Recovering Purity with Comonads & Capabilities

Vikraman Choudhury [1,2] Neel Krishnaswami [2]

[1]Indiana University

[2]University of Cambridge

September 23, 2019

We can extend a pure functional language with monads to encode effects.

We can extend a pure functional language with monads to encode effects.

Can we extend an *impure* functional language to encode the *absence of effects*?

```
map1 : ∀ a b. (a → b) → List a → List b
map1 f []         = []
map1 f (x :: xs) = f x :: map1 f xs
```

```
map1 : ∀ a b. (a → b) → List a → List b
map1 f []        = []
map1 f (x :: xs) = f x :: map1 f xs
```

```
map2 : ∀ a b. (a → b) → List a → List b
map2 f ys =
  let rec loop xs acc =
    match xs with
    | []      → List.reverse acc
    | x :: xs → loop xs (f x :: acc)
  in
  loop ys []
```

```
let xs : List String = ["left "; "to "; "right "]

let f : String → String = fun s → stdout.print(s); s


let ys1 = map1 f xs  -- Prints "right to left " to stdout

let ys2 = map2 f xs  -- Prints "left to right " to stdout
```

```
let xs : List Int = [1; 2; 3]

let f : Int → Int = fun n → stdout.print("a") ; n + 1

let g : Int → Int = fun n → stdout.print("b") ; n + 1


let ys1 = map1 f (map1 g xs) -- Prints "bbbaaa" to stdout

let ys2 = map1 (f ∘ g) xs    -- Prints "bababa" to stdout
```

```
map : ∀ a b. Pure (a → b) → List a → List b
```

$$\mathtt{map} : \forall\ \mathtt{a}\ \mathtt{b}.\ \mathbf{Pure}\ (\mathtt{a} \to \mathtt{b}) \to \mathbf{List}\ \mathtt{a} \to \mathbf{List}\ \mathtt{b}$$

$$\epsilon : \forall\ \mathtt{a}.\ \mathbf{Pure}\ \mathtt{a} \to \mathtt{a}$$
$$\delta : \forall\ \mathtt{a}.\ \mathbf{Pure}\ \mathtt{a} \xrightarrow{\sim} \mathbf{Pure}\ (\mathbf{Pure}\ \mathtt{a})$$

```
map : ∀ a b. Pure (a → b) → List a → List b
```

```
ϵ : ∀ a. Pure a → a
δ : ∀ a. Pure a ⥲ Pure (Pure a)
```

Frank Pfenning, and Rowan Davies.
"A judgmental reconstruction of modal logic."
MSCS (2001)

TYPES	A, B	$::=$	$\mathsf{unit} \mid A \times B \mid A \Rightarrow B \mid \mathsf{str} \mid \mathsf{cap}$
TERMS	e	$::=$	$() \mid (e_1\,, e_2) \mid \mathsf{fst}\, e \mid \mathsf{snd}\, e$
		$\mid$	$x \mid \lambda x : A.\, e \mid e_1\, e_2$
		$\mid$	$s \mid e_1 \cdot \mathsf{print}(e_2)$
VALUES	v	$::=$	$x \mid () \mid (v_1\,, v_2) \mid \lambda x : A.\, e \mid s$
CONTEXTS	Γ, Δ, Ψ	$::=$	$\cdot \mid \Gamma, x : A$
SUBSTITUTIONS	θ, ϕ	$::=$	$\langle\rangle \mid \langle \theta, v/x \rangle$

$$
\begin{array}{llcl}
\textsc{Types} & A, B & ::= & \mathsf{unit} \mid A \times B \mid A \Rightarrow B \mid \mathsf{str} \mid \mathsf{cap} \mid \Box A \\
\textsc{Terms} & e & ::= & () \mid (e_1\,, e_2) \mid \mathsf{fst}\, e \mid \mathsf{snd}\, e \\
 & & \mid & x \mid \lambda x : A.\, e \mid e_1\, e_2 \\
 & & \mid & s \mid e_1 \cdot \mathsf{print}(e_2) \\
 & & \mid & \mathsf{box}\, \boxed{e} \mid \mathsf{let}\, \mathsf{box}\, \boxed{x} = e_1\, \mathsf{in}\, e_2 \\
\textsc{Values} & v & ::= & x \mid () \mid (v_1\,, v_2) \mid \lambda x : A.\, e \mid s \mid \mathsf{box}\, \boxed{e} \\
\textsc{Qualifiers} & q, r & ::= & \circ \mid \bullet \\
\textsc{Contexts} & \Gamma, \Delta, \Psi & ::= & \cdot \mid \Gamma, x : A^q \\
\textsc{Substitutions} & \theta, \phi & ::= & \langle\rangle \mid \langle \theta, e^q / x \rangle
\end{array}
$$

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A}\ \textsc{Var} \qquad \frac{\Gamma, x : A \vdash e : B}{\Gamma \vdash \lambda x : A.\, e : A \Rightarrow B}\ \Rightarrow \mathrm{I}$$

$$\frac{\Gamma \vdash e_1 : A \Rightarrow B \qquad \Gamma \vdash e_2 : A}{\Gamma \vdash e_1\, e_2 : B}\ \Rightarrow \mathrm{E}$$

$$\frac{\Gamma \vdash e_1 : \mathsf{cap} \qquad \Gamma \vdash e_2 : \mathsf{str}}{\Gamma \vdash e_1 \cdot \mathsf{print}(e_2) : \mathsf{unit}}\ \textsc{Print}$$

$$\frac{x : A^{q} \in \Gamma}{\Gamma \vdash x : A}\ \textsc{Var} \qquad \frac{\Gamma, x : A^{\bullet} \vdash e : B}{\Gamma \vdash \lambda x : A.\, e : A \Rightarrow B}\ \Rightarrow \mathrm{I}$$

$$\frac{\Gamma \vdash e_1 : A \Rightarrow B \qquad \Gamma \vdash e_2 : A}{\Gamma \vdash e_1\, e_2 : B}\ \Rightarrow \mathrm{E}$$

$$\frac{\Gamma \vdash e_1 : \mathsf{cap} \qquad \Gamma \vdash e_2 : \mathsf{str}}{\Gamma \vdash e_1 \cdot \mathsf{print}(e_2) : \mathsf{unit}}\ \textsc{Print}$$

$$\frac{\Gamma^{\circ} \vdash e : A}{\Gamma \vdash \mathsf{box}\ \boxed{e} : \square A}\ \square\mathrm{I} \qquad \frac{\Gamma \vdash e_1 : \square A \qquad \Gamma, x : A^{\circ} \vdash e_2 : B}{\Gamma \vdash \mathsf{let}\ \mathsf{box}\ \boxed{x} = e_1\ \mathsf{in}\ e_2 : B}\ \square\mathrm{E}$$

$$\begin{aligned} (\cdot)^{\circ} &:= \cdot \\ (\Gamma, x : A^{\circ})^{\circ} &:= \Gamma^{\circ}, x : A^{\circ} \\ (\Gamma, x : A^{\bullet})^{\circ} &:= \Gamma^{\circ} \end{aligned}$$

$$\frac{\Gamma, x : A \vdash e : B \qquad \Gamma \vdash v : A}{\Gamma \vdash (\lambda x : A.\, e)\, v \approx [v/x]e : B} \Rightarrow \beta$$

$$\frac{\Gamma \vdash e : A \Rightarrow B}{\Gamma \vdash e \approx \lambda x : A.\, e\, x : A \Rightarrow B} \Rightarrow \eta$$

$$\frac{\Gamma, x : A^{\bullet} \vdash e : B \qquad \Gamma \vdash v : A}{\Gamma \vdash (\lambda x : A.\, e)\, v \approx [v/x]e : B} \Rightarrow \beta$$

$$\frac{\Gamma \vdash v : A \Rightarrow B}{\Gamma \vdash v \approx \lambda x : A.\, v\, x : A \Rightarrow B} \Rightarrow \eta\text{-IMPURE}$$

$$\frac{\Gamma \vdash^{\circ} e : A \Rightarrow B}{\Gamma \vdash e \approx \lambda x : A.\, e\, x : A \Rightarrow B} \Rightarrow \eta\text{-PURE}$$

$$\frac{\Gamma^{\circ} \vdash e_1 : A \qquad \Gamma, x : A^{\circ} \vdash e_2 : B}{\Gamma \vdash \mathsf{let\,box}\,\boxed{x} = \mathsf{box}\,\boxed{e_1}\,\mathsf{in}\,e_2 \approx [e_1/x]e_2 : B}\ \square\beta$$

$$\frac{\Gamma \vdash \mathcal{E}\langle\!\langle e \rangle\!\rangle : B \qquad \begin{array}{c}\Gamma \vdash e : \square A \\ \Gamma \vdash \mathsf{let\,box}\,\boxed{x} = e\,\mathsf{in}\,\mathcal{E}\langle\!\langle \mathsf{box}\,\boxed{x} \rangle\!\rangle : B\end{array}}{\Gamma \vdash \mathcal{E}\langle\!\langle e \rangle\!\rangle \approx \mathsf{let\,box}\,\boxed{x} = e\,\mathsf{in}\,\mathcal{E}\langle\!\langle \mathsf{box}\,\boxed{x} \rangle\!\rangle : B}\ \square\eta\text{-IMPURE}$$

$$\frac{\Gamma \vdash \mathcal{C}\langle\!\langle e \rangle\!\rangle : B \qquad \begin{array}{c}\Gamma \vdash^{\circ} e : \square A \\ \Gamma \vdash \mathsf{let\,box}\,\boxed{x} = e\,\mathsf{in}\,\mathcal{C}\langle\!\langle \mathsf{box}\,\boxed{x} \rangle\!\rangle : B\end{array}}{\Gamma \vdash \mathcal{C}\langle\!\langle e \rangle\!\rangle \approx \mathsf{let\,box}\,\boxed{x} = e\,\mathsf{in}\,\mathcal{C}\langle\!\langle \mathsf{box}\,\boxed{x} \rangle\!\rangle : B}\ \square\eta\text{-PURE}$$

$\mathcal{C}$: Cat

Let $\mathcal{C}$ be a fixed set of capabilities.
A capability space X is a set $|X|$, with a weight function w_X.
$\mathcal{C}$ is the category of capability spaces.

$$\begin{aligned}
\mathrm{Obj}_{\mathcal{C}} &:= (|X| : \mathrm{Set},\ w_X : |X| \to \wp(\mathcal{C})) \\
\mathrm{Hom}_{\mathcal{C}}(X, Y) &:= \left\{ f \in |X| \to |Y| \;\middle|\; \begin{array}{l} \forall x \in |X|, \\ w_Y(f(x)) \subseteq w_X(x) \end{array} \right\}
\end{aligned}$$

$\mathscr{C}$: Cat

Let $\mathcal{C}$ be a fixed set of capabilities.
A capability space X is a set $|X|$, with a weight function w_X.
$\mathscr{C}$ is the category of capability spaces.

$$\begin{aligned} \mathrm{Obj}_{\mathscr{C}} &:= (|X| : \mathrm{Set},\ w_X : |X| \to \wp(\mathcal{C})) \\ \mathrm{Hom}_{\mathscr{C}}(X, Y) &:= \left\{ f \in |X| \to |Y| \;\middle|\; \begin{array}{l} \forall x \in |X|, \\ w_Y(f(x)) \subseteq w_X(x) \end{array} \right\} \end{aligned}$$

$\mathscr{C}$ is a cartesian closed category!

$\square : \mathcal{C} \to \mathcal{C}$

$$
\begin{aligned}
|\square A| &:= \left\{ a \in |A| \;\middle|\; w_A(a) = \emptyset \right\} \\
w_{\square A}(a) &:= w_A(a) = \emptyset
\end{aligned}
$$

$\square$ is a strong monoidal idempotent comonad.

$\square : \mathcal{C} \to \mathcal{C}$

$$
\begin{aligned}
|\square A| &:= \left\{ a \in |A| \,\middle|\, w_A(a) = \emptyset \right\} \\
w_{\square A}(a) &:= w_A(a) = \emptyset
\end{aligned}
$$

$\square$ is a strong monoidal idempotent comonad.

$T : \mathcal{C} \to \mathcal{C}$

$$
\begin{aligned}
|T(A)| &= |A| \times (\mathcal{C} \to \Sigma^*) \\
w_{T(A)}(a, o) &= w_A(a) \cup \left\{ c \in \mathcal{C} \,\middle|\, o(c) \neq \varepsilon \right\}
\end{aligned}
$$

T is a strong monad.

$\square : \mathcal{C} \to \mathcal{C}$

$$|\square A| \quad := \quad \left\{ a \in |A| \;\middle|\; w_A(a) = \emptyset \right\}$$
$$w_{\square A}(a) \quad := \quad w_A(a) = \emptyset$$

$\square$ is a strong monoidal idempotent comonad.

