A Contextual Reconstruction of Monadic Reflection

Toru Kawata
Department of Computer Science, The University of Tokyo, Tokyo, Japan
kawata@lyon.is.s.u-tokyo.ac.jp

Abstract

With the help of an idea of contextual modal logic, we define a logical system λ^{refl} that incorporates monadic reflection, and then investigate delimited continuations through the lens of monadic reflection. Technically, we firstly prove a certain universality of continuation monad, making the character of monadic reflection a little more clear. Next, moving focus to delimited continuations, we present a macro definition of shift/reset by monadic reflection. We then prove that $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$, a restriction of λ^{refl}, has exactly the same provability as $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$, a system that incorporates shift/reset. Our reconstruction of monadic reflection opens up a path for investigation of delimited continuations with familiar monadic language.

2012 ACM Subject Classification Theory of computation → Type theory

Keywords and phrases Monadic Reflection, Delimited Continuations, shift/reset, Contextual Modal Logic, Curry-Howard Isomorphism

Digital Object Identifier 10.4230/LIPIcs.CSL.2018.27

Acknowledgements I would like to thank Yoshihiko Kakutani, Yuichi Nishiwaki, and Yuito Murase for their valuable comments on contextual modality. I also thank the anonymous reviewers for their valuable comments.

1 Introduction

Every light is accompanied by darkness. Every term is accompanied by its continuation. Suppose that the part indicated by the underline in the following term is being evaluated:

$$1 + \underline{2 \times 3} - 4$$

In this case, the continuation of 2×3 is intuitively $\mathcal{A}\,(1 + [\,] - 4)$, where $\mathcal{A}\,(M)$ means "compute M and then quit". In other words, a continuation is the rest of the computation, and is therefore expressed as a term with just one "hole". Some calculi[5][21] can turn a continuation into a function and use it in terms as if it were an ordinary function. Such manipulations of continuations are well-understood today, and indeed it is a well-known fact that these can be characterized as classical inferences via the Curry-Howard Isomorphism[10][21]. In the correspondence, the type of $\mathcal{A}\,(1 + [\,] - 4)$ is understood to be $\mathbb{N} \to \bot$, assuming that $\mathbb{N}$ is the type of natural numbers.

Delimited continuation[4][6][11] is a variant of continuation. In operational terms, a delimited continuation is explained as the rest of the computation *up to the nearest enclosing delimiter*. For example, consider the following term:

$$\lceil 1 + \underline{2 \times 3} \rceil - 4$$

where we denote the delimiter by $\lceil - \rceil$. In this case, the delimited continuation of 2×3 is $1 + [\,]$. As in the case of ordinary continuations, there are calculi that can turn a delimited

continuation into a function[4][15]. This perhaps seemingly trivial modification is in fact significant, enhancing the expressibility of continuations drastically. Delimited continuations are so expressive that it can realize, for example, coroutines, non-determinisim, and statically typed printf[2]. This expressibility essentially comes from the fact that delimited continuations are composable as functions, whereas ordinary continuations are not since their codomains are $\bot$.

In addition to the above intriguing applications of delimited continuations, there exists one more elegant use of it: *monadic reflection*[7][8]. Monadic reflections allow programmers to write programs that involve monads as if those monads are "built-in" to the language, or in Haskell terminology, without do-notation. A little more specifically, in a system with monadic reflection, a monadic term $M : TA$ can be turned into a non-monadic term $\mathsf{reflect}^T M : A$ without disrupting the intuitive behavior of M, easing the trouble of writing effectful programs in a purely-functional language.

On the other hand, however, delimited continuations are logically complicated. In contrast to the fact that ordinary continuations correspond to classical inferences, there does not seem to exist such a simple correspondence for delimited ones. Although there exists, for example, an interesting work that embeds a system with delimited continuations into a variant of classical logic[1], the embedding is still not "the end of the story" in that the original system with delimited continuations[4] does not allow classical inferences.

Thus, to sum up the plot, an elegant application is being derived from a rather opaque starting point. In this paper, with the help of an idea of contextual modal logic, we invert the direction of this storyline: We directly define a logical system λ^{refl} that incorporates monadic reflection, and then investigate delimited continuations through the lens of monadic reflection, obtaining a better understanding of delimited continuations. Our contribution in this paper is now summaried as follows:

- A modal logical system that incorporates monadic reflection,
- The universality of continuation monad with respect to monadic reflection,
- Logical understanding of delimited continuations via "quasi-double negation",
- Equivalence of monadic reflection and delimited continuation in provability.

In the next section, we overview the idea of Fitch-style contextual modal logic to obtain the foundational idea of context layering. In the section, we also briefly review ordinary modal logic since it might be an unfamiliar concept for researchers on delimited continuations. Readers who are familiar with modal logic can safely skip the first part of the section. Using the idea of contextual modal logic, in Section 3, we define a system λ^{refl} that incorporates monadic reflection, and show the univesality of continuation monad with respect to reflection, making the character of monadic reflection a little more clear. We also present a macro definition of shift/reset in the section. Section 4 is dedicated to an investigation of delimited continuations via the lens of monadic reflection. We show certain equivalence of these two concepts with respect to provability in the section. In Section 5, we put our contribution into perspective by comparing our work to related work. Section 6 concludes this paper with discussion of future work.

2 Fitch-Style Contextual Modal Logic

2.1 A Quick Tour of Modal Logic

Modal logic is a logic in which we can add certain modalities, or "flavors", to propositions. In modal logic, one can state that a proposition is "necessarily" true, "possibly" true, or "should be" true, etc. Let us focus on necessity and possibility here. Typically, when a

proposition A is necessarily or possibly true under an assumption Γ, one writes $\Gamma \vdash \Box A$ or $\Gamma \vdash \Diamond A$, respectively. Using these operators, one can extend ordinary logic into modal one.

But what after all is necessity? What is possibility? The critical problem of modal logic in its early days was the very fact that people did not share any firm consensus on these concepts. Is a proposition $\Box A \to \Box\Box A$ true? How about $\Box(A \to B) \to (\Box A \to \Box B)$? Since people did not have any precise definitions of modalities, the answer naturally differed among people, which resulted in a lot of different variant of modal logics.

