

$$\hat{s}[\![\sigma]\!] = (s[\![\sigma]\!], \mathrm{DEF}^{\mathrm{val}}_{\sigma}) \quad \text{and} \quad \hat{T}\hat{s}[\![\sigma]\!] = (Ts[\![\sigma]\!], \mathrm{DEF}_{\sigma})$$

the definable maps are exactly those
that lift to $(\mathbb{K}_{\mathcal{M},s[\![-]\!]}, \hat{T}, \hat{s})$

The route so far

- ✓ the model we've just seen, abstractly
- ✓ restricting to definable morphisms
- 3. the abstract OHR construction $\rightsquigarrow$ follows pattern just seen
- 4. getting full abstraction

The abstract OHR construction

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

---

$$\begin{array}{c} \mathcal{M} \\ \circlearrowleft \\ T \end{array}$$

A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces

---

$$\prod_{i \in \mathbb{I}} \begin{pmatrix} \text{predicates} \\ \text{on } \mathbb{B}_i \end{pmatrix}$$

$$\Big\downarrow \text{ fibration for logical relations}$$

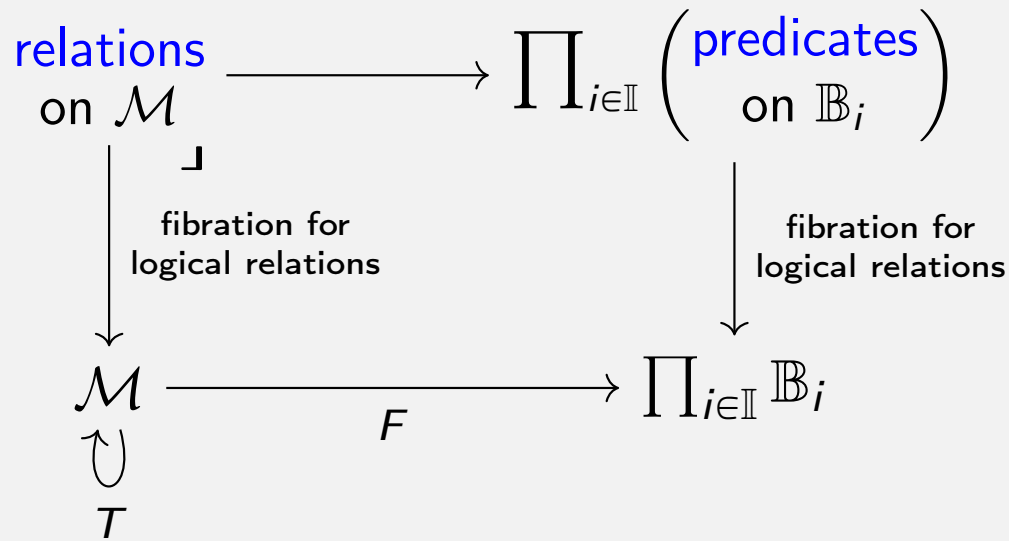
$$\prod_{i \in \mathbb{I}} \mathbb{B}_i$$

$$\mathcal{M} \circlearrowright T$$

NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ

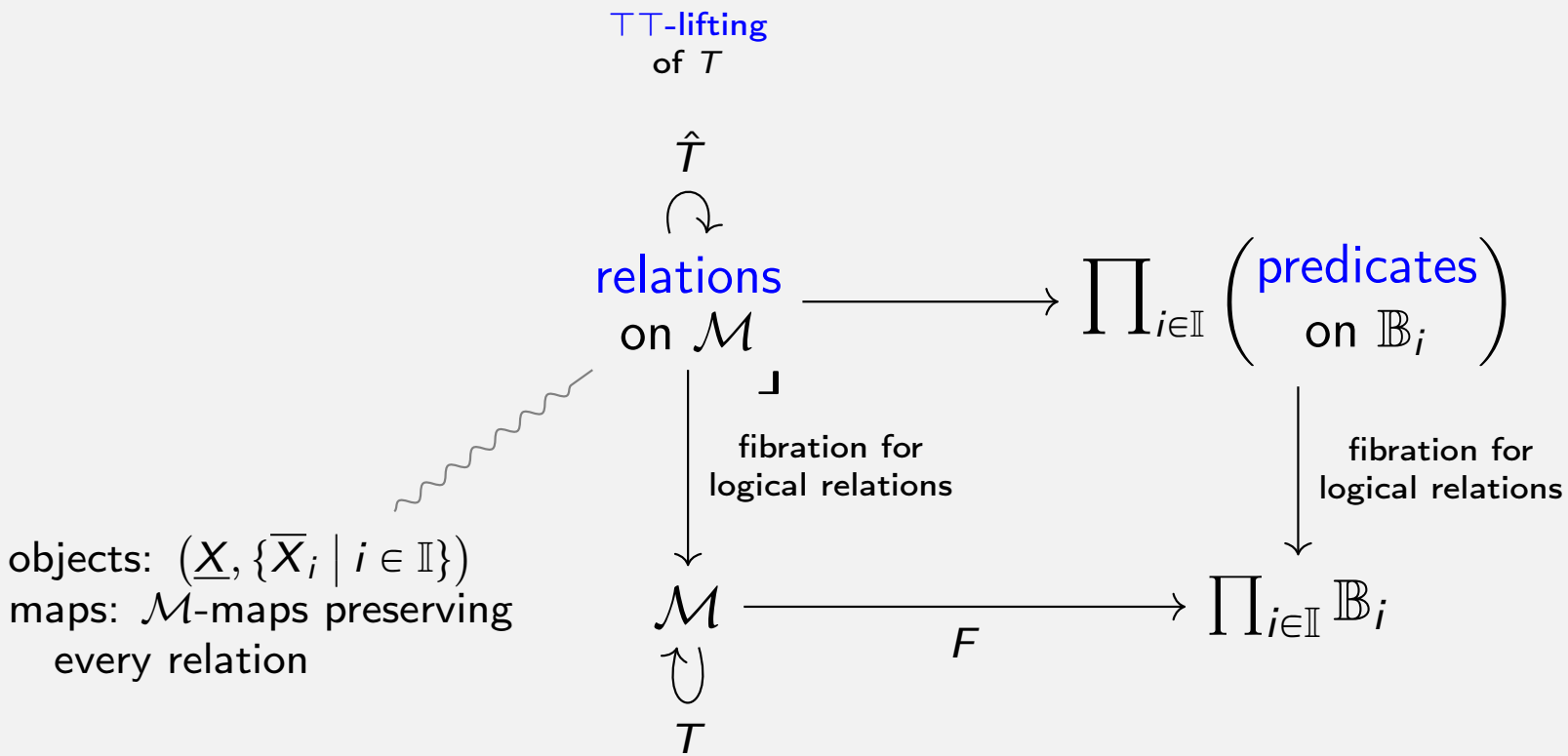
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ

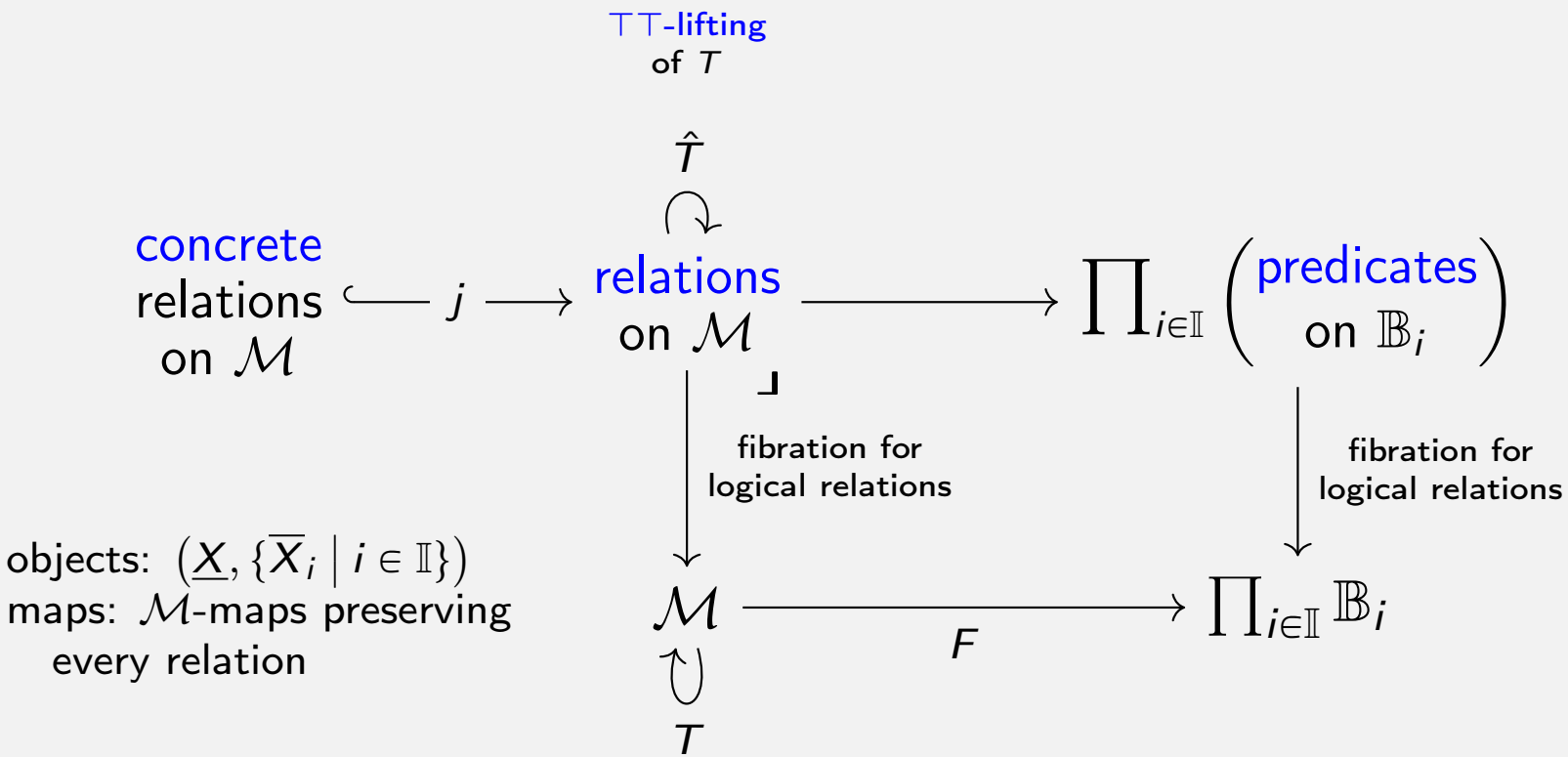
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ

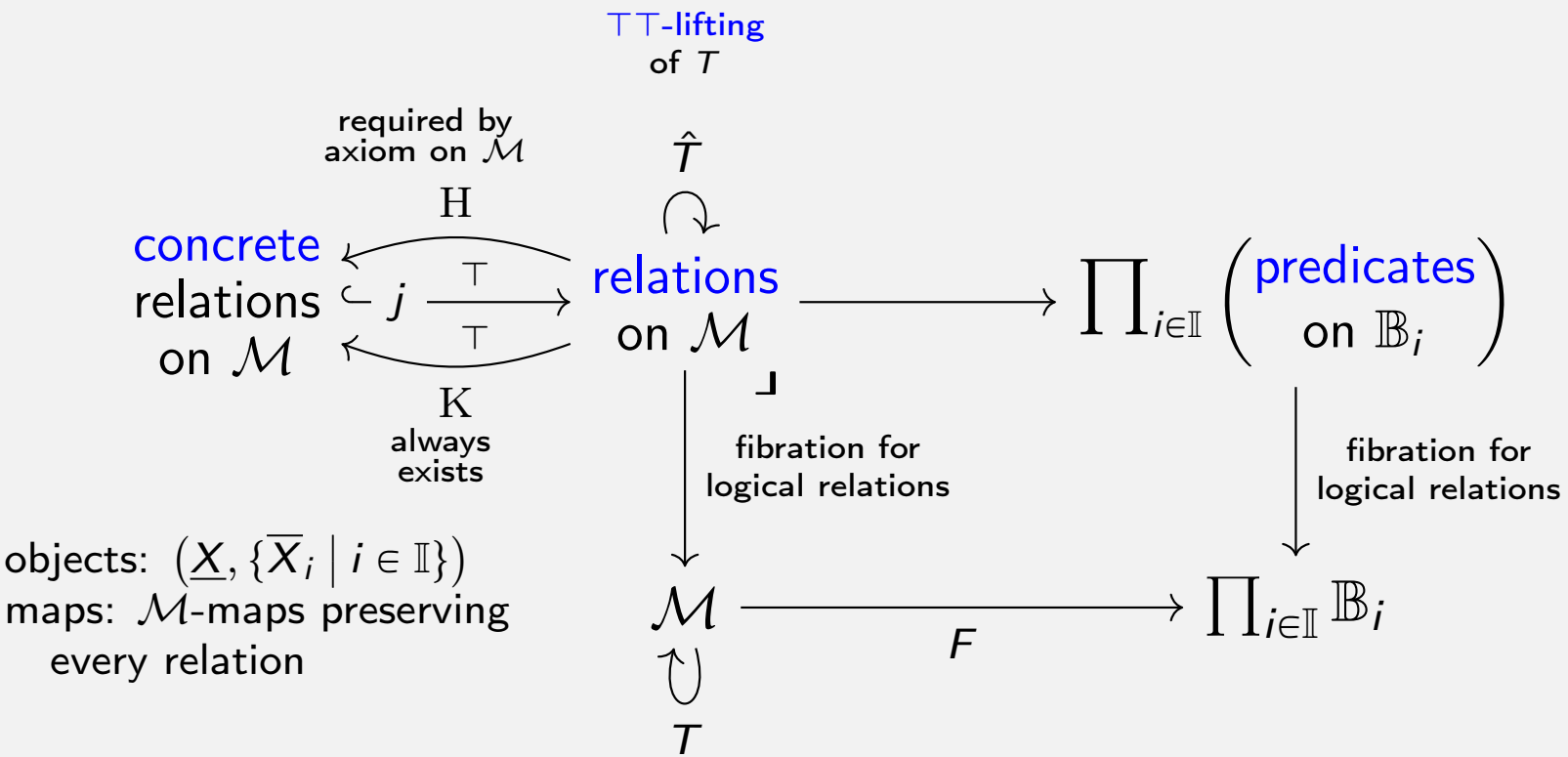
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ

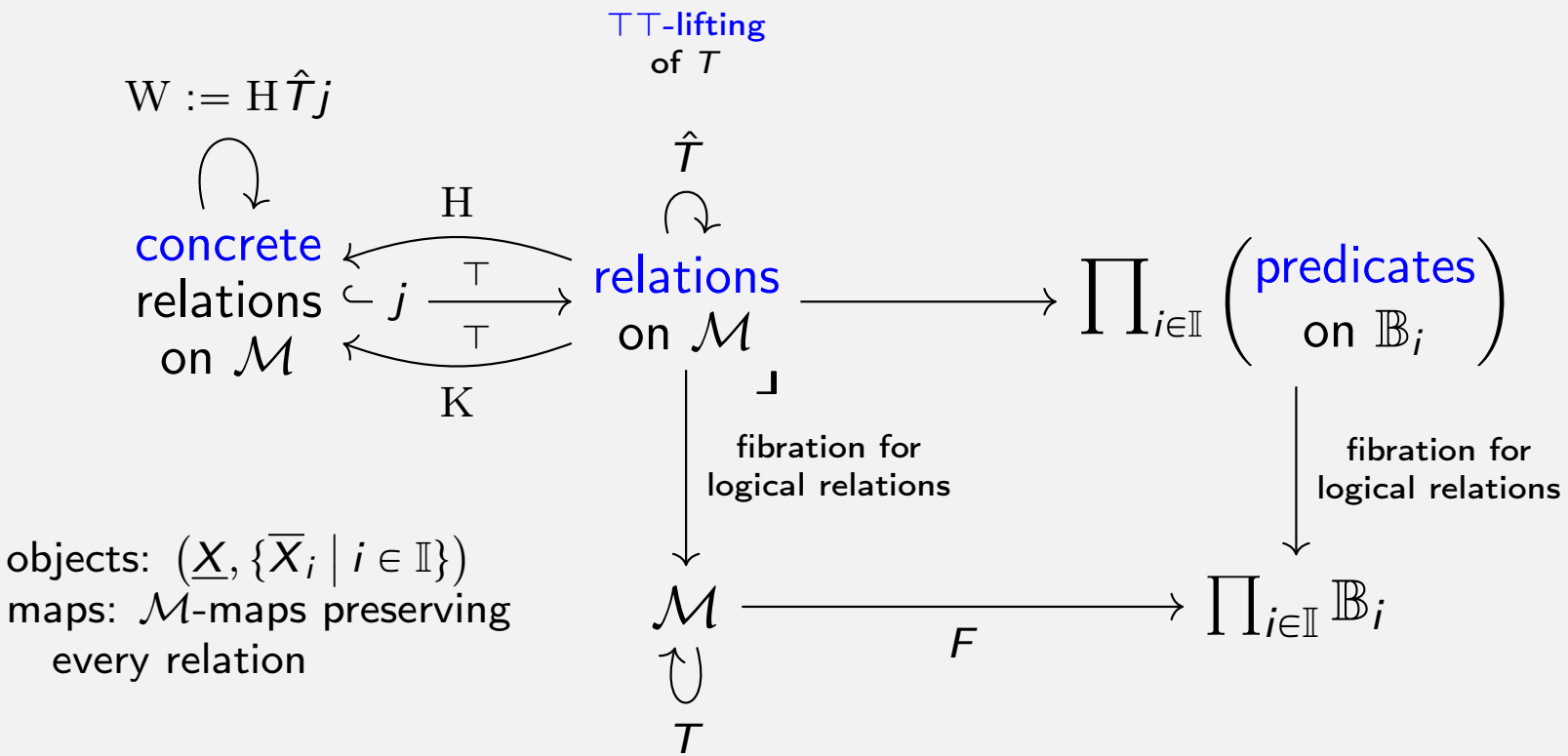
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ

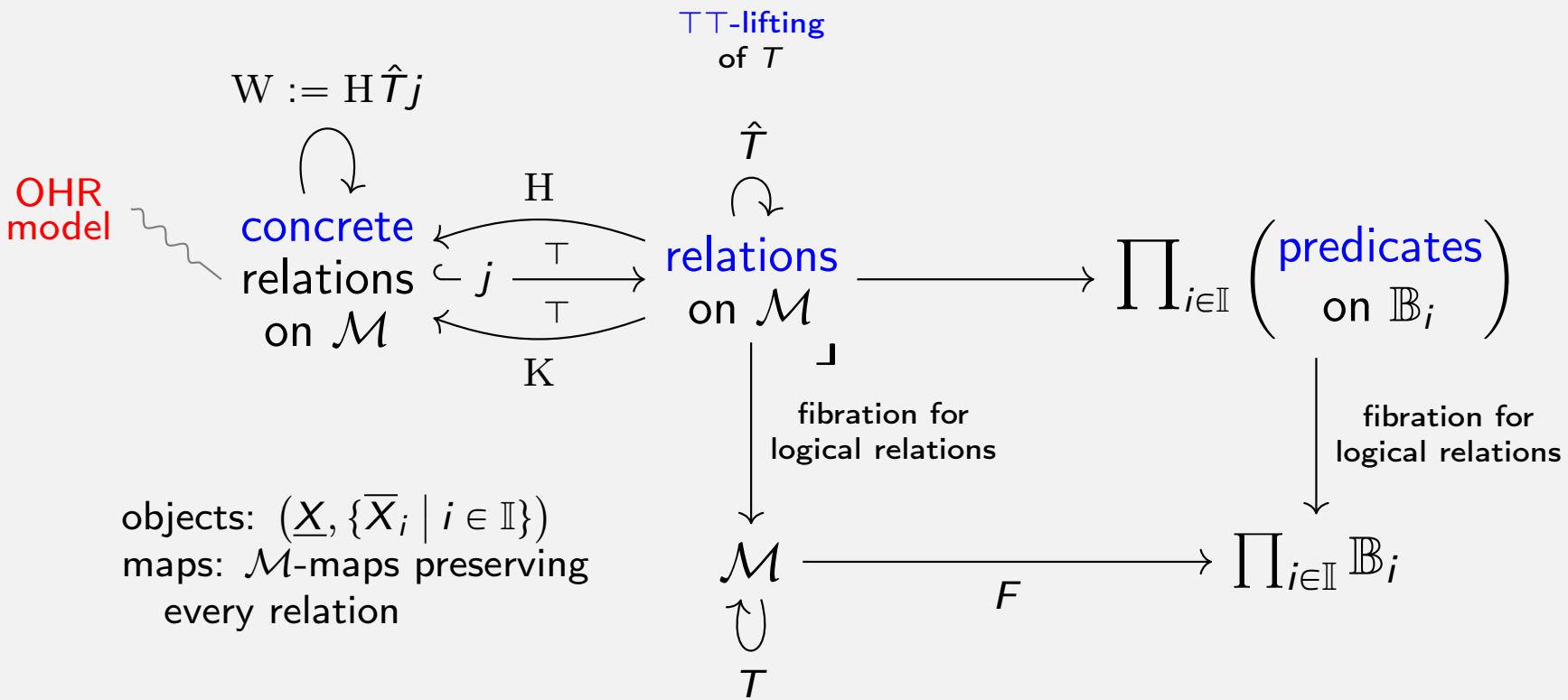
1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

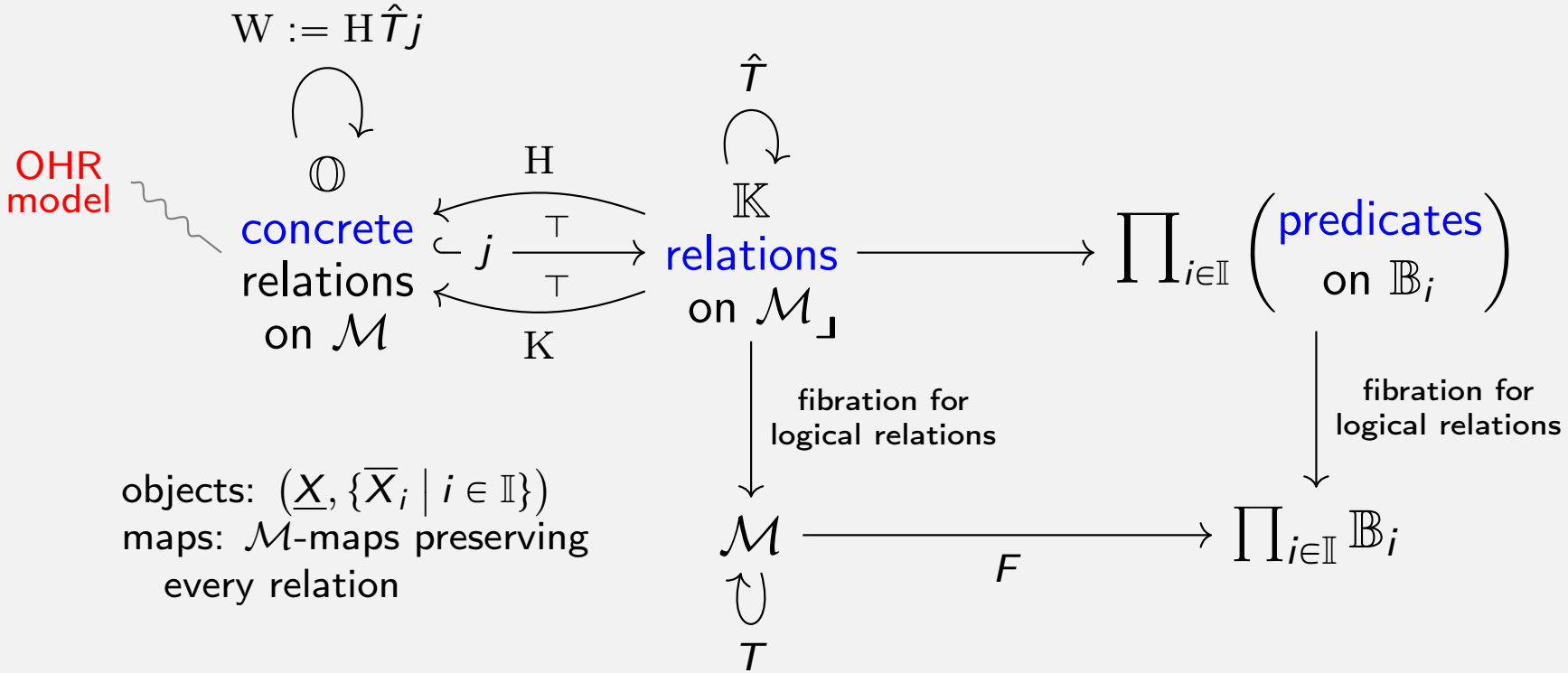
A recipe for removing bad morphisms κ

1. Use relations $\rightsquigarrow$ stops them being morphisms
2. Use concreteness $\rightsquigarrow$ cuts them out function spaces



