Haskell におけるモナドとアローの同値性

Ziphil Aleshlas

2019 年 3 月 ? 日

1. 初めに

副作用のある処理を純粋な関数として扱うための方法には，古くから知られているモナドを用いるものと，Hughes[1] によって提唱されたアローを用いるものがある．関数型プログラミング言語の Haskell ではともに型クラスとして実装されており，モナドは `Monad` クラスとして，アローは `Arrow` クラスとして利用できる．

モナドとアローという 2 つの概念の間には密接な関係があり，計算の実行そのものも計算になる (この計算は app で表される) という構造をアローに付け加えることで，モナドと等価なものが得られることが知られている．このノートでは，この「等価である」という関係を型理論的に定式化し，モナドと app を加えたアローとが等価であることを形式的に証明することを目標とする．この証明の方針は，Lindley–Wadler–Yallop[2] を参考にした．

なお，Haskell に関する予備知識は要求しない (知っていると理解しやすいという程度) が，型理論に関する細かい説明は省くので，型理論へのある程度の慣れは要求する．

2. 定義

2.1. 型理論としての定式化

まず，モナドとアローを等号付きの型理論として定式化するため，そのベースとなる理論を用意する．ここでは，通常の単純型付きラムダ計算の体系に直積 (タプル型) を追加したものを用いることにする．

定義 1. 理論 $\lambda_{1\times}$ を以下のように定義する．まず，型 $A, B, \cdots$ および項 $M, N, \cdots$ を，

$$\begin{aligned} A, B &::= \alpha \mid 1 \mid A \times B \mid A \to B \\ M, N &::= a \mid \star \mid \langle M, N \rangle \mid \mathtt{fst}\, M \mid \mathtt{snd}\, M \mid \lambda a.\, M \mid MN \end{aligned}$$

によって定める．ここで，α と a はそれぞれ原子型と変項を表している．次に，型割り当ての推論規

則は,

$$\frac{}{\Gamma,\, a\colon A \vdash a\colon A} \qquad \frac{}{\Gamma \vdash \star\colon 1}$$

$$\frac{\Gamma \vdash M\colon A \quad \Gamma \vdash N\colon B}{\Gamma \vdash \langle M,\, N\rangle\colon A \times B} \qquad \frac{\Gamma \vdash M\colon A \times B}{\Gamma \vdash \mathtt{fst}\, M\colon A} \qquad \frac{\Gamma \vdash M\colon A \times B}{\Gamma \vdash \mathtt{snd}\, M\colon B}$$

$$\frac{\Gamma,\, a\colon A \vdash M\colon B}{\Gamma \vdash \lambda a.\, M\colon A \to B} \qquad \frac{\Gamma \vdash M\colon A \to B \quad \Gamma \vdash N\colon A}{\Gamma \vdash MN\colon B}$$

とする．ここで，型割り当ての左辺に出てくる型環境は変項と型のペアの集合であるとし，順序や個数の区別はしないこととする．最後に，同じ型をもつ項の間の等式を,

$$\begin{aligned}
\star &= M \\
\mathtt{fst}\langle M,\, N\rangle &= M \\
\mathtt{snd}\langle M,\, N\rangle &= N \\
\langle \mathtt{fst}\, M,\, \mathtt{snd}\, M\rangle &= M \\
(\lambda a.\, M)N &= M[a := N] \\
\lambda a.\, Ma &= M
\end{aligned}$$

で定め，合同関係になるように拡張する．なお，最後の等式において M は自由変項に a を含まないものとする．

記号に関しては，Haskell で用いられているものではなく数学の慣習に従っている．例えば，2 つの型 A, B のタプルの型は $A \times B$ のように積の形で書き，ユニット型は 1 で表すことにしている．

これ以降，簡単のため以下の略記を用いることがある．

$$\begin{aligned}
\mathtt{id} &: A \to A \\
&\equiv \lambda a.\, a \\
\mathtt{swap} &: A \times B \to B \times A \\
&\equiv \lambda p.\, \langle \mathtt{snd}\, p,\, \mathtt{fst}\, p\rangle \\
\mathtt{assoc} &: (A \times B) \times C \to A \times (B \times C) \\
&\equiv \lambda p.\, \langle \mathtt{fst}(\mathtt{fst}\, p),\, \langle \mathtt{snd}(\mathtt{fst}\, p),\, \mathtt{snd}\, p\rangle\rangle \\
\langle \circ \rangle &: (B \to C) \to (A \to B) \to A \to C \\
&\equiv \lambda g.\, \lambda f.\, \lambda a.\, g(f\, a) \\
\langle \divideontimes \rangle &: (A \to C) \to (B \to D) \to A \times B \to C \times D \\
&\equiv \lambda f.\, \lambda g.\, \lambda p.\, \langle f(\mathtt{fst}\, p),\, g(\mathtt{snd}\, p)\rangle
\end{aligned}$$

