Update monads:
Cointerpreting directed containers

Danel Ahman, U. of Edinburgh
Tarmo Uustalu, Inst. of Cybernetics, Tallinn

TYPES 2013, 23–26 April 2013
Background: Three famous monads

Reader monad

- $S$ — a set
- $T X = S \to X$

State monad

- $S$ — a set
- $T X = S \to S \times X$

Writer monad

- $(P, o, \oplus)$ — a monoid
- $T X = P \times X$

$S$ — states, $P$ — updates (alt. "programs")
This talk: A unification (+ a little more)

Update monad
\[ S \rightarrow (P, o, \oplus) \rightarrow P \times X \]

Reader monad
\[ S \rightarrow X \]

State monad
\[ S \rightarrow S \times X \]

Writer monad
\[ (P, o, \oplus) \rightarrow P \times X \]
This talk: A unification (+ a little more)

Update monad

\[ T \mathcal{X} = S \rightarrow P \times X \]

\[ cf. \ T \mathcal{X} = \Pi_{s : S} (sP \times X) \] by Kammar and Plotkin

Reader monad

\[ S \rightarrow X \]

State monad

\[ S \rightarrow S \times X \]

Writer monad

\[ (P, o, \oplus) \rightarrow a \ monoid \]

\[ T \mathcal{X} = P \times X \]
Monoids, monoid actions

- A monoid on a set $P$ is given by
  \[
  o : P, \\
  \oplus : P \rightarrow P \rightarrow P,
  \]
  \[
  p \oplus o = p, \\
  o \oplus p = p, \\
  (p \oplus p') \oplus p'' = p \oplus (p' \oplus p'')
  \]

- An action of a monoid $(P, o, \oplus)$ on a set $S$ is given by
  \[
  \downarrow: S \rightarrow P \rightarrow S,
  \]
  \[
  s \downarrow o = s, \\
  s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'
  \]
A set $S$, monoid $(P, o, \oplus)$ and action $\downarrow$ give a monad via

$$TX = S \to P \times X$$

$$\eta : \forall \{X\}. X \to S \to P \times X$$

$$\eta x = \lambda s. (o, x)$$

$$\mu : \forall \{X\}. (S \to P \times (S \to P \times X)) \to S \to P \times X$$

$$\mu f = \lambda s. \text{let } (p, g) = f s;$$

$$(p', x) = g (s \downarrow p)$$

$$\text{in } (p \oplus p', x)$$
Reader and writer monads as instances

- Recall update monads:
  \[ T \times X = S \rightarrow (P \times X) \]

- Reader monads:
  update monads with \((P, o, \oplus)\) and \(\downarrow\) trivial

- Writer monads:
  update monads with \(S\) and \(\downarrow\) trivial

- State monads:
  embed into update monads
  with \(P\) the free monoid
  on the overwrite semi-group \((S, \bullet)\) with \(s \bullet s' = s'\)
Update monad example: writing into a buffer

- \( S = E^* \times \text{Nat} \)  
  \text{(buffer content and free space)}

- \( P = E^* \)  
  \text{(sequence of values written)}

- \( o = [] \)

- \( p \oplus p' = p \mathbin{++} p' \)

- \( (s, n) \downarrow p = (s \mathbin{++} (p|n), n - \text{length}(p|n)) \)
  \text{(p|n is p truncated to length n)}
Algebras of update monads

An algebra of such a monad is a set $X$ with an operation

$$\text{act} : (S \to P \times X) \to X$$

$$x = \text{act}(\lambda s. o, x)$$
$$\text{act}(\lambda s. p, \text{act}(\lambda s'. p', x))$$
$$= \text{act}(\lambda s. p \oplus p'[s \downarrow p/s'], x[s \downarrow p/s'])$$

or, equivalently a pair of operations (cf. algebraic effects)

