Monadic reflection in Lax Logic

Tristan Crolard

LACL – University of East Paris

TPDC'11
Novi Sad – May 29, 2011

Introduction

- Revisit "*Representing monads*" [Filinski, 1994] from a logical standpoint.
- **Goal:** understand the logical meaning of **shift**/**reset** in the restricted framework of a major application, i.e. implementing monadic reflection.
- Through the formulas-as-types interpretation, a monad $\Diamond_{\sqcup}$ corresponds to the modality from Lax logic [Curry, 1952]:

$$\vdash \mathit{unit} : \varphi \Rightarrow \Diamond\varphi \qquad\qquad \vdash \mathit{bind} : (\varphi \Rightarrow \Diamond\psi) \Rightarrow \Diamond\varphi \Rightarrow \Diamond\psi$$

- Monadic reflection is given by these logical rules:

$$\frac{\Gamma \vdash\ t : \varphi}{\Gamma \vdash\ [t] : \Diamond\varphi}\ [\text{reify}] \qquad \frac{\Gamma \vdash\ t : \Diamond\varphi}{\Gamma \vdash\ \mu(t) : \varphi}\ [\text{reflect}]$$

- In this talk, we consider only *provability*.

Example: the exception monad

$$\varepsilon \quad : \quad \mathit{type}.$$
$$\diamondsuit\varphi \quad = \quad \varphi \vee \varepsilon.$$

$$\mathit{unit} \quad = \quad \lambda a\colon \varphi.\mathbf{inl}\, a.$$
$$\mathit{bind}\, f \quad = \quad \lambda t.\mathbf{case}\, t\, \mathbf{of}\ (\mathbf{inl}\, a) \mapsto f\ a \mid (\mathbf{inr}\, b) \mapsto \mathbf{inr}\, b.$$

Defining **raise** and **handle** in direct style:

$$\mathbf{raise} \quad = \quad \mu(\mathbf{inr}\, t).$$

$$t\, \mathbf{handle}\, e \mapsto h \quad = \quad \mathbf{case}\ [t]\ \mathbf{of}\ (\mathbf{inl}\, a) \mapsto a \mid (\mathbf{inr}\, e) \mapsto h.$$

Moggi's monadic translation (CBV)

Translation of types (σ atomic):

- $\sigma^{\Diamond} \equiv \sigma$
- $(\varphi \Rightarrow \psi)^{\Diamond} \equiv \varphi^{\Diamond} \Rightarrow \Diamond\psi^{\Diamond}$
- $(\Diamond\psi)^{\Diamond} \equiv \Diamond\varphi^{\Diamond}$

Translation of terms:

- $x^{\Diamond} \equiv unit\ x$
- $(\lambda x.t)^{\Diamond} \equiv unit\ \lambda x.t^{\Diamond}$
- $(t_1\ t_2)^{\Diamond} \equiv bind\,(\lambda f.(bind\ f\ t_2^{\Diamond}))\ t_1^{\Diamond}$
- $\mu(t)^{\Diamond} \equiv bind\ id\ t^{\Diamond}$
- $[\![t]\!]^{\Diamond} \equiv unit\ t^{\Diamond}$

Lemma. *If* $\Gamma \vdash t\colon \varphi$ *is derivable then* $\Gamma^{\Diamond} \vdash t^{\Diamond}\colon \Diamond\varphi^{\Diamond}$ *is derivable.*

Proof.

$$\frac{\Gamma^{\Diamond} \vdash t^{\Diamond} : \Diamond\varphi^{\Diamond}}{\Gamma^{\Diamond} \vdash unit\ t^{\Diamond} : \Diamond\Diamond\varphi^{\Diamond}}\ [\text{unit}] \qquad \frac{\Gamma^{\Diamond} \vdash t^{\Diamond} : \Diamond\Diamond\varphi^{\Diamond}}{\Gamma^{\Diamond} \vdash bind\ id\ t^{\Diamond} : \Diamond\varphi^{\Diamond}}\ [\text{join}]$$

Filinski's CPS-translation (CBV)

Define $\nabla\varphi = (\varphi \Rightarrow \Diamond o) \Rightarrow \Diamond o$ where o is some universal answer type (not sound).