`id` は恒等関数であり，`swap` はタプルの要素の左右を反転する関数で，`assoc` は二重のタプルの結合を変える関数である．これら全て，同名の関数が Haskell でも定義されている．さらに，$\circ$ は関数合成をする演算子で，$\divideontimes$ は 2 つの関数をタプルの要素それぞれに適用する演算子である．Haskell での記法に倣って，記号を引数に中置するときはそのまま書き，普通の関数のように引数に前置するときは括弧で囲むことにする．

次に，モナドとアローのそれぞれを扱うために，この体系を拡張する．

定義 2. 理論 **Monad** を以下のように定義する．まず，型 $A, B, \cdots$ および項 $M, N, \cdots$ を，$\lambda_{1\times}$ のものに加え，

$$A, B ::= \cdots \mid \#A$$
$$M, N ::= \cdots \mid \mathtt{return} \mid \langle \rightarrowtail\ll \rangle$$

と拡張する．ここで追加した 2 つの定数は，

$$\frac{}{\Gamma \vdash \mathtt{return} : A \to \#A} \qquad \frac{}{\Gamma \vdash \langle \rightarrowtail\ll \rangle : (A \to \#B) \to \#A \to \#B}$$

という多相型をもつ．さらに，3 つの等式

$$F \rightarrowtail\ll \mathtt{return}\, M = FM \tag{M_1}$$
$$\mathtt{return} \rightarrowtail\ll M = M \tag{M_2}$$
$$G \rightarrowtail\ll (F \rightarrowtail\ll M) = (\lambda a.\, G \rightarrowtail\ll Fa) \rightarrowtail\ll M \tag{M_3}$$

を満たすとする．

定義 3. 理論 **ArrowApply** を以下のように定義する．まず，型 $A, B, \cdots$ および項 $M, N, \cdots$ を，$\lambda_{1\times}$ のものに加え，

$$A, B ::= \cdots \mid A \rightsquigarrow B$$
$$M, N ::= \cdots \mid \mathtt{arr} \mid \langle \lll \rangle \mid \mathtt{first} \mid \mathtt{app}$$

と拡張する．ここで追加した 4 つの定数は，

$$\frac{}{\Gamma \vdash \mathtt{arr} : (A \to B) \to (A \rightsquigarrow B)}$$
$$\frac{}{\Gamma \vdash \langle \lll \rangle : (B \rightsquigarrow C) \to (A \rightsquigarrow B) \to (A \rightsquigarrow C)} \qquad \frac{}{\Gamma \vdash \mathtt{first} : (A \rightsquigarrow B) \to (A \times C \rightsquigarrow B \times C)}$$
$$\frac{}{\Gamma \vdash \mathtt{app} : (A \rightsquigarrow B) \times A \rightsquigarrow B}$$

という多相型をもつ．さらに，12 個の等式

$$\mathtt{arr}\,\mathtt{id} \lll F = F \tag{C_1}$$
$$F \lll \mathtt{arr}\,\mathtt{id} = F \tag{C_2}$$
$$H \lll (G \lll F) = (H \lll G) \lll F \tag{C_3}$$
$$\mathtt{arr}(G \circ F) = \mathtt{arr}\, G \lll \mathtt{arr}\, F \tag{A_1}$$
$$\mathtt{first}(\mathtt{arr}\, F) = \mathtt{arr}(F \gg\!\!* \mathtt{id}) \tag{A_2}$$
$$\mathtt{first}(G \lll F) = \mathtt{first}\, G \lll \mathtt{first}\, F \tag{A_3}$$
$$\mathtt{arr}\,\mathtt{fst} \lll \mathtt{first}\, F = F \lll \mathtt{arr}\,\mathtt{fst} \tag{A_4}$$
$$\mathtt{arr}(\mathtt{id} \gg\!\!* G) \lll \mathtt{first}\, F = \mathtt{first}\, F \lll \mathtt{arr}(\mathtt{id} \gg\!\!* G) \tag{A_5}$$
$$\mathtt{arr}\,\mathtt{assoc} \lll \mathtt{first}(\mathtt{first}\, F) = \mathtt{first}\, F \lll \mathtt{arr}\,\mathtt{assoc} \tag{A_6}$$
$$\mathtt{app} \lll \mathtt{first}(\mathtt{arr}(\lambda a.\ \mathtt{arr}(\lambda b.\, \langle a, b \rangle))) = \mathtt{arr}\,\mathtt{id} \tag{P_1}$$

$$\mathtt{app} \lll \mathtt{first}(\mathtt{arr}\langle \lll F\rangle) = \mathtt{app} \lll \mathtt{second}\, F \qquad [\mathrm{P}_2]$$
$$\mathtt{app} \lll \mathtt{first}(\mathtt{arr}\langle F \lll\rangle) = F \lll \mathtt{app} \qquad [\mathrm{P}_3]$$

