Monads and Meta-lambda Calculus

Daisuke Bekki

Ochanomizu University
Ootsuka 2-1-1, Bunkyo-ku, Tokyo, Japan

1 Background: Monads in Category Theory, Programming Language, and Natural Language

The notion of monads originates in homological algebra and category theory: a monad in a category $\mathcal{C}$ is a triple $\langle \boldsymbol{T}, \eta, \mu \rangle$ that consists of a functor $\boldsymbol{T} : \mathcal{C} \longrightarrow \mathcal{C}$ and two natural transformations:

$$\eta : \boldsymbol{Id}_{\mathcal{C}} \stackrel{\cdot}{\longrightarrow} \boldsymbol{T}, \qquad \mu : \boldsymbol{T}^2 \stackrel{\cdot}{\longrightarrow} \boldsymbol{T}$$

such that the following diagrams commute for any object A in $\mathcal{C}$.

$$\begin{tikzcd}
\boldsymbol{T}^3 A \arrow[r, "\boldsymbol{T}\mu_A"] \arrow[d, "\mu_{\boldsymbol{T}A}"'] & \boldsymbol{T}^2 A \arrow[d, "\mu_A"] \\
\boldsymbol{T}^2 A \arrow[r, "\mu_A"'] & \boldsymbol{T} A
\end{tikzcd}
\qquad
\begin{tikzcd}
\boldsymbol{T} A \arrow[r, "\eta_{\boldsymbol{T}A}"] \arrow[dr, "\boldsymbol{Id}_{\mathcal{C}}"'] & \boldsymbol{T}^2 A \arrow[d, "\mu_A"] & \boldsymbol{T} A \arrow[l, "\boldsymbol{T}\eta_A"'] \arrow[dl, "\boldsymbol{Id}_{\mathcal{C}}"] \\
 & \boldsymbol{T} A &
\end{tikzcd}$$

Lambek (1980) established categorical semantics of typed lambda calculi (hereafter TLC), showing that TLC are equivalent to Cartesian closed categories (CCC), in which TLC terms are interpreted as morphisms.

These studies converged to the 'monadic' categorical semantics of TLC in Moggi (1989), where each lambda term is interpreted as a morphism in the Kleisli category generated by a certain monad. This setting is intended to uniformly encapsulate 'impure' aspects of functional programming languages, such as side-effects, exceptions and continuations, within the enhanced data types specified by the monad, and hide them within 'pure' structures of TLC. The method requires, however, some tangled notions such as tensorial strength (Kock (1970)) for the definition of lambda abstraction, evaluation and products.

This complexity motivated Wadler (1992) to propose a simplified model, known as *monad comprehension*, which generalizes the notion of list comprehension. Results from this study were incorporated into the programming language Haskell, and this showed that the monadic analyses can treat a wider range of computational concepts than Moggi (1989) had enumerated, such as state readers, array update, non-determinism, inputs/outputs, and even parsers and interpreters.

Shan (2001) showed that the results of monadic analyses can be imported to the field of natural language semantics, where various semantic/pragmatic/computational aspects such as non-determinism, focus, intensionality, variable

H. Hattori et al. (Eds.): JSAI 2008, LNAI 5447, pp. 193–208, 2009.

binding, continuation and quantification, can be uniformly represented as monads, just as the 'impure' aspects in programming languages.

This enterprise, encapsulation of 'impure' aspects of computation by monads, seems to be an attractive prospect, especially given the lack of standard formal models for the interfaces between semantics/pragmatics or semantics/computation. On the other hand, monadic analyses, as they have drifted among different fields, seem to have become gradually dissociated from the original monad concept.

This paper aims to restore the relation between monadic analyses and categorical monads. In other words, I aim to combine the recent advances in monadic analyses with the categorical semantics of TLC along the lines of Lambek (1980). This is realized through "Meta-Lambda Calculus" (henceforth MLC), an extension of TLC, which is defined in the following way.

2 Meta-lambda Calculus

The syntax of MLC is specified by a quintuple $\langle \mathcal{GT}, \mathcal{C}on, \mathcal{M}con, \Gamma, \Delta \rangle$, which respectively represents a finite collection of ground types, constant symbols, *meta*-constant symbols, variables and *meta*-variables.

Definition 1 (Type and Meta-types). *Given a quintuple, the collections of types (notation $\mathcal{T}yp$) and meta-types (notation $\mathcal{M}typ$) are defined by the following BNF grammar.*

$$\begin{aligned} \mathcal{T}yp &:= \mathcal{GT} \mid \mathcal{T}yp\mathcal{T}yp \mid unit \mid \mathcal{T}yp \times \cdots \times \mathcal{T}yp \\ \mathcal{M}typ &:= \mathcal{T}yp \mid \mathcal{M}typ \mapsto \mathcal{M}typ \mid munit \mid \mathcal{M}typ \otimes \cdots \otimes \mathcal{M}typ \end{aligned}$$

Definition 2 (Type Assignment). *The type assignment function Σ maps each variable (a member of Γ) to a member of $\mathcal{T}yp$ and each meta-variable (a member of Δ) to a member of $\mathcal{M}typ$. The sets of constant symbols, meta-constant symbols, variables and meta-variables of type τ are respectively defined as follows.*

$$\begin{aligned} \mathcal{C}on^{\tau} &\stackrel{def}{\equiv} \{x \in \mathcal{C}on \mid \Sigma(x) = \tau\} & \Gamma^{\tau} &\stackrel{def}{\equiv} \{x \in \Gamma \mid \Sigma(x) = \tau\} \\ \mathcal{M}con^{\tau} &\stackrel{def}{\equiv} \{x \in \mathcal{M}con \mid \Sigma(x) = \tau\} & \Delta^{\tau} &\stackrel{def}{\equiv} \{x \in \Delta \mid \Sigma(x) = \tau\} \end{aligned}$$

2.1 Interpretation via Covariant Hom-Functor

An *interpretation* of MLC terms is specified by a quadruple $\langle \mathcal{E}, valT, valC, valMC \rangle$ where $\mathcal{E}$ is a Cartesian closed category with small hom-sets, $valT$ is a function that sends each $\tau \in \mathcal{GT}$ to an object in $\mathcal{E}$, $valC$ and $valMC$ are functions that send each $c \in \mathcal{C}on^{\tau}$ and each $\gamma \in \mathcal{M}con^{\tau}$ to a global element in $\mathcal{E}$ and $\mathcal{S}et$ respectively. Given a quadruple, *the type interpretation* $[\![-]\!]$ and *meta-type interpretation* $\lceil\!\lceil - \rceil\!\rceil$ are defined as follows.

Definition 3 (Interpretation of Types and Meta-types). $[\![-]\!]$ *maps each member of $\mathcal{T}yp$ to an object in $\mathcal{E}$ and $\lceil\!\lceil - \rceil\!\rceil$ maps each member of $\mathcal{M}typ$ to an object in $\mathcal{S}et$ via the following rules:*

$$
\begin{array}{rl|rl}
\lfloor\!\lfloor \tau\ (\in \mathcal{GT}) \rfloor\!\rfloor &= valT(\tau) & \lceil\!\lceil \sigma\ (\in \mathcal{T}yp) \rceil\!\rceil &= \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, \lfloor\!\lfloor \sigma \rfloor\!\rfloor) \\
\lfloor\!\lfloor \tau_1 \tau_2 \rfloor\!\rfloor &= \lfloor\!\lfloor \tau_2 \rfloor\!\rfloor^{\lfloor\!\lfloor \tau_1 \rfloor\!\rfloor} & \lceil\!\lceil \sigma_1 \mapsto \sigma_2 \rceil\!\rceil &= \lceil\!\lceil \sigma_2 \rceil\!\rceil^{\lceil\!\lceil \sigma_1 \rceil\!\rceil} \\
\lfloor\!\lfloor unit \rfloor\!\rfloor &= 1 & \lceil\!\lceil munit \rceil\!\rceil &= * \\
\lfloor\!\lfloor \tau_1 \times \cdots \times \tau_n \rfloor\!\rfloor &= \lfloor\!\lfloor \tau_1 \rfloor\!\rfloor \times \cdots \times \lfloor\!\lfloor \tau_n \rfloor\!\rfloor & \lceil\!\lceil \sigma_1 \otimes \cdots \otimes \sigma_n \rceil\!\rceil &= \lceil\!\lceil \sigma_1 \rceil\!\rceil \times \cdots \times \lceil\!\lceil \sigma_n \rceil\!\rceil
\end{array}
$$

where 1 *is a selected terminal object in* $\mathcal{E}$. *Let* $*$ *be any one-point set, which is a terminal object in* $\mathcal{S}et$. *Suppose that* $\Gamma = x_1, \ldots, x_m$ *and* $\Delta = X_1, \ldots, X_n$, *then* $\lfloor\!\lfloor \Gamma \rfloor\!\rfloor = \lfloor\!\lfloor \Sigma(x_1) \rfloor\!\rfloor \times \cdots \times \lfloor\!\lfloor \Sigma(x_m) \rfloor\!\rfloor$ *and* $\lceil\!\lceil \Delta \rceil\!\rceil = \lceil\!\lceil \Sigma(X_1) \rceil\!\rceil \times \cdots \times \lceil\!\lceil \Sigma(X_n) \rceil\!\rceil$.