Translation of types (σ atomic):

- $\sigma^{\nabla} \equiv \sigma$
- $(\varphi \Rightarrow \psi)^{\nabla} \equiv \varphi^{\nabla} \Rightarrow \nabla\psi^{\nabla}$
- $(\Diamond\psi)^{\nabla} \equiv \Diamond\varphi^{\nabla}$

Translation of terms:

- $x^{\nabla} \equiv \lambda k.k\ x$
- $(\lambda x.t)^{\nabla} \equiv \lambda k.k\ (\lambda x.t^{\nabla})$
- $(t_1\ t_2)^{\nabla} \equiv \lambda k.t_1^{\nabla}\ (\lambda f.t_2^{\nabla}\ (\lambda a.f\ a\ k))$
- $\mu(t)^{\nabla} \equiv \lambda k.t^{\nabla}\ (bind\, k)$
- $[t]^{\nabla} \equiv \lambda k.k\ (t^{\nabla}\ unit)$

Delimited control

Reflect/reify are *definable in direct style* from **shift**/**reset** [Filinski, 1994]

$$\begin{array}{rcl} [\![t]\!] & = & \textbf{reset}\ (\mathit{unit}\ t). \\ \mu(t) & = & \textbf{shift}\ (\lambda k.\mathit{bind}\ k\ t). \end{array}$$

That is, these equations are valid:

- $(\textbf{reset}\ (\mathit{unit}\ t))^{\nabla} = [\![t]\!]^{\nabla}$
- $(\textbf{shift}\ (\lambda k.\mathit{bind}\ k\ t))^{\nabla} = \mu(t)^{\nabla}$

where:

- $\textbf{reset}^{\nabla} = \lambda m.\lambda c.(c\,(m\ \mathit{id}))$
- $\textbf{shift}^{\nabla} = \lambda h.\lambda c.(h\,(\lambda v.\lambda c'.c'\,(c\,v))\ \mathit{id})$

Answer type polymorphism

Footnote from "*Representing monads*":

> "Alternatively, with a little more care, we can take $\nabla\varphi = \forall\alpha(\varphi \to \Diamond\alpha) \to \Diamond\alpha$; it is straightforward to check that both the term translation and the operations defined in the following can in fact be typed according to this schema."

So, let us do it *carefully*:

- Formalization in Twelf (work in progress)
- Experimenting with TeXmacs as a front end
  (the Twelf source is generated from the slides)

Plan of the rest of the talk

- Formalize System F in Twelf
- Check that the operations are well-typed in direct style
- Check that the CPS-translations of the operations are well-typed
- Interpret the logical type of **shift** for the usual monads:
  - continuation monad
  - state monad
  - exception monad

System F (HOAS)

Types

$$type \quad : \quad \textbf{type}.$$
$$\sqcup \Rightarrow \sqcup \quad : \quad type \rightarrow type \rightarrow type.$$
$$\sqcup \wedge \sqcup \quad : \quad type \rightarrow type \rightarrow type.$$
$$\sqcup \vee \sqcup \quad : \quad type \rightarrow type \rightarrow type.$$

$$\forall_{\sqcup . \sqcup} \quad : \quad (type \rightarrow type) \rightarrow type.$$
$$\textbf{binding} \quad 1 \mapsto 2 \quad \textbf{in} \quad \forall_{\sqcup . \sqcup}$$

$$\textbf{void} \quad = \quad \forall \beta . \beta.$$

Terms

$term \quad : \quad \textbf{type}.$

Abstraction

$\lambda_{\sqcup} : _{\sqcup} . _{\sqcup} \quad : \quad type \rightarrow (term \rightarrow term) \rightarrow term.$

$\textbf{binding} \quad 1 \mapsto 3 \quad \textbf{in} \quad \lambda_{\sqcup} : _{\sqcup} . _{\sqcup}$

Application

$_{\sqcup} \ _{\sqcup} \quad : \quad term \rightarrow term \rightarrow term.$

Polymorphic abstraction

$\Lambda_{\sqcup} . _{\sqcup} \quad : \quad (type \rightarrow term) \rightarrow term.$

$\textbf{binding} \quad 1 \mapsto 2 \quad \textbf{in} \quad \Lambda_{\sqcup} . _{\sqcup}$

Instantiation

$$\_\{\_\} \quad : \quad term \;\to\; type \;\to\; term.$$

Derived syntax for **let**

$$\textbf{let}\ \_ : \_ = \_\ \textbf{in}\ \_ \quad = \quad [\tau]\,[u]\,[t]\,(\lambda x{:}\tau.t[x])\ u.$$

binding $1 \mapsto 4$ **in** **let** $\_ : \_ = \_$ **in** $\_$

Pairing

$$\langle \_, \_ \rangle \quad : \quad term \;\to\; term \;\to\; term.$$

Pattern matching

$$\textbf{let}\ \langle \_, \_ \rangle = \_\ \textbf{in}\ \_ \quad : \quad term \;\to\; (term \;\to\; term \;\to\; term) \;\to\; term.$$

binding $2 \mapsto 4$ **in** **let** $\langle \_, \_ \rangle = \_$ **in** $\_$
binding $1 \mapsto 4$ **in** **let** $\langle \_, \_ \rangle = \_$ **in** $\_$