を満たすとする．なお，式中に出てくる $\langle \lll F\rangle$ や $\langle F \lll\rangle$ は，Haskell でも使われる演算子のセクションを表し，それぞれ $\lambda h.\, h \lll F$ と $\lambda h.\, F \lll h$ の略記である．

モナドとアローはそれぞれ $\#$ と $\rightsquigarrow$ で表した．また，関数適用の記号と順番を揃えるため，モナドやアローの合成は全て左向きにした．

以下の略記も用いる[*1]．second については，上の定義中でも一度用いている．

$$\begin{aligned}\langle \lll\rangle &: (B \to \#C) \to (A \to \#B) \to (A \to \#C)\\ &\equiv \lambda g.\, \lambda f.\, \lambda a.\, g \gg\!= f\, a\\ \mathtt{second} &: (A \rightsquigarrow B) \to (C \times A \rightsquigarrow C \times B)\\ &\equiv \lambda f.\, \mathtt{arr}\, \mathtt{swap} \lll \mathtt{first}\, f \lll \mathtt{arr}\, \mathtt{swap}\end{aligned}$$

これらの演算子や関数も，同名のものが Haskell に定義されている．

理論と Haskell との対応は，次のように述べることができる．ベースとして定義した $\lambda_{1\times}$ は，型クラスのない Haskell に対応する．この $\lambda_{1\times}$ を拡張した理論は，Haskell の型クラスに対応する．型クラスとは既存の型に構造を追加するものだと捉えることができるが，構成できる型や項を $\lambda_{1\times}$ から拡張することで，この追加された構造を扱っているのである．例えば，Haskell においてモナドを表す型クラスには return と $\gg\!=$ が定義されているが，それを型理論的に写し取ったのが **Monad** である．

2.2. 理論の変換と同値性

続いて，2 つの理論の同値性を定義する．まず，ある理論を別の理論に変換することを考える．

定義 4. 等号理論 $\mathfrak{T}$, $\mathfrak{U}$ に対し，$\mathfrak{T}$ から $\mathfrak{U}$ への**変換** (translation) とは，

- $\mathfrak{T}$ の型 A に対し，$\mathfrak{U}$ の型 $[\![A]\!]$ が定まっている．
- $\mathfrak{T}$ の項 M に対し，$\mathfrak{U}$ の項 $[\![M]\!]$ が定まっている．

という情報から成り，2 つの条件

- $\mathfrak{T}$ での型割り当て $\Gamma \vdash_{\mathfrak{T}} M\colon A$ に対し，$\mathfrak{U}$ での型割り当て $[\![\Gamma]\!] \vdash_{\mathfrak{U}} [\![M]\!]\colon [\![A]\!]$ が成立する．
- $\mathfrak{T}$ での等式 $\Gamma \vdash_{\mathfrak{T}} M = N\colon A$ に対し，$\mathfrak{U}$ での型割り当て $[\![\Gamma]\!] \vdash_{\mathfrak{U}} [\![M]\!] = [\![N]\!]\colon [\![A]\!]$ が成立する．

を満たさなければならない．ここで，型環境の変換は，

$$[\![x_1\colon C_1, \cdots, x_n\colon C_n]\!] \equiv x_1\colon [\![C_1]\!], \cdots, x_n\colon [\![C_n]\!]$$

