Examples

$f : \forall \alpha, \beta . \alpha \times \beta \to \beta \times \alpha$

$\Rightarrow \approx$ "$f(a,b) = (b,a)$"

i.e. $\forall a : 1 \to A, \forall b : 1 \to B$

$\langle a, b \rangle ; f_{A,B} = \langle b, a \rangle$

Proof: $\langle a, b \rangle ; f_{A,B} =$

$\langle id_1, id_1 \rangle ; (a \times b) ; f_{A,B} =$

$1 \to 1 \times 1$

$\langle id_1, id_1 \rangle ; f_{1,1} ; (b \times a) =$

- $1 \simeq 1 \times 1 \to 1 \times 1 \simeq 1$
- $\Rightarrow 1 \times 1$ is final (Exercise!)
- $\Rightarrow f_{1\times 1} = ! = id_{1\times 1}$

$\langle id_1, id_1 \rangle ; (b \times a) =$

$\langle b, a \rangle$

Example (informal)

Suppose we allow the type τ^* for "lists of τ".

τ^* is functorial:

$\forall g : A \to B$

$\Rightarrow g^* : A^* \to B^*$ is pointwise application

$\langle t_1, \ldots \rangle \longmapsto \langle g(t_1), \ldots \rangle$

If we have $f : \forall \alpha.\ \alpha \to \alpha^*$

then $f_{\mathbb{N}}(5)$ can be

$\langle\rangle, \langle 5 \rangle, \langle 5, 5 \rangle, \ldots$

but can NOT be e.g. $\langle 7 \rangle$

Indeed, if we take

$g : \mathbb{N} \to \mathbb{N}$

$g(5) = 5$

$g(7) \neq 7$

then, by naturality / parametricity:

$f_{\mathbb{N}}(g(n)) = g^*(f_{\mathbb{N}}(n))$

hence

$f_{\mathbb{N}}(g(5)) = f_{\mathbb{N}}(5) \overset{\leftarrow}{=} \langle 7 \rangle$ by contrad.

$g^*(f_{\mathbb{N}}(5)) = g^*(\langle 7 \rangle) = \langle g(7) \rangle \neq \langle 7 \rangle$

Note that a "monomorphic"

$f : \mathbb{N} \to \mathbb{N}^*$

can satisfy $f(5) = \langle 7 \rangle$

but polymorphic functions can not do that!

Let $f : \forall \alpha.\ \alpha^* \longrightarrow \alpha^*$

and assume $f_{\mathbb{N}}(\langle 1,2,3\rangle) = \langle 2,3,3,1\rangle$

Then $\forall \tau\ \forall x,y,z : \tau$

$f_\tau(\langle x,y,z\rangle) = \langle y,z,z,x\rangle$

Indeed, let $g : \mathbb{N} \longrightarrow \tau$

$g(1) = x$

$g(2) = y$

$g(3) = z$

by parametricity:

$g^* ; f_\tau = f_{\mathbb{N}} ; g^*$

in particular

$f_\tau(g^*(\langle 1,2,3\rangle)) = g^*(f_{\mathbb{N}}(\langle 1,2,3\rangle))$

$\|$ $\|$

$f_\tau(\langle g(1), \dots\rangle)$ $g^*(\langle 2,3,3,1\rangle)$

$\|$ $\|$

$f_\tau(\langle x,y,z\rangle)$ $\langle g(2), g(3), \dots\rangle$

$\|$

$\langle y,z,z,x\rangle$

Def: A biCCC has set-like coproduct

iff $\forall g : 1 \to A+B$

then, either

$g = \overline{g} ; in_1$ for some $\overline{g} : 1 \to A$

or

$g = \hat{g} ; in_2$ for some $\hat{g} : 1 \to B$

In particular $\forall x : 1 \to 1+1 \cong 2$

we have $x = in_1$ or $x = in_2$

In a parametric model with set-like coproducts,

$$f : \forall \alpha.\ \alpha \times \alpha \longrightarrow \alpha$$

must behave as a projection

$\simeq$ "$f(x,y) = x \ \forall x,y$" or

"$f(x,y) = y \ \forall x,y$"

Formally:

$(\forall x,y : 1 \to A.\ \langle x,y \rangle ; f_A = x)$ OR

$(\ = y)$

Consider $h : 1 \longrightarrow 2$

$$h = \langle in_1, in_2 \rangle ; f_2$$

$$1 \longrightarrow 2 \times 2 \longrightarrow 2$$

$\Rightarrow h = in_1$ OR $h = in_2$

Then:

