Monads and Interaction
Lecture 2bis

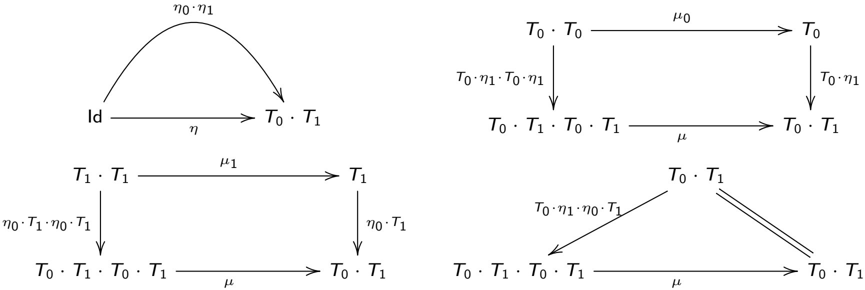
Tarmo Uustalu, Reykjavik University

OPLSS 2021, Eugene, OR, 14–25 June 2021

Compatible compositions of monads

Compatible compositions of monads

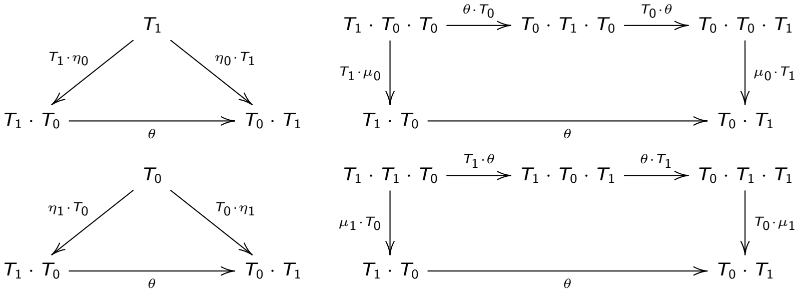
- A *compatible composition* of two monads (T_0, η_0, μ_0), (T_1, η_1, μ_1) is a monad structure (η, μ) on $T = T_0 \cdot T_1$ satisfying



- Equations 1-3 say just that $T_0 \cdot \eta_1$ and $\eta_0 \cdot T_1$ are monad morphisms between (T_0, η_0, μ_0) resp. (T_1, η_1, μ_1) and (T, η, μ).
  Equation 1 fixes that $\eta = \eta_0 \cdot \eta_1$; so there is some freedom only about μ.
- Equation 4 is called *middle unitality*.

Distributive laws of monads

- A *distributive law* of a monad (T_1, η_1, μ_1) over (T_0, η_0, μ_0) is a natural transformation $\theta : T_1 \cdot T_0 \to T_0 \cdot T_1$ such that



Compatible compositions = distributive laws

- Compatible compositions of (T_0, η_0, μ_0), (T_1, η_1, μ_1) are in a bijection with distributive laws of (T_1, η_1, μ_1) over (T_0, η_0, μ_0).
- Given μ, one recovers θ by

$$\theta = \; T_1 \cdot T_0 \xrightarrow{\eta_0 \cdot T_1 \cdot T_0 \cdot \eta_1} T_0 \cdot T_1 \cdot T_0 \cdot T_1 \xrightarrow{\mu} T_0 \cdot T_1$$

- Given θ, μ is defined by

$$\mu = \; T_0 \cdot T_1 \cdot T_0 \cdot T_1 \xrightarrow{T_0 \cdot \theta \cdot T_1} T_0 \cdot T_0 \cdot T_1 \cdot T_1 \xrightarrow{\mu_0 \cdot \mu_1} T_0 \cdot T_1$$

Algebras of compatible compositions

- Given a distributive law θ, a θ-*pair of algebras* is given by a set X with a (T_0, η_0, μ_0)-algebra structure (X, ξ_0) and a (T_1, η_1, μ_1)-algebra structure (X, ξ_1) such that

$$\begin{array}{ccccc} T_1X & \xrightarrow{\xi_1} & X & \xleftarrow{\xi_0} & T_0X \\ {\scriptstyle T_1\xi_0}\uparrow & & & & \uparrow{\scriptstyle T_0\xi_1} \\ T_1(T_0X) & & \xrightarrow{\theta_X} & & T_0(T_1X) \end{array}$$

- Such pairs of algebras are in a bijection with (T, η, μ)-algebras.
- Given ξ_0, ξ_1, one constructs ξ as
  - $\xi = T_0(T_1X) \xrightarrow{T_0\xi_1} T_0X \xrightarrow{\xi_0} X$.
- Given ξ, one defines ξ_0 and ξ_1 by
  - $\xi_0 = T_0X \xrightarrow{T_0\eta_{1X}} T_0(T_1X) \xrightarrow{\xi} X$,
  - $\xi_1 = T_1X \xrightarrow{\eta_{0\,T_1X}} T_0(T_1X) \xrightarrow{\xi} X$.