によって定まるものとする．

[*1] $\lll$ おさかな．$\lll\!\sim$．

2 つの理論 $\mathfrak{T}, \mathfrak{U}$ があるとき，その間の変換 $[\![\text{-}]\!]\colon \mathfrak{T} \to \mathfrak{U}$ とは，Haskell において $\mathfrak{T}$ で表される型クラスが $\mathfrak{U}$ で表される型クラスを拡張 (もしくは実装) しているのだと解釈できる．例えば，型 R を 1 つ固定すれば，定義域が R の関数の型 $R \to \text{-}$ はモナドになる．実際，Haskell でもそのようなインスタンスの定義がなされている．このことを型理論的には，変換 $[\![\text{-}]\!]\colon \mathbf{Monad} \to \lambda_{1\times}$ であって，

$$\begin{aligned}
[\![\texttt{return}]\!] &: [\![A]\!] \to R \to [\![A]\!] \\
&\equiv \lambda a.\, \lambda r.\, a \\
[\![\langle \mathbin{=\!\!\ll} \rangle]\!] &: ([\![A]\!] \to R \to [\![B]\!]) \to (R \to [\![A]\!]) \to R \to [\![B]\!] \\
&\equiv \lambda f.\, \lambda h.\, \lambda r.\, f(hr)r
\end{aligned}$$

を満たすようなものが存在すると述べることができる．

2 つの理論が同値であるとは，双方向の変換があって，2 回変換すると本質的に同じものに戻ってくることとして定義される．ここで「本質的に」と言ったのは，全く同じものに戻ってくるとしてしまうと条件が強すぎるので，型の同型の違いのズレを許すことにするという意味である[*2]．

定義 5. 理論 $\mathfrak{T}$ の型 A, B に対し，A と B が**同型** (isomorphic) であるとは，以下の状況を満たすことである．すなわち，2 つの閉項 $\Phi\colon A \to B, \bar{\Phi}\colon B \to A$ があって，任意の型割り当て $\Gamma \vdash M\colon A$ および $\Delta \vdash N\colon B$ に対し，

$$\begin{aligned}
\Gamma &\vdash \bar{\Phi}(\Phi M) = M\colon A \\
\Delta &\vdash \Phi(\bar{\Phi} N) = N\colon B
\end{aligned}$$

が成り立つ．

定義 6. 理論 $\mathfrak{T}, \mathfrak{U}$ に対し，$\mathfrak{T}$ と $\mathfrak{U}$ が**同値** (equivalent) であるとは，以下の状況を満たすことである．すなわち，双方向の変換 $[\![\text{-}]\!]\colon \mathfrak{T} \to \mathfrak{U}, \{\!|\text{-}|\!\}\colon \mathfrak{U} \to \mathfrak{T}$ があって，4 つの条件

- 任意の $\mathfrak{T}$ の型 A に対し，型の同型 $\Phi_A\colon A \to \{\!|[\![A]\!]|\!\}$ が存在する．
- 任意の $\mathfrak{U}$ の型 B に対し，型の同型 $\Psi_B\colon B \to [\![\{\!|B|\!\}]\!]$ が存在する．
- 任意の $\mathfrak{T}$ の型割り当て $\Gamma \vdash_{\mathfrak{T}} M\colon A$ に対し，

  $$\Gamma \vdash_{\mathfrak{T}} \{\!|[\![M]\!]|\!\}[\Gamma := \Phi\Gamma] = \Phi_A M\colon \{\!|[\![A]\!]|\!\}$$

  が成り立つ．なお，上式における代入 $[\Gamma := \Phi\Gamma]$ は，$\Gamma \equiv x_1\colon C_1, \cdots, x_n\colon C_n$ とおくとき，

  $$[\Gamma := \Phi\Gamma] \equiv [x_1 := \Phi_{C_1} x_1, \cdots, x_n := \Phi_{C_n} x_n]$$

  を意味する．
- 任意の $\mathfrak{U}$ の型割り当て $\Delta \vdash_{\mathfrak{U}} N\colon B$ に対し，

  $$\Delta \vdash_{\mathfrak{U}} [\![\{\!|N|\!\}]\!][\Delta := \Psi\Delta] = \Psi_B N\colon [\![\{\!|B|\!\}]\!]$$

  が成り立つ．代入 $[\Delta := \Psi\Delta]$ の意味は，上と同様である．

[*2] 圏論を知っているのならば，圏同型ではなく圏同値に相当する概念だと言えば分かりやすいだろう．

が全て成り立つ．

3 つ目と 4 つ目の条件について少し補足をしておこう．型割り当て $\Gamma \vdash M\colon A$ があったときに，同じ型環境のもと型 $\{\!|\llbracket A \rrbracket|\!\}$ の項を作る方法は 2 通りある．まずは単純に M に Φ_A を適用して，$\Phi_A M$ を得る方法である．一方，M に $\{\!|\llbracket\text{-}\rrbracket|\!\}$ を施すことで型割り当て $\{\!|\llbracket\Gamma\rrbracket|\!\} \vdash \{\!|\llbracket M\rrbracket|\!\}\colon \{\!|\llbracket A\rrbracket|\!\}$ が得られるので，ここから $[\Gamma := \Phi\Gamma]$ という代入を行って型環境を Γ に戻すという方法も可能である．上の定義の 3 つ目と 4 つ目の条件は，この 2 通りの方法で作った項が等しくなることを保証しているのである[*3]．なお，以降ではこの条件を Φ や Ψ の自然性と呼ぶ．