Fortunately, around 1960, this situation was recovered by Kripke[16][17], who presented a semantics for modal logic. He presented formal definitions of the two modalities in his semantics which employs a concept that would excite Sci-Fi enthusiasts and even others: possible worlds. His semantics introduces a set of possible worlds and an "accessibility relation" between them. Under this framework, for example, the meaning of necessity is defined as follows: $\Box A$ is true in a possible world w if and only if A is true in any world w' that is accessible from the world w. The various aspects of necessity are then characterized by how this accessibility relation is defined. For instance, if the accessibility relation under consideration is reflexive, the box modality validates $\Box A \to A$. If the relation if transitive, the modality validates $\Box A \to \Box\Box A$, etc. In this way, he gave formal definitions of "necessity" and "possibility", or whatever they are called, in modal logic.

2.2 Fitch-Style Contextual Modal Logic

There still remains a pitfall. Modalities become tricky when they are combined with bindings. Let us take the example by Quine[23]. The sentence $\Box(8 > 7)$ is true as long as our box modality considered here validates $A \to \Box A$. Now, we know that the number of the planets in our solar system is 8. Let us define

$$N := (\text{the number of the planets in our solar system}).$$

Then $N = 8$ is true. Now, however, the statement $\Box(N > 7)$ is intuitively false, since we can consider a possible world in which we have, for instance, only 5 planets in our solar system. This phenomenon suggests that box modality has a delicate character with respect to bindings. Some might think that the box modality should contain information about its argument just like universal/existential quantifiers.

And here comes contextual modal logic. Contextual modal logic[19] is an attempt to generalize the modalities with contextual information. In this logic, the box modality is generalized from $\Box A$ to $[\Gamma]A$. The latter proposition intuitively reads as "A is necessarily true under Γ". $[\Gamma]A$ degenerates to the ordinary box modality when Γ is empty. By generalizing box modality in this way, one can obtain a more fine-grained modality.

Multi-level Fitch-style contextual modal logic[20] is a variant of contextual modal logic. In the logic, the form of judgments are generalized to $\Gamma_1\,;\ \ldots\ ;\ \Gamma_n \vdash A$. Intuitively, this reads as "A is true under Γ_n, is true under Γ_{n-1}, ..., is true under Γ_1". In other words, this logic extends our vocabulary in the system, allowing statements of the form not only

$$A \text{ is true under } \Gamma$$

but also

$$\text{"}A \text{ is true under } \Gamma_2\text{" is true under } \Gamma_1.$$

By internalizing the meaning of this statement, we can incorporate quotation to our logic. For example, we can consider the following inference rules

$$\frac{\vec{\Pi}\,;\ \Gamma \vdash A}{\vec{\Pi} \vdash [\Gamma]A}\,\Box_{\mathsf{i}} \qquad \frac{\vec{\Pi} \vdash [\Gamma]A}{\vec{\Pi}\,;\ \Gamma \vdash A}\,\Box_{\mathsf{e}}$$

which internalize quotation, or the relative pronoun "that", as the box modality. It would be worth noting that this box modality defined above characterizes the so-called K modality. Furthermore, by extending these rules in a certain way, we can also characterize T, K4, and S4 modalities. Interested readers are referred to [18][20]. In this paper, however, we think that this quick explanation is sufficient for our purpose, and therefore do not go into further details here.

3 A Contextual Reconstruction of Monadic Reflection

3.1 A Logical System with Monadic Reflection

3.1.1 Syntax

The syntax of λ^{refl} is defined as follows. Here, the x and t in the following definition are a variable and a type-variable, respectively. We assume that the set of variables and that of type-variables are distinct.

$$\begin{aligned}
A, B &::= t \mid A \to A \\
T &::= [\,] \mid t \mid T \to T \\
M, N, P, Q &::= x \mid \lambda x.\, M \mid M@M \mid \mathsf{reflect}^T M \mid \mathsf{reify}^T M \\
\Gamma, \Delta &::= \emptyset \mid x : A, \Gamma \\
\vec{\Pi} &::= \cdot \mid \Gamma\,;\ \vec{\Pi} \\
\vec{L} &::= \cdot \mid T\,;\ \vec{L}
\end{aligned}$$

We define $T[A]$ to be the proposition obtained by replacing all the occurrances of $[\,]$ in T with A. For example, when $T = [\,] \to B$, $T[A]$ is $A \to B$. Although the definition of T in the rule above is somewhat restricted, we can easily generalize it if necessary. We denote $T[A]$ by TA. We also abbreviate $\Gamma_1\,;\ \ldots\ ;\ \Gamma_n\,;\ \cdot$ as $\Gamma_1\,;\ \ldots\ ;\ \Gamma_n$, and $T_1\,;\ \ldots\ ;\ T_n\,;\ \cdot$ as $T_1\,;\ \ldots\ ;\ T_n$. Every judgment of λ^{refl} is of the form $\vec{L} \mid \vec{\Pi} \vdash M : A$.

3.1.2 Logic

The type system of λ^{refl} is as follows.

$$\frac{}{\vec{L}\,;\ T \mid \vec{\Pi}\,;\ \Gamma, x : A \vdash x : A}\,\mathsf{var}$$

$$\frac{\vec{L} \mid \vec{\Pi}\,;\ \Gamma, x : A \vdash M : B}{\vec{L} \mid \vec{\Pi}\,;\ \Gamma \vdash \lambda x.\, M : A \to B}\,\to_{\mathsf{i}} \qquad \frac{\vec{L} \mid \vec{\Pi} \vdash M : A \to B \qquad \vec{L} \mid \vec{\Pi} \vdash N : A}{\vec{L} \mid \vec{\Pi} \vdash M@N : B}\,\to_{\mathsf{e}}$$

$$\frac{T\,;\ \vec{L} \mid \emptyset\,;\ \vec{\Pi} \vdash M : A}{\vec{L} \mid \vec{\Pi} \vdash \mathsf{reify}^T M : TA}\,\mathsf{reify} \qquad \frac{\vec{L} \mid \vec{\Pi} \vdash M : TA}{T\,;\ \vec{L} \mid \Gamma\,;\ \vec{\Pi} \vdash \mathsf{reflect}^T M : A}\,\mathsf{reflect}$$