$\langle x,y \rangle ; f_A = [\beta]$

$\langle in_1 ; [x,y], in_2 ; [x,y] \rangle ; f_A =$

$1 \to 2 \to A \qquad 1 \to 2 \to A$

$\langle in_1, in_2 \rangle ; ([x,y] \times [x,y]) ; f_A =$

$\langle in_1, in_2 \rangle ; f_2 ; [x,y] =$

$h ; [x,y] =$

if $h = in_1$: $in_1 ; [x,y] = [\beta]$, x

if $h = in_2$: $in_2 ; [x,y] = [\beta]$, y

In a paramet. model with set-like +

$$\forall x : 1 \to A+B.\quad x ; \sigma_{A,B} = x ; [in_2, in_1]$$

By cases on x. Assume $x = \bar{x} ; in_1$

(the case $x = \hat{x} ; in_2$ is similar)

$$x = \bar{x} ; in_1 = in_1 ; [\bar{x} ; in_1 , \text{anything}]$$

$$1 \to 1 + \square \to A+B$$

$$\text{e.g.} \quad = in_1 ; [\bar{x} ; in_1 , !_B ; in_2]$$

$$1 \to 1+0 \to A+B$$

$$= in_1 ; (\bar{x} + !_B)$$

Hence:

$$x ; \sigma_{A,B} = \bar{x} ; in_1 ; \sigma_{A,B} =$$

$$in_1 ; (\bar{x} + !_B) ; \sigma_{A,B} =$$

$$in_1 ; \sigma_{1,0} ; (!_B + \bar{x}) =$$

$1 \to 1+0 \to 0+1 \simeq 1 \Rightarrow 0+1$ final

$$! ; (!_B + \bar{x}) =$$

$$in_2 ; (!_B + \bar{x}) =$$

$$in_2 ; [!_B ; in_1 , \bar{x} ; in_2] = [B]$$

$$\bar{x} ; in_2 = [\beta]$$

$$\bar{x} ; (in_1 ; [in_2, in_1]) =$$

$$x ; [in_2, in_1]$$

In real-world programming languages

Ocaml, Haskell, ... "satisfy parametricity"

(except for non termination)

Java, Scala do not

Issues: "instanceof", reflection

(Java) $\forall \alpha. \alpha \to \alpha$

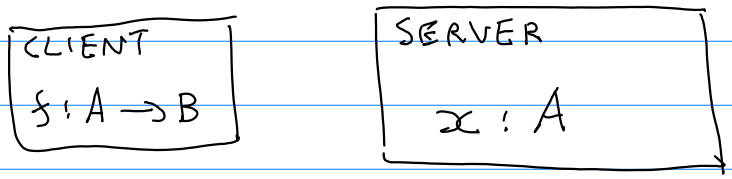
Ex:
```
<A> A foo(A x){
    if (x instanceof String){
        return (A) "hi!";
    } else
    return x;
}
```

violates parametricity (since foo ≠ id).

A gentle introduction to the Yoneda Lemma

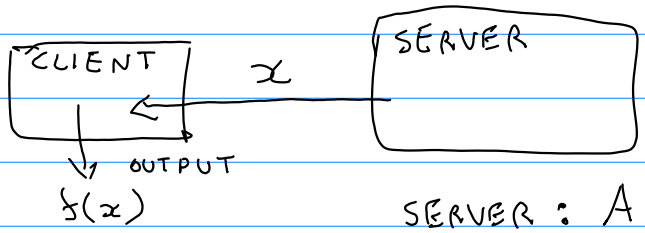
(a particular case)

Consider a network



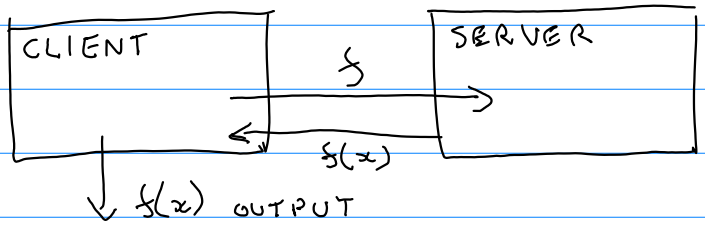
the client wants to compute $f(x)$

Solution 1 : download



SERVER : A

Solution 2 : mobile code



SERVER : $(A \to B) \to B$

the server can be made polymorphic :

SERVER : $\forall B.\ (A \to B) \to B$

Intuition : these two solutions are equivalent

lemma (Yoneda, simplified case)

For any type τ, with $\alpha \notin fv(\tau)$ in parametric models we have

$$\tau \simeq \forall \alpha.\, (\tau \to \alpha) \to \alpha$$

Immediate consequences: the encodings!

$$\begin{aligned}
0 &\simeq \forall \alpha.\, (0 \to \alpha) \to \alpha \\
&\simeq \forall \alpha.\, 1 \to \alpha \\
&\simeq \forall \alpha.\, \alpha \\
1 &\simeq \forall \alpha.\, (1 \to \alpha) \to \alpha \\
&\simeq \forall \alpha.\, \alpha \to \alpha \\
\tau \times \sigma &\simeq \forall \alpha.\, ((\tau \times \sigma) \to \alpha) \to \alpha \\
&\simeq \forall \alpha.\, (\tau \to \sigma \to \alpha) \to \alpha \\
\tau + \sigma &\simeq \forall \alpha.\, ((\tau + \sigma) \to \alpha) \to \alpha \\
&\simeq \forall \alpha.\, ((\tau \to \alpha) \times (\sigma \to \alpha)) \to \alpha \\
&\simeq \forall \alpha.\, (\tau \to \alpha) \to (\sigma \to \alpha) \to \alpha
\end{aligned}$$