以上によって，理論が同値とはどういった意味なのかが数学的に定式化できた．次の節で，**Monad** と **ArrowApply** がこの意味で同値になることを証明する．

3. 証明

3.1. 方針

このノートで示したい主張は以下のように述べられる．

定理 7. 2 つの理論 **Monad**, **ArrowApply** は同値である．

この定理を，以下の 3 つの補題に分けて証明する．それぞれを続く 3 つの節で証明していく．

補題 8. 理論の変換 $\llbracket\text{-}\rrbracket\colon$ **Monad** $\to$ **ArrowApply** が存在する．

補題 9. 理論の変換 $\{\!|\text{-}|\!\}\colon$ **ArrowApply** $\to$ **Monad** が存在する．

補題 10. 上で得られた 2 つの変換 $\llbracket\text{-}\rrbracket\colon$ **Monad** $\to$ **ArrowApply**, $\{\!|\text{-}|\!\}\colon$ **ArrowApply** $\to$ **Monad** は理論同値を与える．

3.2. モナドからアローへ

まずは，**Monad** から **ArrowApply** への変換 $\llbracket\text{-}\rrbracket\colon$ **Monad** $\to$ **ArrowApply** を定義する．Haskell の言葉で述べれば，`ArrowApply` から `Monad` のインスタンスを作るということになる．Haskell では，これは `ArrowMonad` クラスを介して実現している．この実装を型理論的に写し取れば良い．

$\llbracket\text{-}\rrbracket$ の型に関する対応は，

$$\llbracket\alpha\rrbracket \equiv \alpha$$
$$\llbracket 1\rrbracket \equiv 1$$
$$\llbracket A \times B\rrbracket \equiv \llbracket A\rrbracket \times \llbracket B\rrbracket$$

[*3] 圏論的には，$\mathfrak{T}, \mathfrak{U}$ が圏で，$\llbracket\text{-}\rrbracket, \{\!|\text{-}|\!\}$ はその間の圏同値を与える関手であり，Φ, Ψ はその圏同値に付随する自然同型だと捉えられる．3 つ目と 4 つ目の条件は，Φ や Ψ が自然変換になっていると述べているにすぎない．

$$[\![A \to B]\!] \equiv [\![A]\!] \to [\![B]\!]$$
$$[\![\#A]\!] \equiv 1 \rightsquigarrow [\![A]\!]$$

と定める．さらに，[\![-]\!] の項に関する対応は，

$$\begin{aligned}
[\![\mathtt{return}]\!] &: [\![A]\!] \to (1 \rightsquigarrow [\![A]\!]) \\
&\equiv \lambda a.\ \mathtt{arr}(\lambda u.\ a) \\
[\![\langle \gg\!= \rangle]\!] &: ([\![A]\!] \to (1 \rightsquigarrow [\![B]\!])) \to (1 \rightsquigarrow [\![A]\!]) \to (1 \rightsquigarrow [\![B]\!]) \\
&\equiv \lambda f.\, \lambda s.\, \mathtt{app} \lll \mathtt{arr}(\lambda a.\, \langle f\, a, \star \rangle) \lll s
\end{aligned}$$

とする．これ以外の形の項については，自明な対応を用いる[*4]．

これが等号理論の変換になるには，定義 4 の 2 つの条件を満たさなければならない．1 つ目の条件 (型を保つこと) は定義から明らかなので，2 つ目の条件 (等式を保つこと) を確かめる．そのためには，**Monad** に追加で定まっている 3 つの等式 M_1, M_2, M_3 の両辺に [\![-]\!] を施したものが，**ArrowApply** でも等しくなっていることを示せば十分である．

≡ ˈ ω ˈ ≡ がんばれ!