In contrast to the standard categorical semantics[1] of TLC where an interpretation $[\![-]\!]$ of a lambda term of the type τ is a morphism: $\lfloor\!\lfloor \Gamma \rfloor\!\rfloor \longrightarrow \lfloor\!\lfloor \tau \rfloor\!\rfloor$ in $\mathcal{E}$, the categorical semantics of MLC utilizes a *covariant hom-functor*[2] $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, -) : \mathcal{E} \longrightarrow \mathcal{S}et$, by which a morphism $f : A \longrightarrow B$ in $\mathcal{E}$ is mapped to the morphism $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, f) : \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, A) \longrightarrow \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, B)$ in $\mathcal{S}et$. $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, f)$ is also written as f_* and called "composition with f on the left" or "the map induced by f." $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, f)$ maps a morphism $a : \lfloor\!\lfloor \Gamma \rfloor\!\rfloor \longrightarrow A$ in $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, A)$ to $f \circ a : \lfloor\!\lfloor \Gamma \rfloor\!\rfloor \longrightarrow B$ in $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, B)$. The two morphisms $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, f)$ and $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, g)$ induced by two composable morphisms $f : A \longrightarrow B$ and $g : B \longrightarrow C$ are also composable in $\mathcal{S}et$, as indicated in the following diagram.

$$
\begin{tikzcd}
\lfloor\!\lfloor \Gamma \rfloor\!\rfloor \arrow[d, "a"'] & & \\
A \arrow[r, "f"] & B \arrow[r, "g"] & C
\end{tikzcd}
$$

Now the following bijection is a natural isomorphism (tp and $\overline{tp}$ are *transposes* of each other).[3]

$$
(1) \qquad \mathcal{S}et(*, \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, -)) \underset{tp}{\overset{\overline{tp}}{\rightleftarrows}} \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, -)
$$

Then, any interpretation of a TLC term M of type τ by the standard categorical semantics, which is also an element of $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, \lfloor\!\lfloor \tau \rfloor\!\rfloor)$, is mapped to the corresponding element in $\mathcal{S}et(*, \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, \lfloor\!\lfloor \tau \rfloor\!\rfloor))$, via the (component $tp_{\lfloor\!\lfloor \tau \rfloor\!\rfloor}$ of) natural transformation tp. tp is specified by the universal arrow which obtains by applying $id_{\lfloor\!\lfloor \Gamma \rfloor\!\rfloor}$ to tp itself. Namely, $tp([\![M]\!]) = \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, [\![M]\!]) \circ tp(id_{\lfloor\!\lfloor \Gamma \rfloor\!\rfloor})$, where $\langle \lfloor\!\lfloor \Gamma \rfloor\!\rfloor, tp(id_{\lfloor\!\lfloor \Gamma \rfloor\!\rfloor}) \rangle$ is a universal arrow from $*$ to $\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, -)$.

$$
\begin{tikzcd}
\lfloor\!\lfloor \Gamma \rfloor\!\rfloor \arrow[d, dotted, "{[\![M]\!]}"'] & \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, \lfloor\!\lfloor \Gamma \rfloor\!\rfloor) \arrow[d, "{\mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, [\![M]\!])}"'] & * \arrow[l, "{u = tp(id_{\lfloor\!\lfloor \Gamma \rfloor\!\rfloor})}"'] \arrow[dl, "{tp([\![M]\!])}"] \\
\lfloor\!\lfloor \tau \rfloor\!\rfloor & \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, \lfloor\!\lfloor \tau \rfloor\!\rfloor) &
\end{tikzcd}
$$

---

[1] See Lambek (1980), Lambek and Scott (1986) and Crole (1993), among others.

[2] See MacLane (1997), p.34.

[3] Proof of (1) is found in the proof of Proposition 2 in MacLane (1997), p.60. For an element $f \in \mathcal{S}et(*, \mathcal{E}(\lfloor\!\lfloor \Gamma \rfloor\!\rfloor, -))$, $\overline{tp}(f) = f(*)$.

Let $\mathcal{S}_\mathcal{E}$ be the subcategory of $\mathcal{S}et$, defined by the 'shadow' of the functor $\mathcal{E}(\llbracket \Gamma \rrbracket, -)$. Then the interpretation of a TLC term of type τ is embedded in $\mathcal{S}_\mathcal{E}$ as morphisms $* \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau \rrbracket)$.

In general, the interpretation of an MLC term of type τ is a morphism $\llbracket \Delta \rrbracket \longrightarrow \llbracket \tau \rrbracket$. When $\tau \in \mathcal{T}yp$, $\llbracket \tau \rrbracket = \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau \rrbracket)$. Therefore, elements of MLC with no meta-variables (namely, $\Delta = \{\}$ and $\llbracket \Delta \rrbracket = *$) are in one-to-one correspondence with elements of TLC.

2.2 Syntax and Semantics

Given a quintuple $\langle \mathcal{GT}, \mathcal{C}on, \mathcal{M}con, \Gamma, \Delta \rangle$, *the set of meta-lambda terms of type* $\tau \in \mathcal{M}typ$ in MLC (notation Λ^τ) is recursively defined as follows.

Variables

$$\frac{x_i \in \Gamma^\tau}{x_i \in \Lambda^\tau, \quad \llbracket x_i \rrbracket = tp(\pi_i) \circ !_{\llbracket \Delta \rrbracket} : \llbracket \Delta \rrbracket \longrightarrow * \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau \rrbracket)}$$

Meta-Variables

$$\frac{X_i \in \Delta^\tau}{X_i \in \Lambda^\tau, \quad \llbracket X_i \rrbracket = \pi_i : \llbracket \Delta \rrbracket \longrightarrow \llbracket \tau \rrbracket}$$

Since a projection π_i in $\mathcal{E}$ is a morphism $\llbracket \Gamma \rrbracket \longrightarrow \llbracket \tau_i \rrbracket$, its transpose $tp(\pi_i)$ is a morphism $* \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau_i \rrbracket)$ in $\mathcal{S}et$. A meta-variable X_i is just interpreted as the projection π_i in $\mathcal{S}_\mathcal{E}$, which selects the i-th member of Δ and returns its value.

Constant Symbols

$$\frac{c \in \mathcal{C}on^\tau}{c \in \Lambda^\tau, \quad \llbracket c \rrbracket = tp(valC(c) \circ !_{\llbracket \Gamma \rrbracket}) \circ !_{\llbracket \Delta \rrbracket} : \llbracket \Delta \rrbracket \longrightarrow * \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau \rrbracket)}$$

Meta-Constant Symbols

$$\frac{\gamma \in \mathcal{M}con^\tau}{\gamma \in \Lambda^\tau, \quad \llbracket \gamma \rrbracket = valMC(\gamma) \circ !_{\llbracket \Delta \rrbracket} : \llbracket \Delta \rrbracket \longrightarrow * \longrightarrow \llbracket \tau \rrbracket}$$

Constant and meta-constant symbols are interpreted via $valC$ and $valMC$, which associate them with global elements in $\mathcal{E}$ and $\mathcal{S}_\mathcal{E}$.