Injections

$\mathbf{inl}\ \sqcup \quad : \quad term \ \to \ term.$
$\mathbf{inr}\ \sqcup \quad : \quad term \ \to \ term.$

Pattern matching

$\mathbf{case}\ \sqcup\ \mathbf{of}\ (\mathbf{inl}\ \sqcup) \mapsto \sqcup \mid (\mathbf{inr}\ \sqcup) \mapsto \sqcup \quad : \quad term \ \to \ (term \ \to \ term) \ \to \ (term \ \to \ term) \ \to \ term.$

$\mathbf{binding} \quad 4 \mapsto 5 \quad \mathbf{in} \quad \mathbf{case}\ \sqcup\ \mathbf{of}\ (\mathbf{inl}\ \sqcup) \mapsto \sqcup \mid (\mathbf{inr}\ \sqcup) \mapsto \sqcup$
$\mathbf{binding} \quad 2 \mapsto 3 \quad \mathbf{in} \quad \mathbf{case}\ \sqcup\ \mathbf{of}\ (\mathbf{inl}\ \sqcup) \mapsto \sqcup \mid (\mathbf{inr}\ \sqcup) \mapsto \sqcup$

Monadic constants

$unit \quad : \quad term.$
$bind \quad : \quad term.$

Delimited control operators

$\mathbf{reset} \quad : \quad term.$
$\mathbf{shift} \quad : \quad term.$

Typing judgment

$\vdash \sqcup : \sqcup \quad : \quad \mathit{term} \rightarrow \mathit{type} \rightarrow \mathbf{type}.$

$$\frac{\{x\} \vdash x : \varphi \rightarrow \vdash t[x] : \psi}{\vdash \lambda x{:}\,\varphi.t[x] : \varphi \Rightarrow \psi}\ [\text{lam}]$$

$$\frac{\vdash t_1 : \varphi \Rightarrow \psi \quad \vdash t_2 : \varphi}{\vdash t_1\ t_2 : \psi}\ [\text{app}]$$

$$\frac{\{\alpha\} \vdash t[\alpha] : \psi[\alpha]}{\vdash \Lambda\alpha.t[\alpha] : \forall\alpha.\psi[\alpha]}\ [\text{abs}]$$

$$\frac{\vdash t : \forall\alpha.\psi[\alpha]}{\vdash t\{\varphi\} : \psi[\varphi]}\ [\text{inst}]$$

Typing judgment

$$\frac{\vdash t_1 : \varphi \quad \vdash t_2 : \psi}{\vdash \langle t_1, t_2 \rangle : \varphi \wedge \psi}\ [\text{pair}]$$

$$\frac{\{x\} \vdash x : \varphi \ \rightarrow\ (\{y\} \vdash y : \psi \ \rightarrow\ \vdash u[x][y] : \tau) \quad \vdash t : \varphi \wedge \psi}{\vdash \mathbf{let}\ \langle x, y \rangle = t\ \mathbf{in}\ u[x][y] : \tau}\ [\text{match}]$$

$$\frac{\vdash t : \psi}{\vdash \mathbf{inr}\, t : \varphi \vee \psi}\ [\text{inr}]$$

$$\frac{\vdash t : \varphi}{\vdash \mathbf{inl}\, t : \varphi \vee \psi}\ [\text{inl}]$$

$$\frac{\{x\} \vdash x : \varphi \ \rightarrow\ \vdash u_1[x] : \phi \quad \{y\} \vdash y : \psi \ \rightarrow\ \vdash u_2[y] : \phi \quad \vdash t : \varphi \vee \psi}{\vdash \mathbf{case}\, t\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u_1[x] \mid (\mathbf{inr}\, y) \mapsto u_2[y] : \phi}\ [\text{case}]$$

Lax logic

Primitive monad or lax modality [Curry, 1952]

$\Diamond_{\sqcup} \quad : \quad type \ \rightarrow \ type.$

$$\vdash unit : \varphi \Rightarrow \Diamond\varphi \ ^{[\text{unit}]}$$

$$\vdash bind : (\varphi \Rightarrow \Diamond\psi) \Rightarrow \Diamond\varphi \Rightarrow \Diamond\psi \ ^{[\text{bind}]}$$

Delimited control

Fixed answer type

$o \quad : \quad \mathit{type}.$

$$\vdash \mathbf{reset} : \Diamond\varphi \Rightarrow \Diamond\varphi \ ^{[\mathrm{reset}]}$$

$$\vdash \mathbf{shift} : ((\varphi \Rightarrow \Diamond o) \Rightarrow \Diamond o) \Rightarrow \varphi \ ^{[\mathrm{shift}]}$$