以上により，補題 8 が示された．

3.3. アローからモナドへ

次は逆に，**ArrowApply** から **Monad** への変換 {\|-\|}: **ArrowApply** → **Monad** を作る．Haskell においては，`Monad` から `ArrowApply` のインスタンスを作るということに対応するが，これは `Kleisli` クラスを介して行われている．ここでも，その実装を写し取れば良い．

{\|-\|} の型に関する対応は，

$$\{\!|A \rightsquigarrow B|\!\} \equiv \{\!|A|\!\} \to \#\{\!|B|\!\}$$

と定め，{\|-\|} の項に関する対応は，

$$\begin{aligned}
\{\!|\mathtt{arr}|\!\} &: (\{\!|A|\!\} \to \{\!|B|\!\}) \to (\{\!|A|\!\} \to \#\{\!|B|\!\}) \\
&\equiv \lambda f.\, \mathtt{return} \circ f \\
\{\!|\langle \lll \rangle|\!\} &: (\{\!|B|\!\} \to \#\{\!|C|\!\}) \to (\{\!|A|\!\} \to \#\{\!|B|\!\}) \to (\{\!|A|\!\} \to \#\{\!|C|\!\}) \\
&\equiv \lambda g.\, \lambda f.\, g \lll f \\
\{\!|\mathtt{first}|\!\} &: (\{\!|A|\!\} \to \#\{\!|B|\!\}) \to \{\!|A|\!\} \times \{\!|C|\!\} \to \#(\{\!|B|\!\} \times \{\!|C|\!\}) \\
&\equiv \lambda f.\, \lambda p.\, (\lambda b.\ \mathtt{return}\langle b, \mathtt{snd}\, p\rangle) \lll f(\mathtt{fst}\, p) \\
\{\!|\mathtt{app}|\!\} &: (\{\!|A|\!\} \to \#\{\!|B|\!\}) \times \{\!|A|\!\} \to \#\{\!|B|\!\} \\
&\equiv \lambda p.\, (\mathtt{fst}\, p)(\mathtt{snd}\, p)
\end{aligned}$$

とする．これ以外の形の型や項については，自明な対応とする．

[*4] 例えば，$[\![\mathtt{fst}\, M]\!] \equiv \mathtt{fst}[\![M]\!]$ などである．

これが等号理論の変換となることを確かめるには，前節の場合と同様に，**ArrowApply** に追加で定まっている 12 個の等式 $C_1, \cdots, P_3$ の両辺に $\{\!|\text{-}|\!\}$ を施したものが，**Monad** でも等しくなっていることを示せば良い．

≡ ˈ ω ˈ ≡ ぼくがやるの?

以上により，補題 9 が示された．

3.4. 同値性

最後に，ここまでで構成した 2 つの変換 $[\![\text{-}]\!]$: **Monad** → **ArrowApply**, $\{\!|\text{-}|\!\}$: **ArrowApply** → **Monad** が理論同値を与えることを示す．

まず，**Monad** の各型 $\tilde{A}$ に対し，型同型を与える項 $\Phi_{\tilde{A}}\colon \tilde{A} \to \{\!|[\![\tilde{A}]\!]|\!\}$, $\bar{\Phi}_{\tilde{A}}\colon \{\!|[\![\tilde{A}]\!]|\!\} \to \tilde{A}$ を構成する．これは相互再帰的に，

$$
\begin{aligned}
\Phi_\alpha &: \alpha \to \alpha & \bar{\Phi}_\alpha &: \alpha \to \alpha \\
&\equiv \mathtt{id} & &\equiv \mathtt{id} \\
\Phi_1 &: 1 \to 1 & \bar{\Phi}_1 &: 1 \to 1 \\
&\equiv \mathtt{id} & &\equiv \mathtt{id} \\
\Phi_{A\times B} &: A \times B \to \{\!|[\![A]\!]|\!\} \times \{\!|[\![B]\!]|\!\} & \bar{\Phi}_{A\times B} &: \{\!|[\![A]\!]|\!\} \times \{\!|[\![B]\!]|\!\} \to A \times B \\
&\equiv \lambda p.\, \langle \Phi_A(\mathtt{fst}\, p), \Phi_B(\mathtt{snd}\, p)\rangle & &\equiv \lambda p.\, \langle \bar{\Phi}_A(\mathtt{fst}\, p), \bar{\Phi}_B(\mathtt{snd}\, p)\rangle \\
\Phi_{A\to B} &: (A \to B) \to \{\!|[\![A]\!]|\!\} \to \{\!|[\![B]\!]|\!\} & \bar{\Phi}_{A\to B} &: (\{\!|[\![A]\!]|\!\} \to \{\!|[\![B]\!]|\!\}) \to A \to B \\
&\equiv \lambda f.\, \lambda a.\, \Phi_B(f(\bar{\Phi}_A a)) & &\equiv \lambda f.\, \lambda a.\, \bar{\Phi}_B(f(\Phi_A a)) \\
\Phi_{\#A} &: \#A \to 1 \to \#\{\!|[\![A]\!]|\!\} & \bar{\Phi}_{\#A} &: (1 \to \#\{\!|[\![A]\!]|\!\}) \to \#A \\
&\equiv \lambda x.\, \lambda u.\, \mathtt{return} \circ \Phi_A \mathbin{=\!\!\ll} x & &\equiv \lambda f.\, \mathtt{return} \circ \bar{\Phi}_A \mathbin{=\!\!\ll} f\star
\end{aligned}
$$