There exist two side-conditions. Firstly, the rule $\to_{\mathsf{i}}$ can be applied only when M does not have any $\mathsf{reflect}$s that are not encapsulated by reify. This side-condition is required to define the reduction of this system in the ordinary call-by-value way. At the same time, however, this side-condition can seemingly be dropped by allowing reduction in abstraction, and we will revisit this point later. Secondly, when the rules reify or $\mathsf{reflect}$ are applied, the following two rules must be admissible without appealing reify and $\mathsf{reflect}$:

$$\frac{\vec{L} \mid \vec{\Pi} \vdash M : A}{\vec{L} \mid \vec{\Pi} \vdash \mathsf{return}^T\, M : TA}\ \mathsf{return}$$

$$\frac{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash M : TA \qquad \vec{L} \mid \vec{\Pi};\ \Gamma, x : A \vdash N : TB}{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash M \rhd_x^T N : TB}\ \mathsf{bind}$$

where $\mathsf{return}^T\, M$ and $M \rhd_x^T N$ are "macros" defined by terms in the system[1]. For example, suppose $T = [\,]$. For this identity effect T, the rule return is vacuously admissible. The rule bind is also admissible for this effect since we have the following derivation tree:

$$\frac{\dfrac{\vec{L} \mid \vec{\Pi};\ \Gamma, x : A \vdash N : B}{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash \lambda x.\, N : A \to B} \qquad \vec{L} \mid \vec{\Pi};\ \Gamma \vdash M : A}{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash (\lambda x.\, N)@M : B}$$

Thus, we are allowed to apply the rules reify and $\mathsf{reflect}$ when $T = [\,]$. In this case, $\mathsf{return}^{[\,]}\, M$ and $M \rhd_x^{[\,]} N$ are understood as macros for M and $(\lambda x.\, N)@M$, respectively.

▶ **Definition 1.** A proposition A is *provable in* λ^{refl} when there exists a term N such that $[\,] \mid \emptyset \vdash N : A$ is derivable in λ^{refl}. Such N is said to be *well-typed*.

We write $\lambda^{\mathsf{refl}} \vdash A$ when A is provable in λ^{refl}. We also write $\lambda^{\mathsf{refl}} \vdash \mathcal{J}$ when $\mathcal{J}$ is derivable in λ^{refl}. We use these notations for subsystems of λ^{refl} that we will define later, too.

3.1.3 Reduction

Values V, contexts E, and pure-contexts F are defined as follows.

$$\begin{aligned} V &::= x \mid \lambda x.\, M \\ E &::= [\,] \mid V@E \mid E@M \mid \mathsf{reify}^T\, E \\ F &::= [\,] \mid V@F \mid F@M \end{aligned}$$

We define $E[M]$ in the same way as $T[A]$. The reduction of this system is defined as follows.

$$\begin{aligned} E[(\lambda x.\, M)@V] &\rightsquigarrow E[M\{x := V\}] \\ E[\mathsf{reify}^T\, F[\mathsf{reflect}^T\, M]] &\rightsquigarrow E[M \rhd_x^T \mathsf{reify}^T\, F[x]] \\ E[\mathsf{reify}^T\, V] &\rightsquigarrow E[\mathsf{return}^T\, V] \end{aligned}$$

Here, the second rule "factors out" a detour resulted from reify/reflect. Specifically, the rule translates a derivation tree

[1] Note that we do not impose the laws of monads here.

$$\dfrac{\dfrac{\dfrac{\dfrac{(\cdots)}{\vec{L} \mid \vec{\Pi} \vdash M : TA}}{T\,;\ \vec{L} \mid \emptyset\,;\ \vec{\Pi} \vdash \mathsf{reflect}^T M : A}}{T\,;\ \vec{L} \mid \emptyset\,;\ \vec{\Pi} \vdash F[\mathsf{reflect}^T M] : B}}{\vec{L} \mid \vec{\Pi} \vdash \mathsf{reify}^T F[\mathsf{reflect}^T M] : TB}$$

into

$$\dfrac{\dfrac{(\cdots)}{\vec{L} \mid \vec{\Pi} \vdash M : TA} \qquad \dfrac{\dfrac{\dfrac{}{T\,;\ \vec{L} \mid \emptyset\,;\ \vec{\Pi}, x : A \vdash x : A}}{T\,;\ \vec{L} \mid \emptyset\,;\ \vec{\Pi}, x : A \vdash F[x] : B}}{\vec{L} \mid \vec{\Pi}, x : A \vdash \mathsf{reify}^T F[x] : TB}}{\vec{L} \mid \vec{\Pi} \vdash M \rhd_x^T \mathsf{reify}^T F[x] : TB}$$

The following elementary properties can be shown by ordinary induction.

▶ **Proposition 2.** *If a judgment $\vec{L} \mid \vec{\Pi}\,;\ \Gamma \vdash M : A$ is derivable in λ^{refl}, $\vec{L} \mid \vec{\Pi}\,;\ \Gamma, x : B \vdash M : A$ is also derivable for any fresh variable x and any type B.*

▶ **Proposition 3.** *If M is a well-typed term of type A, $M \rightsquigarrow N$ implies $N : A$.*

▶ **Proposition 4.** *If $\vec{L} \mid \vec{\emptyset} \vdash M : A$ is derivable, M is one of the followings:*
- $M = \lambda x.\, P$
- $M = E[R]$
- $M = F[\mathsf{reflect}^T P]$

where R is one of $(\lambda x.\, P)@V, \mathsf{reify}^T F[\mathsf{reflect}^T P], \mathsf{reify}^T V$.

The last proposition implies the progress property:

▶ **Corollary 5.** *Every well-typed term M is either a value, or can be uniquely written as $E[R]$, where R is one of $(\lambda x.\, P)@V, \mathsf{reify}^T F[\mathsf{reflect}^T P], \mathsf{reify}^T V$.*

3.2 $\lambda_2^{\mathsf{refl}}, \lambda_{2\mathsf{cont}}^{\mathsf{refl}}$: Restrictions of λ^{refl}

Now, let us investigate the character of continuation monad. The system $\lambda_2^{\mathsf{refl}}$ is defined to be the system obtained from λ^{refl} by restricting the form of judgments to the following two:
- $[\,] \mid \Gamma \vdash M : A$
- $T\,;\ [\,] \mid \emptyset\,;\ \Gamma \vdash M : A$