Monadic reflection

Reflect/reify are definable from **shift**/**reset**:

$$\begin{array}{rcl} [t] & = & \textbf{reset}\ (\mathit{unit}\ t). \\ \mu(t) & = & \textbf{shift}\ (\lambda k{:}\,\varphi \Rightarrow \Diamond o.\mathit{bind}\ k\ t). \end{array}$$

Lemma. *The following typing rules are derivable:*

$$\frac{\vdash t : \varphi}{\vdash [t] : \Diamond\varphi}\ [\text{reify}] \qquad \frac{\vdash t : \Diamond\varphi}{\vdash \mu(t) : \varphi}\ [\text{reflect}]$$

%solve $\vdash t : \varphi \ \rightarrow\ \vdash [t] : \Diamond\varphi$
%solve $\vdash t : \Diamond\varphi \ \rightarrow\ \vdash \mu(t) : \varphi$

Polymorphic monadic reflection

Polymorphic type for **shift**:

$$\vdash \mathbf{shift} : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha) \Rightarrow \varphi \ ^{[\text{shift}]}$$

Reflect is still definable from **shift**:

$$\mu(t) \quad = \quad \mathbf{shift}\ (\Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t).$$

Lemma. *The following typing rule is derivable:*

$$\frac{\vdash t : \Diamond\varphi}{\vdash \mu(t) : \varphi}\ ^{[\text{reflect}]}$$

%solve $\vdash t : \Diamond\varphi \ \rightarrow\ \vdash \mu(t) : \varphi$

Remark. $\forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha)$ is equivalent to $\Diamond\varphi$.

%solve $\vdash t : \Diamond\varphi \ \rightarrow\ \vdash \Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha)$
%solve $\vdash f : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha) \ \rightarrow\ \vdash (f\,\{\varphi\}\ \mathit{unit}) : \Diamond\varphi$

Derived typing rule for reify

lemma $\vdash t : \varphi \implies \vdash [t] : \Diamond\varphi$

Proof.

$$\dfrac{\mathcal{D}_{\text{of}_1}}{\vdash t : \varphi} \implies \dfrac{\dfrac{}{\vdash \textbf{reset} : \Diamond\varphi \Rightarrow \Diamond\varphi}\,[\text{reset}] \qquad \dfrac{\dfrac{}{\vdash unit : \varphi \Rightarrow \Diamond\varphi}\,[\text{unit}] \qquad \dfrac{\mathcal{D}_{\text{of}_1}}{\vdash t : \varphi}}{\vdash unit\ t : \Diamond\varphi}\,[\text{app}]}{\vdash \textbf{reset}\ (unit\ t) : \Diamond\varphi}\,[\text{app}]\ [\&]$$

%mode $+\mathcal{D}_{\text{of}_1} \implies -\mathcal{D}_{\text{of}_2}$
%worlds $()\ \mathcal{D}_{\text{of}_1} \implies \mathcal{D}_{\text{of}_2}$
%total $\{\}\ \mathcal{D}_{\text{of}_1} \implies \mathcal{D}_{\text{of}_2}$

Derived typing rule for reflect

lemma $\vdash t : \Diamond\varphi \quad \Longrightarrow \quad \vdash \mu(t) : \varphi$

Proof.

$$\begin{array}{c}\mathcal{D}_{\mathrm{of}_t} \\ \vdash t : \Diamond\varphi\end{array} \quad \Longrightarrow \quad {\scriptstyle[\text{shift}]}\ \dfrac{[\alpha]\ \dfrac{\dfrac{[k]\ \dfrac{[\mathcal{D}_{\mathrm{of}_k}]\ \dfrac{{\scriptstyle[\text{bind}]}\ \dfrac{\begin{array}{c}\mathcal{D}_{\mathrm{of}_k} \\ \vdash k : \varphi \Rightarrow \Diamond\alpha\end{array}}{\vdash \mathit{bind}\ k : \Diamond\varphi \Rightarrow \Diamond\alpha}\ {\scriptstyle[\text{app}]} \qquad \begin{array}{c}\mathcal{D}_{\mathrm{of}_t} \\ \vdash t : \Diamond\varphi\end{array}}{\vdash \mathit{bind}\ k\ t : \Diamond\alpha}\ {\scriptstyle[\text{app}]}}{\vdash k : \varphi \Rightarrow \Diamond\alpha \ \rightarrow\ \vdash \mathit{bind}\ k\ t : \Diamond\alpha}}{\{k\}\ \vdash k : \varphi \Rightarrow \Diamond\alpha \ \rightarrow\ \vdash \mathit{bind}\ k\ t : \Diamond\alpha}}{\vdash \lambda k{:}\, \varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t : (\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha}\ {\scriptstyle[\text{lam}]}}{\vdash \Lambda\alpha.\lambda k{:}\, \varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha)}\ {\scriptstyle[\text{abs}]}$$

$$\dfrac{\vdash \Lambda\alpha.\lambda k{:}\, \varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha)}{\vdash \mathbf{shift}\ (\Lambda\alpha.\lambda k{:}\, \varphi \Rightarrow \Diamond\alpha.\mathit{bind}\ k\ t) : \varphi}\ {\scriptstyle[\text{app}]}$$