と定義する．これらが互いに逆になっていることは容易に確かめられる．

続いて，**Monad** の各項 $\tilde{M}$ に対する Φ の自然性を確かめる．$\tilde{M}$ が $\mathtt{return}$, $\langle \mathbin{=\!\!\ll} \rangle$ のどちらかである場合のみ非自明なので，この場合のみを考える．

$\mathtt{return}$ の場合は，まず，

$$
\begin{aligned}
\{\!|[\![\mathtt{return}]\!]|\!\} &= \{\!|\lambda a.\ \mathtt{arr}(\lambda u.\, a)|\!\} \\
&= \lambda a.\, \mathtt{return} \circ (\lambda u.\, a) \\
&= \lambda a.\, \lambda u.\ \mathtt{return}\, a
\end{aligned}
$$

が成り立つ．一方で，

$$
\begin{aligned}
\Phi_{A\to \#A}\, \mathtt{return} &= \lambda a.\, \Phi_{\#A}(\mathtt{return}(\bar{\Phi}_A a)) \\
&= \lambda a.\, \lambda u.\, \mathtt{return} \circ \Phi_A \mathbin{=\!\!\ll} \mathtt{return}(\bar{\Phi}_A a) \\
&= \lambda a.\, \lambda u.\, (\mathtt{return} \circ \Phi_A)(\bar{\Phi}_A a) \\
&= \lambda a.\, \lambda u.\ \mathtt{return}\, a
\end{aligned}
$$

となるので，上の計算結果と一致する．

$\langle \twoheadrightarrow\ll \rangle$ の場合は，まず，

$$\begin{aligned}
\{\!|[\![\langle \twoheadrightarrow\ll \rangle]\!]|\!\} &= \{\!|\lambda f.\, \lambda g.\ \mathtt{app} \lll \mathtt{arr}(\lambda a.\, \langle f a, \star\rangle) \lll g|\!\} \\
&= \lambda f.\, \lambda g.\, \{\!|\mathtt{app}|\!\} \lll\!\ll \{\!|\mathtt{arr}(\lambda a.\, \langle f a, \star\rangle)|\!\} \lll\!\ll g \\
&= \lambda f.\, \lambda g.\, (\lambda p.\, (\mathtt{fst}\, p)(\mathtt{snd}\, p)) \lll\!\ll (\lambda a.\ \mathtt{return}\langle f a, \star\rangle) \lll\!\ll g \\
&= \lambda f.\, \lambda g.\, (\lambda a.\, (\lambda p.\, (\mathtt{fst}\, p)(\mathtt{snd}\, p)) \twoheadrightarrow\!\ll \mathtt{return}\langle f a, \star\rangle) \lll\!\ll g \\
&= \lambda f.\, \lambda g.\, (\lambda a.\, f a \star) \lll\!\ll g \\
&= \lambda f.\, \lambda g.\, \lambda u.\, (\lambda a.\, f a \star) \twoheadrightarrow\!\ll g u \\
&= \lambda f.\, \lambda g.\, \lambda u.\, f' \twoheadrightarrow\!\ll g \star
\end{aligned}$$

と計算できる．なお，最後の等式において $f' \equiv \lambda a.\, f a \star$ とおいた．さて一方で，

$$\begin{aligned}
\Phi_{(A\to\#B)\to\#A\to\#B}\langle \twoheadrightarrow\ll \rangle &= \lambda f.\, \Phi_{\#A\to\#B}(\langle \twoheadrightarrow\ll \rangle(\bar{\Phi}_{A\to\#B} f)) \\
&= \lambda f.\, \lambda g.\, \Phi_{\#B}(\langle \twoheadrightarrow\ll \rangle(\bar{\Phi}_{A\to\#B} f)(\bar{\Phi}_{\#A} g)) \\
&= \lambda f.\, \lambda g.\, \Phi_{\#B}(\bar{\Phi}_{A\to\#B} f \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g) \\
&= \lambda f.\, \lambda g.\, \lambda u.\, \mathtt{return} \circ \Phi_B \twoheadrightarrow\!\ll \bar{\Phi}_{A\to\#B} f \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g
\end{aligned}$$