$$\text{lkp} : (S \to X) \to X$$
$$\text{upd} : P \times X \to X$$

$$x = \text{lkp}(\lambda s. \text{upd}(o, x))$$
$$\text{upd}(p, (\text{upd}(p', x))) = \text{upd}(p \oplus p', x)$$
$$\text{lkp}(\lambda s. \text{upd}(p, \text{lkp}(\lambda s'. x))) = \text{lkp}(\lambda s. \text{upd}(p, x[s \downarrow p/s']))$$
The operations

\[ \text{act} : (S \to P \times X) \to X \]
\[ \text{lkp} : (S \to X) \to X \]
\[ \text{upd} : P \times X \to X \]

are interdefinable via

\[ \text{lkp} (\lambda s. x) = \text{act} (\lambda s. (o, x)) \]
\[ \text{upd} (p, x) = \text{act} (\lambda s. (p, x)) \]
\[ \text{act} (\lambda s. (p, x)) = \text{lkp} (\lambda s. \text{upd} (p, x)) \]
Update monads as compatible compositions

The update monad for $S$, $(P, o, \oplus)$, $\downarrow$ is the compatible composition the reader and writer monads

$$T_0 X = S \to X$$
$$T_1 X = P \times X$$

$$\eta_0 : \forall \{X\}. X \to S \to X$$
$$\eta_0 x = \lambda s. x$$

$$\eta_1 : \forall \{X\}. X \to P \to X$$
$$\eta_1 x = (o, x)$$

$$\mu_0 : \forall \{X\}. (S \to (S \to X)) \to S \to X$$
$$\mu_0 f = \lambda s. f s s$$

$$\mu_1 : \forall \{X\}. (P \times (P \times X)) \to P \times X$$
$$\mu_1 ((p, p'), x) = (p \oplus p', x)$$

for the distributive law

$$\lambda : \forall \{X\}. P \times (S \to X) \to (S \to P \times X)$$
$$\lambda (p, f) = \lambda s. (p, f (s \downarrow p))$$
Update algebras as compatible pairs of reader and writer algebras

An algebra of the update monad for $S$, $(P, o, \oplus)$, $\downarrow$ is a set $X$ carrying algebras of both the reader and writer monad

\[
\begin{align*}
\text{lkp} : (S \rightarrow X) & \rightarrow X & \text{upd} : P \times X & \rightarrow X \\
\text{lkp} (\lambda s. x) & = x & \text{upd} (o, x) & = x \\
\text{lkp} (\lambda s. (\text{lkp} \lambda s'. x)) & = \text{lkp} (\lambda s. x[s/s']) & \text{upd} (p, \text{upd}(p', x)) & = \text{upd} (p \oplus p', x)
\end{align*}
\]

satisfying an additional compatibility condition

\[
\text{upd} (p, \text{lkp} (\lambda s'. x)) = \text{lkp} (\lambda s. \text{upd} (p, x[s \downarrow p/s']))
\]
A finer version

Rather than

\[ S \rightarrowtext{a set} \]
\[ (P, o, \oplus) \rightarrowtext{a monoid} \]
\[ \downarrow \rightarrowtext{an action} \]
\[ TX = S \rightarrow P \times X \]

consider

\[ (S, P, \downarrow, o, \oplus) \rightarrowtext{a directed container} \]
\[ TX = \prod_{s : S} P_s \times X \]

\[ S \rightarrowtext{states, } P_s \rightarrowtext{updates enabled (or safe) in state } s \]
Directed containers

- A directed container is

\[
S \text{ a set,} \quad P \text{ a } S\text{-indexed family,}
\]

\[
\downarrow : \prod s : S. Ps \to S,
\]

\[
o : \prod \{s : S\}. Ps
\]

\[
\oplus : \prod \{s : S\}. \prod p : Ps. P(s \downarrow p) \to Ps,
\]

\[
s \downarrow o = s,
\]

\[
s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p',
\]

\[
p \oplus o = p,
\]

\[
o \oplus p = p,
\]

\[
(p \oplus p') \oplus p'' = p \oplus (p' \oplus p''),
\]
Monads from directed containers