Product

$$\frac{}{\langle\ \rangle \in \Lambda^{unit}, \quad \llbracket \langle\ \rangle \rrbracket = tp(!_{\llbracket \Gamma \rrbracket}) \circ !_{\llbracket \Delta \rrbracket} : \llbracket \Delta \rrbracket \longrightarrow * \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket unit \rrbracket)}$$

$$\frac{M_1 \in \Lambda^{\tau_1} \quad \cdots \quad M_n \in \Lambda^{\tau_n} \quad \tau_1 \in \mathcal{T}yp \quad \cdots \quad \tau_n \in \mathcal{T}yp}{\begin{array}{c}\langle M_1, \ldots, M_n \rangle \in \Lambda^{\tau_1 \times \cdots \times \tau_n}, \quad \llbracket \langle M_1, \ldots, M_n \rangle \rrbracket = \ltimes \circ \langle \llbracket M_1 \rrbracket, \ldots, \llbracket M_n \rrbracket \rangle : \\ \llbracket \Delta \rrbracket \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau_1 \rrbracket) \times \cdots \times \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau_n \rrbracket) \\ \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau_1 \rrbracket \times \cdots \times \llbracket \tau_n \rrbracket)\end{array}}$$

$$\frac{P \in \Lambda^{\tau_1 \times \cdots \times \tau_n} \quad \tau_1 \times \cdots \times \tau_n \in \mathcal{T}yp \quad 1 \leq k \leq n}{\pi_k(P) \in \Lambda^{\tau_k}, \quad \llbracket \pi_k(P) \rrbracket = \mathcal{E}(\llbracket \Gamma \rrbracket, \pi_k) \circ \llbracket P \rrbracket : \llbracket \Delta \rrbracket \longrightarrow \mathcal{E}(\llbracket \Gamma \rrbracket, \llbracket \tau_k \rrbracket)}$$

The morphism $\ltimes$, together with its inverse $\rtimes$, is an isomorphism between the following two objects in $\mathcal{S}et$.

$$\mathcal{E}(\llbracket\Gamma\rrbracket, A) \times \mathcal{E}(\llbracket\Gamma\rrbracket, B) \cong \mathcal{E}(\llbracket\Gamma\rrbracket, A \times B)$$

$$\begin{array}{rl} \ltimes : & f, g, \ldots \mapsto \langle f, g, \ldots\rangle \\ \rtimes : & f \mapsto \pi_1 \circ f,\ \pi_2 \circ f, \ldots \end{array}$$

The pair of $\ltimes$ and $\rtimes$ is a bijection since the hom-functor $\mathcal{E}(\llbracket\Gamma\rrbracket, -)$ preserves all finite limits.[4]

Meta-Product

$$\frac{}{\langle\!\langle\ \rangle\!\rangle \in \Lambda^{munit}, \quad \llbracket\langle\!\langle\ \rangle\!\rangle\rrbracket = !_{\llbracket\Delta\rrbracket} : \llbracket\Delta\rrbracket \longrightarrow \llbracket munit\rrbracket}$$

$$\frac{M_1 \in \Lambda^{\tau_1} \quad \cdots \quad M_n \in \Lambda^{\tau_n}}{\begin{array}{l}\langle\!\langle M_1, \ldots, M_n\rangle\!\rangle \in \Lambda^{\tau_1 \otimes \cdots \otimes \tau_n}, \\ \llbracket\langle\!\langle M_1, \ldots, M_n\rangle\!\rangle\rrbracket = \langle\llbracket M_1\rrbracket, \ldots, \llbracket M_n\rrbracket\rangle : \llbracket\Delta\rrbracket \longrightarrow \llbracket\tau_1\rrbracket \times \cdots \times \llbracket\tau_n\rrbracket\end{array}}$$

$$\frac{P \in \Lambda^{\tau_1 \otimes \cdots \otimes \tau_n} \quad 1 \le k \le n}{\pi_k(P) \in \Lambda^{\tau_k}, \quad \llbracket\pi_k(P)\rrbracket = \pi_k \circ \llbracket P\rrbracket : \llbracket\Delta\rrbracket \longrightarrow \llbracket\tau_k\rrbracket}$$

Meta-products roughly correspond to products in $\mathcal{S}_{\mathcal{E}}$, whose interpretation is similar to the interpretation of (normal) products in TLC.

Meta-Lambda Abstraction

$$\frac{X \in \Delta^{\tau} \quad M \in \Lambda^{\sigma}}{\zeta X_i.M \in \Lambda^{\tau \mapsto \sigma}, \quad \llbracket\zeta X_i.M\rrbracket = \lambda(\llbracket M\rrbracket) \circ {\pi^{\Delta}}_{\Delta\backslash X_i} : \llbracket\Delta\rrbracket \longrightarrow \llbracket\sigma\rrbracket^{\llbracket\tau\rrbracket}}$$

Meta-Functional Application

$$\frac{M \in \Lambda^{\tau \mapsto \sigma} \quad N \in \Lambda^{\tau}}{M\lceil N\rceil \in \Lambda^{\sigma}, \quad \llbracket M\lceil N\rceil\rrbracket = ev \circ \langle\llbracket M\rrbracket, \llbracket N\rrbracket\rangle : \llbracket\Delta\rrbracket \longrightarrow \llbracket\sigma\rrbracket}$$

Meta-Substitution

$$\frac{M \in \Lambda^{\sigma} \quad N \in \Lambda^{\tau} \quad X \in \Delta^{\tau}}{M[N/X_i] \in \Lambda^{\sigma}, \quad \llbracket M[N/X_i]\rrbracket = \llbracket M\rrbracket \circ \langle{\pi^{\Delta}}_{\Delta\backslash X_i}, \llbracket N\rrbracket\rangle : \llbracket\Delta\rrbracket \longrightarrow \llbracket\sigma\rrbracket}$$

The interpretation of meta-lambda abstraction, functional application and substitution in MLC is similar to the interpretation of (normal) elements of TLC.[5] Just as beta conversion is sound in TLC, meta-beta conversion is sound in MLC:

Theorem 4 (Meta-Beta Conversion)

$$(\zeta X.M)\lceil N\rceil = M[N/X]$$

[4] See "Theorem 1" and its proof in MacLane (1997), p.116.

[5] The projection ${\pi^{\Delta}}_{\Delta\backslash X_i}$ is defined as the morphism $\langle\pi_1, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_n\rangle$, which maps $\langle X_1, \ldots, X_{i-1}, X_i, X_{i+1}, \ldots, X_n\rangle$ to $\langle X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n\rangle$, namely, the morphism which forget the i-th element. This construction crucially depends on the assumption that Δ is a finite set.

This is proved by a standard triangular identity for a Cartesian closed category.

Proof

$$
\begin{aligned}
\llbracket(\zeta X.M)\lceil N\rceil\rrbracket &= ev \circ \langle \llbracket \zeta X.M \rrbracket, \llbracket N \rrbracket\rangle \\
&= ev \circ \langle \lambda(\llbracket M \rrbracket) \circ {\pi^{\Delta}}_{\Delta\backslash X}, \llbracket N \rrbracket\rangle \\
&= ev \circ (\lambda(\llbracket M \rrbracket) \times id) \circ \langle {\pi^{\Delta}}_{\Delta\backslash X}, \llbracket N \rrbracket\rangle \\
&= \llbracket M \rrbracket \circ \langle {\pi^{\Delta}}_{\Delta\backslash X}, \llbracket N \rrbracket\rangle \\
&= \llbracket M[N/X] \rrbracket
\end{aligned}
$$

$$
\begin{tikzcd}
\mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor)^{\mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\tau\rfloor\!\rfloor)} \times \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\tau\rfloor\!\rfloor) \arrow[r, "ev"] & \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor) \\
\lceil\!\lceil\Delta\rceil\!\rceil = \lceil\!\lceil\Delta\rceil\!\rceil' \times \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\tau\rfloor\!\rfloor) \arrow[u, "\lambda(\llbracket M\rrbracket)\times id"] \arrow[ur, "\llbracket M\rrbracket"'] &
\end{tikzcd}
$$

□

Lambda Abstraction

$$
\frac{x \in \Gamma^{\tau} \quad M \in \Lambda^{\sigma} \quad \sigma,\tau \in \mathcal{T}yp}{\lambda x.M \in \Lambda^{\tau\sigma}, \quad \llbracket \lambda x.M \rrbracket = {\leftthreetimes^{\lfloor\!\lfloor\tau\rfloor\!\rfloor}}_{\lfloor\!\lfloor\sigma\rfloor\!\rfloor} \circ \llbracket M \rrbracket : \lceil\!\lceil\Delta\rceil\!\rceil \longrightarrow \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor^{\lfloor\!\lfloor\tau\rfloor\!\rfloor})}
$$

Functional Application

$$
\frac{M \in \Lambda^{\tau\sigma} \quad N \in \Lambda^{\tau} \quad \sigma,\tau \in \mathcal{T}yp}{\begin{aligned} &M(N) \in \Lambda^{\sigma}, \quad \llbracket M(N) \rrbracket = \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, ev) \circ \ltimes \circ \langle \llbracket M \rrbracket, \llbracket N \rrbracket \rangle : \\ &\lceil\!\lceil\Delta\rceil\!\rceil \longrightarrow \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor^{\lfloor\!\lfloor\tau\rfloor\!\rfloor}) \times \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\tau\rfloor\!\rfloor) \\ &\longrightarrow \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor^{\lfloor\!\lfloor\tau\rfloor\!\rfloor} \times \lfloor\!\lfloor\tau\rfloor\!\rfloor) \\ &\longrightarrow \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\tau\rfloor\!\rfloor) \end{aligned}}
$$

Substitution

$$
\frac{M \in \Lambda^{\sigma} \quad N \in \Lambda^{\tau} \quad x \in \Gamma^{\tau} \quad \sigma,\tau \in \mathcal{T}yp}{M[N/x] \in \Lambda^{\sigma}, \quad \llbracket M[N/x] \rrbracket = \mathcal{S}ub \circ \langle \llbracket M \rrbracket, \llbracket N \rrbracket \rangle : \lceil\!\lceil\Delta\rceil\!\rceil \longrightarrow \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, \lfloor\!\lfloor\sigma\rfloor\!\rfloor)}
$$

For the definition of normal lambda abstractions, functional application and substitutions, some additional notions are required: the morphism $\leftthreetimes$ and $\mathcal{S}ub$.