Distributive laws of exceptions monads

- Suppose $\mathcal{C}$ has finite coproducts.

- The exceptions monad for E distributes in a unique way over any monad (T_0, η_0, μ_0).
- $\theta_X : E + T_0 X \to T_0(E + X)$
  $\theta\,(\mathsf{inl}\,e) = \eta_0\,(\mathsf{inl}\,e)$,
  $\theta\,(\mathsf{inr}\,c) = T_0\,\mathsf{inr}$

- So we have a unique monad structure on $TX = T_0(E + X)$ compatible with the two monads.

Distributive laws of writer monads

- Suppose $\mathcal{C}$ has finite products.

- There is a distributive law of the writer monad for $(P, \mathrm{o}, \oplus)$ over any strong monad (T_0, η_0, μ_0).
- $\theta : P \times T_0X \to T_0(P \times X)$
  $\theta\,(p, c) = \mathrm{st}\,(p, c)$

- This gives a monad structure on $TX = T_0(P \times X)$ compatible with the two monads.

- This example generalizes to any monoidal category.

Distr. laws of writer monads over reader monads

- Suppose $\mathcal{C}$ is Cartesian closed.

- A *right action* of a monoid $(P, \text{o}, \oplus)$ on an object S is a map $\downarrow : S \times P \to S$ satisfying $s \downarrow \text{o} = s$, $s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'$.

- Distributive laws of the writer monad for $(P, \text{o}, \oplus)$ over the reader monad for S are in a bijection with right actions of $(P, \text{o}, \oplus)$ on S.

- The compatible composition of the two monads determined by a right action $\downarrow$ is
  - $T\,X = S \Rightarrow P \times X$
  - $\eta\,x = \lambda s.\,(\text{o}, x)$
  - $\mu\,f = \lambda s.\,\text{let}\ \ (p, g) = f\,s$
    $\qquad\qquad\qquad\quad (p', x) = g\,(s \downarrow p)$
    $\qquad\qquad\ \text{in}\ (p \oplus p', x)$

  —the *update monad* for S, $(P, \text{o}, \oplus)$, $\downarrow$.

- This example generalizes to any monoidal closed category.

State logging

- Take S to be some object (of states).
- Take $P = \text{List}\, S$, $\text{o} = []$, $\oplus = +\!\!+$ (state logs).

- $s \downarrow [] = s$
  $s \downarrow (s' :: ss) = s' \downarrow ss$

  (so $s \downarrow ss$ is the last element of $s :: ss$)

Reading a stack and popping

- Take $S = \mathsf{List}\, E$ (states of a stack for some object E of elements).
- Take $P = \mathsf{Nat}$, $\mathsf{o} = 0$, $\oplus = +$ (possible numbers of elements to pop).
- $es \downarrow n = removelast\, n\, es$.

Reading a stack and pushing

- Take again $S = \mathsf{List}\, E$.
- Take $P = \mathsf{List}\, E$, $\mathrm{o} = []$, $\oplus = +\!\!+$ (lists of elements to push on the stack).
- $es \downarrow es' = es +\!\!+ es'$.
- (So here we choose $(S, \downarrow)$ to be the initial $(P, \mathrm{o}, \oplus)$-set—which is always a possibility.)

