# モナドって何だよ

型推栄 (@\_jj1lis\_uec)

2025年8月2日

### 1 はじめに

プログラミング言語におけるモナドって何? という直感が全然生えてこないので、まとめてみて理解を試みる. よかったら読んでみてください. あと間違いがあればご指摘お願いします.

本稿では以下の知識を仮定する.

- 圏論の入門書 (Mac Lane とか Leinster とか) の最初の数章を読んだことがある.
- Haskell について多少は知っている (でもモナドは知らない).
- 型について多少は知っている.

### 2 圏論におけるモナド

文献によって定義の揺れなどがあるだろうから、本稿で採用するモナドの定義を述べておく.

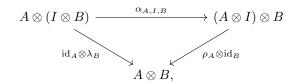
定義 1. (1) V を圏,  $I \in V$  をその対象, $\otimes: V \times V \to V$  を関手とする。 $\otimes (A,B) \in V$  は  $A \otimes B$  のようにも書く.次のような自然同型  $\alpha,\lambda,\rho$  が存在するとき,V はモノイダル(monoidal)であると言う.なお以下では対象  $I \in V$  と関手  $I: \mathbf{1} \to V$  と同一視していることに注意せよ.

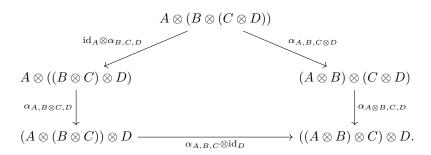
$$\alpha: \otimes \circ (\mathrm{id}_{\mathcal{V}} \times \otimes) \Rightarrow \otimes \circ (\otimes \times \mathrm{id}_{\mathcal{V}}), 
\lambda: \pi_{2} \Rightarrow I \times \mathrm{id}_{\mathcal{V}}, 
\rho: \pi_{1} \Rightarrow \mathrm{id}_{\mathcal{V}} \times I,$$

$$\mathcal{V} \times \mathcal{V} \times \mathcal{V} \xrightarrow{\otimes \times \mathrm{id}_{\mathcal{V}}} \mathcal{V} \times \mathcal{V} \qquad \mathbf{1} \times \mathcal{V} \xrightarrow{I \times \mathrm{id}_{\mathcal{V}}} \mathcal{V} \times \mathcal{V} \xleftarrow{\mathrm{id}_{\mathcal{V}} \times I} \mathcal{V} \times \mathbf{1}$$

$$\downarrow_{\mathrm{id}_{\mathcal{V}} \times \otimes} \downarrow \qquad \qquad \downarrow_{\otimes} \qquad \qquad \downarrow_{\otimes}$$

加えて, 任意の  $A,B,C,D\in\mathcal{V}$  に対して以下の図式は可換となる必要がある. ただしここで射  $\alpha_{(A,B,C)}$  を簡単に  $\alpha_{A,B,C}$  とも書いている:



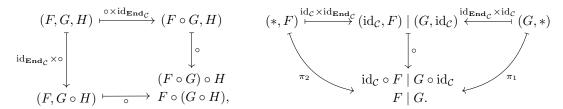


このとき I を単位対象 (unit object),  $\otimes$  をテンソル積 (tensor product) と呼ぶ. また  $\mathcal V$  を 6 つ組  $(\mathcal V,\otimes,I,\alpha,\lambda,\rho)$  として明記する場合もある.

(2)  $(V, \otimes, I, \alpha, \lambda, \rho)$  をモノイダル圏とする. V のモノイド対象  $(monoid\ object)$  ないし単にモノイドとは、対象  $M \in V$ 、乗法 (multiplication) と呼ばれる射  $\mu \colon M \otimes M \to M$ 、単位 (unit) と呼ばれる射 $\eta \colon I \to M$  から成る 3 つ組  $(M, \mu, \eta)$  であって、以下の図式を可換にするもののことである.

(3) モナド (monad) とは、自己関手の圏におけるモノイド対象である.

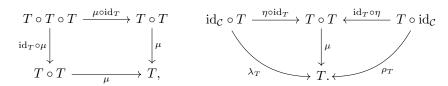
特にモナドについて補足しよう.所与の圏  $\mathcal C$  に対して,関手圏  $[\mathcal C,\mathcal C]$ —以後は  $\mathbf{End}_{\mathcal C}$  と書く—は,テンソル積として関手の合成。,単位対象として恒等関手  $\mathrm{id}_{\mathcal C}$  を取ることでモノイダル圏をなす.以下の図式を見れば分かる通り,定義  $\mathbf 1$  (1) における自然変換  $\alpha,\lambda,\rho$  はすべて恒等自然変換となる.



このとき  $\mathbf{End}_{\mathcal{C}}$  の対象(すなわち関手) $T: \mathcal{C} \to \mathcal{C}$  と射(すなわち自然変換) $\mu: T \circ T \Rightarrow T$  および  $\eta: \mathrm{id}_{\mathcal{C}} \Rightarrow T$  の組  $(T, \mu, \eta)$  は,以下の条件を満たすとき  $\mathbf{End}_{\mathcal{C}}$  のモノイド対象となる.

$$\begin{split} \mu \cdot (T\mu) &= \mu \cdot (\mu T), \\ \mu \cdot (T\eta) &= \mu \cdot (\eta T) = \mathrm{id}_T. \end{split}$$

これはまさに、図式 2.2 と同じことを言っているに過ぎない.



ただしここで次のような記法を用いはじめた.まず・は  $\mathbf{End}_{\mathcal{C}}$  における射の合成である.また  $T\mu$  は各対象  $A\in\mathcal{C}$  について

$$(T\mu)_A = T(\mu_A) \colon T \circ T \circ T(A) \to T \circ T(A)$$

とすることで定義される自然変換  $T\circ T\circ T\Rightarrow T\circ T$  を表す. 同様に  $\mu T\colon T\circ T\circ T\Rightarrow T\circ T$  で  $(\mu T)_A=\mu_{T(A)}$  により定まる自然変換を表す.この  $(T,\mu,\eta)$  が  $\mathcal C$  におけるモナドである.

# 3 Haskell におけるモナド

ようやく本題に入ろう. 関数型プログラミング言語は数多あるが,本稿では Haskell を用いて説明を行う. Haskell の型を対象,関数を射と見て,その全体を Hask と書くことにする. 実は Hask は圏である.

### 注意 2.