The substitution rule for meta-variables, which is not described in detail in this paper for the sake of space, is supposed to be immune with respect to the binding of (normal) variables, as the following equation implies.

(2) $$(\zeta X.\lambda x.X)\lceil x\rceil = (\lambda x.X)[x/X] = \lambda x.x$$

This means that the following map Λ is representable at the level of the object language, which is not the case in TLC.

(3) $$\leftthreetimes : \phi \mapsto \lambda x.\phi$$

Lambda Abstraction Map $\leftthreetimes$: The *lambda abstraction map* $\leftthreetimes^{D}$ is a natural transformation: $\mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, -) \xrightarrow{\cdot} \mathcal{E}(\lfloor\!\lfloor\Gamma\rfloor\!\rfloor, -^{D})$, whose component is written as

$\leftthreetimes^D{}_A$. The functor $\mathcal{E}(\llbracket\Gamma\rrbracket, -^D) : \mathcal{E} \longrightarrow \mathcal{S}et$ maps a morphism $f : A \longrightarrow B$ in $\mathcal{E}$ to a morphism between hom-sets that maps any morphism $a : \llbracket\Gamma\rrbracket \longrightarrow A^D \in \mathcal{E}(\llbracket\Gamma\rrbracket, A^D)$ to $\lambda(f \circ \overline{a}) : \Gamma \longrightarrow B^D \in \mathcal{E}(\llbracket\Gamma\rrbracket, B^D)$.

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, dotted, "\mathcal{E}(\llbracket\Gamma\rrbracket,-^D)"'] & B \arrow[d, dotted, "\mathcal{E}(\llbracket\Gamma\rrbracket,-^D)"] \\
\mathcal{E}(\llbracket\Gamma\rrbracket, A^D) \arrow[r, "\mathcal{E}(\llbracket\Gamma\rrbracket,f^D)"'] & \mathcal{E}(\llbracket\Gamma\rrbracket, B^D)
\end{tikzcd}$$

The intuition behind the map $\mathcal{E}(\llbracket\Gamma\rrbracket, f^D)$ is illustrated in the following diagram.

$$\begin{tikzcd}
B^D \times D \arrow[rr, "ev"] & & B \\
A^D \times D \arrow[r, "ev"] & A \arrow[ur, "f"'] & \\
\llbracket\Gamma\rrbracket \times D \arrow[u, "a \times id_D"] \arrow[ur, "\overline{a}"'] \arrow[uu, bend left, "\lambda(f\circ\overline{a})\times id_D"] & &
\end{tikzcd}$$

Now, by the Yoneda lemma, the following isomorphism exists, and each natural transformation from $(\mathcal{E}(\llbracket\Gamma\rrbracket, -)$ to $\mathcal{E}(\llbracket\Gamma\rrbracket, -^D)$ is specified by an element of $\mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\Gamma\rrbracket^D)$.

$$Nat(\mathcal{E}(\llbracket\Gamma\rrbracket, -), \mathcal{E}(\llbracket\Gamma\rrbracket, -^D) \cong \mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\Gamma\rrbracket^D)$$

The natural transformation $\leftthreetimes^D$ is thus specified by the morphism $\lambda(id_{\llbracket\Gamma\rrbracket}) \circ \pi^\Gamma{}_{\Gamma\backslash D} : \llbracket\Gamma\rrbracket \longrightarrow \llbracket\Gamma\rrbracket^D$ that makes the following diagram commute.[6]

$$\begin{tikzcd}
\llbracket\Gamma\rrbracket \arrow[r, "\pi^\Gamma{}_{\Gamma\backslash D}\times\pi^\Gamma{}_D"] & \llbracket\Gamma\rrbracket' \times D \arrow[r, "\lambda(id_{\llbracket\Gamma\rrbracket})\times id_D"] \arrow[dr, "id_{\llbracket\Gamma\rrbracket}"'] & \llbracket\Gamma\rrbracket^D \times D \arrow[d, "ev"] \\
& & \llbracket\Gamma\rrbracket
\end{tikzcd}$$

(where $\llbracket\Gamma\rrbracket = \llbracket\Gamma\rrbracket' \times D$)

Then each component $\leftthreetimes^D{}_A$ is obtained by applying this morphism to $\mathcal{E}(\llbracket\Gamma\rrbracket, f^D)$, which sends a morphism $f : \llbracket\Gamma\rrbracket \longrightarrow A$ to the morphism from $\llbracket\Gamma\rrbracket$ to B.

$$\begin{aligned}
\leftthreetimes^D{}_A \ &: \ f \mapsto \mathcal{E}(\llbracket\Gamma\rrbracket, f^D)(\lambda(id_{\llbracket\Gamma\rrbracket}) \circ \pi^\Gamma{}_{\Gamma\backslash D}) \\
&= f \mapsto \lambda(f \circ \overline{\lambda(id_{\llbracket\Gamma\rrbracket}) \circ \pi^\Gamma{}_{\Gamma\backslash D}})
\end{aligned}$$

[6] The projection $\pi^\Gamma{}_{\Gamma\backslash A_i}$ is defined as the morphism $\langle\pi_1, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_n\rangle$, and $\pi^\Gamma{}_{A_i}$ as π_i, given that $\llbracket\Gamma\rrbracket = A_1 \times \cdots \times A_i \times \cdots \times A_n$. As mentioned in fn.5, this depends on the assumption that Γ is a finite set.

Substitution Map $\mathcal{S}ub$: Substitution map $\mathcal{S}ub$ is a morphism from $\mathcal{E}(\llbracket\Gamma\rrbracket, A) \times \mathcal{E}(\llbracket\Gamma\rrbracket, B)$ to $\mathcal{E}(\llbracket\Gamma\rrbracket, A)$ defined as follows, where $m : \llbracket\Gamma\rrbracket \longrightarrow A$, $n : \llbracket\Gamma\rrbracket \longrightarrow B$ and $\llbracket\Sigma(x)\rrbracket = B$.

$$\mathcal{S}ub : m, n \mapsto m \circ \langle {\pi^{\Gamma}}_{\Gamma\backslash x}, n\rangle$$

Theorem 5 (Normal Beta Conversion)

$$(\lambda x.M)(N) = M[N/x]$$

Proof. In the diagram below, the maps $m : \Gamma \longrightarrow \llbracket\sigma\rrbracket)$ and $n : \Gamma \longrightarrow \llbracket\tau\rrbracket$ are mapped successively to $\langle\lambda(m \circ \overline{\lambda(id) \circ {\pi^{\Gamma}}_{\Gamma\backslash x}}), n\rangle$.

$$\begin{array}{ccccc}
\mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\sigma\rrbracket^{\llbracket\tau\rrbracket}) \times \mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\tau\rrbracket) & \xrightarrow{\sim} & \mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\sigma\rrbracket^{\llbracket\tau\rrbracket} \times \llbracket\tau\rrbracket) & \xrightarrow{\mathcal{E}(\llbracket\Gamma\rrbracket, ev)} & \mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\sigma\rrbracket) \\
\uparrow {\scriptstyle \lambda^{\llbracket\tau\rrbracket}{}_{\llbracket\sigma\rrbracket} \times id_{\mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\tau\rrbracket)}} & & & & \\
\mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\sigma\rrbracket) \times \mathcal{E}(\llbracket\Gamma\rrbracket, \llbracket\tau\rrbracket) & & & & \\
\uparrow {\scriptstyle \langle\llbracket M\rrbracket, \llbracket N\rrbracket\rangle} & & & & \\
\llbracket\Delta\rrbracket & & & &
\end{array}$$

which is then proved to be equivalent to the substitution map $\mathcal{S}$ as follows.