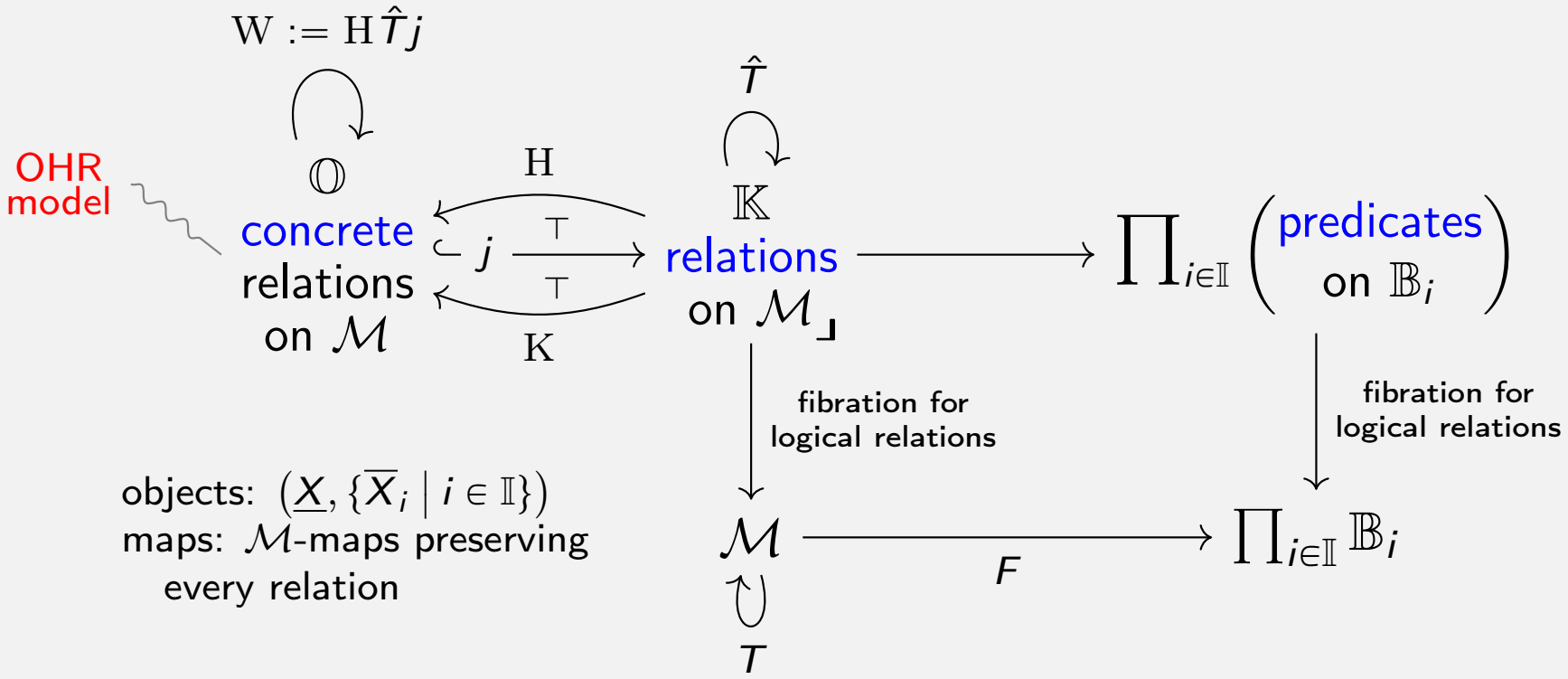
NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

A recipe for removing bad morphisms κ



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

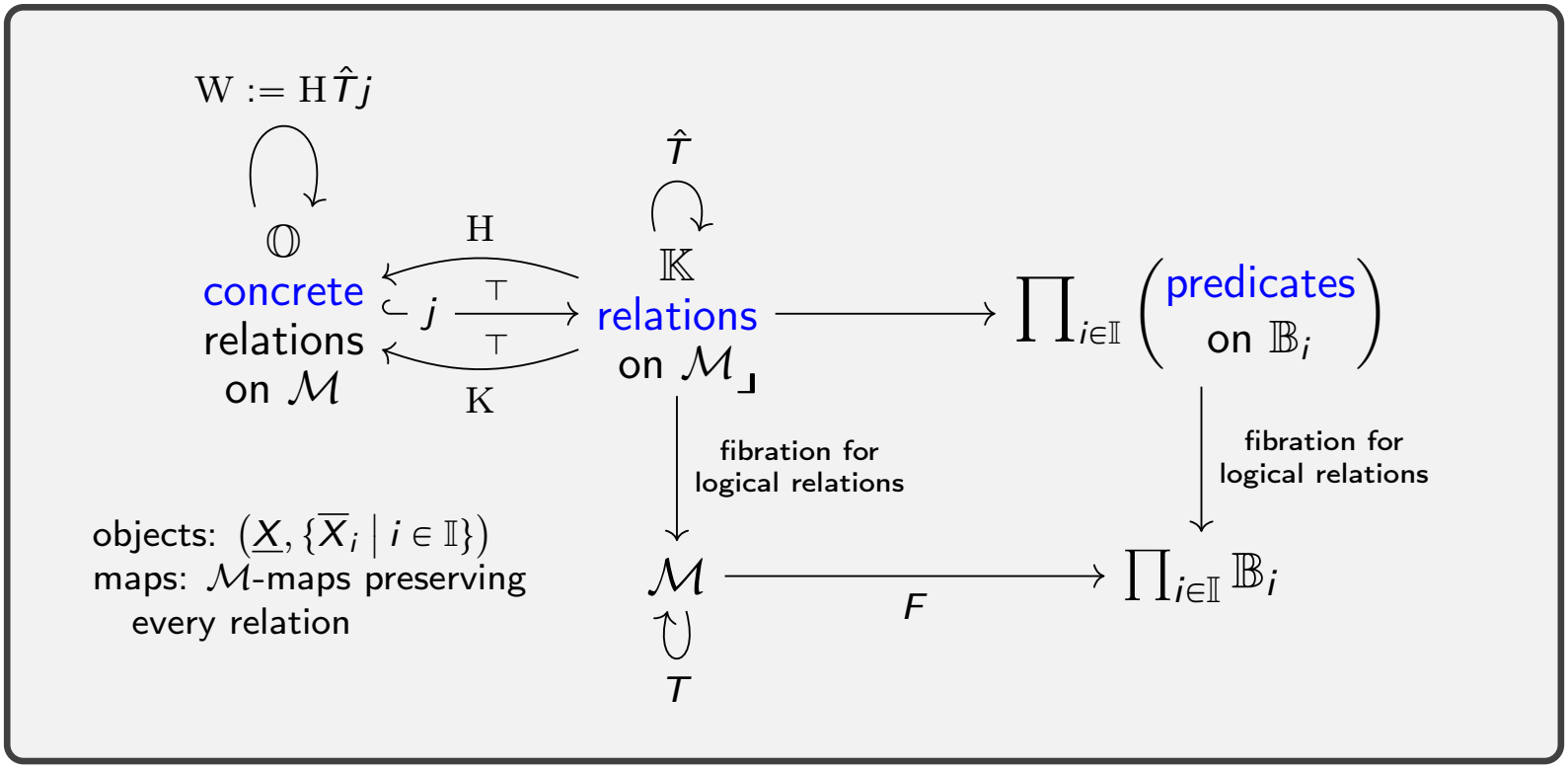
A recipe for removing bad morphisms κ



NB: $\mathbb{I}$-many categories on RHS $\rightsquigarrow$ $\mathbb{I}$-many relations

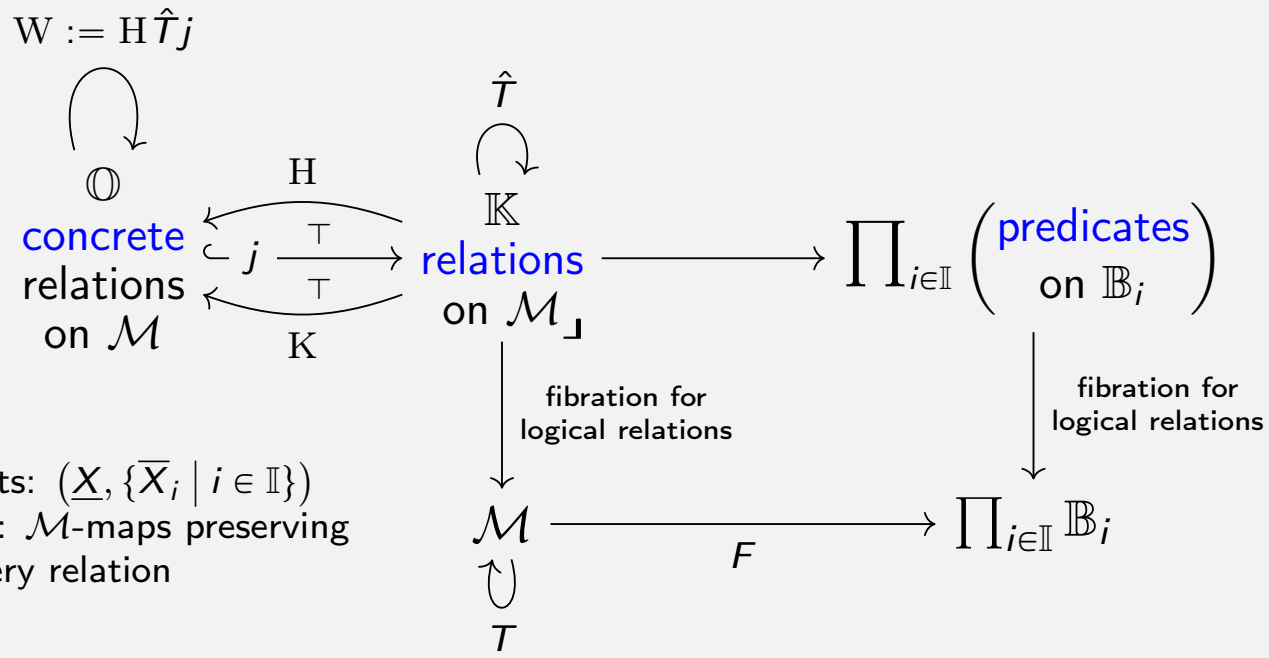
Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$



Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

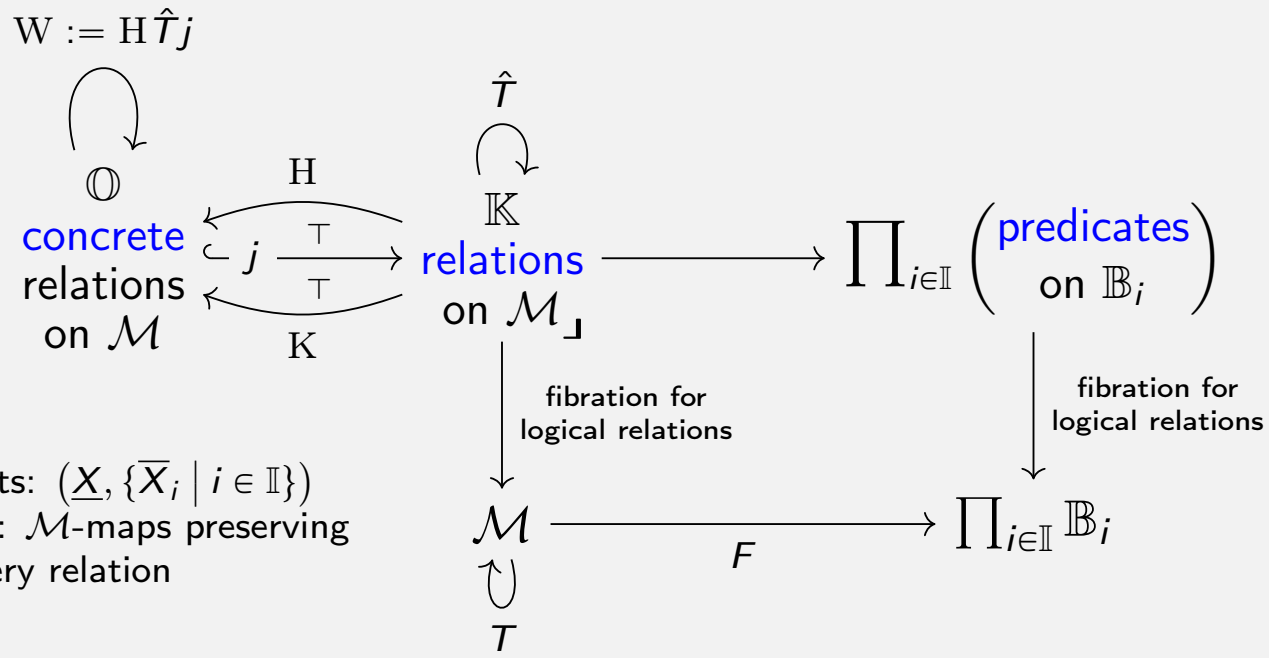


Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!] \text{ preserves every relation, } \ldots$$

