(Co)Monads

Alyson Mei

December 23, 2025

Contents

1 Introduction

Conventions:

– Standard polymorphic notation is used freely when no ambiguity arises;

– All diagrams are assumed to commute unless stated otherwise.

TODO:

– Specify the Δ framework;

– Check the details for the Exception monad;

– Consider algebras, Eilenberg–Moore category, etc.;

– Write the comonads section.

2 Monads

2.1 Monoid object and Monad

Definition 2.1 (Monoid object)**.** Let $(\mathcal{C}, \otimes, I)$ be a monoidal category. A monoid object in $\mathcal{C}$ consists of:

- an object $M \in \mathrm{obj}(\mathcal{C})$,
- a multiplication morphism
$$\mu : M \otimes M \to M,$$
- a unit morphism
$$\eta : I \to M,$$

satisfying the unit and associativity diagrams:

$$\begin{tikzcd} M \arrow[r, "M\otimes\eta"] \arrow[dr, "\mathrm{id}"'] & M \otimes M \arrow[d, dashed, "\mu"] & M \arrow[l, "\eta\otimes M"'] \arrow[dl, "\mathrm{id}"] \\ & M & \end{tikzcd} \qquad \begin{tikzcd} M^{\otimes 3} \arrow[r, "\mu\otimes M"] \arrow[d, "M\otimes\mu"'] & M^{\otimes 2} \arrow[d, "\mu"] \\ M^{\otimes 2} \arrow[r, "\mu"'] & M \end{tikzcd} .$$

Example 2.2.

- Categories with finite products $(\mathcal{C}, \times, 1)$ are monoidal:
  - In $\mathcal{S}et$, monoid objects are ordinary monoids;
  - In $\mathrm{Vect}_{\mathbf{k}}$, monoid objects are associative unital algebras over a field $\mathbf{k}$;
- The category of endofunctors $([\mathcal{C}, \mathcal{C}], \circ, \mathrm{id})$ is monoidal; monoid objects in this category are called monads.

Definition 2.3 (Monad)**.** Monad is a triple

$$(M : \mathcal{C} \to \mathcal{C}, \eta : \mathrm{id}_{\mathcal{C}} \to M, \mu : M^2 \to M),$$

satisfying the unit and associativity diagrams:

$$\begin{tikzcd} M \arrow[r, "\eta\circ M"] \arrow[dr, "\mathrm{id}_M"'] & M^2 \arrow[d, "\mu"] & M \arrow[l, "M\circ\eta"'] \arrow[dl, "\mathrm{id}_M"] \\ & M & \end{tikzcd} \qquad \begin{tikzcd} M^3 \arrow[r, "M\circ\mu"] \arrow[d, "\mu\circ M"'] & M^2 \arrow[d, "\mu"] \\ M^2 \arrow[r, "\mu"'] & M \end{tikzcd} .$$

2.2 Kleisli category

Definition 2.4 (Kleisli category)**.** Let (T, η, μ) be a monad on $\mathcal{C}$. The Kleisli category $\mathrm{Kl}(T)$ of the monad T is defined as follows:

- Objects of $\mathrm{Kl}(T)$ are the same as in $\mathcal{C}$,
- For $A, B \in \mathrm{obj}(\mathcal{C})$, a morphism $f^\bullet : A \to B$ in $\mathrm{Kl}(T)$ is a morphism $f : A \to TB$ in $\mathcal{C}$,
- Composition:

$$\begin{tikzcd} A \arrow[d, "f"'] \\ TB \end{tikzcd} \quad \begin{tikzcd} B \arrow[d, "g"'] \\ TC \end{tikzcd} \quad \begin{tikzcd} TB \arrow[d, "Tg"'] & A \arrow[l, "f"'] \arrow[d, "\mu_C \circ Tg \circ f"] \\ T^2C \arrow[r, "\mu_C"'] & TC \end{tikzcd} \quad \begin{tikzcd} A \arrow[d, "f^\bullet \circ g^\bullet \in \mathrm{Kl}(T)"] \\ C \end{tikzcd},$$

- Identity morphism $\mathrm{id}^\bullet_A$ in $\mathrm{Kl}(T)$ is a morphism $\eta_A : A \to TA$ in $\mathcal{C}$:

$$\begin{tikzcd} A \arrow[r, "\eta_A"] \arrow[d, "f"'] & TA \arrow[d, "Tf"] \\ TB \arrow[r, "\eta_{TB}"'] & T^2B \end{tikzcd} \quad \begin{tikzcd} TA \arrow[d, "Tf"'] & A \arrow[l, "\eta_A"'] \arrow[d, "\mu_B \circ \eta_{TB} \circ f = f"] \\ T^2B \arrow[r, "\mu_B"'] & TB \end{tikzcd} \quad \begin{tikzcd} A \arrow[d, "f \circ \mathrm{id}^\bullet_A"] \\ B \end{tikzcd}$$

$$\begin{tikzcd} TA \arrow[d, "T\eta_A"'] & C \arrow[l, "g"'] \arrow[d, "\mu_A \circ T\eta_A \circ g = g"] \\ T^2A \arrow[r, "\mu_A"'] & TA \end{tikzcd} \quad \begin{tikzcd} A \arrow[d, "\mathrm{id}^\bullet_A \circ g"] \\ B \end{tikzcd}.$$