A directed container \((S, P, \downarrow, o, \oplus)\) yields a monad via

\[
T X = \prod s : S. P s \times X
\]

\[
\eta : \forall \{X\}. X \rightarrow \prod s : S. P s \times X
\]
\[
\eta x = \lambda s. (o, x)
\]

\[
\mu : \forall \{X\}. (\prod s : S. P s \times (\prod s' : S. P s' \times X)) \rightarrow \prod s : S. P s \times X
\]
\[
\mu f = \lambda s. \text{let } (p, g) = f s; \\
\text{in } (p \oplus p', x)
\]
Example: writing into a buffer (a finer version)

- $S = E^* \times \text{Nat}$  
  (buffer content and free space)

- $P(s, n) = E^{\leq n}$  
  (sequence of values written)

- $o = []$

- $p \oplus p' = p \mathbin{\|} p'$

- $(s, n) \downarrow p = (s \mathbin{\|} p, n - \text{length}(p))$
Monads from directed containers: Algebras

An algebra for the monad for the directed container \((S, P, \downarrow, o, \oplus)\) is a set \(X\) with an operation

\[
\text{act} : (\prod s : S. P s \times X) \rightarrow X
\]

\[
x = \text{act} (\lambda s. o, x)
\]

\[
\text{act} (\lambda s. p, \text{act} (\lambda s'. p', x))
\]

\[
= \text{act} (\lambda s. p \oplus p'[s \downarrow p/s'], x[s \downarrow p/s'])
\]
Directed container morphisms, monad morphisms

- A morphism between two directed containers $(S', P', ↓', o', ⊕')$ and $(S, P, ↓, o, ⊕)$ is given by

  \[ t : S' \to S \]
  \[ q : \prod\{s : S'\} \cdot P(t s) \to P' s \]

  \[ t(s ↓' q p) = t s ↓ p \]
  \[ o' = q o \]
  \[ q p ⊕' q p' = q(p ⊕ p) \]

- It yields a morphism between the monads $(T, η, µ)$ and $(T', η', µ')$ via

  \[ τ : \forall\{X\}. (\prod s : S. P s \times X) \to \prod s : S'. P' s \times X \]
  \[ τ f = λs. \text{let} \ (p, x) = f(t s) \text{ in} \ (q p, x) \]

- Notice the reversal of arrow directions!
Directed containers and comonads

(Directed containers and comonads)

\[ \text{DCont} \cong \text{Comonoids}(\text{Cont}) \xrightarrow{\sim} \text{Comonoids}([\text{Set}, \text{Set}]) \xrightarrow{\sim} \text{Comonoids}([\text{Set}, \text{Set}]) \xrightarrow{U} \text{[Set, Set]} \xrightarrow{\text{mon.}} \]

\[ [S, P]^c X = \Sigma s : S. P s \rightarrow X \]
Directed containers and monads

(the new picture)

\[ \text{DCont}^{\text{op}} \cong (\text{Comonoids} (\text{Cont}))^{\text{op}} \]
\[ \cong \text{Monoids}(\text{Cont}^{\text{op}}) \]
\[ \Rightarrow \text{Monads}([\text{Set}, \text{Set}]) \]
\[ \Rightarrow \text{Monoids}([\text{Set}, \text{Set}]) \]

\[ \langle \langle - \rangle \rangle^{dc} \]
\[ \Rightarrow \text{Monads}([\text{Set}, \text{Set}]) \]
\[ \Rightarrow \text{Monoids}([\text{Set}, \text{Set}]) \]

\[ \langle \langle - \rangle \rangle^c \text{ lax mon.} \]

\[ U \]

\[ \Rightarrow \text{Cont}^{\text{op}} \text{ mon.} \]

\[ \Rightarrow \text{Cont}^{\text{op}} \text{ mon.} \]

\[ U \]

\[ \Rightarrow [\text{Set}, \text{Set}] \text{ mon.} \]

\[ \Rightarrow [\text{Set}, \text{Set}] \text{ mon.} \]

\[ \langle \langle S, P \rangle \rangle^c X = \Pi s : S. P s \times X \]