%mode $+\mathcal{D}_{\mathrm{of}_1} \quad \Longrightarrow \quad -\mathcal{D}_{\mathrm{of}_2}$
%worlds $()\quad \mathcal{D}_{\mathrm{of}_1} \quad \Longrightarrow \quad \mathcal{D}_{\mathrm{of}_2}$
%total $\{\}\quad \mathcal{D}_{\mathrm{of}_1} \quad \Longrightarrow \quad \mathcal{D}_{\mathrm{of}_2}$

Different continuation monads

1. Continuation monad
   $\nabla\varphi \quad = \quad (\varphi \Rightarrow o) \Rightarrow o.$

2. Modal continuation monad
   $\nabla\varphi \quad = \quad (\varphi \Rightarrow \Diamond o) \Rightarrow \Diamond o.$

3. Polymorphic continuation monad
   $\nabla\varphi \quad = \quad \forall\alpha.(\varphi \Rightarrow \alpha) \Rightarrow \alpha.$

4. Polymorphic modal continuation monad
   $\nabla\varphi \quad = \quad \forall\alpha.(\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha.$

Remark. Cases 1 and 3 are obtained by taking $\Diamond$ as the identity monad.

Modal continuation monad

$$\nabla\varphi \quad = \quad (\varphi \Rightarrow \Diamond o) \Rightarrow \Diamond o.$$

$$\begin{array}{lcl} unit_{\nabla} & = & \lambda t{:}\,\varphi.\lambda k{:}\,\varphi \Rightarrow \Diamond o.(k\ t). \\ bind_{\nabla} & = & \lambda k{:}\,\varphi \Rightarrow \nabla\psi.\lambda m{:}\,\nabla\varphi.\lambda c{:}\,\psi \Rightarrow \Diamond o.m\ (\lambda v{:}\,\varphi.k\ v\ c). \end{array}$$

%solve $\vdash unit_{\nabla} : \varphi \Rightarrow \nabla\varphi$

%solve $\vdash bind_{\nabla} : (\varphi \Rightarrow \nabla\psi) \Rightarrow \nabla\varphi \Rightarrow \nabla\psi$

Polymorphic modal continuation monad

$$\nabla\varphi \quad = \quad \forall\alpha.(\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha.$$

$$\begin{array}{lcl} unit_{\nabla} & = & \lambda t{:}\,\varphi.\Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.(k\ t). \\ bind_{\nabla} & = & \lambda m{:}\,\nabla\varphi.\lambda k{:}\,\varphi \Rightarrow \nabla\psi.\Lambda\alpha.\lambda c{:}\,\psi \Rightarrow \Diamond\alpha.m\{\alpha\}\ (\lambda v{:}\,\varphi.(k\ v)\{\alpha\}\ c). \end{array}$$

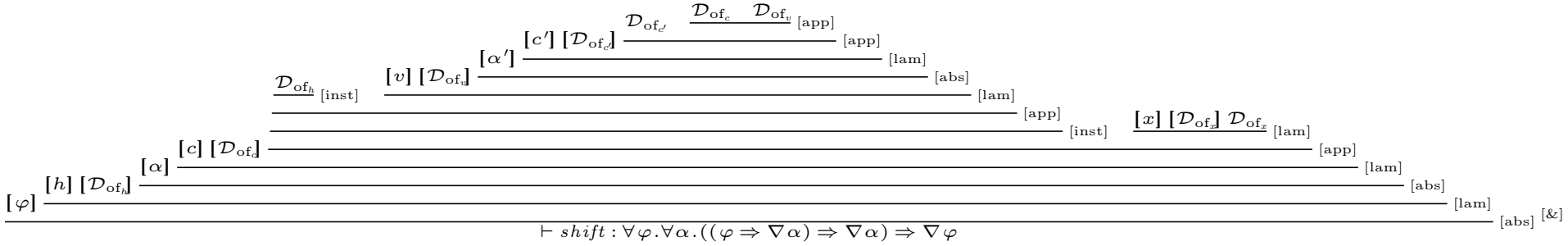
$$\begin{array}{ll} \textbf{\%solve} & \vdash unit_{\nabla} : \varphi \Rightarrow \nabla\varphi \\ \textbf{\%solve} & \vdash bind_{\nabla} : \nabla\varphi \Rightarrow (\varphi \Rightarrow \nabla\psi) \Rightarrow \nabla\psi \end{array}$$