2.3 Monads and adjunctions

Theorem 2.5 (Monads $\approx$ adjunctions)**.** Every adjunction gives rise to a monad, and every monad arises (up to equivalence) from an adjunction.

From adjunction to monad.
Let $F : \mathcal{C} \to \mathcal{D}$, $G : \mathcal{D} \to \mathcal{C}$, and F be the left adjoint to G:

$$\eta : \mathrm{id}_{\mathcal{C}} \to GF, \ \varepsilon : FG \to \mathrm{id}_{\mathcal{D}},$$

$$\begin{tikzcd} F \arrow[r, "F\eta"] \arrow[dr, "\mathrm{id}"'] & FGF \arrow[d, "\varepsilon F"] \\ & F \end{tikzcd} \qquad \begin{tikzcd} GFG \arrow[d, "G\varepsilon"'] & G \arrow[l, "\eta G"'] \arrow[dl, "\mathrm{id}"] \\ G & \end{tikzcd} .$$

We define the underlying functor M of the monad as

$$M := GF : \mathcal{C} \to \mathcal{C}.$$

Its unit is exactly η. We can define the multiplication $\mu : M^2 \to M$ using the counit:

$$\mu := G\varepsilon F.$$

The ε here essentially collapses the forget-free structure from GF.

The unit and associativity diagrams: //UNFINISHED

$$\begin{tikzcd} GF \arrow[r, "\eta GF"] \arrow[dr, "(G\varepsilon\circ\eta G)F=\mathrm{id}"'] & GFGF \arrow[d, "\mu=G\varepsilon F"] \\ & GF \end{tikzcd} \qquad \begin{tikzcd} GFGF \arrow[d, "\mu=G\varepsilon F"'] & GF \arrow[l, "GF\eta"'] \arrow[dl, "G(\varepsilon F\circ F\eta)=\mathrm{id}"] \\ GF & \end{tikzcd} ,$$

$$\begin{tikzcd} GFGFGF \arrow[r, "G\varepsilon FGF"] \arrow[d, "GFG\varepsilon F"'] & GFGF \arrow[d, "G\varepsilon F"] \\ GFGF \arrow[r, "G\varepsilon F"'] & GF \end{tikzcd} .$$

2.4 Examples

2.4.1 List monad

Definition and a monad structure.
Consider the functor List : $\mathcal{S}et \to \mathcal{S}et$, which sends a set to the set of all finite lists of its elements:

$$\begin{tikzcd} A \arrow[d, "f"] \\ B \end{tikzcd} \qquad \begin{tikzcd} \text{List } A \arrow[d, "\text{List } f"] \\ \text{List } B \end{tikzcd} \qquad \begin{tikzcd} {[a_1, a_2, ...a_n]} \arrow[d, mapsto, "\text{List } f"] \\ {[f(a_1), f(a_2), ..., f(a_n)]} \end{tikzcd} .$$

From a categorical perspective, the functor List can be presented as follows:

$$\begin{tikzcd} \mathcal{S}et \arrow[r, "\Delta"] \arrow[d, "\text{List}"'] & \mathcal{S}et^{\mathbb{N}} \arrow[d, "\prod_{n\in\mathbb{N}} \times^n"] \\ \mathcal{S}et & \mathcal{S}et^{\mathbb{N}} \arrow[l, "\coprod_{n\in\mathbb{N}}"] \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, mapsto] \arrow[d, mapsto, "\text{List}(A)"'] & A^{\mathbb{N}} \arrow[d, mapsto] \\ \coprod_{n\in\mathbb{N}} A^n & (A^n)_{n\in\mathbb{N}} \arrow[l, mapsto] \end{tikzcd} .$$

This presentation gives rise to the n API:

- The collection of constructors

$$[\ ]_{n,A} := A^n \xrightarrow{i_n} \text{List } A := \lambda a_1 \dots a_n.[a_1, \dots, a_n],$$

  in particular:

  - The single-element list constructor

$$\eta_A : A \to \text{List } A := \lambda a.[a],$$

  - The empty list constructor

$$[\ ]_0 : 1 \to \text{List } A := \lambda a.[];$$

- Length of a list via the power of A in the coproduct:

$$\text{len}_A : \text{List } A \to \mathbf{N};$$

- List concatenation via the natural isomorphism $A^n \times A^m \cong A^{n+m}$:

$$\begin{tikzcd} \text{id} \times \text{id} \arrow[r, "i \times i"] \arrow[d, "\simeq"'] & \text{List} \times \text{List} \arrow[d, "+"] \\ \text{id} \arrow[r, "i"'] & \text{List} \end{tikzcd} \qquad \begin{tikzcd} A^m \times A^n \arrow[r, "i_m \times i_n"] \arrow[d, "\cong"'] & \text{List } A \times \text{List } A \arrow[d, "+_A"] \\ A^{m+n} \arrow[r, "i_{m+n}"'] & \text{List } A \end{tikzcd} ,$$

$$+_A : \text{List } A \times \text{List } A \to \text{List } A := \lambda lm.[l_0, ...l_{\text{len}(l)-1}, m_0, ..., m_{\text{len}(m)-1}],$$
$$+_A : (\text{List } A)^n \to \text{List } A := \lambda l_0...l_n.l_0 + ... + l_n;$$

– List destructors:

$$\mu_A : \mathrm{List}(\mathrm{List}\, A) \to \mathrm{List}(A) := \lambda l.\ \sum_{i=0}^{\mathrm{len}(l)-1} l_i,$$

$$\mu_A : \mathrm{List}^n\, A \to \mathrm{List}\, A := \mu_A^{\circ n}.$$

First of all, this API provides an internal monoid structure on lists: for a fixed set A, we have a monoid object

$$(\mathrm{List}\, A, [], +_A)$$

in $\mathcal{S}et$. The corresponding diagrams are straightforward in this case.

Second, it provides a monad structure on List:

$$(\mathrm{List}, \eta : \mathrm{id} \to \mathrm{List}, \mu : \mathrm{List}^2 \to \mathrm{List}).$$

The corresponding diagrams:

$$\begin{tikzcd}
\mathrm{List}\, A \arrow[r, "\eta_{\mathrm{List}\, A}"] \arrow[dr, "\mathrm{id}"'] & \mathrm{List}(\mathrm{List}\, A) \arrow[d, "\mu_A"] \\
& \mathrm{List}\, A
\end{tikzcd}
\qquad
\begin{tikzcd}
l \arrow[r, mapsto] \arrow[dr, mapsto] & {[l]} \arrow[d, mapsto] \\
& l = [l]_0
\end{tikzcd}\ ,$$

$$\begin{tikzcd}
\mathrm{List}(\mathrm{List}\, A) \arrow[d, "\mu_A"'] & \mathrm{List}\, A \arrow[l, "\mathrm{List}\, \eta_A"'] \arrow[dl, "\mathrm{id}"] \\
\mathrm{List}\, A &
\end{tikzcd}
\qquad
\begin{tikzcd}
{[[l_0], ..., [l_{\mathrm{len}(l)-1}]]} \arrow[d, mapsto] & l \arrow[l, mapsto] \arrow[dl, mapsto] \\
l = \sum_{i=0}^{\mathrm{len}(l)-1} [l_i] &
\end{tikzcd}\ ,$$

$$\begin{tikzcd}
\mathrm{List}^3\, A \arrow[r, "\mathrm{List}\, \mu_A"] \arrow[d, "\mu_{\mathrm{List}\, A}"'] & \mathrm{List}^2\, A \arrow[d, "\mu_A"] \\
\mathrm{List}^2\, A \arrow[r, "\mu_A"'] & \mathrm{List}\, A
\end{tikzcd}
\qquad
\begin{tikzcd}
l \arrow[r, mapsto] \arrow[d, mapsto] & {[\sum_{j=0}^{\mathrm{len}(l_0)-1} (l_0)_j, ..., \sum_{j=0}^{\mathrm{len}(l_{\mathrm{len}(l)-1}-1)} (l_{\mathrm{len}(l)-1})_j]} \arrow[d, mapsto] \\
\sum_{i=0}^{\mathrm{len}(l)-1} l_i \arrow[r, mapsto] & \sum_{i,j} (l_i)_j
\end{tikzcd}\ .$$