$$\begin{aligned}
m, n \mapsto\ & ev \circ \langle\lambda(m \circ \overline{\lambda(id) \circ {\pi^{\Gamma}}_{\Gamma\backslash x}}), n\rangle \\
=\ & m \circ \overline{\lambda(id_{\Gamma}) \circ {\pi^{\Gamma}}_{\Gamma\backslash x}} \circ \langle id, n\rangle \\
=\ & m \circ ev \circ (\lambda(id) \circ {\pi^{\Gamma}}_{\Gamma\backslash x} \times id) \circ \langle id, n\rangle \\
=\ & m \circ ev \circ \langle\lambda(id) \circ {\pi^{\Gamma}}_{\Gamma\backslash x} \circ id, id \circ n\rangle \\
=\ & m \circ ev \circ \langle\lambda(id) \circ {\pi^{\Gamma}}_{\Gamma\backslash x}, n\rangle \\
=\ & m \circ ev \circ (\lambda(id) \times id) \circ ({\pi^{\Gamma}}_{\Gamma\backslash x} \times n) \\
=\ & m \circ \langle{\pi^{\Gamma}}_{\Gamma\backslash x}, n\rangle
\end{aligned}$$

□

3 Transformation with Internal Monads

3.1 Δ-Indexed Category

In a category $\mathcal{S}_{\mathcal{E}}$, associate with each object $A \in \llbracket\Delta\rrbracket$ a new object A_{Δ} and to each arrow $f : \llbracket\Delta\rrbracket \times A \longrightarrow B$ in $\mathcal{C}$ a new arrow $f_{\Delta} : A_{\Delta} \longrightarrow B_{\Delta}$. These new objects and arrows constitute a category when the composite of f_{Δ} with $g_{\Delta} : B_{\Delta} \longrightarrow C_{\Delta}$ is defined by:

$$g_{\Delta} \circ f_{\Delta} \stackrel{def}{\equiv} g \circ \langle\pi_1, f\rangle$$

Let us call the category constructed as above the Δ-indexed category of $\mathcal{S}_{\mathcal{E}}$ (notation $\Delta \times \mathcal{S}_{\mathcal{E}}$).

Theorem 6. *For any MLC terms $M \in \Lambda^{A \mapsto B}$ and $N \in \Lambda^{B \mapsto C}$, the following equation holds for morphisms in a Δ-indexed category.*

$$(\overline{\llbracket N \rrbracket})_\Delta \circ (\overline{\llbracket M \rrbracket})_\Delta = (\overline{\llbracket N \circ M \rrbracket})_\Delta$$

Proof. Equality is proved as follows.

$$\begin{aligned}
(\overline{\llbracket N \rrbracket})_\Delta \circ (\overline{\llbracket M \rrbracket})_\Delta &= \overline{\llbracket N \rrbracket} \circ \langle \pi_1, \overline{\llbracket M \rrbracket} \rangle \\
&= ev \circ (\llbracket N \rrbracket \times id_B) \circ \langle \pi_1, ev \circ (\llbracket M \rrbracket \times id_A) \rangle \\
&= ev \circ \langle \llbracket N \rrbracket \circ \pi_1, ev \circ (\llbracket M \rrbracket \times id_A) \rangle
\end{aligned}$$

$$\begin{aligned}
(\overline{\llbracket N \circ M \rrbracket})_\Delta &= \overline{\llbracket N \circ M \rrbracket} \\
&= \overline{\llbracket \zeta X. N \lceil M \lceil X \rceil \rceil \rrbracket} \\
&= \overline{\lambda(\llbracket N \lceil M \lceil X \rceil \rceil \rrbracket) \circ {\pi^{\Delta}}_{\Delta \backslash X}} \\
&= \overline{\lambda(ev \circ \langle \llbracket N \rrbracket, \llbracket M \lceil X \rceil \rrbracket \rangle) \circ {\pi^{\Delta}}_{\Delta \backslash X}} \\
&= ev \circ (\lambda(ev \circ \langle \llbracket N \rrbracket, \llbracket M \lceil X \rceil \rrbracket \rangle) \circ {\pi^{\Delta}}_{\Delta \backslash X} \times id_A) \\
&= ev \circ (\lambda(ev \circ \langle \llbracket N \rrbracket, \llbracket M \lceil X \rceil \rrbracket \rangle \times id_A) \circ ({\pi^{\Delta}}_{\Delta \backslash X} \times id_A) \\
&= ev \circ \langle \llbracket N \rrbracket, \llbracket M \lceil X \rceil \rrbracket \rangle \circ ({\pi^{\Delta}}_{\Delta \backslash X} \times id_A) \\
&= ev \circ \langle \llbracket N \rrbracket, ev \circ \langle \llbracket M \rrbracket, \pi_A \rangle \rangle \circ \pi_1 \qquad \text{—}(\dagger) \\
&= ev \circ \langle \llbracket N \rrbracket \circ \pi_1, ev \circ \langle \llbracket M \rrbracket, \pi_A \rangle \circ \pi_1 \rangle \\
&= ev \circ \langle \llbracket N \rrbracket \circ \pi_1, ev \circ (\llbracket M \rrbracket \times id_A) \rangle \qquad \text{—}(\ddagger)
\end{aligned}$$

□

Two remarks should be made concerning this proof. Firstly, the substitution (†) is due to the equation ${\pi^{\Delta}}_{\Delta \backslash X} \times id_A = \pi_1$, the proof of which is illustrated in the following diagram.

$$\begin{tikzcd}
 & \lceil\!\lceil \Delta \rceil\!\rceil \times A \arrow[dl, "\pi_1"'] \arrow[d, "{\pi^{\Delta}}_{\Delta \backslash X} \times id_A"] \\
\lceil\!\lceil \Delta \rceil\!\rceil \arrow[r, leftrightarrow, "\sim"] & \lceil\!\lceil \Delta \rceil\!\rceil' \times A
\end{tikzcd}$$

The second remark concerns the substitution (‡), which depends on the equation $\langle \llbracket M \rrbracket, \pi_A \rangle \circ \pi_1 = (\llbracket M \rrbracket \times id_A)$ that holds under the condition that $\lceil\!\lceil \Delta \rceil\!\rceil \cong \lceil\!\lceil \Delta \rceil\!\rceil \times A$, which in turn depends on our assumption that Δ includes all occurrences of meta-variables. The bijection between $\lceil\!\lceil \Delta \rceil\!\rceil$ and $\lceil\!\lceil \Delta \rceil\!\rceil \times A$ is illustrated in the following diagram.

$$\begin{tikzcd}
\lceil\!\lceil \Delta \rceil\!\rceil & \lceil\!\lceil \Delta \rceil\!\rceil \times A \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & A \\
 & \lceil\!\lceil \Delta \rceil\!\rceil \arrow[ul, leftrightarrow, "id_{\lceil\!\lceil \Delta \rceil\!\rceil}"] \arrow[u, dotted, "\langle id_{\lceil\!\lceil \Delta \rceil\!\rceil}, \pi_A \rangle"] \arrow[ur, "\pi_A"'] &
\end{tikzcd}$$

Under this assumption, the equation $\langle \llbracket M \rrbracket, \pi_A \rangle \circ \pi_1 = (\llbracket M \rrbracket \times id_A)$ can be shown to hold as follows:

$$
\begin{tikzcd}
\llbracket \Delta \rrbracket \arrow[d, "\llbracket M \rrbracket"'] \arrow[rr, bend left, "\pi_A"] & \llbracket \Delta \rrbracket \times A \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\langle \llbracket M \rrbracket, id_A \rangle"] & A \arrow[d, "id_A"] \\
B & B \times A \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & A
\end{tikzcd}
$$

3.2 Internal Monad

Definition 7 (Internal Monad). *An internal monad is a triple $\langle \boldsymbol{T}, \eta, \mu \rangle$ of meta-lambda terms which satisfies the following condition: the triple $\langle \overline{\boldsymbol{T}}, \overline{\eta}, \overline{\mu} \rangle$ constitutes a categorical monad in $\Delta \times \mathcal{S}_{\mathcal{E}}$, where $\overline{\boldsymbol{T}}, \overline{\eta}, \overline{\mu}$ are defined as follows:*

$$\overline{\boldsymbol{T}} \stackrel{def}{\equiv} f \mapsto \overline{\llbracket \boldsymbol{T} \lceil \zeta X. f \lceil X \rceil \rceil \rrbracket} \text{ (as the arrow function of the functor } \overline{\boldsymbol{T}}\text{)}$$
$$\overline{\eta} \stackrel{def}{\equiv} \overline{\llbracket \eta \rrbracket} \text{ (as a component of a natural transformation)}$$
$$\overline{\mu} \stackrel{def}{\equiv} \overline{\llbracket \mu \rrbracket} \text{ (as a component of a natural transformation)}$$

By specifying an internal monad, monadic analyses can be represented by the following interpretation schema.

Definition 8 (Transformation with Internal Monad (call-by-value))

$$
\begin{aligned}
\llbracket x \rrbracket_{\boldsymbol{T}} &= \eta \lceil x \rceil \\
\llbracket \lambda x.M \rrbracket_{\boldsymbol{T}} &= \boldsymbol{T} \lceil \zeta X. \lambda x. X \rceil \lceil \llbracket M \rrbracket_{\boldsymbol{T}} \rceil \\
\llbracket M(N) \rrbracket_{\boldsymbol{T}} &= \mu(\boldsymbol{T} \lceil \zeta X. (\boldsymbol{T} \lceil \zeta Y. X(Y) \rceil \lceil \llbracket N \rrbracket_{\boldsymbol{T}} \rceil \rceil \lceil \llbracket M \rrbracket_{\boldsymbol{T}} \rceil) \\
\llbracket \langle M, N \rangle \rrbracket_{\boldsymbol{T}} &= \mu(\boldsymbol{T} \lceil \zeta Y. (\boldsymbol{T} \lceil \zeta X. \langle X, Y \rangle \rceil \lceil \llbracket M \rrbracket_{\boldsymbol{T}} \rceil \rceil \lceil \llbracket N \rrbracket_{\boldsymbol{T}} \rceil)
\end{aligned}
$$

Theorem 9. *If a triple $\langle \boldsymbol{T}, \eta, \mu \rangle$ satisfies the following four equations, the corresponding triple $\langle \overline{\boldsymbol{T}}, \overline{\eta}, \overline{\mu} \rangle$ becomes a (categorical) monad; therefore the triple $\langle \boldsymbol{T}, \eta, \mu \rangle$ qualifies as an internal monad:*

$\boldsymbol{T}$ conditions:	$\boldsymbol{T} \lceil \zeta X.X \rceil = \zeta X.X$	$\boldsymbol{T} \lceil g \rceil \circ \boldsymbol{T} \lceil f \rceil = \boldsymbol{T} \lceil g \circ f \rceil$
η and μ conditions:	$\boldsymbol{T} \lceil f \rceil \circ \eta = \eta \circ f$	$\boldsymbol{T} \lceil f \rceil \circ \mu = \mu \circ \boldsymbol{T} \lceil \boldsymbol{T} \lceil f \rceil \rceil$
Square identity:	$\mu \circ \boldsymbol{T} \lceil \mu \rceil = \mu \circ \mu$	
Triangular identity:	$\mu \circ \eta = \zeta X.X$	$\mu \circ \boldsymbol{T} \lceil \eta \rceil = \zeta X.X$

Proof. The following equations show that $\overline{\boldsymbol{T}}$ is an endo-functor in the Δ-indexed category $\Delta \times \mathcal{S}et_{\mathcal{E}}$.

$$
\begin{aligned}
(\overline{\llbracket \boldsymbol{T} \lceil \zeta X.(g \circ f) \lceil X \rceil \rceil \rrbracket})_\Delta &= (\overline{\llbracket \boldsymbol{T}(\zeta X.g \lceil X \rceil \circ \zeta X.f \lceil X \rceil) \rrbracket})_\Delta \\
&= (\overline{\llbracket \boldsymbol{T} \lceil \zeta X.g \lceil X \rceil \rceil \rrbracket})_\Delta \circ (\overline{\llbracket \boldsymbol{T} \lceil \zeta X.f \lceil X \rceil \rceil \rrbracket})_\Delta
\end{aligned}
$$

$$
\begin{aligned}
(\overline{\llbracket \boldsymbol{T}\lceil \zeta X_i.id\lceil X_i\rceil\rceil\rrbracket})_\Delta &= (\overline{\llbracket \boldsymbol{T}\lceil \zeta X_i.X_i\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \zeta X_i.X_i\rrbracket})_\Delta \\
&= (\overline{\lambda(\llbracket X_i\rrbracket)\circ \pi^{\Delta}{}_{\Delta\backslash X_i}})_\Delta \\
&= (\llbracket X_i\rrbracket)_\Delta \\
&= (\pi_i)_\Delta \\
&= id_\Delta
\end{aligned}
$$

The following equations show that $\overline{\eta} : Id_{\Delta\times\mathcal{S}et_{\mathcal{E}}} \stackrel{\cdot}{\longrightarrow} \overline{\boldsymbol{T}}$ and $\overline{\mu} : \overline{\boldsymbol{T}}^2 \stackrel{\cdot}{\longrightarrow} \overline{\boldsymbol{T}}$ are natural transformations.

$$
\begin{aligned}
&(\overline{\boldsymbol{T}}\lceil f\rceil)_\Delta \circ (\overline{\eta})_\Delta \\
&= (\overline{\llbracket \boldsymbol{T}\lceil \zeta X.f\lceil X\rceil\rceil\rrbracket} \circ \overline{\llbracket \eta\rrbracket})_\Delta \\
&= (\overline{\llbracket \boldsymbol{T}\lceil \zeta X.f\lceil X\rceil\rceil \circ \eta\rrbracket})_\Delta \\
&= (\overline{\llbracket \eta\circ f\rrbracket})_\Delta \\
&= (\overline{\llbracket \eta\rrbracket})_\Delta \circ (\overline{\llbracket f\rrbracket})_\Delta \\
&= (\overline{\eta})_\Delta \circ f_\Delta
\end{aligned}
\qquad
\begin{aligned}
&(\overline{\boldsymbol{T}}\lceil f\rceil)_\Delta \circ (\overline{\mu})_\Delta \\
&= (\overline{\llbracket \boldsymbol{T}\lceil \zeta X.f\lceil X\rceil\rceil\rrbracket} \circ \overline{\llbracket \mu\rrbracket})_\Delta \\
&= (\overline{\llbracket \boldsymbol{T}\lceil \zeta X.f\lceil X\rceil\rceil \circ \mu\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\circ \boldsymbol{T}\lceil \boldsymbol{T}\lceil f\rceil\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\rrbracket})_\Delta \circ (\overline{\llbracket \boldsymbol{T}\lceil \boldsymbol{T}\lceil f\rceil\rceil\rrbracket})_\Delta \\
&= (\overline{\mu})_\Delta \circ (\overline{\boldsymbol{T}}\left\lceil \overline{\boldsymbol{T}}\lceil f\rceil\right\rceil{}_\Delta
\end{aligned}
$$

The square and triangular identities ensure that the triple constitutes a monad in a given category.

$$
\begin{aligned}
(\overline{\mu})_\Delta \circ (\overline{\boldsymbol{T}}\lceil\overline{\mu}\rceil)_\Delta &= (\overline{\llbracket \mu\rrbracket})_\Delta \circ (\overline{\llbracket \boldsymbol{T}\lceil \mu\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\circ \boldsymbol{T}\lceil \mu\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\circ \mu\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\rrbracket})_\Delta \circ (\overline{\llbracket \mu\rrbracket})_\Delta \\
&= (\overline{\mu})_\Delta \circ (\overline{\mu})_\Delta
\end{aligned}
\qquad
\begin{aligned}
(\overline{\mu})_\Delta \circ (\overline{\eta})_\Delta &= (\overline{\llbracket \mu\rrbracket})_\Delta \circ (\overline{\llbracket \eta\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\circ \eta\rrbracket})_\Delta \\
&= (\overline{\llbracket \zeta X.X\rrbracket})_\Delta \\
(\overline{\mu})_\Delta \circ (\overline{\boldsymbol{T}}\lceil\overline{\eta}\rceil)_\Delta &= (\overline{\llbracket \mu\rrbracket})_\Delta \circ (\overline{\llbracket \boldsymbol{T}\lceil \eta\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \mu\circ \boldsymbol{T}\lceil \eta\rceil\rrbracket})_\Delta \\
&= (\overline{\llbracket \zeta X.X\rrbracket})_\Delta
\end{aligned}
$$

□

4 Examples of Internal Monads

4.1 Non-determinism

The sentence (4) is ambiguous with respect to at least two factors: the antecedent of the pronoun 'he' and the lexical meaning of 'a suit' (clothing or a legal action).

(4) He brought a suit.

Suppose that there are currently two possible antecedents for the subject pronoun, say 'John' and 'Bill'. Then, in the context of standard natural language processing, a parser is expected to spell out the following set of semantic representations for the input sentence (4).

(5) $\{brought'(suit_1)(j'), brought'(suit_2)(j'),$
$brought'(suit_1)(b'), (brought')(suit_2)(b')\}$

But this kind of 'duplication' of output trees is known to bring about a combinatorial explosion in parsing complexity, which has motivated the pursuit of an 'information packing' strategy. Now, let us consider the following interpretation rules from TLC to TLC.

(6)
$$\begin{aligned} [\![x]\!]_{nd} &= \{x\} \\ [\![\lambda x.M]\!]_{nd} &= \{\lambda x.[\![M]\!]_{nd}\} \\ [\![M(N)]\!]_{nd} &= \{mn \mid m \in [\![M]\!]_{nd} \wedge n \in [\![N]\!]_{nd}\} \\ [\![\langle M, N\rangle]\!]_{nd} &= \{\langle m, n\rangle \mid m \in [\![M]\!]_{nd} \wedge n \in [\![N]\!]_{nd}\} \end{aligned}$$

Then, each ambiguity due to the antecedent of 'he' and the lexical ambiguity of 'a suit' can be lexically represented in the following way.