Polymorphic continuation monad (shift)

$$\begin{array}{lcl}
\nabla\varphi & = & \forall\alpha.(\varphi \Rightarrow \alpha) \Rightarrow \alpha. \\
\mathit{shift} & = & \Lambda\varphi.\lambda h{:}\,\forall\alpha.((\varphi \Rightarrow \nabla\alpha) \Rightarrow \nabla\alpha). \\
 & & \quad \Lambda\alpha.\lambda c{:}\,\varphi \Rightarrow \alpha. \\
 & & \quad\quad ((h\{\alpha\}\ (\lambda v{:}\,\varphi.\Lambda\alpha'.\lambda c'{:}\,\alpha \Rightarrow \alpha'.c'\ (c\ v)))\{\alpha\}\ \lambda x{:}\,\alpha.x).
\end{array}$$

lemma $\vdash \mathit{shift} : \forall\varphi.(\forall\alpha.((\varphi \Rightarrow \nabla\alpha) \Rightarrow \nabla\alpha)) \Rightarrow \nabla\varphi$

Proof.



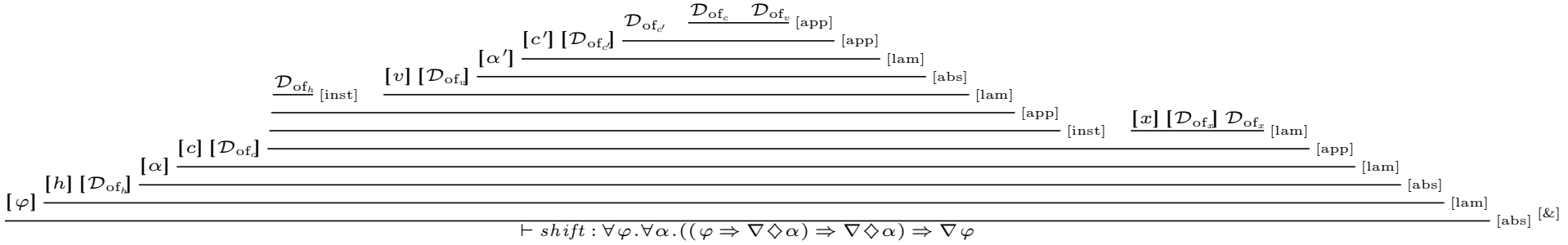
%mode $-\,\mathcal{D}_{\mathrm{of}}$
%worlds $()\quad \mathcal{D}_{\mathrm{of}}$
%total $\{\}\quad \mathcal{D}_{\mathrm{of}}$

Polymorphic modal continuation monad (shift)

$$\nabla\varphi \quad = \quad \forall\alpha.(\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha.$$
$$\mathit{shift} \quad = \quad \Lambda\varphi.\lambda h{:}\,\forall\alpha.((\varphi \Rightarrow \nabla\Diamond\alpha) \Rightarrow \nabla\Diamond\alpha).$$
$$\Lambda\alpha.\lambda c{:}\,\varphi \Rightarrow \Diamond\alpha.$$
$$((h\{\alpha\}\ (\lambda v{:}\,\varphi.\Lambda\alpha'.\lambda c'{:}\,\Diamond\alpha \Rightarrow \Diamond\alpha'.c'\ (c\ v)))\{\alpha\}\ \lambda x{:}\,\Diamond\alpha.x).$$

lemma $\quad \vdash \mathit{shift} : \forall\varphi.(\forall\alpha.((\varphi \Rightarrow \nabla\Diamond\alpha) \Rightarrow \nabla\Diamond\alpha)) \Rightarrow \nabla\varphi$

Proof.



%mode $\quad -\mathcal{D}_{\mathrm{of}}$
%worlds $\quad ()\quad \mathcal{D}_{\mathrm{of}}$
%total $\quad \{\}\quad \mathcal{D}_{\mathrm{of}}$

Example: the continuation monad

$$\Diamond\varphi \quad = \quad (\varphi \Rightarrow o) \Rightarrow o.$$

$$\begin{aligned} unit_{\varphi} \quad &= \quad \lambda t{:}\,\varphi.\lambda k{:}\,\varphi \Rightarrow o.(k\ t). \\ bind_{\varphi,\psi} \quad &= \quad \lambda k{:}\,\varphi \Rightarrow \Diamond\psi.\lambda m{:}\,\Diamond\varphi.\lambda c{:}\,\psi \Rightarrow o.m\ (\lambda v{:}\,\varphi.k\ v\ c). \end{aligned}$$

$$\begin{aligned} &\textbf{\%solve} \quad \vdash unit_{\varphi} : \varphi \Rightarrow \Diamond\varphi \\ &\textbf{\%solve} \quad \vdash bind_{\varphi,\psi} : (\varphi \Rightarrow \Diamond\psi) \Rightarrow \Diamond\varphi \Rightarrow \Diamond\psi \end{aligned}$$