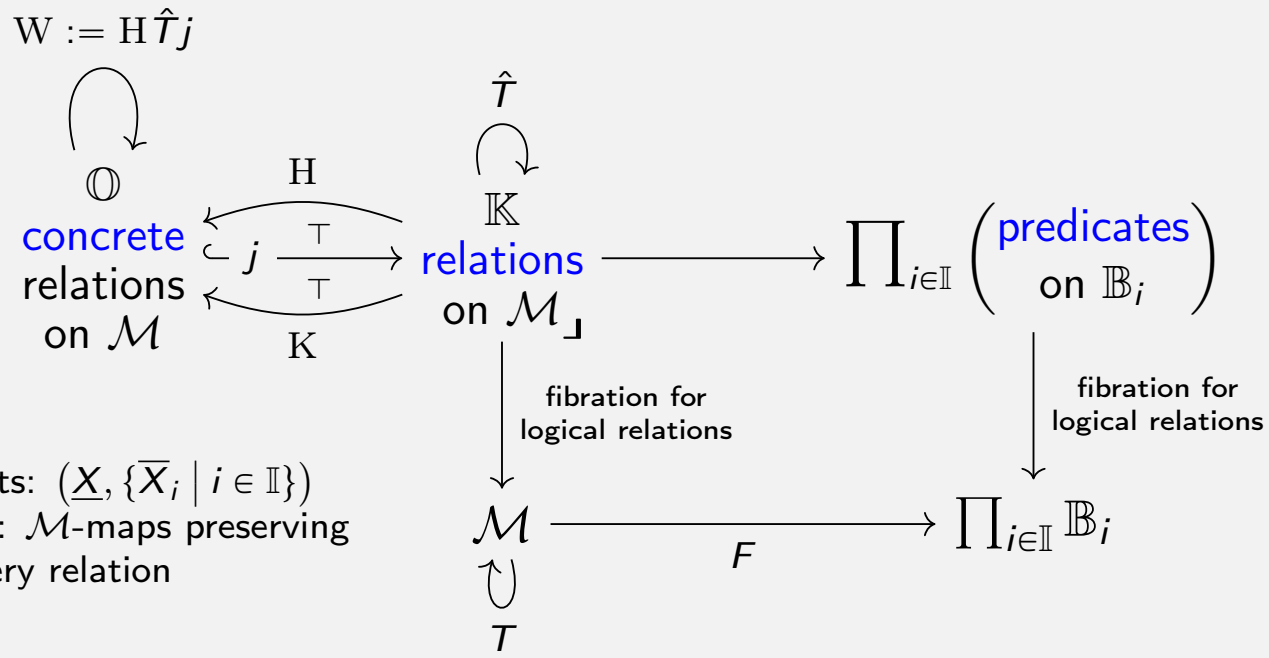


Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!] \text{ preserves every relation, } \ldots$$

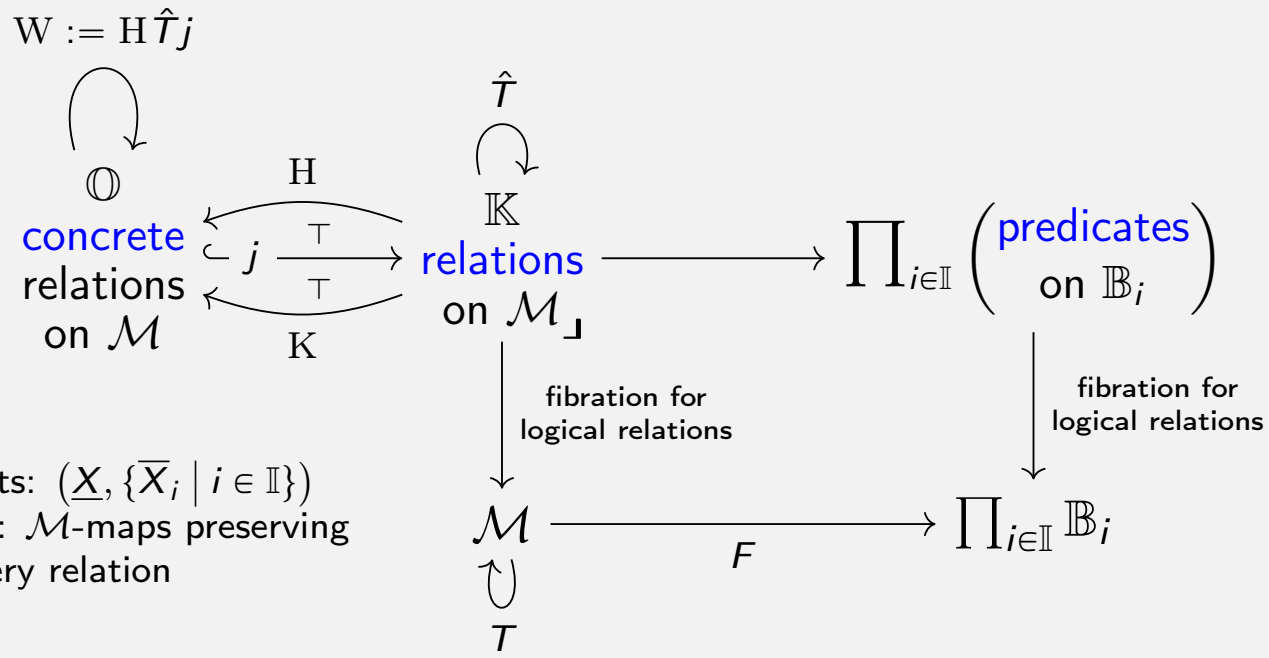


Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!] \text{ preserves every relation, } \ldots$$

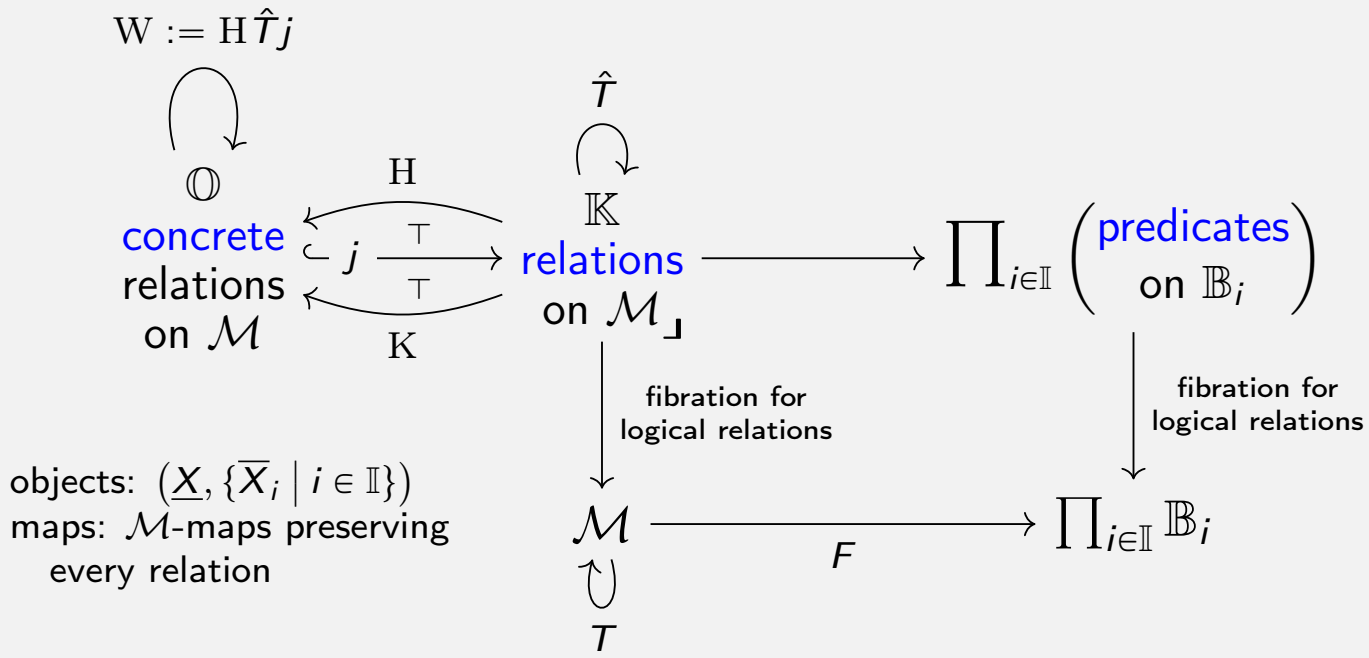


Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...

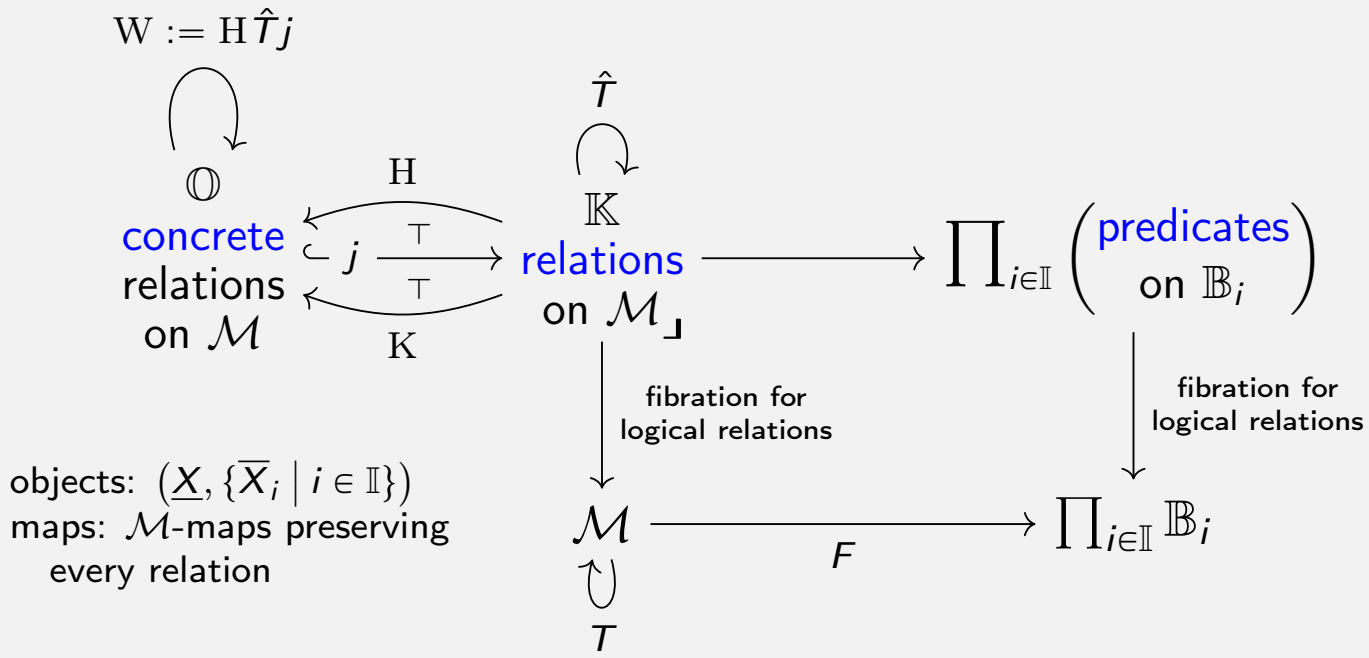


Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable



Key trick: choose $\mathbb{I}$, F, $\mathbb{B}_i$ and $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ with.

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable $\rightsquigarrow$ so every map is definable!

Fact:

Fact:

1. $\mathbb{O}$ is always well-pointed;

Fact:

1. $\mathbb{O}$ is always well-pointed;
2. any well-pointed model with every morphism definable is fully abstract.

Fact:

1. $\mathbb{O}$ is always well-pointed;
2. any well-pointed model with every morphism definable is fully abstract.

$\Rightarrow$ it remains to instantiate the construction

Fact:

1. $\mathbb{O}$ is always well-pointed;
2. any well-pointed model with every morphism definable is fully abstract.

$\Rightarrow$ it remains to instantiate the construction

Strategy: see what we need as we go along!