$T : \mathcal{C} \to \mathcal{C}$

$$|T(A)| \quad = \quad |A| \times (\mathcal{C} \to \Sigma^*)$$
$$w_{T(A)}(a, o) \quad = \quad w_A(a) \cup \left\{ c \in \mathcal{C} \;\middle|\; o(c) \neq \varepsilon \right\}$$

T is a strong monad.

$\square$ *cancels* T

$$\phi_A : \square T A \xrightarrow{\sim} \square A$$

$[\![A]\!] : \mathrm{Obj}_{\mathcal{C}}$

$$
\begin{aligned}
[\![\top]\!] &:= 1 \\
[\![A \times B]\!] &:= [\![A]\!] \times [\![B]\!] \\
[\![A \Rightarrow B]\!] &:= [\![A]\!] \to T[\![B]\!] \\
[\![\Box A]\!] &:= \Box [\![A]\!]
\end{aligned}
$$

$[\![A]\!] : \mathrm{Obj}_{\mathcal{C}}$

$$
\begin{aligned}
[\![\top]\!] &:= 1 \\
[\![A \times B]\!] &:= [\![A]\!] \times [\![B]\!] \\
[\![A \Rightarrow B]\!] &:= [\![A]\!] \to T[\![B]\!] \\
[\![\Box A]\!] &:= \Box[\![A]\!]
\end{aligned}
$$

$[\![\Gamma \vdash e : A]\!] : \mathrm{Hom}_{\mathcal{C}}([\![\Gamma]\!], T[\![A]\!])$

$$
\left[\!\!\left[\frac{\Gamma, x : A^{\bullet} \vdash e : B}{\Gamma \vdash \lambda x : A.\, e : A \Rightarrow B} \right]\!\!\right] := \mathsf{curry}\,([\![\Gamma, x : A^{\bullet} \vdash e : B]\!]) \,;\, \eta_{A \to TB}
$$

$$
\left[\!\!\left[\frac{\Gamma \vdash e : A \times B}{\Gamma \vdash \mathsf{fst}\, e : A} \right]\!\!\right] := [\![\Gamma \vdash e : A \times B]\!] \,;\, T\pi_1
$$

$$\ldots$$

Syntactic Substitution

If $\Gamma \vdash \theta : \Delta$ and $\Delta \vdash e : A$, then $\Gamma \vdash \theta(e) : A$.

Semantic Substitution

If $\Gamma \vdash \theta : \Delta$ and $\Delta \vdash e : A$, then

$$[\![\Gamma \vdash \theta(e) : A]\!] = [\![\Gamma \vdash \theta : \Delta]\!] \,;\, [\![\Delta \vdash e : A]\!].$$

Soundness

If $\Gamma \vdash e_1 \approx e_2 : A$, then $[\![\Gamma \vdash e_1 : A]\!] = [\![\Gamma \vdash e_2 : A]\!]$.

TYPES	unit	:=	unit
	$A \Rightarrow B$	:=	$\square\, A \Rightarrow B$
CONTEXTS	$\cdot$	:=	$\cdot$
	$\Gamma, x : A$	:=	$\Gamma, x : A^{\circ}$
TERMS	$()$	:=	$()$
	x	:=	x
	$\lambda x : A.\, e$	:=	$\lambda z : \square\, A.\ \text{let box}\ \boxed{x} = z\ \text{in}\ e$
	$e_1\, e_2$	:=	$e_1\ \text{box}\ \boxed{e_2}$

Type preserving

If $\Gamma \vdash_\lambda e : A$, then $\Gamma \vdash e : A$.

Equality preserving

If $\Gamma \vdash_\lambda e_1 \approx e_2 : A$, then $\Gamma \vdash e_1 \approx e_2 : A$.

Conservative extension

If $\Gamma \vdash_\lambda e_1 : A$ and $\Gamma \vdash_\lambda e_1 : A$ and $\Gamma \vdash e_1 \approx e_2 : A$,

then $\Gamma \vdash_\lambda e_1 \approx e_2 : A$.

```
map : ∀ a b. Pure (a → b) → List a → List b
map (box f) []        = []
map (box f) (x :: xs) = f x :: map (box f) xs
```

Recovering Purity with Comonads & Capabilities

`https://arxiv.org/abs/1907.07283`[1]

Thank you!

[1] Disclaimer: There is a bug in definition 4.15 on page 13