$$\vdash \textbf{reset} : \Diamond\varphi \Rightarrow \Diamond\varphi \ ^{[\text{reset}]}$$

$$\vdash \textbf{shift} : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha) \Rightarrow \varphi \ ^{[\text{shift}]}$$

$$\begin{aligned} [t] \quad &= \quad \textbf{reset}\ (unit_{\varphi}\ t). \\ \mu(t) \quad &= \quad \textbf{shift}\ (\Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.bind_{\varphi,\alpha}\ k\ t). \end{aligned}$$

$$\begin{aligned} &\textbf{\%solve} \quad \vdash t : \varphi \ \rightarrow \ \vdash [t] : \Diamond\varphi \\ &\textbf{\%solve} \quad \vdash t : \Diamond\varphi \ \rightarrow \ \vdash \mu(t) : \varphi \end{aligned}$$

Example: the continuation monad

Defining **escape** in direct style:

$$\textbf{escape} \quad = \quad \lambda h\colon (\varphi \Rightarrow \psi) \Rightarrow \varphi.\mu(\lambda c\colon \varphi \Rightarrow o.\,[h\ \lambda a\colon \varphi.\mu(\lambda c'\colon \psi \Rightarrow o.c\ a)]\ c).$$

%solve $\vdash \textbf{escape} : ((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$

What is the logical meaning of $\Diamond\varphi \Rightarrow \varphi$?

Since $\Diamond\varphi \equiv (\varphi \Rightarrow o) \Rightarrow o$, for some formula o, we get $\neg\neg\varphi \Rightarrow \varphi$ which extends the logic to classical logic if we take $o = \bot$, but this axiom is incoherent if o is a theorem (note that it is always at least classical logic).

Example: the state monad

$$\sigma \quad : \quad \mathit{type}.$$
$$\Diamond\varphi \quad = \quad \sigma \Rightarrow (\varphi \wedge \sigma).$$

$$\mathit{unit}_{\varphi} \quad = \quad \lambda a{:}\,\varphi.\lambda s{:}\,\sigma.\langle a, s\rangle.$$
$$\mathit{bind}_{\varphi,\psi} \quad = \quad \lambda f{:}\,\varphi \Rightarrow \Diamond\psi.\lambda t{:}\,\Diamond\varphi.\lambda s{:}\,\sigma.\mathbf{let}\ \langle x, s'\rangle = t\ s\ \mathbf{in}\ f\ x\ s'.$$

$$\%\mathbf{solve} \quad \vdash \mathit{unit}_{\varphi} : \varphi \Rightarrow \Diamond\varphi$$
$$\%\mathbf{solve} \quad \vdash \mathit{bind}_{\varphi,\psi} : (\varphi \Rightarrow \Diamond\psi) \Rightarrow \Diamond\varphi \Rightarrow \Diamond\psi$$

$$\vdash \mathbf{reset} : \Diamond\varphi \Rightarrow \Diamond\varphi \ ^{[\mathrm{reset}]}$$
$$\vdash \mathbf{shift} : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha) \Rightarrow \varphi \ ^{[\mathrm{shift}]}$$

$$[t] \quad = \quad \mathbf{reset}\ (\mathit{unit}_{\varphi}\ t).$$
$$\mu(t) \quad = \quad \mathbf{shift}\ (\Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.\mathit{bind}_{\varphi,\alpha}\ k\ t).$$

$$\%\mathbf{solve} \quad \vdash t : \varphi \ \rightarrow\ \vdash [t] : \Diamond\varphi$$
$$\%\mathbf{solve} \quad \vdash t : \Diamond\varphi \ \rightarrow\ \vdash \mu(t) : \varphi$$

Example: the state monad

Defining **fetch** and **store** in direct style:

$$\begin{aligned}\mathbf{unit} &= \forall\alpha.\alpha \Rightarrow \alpha.\\ \langle\rangle &= \Lambda\alpha.\lambda x{:}\,\alpha.x.\end{aligned}$$

$$\mathbf{\%solve} \quad \vdash \langle\rangle : \mathbf{unit}$$

$$\begin{aligned}\mathbf{store} &= \lambda n{:}\,\sigma.\mu(\lambda s{:}\,\sigma.\langle\langle\rangle, n\rangle).\\ \mathbf{fetch} &= \lambda x{:}\,\mathbf{unit}.\mu(\lambda s{:}\,\sigma.\langle s, s\rangle).\end{aligned}$$

$$\begin{aligned}&\mathbf{\%solve} \quad \vdash \mathbf{fetch} : \mathbf{unit} \Rightarrow \sigma\\ &\mathbf{\%solve} \quad \vdash \mathbf{store} : \sigma \Rightarrow \mathbf{unit}\end{aligned}$$

What is the logical meaning of $\Diamond\varphi \Rightarrow \varphi$?