Getting full abstraction

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

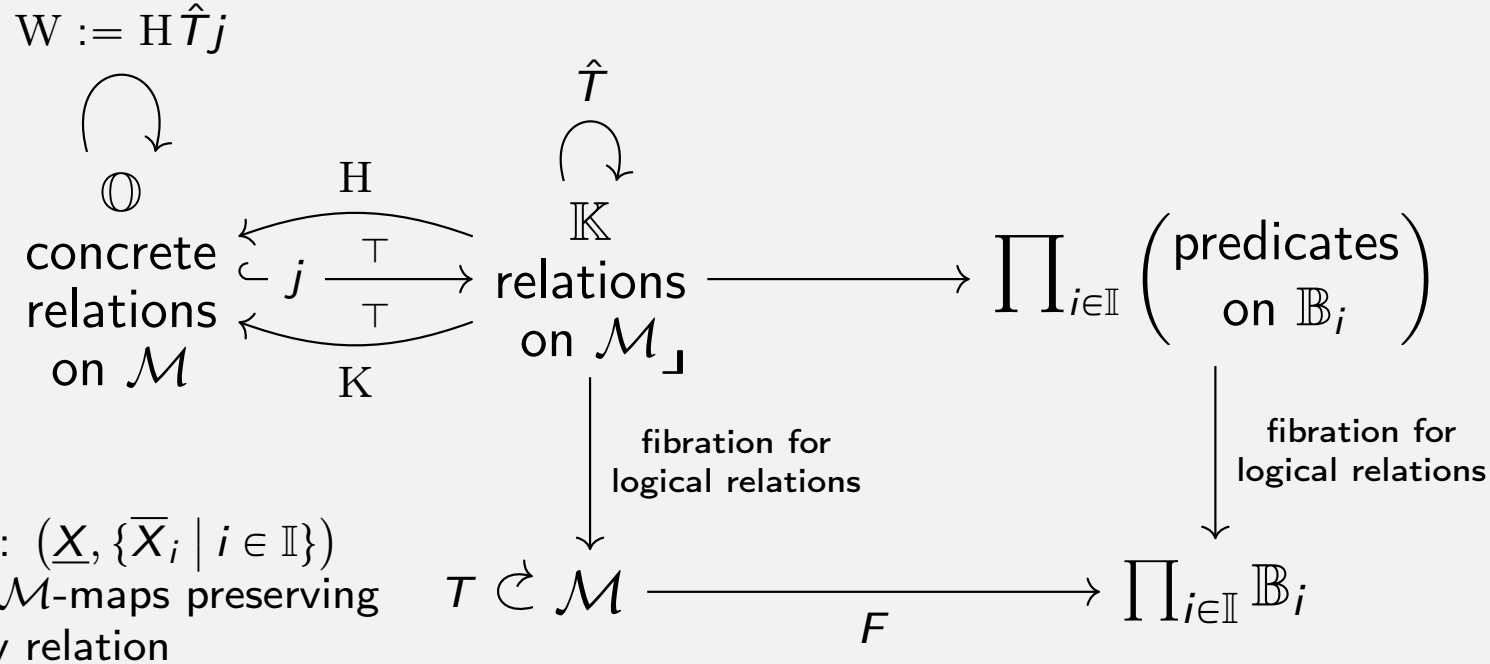
$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

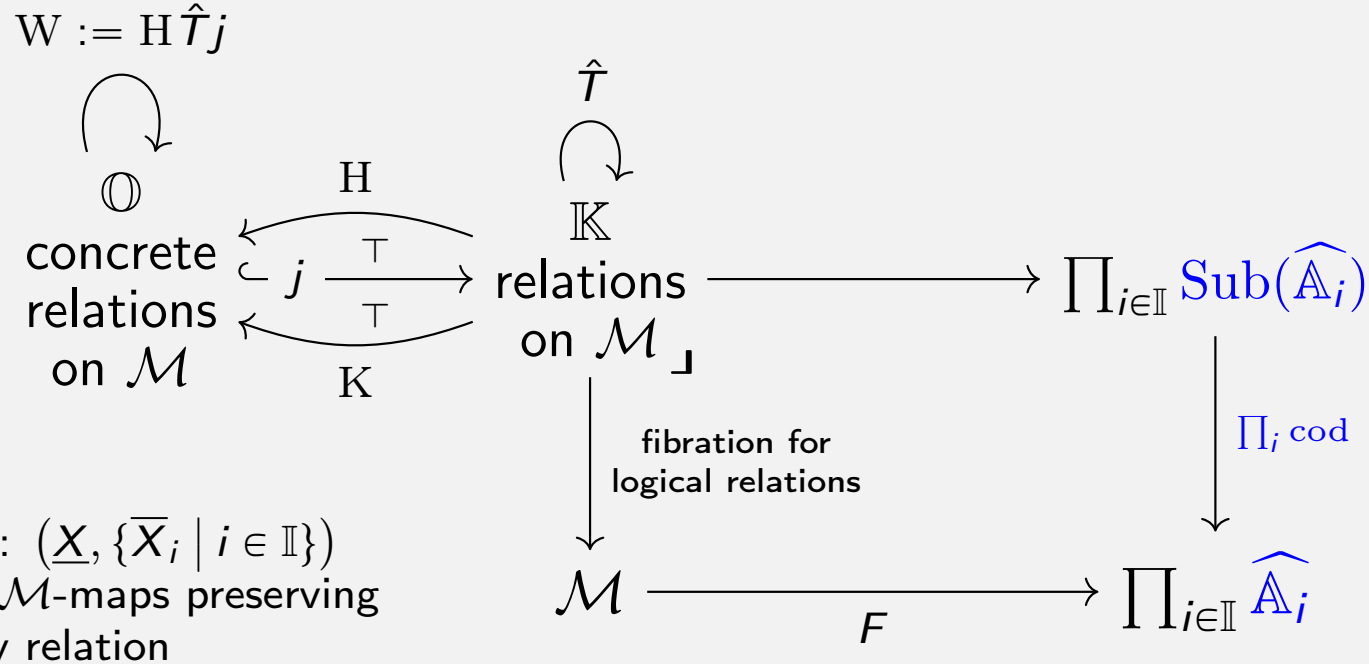


Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

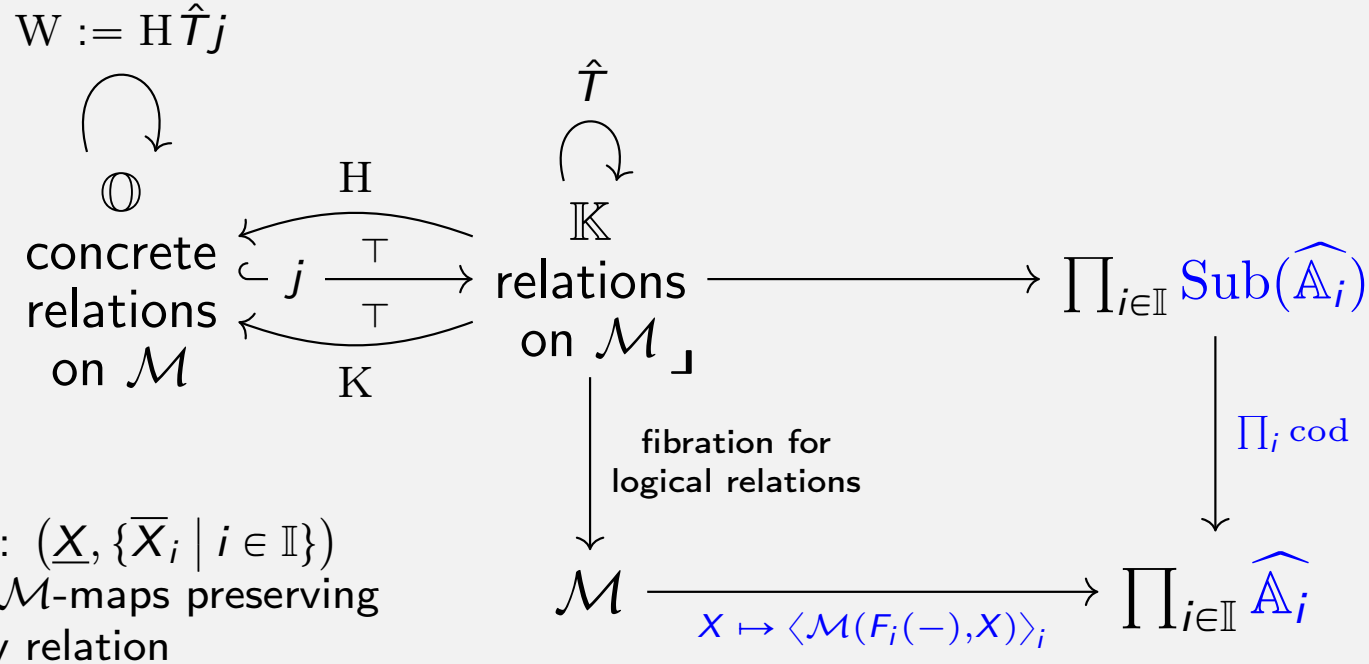


Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable



Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

---

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve DEF

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$ in which maps preserve DEF ⇝ construct the DEF predicate

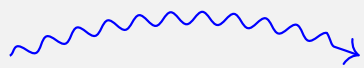
Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$  construct the
in which maps preserve DEF  DEF predicate

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

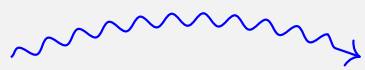
$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation



construct the
DEF predicate

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

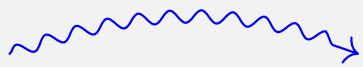
$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation



construct the
DEF predicate

identify DEF as one
of the preserved predicates

Key trick: Choose data and an interpretation $\hat{s}$ so that $\exists i_0 \in \mathbb{I}$ such that

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

then:

$f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ preserves every relation, ...
so f preserves DEF...
so f is definable

Chickens and eggs ⇝ *c.f.* impredicativity

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation

⇝ construct the DEF predicate

⇝ identify DEF as one of the preserved predicates

Chickens and eggs ⇝ *c.f.* impredicativity

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation

⇝ construct the
DEF predicate

⇝ identify DEF as one
of the preserved predicates

Chickens and eggs ⇝ *c.f.* impredicativity

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation

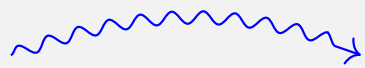
construct the
DEF predicate

identify DEF as one
of the preserved predicates

Need to quantify over relations on $\mathbb{O}$ before constructing $\mathbb{O}$

Chickens and eggs ⇝ *c.f.* impredicativity

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation

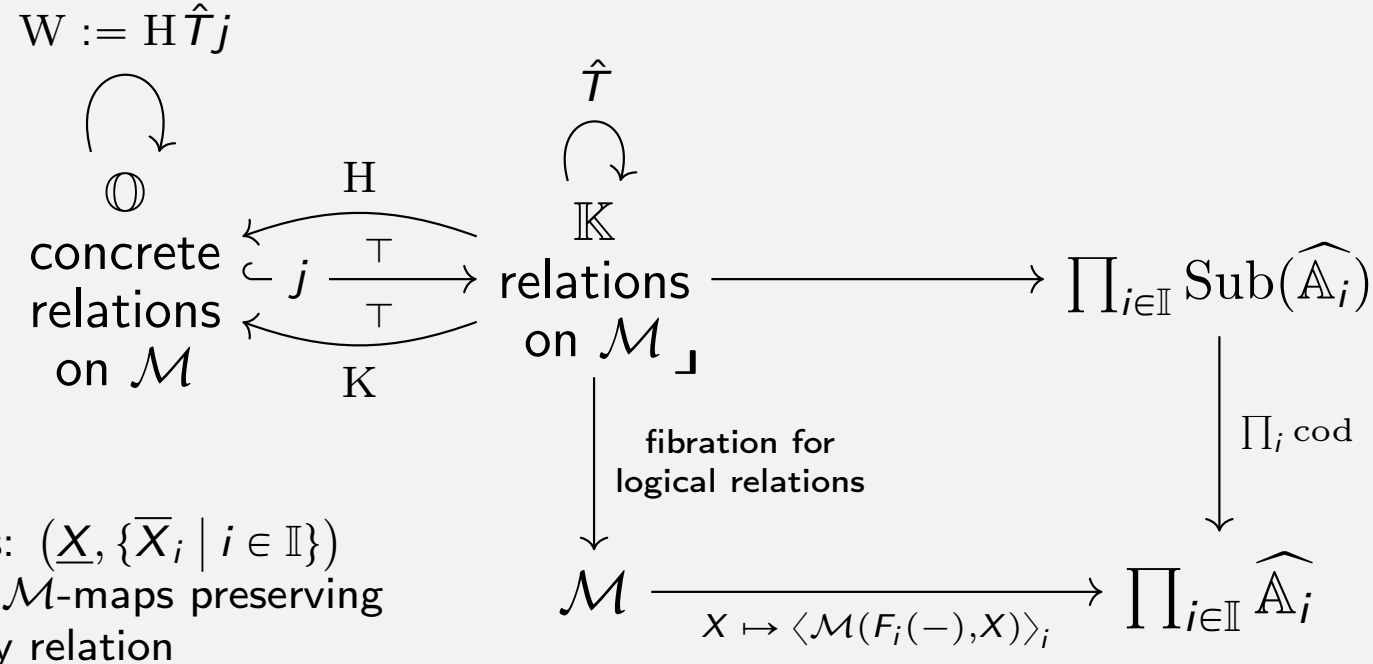


construct the
DEF predicate

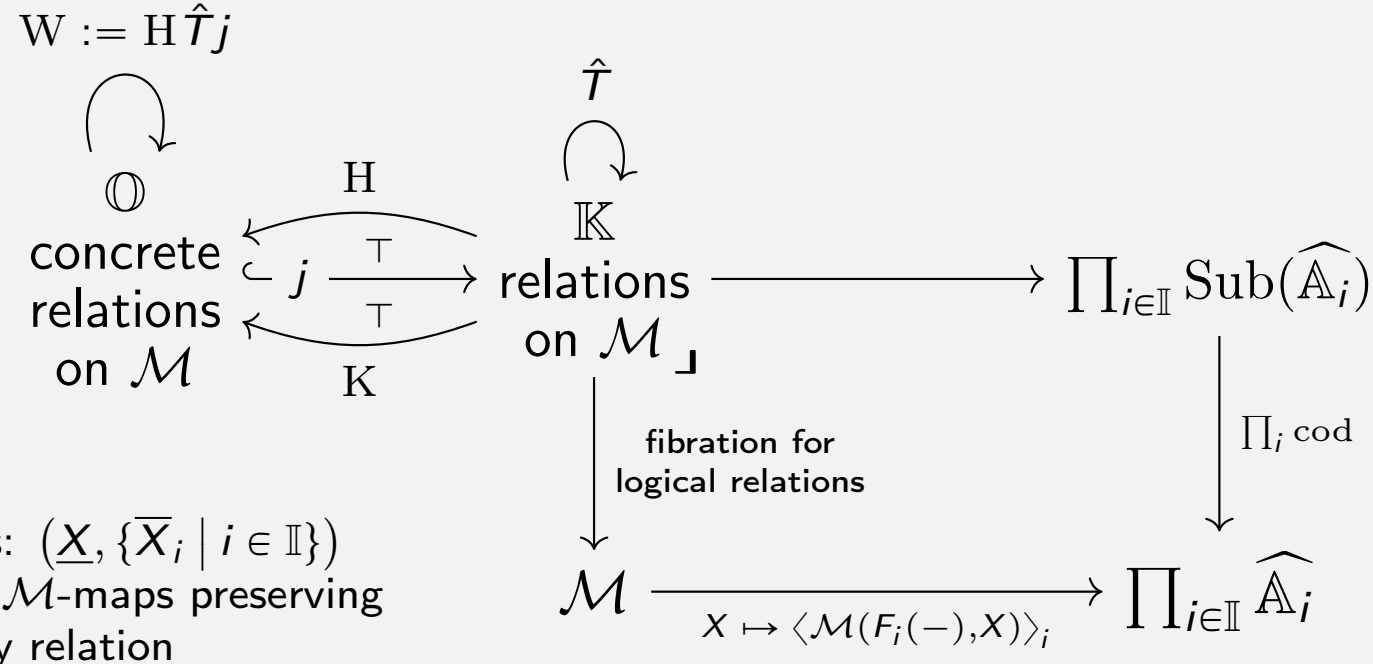
identify DEF as one
of the preserved predicates

Need to quantify over relations on $\mathbb{O}$ before constructing $\mathbb{O}$

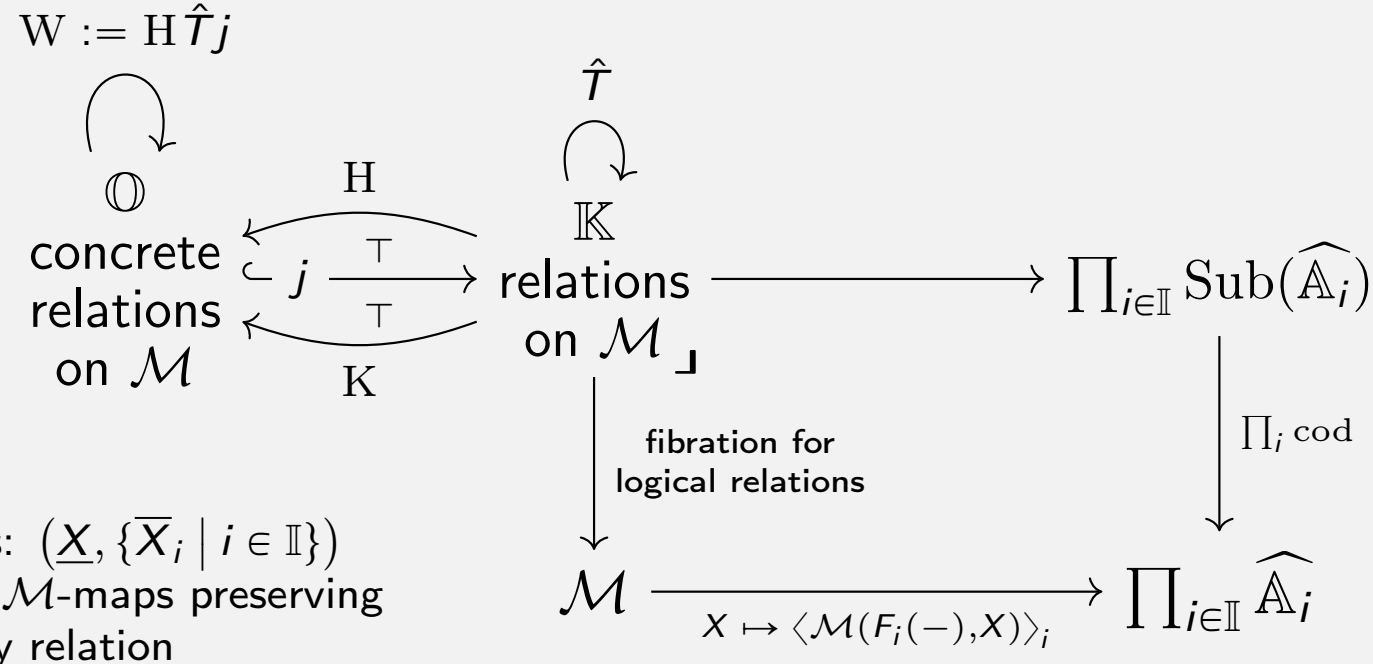
Solution: relations on $\mathbb{O}$ are relations on $\mathcal{M}$!



objects: $\left(\underline{X}, \{\overline{X}_i \mid i \in \mathbb{I}\}\right)$
maps: $\mathcal{M}$-maps preserving every relation

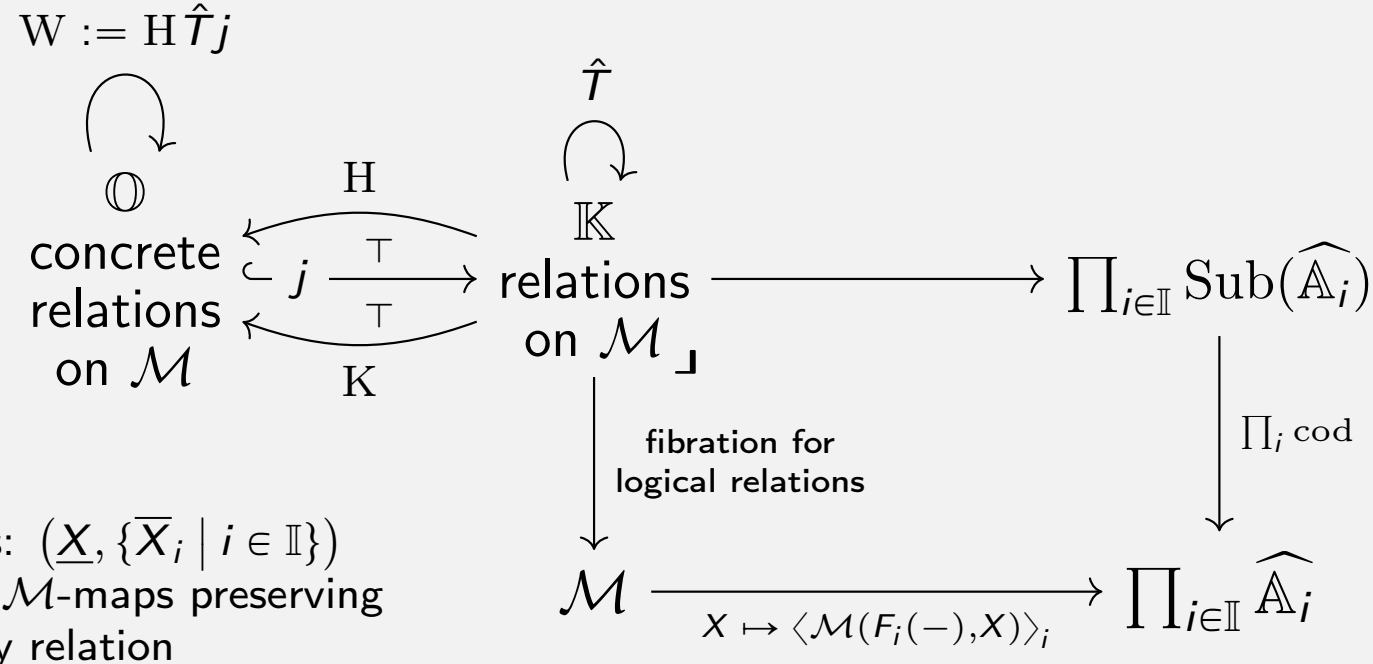


Have $U : \mathbb{O} \to \mathcal{M}$



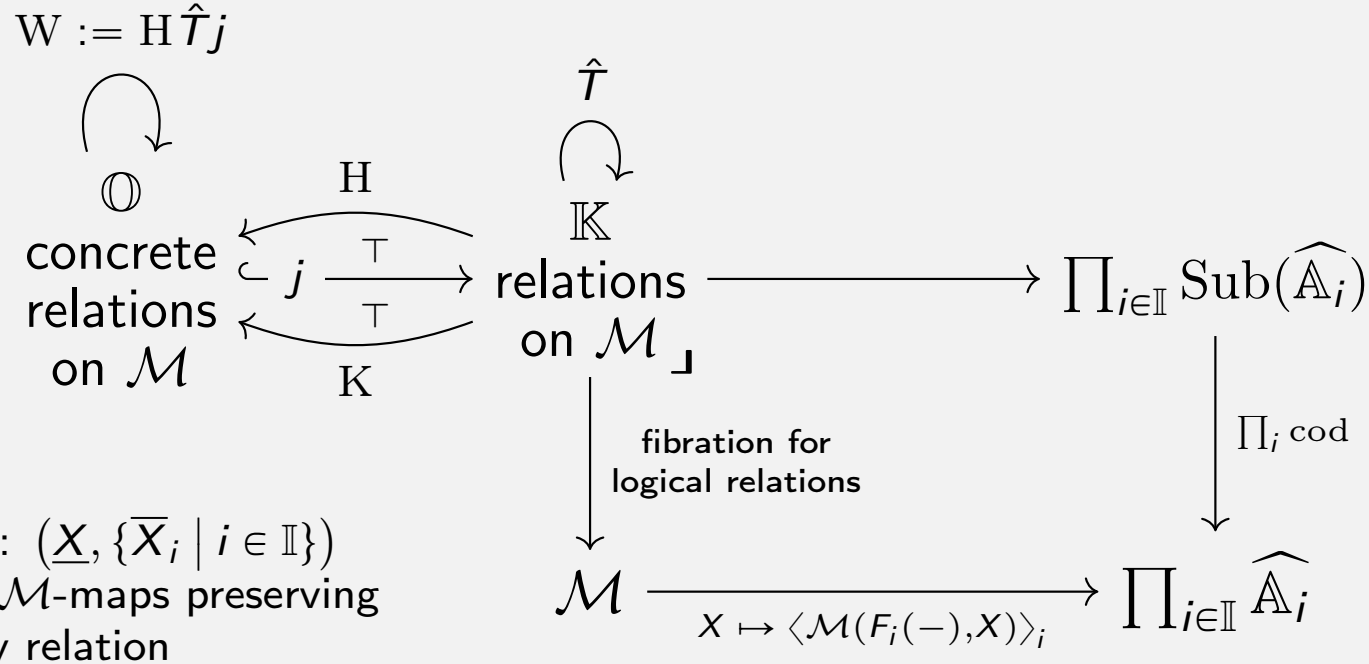
Have $U : \mathbb{O} \to \mathcal{M}$ so for any $X \in \mathbb{O}$ and $\mathcal{R}$ on X we get

$$\mathcal{R}(\Gamma) \subseteq \mathbb{O}(F(\Gamma), X)$$



Have $U : \mathbb{O} \to \mathcal{M}$ so for any $X \in \mathbb{O}$ and $\mathcal{R}$ on X we get

$$\mathcal{R}(\Gamma) \subseteq \mathbb{O}(F(\Gamma), X) \subseteq \mathcal{M}(UF(\Gamma), \underline{X})$$

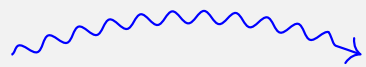


Have $U : \mathbb{O} \to \mathcal{M}$ so for any $X \in \mathbb{O}$ and $\mathcal{R}$ on X we get

$$\mathcal{R}(\Gamma) \subseteq \mathbb{O}(F(\Gamma), X) \subseteq \mathcal{M}(UF(\Gamma), \underline{X})$$

$\rightsquigarrow$ relations over $\mathbb{O}$ induce relations over $\mathcal{M}$!