(7)
$$\begin{aligned} [\![he']\!]_{nd} &= \{j', b'\} \\ [\![suit']\!]_{nd} &= \{suit_1, suit_2\} \end{aligned}$$

Using these expressions, the set of representations (5) can be packed into the single representation (8).

(8) $(brought'(suit'))(he')$

The non-deterministic aspects in the sentence (4) are successfully encapsulated and hidden within (8), for the following equations show that the interpretation $[\![-]\!]_{nd}$ of (8) is equivalent to (5).

(9)
$$\begin{aligned} & [\![(brought'(suit'))]\!]_{nd} \\ &= \{mn \mid m \in [\![brought']\!]_{nd} \wedge n \in [\![suit']\!]_{nd}\} \\ &= \{mn \mid m \in \{brought'\} \wedge n \in \{suit_1, suit_2\}\} \\ &= \{brought'(suit_1), brought'(suit_2)\} \end{aligned}$$

$$\begin{aligned} & [\![(brought'(suit'))(he')]\!]_{nd} \\ &= \{mn \mid m \in [\![brought'(suit')]\!]_{nd} \wedge n \in [\![he']\!]_{nd}\} \\ &= \{mn \mid m \in \{brought'(suit_1), brought'(suit_2)\} \wedge n \in \{j', b'\}\} \\ &= \{brought'(suit_1)(j'), brought(suit_2)(j'), \\ &\quad\ brought'(suit_1)(b'), brought(suit_2)(b')\} \end{aligned}$$

Interpretation $[\![-]\!]_{nd}$ in (6) is specified by the internal monad, consisting of the following triple, which gives rise to the definition (6) via the interpretation schema Definition 8.

Definition 10 (Internal Monad of Non-determinism)

$$\begin{aligned} \boldsymbol{T}_{nd} &= \zeta f.\zeta X.\{f\lceil Y\rceil \mid Y \in X\} \\ \eta_{nd} &= \zeta X.\{X\} \\ \mu_{nd} &= \zeta X.\bigcup X \end{aligned}$$

Theorem 11. $\langle \boldsymbol{T}_{nd}, \eta_{nd}, \mu_{nd}\rangle$ *is an internal monad.*

Proof. $\underline{\boldsymbol{T}\text{ conditions:}}$

$$\begin{aligned}\boldsymbol{T}_{nd}\lceil \zeta X.X\rceil &= \zeta X.\{(\zeta X.X)\lceil Y\rceil \mid Y \in X\}\\ &= \zeta X.\{Y \mid Y \in X\}\\ &= \zeta X.X\end{aligned}$$

$$\begin{aligned}\boldsymbol{T}_{nd}\lceil g\rceil \circ \boldsymbol{T}_{nd}\lceil f\rceil &= \zeta X.\{g\lceil Y\rceil \mid Y \in X\} \circ \zeta X.\{f\lceil Y\rceil \mid Y \in X\}\\ &= \zeta X.\{g\lceil Y\rceil \mid Y \in \{f\lceil Z\rceil \mid Z \in X\}\}\\ &= \zeta X.\{(g \circ f)\lceil Y\rceil \mid Y \in X\}\\ &= \boldsymbol{T}_{nd}\lceil g \circ f\rceil\end{aligned}$$

$\underline{\eta\text{ and }\mu\text{ conditions:}}$

$$\begin{aligned}&\boldsymbol{T}_{nd}\lceil f\rceil \circ \eta_{nd}\\ =&(\zeta X.\{f\lceil Y\rceil \mid Y \in X\}) \circ (\zeta X.\{X\})\\ =&\zeta X.\{f\lceil Y\rceil \mid Y \in \{X\}\}\\ =&\zeta X.\{f\lceil X\rceil\}\\ =&\zeta X.\{X\} \circ f\\ =&\eta_{nd} \circ f\end{aligned}$$

$$\begin{aligned}&\boldsymbol{T}_{nd}\lceil f\rceil \circ \mu_{nd}\\ =&(\zeta X.\{f\lceil Y\rceil \mid Y \in X\}) \circ (\zeta X.\bigcup X)\\ =&\zeta X.\{f\lceil Y\rceil \mid Y \in \bigcup X\}\\ =&\zeta X.\bigcup\{f\lceil Y\rceil \mid Y \in \{f\lceil Z\rceil \mid Z \in X\}\}\\ =&\mu_{nd} \circ \boldsymbol{T}\lceil \boldsymbol{T}\lceil f\rceil\rceil\end{aligned}$$

$\underline{\text{Square identity:}}$

$$\begin{aligned}\mu_{nd} \circ \boldsymbol{T}_{nd}\lceil \mu_{nd}\rceil &= (\zeta X.\bigcup X) \circ \zeta X.\left\{(\zeta X.\bigcup X)\lceil Y\rceil \mid Y \in X\right\}\\ &= (\zeta X.\bigcup X) \circ \zeta X.\left\{\bigcup Y \mid Y \in X\right\}\\ &= (\zeta X.\bigcup X) \circ (\zeta X.\bigcup X)\\ &= \mu_{nd} \circ \mu_{nd}\end{aligned}$$

$\underline{\text{Triangular identity:}}$

$$\begin{aligned}&\mu_{nd} \circ \eta_{nd}\\ =\;&\zeta X.\bigcup X \circ \zeta X.\{X\}\\ =\;&\zeta X.\bigcup\{X\}\\ =\;&\zeta X.X\end{aligned}$$

$$\begin{aligned}&\mu_{nd} \circ \boldsymbol{T}_{nd}\lceil \eta_{nd}\rceil\\ =\;&\zeta X.\bigcup X \circ \zeta X.\{(\zeta X.\{X\})\lceil Y\rceil \mid Y \in X\}\\ =\;&\zeta x.\bigcup\{\{Y\} \mid Y \in X\}\\ =\;&\zeta X.X\end{aligned}$$

□

4.2 Contextual Parameters

Semantic representations sometimes make reference to various kinds of contextual parameters such as speaker/hearer, topic, point of view, for instance. Contextual parameters are often treated as being ‘free variables’ or ‘global variables’, but strictly speaking, this is not accurate, since the their values can be overwritten in the middle of the sentences. For example, in the sentence (10), each of the two occurrences of ‘you’ refers to the hearer at the moment, but its denotation changes by ostention.

(10) (Pointing to John) You passed, (pointing to Mary) and you passed.

Again, if we interpret TLC terms by the following set of rules, contextual parameters can be easily referenced in semantic representations, and can also be changed, even halfway through a single sentence.

(11)
$$
\begin{aligned}
[\![x]\!]_{cp} &= \lambda h.\langle x, h\rangle \\
[\![\lambda x.M]\!]_{cp} &= \lambda h.\langle \lambda x.[\![M]\!]_{cp}, h\rangle \\
[\![M(N)]\!]_{cp} &= \lambda h.let\ M' = \pi_1[\![M]\!]_{cp}h,\ let\ h' = \pi_2[\![M]\!]_{cp}h, \\
&\qquad let\ N' = \pi_1[\![N]\!]_{cp}h',\ let\ h'' = \pi_2[\![N]\!]_{cp}h',\ \langle M'(N'), h''\rangle \\
[\![\langle M, N\rangle]\!]_{cp} &= \lambda h.let\ M' = \pi_1[\![M]\!]_{cp}h,\ let\ h' = \pi_2[\![M]\!]_{cp}h, \\
&\qquad let\ N' = \pi_1[\![N]\!]_{cp}h',\ let\ h'' = \pi_2[\![N]\!]_{cp}h',\ \langle\langle M', N'\rangle, h''\rangle
\end{aligned}
$$

The following definitions provide a method to set the current hearer x to the corresponding contextual parameter, and a method to reference it.

(12)
$$
\begin{aligned}
[\![set\_hearer(x)]\!]_{cp} &= \lambda h.\langle \top, x\rangle \\
[\![hearer()]\!]_{cp} &= \lambda h.\langle h, h\rangle
\end{aligned}
$$

Then, the semantic representation of the sentence (10) can be simply stated as follows.

(13) $set\_hearer(j') \wedge passed'(hearer()) \wedge set\_hearer(m') \wedge passed'(hearer())$

When (13) is interpreted by $[\![-]\!]_{cp}$, each occurrence of 'you' successfully refers to the intended individual, although the two representations for 'you passed' in (13) are exactly the same. Suppose that $A \wedge B = \wedge(\langle A, B\rangle)$.