Since $\Diamond\varphi \equiv \sigma \Rightarrow (\varphi \wedge \sigma)$, for some formula σ, we get $(\sigma \Rightarrow (\varphi \wedge \sigma)) \Rightarrow \varphi$ which is not valid in general. This axiom is derivable if σ is a theorem, but it is incoherent if we take $\sigma = \bot$.

Example: the exception monad

$$\varepsilon \;:\; \mathit{type}.$$
$$\Diamond\varphi \;=\; \varphi \vee \varepsilon.$$

$$\mathit{unit}_{\varphi} \;=\; \lambda a{:}\,\varphi.\mathbf{inl}\,a.$$
$$\mathit{bind}_{\varphi,\psi} \;=\; \lambda f{:}\,\varphi \Rightarrow \Diamond\psi.\lambda t{:}\,\Diamond\varphi.\mathbf{case}\,t\,\mathbf{of}\;(\mathbf{inl}\,a) \mapsto f\;\;a \mid (\mathbf{inr}\,b) \mapsto \mathbf{inr}\,b.$$

%solve $\vdash \mathit{unit}_{\varphi} : \varphi \Rightarrow \Diamond\varphi$
%solve $\vdash \mathit{bind}_{\varphi,\psi} : (\varphi \Rightarrow \Diamond\psi) \Rightarrow \Diamond\varphi \Rightarrow \Diamond\psi$

$\vdash \mathbf{reset} : \Diamond\varphi \Rightarrow \Diamond\varphi$ [reset]

$\vdash \mathbf{shift} : \forall\alpha.((\varphi \Rightarrow \Diamond\alpha) \Rightarrow \Diamond\alpha) \Rightarrow \varphi$ [shift]

$$[t] \;=\; \mathbf{reset}\;(\mathit{unit}_{\varphi}\;t).$$
$$\mu(t) \;=\; \mathbf{shift}\;(\Lambda\alpha.\lambda k{:}\,\varphi \Rightarrow \Diamond\alpha.\mathit{bind}_{\varphi,\alpha}\;k\;t).$$

%solve $\vdash t : \varphi \;\rightarrow\; \vdash [t] : \Diamond\varphi$
%solve $\vdash t : \Diamond\varphi \;\rightarrow\; \vdash \mu(t) : \varphi$

Example: the exception monad

Defining **raise** and **handle** in direct style:

$$\begin{array}{lcl} \mathbf{raise} & = & \lambda e{:}\,\varepsilon.\mu(\mathbf{inr}\,e). \\ \mathbf{handle} & = & \lambda t{:}\,\varphi.\lambda h{:}\,\varepsilon \Rightarrow \varphi.\mathbf{case}\ [t]\ \mathbf{of}\ (\mathbf{inl}\,a) \mapsto a \mid (\mathbf{inr}\,e) \mapsto h\ e. \end{array}$$

$$\begin{array}{ll} \%\mathbf{solve} & \vdash \mathbf{raise} : \varepsilon \Rightarrow \alpha \\ \%\mathbf{solve} & \vdash \mathbf{handle} : \varphi \Rightarrow (\varepsilon \Rightarrow \varphi) \Rightarrow \varphi \end{array}$$

What is the logical meaning of $\Diamond\varphi \Rightarrow \varphi$?

Since $\Diamond\varphi \equiv \varphi \vee \varepsilon$, for some formula ε, we get $(\varphi \vee \varepsilon) \Rightarrow \varphi$ which is not valid in general. This axiom is incoherent if ε is a theorem, but it is derivable if $\neg\varepsilon$ is derivable.

Concluding remarks

- Depending on $\diamondsuit$, the type of **shift** can be:
  - *intuitionistic*
  - *classical*
  - *incoherent*
- However a proof of $\vdash \bot$ is translated into a proof $\vdash \diamondsuit \bot$
  (and the target logic is consistent since $\diamondsuit$ is defined)
- In a dependently typed framework $\vdash \mathbf{reset} : \diamondsuit\varphi \Rightarrow \diamondsuit\varphi$ is useless.
  Is it just an optimization?
- Similarly, we can replace **shift** by the $\mathcal{D}$-operator.
- Why should we expect **shift** to be logically sound?

Bibliography

[1] H. B. Curry. The elimination theorem when modality is present. *The Journal of Symbolic Logic*, 17 (4): 249–265, 1952. ISSN 0022-4812.

[2] A. Filinski. Representing monads. In *Conference Record of the Twenty-First Annual Symposium on Principles of Programming Languages*, pages 446–457, Portland, Oregon, January 1994.