が成り立つ．ここで，

$$\begin{aligned}
\bar{\Phi}_{A\to\#B} f \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g &= (\lambda a.\, \bar{\Phi}_{\#B}(f(\Phi_A a))) \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g \\
&= (\lambda a.\, \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll f(\Phi_A a)\star) \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g \\
&= (\lambda a.\, \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll (f' \circ \Phi_A) a) \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g \\
&= \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll f' \circ \Phi_A \twoheadrightarrow\!\ll \bar{\Phi}_{\#A} g \\
&= \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll f' \circ \Phi_A \twoheadrightarrow\!\ll \mathtt{return} \circ \bar{\Phi}_A \twoheadrightarrow\!\ll g\star \\
&= \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll (\lambda a.\, f' \circ \Phi_A \twoheadrightarrow\!\ll \mathtt{return}(\bar{\Phi}_A a)) \twoheadrightarrow\!\ll g\star \\
&= \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll f' \twoheadrightarrow\!\ll g\star
\end{aligned}$$

であるから，上の式に代入して，

$$\begin{aligned}
\Phi_{(A\to\#B)\to\#A\to\#B}\langle \twoheadrightarrow\ll \rangle &= \lambda f.\, \lambda g.\, \lambda u.\, \mathtt{return} \circ \Phi_B \twoheadrightarrow\!\ll \mathtt{return} \circ \bar{\Phi}_B \twoheadrightarrow\!\ll f' \twoheadrightarrow\!\ll g\star \\
&= \lambda f.\, \lambda g.\, \lambda u.\, (\lambda b.\, \mathtt{return} \circ \Phi_B \twoheadrightarrow\!\ll \mathtt{return}(\bar{\Phi}_B b)) \twoheadrightarrow\!\ll f' \twoheadrightarrow\!\ll g\star \\
&= \lambda f.\, \lambda g.\, \lambda u.\, \mathtt{return} \twoheadrightarrow\!\ll f' \twoheadrightarrow\!\ll g\star \\
&= \lambda f.\, \lambda g.\, \lambda u.\, f' \twoheadrightarrow\!\ll g\star
\end{aligned}$$

を得る．これは最初に計算した $\{\!|[\![\langle \twoheadrightarrow\ll \rangle]\!]|\!\}$ の結果と一致する．

次に，**ArrowApply** の各型 $\tilde{A}$ に対し，型同型を与える項 $\Psi_{\tilde{A}}\colon \tilde{A} \to [\![\{\!|\tilde{A}|\!\}]\!]$, $\bar{\Psi}_{\tilde{A}}\colon [\![\{\!|\tilde{A}|\!\}]\!] \to \tilde{A}$ を，

$$\begin{aligned}
\Psi_{A\rightsquigarrow B} &:\ (A \rightsquigarrow B) \to [\![\{\!|A|\!\}]\!] \to (1 \rightsquigarrow [\![\{\!|B|\!\}]\!]) \\
&\quad \equiv \lambda s.\, \lambda a.\ \mathtt{arr}\, \Psi_B \lll s \lll \mathtt{arr}(\lambda u.\, \bar{\Psi}_A a) \\
\bar{\Psi}_{A\rightsquigarrow B} &:\ ([\![\{\!|A|\!\}]\!] \to (1 \rightsquigarrow [\![\{\!|B|\!\}]\!])) \to (A \rightsquigarrow B) \\
&\quad \equiv \lambda f.\ \mathtt{arr}\, \bar{\Psi}_B \lll \mathtt{app} \lll \mathtt{arr}(\lambda a.\, \langle f(\Psi_A a), \star\rangle)
\end{aligned}$$

によって定義する．$\tilde{A}$ が $A \rightsquigarrow B$ という形以外の場合は，Φ の定義と同様なので省略した．これらが互いに逆になっていることは容易に確かめられる．容易ではない．

続いて，**ArrowApply** の各項 $\tilde{M}$ に対する Ψ の自然性を確かめる．前のときと同様に，$\tilde{M}$ が `arr`, $\langle\lll\rangle$, `first`, `app` のいずれかである場合のみを考える．

≡ ' ω ' ≡ 結構しんどいと思う．

以上により，補題 10 が示され，このノートの主定理 7 が証明された．

参考文献

[1] J. Hughes (2000)「Generalising monads to arrows」『Science of Computer Programming』37:67–111

[2] S. Lindley, P. Wadler, J. Yallop (2011)「Idioms are oblivious, arrows are meticulous, monads are promiscuous」『Electronic Notes in Theoretical Computer Science』229(5):97–117