Kleisli category Kl(List)**.**
One of the possible interpretations of $f : A \to TB$ is a non-deterministic calculation: a is mapped by f to the list of all possible results. The composition corresponds to getting all possible results from sequential non-deterministic calculations:

$$\begin{tikzcd}
A \arrow[r, "f^\bullet"] \arrow[dr, "g^\bullet \circ f^\bullet"'] & B \arrow[d, "g^\bullet"] \\
& C
\end{tikzcd}
\qquad
\begin{tikzcd}
{[b_0, ..., b_n]} \arrow[d, mapsto, "{Tg=[g_0(b_0),...,g_n(b_n)]}"'] & & a \arrow[ll, mapsto, "f"'] \arrow[d, mapsto] \\
{[[b_0^0, b_1^0, ..., b_{m_0}^0], ...[b_0^n, b_1^n, ..., b_{m_n}^n]]} \arrow[rr, mapsto, "\mu_B = \sum_i g_i(b_i)"'] & & {[b_0^0, b_1^0, ..., b_{m_0}^0, ..., b_0^n, b_1^n, ..., b_{m_n}^n]}
\end{tikzcd}\ .$$

2.4.2 Exception monad

Definition and a monad structure.
Consider a type E of exceptions (error values) and a functor Exception : $\mathcal{S}et \to \mathcal{S}et$, which sends a type to its coproduct with the exceptions type:

$$\begin{tikzcd} A \arrow[d, "f"'] & A+E \arrow[d, "f+\mathrm{id}"] \\ B & B+E \end{tikzcd} .$$

The monad structure

$$(\mathrm{Exception}, \eta : \mathrm{id} \to \mathrm{Exception}, \mu : \mathrm{Exception}^2 \to \mathrm{Exception}).$$

is defined as follows:

- Unit represents successful calculation:

$$\eta_A := i_1 : A \to A + E,$$

- Multiplication represents the error propagation:

$$\mu_A : A + E + E \to A + E = \mathrm{id}_A + \nabla_{E,E} = \nabla_{A+E,E},$$

  where ∇ is the codiagonal map.

The diagrams are obvious on $\mathcal{S}$et:

$$\begin{tikzcd} A+E \arrow[r, "\eta_{A+E}=i_{1,A+E}"] \arrow[dr, "\mathrm{id}"'] & (A+E)+E \arrow[d, "\nabla_{A+E,E}"] \\ & A+E \end{tikzcd} \qquad \begin{tikzcd} a_A \mid\mid \mathrm{error}_E \arrow[r, mapsto] \arrow[dr, mapsto] & (a \mid\mid \mathrm{error})_{A+E} \arrow[d, mapsto] \\ & a_A \mid\mid \mathrm{error}_E \end{tikzcd} ,$$

$$\begin{tikzcd} (A+E)+E \arrow[d, "\nabla_{A+E,E}"'] & A+E \arrow[l, "\eta_A+\mathrm{id}_E=i_{1,A}+\mathrm{id}_E"'] \arrow[dl, "\mathrm{id}"] \\ A+E & \end{tikzcd} \qquad \begin{tikzcd} a_{A+E} \mid\mid \mathrm{error}_E \arrow[d, mapsto] & a_A \mid\mid \mathrm{error}_E \arrow[l, mapsto] \arrow[dl, mapsto] \\ a_A \mid\mid \mathrm{error}_E & \end{tikzcd} ,$$

$$\begin{tikzcd} ((A+E)+E)+E \arrow[r, "\nabla_{A+E,E}+\mathrm{id}_E"] \arrow[d, "\nabla_{(A+E)+E,E}"'] & (A+E)+E \arrow[d, "\nabla_{A+E,E}"] \\ (A+E)+E \arrow[r, "\nabla_{A+E,E}"'] & A+E \end{tikzcd} \qquad \begin{tikzcd} a_A \mid\mid \mathrm{error}_{E^{(1)}+E^{(2)}+E^{(3)}} \arrow[r, mapsto] \arrow[d, mapsto] & a_A \mid\mid \mathrm{error}_{E^{(2)}+E^{(3)}} \arrow[d, mapsto] \\ a_A \mid\mid \mathrm{error}_{E^{(1)}+E^{(2)}} \arrow[r, mapsto] & a_A \mid\mid \mathrm{error}_E \end{tikzcd} .$$

Here $a_A \mid\mid b_B$ informally denotes the element of $A + B$, subscripts annotate the domains.

Kleisli category Kl(Exception)**.**
The Kleisli composition for the Exception monad tracks the source of the possible error:

$$\begin{tikzcd} A \arrow[r, "f^\bullet"] \arrow[dr, "g^\bullet \circ f^\bullet"'] & B \arrow[d, "g^\bullet"] \\ & C \end{tikzcd} \qquad \begin{tikzcd} f(a)_B \mid\mid \mathrm{error}_E \arrow[d, mapsto, "g+\mathrm{id}_E"'] & & a_A \arrow[ll, mapsto, "f"'] \arrow[d, mapsto] \\ (gf(a)_C \mid\mid \mathrm{error})_{C+E} \mid\mid \mathrm{error}_E \arrow[rr, mapsto, "\mu_C"'] & & gf(a)_C \mid\mid \mathrm{error}_E \end{tikzcd} .$$

3 Comonads