On the call-by-value CPS transform and its semantics

Carsten Führmann [a,*,1], Hayo Thielecke [b]

[a] *University of Bath, Department of Computer Science, Claverton Down, Bath BA2 7AY, UK*

[b] *School of Computer Science, The University of Birmingham, Birmingham B15 2TT, UK*

Abstract

We investigate continuations in the context of idealized call-by-value programming languages. On the semantic side, we analyze the categorical structures that arise from continuation models of call-by-value languages. On the syntactic side, we study the call-by-value continuation-passing transformation as a translation between equational theories. Among the novelties are an unusually simple axiomatization of control operators and a strengthened completeness result with a proof based on a delaying transform.

Key words: continuations, lambda-calculus, semantics, categories, monads
1991 MSC: 68Q55, 03B70, 18C20, 18C50

1 Introduction

Continuations are one of the fundamental concepts in the semantics of programming languages. Intuitively, a continuation represents the meaning of the "rest of the computation" [1,2]. The way in which the continuation is passed around makes explicit all control transfers, such as function calls and returns. In particular, manipulating the continuation in a non-standard fashion accounts for control transfers, that is, jumps. Originally, the obvious control construct to model was `goto`; more powerful constructs, such as `call/cc` in

* Corresponding author.
Email addresses: `C.Fuhrmann@bath.ac.uk` (Carsten Führmann), `H.Thielecke@cs.bham.ac.uk` (Hayo Thielecke).
[1] With partial support of the UK EPSRC via research grant GR/R29680/01.

Scheme, have been introduced to give the programmer explicit access to continuations.

We restrict our attention in this paper to call-by-value, because the programming languages with continuations that motivate our work are call-by-value. The canonical framework for call-by-value languages with effects is Moggi's computational λ-calculus (λ_C-calculus).

Continuations can be added to the λ_C-calculus in that we can provide an operation to give the programmer explicit access to them. The canonical construct for doing this is the `call/cc` construct in Scheme [3] and the New Jersey dialect of Standard ML [4]. We will use a variant of `call/cc`, Felleisen's $\mathcal{C}$-operator, which is technically more convenient. Specifically, we prove that such a control operator can be added to the computational axioms in a surprisingly simple way, by requiring a certain map to have an inverse.

The meaning of such control operators is given by their manipulation of continuations, although what counts as a continuation depends on the formalism: continuations can be semantic or syntactic entities, or even evaluation contexts in an operational semantics. Making all control transfers explicit by use of continuations is called "continuation-passing style" [5], or "CPS" for short. Continuation-passing has both a semantic and a syntactic side.

On the syntactic side, one can compile λ-terms by systematically introducing continuations everywhere, a translation known as "CPS transform". The target language of this transform is usually considered to be a subset of the λ-calculus. However, it is a very stylized subset, in that it admits a very "imperative" reading in terms of jumping [5]. We aim to do justice to the target language by introducing a new calculus, which we call CPS calculus, designed to bring out the jumping, imperative nature of continuation-passing. The more traditional presentations of CPS are then recovered in that the CPS calculus admits a translation into λ-calculus.

On the semantic side, we study two fundamentally different classes of models. For the first class of models, given by "response categories" (mild generalizations of cartesian-closed categories), the interpretation of the λ_C-calculus can be factored into a CPS transform followed by a trivial interpretation of the target language. We shall present a novel completeness proof for this class of models by a "delaying" transform from the target language back into the λ_C-calculus.

By contrast, the second class of models, which we shall introduce as "$\mathcal{C}$-categories", is direct in that it does not mirror the detour via the CPS transform. First we shall introduce direct models for the λ_C-calculus in general. Then we shall obtain $\mathcal{C}$-categories by adding a categorical version of the abovementioned requirement that a certain map have an inverse. We shall show that, in a precise sense, $\mathcal{C}$-categories are to response categories what the λ_C-calculus with control operators is to the CPS language, in that the two kinds of models are connected by a semantic version of the CPS transform.

Finally, we shall prove that every $\mathcal{C}$-category arises from a response category. In fact, we shall derive this from an even stronger theorem about the relationship between direct models and monadic model in the sense of Moggi.

1.1 The historical background.

Continuations as a concept in programming languages emerged during the 1960s in several places and in different guises [6]; the term continuation was first used by Strachey and Wadsworth [1]. First-class control operators also made their first appearance in the 1960s; the first one was Landin's **J**-operator [7,8], a direct ancestor of the operators which we use in this paper. CPS transformations, a syntactic technique for introducing continuations, were first published by Fischer [9] and Plotkin [10], though the term *continuation-passing transformation* itself was only coined later by Steele [5]. Felleisen et. al. were the first to axiomatize control operators [11]. Felleisen and Sabry gave a completeness proof for the CPS transforms [12]. Their techniques were syntactic, whereas Hofmann [13], at about the same time, used categorical techniques. The typing of control operators (corresponding to classical logic, although we do not explore the logic side here) was discovered by Griffin [14], and incorporated by Duba, Harper and MacQueen in the New Jersey dialect of Standard ML [4].

On the categorical side, Filinski [15] pioneered the treatment of continuation operations as categorical structure in their own right. Moggi's programme of monads as notions of computation [16] is a general account of various effects, which specializes to the CPS transform for the case of the continuations monad $R^{R^{(-)}}$. More specifically, looking at the Kleisli category of this monad amounts to direct style, while making the monad explicit amounts to continuation-passing style. Power and Robinson's alternative account of notions of computation uses premonoidal categories [17] rather than monads. This approach, together with the self-adjointness of the continuation functor, was used in Thielecke's thesis [18]. Selinger's more recent control-categories and co-control categories [19] are also based on premonoidal structure. They provide a semantics of control operators where call-by-name and call-by-value languages are dual to each other.

In this article, we present a state-of-the-art account of the axiomatics and categorical semantics of control operators in a *simply-typed call-by-value* setting, plus several new results. The immediate background to this work is given by both authors' Edinburgh PhD theses [20,18], which aimed at direct accounts of λ_C-calculus and continuations in particular.

1.2 Contributions.

The contributions of this paper include the following.

- A very simple axiomatization of control operators, both in the syntactic and the categorical setting. Given a general account of call-by-value, all we need is to require a certain map to have an inverse.
- An account of the target language of the CPS transform as a calculus in its own right. The intention is to make the connection with intermediate languages in CPS compilers [5,21] more explicit. Furthermore, since the CPS calculus is very close to the π-calculus, the role of CPS in translating call-by-value λ-calculus into the π-calculus is also clarified.
- A delaying transform, which is the basis of a simplified completeness proof.
- A very general result stating that a certain category of direct models is a full reflective subcategory of a certain category of monads. From this result we derive that every $\mathcal{C}$-category arises from a response category.

Our completeness results are universally quantified over theories, as usual in logic. Thus they differ from Sabry and Felleisen's result on the completeness of the CPS transform [12], which is essentially only for the empty theory. (However, the result in [12] is not superseded by ours, because it is stated in an untyped setting, whereas all of our results rely on types.)

Also, our completeness results are unusually strong in the sense that the term model satisfies the equalizer requirement. As we shall explain, this establishes a link with sober spaces in the sense of [22].

1.3 Construction of this article.

In Section 2, we recall some facts about the λ_C-calculus, monads, and premonoidal categories. Section 3 introduces our new axiomatization of control operators. Section 4 presents the CPS calculus and the CPS transform into it. Section 5 relates that transform with the well-known CPS transform into the λ-calculus. In Section 6, we introduce the delaying transform and use it to build a sober CPS term model (which implies completeness of the CPS transform). Section 7 introduces *abstract Kleisli-categories*, and explains their relationship with monads. Section 8 uses those categories as *direct models* of the λ_C-calculus, and presents results about soundness, completeness, initiality, and internal language. Section 9 specializes those models to accommodate control operators, and contains the main structural theorem stating that every such model arises from a continuations monad. Section 10 concludes.

2 Preliminaries

2.1 The λ_C-calculus

The λ_C-calculus [23] has proved itself useful for reasoning about call-by-value programs. Its syntax, typing, and axioms on the well-typed terms are summarized in Figure 1. The letter b ranges over base types, x, y range over variables, and c^A ranges over (typed) constants. We shall often abbreviate $\lambda x : A.M$ by $\lambda x.M$. The expression $\mathtt{let}\, x\, \mathtt{be}\, M\, \mathtt{in}\, N$ is syntactic sugar for $(\lambda x.N)M$. As usual, we consider terms modulo α-equivalence—that is, the collision-free renaming of variables bound by λ. The expression $M[N/x]$ stands for the term that results from substituting N for all free occurrences of the variable x in M, avoiding variable capture by renaming bound variables. The letter Γ stands for *environments*, which are non-repetitive sequences $x_1 : A_1, \ldots, x_n : A_n$ of typed variables. The λ_C*-language* over a set of base types and a set of constants is the set of judgments $\Gamma \vdash M : A$ which is generated by the typed term formation rules in Figure 1. A λ*-theory* $\mathcal{T}$ over a λ_C-language $\mathcal{L}$ is a set of equations $\Gamma \vdash M \equiv N : A$, where $\Gamma \vdash M : A$ and $\Gamma \vdash N : A$ are in $\mathcal{L}$, that contains all equations described in Figure 1, and is a congruence stable under weakening and permutation. We write $|\mathcal{T}|$ for the underlying language of $\mathcal{T}$.

In this article, we consider expressions M only together with their environment Γ—that is, we consider judgments of the form $\Gamma \vdash M : A$, and never bare expressions M. From here on, "expression" means a judgment of the form $\Gamma \vdash M : A$. However, we may just write M if Γ is obvious or if the discussion applies to all admissible Γ.

We define the *factorization* of a type A to be the sequence $A_1, \ldots, A_n$ of types such that none of the A_i is a product or unit type, and A is the product of the A_i up to bracketing and occurrences of 1. For example, if A is

$$((B_0 \to B_1) \times (1 \to B_2)) \times (((B_1 \to 1) \times (B_2 \to B_1)) \times 1))$$

then the factorization of A is the sequence A_1, A_2, A_3, A_4 where $A_1 = B_0 \to B_1$, $A_2 = 1 \to B_2$, $A_3 = B_1 \to 1$, and $A_4 = B_2 \to B_1$. If $A_1, \ldots, A_n$ is the factorization of A and $x_1 : A_1, \ldots x_n : A_n$ are variables, then x_A stands for the evident "n-tuple" of type A built from the x_i. (In our example, we have $x_A = ((x_1, x_2), ((x_3, x_4), ())).$)

Now let $\Gamma \vdash M : A$ and $\Gamma, y_1 : A_1, \ldots, y_n : A_n \vdash N : B$ be expressions such

$$\begin{array}{rl}
\text{Types} & A, B ::= b \mid A \to B \mid A \times B \mid 1 \\
\text{Expressions} & M, N ::= x \mid c^A \mid \lambda x : A.M \mid M\,N \mid (M, N) \mid \pi_i M \mid () \\
\text{Values} & V, U ::= x \mid c^A \mid \lambda x : A.M \mid (V, U) \mid \pi_i V \mid ()
\end{array}$$

$$\frac{}{\Gamma \vdash x : A}\; x : A \in \Gamma \qquad\qquad \Gamma \vdash c^A : A$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash (M, N) : A \times B} \qquad \frac{\Gamma \vdash M : A_1 \times A_2}{\Gamma \vdash \pi_i(M) : A_i} \qquad \Gamma \vdash () : 1$$

$$\begin{array}{rll}
\mathtt{let}\, x\, \mathtt{be}\, V\, \mathtt{in}\, M \equiv & M[V/x] & \\
\lambda x.Vx \equiv & V & (x \notin \mathrm{FV}(V)) \\
\pi_i(V_1, V_2) \equiv & V_i & \\
(\pi_1(V), \pi_2(V)) \equiv & V & \\
V \equiv & () & \\
\mathtt{let}\, x\, \mathtt{be}\, M\, \mathtt{in}\, x \equiv & M & \\
\mathtt{let}\, y\, \mathtt{be}\, (\mathtt{let}\, x\, \mathtt{be}\, L\, \mathtt{in}\, M)\, \mathtt{in}\, N \equiv & \mathtt{let}\, x\, \mathtt{be}\, L\, \mathtt{in}\, \mathtt{let}\, y\, \mathtt{be}\, M\, \mathtt{in}\, N & (x \notin \mathrm{FV}(N)) \\
MN \equiv & \mathtt{let}\, f\, \mathtt{be}\, M\, \mathtt{in}\, \mathtt{let}\, x\, \mathtt{be}\, N\, \mathtt{in}\, f x & \\
(M, N) \equiv & \mathtt{let}\, x\, \mathtt{be}\, M\, \mathtt{in}\, \mathtt{let}\, y\, \mathtt{be}\, N\, \mathtt{in}\, (x, y) & \\
\pi_i(M) \equiv & \mathtt{let}\, x\, \mathtt{be}\, M\, \mathtt{in}\, \pi_i(x) &
\end{array}$$

Fig. 1. The λ_C-calculus

that $A_1, \ldots, A_n$ is the factorization of A. We define

$$\begin{aligned}(\mathtt{let}\, y_1, \ldots, y_n\, \mathtt{be}\, M\, \mathtt{in}\, N) =& (\mathtt{let}\, z\, \mathtt{be}\, M\, \mathtt{in}\, \mathtt{let}\, y_1\, \mathtt{be}\, p_1(z)\, \mathtt{in}\, \ldots \\ & \mathtt{let}\, y_n\, \mathtt{be}\, p_n(z)\, \mathtt{in}\, N)\end{aligned}$$

where z is a fresh variable of type A, and $p_i(z)$ is the obvious repeated application of π_1 and π_2 to z. Also, we define

$$\lambda(y_1,\ldots,y_n):A.N = \lambda x:A.\mathtt{let}\,y_1,\ldots,y_n\,\mathtt{be}\,x\,\mathtt{in}\,N$$

where x is fresh.

2.2 Algebraic values

The notion of algebraic value is taken from [24]. An algebraic value is an expression M that can be substituted for the occurrences of a formal parameter x in any procedure body N whenever M is passed as the actual parameter. Formally, an expression $\Gamma \vdash M : A$ is defined to be an algebraic value of a λ_C-theory $\mathcal{T}$ if every well-formed equation

$$\Gamma' \vdash \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,N \equiv N[M/x] : B$$

is a theorem of $\mathcal{T}$ whenever Γ' contains Γ.

Every algebraic value M of function type is equivalent to a value, because $M \equiv \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,x \equiv \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,\lambda y.xy \equiv \lambda y.My$. At other types this can be false. For example, in λ_C-theories where $- : \mathtt{int} \longrightarrow \mathtt{int}$ is interpreted as integer negation, the algebraic value $x : \mathtt{int} \vdash -x : \mathtt{int}$ is not equivalent to a value. (This follows from a simple inductive argument, using only that the expression is not equivalent to a constant or a variable, and that its type is a base type.)

Lemma 1 *If $\Gamma \vdash M : A$ and $\Gamma, x : A \vdash L : B$ are algebraic values, then so is $\Gamma \vdash \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,L : B$.*

To see this, consider

$$\begin{aligned}\mathtt{let}\,y\,\mathtt{be}\,(\mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,L)\,\mathtt{in}\,N &\equiv \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,\mathtt{let}\,y\,\mathtt{be}\,L\,\mathtt{in}\,N\\ &\equiv \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,N[L/y]\\ &\equiv N[L[M/x]/y]\\ &\equiv N[\mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,L/y].\end{aligned}$$

Remark 2 *The notion of algebraic value is not well defined for expressions without environment. For example, for every $\Gamma \vdash M : A$, the weakened version $\Gamma, x : 0 \vdash M : A$ is an algebraic value if 0 denotes the initial object.*

Lemma 3 *If in a λ_C-theory it holds that*

$$\Gamma \vdash \mathtt{let}\,x\,\mathtt{be}\,M\,\mathtt{in}\,\lambda().x \equiv \lambda().M : 1 \to A \qquad (1)$$

then $\Gamma \vdash M : A$ is an algebraic value.

To see this, assume Equation 1, and consider

$$\begin{aligned}
\texttt{let}\,x\,\texttt{be}\,M\,\texttt{in}\,N &\equiv \texttt{let}\,y\,\texttt{be}\,(\texttt{let}\,x\,\texttt{be}\,M\,\texttt{in}\,\lambda().x)\,\texttt{in}\,N[y()/x] \\
&\equiv \texttt{let}\,y\,\texttt{be}\,\lambda().M\,\texttt{in}\,N[y()/x] \\
&\equiv N[M/x].
\end{aligned}$$

2.3 Monads

As usual in category theory, we present a monad on a category $\mathbf{C}$ as a triple $T = (T, \mu, \eta)$, where T is the underlying functor $\mathbf{C} \longrightarrow \mathbf{C}$, $\mu : T \circ T \longrightarrow T$ is the natural transformation called *multiplication*, and $\eta : Id \longrightarrow T$ is the natural transformation called *unit*. (For an introduction to monads, see [25].) Given a category $\mathbf{C}$ with finite products, a strong monad on $\mathbf{C}$ is a monad T together with a natural transformation $t : A \times TB \longrightarrow T(A \times B)$ (its "strength") that satisfies the four equations below, where $\alpha : (A \times B) \times C \longrightarrow A \times (B \times C)$ and $\rho : 1 \times A \longrightarrow A$ are the evident maps associated with the finite products:

$$\begin{aligned}
T\rho \circ t &= \rho \\
T\alpha \circ t &= t \circ (id \times t) \circ \alpha \\
t \circ (id \times \eta) &= \eta \\
t \circ (id \times \mu) &= \mu \circ Tt \circ t.
\end{aligned}$$

If T is a strong monad on a category $\mathbf{C}$, then $\mathbf{C}$ is said to have T*-exponentials* if for all objects A and B, the exponential $(TB)^A$ exists.

The next definition is taken from [26]. A *monad morphism* from a monad T on $\mathbf{C}$ to a monad T' on $\mathbf{C}'$ is a functor $U : \mathbf{C} \longrightarrow \mathbf{C}'$ together with a natural transformation $\sigma : UT \longrightarrow T'U$ such that "U preserves μ and η up to σ"—that is, the two diagrams below commute.

$$\begin{tikzcd}
UTT \arrow[r, "\sigma T"] \arrow[d, "U\mu"'] & T'UT \arrow[r, "T'\sigma"] & T'T'U \arrow[d, "\mu' U"] \\
UT \arrow[rr, "\sigma"] & & T'U
\end{tikzcd}
\qquad
\begin{tikzcd}
U\,Id \arrow[r, phantom, "="] \arrow[d, "U\eta"'] & Id\,U \arrow[d, "\eta' U"] \\
UT \arrow[r, "\sigma"] & T'U
\end{tikzcd}
\qquad (2)$$

We call (U, σ) *tight* if σ is an isomorphism. A *strong monad morphism* from a strong monad T to a strong monad T' is a monad morphism (U, σ) from T to T' such that U preserves finite products and, letting $U_2 : U(A \times B) \longrightarrow UA \times UB$ be the evident natural isomorphism, the diagram below commutes for all objects A and B.

$$\begin{tikzcd}
U(A \times TB) \arrow[r, "Ut"] \arrow[d, "U_2"'] & UT(A \times B) \arrow[r, "\sigma"] & T'U(A \times B) \arrow[d, "T'U_2"] \\
UA \times UTB \arrow[r, "UA \times \sigma"] & UA \times T'UB \arrow[r, "t'"] & T'(UA \times UB)
\end{tikzcd} \tag{3}$$

Now suppose that T and T' have T-exponentials, and that $(U, \sigma) : T \longrightarrow T'$ is a strong monad morphism. Then, for objects A and B, we call (U, σ) *closed* if the map $U((TB)^A) \longrightarrow (T'UB)^{UA}$ that arises as the adjoint mate of $U((TB)^A) \times UA \cong U((TB)^A \times A) \xrightarrow{Uev} UTB \xrightarrow{\sigma} T'UB$ is an isomorphism for all A and B.

2.4 Premonoidal categories

The following definitions are taken from [17]. A *binoidal category* is a category $\mathbf{K}$ together with

- For each object A, a functor $A \otimes (-) : \mathbf{K} \longrightarrow \mathbf{K}$, and
- For each object B, a functor $(-) \otimes B : \mathbf{K} \longrightarrow \mathbf{K}$

such that for all objects A and B it holds that $(A \otimes (-))(B) = ((-) \otimes B)(A)$. For the joint value, we write $A \otimes B$. We say that morphisms $f : A \longrightarrow A'$ and $g : B \longrightarrow B'$ of a binoidal category $\mathbf{K}$ *commute* if the two diagram below commute.

$$\begin{tikzcd}
A \otimes B \arrow[r, "f \otimes B"] \arrow[d, "A \otimes g"'] & A' \otimes B \arrow[d, "A' \otimes g"] \\
A \otimes B' \arrow[r, "f \otimes B'"] & A' \otimes B'
\end{tikzcd}
\qquad
\begin{tikzcd}
B \otimes A \arrow[r, "B \otimes f"] \arrow[d, "g \otimes A"'] & B \otimes A' \arrow[d, "g \otimes A'"] \\
B' \otimes A \arrow[r, "B' \otimes f"] & B' \otimes A'
\end{tikzcd}$$

A morphism $f : A \longrightarrow A'$ is called *central* if it commutes with every morphism. The *center* $Cen(\mathbf{K})$ of $\mathbf{K}$ is defined as the subcategory of $\mathbf{K}$ given by all objects and the central morphisms. A *symmetric premonoidal category* is a binoidal category together with an object I and natural isomorphisms $A \otimes (B \otimes C) \cong (A \otimes B) \otimes C$, $A \otimes B \cong B \otimes A$, $A \otimes I \cong A$, and $I \otimes A \cong A$ with central components that satisfy the coherence conditions known from symmetric monoidal categories (see e.g. [25]). Given symmetric premonoidal categories $\mathbf{K}$ and $\mathbf{K}'$, a functor $U : \mathbf{K} \longrightarrow \mathbf{K}'$ is called *strict symmetric premonoidal* if it preserves all symmetric premonoidal structure on the nose and sends central morphisms to central morphisms. A *Freyd category* [27] consists of a category $\mathbf{C}$ with explicitly-given finite products, a symmetric premonoidal category $\mathbf{K}$, and a strict symmetric premonoidal functor $F : \mathbf{C} \longrightarrow \mathbf{K}$. A *closed Freyd-category* is a Freyd category $F : \mathbf{C} \longrightarrow \mathbf{K}$ together with a right adjoint $A \rightharpoonup (-)$ to the functor $F(-) \otimes A : \mathbf{C} \longrightarrow \mathbf{K}$ for every object A.

3 Continuation operators as an inverse

For every λ_C-theory, define

$$\mathtt{ftoc}^{O}_{A,B} := \lambda f.\lambda(x,k).k(fx) : (A \to B) \to (A \times (B \to O)) \to O$$

for types A, B, and O. In this section, we shall prove that well-known control operators to manipulate continuations correspond to an inverse of $\mathtt{ftoc}^{O}$.

The name `ftoc` stands for “function to continuation”. The rationale behind this terminology is that `ftoc` sends a call-by-value function $A \to B$ to a continuation that accepts a pair $(x : A, k : B \to O)$ where x is the input and k is the continuation to which the output is passed.

As a reference, we use Hofmann’s axiomatization from [13]. It fixes a type O and provides, for every type B, constants $\mathcal{C}_B : ((B \to O) \to O) \to B$ and $\mathcal{A}_B : O \to B$. Hofmann’s axioms are

$$\begin{aligned} V(\mathcal{A}_A M) &\equiv \mathcal{A}_B M && (\mathcal{A}\text{-}\textsc{Abs}) \\ \mathcal{A}_O M &\equiv M && (\mathcal{A}_O\text{-}\textsc{Id}) \\ V(\mathcal{C}_A M) &\equiv \mathcal{C}_B(\lambda k.M(k \circ V)) && (\mathcal{C}\text{-}\textsc{Nat}) \\ \mathcal{C}_A(\lambda k.kM) &\equiv M && (\mathcal{C}\text{-}\textsc{App}) \end{aligned}$$

where $k \circ V$ stands for $\lambda x.k(Vx)$. (Recall from Section 2.2 that values and algebraic values coincide at function type, so there is no ambiguity in the use of V above.) Hofmann proved soundness and completeness of these axioms with respect to CPS semantics. (Later in this article, we shall prove similar results—in fact, a slightly stronger completeness result—with a different technique.)

We shall proceed in two steps. First, we shall note that $\mathcal{A}$ is derivable from $\mathcal{C}$, and then prove that two simple axioms for $\mathcal{C}$ are enough to obtain Hofmann’s four axioms:

Theorem 4 *A type-indexed family of closed algebraic values $\mathcal{C}_B : ((B \to O) \to O) \to B$ satisfies $\mathcal{C}$-*App *and*

$$\lambda k.k(\mathcal{C}_B V) \equiv V \qquad (\mathcal{C}\text{-}\textsc{Delay})$$

*if and only if for every type B there is a closed algebraic value $\mathcal{A}_B : O \to B$ such that the rules $\mathcal{A}$-*Abs*, $\mathcal{A}_O$-*Id*, $\mathcal{C}$-*Nat*, and $\mathcal{C}$-*App *hold.*

This is remarkable from a categorical point of view, because it states that the existence of a certain kind of natural transformation $((B \to O) \to O) \longrightarrow B$ implies that O is initial. (We shall make this precise in Lemma 81.)

Second, we prove that an operator $\mathcal{C}$ as in the theorem above is equivalent to having an inverse to `ftoc`:

Theorem 5 *For every λ_C-theory $\mathcal{T}$ and all types B and O, the following are equivalent:*

(1) $\mathtt{ftoc}^O_{A,B}$ has an inverse for all A.
(2) $\mathtt{ftoc}^O_{1,B}$ has an inverse.
*(3) There is a closed algebraic value $\mathcal{C}_B : ((B \to O) \to O) \to B$ that satisfies $\mathcal{C}$-*App *and the rule $\lambda k.k(\mathcal{C}_B V) \equiv V$ ($\mathcal{C}$-*Delay*).*

Also, up to $\equiv$, there is at most one $\mathcal{C}_B$ as in Condition 3.

For proving Theorem 4 we shall use the following lemma, which essentially states that O denotes an initial object.

Lemma 6 *If a λ_C-theory has a type-indexed family of closed algebraic values $\mathcal{A}_B : O \to B$ satisfying ($\mathcal{A}$-*Abs*) and ($\mathcal{A}_O$-*Id*), then for every algebraic value $\Gamma \vdash V : O \to B$ it holds that*

$$\Gamma \vdash V \equiv \mathcal{A}_B : O \to B \tag{4}$$

and $x : O \vdash \mathcal{A}_B x : B$ is an algebraic value.

PROOF. For the first claim, consider $V \equiv \lambda x.Vx \equiv \lambda x.V(\mathcal{A}_O x) \equiv \lambda x.\mathcal{A}_B x \equiv \mathcal{A}_B$. The second claim means that all equations of the form below hold:

$$x : O, y_1 : A_1, \ldots, y_n : A_n \vdash \mathtt{let}\, z\, \mathtt{be}\, \mathcal{A}_B x\, \mathtt{in}\, N \equiv N[\mathcal{A}_B x/z] : C.$$

This follows from the first claim, because $\lambda x : O.\mathtt{let}\, z\, \mathtt{be}\, \mathcal{A}_B x\, \mathtt{in}\, N \equiv \mathcal{A}_C \equiv \lambda x : O.N[\mathcal{A}_B x/z]$.

Proof of Theorem 4 Suppose that $\mathcal{C}$ satisfies $\mathcal{C}$-App and $\mathcal{C}$-Delay. Then $\mathcal{C}$-Nat holds because

$$\begin{aligned} V(\mathcal{C}_B M) &\equiv \mathcal{C}_B(\lambda k.k(V(\mathcal{C}_B M))) && (\mathcal{C}\text{-App}) \\ &\equiv \mathcal{C}_B(\lambda k.\mathtt{let}\, m\, \mathtt{be}\, M\, \mathtt{in}\, ((\lambda k'.k'(\mathcal{C}_B m))(k \circ V))) \\ &\equiv \mathcal{C}_B(\lambda k.\mathtt{let}\, m\, \mathtt{be}\, M\, \mathtt{in}\, (m(k \circ V)) \\ &\qquad (\mathcal{C}\text{-Delay, applied to } \lambda k'.k'(\mathcal{C}_B m)) \\ &\equiv \mathcal{C}_B(\lambda k.M(k \circ V)). \end{aligned}$$

Now let $\mathcal{A}_B = \lambda x : O.\mathcal{C}(\lambda k : B \to O.x)$. We have

$$\begin{aligned}
V(\mathcal{A}M) &\equiv V(\texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \mathcal{C}(\lambda k.x)) \\
&\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, V(\mathcal{C}(\lambda k.x)) \\
&\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \mathcal{C}(\lambda k.(\lambda k.x)(k \circ V)) \\
&\qquad \text{(by } \mathcal{C}\text{-}\textsc{Nat}\text{, which has just been proved)} \\
&\equiv \mathcal{A}M.
\end{aligned}$$

Now for the proof of $\mathcal{A}_O$-$\textsc{Id}$. We have

$$\begin{aligned}
\mathcal{A}_O M &\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \mathcal{A}_O x \\
&\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, (\lambda k.k(\mathcal{C}(\lambda k.x)))(\lambda y : O.y) \\
&\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, x \qquad \text{by } \mathcal{C}\text{-}\textsc{Delay} \text{ with } V = \lambda k.x \\
&\equiv M.
\end{aligned}$$

Conversely, let $\mathcal{A}_B$ be a closed algebraic value for every type B such that the four equations in the theorem hold. We have $\mathcal{C}$-$\textsc{Delay}$ because

$$\begin{aligned}
\lambda k.k(\mathcal{C}V) &\equiv \lambda k.\mathcal{C}(\lambda k' : O \to O.V(k' \circ k)) && (\mathcal{C}\text{-}\textsc{Nat}) \\
&\equiv \lambda k.\mathcal{C}(\lambda k' : O \to O.V((\lambda y : O.y) \circ k)) && \text{(by Equation 4)} \\
&\equiv \lambda k.\mathcal{C}(\lambda k' : O \to O.(\lambda y : O.y)(Vk)) \\
&\equiv \lambda k.\mathcal{C}(\lambda k' : O \to O.k'(Vk)) && \text{(by Equation 4)} \\
&\equiv \lambda k.Vk && (\mathcal{C}\text{-}\textsc{App}) \\
&\equiv V.
\end{aligned}$$

For proving Theorem 5 we need to collect some more facts. One crucial fact, which is also of general interest, is that all maps $\texttt{ftoc}^O_{A,B}$ are *rigid* in the sense of the following definition.

Definition 7 *A* rigid map *is an algebraic value $F : (A \to B) \to A' \to B'$ such that $F(\lambda x.Mx) \equiv \lambda y.FMy$ for all expressions $M : A \to B$.*

Remark 8 *Rigid functionals where introduced by Filinski [28] and play a crucial rôle in his "recursion-from-iteration" construction. Hasegawa and Kakutani [29] carefully tweaked the notion of rigid functionals to fit in with general axiomatics. The definition we use above is the tweaked version.*

Lemma 9 *Inverses of rigid maps are rigid.*

Because algebraic values of function type are equivalent to their own η-expansion, we have

Lemma 10 *Rigid maps preserve algebraic values.*

Lemma 11 *For all types A, B, and O, $\mathtt{ftoc}^{O}_{A,B}$ is rigid.*

Proof of Theorem 5 2⇒3: Let $\mathtt{ctof}_{1,B}$ be the inverse of $\mathtt{ftoc}_{1,B}$, and let $\mathcal{C}_B = \lambda h.\mathtt{ctof}_{1,B}(\lambda((),k).hk)()$. For the $\mathcal{C}$-App law, consider

$$\begin{aligned}
\mathcal{C}_B(\lambda k.kM) &\equiv \mathtt{ctof}_{1,B}(\lambda((),k).kM)() \\
&\equiv \mathtt{ctof}_{1,B}(\lambda((),k).k((\lambda().M)()))() \\
&\equiv \mathtt{ctof}_{1,B}(\mathtt{ftoc}_{1,B}(\lambda().M))() \\
&\equiv (\lambda().M))() \\
&\equiv M.
\end{aligned}$$

For the $\mathcal{C}$-Delay law, note that by Lemmas 9, 10, and 11, $\mathtt{ctof}_{1,B}$ applied to a lambda-expression is an algebraic value, and consider

$$\begin{aligned}
\lambda k.k(\mathcal{C}_B V) &= \lambda k.k(\mathtt{ctof}_{1,B}(\lambda((),k).Vk)()) \\
&\equiv \mathtt{let}\, f \,\mathtt{be}\, \mathtt{ctof}_{1,B}(\lambda((),k).Vk) \,\mathtt{in}\, \lambda k.k(f()) \\
&\equiv \mathtt{let}\, f \,\mathtt{be}\, \mathtt{ctof}_{1,B}(\lambda((),k).Vk) \,\mathtt{in}\, \lambda k.\mathtt{ftoc}_{1,B} f((),k) \\
&\equiv \lambda k.\mathtt{ftoc}_{1,B}(\mathtt{ctof}_{1,B}(\lambda((),k).Vk))((),k) \\
&\equiv V.
\end{aligned}$$

For 3⇒1, let $\mathcal{C}_B : ((B \to O) \to O) \to B$ be a closed algebraic value such that the $\mathcal{C}$-App and $\mathcal{C}$-Delay laws hold, and let $\mathtt{ctof}_{A,B} = \lambda h.\lambda x.\mathcal{C}_B(\lambda k.h(x,k))$. Then

$$\begin{aligned}
\mathtt{ftoc}_{A,B} \circ \mathtt{ctof}_{A,B} &\equiv \lambda h.\lambda(x,k).k(\mathcal{C}_B(\lambda k.h(x,k))) \\
&\equiv \lambda h.\lambda(x,k).h(x,k) && (\mathcal{C}\text{-Delay}) \\
&\equiv \lambda h.h
\end{aligned}$$

$$\begin{aligned}
\mathtt{ctof}_{A,B} \circ \mathtt{ftoc}_{A,B} &\equiv \lambda f.\lambda x.\mathcal{C}_B(\lambda k.k(fx)) \\
&\equiv \lambda f.\lambda x.fx && (\mathcal{C}\text{-App}) \\
&\equiv \lambda f.f.
\end{aligned}$$

For the uniqueness of $\mathcal{C}_B$, let $\mathcal{C}_B$ and $\mathcal{C}'_B$ be as in Condition 3, and consider

$$\begin{aligned}
\mathcal{C}_B &\equiv \lambda f.\mathcal{C}_B f && \text{(because } \mathcal{C}_B \text{ is an alg. val.)} \\
&\equiv \lambda f.\mathcal{C}'_B(\lambda k.k(\mathcal{C}_B f)) && (\mathcal{C}\text{-App}) \\
&\equiv \lambda f.\mathcal{C}'_B f && (\mathcal{C}\text{-Delay}) \\
&\equiv \mathcal{C}'_B && \text{(because } \mathcal{C}_B \text{ is an alg. val.).}
\end{aligned}$$

Remark 12 *Our map $\mathtt{ctof}_{A,B}$ is the same as Filinski's map* `switch` *[28], except that* `switch` *was never seen as a primitive.*

Definition 13 *A* $\mathcal{C}$-language *is a* λ_C*-language together with a base type* O *and a type-indexed family of constants* $\mathcal{C}_B : ((B \to O) \to O) \to B$. *A* $\mathcal{C}$-theory *is a* λ_C*-theory on a* $\mathcal{C}$*-language that satisfies* $\mathcal{C}$-App *and* $\mathcal{C}$-Delay.

Lemma 14 *In every* $\mathcal{C}$*-theory, an expression* $\Gamma \vdash M : A$ *is an algebraic value if and only if*

$$\Gamma \vdash \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \lambda k.k\, x \equiv \lambda k.k\, M : (A \to O) \to O. \tag{5}$$

To see the right-to-left implication, assume Equation 5, and consider

$$\begin{aligned}
\texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, N &\equiv \texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \texttt{let}\, y\, \texttt{be}\, \lambda k.k\, x\, \texttt{in}\, N[\mathcal{C}\, y/x] && (\mathcal{C}\text{-Delay})\\
&\equiv \texttt{let}\, y\, \texttt{be}\, (\texttt{let}\, x\, \texttt{be}\, M\, \texttt{in}\, \lambda k.k\, x)\, \texttt{in}\, N[\mathcal{C}\, y/x] &&\\
&\equiv \texttt{let}\, y\, \texttt{be}\, \lambda k.k\, M\, \texttt{in}\, N[\mathcal{C}\, y/x] && (\text{Equation 5})\\
&\equiv N[M/x] && (\mathcal{C}\text{-Delay}).
\end{aligned}$$

4 CPS calculus and CPS transform

In this section, we present the target language of the CPS transform as a calculus in its own right. The intention is to make the connection with intermediate languages in CPS compilers [5,21] more explicit.

Furthermore, since the CPS calculus is very close to the π-calculus, the role of CPS in translating call-by-value λ-calculus into the π-calculus is also clarified. The reconstruction of such translations as continuation-passing was independently discovered by a number of researchers [30,18].

4.1 The CPS calculus

The CPS calculus is presented in Figure 2. We distinguish between *primitive expressions* P, Q, which have a type, and *command expressions* L, M, N, which have no type. The *CPS language* over a set of base types b and a set of operators $f : A \longrightarrow B$ is the set of judgments $\Gamma \vdash P : A$ and $\Gamma \vdash M$ which is generated by the term formation rules in Figure 2. We use

$$M\{k\langle x_1, \ldots, x_n\rangle = N\}$$

as "syntactic sugar" for

$$M\{k\langle y\rangle = N[\pi_1(y)/x_1, \ldots, \pi_n(y)/x_n]\},$$

$$A, B ::= b \mid \neg A \mid A \times B \mid 1$$

$$\frac{}{\Gamma \vdash x : A}\, x : A \in \Gamma \qquad \frac{\Gamma \vdash P : A}{\Gamma \vdash f(P) : B}\, f : A \longrightarrow B$$

$$\frac{\Gamma \vdash P : A \qquad \Gamma \vdash Q : B}{\Gamma \vdash (P, Q) : A \times B} \qquad \frac{\Gamma \vdash P : A_1 \times A_2}{\Gamma \vdash \pi_i(P) : A_i} \qquad \Gamma \vdash () : 1$$

$$\frac{\Gamma \vdash P : \neg A \qquad \Gamma \vdash Q : A}{\Gamma \vdash P\langle Q\rangle}$$

$$\frac{\Gamma, k : \neg A \vdash M \qquad \Gamma, x : A \vdash N}{\Gamma \vdash M\{k\langle x\rangle = N\}}$$

$$\pi_i(P_1, P_2) \equiv P_i \qquad (\pi_1(P), \pi_2(P)) \equiv P \qquad P \equiv ()$$

$$\begin{array}{ll}
(\textsc{Distr}) & L\{m\langle x\rangle = M\}\{n\langle y\rangle = N\} \equiv L\{n\langle y\rangle = N\}\{m\langle x\rangle = M\{n\langle y\rangle = N\}\} \\
& \qquad (m \neq n \neq x \text{ and } m, x \notin \mathrm{FV}(N)) \\
(\mathrm{GC}) & M\{k\langle x\rangle = N\} \equiv M \qquad (k \notin \mathrm{FV}(M)) \\
(\textsc{Jmp}) & k\langle P\rangle\{k\langle x\rangle = M\} \equiv M[P/x] \qquad (k \notin \mathrm{FV}(P)) \\
(\textsc{Eta}) & M\{k\langle x\rangle = k'\langle x\rangle\} \equiv M[k'/k] \\
(\textsc{Contr}) & M\{k\langle x\rangle = N\}\{k'\langle x\rangle = N\} \equiv M[k/k']\{k\langle x\rangle = N\}
\end{array}$$

Fig. 2. The CPS calculus

where y is a fresh variable. A *CPS theory* $\mathcal{T}_{\mathrm{CPS}}$ over a CPS language $\mathcal{L}_{CPS}$ is a set of equations $\Gamma \vdash P \equiv Q : A$, where $\Gamma \vdash P : A$ and $\Gamma \vdash Q : A$ are in $\mathcal{L}_{CPS}$, as well as equations $\Gamma \vdash M \equiv N$, where $\Gamma \vdash M$ and $\Gamma \vdash N$ are in $\mathcal{L}_{CPS}$, such that $\mathcal{T}_{\mathrm{CPS}}$ contains all equations described in Figure 2, and is a congruence stable under weakening and permutation. A *zero* in a *CPS theory* is a type 0 such that there is a primitive expression $\vdash [] : \neg 0$ that satisfies all well-typed equations of the form $M \equiv []$. We abbreviate $P\langle(Q_1, Q_2)\rangle$ by $P\langle Q_1, Q_2\rangle$ and $P\langle()\rangle$ by $P\langle\rangle$.

Remark 15 *In the theoretical literature on continuations, the target language of the CPS transform is taken to be a subset of the λ-calculus. In some compilers, most notably Appel's original compiler for Standard ML of New Jersey [21], a specialized notation for CPS is used. The CPS calculus is designed to convey the spirit of such intermediate languages, in particular their imperative semantics in terms of jumping.*

If we restrict ourselves to the subset where primitive expressions can only be variables, then command expressions of the CPS calculus can be translated into processes of the π-calculus as follows:

$$\begin{aligned} (k\langle x\rangle)^\pi &= \overline{k}\langle x\rangle \\ (M\{k\langle x\rangle = N\})^\pi &= (\nu k)(M^\pi \mid !k(x).(N^\pi)). \end{aligned}$$

Intuitively, a continuation is encoded as a write-only channel, and jumping becomes sending along such a channel.

4.2 The CPS transform

A *continuation-passing-style transform* (CPS transform) $\kappa : \mathcal{L} \longrightarrow \mathcal{T}_{\mathrm{CPS}}$ from a λ_C-language $\mathcal{L}$ into a CPS theory $\mathcal{T}_{\mathrm{CPS}}$ is a map κ that sends every type A of $\mathcal{L}$ to a type A^κ of $\mathcal{T}_{\mathrm{CPS}}$, and every expression M of $\mathcal{L}$ together with a variable $k \notin \mathrm{FV}(M)$ to an expression $M^\kappa(k)$ of $\mathcal{T}_{\mathrm{CPS}}$, such that

(1) On types, κ behaves according to the rules

$$(A \to B)^\kappa = \neg(A^\kappa \times \neg B^\kappa) \qquad (A \times B)^\kappa = A^\kappa \times B^\kappa \qquad 1^\kappa = 1,$$

(2) For every expression $\Gamma \vdash M : B$ in $\mathcal{L}$, the expression $\Gamma^\kappa, k : \neg B^\kappa \vdash M^\kappa(k)$ is well typed in $\mathcal{T}_{\mathrm{CPS}}$, where Γ^κ stands for $x_1 : A_1^\kappa, \ldots, x_n : A_n^\kappa$ whenever Γ is $x_1 : A_1, \ldots, x_n : A_n$,
(3) On expressions, κ behaves according to the rules

$$\begin{aligned} x^\kappa(k) &= k\langle x\rangle \\ c^\kappa(k) &= k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\} \\ &\qquad \text{where } k \notin \mathrm{FV}(P),\ k \neq l_i, \text{ and } k \notin \mathrm{FV}(L_i) \text{ for } i \in \{1, \ldots, n\} \\ (\lambda x.M)^\kappa(k) &= k\langle h\rangle\{h\langle x, l\rangle = M^\kappa(l)\} \\ (MN)^\kappa(k) &= M^\kappa(k_1)\{k_1\langle m\rangle = N^\kappa(k_2)\{k_2\langle n\rangle = m\langle n, k\rangle\}\} \\ (M, N)^\kappa(k) &= M^\kappa(k_1)\{k_1\langle m\rangle = N^\kappa(k_2)\{k_2\langle n\rangle = k\langle m, n\rangle\}\} \\ (\pi_i M)^\kappa(k) &= M^\kappa(k')\{k'\langle m_1, m_2\rangle = k\langle m_i\rangle\} \\ ()^\kappa(k) &= k\langle\rangle. \end{aligned}$$

(P and $L_1, \cdots, L_n$ must be the same for every occurrence of c.)

We call an expression L of $\mathcal{L}$ a *κ-value* if it holds in $\mathcal{T}_{\mathrm{CPS}}$ that

$$L^\kappa(k) \equiv k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}$$

for some P and $L_1, \ldots, L_n$ such that $k \notin \mathrm{FV}(P)$, $k \neq l_i$, and $k \notin \mathrm{FV}(L_i)$ for $i \in \{1, \ldots, n\}$.

Lemma 16 *For every CPS transform κ, every value V is a κ-value.*

PROOF. By induction on V.

Lemma 17 *Let $\kappa : \mathcal{L} \longrightarrow \mathcal{T}_{\mathrm{CPS}}$ be a CPS transform and let $\Gamma \vdash L : A$ be a κ-value with $L^\kappa(k) \equiv k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}$. Then for every expression $\Gamma, x : A \vdash M : B$ it holds in $\mathcal{T}_{\mathrm{CPS}}$ that*

$$(M[L/x])^\kappa(k) \equiv M^\kappa(k)[P/x]\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}.$$

PROOF. By induction over M.

Lemma 18 *For every CPS transform $\kappa : \mathcal{L} \longrightarrow \mathcal{T}_{\mathrm{CPS}}$, every κ-value $\Gamma \vdash L : A$ of $\mathcal{L}$, and every expression $\Gamma, x : A \vdash M : B$ of $\mathcal{L}$, it holds in $\mathcal{T}_{\mathrm{CPS}}$ that*

$$(\mathtt{let}\, x\, \mathtt{be}\, L\, \mathtt{in}\, M)^\kappa(k) \equiv (M[L/x])^\kappa(k).$$

PROOF. Let $L^\kappa(k) \equiv k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}$, and consider

$$\begin{array}{ll}
(\mathtt{let}\, x\, \mathtt{be}\, L\, \mathtt{in}\, M)^\kappa(k') & \\
\equiv L^\kappa(k)\{k\langle x\rangle = M^\kappa(k')\} & \\
\equiv k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}\{k\langle x\rangle = M^\kappa(k')\} & \\
\equiv k\langle P\rangle\{k\langle x\rangle = M^\kappa(k')\}\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\} & (\textsc{Distr}, \mathrm{GC}) \\
\equiv M^\kappa(k')[P/x]\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\} & (\textsc{Jmp}) \\
\equiv (M[L/x])^\kappa(k') & (\text{Lemma 17}).
\end{array}$$

Corollary 19 *Every CPS transform κ validates the rule $\mathtt{let}\, x\, \mathtt{be}\, V\, \mathtt{in}\, M \equiv M[V/x]$.*

Proposition 20 ***(Soundness)*** *For every CPS transform $\kappa : \mathcal{L} \to \mathcal{T}_{\mathrm{CPS}}$ into a CPS theory $\mathcal{T}_{\mathrm{CPS}}$, the well-typed equations $\Gamma \vdash M \equiv N : A$ over $\mathcal{L}$ for which $\Gamma^\kappa, k : \neg A^\kappa \vdash M^\kappa(k) \equiv N^\kappa(k)$ holds in $\mathcal{T}_{\mathrm{CPS}}$ form a λ_C-theory $\kappa^{-1}(\mathcal{T}_{\mathrm{CPS}})$. Moreover, if $\mathcal{L}$ is a $\mathcal{C}$-language, $\mathcal{T}_{\mathrm{CPS}}$ is a CPS theory with a zero 0, and $O^\kappa = 0$, and $\mathcal{C}^\kappa_B(k) \equiv k\langle f\rangle\{f\langle h, l\rangle = h\langle m, []\rangle\{m\langle x, []\rangle = l\langle x\rangle\}\}$, then $\kappa^{-1}(\mathcal{T}_{\mathrm{CPS}})$ is a $\mathcal{C}$-theory.*

PROOF. We have to check that κ validates the equational rules in Figure 1, as well as the rules $\mathcal{C}$-App and $\mathcal{C}$-Delay. As stated by Corollary 19, κ validates the rule $\mathtt{let}\, x\, \mathtt{be}\, V\, \mathtt{in}\, M \equiv M[V/x]$, and therefore it suffices to check the other equational rules under the assumption that V is a variable. This amounts to straightforward (albeit laborious) calculations in the CPS calculus.

5 Response calculus and lambda transform

In this section, we show that the well-known CPS transform into the λ-calculus is given by our CPS transform into the CPS calculus followed by a "lambda transform" from the latter into the λ-calculus. We shall need only the fragment of the λ-calculus which is given by restricting function types to the form $A \to R$, for some fixed *response type* R. We shall call this fragment the "response calculus".

5.1 The response calculus

The *response language* over given base types and operators is the set of expressions $\Gamma \vdash M : A$ generated by the term formation rules in Figure 3. A *response theory* $\mathcal{T}_R$ over a response language $\mathcal{L}_R$ is a set of equations $\Gamma \vdash M \equiv N : A$ where $\Gamma \vdash M : A$ and $\Gamma \vdash N : A$ are in $\mathcal{L}_R$ that contains all equations described in Figure 3, and is a congruence stable under weakening and permutation. We write $|\mathcal{T}_R|$ for the underlying language of $\mathcal{T}_R$. So, the response calculus is like the ordinary lambda calculus with product and unit types, except that function types are restricted to the form $A \to R$.

The evident semantics of the response calculus is given by categories with finite products and an object R for which there are exponentials R^A. We call such categories *response categories.*

Remark 21 *Our response categories are a generalization Selinger's response categories [19]: the latter are required to have finite sums, over which the finite products must distribute, and the evident map $\eta : A \longrightarrow R^{R^A}$ must be a mono.*

A *zero* in a response theory is a type 0 such that there is an expression $\vdash [] : 0 \to R$ that satisfies all well-typed equations of the form $M \equiv []$. From a categorical point of view, this means requiring that the object R^0 is terminal[2].

[2] But 0 need not be initial: while there is a unique morphism from 0 to R, there need not be a unique morphism from 0 to every object.

Types $A, B ::= R \mid A \to R \mid A \times B \mid 1 \mid b$ where b ranges over base types

$$\frac{}{\Gamma \vdash x : A}\, x : A \in \Gamma \qquad \frac{\Gamma \vdash M : A}{\Gamma \vdash f(M) : B}\text{ if } f : A \longrightarrow B$$

$$\frac{\Gamma, x : A \vdash M : R}{\Gamma \vdash \lambda x : A.M : A \to R} \qquad \frac{\Gamma \vdash M : A \to R \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : R}$$

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash (M, N) : A \times B} \qquad \frac{\Gamma \vdash M : A_1 \times A_2}{\Gamma \vdash \pi_i(M) : A_i} \qquad \Gamma \vdash () : 1$$

$$\begin{aligned} (\lambda x.M)N &\equiv M[N/x] \\ \lambda x.Mx &\equiv M && (x \notin \mathrm{FV}(M)) \\ \pi_i(M_1, M_2) &\equiv M_i \\ (\pi_1 M, \pi_2 M) &\equiv M \\ M &\equiv () \end{aligned}$$

Fig. 3. The response calculus

5.2 The lambda transform

A *lambda transform* $(-)^\lambda : \mathcal{L}_{\mathrm{CPS}} \longrightarrow \mathcal{L}_R$ from a CPS language $\mathcal{L}_{CPS}$ into a response language $\mathcal{L}_R$ is a map $(-)^\lambda$ that sends every type A of $\mathcal{L}_{CPS}$ to a type A^λ of $\mathcal{L}_R$, every primitive expression P of $\mathcal{L}_{CPS}$ to an expression P^λ of $\mathcal{L}_R$, and every command expression M of $\mathcal{L}_{CPS}$ to an expression M^λ of $\mathcal{L}_R$, such that

(1) On types, $(-)^\lambda$ behaves according to the rules

$$(\neg A)^\lambda = A^\lambda \to R \qquad (A \times B)^\lambda = A^\lambda \times B^\lambda \qquad 1^\lambda = 1,$$

(2) For every primitive expression $\Gamma \vdash P : A$ in $\mathcal{L}_{CPS}$, $\Gamma^\lambda \vdash P^\lambda : A^\lambda$ is well-typed in $\mathcal{L}_R$, and for a command expression $\Gamma \vdash M$ in $\mathcal{L}_{CPS}$, the expression $\Gamma^\lambda \vdash M^\lambda : R$ is well typed in $\mathcal{L}_R$, where Γ^λ stands for x_1 :

$A_1^\lambda, \ldots, x_n : A_n^\lambda$ whenever Γ is $x_1 : A_1, \ldots, x_n : A_n$,

(3) On primitive expressions, $(-)^\lambda$ behaves according to the rules

$$\begin{aligned} x^\lambda &= x \\ (P, Q)^\lambda &= (P^\lambda, Q^\lambda) \\ (\pi_i(P))^\lambda &= \pi_i(P^\lambda) \\ ()^\lambda &= () \\ (f(P))^\lambda &= M_\lambda^f[P^\lambda/x] \end{aligned}$$

where $x : A^\lambda \vdash M_\lambda^f : B^\lambda$ is an expression in $\mathcal{L}_R$ if $f : A \longrightarrow B$ is an operator of $\mathcal{L}_{CPS}$. On command expressions, $(-)^\lambda$ behaves according to the rules

$$(M\{k\langle x\rangle = N\})^\lambda = (\lambda k.M^\lambda)(\lambda x.N^\lambda) \qquad (P\langle Q\rangle)^\lambda = P^\lambda Q^\lambda.$$

Proposition 22 ***(Soundness)*** *For every lambda transform $\lambda : \mathcal{L}_{\mathrm{CPS}} \to \mathcal{T}_R$ into a response theory $\mathcal{T}_R$, the well-typed equations $\Gamma \vdash M \equiv N$ over $\mathcal{L}_{\mathrm{CPS}}$ for which $\Gamma^\lambda \vdash M^\lambda \equiv N^\lambda : R$ holds in $\mathcal{T}_R$ form a CPS theory $\lambda^{-1}(\mathcal{T}_R)$. Moreover, if $\mathcal{T}_R$ has a zero 0, and $\mathcal{L}_{\mathrm{CPS}}$ has a closed primitive expression $\vdash [] : \neg O$ for some type O such that $O^\lambda = 0$, then O is a zero in $\lambda^{-1}(\mathcal{T}_R)$.*

5.3 The CPS transform into the lambda calculus.

Next we show that the well-known call-by-value CPS transform into the lambda calculus (here: response calculus) is essentially the composition $\lambda \circ \kappa$. A CPS transform $\gamma : \mathcal{L} \longrightarrow \mathcal{T}_R$ from a λ_C-language $\mathcal{L}$ into a response theory $\mathcal{T}_R$ is a map γ from types of $\mathcal{L}$ to types of $\mathcal{T}_R$ and from expressions of $\mathcal{L}$ to expressions of $\mathcal{T}_R$, such that

(1) On types, γ behaves according to the rules

$$(A \to B)^\gamma = (A^\gamma \times (B^\gamma \to R)) \to R \quad (A \times B)^\gamma = A^\gamma \times B^\gamma \quad 1^\gamma = 1,$$

(2) For every expression $\Gamma \vdash M : B$ in $\mathcal{L}$, the expression $\Gamma^\gamma \vdash M^\gamma : (B^\gamma \to R) \to R$ is in well-typed in $\mathcal{T}_R$, where Γ^γ stands for $x_1 : A_1^\gamma, \ldots, x_n : A_n^\gamma$ whenever Γ is $x_1 : A_1, \ldots, x_n : A_n$,

(3) On expressions, γ behaves according to the rules

$$\begin{aligned}
x^\gamma &= \lambda k.kx \\
c^\gamma &= \lambda k.k(P_c) \quad (k \notin \mathrm{FV}(P_c)) \\
(\lambda x.M)^\gamma &= \lambda k.k(\lambda(x,h).(M^\gamma h)) \\
(MN)^\gamma &= \lambda k.M^\gamma(\lambda m.N^\gamma(\lambda n.m(n,k))) \\
(M,N)^\gamma &= \lambda k.M^\gamma(\lambda m.N^\gamma(\lambda n.k(m,n)) \\
(\pi_i(M))^\gamma &= \lambda k.M(\lambda m.k(\pi_i m)) \\
()^\gamma &= \lambda k.k().
\end{aligned}$$

Proposition 23 *Let $\kappa : \mathcal{L} \longrightarrow \mathcal{T}_{\mathrm{CPS}}$ be a CPS transform, $\lambda : \mathcal{L}_{CPS} \longrightarrow \mathcal{T}_R$ a lambda transform, and $\gamma : \mathcal{L} \longrightarrow \mathcal{T}_R$ a CPS transform into the response calculus such that $(b^\kappa)^\lambda = b^\gamma$ and $c^\gamma k \equiv (c^\kappa(k))^\lambda$ for all base types b and constants c of $\mathcal{L}$. Then it holds in $\mathcal{T}_R$ that*

$$M^\gamma k \equiv (M^\kappa(k))^\lambda.$$

PROOF. By induction over M.

Now we turn to proving the soundness of γ. It can be proved by using Proposition 23 and the fact that sound maps compose. However, there is an alternative, self-contained proof which analogous to the one for κ:

Lemma 24 *For every CPS transform γ, for every value V it holds that $V^\gamma \equiv \lambda k.kP$ for some P.*

Remark 25 *The form $k\langle P\rangle\{l_1\langle x_1\rangle = L_1\}\ldots\{l_n\langle x_n\rangle = L_n\}$ of V^κ had to be more complicated than the form $\lambda k.kP$ of V^γ because the syntax of the CPS calculus does not admit expressions like $k\langle \mathtt{let}\, l_1\, \mathtt{be}\, L_1\, \mathtt{in}\, \ldots \mathtt{let}\, l_n\, \mathtt{be}\, L_n\, \mathtt{in}\, P\rangle$.*

Lemma 26 *Let $\gamma : \mathcal{L} \longrightarrow \mathcal{T}_R$ be a CPS transform, and let $\Gamma \vdash L : A$ be an expression of $\mathcal{L}$ such that $L^\gamma \equiv \lambda k.kP$ for some P. Then for every expression $\Gamma, x : A \vdash M : B$ it holds in $\mathcal{T}_R$ that $(M[L/x])^\gamma \equiv M^\gamma[P/x]$.*

Lemma 27 *For every CPS transform $\kappa : \mathcal{L} \longrightarrow \mathcal{T}_R$, every expression $\Gamma \vdash L : A$ with $L^\gamma = \lambda k.kP$, and every expression $\Gamma, x : A \vdash M : B$ of $\mathcal{L}$, it holds in $\mathcal{T}_R$ that*

$$(\mathtt{let}\, x\, \mathtt{be}\, L\, \mathtt{in}\, M)^\kappa(k) \equiv (M[L/x])^\kappa(k).$$

Corollary 28 *Every CPS transform γ validates the rule $\mathtt{let}\, x\, \mathtt{be}\, V\, \mathtt{in}\, M \equiv M[V/x]$.*

Proposition 29 ***(Soundness)*** *For every CPS transform $\gamma : \mathcal{L} \to \mathcal{T}_R$, the well-typed equations $\Gamma \vdash M \equiv N : A$ over $\mathcal{L}$ for which $\Gamma^\gamma \vdash M^\gamma \equiv N^\gamma : (A^\gamma \to R) \to R$ holds in $\mathcal{T}_R$ form a λ_C-theory $\gamma^{-1}(\mathcal{T}_R)$. Moreover, if $\mathcal{L}$ is a $\mathcal{C}$-language, $\mathcal{T}_R$ has a zero 0, and $O^\gamma = 0$, and ${\mathcal{C}_B}^\gamma \equiv \lambda k.k(\lambda(h,l).h(\lambda(x,[]).lx,[]))$ then $\gamma^{-1}(\mathcal{T}_R)$ is a $\mathcal{C}$-theory*[3] .

6 The delaying transform

Traditionally, the CPS transform is considered to be a semantics of the λ_C-calculus (with and without control operators). But there is also a sound transform, which we shall introduce as the "delaying transform", in the opposite direction. This transform, which can be seen as a semantics of the response calculus in terms of the λ_C-calculus, pushes the balance towards considering the λ_C-calculus to be fundamental rather then derived.

As we shall see, the delaying transform provides a remarkable way of defining a CPS term model of a λ_C-theory, and thus yields a novel completeness proof for the CPS transform (with respect to $\mathcal{C}$-theories). Also, our CPS term model satisfies the equalizer requirement, so our completeness result is stronger than usual. However, independently of completeness, we consider the delaying transform to be interesting in its own right.

Definition 30 *A* delaying transform $\delta : \mathcal{L}_R \longrightarrow \mathcal{T}$ *from a response language $\mathcal{L}_R$ to a λ_C-theory $\mathcal{T}$ with a chosen type O is a map δ from types of $\mathcal{L}_R$ to types of $\mathcal{T}$ and from expressions of $\mathcal{L}_R$ to expressions of $\mathcal{T}$ such that*

$$R^\delta = 1 \to O \qquad (A \to R)^\delta = A^\delta \to O \qquad (A \times B)^\delta = A^\delta \times B^\delta \qquad 1^\delta = 1$$

and every expression $x_1 : A_1, \ldots, x_n : A_n \vdash M : B$ is sent to an expression $x_1 : A_1^\delta, \ldots, x_n : A_n^\delta \vdash M^\delta : B^\delta$ according to the rules

$$\begin{aligned}
x^\delta &= x \\
(f(M))^\delta &= f^\delta(M^\delta) \\
(\lambda x.M)^\delta &= \lambda x.(M^\delta)() \\
(MN)^\delta &= \lambda().M^\delta\, N^\delta \\
(M,N)^\delta &= (M^\delta, N^\delta) \\
(\pi_i(M))^\delta &= \pi_i(M^\delta) \\
()^\delta &= ()
\end{aligned}$$

[3] If the type of ${\mathcal{C}_B}^\gamma$ did not contain occurrences of the terminal type $0 \to R$, we could use the simpler, more familiar condition ${\mathcal{C}_B}^\gamma \equiv \lambda k.k(\lambda(h,l).hl)$.

where, letting $A \longrightarrow B$ be the typing of the operator f above, f^δ is a closed expression of type $A^\delta \to B^\delta$ such that $x : A^\delta \vdash f^\delta x : B^\delta$ is an algebraic value.

Lemma 31 *If $\delta : \mathcal{L}_R \longrightarrow \mathcal{T}$ is a delaying transform, then for every expression $\Gamma \vdash M : A$ in $\mathcal{L}_R$, the expression $\Gamma^\delta \vdash M^\delta : A^\delta$ is an algebraic value.*

Lemma 32 *If $\delta : \mathcal{L}_R \longrightarrow \mathcal{T}$ is a delaying transform, then for all expressions $\Gamma, x : A \vdash M : B$ and $\Gamma \vdash N : A$ of $\mathcal{L}_R$, the equation $\Gamma^\delta \vdash (M[N/x])^\delta \equiv M^\delta[N^\delta/x] : B^\delta$ holds in $\mathcal{T}$.*

Proposition 33 ***(Soundness)*** *For every delaying transform $\delta : \mathcal{L}_R \to \mathcal{T}$, the equations $\Gamma \vdash M \equiv N : A$ over $\mathcal{L}_R$ for which the equation $\Gamma^\delta \vdash M^\delta \equiv N^\delta : A^\delta$ holds in $\mathcal{T}$ form a response theory $\delta^{-1}(\mathcal{T})$. Moreover, if $\mathcal{L}_R$ has a type 0 such that $0^\delta = O$ and an expression $\vdash [] : 0 \to R$, then 0 is a zero in $\mathcal{L}_R$.*

PROOF. For the first claim, simply check the equations in Figure 3 with the help of Lemmas 1, 31, and 32. For the second claim, let $0^\delta = O$ and $[]^\delta = \mathcal{A}_O$. We must check that all well-typed equations of the form $M \equiv []$ hold in $\mathcal{L}_R$, that is, $M^\delta \equiv []^\delta$ holds in $\mathcal{T}$. By Lemma 31, M^δ and $[]^\delta$ are algebraic values. In $\mathcal{T}$ all algebraic values of type $O \to O$ are equivalent to $\mathcal{A}_O$, so the claim follows from Lemma 6.

Remark 34 *It is part of the functional programming folklore that one can encode a form of call-by-name (although some prefer to call it "lazy") in call-by-value by delaying, or "thunking"*[4] *all arguments. Since we do not evaluate under λ-abstractions, prefixing an expression with a λ for a dummy argument, say the empty tuple $()$, effectively delays evaluation of the expression until the time when the dummy argument is supplied. Systematically introducing such dummy arguments everywhere amounts to the following transformation:*

$$\begin{aligned} x^\theta &= x() \\ (\lambda x.M)^\theta &= \lambda x.(M^\theta) \\ (MN)^\theta &= (M^\theta)(\lambda().N^\theta). \end{aligned}$$

Such a transform has been studied by Danvy and Hatcliff [32]. The delaying transform is based on a similar idea, but whereas the thunking transform delays the arguments*, the delaying transform delays the* results *of function calls. Both the thunking transform and our delaying transform validate the unrestricted β-law. However, the former does not validate the unrestricted η-law. Thus, it encodes a "lazy" semantics. By contrast, the delaying transform validates the unrestricted η-law, and can thus be seen as a "true call-by-name" transform (except that the source language is type-restricted).*

[4] The technical term "thunk" originates in ALGOL60, since it used a similar mechanism to implement call-by-name [31].

For every $\mathcal{C}$-theory $\mathcal{T}$, define a response language $\mathcal{L}_R$ over the same base types by giving operators as follows. For any type A of $\mathcal{T}$, let $A^{\delta_\mathcal{T}}$ stand for the identity-on-base-types delaying transform of A. For all types $A_1, \ldots, A_n$ and B of $\mathcal{L}_R$, and for every closed expression $M : A_1^{\delta_\mathcal{T}} \times \cdots \times A_n^{\delta_\mathcal{T}} \longrightarrow B^{\delta_\mathcal{T}}$ of $\mathcal{T}$ such that $x_1 : A_1^{\delta_\mathcal{T}}, \ldots, x_n : A_n^{\delta_\mathcal{T}} \vdash M(x_1, \ldots, x_n) : B^{\delta_\mathcal{T}}$ is an algebraic value, let $\mathcal{L}_R$ have an operator $f_M : A_1 \times \cdots \times A_n \longrightarrow B$, and let those be the only operators of $\mathcal{L}_R$. Define $\delta_\mathcal{T} : \mathcal{L}_R \longrightarrow \mathcal{T}$ to be the identity-on-base-types delaying transform that sends f_M to M. By soundness of the delaying transform, $\delta_\mathcal{T}^{-1}(\mathcal{T})$ is a response theory, and the type O is a zero with unique map $[] = f_{\lambda().\mathcal{A}_O}() : O \to R$.

Next, we turn to proving that $\delta_\mathcal{T}^{-1}(\mathcal{T})$ forms a "term" model of $\mathcal{T}$ (Proposition 39). The proof relies on the fact that a CPS transform followed by a delaying transform essentially sends an expression M to $\lambda k.kM$ (Lemma 38). "Essentially" means here that this claim holds up to a recursive-on-types version $\texttt{ftoc}^+$ of the isomorphism $\texttt{ftoc}$.

For every CPS transform γ and every delaying transform δ in the opposite direction, the composed transform $\delta \circ \gamma$ recursively replaces types of the form $A \to B$ by $A \times (B \to O) \to O$. Let $\mathcal{T}$ be a $\mathcal{C}$-theory, and let $\widehat{(-)}$ be the map from types to types that does this recursive replacement, and sends every base type to itself. Using $\texttt{ftoc}$ and $\texttt{ctof}$, we define type-indexed families of maps $\texttt{ftoc}^+_A : A \to \widehat{A}$ and $\texttt{ctof}^+_A : \widehat{A} \to A$ by mutual recursion:

$$\begin{aligned}
\texttt{ftoc}^+_b &= \lambda x.x \\
\texttt{ftoc}^+_1 &= \lambda x.x \\
\texttt{ftoc}^+_{A\times B} &= \lambda(x,y).(\texttt{ftoc}^+_A x, \texttt{ftoc}^+_B y) \\
\texttt{ftoc}^+_{A\to B} &= \lambda f.\texttt{ftoc}_{\widehat{A},\widehat{B}}(\texttt{ftoc}^+_B \circ f \circ \texttt{ctof}^+_A)
\end{aligned}$$

$$\begin{aligned}
\texttt{ctof}^+_b &= \lambda x.x \\
\texttt{ctof}^+_1 &= \lambda x.x \\
\texttt{ctof}^+_{A\times B} &= \lambda(x,y).(\texttt{ctof}^+_A x, \texttt{ctof}^+_B y) \\
\texttt{ctof}^+_{A\to B} &= \lambda h.\texttt{ctof}^+_B \circ (\texttt{ctof}_{\widehat{A},\widehat{B}} h) \circ \texttt{ftoc}^+_A.
\end{aligned}$$

Lemma 35 *All maps* $\texttt{ftoc}^+_A$ *and* $\texttt{ctof}^+_A$ *preserve algebraic values.*

PROOF. By induction over A, using the fact that $\texttt{ftoc}_{\widehat{A},\widehat{B}}$ is rigid and Lemma 10.

Lemma 36 *The maps* $\texttt{ftoc}^+_A$ *and* $\texttt{ctof}^+_A$ *are mutually inverse.*

PROOF. By induction over A, using Lemma 35.

Lemma 37 *In every $\mathcal{C}$-theory it holds that* $\texttt{ftoc}^+_{((B\to O)\to O)\to B}\mathcal{C}_B \equiv \lambda(h,l).h(\lambda(x,y).lx, \mathcal{A}_O)$.

PROOF. The proof proceeds in three steps:

$$\begin{aligned}
&\texttt{ctof}_{B,O}\\
&\equiv \lambda h.\lambda x.\mathcal{C}_O(\lambda k : O \to O.h(x,k))\\
&\equiv \lambda h.\lambda x.\mathcal{C}_O(\lambda k : O \to O.k(h(x,k))) && \text{(as } k = \lambda y.y \text{ by Lemma 6)}\\
&\equiv \lambda h.\lambda x.h(x, \mathcal{A}_O) && \text{(by (}\mathcal{C}\text{-App))} \qquad (6)
\end{aligned}$$

$$\begin{aligned}
&\texttt{ctof}^+_{(B\to O)\to O}\\
&\equiv \lambda h.\lambda f.\texttt{ctof}_{\widehat{B\to O},O}h(\texttt{ftoc}^+_{B\to O}f)\\
&\equiv \lambda h.\lambda f.h(\texttt{ftoc}^+_{B\to O}f, \mathcal{A}_O) && \text{(by Equation 6)}\\
&\equiv \lambda h.\lambda f.h(\texttt{ftoc}_{\widehat{B},O}(f \circ \texttt{ctof}^+_B), \mathcal{A}_O) && (7)
\end{aligned}$$

$$\begin{aligned}
&\texttt{ftoc}^+_{((B\to O)\to O)\to B}\mathcal{C}_B\\
&\equiv \lambda(h,k).(k \circ \texttt{ftoc}^+_B)(\mathcal{C}(\texttt{ctof}^+_{(B\to O)\to O}h))\\
&\equiv \lambda(h,k).\texttt{ctof}^+_{(B\to O)\to O}h(k \circ \texttt{ftoc}^+_B) && \text{(by (}\mathcal{C}\text{-Delay))}\\
&\equiv \lambda(h,k).h(\texttt{ftoc}_{\widehat{B},O}(k \circ \texttt{ftoc}^+_B \circ \texttt{ctof}^+_B), \mathcal{A}_O) && \text{(by Equation 7)}\\
&\equiv \lambda(h,k).h(\texttt{ftoc}_{\widehat{B},O}k, \mathcal{A}_O)\\
&\equiv \lambda(h,k).h(\lambda(x,y:O).kx, \mathcal{A}_O).
\end{aligned}$$

Now back to the proof that $\delta_{\mathcal{T}}^{-1}(\mathcal{T})$ forms a model. Define $\gamma_{\mathcal{T}} : \mathcal{T} \longrightarrow \delta_{\mathcal{T}}^{-1}(\mathcal{T})$ to be the identity-on-base-types CPS transform that sends every constant c to $\lambda k.k(f_{\lambda().\texttt{ftoc}^+c}())$, except for the control operator $\mathcal{C}$, which is sent to $\lambda k.k(\lambda(h,l).h(\lambda(x,y:0).lx, []))$. The following diagram outlines the situation:

$$\begin{tikzcd}
\mathcal{T} \arrow[r, shift left, "\gamma_{\mathcal{T}}"] & \delta_{\mathcal{T}}{}^{-1}(\mathcal{T}). \arrow[l, shift left, "\delta_{\mathcal{T}}"]
\end{tikzcd}$$

It holds that $(c^{\gamma_{\mathcal{T}}})^{\delta_{\mathcal{T}}} \equiv \lambda k.k(\texttt{ftoc}^+\, c)$ for every constant c. (The case $c \neq \mathcal{C}$ is immediate, and the case $c = \mathcal{C}$ follows from Lemma 37.) Therefore, the following Lemma applies to $\gamma_{\mathcal{T}}$ and $\delta_{\mathcal{T}}$.

Lemma 38 *Let $\mathcal{T}$ be a $\mathcal{C}$-theory, and let $\mathcal{L}_R$ be a response language over the same base types. Let $\delta : \mathcal{L}_R \longrightarrow \mathcal{T}$ be an identity-on-base-types delaying transform, and let $\gamma : \mathcal{T} \longrightarrow \delta^{-1}(\mathcal{T})$ be an identity-on-base-types CPS transform. If $(c^\gamma)^\delta \equiv \lambda k.k(\mathtt{ftoc}^+ c)$ holds for every constant c of $\mathcal{T}$, then for every expression $x_1 : A_1, \ldots, x_n : A_n \vdash M : A$ of $\mathcal{T}$, the equation below holds in $\mathcal{T}$.*

$$x_1 : (A_1^\gamma)^\delta, \ldots, x_n : (A_n^\gamma)^\delta \vdash (M^\gamma)^\delta \equiv \lambda k.k(\mathtt{ftoc}^+ M[\mathtt{ctof}^+ x_i/x_i]) : ((A^\gamma)^\delta \to 0) \to 0$$

PROOF. By induction on M, where the application and projection cases follow from the equations

$$\lambda k.\mathtt{ftoc}^+ L\,(\mathtt{ftoc}^+\, N, k) \equiv \lambda k.k(\mathtt{ftoc}^+(L\,N)) \quad \pi_i(\mathtt{ftoc}^+\, N) \equiv \mathtt{ftoc}^+(\pi_i(N))$$

which can easily be checked.

Proposition 39 *The interpretation $\gamma_\mathcal{T} : \mathcal{T} \longrightarrow \delta_\mathcal{T}^{-1}(\mathcal{T})$ is a model of $\mathcal{T}$ such that denotational equality implies equivalence in $\mathcal{T}$.*

To see this, consider

$$\begin{aligned}
& M^{\gamma_\mathcal{T}} \equiv N^{\gamma_\mathcal{T}} \in {\delta_\mathcal{T}}^{-1}(\mathcal{T}) \\
\iff\ & (M^{\gamma_\mathcal{T}})^{\delta_\mathcal{T}} \equiv (N^{\gamma_\mathcal{T}})^{\delta_\mathcal{T}} \in \mathcal{T} \\
\iff\ & \lambda k.k(\mathtt{ftoc}^+(M[\mathtt{ctof}^+ x_i/x_i])) \equiv \lambda k.k(\mathtt{ftoc}^+(N[\mathtt{ctof}^+ x_i/x_i])) \in \mathcal{T} \\
& \quad \text{(by Lemma 38)} \\
\iff\ & \mathtt{ftoc}^+(M[\mathtt{ctof}^+ x_i/x_i]) \equiv \mathtt{ftoc}^+(N[\mathtt{ctof}^+ x_i/x_i]) \in \mathcal{T} \qquad (\mathcal{C}\text{-}\textsc{App}) \\
\iff\ & M \equiv N \in \mathcal{T} \qquad \text{(because } \mathtt{ftoc}^+ \text{ and } \mathtt{ctof}^+ \text{ are isos)}
\end{aligned}$$

The following (essentially well-known) completeness theorem follows immediately.

Theorem 40 ***(Completeness)*** *$\mathcal{C}$-theories are complete for CPS transforms into response theories with zero.*

6.2 The CPS-calculus term model

We could have built the term model of a $\mathcal{C}$-theory from the CPS calculus rather than the response calculus, using the same basic technique as in the previous section. Here, we shall derive the CPS-calculus term model by "pulling back" the response-calculus term model along the lambda-transform.

Let $\mathcal{T}$ be a $\mathcal{C}$-theory $\mathcal{T}$. Let $\mathcal{L}_{CPS}$ be the CPS language with the same base types, and operators $f_c : 1 \longrightarrow \overline{A}$ for every constant $c^A \neq \mathcal{C}$ of $\mathcal{T}$, where $\overline{A}$ is the identity-on-base-types CPS transform of A. Define $\lambda_\mathcal{T} : \mathcal{L}_{CPS} \longrightarrow \delta_\mathcal{T}^{-1}(\mathcal{T})$

to be the identity-on-base types lambda transform that sends $f_c : 1 \longrightarrow \overline{A}$ to $f_c : 1 \longrightarrow A^{\gamma_{\mathcal{T}}}$. Define $\kappa_{\mathcal{T}} : |\mathcal{T}| \longrightarrow \mathcal{L}_{CPS}$ to be the identity-on-base types CPS transform with $c^{\kappa_{\mathcal{T}}}(k) = k\langle f_c()\rangle$ and $\mathcal{C}_B^{\kappa_{\mathcal{T}}}(k) = k\langle f\rangle\{f\langle h, l\rangle = h\langle m, []\rangle\{m\langle x, []\rangle = l\langle x\rangle\}\}$.

Proposition 41 *For every $\mathcal{C}$-theory $\mathcal{T}$, it holds that $\kappa_{\mathcal{T}}^{-1}(\lambda_{\mathcal{T}}{}^{-1}(\delta_{\mathcal{T}}^{-1}(\mathcal{T}))) = \mathcal{T}$.*

PROOF. We have

$$\begin{array}{lll}
M \equiv N & \in \kappa_{\mathcal{T}}^{-1}(\gamma_{\mathcal{T}}{}^{-1}(\delta_{\mathcal{T}}{}^{-1}(\mathcal{T}))) & \\
\iff (M^{\kappa_{\mathcal{T}}}(k))^{\lambda_{\mathcal{T}}} \equiv (N^{\kappa_{\mathcal{T}}}(k))^{\lambda_{\mathcal{T}}} & \in \delta_{\mathcal{T}}{}^{-1}(\mathcal{T}) & \\
\iff M^{\gamma_{\mathcal{T}}}k \equiv N^{\gamma_{\mathcal{T}}}k & \in \delta_{\mathcal{T}}{}^{-1}(\mathcal{T}) & \text{(Prop. 23)} \\
\iff M^{\gamma_{\mathcal{T}}} \equiv N^{\gamma_{\mathcal{T}}} & \in \delta_{\mathcal{T}}{}^{-1}(\mathcal{T}) & \\
\iff M \equiv N & \in \gamma_{\mathcal{T}}^{-1}(\delta_{\mathcal{T}}{}^{-1}(\mathcal{T})) & \\
\iff M \equiv N & \in \mathcal{T} & \text{(Prop. 39).}
\end{array}$$

The right-to-left inclusion in Proposition 41 means that the interpretation $\kappa_{\mathcal{T}} : \mathcal{T} \longrightarrow \lambda_{\mathcal{T}}^{-1}(\delta_{\mathcal{T}}^{-1}(T))$ is a model. The left-to-right inclusion means that equality in that model implies equivalence in $\mathcal{T}$. So we have

Theorem 42 *$\mathcal{C}$-theories are complete for CPS transforms into the CPS calculus.*

6.3 The equalizer requirement and sobriety

When a monad T is used to represent a computational effect (as described by Moggi, see e.g. [23]), an expression of type TA represents a computation which yields a value of type A *if it returns*, but—whether it returns or not—may also have an effect like throwing an exception or changing the contents of a variable. The monad's unit $\eta_A : A \longrightarrow TA$ sends a value to the effect-free computation that returns that value.

In this article, T ranges over continuations monads $R^{R^{(-)}}$. In the response calculus, we have $\eta_A = \lambda x : A.\lambda k : A \to R.kx$. Expressions ϕ of type $A \to R$ can be seen as observations about expressions of type A. Because $\eta M\phi = \phi M$, the expression ηM represents the results of all possible observations of M.

For every monad representing a computational effect, it is an evident sanity condition for η_A to be a mono. For a continuations monad, this means that a value is uniquely determined by the observations that can be made about it!

So one wonders when an arbitrary expression $\Gamma \vdash M : TA$ should be the computation corresponding to a (necessarily unique) value. It was shown in [33] that for "good" models this is so if and only if the denotation $f : \Gamma \longrightarrow TA$ of M satisfies the equation

$$\eta_{TA} \circ f = T\eta_A \circ f. \tag{8}$$

That is, such models are characterized by the following condition: if some morphism $f : A \longrightarrow TB$ satisfies Equation 8, then there is a unique $f' : A \longrightarrow B$ such that $f = \eta \circ f'$. In other words,

$$\begin{tikzcd} A \arrow[r, "\eta_A"] & TA \arrow[r, shift left, "\eta_{TA}"] \arrow[r, shift right, "T\eta_A"'] & TTA \end{tikzcd} \tag{9}$$

must be an equalizer diagram for all objects A. Führmann [33] showed (in categorical terms, which will be summarized in Section 7) that for every λ_C-theory $\mathcal{T}$, among the fully complete models whose objects are denotable, there is a unique one that satisfies the equalizer requirement.

Taylor [22] explored Equation 8 in the continuations case, in particular under a topological interpretation where R is the Sierpinski space Σ. (In fact, our discussion of "observations" above follows Taylor's article[5].) Taylor calls a space A *sober* if Diagram 9 is an equalizer. He shows that $f : 1 \longrightarrow \Sigma^{\Sigma^B}$ (which can be seen as a set of open sets) solves Equation 8 if and only if it is the sets of open neighborhoods of a unique point. As explained in [22], it follows that categorical sobriety agrees with topological sobriety. Furthermore, Taylor shows how to turn a category with an exponentiating object Σ into a category all whose object are sober. This is, up to isomorphism, a special case of the Führmann's construction mentioned above.

Next, we prove that the unique model satisfying the equalizer requirement is $\delta_{\mathcal{T}}^{-1}(\mathcal{T})$ (Proposition 44). As a consequence, we can strengthen the completeness result from the previous section and obtain the novel Theorem 45.

Note that, in the response calculus, Equation 8 is

$$\lambda k.kM \equiv \lambda k.M(\lambda x.k(\lambda l.lx)). \tag{10}$$

Lemma 43 *Let $\delta : \mathcal{L}_R \longrightarrow \mathcal{T}$ be a delaying transform. Then for every expression $\Gamma \vdash M : TA$ of $\delta^{-1}(\mathcal{T})$, Equation 10 holds if and only if $\Gamma^\delta \vdash \mathcal{C}(M^\delta) : A^\delta$ is an algebraic value.*

[5] Warning: the Sierpinski space provides a very limited notion of observation. In a model of a real-life programming language, R might contain every string that a program can print, every sound a program can produce, and so on.

PROOF. Applying δ to Equation 10 yields the following equation in $\mathcal{T}$.

$$\lambda k.k\, M^\delta \equiv \lambda k.M^\delta(\lambda x.k(\lambda l.l\, x)) \tag{11}$$

If $\mathcal{C}M^\delta$ is an algebraic value, Equation 11 holds because

$$\begin{array}{ll}
\lambda k.M^\delta(\lambda x.k(\lambda l.l\, x)) & \\
\quad \equiv \lambda k.(\lambda l.l(\mathcal{C}(M^\delta)))(\lambda x.k(\lambda l.l\, x)) & \text{(by } \mathcal{C}\text{-Delay, which applies because } M^\delta \\
 & \quad \text{is an alg. val. by Lemma 31)} \\
\quad \equiv \lambda k.(\lambda x.k(\lambda l.l\, x))(\mathcal{C}(M^\delta)) & \\
\quad \equiv \lambda k.k(\lambda l.l\,(\mathcal{C}(M^\delta))) & \text{(because } \mathcal{C}(M^\delta) \text{ is an algebraic value)} \\
\quad \equiv \lambda k.k(M^\delta) & (\mathcal{C}\text{-Delay}).
\end{array}$$

Conversely, Equation 11 implies

$$\begin{array}{rll}
\lambda l.l\,(\mathcal{C}(M^\delta)) & \equiv M^\delta & (\mathcal{C}\text{-Delay}) \\
 & \equiv \mathcal{C}(\lambda k.k(M^\delta)) & (\mathcal{C}\text{-App}) \\
 & \equiv \mathcal{C}(\lambda k.M^\delta(\lambda x.k(\lambda l.l\, x))) & \text{(Equation 11)} \\
 & \equiv (\lambda x.\lambda l.l\, x)(\mathcal{C}(M^\delta)) & (\mathcal{C}\text{-Nat}) \\
 & \equiv \texttt{let}\, x\, \texttt{be}\, \mathcal{C}(M^\delta)\, \texttt{in}\, \lambda l.l\, x &
\end{array}$$

which by Lemma 14 implies that $\mathcal{C}(M^\delta)$ is an algebraic value.

Proposition 44 *For every λ_C-theory $\mathcal{T}$, the term model $\delta_{\mathcal{T}}^{-1}(\mathcal{T})$ satisfies the equalizer requirement.*

PROOF. Let δ stand short for $\delta_{\mathcal{T}}$ in this proof. Suppose that Equation 10 holds for some expression $x_1 : A_1, \ldots, x_n : A_n \vdash M : TA$ of $\delta^{-1}(\mathcal{T})$. We need a unique (up to $\equiv$) expression $x_1 : A_1, \ldots, x_n : A_n \vdash M' : A$ such that the equation $M \equiv \eta(M')$ holds in $\delta^{-1}(\mathcal{T})$. We claim that the required M' is $f_{\lambda\vec{x}.\mathcal{C}(M^\delta)}(\vec{x})$. For this to exist, $\vec{x} \vdash \mathcal{C}(M^\delta) : A^\delta$ must be an algebraic value, which it is by Lemma 43. To see that $M \equiv \eta(M')$, consider

$$\begin{array}{rll}
(\eta(M'))^\delta & \equiv (\lambda k.kM')^\delta & \\
 & \equiv (\lambda k.k(\mathcal{C}\,(M^\delta)) & \\
 & \equiv M^\delta & (\mathcal{C}\text{-Delay}).
\end{array}$$

For uniqueness of M', suppose that $M \equiv \eta(N)$, and consider

$$\begin{array}{rll}
N^\delta & \equiv \mathcal{C}(\lambda k.k(N^\delta)) & (\mathcal{C}\text{-App}) \\
 & \equiv \mathcal{C}((\eta(N))^\delta) \equiv \mathcal{C}(M^\delta) \equiv (M')^\delta. &
\end{array}$$

Theorem 45 ***(Completeness)*** *$\mathcal{C}$-theories are complete for CPS transforms into response theories with zero that satisfy the equalizer requirement.*

7 Abstract Kleisli-categories

An interpretation $[\![-]\!]$ of a λ_C-expression $\Gamma \vdash M : A$ in a response category (given syntactically as a CPS transform) yields a morphism $[\![\Gamma]\!] \longrightarrow R^{R^{[\![A]\!]}}$. More generally, in a λ_C-model with monad T, it yields a morphism $[\![\Gamma]\!] \longrightarrow T[\![A]\!]$. By contrast, in this section we shall define a notion of "direct" interpretation where $\Gamma \vdash M : A$ denotes a morphism $[\![\Gamma]\!] \longrightarrow [\![A]\!]$ in an "abstract Kleisli category".

We introduced control theories as special λ_C-theories; similarly, we shall first introduce direct models of the λ_C-calculus (the categories in this section, the semantics in the next) and specialize them to continuations models later—our "$\mathcal{C}$-categories" (Section 9), of which Selinger's "co-control categories" [19] are a special case.

We shall introduce the direct models in three steps: (1) "abstract Kleisli-categories", which correspond to monads, (2) "precartesian abstract Kleisli-categories", which correspond to strong monads on categories with finite products, and (3) "precartesian-closed abstract Kleisli-categories", which correspond to strong monads on categories with finite products and T-exponentials. Our emphasis is on structural theorems that explain what we mean by "correspond": Theorems 48, 56, and 62. These theorems are very strong indeed—for example, we shall obtain the fact that every $\mathcal{C}$-category arises from a continuations monad (similar to Theorem 2.18 in [19]) as a corollary of Theorem 62

Remark 46 *This section goes slightly beyond [33]. In particular, morphisms between monads and morphisms between abstract Kleisli-categories have been generalized because of practical needs as well as mathematical taste.*

7.1 Abstract Kleisli-categories

Definition 47 *An* abstract Kleisli-category *is a category* $\mathbf{K}$ *together with a functor* $L : \mathbf{K} \to \mathbf{K}$*, a transformation*[6] $\vartheta_A : A \longrightarrow LA$ *(called* thunk*), and a natural transformation* $LA \xrightarrow{\varepsilon_A} A$ *(called* force*) such that* $\vartheta_L : L \to L^2$ *is a*

[6] By "transformation" from a functor F to a functor G we mean an object-indexed family of arrows $\varphi_A : FA \longrightarrow GA$ (not necessarily natural).

natural transformation, and the following diagrams commute.

$$\begin{tikzcd}
A \arrow[r, "\vartheta"] \arrow[d, "\vartheta"'] & LA \arrow[d, "L\vartheta"] \\
LA \arrow[r, "\vartheta_L"] & L^2A
\end{tikzcd}
\qquad
\begin{tikzcd}
A \arrow[r, "\vartheta"] \arrow[dr, "id"'] & LA \arrow[d, "\varepsilon"] \\
 & A
\end{tikzcd}
\qquad
\begin{tikzcd}
LA \arrow[r, "\vartheta_L"] \arrow[dr, "id"'] & L^2A \arrow[d, "L\varepsilon"] \\
 & LA
\end{tikzcd}$$

Given any category $\mathbf{C}$ with a monad T, the Kleisli-category $\mathbf{C}_T$ forms an abstract Kleisli-category. The endofunctor $L : \mathbf{C}_T \to \mathbf{C}_T$ is obtained as the composite $\mathbf{C}_T \xrightarrow{G_T} \mathbf{C} \xrightarrow{F_T} \mathbf{C}_T$ around the adjunction determined by the monad. Thus on objects we have $LA = TA$. The map ϑ_A in $\mathbf{C}_T$ is $F_T\eta_A$—that is, $A \xrightarrow{\eta\circ\eta} T^2A$ in $\mathbf{C}$. The map ε_A in $\mathbf{C}_T$ is just the counit of the adjunction, which is explicitly given by the identity $TA \xrightarrow{id} TA$ in $\mathbf{C}$.

A morphism $A \xrightarrow{f} B$ in an abstract Kleisli-category $\mathbf{K}$ is called *thunkable* if the diagram below commutes.

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "\vartheta"'] & B \arrow[d, "\vartheta"] \\
LA \arrow[r, "Lf"] & LB
\end{tikzcd}$$

We write $\mathbf{K}_\vartheta$ for the subcategory of $\mathbf{K}$ given by the thunkable maps. Because ϑ_L is a natural transformation, all morphisms in the image of L are in $\mathbf{K}_\vartheta$. The functor $L : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$ is right adjoint to the inclusion $J : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}$ with unit ϑ and counit ε. We write

$$[-] : \mathbf{K}(A, B) \cong \mathbf{K}_\vartheta(A, LB)$$

for the adjunction isomorphism.

Given abstract Kleisli-categories $\mathbf{K}$ and $\mathbf{K}'$, a *morphism of abstract Kleisli-categories* from $\mathbf{K}$ to $\mathbf{K}'$ is defined to be a functor $V : \mathbf{K} \to \mathbf{K}'$ that preserves thunkable morphisms. Let $\mathbf{AKl}$ be the resulting category. We call V *tight* if for every object A, the morphism $[V\varepsilon_A] : VLA \longrightarrow LVA$ is an iso. (Recall that we also defined tightness for monad morphisms in the preliminaries.) The following theorem, which is a strengthened version of Theorem 5.3 in [33], is crucial for the proof of our main representation theorem (Theorem 84), and also useful for understanding sobriety (by being the basis of Proposition 53). We write $\mathbf{Mnd}$ for the category whose objects are categories with monads $(\mathbf{C}, T, \eta, \mu)$ and whose morphisms are monad morphisms.

Theorem 48 *The construction of the Kleisli-category forms a functor*

Mnd $\longrightarrow$ **AKl** *with a full and faithful right adjoint. Moreover, both adjoint functors preserve tight morphisms, and the components of the unit and counit are tight.*

So **AKl** is a reflective subcategory of **Mnd**, and the same relationship exists between the two subcategories of tight morphisms.

The rest of this section constitutes the proof of Theorem 48. We shall introduce an auxiliary category **Adj**, prove that there is an (adjoint) equivalence between **Mnd** and **Adj** (Lemma 51), and give a functor **Adj** $\longrightarrow$ **AKl** with a full and faithful right adjoint (Lemma 52). Theorem 48 then follows immediately be composing the two reflections. The objects of **Adj** are adjunctions $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ such that $\mathbf{C}$ and $\mathbf{K}$ have the same objects, and the left adjoint $F : \mathbf{C} \longrightarrow \mathbf{K}$ is the identity on objects. The morphisms in **Adj** from $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ to $F' \dashv G' : \mathbf{K}' \longrightarrow \mathbf{C}'$ are pairs of functors $(U : \mathbf{C} \longrightarrow \mathbf{C}', V : \mathbf{K} \longrightarrow \mathbf{K}')$ such that the diagram below commutes.

$$\begin{tikzcd}
\mathbf{K} \arrow[r, "V"] & \mathbf{K}' \\
\mathbf{C} \arrow[u, "F"] \arrow[r, "U"] & \mathbf{C}' \arrow[u, "F'"']
\end{tikzcd} \qquad (12)$$

Importantly, we do not require the square involving G and G' to commute—that is, we do not require $UG = G'V$. However, there is a natural transformation $UGA \longrightarrow G'VA$, which we call τ_A, given by the adjoint mate of $V\varepsilon_A$. We call $(U : \mathbf{C} \longrightarrow \mathbf{C}', V : \mathbf{K} \longrightarrow \mathbf{K}')$ *tight* if τ_A is an isomorphism for every object A. The morphism $[V\varepsilon_A]$ in the definition of tight morphisms of abstract Kleisli-categories coincides with τ_A, so the two notions of tightness agree.

Remark 49 *The notion of tightness is independent of the choice of right adjoints. That is, if (U, V) is a tight morphism in* **Adj** *from $F \dashv G$ to $F' \dashv G'$, and H (resp. H') is another right adjoint of F (resp. F'), then (U, V) is also a tight morphism from $F \dashv H$ to $F' \dashv H'$. This is so because the natural transformation $UH \longrightarrow H'V$ is the same as the natural isomorphism $UG \longrightarrow G'V$ up to the isomorphisms between G and H (resp. G' and H').*

Lemma 50 *Let $F \dashv G$ and $F' \dashv G'$ be objects of* **Adj**. *Let T and T' be the induced monads, and let $U : T \longrightarrow T'$ be a functor. Then to give a natural transformation $\sigma : UT \longrightarrow T'U$ that makes U into a monad morphism is to give a functor $V : \mathbf{K} \longrightarrow \mathbf{K}'$ that makes Diagram 12 commute. Moreover, (U, σ) is tight if and only (U, V) is tight.*

PROOF. Given σ, define $VA = UA$ for every object A, and define Vf by sending $f \in \mathbf{K}(A,B)$ through the map $\mathbf{K}(A,B) = \mathbf{K}(FA,B) \cong \mathbf{C}(A,GB) = \mathbf{C}(A,TB) \xrightarrow{U} \mathbf{C}(UA,UTB) \xrightarrow{\mathbf{C}(UA,\sigma_B)} \mathbf{C}(UA,T'UB) = \mathbf{C}(UA,G'UB) \cong \mathbf{K}'(F'UA,UB) = \mathbf{K}'(UA,UB)$. Conversely, given V, define $\sigma_A = \tau_{FA}$. It follows from routine calculations that these two constructions are mutually inverse. The two notions of tightness correspond because τ agrees with σ.

Lemma 51 *The construction of the Kleisli category forms an (adjoint) equivalence between* **Mnd** *and* **Adj**. *Moreover, both adjoint functors preserve tight morphisms, and the components of the unit and counit isos are tight.*

PROOF. The required functor $\mathbf{Mnd} \longrightarrow \mathbf{Adj}$ sends T to $F_T \dashv G_T$, and (U,σ) to (U,V) according to Lemma 50. The required functor $\mathbf{Adj} \longrightarrow \mathbf{Mnd}$ sends $F \dashv G$ to the induced monad, and (U,V) to (U,σ) according to Lemma 50. By the same lemma, the composed functor $\mathbf{Mnd} \longrightarrow \mathbf{Adj} \longrightarrow \mathbf{Mnd}$ is the identity. The functor $\mathbf{Adj} \longrightarrow \mathbf{Mnd} \longrightarrow \mathbf{Adj}$ sends $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ to $F_T \dashv G_T : \mathbf{C}_T \longrightarrow \mathbf{C}$, where T is the monad induced by $F \dashv G$. As for any adjunction, there is a unique comparison functor $H : \mathbf{C}_T \longrightarrow \mathbf{K}$ that mediates between the two adjunctions. In particular, $(Id_{\mathbf{C}}, H)$ is a morphism in **Adj** from $F_T \dashv G_T$ to $F \dashv G$. Because F is the identity on objects, H has an inverse (which sends $f \in \mathbf{K}(A,B) = \mathbf{K}(FA,B)$ through the map $\mathbf{K}(FA,B) \cong \mathbf{C}(A,GB) = \mathbf{C}(A,TB) = \mathbf{C}_T(A,B)$). It follows from a routine calculation that the construction of $(Id_{\mathbf{C}}, H)$ is natural in $F \dashv G$. So the functor $\mathbf{Adj} \longrightarrow \mathbf{Mnd} \longrightarrow \mathbf{Adj}$ is naturally isomorphic to the identity functor. Checking the tightness of $(Id_{\mathbf{C}}, H)$ is straightforward.

Lemma 52 **AKl** *is a full reflective subcategory of* **Adj**. *Moreover, both adjoint functors preserve tight morphisms, and the components of the unit and counit are tight.*

PROOF. The right adjoint $\mathbf{AKl} \longrightarrow \mathbf{Adj}$ sends $\mathbf{K}$ to $J \dashv L : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$ (where J is the inclusion), and it sends $H : \mathbf{K} \longrightarrow \mathbf{K}'$ to $(H : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}'_\vartheta, H : \mathbf{K} \longrightarrow \mathbf{K}')$. The left adjoint $\mathbf{Adj} \longrightarrow \mathbf{AKl}$ sends an adjunction $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ to $\mathbf{K}$ with $L = FG$, $\vartheta_A = F\eta_A$, and the force map given by the counit of the adjunction. A morphism (U,V) from $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ to $F' \dashv G' : \mathbf{K}' \longrightarrow \mathbf{C}'$ is sent to $V : \mathbf{K} \longrightarrow \mathbf{K}'$. To see that V preserves thunkable morphisms, let $f \in \mathbf{K}_\vartheta(A, B)$ and consider the two diagrams below.

$$\begin{tikzcd}
VA \arrow[r, "VF\eta"] \arrow[d, "Vf"'] & VFGA \arrow[d, "VFGf"] \\
VB \arrow[r, "VF\eta"'] & VFGB
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & F'G'VB \arrow[ddd, "F'G'Vf"] \\
F'UA \arrow[r, "F'U\eta"'] \arrow[d, "Vf"'] \arrow[urr, "F'\eta'"] & F'UGA \arrow[d, "F'UGf"] \arrow[ur, "F'\tau"] & \\
F'UB \arrow[r, "FU\eta"] \arrow[drr, "F'\eta'"'] & F'UGB \arrow[dr, "F'\tau"] & \\
 & & F'G'VB
\end{tikzcd}$$

The left diagram commutes because it is the thunkability square for f sent through V. The left square in the right diagram is the same as the left diagram because $VF = F'U$. The right square in the right diagram commutes due to the naturality of τ. The two triangles commute because $\tau_A = \sigma_A$ for all A by Lemma 50, and because, by the definition of monad morphism, U preserves η up to σ. So the outer square of the right diagram commutes, and it states that Vf is thunkable.

To see that the morphism V in **AKl** is tight if the morphism (U, V) in **Adj** is tight, note that the map $[V\varepsilon_A]$ required to be an iso for the former is the image under F' of the map required to be an iso for the latter.

Obviously, the composition $\mathbf{AKl} \longrightarrow \mathbf{Adj} \longrightarrow \mathbf{AKl}$ is the identity. The counit of the required reflection is simply the identity natural transformation on that identity functor. The unit, which has to mediate between $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ and $J \dashv L : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$, consists of the identity functor on $\mathbf{K}$ and the map $\mathbf{C} \longrightarrow \mathbf{K}_\vartheta$ that arises as the co-restriction of F. Checking the given data form an adjunction is now easy.

Proof of Theorem 48 By composing Lemmas 51 and 52.

The following result (Theorem 5.23 from [33], which we shall not use in this article) provides additional justification for abstract Kleisli-categories by establishing a link with the important notion of sobriety (which we discussed in Section 6.3).

Proposition 53 *A monad is in the image of the full and faithful functor* $\mathbf{AKl} \longrightarrow \mathbf{Mnd}$ *if and only if it satisfies the equalizer requirement.*

In this section, we introduce extra structure on abstract Kleisli-categories which corresponds to upgrading from monads to strong monads on categories with finite products.

Definition 54 *A* precartesian abstract Kleisli-category $\mathbf{K}$ *is an abstract Kleisli-category together with finite products on* $\mathbf{K}_\vartheta$ *and a symmetric premonoidal structure on* $\mathbf{K}$ *such that the inclusion* $J : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}$ *is a strict symmetric premonoidal functor.*

In other words, a precartesian abstract Kleisli-category is an abstract Kleisli-category with the extra structure of a Freyd category.

Let $\delta_A : A \longrightarrow A \otimes A$ be the diagonal associated with the finite products on $\mathbf{K}_\vartheta$, let π_1 and π_2 be the projections, let $!_A : A \longrightarrow I$ be the unique thunkable map, and let $\langle f, g\rangle$ stand for $(id \otimes g) \circ (f \otimes id) \circ \delta$. The next proposition, taken from [33], is an equational characterization of precartesian abstract Kleisli-categories[7]. A morphism $f : A \longrightarrow B$ is called *copyable* if $\delta_B \circ f = (B \otimes f) \circ (f \otimes A) \circ \delta_A$, and *discardable* if $!_B \circ f = !_A$.

Proposition 55 *Let* $\mathbf{K}$ *be an abstract Kleisli-category together with a binoidal structure* $\otimes$*, an object* I*, and transformations* $\delta_A : A \to A \otimes A$, $\pi_i : A_1 \otimes A_2 \to A_i$*, and* $!_A : A \to I$*. Then* $\mathbf{K}$ *with these data forms a precartesian abstract Kleisli-category if and only if*

(1) All morphisms of the form $[f]$ *are central, copyable, and discardable.*
(2) All components of δ, π_i*, and* $!$*, as well as all morphisms of the form* $A \otimes [f]$ *and* $[f] \otimes A$*, are thunkable.*
(3) The transformations below are natural in each argument (w.r.t. all *morphisms in* $\mathbf{K}$*).*

$$\begin{aligned} \pi_1 &: A \otimes I \longrightarrow A \\ \langle \pi_1 \circ \pi_1, \langle \pi_2 \circ \pi_1, \pi_2\rangle\rangle &: (A \otimes B) \otimes C \longrightarrow A \otimes (B \otimes C) \\ \pi_2 &: I \otimes A \longrightarrow A \\ \langle \pi_2, \pi_1\rangle &: A \otimes B \longrightarrow B \otimes A \end{aligned}$$

[7] The condition $! = id$ was overlooked in [33].

(4) The following equations hold whenever they "type-check".

$$\begin{aligned}(\pi_1 \otimes \pi_2) \circ \delta &= id \\ \pi_i \circ \delta &= id \\ \pi_1 \circ (id \otimes !) &= \pi_1 \\ \pi_2 \circ (! \otimes id) &= \pi_2 \\ ! &= id\end{aligned}$$

A *morphism of Freyd categories* (U, V) from $F : \mathbf{C} \longrightarrow \mathbf{K}$ to $F' \dashv G' : \mathbf{C}' \longrightarrow \mathbf{K}'$ consists of a functor $U : \mathbf{C} \longrightarrow \mathbf{C}'$ that preserves finite products and a functor $V : \mathbf{K} \longrightarrow \mathbf{K}'$ such that, letting $U_2(A, B) : U(A \times B) \longrightarrow UA \times UB$ be the evident natural isomorphism, the object-indexed family of maps $F'U_2(A, B) : V(A \otimes B) \longrightarrow VA \otimes VB$ is natural in A and B. A *Freyd adjunction* is defined to be a Freyd category $F : \mathbf{C} \longrightarrow \mathbf{K}$ together with a right adjoint G of F. A *morphism of Freyd adjunctions* is defined to be a morphism of Freyd categories that also is a morphism in **Adj**. We write $\mathbf{Adj}_\otimes$ for the resulting category. A *morphism of precartesian abstract Kleisli-categories* is simply a morphism of Freyd adjunctions. We write $\mathbf{AKl}_\otimes$ for the resulting category. Furthermore, we write $\mathbf{Mnd}_t$ for the category whose objects are strong monads on categories with finite products and whose morphisms are strong monad morphisms.

Theorem 56 *The reflection between* **Mnd** *and* **AKl** *forms a reflection between* $\mathbf{Mnd}_t$ *and* $\mathbf{AKl}_\otimes$.

As in the case of Theorem 48, we prove this theorem by composing two adjunctions, given by Lemmas 57 and 58.

Lemma 57 *The equivalence between* **Mnd** *and* **Adj** *forms an equivalence between* $\mathbf{Mnd}_t$ *and* $\mathbf{Adj}_\otimes$.

PROOF. Let T be a strong monad on a category $\mathbf{C}$ with finite products. For a morphism $f \in \mathbf{C}_T(A, B)$ and an object C, define $C \otimes f$ to be $C \times A \xrightarrow{C \times f} C \times TB \xrightarrow{t} T(C \times B)$, and $f \otimes C$ symmetrically. Then $\mathbf{C}_T$ together with $\otimes$ forms a symmetric premonoidal category, where the four required natural isomorphisms are given as the images under the left adjoint F_T of the evident maps definable from the finite products of $\mathbf{C}$, and furthermore F_T is a strict symmetric premonoidal functor. (This is the left-to-right part of Corollary 4.2 in [17].)

Conversely, let $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ be a Freyd adjunction, and let T be the induced monad on $\mathbf{C}$. Then the required strength $t_{A,B}$ is given as the adjoint mate of $A \otimes \varepsilon_B$, where ε is the counit of the adjunction.

Now consider the situation in Lemma 50 and assume that the two adjunctions there are Freyd adjunctions. Let (U,σ) be a morphism $T \longrightarrow T'$ in **Mnd**, and let (U,V) be the corresponding morphism from $F \dashv G$ to $F' \dashv G'$ of **Adj**. We need to show that (U,σ) is in $\mathbf{Mnd}_t$ if and only if (U,V) is in $\mathbf{Adj}_\otimes$. This is so because Equation 3 corresponds to the naturality of $F'U_2(A,B) : V(A \otimes B) \longrightarrow (VA) \otimes (VB)$.

Now for the unit and counit isos. The iso $(Id_{\mathbf{C}}, H)$ of **Adj** from the proof of Lemma 51 is easily shown to be a morphism in $\mathbf{Adj}_\otimes$. The identity functor $\mathbf{Mnd} \longrightarrow \mathbf{Adj} \longrightarrow \mathbf{Mnd}$ from Lemma 51 also forms the identity functor $\mathbf{Mnd}_t \longrightarrow \mathbf{Mnd}_t$, because it does not change the strength, and the components of the natural identity on that identity functor are trivially morphisms in $\mathbf{Mnd}_t$.

Lemma 58 *The reflection between* **Adj** *and* **AKl** *forms a reflection between* $\mathbf{Adj}_\otimes$ *and* $\mathbf{AKl}_\otimes$*.*

PROOF. That the functor $\mathbf{AKl} \longrightarrow \mathbf{Adj}$ forms a functor $\mathbf{AKl}_\otimes \longrightarrow \mathbf{Adj}_\otimes$ is trivial. For the opposite direction, let $F \dashv G : \mathbf{K} \to \mathbf{C}$ be a Freyd adjunction. To see that the abstract Kleisli-category $J \dashv L : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$ is a precartesian one, we use Proposition 55. The required transformations δ, π_i, and $!$ are the images under F of the evident maps of **C**. Note that for every morphism f in **K**, we have $[f] = F(f^\sharp)$, where $f^\sharp$ is the adjoint mate of f in **C**. Condition 1 holds because every morphism of the form Fg is central (since F is a symmetric premonoidal functor) and copyable and discardable (which follows from sending the equations $! \circ g =!$ and $\langle id, id\rangle \circ g = (g \times g) \circ \langle id, id\rangle$ through F). For Condition 2, note that all maps listed there are in the image of F (in particular, $A \otimes [f] = A \otimes F(f^\sharp) = F(A \times f^\sharp)$, and therefore thunkable. Condition 3 holds because the maps required to be natural coincide with the natural isos that belong to the premonoidal structure of **K**. The equations in Condition 4 follow from sending the corresponding equations in **C** through F.

For the morphism part, let (U,V) be a morphism of Freyd adjunctions from $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ to $F' \dashv G' : \mathbf{K}' \longrightarrow \mathbf{C}'$. To see that the restriction of V to $\mathbf{K}_\vartheta \longrightarrow \mathbf{K}'_\vartheta$ preserves finite products, send the product cone $A_1 \xleftarrow{F\pi_1} A_1 \otimes A_2 \xrightarrow{F\pi_2} A_2$ of $\mathbf{K}_\vartheta$ through V. The resulting cone is $VA_1 \xleftarrow{FU\pi_1} V(A_1 \otimes A_2) \xrightarrow{FU\pi_2} VA_2$, which is isomorphic to the product cone $VA_1 \xleftarrow{F\pi_1} (VA_1) \otimes (VA_2) \xrightarrow{F\pi_2} VA_2$ of $\mathbf{K}'_\vartheta$, where the mediating isomorphism is $F'U_2 : V(A_1 \otimes A_2) \longrightarrow VA_1 \otimes VA_2$.

The unit of the reflection between **Adj** and **AKl** arose from the co-restriction $\mathbf{C} \longrightarrow \mathbf{K}_\vartheta$ of F; it strictly preserves finite products by definition of the finite products on $\mathbf{K}_\vartheta$. The functor $\mathbf{AKl} \longrightarrow \mathbf{Adj} \longrightarrow \mathbf{AKl}$, which we know to be the identity from the proof of Lemma 52, also forms the identity

$\mathbf{AKl}_\otimes \longrightarrow \mathbf{AKl}_\otimes$. Again, the counit is the natural identity on that identity functor, and its components are trivially morphisms in $\mathbf{AKl}_\otimes$.

7.3 Closed structure

In this section, we study the property of abstract Kleisli-categories that corresponds to T-exponentials.

Definition 59 *A* precartesian-closed abstract Kleisli-category[8] $\mathbf{K}$ *is a precartesian abstract Kleisli-category together with a right adjoint* $A \rightharpoonup (-)$ *to the functor* $(-) \otimes A : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}$ *for every object* A*.*

So a precartesian-closed abstract Kleisli-category is a special case of a closed Freyd-category. For morphisms $f : A \longrightarrow A'$ and $g : B \longrightarrow B'$ of a precartesian-closed abstract Kleisli-category, define $f \rightharpoonup g$ to be the map $(A' \rightharpoonup B) \longrightarrow (A \rightharpoonup B')$ that arises as the adjoint mate of $(A' \rightharpoonup B) \otimes A \xrightarrow{id \otimes f} (A' \rightharpoonup B) \otimes A' \xrightarrow{apply} B \xrightarrow{g} B'$. Unlike $\otimes$, the operation $\rightharpoonup$ is a bifunctor, of type $\mathbf{K}^{op} \times \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$. The functor $I \rightharpoonup (-) : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$ is right adjoint to $(-) \otimes I : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}$, and therefore also to the inclusion functor $\mathbf{K}_\vartheta \longrightarrow \mathbf{K}$. Because $L : \mathbf{K}_\vartheta \longrightarrow \mathbf{K}$ is also right adjoint to the inclusion functor, L and $I \rightharpoonup (-)$ are naturally isomorphic. We write ι_A for the isomorphism $LA \cong I \rightharpoonup A$. In fact, ι_A is the adjoint mate of $LA \otimes I \cong LA \xrightarrow{\varepsilon} A$.

Proposition 60 *Let* $\mathbf{K}$ *be a precartesian abstract Kleisli-category together with, for all objects* A *and* B*, an object* $A \rightharpoonup B$*, and transformations* $\Lambda : \mathbf{K}(A \otimes B, C) \longrightarrow \mathbf{K}_\vartheta(A, B \rightharpoonup C)$ *and* $apply : (A \rightharpoonup B) \otimes A \longrightarrow B$*. Then* $\mathbf{K}$ *with these data is a precartesian-closed abstract Kleisli-category if and only if*

$$apply \circ (\Lambda f \otimes A) = f \qquad \Lambda apply = id \qquad (\Lambda g) \circ [f] = \Lambda(g \circ ([f] \otimes id)).$$

We write $\mathbf{Adj}_\otimes^\rightharpoonup$ for the category whose objects are closed Freyd-categories and whose morphisms are morphisms of Freyd categories, and $\mathbf{AKl}_\otimes^\rightharpoonup$ for the full subcategory of $\mathbf{Adj}_\otimes^\rightharpoonup$ whose objects are precartesian-closed abstract Kleisli-categories. We do not require morphisms to preserve exponentials, because there exist important counterexamples (see Remark 67). Instead, we address the issue with an extra condition: a morphism (U, V) from $F \dashv G : \mathbf{C} \longrightarrow \mathbf{K}$ to $F' \dashv G' : \mathbf{K}' \longrightarrow \mathbf{C}'$ in $\mathbf{Adj}_\otimes^\rightharpoonup$ is called *closed* if, for all objects A and B, the map in $\mathbf{C}'(U(A \rightharpoonup B), VA \rightharpoonup VB)$ that arises as the adjoint mate of $V(A \rightharpoonup B) \otimes VA \cong V((A \rightharpoonup B) \otimes A) \xrightarrow{V apply} VB$ is an isomorphism.

[8] called "direct λ_C-models" in [33] and "computational abstract Kleisli-categories" in [20].

Remark 61 *Closed morphisms are tight: if (U, V) is closed then it is tight because the natural transformation in the inner square below is an isomorphism. (Recall that by Remark 49, tightness does not depend on the choice of right adjoint.)*

$$\begin{tikzcd}
\mathbf{K} \arrow[rr, "V"] \arrow[d, shift left=2, "I \rightharpoonup (-)"] \arrow[d, phantom, "\dashv"] & \arrow[d, phantom, "\Rightarrow"] & \mathbf{K}' \arrow[d, shift right=2, "VI \rightharpoonup (-)"'] \arrow[d, phantom, "\vdash"] \\
\mathbf{C} \arrow[u, shift left=2, "F"] \arrow[rr, "U"'] & & \mathbf{C}' \arrow[u, shift right=2, "F'"']
\end{tikzcd}$$

We write $\mathbf{Mnd}_t^T$ for the category whose objects are strong monads on categories with finite products and T-exponentials and whose morphisms are strong monad morphisms.

Theorem 62 *The reflection between $\mathbf{Mnd}_t$ and $\mathbf{AKl}_\otimes$ forms a reflection between $\mathbf{Mnd}_t^T$ and $\mathbf{AKl}_\otimes^\rightharpoonup$. Moreover, both adjoint functors preserve closed morphisms, and the unit and counit are closed.*

PROOF. By composing Lemmas 65 and 66.

Lemma 63 *Let $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ be a Freyd adjunction, and let T be the induced strong monad. Then to give a right adjoint to $F(-) \otimes A : \mathbf{C} \longrightarrow \mathbf{K}$ for all objects A is to give T-exponentials to $\mathbf{C}$.*

PROOF. $A \rightharpoonup B$ corresponds to $(TB)^A$, and $apply_B^A \in \mathbf{K}((A \rightharpoonup B) \otimes A, B)$ corresponds to $ev_B^A \in \mathbf{C}((TB)^A \times A, TB)$: the two classes of maps are adjoint mates, and the universal property of one easily translates into the universal property of the other.

Lemma 64 *Consider the situation in Lemma 50, where $F \dashv G$ and $F' \dashv G'$ are closed Freyd-categories an T and T' are the induced strong monads. Then (U, σ) is closed if and only if (U, V) is closed.*

PROOF. The map $\mathbf{C}'(U(A \rightharpoonup B), (VA) \rightharpoonup (VB))$ from the definition of closed morphism of closed Freyd-categories coincides with the map $\mathbf{C}'(U((TB)^A), (T'UB)^{UA}$ from the definition of closed morphism of strong monads. So one is an isomorphism if the other one is.

Lemma 65 *The (adjoint) equivalence between $\mathbf{Mnd}_t$ and $\mathbf{Adj}_\otimes$ forms an equivalence between $\mathbf{Mnd}_t^T$ and $\mathbf{Adj}_\otimes^\rightharpoonup$. Moreover, both adjoint functors preserve closed morphisms, and the unit and counit isos are closed.*

PROOF. The object parts of the functors are addressed by Lemma 63. For the morphism parts, we only have to show that both functors preserve closed morphisms. This is immediate from Lemma 64. Checking that the components of the unit and counit isos are closed is straightforward.

Lemma 66 *The reflection between* $\mathbf{Adj}_\otimes$ *and* $\mathbf{AKl}_\otimes$ *forms a reflection between* $\mathbf{Adj}_\otimes^{\rightharpoonup}$ *and* $\mathbf{AKl}_\otimes^{\rightharpoonup}$*. Moreover, both adjoint functors preserve closed morphisms, and the unit and counit isos are closed.*

PROOF. That the functor $\mathbf{AKl}_\otimes \longrightarrow \mathbf{Adj}_\otimes$ forms a functor $\mathbf{AKl}_\otimes^{\rightharpoonup} \longrightarrow \mathbf{Adj}_\otimes^{\rightharpoonup}$ is trivial. For the converse, let $F \dashv G : \mathbf{K} \longrightarrow \mathbf{C}$ be a closed Freyd-category with universal map $apply_B^A \in \mathbf{K}((A \rightharpoonup B) \otimes A, B)$. Then the precartesian abstract Kleisli-category $J \dashv L : \mathbf{K} \longrightarrow \mathbf{K}_\vartheta$ together with $\rightharpoonup$ and *apply* is also a closed Freyd-category, with the adjunction isomorphism $\mathbf{K}(A \otimes B, C) \cong \mathbf{C}(A, B \rightharpoonup C) \xrightarrow{F} \mathbf{K}(A, B \rightharpoonup C)$. (For a calculation proving this fact, see Proposition 2.25 in [33].) To see that the functor $\mathbf{Adj}_\otimes \longrightarrow \mathbf{AKl}_\otimes$ preserves closed morphisms, note that the map in $\mathbf{K}'_{\vartheta'}(V(A \rightharpoonup B), (VA) \rightharpoonup (VB))$ required to be an isomorphism is the image under F' of the map in $\mathbf{C}'(U(A \rightharpoonup B), (VA) \rightharpoonup (VB))$. Checking the tightness of the unit and counit is straightforward.

Remark 67 *To see that one should not require morphisms of* $\mathbf{AKl}_\otimes^{\rightharpoonup}$ *or morphisms of* $\mathbf{Mnd}_t^T$ *to be closed, consider the following example. Let* $\mathbf{C}$ *be a cartesian-closed category, and let* T *be a strong monad on* $\mathbf{C}$*. Let* O *be any object of* $\mathbf{C}$*, and let* T' *be the continuations monad with* $R = TO$*, that is,* $T'O = (TO)^{(TO)^A}$*. Then the identity functor on* $\mathbf{C}$ *turns out to form a strong monad morphism from* $(\mathbf{C}, T)$ *to* $(\mathbf{C}, T')$*, where the required natural transformation* $\sigma_A : TA \longrightarrow T'A = (TO)^{(TO)^A}$ *is the adjoint mate of*

$$TA \times (TO)^A \cong (TO)^A \times TA \xrightarrow{t} T((TO)^A \times A) \xrightarrow{Tev} TTO \xrightarrow{\mu} TO.$$

Thus, any monad can be embedded into a continuations monad by a strong monad morphism. However, this strong monad morphism is not generally closed. For if it where closed, then the map $(TB)^A \longrightarrow (T'B)^A = ((TO)^{(TO)^B})^A$ *which arises as the adjoint mate of*

$$(TB)^A \times A \xrightarrow{ev} TB \xrightarrow{\sigma} T'B$$

would have to be an isomorphism for all A *and* B*. But this is not generally true. For example, if* T *is the identity monad, then the map, which has now type* $B^A \longrightarrow (O^{O^B})^A$*, turns out to be the one given by* $\lambda f : B^A.\lambda x : A.\lambda g : O^B.g(fx)$*, which is not generally an isomorphism.*

8 Direct models of the λ_C-calculus

In this section, we shall discuss the interpretation of λ_C-languages in precartesian-closed abstract Kleisli-categories. In particular, we show how to construct the initial model of any λ_C-theory, and derive a completeness result; conversely, we show how to construct the internal language of a precartesian-closed abstract Kleisli-category. As a vital tool, we shall introduce a notion of "reverse interpretation" describing how to express categorical structure in the λ_C-calculus.

A *direct interpretation* of a λ_C-language $\mathcal{L}$ is a precartesian-closed abstract Kleisli-category $\mathbf{K}$ together with a map $\mathbf{K}[\![-]\!]$ (short, $[\![-]\!]$) sending types of $\mathcal{L}$ to objects of $\mathbf{K}$ such that

$$[\![A \times B]\!] = [\![A]\!] \otimes [\![B]\!] \qquad [\![1]\!] = I \qquad [\![A \to B]\!] = [\![A]\!] \rightharpoonup [\![B]\!]$$

and expressions of $\mathcal{L}$ to morphisms in $\mathbf{K}$ according to the rules in Figure 4, where $[\![\Gamma]\!]$ is defined as $[\![(\cdots(A_1 \times A_2) \times \cdots \times A_n)]\!]$ whenever $\Gamma = x_1 : A_1, \ldots, x_n : A_n$. (Thus, a direct interpretation is uniquely determined by its behavior on base types and constants.) A *direct model* of a λ_C-theory $\mathcal{T}$ is a direct interpretation of $|\mathcal{T}|$ that validates all equations of $\mathcal{T}$. We shall omit the word "direct" when it is clear that the range of an interpretation is a precartesian-closed abstract Kleisli-category.

Proposition 68 (Soundness) *For every direct interpretation* $\mathbf{K}[\![-]\!]$ *of a* λ_C*-language* $\mathcal{L}$*, the well-typed equations* $\Gamma \vdash M \equiv N : A$ *over* $\mathcal{L}$ *such that* $\mathbf{K}[\![\Gamma \vdash M : A]\!] = \mathbf{K}[\![\Gamma \vdash N : A]\!]$ *form a* λ_C*-theory.*

Now we turn towards defining the a notion of "reverse interpretation". First, some auxiliary definitions. The *categorical types* over a collection of base types b are defined by

$$A, B ::= LA \mid A \otimes B \mid I \mid A \rightharpoonup B \mid b.$$

The *categorical expressions* over collections of base types b and operators $h : A \longrightarrow B$ are given by

$$\begin{aligned} f, g ::= {} & id_A \mid g \circ f \mid A \otimes f \mid f \otimes A \mid \delta_A \mid \pi_1^{A_1,A_2} \mid \pi_2^{A_1,A_2} \\ & \mid !_A \mid [f] \mid \varepsilon_A \mid \Lambda f \mid apply_{A,B} \mid h. \end{aligned}$$

where the evident typability is required. A *categorical language* is defined to be the collection of categorical expressions over some base types and operators.

A *reverse interpretation* from a categorical language $\mathcal{L}_{cat}$ into a λ_C-language

$$[\![x_1 : A_1, \ldots, x_n : A_n \vdash x_i : A_i]\!] = (\cdots([\![A_1]\!] \otimes [\![A_2]\!]) \cdots \otimes [\![A_n]\!]) \xrightarrow{\pi_i} [\![A_i]\!]$$

$$[\![\Gamma \vdash c : A]\!] = [\![\Gamma]\!] \xrightarrow{!} I \xrightarrow{[\![c]\!]} [\![A]\!] \qquad ([\![c]\!] \text{ thunkable})$$

$$\frac{[\![\Gamma, x : A \vdash M : B]\!] = [\![\Gamma]\!] \otimes [\![A]\!] \xrightarrow{f} [\![B]\!]}{[\![\Gamma \vdash \lambda x : A.M : A \to B]\!] = [\![\Gamma]\!] \xrightarrow{\Lambda f} ([\![A]\!] \rightharpoonup [\![B]\!])}$$

$$\frac{[\![\Gamma \vdash M : A \to B]\!] = [\![\Gamma]\!] \xrightarrow{f} ([\![A]\!] \rightharpoonup [\![B]\!]) \qquad [\![\Gamma \vdash N : A]\!] = [\![\Gamma]\!] \xrightarrow{g} [\![A]\!]}{[\![\Gamma \vdash MN : B]\!] = [\![\Gamma]\!] \xrightarrow{\langle f,g \rangle} ([\![A]\!] \rightharpoonup [\![B]\!]) \otimes [\![A]\!] \xrightarrow{apply} [\![B]\!]}$$

$$\frac{[\![\Gamma \vdash M : A]\!] = [\![\Gamma]\!] \xrightarrow{f} [\![A]\!] \qquad [\![\Gamma \vdash N : B]\!] = [\![\Gamma]\!] \xrightarrow{g} [\![B]\!]}{[\![\Gamma \vdash (M, N) : A \times B]\!] = [\![\Gamma]\!] \xrightarrow{\langle f,g \rangle} [\![A]\!] \otimes [\![B]\!]}$$

$$\frac{[\![\Gamma \vdash M : A_1 \times A_2]\!] = [\![\Gamma]\!] \xrightarrow{f} [\![A_1]\!] \otimes [\![A_2]\!]}{[\![\Gamma \vdash \pi_i(M) : A_i]\!] = [\![\Gamma]\!] \xrightarrow{f} [\![A_1]\!] \otimes [\![A_2]\!] \xrightarrow{\pi_i} [\![A_i]\!]}$$

$$[\![\Gamma \vdash () : 1]\!] = [\![\Gamma]\!] \xrightarrow{!} I$$

Fig. 4. Semantics of the λ_C-calculus in precartesian-closed abstract Kleisli-categories

$\mathcal{L}$ is defined to be a map $(-)^r$ that sends types of $\mathcal{L}_{cat}$ to types of $\mathcal{L}$ such that

$$(A \otimes B)^r = A^r \times B^r \quad I^r = 1 \quad (A \rightharpoonup B)^r = A^r \to B^r \quad (LA)^r = 1 \rightharpoonup A^r.$$

and every expression $f : A \longrightarrow B$ of $\mathcal{L}_{cat}$ to an expression $(x_1 : A_1, \ldots, x_n : A_n \vdash M : B^r)$ of $\mathcal{L}$, where the sequence $A_1, \ldots, A_n$ is the factorization of A^r, following the rules in Figure 5. (Thus, a reverse interpretation is uniquely

$$\begin{array}{c|l}
(f : A \longrightarrow B)^r & = \Gamma \vdash M : B^r \\
(g : B \longrightarrow C)^r & = y_1 : B_1, \ldots, y_n : B_n \vdash N : C^r \\
\cline{1-1}
(g \circ f : A \longrightarrow C)^r & = \Gamma \vdash \mathtt{let}\, y_1, \ldots, y_n \,\mathtt{be}\, M \,\mathtt{in}\, N : C^r
\end{array}$$

$$(id : A \longrightarrow A)^r = x_1 : A_1 \ldots, x_n : A_n \vdash x_{A^r} : A^r$$

$$\begin{array}{c|l}
(f : A \longrightarrow B)^r & = \Gamma \vdash M : B^r \\
\cline{1-1}
(C \otimes f : C \otimes A \longrightarrow C \otimes B)^r & = y_1 : C_1, \ldots, y_n : C_n, \Gamma \vdash (y_{C^r}, M) : C^r \times B^r
\end{array}$$

$$\begin{array}{c|l}
(f : A \longrightarrow B)^r & = \Gamma \vdash M : B^r \\
\cline{1-1}
(f \otimes C : A \otimes C \longrightarrow B \otimes C)^r & = \Gamma, y_1 : C_1, \ldots, y_n : C_n \vdash (M, y_{B^r}) : B^r \times C^r
\end{array}$$

$$(\delta : A \longrightarrow A \otimes A)^r = x_1 : A_1 \ldots, x_n : A_n \vdash (x_{A^r}, x_{A^r}) : A^r \times A^r$$

$$(\pi_1 : A \otimes B \longrightarrow A)^r = x_1 : A_1, \ldots, x_n : A_n, y_1 : B_1, \ldots, y_n : B_m \vdash x_{A^r} : A^r$$

$$(\pi_2 : A \otimes B \longrightarrow A)^r = x_1 : A_1 \ldots, x_n : A_n, y_1 : B_1, \ldots, y_n : B_m \vdash y_{B^r} : B^r$$

$$(! : A \longrightarrow I)^r = x_1 : A_1, \ldots, x_n : A_n \vdash () : 1$$

$$\begin{array}{c|l}
(f : A \longrightarrow B)^r & = \Gamma \vdash M : B^r \\
\cline{1-1}
([f] : A \longrightarrow LB)^r & = \Gamma \vdash \lambda().M : 1 \to B^r
\end{array}$$

$$(\varepsilon : LA \longrightarrow A)^r = x : 1 \to A^r \vdash x() : A^r$$

$$\begin{array}{c|l}
(f : A \otimes B \longrightarrow C)^r & = \Gamma, y_1 : B_1, \ldots, y_n : B_m \vdash M : C^r \\
\cline{1-1}
(\Lambda f : A \longrightarrow (B \rightharpoonup C))^r & = \Gamma \vdash \lambda(y_1, \ldots, y_n) : B^r.M : B^r \to C^r
\end{array}$$

$$(\mathit{apply} : (A \rightharpoonup B) \otimes A \longrightarrow B)^r = f : A^r \to B^r, x_1 : A_1, \ldots, x_n : A_n \vdash f(x_{B^r}) : B^r$$

Fig. 5. Reverse interpretation of categorical expressions in the λ_C-calculus

determined by its behavior on base types b and operators h.)

Proposition 69 (Reverse soundness) *For every reverse interpretation $r : \mathcal{L}_{cat} \longrightarrow \mathcal{T}$ into a λ_C-theory $\mathcal{T}$, the resulting congruence on $\mathcal{L}_{cat}$ induces a precartesian-closed abstract Kleisli-category on $\mathcal{L}_{cat}$.*

PROOF. By checking the equations given by Propositions 55 and 60. (Because all equations must be sent through $(-)^r$, this amounts to mechanical (albeit laborious) calculations in the λ_C-calculus.)

Now we turn towards defining the term model of a λ_C-theory $\mathcal{T}$. Let $\mathcal{L}_{cat}(\mathcal{T})$ be the categorical language with the same base types as $\mathcal{T}$ and operators $h_c^A : I \longrightarrow A$ for every constant $c^{(A^r)}$ of $\mathcal{T}$. Define $r(\mathcal{T})$ to be the identity-on-base-types reverse interpretation that sends h_c^A to $(\vdash c^{(A^r)} : A^r)$. We write $\mathbf{K}_\mathcal{T}$ for the category induced on $\mathcal{L}_{cat}(\mathcal{T})$ by $r(\mathcal{T})$, and given an expression f of $\mathcal{L}_{cat}(\mathcal{T})$, we write $\mathbf{K}_\mathcal{T}\langle\!\lbrack f \rbrack\!\rangle$ for the equivalence class of f with respect to $r(\mathcal{T})$. Now define $\mathbf{K}_\mathcal{T}[\![-]\!] : |\mathcal{T}| \longrightarrow \mathbf{K}_\mathcal{T}$ to be the identity-on-base-types interpretation that sends c^A to $\mathbf{K}_\mathcal{T}\langle\!\lbrack h_c^{(\!\lbrack A \rbrack\!)} \rbrack\!\rangle$, where $(\!\lbrack A \rbrack\!)$ is the evident categorical type (without occurrences of L) corresponding to A. (Note that every $\mathbf{K}\langle\!\lbrack h_c^B \rbrack\!\rangle$ is thunkable, because $r(\mathcal{T})$ sends the equation $\vartheta \circ h_c^B = Lh_c^B \circ \vartheta$, which is equivalent to $[id] \circ h_c^B = [h_c^B]$, to $\vdash \mathtt{let}\, x\, \mathtt{be}\, c^{(B^r)}\, \mathtt{in}\, \lambda().x \equiv \lambda().c^{(B^r)} : B^r$). We shall write $\mathbf{K}_\mathcal{T}(\!\lbrack \Gamma \vdash M : A \rbrack\!)$ for the expression of $\mathcal{L}_{cat}(\mathcal{T})$ associated with $\Gamma \vdash M : A$. (So we have $\mathbf{K}_\mathcal{T}[\![\Gamma \vdash M : A]\!] = \mathbf{K}_\mathcal{T}\langle\!\lbrack \mathbf{K}_\mathcal{T}(\!\lbrack \Gamma \vdash M : A \rbrack\!) \rbrack\!\rangle$.)

Theorem 70 (Term model) *The theory on $|\mathcal{T}|$ induced by the interpretation $\mathbf{K}_\mathcal{T}[\![-]\!]$ is $\mathcal{T}$.*

PROOF. Let $\Gamma' \vdash M' : A'$ be $(\mathbf{K}_\mathcal{T}(\!\lbrack \Gamma \vdash M : A \rbrack\!))^{r(\mathcal{T})}$ and let $\Gamma' \vdash N' : A'$ be $(\mathbf{K}_\mathcal{T}(\!\lbrack \Gamma \vdash N : A \rbrack\!))^{r(\mathcal{T})}$. By definition of $\mathbf{K}_\mathcal{T}$, the equation $\Gamma \vdash M \equiv N : A$ holds in the theory induced by $\mathbf{K}_\mathcal{T}[\![-]\!]$ (i.e. $\mathbf{K}_\mathcal{T}[\![\Gamma \vdash M : A]\!] = \mathbf{K}_\mathcal{T}[\![\Gamma \vdash N : A]\!]$) if and only if $\Gamma' \vdash M' \equiv N' : A'$ holds in $\mathcal{T}$. As can be proved by induction on M, the expression $\Gamma' \vdash M' : A'$ is the same as $\Gamma \vdash M : A$ except for renaming variables in Γ and splitting variables of product types into multiple variables of non-product type. Therefore, $\Gamma' \vdash M' \equiv N' : A'$ holds in $\mathcal{T}$ if and only if $\Gamma \vdash M \equiv N : A$ holds in $\mathcal{T}$.

Corollary 71 (Completeness) *If an equation holds in all direct models of a λ_C-theory $\mathcal{T}$, it holds in $\mathcal{T}$.*

Before we proceed with Theorems 73 and 74, we need to address a technical issue, which arises because λ_C-languages have no types of the form LA. Consider the diagram below

$$\begin{tikzcd}
\mathbf{K} & \\
\mathcal{L}_{cat} \arrow[u, "\mathbf{K}\langle\!\lbrack - \rbrack\!\rangle"] \arrow[r, "r"'] & \mathcal{L} \arrow[ul, "\mathbf{K}[\![-]\!]"']
\end{tikzcd} \tag{13}$$

where $\mathcal{L}$ is a λ_C-language, $\mathbf{K}$ is a precartesian-closed abstract Kleisli-category, $\mathcal{L}_{cat}$ is a categorical language, $\mathbf{K}[\![-]\!]$ is an interpretation, r is a reverse interpretation, and $\mathbf{K}\langle\!\langle-\rangle\!\rangle$ is a categorical interpretation (in the evident sense). Suppose that for every base type b of $\mathcal{L}_{cat}$ it holds that $\mathbf{K}[\![b^r]\!] = \mathbf{K}\langle\!\langle b\rangle\!\rangle$.

Then, crucially, for *every* type A of $\mathcal{L}_{cat}$ there is a thunkable isomorphism $\iota^+_A : \mathbf{K}\langle\!\langle A\rangle\!\rangle \cong \mathbf{K}[\![A^r]\!]$, namely the recursive version of $\iota_A : LA \cong (I \rightharpoonup A)$ from Section 7.3:

$$\iota^+_b = id_{\mathbf{K}[\![b]\!]} \quad \iota^+_{A\otimes B} = \iota^+_A \otimes \iota^+_B \quad \iota^+_{A\rightharpoonup B} = (\iota^+_A)^{-1} \rightharpoonup \iota^+_B \quad \iota^+_{LA} = \iota_{\mathbf{K}[\![A^r]\!]} \circ L(\iota^+_A).$$

In other words, on types, Diagram 13 commutes up to ι^+. Next, we consider the situation for expressions, in the diagram below, where $A_1, \ldots, A_n$ is the factorization of A^r, and the isomorphism is built in the evident way from ι^+ and the associativity map for $\otimes$.

$$\begin{tikzcd}
\mathbf{K}(\langle\!\langle A\rangle\!\rangle, \langle\!\langle B\rangle\!\rangle) \arrow[r, phantom, "\cong"] & \mathbf{K}([\![A_1]\!] \otimes \cdots \otimes [\![A_n]\!], [\![B^r]\!]) \\
\mathcal{L}_{cat}(A, B) \arrow[u, "\mathbf{K}\langle\!\langle-\rangle\!\rangle"] \arrow[r, "r"'] & \mathcal{L}(A_1, \ldots, A_n \vdash B^r) \arrow[u, "\mathbf{K}[\![-]\!]"']
\end{tikzcd} \tag{14}$$

The following lemma plays a key rôle in the proofs of Theorems 73 and 74:

Lemma 72 *If Diagram 14 commutes for every* operator $h \in \mathcal{L}_{cat}(A, B)$*, then it already commutes for every expression* $f \in \mathcal{L}_{cat}(A, B)$*.*

PROOF. By induction on f.

Theorem 73 (Initiality) *For every direct model* $\mathbf{K}[\![-]\!]$ *of a* λ_C*-theory* $\mathcal{T}$*, there is a unique functor* $U : \mathbf{K}_\mathcal{T} \longrightarrow \mathbf{K}$ *that preserves the precartesian-closed abstract Kleisli-structure on the nose and satisfies the equation* $U \circ \mathbf{K}_\mathcal{T}[\![-]\!] = \mathbf{K}[\![-]\!]$*.*

PROOF. Diagram 15 sums up the definitions we shall make in this proof.

For uniqueness of U, let $U : K_\mathcal{T} \longrightarrow K$ be a functor as in the theorem. Then for each base type b it holds that $Ub = U(\mathbf{K}_\mathcal{T}[\![b]\!]) = \mathbf{K}[\![b]\!]$, and for each constant c^A it holds that $U(\mathbf{K}_\mathcal{T}\langle\!\langle h_c^{\langle\!\langle A\rangle\!\rangle}\rangle\!\rangle) = U(\mathbf{K}_\mathcal{T}[\![\vdash c : A]\!]) = \mathbf{K}[\![\vdash c : A]\!]$. Therefore, letting $\mathbf{K}\langle\!\langle-\rangle\!\rangle$ stand for $U(\mathbf{K}_\mathcal{T}\langle\!\langle-\rangle\!\rangle)$, the categorical interpretation $\mathbf{K}\langle\!\langle-\rangle\!\rangle$ is determined at $h_c^{\langle\!\langle A\rangle\!\rangle}$. However, there exist h_c^B for *every* B such that $B^r = A$ (not only for $B = \langle\!\langle A\rangle\!\rangle$), and we need $\mathbf{K}\langle\!\langle-\rangle\!\rangle$ to be determined at all those h_c^B. Fortunately, we have $\iota^+ : \mathbf{K}_\mathcal{T}\langle\!\langle B\rangle\!\rangle \longrightarrow \mathbf{K}_\mathcal{T}[\![B^r]\!] = \mathbf{K}_\mathcal{T}[\![A]\!]$. Also, $\mathbf{K}_\mathcal{T}\langle\!\langle h_c^{\langle\!\langle A\rangle\!\rangle}\rangle\!\rangle$ is equal to $I \xrightarrow{\mathbf{K}_\mathcal{T}\langle\!\langle h_c^B\rangle\!\rangle} \mathbf{K}_\mathcal{T}\langle\!\langle B\rangle\!\rangle \xrightarrow{\iota^+} \mathbf{K}_\mathcal{T}[\![A]\!]$. (This equation can be checked by sending it through $r(\mathcal{T})$, using the fact that $(\iota^+)^{r(\mathcal{T})}$ is essentially the identity—that is, a tuple of variables.) Furthermore, the diagram below commutes, as can be checked by induction over A.

$$\begin{tikzcd}
U\mathbf{K}_{\mathcal{T}}\llparenthesis A\rrparenthesis \arrow[r, phantom, "="] \arrow[d, "U\iota^+_A"'] & \mathbf{K}\llparenthesis A\rrparenthesis \arrow[d, "\iota^+_A"] \\
U\mathbf{K}_{\mathcal{T}}\llbracket A^r\rrbracket \arrow[r, phantom, "="] & \mathbf{K}\llbracket A^r\rrbracket
\end{tikzcd}$$

It follows that $\mathbf{K}\llparenthesis h_c^B\rrparenthesis = U(\mathbf{K}_{\mathcal{T}}\llparenthesis h_c^B\rrparenthesis) = U(\iota^+)^{-1} \circ U(\mathbf{K}_{\mathcal{T}}\llparenthesis h_c^{\llparenthesis A\rrparenthesis}\rrparenthesis) = (\iota^+)^{-1} \circ \mathbf{K}\llbracket \vdash c : A\rrbracket$. So $\mathbf{K}\llparenthesis - \rrparenthesis$ is uniquely determined. Because $\mathbf{K}_{\mathcal{T}}\llparenthesis - \rrparenthesis$ is surjective, U is uniquely determined.

For the existence of U, let $\mathbf{K}\llparenthesis - \rrparenthesis : \mathcal{L}_{cat} \longrightarrow \mathbf{K}$ be the categorical interpretation such that $\mathbf{K}\llparenthesis b\rrparenthesis = \mathbf{K}\llbracket b\rrbracket$ and $\mathbf{K}\llparenthesis h_c^B\rrparenthesis = (\iota^+)^{-1} \circ \mathbf{K}\llbracket \vdash c : A\rrbracket$. To obtain a well-defined U, we must prove that $\mathbf{K}_{\mathcal{T}}\llparenthesis f\rrparenthesis = \mathbf{K}_{\mathcal{T}}\llparenthesis g\rrparenthesis$ implies $\mathbf{K}\llparenthesis f\rrparenthesis = \mathbf{K}\llparenthesis g\rrparenthesis$. Letting $(\Gamma \vdash M : A) = f^{r(\mathcal{T})}$ and $(\Gamma \vdash N : A) = g^{r(\mathcal{T})}$, the equation $\mathbf{K}_{\mathcal{T}}\llparenthesis f\rrparenthesis = \mathbf{K}_{\mathcal{T}}\llparenthesis g\rrparenthesis$ means that $\Gamma \vdash M \equiv N : A$ holds in $\mathcal{T}$, which implies $\mathbf{K}\llbracket f^{r(\mathcal{T})}\rrbracket = \mathbf{K}\llbracket g^{r(\mathcal{T})}\rrbracket$ because $\mathbf{K}\llbracket - \rrbracket$ is a model. By Lemma 72, $\mathbf{K}\llbracket f^{r(\mathcal{T})}\rrbracket$ is the same as $\mathbf{K}\llparenthesis f\rrparenthesis$, up to isomorphism. So we have $\mathbf{K}\llparenthesis f\rrparenthesis = \mathbf{K}\llparenthesis g\rrparenthesis$.

$$\begin{tikzcd}
\mathbf{K}_{\mathcal{T}} \arrow[rr, "U"] & & \mathbf{K} \\
& & \\
\mathcal{L}_{cat} \arrow[uu, "\mathbf{K}_{\mathcal{T}}\llparenthesis - \rrparenthesis"] \arrow[uurr, "\mathbf{K}\llparenthesis - \rrparenthesis"] \arrow[rr, "r"'] & & \mathcal{L} \arrow[uu, "\mathbf{K}\llbracket - \rrbracket"'] \arrow[uull, "\mathbf{K}_{\mathcal{T}}\llbracket - \rrbracket"]
\end{tikzcd} \tag{15}$$

We define the *internal language* $\mathcal{L}(\mathbf{K})$ of $\mathbf{K}$ to be the λ_C-language with the objects of $\mathbf{K}$ as base types and a constant $\lceil f\rceil^{A\to B}$ for every $f \in \mathbf{K}(\llbracket A\rrbracket, \llbracket B\rrbracket)$ where $\llbracket A\rrbracket$ is the identity-on-base-types interpretation of A. Now let $\mathbf{K}\llbracket - \rrbracket$ be the identity-on-base-types interpretation of $\mathcal{L}(\mathbf{K})$ that sends each constant $\lceil f\rceil^{A\to B}$ to the Λ-mate of $I \otimes \llbracket A\rrbracket \cong \llbracket A\rrbracket \xrightarrow{f} \llbracket B\rrbracket$. We define the *internal theory* $\mathcal{T}(\mathbf{K})$ of $\mathbf{K}$ to be the theory on $\mathcal{L}(\mathbf{K})$ induced by $\mathbf{K}\llbracket - \rrbracket$.

We are now aiming to show that the internal theory can be used to check categorical equations. First, we need some terminology. Let $\mathcal{L}_{cat}(\mathbf{K})$ be the categorical language whose base types are the objects of $\mathbf{K}$ and whose operators are of the form $h : A \longrightarrow B$ where A and B are types of $\mathcal{L}_{cat}(\mathbf{K})$

and $h \in \mathbf{K}([\![A]\!], [\![B]\!])$. Let $\mathbf{K}\langle\!|-|\!\rangle$ be the evident interpretation of $\mathcal{L}_{cat}(\mathbf{K})$ in $\mathbf{K}$. (Note that this interpretation is what we intuitively use in real-life categorical reasoning. After all, we use signs like $\otimes$ and Λ on paper.) There is a second interpretation of $\mathcal{L}_{cat}(\mathbf{K})$, namely the identity-on-base types reverse interpretation $r(\mathbf{K})$ into $\mathcal{L}(\mathbf{K})$. We want it to interpret every operator $h : A \longrightarrow B$ of $\mathcal{L}_{cat}(\mathbf{K})$ by a constant $\ulcorner h' \urcorner : A^{r(\mathbf{K})} \to B^{r(\mathbf{K})}$ in $\mathcal{L}(\mathbf{K})$—that is, a we need an element h' of $\in \mathbf{K}([\![A^{r(\mathbf{K})}]\!], [\![B^{r(\mathbf{K})}]\!])$. The domain and codomain of $h : [\![A]\!] \longrightarrow [\![B]\!]$ are not exactly what we need, because A and B may contain the type constructor L, but fortunately, we can use ι^+: we define $h' = \iota_B \circ h \circ \iota_A^{-1}$, and $h^{r(\mathbf{K})} = (x_1 : A_1, \ldots, x_n : A_n \vdash \ulcorner h' \urcorner (x_{A^{r(\mathbf{K})}}) : B^{r(\mathbf{K})})$, where $A_1, \ldots, A_n$ is the factorization of $A^{r(\mathbf{K})}$.

The following proposition states that the denotation of every categorical expression f is expressed (up to isomorphism) by the reverse interpretation of f. So we can use the internal language for all equational reasoning in $\mathbf{K}$.

Theorem 74 (Internal language) *For every precartesian-closed abstract Kleisli-category $\mathbf{K}$, the following diagram commutes (where $A_1, \ldots, A_n$ is the factorization of A^r, and the isomorphism is built in the evident way from ι^+ and the associativity map for $\otimes$).*

$$\begin{tikzcd}
\mathbf{K}(\langle\!|A|\!\rangle, \langle\!|B|\!\rangle) \arrow[r, phantom, "\cong"] & K([\![A_1]\!] \otimes \cdots \otimes [\![A_n]\!], [\![B^r]\!]) \\
\mathcal{L}_{cat}(\mathbf{K})(A, B) \arrow[u, "\mathbf{K}\langle\!|-|\!\rangle"] \arrow[r, "r(\mathbf{K})"'] & \mathcal{L}(\mathbf{K})(A_1, \ldots, A_n \vdash B^r) \arrow[u, "\mathbf{K}[\![-]\!]"']
\end{tikzcd}$$

(So in particular, $\mathbf{K}\langle\!|-|\!\rangle$ and $r(\mathbf{K})$ validate the same equations.)

PROOF. This is a special case of Lemma 72.

Lemma 75 *Let $\mathbf{K}[\![-]\!] : \mathcal{T} \longrightarrow \mathbf{K}$ be a direct model, let $\Gamma \vdash M : A$ be an expression of $\mathcal{L}$, and let $f = \mathbf{K}[\![\Gamma \vdash M : A]\!]$. If $\Gamma \vdash M : A$ is an algebraic value, then f is thunkable. The converse holds if the theory on $|\mathcal{T}|$ induced by $\mathbf{K}[\![-]\!]$ is $\mathcal{T}$.*

PROOF. If $\Gamma \vdash M : A$ is an algebraic value, then we have Equation 1 in $\mathcal{T}$. Because $\mathbf{K}$ is a model, it holds that $(\Lambda \pi_1^{[\![A]\!], I}) \circ f = \Lambda(f \circ \pi_1^{[\![\Gamma]\!], I})$. Therefore we have

$$\vartheta_{[\![A]\!]} \circ f = \iota_{[\![A]\!]}^{-1} \circ (\Lambda \pi_1^{[\![A]\!], I}) \circ f = \iota_{[\![A]\!]}^{-1} \circ \Lambda(f \circ \pi_1^{\Gamma, I}) = [f] = Lf \circ \vartheta_{[\![\Gamma]\!]}.$$

Now suppose that the theory induced by $\mathbf{K}[\![-]\!]$ is $\mathcal{T}$. If f is thunkable, then

$$\iota^{-1}_{[\![A]\!]} \circ (\Lambda \pi_1^{[\![A]\!],I}) \circ f = \vartheta_{[\![A]\!]} \circ f = Lf \circ \vartheta_{[\![\Gamma]\!]} = [f] = \iota^{-1}_{[\![A]\!]} \circ \Lambda(f \circ \pi_1^{\Gamma,I}).$$

Because the left side is denoted by $\mathtt{let}\, x\, \mathtt{be}\, M\, \mathtt{in}\, \lambda().x$ and the right side by $\lambda().M$, we have Equation 1 in the theory induced by $\mathbf{K}$ and therefore in $\mathcal{T}$. By Lemma 3, $\Gamma \vdash M : A$ is an algebraic value.

9 $\mathcal{C}$-categories

In Section 8, we showed a correspondence between precartesian-closed abstract Kleisli-categories and λ_C-theories. In this section, we study those precartesian-closed abstract Kleisli-categories that correspond to the special case of $\mathcal{C}$-theories.

Let $\mathbf{K}$ be a precartesian-closed abstract Kleisli-category, and let O be an object of $\mathbf{K}$. For a morphism $f : A \longrightarrow B$ in $\mathbf{K}$, let $\widehat{f} : A \longrightarrow (B \rightharpoonup O) \rightharpoonup O$ be the adjoint mate of $A \otimes (B \rightharpoonup O) \xrightarrow{f \otimes (B \rightharpoonup O)} B \otimes (B \rightharpoonup O) \cong (B \rightharpoonup O) \otimes B \xrightarrow{apply} O$. The map $f \mapsto \widehat{f}$ is the internal-language counterpart of the map $M \mapsto \lambda k.kM$. It forms a transformation $\mathbf{K}(A, B) \cong \mathbf{K}_\vartheta(A, (B \to O) \to O)$ which is natural with respect to thunkable morphisms in A and all morphisms in B. To see the restricted naturality in A, let $f \in \mathbf{K}_\vartheta(A', A)$ and $g \in \mathbf{K}(A, B)$. We need $\widehat{g \circ f} = \widehat{g} \circ f$. By Lemma 75, that f is thunkable means that its internal-language representation $x : A' \vdash \lceil f \rceil\, x : A$ is an algebraic value. The naturality claim holds because, letting x be a variable of type A', we have

$$\begin{aligned}
\lceil \widehat{g \circ f} \rceil\, x &\equiv \lambda k.k(\lceil g \circ f \rceil\, x) \\
&\equiv \lambda k.k(\lceil g \rceil\, (\lceil f \rceil\, x)) \\
&\equiv \mathtt{let}\, y\, \mathtt{be}\, \lceil f \rceil\, x\, \mathtt{in}\, \lambda k.k(\lceil g \rceil\, y) && \text{because } \lceil f \rceil\, x \text{ is an algebraic value} \\
&\equiv \mathtt{let}\, y\, \mathtt{be}\, \lceil f \rceil\, x\, \mathtt{in}\, \lceil \widehat{g} \rceil\, y \\
&\equiv \lceil \widehat{g} \rceil\, (\lceil f \rceil\, x) && \text{because } \lceil f \rceil\, x \text{ is an algebraic value} \\
&\equiv \lceil \widehat{g} \circ f \rceil\, x.
\end{aligned}$$

The unrestricted naturality of $f \mapsto \widehat{f}$ in B follows from another simple internal-language argument, which we leave to the reader. The categorical versions of the $\mathcal{C}$-App and $\mathcal{C}$-Delay rules are

$$\begin{aligned}
\mathcal{C} \circ \widehat{f} &= f && (\mathcal{C}\text{-App}) \\
\widehat{\mathcal{C}} &= id && (\mathcal{C}\text{-Delay}).
\end{aligned}$$

where $\mathcal{C}$ ranges over morphisms $((B \rightharpoonup O) \rightharpoonup O) \longrightarrow B$. Such a family $\mathcal{C}$ exists if and only if the natural transformation $f \mapsto \widehat{f}$ is and isomorphism, in which case we have an adjunction whose counit is $\mathcal{C}$. This motivates the following definition:

Definition 76 *A* $\mathcal{C}$-category *is a precartesian-closed abstract Kleisli-category together with an object O such that the natural transformation $\mathbf{K}(A,B) \cong \mathbf{K}_\vartheta(A,(B \to O) \to O)$ is an isomorphism. We write $\mathcal{C}$ for the counit of the adjunction.*

Evidently, $\mathcal{C}$-categories are to $\mathcal{C}$-theories what precartesian-closed abstract Kleisli-categories are to λ_C-theories. That is, all propositions and theorems in Section 8 hold mutatis mutandis.

For the record, we state the categorical version of Theorem 5. For objects A and B of $\mathbf{K}$, let $\mathit{ftoc}^O_{A,B} : (A \rightharpoonup B) \longrightarrow (A \otimes (B \rightharpoonup O)) \rightharpoonup O$ be the morphism given in the internal language of $\mathbf{K}$ by the expression $\mathtt{ftoc}^O_{A,B}$.

Theorem 77 *For every precartesian-closed abstract Kleisli-category and all objects B and O, the following are equivalent:*

(1) $\mathit{ftoc}^O_{A,B}$ has an inverse for all A.
(2) $\mathit{ftoc}^O_{1,B}$ has an inverse.
*(3) There is a morphism $\mathcal{C}_B : ((B \rightharpoonup O) \rightharpoonup O) \longrightarrow B$ satisfying $\mathcal{C}$-*App *and $\mathcal{C}$-*Delay.

*Also, there is at most one $\mathcal{C}_B : ((B \rightharpoonup O) \rightharpoonup O) \longrightarrow B$ satisfying $\mathcal{C}$-*App *and $\mathcal{C}$-*Delay*.*

9.1 CPS transform and Kleisli construction

Let $\mathbf{C}$ be a response category, let T be the continuations monad on $\mathbf{C}$, and let $\mathbf{C}_T$ be the resulting precartesian-closed abstract Kleisli-category. We show that the internal-language representation of the construction of $\mathbf{C}_T$ from $\mathbf{C}$ is given by the CPS transform. From that result, we shall prove that $\mathbf{C}_T$ is a $\mathcal{C}$-category if $\mathbf{C}$ has a zero.

For every identity-on-base-types CPS transform $\gamma : \mathcal{L}(\mathbf{C}_T) \longrightarrow \mathcal{L}(\mathbf{C})$ and every type A of $\mathcal{L}(\mathbf{C}_T)$ it holds that $\mathbf{C}[\![A^\gamma]\!] = \mathbf{C}_T[\![A]\!]$. We shall simply write $[\![A]\!]$ for the joint value.

Proposition 78 *Let $\gamma : \mathcal{L}(\mathbf{C}_T) \longrightarrow \mathcal{L}(\mathbf{C})$ be the identity-on-base-types CPS transform such that $\mathbf{C}[\![\lceil f \rceil^\gamma]\!] = \mathbf{C}_T[\![\lceil f \rceil]\!]$ for every constant $\lceil f \rceil$ of $\mathcal{L}(\mathbf{C}_T)$. Then for every expression $\Gamma \vdash M : A$ of $\mathcal{L}(\mathbf{C}_T)$ it holds that*

$$\mathbf{C}[\![\Gamma^\gamma \vdash M^\gamma : (A^\gamma \to R) \to R]\!] = \mathbf{C}_T[\![\Gamma \vdash M : A]\!].$$

In other words, the following diagram commutes.

$$\begin{tikzcd}
\mathbf{C}_T(\llbracket\Gamma\rrbracket, \llbracket A\rrbracket) \arrow[d, equal] & \mathcal{L}(\mathbf{C}_T)(\Gamma, A) \arrow[l, "\mathbf{C}_T\llbracket - \rrbracket"'] \arrow[d, "\gamma"] \\
\mathbf{C}(\llbracket\Gamma\rrbracket, R^{R^{\llbracket A\rrbracket}}) & \mathcal{L}(\mathbf{C})(\Gamma^\gamma, (A^\gamma \to R) \to R) \arrow[l, "\mathbf{C}\llbracket - \rrbracket"']
\end{tikzcd}$$

PROOF. By induction over M.

Remark 79 *For γ to exist, the internal-language representation of $\mathbf{C}_T\llbracket\lceil f\rceil\rrbracket$ must be of the form $\lambda k.kP$ according to the definition of the CPS transform. This is true, because according to our interpretation of internal languages, $\mathbf{C}_T\llbracket\lceil f\rceil\rrbracket$ has the form Λg (where $g = I \otimes \llbracket\Gamma\rrbracket \cong \llbracket\Gamma\rrbracket \xrightarrow{f} \llbracket A\rrbracket$). But $\Lambda g = F_T\lambda_T g = \eta(\lambda_T g)$, and η is $x \vdash \lambda k.kx$ in $\mathcal{L}(\mathbf{C})$.*

Proposition 80 *If $\mathbf{C}$ is a response category with zero, then $\mathbf{C}_T$ is a $\mathcal{C}$-category.*

PROOF. Let $\mathcal{C}_A$ be the evident element of $\mathbf{C}_T((A \rightharpoonup 0) \rightharpoonup 0, A) = \mathbf{C}(R^{R^{A\times R^0}\times R^0}, R^{R^A})$. We check the categorical versions of $\mathcal{C}$-App and $\mathcal{C}$-Delay. Their reverse interpretations are

$\Gamma \vdash \lceil\mathcal{C}\rceil\,(\lambda k.kM) \equiv M : A \quad x : (A \to 0) \to 0 \vdash \lambda k.k(\lceil\mathcal{C}\rceil\, x) \equiv x : (A \to 0) \to 0$

if $\Gamma \vdash M : A$ is the reverse interpretation of f. Let γ be the CPS transform from Proposition 78. By Proposition 29 (soundness of γ), it suffices to prove that $(\lceil\mathcal{C}\rceil)^\gamma \equiv \lambda k.k(\lambda(h,l).h(\lambda(x,[]).lx,[]))$. By Proposition 78 it suffices to prove that

$$\mathbf{C}_T\llbracket\lceil\mathcal{C}\rceil\rrbracket = \mathbf{C}\llbracket\lambda k.k(\lambda(h,l).h(\lambda(x,[]).lx,[]))\rrbracket. \tag{16}$$

By definition, $\mathbf{C}_T(\lceil\mathcal{C}\rceil)$ is the Λ-mate of $((A \rightharpoonup 0) \rightharpoonup 0) \otimes I \cong (A \rightharpoonup 0) \rightharpoonup 0) \xrightarrow{\mathcal{C}} A$. Equation 16 follows (after some calculation) because we defined $\mathcal{C}$ as the evident element of $\mathbf{C}(R^{R^{A\times R^0}\times R^0}, R^{R^A})$.

9.2 The structure of $\mathcal{C}$-categories

In this section, we study the structure specific to $\mathcal{C}$-categories (as opposed to arbitrary precartesian-closed abstract Kleisli-categories), culminating in the Theorem 84, which states that every $\mathcal{C}$-category arises from a continuations monad.

Lemma 81 *In every $\mathcal{C}$-category $\mathbf{K}$, the object O is initial, and for every object B, the unique map $O \to B$ is in $\mathbf{K}_\vartheta$.*

PROOF. We use the internal theory $\mathcal{T}(\mathbf{K})$. For every $f \in \mathbf{K}(O, B)$ we have $f = \mathbf{K}[\![x : O \vdash \lceil f \rceil\, x : B]\!]$. By Lemma 6 it holds in $\mathcal{T}(\mathbf{K})$ that $\lceil f \rceil \equiv \mathcal{A}_B$, and therefore $f = \mathbf{K}[\![x : O \vdash \mathcal{A}_B x : B]\!]$. Again by Lemma 6, $x : O \vdash \mathcal{A}_B x : B$ is an algebraic value. So f is thunkable by Lemma 8.

Proposition 82 *For every $\mathcal{C}$-category $\mathbf{K}$, the subcategory $\mathbf{K}_\vartheta$ is a response category with $R = I \rightharpoonup O$ $R^A = A \rightharpoonup O$ and zero O.*

PROOF. The required map $ev \in \mathbf{K}_\vartheta(R^A \otimes A, R)$ is the Λ-mate of $(R^A \otimes A) \otimes I \cong R^A \otimes A \xrightarrow{apply} O$. To see the universal property of ev, note that for every $f \in \mathbf{K}_\vartheta(A \otimes B, R)$, a morphism $g \in \mathbf{K}_\vartheta(A, R^B)$ solves the equation

$$\begin{tikzcd}
R^B \otimes B \arrow[r, "ev"] & R \\
A \otimes B \arrow[u, "g \otimes B"] \arrow[ur, "f"'] &
\end{tikzcd}$$

if and only if g is the Λ-mate of $A \otimes B \cong (A \otimes B) \otimes I \xrightarrow{\Lambda^{-1} f} O$.

By Lemma 81, O is initial in $\mathbf{K}_\vartheta$. In particular, R^O is terminal in $\mathbf{K}_\vartheta$—that is, O is a zero.

So we have two strong monads with T-exponentials on $\mathbf{K}_\vartheta$: the continuations monad with the underlying functor $R^{R^{(-)}}$, and the monad with the underlying functor given by L (i.e. the monad obtained by sending $\mathbf{K}$ through the functor $\mathbf{AKl}^{\rightharpoonup}_{\otimes} \longrightarrow \mathbf{Mnd}^T_t$ given in Theorem 62). We shall prove the key fact that these two monads are isomorphic. It helps here to give the reverse interpretations of both monads' operators. The monad based on L has $T = L$, $(TA)^B = A \rightharpoonup B$, $\eta = \vartheta$, $\mu = L\varepsilon$, and $t = [id \otimes \varepsilon]$. Applying the reverse interpretation (and emptying the environments Γ of the resulting expressions by introducing an outermost λ-abstraction) yields

$$\begin{aligned}
\eta_A &= \lambda x : A.\lambda().x \\
\mu_A &= \lambda f : 1 \to 1 \to A.\lambda().f()() \\
t_{A,B} &= \lambda(x : A, f : 1 \to B).\lambda().(f(), x) \\
T &= \lambda f : A \to B.\lambda g : 1 \to A.\lambda().(f(g()).
\end{aligned}$$

The operators of the continuations monad are obtained as follows: first, one writes the usual definition of the continuations monad in the response category $\mathbf{K}_\vartheta$. Next, one expresses that definition in terms of the structure of $\mathbf{K}$, where exponentials are treated as in the proof of Proposition 82. Finally, one applies the reverse interpretation. This slightly laborious endeavor results in the data below, where $R^{(f:A\to B)} = \lambda h : B \to O.\lambda x : A.h(fx)$.

$$\begin{aligned}
\eta'_A &= \lambda x : A.\lambda k : A \to O.k\,x \\
\mu'_A &= R^{(\eta'_{A\to O})} \\
t'_{A,B} &= \lambda(x : A, h : (B \to O) \to O).\lambda k : (A \times B) \to O.h(\lambda y : B.k(x,y)) \\
T' &= \lambda f : A \to B.R^{R f}
\end{aligned}$$

For T-exponentials, we have $(T'B)^A = (A \times (B \to O)) \to O$.

Proposition 83 *For every $\mathcal{C}$-category $\mathbf{K}$, the identity functor on $\mathbf{K}_\vartheta$ forms a closed isomorphism of strong monads between the monad based on L and the continuations monad.*

PROOF. Let T be the monad based on L, and let T' be the continuations monad. The required natural transformation $\sigma_A : TA \longrightarrow T'A$ is $\widehat{\varepsilon}$. The reverse interpretation of this map is $\lambda f : 1 \to A.\lambda k : A \to O.k(f())$. The naturality of σ, as well as Diagrams 2 and 3 are easily checked by using the internal-language representations of the two monads (note that U_2 is the identity). For the strong monad morphism to be closed, the map $(TB)^A \longrightarrow (T'B)^A$ that arises as the adjoint mate of $(TB)^A \times A \xrightarrow{ev} TB \xrightarrow{\sigma} T'B$ must be an isomorphism for all A and B. In the internal language of $\mathbf{K}$, this map turns out to be $\lambda f : A \to B.\lambda(x : A, k : B \to O).k(f\,x)$. But this is essentially `ftoc` and therefore has an inverse, `ctof`. Because our monad morphism (Id, σ) is closed, it is also tight by Remark 61—that is, σ is an isomorphism. So (Id, σ) is an isomorphism, with inverse (Id, σ^{-1}).

Finally, the main structural theorem for $\mathcal{C}$-categories. Its proof is a "killer application" for the reflection theorems in Section 7, which culminated in Theorem 62.

Theorem 84 *Every $\mathcal{C}$-category arises from the continuations monad of a response category with zero, up to a closed isomorphism of precartesian-closed abstract Kleisli-categories.*

PROOF. Let $\mathbf{K}$ be a $\mathcal{C}$-category, and consider the diagram below.

$$\begin{tikzcd}
\mathbf{AKl}^{\rightharpoonup}_{\otimes} \arrow[d, shift left=1.5] \arrow[d, phantom, "\dashv"] & \mathbf{K} \arrow[rrd, mapsto] & \cong & (\mathbf{K}_\vartheta)_L & \cong & (\mathbf{K}_\vartheta)_{R^{R^{(-)}}} \\
\mathbf{Mnd}^T_t \arrow[u, shift left=1.5] & & & (\mathbf{K}_\vartheta, L) \arrow[u, mapsto] & \cong^{\sigma} & (\mathbf{K}_\vartheta, R^{R^{(-)}}) \arrow[u, mapsto]
\end{tikzcd}$$

The functor $\mathbf{AKl}^{\rightharpoonup}_{\otimes} \longrightarrow \mathbf{Mnd}^T_t$ from Theorem 62 sends $\mathbf{K}$ to the strong monad L on the category $\mathbf{K}_\vartheta$ with finite products and T-exponentials. The functor $\mathbf{Mnd}^T_t \longrightarrow \mathbf{AKl}^{\rightharpoonup}_{\otimes}$ sends this monad to $(\mathbf{K}_\vartheta)_L$, which by Theorem 62 is isomorphic to $\mathbf{K}$ via a closed morphism in $\mathbf{AKl}^{\rightharpoonup}_{\otimes}$. (The isomorphism is given by the counit of the reflection between $\mathbf{AKl}^{\rightharpoonup}_{\otimes}$ and $\mathbf{Mnd}^T_t$.) By Proposition 83, there is an closed isomorphism in $\mathbf{Mnd}^T_t$ from the monad L to the continuations monad $R^{R^{(-)}}$ on the response category $\mathbf{K}_\vartheta$. The functor $\mathbf{Mnd}^T_t \longrightarrow \mathbf{AKl}^{\rightharpoonup}_{\otimes}$ sends this isomorphism to a closed isomorphism $(\mathbf{K}_\vartheta)_L \cong (\mathbf{K}_\vartheta)_{R^{R^{(-)}}}$ in $\mathbf{AKl}^{\rightharpoonup}_{\otimes}$.

Remark 85 *It is in fact possible to adapt the whole of Theorem 62 to continuations, in the sense that the category of $\mathcal{C}$-categories becomes a reflective subcategory of the category of response categories with zero (with suitably-chosen notions of morphism). However, we content ourselves with the more general Theorem 62 and leave the details of its continuations version to the ambitious reader.*

Remark 86 *Selinger's "co-control" categories are $\mathcal{C}$-categories with finite sums satisfying some extra conditions. To see this, recall that our response categories are a generalization of Selinger's response categories [19], which have finite sums over which the products must distribute (see Remark 21). Co-control categories arise from his Selinger's response categories in the same way as $\mathcal{C}$-categories arise from our response categories. As it turns out, the distributive finite sums on the response category induce finite sums on the induced $\mathcal{C}$-category which make it into a co-control category.*

10 Conclusions

One of the themes of this work was the tightly intermeshed nature of the semantic and the syntactic point of view. Category theory, in our view, helps one in identifying the crucial abstract structures. Pragmatically, however, it is often much easier to work with syntax, such as internal languages for the structures under consideration. We have come to appreciate the economy and efficiency of the λ_C-calculus, and the control operator axioms in the style of Felleisen, as a tool for reasoning. In the same vein, syntactic transformations, the CPS transform foremost among them, are a powerful tool. In fact, we have taken a syntactic approach to issues which are normally seen as inherently semantic. Cases in point are the delaying transform and the use of categorical combinators. Such use of syntactic methods in not without parallel in category theory: witness the heroic effort required for Topos Theory by means of diagram-chases, and the relative ease of reasoning in intuitionistic set theory as the internal language of toposes.

Acknowledgements

We would like to thank the anonymous referees and Gordon Plotkin for many useful remarks and suggestions. Also, we acknowledge the use of Paul Taylor's diagram macros.

References

[1] C. Strachey, C. P. Wadsworth, Continuations: A mathematical semantics for handling full jumps, Monograph PRG-11, Oxford University Computing Laboratory, Programming Research Group, Oxford, UK (1974).

[2] C. Strachey, C. P. Wadsworth, Continuations: A mathematical semantics for handling full jumps, Higher-Order and Symbolic Computation 13 (1/2) (2000) 135–152, reprint of [1].

[3] H. Abelson, R. Dybvig, C. Haynes, G. Rozas, N. Adams, D. Friedman, E. Kohlbecker, G. Steele, D. Bartley, R. Halstead, D. Oxley, G. Sussman, G. Brooks, C. Hanson, K. Pitman, M. Wand, Revised^5 report on the algorithmic language Scheme, Higher-order and Symbolic Computation 11 (1) (1998) 7–105.

[4] B. Duba, R. Harper, D. MacQueen, Typing first-class continuations in ML, in: Proc. ACM Symp. Principles of Programming Languages, 1991, pp. 163–173.

[5] G. Steele, Rabbit: A compiler for Scheme, AI Technical Report 474, MIT (May 1978).

[6] J. C. Reynolds, The discoveries of continuations, Lisp and Symbolic Computation 6 (3/4) (1993) 233–247.

[7] P. J. Landin, A generalization of jumps and labels, Report, UNIVAC Systems Programming Research (Aug. 1965).

[8] P. J. Landin, A generalization of jumps and labels, Higher-Order and Symbolic Computation 11 (2).

[9] M. J. Fischer, Lambda-calculus schemata, in: Proceedings ACM Conference on Proving Assertions about Programs, Los Cruces, 1972, pp. 104–109, sIGPLAN Notices, 7(1), January 1972.

[10] G. D. Plotkin, Call-by-name, call-by-value and the λ-calculus, Theoretical Computer Science 1 (2) (1975) 125–159.

[11] M. Felleisen, D. P. Friedman, E. E. Kohlbecker, B. Duba, Reasoning with continuations, Proceedings of the Symposium on Logic in Computer Science (1986) 131–141.

[12] A. Sabry, M. Felleisen, Reasoning about programs in continuation-passing style, Lisp and Symbolic Computation 6 (3/4) (1993) 289–360.

[13] M. Hofmann, Sound and complete axiomatisations of call-by-value control operators, Mathematical Structures in Computer Science 5 (1995) 461–482.

[14] T. G. Griffin, A formulae-as-types notion of control, in: Principles of Programming Languages (POPL '90), ACM, 1990, pp. 47–58.

[15] A. Filinski, Declarative continuations: an investigation of duality in programming language semantics, in: D. H. Pitt, et al. (Eds.), Category Theory and Computer Science, no. 389 in Lecture Notes in Computer Science, Springer-Verlag, 1989, pp. 224–249.

[16] E. Moggi, Notions of computation and monads, Information and Computation 93 (1).

[17] J. Power, E. Robinson, Premonoidal categories and notions of computation, Mathematical Structures in Computer Science 7 (5) (1997) 453–468.

[18] H. Thielecke, Categorical structure of continuation passing style, Ph.D. thesis, University of Edinburgh (1997).

[19] P. Selinger, Control categories and duality: on the categorical semantics of the lambda-mu calculus, Mathematical Structures in Computer Science 11 (2001) 207–260.

[20] C. Führmann, The structure of call-by-value, Ph.D. thesis, Division of Informatics, University of Edinburgh (2000).

[21] A. W. Appel, Compiling with Continuations, Cambridge University Press, 1992.

[22] P. Taylor, Sober spaces and continuations, Theory and Applications of Categories 10.

[23] E. Moggi, Computational lambda-calculus and monads, in: Proceedings 4th Annual IEEE Symp. on Logic in Computer Science, LICS'89, Pacific Grove, CA, USA, 5–8 June 1989, IEEE Computer Society Press, Washington, DC, 1989, pp. 14–23.
URL `citeseer.nj.nec.com/moggi89computational.html`

[24] C. Führmann, Varieties of effects, in: Proceedings FOSSACS 2002, Vol. 2303 of LNCS, Springer, 2002, pp. 144–158.

[25] S. M. Lane, Categories for the Working Mathematician, Graduate Texts in Mathematics, Springer-Verlag, 1971.

[26] E. Moggi, Computational lambda-calculus and monads, Tech. Rep. ECS-LFCS-88-66, Edinburgh Univ., Dept. of Comp. Sci. (1988).

[27] J. Power, H. Thielecke, Closed Freyd- and kappa-categories, in: Proc. ICALP '99, Vol. 1644 of LNCS, Springer, 1999.

[28] A. Filinski, Recursion from iteration, Lisp and Symbolic Computation 7 (1) (1994) 11–38.

[29] M. Hasegawa, Y. Kakutani, Axioms for recursion in call-by-value, Higher-Order and Symbolic Computation 15 (2-3) (2002) 235–264.

[30] G. Boudol, Pi-calculus in direct style, in: Symposium on Principles of Programming Languages (POPL'97), ACM, 1997.

[31] P. Z. Ingerman, Thunks: a way of compiling procedure statements with some comments on procedure declarations., Communications of the ACM 4 (1) (1961) 55–58.

[32] J. Hatcliff, O. Danvy, Thunks and the λ-calculus, Journal of Functional Programming 7 (2) (1997) 303–319.

[33] C. Führmann, Direct models of the computational lambda-calculus, in: Proceedings MFPS XV, Vol. 20 of Electronic Notes in Theoretical Computer Science, Elsevier, New Orleans, 1999.