(14)
$$
\begin{aligned}
[\![\wedge]\!]_{cp} &= \lambda h.\langle \wedge, h\rangle \\
[\![passed']\!]_{cp} &= \lambda h.\langle passed', h\rangle \\
[\![passed'(hearer())]\!]_{cp} &= \lambda h.\langle passed'(h), h\rangle \\
[\![\langle set\_hearer(j'), passed'(hearer())\rangle]\!]_{cp} &= \lambda h.\langle\langle \top, passed'(j')\rangle, j'\rangle \\
[\![set\_hearer(j') \wedge passed'(hearer())]\!]_{cp} &= \lambda h.\langle \wedge(\top, passed'(j')), j'\rangle \\
&= \lambda h.\langle passed'(j'), j'\rangle
\end{aligned}
$$

Therefore, the following result obtains.

(15) $[\![set\_hearer(j') \wedge passed'(hearer()) \wedge set\_hearer(m') \wedge passed'(hearer())]\!]_{cp}$
$= \lambda h.\langle passed'(j') \wedge passed'(m'), m'\rangle$

The interpretation $[\![-]\!]_{cp}$ in (11) is specified by the following internal monad. Again, this gives rise to the definition (11) via the interpretation schema Definition 8.

Definition 12 (Internal Monad of Contextual Parameters)

$$
\begin{aligned}
\boldsymbol{T}_{cp} &= \zeta f.\zeta X.\lambda h.\langle f\ \lceil \pi_1(Xh)\rceil, \pi_2(Xh)\rangle \\
\eta_{cp} &= \zeta X.\lambda h.\langle X, h\rangle \\
\mu_{cp} &= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh))
\end{aligned}
$$

Theorem 13. *$\langle \boldsymbol{T}_{cp}, \eta_{cp}, \mu_{cp} \rangle$ is an internal monad.*

Proof. $\underline{\boldsymbol{T}\text{ conditions:}}$

$$\begin{aligned}
\boldsymbol{T}_{cp}\lceil \zeta X.X \rceil &= \zeta X.\lambda h.\langle (\zeta X.X)\lceil \pi_1(Xh) \rceil, \pi_2(Xh) \rangle \\
&= \zeta X.\lambda h.\langle \pi_1(Xh), \pi_2(Xh) \rangle \\
&= \zeta X.\lambda h.Xh \\
&= \zeta X.X
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{T}_{cp}\lceil g \rceil \circ \boldsymbol{T}_{cp}\lceil f \rceil &= \zeta X.\lambda h.\langle g\lceil \pi_1(Xh) \rceil, \pi_2(Xh) \rangle \circ \zeta X.\lambda h.\langle f\lceil \pi_1(Xh) \rceil, \pi_2(Xh) \rangle \\
&= \zeta X.\lambda h.\langle g\lceil f\lceil \pi_1(Xh) \rceil \rceil, \pi_2(Xh) \rangle \\
&= \boldsymbol{T}_{cp}\lceil g \circ f \rceil
\end{aligned}$$

$\underline{\eta \text{ and } \mu \text{ conditions:}}$

$$\begin{aligned}
\boldsymbol{T}_{cp}\lceil f \rceil \circ \eta_{cp} &= \zeta X.\lambda h.\langle f\lceil \pi_1(Xh) \rceil, \pi_2(Xh) \rangle \circ \zeta X.\lambda h.\langle X, h \rangle \\
&= \zeta X.\lambda h.\langle fX, h \rangle \\
&= \eta_{cp} \circ f
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{T}_{cp}\lceil f \rceil \circ \mu_{cp} &= \zeta X.\lambda h.\langle f\lceil \pi_1(Xh) \rceil, \pi_2(Xh) \rangle \circ \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \\
&= \zeta X.\lambda h.\langle f\lceil \pi_1((\pi_1(Xh))(\pi_2(Xh))) \rceil, \pi_2((\pi_1(Xh))(\pi_2(Xh))) \rangle \\
&= \zeta X.\lambda h.(\lambda h'.\langle f\lceil \pi_1((\pi_1(Xh))h') \rceil, \pi_2((\pi_1(Xh))h') \rangle)(\pi_2(Xh)) \\
&= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \\
&\quad \circ \zeta X.\lambda h.\langle \lambda h'.\langle f\lceil \pi_1((\pi_1(Xh))h') \rceil, \pi_2((\pi_1(Xh))h') \rangle, \pi_2(Xh) \rangle \\
&= \mu_{cp} \circ \boldsymbol{T}\lceil \boldsymbol{T}\lceil f \rceil \rceil
\end{aligned}$$

<u>Square identity:</u>

$$\begin{aligned}
\mu_{cp} \circ \boldsymbol{T}_{cp}\lceil \mu_{cp} \rceil &= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \\
&\quad \circ \zeta X.\lambda h.\langle (\zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)))(\pi_1(Xh)), \pi_2(Xh) \rangle \\
&= \zeta X.\lambda h.(\lambda h'.(\pi_1((\pi_1(Xh))h'))(\pi_2((\pi_1(Xh))h')))(\pi_2(Xh)) \\
&= \zeta X.\lambda h.(\pi_1((\pi_1(Xh))(\pi_2(Xh))))(\pi_2((\pi_1(Xh))(\pi_2(Xh)))) \\
&= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \circ \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \\
&= \mu_{cp} \circ \mu_{cp}
\end{aligned}$$

<u>Triangular identity:</u>

$$\begin{aligned}
\mu_{cp} \circ \eta_{cp} &= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \circ \zeta X.\lambda h.\langle X, h \rangle \\
&= \zeta X.\lambda h.Xh \\
&= \zeta X.X
\end{aligned}$$

$$
\begin{aligned}
\mu_{cp} \circ \boldsymbol{T}_{cp} \lceil \eta_{cp} \rceil &= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \\
&\quad \circ \zeta X.\lambda h.\langle (\zeta X.\lambda h.\langle X, h\rangle)(\pi_1(Xh)), \pi_2(Xh)\rangle \\
&= \zeta X.\lambda h.(\pi_1(Xh))(\pi_2(Xh)) \circ \ \zeta X.\lambda h.\langle \lambda h.\langle \pi_1(Xh), h\rangle, \pi_2(Xh)\rangle \\
&= \zeta X.\lambda h.(\lambda h.\langle \pi_1(Xh), h\rangle)(\pi_2(Xh)) \\
&= \zeta X.\lambda h.\langle \pi_1(Xh), \pi_2(Xh)\rangle \\
&= \zeta X.\lambda h.Xh \\
&= \zeta X.X
\end{aligned}
$$

□

5 Conclusion

Meta-Lambda Calculus (MLC) is an extended TLC with meta-constructions, whose categorical semantics is defined by means of a hom-functor from a Cartesian closed category to $\mathcal{S}et$. In this setting, TLC is naturally regarded as a special case of MLC, namely, MLC with no meta-variables. I also proved that both normal and meta- beta conversions are sound in this categorical semantics.

Each computational monad in "monadic analyses" is specified by an "internal monad", which is a triple of MLC terms, that serves as a parameter in the transformation rules. I proved that if a triple satisfies the set of conditions ($\boldsymbol{T}$ conditions, η and μ conditions, square identity, triangular identity), there exists a categorical monad in "Δ-indexed category" which exactly corresponds to that triple.

As examples of such computational monads, I presented two internal monads for non-determinism and contextual parameters, and proved that they indeed satisfy the conditions for internal monads.

References

[Crole (1993)] Crole, R.L.: Categories for Types. Cambridge University Press, Cambridge (1993)

[Kock (1970)] Kock, A.: Strong functors and monoidal monads, Various Publications Series 11. Aarhus Universitet (August 1970)

[Lambek (1980)] Lambek, J.: From λ-calculus to cartesian closed categories. In: Seldin, J.P., Hindley, J.R. (eds.) To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism. Academic Press, London (1980)

[Lambek and Scott (1986)] Lambek, J., Scott, P.J.: Introduction to higher order categorical logic. Cambridge University Press, Cambridge (1986)

[MacLane (1997)] MacLane, S.: Categories for the Working Mathematician, 2nd edn. Graduate Texts in Mathematics. Springer, Heidelberg (1997)

[Moggi (1989)] Moggi, E.: Computational lambda-calculus and monads. In: Proceedings of Fourth Annual IEEE Symposium on Logic in Computer Science, pp. 14–23 (1989)

[Shan (2001)] Shan, C.-c.: Monads for natural language semantics. In: Striegnitz, K. (ed.) The ESSLLI-2001 student session (13th European summer school in logic, language and information, 2001), pp. 285–298 (2001)

[Wadler (1992)] Wadler, P.: Comprehending Monads. In: Mathematical Structure in Computer Science, vol. 2, pp. 461–493. Cambridge University Press, Cambridge (1992)