Matching pairs of monoid actions

- Suppose $\mathcal{C}$ has finite products.
- A *matching pair of actions* of two monoids $(P_0, o_0, \oplus_0)$ and $(P_1, o_1, \oplus_1)$ on each other is pair of maps $\searrow : P_1 \times P_0 \to P_0$ and $\swarrow : P_1 \times P_0 \to P_1$ such that

$$
\begin{aligned}
o_1 \searrow p_0 &= p_0 \\
(p_1 \oplus_1 p_1') \searrow p_0 &= p_1 \searrow (p_1' \searrow p_0) \\
p_1 \searrow o_0 &= o_0 \\
p_1 \searrow (p_0 \oplus_0 p_0') &= (p_1 \searrow p_0) \oplus_0 ((p_1 \swarrow p_0) \searrow p_0')
\end{aligned}
$$

$$
\begin{aligned}
p_1 \swarrow o_0 &= p_1 \\
p_1 \swarrow (p_0 \oplus_0 p_0') &= (p_1 \swarrow p_0) \swarrow p_0' \\
o_1 \swarrow p_0 &= o_1 \\
(p_1 \oplus_1 p_1') \swarrow p_0 &= (p_1 \swarrow (p_1' \searrow p_0)) \oplus_1 (p_1' \swarrow p_0)
\end{aligned}
$$

Zappa-Szép product of monoids

- A *Zappa-Szép product* (aka *knit product*, *bicrossed product*, *bilateral semidirect product*) of two monoids $(P_0, o_0, \oplus_0)$ and $(P_1, o_1, \oplus_1)$ is a monoid structure $(o, \oplus)$ on $P = P_0 \times P_1$ such that

$$\begin{aligned} o &= (o_0, o_1) \\ (p, o_1) \oplus (p', o_1) &= (p \oplus_0 p', o_1) \\ (o_0, p) \oplus (o_0, p') &= (o_0, p \oplus_1 p') \\ (p, o_1) \oplus (o_0, p') &= (p, p') \end{aligned}$$

- Zappa-Szép products of $(P_0, o_0, \oplus_0)$ and $(P_1, o_1, \oplus_1)$ are in a bijective correspondence with matching pairs of actions of $(P_0, o_0, \oplus_0)$ and $(P_1, o_1, \oplus_1)$.
- Given $\oplus$, one constructs $\searrow$ and $\swarrow$ by
  - $(p_1 \searrow p_0, p_1 \swarrow p_0) = (o_0, p_1) \oplus (p_0, o_1)$
- Given $\searrow$ and $\swarrow$, $\oplus$ is defined by
  - $(p_0, p_1) \oplus (p_0', p_1') = (p_0 \oplus_0 (p_1 \swarrow p_0'), (p_1 \searrow p_0') \oplus_1 p_1')$

Two writer monads

- Compatible compositions of writer monads for $(P_0, \mathsf{o}_0, \oplus_0)$ and $(P_1, \mathsf{o}_1, \oplus_1)$ are in a bijection with matching pairs of actions of the two monoids.
- They are isomorphic to writer monads for the corresponding Zappa-Szép products.

- This example generalizes to any monoidal category.

Combining popping and pushing

- Take $(P_0, o_0, \oplus_0) = (\mathrm{Nat}, 0, +)$, $(P_1, o_1, \oplus_1) = (\mathrm{List}\, E, [], +\!\!+)$ where E is some set.

- $es \searrow n = n \mathbin{\dot{-}} \mathit{length}\, es$,
  $es \swarrow n = \mathit{removelast}\, n\, es$.
- $(n, es) \oplus (n', es')$
  $= (n + (n' \mathbin{\dot{-}} \mathit{length}\, es), (\mathit{removelast}\, n'\, es) +\!\!+ es')$

- Pairs (n, es) represent net effects of sequences of pop, push instructions on a stack: some number of elements is removed from and some new specific elements are added to the stack.

Combining reading, popping, pushing

- How do I now show that
  - $TX = \mathsf{List}\, E \Rightarrow \mathsf{Nat} \times (\mathsf{List}\, E \times X)$

  is a monad?
- This is of the form $T_0 \cdot T_1 \cdot T_2$ where
  - $T_0 X = \mathsf{List}\, E \Rightarrow X$
  - $T_1 X = \mathsf{Nat} \times X$
  - $T_2 X = \mathsf{List}\, E \times X$
- We already know that
  - $T_{01} = T_0 \cdot T_1$
  - $T_{02} = T_0 \cdot T_2$
  - $T_{12} = T_1 \cdot T_2$

  are compatible compositions of monads.
- We want to be sure that $(T_0 \cdot T_1) \cdot T_2$ and $T_0 \cdot (T_1 \cdot T_2)$ are compatible compositions of monads.
- Moreover, they'd better be the same monad!

- In terms of distributive laws, this only requires checking the Yang-Baxter equation: