Algebraic effects and handlers for arrows

TAKAHIRO SANADA

Research Institute for Mathematical Sciences,
Kyoto University
(e-mail: tsanada@kurims.kyoto-u.ac.jp)

Abstract

We present an arrow calculus with operations and handlers, and its operational and denotational semantics. The calculus is an extension of Lindley, Wadler and Yallop's arrow calculus.

The denotational semantics is given using strong (pro)monad $\mathscr{A}$ in the bicategory of categories and profunctors. The construction of appropriate strong monad $\mathscr{A}$ is not trivial because of a size problem. To build denotational semantics, we investigate what $\mathscr{A}$-algebras are, and a handler is interpreted as an $\mathscr{A}$-homomorphisms between $\mathscr{A}$-algebras.

The syntax and operational semantics are derived from the observations on $\mathscr{A}$-algebras. We prove the soundness and adequacy theorem of the operational semantics for the denotational semantics.

1 Introduction

Hughes (2000) introduced the notion of *arrow* as an extension of the notion of monad for Haskell to capture non-monadic computational effects, such as a parser combinator. As a syntactic development, the arrow calculus was introduced by Lindley et al. (2010). Their calculus is an arrow version of Moggi's metalanguage (Moggi (1991)). As a semantic development, Heunen and Jacobs (2006); Jacobs et al. (2009); Asada (2010) revealed that arrows are strong monads in the bicategory **Prof** of categories and profunctors.

It is a natural question whether we can construct an arrow version of *algebraic effects* and *effect handlers* since arrows are an extension of monads. Plotkin and Power (2001*a*,b) presented an algebraic view for computational effects. Plotkin and Pretnar (2013) provided effect handlers as a way to implement effects. Algebraic effects and effect handlers are the foundations of programming language with effects that correspond to finitary (or more generally ranked) monads. Can we obtain an arrow version of such foundations?

Lindley (2014) defined an effect system λ_{flow} which has algebraic effects and handlers for arrows, monads and idioms. However, the effect system λ_{flow} is not satisfactory for the following reasons.

- The theoretical background of algebraic effects for arrows is ambiguous. Any categorical explanation of algebraic theories for arrows is not given.
- The syntax is complicated. It is unclear why the construction of handlers is given in that way.

- Denotational semantics is not defined. It seems hard to give denotational semantics because the algebraic foundation of effects and handlers is not discussed enough.

We present *an arrow calculus with operations and handlers* as an extension of the arrow calculus. We discuss a categorical foundation for algebraic theories for arrows and give denotational semantics for our calculus by constructing an appropriate strong monad in (**Ens**-)**Prof**. As a main result, the soundness and adequacy theorems of the operational semantics with respect to denotational semantics is proven.

Our contributions are as follows.

- We describe *algebras* for arrows from a 2-categorical point of view.
- We present *an arrow calculus with operations and handlers* based on the notions of algebras for arrows. The *progress* and *preservation* theorems for the calculus are shown.
- We give a denotational semantics for the calculus and prove the *soundness* theorem. There are the following non-trivial points in defining the denotational semantics.

  - The "smallness" of an appropriate strong monad in (**Ens**-)**Prof**. The collection of *arrow terms*, which are arrow analogues of terms in ordinary algebraic theories, is a proper class, not a set. Hence, the "smallness" of a monad that we construct is not trivial. We prove the "smallness" of the monad by counting the number of *normal forms* of arrow terms, which was introduced by Yallop (2010) for Haskell programs of arrow types.
  - A treatment of strength to construct an algebra from a handler. Unlike ordinary handlers, we need a trick to define interpretation of handlers for arrows because of the strength of strong monads in **Prof**. We define interpretation $[\![-]\!]^S$ with a set S as a parameter to construct an algebra from a handler.

1.1 Arrows in Haskell

Hughes (2000, 2004) introduced arrows as a generalisation of monads. In Haskell, arrows are defined using a type class[1].

```haskell
class Arrow a where
    arr :: (x -> y) -> a x y
    (>>>) :: a x y -> a y z -> a x z
    first :: a x y -> a (x, z) (y, z)
```

---

[1] In Haskell, for types x and y, (x, y) is the product type of x and y.

An instance of `Arrow` is required to satisfy the following *arrow laws*.

$$(a \ggg b) \ggg c = a \ggg (b \ggg c) \tag{1.1}$$

$$\mathtt{arr}\ (g \circ f) = \mathtt{arr}\ g \ggg \mathtt{arr}\ f \tag{1.2}$$

$$\mathtt{arr\ id} \ggg a = a \tag{1.3}$$

$$a \ggg \mathtt{arr\ id} = a \tag{1.4}$$

$$\mathtt{first}\ a \ggg \mathtt{arr}\ (\mathrm{id} \times f) = \mathtt{arr}\ (\mathrm{id} \times f) \ggg \mathtt{first}\ a \tag{1.5}$$

$$\mathtt{first}\ a \ggg \mathtt{arr}\ \pi_1 = \mathtt{arr}\ \pi_1 \ggg a \tag{1.6}$$

$$\mathtt{first}\ a \ggg \mathtt{arr}\ \alpha = \mathtt{arr}\ \alpha \ggg \mathtt{first}\ (\mathtt{first}\ a) \tag{1.7}$$

$$\mathtt{first}\ (\mathtt{arr}\ f) = \mathtt{arr}\ (f \times \mathrm{id}) \tag{1.8}$$

$$\mathtt{first}\ (a \ggg b) = \mathtt{first}\ a \ggg \mathtt{first}\ b \tag{1.9}$$

where $\pi_1 : X \times Y \to X$ is a projection and $\alpha : (X \times Y) \times Z \to X \times (Y \times Z)$ is an associator.

We explain some intuition of the type class `Arrow`. Let `A` be an instance of `Arrow`. A type `A x y` is the type of computation whose input type is `x` and output type is `y`. The function `arr` makes pure computation `arr` f of type `A x y` from a function f of type `x -> y`. The function ($\ggg$) composes two computations f of type `A x y` and g of type `A y z` and returns a computation $f \ggg g$ of type `A x z`. The function `first` introduces an additional type z to the input and output types of a computation f of type `A x y` and returns a computation `first` f of type `A (x, z) (y, z)`. The computation `first` f takes an input of type `(x, z)`, applies f to the first component of the input and does nothing to the second component of the input, and returns an output of type `(y, z)`.

1.2 An example: logic circuit simulation by effects and handlers

Suppose we want to write a simulator for logic circuits. Logic circuits are composed of *wires* and *gates*. Wires connect gates and transmit boolean values. There are different types of gates, such as AND, OR, NOT and NAND. The gates AND, OR, and NAND takes two boolean values as inputs and outputs one boolean value. The gate NOT takes one boolean value as an input and outputs one boolean value. See Table 1 for the definitions as boolean functions of the gates.

Table 1. Logic gates

	AND	OR	NOT	NAND
Symbol				
Boolean function	$\mathrm{AND}(x, y)$ $= x \wedge y$	$\mathrm{OR}(x, y)$ $= x \vee y$	$\mathrm{NOT}(x)$ $= \neg x$	$\mathrm{NAND}(x, y)$ $= \neg(x \wedge y)$

1.2.1 Logic circuit simulation by ordinary algebraic effects and handlers

We can write a simulator for logic circuits in a language with ordinary algebraic effects and handlers. Let a signature $\Sigma_{\mathrm{LC}} = \{\mathrm{AND} : \mathrm{Bool} \times \mathrm{Bool} \twoheadrightarrow \mathrm{Bool}, \mathrm{OR} : \mathrm{Bool} \times \mathrm{Bool} \twoheadrightarrow \mathrm{Bool}, \mathrm{NOT} : \mathrm{Bool} \twoheadrightarrow \mathrm{Bool}, \mathrm{NAND} : \mathrm{Bool} \times \mathrm{Bool} \twoheadrightarrow \mathrm{Bool}\}$. For example, $\mathrm{AND} : \mathrm{Bool} \times$

Bool $\twoheadrightarrow$ Bool $\in \Sigma_{\text{LC}}$ is an algebraic operation that takes two boolean values as inputs and outputs one boolean value.

Firstly we write a handler to implement NAND and OR by AND and NOT.

$$H_1 = \{\text{NAND}(x, y), k \mapsto \textbf{let } u \Leftarrow \text{AND}(x, y) \textbf{ in let } v \Leftarrow \text{NOT}(u) \textbf{ in } k(v)$$
$$\text{OR}(x, y), k \mapsto \textbf{let } u \Leftarrow \text{NOT}(x) \textbf{ in let } v \Leftarrow \text{NOT}(y) \textbf{ in let } w \Leftarrow \text{AND}(u, v) \textbf{ in } k(w)\}$$
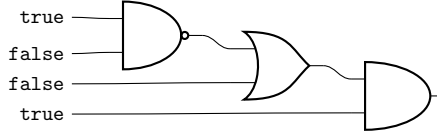
Secondly we write a handler to implement NOT and AND.

$$H_2 = \{\text{AND}(x, y), k \mapsto \textbf{if } x \textbf{ then}(\textbf{if } y \textbf{ then } k(\texttt{true}) \textbf{ else } k(\texttt{false})) \textbf{ else } k(\texttt{false})$$
$$\text{NOT}(x), k \mapsto \textbf{if } x \textbf{ then } k(\texttt{false}) \textbf{ else } k(\texttt{true})\}$$

By the handlers H_1 and H_2, we can simulate logic circuits. For example, we define a program P as

$$P = (\textbf{let } x \Leftarrow \text{NAND}(\texttt{true}, \texttt{false}) \textbf{ in let } y \Leftarrow \text{OR}(x, \texttt{false}) \textbf{ in } \text{AND}(y, \texttt{true})). \quad (1.10)$$

The program P corresponds to the following logic circuit.

Then, we can obtain the simulation result of P by handling it with H_1 and H_2:

$$\textbf{handle}(\textbf{handle } P \textbf{ with } H_1) \textbf{ with } H_2 \rightarrow^* \texttt{true}.$$

The advantage of using handlers is that the structure of a logic circuit can be separated from its implementation. We can use different implementation for logic circuits using the following handlers:

$$H_3 = \left\{ \begin{array}{l} \text{AND}(x, y), k \\ \quad \mapsto \textbf{let } u \Leftarrow \text{NAND}(x, y) \textbf{ in let } v \Leftarrow \text{NOT}(u) \textbf{ in } k(v) \\ \text{OR}(x, y), k \\ \quad \mapsto \textbf{let } u \Leftarrow \text{NOT}(x) \textbf{ in let } v \Leftarrow \text{NOT}(y) \textbf{ in let } w \Leftarrow \text{NAND}(u, v) \textbf{ in } k(w) \end{array} \right\}$$

$$H_4 = \{\text{NOT}(x), k \mapsto \textbf{let } u \Leftarrow \text{NAND}(x, x) \textbf{ in } k(u)\},$$

$$H_5 = \{\text{NAND}(x, y), k \mapsto \textbf{if } x \textbf{ then}(\textbf{if } y \textbf{ then } k(\texttt{false}) \textbf{ else } k(\texttt{true})) \textbf{ else } k(\texttt{true})\}.$$

We have $\textbf{handle}(\textbf{handle}(\textbf{handle } P \textbf{ with } H_3) \textbf{ with } H_4) \textbf{ with } H_5 \rightarrow^* \texttt{true}.$

1.2.2 A problem of the approach with ordinary effects and handlers

Although we can write logic circuits with ordinary algebraic effects, the expressive power of programming languages with such effects is too high. This is because it is possible to describe *dynamic-* or *meta-operations* on circuits that cannot be realised in normal circuits. For example, let Q be the following program:

$$Q = (\textbf{let } x \Leftarrow \text{AND}(\texttt{true}, \texttt{false}) \textbf{ in if } x \textbf{ then } \text{AND}(x, \texttt{true}) \textbf{ else } \text{NOT}(x)). \quad (1.11)$$

This program corresponds to the following "logic circuit."
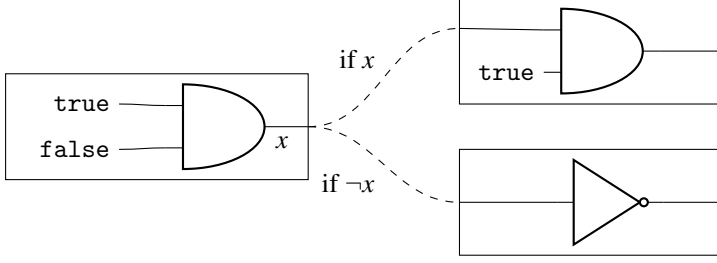
The above "logic circuit" has dynamic selection of circuits according to the output value of the first AND gate. Since such dynamic selection is an "out-of-circuit" operation, we want to restrict the possibility of writing such a program.

1.2.3 Logic circuit simulation by the arrow calculus with operations and handlers

Since arrows are generalisation of monads, the expression power of arrows is weaker than that of monads. We can exploit the constraints to restrict dynamic selection of logic circuits. Algebraic effects that correspond to arrows, which we introduce as the *arrow calculus with operation and handlers* in this paper, has a restriction such that it is impossible to perform conditional branching on their output to select subsequent algebraic effects.

The formal syntax of the arrow calculus with operations and handlers is given in Section 4.1. Here, we give informal descriptions of the syntax and explain the restriction.

The arrow calculus has two kinds of judgements:

$$\Gamma \vdash M : A \quad \text{and} \quad \Gamma \,\mathring{,}\, \Delta \vdash P : A$$

where $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ and $\Delta = y_1 : B_1, \ldots, y_m : B_m$ are typing environments and A is a type. The *term M* is a pure function of type A and the *command P* is a computation which takes inputs of types Δ and returns an output of type A under the context Γ. Compared to the arrow class in Haskell (Section 1.1), the command P corresponds to a Haskell function of type "$\Gamma \to \text{Ar}\ \Delta\ A$" where Ar is an instance of the arrow class Arrow.

If we have a pure function M from Δ to A, we can obtain a pure computation:

$$\frac{\Gamma, \Delta \vdash M : A}{\Gamma \,!\, \Delta \vdash \lfloor M \rfloor \,!\, A}$$

This rule corresponds to arr of the class Arrow in Haskell.

For an operation $\text{op} \in \Sigma$, we can perform the operation with an input M:

$$\frac{\text{op} : \gamma \twoheadrightarrow \delta \in \Sigma \quad \Gamma, \Delta \vdash M : \gamma}{\Gamma \,\mathring{,}\, \Delta \vdash \text{op}(M) \,!\, \delta}$$

Abstraction and application of commands are given by the following rules, respectively:

$$\frac{\Gamma \,\mathring{,}\, x : A \vdash P \,!\, B}{\Gamma \vdash \lambda^{\bullet} x : A.P : A \rightsquigarrow B} \quad \text{and} \quad \frac{\Gamma \vdash L : A \rightsquigarrow B \quad \Gamma, \Delta \vdash M : A}{\Gamma \,\mathring{,}\, \Delta \vdash L \bullet M \,!\, B}$$

where the type $A \rightsquigarrow B$ corresponds to a type $\text{Ar}\ A\ B$ in Haskell. The rule for $L \bullet M$ corresponds to ($\ggg$) of the class Arrow in Haskell.

We also have sequential composition **let** $x \Leftarrow P$ **in** Q of commands P and Q, which corresponds to $(\ggg)$ and `first` in Haskell:

$$\frac{\Gamma \mathbin{;} \Delta \vdash P \mathbin{!} A \quad \Gamma \mathbin{;} x : A, \Delta \vdash Q \mathbin{!} B}{\Gamma \mathbin{;} \Delta \vdash \mathbf{let}\ x \Leftarrow P\ \mathbf{in}\ Q \mathbin{!} B}$$

The important difference from the ordinary algebraic effects is that we *cannot* add a conditional branching like **if** M **then** P_1 **else** P_2 to the syntax of the arrow calculus, although we can add **if** M **then** N_1 **else** N_2 to the arrow calculus where M, N_1 and N_2 are terms and P_1 and P_2 are commands. This restriction comes from semantic observation. An algebra of a promonad, which is a semantic counterpart of arrows, is a sequence (without branching) of operations whereas an algebra of an ordinary monad is a tree of operations. Hence, we cannot add a conditional branching to the arrow calculus because the algebraic structure has no branching, and we can do conditional branching in a language with ordinary algebraic effects because the algebraic structure has branching. This observation is informally described by Lindley (2014), and we give a formal explanation in Section 3.

Let's return to the simulation of logic circuits. Now we can restrict the dynamic selection of circuits by using the constraints of the arrow calculus. The program P defined by (1.10) is also a valid program in the arrow calculus, and Q defined by (1.11) is not a program in the arrow calculus.

In the arrow calculus with operations and handlers, handlers H_1' and H_2' corresponding to H_1 and H_2 are defined as follows.

$$H_1' = \left\{ \begin{array}{ll} \mathsf{NAND}, k : \mathsf{Bool} \rightsquigarrow \mathsf{Bool} \mathbin{;} z : \mathsf{Bool} \times \mathsf{Bool} & \mapsto \quad \begin{array}{l} \mathbf{let}\ u \Leftarrow \mathsf{AND}(z)\ \mathbf{in} \\ \mathbf{let}\ v \Leftarrow \mathsf{NOT}(u)\ \mathbf{in}\ k \bullet v \end{array} \\[2em] \mathsf{OR}, k : \mathsf{Bool} \rightsquigarrow \mathsf{Bool} \mathbin{;} z : \mathsf{Bool} \times \mathsf{Bool} & \mapsto \quad \begin{array}{l} \mathbf{let}\ u \Leftarrow \mathsf{NOT}(\mathbf{fst}\ z)\ \mathbf{in} \\ \mathbf{let}\ v \Leftarrow \mathsf{NOT}(\mathbf{snd}\ z)\ \mathbf{in} \\ \mathbf{let}\ w \Leftarrow \mathsf{AND}(\langle u, v \rangle)\ \mathbf{in} \\ k \bullet w \end{array} \end{array} \right\}$$

$$H_2' = \left\{ \begin{array}{l} \mathsf{AND}, k : \mathsf{Bool} \rightsquigarrow \mathsf{Bool} \mathbin{;} z : \mathsf{Bool} \times \mathsf{Bool} \\ \mapsto k \bullet (\mathbf{if}(\mathbf{fst}\ z)\ \mathbf{then}(\mathbf{if}(\mathbf{snd}\ z)\ \mathbf{then}\ \mathtt{true}\ \mathbf{else}\ \mathtt{false})\ \mathbf{else}\ \mathtt{false}) \\ \mathsf{NOT}, k : \mathsf{Bool} \rightsquigarrow \mathsf{Bool} \mathbin{;} x : \mathsf{Bool} \\ \mapsto k \bullet (\mathbf{if}\ x\ \mathbf{then}\ \mathtt{false}\ \mathbf{else}\ \mathtt{true}) \end{array} \right\}$$

Note that, in the construction of H_2', we can use **if**'s because these **if**'s select terms, not commands.

We can simulate logic circuits by handling P with H_1' and H_2':

$$\mathbf{handle}(\mathbf{handle}\ P\ \mathbf{with}\ H_1')\ \mathbf{with}\ H_2' \rightarrow^* \lfloor \mathtt{true} \rfloor.$$

We can also construct handlers H_3', H_4' and H_5' corresponding to H_3, H_4 and H_5. For more details, see Section 4.3.1.

1.3 The structure of this paper

The rest of this paper is organised as follows. Section 2 is a section on categorical preliminaries. In Section 3, we describe *algebras* for a monad in the bicategory of categories and profunctors, and observe the universality of a free algebra. The arrow calculus with operations and handlers is introduced in Section 4. Typing rules and operational semantics are presented. In Section 5, we define the denotational semantics for the arrow calculus with operations and handlers. The definition of models is given in Section 5.1. We tackle the "smallness" problem and construct a model in Section 5.2. In Section 5.3, we define the interpretation with attention to the treatment of strength. Soundness and adequacy is shown in Section 5.4.

2 Preliminaries on category theory

In this paper, we assume that the readers are familiar with basic notions of category theory such as adjunction, monads, presheaves, the Yoneda lemma, Eilenberg-Moore categories and monoidal categories as described in a textbook by Mac Lane (1971). A textbook by Leinster (2014) is also a good reference for category theory. We use advanced topics of category theory such as coends, 2-categories, bicategories and enriched categories. Readers unfamiliar with coends are referred to Appendix 1. Readers unfamiliar with higher categories such as 2-categories, bicategories and enriched categories are referred to Appendix 2.

Throughout this paper, we use the following notation.

Notation 2.1. We denote the categories of sets and maps, and classes and maps of classes as **Set** and **Ens**, respectively. The 2-category of small categories and functors is denoted by **Cat**. For a category $\mathbb{C}$, we write the set of objects of $\mathbb{C}$ as $\mathrm{Ob}(\mathbb{C})$. We write $\mathrm{Id}_{\mathbb{C}}$ for the identity functor on $\mathbb{C}$. When a category $\mathbb{C} = (\mathbb{C}, \otimes, J)$ is symmetric monoidal closed, we write $\Lambda\colon \mathbb{C}(A \otimes B, C) \to \mathbb{C}(A, B \Rightarrow C)$ for the currying operator.

2.1 Profunctors

Arrows can be seen as strong monads in the bicategory **Prof** of categories and profunctors (Heunen and Jacobs (2006); Jacobs et al. (2009); Asada (2010)). We review profunctors and strong monads in **Prof**.

In the following definition of profunctors, we use coends, which are defined and gave an informal description in Appendix 1. For more detailed explanation of coends, see also (Mac Lane, 1971, Section 9) or (Loregian, 2021, Section 1).

Definition 2.2 (profunctor, Bénabou (2000))**.** Let $\mathbb{C}$ and $\mathbb{D}$ be categories. A *profunctor* $F\colon \mathbb{C} \nrightarrow \mathbb{D}$ from $\mathbb{C}$ to $\mathbb{D}$ is a functor $F\colon \mathbb{D}^{\mathrm{op}} \times \mathbb{C} \to \mathbf{Set}$. For profunctors $F\colon \mathbb{C} \nrightarrow \mathbb{D}$ and $G\colon \mathbb{D} \nrightarrow \mathbb{E}$, their composite $G \circ F\colon \mathbb{C} \nrightarrow \mathbb{E}$ is defined by taking the coend:

$$(G \circ F)(E, C) = \int^{D \in \mathbb{D}} G(E, D) \times F(D, C) \tag{2.1}$$

Table 2. Analogy between profunctors and binary relations

	A profunctor $F\colon \mathbb{C} \nrightarrow \mathbb{D}$	A binary relation $r \subseteq C \times D$
Substance	A functor $F\colon \mathbb{D}^{\mathrm{op}} \times \mathbb{C} \to \mathbf{Set}$	A function $r\colon D \times C \to 2$
Composition	$(G \circ F)(E,C) =$ $\int^D G(E,D) \times F(D,C)$	$(s \circ r)(e,c) \iff$ $\exists d.\, s(e,d) \wedge r(d,c)$
Identity	$\mathrm{I}_\mathbb{C}(C,C') = \mathbb{C}(C,C')$	$i(c,c') = \begin{cases} 0 & \text{if } c \neq c' \\ 1 & \text{if } c = c' \end{cases}$

for $E \in \mathrm{Ob}(\mathbb{E})$ and $C \in \mathrm{Ob}(\mathbb{C})$. The *identity profunctor* $\mathrm{I}_\mathbb{C}\colon \mathbb{C} \nrightarrow \mathbb{C}$ is defined by

$$\mathrm{I}_\mathbb{C}(C,D) = \mathbb{C}(C,D)$$

for $C, D \in \mathrm{Ob}(\mathbb{C})$. A *2-cell* $\alpha\colon F \Rightarrow G$ between profunctors $F, G\colon \mathbb{C} \nrightarrow \mathbb{D}$ is a natural transformation.

A profunctor $F\colon \mathbb{C} \nrightarrow \mathbb{D}$ is an analogue of a binary relation $r \subseteq C \times D$ between two sets C and D. We identify the binary relation r with its characteristic function $D \times C \to 2$. A composition $s \circ r \subseteq C \times E$ of two relations $r \subseteq C \times D$ and $s \subseteq D \times E$ is defined as follows:

$$(s \circ r)(e,c) \iff \exists d \in D.\, s(e,d) \wedge r(d,c).$$

This definition of compositions is similar to the definition of compositions of profunctors (2.1) because the coend operator $\int^{D \in \mathbb{D}}(-)$ is an analogue of an existential quantifier $\exists D \in \mathbb{D}.\,(-)$ as described in Appendix 1. The correspondence between profunctors and binary relations is summarised in Table 2.

A profunctor $G\colon 1 \nrightarrow \mathbb{C}$ is a presheaf $G\colon \mathbb{C}^{\mathrm{op}} \to \mathbf{Set}$ on $\mathbb{C}$. We have $(H \circ G) \circ F \cong H \circ (G \circ F)$ and $\mathrm{I}_\mathbb{D} \circ F \cong F \cong F \circ \mathrm{I}_\mathbb{C}$. That is, associativity and unitality hold *up to natural isomorphism*, not strictly. Moreover, the class of small categories and profunctors forms a bicategory. Roughly speaking, a bicategory is a "category" whose hom-sets have a category structure and whose composition and identity are associative and unital up to isomorphism. See Appendix 2.2 or (Borceux, 1994, Section 7.7) for the definition of bicategories. We write the bicategory of profunctors as **Prof**.

From a functor $F\colon \mathbb{C} \to \mathbb{D}$, we can obtain two profunctors $F_*\colon \mathbb{C} \nrightarrow \mathbb{D}$ and $F^*\colon \mathbb{D} \nrightarrow \mathbb{C}$ which are defined by

$$F_*(D,C) = \mathbb{D}(D,FC), \qquad F^*(C,D) = \mathbb{D}(FC,D)$$

on objects $C \in \mathrm{Ob}(\mathbb{C})$ and $D \in \mathrm{Ob}(\mathbb{D})$. For morphisms f in $\mathbb{C}$ and g in $\mathbb{D}$, $F_*(g,f)$ and $F^*(f,g)$ are also be defined appropriately.

2.2 Monads in the bicategory of profunctors

To capture notion of computations, monads have been used in functional programming (Moggi (1989, 1991)). In this paper, to capture notion of computation, we use monads *in* the bicategory **Prof** instead of ordinary monads, that is monads *in* the 2-category **Cat**. For the definition of monads in 2-categories and bicategories, see Appendices 2.1 and 2.2.

We call a monad in **Prof** a *promonad*. A strong promonad is a promonad with a strength.

Definition 2.3 (strong promonad, Asada (2010); Asada and Hasuo (2010))**.** Let $\mathbb{C} = (\mathbb{C}, \otimes, J)$ be a monoidal category. A *strong promonad* on $\mathbb{C}$ is a profunctor $\mathscr{A}\colon \mathbb{C} \nrightarrow \mathbb{C}$ with 2-cells $\eta^{\mathscr{A}}\colon \mathrm{I}_{\mathbb{C}} \Rightarrow \mathscr{A}$, $\mu^{\mathscr{A}}\colon \mathscr{A} \circ \mathscr{A} \Rightarrow \mathscr{A}$, and $\sigma^{\mathscr{A}}\colon (\otimes_*) \circ (\mathscr{A} \times \mathrm{I}_{\mathbb{C}}) \Rightarrow \mathscr{A} \circ (\otimes_*)$:

satisfying the axioms shown in Fig. 1.

Hughes (2000); Jacobs et al. (2009); Asada (2010) showed a promonad corresponds to an arrow type in Haskell. The unit $\eta\colon \mathrm{I}_{\mathbb{C}} \Rightarrow \mathscr{A}$ of a promonad $\mathscr{A}$ corresponds to `arr : x -> y -> A x y`:

$$\frac{\eta\colon \mathrm{I}_{\mathbb{C}} \Rightarrow \mathscr{A}}{\eta_{A,B}\colon \mathbb{C}(A,B) \to \mathscr{A}(A,B) \quad \text{natural in } A \text{ and } B}$$

The multiplication $\mu\colon \mathscr{A} \circ \mathscr{A} \Rightarrow \mathscr{A}$ of the promonad $\mathscr{A}$ corresponds to $(\ggg)\colon$ `A x y -> A y z -> A x z`:

$$\frac{\mu\colon \mathscr{A} \circ \mathscr{A} \Rightarrow \mathscr{A}}{\frac{\mu_{A,C}\colon \int^{B\in\mathbb{C}} \mathscr{A}(A,B) \times \mathscr{A}(B,C) \to \mathscr{A}(A,C) \quad \text{natural in } A \text{ and } C}{\mu_{A,B,C}\colon \mathscr{A}(A,B) \times \mathscr{A}(B,C) \to \mathscr{A}(A,C) \quad \text{natural in } A \text{ and } C, \text{ and extranatural in } B}}$$

The strength $\sigma\colon \otimes_* \circ (\mathscr{A} \times \mathrm{I}_{\mathbb{C}}) \Rightarrow \mathscr{A} \circ \otimes_*$ of the promonad $\mathscr{A}$ corresponds to `first : A x y -> A (x, z) (y, z)`. The proof of the correspondence is complicated, see (Asada, 2010, Theorem 14):

$$\frac{\sigma\colon \otimes_* \circ (\mathscr{A} \times \mathrm{I}_{\mathbb{C}}) \Rightarrow \mathscr{A} \circ \otimes_*}{\mathscr{A}(A,B) \to \mathscr{A}(A \otimes C, B \otimes C) \quad \text{natural in } A \text{ and } B, \text{ and extranatural in } C}$$

From a monad, we can obtain a promonad:

Proposition 2.4. *Let $\mathscr{T}\colon \mathbb{C} \to \mathbb{C}$ be a monad. The profunctor $\mathscr{T}_*\colon \mathbb{C} \nrightarrow \mathbb{C}$ is a promonad.*

Note that the profunctor $T_*\colon \mathbb{C} \nrightarrow \mathbb{C}$ is a functor $\mathbb{C}^{\mathrm{op}} \times \mathbb{C} \to \mathbf{Set}$, and we have $\mathscr{T}_*(C,D) = \mathbb{C}(C, \mathscr{T}D) = \mathscr{K}\!\ell(\mathscr{T})(C,D)$ where $\mathscr{K}\!\ell(\mathscr{T})$ is the Kleisli category for the monad $\mathscr{T}$. Hence, each monad can be regarded as a promonad. In this sense, arrows are generalisation of monads.

For a monad $\mathscr{T}\colon \mathbb{C} \to \mathbb{C}$ in **Cat**, there exists a category $\mathscr{E}\!\mathscr{M}(\mathscr{T})$ called the *Eilenberg–Moore category* of $\mathscr{T}$. It satisfies the following property (Street (1972)):

$$\mathbf{Cat}(\mathbb{D}, \mathscr{E}\!\mathscr{M}(\mathscr{T})) \cong \mathscr{E}\!\mathscr{M}(\mathbf{Cat}(\mathbb{D}, \mathscr{T})) \tag{2.2}$$

where $\mathbf{Cat}(\mathbb{D}, \mathscr{T})\colon \mathbf{Cat}(\mathbb{D}, \mathbb{C}) \to \mathbf{Cat}(\mathbb{D}, \mathbb{C})$ on the right-hand side is a monad on the functor category $\mathbf{Cat}(\mathbb{D}, \mathbb{C})$ defined by $\mathbf{Cat}(\mathbb{D}, \mathscr{T})(F) = \mathscr{T} \circ F$. The category $\mathscr{E}\!\mathscr{M}(\mathbf{Cat}(\mathbb{D}, \mathscr{T}))$ is the Eilenberg–Moore category of the monad $\mathbf{Cat}(\mathbb{D}, \mathscr{T})$.
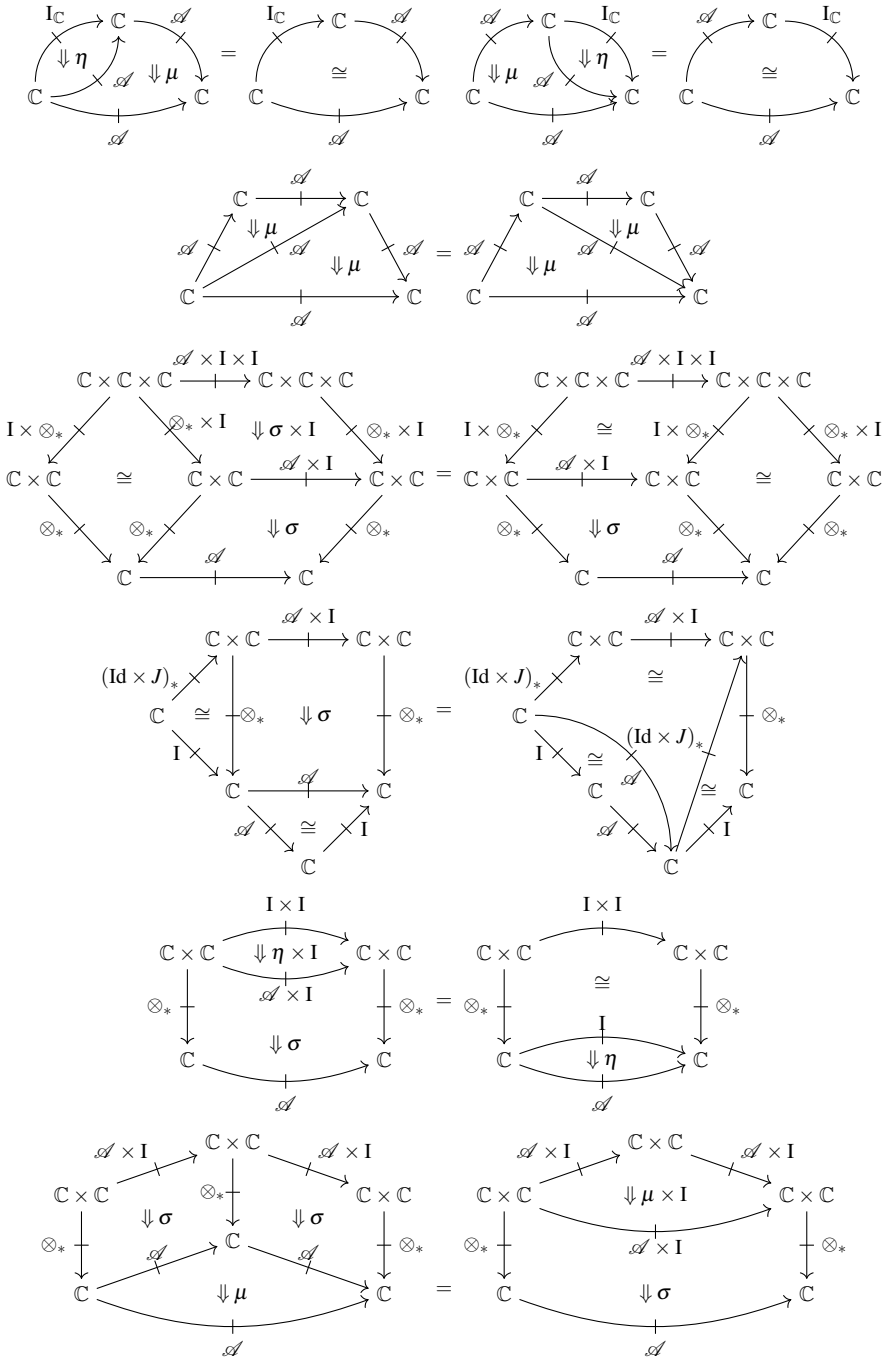
Fig. 1. Axioms for a strong promonad

For a promonad $\mathscr{A} : \mathbb{C} \nrightarrow \mathbb{C}$ in **Prof**, there exists a category $\mathscr{EM}(\mathscr{A})$ (Wood (1985)) that satisfies

$$\mathbf{Prof}(\mathbb{D}, \mathscr{EM}(\mathscr{A})) \simeq \mathscr{EM}(\mathbf{Prof}(\mathbb{D}, \mathscr{A})) \tag{2.3}$$

where $\mathbf{Prof}(\mathbb{D}, \mathscr{A}) : \mathbf{Prof}(\mathbb{D}, \mathbb{C}) \to \mathbf{Prof}(\mathbb{D}, \mathbb{C})$ on the right-hand side is a monad on the profunctor category $\mathbf{Prof}(\mathbb{D}, \mathbb{C})$ defined by $\mathbf{Prof}(\mathbb{D}, \mathscr{A})(F) = \mathscr{A} \circ F$.

Definition 2.5 (the Eilenberg-Moore category of a promonad, Wood (1985)). Let $\mathscr{A} : \mathbb{C} \nrightarrow \mathbb{C}$ be a promonad in **Prof**. The *Eilenberg-Moore category* $\mathscr{EM}(\mathscr{A})$ of $\mathscr{A}$ is defined as follows:

- $\mathrm{Ob}(\mathscr{EM}(\mathscr{A})) = \mathrm{Ob}(\mathbb{C})$.
- For $A, B \in \mathrm{Ob}(\mathscr{EM}(\mathscr{A}))$, $\mathscr{EM}(\mathscr{A})(A, B) = \mathscr{A}(A, B)$.
- For $A \in \mathrm{Ob}(\mathscr{EM}(\mathscr{A}))$, the identity on A is $\eta_{A,A}^{\mathscr{A}}(\mathrm{id}_A) \in \mathscr{A}(A, A)$.
- For $a \in \mathscr{EM}(\mathscr{A})(A, B)$ and $b \in \mathscr{EM}(\mathscr{A})(B, C)$, the composition $b \circ a$ is $\mu_{A,B,C}^{\mathscr{A}}(a, b)$.

There is an identity-on-object functor $J : \mathbb{C} \to \mathscr{EM}(\mathscr{A})$ defined by $\eta_{A,B}^{\mathscr{A}} : \mathbb{C}(A, B) \to \mathscr{A}(A, B)$. The functor J induces an adjunction $J_* \dashv J^*$ in **Prof**:

$$\mathbb{C} \underset{J^*}{\overset{J_*}{\underset{\longleftarrow}{\overset{\longrightarrow}{\dashv}}}} \mathscr{EM}(\mathscr{A}) \ .$$

The promonad $\mathscr{A}$ coincides with the composition of J_* and J^* i.e., $\mathscr{A} \cong J^* \circ J_*$.

2.3 Size issues

An arrow corresponds to a strong promonad, but when trying to interpret a programming language with, for example, **Set**, one faces a problem of the size of a set. That is, an endoprofunctor $\mathscr{A} : \mathbf{Set} \nrightarrow \mathbf{Set}$ must be a functor $\mathscr{A} : \mathbf{Set}^{\mathrm{op}} \times \mathbf{Set} \to \mathbf{Ens}$, not a functor $\mathbf{Set}^{\mathrm{op}} \times \mathbf{Set} \to \mathbf{Set}$, because the composition of profunctors is defined by a coend. Hence, the interpretation $\mathscr{A}(\llbracket A \rrbracket, \llbracket B \rrbracket)$ of an arrow type $A \rightsquigarrow B$ (in a Haskell-like notation, $\mathtt{Ar}\ A\ B$, where $\mathtt{Ar}$ is an instance of the class $\mathtt{Arrow}$) is a class, not a set. Asada (2010) introduced $\mathbb{V}$-*small profunctors* in $\mathbb{V}'$-**Prof** to solve the size problem.

Readers who are not concerned about size issues may skip the rest of this section. For the definition of enriched categorical notions such as $\mathbb{V}$-categories and $\mathbb{V}$-functors, see Appendix 2.3 and (Kelly, 1982, Section 1).

Definition 2.6 ($\mathbb{V}$-profunctors). Let $\mathbb{V}$ be a symmetric monoidal category and $\mathbb{C}$ and $\mathbb{D}$ be $\mathbb{V}$-categories. A $\mathbb{V}$-*profunctor* F from $\mathbb{C}$ to $\mathbb{D}$, written $F : \mathbb{C} \nrightarrow \mathbb{D}$, is a $\mathbb{V}$-functor $F : \mathbb{D}^{\mathrm{op}} \otimes \mathbb{C} \to \mathbb{V}$. A 2-*cell* α from $\mathbb{V}$-profunctors $F : \mathbb{C} \nrightarrow \mathbb{D}$ to $G : \mathbb{C} \to \mathbb{D}$, written $\alpha : F \Rightarrow G$ is a $\mathbb{V}$-natural transformation between the $\mathbb{V}$-functors. For two $\mathbb{V}$-profunctors $F : \mathbb{C} \nrightarrow \mathbb{D}$ and $G : \mathbb{D} \nrightarrow \mathbb{E}$, their composite $G \circ F : \mathbb{C} \nrightarrow \mathbb{E}$ is defined by the following coend in $\mathbb{V}$:

$$(G \circ F)(E, C) = \int^{D \in \mathbb{D}} G(E, D) \otimes F(D, C).$$

The identity $\mathbb{V}$-profunctor $I_\mathbb{C} : \mathbb{C} \nrightarrow \mathbb{C}$ is defined by

$$I_\mathbb{C}(C, D) = \mathbb{C}(C, D).$$

The collection of small $\mathbb{V}$-categories, $\mathbb{V}$-profunctors and 2-cells forms a bicategory $\mathbb{V}$-**Prof**. The bicategory **Prof** is the special case of $\mathbb{V}$-**Prof** where $\mathbb{V} = (\mathbf{Set}, \times, 1)$.

Definition 2.7 (Asada (2010)). Let $\mathbb{V}$ be a symmetric monoidal category, $\mathbb{V}'$ be a sufficiently cocomplete symmetric monoidal closed category, and $J : \mathbb{V} \to \mathbb{V}'$ be a symmetric strong monoidal fully faithful functor. A $\mathbb{V}'$-profunctor $F : \mathbb{C} \nrightarrow \mathbb{D}$ is $\mathbb{V}$-*small* if there exists a $\mathbb{V}'$-functor $F^\circ : \mathbb{D}^{op} \otimes \mathbb{C} \to \mathbb{V}$ such that $J \circ F^\circ = F$:

$$
\begin{array}{ccc}
\mathbb{D}^{op} \otimes \mathbb{C} & \xrightarrow{\;F\;} & \mathbb{V}' \\
 & \searrow^{F^\circ} & \uparrow^{J} \\
 & & \mathbb{V}
\end{array}
\qquad (2.4)
$$

Let $J : \mathbf{Set} \to \mathbf{Ens}$ be the embedding. A **Set**-small strong **Ens**-promonad $\mathscr{A} : \mathbf{Set} \nrightarrow \mathbf{Set}$ is an **Ens**-functor $\mathscr{A} : \mathbf{Set}^{op} \times \mathbf{Set} \to \mathbf{Ens}$ with 2-cells η, μ and σ which make $\mathscr{A}$ a strong **Ens**-promonad, and an **Ens**-functor $\mathscr{A}^\circ : \mathbf{Set}^{op} \times \mathbf{Set} \to \mathbf{Set}$ satisfying

$$
\begin{array}{ccc}
\mathbf{Set}^{op} \times \mathbf{Set} & \xrightarrow{\;\mathscr{A}\;} & \mathbf{Ens} \\
 & \searrow^{\mathscr{A}^\circ} & \uparrow^{J} \\
 & & \mathbf{Set}
\end{array}
$$

If a suitable **Set**-small strong **Ens**-promonad $\mathscr{A} : \mathbf{Set} \nrightarrow \mathbf{Set}$ exists, then we can define the interpretation of an arrow type $A \rightsquigarrow B$ (which will be introduced in Section 4.1) by $[\![A \rightsquigarrow B]\!] = \mathscr{A}^\circ([\![A]\!], [\![B]\!])$.

We will introduce an arrow calculus with operations and handlers in Section 4.1 and define models of the calculus as appropriate small promonads (Definition 5.1). As a concrete model, we will construct a **Set**-small strong **Ens**-promonad $\mathscr{A}$ which have sufficient structures to interpret the arrow calculus in Section 5.2.

3 Algebras of arrows

For an ordinary monad $\mathscr{T} : \mathbb{C} \to \mathbb{C}$, a $\mathscr{T}$-algebra is a pair $\langle A, a \rangle$ of an object $A \in \mathbb{C}$ and a morphism $a : \mathscr{T}A \to A$ in $\mathbb{C}$ satisfying appropriate axioms, that is an object of the Eilenberg-Moore category $\mathscr{EM}(\mathscr{T})$. An ordinary effect handler is interpreted as a homomorphism between two $\mathscr{T}$-algebras.

What is an $\mathscr{A}$-algebra for a promonad $\mathscr{A} : \mathbb{C} \nrightarrow \mathbb{C}$? We answer this question in the next section (Section 3.1) from a 2-categorical point of view. We also discuss the universality of a free $\mathscr{A}$-algebra in Section 3.2. A handler for arrows is interpreted as a homomorphism between $\mathscr{A}$-algebras in the sense of Section 3.1.

3.1 Algebras of promonads

Let **1** be the category with a single object and a single morphism (the identity). For a monad $\mathscr{T}\colon \mathbb{C}\to\mathbb{C}$, a $\mathscr{T}$-algebra is a functor $\mathbf{1}\to\mathscr{EM}(\mathscr{T})$ in **Cat**. Similarly, for a promonad $\mathscr{A}\colon \mathbb{C}\nrightarrow\mathbb{C}$, we call a profunctor $\mathbf{1}\nrightarrow\mathscr{EM}(\mathscr{A})$ in **Prof** an $\mathscr{A}$-*algebra* where $\mathscr{EM}(\mathscr{A})$ is the Eilenberg-Moore category of $\mathscr{A}$ (Definition 2.5).

By (2.2), a $\mathscr{T}$-algebra $a\colon \mathbf{1}\to\mathscr{EM}(\mathscr{T})$ corresponds to a **Cat**$(\mathbf{1},\mathscr{T})$-algebra α, that is the following equation holds:

where $A\in\mathbb{C}$, $\ulcorner A\urcorner\colon \mathbf{1}\to\mathbb{C}$ is the constant functor, and the 2-cell ξ is defined by $\xi_f = f\colon \mathscr{T}U(f)\to Uf$ for a $\mathscr{T}$-algebra $f\colon \mathscr{T}A\to A$. Hence, specifying a $\mathscr{T}$-algebra $a\colon \mathbf{1}\to\mathscr{EM}(\mathscr{T})$ is equivalent to specifying a morphism $\alpha\colon \mathscr{T}A\to A$ satisfying ordinary equations for a $\mathscr{T}$-algebra.

Similarly, by (2.3), a $\mathscr{A}$-algebra $a\colon \mathbf{1}\nrightarrow\mathscr{EM}(\mathscr{A})$ corresponds to a **Prof**$(\mathbf{1},\mathscr{A})$-algebra α up to isomorphism, that is

The **Prof**$(\mathbf{1},\mathscr{A})$-algebra α satisfies the following equations.

$$(3.1)$$

$$(3.2)$$

Note that (3.1) is equivalent to $\alpha_{X,Y}(\eta_{X,Y}(f),k)=(Gf)(k)$ for any $X,Y\in\mathbb{C}$, $f\in\mathbb{C}(X,Y)$ and $k\in GY$, and (3.2) is equivalent to $\alpha_{X,Z}(\mu_{X,Y,Z}(a,b),k)=\alpha_{X,Y}(a,\alpha_{Y,Z}(b,k))$ for any $X,Y,Z\in\mathbb{C}$, $a\in\mathscr{A}(X,Y)$, $b\in\mathscr{A}(Y,Z)$ and $k\in GZ$.

Hence, specifying an $\mathscr{A}$-algebra $a\colon \mathbf{1}\nrightarrow \mathscr{E}\mathscr{M}(\mathscr{A})$ is equivalent to specifying a presheaf $G\colon \mathbb{C}^{\mathrm{op}}\to\mathbf{Set}$ and a 2-cell $\alpha\colon \mathscr{A}\circ G\Rightarrow G$ in $\mathbf{Prof}$ satisfying (3.1) and (3.2). We call such a pair $\langle G,\alpha\rangle$ also an *algebra*.

Definition 3.1 (algebras and homomorphisms). Let $\mathscr{A}\colon \mathbb{C}\nrightarrow\mathbb{C}$ be a promonad. An *algebra* for $\mathscr{A}$ is a pair $\langle G,\alpha\rangle$ of a presheaf $G\colon \mathbb{C}^{\mathrm{op}}\to\mathbf{Set}$ and a 2-cell $\alpha\colon \mathscr{A}\circ G\Rightarrow G$ in $\mathbf{Prof}$. A homomorphism h from an algebra $\langle G,\alpha\rangle$ to an algebra $\langle H,\beta\rangle$, written $h\colon \langle G,\alpha\rangle\to \langle H,\beta\rangle$, is a 2-cell $h\colon G\Rightarrow H$ in $\mathbf{Prof}$ such that $\beta\circ(\mathscr{A}h)=h\circ\alpha$.

In other words, a homomorphism $h\colon \langle G,\alpha\rangle\to\langle H,\beta\rangle$ is a natural transformation from G to H that makes the following diagram commute:

$$
\begin{array}{ccc}
\mathscr{A}(X,Y)\times GY & \xrightarrow{\;\alpha_{X,Y}\;} & GX \\
{\scriptstyle \mathscr{A}(X,Y)\times h_Y}\big\downarrow & & \big\downarrow{\scriptstyle h_X} \\
\mathscr{A}(X,Y)\times HY & \xrightarrow[\;\beta_{X,y}\;]{} & HX
\end{array}
$$

for any $X,Y\in\mathbb{C}$.

3.2 Arrow handlers as homomorphisms between algebras

A free $\mathscr{A}$-algebra for a promonad $\mathscr{A}$ has a universal property, which is similar to the universal property for a free $\mathscr{T}$-algebra for an ordinary monad $\mathscr{T}$. Theorem 3.2 shows the universality of a free $\mathscr{A}$-algebra. An effect handler for arrows is interpreted by the homomorphism induced by the universality.

Theorem 3.2. *Let $\mathscr{A}$ be a promonad on $\mathbb{C}$, $C\in\mathbb{C}$, $G\colon \mathbf{1}\nrightarrow\mathbb{C}$ be a profunctor, that is a presheaf on $\mathbb{C}$, and $\langle G,\alpha\colon \mathscr{A}\circ G\Rightarrow G\rangle$ be an $\mathscr{A}$-algebra. If $\phi\colon \mathbb{C}(-,C)\to G$ is a morphism between presheaves, then there exists a unique homomorphism*

$$\phi^{\dagger}\colon \left\langle \mathscr{A}(-,C),\mu^{\mathscr{A}}_{-,C}\colon \mathscr{A}\circ\mathscr{A}(-,C)\Rightarrow\mathscr{A}(-,C)\right\rangle\to\langle G,\alpha\colon \mathscr{A}\circ G\Rightarrow G\rangle$$

between $\mathscr{A}$-algebras that makes the following diagram commute up to isomorphism:

$$
\begin{array}{ccc}
\mathbb{C}(-,C) & \xrightarrow{\;\eta^{\mathscr{A}}_{-,C}\;} & \mathscr{A}(-,C) \\
 & {\scriptstyle \phi}\searrow & \big\downarrow{\scriptstyle \phi^{\dagger}} \\
 & & G
\end{array}\quad .
\tag{3.3}
$$

Proof We define $\phi^{\dagger}=\alpha\circ(\mathscr{A}\phi)\circ\rho^{-1}\colon \mathscr{A}(-,C)\to G$ where ρ is the right unitor:

$$\mathscr{A}(-,C)\xrightarrow[\;\rho^{-1}\;]{\cong}(\mathscr{A}\circ \mathrm{I}_{\mathbb{C}})(-,C)=\mathscr{A}\circ(\mathbb{C}(-,C))\xrightarrow{\;\mathscr{A}\phi\;}\mathscr{A}\circ G\xrightarrow{\;\alpha\;}G. \tag{3.4}$$

We check that $\phi^\dagger$ makes the diagram (3.3) commute. In the following diagram, the bottom triangle commutes by (3.1).

$$
\begin{array}{ccc}
\mathbb{C}(-,C) & \xrightarrow{\eta^{\mathscr{A}}_{-,C}} & \mathscr{A}(-,C) \\
\phi \downarrow & & \downarrow \cong \\
& & \mathscr{A} \circ \mathbb{C}(-,C) \\
& & \downarrow \mathscr{A}\phi \\
G & & \mathscr{A} \circ G \\
\cong \downarrow \quad \nearrow^{\eta^{\mathscr{A}}G} & & \downarrow \alpha \\
I_{\mathbb{C}} \circ G & \xrightarrow{\cong} & G
\end{array}
$$

To show that the above square commutes, it suffices to show the commutativity of the following diagram for each object A of $\mathbb{C}$:

$$
\begin{array}{ccc}
\mathbb{C}(A,C) & \xrightarrow{\eta^{\mathscr{A}}_{A,C}} & \mathscr{A}(A,C) \\
\phi_A \downarrow & & \downarrow \cong \\
GA & & \int^{B\in\mathbb{C}} \mathscr{A}(A,B) \times \mathbb{C}(B,C) \\
\cong \downarrow & & \downarrow \mathscr{A}\phi \\
\int^{B\in\mathbb{C}} \mathbb{C}(A,B) \times GB & \xrightarrow{\eta^{\mathscr{A}}G} & \int^{B\in\mathbb{C}} \mathscr{A}(A,B) \times GB
\end{array}
$$

$$
\begin{array}{ccc}
(f:A\to C) & \xmapsto{\eta^{\mathscr{A}}_{A,C}} & \eta^{\mathscr{A}}_{A,C}(f) \\
\phi_A \downarrow & & \downarrow \cong \\
\phi_A(f) & & [\eta^{\mathscr{A}}_{A,C}(f), \mathrm{id}_C] \\
\cong \downarrow & & \downarrow \mathscr{A}\phi \\
[\mathrm{id}_A, \phi_A(f)] & \xmapsto{\eta^{\mathscr{A}}G} [\eta^{\mathscr{A}}_{A,A}(\mathrm{id}_A), \phi_A(f)] & [\eta^{\mathscr{A}}_{A,C}(f), \phi_C(\mathrm{id}_C)]
\end{array}
$$

where $[x,y]$ denotes the equivalence class of a pair (x,y), see the proof of Proposition 1.3 in Appendix 1. By chasing the diagram, it is enough to show $[\eta^{\mathscr{A}}_{A,A}(\mathrm{id}_A), \phi_A(f)] = [\eta^{\mathscr{A}}_{A,C}(f), \phi_C(\mathrm{id}_C)]$ in $\int^{B\in\mathbb{C}} \mathscr{A}(A,B) \times GB$ for each $A \in \mathbb{C}$ and $f: A \to C$. Consider the following commutative diagram.

$$
\begin{array}{ccc}
\mathscr{A}(A,A) \times GC & \xrightarrow{\mathscr{A}(A,A)\times Gf} & \mathscr{A}(A,A) \times GA \\
\mathscr{A}(A,f)\times GC \downarrow & & \downarrow \omega_A \\
\mathscr{A}(A,C) \times GC & \xrightarrow{\omega_C} & \int^{B\in\mathbb{C}} \mathscr{A}(A,B) \times GB
\end{array}
$$

For $\langle \eta(\mathrm{id}_A), \phi_C(\mathrm{id}_C)\rangle \in \mathscr{A}(A,A) \times GC$, we have

$$\omega_A((\mathscr{A}(A,A) \times G(f))\langle \eta(\mathrm{id}_A), \phi_C(\mathrm{id}_C)\rangle)$$
$$= \omega_A(\langle \eta(\mathrm{id}_A), (Gf)(\phi_C(\mathrm{id}_C))\rangle)$$
$$= \omega_A(\langle \eta(\mathrm{id}_A), \phi_C(f)\rangle) \qquad \text{(by the naturality of } \phi)$$
$$= [\eta(\mathrm{id}_A), \phi_C(f)]$$

and

$$\omega_C((\mathscr{A}(A,f) \times G(C))\langle \eta(\mathrm{id}_A), \phi_C(\mathrm{id}_C)\rangle)$$
$$= \omega_C(\langle \mathscr{A}(A,f)(\eta(\mathrm{id}_A)), \phi_C(\mathrm{id}_C)\rangle)$$
$$= \omega_C(\langle \eta(f), \phi_C(\mathrm{id}_C)\rangle) \qquad \text{(by the naturality of } \eta_{A,-})$$
$$= [\eta(f), \phi_C(\mathrm{id}_C)].$$

Hence, we have $[\eta(\mathrm{id}_A), \phi(f)] = [\eta(f), \phi(\mathrm{id}_C)]$.

Next, we show that $\phi^\dagger$ is a homomorphism $\mu^{\mathscr{A}}_{-,C} \to \alpha$ of $\mathscr{A}$-algebras. It suffices to show the commutativity of the left and right square in the following diagram. The right square commutes because α is an $\mathscr{A}$-algebra (3.2), and the left square commutes by the naturality of μ and ϕ and some calculation similar to the above argument.

By the Yoneda lemma, giving a morphism $\phi \colon \mathbb{C}(-,C) \to G$ between presheaves is equivalent to giving an element $p \in GC$. In the proof of Theorem 3.2, $\phi^\dagger$ is defined by (3.4). Hence, for $a \in \mathscr{A}(A,C)$, $\phi^\dagger(a)$ is calculated to be $\alpha([a,p])$. In summary, we obtain the following corollary.

Corollary 3.3. *Let $\mathscr{A}$ be a promonad on $\mathbb{C}$, $C \in \mathbb{C}$, G be a presheaf on $\mathbb{C}$, and $\langle G, \alpha\rangle$ be an $\mathscr{A}$-algebra. For an element $p \in GC$, there is a homomorphism*

$$h \colon \left\langle \mathscr{A}(-,C), \mu^{\mathscr{A}}_{-,C} \colon \mathscr{A} \circ \mathscr{A}(-,C) \Rightarrow \mathscr{A}(-,C)\right\rangle \to \langle G, \alpha \colon \mathscr{A} \circ G \Rightarrow G\rangle$$

satisfying $h_A(a) = \alpha([a,p])$ for any $A \in \mathbb{C}$ and $a \in \mathscr{A}(A,C)$, and $h(\eta^{\mathscr{A}}_{A,C}(f)) = (Gf)(p)$ for any $A \in \mathbb{C}$ and $f \colon A \to C$ in $\mathbb{C}$.

When G in Corollary 3.3 is a presheaf $\mathscr{A}'(-,D)$ for another promonad $\mathscr{A}' \colon \mathbb{C} \nrightarrow \mathbb{C}$ and an object $D \in \mathbb{C}$, we obtain the following corollary.

Corollary 3.4. *Let $\mathscr{A}$ and $\mathscr{A}'$ be promonads on $\mathbb{C}$, $D \in \mathbb{C}$, and α be a family $(\alpha_{A,B} \colon \mathscr{A}(A, B) \times \mathscr{A}'(B, D) \to \mathscr{A}'(A, D))_{A,B \in \mathbb{C}}$ of maps which is natural in A and extranatural in B and satisfies (3.1) and (3.2) for $G = \mathscr{A}'(-, D)$. For an element $p \in \mathscr{A}'(C, D)$, there is a homomorphism*

$$h \colon \left\langle \mathscr{A}(-, C), \mu^{\mathscr{A}} \right\rangle \to \left\langle \mathscr{A}'(-, D), \alpha \right\rangle$$

such that $h(a) = \alpha_{A,C}(a, p)$ for any $A \in \mathbb{C}$ and $a \in \mathscr{A}(A, C)$, and $h_A(\eta^{\mathscr{A}}_{A,C}(f)) = \mathscr{A}'(f, D)(p)$ for any $A \in \mathbb{C}$ and $f \colon A \to C$ in $\mathbb{C}$.

We use Corollary 3.4 to interpret handlers for arrows.

Remark 3.5. Note that we can prove Theorem 3.2 in another way. From the promonad $\mathscr{A} \colon \mathbb{C} \nrightarrow \mathbb{C}$, we can obtain a cocontinuous monad $\widetilde{\mathscr{A}} \colon [\mathbb{C}^{\mathrm{op}}, \mathbf{Set}] \to [\mathbb{C}^{\mathrm{op}}, \mathbf{Set}]$ by the following construction. Then, using the universality of free $\widetilde{\mathscr{A}}$-algebra, Theorem 3.2 is proved.

$$
\cfrac{
  \cfrac{
    \cfrac{
      F \colon \mathbb{C} \nrightarrow \mathbb{D}
    }{
      F \colon \mathbb{D}^{\mathrm{op}} \times \mathbb{C} \to \mathbf{Set}
    }
  }{
    \overline{F} \colon \mathbb{C} \to [\mathbb{D}^{\mathrm{op}}, \mathbf{Set}]
  } \text{ currying}
}{
  \widetilde{F} \colon [\mathbb{C}^{\mathrm{op}}, \mathbf{Set}] \to [\mathbb{D}^{\mathrm{op}}, \mathbf{Set}] \colon \text{cocontinuous}
} \text{ left Kan extension}
$$

4 The arrow calculus with operations and handlers

The *arrow calculus* was introduced by Lindley et al. (2010, 2011). We add operations and handlers to their calculus.

4.1 Syntax and typing rules

Let *BType* be a set of base types, and Σ be a set of operations. We assume that two base types γ (*coarity*) and δ (*arity*) are assigned for each operation op $\in \Sigma$. We write op$: \gamma \to \delta$ when the coarity and arity of op are γ and δ, respectively. The syntax is shown in Figure 2. The difference from the original arrow calculus (Lindley et al. (2010, 2011)) is the addition of op(M), **handle** R **with** H to the commands and handlers H.

We call a type A *primitive* (written $\Phi(A)$) if A is constructed only by β, $\times$ and $\to$. Formally, $\Phi(A)$ is defined as follows.

$$\frac{\beta \in BType}{\Phi(\beta)} \qquad \frac{\Phi(A) \quad \Phi(B)}{\Phi(A \times B)} \qquad \frac{\Phi(A) \quad \Phi(B)}{\Phi(A \to B)}$$

The typing rules for the arrow calculus are shown in Figure 3. The notion of primitive types is used in the rule T-HANDLER.

A type $A \rightsquigarrow B$ is that of effectful computation such that the type of its inputs is A and the type of its outputs is B.

As we described in Section 1.2.3, a term M is a pure function and a command P is an effectful computation. The command $\lfloor M \rfloor$ is a pure computation, which corresponds to

Types	$A, B, C, D ::= \beta \mid A \times B \mid A \to B \mid A \rightsquigarrow B$ where $\beta \in BType$
Environments	$\Gamma, \Delta ::= \diamond \mid \Gamma, x : A$
Terms	$M, N, L ::= x \mid \langle M, N \rangle \mid \mathbf{fst}\, M \mid \mathbf{snd}\, M \mid \lambda x : A.M \mid MN \mid \lambda^{\bullet} x : A.P$
Values	$V, W ::= x \mid \langle V, W \rangle \mid \lambda x : A.M \mid \lambda^{\bullet} x : A.P$
Commands	$P, Q, R ::= \lfloor M \rfloor \mid \mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q \mid L \bullet M \mid \mathsf{op}(M) \mid \mathbf{handle}\, R \,\mathbf{with}\, H$
Handlers	$H ::= \{\, \mathring{,}\, x : C \mapsto P \} \cup \{ \mathsf{op}, k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \mapsto Q_{\mathsf{op}} \}_{\mathsf{op} \in \Sigma}$

Fig. 2. The syntax of the arrow calculus with operations and handlers

The derivation of $\boxed{\Gamma \vdash M : A}$

$$\frac{}{\Gamma, x : A \vdash x : A}\ \text{T-VAR} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M : A \to B}\ \text{T-ABS}$$

$$\frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ \text{T-APP} \qquad \frac{\Gamma \,\mathring{,}\, x : A \vdash P : B}{\Gamma \vdash \lambda^{\bullet} x : A.P : A \rightsquigarrow B}\ \text{T-CABS}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \times B}\ \text{T-PAIR} \qquad \frac{\Gamma \vdash M : A \times B}{\Gamma \vdash \mathbf{fst}\, M : A}\ \text{T-FST} \qquad \frac{\Gamma \vdash M : A \times B}{\Gamma \vdash \mathbf{snd}\, M : B}\ \text{T-SND}$$

The derivation of $\boxed{\Gamma \,\mathring{,}\, \Delta \vdash P \,!\, A}$

$$\frac{\Gamma, \Delta \vdash M : A}{\Gamma \,\mathring{,}\, \Delta \vdash \lfloor M \rfloor \,!\, A}\ \text{T-PURE} \qquad \frac{\Gamma \vdash L : A \rightsquigarrow B \quad \Gamma, \Delta \vdash M : A}{\Gamma \,\mathring{,}\, \Delta \vdash L \bullet M \,!\, B}\ \text{T-CAPP}$$

$$\frac{\Gamma \,\mathring{,}\, \Delta \vdash P \,!\, A \quad \Gamma \,\mathring{,}\, x : A, \Delta \vdash Q \,!\, B}{\Gamma \,\mathring{,}\, \Delta \vdash \mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q \,!\, B}\ \text{T-LET} \qquad \frac{\mathsf{op} : \gamma \to \delta \in \Sigma \quad \Gamma, \Delta \vdash M : \gamma}{\Gamma \,\mathring{,}\, \Delta \vdash \mathsf{op}(M) \,!\, \delta}\ \text{T-OP}$$

$$\frac{\Gamma \,\mathring{,}\, \Delta \vdash P \,!\, C \quad \vdash H : C \Rightarrow D}{\Gamma \,\mathring{,}\, \Delta \vdash \mathbf{handle}\, P \,\mathbf{with}\, H \,!\, D}\ \text{T-HANDLE}$$

The derivation of $\boxed{\vdash H : C \Rightarrow D}$

$$\frac{\Phi(C) \wedge \Phi(D) \quad \diamond \,\mathring{,}\, x : C \vdash P \,!\, D \quad (k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \vdash Q_{\mathsf{op}} \,!\, D)_{(\mathsf{op} : \gamma \to \delta) \in \Sigma}}{\vdash \{\, \mathring{,}\, x : C \mapsto P \} \cup \{ \mathsf{op}, k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \mapsto Q_{\mathsf{op}} \}_{\mathsf{op} \in \Sigma} : C \Rightarrow D}\ \text{T-HANDLER}$$

Fig. 3. Typing rules for the arrow calculus with operations and handlers

$\mathtt{arr}$ in Haskell. The command $L \bullet M$ is a command which invokes an arrow L with an input M. It corresponds to $(\ggg)$ in Haskell. The command $\mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q$ is a sequential composition of commands P and Q, which corresponds to $(\ggg)$ and $\mathtt{first}$ in Haskell. The command $\mathsf{op}(M)$ is an operation invocation with an input M.

A judgement $\Gamma \vdash M : A$ for a term means that the term M is a pure function from Γ to A. A judgement $\Gamma \vdash P \,!\, A$ for a command means that the command P is a computation such that the type of its inputs is Δ and the type of its outputs is A under the context Γ.

In the construction of handlers:

$$H = \{\, \mathring{,}\, x : C \mapsto P \} \cup \{ \mathsf{op}, k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \mapsto Q_{\mathsf{op}} \}_{\mathsf{op} \in \Sigma}, \qquad (4.1)$$

the command P corresponds to the $p \in \mathcal{A}'(C, D)$ in Corollary 3.4, and the family $\{Q_{\mathsf{op}}\}_{\mathsf{op} \in \Sigma}$ of commands determines an algebraic structure α in Corollary 3.4.

Handlers for arrows is very similar to handlers for ordinary monads. The difference is that, in the handler (4.1), $x : C$ and $z : \gamma$ are not ordinary contexts, but inputs of the effectful computation P and Q_{op}, respectively.

4.2 Operational semantics

We present the reduction rules for closed terms $\diamond \vdash M : A$ and closed commands $\diamond \mathbin{\mathring{,}} \diamond \vdash P \,!\, A$. We define two kinds of evaluation contexts as follows.

$$\mathcal{E} ::= [-] \mid \mathcal{E}N \mid V\mathcal{E} \mid \mathcal{E} \bullet N \mid (\lambda^{\bullet}x.P) \bullet \mathcal{E} \mid \lfloor \mathcal{E} \rfloor \mid \mathsf{op}(\mathcal{E}) \mid \mathbf{fst}\,\mathcal{E} \mid \mathbf{snd}\,\mathcal{E} \mid \langle \mathcal{E}, M \rangle \mid \langle V, \mathcal{E} \rangle$$
$$\mathcal{F} ::= [-] \mid \mathbf{let}\, x \Leftarrow \mathcal{F} \,\mathbf{in}\, Q$$

We put a term in the hole of a context $\mathcal{E}[-]$ and a command in the hole of a context $\mathcal{F}[-]$. The reduction relation are shown in Figure 4.
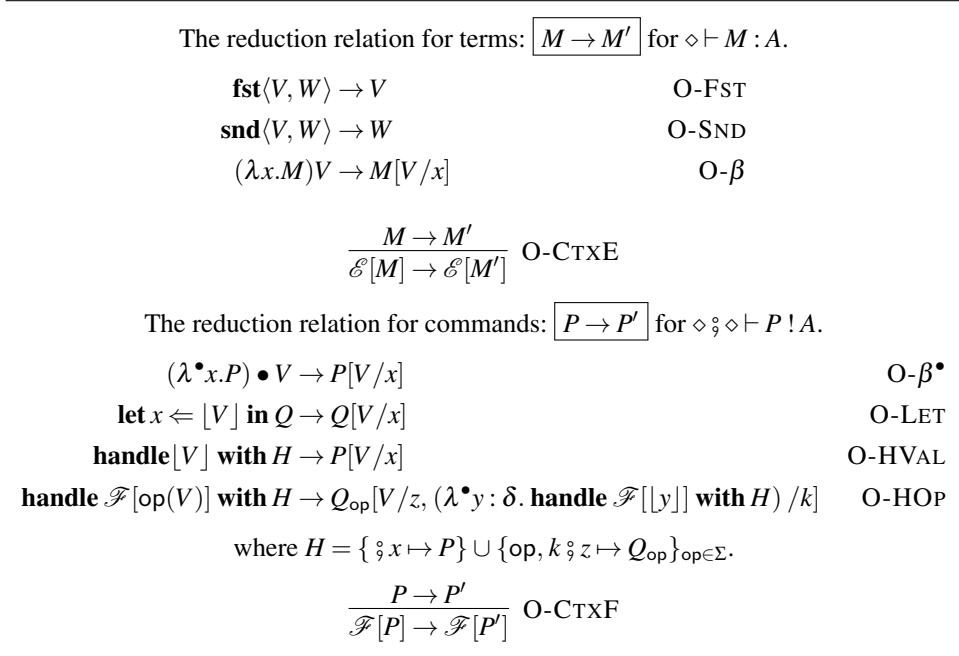

The reduction relation for terms: $\boxed{M \to M'}$ for $\diamond \vdash M : A$.

$$\mathbf{fst}\langle V, W \rangle \to V \qquad\qquad \text{O-Fst}$$
$$\mathbf{snd}\langle V, W \rangle \to W \qquad\qquad \text{O-Snd}$$
$$(\lambda x.M)V \to M[V/x] \qquad\qquad \text{O-}\beta$$

$$\frac{M \to M'}{\mathcal{E}[M] \to \mathcal{E}[M']} \ \text{O-CtxE}$$

The reduction relation for commands: $\boxed{P \to P'}$ for $\diamond \mathbin{\mathring{,}} \diamond \vdash P \,!\, A$.

$$(\lambda^{\bullet}x.P) \bullet V \to P[V/x] \qquad\qquad \text{O-}\beta^{\bullet}$$
$$\mathbf{let}\, x \Leftarrow \lfloor V \rfloor \,\mathbf{in}\, Q \to Q[V/x] \qquad\qquad \text{O-Let}$$
$$\mathbf{handle}\lfloor V \rfloor \,\mathbf{with}\, H \to P[V/x] \qquad\qquad \text{O-Hval}$$
$$\mathbf{handle}\,\mathcal{F}[\mathsf{op}(V)] \,\mathbf{with}\, H \to Q_{\mathsf{op}}[V/z, (\lambda^{\bullet}y : \delta.\, \mathbf{handle}\,\mathcal{F}[\lfloor y \rfloor] \,\mathbf{with}\, H)\,/k] \qquad \text{O-Hop}$$

$$\text{where } H = \{\mathbin{\mathring{,}}\, x \mapsto P\} \cup \{\mathsf{op}, k \mathbin{\mathring{,}} z \mapsto Q_{\mathsf{op}}\}_{\mathsf{op} \in \Sigma}.$$

$$\frac{P \to P'}{\mathcal{F}[P] \to \mathcal{F}[P']} \ \text{O-CtxF}$$

Fig. 4. Operational semantics.

Type preservation (Proposition 4.3) and progress (Proposition 4.1) hold for the arrow calculus with operations and handlers.

Proposition 4.1 (progress). *The following hold.*

1. *For any well-typed term $\diamond \vdash M : A$, there exists a term M' such that $M \to M'$ or M is a value.*

2. *For any well-typed command $\diamond \mathbin{;} \diamond \vdash P \mathbin{!} A$, one of following holds.*

    a. *There exists a command P' such that $P \to P'$.*
    b. *$P = \lfloor V \rfloor$ for some value V.*
    c. *$P = \mathscr{F}[\mathsf{op}(V)]$ for some operation op, value V, and context $\mathscr{F}$.*

Lemma 4.2 (substitution). *Let Γ and Δ be contexts. The following hold.*

1. *If $\Gamma, x : A \vdash M : B$ and $\Gamma \vdash N : A$ are derivable, then $\Gamma \vdash M[N/x] : B$ is derivable.*
2. *If $\Gamma, x : A \mathbin{;} \Delta \vdash P \mathbin{!} B$ and $\Gamma \vdash N : A$ are derivable, then $\Gamma \mathbin{;} \Delta \vdash P[N/x] : B$ is derivable.*
3. *If $\Gamma \mathbin{;} x : A, \Delta \vdash P \mathbin{!} B$ and $\Gamma, \Delta \vdash N : A$ are derivable, then $\Gamma \mathbin{;} \Delta \vdash P[N/x] : B$ is derivable.*

Proof The proof is by induction on the derivations.   ∎

Proposition 4.3 (Type preservation). *The following hold.*

1. *For any well-typed term $\diamond \vdash M : A$, if $M \to M'$ then $\diamond \vdash M' : A$.*
2. *For any well-typed command $\diamond \mathbin{;} \diamond \vdash P \mathbin{!} A$, if $P \to P'$ then $\diamond \mathbin{;} \diamond \vdash P' : A$.*

Proof The proof is by induction on the derivation of $M \to M'$ and $P \to P'$, and Lemma 4.2.
∎

4.3 Example

In the description of handlers, clauses that just forward its input shall be omitted. For example, let $\Sigma = \{\mathsf{op}_1, \mathsf{op}_2\}$, we write

$$H = \{\mathsf{op}_1, k \mathbin{;} z \mapsto Q_1\}$$

for $H = \{ \mathbin{;} x \mapsto \lfloor x \rfloor\} \cup \{\mathsf{op}_1, k \mathbin{;} z \mapsto Q_1, \ \mathsf{op}_2, k \mathbin{;} z \mapsto \mathbf{let}\ u \Leftarrow \mathsf{op}_2(z) \ \mathbf{in}\ k \bullet u\}$.

4.3.1 Logic circuit simulation

We explain the details of the example of logic circuit simulation in Section 1.2.3. Let $BType = \{\mathrm{Bool}\}$. The base type Bool is a type for boolean values. We add the following constants to the arrow calculus.

$$\mathtt{true} : \mathrm{Bool}, \quad \mathtt{false} : \mathrm{Bool}.$$

Let Σ_{LC} be the signature of logic circuits defined in Section 1.2. We extend the arrow calculus by adding **if** M **then** N_1 **else** N_2 to terms with the following typing rule:

$$\frac{\Gamma \vdash M : \mathrm{Bool} \quad \Gamma \vdash N_1 : A \quad \Gamma \vdash N_2 : A}{\Gamma \vdash \mathbf{if}\ M\ \mathbf{then}\ N_1\ \mathbf{else}\ N_2 : A}$$

The operational semantics is extended by adding the following evaluation context and reduction relations.

$$\mathscr{E} ::= \cdots \mid \mathbf{if}\ \mathscr{E}\ \mathbf{then}\ N_1\ \mathbf{else}\ N_2,$$

$$\textbf{if } \texttt{true} \textbf{ then } N_1 \textbf{ else } N_2 \to N_1,$$
$$\textbf{if } \texttt{false} \textbf{ then } N_1 \textbf{ else } N_2 \to N_2.$$

Let Q_{NAND} be $\textbf{let } u \Leftarrow \text{AND}(z) \textbf{ in let } v \Leftarrow \text{NOT}(u) \textbf{ in } k \bullet v$ and Q_{OR} be $\textbf{let } u \Leftarrow \text{NOT}(\textbf{fst } z) \textbf{ in let } v \Leftarrow \text{NOT}(\textbf{snd } z) \textbf{ in let } w \Leftarrow \text{AND}(\langle u, v \rangle) \textbf{ in } k \bullet w$. The handler $H_1' = \{\text{NAND}, k \,\fatsemi\, z \mapsto Q_{\text{NAND}}, \ \text{OR}, k \,\fatsemi\, z \mapsto Q_{\text{OR}}\}$ and H_2' defined in Section 1.2.3 is well-typed: $\vdash H_1' : \text{Bool} \Rightarrow \text{Bool}$ and $\vdash H_2' : \text{Bool} \Rightarrow \text{Bool}$.

We check the derivation of $\vdash H_1' : \text{Bool} \Rightarrow \text{Bool}$. Let $\Gamma = k : \text{Bool} \rightsquigarrow \text{Bool}$, $\Delta = z : \text{Bool} \times \text{Bool}$ and $\Delta' = (\Delta, u : \text{Bool})$. The derivation of $\Gamma \,\fatsemi\, \Delta \vdash Q_{\text{NAND}} \,!\, \text{Bool}$ is as follows.

$$
\cfrac{
  \cfrac{
    \text{AND} : \text{Bool} \times \text{Bool} \rightarrow \text{Bool}
    \quad
    \Gamma, \Delta \vdash z : \text{Bool} \times \text{Bool}
  }{
    \Gamma \,\fatsemi\, \Delta \vdash \text{AND}(z) \,!\, \text{Bool}
  }
  \qquad
  \cfrac{
    \cfrac{
      \text{NOT} : \text{Bool} \rightarrow \text{Bool}
      \quad
      \Gamma, \Delta' \vdash u : \text{Bool}
    }{
      \Gamma \,\fatsemi\, \Delta' \vdash \text{NOT}(u) \,!\, \text{Bool}
    }
    \quad
    \cfrac{
      \Gamma \vdash k : \text{Bool} \rightsquigarrow \text{Bool}
      \quad
      \Gamma, \Delta', v : \text{Bool} \vdash v : \text{Bool}
    }{
      \Gamma \,\fatsemi\, \Delta', v : \text{Bool} \vdash k \bullet v \,!\, \text{Bool}
    }
  }{
    \Gamma \,\fatsemi\, \Delta' \vdash \textbf{let } v \Leftarrow \text{NOT}(u) \textbf{ in } k \bullet v \,!\, \text{Bool}
  }
}{
  \Gamma \,\fatsemi\, \Delta \vdash Q_{\text{NAND}} \,!\, \text{Bool}
}
$$

Similarly, we can derive $\Gamma \,\fatsemi\, \Delta \vdash Q_{\text{OR}} \,!\, \text{Bool}$. Hence, the derivation of $\vdash H_1' : \text{Bool} \Rightarrow \text{Bool}$ is as follows:

$$
\cfrac{
  \cfrac{
    x : \text{Bool} \vdash x : \text{Bool}
  }{
    \diamond \,\fatsemi\, x : \text{Bool} \vdash \lfloor x \rfloor \,!\, \text{Bool}
  }
  \qquad
  (k : \delta \rightsquigarrow \text{Bool} \,\fatsemi\, z : \gamma \vdash Q_{\text{op}} \,!\, \text{Bool})_{\text{op} \in \Sigma_{\text{LC}}}
}{
  \vdash H_1' : \text{Bool} \Rightarrow \text{Bool}
}
$$

where $Q_{\text{AND}} = \textbf{let } u \Leftarrow \text{AND}(z) \textbf{ in } k \bullet u$ and $Q_{\text{NOT}} = \textbf{let } u \Leftarrow \text{NOT}(z) \textbf{ in } k \bullet u$.

The evaluation of $\textbf{handle}(\textbf{handle } \text{NAND}(\langle \texttt{true}, \texttt{false} \rangle) \textbf{ with } H_1') \textbf{ with } H_2'$ is shown in Figure 5.

4.3.2 Read only state

Let $BType = \{\text{Unit}, \text{Bool}\}$ and $\Sigma = \{\text{get} : \text{Unit} \rightarrow \text{Bool}\}$. The operation get is expected to read a state of type Bool and return the stored value. We can implement get by handler.

The following judgements are derivable:

$$
\cfrac{
  x : \text{Bool} \vdash x : \text{Bool}
}{
  \diamond \,\fatsemi\, x : \text{Bool} \vdash \lfloor x \rfloor \,!\, \text{Bool}
}
$$

$$
\cfrac{
  k : \text{Bool} \rightsquigarrow \text{Bool} \vdash k : \text{Bool} \rightsquigarrow \text{Bool}
  \quad
  k : \text{Bool} \rightsquigarrow \text{Bool}, z : \text{Unit} \vdash \texttt{true} : \text{Bool}
}{
  k : \text{Bool} \rightsquigarrow \text{Bool} \,\fatsemi\, z : \text{Unit} \vdash k \bullet \texttt{true} \,!\, \text{Bool}
}.
$$

Hence, we can construct a handler $H = \{\,\fatsemi\, x \mapsto \lfloor x \rfloor; \text{get}, k \,\fatsemi\, z \mapsto k \bullet \texttt{true}\}$ and derive $\vdash H : \text{Bool} \Rightarrow \text{Bool}$. This handler H implements get so that the stored value is $\texttt{true}$. For example, the reduction of a program $\textbf{handle } \text{get}(\langle \rangle) \textbf{ with } H$ is as follows.

$$
\begin{aligned}
\textbf{handle } \text{get}(\langle \rangle) \textbf{ with } H &\to (\lambda^\bullet y. \, \textbf{handle} \lfloor y \rfloor \textbf{ with } H) \bullet \texttt{true} \\
&\to (\textbf{handle} \lfloor y \rfloor \textbf{ with } H)[\texttt{true}/y] \\
&= \textbf{handle} \lfloor \texttt{true} \rfloor \textbf{ with } H \\
&\to \lfloor x \rfloor [\texttt{true}/x] \\
&= \lfloor \texttt{true} \rfloor
\end{aligned}
$$

$$\textbf{handle}(\textbf{handle } \mathsf{NAND}(\langle \mathtt{true}, \mathtt{false} \rangle) \textbf{ with } H_1') \textbf{ with } H_2'$$

$$\rightarrow \textbf{handle } Q_{\mathsf{NAND}}[\langle \mathtt{true}, \mathtt{false} \rangle / z, (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') / k] \textbf{ with } H_2'$$

$$= \textbf{handle} \begin{pmatrix} \textbf{let } u \Leftarrow \mathsf{AND}(\langle \mathtt{true}, \mathtt{false} \rangle) \textbf{ in} \\ \textbf{let } v \Leftarrow \mathsf{NOT}(u) \textbf{ in} \\ (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') \bullet v \end{pmatrix} \textbf{ with } H_2'$$

$$\rightarrow \left(\lambda^\bullet y.\, \textbf{handle} \begin{pmatrix} \textbf{let } u \Leftarrow \lfloor y \rfloor \textbf{ in let } v \Leftarrow \mathsf{NOT}(u) \textbf{ in} \\ (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') \bullet v \end{pmatrix} \textbf{ with } H_2' \right)$$

$$\bullet \begin{pmatrix} \textbf{if}(\textbf{fst}\langle \mathtt{true}, \mathtt{false} \rangle) \\ \textbf{then}(\textbf{if}(\textbf{snd}\langle \mathtt{true}, \mathtt{false} \rangle) \textbf{ then } \mathtt{true} \textbf{ else } \mathtt{false}) \\ \textbf{else } \mathtt{false} \end{pmatrix}$$

$$\rightarrow^* \left(\lambda^\bullet y.\, \textbf{handle} \begin{pmatrix} \textbf{let } u \Leftarrow \lfloor y \rfloor \textbf{ in let } v \Leftarrow \mathsf{NOT}(u) \textbf{ in} \\ (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') \bullet v \end{pmatrix} \textbf{ with } H_2' \right) \bullet \mathtt{false}$$

$$\rightarrow^* \textbf{handle} \begin{pmatrix} \textbf{let } v \Leftarrow \mathsf{NOT}(\mathtt{false}) \textbf{ in} \\ (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') \bullet v \end{pmatrix} \textbf{ with } H_2'$$

$$\rightarrow \left(\lambda^\bullet y'.\, \textbf{handle} \begin{pmatrix} \textbf{let } v \Leftarrow \lfloor y' \rfloor \textbf{ in} \\ (\lambda^\bullet y.\, \textbf{handle} \lfloor y \rfloor \textbf{ with } H_1') \bullet v \end{pmatrix} \textbf{ with } H_2' \right) \bullet \begin{pmatrix} \textbf{if } \mathtt{false} \\ \textbf{then } \mathtt{false} \\ \textbf{else } \mathtt{true} \end{pmatrix}$$

$$\rightarrow^* \textbf{handle handle} \lfloor \mathtt{true} \rfloor \textbf{ with } H_1' \textbf{ with } H_2'$$

$$\rightarrow \textbf{handle} \lfloor \mathtt{true} \rfloor \textbf{ with } H_2'$$

$$\rightarrow \lfloor \mathtt{true} \rfloor$$

Fig. 5. The evaluation of $\textbf{handle}(\textbf{handle } \mathsf{NAND}(\langle \mathtt{true}, \mathtt{false} \rangle) \textbf{ with } H_1') \textbf{ with } H_2'$.

5 Denotational semantics

5.1 Models of arrow calculus

We define models of the arrow calculus, in which the arrow calculus with operations is interpreted.

Definition 5.1 (a model of arrow calculus). A *model of the arrow calculus* consists of the following data.

- A cartesian closed category (ccc) $\mathbb{C}$.
- A cocomplete cartesian closed category $\mathbb{C}'$.
- A cartesian fully faithful functor $J \colon \mathbb{C} \to \mathbb{C}'$.
- A $\mathbb{C}$-small strong promonad $\mathscr{A} \colon \mathbb{C} \nrightarrow \mathbb{C}$ on $\mathbb{C}$ in $\mathbb{C}'$-**Prof** (Definition 2.3 and 2.7).

For a ccc $\mathbb{C}$, a cocomplete ccc $\mathbb{C}'$ and a cartesian fully faithful functor $J \colon \mathbb{C} \to \mathbb{C}'$, the identity $\mathbb{C}'$-profunctor $\mathrm{I}_{\mathbb{C}} \colon \mathbb{C} \nrightarrow \mathbb{C}$ on $\mathbb{C}$ defined by $\mathrm{I}_{\mathbb{C}}(A, B) = (JA \Rightarrow JB) = J(A \Rightarrow B)$ is a

$\mathbb{C}$-small strong promonad on $\mathbb{C}$ in $\mathbb{C}'$-**Prof**. We have $I_{\mathbb{C}}^{\circ}(A, B) = (A \Rightarrow B)$ and $J \circ I_{\mathbb{C}}^{\circ} = I_{\mathbb{C}}$:

$$
\begin{array}{ccc}
\mathbb{C}^{\mathrm{op}} \times \mathbb{C} & \xrightarrow{\;\;I_{\mathbb{C}}\;\;} & \mathbb{C}' \\
& \searrow_{I_{\mathbb{C}}^{\circ}} & \uparrow_{J} \quad . \\
& & \mathbb{C}
\end{array}
$$

Definition 5.2 (interpretation). Given a model of arrow calculus $(\mathbb{C}, \mathbb{C}', J, \mathscr{A})$, interpretation $[\![\beta]\!] \in \mathbb{C}$ of each base type $\beta \in BType$ and interpretation $[\![\mathrm{op}]\!]\colon \mathscr{A}([\![\delta]\!], -) \Rightarrow \mathscr{A}([\![\gamma]\!], -)$ of each operation $\mathrm{op}\colon \gamma \twoheadrightarrow \delta$, we define interpretation of the arrow calculus with operations (without handlers). The interpretation $[\![A]\!]$ of a type A is defined as a natural extension of $[\![\beta]\!]$ with $[\![A \rightsquigarrow B]\!] = \mathscr{A}^{\circ}([\![A]\!], [\![B]\!])$. For a term $\Gamma \vdash M\colon A$ and a command $\Gamma \,\fatsemi\, \Delta \vdash P \,!\, A$, their interpretation $[\![M]\!]\colon [\![\Gamma]\!] \to [\![A]\!]$ and $[\![P]\!]\colon [\![\Gamma]\!] \to \mathscr{A}^{\circ}([\![\Delta]\!], [\![A]\!])$ are defined as in Figure 6.

We discuss interpretation of handlers in Section 5.3.

5.2 Construction of a model of the arrow calculus

We give a semantics of the arrow calculus with operations and handlers using a **Set**-small strong **Ens**-promonad $\mathscr{A}$ on **Set**. We fix an interpretation of the base types $[\![-]\!]\colon BType \to$ **Set**. Here $BType$ is regarded as a discrete category.

First, we construct a map $\mathrm{Arr}_{\Sigma}\colon \mathrm{Ob}(\mathbf{Set}^{\mathrm{op}}) \times \mathrm{Ob}(\mathbf{Set}) \to \mathrm{Ob}(\mathbf{Ens})$.

Definition 5.3 (arrow term). For sets A and B, $\mathrm{Arr}_{\Sigma}(A, B)$ is defined to be the smallest class satisfying the rules in Figure 7. We call an element in $\mathrm{Arr}_{\Sigma}(A, B)$ an *arrow term*. An *equation* is a pair of arrow terms (t, t') where $t, t' \in \mathrm{Arr}_{\Sigma}(A, B)$ for some sets A and B. A *theory* is a pair (Σ, E) of signature Σ and a set E of equations.

In the following, we consider only the case $E = \emptyset$.

Unfortunately, $\mathrm{Arr}_{\Sigma}(A, B)$ is a proper class because it contains $a \ggg b$ for any $X \in \mathbf{Set}$, $a \in \mathrm{Arr}_{\Sigma}(A, X)$ and $b \in \mathrm{Arr}_{\Sigma}(X, B)$. However, we can define an equivalence relation $\sim$ on $\mathrm{Arr}_{\Sigma}(A, B)$, and obtain **Set**-small strong **Ens**-promonad $\mathrm{Arr}_{\Sigma}(-_1, -_2)/{\sim}\colon \mathbf{Set} \nrightarrow \mathbf{Set}$.

Interpretation $[\![\Gamma \vdash M : A]\!] : [\![\Gamma]\!] \to [\![A]\!]$ of terms

T-CABS: $[\![\Gamma \vdash \lambda^\bullet x : A.P : A \rightsquigarrow B]\!] = [\![P]\!]$

and for the other rules, $[\![\Gamma \vdash M : A]\!]$ is defined straightforwardly.

Interpretation $[\![\Gamma \, \mathring{,} \, \Delta \vdash P \, ! \, A]\!] : [\![\Gamma]\!] \to \mathscr{A}^\circ([\![\Delta]\!], [\![A]\!])$ of commands

T-PURE:

$$[\![\Gamma \, \mathring{,} \, \Delta \vdash \lfloor M \rfloor \, ! \, A]\!] = \left([\![\Gamma]\!] \xrightarrow{\Lambda[\![M]\!]} \mathrm{I}^\circ_\mathbb{C}([\![\Delta]\!], [\![A]\!]) \xrightarrow{\eta} \mathscr{A}^\circ([\![\Delta]\!], [\![A]\!]) \right)$$

T-CAPP:

$$[\![\Gamma \, \mathring{,} \, \Delta \vdash L \bullet M \, ! \, B]\!] = \left([\![\Gamma]\!] \xrightarrow{\langle m, [\![L]\!] \rangle} \mathscr{A}([\![\Delta]\!], [\![A]\!]) \times \mathscr{A}([\![A]\!], [\![B]\!]) \xrightarrow{\mu} \mathscr{A}([\![\Delta]\!], [\![B]\!]) \right)$$

where

$$m = \left([\![\Gamma]\!] \xrightarrow{\Lambda[\![M]\!]} \mathrm{I}^\circ([\![\Delta]\!], [\![A]\!]) \xrightarrow{\eta} \mathscr{A}([\![\Delta]\!], [\![A]\!]) \right)$$

T-LET:

$$[\![\Gamma \, \mathring{,} \, \Delta \vdash \mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q \, ! \, B]\!]$$

$$= \left(\begin{array}{l} [\![\Gamma]\!] \xrightarrow{\langle d, p, [\![Q]\!] \rangle} \mathscr{A}([\![\Delta]\!], [\![\Delta]\!]^2) \times \mathscr{A}([\![\Delta]\!]^2, [\![A]\!] \times [\![\Delta]\!]) \times \mathscr{A}([\![A]\!] \times [\![\Delta]\!], [\![B]\!]) \\ \xrightarrow{\mu \times \mathrm{id}} \mathscr{A}([\![\Delta]\!], [\![A]\!] \times [\![\Delta]\!]) \times \mathscr{A}([\![A]\!] \times [\![\Delta]\!], [\![B]\!]) \xrightarrow{\mu} \mathscr{A}([\![\Delta]\!], [\![B]\!]) \end{array} \right)$$

where

$$d = \left([\![\Gamma]\!] \xrightarrow{!} 1 \xrightarrow{\Lambda(\lambda x.\langle x,x \rangle)} \mathrm{I}^\circ([\![\Delta]\!], [\![\Delta]\!]^2) \xrightarrow{\eta} \mathscr{A}([\![\Delta]\!], [\![\Delta]\!]^2) \right)$$

$$p = \left([\![\Gamma]\!] \xrightarrow{[\![P]\!]} \mathscr{A}([\![\Delta]\!], [\![A]\!]) \xrightarrow{\sigma} \mathscr{A}([\![\Delta]\!]^2, [\![A]\!] \times [\![\Delta]\!]) \right)$$

T-OP:

$$[\![\Gamma \, \mathring{,} \, \Delta \vdash \mathsf{op}(M) \, ! \, \delta]\!] = \left([\![\Gamma]\!] \xrightarrow{\langle m,o \rangle} \mathscr{A}([\![\Delta]\!], [\![\gamma]\!]) \times \mathscr{A}([\![\gamma]\!], [\![\delta]\!]) \xrightarrow{\mu} \mathscr{A}([\![\Delta]\!], [\![\delta]\!]) \right)$$

where

$$m = \left([\![\Gamma]\!] \xrightarrow{\Lambda[\![M]\!]} \mathrm{I}^\circ([\![\Delta]\!], [\![\gamma]\!]) \xrightarrow{\eta} \mathscr{A}([\![\Delta]\!], [\![\gamma]\!]) \right)$$

$$o = \left([\![\Gamma]\!] \xrightarrow{!} 1 \xrightarrow{\Lambda(\mathrm{id})} \mathrm{I}^\circ([\![\delta]\!], [\![\delta]\!]) \xrightarrow{\eta} \mathscr{A}([\![\delta]\!], [\![\delta]\!]) \xrightarrow{[\![\mathsf{op}]\!]_{[\![\delta]\!]}} \mathscr{A}([\![\gamma]\!], [\![\delta]\!]) \right)$$

Fig. 6. The categorical semantics

$$\frac{f \in \mathbf{Set}(A,B)}{\mathrm{arr}(f) \in \mathrm{Arr}_\Sigma(A,B)} \quad \frac{\mathrm{op} : \gamma \to \delta \in \Sigma}{\mathrm{op} \in \mathrm{Arr}_\Sigma([\![\gamma]\!], [\![\delta]\!])} \quad \frac{a \in \mathrm{Arr}_\Sigma(A,B) \quad b \in \mathrm{Arr}_\Sigma(B,C)}{a \ggg b \in \mathrm{Arr}_\Sigma(A,C)}$$

$$\frac{a \in \mathrm{Arr}_\Sigma(A,B)}{\mathrm{first}_C(a) \in \mathrm{Arr}_\Sigma(A \times C, B \times C)}$$

Fig. 7. Construction of $\mathrm{Arr}_\Sigma(A,B) \in \mathbf{Ens}$ for $A, B \in \mathbf{Set}$.

The equivalence relation $\sim$ is defined as the smallest congruence relation satisfying the following axioms (5.1) to (5.9), which correspond to the arrow laws (1.1) to (1.9).

$$(a \ggg b) \ggg c \sim a \ggg (b \ggg c) \tag{5.1}$$

$$\mathrm{arr}(g \circ f) \sim \mathrm{arr}(g) \ggg \mathrm{arr}(f) \tag{5.2}$$

$$\mathrm{arr}(\mathrm{id}) \ggg a \sim a \tag{5.3}$$

$$a \ggg \mathrm{arr}(\mathrm{id}) \sim a \tag{5.4}$$

$$\mathrm{first}(a) \ggg \mathrm{arr}(\mathrm{id} \times f) \sim \mathrm{arr}(\mathrm{id} \times f) \ggg \mathrm{first}(a) \tag{5.5}$$

$$\mathrm{first}(a) \ggg \mathrm{arr}(\pi_1) \sim \mathrm{arr}(\pi_1) \ggg a \tag{5.6}$$

$$\mathrm{first}(a) \ggg \mathrm{arr}(\alpha) \sim \mathrm{arr}(\alpha) \ggg \mathrm{first}(\mathrm{first}(a)) \tag{5.7}$$

$$\mathrm{first}(\mathrm{arr}(f)) \sim \mathrm{arr}(f \times \mathrm{id}) \tag{5.8}$$

$$\mathrm{first}(a \ggg b) \sim \mathrm{first}(a) \ggg \mathrm{first}(b) \tag{5.9}$$

We denote $\mathrm{Arr}_\Sigma(A,B)/\sim$ as $\mathscr{A}_\Sigma(A,B)$. We define $\mathrm{second}_X(a) = \mathrm{arr}(sym_{A,X}) \ggg \mathrm{first}_X(a) \ggg \mathrm{arr}(sym_{X,B})$ for $a \in \mathrm{Arr}_\Sigma(A,B)$, where the map $sym_{A,X} : A \times X \to X \times A$ is defined by $sym_{A,X}(a,x) = (x,a)$.

We use string diagram like notation introduced by Asada and Hasuo (2010). Arrow terms are depicted as follows.

$$A - \boxed{f} - B = \mathrm{arr}(f) \qquad \text{for } f : A \to B \text{ in } \mathbf{Set}$$

$$[\![\gamma]\!] - \boxed{\mathrm{op}} - [\![\delta]\!] = \mathrm{op} \qquad \text{for } \mathrm{op} : \gamma \to \delta \in \Sigma$$

$$A - \boxed{a}^{B}\boxed{b} - C = a \ggg b \qquad \text{for } a \in \mathrm{Arr}_\Sigma(A,B) \text{ and } b \in \mathrm{Arr}_\Sigma(B,C)$$

$$\begin{array}{c} A - \boxed{a} - B \\ C \text{————} C \end{array} = \mathrm{first}_C(a) \qquad \text{for } a \in \mathrm{Arr}_\Sigma(A,B)$$

Especially, we write

$$A \prec {}^{A}_{A} = A - \boxed{d} {}^{A}_{A} \qquad \text{for the map } d : A \to A \times A \text{ defined by } d(a) = (a,a),$$

$$\begin{array}{c} A \\ B \end{array} \rhd - A = \begin{array}{c} A \\ B \end{array} \boxed{\pi_1} - A \qquad \text{for the first projection } \pi_1 : A \times B \to A,$$

$$\begin{array}{c} A \\ B \end{array} \rhd - B = \begin{array}{c} A \\ B \end{array} \boxed{\pi_2} - B \qquad \text{for the second projection } \pi_2 : A \times B \to B.$$

The **Set**-smallness of $\mathrm{Arr}(-_1, -_2)/\sim$ is proven from Proposition 5.4. It says that every arrow term is equivalent to the normal form. The normal form was introduced by Yallop
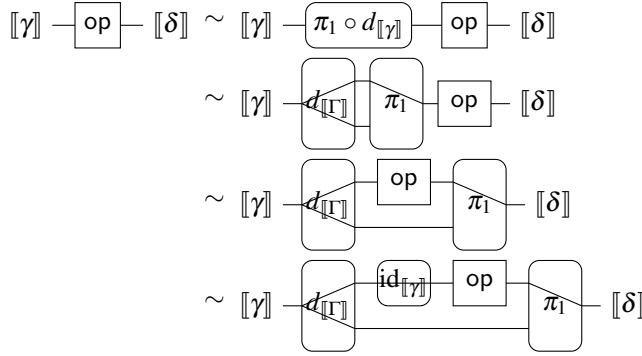
Fig. 8. In case $a = \mathsf{op}$

(2010) for arrows in Haskell to compare two programs. The normal form here is essentially the same as his except that it is defined for arrow terms $a \in \mathrm{Arr}_\Sigma(A, B)$. The size of the collection of all normal forms is small.

Proposition 5.4 (normal form). *Let A and B be sets. For any $a \in \mathrm{Arr}_\Sigma(A, B)$, there exist a natural number $n \in \mathbb{N}$, a sequence of operations $(\mathsf{op}_i \colon \gamma_i \twoheadrightarrow \delta_i)_{i=1,\ldots,n}$, a sequence of maps $(f_i \colon [\![\delta_{i-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A \to [\![\gamma_i]\!])_{i=1,\ldots,n}$ and $g \colon [\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A \to B$ such that*

$$
\begin{aligned}
a \sim{} & \mathrm{arr}(d_A) \ggg \mathrm{first}_A(\mathrm{arr}(f_1) \ggg \mathsf{op}_1) \\
& \ggg \mathrm{arr}(d_{[\![\delta_1]\!] \times A}) \ggg \mathrm{first}_{[\![\delta_1]\!] \times A}(\mathrm{arr}(f_2) \ggg \mathsf{op}_2) \\
& \ggg \cdots \\
& \ggg \mathrm{arr}(d_{[\![\delta_{n-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{first}_{[\![\delta_{n-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A}(\mathrm{arr}(f_n) \ggg \mathsf{op}_n) \\
& \ggg \mathrm{arr}(g)
\end{aligned}
\tag{5.10}
$$

where $d_X \colon X \to X \times X$ is the diagonal map: $d_X(x) = (x, x)$. We write the arrow term of the right-hand side as $\mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n}, (f_i)_{i=1,\ldots,n}; g)$.

Proof [Proof sketch] We prove by induction on the structure of $a \in \mathrm{Arr}_\Sigma(A, B)$. Since the proof is very long, we only prove the following two cases here and the rest of the proof is sent to the appendix.

In case $a = \mathrm{arr}(f)$ for some $f \colon A \to B$. The proposition trivially holds for $n = 0$.

In case $a = \mathsf{op}$ for some $(\mathsf{op} \colon \gamma \twoheadrightarrow \delta) \in \Sigma$ with $[\![\gamma]\!] = A$ and $[\![\delta]\!] = B$. We have

$$
\begin{aligned}
a ={}& \mathsf{op} \\
\sim{}& \mathrm{arr}(\mathrm{id}) \ggg \mathsf{op} \\
\sim{}& \mathrm{arr}(\pi_1 \circ d_A) \ggg \mathsf{op} \\
\sim{}& \mathrm{arr}(d_A) \ggg \mathrm{arr}(\pi_1) \ggg \mathsf{op} \\
\sim{}& \mathrm{arr}(d_A) \ggg \mathrm{first}(\mathsf{op}) \ggg \mathrm{arr}(\pi_1) \\
\sim{}& \mathrm{arr}(d_A) \ggg \mathrm{first}_A(\mathrm{arr}(\mathrm{id}_{[\![\gamma]\!]}) \ggg \mathsf{op}) \ggg \mathrm{arr}(\pi_1).
\end{aligned}
$$

See Figure 8. ∎

We call an arrow term in the form $\mathrm{cf}\,((\mathsf{op}_i)_i, (f_i)_i; g)$ in the right-hand side of (5.10) a *canonical arrow term*. The collection $\mathscr{A}_\Sigma^\circ(A, B)$ of all canonical arrow terms in $\mathrm{Arr}_\Sigma(A, B)$ is a *set*, not a proper class, because

$$\mathscr{A}_\Sigma^\circ(A, B) = \bigcup_{n \in \mathbb{N}} \left\{ \mathrm{cf}\,((\mathsf{op}_i)_{i=1}^n, (f_i)_{i=1}^n; g) \;\middle|\; \begin{array}{l} (\mathsf{op}_i)_i \in \Sigma^n \\ (f_i \colon [\![\delta_{i-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A \to [\![\gamma_i]\!])_i, \\ g \colon [\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A \to B \end{array} \right\}.$$

We identify an equivalence class $[a]$ for an arrow term $a \in \mathrm{Arr}_\Sigma(A, B)$ with its normal form.

The map $\mathscr{A}_\Sigma \colon \mathrm{Ob}(\mathbf{Set}^{\mathrm{op}}) \times \mathrm{Ob}(\mathbf{Set}) \to \mathrm{Ob}(\mathbf{Ens})$ is extended to an $\mathbf{Ens}$-profunctor $\mathscr{A}_\Sigma \colon \mathbf{Set} \nrightarrow \mathbf{Set}$ by defining $\mathscr{A}_\Sigma(f, g)([a]) = [\mathrm{arr}(f) \ggg a \ggg \mathrm{arr}(g)]$ for $f \colon A' \to A$, $g \colon B \to B'$ and $a \in \mathrm{Arr}_\Sigma(A, B)$. We have constructed a model of the arrow calculus.

Proposition 5.5. *The data* $(\mathbf{Set}, \mathbf{Ens}, J, \mathscr{A}_\Sigma)$ *is a model of the arrow calculus (Definition 5.1).*

Note that the model $(\mathbf{Set}, \mathbf{Ens}, J, \mathscr{A}_\Sigma)$ is the free model in the sense that if $(\mathbf{Set}, \mathbf{Ens}, J, \mathscr{A})$ is also a model then there is a unique 2-cell $h \colon \mathscr{A}_\Sigma \Rightarrow \mathscr{A}$ which is compatible with units and multiplications of $\mathscr{A}_\Sigma$ and $\mathscr{A}$.

5.3 Interpretation of handlers

We interpret handlers in the model $(\mathbf{Set}, \mathbf{Ens}, J, \mathscr{A}_\Sigma)$ of the arrow calculus. We fix an interpretation of base types $[\![-]\!] \colon BType \to \mathrm{Ob}(\mathbf{Set})$.

5.3.1 The problem of strength

Unlike in the case of the strong monad $\mathscr{T} \colon \mathbf{Set} \to \mathbf{Set}$ in $\mathbf{Cat}$, in the case of the strong promonad $\mathscr{A} \colon \mathbf{Set} \nrightarrow \mathbf{Set}$ in $\mathbf{Ens}$-$\mathbf{Prof}$, the treatment of the strength is non-trivial. To interpret a handler as a $\mathscr{A}_\Sigma^\circ$-homomorphism, we have to construct a family of maps

$$(\alpha_{A,B} \colon \mathscr{A}_\Sigma^\circ(A, B) \times \mathscr{A}_\Sigma^\circ(B, [\![D]\!]) \to \mathscr{A}_\Sigma^\circ(A, [\![D]\!]))_{A,B} \tag{5.11}$$

from a handler $H = \{\, \mathring{\,}\, x \colon C \mapsto P \} \cup \{\mathsf{op}, k \colon \delta \rightsquigarrow D \,\mathring{\,}\, z \colon \gamma \mapsto Q_{\mathsf{op}}\}_{\mathsf{op}}$:

$$\frac{\diamond \,\mathring{\,}\, x \colon C \vdash P \,!\, D \quad (k \colon \delta \rightsquigarrow D \,\mathring{\,}\, z \colon \gamma \vdash Q_{\mathsf{op}} \,!\, D)_{\mathsf{op} \colon \gamma \to \delta}}{\vdash H \colon C \Rightarrow D} \;\; \text{T-Handler}.$$

The problem is that we cannot define the maps (5.11) since there is no way to define $\alpha(\mathrm{first}_S(\mathsf{op}), b)$ for $\mathsf{op} \colon \gamma \to \delta$ and $b \in \mathscr{A}_\Sigma^\circ([\![\delta]\!] \times S, [\![D]\!])$:

$$\begin{aligned} \mathscr{A}_\Sigma^\circ([\![\gamma]\!] \times S, [\![\delta]\!] \times S) \times \mathscr{A}_\Sigma^\circ([\![\delta]\!] \times S, [\![D]\!]) \to \mathscr{A}_\Sigma^\circ([\![\gamma]\!] \times S, [\![D]\!]) \\ (\mathrm{first}_S(\mathsf{op}), b) \mapsto \alpha(\mathrm{first}_S(\mathsf{op}), b). \end{aligned} \tag{5.12}$$

To define the above map, we need maps $\mathscr{A}_\Sigma^\circ([\![\delta]\!] \times S, [\![D]\!]) \to \mathscr{A}_\Sigma^\circ([\![\gamma]\!] \times S, [\![D]\!])$ as an interpretation of Q_{op}. However, in the naïve interpretation (Definition 5.2), the command Q_{op} is interpreted as a map $[\![Q_{\mathsf{op}}]\!] \colon \mathscr{A}_\Sigma^\circ([\![\delta]\!], [\![D]\!]) \to \mathscr{A}_\Sigma^\circ([\![\gamma]\!], [\![D]\!])$.

5.3.2 Interpretation with a parameter

We solve this problem by using an additional parameter $S \in \mathrm{Ob}(\mathbf{Set})$ in the interpretation of terms. This additional parameter enables us to define maps $\mathscr{A}_\Sigma^\circ([\![\delta]\!] \times S, [\![D]\!]) \to \mathscr{A}_\Sigma^\circ([\![\gamma]\!] \times S, [\![D]\!])$ as an interpretation of Q_{op}. For a type A and a set S, we define the interpretation $[\![A]\!]^S$ as an extension of $[\![\beta]\!]$ for $\beta \in BType$.

$$[\![\beta]\!]^S = [\![\beta]\!]$$
$$[\![A \times B]\!]^S = [\![A]\!]^S \times [\![B]\!]^S$$
$$[\![A \to B]\!]^S = [\![A]\!]^S \Rightarrow [\![B]\!]^S$$
$$[\![A \rightsquigarrow B]\!]^S = \mathscr{A}_\Sigma^\circ([\![A]\!]^S \times S, [\![B]\!]^S)$$

The interpretation $[\![\Gamma]\!]^S$ of a context Γ is defined as an extension of the interpretation of types:

$$[\![\diamond]\!]^S = 1, \qquad [\![\Gamma, x : A]\!]^S = [\![\Gamma]\!]^S \times [\![A]\!]^S.$$

The key is the interpretation $[\![A \rightsquigarrow B]\!]^S$ of a type $A \rightsquigarrow B$. Adding S to the "input argument" of the arrow allows us to deal with strength.

If a type A is primitive, its interpretation is independent of S, that is we can show the following lemma by the definitions.

Lemma 5.6. *If a type A is primitive ($\Phi(A)$), then $[\![A]\!]^S = [\![A]\!]^1$ for any S.*

Next, we define interpretation of terms and commands. For judgements $\Gamma \vdash M : A$ and $\Gamma \, \mathring{,} \, \Delta \vdash P : A$, we want to define:

$$[\![\Gamma \vdash M : A]\!]^S \colon [\![\Gamma]\!]^S \to [\![A]\!]^S,$$
$$[\![\Gamma \, \mathring{,} \, \Delta \vdash P \, ! \, A]\!]^S \colon [\![\Gamma]\!]^S \to \mathscr{A}_\Sigma^\circ([\![\Delta]\!]^S \times S, [\![A]\!]^S).$$

Let $e \in [\![\Gamma]\!]^S$. The interpretation of terms is defined as follows.

$$[\![\Gamma, y : A \vdash y : A]\!]^S(e, a) = a \qquad\qquad\qquad a \in [\![A]\!]^S$$
$$[\![\Gamma \vdash \langle M_1, M_2 \rangle : A_1 \times A_2]\!]^S(e) = ([\![M_1]\!]^S(e), [\![M_2]\!]^S(e))$$
$$[\![\Gamma \vdash \mathbf{fst}\, M : A]\!]^S(e) = \pi_1([\![M]\!]^S(e))$$
$$[\![\Gamma \vdash \mathbf{snd}\, M : A]\!]^S(e) = \pi_2([\![M]\!]^S(e))$$
$$[\![\Gamma \vdash MN : B]\!]^S(e) = [\![M]\!]^S(e)\left([\![N]\!]^S(e)\right)$$
$$[\![\Gamma \vdash \lambda y : A.M : A \to B]\!]^S(e) = [\![M]\!]^S(e, -)$$
$$[\![\Gamma \vdash \lambda^\bullet y : A.P : A \rightsquigarrow B]\!]^S(e) = [\![P]\!]^S(e)$$

The interpretation of commands is defined as follows.

$$\llbracket \Gamma \, \mathring{,} \, \Delta \vdash \lfloor M \rfloor \,!\, A \rrbracket^S(e) = \mathrm{arr}\left(\llbracket \Delta \rrbracket^S \times S \xrightarrow{\pi_1} \llbracket \Delta \rrbracket^S \xrightarrow{\llbracket M \rrbracket^S(e,-)} \llbracket A \rrbracket^S \right)$$

$$\llbracket \Gamma \, \mathring{,} \, \Delta \vdash L \bullet M \,!\, B \rrbracket^S(e) = \mathrm{first}_S\left(\mathrm{arr}\left(\llbracket M \rrbracket^S(e,-) \right) \right) \ggg \llbracket L \rrbracket^S(e)$$

$$\llbracket \Gamma \, \mathring{,} \, \Delta \vdash \mathsf{op}(M) \,!\, \delta \rrbracket^S(e) = \mathrm{arr}\left(\llbracket \Delta \rrbracket^S \times S \xrightarrow{\pi_1} \llbracket \Delta \rrbracket^S \xrightarrow{\llbracket M \rrbracket^S(e,-)} \llbracket \gamma \rrbracket \right) \ggg \mathsf{op}$$

$$\llbracket \Gamma \, \mathring{,} \, \Delta \vdash \mathbf{let}\, y \Leftarrow P \,\mathbf{in}\, Q \,!\, B \rrbracket^S(e) = \mathrm{arr}\,(d) \ggg \mathrm{first}\left(\llbracket P \rrbracket^S(e) \right) \ggg \llbracket Q \rrbracket^S(e)$$

$$\text{where } d(z) = (z,z) \text{ for } z \in \llbracket \Delta \rrbracket^S \times S.$$

The interpretation of $\llbracket \mathbf{let}\, y \Leftarrow P \,\mathbf{in}\, Q \rrbracket^S(e)$ is illustrated as follows.
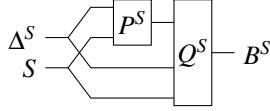
To interpret handling $\Gamma \, \mathring{,} \, \Delta \vdash \mathbf{handle}\, R \,\mathbf{with}\, H \,!\, D$, we construct an algebra from $H = \{\, \mathring{,} \, y : C \mapsto P \,!\, D\} \cup \{\mathsf{op}, k : \delta \rightsquigarrow D \, \mathring{,} \, c : \gamma \mapsto Q_{\mathsf{op}} \,!\, D\}_{\mathsf{op}:\gamma\to\delta\in\Sigma}$. Note that the derivation is

$$\cfrac{\Gamma \, \mathring{,} \, \Delta \vdash R \,!\, C \qquad \cfrac{\Phi(C) \wedge \Phi(D) \qquad \cfrac{\begin{array}{c}\diamond \, \mathring{,} \, y : C \vdash P \,!\, D \\ (k : \delta \rightsquigarrow D \, \mathring{,} \, c : \gamma \vdash Q_{\mathsf{op}} \,!\, D)_{\mathsf{op}:\gamma\to\delta\in\Sigma}\end{array}}{\vdash H : C \Rightarrow D} \text{ T-Handler}}{\Gamma \, \mathring{,} \, \Delta \vdash \mathbf{handle}\, R \,\mathbf{with}\, H \,!\, D}}{} \text{ T-Handle}$$

By Lemma 5.6, we have $\llbracket C \rrbracket^S = \llbracket C \rrbracket^1$ and $\llbracket D \rrbracket^S = \llbracket D \rrbracket^1$. We have maps:

$$\llbracket Q_{\mathsf{op}} \rrbracket^S : \mathscr{A}^{\circ}_{\Sigma}(\llbracket \delta \rrbracket \times S, \llbracket D \rrbracket^1) \to \mathscr{A}^{\circ}_{\Sigma}(\llbracket \gamma \rrbracket \times S, \llbracket D \rrbracket^1) \quad \text{for each } \mathsf{op}: \gamma \to \delta \in \Sigma.$$

The maps $\llbracket Q_{\mathsf{op}} \rrbracket^S$ induce an $\mathscr{A}_{\Sigma}$-algebra α on the presheaf $\mathscr{A}_{\Sigma}(-, \llbracket D \rrbracket)$ as follows.

$$\alpha : \mathscr{A}^{\circ}_{\Sigma}(A, B) \times \mathscr{A}^{\circ}_{\Sigma}(B, \llbracket D \rrbracket^1) \to \mathscr{A}^{\circ}_{\Sigma}(A, \llbracket D \rrbracket^1)$$
$$(\mathrm{arr}(f), a) \mapsto \mathrm{arr}(f) \ggg a$$

$$\alpha : \mathscr{A}^{\circ}_{\Sigma}(A, B) \times \mathscr{A}^{\circ}_{\Sigma}(B, \llbracket D \rrbracket^1) \to \mathscr{A}^{\circ}_{\Sigma}(A, \llbracket D \rrbracket^1)$$
$$(b_1 \ggg b_2, a) \mapsto \alpha(b_1, \alpha(b_2, a))$$

$$\alpha : \mathscr{A}^{\circ}_{\Sigma}(\llbracket \gamma \rrbracket, \llbracket \delta \rrbracket) \times \mathscr{A}^{\circ}_{\Sigma}(\llbracket \delta \rrbracket, \llbracket D \rrbracket^1) \to \mathscr{A}^{\circ}_{\Sigma}(\llbracket \gamma \rrbracket, \llbracket D \rrbracket^1)$$
$$(\mathsf{op}, a) \mapsto \llbracket Q_{\mathsf{op}} \rrbracket^1(a)$$

$$\alpha : \mathscr{A}^{\circ}_{\Sigma}(\llbracket \gamma \rrbracket \times S, \llbracket \delta \rrbracket \times S) \times \mathscr{A}^{\circ}_{\Sigma}(\llbracket \delta \rrbracket \times S, \llbracket D \rrbracket^1) \to \mathscr{A}^{\circ}_{\Sigma}(\llbracket \gamma \rrbracket \times S, \llbracket D \rrbracket^1)$$
$$(\mathrm{first}_S(\mathsf{op}), a) \mapsto \llbracket Q_{\mathsf{op}} \rrbracket^S(a)$$

Note that by considering the normal form of $b \in \mathscr{A}_{\Sigma}(A, B)$ which does not appear in the above definition, $\alpha(b, a)$ is defined, and the α is well-defined. For example, we have $\alpha(\mathrm{first}_S(\mathrm{arr}(f) \ggg \mathsf{op}), a) = \alpha(\mathrm{arr}(f \times \mathrm{id}_S) \ggg \mathrm{first}_S(\mathsf{op}), a) = \alpha(\mathrm{arr}(f \times \mathrm{id}_S), \alpha(\mathrm{first}_S(\mathsf{op}), a)) = \mathrm{arr}(f \times \mathrm{id}_S) \ggg \alpha(\mathrm{first}_S(\mathsf{op}), a) = \mathrm{arr}(f \times \mathrm{id}_S) \ggg \llbracket Q_{\mathsf{op}} \rrbracket^S(a)$.

We also have $\llbracket P \rrbracket^1 \in \mathscr{A}_\Sigma(\llbracket C \rrbracket^1, \llbracket D \rrbracket^1)$. Hence, by Corollary 3.4, there exists a homomorphism $h \colon \mu \to \alpha$ such that the following diagram commutes:

$$
\begin{array}{ccc}
\mathbf{Set}(-, \llbracket C \rrbracket^1) & \xrightarrow{\ \eta\ } & \mathscr{A}_\Sigma^\circ(-, \llbracket C \rrbracket^1) \\[2mm]
& \searrow{\scriptstyle [(-) \ggg \llbracket P \rrbracket^1]} & \Big\downarrow{\scriptstyle h} \\[2mm]
& & \mathscr{A}_\Sigma^\circ(-, \llbracket D \rrbracket^1)
\end{array} \quad .
$$

We use the homomorphism h to interpret **handle R with H**:

$$\llbracket \Gamma \,\mathbin{\fatsemi}\, \Delta \vdash \mathbf{handle}\ R\ \mathbf{with}\ H \,!\, D \rrbracket^S(e) = h_{\llbracket \Delta \rrbracket^S}(\llbracket R \rrbracket^S(e)) \quad \text{for } e \in \llbracket \Gamma \rrbracket^S$$

and write $\llbracket H \rrbracket$ for the homomorphism h.

The denotational semantics $\llbracket - \rrbracket^S$ defined here is compatible with the categorical semantics $\llbracket - \rrbracket$ (Definition 5.2) in the model $(\mathbf{Set}, \mathbf{Ens}, J, \mathscr{A}_\Sigma)$ in the following sense.

Proposition 5.7. *Let Γ and Δ be primitive contexts and A be a primitive type. Let M be a term and P be a command of the arrow calculus with operations (without handlers). The following hold.*

1. *If $\Gamma \vdash M : A$ then $\llbracket M \rrbracket^S = \llbracket M \rrbracket$ for any S.*
2. *If $\Gamma \,\mathbin{\fatsemi}\, \Delta \vdash P \,!\, A$ then $(\mathrm{arr}(j_s) \ggg \llbracket P \rrbracket^S(e)) = \llbracket P \rrbracket(e)$ for any S, $s \in S$ and $e \in \llbracket \Gamma \rrbracket$ where $j_s(z) = (z, s)$ for $z \in \llbracket \Delta \rrbracket$.*

Proposition 5.8. *Let Γ and Δ be contexts and A be a type. Let M be a term and P be a command of the arrow calculus with operations (without handlers). The following hold.*

1. *If $\Gamma \vdash M : A$ then $\llbracket M \rrbracket^1 = \llbracket M \rrbracket$.*
2. *If $\Gamma \,\mathbin{\fatsemi}\, \Delta \vdash P \,!\, A$ then $\llbracket P \rrbracket^1 = \llbracket P \rrbracket$.*

Hence, we write $\llbracket A \rrbracket^1$ and $\llbracket \Gamma \rrbracket^1$ simply as $\llbracket A \rrbracket$ and $\llbracket \Gamma \rrbracket$, respectively.

Remark 5.9. Let $\mathbb{C}$ be a Cartesian closed category, $\mathbb{C}'$ be a cocomplete Cartesian closed category and $J \colon \mathbb{C} \to \mathbb{C}'$ be a strong Cartesian fully faithful functor. For an ordinary strong monad $\mathscr{T} \colon \mathbb{C} \to \mathbb{C}$, we do not have the problem on the strength. The reason is as follows. Let $\mathscr{A} \colon \mathbb{C} \nrightarrow \mathbb{C}$ be a strong promonad $\mathscr{T}_*$ (Proposition 2.4) in $\mathbb{C}'$-**Prof** defined by

$$\mathscr{A}(A, B) = \mathbb{C}(A, \mathscr{T}B) = (JA \Rightarrow J\mathscr{T}B) \in \mathbb{C}'$$

for a strong monad $\mathscr{T} \colon \mathbb{C} \to \mathbb{C}$. This promonad $\mathscr{A}$ is $\mathbb{C}$-small. Judgements $\{k \colon \delta \rightsquigarrow D \,\mathbin{\fatsemi}\, z \colon \gamma \vdash Q_{\mathsf{op}} : D\}_{\mathsf{op} \in \Sigma}$ are interpreted as morphisms

$$\llbracket Q_{\mathsf{op}} \rrbracket \colon \llbracket \gamma \rrbracket \times (\llbracket \delta \rrbracket \Rightarrow \mathscr{T}\llbracket D \rrbracket) \to \mathscr{T}\llbracket D \rrbracket, \qquad \mathsf{op} \in \Sigma$$

in $\mathbb{C}$. An $\mathscr{A}$-algebra α can be constructed from the set of morphisms $\{\llbracket Q_{\mathsf{op}} \rrbracket\}_{\mathsf{op} \in \Sigma}$. The map (5.12) is defined as follows.

$$
\begin{aligned}
&\mathbb{C}(\llbracket \gamma \rrbracket \times S, \mathscr{T}(\llbracket \delta \rrbracket \times S)) \times \mathbb{C}(\llbracket \delta \rrbracket \times S, \mathscr{T}\llbracket D \rrbracket) \to \mathbb{C}(\llbracket \gamma \rrbracket \times S, \mathscr{T}\llbracket D \rrbracket) \\
&\qquad\qquad (\mathrm{first}_S(\mathsf{op}), b) \mapsto \Lambda^{-1}(\Lambda\llbracket Q_{\mathsf{op}} \rrbracket \circ (\Lambda b)).
\end{aligned}
\tag{5.13}
$$

The key is that if $\mathscr{A}(A,B) = \mathbb{C}(A, \mathscr{T}B)$ we have

$$\mathscr{L}_{A,B,C} \colon \mathscr{A}^{\circ}(C \times A, B) \cong \mathscr{A}^{\circ}(C, \mathscr{A}^{\circ}(A,B)) \qquad \text{for any } A, B, C \in \mathbb{C}. \tag{5.14}$$

Conversely, assume that (5.14) holds. We have

$$\mathscr{L}_{A,B,\mathscr{A}^{\circ}(A,B)}^{-1} \colon \mathscr{A}^{\circ}(\mathscr{A}^{\circ}(A,B), \mathscr{A}^{\circ}(A,B)) \to \mathscr{A}^{\circ}(\mathscr{A}^{\circ}(A,B) \times A, B)$$

$$\mathrm{id}_{\mathscr{A}^{\circ}(A,B)} \mapsto \mathscr{L}_{A,B,\mathscr{A}^{\circ}(A,B)}^{-1}(\mathrm{id}_{\mathscr{A}^{\circ}(A,B)}) =: \mathsf{app}_{A,B}.$$

The arrow with the maps $\mathsf{app}_{A,B}$ are known as a higher-order arrow (Hughes (2000)), which is equivalent to an ordinary monad (i.e. $\mathscr{A}(A,B) = \mathbb{C}(A, \mathscr{T}B)$ for a monad $\mathscr{T}$).

5.4 Soundness and adequacy

We prove the soundness (Theorem 5.12) of the operational semantics in Section 4.2 for the denotational semantics in Section 5.3.

First, observe the denotations of substituted terms and commands.

Lemma 5.10. *The following hold.*

1. *If* $\Gamma, x \colon A \vdash M \colon B$ *and* $\Gamma \vdash V \colon A$, *then* $[\![M[V/x]]\!]^S(c) = [\![M]\!]^S(c, [\![V]\!]^S(c))$.
2. *If* $\Gamma, x \colon A \,\mathbin{;}\, \Delta \vdash P \mathbin{!} B$ *and* $\Gamma \vdash V \colon A$, *then* $[\![P[V/x]]\!]^S(c) = [\![P]\!]^S(c, [\![V]\!]^S(c))$.
3. *If* $\Gamma \,\mathbin{;}\, x \colon A, \Delta \vdash P \mathbin{!} B$ *and* $\Gamma, \Delta \vdash V \colon A$, *then*

$$[\![P[V/x]]\!]^S(c) = \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S})) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg [\![P]\!]^S(c)$$

$$= \begin{array}{c} \Delta^S \longleftarrow \boxed{[\![V]\!]^S(c,-)} \\ \\ S \longrightarrow \end{array} \boxed{[\![P]\!]^S(c)} - B^S$$

The following lemma is used to show the soundness for $(\textbf{handle } \mathscr{F}[\mathsf{op}(V)] \textbf{ with } H) \to Q_{\mathsf{op}}[V/z, (\lambda^{\bullet}y \colon \delta.\ \textbf{handle } \mathscr{F}[\lfloor y \rfloor] \textbf{ with } H)/k]$.

Lemma 5.11. *If* $\diamond \,\mathbin{;}\, \diamond \vdash \mathscr{F}[\mathsf{op}(V)] \mathbin{!} C$, *then*

$$[\![\mathscr{F}[\mathsf{op}(V)]]\!]^S(\star) = \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S) \ggg \mathsf{op}) \ggg [\![\mathscr{F}[\lfloor y \rfloor]]\!]^S(\star)$$

holds.

Combining Lemma 5.10 and Lemma 5.11, we obtain the soundness theorem for the arrow calculus with operations and handlers.

Theorem 5.12 (soundness)**.** *The following hold.*

1. *If* $\diamond \vdash M \colon A$ *and* $M \to M'$, *then* $[\![M]\!]^S = [\![M']\!]^S$ *for any S.*
2. *If* $\diamond \,\mathbin{;}\, \diamond \vdash P \mathbin{!} A$ *and* $P \to P'$, *then* $[\![P]\!]^S = [\![P']\!]^S$ *for any S.*

Next, we prove the adequacy theorem using logical relations as done in (Bauer and Pretnar (2013); Sanada (2023)). The logical relations relate programs of type A and elements of $[\![A]\!]$. Let $BType = \{\mathsf{Unit}\}$. We add a constant $\langle\rangle$ to the terms and values. We also

add the following derivation rules to the arrow calculus:

$$\frac{}{\Gamma \vdash \langle\rangle \,!\, \text{Unit}} \ \text{T-UNIT}.$$

The interpretation $[\![\text{Unit}]\!]$ is the singleton set $\{\star\}$ and $[\![\langle\rangle]\!]^S \colon [\![\Gamma]\!]^S \to [\![\text{Unit}]\!]^S = \{\star\}$ is the unique map.

Definition 5.13 (logical relation). We define relations $(\lhd_A) \subseteq [\![A]\!] \times \{M \mid \diamond \vdash M : A\}$ and $(\blacktriangleleft_A) \subseteq \mathscr{A}_\Sigma(1, [\![A]\!]) \times \{P \mid \diamond\,\mathbf{;}\,\diamond \vdash P : A\}$ for each type A as follows:

$$\begin{aligned}
\star \lhd_{\text{Unit}} M &\iff M \to^* \langle\rangle \\
v \lhd_{A_1 \times A_2} M &\iff (M \to^* \langle V_1, V_2\rangle) \wedge (\pi_1(v) \lhd_{A_1} V_1) \wedge (\pi_2(v) \lhd_{A_2} V_2) \\
f \lhd_{A \to B} M &\iff (M \to^* \lambda x \colon A.M') \wedge \forall N.\forall w.(w \lhd_A N \implies fw \lhd_B MN) \\
a \lhd_{A \rightsquigarrow B} M &\iff (M \to^* \lambda^\bullet x \colon A.P) \wedge \forall N.\forall w.(w \lhd_A N \implies \text{arr}(w) \ggg a \blacktriangleleft_B M \bullet N)
\end{aligned}$$

and $a \blacktriangleleft_A P$ if

- $P \to^* \lfloor V \rfloor$ for a value V and there exists $v \in [\![A]\!]$ such that $a = \text{arr}(v)$ and $v \lhd_A V$, or
- $P \to^* \mathscr{F}[\text{op}(V)]$ for a value V and $\text{op} \colon \gamma \twoheadrightarrow \delta \in \Sigma$ and there exist $v \in [\![\gamma]\!]$ with $v \lhd_\gamma V$ and $b \in \mathscr{A}_\Sigma([\![\delta]\!], [\![A]\!])$ such that $a = \text{arr}(v) \ggg \text{op} \ggg b$ and $w \lhd_\delta W \implies \text{arr}(w) \ggg b \blacktriangleleft_A \mathscr{F}[\lfloor W \rfloor]$ for any $w \in [\![\delta]\!]$ and $\diamond \vdash W : \delta$.

Lemma 5.14. *The following hold.*

1. *If $M \to^* M'$ and $v \lhd_A M'$, then $v \lhd_A M$.*
2. *If $M \to^* M'$ and $v \lhd_A M$, then $v \lhd_A M'$.*
3. *If $P \to^* P'$ and $a \blacktriangleleft_A P'$, then $a \blacktriangleleft_A P$.*
4. *If $P \to^* P'$ and $a \blacktriangleleft_A P$, then $a \blacktriangleleft_A P'$.*

We can prove the following theorem using Lemma 5.14.

Theorem 5.15. *Let $\Gamma = x_1 : A_1, \ldots, x_m : A_m$ and $\Delta = y_1 : B_1, \ldots, y_n : B_n$. The following hold.*

1. *For $\Gamma \vdash M : A$ and $v_i \in [\![A_i]\!]$ and V_i with $v_i \lhd_{A_i} V_i$ for each $i \in \{1, \ldots, m\}$,*

$$[\![M]\!](v_1, \ldots, v_m) \lhd_A M[V_1/x_1, \ldots, V_m/x_m].$$

2. *For $\Gamma\,\mathbf{;}\,\Delta \vdash P : C$, $v_i \in [\![A_i]\!]$ and V_i with $v_i \lhd_{A_i} V_i$ for each $i \in \{1, \ldots, m\}$ and $w_j \in [\![B_j]\!]$ and W_j with $w_j \lhd_{B_j} W_j$ for each $j \in \{1, \ldots, n\}$,*

$$\begin{aligned}
&\text{arr}(\langle w_1, \ldots, w_n\rangle) \ggg [\![P]\!](v_1, \ldots, v_m) \\
&\blacktriangleleft_C P[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].
\end{aligned}$$

Proof [Proof sketch] The proof is done by induction on the derivation of $\Gamma \vdash M : A$ and $\Gamma\,\mathbf{;}\,\Delta \vdash P \,!\, C$.

Table 3. Translation between the arrow calculus and Paterson's notation

Arrow calculus with operations	Paterson's notation
$\lambda^\bullet x : A.P$	**proc** $x \to P$
$L \bullet M$	$L \prec M$
$\lfloor M \rfloor$	returnA $\prec M$
$\mathsf{op}(M)$	op $\prec M$
let $x \Leftarrow P$ **in** Q	**do**$\{x \leftarrow P; Q\}$

Here, we only show the most non-trivial case T-CABS, and the rest of the proof is sent to the appendix. The derivation is

$$\frac{\Gamma \,\mathbin{\raise1pt\hbox{$\scriptstyle\circ$}\kern-1pt\raise-2pt\hbox{$\scriptstyle\circ$}}\, y : B \vdash P \,!\, C}{\Gamma \vdash \lambda^\bullet y : B.P \,!\, B \rightsquigarrow C} \;\; \text{T-CABS}.$$

By the induction hypothesis, we have

$$\mathsf{arr}(w) \ggg \llbracket P \rrbracket(v_1, \ldots, v_m) \blacktriangleleft_C P[V_1/x_1, \ldots, V_m/x_m, W/y] \tag{5.15}$$

for any w and W with $w \vartriangleleft_B W$. Given N and w satisfying $w \vartriangleleft_B N$, we have $N \to^* W$ for a value W by the definition of $\vartriangleleft_B$. Applying Lemma 5.14(2), we have $w \vartriangleleft_B W$. Hence, (5.15) holds for this w and W. Now, we have

$$(\lambda^\bullet y : B.P[V_1/x_1, \ldots, V_m/x_m]) \bullet N \to^* (\lambda^\bullet y : B.P[V_1/x_1, \ldots, V_m/x_m]) \bullet W$$
$$\to P[V_1/x_1, \ldots, V_m/x_m, W/y].$$

Thus, applying Lemma 5.14(3), we have

$$\mathsf{arr}(w) \ggg \llbracket P \rrbracket(v_1, \ldots, v_m) \blacktriangleleft_C (\lambda^\bullet y : B.P[V_1/x_1, \ldots, V_m/x_m]) \bullet N.$$

Since $\llbracket P \rrbracket = \llbracket \lambda^\bullet y : B.P \rrbracket$, we have

$$\llbracket \lambda^\bullet y : B.P \rrbracket(v_1, \ldots, v_m) \vartriangleleft_{B \rightsquigarrow C} (\lambda^\bullet y : B.P[V_1/x_1, \ldots, V_m/x_m]).$$

∎

As a corollary of the above theorem, we can show adequacy.

Corollary 5.16 (adequacy)**.** *If* $\diamond \,\mathbin{\raise1pt\hbox{$\scriptstyle\circ$}\kern-1pt\raise-2pt\hbox{$\scriptstyle\circ$}}\, \diamond \vdash P \,!\, \mathsf{Unit}$ *and* $\llbracket P \rrbracket = \mathsf{arr}(\star) \in \mathscr{A}_\Sigma(1, \llbracket \mathsf{Unit} \rrbracket)$*, then* $P \to^* \lfloor \langle \rangle \rfloor$*.*

Proof By Theorem 5.15, we have $\mathsf{arr}(\star) = \llbracket P \rrbracket \blacktriangleleft_{\mathsf{Unit}} P$. By the definition of $\blacktriangleleft_{\mathsf{Unit}}$, we have $P \to^* \lfloor V \rfloor$ for a value V and $\star \vartriangleleft_{\mathsf{Unit}} V$. By the definition of $\vartriangleleft_{\mathsf{Unit}}$, we have $V = \langle \rangle$. Hence, $P \to^* \lfloor \langle \rangle \rfloor$. ∎

6 Related work

Paterson (2001) introduced a notation for arrows. As mentioned by Lindley et al. (2010), there is a translation between the arrow calculus and Paterson's notation, see Table 3.

There is another approach to semantics for arrows: *Freyd categories* (Atkey (2011)). As (Asada, 2010, Theorem 23) proved, small strong monads on $\mathbb{C}$ in $\widehat{\mathbb{C}}$-**Prof** with respect to the Yoneda embedding $\text{よ} : \mathbb{C} \to \widehat{\mathbb{C}}$ are equivalent to arrows in the sense of (Atkey, 2011,

Definition 2.1). We have adopted the profunctor approach because it is easier to consider with regard to algebras, which are the basis of handlers.

There are some notions of algebras for arrows or profunctors. Jacobs et al. (2009) defined an *algebra* for a promonad $\mathscr{A}\colon \mathbb{C} \nrightarrow \mathbb{C}$ as a 2-cell $\chi\colon \mathscr{A} \Rightarrow I_{\mathbb{C}}$ subject to some axioms (Jacobs et al., 2009, Definition 6.5), which is different from ours.

In nLab (2021), an *algebra* for a profunctor $H\colon \mathbb{C} \nrightarrow \mathbb{C}$ is defined as a pair (X, x) of an object $X \in \mathbb{C}$ and an element $x \in H(X, X)$, which does not induce our definition of algebras.

Altenkirch et al. (2010) introduced *relative monads* as a generalisation of monads. Let $\mathbb{C}$ be a category and $\widehat{\mathbb{C}} = [\mathbb{C}^{\mathrm{op}}, \mathbf{Set}]$. A relative monad $\mathscr{T}\colon \mathbb{C} \to \widehat{\mathbb{C}}$ on the Yoneda embedding $\yen\colon \mathbb{C} \to \widehat{\mathbb{C}}$ corresponds to a promonad on $\mathbb{C}$ (Altenkirch et al., 2010, Theorem 9). An Eilenberg-Moore algebra for $\mathscr{T}$ is a pair (G, χ) of an object $G \in \widehat{\mathbb{C}}$ and a natural transformation $\chi = \{\chi_Z\colon \widehat{\mathbb{C}}(\yen Z, G) \to \widehat{\mathbb{C}}(\mathscr{T}Z, G)\}_Z$ subject to some axioms (Altenkirch et al., 2010, Definition 3). Eilenberg-Moore algebras for a relative monad $\mathscr{T}$ are equivalent to algebras in the sense of Definition 3.1 for the promonad $\mathscr{A}$ defined by $\mathscr{A}(X, Y) = \mathscr{T}YX$. Especially, giving $\chi_Z\colon \widehat{\mathbb{C}}(\yen Z, G) \to \widehat{\mathbb{C}}(\mathscr{T}Z, G)$ natural in Z is equivalent to giving $\alpha\colon \mathscr{A}(Y, Z) \times GY \to GZ$ natural in Z and extranatural in Y because we have

$$
\begin{aligned}
&\int_{Z \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}\left(\widehat{\mathbb{C}}(\yen Z, G), \widehat{\mathbb{C}}(\mathscr{T}Z, G)\right) \\
&\cong \int_{Z \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}\left(GZ, \widehat{\mathbb{C}}(TZ, G)\right) \qquad \text{the Yoneda lemma} \\
&\cong \int_{Z \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}\left(GZ, \int_{Y \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}(\mathscr{T}ZY, GY)\right) \\
&\cong \int_{Z \in \mathbb{C}^{\mathrm{op}}} \int_{Y \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}\left(GZ, \mathbf{Set}(\mathscr{T}ZY, GY)\right) \\
&\cong \int_{Z \in \mathbb{C}^{\mathrm{op}}} \int_{Y \in \mathbb{C}^{\mathrm{op}}} \mathbf{Set}\left(\mathscr{A}(Y, Z) \times GZ, GY\right).
\end{aligned}
$$

Here, we used *end* $\int_X(-)$, which is the dual notion of coend. Uustalu (2010) introduced *strong relative monads*, which corresponds to strong promonads.

Pieters et al. (2020) introduced handlers for *monoidal effects*. In their framework, an *inductive handler* for arrows (without static parameters) is constructed by giving a 2-cell $(I_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A}) \Rightarrow \mathscr{A}$ in the bicategory of (strong) profunctors, where $\vec{\Sigma}(A, B) = \coprod_{(\mathrm{op}\colon \gamma \to \delta) \in \Sigma} \mathbb{C}(A, [\![\gamma]\!] \times ([\![\delta]\!] \Rightarrow B))$. A 2-cell $(I_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A}) \Rightarrow \mathscr{A}$ consists of

- a family of maps $\eta_{A,B}\colon \mathbb{C}(A, B) \to \mathscr{A}(A, B)$ natural in A and B, and
- families of maps $\iota_{\mathrm{op},A,B,C}\colon \mathbb{C}(A, [\![\gamma]\!] \times ([\![\delta]\!] \Rightarrow C)) \times \mathscr{A}(C, B) \to \mathscr{A}(A, B)$ natural in A and B and extranatural in C.

From this semantic structure, Pieters et al. (2020) defined syntax of inductive handlers for arrows (without static parameters) as follows.

```
ihandler
| ε (f: A→B) ↦ N : P(A,B)
| op (k: A→(γ,δ→C), l: P(C,B)) ↦ M_op : P(A,B)
```

From an inductive handler we can obtain a handler in the sense of this paper because we have the following map in **Set**:

Proposition 6.1. *There is a map*

$$\Xi\colon \int_A \int_B \mathbf{Set}\left((I_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A})(A, B), \mathscr{A}(A, B)\right)$$
$$\to \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \twoheadrightarrow \delta} \mathbf{Set}(\mathscr{A}(\llbracket \delta \rrbracket, B), \mathscr{A}(\llbracket \gamma \rrbracket, B))$$

for any set B.

Hence, given a 2-cell $h\colon (I_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A}) \Rightarrow \mathscr{A}$, we have $\pi_1(\Xi(h)) \in \mathscr{A}(B, B)$ and $\{\pi_{\mathsf{op}}(\pi_2(\Xi(h)))\colon \mathscr{A}(\llbracket \delta \rrbracket, B) \to \mathscr{A}(\llbracket \gamma \rrbracket, B)\}_{\mathsf{op}}$, which determine a handler in the sense of this paper. Note that inductive handlers cannot modify the answer type whereas handlers in the sense of this paper can.

7 Conclusion and future work

We have introduced an arrow calculus with operations and handlers, and defined its operational semantics and denotational semantics. The calculus design is based on categorical observations. The preservation and progress theorems are proved. We have also proved soundness and adequacy.

For future work, we plan to investigate the following topics.

λ_{flow} is a calculus with handlers for monads, arrows and idioms (Lindley (2014)). The relationship between the arrow calculus with operations and handlers and λ_{flow} is to be investigated. Is there translation from the arrow calculus with operations and handlers to Lindley's λ_{flow}?

Combination of handlers for monads and arrows in categorical way is interesting to investigate. Monoidal effects by Pieters et al. (2020) are one answer to this question. As another answer, can we use double categorical frameworks, focusing on the relationship between monads and promonads?

An algebraic theory corresponds to a finitary monad (Adamek and Rosicky (1994)). Can we develop a promonad version of such theory?

We are investigating further examples of the use of arrow calculus with operations and handlers. As explained in Section 1.2, we cannot use conditional branching on the output of effects in the arrow calculus. This feature is useful to implement neural network for instance. In neural network programming, we usually do not change layers dynamically depends on the output of previous layers. Moreover, handlers enable us to separate the structure of networks and their implementation.

Conflicts of interest

None.

References

Adamek, J. & Rosicky, J. (1994) *Locally Presentable and Accessible Categories*. London Mathematical Society Lecture Note Series. Cambridge University Press.

Altenkirch, T., Chapman, J. & Uustalu, T. (2010) Monads need not be endofunctors. Foundations of Software Science and Computational Structures. Berlin, Heidelberg. Springer Berlin Heidelberg. pp. 297–311.

Asada, K. (2010) Arrows are strong monads. Proceedings of the Third ACM SIGPLAN Workshop on Mathematically Structured Functional Programming. New York, NY, USA. Association for Computing Machinery. p. 33–42.

Asada, K. & Hasuo, I. (2010) Categorifying computations into components via arrows as profunctors. *Electronic Notes in Theoretical Computer Science*. **264**(2), 25–45. Proceedings of the Tenth Workshop on Coalgebraic Methods in Computer Science (CMCS 2010).

Atkey, R. (2011) What is a categorical model of arrows? *Electronic Notes in Theoretical Computer Science*. **229**(5), 19–37. Proceedings of the Second Workshop on Mathematically Structured Functional Programming (MSFP 2008).

Bauer, A. & Pretnar, M. (2013) An effect system for algebraic effects and handlers. Algebra and Coalgebra in Computer Science. Berlin, Heidelberg. Springer Berlin Heidelberg. pp. 1–16.

Borceux, F. (1994) *Handbook of Categorical Algebra*. vol. 1 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press.

Bénabou, J. (2000) Distributors at work. Lecture notes by Thomas Streicher, https://www2.mathematik.tu-darmstadt.de/~streicher/FIBR/DiWo.pdf.

Fujii, S. (2021) Introduction to profunctors. Available on YouTube, https://www.youtube.com/playlist?list=PLOtyxrAiMd3BvBc0XSU6JB6rKmrmhgDNG.

Heunen, C. & Jacobs, B. (2006) Arrows, like monads, are monoids. *Electronic Notes in Theoretical Computer Science*. **158**, 219–236. Proceedings of the 22nd Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXII).

Hughes, J. (2000) Generalising monads to arrows. *Science of Computer Programming*. **37**(1), 67–111.

Hughes, J. (2004) Programming with arrows. Advanced Functional Programming.

Jacobs, B., Heunen, C. & Hasuo, I. (2009) Categorical semantics for arrows. *J. Funct. Program.* **19**, 403–438.

Kelly, G. M. (1982) *Basic concepts of enriched category theory*. Cambridge University Press Cambridge. New York.

Leinster, T. (2014) *Basic Category Theory*. Cambridge Studies in Advanced Mathematics. Cambridge University Press.

Lindley, S. (2014) Algebraic effects and effect handlers for idioms and arrows. Proceedings of the 10th ACM SIGPLAN Workshop on Generic Programming. ACM. pp. 47–58.

Lindley, S., Wadler, P. & Yallop, J. (2010) The arrow calculus. *Journal of Functional Programming*. **20**(1), 51–69.

Lindley, S., Wadler, P. & Yallop, J. (2011) Idioms are oblivious, arrows are meticulous, monads are promiscuous. *Electronic Notes in Theoretical Computer Science*. **229**(5), 97–117. Proceedings of

the Second Workshop on Mathematically Structured Functional Programming (MSFP 2008).

Loregian, F. (2021) *(Co)end Calculus*. London Mathematical Society Lecture Note Series. Cambridge University Press.

Mac Lane, S. (1971) *Categories for the Working Mathematician*. Springer-Verlag. New York. Graduate Texts in Mathematics, Vol. 5.

Moggi, E. (1989) Computational lambda-calculus and monads. Fourth Annual Symposium on Logic in Computer Science. pp. 14–23.

Moggi, E. (1991) Notions of computation and monads. *Information and Computation*. **93**(1), 55–92. Selections from 1989 IEEE Symposium on Logic in Computer Science.

nLab. (2021) algebra for a profunctor. nLab, Ritrieved November 7, 2023, from `https://ncatlab.org/nlab/show/algebra+for+a+profunctor`.

Paterson, R. A. (2001) A new notation for arrows. ACM SIGPLAN International Conference on Functional Programming.

Pieters, R. P., Rivas, E. & Schrijvers, T. (2020) Generalized monoidal effects and handlers. *Journal of Functional Programming*. **30**, e23.

Plotkin, G. & Power, J. (2001a) Adequacy for algebraic effects. Foundations of Software Science and Computation Structures. Berlin, Heidelberg. Springer Berlin Heidelberg. pp. 1–24.

Plotkin, G. & Power, J. (2001b) Semantics for algebraic operations. *Electronic Notes in Theoretical Computer Science*. **45**, 332–345. MFPS 2001.

Plotkin, G. D. & Pretnar, M. (2013) Handling algebraic effects. *Logical Methods in Computer Science*. **Volume 9, Issue 4**.

Sanada, T. (2023) Category-Graded Algebraic Theories and Effect Handlers. *Electronic Notes in Theoretical Informatics and Computer Science*. **Volume 1 - Proceedings of MFPS XXXVIII**.

Street, R. (1972) The formal theory of monads. *Journal of Pure and Applied Algebra*. **2**(2), 149–168.

Uustalu, T. (2010) Strong relative monads. Short contribution in International Workshop on Coalgebraic Methods in Computer Science 2010.

Wood, R. J. (1985) Proarrows II. *Cahiers de Topologie et Géométrie Différentielle Catégoriques*. **26**(2), 135–168.

Yallop, J. (2010) *Abstraction for web programming*. Ph.D. thesis. University of Edinburgh.

1 Coends

In this section, we review the definition and construction of coends. We also give an informal description of coends.

Definition 1.1 (extranatural transformation)**.** Let $F, G\colon \mathbb{C}^{\mathrm{op}} \times \mathbb{C} \to \mathbb{D}$ be functors. An *extranatural transformation* ϕ from F to G, we write $\phi\colon F \Rightarrow G$, is a family of morphisms $(\phi_C\colon F(C,C) \to G(C,C))_{C \in \mathrm{Ob}(\mathbb{C})}$ such that the following diagram commutes for any morphisms $f\colon C \to C'$ in $\mathbb{C}$:

$$
\begin{array}{ccc}
& F(C,C) \xrightarrow{\ \phi_C\ } G(C,C) & \\
F(f,C) \nearrow & & \searrow G(C,f) \\
F(C',C) & & G(C,C') \\
F(C',f) \searrow & & \nearrow G(f,C') \\
& F(C',C') \xrightarrow{\ \phi_{C'}\ } G(C',C') &
\end{array}
$$

In the following, we deal only with extranatural transformations whose codomain (G) is a constant functor.

A coend is a pair of an object and an extranatural transformation defined for a functor. It has a universal property like a colimit.

Definition 1.2 (coend)**.** Let $F\colon \mathbb{C}^{\mathrm{op}} \times \mathbb{C} \to \mathbb{D}$ be a functor. A *coend* of F is a pair of an object $\int^{C \in \mathbb{C}} F(C,C) \in \mathbb{D}$ and an extranatural transformation

$$
\omega\colon F \Rightarrow \int^{C \in \mathbb{C}} F(C,C)
$$

satisfying the following universal property. If $\phi\colon F \Rightarrow X$ is an extranatural transformation to an object $X \in \mathbb{D}$, then there exists a unique morphism $\kappa\colon \int^{C \in \mathbb{C}} F(C,C) \to X$ such that $\phi = \kappa \circ \omega$:

The existence of a coend of $F\colon \mathbb{C}^{\mathrm{op}} \times \mathbb{C} \to \mathbb{D}$ depends on the properties of $\mathbb{C}$, $\mathbb{D}$ and F. The following proposition is known for the cases we often use in this paper.

Proposition 1.3. *If $\mathbb{C}$ is small, for the functor $F\colon \mathbb{C}^{\mathrm{op}} \times \mathbb{C} \to \mathbf{Set}$, the coend $\omega\colon F \Rightarrow \int^{C \in \mathbb{C}} F(C,C)$ of F always exists.*

Proof We only construct the set $\int^{C \in \mathbb{C}} F(C,C)$ and ω, and the proof of universality is left to the reader. Firstly, we define an equivalence relation $\sim$ on a set $\coprod_{C \in \mathrm{Ob}(\mathbb{C})} F(C,C)$ as follows. For $a \in F(C,C)$ and $b \in F(C',C')$, $a \sim b$ if there exists a morphism $f\colon C \to C'$ in $\mathbb{C}$ and $c \in F(C',C)$ such that $F(f,C)(c) = a$ and $F(C',f)(c) = b$.

$$
\begin{array}{ccc}
& F(C',C) & \\
{\scriptstyle F(f,C)}\swarrow & & \searrow{\scriptstyle F(C',f)} \\
F(C,C) & & F(C',C')
\end{array}
\qquad
\begin{array}{ccc}
& c & \\
{\scriptstyle F(f,C)}\swarrow & & \searrow{\scriptstyle F(C',f)} \\
a & & b
\end{array}
$$

We write $[a]$ for the equivalence class of $a \in F(C,C)$ in $\sim$. Now, we define

$$
\int^{C \in \mathbb{C}} F(C,C) := \left(\coprod_{C \in \mathrm{Ob}(\mathbb{C})} F(C,C) \right) / \sim .
$$

We define $\omega_C \colon F(C,C) \to \int^{C \in \mathbb{C}} F(C,C)$ as the canonical injection:

$$
\omega_C \colon F(C,C) \to \int^{C \in \mathbb{C}} F(C,C)
$$
$$
a \mapsto [a].
$$

∎

Informally, we can think of $\int^{C \in \mathbb{C}}$ as an existential quantifier $\exists$ over $C \in \mathrm{Ob}(\mathbb{C})$. An element w of $\int^{C \in \mathbb{C}} F(C,C)$ is regarded as a witness of a proposition that there exists $C \in \mathrm{Ob}(\mathbb{C})$ such that $F(C,C)$ holds.

2 2-categories, bicategories and enriched categories

2.1 2-categories

Roughly speaking, a 2-category is a category whose hom-sets have a categorical structure.

Definition 2.1 (2-categories). A 2-*category* $\mathbf{C}$ consists of the following data.

- A class $\mathrm{Ob}(\mathbf{C})$. We call an element a of $\mathrm{Ob}(\mathbf{C})$ an *object* or a 0-*cell*.
- A family $\{\mathbf{C}(a,b)\}_{a,b \in \mathrm{Ob}(\mathbf{C})}$ of categories, called *hom-categories*. We call an object f of $\mathbf{C}(a,b)$ a *morphism* or a 1-*cell* from a to b of $\mathbf{C}$. A morphism $\alpha\colon f \to g$ in $\mathbf{C}(a,b)$ is called a 2-*cell* from f to g of $\mathbf{C}$.
- An *identity* functor $\mathrm{id}_a \colon \mathbf{1} \to \mathbf{C}(a,a)$ for each $a \in \mathrm{Ob}(\mathbf{C})$.
- A *composition* functor $\mathbf{c}_{a,b,c} \colon \mathbf{C}(b,c) \times \mathbf{C}(a,b) \to \mathbf{C}(a,c)$ for each $a,b,c \in \mathrm{Ob}(\mathbf{C})$

subject to the following axioms expressed by the (strict) commutativity of

$$
\begin{array}{ccc}
(\mathbf{C}(c,d) \times \mathbf{C}(b,c)) \times \mathbf{C}(a,b) & \xrightarrow{\;\mathbf{c}\times\mathbf{C}(a,b)\;} & \mathbf{C}(b,d) \times \mathbf{C}(a,b) \\
\Big\downarrow{\scriptstyle\cong} & & \Big\downarrow{\scriptstyle\mathbf{c}} \\
\mathbf{C}(c,d) \times (\mathbf{C}(b,c) \times \mathbf{C}(a,b)) \xrightarrow[\mathbf{C}(c,d)\times\mathbf{c}]{} \mathbf{C}(c,d) \times \mathbf{C}(a,c) & \xrightarrow{\;\mathbf{c}\;} & \mathbf{C}(a,d)
\end{array}
$$

$$\text{(2.1)}$$

$$
\begin{array}{ccc}
\mathbf{1} \times \mathbf{C}(a,b) & & \mathbf{C}(a,b) \times \mathbf{1} \\
{\scriptstyle \mathrm{id}_b \times \mathbf{C}(a,b)}\Big\downarrow \quad \searrow{\scriptstyle\cong} & & {\scriptstyle\cong}\swarrow \quad \Big\downarrow{\scriptstyle \mathbf{C}(a,b)\times\mathrm{id}_a} \\
\mathbf{C}(b,b) \times \mathbf{C}(a,b) \xrightarrow[\mathbf{c}]{} \mathbf{C}(a,b) & \xleftarrow[\mathbf{c}]{} & \mathbf{C}(a,b) \times \mathbf{C}(a,a)
\end{array}
$$

$$\text{(2.2)}$$

We write $f \colon a \to b$ for a 1-cell f from a to b, and $\alpha \colon f \Rightarrow g$ for a 2-cell α from f to g:

$$
a \underset{g}{\overset{f}{\rightrightarrows}} \; {\Downarrow \alpha} \; b.
$$

We also write $g \circ f$ for the composition $\mathbf{c}(g,f)$ of 1-cells $f \colon a \to b$ and $g \colon b \to c$.

For 1-cells, the axiom (2.1) is the associativity of composition and (2.2) is the unitality of composition. Let $f \colon a \to b$, $g \colon b \to c$ and $h \colon c \to d$ be 1-cells in a 2-category $\mathbf{C}$. The associativity of composition (2.1) for the 1-cells f, g and h is

$$
(h \circ g) \circ f = h \circ (g \circ f) \tag{2.3}
$$

in the hom-category $\mathbf{C}(a,d)$, and the unitality (2.2) for the 1-cell f is

$$
(\mathrm{id}_b \circ f) = f = (f \circ \mathrm{id}_a) \tag{2.4}
$$

in the hom-category $\mathbf{C}(a,b)$.

An example of 2-categories is the 2-category **Cat**, whose 0-cells, 1-cells and 2-cells are small categories, functors and natural transformation, respectively.

2-category theory is a formal language to describe the ordinary category theory. For example, a definition of monads *in* 2-categories is as follows.

Definition 2.2 (monads in 2-categories)**.** Let $\mathbf{C}$ be a 2-category. A *monad* in $\mathbf{C}$ is an endo 1-cell $t \colon c \to c$ equipped with

- a 2-cell $\eta \colon \mathrm{id}_c \Rightarrow t$ called a *unit* and
- a 2-cell $\mu \colon t \circ t \Rightarrow t$ called a *multiplication*

such that the following axioms hold:

$$t \xrightarrow{c} \overset{t}{\underset{t}{\Downarrow \mu}} c \xrightarrow{t} c \xrightarrow{t} \Downarrow \mu \xrightarrow{t} \quad = \quad t \xrightarrow{c} c \xrightarrow{t} c \xrightarrow{t} \Downarrow \mu \xrightarrow{t} c$$

Monads in the 2-category **Cat** in the sense of Definition 2.2 coincide with ordinary monads.

2.2 Bicategories

In the definition of 2-categories, the axiom (2.1) and (2.2) are strict in the sense that the equalities hold in the hom-categories. From a category-theoretic principle, these axioms may be too strict because we want to identify two functors in (2.1) and (2.2) which are not only strictly equal, but *naturally isomorphic*.

Definition 2.3 (bicategories)**.** A *bicategory* **C** consists of the same data in the definition of 2-categories (Definition 2.1) which make (2.1) and (2.2) commute *up to natural isomorphism*.

For 1-cells $f\colon a \to b$, $g\colon b \to c$ and $h\colon c \to d$ in a bicategory **C**, the associativity of composition (2.1) and the unitality (2.2) are expressed as follows:

$$(h \circ g) \circ f \cong h \circ (g \circ f) \qquad \text{in the category } \mathbf{C}(a, d), \qquad (2.5)$$

$$(\mathrm{id}_b \circ f) \cong f \cong (f \circ \mathrm{id}_a) \qquad \text{in the category } \mathbf{C}(a, b). \qquad (2.6)$$

The equalities in (2.3) and (2.4) are replaced by the isomorphisms.

We can obtain a definition of monads *in* bicategories from the definition of monads in 2-categories (Definition 2.2).

2.3 Enriched categories

In the definition of 2-categories (Definition 2.1), the monoidal structure of the category of small categories, we write $\mathbf{Cat}_0 = (\mathbf{Cat}_0, \times, \mathbf{1})$, are essential to describe the identity functors id_a and composition functors $\mathbf{c}_{a,b,c}$ and the axioms. The descriptions of the identity and composition in ordinary categories also use the monoidal structure of $\mathbf{Set} = (\mathbf{Set}, \times, 1)$. We generalise (2-)categories from this perspective and obtain the following definition of *enriched categories*.

Definition 2.4 (enriched category)**.** Let $\mathbb{V} = (\mathbb{V}, \otimes, J)$ be a symmetric monoidal category. A $\mathbb{V}$-*enriched category* (or simply a $\mathbb{V}$-*category*) $\mathbb{C}$ consists of the following data.

- A class $\mathrm{Ob}(\mathbb{C})$ of *objects*.
- A family of objects $\{\mathbb{C}(a, b)\}_{a,b \in \mathrm{Ob}(\mathbb{C})}$ of $\mathbb{V}$. We call $\mathbb{C}(a, b) \in \mathbb{V}$ a *hom-object*.
- A morphism $\mathrm{id}_a\colon J \to \mathbb{C}(a, a)$ for each $a \in \mathrm{Ob}(\mathbb{C})$, called an *identity*.
- A morphism $\mathbf{c}_{a,b,c}\colon \mathbb{C}(b, c) \otimes \mathbb{C}(a, b) \to \mathbb{C}(a, c)$ for each $a, b, c \in \mathrm{Ob}(\mathbb{C})$, called a *composition*

subject to the following axioms expressed by the commutativity of

$$
\begin{array}{ccc}
(\mathbb{C}(c,d) \otimes \mathbb{C}(b,c)) \otimes \mathbb{C}(a,b) & \xrightarrow{\;\mathbf{c} \otimes \mathbb{C}(a,b)\;} & \mathbb{C}(b,d) \otimes \mathbb{C}(a,b) \\
\;\downarrow{\cong} & & \;\downarrow{\mathbf{c}} \\
\mathbb{C}(c,d) \otimes (\mathbb{C}(b,c) \otimes \mathbb{C}(a,b)) \xrightarrow[\mathbb{C}(c,d)\otimes\mathbf{c}]{} \mathbb{C}(c,d) \otimes \mathbb{C}(a,c) & \xrightarrow{\;\mathbf{c}\;} & \mathbb{C}(a,d)
\end{array}
\tag{2.7}
$$

$$
\begin{array}{ccccc}
J \otimes \mathbb{C}(a,b) & & & & \mathbb{C}(a,b) \otimes J \\
\;\downarrow{\mathrm{id}_b \otimes \mathbb{C}(a,b)} & \searrow{\cong} & & \nwarrow{\cong} & \;\downarrow{\mathbb{C}(a,b)\times\mathrm{id}_a} \\
\mathbb{C}(b,b) \otimes \mathbb{C}(a,b) \xrightarrow[\mathbf{c}]{} & \mathbb{C}(a,b) & \xleftarrow[\mathbf{c}]{} & \mathbb{C}(a,b) \times \mathbb{C}(a,a)
\end{array}
\tag{2.8}
$$

An ordinary category is a **Set**-enriched category and a 2-category in the sense of Definition 2.1 is a **Cat**$_0$-enriched category.

The enriched version of a functor between categories is called a $\mathbb{V}$-*functor*, which preserves $\mathbb{V}$-category structures. See Kelly (1982) for the definition.

3 Proofs for Section 4 (The arrow calculus with operations and handlers)

Proposition 4.1 (progress). *The following hold.*

1. *For any well-typed term $\diamond \vdash M : A$, there exists a term M' such that $M \to M'$ or M is a value.*
2. *For any well-typed command $\diamond \mathbin{\mathring{,}} \diamond \vdash P \mathbin{!} A$, one of following holds.*

   a. *There exists a command P' such that $P \to P'$.*
   b. *$P = \lfloor V \rfloor$ for some value V.*
   c. *$P = \mathscr{F}[\mathsf{op}(V)]$ for some operation op, value V, and context $\mathscr{F}$.*

Proof (1). By induction on the derivation of $\diamond \vdash M : A$.

In case T-VAR. This case cannot happen because the context is empty.

In case T-ABS. In this case, we have $M = \lambda x.N$ for some term N, and M is a value.

$$
\frac{x:A \vdash N:B}{\diamond \vdash \lambda x.N : A \to B} \; \text{T-ABS}.
$$

In case T-APP. We have

$$
\frac{\diamond \vdash M_1 : B \to A \quad \diamond \vdash M_2 : B}{\diamond \vdash M_1 M_2 : A} \; \text{T-ABS}.
$$

By the induction hypothesis, M_1 is a value or $M_1 \to M_1'$ for some M_1', and M_2 is a value or $M_2 \to M_2'$ for some M_2. If $M_1 \to M_1'$ holds for some M_1', then we have $M_1 M_2 \to M_1' M_2$. If M_1 is a value, then $M_1 = \lambda x.N$ for some N because $\diamond \vdash M_1 : A \to B$. We have two subcases: If M_2 is a value V, then we have $M_1 M_2 = (\lambda x.N)V \to N[V/x]$. If $M_2 \to M_2'$ holds for some M_2', then we have $M_1 M_2 = (\lambda x.N)M_2 \to (\lambda x.N)M_2'$.

In case T-CABS. In this case, we have $M = \lambda^\bullet x.N$ for some term N, and M is a value.

$$\frac{\diamond \, \fatsemi \, x : A \vdash N \,!\, B}{\diamond \vdash \lambda^\bullet x.N : A \rightsquigarrow B} \ \text{T-CABS}.$$

In case T-PAIR.

$$\frac{\diamond \vdash M_1 : A \quad \diamond \vdash M_2 : B}{\diamond \vdash \langle M_1, M_2 \rangle : A \times B} \ \text{T-PAIR}.$$

By the induction hypothesis, M_1 is a value or $M_1 \to M_1'$ for some M_1', and M_2 is a value or $M_2 \to M_2'$ for some M_2. If $M_1 \to M_1'$ holds for some M_1', then we have $\langle M_1, M_2 \rangle \to \langle M_1', M_2 \rangle$. If M_1 is a value V_1, we have two subcases: if M_2 is a value V_2, then $M = \langle M_1, M_2 \rangle = \langle V_1, V_2 \rangle$ is a value. If $M_2 \to M_2'$ holds for some M_2', then we have $\langle M_1, M_2 \rangle = \langle V_1, M_2 \rangle \to \langle V_1, M_2' \rangle$.

In case T-FST, T-SND. Straightforward.

(2). By induction on the derivation of $\diamond \, \fatsemi \, \diamond \vdash P \,!\, A$.

In case T-PURE. We have

$$\frac{\diamond \vdash M : A}{\diamond \, \fatsemi \, \diamond \vdash \lfloor M \rfloor \,!\, A} \ \text{T-PURE}$$

for some term M. By (1), M is a value or $M \to M'$ holds for some M'. If M is a value V, then $P = \lfloor V \rfloor$ and this satisfies (b). If $M \to M'$ holds for some M', then we have $P = \lfloor M \rfloor \to \lfloor M' \rfloor$.

In case T-CAPP. We have

$$\frac{\diamond \vdash L : B \rightsquigarrow A \quad \diamond \vdash N : B}{\diamond \, \fatsemi \, \diamond \vdash L \bullet N \,!\, A} \ \text{T-CAPP}.$$

By the similar argument to **Case** T-APP, we have either $L \bullet N \to L' \bullet N$ for some L', $L = (\lambda^\bullet x.L_1)$ and $L \bullet N \to L \bullet N'$ for some N', or $L = (\lambda^\bullet x.L_1)$, N is a value, and $L \bullet N = (\lambda^\bullet x.L_1) \bullet V \to L_1[V/x]$ for some N'.

In case T-LET. We have

$$\frac{\diamond \, \fatsemi \, \diamond \vdash P \,!\, B \quad \diamond \, \fatsemi \, x : B \vdash Q \,!\, A}{\diamond \, \fatsemi \, \diamond \vdash \mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q \,!\, A} \ \text{T-LET}.$$

By the induction hypothesis, we have the following three cases.

1. $P \to P'$ for some command P'. We have $(\mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q) \to (\mathbf{let}\, x \Leftarrow P' \,\mathbf{in}\, Q)$.
2. $P = \lfloor V \rfloor$ for some value term V. We have $(\mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q) = (\mathbf{let}\, x \Leftarrow \lfloor V \rfloor \,\mathbf{in}\, Q) \to Q[V/x]$.
3. $P = \mathscr{F}[\mathsf{op}(V)]$ for some value term V, operation symbol $\mathsf{op} \in \Sigma$ and context $\mathscr{F}$. We have $P = \mathscr{F}'[\mathsf{op}(V)]$ for $\mathscr{F}' = (\mathbf{let}\, x \Leftarrow \mathscr{F} \,\mathbf{in}\, Q)$.

In case T-OP. We have

$$\frac{\mathsf{op} : \gamma \twoheadrightarrow \delta \quad \diamond \vdash M : \gamma}{\diamond \, \fatsemi \, \diamond \vdash \mathsf{op}(M) \,!\, \delta} \ \text{T-OP}.$$

By the induction hypothesis, M is a value or $M \to M'$ for some M'. If M is a value V, then $P = \mathsf{op}(V) = \mathscr{F}[\mathsf{op}(V)]$ for $\mathscr{F} = [-]$. If $M \to M'$ holds, then $P = \mathsf{op}(M) \to \mathsf{op}(M')$.

In case T-HANDLE. We have

$$\frac{\diamond \,\overset{\circ}{\circ}\, \diamond \vdash R \,!\, C \quad \vdash H : C \Rightarrow D}{\diamond \,\overset{\circ}{\circ}\, \diamond \vdash \mathbf{handle}\ R\ \mathbf{with}\ H \,!\, D}\ \text{T-HANDLE}$$

where $H = \{\, \overset{\circ}{\circ}\, x : C \mapsto P\} \cup \{k : \delta \rightsquigarrow D \,\overset{\circ}{\circ}\, z : \gamma \vdash Q_{\mathsf{op}}\}_{\mathsf{op} \in \Sigma}$. By the induction hypothesis, we have the following three cases.

1. $R \to R'$ for some command R'. We have $(\mathbf{handle}\ R\ \mathbf{with}\ H) \to (\mathbf{handle}\ R'\ \mathbf{with}\ H)$.
2. $R = \lfloor V \rfloor$ for some value term V. We have

$$(\mathbf{handle}\ R\ \mathbf{with}\ H) = (\mathbf{handle}\ \lfloor V \rfloor\ \mathbf{with}\ H) \to P[V/x].$$

3. $R = \mathscr{F}[\mathsf{op}(V)]$ for some value term V, operation symbol $\mathsf{op} \in \Sigma$ and context $\mathscr{F}$. We have

$$(\mathbf{handle}\ R\ \mathbf{with}\ H) = (\mathbf{handle}\ \mathscr{F}[\mathsf{op}(V)]\ \mathbf{with}\ H)$$
$$\to Q_{\mathsf{op}}[V/z, (\lambda^{\bullet}y.\ \mathbf{handle}\lfloor y \rfloor\ \mathbf{with}\ H)/k].$$

∎

4 Proofs for Section 5 (Denotational semantics)

Proposition 5.4 (normal form)**.** *Let A and B be sets. For any $a \in \mathrm{Arr}_{\Sigma}(A, B)$, there exist a natural number $n \in \mathbb{N}$, a sequence of operations $(\mathsf{op}_i : \gamma_i \to \delta_i)_{i=1,\ldots,n}$, a sequence of maps $(f_i : [\![\delta_{i-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A \to [\![\gamma_i]\!])_{i=1,\ldots,n}$ and $g : [\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A \to B$ such that*

$$
\begin{aligned}
a \sim \mathrm{arr}(d_A) &\ggg \mathrm{first}_A(\mathrm{arr}(f_1) \ggg \mathsf{op}_1) \\
&\ggg \mathrm{arr}(d_{[\![\delta_1]\!] \times A}) \ggg \mathrm{first}_{[\![\delta_1]\!] \times A}(\mathrm{arr}(f_2) \ggg \mathsf{op}_2) \\
&\ggg \cdots \\
&\ggg \mathrm{arr}(d_{[\![\delta_{n-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{first}_{[\![\delta_{n-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A}(\mathrm{arr}(f_n) \ggg \mathsf{op}_n) \\
&\ggg \mathrm{arr}(g)
\end{aligned}
\tag{4.10}
$$

where $d_X : X \to X \times X$ is the diagonal map: $d_X(x) = (x, x)$. We write the arrow term of the right-hand side as $\mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n}, (f_i)_{i=1,\ldots,n}; g)$.

Proof We prove by induction on the structure of $a \in \mathrm{Arr}_{\Sigma}(A, B)$.

In case $a = \mathrm{arr}(f)$ **for some** $f : A \to B$. This case is proved in the main text.

In case $a = \mathsf{op}$ **for some** $(\mathsf{op} : \gamma \to \delta) \in \Sigma$ **with** $[\![\gamma]\!] = A$ **and** $[\![\delta]\!] = B$. This case is proved in the main text.

In case $a = \mathrm{first}_X(a')$ **for some** $a' \in \mathrm{Arr}_{\Sigma}(A', B')$ **with** $A' \times X = A$ **and** $B' \times X = B$. By the induction hypothesis, we have

$$a' \sim \mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n}, (f_i)_{i=1,\ldots,n}; g)$$

for some $n \in \mathbb{N}$, a sequence of operations $(\mathsf{op}_i : \gamma_i \to \delta_i)_{i=1,\ldots,n}$, a sequence of maps $(f_i' : [\![\delta_{i-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A' \to [\![\gamma_i]\!])_{i=1,\ldots,n}$ and $g' : [\![\delta_m]\!] \times \cdots \times [\![\delta_1]\!] \times A' \to B'$.

We show $a = \mathrm{first}_X(a') \sim \mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n}, (f_i')_{i=1,\ldots,n}; g \times \mathrm{id}_X)$ for some f_i' ($i = 1, \ldots, n$) by induction on n.

In the base case, $n = 0$, we have

$$
\begin{aligned}
a &= \mathrm{first}_X(a') \\
&\sim \mathrm{first}_X(\mathrm{cf}\,((),();g)) \\
&= \mathrm{first}_X(\mathrm{arr}(g)) \\
&\sim \mathrm{arr}(g \times \mathrm{id}_X) \\
&= \mathrm{cf}\,((),();g \times \mathrm{id}_X)
\end{aligned}
$$

We assume that the claim holds in case n, and show the claim in case $n + 1$. We have

$$
\begin{aligned}
a &= \mathrm{first}_X(a') \\
&\sim \mathrm{first}_X(\mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n+1}, (f_i)_{i=1,\ldots,n+1};g)) \\
&= \mathrm{first}_X(\mathrm{arr}(d_{A'}) \ggg \mathrm{first}_{A'}(\mathrm{arr}(f_1) \ggg \mathsf{op}_1) \ggg \mathrm{cf}\,((\mathsf{op}_{1+i})_{i=1,\ldots,n}, (f_{1+i})_{i=1,\ldots,n};g)) \\
&\sim \mathrm{first}_X(\mathrm{arr}(d_{A'}) \ggg \mathrm{first}_{A'}(\mathrm{arr}(f_1) \ggg \mathsf{op}_1)) \\
&\qquad \ggg \mathrm{first}_X(\mathrm{cf}\,((\mathsf{op}_{1+i})_{i=1,\ldots,n}, (f_{1+i})_{i=1,\ldots,n};g)) \\
&\sim \mathrm{arr}(d_{A' \times X}) \ggg \mathrm{first}_{A' \times X}(\mathrm{arr}(f_1 \circ \pi_1) \ggg \mathsf{op}_1) \\
&\qquad \ggg \mathrm{first}_X(\mathrm{cf}\,((\mathsf{op}_{1+i})_{i=1,\ldots,n}, (f_{1+i})_{i=1,\ldots,n};g)) \\
&\sim \mathrm{arr}(d_A) \ggg \mathrm{first}_A(\mathrm{arr}(f_1 \circ \pi_1) \ggg \mathsf{op}_1) \\
&\qquad \ggg \mathrm{cf}\,((\mathsf{op}_{1+i})_{i=1,\ldots,n}, (f'_{1+i})_{i=1,\ldots,n};g \times \mathrm{id}_X) \\
&\sim \mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,n}, (f'_i)_{i=1,\ldots,n};g \times \mathrm{id}_X)
\end{aligned}
$$

where $f'_1 = f_1 \circ \pi_1 : A' \times X \to [\![\gamma_1]\!]$. See Figure 9.

In case $a = b \ggg c$ **for some** $b \in \mathrm{Arr}_\Sigma(A,X)$ **and** $c \in \mathrm{Arr}_\Sigma(X,B)$. By the induction hypothesis, we have

$$
b \sim \mathrm{cf}\,((\mathsf{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m};g')
$$

for some $m \in \mathbb{N}$, $(\mathsf{op}_i \colon \gamma_i \rightarrowtail \delta_i)_{i=1,\ldots,m}$, $(f_i \colon [\![\delta_{i-1}]\!] \times \cdots \times [\![\delta_1]\!] \times A \to [\![\gamma_i]\!])_{i=1,\ldots,m}$ and $g' \colon [\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A \to X$, and

$$
c \sim \mathrm{cf}\,((\mathsf{op}_{m+i})_{i=1,\ldots,m}, (f'_i)_{i=1,\ldots,m};g'')
$$

for some $n \in \mathbb{N}$, $(\mathsf{op}_{m+i} \colon \gamma_{m+i} \rightarrowtail \delta_{m+i})_{i=1,\ldots,n}$, $(f'_i \colon [\![\delta_{m+i-1}]\!] \times \cdots \times [\![\delta_{m+1}]\!] \times A \to [\![\gamma_{m+i}]\!])_{i=1,\ldots,n}$ and $g'' \colon [\![\delta_{m+n+1}]\!] \times \cdots \times [\![\delta_{m+1}]\!] \times X \to B$. We show $a = b \ggg c \sim \mathrm{cf}\,((\mathsf{op})_{i=1,\ldots,m+n}, (f_i)_{i=1,\ldots,m+n};g)$ for some $(f_i)_{i=m+1,\ldots,m+n}$ and g by induction on n.

In the base case, $n = 0$, we have

$$
\begin{aligned}
a &= b \ggg c \\
&\sim \mathrm{cf}\,((\mathsf{op})_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m};g') \ggg \mathrm{cf}\,((),();g'') \\
&= \mathrm{cf}\,((\mathsf{op})_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m};g') \ggg \mathrm{arr}(g'') \\
&\sim \mathrm{cf}\,((\mathsf{op})_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m};g'' \circ g')\,.
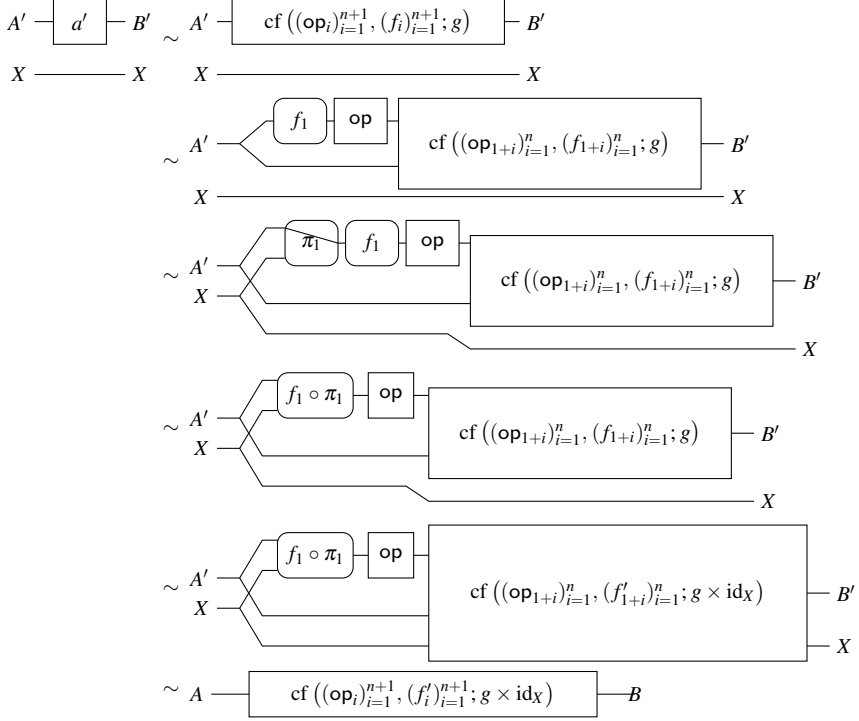\end{aligned}
$$

Fig. 9. In case $a = \mathrm{first}_X(a')$ and $a' \sim \mathrm{cf}\left((\mathrm{op})_{i=1}^{n+1}, (f_i)_{i=1}^{n+1}; g\right)$.

We assume that the claim holds in case n, and show the claim in case $n+1$. We have

$$a = b \ggg c$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; g'\right) \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=1,\ldots,n+1}, (f_i')_{i=1,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right) \ggg \mathrm{arr}(g')$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=1,\ldots,n+1}, (f_i')_{i=1,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right)$$
$$\qquad \ggg \mathrm{arr}(g') \ggg \mathrm{arr}(d_X) \ggg \mathrm{first}(\mathrm{arr}(f_1') \ggg \mathrm{op}_{m+1})$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=2,\ldots,n+1}, (f_i')_{i=2,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right)$$
$$\qquad \ggg \mathrm{arr}(d_{[\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{arr}(g' \times g') \ggg \mathrm{arr}(f_1' \times \mathrm{id}) \ggg \mathrm{first}(\mathrm{op}_{m+1})$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=2,\ldots,n+1}, (f_i')_{i=2,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right)$$
$$\qquad \ggg \mathrm{arr}(d_{[\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{arr}((f_1' \circ g') \times g') \ggg \mathrm{first}(\mathrm{op}_{m+1})$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=2,\ldots,n+1}, (f_i')_{i=2,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right)$$
$$\qquad \ggg \mathrm{arr}(d_{[\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{first}(\mathrm{arr}(f_1' \circ g')) \ggg \mathrm{arr}(\mathrm{id} \times g') \ggg \mathrm{first}(\mathrm{op}_{m+1})$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=2,\ldots,n+1}, (f_i')_{i=2,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m}, (f_i)_{i=1,\ldots,m}; \mathrm{id}\right)$$
$$\qquad \ggg \mathrm{arr}(d_{[\![\delta_n]\!] \times \cdots \times [\![\delta_1]\!] \times A}) \ggg \mathrm{first}(\mathrm{arr}(f_1' \circ g')) \ggg \mathrm{first}(\mathrm{op}_{m+1}) \ggg \mathrm{arr}(\mathrm{id} \times g')$$
$$\qquad \ggg \mathrm{cf}\left((\mathrm{op}_{m+i})_{i=2,\ldots,n+1}, (f_i')_{i=2,\ldots,n+1}; g''\right)$$
$$\sim \mathrm{cf}\left((\mathrm{op}_i)_{i=1,\ldots,m+1}, (f_i)_{i=1,\ldots,m+1}; \mathrm{id} \times g'\right) \ggg \mathrm{cf}\left((\mathrm{op}_{m+1+i})_{i=1,\ldots,n}, (f_{1+i}')_{i=1,\ldots,n}; g''\right)$$
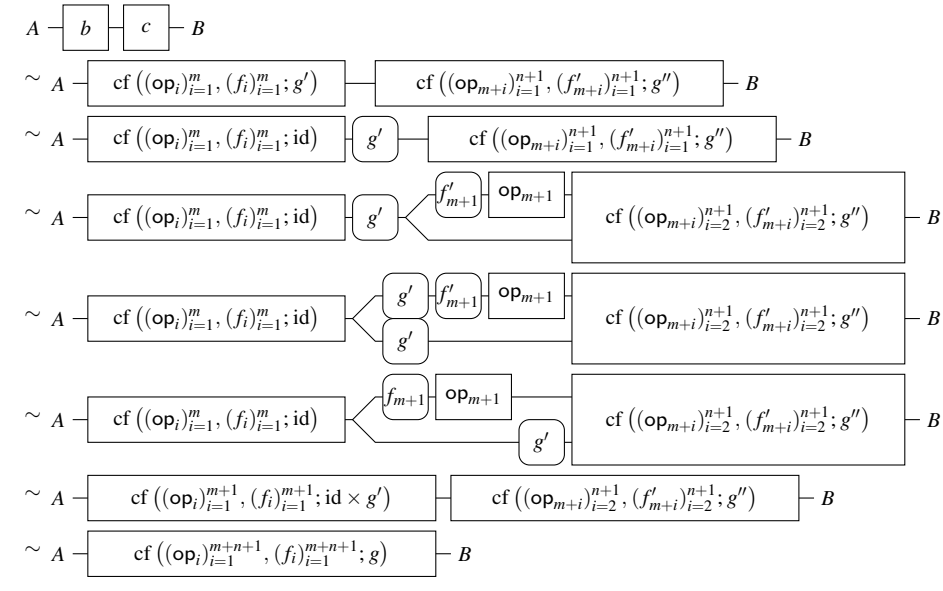
Fig. 10. In case $a = b \ggg c$ and $c \sim \mathrm{cf}\left((\mathsf{op}_i)_{i=1}^{n+1}, (f_i')_{i=1}^{n+1}; g''\right)$.

where $f_{m+1} = f_1' \circ g'$. See Figure 10. By the induction hypothesis, we obtain

$$a \sim \mathrm{cf}\left((\mathsf{op}_i)_{i=1,\ldots,m+1+n}, (f_i)_{i=1,\ldots,m+1+n}; g\right)$$

for some $(f_{m+1+i})_{i=1,\ldots,n}$ and g.

∎

Lemma 5.10. *The following hold.*

1. *If* $\Gamma, x : A \vdash M : B$ *and* $\Gamma \vdash V : A$, *then* $[\![M[V/x]]\!]^S(c) = [\![M]\!]^S(c, [\![V]\!]^S(c))$.
2. *If* $\Gamma, x : A \mathbin{\mathring{,}} \Delta \vdash P \mathbin{!} B$ *and* $\Gamma \vdash V : A$, *then* $[\![P[V/x]]\!]^S(c) = [\![P]\!]^S(c, [\![V]\!]^S(c))$.
3. *If* $\Gamma \mathbin{\mathring{,}} x : A, \Delta \vdash P \mathbin{!} B$ *and* $\Gamma, \Delta \vdash V : A$, *then*

$$[\![P[V/x]]\!]^S(c) = \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S})) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c, -)))) \ggg [\![P]\!]^S(c)$$

Proof The proof is by induction on the derivations.

(3). **In case** T-PURE. The derivation is

$$\dfrac{\Gamma, x : A, \Delta \vdash M : B}{\Gamma \mathbin{\mathring{,}} x : A, \Delta \vdash \lfloor M \rfloor \mathbin{!} B} \ \text{T-PURE}.$$

By the induction hypothesis (1), we have

$$[\![M[V/x]]\!]^S(c, -) = [\![M]\!]^S(c, [\![V]\!]^S(c, -), -) \circ d_{[\![\Delta]\!]^S}$$

for any $c \in [\![\Gamma]\!]^S$. Hence, we have

$$[\![\lfloor M \rfloor [V/x]]\!]^S(c) = [\![\lfloor M[V/x]\rfloor]\!]^S(c)$$
$$= \mathrm{arr}([\![M[V/x]]\!]^S(c,-) \circ \pi_1)$$
$$= \mathrm{arr}([\![M]\!]^S(c,[\![V]\!]^S(c,-),-) \circ d_{[\![\Delta]\!]^S} \circ \pi_1)$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{arr}([\![M]\!]^S(c,[\![V]\!]^S(c,-),-))$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{arr}([\![M]\!]^S(c,-,-) \circ ([\![V]\!]^S(c,-) \times \mathrm{id}_{[\![\Delta]\!]^S}))$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-)) \ggg \mathrm{arr}([\![M]\!]^S(c,-,-))$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-))) \ggg \mathrm{arr}(\pi_1) \ggg \mathrm{arr}([\![M]\!]^S(c,-,-))$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-))) \ggg [\![\lfloor M \rfloor]\!]^S(c).$$

In case T-CAPP. The derivation is

$$\frac{\Gamma \vdash L : C \rightsquigarrow B \quad \Gamma, x:A, \Delta \vdash M : C}{\Gamma \,\mathbin{\raisebox{0.3ex}{\scriptsize\(;\)}}\, x:A, \Delta \vdash L \bullet M \,!\, B} \;\text{T-CAPP}.$$

By the induction hypothesis (1), we have

$$[\![M[V/x]]\!]^S(c,-) = [\![M]\!]^S(c,[\![V]\!]^S(c,-),-) \circ d_{[\![\Delta]\!]^S}$$

for any $c \in [\![\Gamma]\!]^S$. Hence, we have

$$[\![(L \bullet M)[V/x]]\!]^S = [\![L \bullet (M[V/x])]\!]^S$$
$$= \mathrm{first}_S(\mathrm{arr}([\![M[V/x]]\!]^S(c,-))) \ggg [\![L]\!]^S(c)$$
$$= \mathrm{first}_S(\mathrm{arr}([\![M]\!]^S(c,[\![V]\!]^S(c,-),-) \circ d_{[\![\Delta]\!]^S})) \ggg [\![L]\!]^S(c)$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{arr}([\![M]\!]^S(c,[\![V]\!]^S(c,-),-))) \ggg [\![L]\!]^S(c)$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-)) \ggg \mathrm{arr}([\![M]\!]^S(c,-,-))) \ggg [\![L]\!]^S(c)$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-))) \ggg \mathrm{first}_S(\mathrm{arr}([\![M]\!]^S(c,-,-))) \ggg [\![L]\!]^S(c)$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}([\![V]\!]^S(c,-))) \ggg [\![L \bullet M]\!]^S(c).$$

In case T-LET. The derivation is

$$\frac{\Gamma \,\mathbin{\raisebox{0.3ex}{\scriptsize\(;\)}}\, x:A, \Delta \vdash P \,!\, C \quad \Gamma \,\mathbin{\raisebox{0.3ex}{\scriptsize\(;\)}}\, y:C, x:A, \Delta \vdash Q \,!\, B}{\Gamma \,\mathbin{\raisebox{0.3ex}{\scriptsize\(;\)}}\, x:A, \Delta \vdash \mathbf{let}\, y \Leftarrow P \,\mathbf{in}\, Q \,!\, B} \;\text{T-LET}.$$

By the induction hypothesis (3), we have

$$[\![P[V/x]]\!]^S(c) = \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg [\![P]\!]^S(c)$$

$$\llbracket Q[V/x] \rrbracket^S(c) = $$

$$= $$

Hence, we have

$$\llbracket (\textbf{let } y \Leftarrow P \textbf{ in } Q)[V/x] \rrbracket^S(c) = \llbracket \textbf{let } y \Leftarrow P[V/x] \textbf{ in } Q[V/x] \rrbracket^S(c)$$

$$= $$

$$= $$

$$= \mathrm{first}_S(\mathrm{arr}(d_{\llbracket \Delta \rrbracket^S}) \ggg \mathrm{first}_{\llbracket \Delta \rrbracket^S}(\mathrm{arr}(\llbracket V \rrbracket^S(c,-)))) \ggg \llbracket \textbf{let } y \Leftarrow P \textbf{ in } Q \rrbracket^S(c).$$

In case T-OP. The derivation is

$$\frac{\mathsf{op} : \gamma \rightarrowtail \delta \in \Sigma \quad \Gamma, x:A, \Delta \vdash M : \delta}{\Gamma \,\fatsemi\, x:A, \Delta \vdash \mathsf{op}(M) \,!\, \delta} \ \text{T-OP}.$$

By the induction hypothesis (1), we have

$$\llbracket M[V/x] \rrbracket^S(c,-) = \llbracket M \rrbracket^S(c, \llbracket V \rrbracket^S(c,-), -) \circ d_{\llbracket \Delta \rrbracket^S}$$

for any $c \in \llbracket \Gamma \rrbracket^S$. Hence, we have

$$\llbracket \mathsf{op}(M)[V/x] \rrbracket^S(c) = \llbracket \mathsf{op}(M[V/x]) \rrbracket^S(c)$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(\llbracket M[V/x] \rrbracket^S(c),-) \ggg \mathsf{op}$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(\llbracket M \rrbracket^S(c, \llbracket V \rrbracket^S(c,-),-) \circ d_{\llbracket \Delta \rrbracket^S}) \ggg \mathsf{op}$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(d_{\llbracket \Delta \rrbracket^S}) \ggg \mathrm{arr}(\llbracket M \rrbracket^S(c, \llbracket V \rrbracket^S(c,-),-)) \ggg \mathsf{op}$$
$$= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(d_{\llbracket \Delta \rrbracket^S}) \ggg \mathrm{first}_{\llbracket \Delta \rrbracket^S}(\llbracket V \rrbracket^S(c,-)) \ggg \mathrm{arr}(\llbracket M \rrbracket^S(c,-,-)) \ggg \mathsf{op}$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{\llbracket \Delta \rrbracket^S}) \ggg \mathrm{first}_{\llbracket \Delta \rrbracket^S}(\llbracket V \rrbracket^S(c,-))) \ggg \mathrm{arr}(\pi_1) \ggg \mathrm{arr}(\llbracket M \rrbracket^S(c,-,-)) \ggg \mathsf{op}$$
$$= \mathrm{first}_S(\mathrm{arr}(d_{\llbracket \Delta \rrbracket^S}) \ggg \mathrm{first}_{\llbracket \Delta \rrbracket^S}(\llbracket V \rrbracket^S(c,-))) \ggg \mathrm{arr}(\pi_1) \ggg \llbracket \mathsf{op}(M) \rrbracket^S(c).$$

In case T-HANDLE. The derivation is

$$\frac{\Gamma \,\fatsemi\, x:A, \Delta \vdash P \,!\, C \quad \vdash H : C \Rightarrow D}{\Gamma \,\fatsemi\, x:A, \Delta \vdash \textbf{handle } P \textbf{ with } H \,!\, D} \ \text{T-HANDLE}.$$

By the induction hypothesis (3), we have

$$[\![P[V/x]]\!]^S(c) = \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg [\![P]\!]^S(c).$$

Hence, we have

$$\begin{aligned}
&[\![(\mathbf{handle}\ P\ \mathbf{with}\ H)[V/x]]\!]^S(c) = [\![\mathbf{handle}\ P[V/x]\ \mathbf{with}\ H]\!]^S(c)\\
&= h\left([\![P[V/x]]\!]^S(c)\right)\\
&= h\left(\mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg [\![P]\!]^S(c)\right)\\
&= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg h\left([\![P]\!]^S(c)\right)\\
&= \mathrm{first}_S(\mathrm{arr}(d_{[\![\Delta]\!]^S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S}(\mathrm{arr}([\![V]\!]^S(c,-)))) \ggg [\![\mathbf{handle}\ P\ \mathbf{with}\ H]\!]^S(c).
\end{aligned}$$

$\blacksquare$

Lemma 5.11. *If* $\diamond\ \mathring{,}\ \diamond \vdash \mathscr{F}[\mathsf{op}(V)] \mathbin{!} C$, *then*

$$[\![\mathscr{F}[\mathsf{op}(V)]]\!]^S(\star) = \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S) \ggg \mathsf{op}) \ggg [\![\mathscr{F}[\lfloor y\rfloor]]\!]^S(\star)$$

holds.

Proof The proof is by induction on the structure of $\mathscr{F}$.

If $\mathscr{F} = [-]$, then we have

$$\begin{aligned}
[\![\mathsf{op}(V)]\!]^S(\star) &= \mathrm{arr}([\![V]\!]^S(\star) \circ \pi_1) \ggg \mathsf{op}\\
&= \mathrm{arr}(\pi_1) \ggg \mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}\\
&= \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}) \ggg \mathrm{arr}(\pi_1)\\
&= \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}) \ggg \mathrm{arr}(\pi_1)\\
&= \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}) \ggg [\![\lfloor y\rfloor]\!]^S(\star).
\end{aligned}$$

Otherwise, $\mathscr{F} = (\mathbf{let}\ x \Leftarrow \mathscr{F}'\ \mathbf{in}\ P)$, we have

$$\begin{aligned}
&[\![\mathscr{F}[\mathsf{op}(V)]]\!]^S(\star)\\
&= [\![\mathbf{let}\ x \Leftarrow \mathscr{F}'[\mathsf{op}(V)]\ \mathbf{in}\ P]\!]^S(\star)\\
&= \mathrm{arr}(d_{[\![\Delta]\!]^S \times S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S \times S}([\![\mathscr{F}'[\mathsf{op}(V)]]\!]^S(\star)) \ggg [\![P]\!]^S(\star)\\
&= \mathrm{arr}(d_{[\![\Delta]\!]^S \times S}) \ggg \mathrm{first}_{[\![\Delta]\!]^S \times S}(\mathrm{first}_S(\mathrm{arr}([\![V]\!]^S) \ggg \mathsf{op}) \ggg [\![\mathscr{F}'[\lfloor y\rfloor]]\!]^S(\star)) \ggg [\![P]\!]^S(\star)\\
&= \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}) \ggg [\![\mathbf{let}\ x \Leftarrow \lfloor y\rfloor\ \mathbf{in}\ P]\!]^S(\star)\\
&= \mathrm{first}_S(\mathrm{arr}([\![V]\!]^S(\star)) \ggg \mathsf{op}) \ggg [\![\mathscr{F}[\lfloor y\rfloor]]\!]^S(\star).
\end{aligned}$$

by the induction hypothesis. $\blacksquare$

Theorem 5.12 (soundness). *The following hold.*

1. *If* $\diamond \vdash M : A$ *and* $M \to M'$, *then* $[\![M]\!]^S = [\![M']\!]^S$ *for any S.*
2. *If* $\diamond\ \mathring{,}\ \diamond \vdash P \mathbin{!} A$ *and* $P \to P'$, *then* $[\![P]\!]^S = [\![P']\!]^S$ *for any S.*

Proof By induction on the derivations $M \to M'$ and $P \to P'$.

(1). **In case** O-FST. We have $\diamond \vdash \mathbf{fst}\langle V_1, V_2 \rangle : A_1$ and $\mathbf{fst}\langle V_1, V_2 \rangle \to V_1$. The derivation of $\diamond \vdash \mathbf{fst}\langle V_1, V_2 \rangle : A_1$ is

$$\dfrac{\dfrac{\diamond \vdash V_1 : A_1 \quad \diamond \vdash V_2 : A_2}{\diamond \vdash \langle V_1, V_2 \rangle : A_1 \times A_2} \text{ T-PAIR}}{\diamond \vdash \mathbf{fst}\langle V_1, V_2 \rangle : A_1} \text{ T-FST} \quad .$$

We have

$$[\![\mathbf{fst}\langle V_1, V_2 \rangle]\!]^S(\star) = \pi_1([\![V_1]\!]^S(\star), [\![V_2]\!]^S(\star)) = [\![V_1]\!]^S(\star)$$

for any S and $\star \in [\![\diamond]\!]^S = 1 = \{\star\}$.

In case O-SND. Similar to the case O-FST.