The system $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$ is also defined to be the system obtained by restricting the form of judgments to the following two:
- $[\,] \mid \Gamma \vdash M : A$
- $\mathsf{cont}_B\,;\ [\,] \mid \emptyset\,;\ \Gamma \vdash M : A$

where cont_A stands for $([\,] \to A) \to A$. For this $\mathsf{cont}_{(-)}$, the rule return is admissible by the following derivation:

$$\dfrac{\dfrac{}{\vec{L} \mid \vec{\Pi}\,;\ \Gamma, k : A \to B \vdash k : A \to B} \qquad \dfrac{\vec{L} \mid \vec{\Pi}\,;\ \Gamma \vdash M : A}{\vec{L} \mid \vec{\Pi}\,;\ \Gamma, k : A \to B \vdash M : A}}{\dfrac{\vec{L} \mid \vec{\Pi}\,;\ \Gamma, k : A \to B \vdash k@M : B}{\vec{L} \mid \vec{\Pi}\,;\ \Gamma \vdash \lambda k.\, k@M : \mathsf{cont}_B A}}$$

and the rule bind is also admissible as follows:

$$\dfrac{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash M : \mathsf{cont}_C A \qquad \dfrac{\dfrac{\vec{L} \mid \vec{\Pi};\ \Gamma, x : A \vdash N : \mathsf{cont}_C B}{\vec{L} \mid \vec{\Pi};\ \Delta \vdash N : \mathsf{cont}_C B} \qquad \dfrac{\dfrac{}{\vec{L} \mid \vec{\Pi};\ \Delta \vdash w : B \to C}}{\vec{L} \mid \vec{\Pi};\ \Delta \vdash N@w : C}}{\vec{L} \mid \vec{\Pi};\ \Gamma, w : B \to C \vdash \lambda x.\, N@w : A \to C}}{\dfrac{\vec{L} \mid \vec{\Pi};\ \Gamma, w : B \to C \vdash M@(\lambda x.\, N@w) : C}{\vec{L} \mid \vec{\Pi};\ \Gamma \vdash \lambda w.\, M@(\lambda x.\, N@w) : \mathsf{cont}_C B}}$$

where Δ is a shorthand for $\Gamma, x : A, w : B \to C$. These admissibilities give us a license to apply reify and reflect for this effect.

We will write $\mathsf{reify}^A M$ for $\mathsf{reify}^{\mathsf{cont}_A} M$ and $\mathsf{reflect}^A M$ for $\mathsf{reflect}^{\mathsf{cont}_A} M$.

3.3 Universality of Continuation Monad

The system $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$ is obviously a subsystem of $\lambda_2^{\mathsf{refl}}$, and therefore any proposition that can be proved in $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$ can also be proved in $\lambda_2^{\mathsf{refl}}$. Here, we show that the other direction is in fact also true.

▶ **Theorem 6.** *Let $\mathcal{J}$ be a derivable judgment in $\lambda_2^{\mathsf{refl}}$. If $\mathcal{J}$ is of the form $[\,] \mid \Gamma \vdash M : A$, there exists a term N such that $[\,] \mid \Gamma \vdash N : A$ is derivable in $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$. If $\mathcal{J}$ is of the form $T;\ [\,] \mid \emptyset;\ \Gamma \vdash M : A$, for any type B, there exists a term N such that the judgment $\mathsf{cont}_{TB};\ [\,] \mid \emptyset;\ \Gamma \vdash N : A$ is derivable in $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$.*

Proof. We prove the statement by induction on the derivation of $\mathcal{J}$. When $\mathcal{J}$ is derived from var, $\to_{\mathsf{i}}$, or $\to_{\mathsf{e}}$, the proofs are routine. Suppose that $\mathcal{J}$ is derived from reflect and of the form $T;\ [\,] \mid \emptyset;\ \Gamma \vdash \mathsf{reflect}^T M : A$. In this case we have $[\,] \mid \Gamma \vdash M : TA$. By the induction hypothesis, there exists a term N such that $[\,] \mid \Gamma \vdash N : TA$ is derivable in $\lambda_{2\mathsf{cont}}^{\mathsf{refl}}$. Now, we can construct the following valid deriviation tree for any type B:

$$\dfrac{\dfrac{\dfrac{[\,] \mid \Gamma \vdash N : TA}{[\,] \mid \Gamma, k : A \to TB \vdash N : TA} \qquad \dfrac{\dfrac{}{[\,] \mid \Delta \vdash k : A \to TB} \qquad \dfrac{}{[\,] \mid \Delta \vdash x : A}}{[\,] \mid \Gamma, k : A \to TB, x : A \vdash k@x : TB}}{[\,] \mid \Gamma, k : A \to TB \vdash N \rhd_x^T k@x : TB}}{\dfrac{[\,] \mid \Gamma \vdash \lambda k.\, N \rhd_x^T k@x : (A \to TB) \to TB}{\mathsf{cont}_{TB};\ [\,] \mid \emptyset;\ \Gamma \vdash \mathsf{reflect}^{TB}(\lambda k.\, N \rhd_x^T k@x) : A}}$$

where the Δ is a shorthand for $\Gamma, k : A \to TB, x : A$.

Suppose that $\mathcal{J}$ is derived from reify and of the form $[\,] \mid \Gamma \vdash \mathsf{reify}^T M : TA$. In this case, we have $T;\ [\,] \mid \emptyset;\ \Gamma \vdash M : A$. Just as in the previous case, for any type B, there exists a term N such that $\mathsf{cont}_{TB};\ [\,] \mid \emptyset;\ \Gamma \vdash N : A$. Since the B in this judgment is arbitrary, we can take it to be A. In other words, we have $\mathsf{cont}_{TA};\ [\,] \mid \emptyset;\ \Gamma \vdash N : A$. Now we just need to construct the following derivation tree:

$$\dfrac{\mathsf{cont}_{TA};\ [\,] \mid \emptyset;\ \Gamma \vdash N : A}{[\,] \mid \Gamma \vdash \mathsf{reify}^{TA} N : (A \to TA) \to TA} \qquad \dfrac{\dfrac{\dfrac{}{[\,] \mid \Gamma, x : A \vdash x : A}}{[\,] \mid \Gamma, x : A \vdash \mathsf{return}^T x : TA}}{[\,] \mid \Gamma \vdash \lambda x.\, \mathsf{return}^T x : A \to TA}$$
$$[\,] \mid \Gamma \vdash (\mathsf{reify}^{TA} N)@(\lambda x.\, \mathsf{return}^T x) : TA$$

◀

▶ **Corollary 7.** *$\lambda_2^{\mathsf{refl}} \vdash A$ iff $\lambda_{2\mathsf{cont}}^{\mathsf{refl}} \vdash A$.*

Note that we had to state the theorem only for provability, and not for term conversion. One might wonder why we did not define, for example, a translation by

$$\begin{aligned}
x^\natural &= x \\
(\lambda x.\, M)^\natural &= \lambda x.\, M^\natural \\
(M@N)^\natural &= M^\natural @ N^\natural \\
(\mathsf{reflect}^T M)^\natural &= \mathsf{reflect}^{TB}(\lambda k.\, M^\natural \rhd_x^T k@x) \\
(\mathsf{reify}^T M)^\natural &= (\mathsf{reify}^{TA} M^\natural)@(\lambda x.\, \mathsf{return}^T\, x)
\end{aligned}$$

and state that $M \rightsquigarrow N \Leftrightarrow M^\natural \rightsquigarrow N^\natural$. This is because the translation makes the correspondence of redexes unclear. For example, in the following diagram, we cannot reduce the upper-right term into the lower-right term because the "redex" $1+1$ is encapsulated in the lambda abstraction.

$$\begin{array}{ccc}
\mathsf{reflect}^T F[1+1] & \overset{\natural}{\rightsquigarrow} & \mathsf{reflect}^{TB}(\lambda k.\, F^\natural[1+1] \rhd_x^T k@x) \\
{\scriptstyle \mathrm{reduce}}\downarrow & & \vdots\,{\scriptstyle ?} \\
\mathsf{reflect}^T F[2] & \overset{\natural}{\rightsquigarrow} & \mathsf{reflect}^{TB}(\lambda k.\, F^\natural[2] \rhd_x^T k@x)
\end{array}$$

This fact complicates discussion on term conversion. Faced with the complexity, we had to state the theorem only for provability, albeit we believe that our results can be strengthened to include an account of term conversion. In Section 4, we state the equivalence of monadic reflection and delimited continuations again "up to provability". It is also because of this.

3.4 Deriving shift/reset from Monadic Reflection

When we instantiate the T in the rule $\mathsf{reflect}$ with cont_B, the rule behaves as if it were a rule of double-negation:

$$\frac{\vec{L} \mid \vec{\Pi} \vdash M : (A \to B) \to B}{\mathsf{cont}_B\,;\ \vec{L} \mid \Gamma\,;\ \vec{\Pi} \vdash \mathsf{reflect}^B M : A}\ \mathsf{reflect}$$

Of couse, this is not a properly classical inference since we do have a "debt" cont_B. Still, it would be natural to expect that we might be able to handle continuations in some form using this "quasi-double negation", considering that classical inferences correspond to manipulations of continuations via the Curry-Howard Isomorphism.

And this is in fact true. In λ^{refl}, we can define the following macros:

$$\begin{aligned}
\lceil M \rceil^A &= (\mathsf{reify}^A M)@(\lambda z.\, z) \\
\mathcal{S}^A k.\, M &= \mathsf{reflect}^A(\lambda k.\, \lceil M \rceil^A)
\end{aligned}$$

These macros behave in exactly the same way as specified in the ordinary operational semantics of shift/reset. Namely, we can easily check the followings:

$$\begin{aligned}
\lceil F[\mathcal{S}k.\, M] \rceil &\rightsquigarrow^* \lceil M\{k := \lambda x.\, \lceil F[x] \rceil\} \rceil, \\
\lceil V \rceil &\rightsquigarrow^* V.
\end{aligned}$$

Specifically, the former is justified by:

$$\begin{aligned}
&\lceil F[\mathcal{S}k.\, M]\rceil \\
&= (\mathsf{reify}\,(F[\mathsf{reflect}\,(\lambda k.\, \lceil M\rceil)]))@(\lambda z.\, z) \\
&\rightsquigarrow ((\lambda k.\, \lceil M\rceil) \rhd_x \mathsf{reify}\,(F[x]))@(\lambda z.\, z) \\
&= (\lambda w.\, (\lambda k.\, \lceil M\rceil)@(\lambda x.\, (\mathsf{reify}\,(F[x]))@w))@(\lambda z.\, z) \\
&\rightsquigarrow (\lambda k.\, \lceil M\rceil)@(\lambda x.\, (\mathsf{reify}\,(F[x]))@(\lambda z.\, z)) \\
&= (\lambda k.\, \lceil M\rceil)@(\lambda x.\, \lceil F[x]\rceil) \\
&\rightsquigarrow \lceil M\{k := \lambda x.\, \lceil F[x]\rceil\}\rceil
\end{aligned}$$

and the latter by:

$$\lceil V\rceil = (\mathsf{reify}\, V)@(\lambda z.\, z) \rightsquigarrow (\mathsf{return}\, V)@(\lambda z.\, z) = (\lambda k.\, k@V)@(\lambda z.\, z) \rightsquigarrow^* V.$$

4 Investigating Delimited Continuations with Monadic Reflection

4.1 A Logical System with Delimited Continuations

Our logical system with delimited continuations, $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$, is obtained from $\lambda^{\mathsf{s/r}}_{\mathsf{let}}$ [3], by

- fixing the answer types,
- restricting lambda abstractions to be pure, and
- making the application of the rule exp explicit.

4.1.1 Syntax

The syntax of $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ is defined as follows.

$$\begin{aligned}
A, B &::= t \mid A \to A \\
M, N &::= x \mid \lambda x.\, M \mid M@N \mid \mathcal{S}^A k.\, M \mid \lceil M\rceil^A \mid \lfloor M\rfloor^A \\
\Gamma, \Delta &::= \emptyset \mid x : A, \Gamma
\end{aligned}$$

We often omit the type annotation in $\mathcal{S}^A k.\, M$, $\lceil M\rceil^A$, and $\lfloor M\rfloor^A$. The form of the judgments of $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ is either $\Gamma \vdash M : A$ or $B \mid \Gamma \vdash M : A$.

4.1.2 Logic

The type system of $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ is as follows.

$$\frac{}{\Gamma, x : A \vdash x : A}\ \mathsf{var}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.\, M : A \to B} \to_{\mathsf{i}} \qquad \frac{C \mid \Gamma \vdash M : A \to B \qquad C \mid \Gamma \vdash N : A}{C \mid \Gamma \vdash M@N : B} \to_{\mathsf{e}}$$

$$\frac{B \mid \Gamma, k : A \to B \vdash M : B}{B \mid \Gamma \vdash \mathcal{S}^B k.\, M : A}\ \mathsf{shift} \qquad \frac{A \mid \Gamma \vdash M : A}{\Gamma \vdash \lceil M\rceil^A : A}\ \mathsf{reset} \qquad \frac{\Gamma \vdash M : A}{B \mid \Gamma \vdash \lfloor M\rfloor^B : A}\ \mathsf{exp}$$

▶ **Definition 8.** A proposition A is *provable in* $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ if there exists a term N such that $\emptyset \vdash N : A$. We also say that such N is *well-typed*.

We define $\lambda^{\mathsf{s/r}}_{\mathsf{pure}} \vdash A$ and $\lambda^{\mathsf{s/r}}_{\mathsf{pure}} \vdash \mathcal{J}$ as in λ^{refl}.

4.2 Translations

We will compare $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ with $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$. Towards that end, we syntactically distinguish the applications of $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$ at different levels. In other words, we separate the rule $\to_{\mathsf{e}}$ in $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$ into the following two:

$$\frac{[\,]\mid\Gamma\vdash M:A\to B\qquad[\,]\mid\Gamma\vdash N:A}{[\,]\mid\Gamma\vdash M\diamond^B N:B}\to^{\mathsf{pure}}_{\mathsf{e}}$$

$$\frac{\mathsf{cont}_C\,;\ [\,]\mid\emptyset\,;\ \Gamma\vdash M:A\to B\qquad\mathsf{cont}_C\,;\ [\,]\mid\emptyset\,;\ \Gamma\vdash N:A}{\mathsf{cont}_C\,;\ [\,]\mid\emptyset\,;\ \Gamma\vdash M@N:B}\to_{\mathsf{e}}$$

We sometimes omit the annotation in the former application for brevity.

With the preparation above, we define two translations $(-)^\sharp:\lambda^{\mathsf{s/r}}_{\mathsf{pure}}\to\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$ and $(-)^\flat:\lambda^{\mathsf{refl}}_{\mathsf{2cont}}\to\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ as follows:

$$\begin{aligned}
x^\flat &= x & x^\sharp &= x\\
(\lambda x.\,M)^\flat &= \lambda x.\,M^\flat & (\lambda x.\,M)^\sharp &= \lambda x.\,M^\sharp\\
(M@N)^\flat &= M^\flat@N^\flat & (M@N)^\sharp &= M^\sharp@N^\sharp\\
(M\diamond^A N)^\flat &= \lceil\lfloor M^\flat\rfloor^A@\lfloor N^\flat\rfloor^A\rceil^A & &-\\
(\mathsf{reflect}^A M)^\flat &= \mathcal{S}^A k.\,\lfloor M^\flat\rfloor^A@\lfloor k\rfloor^A & (\mathcal{S}^A k.\,M)^\sharp &= \mathsf{reflect}^A(\lambda k.\,(\mathsf{reify}^A M^\sharp)\diamond^A(\lambda x.\,x))\\
(\mathsf{reify}^A M)^\flat &= \lambda k.\,\lceil\lfloor k\rfloor^A@M^\flat\rceil^A & (\lceil M\rceil^A)^\sharp &= (\mathsf{reify}^A M^\sharp)\diamond^A(\lambda x.\,x)\\
&- & (\lfloor M\rfloor^A)^\sharp &= \mathsf{reflect}^A(\lambda k.\,k\diamond^A M^\sharp)
\end{aligned}$$

We extend these translations for $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$-judgments

$$\begin{aligned}
(\Gamma\vdash M:A)^\sharp &= [\,]\mid\Gamma\vdash M^\sharp:A\\
(B\mid\Gamma\vdash M:A)^\sharp &= \mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma\vdash M^\sharp:A
\end{aligned}$$

and for $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$-judgments

$$\begin{aligned}
([\,]\mid\Gamma\vdash M:A)^\flat &= \Gamma\vdash M^\flat:A\\
(\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma\vdash M:A)^\flat &= B\mid\Gamma\vdash M^\flat:A.
\end{aligned}$$

4.3 Equivalence of the Two Systems in Provability

Now, we present the equivalence of monadic reflection and delimited continuation.

▶ **Theorem 9.** *$\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$ and $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}$ possess exactly the same provability. Specifically,*

1. *$\lambda^{\mathsf{s/r}}_{\mathsf{pure}}\vdash\mathcal{J}$ implies $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}\vdash\mathcal{J}^\sharp$*
2. *$\lambda^{\mathsf{refl}}_{\mathsf{2cont}}\vdash\mathcal{J}$ implies $\lambda^{\mathsf{s/r}}_{\mathsf{pure}}\vdash\mathcal{J}^\flat$*

Proof. (1) We prove the statement by the induction on the derivation of $\mathcal{J}$. The proofs for $\mathsf{var},\to_{\mathsf{i}},\to^{\mathsf{pure}}_{\mathsf{e}},\to_{\mathsf{e}}$ are routine. Assume that $\mathcal{J}$ is derived using shift and of the form $B\mid\Gamma\vdash\mathcal{S}^B k.\,M:A$. In this case, we have $B\mid\Gamma,k:A\to B\vdash M:B$. Applying the translation, we obtain a judgment $\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma,k:A\to B\vdash M^\sharp:B$, which is justified by the induction hypothesis. Now we have the following derivation tree

$$\dfrac{\dfrac{\dfrac{\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma, k : A \to B \vdash M^\sharp : B}{[\,]\mid\Gamma, k : A \to B \vdash \mathsf{reify}^B M^\sharp : (B \to B) \to B} \quad \dfrac{\dfrac{}{[\,]\mid\Gamma, k : A \to B, x : B \vdash x : B}}{[\,]\mid\Gamma, k : A \to B \vdash \lambda x.\, x : B \to B}}{[\,]\mid\Gamma, k : A \to B \vdash (\mathsf{reify}^B M^\sharp)\diamond^B(\lambda x.\, x)}}{\dfrac{[\,]\mid\Gamma \vdash \lambda k.\,(\mathsf{reify}^B M^\sharp)\diamond^B(\lambda x.\, x) : (A \to B) \to B}{\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash \mathsf{reflect}^B(\lambda k.\,(\mathsf{reify}^B M^\sharp)\diamond^B(\lambda x.\, x)) : A}}$$

where x is a variable which does not occur in $M^\sharp, \Gamma$. The conclustion of this deriviation tree is equal to the result of the translation of $B \mid \Gamma \vdash \mathcal{S}^B k.\, M : A$.

Assume that $\mathcal{J}$ is derived using reset and of the form $\Gamma \vdash \lceil M \rceil^A : A$. In this case, we have $A \mid \Gamma \vdash M : A$. Translating this judgment, we obtain a valid judgment $\mathsf{cont}_A\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash M^\sharp : A$. Now, we can construct the following tree:

$$\dfrac{\dfrac{\mathsf{cont}_A\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash M^\sharp : A}{[\,]\mid\Gamma \vdash \mathsf{reify}^A M^\sharp : (A \to A) \to A} \quad \dfrac{\dfrac{}{[\,]\mid\Gamma, x : A \vdash x : A}}{[\,]\mid\Gamma \vdash \lambda x.\, x : A \to A}}{[\,]\mid\Gamma \vdash (\mathsf{reify}^A M^\sharp)\diamond^A(\lambda x.\, x) : A}$$

Hence we have $[\,]\mid\Gamma \vdash (\mathsf{reify}^A M^\sharp)\diamond^A(\lambda x.\, x) : A$, which is equal to the result of the translation of $\Gamma \vdash \lceil M \rceil^A : A$.

Assume that $\mathcal{J}$ is derived using exp and of the form $B \mid \Gamma \vdash \lfloor M \rfloor^B : A$. In this case, we have $\Gamma \vdash M : A$, and therefore $[\,]\mid\Gamma \vdash M^\sharp : A$. Now we have the following derivation:

$$\dfrac{\dfrac{\dfrac{}{[\,]\mid\Gamma, k : A \to B \vdash k : A \to B} \quad \dfrac{[\,]\mid\Gamma \vdash M^\sharp : A}{[\,]\mid\Gamma, k : A \to B \vdash M^\sharp : A}}{[\,]\mid\Gamma, k : A \to B \vdash k \diamond^B M^\sharp : B}}{\dfrac{[\,]\mid\Gamma \vdash \lambda k.\, k \diamond^B M^\sharp : (A \to B) \to B}{\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash \mathsf{reflect}^B(\lambda k.\, k \diamond^B M^\sharp) : A}}$$

which concludes our proof for (1).

(2) Again, we prove the statement on the derivation of $\mathcal{J}$. We present proofs only for non-trivial cases: $\mathsf{reflect}, \mathsf{reify}$, and $\to_{\mathsf{e}}^{\mathsf{pure}}$. Assume that $\mathcal{J}$ is derived from $\mathsf{reflect}$ and of the form $\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash \mathsf{reflect}^B M : A$. Then we have $[\,]\mid\Gamma \vdash M : (A \to B) \to B$, which implies $\Gamma \vdash M^\flat : (A \to B) \to B$ by the induction hypothesis. Now we have the following derivation tree:

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash M^\flat : (A \to B) \to B}{B \mid \Gamma \vdash \lfloor M^\flat \rfloor^B : (A \to B) \to B}}{B \mid \Gamma, k : A \to B \vdash \lfloor M^\flat \rfloor^B : (A \to B) \to B} \quad \dfrac{\dfrac{}{\Gamma, k : A \to B \vdash k : A \to B}}{B \mid \Gamma, k : A \to B \vdash \lfloor k \rfloor^B : A \to B}}{\dfrac{B \mid \Gamma, k : A \to B \vdash \lfloor M^\flat \rfloor^B @ \lfloor k \rfloor^B : B}{B \mid \Gamma \vdash \mathcal{S}^B k.\, \lfloor M^\flat \rfloor^B @ \lfloor k \rfloor^B : A}}$$

The conclusion of this tree is equal to $(\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash \mathsf{reflect}^B M : A)^\flat$.

Assume that $\mathcal{J}$ is derived from reify and of the form $[\,]\mid\Gamma \vdash \mathsf{reify}^B M : (A \to B) \to B$. In this case, we have $\mathsf{cont}_B\,;\ [\,]\mid\emptyset\,;\ \Gamma \vdash M : A$, which implies $B \mid \Gamma \vdash M^\flat : A$. The tree that we need to construct is the following one:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, k : A \to B \vdash k : A \to B}}{B \mid \Gamma, k : A \to B \vdash \lfloor k \rfloor^B : A \to B} \quad \dfrac{B \mid \Gamma \vdash M^\flat : A}{B \mid \Gamma, k : A \to B \vdash M^\flat : A}}{B \mid \Gamma, k : A \to B \vdash \lfloor k \rfloor^B @ M^\flat : B}}{\Gamma, k : A \to B \vdash \lceil \lfloor k \rfloor^B @ M^\flat \rceil^B : B}}{\Gamma \vdash \lambda k.\, \lceil \lfloor k \rfloor^B @ M^\flat \rceil^B : (A \to B) \to B}$$

Finally, assume that $\mathcal{J}$ is derived from $\to_{\mathsf{e}}^{\mathsf{pure}}$ and of the form $[\,] \mid \Gamma \vdash M \diamond^B N : B$. In this case, we have $[\,] \mid \Gamma \vdash M : A \to B$ and $[\,] \mid \Gamma \vdash N : A$ for some A. We can therefore construct the following derivation tree:

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash M^\flat : A \to B}{B \mid \Gamma \vdash \lfloor M^\flat \rfloor^B : A \to B} \quad \dfrac{\Gamma \vdash N^\flat : A}{B \mid \Gamma \vdash \lfloor N^\flat \rfloor^B : A}}{B \mid \Gamma \vdash \lfloor M^\flat \rfloor^B @ \lfloor N^\flat \rfloor^B : B}}{\Gamma \vdash \lceil \lfloor M^\flat \rfloor^B @ \lfloor N^\flat \rfloor^B \rceil^B : B}$$

which concludes our proof for (2). ◀

▶ **Corollary 10.** $\lambda_{\mathsf{pure}}^{\mathsf{s/r}} \vdash A$ *iff* $\lambda_{\mathsf{2cont}}^{\mathsf{refl}} \vdash A$.

5 Related Work

5.1 Monadic Reflection

In his seminal work, Filinski[7] introduced the idea of monadic reflection with its implementation by shift/reset. His work is one of our main motivations of this paper. It would be worth to note that the type system of the calculus with shift/reset in his paper is different from the original one[4], and therefore from the one that we have discussed in this paper. Indeed, for example, the original calculus is known to be strongly normalizing, whereas his calculus is not[14].

Forster et al.[9] compares the expressibility of effect handlers, monadic reflection, and delimited continuations. In the paper, among others, they show that delimited continuations and monadic reflection can express each other in untyped setting. At the same time, they show that their translation from delimited continuations to monadic reflection preserves types, whereas the translation of the opposite direction does not. Comparing to their work, our reconstruction can be understood as a proposal of an answer to the question of how we can preserve types in both directions of these translations.

Zeilberger[24] discusses delimited continuations in his somewhat non-standard calculus which incorporates polarity. He claims in the paper that monadic reflection is essentially the isomorphism in the Yoneda lemma. His observation might play an important role when we construct a categorical semantics of our system.

5.2 Logical Meaning of Delmited Continuations

Ariola et al.[1] explains the logical meaning of delimited continuations by translating a system with a dynamic variable, which can interpret shift/reset with answer-type modification, into a logical system with *subtraction*. In terms of classical logic, subtraction $A - B$ is dual to implication $A \to B$. Namely, $A - B = A \wedge (\neg B)$. A virtue of their system would be the fact that it explains the meaning of answer-type modification. At the same time, however, it should be noted that the subtractive system allows classical inference, whereas the original system with shift/reset does not allow double-negation.

Kameyama [12] covers a variant of delimited continuation operators which differs a little from shift/reset. In his work, he consider a side-condition that the operation that corresponds to shift can be used only when the operation that corresponds to reset will appear later. Through the lens of our reconstruction, this condition seems to be closely related to the role of the context stack in our system.

6 Conclusion

We have reconstructed monadic reflection using the idea of contextual modal logic, and exploited it to investigate delimited continuations, obtaining the equivalence of these two concepts in provability.

In this paper, we have focused on delimited continuations with pure abstractions. It might be possible to extend our work and encompass impure abstractions. Indeed, we can extend the pure-contexts in λ^{refl} as follows:

$$F ::= [\,] \mid V@F \mid F@M \mid \lambda x.\, F$$

and drop the side-condition of $\to_{\mathsf{i}}$ that we imposed in Section 3. With some modification to the definition of the values, we have the following reduction, for example:

$$E[\mathsf{reify}^T(\lambda x.\, \mathsf{reflect}^T M)] \rightsquigarrow E[M \rhd_y^T \mathsf{reify}^T(\lambda x.\, y)].$$

Note that our layered context ensures that M does not have x as free variable. By exploiting this fact and the macro-definition of shift/reset in λ^{refl}, we might be able to realize impure functions.

Another direction of future work is categorical semantics of λ^{refl}. Ordinary contextual modal logic already has a categorical semantics based on iterative enrichment[20]. It might be possible to explain λ^{refl} based on their calculus.

Decomposing the effect T into contextual modalities is also an interesting topic. In [22], the authors decompose the modality that corresponds to T in our system into $\Diamond\Box$. Using the contextual possibility, it might be possible to decompose our T and derive the rule $\mathsf{reify}, \mathsf{reflect}$.

We have restriced the depth of the context stack of $\lambda^{\mathsf{refl}}_{\mathsf{2cont}}$ to be 2. It might also be possible to generalize the equivalence between delimited continuation and monadic reflection to the "iterative" one by dropping this restriction, clarifying the character of shiftn/resetn in the CPS hierarchy[13].

References

1. Zena Ariola, Hugo Herbelin, and Amr Sabry. A Type-Theoretic Foundation of Delimited Continuations. *Higher-Order and Symbolic Computation*, 2007.
2. Kenichi Asai. On typing delimited continuations: three new solutions to the printf problem. *Higher-Order and Symbolic Computation*, 22(3):275–291, Sep 2009. doi:10.1007/s10990-009-9049-5.
3. Kenichi Asai and Yukiyoshi Kameyama. Polymorphic delimited continuations. In *Programming Languages and Systems*, pages 239–254. Springer Berlin Heidelberg, 2007. doi:10.1007/978-3-540-76637-7_16.
4. Olivier Danvy and Andrzej Filinski. A functional abstraction of typed contexts. Technical report, Institute of Datalogy,University of Copenhagen, 1989.
5. Matthias Felleisen, Daniel Friedman, Eugene Kohlbecker, and Bruce Duba. A syntactic theory of sequential control. *Theoretical Computer Science*, 52(3):205–237, 1987. doi:10.1016/0304-3975(87)90109-5.

6 Mattias Felleisen. The theory and practice of first-class prompts. In *Proceedings of the 15th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '88, pages 180–190, New York, NY, USA, 1988. ACM. doi:10.1145/73560.73576.

7 Andrzej Filinski. Representing monads. In *Proceedings of the 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '94, pages 446–457. ACM, 1994. doi:10.1145/174675.178047.

8 Andrzej Filinski. Monads in action. *SIGPLAN Not.*, 45(1):483–494, 2010. doi:10.1145/1707801.1706354.

9 Yannick Forster, Ohad Kammar, Sam Lindley, and Matija Pretnar. On the expressive power of user-defined effects: Effect handlers, monadic reflection, delimited control. *Proceedings of the ACM on Programming Languages*, 1(ICFP):13:1–13:29, 2017. doi:10.1145/3110257.

10 Timothy Griffin. A formulae-as-type notion of control. In *Proceedings of the 17th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '90, pages 47–58, New York, NY, USA, 1990. ACM. doi:10.1145/96709.96714.

11 Gregory Johnson. GL – a denotational testbed with continuations and partial continuations as first-class objects. *SIGPLAN Not.*, 22(7):165–176, 1987. doi:10.1145/960114.29668.

12 Yukiyoshi Kameyama. Towards logical understanding of delimited continuations. In *Continuations Workshop*, pages 27–33, 2001.

13 Yukiyoshi Kameyama. Axioms for delimited continuations in the CPS hierarchy. In *Computer Science Logic, 18th International Workshop, CSL 2004, 13th Annual Conference of the EACSL, Karpacz, Poland, September 20-24, 2004, Proceedings*, pages 442–457, 2004. doi:10.1007/978-3-540-30124-0_34.

14 Yukiyoshi Kameyama and Kenichi Asai. Strong normalization of polymorphic calculus for delimited continuations. In *Symbolic Computation in Software Science*, 2008.

15 Oleg Kiselyov and Chung-chieh Shan. A substructural type system for delimited continuations. In *Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007, Paris, France, June 26-28, 2007, Proceedings*, pages 223–239, 2007. doi:10.1007/978-3-540-73228-0_17.

16 Saul Kripke. A completeness theorem in modal logic. *The Journal of Symbolic Logic*, 24(1):1–14, 1959. URL: http://www.jstor.org/stable/2964568.

17 Saul Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16(1963):83–94, 1963.

18 Yuito Murase. Kripke-style contextual modal type theory. Work-in-progress report at Logical Frameworks and Meta-Languages, 2017.

19 Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. Contextual modal type theory. *ACM Transactions on Computational Logic*, 9(3):23:1–23:49, 2008. doi:10.1145/1352582.1352591.

20 Yuichi Nishiwaki, Yoshihiko Kakutani, and Yuito Murase. Modality via iterated enrichment, 2018. arXiv:1804.02809. arXiv:arXiv:1804.02809.

21 Michel Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. *Lecture Notes in Computer Science*, 624:190–201, 1992. doi:10.1007/BFb0013061.

22 Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11(4):511–540, aug 2001. doi:10.1017/S0960129501003322.

23 Willard Quine. Reference and modality. In *Journal of Symbolic Logic*, pages 139–159. Harvard University Press, 1953.

24 Noam Zeilberger. Polarity and the logic of delimited continuations. In *Proceedings of the 2010 25th Annual IEEE Symposium on Logic in Computer Science*, LICS '10, pages 219–227, Washington, DC, USA, 2010. IEEE Computer Society. doi:10.1109/LICS.2010.23.