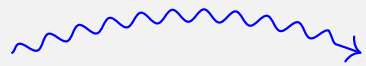
define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation



construct the
DEF predicate

identify DEF as one
of the preserved predicates

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$
in which maps preserve
every possible relation
over $(\mathcal{M}, T, s)$



construct the
DEF predicate

identify DEF as one
of the preserved predicates

Aim:

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$ in which maps preserve every possible relation over $(\mathcal{M}, T, s)$

Aim:

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$ in which maps preserve every possible relation over $(\mathcal{M}, T, s)$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations so that DEF must appear
2. define interpretation to look it up

Aim:

define a model $(\mathbb{O}, \mathrm{W}, \hat{s})$ in which maps preserve every possible relation over $(\mathcal{M}, T, s)$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations so that DEF must appear
2. define interpretation to look it up

Then: will get $\exists i_0 \in \mathbb{I}$

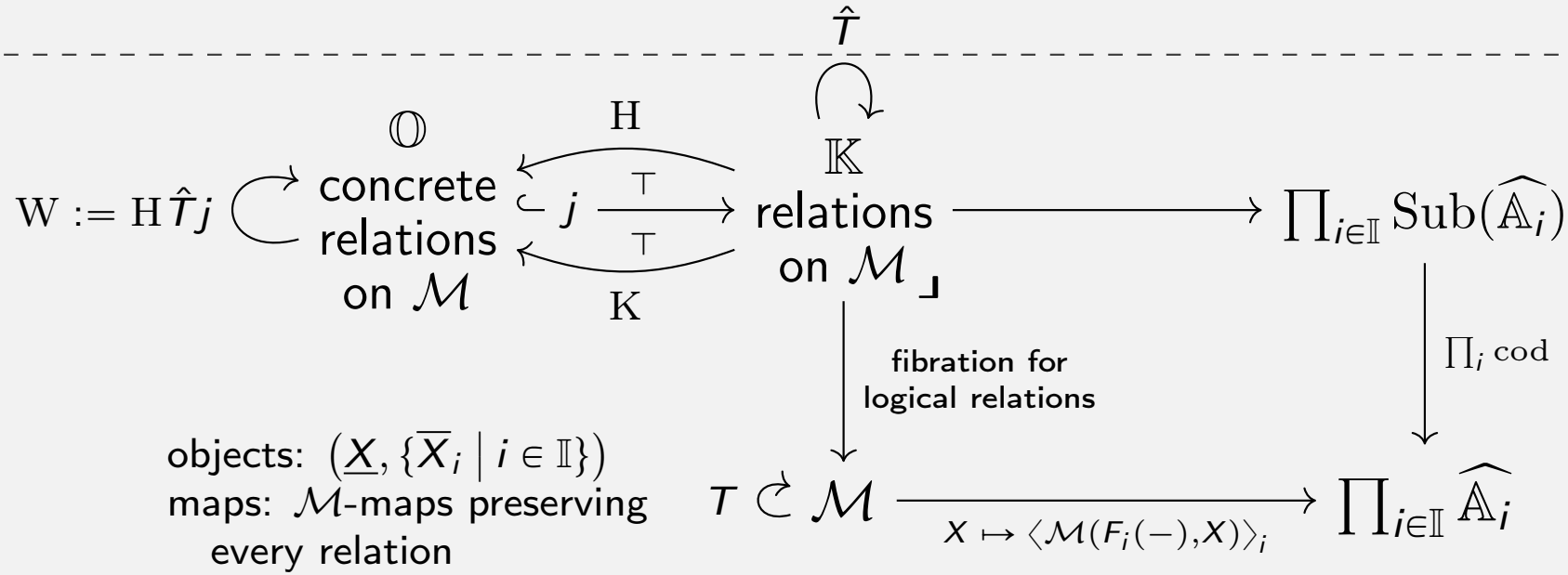
$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma$$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up

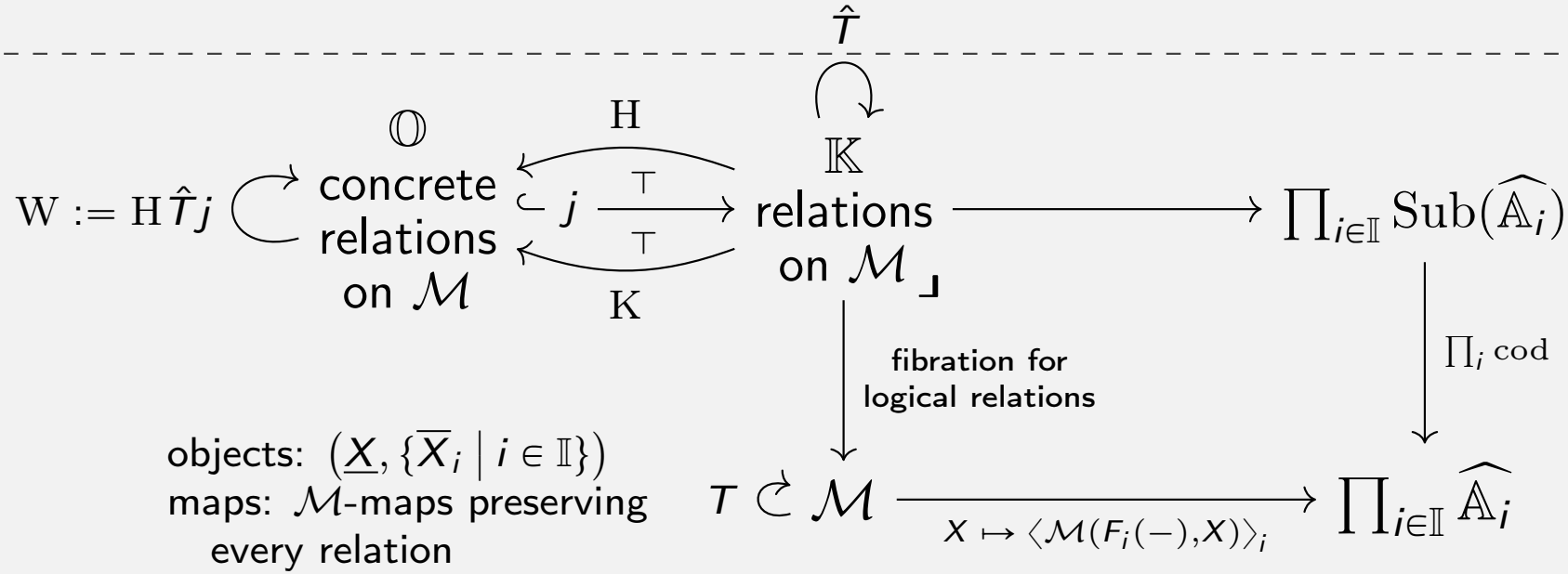
Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up



Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up

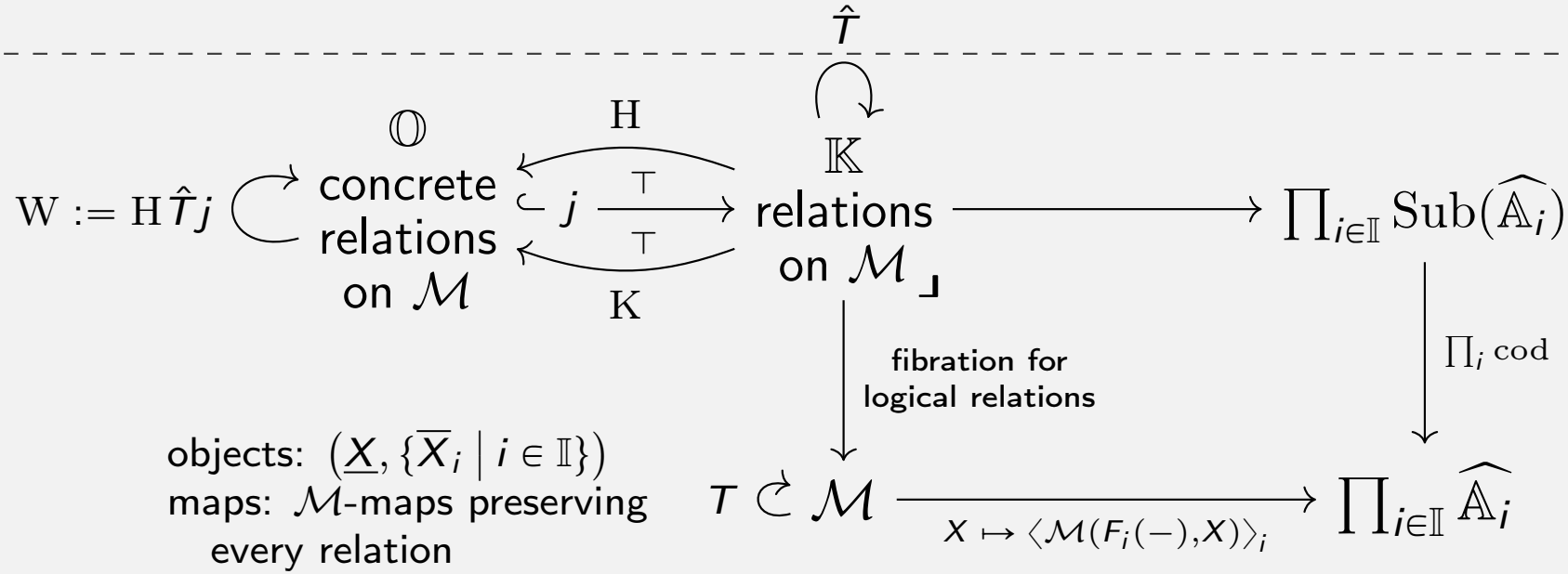


Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up

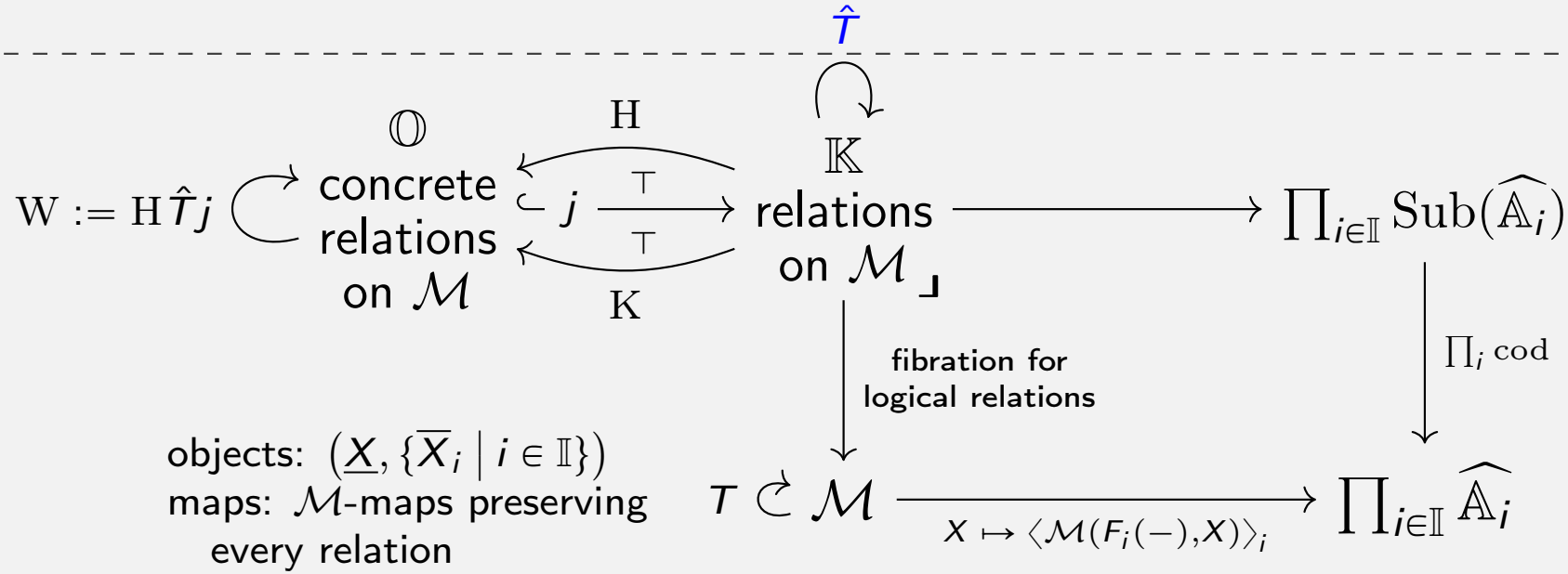


Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up

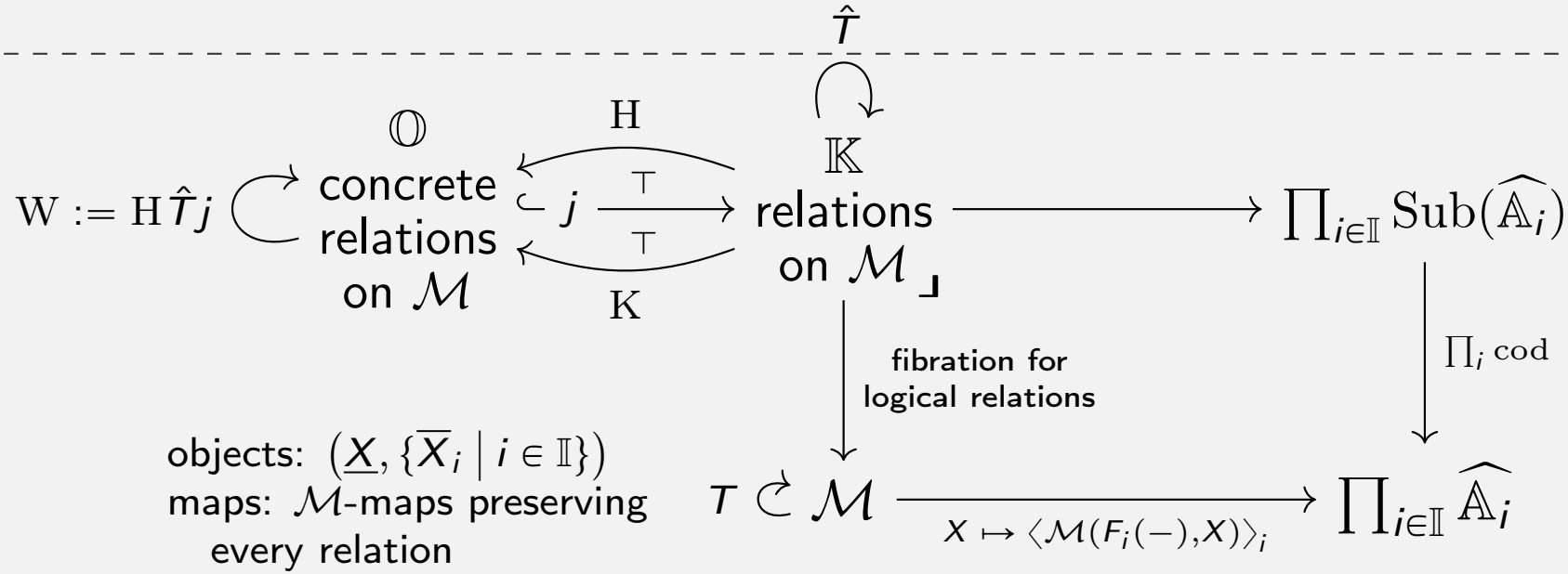


Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$

Tactic:

1. use $\mathbb{I}$ to quantify over all possible relations
   so that DEF must appear
2. define interpretation to look it up



Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$
   $\rightsquigarrow$ to define our semantic interpretation in $\mathbb{O}$

Bake all the data into $\mathbb{I}$

Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$

$$\mathbb{I} \ni ?$$

Bake all the data into $\mathbb{I}$

Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$

---

1. set $\mathrm{Sites} \ni \mathrm{Con}^{\mathrm{op}}$

$$\mathbb{I} \ni (\mathbb{A} \in \mathrm{Sites}, \dots)$$

Bake all the data into $\mathbb{I}$

Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$

---

1. set $\mathrm{Sites} \ni \mathrm{Con}^{\mathrm{op}}$
2. for every $\mathbb{A} \in \mathrm{Sites}$,

$$\mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}],$$

$$\mathbb{I} \ni (\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \dots)$$

Bake all the data into $\mathbb{I}$

Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$

---

1. set $\mathrm{Sites} \ni \mathrm{Con}^{\mathrm{op}}$
2. for every $\mathbb{A} \in \mathrm{Sites}$,
$$\mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}],$$
3. for every $F \in \mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}]$,
$$\mathrm{Lift}(\mathbb{A}, F) = \{\text{monad liftings to } \mathbb{K}_{\mathcal{M},F}\}$$

$$\mathbb{I} \ni \Big(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Fun}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), \dots\Big)$$

Bake all the data into $\mathbb{I}$

Need from every $i \in \mathbb{I}$:

1. category $\mathbb{A}_i$
2. functor $F_i : \mathrm{Con}^{\mathrm{op}} \to \mathcal{M}$
3. a lifting $\hat{T}_i$ of T to $\mathbb{K}_{\mathcal{M},F_i}$
4. an interpretation r in $\mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i}) \hookrightarrow \mathbb{K}_{\mathcal{M},F_i}$

---

1. set $\mathrm{Sites} \ni \mathrm{Con}^{\mathrm{op}}$
2. for every $\mathbb{A} \in \mathrm{Sites}$,
$$\mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}],$$
3. for every $F \in \mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}]$,
$$\mathrm{Lift}(\mathbb{A}, F) = \{\text{monad liftings to } \mathbb{K}_{\mathcal{M},F}\}$$
4. for $\hat{T} \in \mathrm{Lift}(\mathbb{A}, F)$,
$$\mathrm{Interp}(\mathbb{A}, F, \hat{T}) = \{\text{interpretations in } \mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i})\}$$

$$\mathbb{I} \ni \left(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), r \in \mathrm{Interp}(\mathbb{A}, F, \hat{T})\right)$$

Defining the semantic interpretation: just look it up in $\mathbb{I}$!

Defining the semantic interpretation: just look it up in $\mathbb{I}$!

1. set $\mathrm{Sites} \ni \mathrm{Con}^{\mathrm{op}}$
2. for every $\mathbb{A} \in \mathrm{Sites}$,
$$\mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}],$$
3. for every $F \in \mathrm{Func}(\mathbb{A}) = [\mathbb{A}, \mathcal{M}]$,
$$\mathrm{Lift}(\mathbb{A}, F) = \{\text{monad liftings to } \mathbb{K}_{\mathcal{M},F}\}$$
4. for $\hat{T} \in \mathrm{Lift}(\mathbb{A}, F)$,
$$\mathrm{Interp}(\mathbb{A}, F, \hat{T}) = \{\text{interpretations in } \mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i})\}$$

$$\mathbb{I} \ni \left(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), r \in \mathrm{Interp}(\mathbb{A}, F, \hat{T})\right)$$

Defining the semantic interpretation: just look it up in $\mathbb{I}$!

4. for $\hat{T} \in \mathrm{Lift}(\mathbb{A}, F)$,

$$\mathrm{Interp}(\mathbb{A}, F, \hat{T}) = \{\text{interpretations in } \mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i})\}$$

$$\mathbb{I} \ni \Big(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), r \in \mathrm{Interp}(\mathbb{A}, F, \hat{T})\Big)$$

Defining the semantic interpretation: just look it up in $\mathbb{I}$!

4. for $\hat{T} \in \mathrm{Lift}(\mathbb{A}, F)$,
$$\mathrm{Interp}(\mathbb{A}, F, \hat{T}) = \{\text{interpretations in } \mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i})\}$$

$$\mathbb{I} \ni \Big(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), r \in \mathrm{Interp}(\mathbb{A}, F, \hat{T})\Big)$$

carriers: take the interpretation from $\mathcal{M} \rightsquigarrow \underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

Defining the semantic interpretation: just look it up in $\mathbb{I}$!

4. for $\hat{T} \in \mathrm{Lift}(\mathbb{A}, F)$,
$$\mathrm{Interp}(\mathbb{A}, F, \hat{T}) = \{\text{interpretations in } \mathrm{Conc}(\mathbb{K}_{\mathcal{M},F_i})\}$$

$$\mathbb{I} \ni \Big(\mathbb{A} \in \mathrm{Sites}, F \in \mathrm{Func}(\mathbb{A}), \hat{T} \in \mathrm{Lift}(\mathbb{A}, F), r \in \mathrm{Interp}(\mathbb{A}, F, \hat{T})\Big)$$

carriers: take the interpretation from $\mathcal{M} \rightsquigarrow \underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}}$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}}$

Note: $\mathrm{DEF}_\sigma^{\mathrm{val}}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}}$

Note: $\mathrm{DEF}_\sigma^{\mathrm{val}}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big) \subseteq \mathcal{M}\big(U\hat{s}[\![\Gamma]\!], U\mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}}$

Note: $\mathrm{DEF}_\sigma^{\mathrm{val}}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big) \subseteq \mathcal{M}\big(U\hat{s}[\![\Gamma]\!], U\mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking the index i_0

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_\sigma^{\mathrm{val}}$

Note: $\mathrm{DEF}_\sigma^{\mathrm{val}}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big) \subseteq \mathcal{M}\big(U\hat{s}[\![\Gamma]\!], U\mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking the index i_0

4. $r(\beta) := \big(s[\![\beta]\!], \mathrm{DEF}_\beta^{\mathrm{val}}\big)$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma}$

Note: $\mathrm{DEF}^{\mathrm{val}}_{\sigma}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big) \subseteq \mathcal{M}\big(U\hat{s}[\![\Gamma]\!], U\mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking the index i_0

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$

4. $r(\beta) := \big(s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta}\big)$

Picking $i_0 \rightsquigarrow$ look up DEF

Want: $\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma}$

Note: $\mathrm{DEF}^{\mathrm{val}}_{\sigma}(\Gamma) \subseteq \mathbb{O}\big(\hat{s}[\![\Gamma]\!], \mathrm{W}\hat{s}[\![\sigma]\!]\big) \subseteq \mathcal{M}\big(U\hat{s}[\![\Gamma]\!], U\mathrm{W}\hat{s}[\![\sigma]\!]\big)$

Picking the index i_0

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := \big(s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta}\big)$

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$

---

carriers: take the interpretation from $\mathcal{M}$ $\rightsquigarrow$ $\underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$

---

carriers: take the interpretation from $\mathcal{M}$ $\rightsquigarrow$ $\underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

Key lemma

For i_0 as above, and all $\sigma \in \mathrm{Type}$:

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$

---

carriers: take the interpretation from $\mathcal{M}$ $\rightsquigarrow$ $\underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

Key lemma

For i_0 as above, and all $\sigma \in \mathrm{Type}$:

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

Hence every $f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ is definable

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$

---

carriers: take the interpretation from $\mathcal{M}$ $\rightsquigarrow$ $\underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

Key lemma

For i_0 as above, and all $\sigma \in \mathrm{Type}$:

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

Hence every $f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ is definable

Hence $(\mathbb{O}, \mathrm{W}, \hat{s})$ is fully complete and well-pointed

1. $\mathbb{A} := \mathrm{Con}^{\mathrm{op}}$
2. $F := (\mathrm{Con}^{\mathrm{op}} \xrightarrow{\hat{s}[\![-]\!]} \mathbb{O} \xrightarrow{U} \mathcal{M})$
3. $\hat{T}$ chosen by $\top\top$-lifting
4. $r(\beta) := (s[\![\beta]\!], \mathrm{DEF}^{\mathrm{val}}_{\beta})$

---

carriers: take the interpretation from $\mathcal{M}$ $\rightsquigarrow$ $\underline{\hat{s}[\![\beta]\!]} := s[\![\beta]\!]$

relations: use what the index gives:

$$\overline{\hat{s}[\![\beta]\!]}(\mathbb{A}, F, \hat{T}, r) := \big(\text{relation part of } r(\beta)\big)$$

Key lemma

For i_0 as above, and all $\sigma \in \mathrm{Type}$:

$$\overline{\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}^{\mathrm{val}}_{\sigma} \quad \text{and} \quad \overline{\mathrm{W}\hat{s}[\![\sigma]\!]}_{i_0} = \mathrm{DEF}_{\sigma}$$

Hence every $f : \hat{s}[\![\Gamma]\!] \to \mathrm{W}\hat{s}[\![\sigma]\!]$ is definable

Hence $(\mathbb{O}, \mathrm{W}, \hat{s})$ is fully abstract

Summary and future work

Summary and future work

1. Every map definable + well-pointedness
   $\Rightarrow$ full abstraction

Summary and future work

1. Every map definable + well-pointedness
   $\Rightarrow$ full abstraction
2. Definability is a logical relation;
   if f preserves DEF, then f is definable

Summary and future work

1. Every map definable + well-pointedness
   $\Rightarrow$ full abstraction
2. Definability is a logical relation;
   if f preserves DEF, then f is definable
3. Maps in O'Hearn–Riecke model $\mathbb{O}$ preserve
   enough relations

Summary and future work

1. Every map definable + well-pointedness
   $\Rightarrow$ full abstraction
2. Definability is a logical relation;
   if f preserves DEF, then f is definable
3. Maps in O'Hearn–Riecke model $\mathbb{O}$ preserve
   enough relations

Still to do

1. Weakening assumptions: well-pointedness, hull functor H, ...
2. Checking examples: esp. presheaf models (names, ...)
3. Universal property?

Summary and future work

1. Every map definable + well-pointedness
   $\Rightarrow$ full abstraction
2. Definability is a logical relation;
   if f preserves DEF, then f is definable
3. Maps in O'Hearn–Riecke model $\mathbb{O}$ preserve
   enough relations

Still to do

1. Weakening assumptions: well-pointedness, hull functor H, ...
2. Checking examples: esp. presheaf models (names, ...)
3. Universal property?

preprint at cs.ox.ac.uk/people/philip.saville/home.html