In case O-β. We have $\diamond \vdash (\lambda x : A.M)V : B$ and $(\lambda x : A.M)V \to M[V/x]$. The derivation of $\diamond \vdash (\lambda x : A.M)V : B$ is

$$\dfrac{\dfrac{x : A \vdash M : B}{\diamond \vdash \lambda x : A.M : A \to B} \text{ T-ABS} \quad \diamond \vdash V : A}{\diamond \vdash (\lambda x : A.M)V : B} \text{ T-APP} \quad .$$

We have

$$\begin{aligned}
[\![(\lambda x : A.M)V]\!]^S(\star) &= [\![\lambda x : A.M]\!]^S(\star)\left([\![V]\!]^S(\star)\right) \\
&= [\![M]\!]^S(\star, [\![V]\!]^S(\star)) \\
&= [\![M[V/x]]\!]^S(\star) \qquad\qquad \textcolor{blue}{\textit{Lemma} 5.10}
\end{aligned}$$

for any S and $\star \in [\![\diamond]\!]^S = 1 = \{\star\}$.

(2). **In case** O-$\beta^\bullet$. We have $\diamond \,\fatsemi\, \diamond \vdash (\lambda^\bullet x : A.P) \bullet V \mathbin{!} B$ and $(\lambda^\bullet x : A.P) \bullet V \to P[V/x]$. The derivation of $\diamond \,\fatsemi\, \diamond \vdash (\lambda^\bullet x : A.P) \bullet V \mathbin{!} B$ is

$$\dfrac{\dfrac{\diamond \,\fatsemi\, x : A \vdash P \mathbin{!} B}{\diamond \vdash \lambda^\bullet x : A.P : A \rightsquigarrow B} \text{ T-CABS} \quad \diamond \vdash V : A}{\diamond \,\fatsemi\, \diamond \vdash (\lambda^\bullet x : A.P) \bullet V \mathbin{!} B} \text{ T-CAPP}$$

We have

$$\begin{aligned}
[\![(\lambda^\bullet x : A.P) \bullet V]\!]^S(\star) &= \mathrm{first}_S\left(\mathrm{arr}\left([\![V]\!]^S\right)\right) \ggg [\![\lambda^\bullet x : A.P]\!]^S(\star) \\
&= \mathrm{first}_S\left(\mathrm{arr}\left([\![V]\!]^S\right)\right) \ggg [\![P]\!]^S(\star) \\
&= [\![P[V/x]]\!]^S(\star) \qquad\qquad \textcolor{blue}{\textit{Lemma} 5.10}
\end{aligned}$$

for any S and $\star \in [\![\Gamma]\!]^S = 1 = \{\star\}$.

In case O-LET. We have $\diamond \,\fatsemi\, \diamond \vdash \mathbf{let}\, x \Leftarrow \lfloor V \rfloor \mathbf{in}\, Q \mathbin{!} B$ and $\mathbf{let}\, x \Leftarrow \lfloor V \rfloor \mathbf{in}\, Q \to Q[V/x]$. The derivation of $\diamond \,\fatsemi\, \diamond \vdash \mathbf{let}\, x \Leftarrow \lfloor V \rfloor \mathbf{in}\, Q \mathbin{!} B$ is

$$\dfrac{\dfrac{\diamond \vdash V : A}{\diamond \,\fatsemi\, \diamond \vdash \lfloor V \rfloor \mathbin{!} A} \text{ T-PURE} \quad \diamond \,\fatsemi\, x : A \vdash Q \mathbin{!} B}{\diamond \,\fatsemi\, \diamond \vdash \mathbf{let}\, x \Leftarrow \lfloor V \rfloor \mathbf{in}\, Q \mathbin{!} B} \text{ T-LET} \quad .$$

We have

$$
\begin{aligned}
&\llbracket \textbf{let } x \Leftarrow \lfloor V \rfloor \textbf{ in } Q \rrbracket^S(\star) \\
&= \mathrm{arr}\,(d) \ggg \mathrm{first}\left(\llbracket \lfloor V \rfloor \rrbracket^S(\star)\right) \ggg \llbracket Q \rrbracket^S(\star) \\
&= \mathrm{arr}\,(d) \ggg \mathrm{first}\left(\mathrm{arr}\left(\llbracket V \rrbracket^S \circ \pi_1\right)\right) \ggg \llbracket Q \rrbracket^S(\star) \\
&= \mathrm{first}_S(\mathrm{arr}(d)) \ggg \mathrm{first}_{\llbracket \Delta \rrbracket^S \times S}\left(\mathrm{arr}\,\llbracket V \rrbracket^S\right) \ggg \llbracket Q \rrbracket^S(\star) \\
&= \llbracket Q[V/x] \rrbracket^S(\star) \qquad\qquad\qquad\qquad\qquad \textit{Lemma } 5.10
\end{aligned}
$$

for any S and $\star \in \llbracket \diamond \rrbracket^S = 1 = \{\star\}$.

In case O-HVAL. We have $\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle} \lfloor V \rfloor \textbf{ with } H \,!\, D$ and $\textbf{handle} \lfloor V \rfloor \textbf{ with } H \to P[V/x]$ where $H = \{\,\mathring{,}\, x : C \mapsto P\} \cup \{\mathsf{op}, k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \mapsto Q_{\mathsf{op}}\}_{\mathsf{op}:\gamma \to \delta \in \Sigma}$. The derivation of $\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle} \lfloor V \rfloor \textbf{ with } H \,!\, D$ is

$$
\dfrac{\dfrac{\diamond \vdash V : C}{\diamond \,\mathring{,}\, \diamond \vdash \lfloor V \rfloor \,!\, C}\ \text{T-Pure} \qquad \dfrac{\diamond \,\mathring{,}\, x : C \vdash P \,!\, D \quad k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \vdash Q_{\mathsf{op}} \,!\, D}{\vdash H : C \Rightarrow D}\ \text{T-Handler}}{\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle} \lfloor V \rfloor \textbf{ with } H \,!\, D}\ \text{T-Handle}
$$

where C and D are primitive. We have

$$
\begin{aligned}
\llbracket \textbf{handle} \lfloor V \rfloor \textbf{ with } H \rrbracket^S(\star) &= h_{\llbracket \Delta \rrbracket^S}(\llbracket \lfloor V \rfloor \rrbracket^S(\star)) \\
&= h_{\llbracket \Delta \rrbracket^S}(\mathrm{arr}(\llbracket V \rrbracket^S \circ \pi_1)) \\
&= \mathrm{arr}(\llbracket V \rrbracket^S) \ggg \llbracket P \rrbracket \\
&= \llbracket P[V/x] \rrbracket^S(\star) \qquad\qquad \textit{Lemma } 5.10
\end{aligned}
$$

for any S and $\star \in \llbracket \diamond \rrbracket^S = 1 = \{\star\}$.

In case O-HOP. We have $\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle } \mathscr{F}[\mathsf{op}(V)] \textbf{ with } H \,!\, D$ and $\textbf{handle } \mathscr{F}[op(V)] \textbf{ with } H \to Q_{\mathsf{op}}[V/z, (\lambda^\bullet y : \delta. \,\textbf{handle } \mathscr{F}[\lfloor y \rfloor] \textbf{ with } H)/k]$ where $H = \{\,\mathring{,}\, x : C \mapsto P\} \cup \{\mathsf{op}, k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \mapsto Q_{\mathsf{op}}\}_{\mathsf{op}:\gamma \to \delta \in \Sigma}$. The derivation of $\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle } \mathscr{F}[\mathsf{op}(V)] \textbf{ with } H \,!\, D$ is

$$
\dfrac{\diamond \,\mathring{,}\, \diamond \vdash \mathscr{F}[\mathsf{op}(V)] \,!\, C \qquad \dfrac{\diamond \,\mathring{,}\, x : C \vdash P \,!\, D \quad k : \delta \rightsquigarrow D \,\mathring{,}\, z : \gamma \vdash Q_{\mathsf{op}} \,!\, D}{\diamond \vdash H : C \Rightarrow D}\ \text{T-Handler}}{\diamond \,\mathring{,}\, \diamond \vdash \textbf{handle } \mathscr{F}[\mathsf{op}(V)] \textbf{ with } H \,!\, D}\ \text{T-Handle}
$$

where C and D are primitive. We have

$$\llbracket \mathbf{handle}\ \mathscr{F}[\mathsf{op}(V)]\ \mathbf{with}\ H \rrbracket^S(\star)$$

$$= h_{\llbracket \diamond \rrbracket^S}(\llbracket \mathscr{F}[\mathsf{op}(V)] \rrbracket^S(\star))$$

$$= h_1\left(\mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S) \ggg \mathsf{op}) \ggg \llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star)\right) \qquad \textit{Lemma}\ 5.11$$

$$= \alpha\left(\mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S) \ggg \mathsf{op}) \ggg \llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star) \ggg \llbracket P \rrbracket\right)$$

$$= \alpha\left(\mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S) \ggg \mathsf{op}), \alpha\left(\llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star) \ggg \llbracket P \rrbracket\right)\right)$$

$$= \alpha\left(\mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S) \ggg \mathsf{op}), h\left(\llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star)\right)\right)$$

$$= \alpha\left(\mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S) \ggg \mathsf{op}), h\left(\llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star)\right)\right)$$

$$= \mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S)) \ggg \llbracket Q_{\mathsf{op}} \rrbracket^S\left(h\left(\llbracket \mathscr{F}[\lfloor y \rfloor] \rrbracket^S(\star)\right)\right)$$

$$= \mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S)) \ggg \llbracket Q_{\mathsf{op}} \rrbracket^S\left(\llbracket \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H \rrbracket^S(\star)\right)$$

$$= \mathrm{first}_S(\mathrm{arr}(\llbracket V \rrbracket^S)) \ggg \llbracket Q_{\mathsf{op}} \rrbracket^S\left(\llbracket \lambda^{\bullet} y : \delta.\ \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H \rrbracket^S(\star)\right)$$

$$= \llbracket Q_{\mathsf{op}}[V/z, (\lambda^{\bullet} y : \delta.\ \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H)/k] \rrbracket^S(\star). \qquad \textit{Lemma}\ 5.10$$

∎

Lemma 5.14. *The following hold.*

1. *If $M \to^* M'$ and $v \lhd_A M'$, then $v \lhd_A M$.*
2. *If $M \to^* M'$ and $v \lhd_A M$, then $v \lhd_A M'$.*
3. *If $P \to^* P'$ and $a \blacktriangleleft_A P'$, then $a \blacktriangleleft_A P$.*
4. *If $P \to^* P'$ and $a \blacktriangleleft_A P$, then $a \blacktriangleleft_A P'$.*

Proof We can prove (3) and (4) straightforwardly by the definition. To prove (1) and (2), we do induction on the type A.

(1). **In case** Unit. Suppose $M \to^* M'$ and $v \lhd_{\mathrm{Unit}} M'$. We have $v = \star$ and $M' \to^* \langle\rangle$ by the definition of $\lhd_{\mathrm{Unit}}$. Hence, we have $M \to^* M' \to^* \langle\rangle$, which means $v \lhd_{\mathrm{Unit}} M$.

In case $A_1 \times A_2$. Suppose $M \to^* M'$ and $v \lhd_{A_1 \times A_2} M'$. We have $M \to^* M' \to^* \langle V_1, V_2 \rangle$, $\pi_1(v) \lhd_{A_1} V_1$ and $\pi_2(v) \lhd_{A_2} V_2$. This means $v \lhd_{A_1 \times A_2} M$.

In case $A \to B$. Suppose $M \to^* M'$ and $f \lhd_{A \to B} M'$. We have $M \to^* M' \to^* \lambda x : A.M''$ and $(w \lhd_A N \implies fw \lhd_B M'N)$ for any N and w. Given N and w with $w \lhd_A N$, we have $fw \lhd_B M'N$. Since $M \to^* M'$, we have $MN \to^* M'N$. By the induction hypothesis (1), we have $fw \lhd_B MN$. This means $f \lhd_{A \to B} M$.

In case $A \rightsquigarrow B$. Suppose $M \to^* M'$ and $a \lhd_{A \rightsquigarrow B} M'$. We have $M \to^* M' \to^* \lambda^{\bullet} x : A.P$ and $(w \lhd_A N \implies \mathrm{arr}(w) \ggg a \blacktriangleleft_B M' \bullet N)$ for any N and w. Given N and w with $w \lhd_A N$, we have $\mathrm{arr}(w) \ggg a \blacktriangleleft_B M' \bullet N$. Since $M \to^* M'$, we have $M \bullet N \to^* M' \bullet N$. By (3), we have $\mathrm{arr}(w) \ggg a \blacktriangleleft_B M \bullet N$. This means $a \lhd_{A \rightsquigarrow B} M$.

(2). Note that we can show that $M' \to^* V$ if $M \to^* V$ and $M \to^* M'$.

In case Unit. Suppose $M \to^* M'$ and $v \lhd_{\mathrm{Unit}} M$. We have $v = \star$ and $M \to^* \langle\rangle$ by the definition of $\lhd_{\mathrm{Unit}}$. Hence, we have $M' \to^* \langle\rangle$, which means $v \lhd_{\mathrm{Unit}} M'$.

In case $A_1 \times A_2$. Suppose $M \to^* M'$ and $v \vartriangleleft_{A_1 \times A_2} M$. We have $M \to^* \langle V_1, V_2 \rangle$, $\pi_1(v) \vartriangleleft_{A_1} V_1$ and $\pi_2(v) \vartriangleleft_{A_2} V_2$. This means $v \vartriangleleft_{A_1 \times A_2} M$ since we have $M' \to^* \langle V_1, V_2 \rangle$.

In case $A \to B$. Suppose $M \to^* M'$ and $f \vartriangleleft_{A \to B} M$. We have $M \to^* \lambda x : A.M''$ and $(w \vartriangleleft_A N \implies fw \vartriangleleft_B MN)$ for any N and w. Given N and w with $w \vartriangleleft_A N$, we have $fw \vartriangleleft_B MN$. Since $M \to^* M'$, we have $MN \to^* M'N$. By the induction hypothesis (2), we have $fw \vartriangleleft_B M'N$. This means $f \vartriangleleft_{A \to B} M'$.

In case $A \rightsquigarrow B$. Suppose $M \to^* M'$ and $a \vartriangleleft_{A \rightsquigarrow B} M$. We have $M \to^* \lambda^\bullet x : A.P$ and $(w \vartriangleleft_A N \implies \mathrm{arr}(w) \ggg a \blacktriangleleft_B M \bullet N)$ for any N and w. Given N and w with $w \vartriangleleft_A N$, we have $\mathrm{arr}(w) \ggg a \blacktriangleleft_B M \bullet N$. Since $M \to^* M'$, we have $M \bullet N \to^* M' \bullet N$. By (4), we have $\mathrm{arr}(w) \ggg a \blacktriangleleft_B M' \bullet N$. This means $a \vartriangleleft_{A \rightsquigarrow B} M'$. ∎

Theorem 5.15. *Let* $\Gamma = x_1 : A_1, \ldots, x_m : A_m$ *and* $\Delta = y_1 : B_1, \ldots, y_n : B_n$. *The following hold.*

1. *For* $\Gamma \vdash M : A$ *and* $v_i \in [\![A_i]\!]$ *and* V_i *with* $v_i \vartriangleleft_{A_i} V_i$ *for each* $i \in \{1, \ldots, m\}$,

$$[\![M]\!](v_1, \ldots, v_m) \vartriangleleft_A M[V_1/x_1, \ldots, V_m/x_m].$$

2. *For* $\Gamma \mathbin{\mathring{,}} \Delta \vdash P : C$, $v_i \in [\![A_i]\!]$ *and* V_i *with* $v_i \vartriangleleft_{A_i} V_i$ *for each* $i \in \{1, \ldots, m\}$ *and* $w_j \in [\![B_j]\!]$ *and* W_j *with* $w_j \vartriangleleft_{B_j} W_j$ *for each* $j \in \{1, \ldots, n\}$,

$$\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![P]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_C P[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

Proof The proof is done by induction on the derivation of $\Gamma \vdash M : A$ and $\Gamma \mathbin{\mathring{,}} \Delta \vdash P \,!\, C$. Suppose that $\Gamma = x_1 : A_1, \ldots, x_m : A_m$ and $\Delta = y_1 : B_1, \ldots, y_n : B_n$, V_i is a value and $v_i \in [\![A_i]\!]$ with $v_i \vartriangleleft_{A_i} V_i$ for each $i \in \{1, \ldots, m\}$, and W_j is a value and $w_j \in [\![B_j]\!]$ with $w_j \vartriangleleft_{B_j} W_j$ for each $j \in \{1, \ldots, n\}$.

(1) The case T-CABS is proved in the main text, and the other cases are proved straightforwardly by the definition of $\vartriangleleft_A$.

(2) **In case** T-PURE. The derivation is

$$\frac{\Gamma, \Delta \vdash M : A}{\Gamma \mathbin{\mathring{,}} \Delta \vdash \lfloor M \rfloor \,!\, A} \; \text{T-PURE}.$$

By the induction hypothesis, we have

$$[\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n) \vartriangleleft_A M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

By the definition of $\vartriangleleft_A$, there exists a value $\vdash V : A$ such that

$$M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \to^* V.$$

Thus, we obtain $\lfloor M \rfloor[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \to^* \lfloor V \rfloor$. We have

$$\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\lfloor M \rfloor]\!](v_1, \ldots, v_m)$$
$$= \mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg \mathrm{arr}([\![M]\!](v_1, \ldots, v_m, -))$$
$$= \mathrm{arr}([\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n)).$$

Therefore, by the definition of $\blacktriangleleft_A$, we have

$$\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\lfloor M \rfloor]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_A \lfloor M \rfloor [V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

In case T-CAPP. The derivation is

$$\frac{\Gamma \vdash L : A \rightsquigarrow B \quad \Gamma, \Delta \vdash M : A}{\Gamma \, \mathbin{\fatsemi} \Delta \vdash L \bullet M \,!\, B} \ \ \text{T-CABS}.$$

By the induction hypothesis, we have

$$[\![L]\!](v_1, \ldots, v_m) \lhd_{A \rightsquigarrow B} L[V_1/x_1, \ldots, V_m/x_m],$$
$$[\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n) \lhd_A M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

By the definition of $\lhd_{A \rightsquigarrow B}$, we have

$$\mathsf{arr}([\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n)) \ggg [\![L]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_B L[V_1/x_1, \ldots, V_m/x_m] \bullet M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

We also have

$$\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![M \bullet L]\!](v_1, \ldots, v_m)$$
$$= \mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg \mathsf{arr}([\![M]\!](v_1, \ldots, v_m, -)) \ggg [\![L]\!](v_1, \ldots, v_m)$$
$$= \mathsf{arr}([\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n)) \ggg [\![L]\!](v_1, \ldots, v_m)$$

and

$$(L \bullet M)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]$$
$$= L[V_1/x_1, \ldots, V_m/x_m] \bullet M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

Therefore, we have

$$\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![M \bullet L]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_B (L \bullet M)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

In case T-OP. The derivation is

$$\frac{\Gamma, \Delta \vdash M \,!\, \gamma \quad (\mathsf{op} : \gamma \twoheadrightarrow \delta) \in \Sigma}{\Gamma \, \mathbin{\fatsemi} \Delta \vdash \mathsf{op}(M) \,!\, \delta}.$$

By the induction hypothesis, we have

$$[\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n) \lhd_\gamma M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]. \qquad (4.1)$$

By the definition of $\lhd_\delta$, there exists a value U such that

$$M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \to^* U. \qquad (4.2)$$

From (4.1), (4.2), and Lemma 5.14(1), we have

$$[\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n) \lhd_{[\![\gamma]\!]} U. \qquad (4.3)$$

We have

$$
\begin{aligned}
&\mathsf{op}(M)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \\
&= \mathsf{op}(M[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]) \\
&\to^* \mathsf{op}(U) \\
&= \mathscr{F}[\mathsf{op}(U)]
\end{aligned}
$$

for the trivial context $\mathscr{F} = [-]$, and

$$
\begin{aligned}
&\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\mathsf{op}(M)]\!](v_1, \ldots, v_m) \\
&= \mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg \mathsf{arr}([\![M]\!](v_1, \ldots, v_m, -)) \ggg \mathsf{op} \\
&= \mathsf{arr}([\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n)) \ggg \mathsf{op} \\
&= \mathsf{arr}([\![M]\!](v_1, \ldots, v_m, w_1, \ldots, w_n)) \ggg \mathsf{op} \ggg [\![\mathscr{F}[\lfloor y \rfloor]]\!].
\end{aligned}
$$

If $w \lhd_\delta W$ then $\mathsf{arr}(w) \blacktriangleleft_\delta \lfloor W \rfloor = \mathscr{F}[\lfloor W \rfloor]$. Therefore, by the definition of $\blacktriangleleft_\delta$, we have

$$
\begin{aligned}
&\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\mathsf{op}(M)]\!](v_1, \ldots, v_m) \\
&\blacktriangleleft_\delta \mathsf{op}(M)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].
\end{aligned}
$$

In case T-LET. The derivation is

$$
\frac{\Gamma \,\mathbin{;}\, \Delta \vdash P \,!\, B \quad \Gamma \,\mathbin{;}\, y : B, \Delta \vdash Q \,!\, A}{\Gamma \,\mathbin{;}\, \Delta \vdash \mathbf{let}\, x \Leftarrow P \,\mathbf{in}\, Q \,!\, A} \quad .
$$

Let $\qquad P' = P[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \qquad$ and $\qquad Q' = Q[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]$. By the induction hypothesis, we have

$$
\mathsf{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![P]\!](v_1, \ldots, v_m) \blacktriangleleft_A P'
$$

and

$$
\mathsf{arr}(\langle u, w_1, \ldots, w_n \rangle) \ggg [\![Q]\!](v_1, \ldots, v_m) \blacktriangleleft_B Q'[U/y]
$$

for any $u \in [\![B]\!]$ and a value U with $u \lhd_B U$. By the definition of $\blacktriangleleft_A$, there are two cases.

Case $P' \to^* \lfloor U \rfloor$ for a value U and there exists $u \in [\![B]\!]$ such that $\mathsf{arr}(w_1, \ldots, w_n) \ggg [\![P]\!](v_1, \ldots, v_m) = \mathsf{arr}(u)$ and $u \lhd_B U$. We have

$$
\begin{aligned}
&(\mathbf{let}\, y \Leftarrow P \,\mathbf{in}\, Q)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] \\
&= \mathbf{let}\, y \Leftarrow P' \,\mathbf{in}\, Q' \\
&\to^* \mathbf{let}\, y \Leftarrow \lfloor U \rfloor \,\mathbf{in}\, Q' \\
&\to Q'[U/y]
\end{aligned}
$$

and

$$
\begin{aligned}
&\mathsf{arr}(\vec{w}) \ggg [\![\mathbf{let}\, y \Leftarrow P \,\mathbf{in}\, Q]\!](v_1, \ldots, v_m) \\
&= \mathsf{arr}(\vec{w}) \ggg \mathsf{arr}(d) \ggg \mathsf{first}([\![P]\!](v_1, \ldots, v_n)) \ggg [\![Q]\!](v_1, \ldots, v_n) \\
&= \mathsf{arr}(d) \ggg \mathsf{first}(\mathsf{arr}(\vec{w}) \ggg [\![P]\!](v_1, \ldots, v_n)) \ggg \mathsf{second}(\mathsf{arr}(\vec{w})) \ggg [\![Q]\!](v_1, \ldots, v_n) \\
&= \mathsf{arr}(d) \ggg \mathsf{first}(\mathsf{arr}(u)) \ggg \mathsf{second}(\mathsf{arr}(\vec{w})) \ggg [\![Q]\!](v_1, \ldots, v_n) \\
&= \mathsf{arr}(\langle u, w_1, \ldots, w_m \rangle) \ggg [\![Q]\!](v_1, \ldots, v_n)
\end{aligned}
$$

where $\vec{w} = \langle w_1, \ldots, w_n \rangle$. Hence, by Lemma 5.14(3), we obtain

$$\operatorname{arr}(\vec{w}) \ggg [\![\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_B (\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

Case $P' \to^* \mathscr{F}[\mathsf{op}(U)]$ for a value U and $\mathsf{op} : \gamma \to \delta$ and there exists $u \in [\![\gamma]\!]$ with $u \lhd_\gamma U$ and $b \in \mathscr{A}_\Sigma([\![\delta]\!], [\![A]\!])$ such that $\operatorname{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![P]\!](v_1, \ldots, v_m) = \operatorname{arr}(u) \ggg \mathsf{op} \ggg b$ and $w \lhd_\delta W \implies \operatorname{arr}(w) \ggg b \blacktriangleleft_A \mathscr{F}[\lfloor W \rfloor]$ for any $w \in [\![\delta]\!]$ and a value W. We have

$$(\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]$$
$$= \mathbf{let} \, y \Leftarrow P' \, \mathbf{in} \, Q'$$
$$\to^* \mathbf{let} \, y \Leftarrow \mathscr{F}[\mathsf{op}(U)] \, \mathbf{in} \, Q'$$
$$= \mathscr{F}'[\mathsf{op}(U)]$$

where $\mathscr{F}' = \mathbf{let} \, y \Leftarrow \mathscr{F} \, \mathbf{in} \, Q'$, and

$$\operatorname{arr}(\vec{w}) \ggg [\![\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q]\!](\vec{v})$$
$$= \operatorname{arr}(\vec{w}) \ggg \operatorname{arr}(d) \ggg \mathrm{first}([\![P]\!](\vec{v})) \ggg [\![Q]\!](\vec{v})$$
$$= \operatorname{arr}(d) \ggg \mathrm{first}(\operatorname{arr}(\vec{w}) \ggg [\![P]\!](\vec{v})) \ggg \mathrm{second}(\operatorname{arr}(\vec{w})) \ggg [\![Q]\!](\vec{v})$$
$$= \operatorname{arr}(d) \ggg \mathrm{first}(\operatorname{arr}(u) \ggg \mathsf{op} \ggg b) \ggg \mathrm{second}(\operatorname{arr}(\vec{w})) \ggg [\![Q]\!](\vec{v})$$
$$= \operatorname{arr}(u) \ggg \mathsf{op} \ggg b \ggg \operatorname{arr}(\lambda b.\langle b, \vec{w}\rangle) \ggg [\![Q]\!](\vec{v})$$
$$= \operatorname{arr}(u) \ggg \mathsf{op} \ggg \operatorname{arr}(\lambda x.\langle x, \vec{w}\rangle) \ggg \mathrm{first}(b) \ggg [\![Q]\!](\vec{v})$$
$$= \operatorname{arr}(u) \ggg \mathsf{op} \ggg b'$$

where $\vec{w} = \langle w_1, \ldots, w_n \rangle$, $\vec{v} = \langle v_1, \ldots, v_m \rangle$ and $b' = \operatorname{arr}(\lambda x.\langle x, \vec{w}\rangle) \ggg \mathrm{first}(b) \ggg [\![Q]\!](\vec{v})$. Given $w \in [\![\delta]\!]$ and $\diamond \vdash W : \delta$ with $w \lhd_\delta W$, we can show $\operatorname{arr}(w) \ggg b' \blacktriangleleft_B \mathscr{F}'[\lfloor W \rfloor]$. Therefore, we have

$$\operatorname{arr}(\vec{w}) \ggg [\![\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q]\!](v_1, \ldots, v_m)$$
$$\blacktriangleleft_B (\mathbf{let} \, y \Leftarrow P \, \mathbf{in} \, Q)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].$$

In case T-HANDLE. The derivation is

$$\frac{\Gamma \,\mathbf{;}\, \Delta \vdash R \,!\, C \qquad \dfrac{\diamond \,\mathbf{;}\, x : C \vdash P \,!\, D \quad (k : \delta \rightsquigarrow D \,\mathbf{;}\, z : \gamma \vdash Q_{\mathsf{op}} \,!\, D)_{\mathsf{op} \in \Sigma}}{\vdash H : C \Rightarrow D}}{\Gamma \,\mathbf{;}\, \Delta \vdash \mathbf{handle} \, R \, \mathbf{with} \, H \,!\, D}$$

where $H = \{\,\mathbf{;}\, x : C \mapsto P\} \cup \{k : \delta \rightsquigarrow D \,\mathbf{;}\, z : \gamma \mapsto Q_{\mathsf{op}}\}_{\mathsf{op}}$. Let $R' = R[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]$. By the induction hypothesis, we have

$$\operatorname{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![R]\!](v_1, \ldots, v_m) \blacktriangleleft_C R'.$$

By the definition of $\blacktriangleleft_C$, there are two cases.

Case $R' \to^* \lfloor U \rfloor$ for a value U and there exists $u \in [\![C]\!]$ such that $\operatorname{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![R]\!](v_1, \ldots, v_m) = \operatorname{arr}(u)$ and $u \lhd_C U$. By the induction hypothesis and $u \lhd_C U$, we have

$$\operatorname{arr}(u) \ggg [\![P]\!] \blacktriangleleft_D P[U/x].$$

We have

$$\begin{aligned}
&\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\mathbf{handle}\ R\ \mathbf{with}\ H]\!](v_1, \ldots, v_m) \\
&= \mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![H]\!]([\![R]\!](v_1, \ldots, v_m)) \\
&= [\![H]\!](\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![R]\!](v_1, \ldots, v_m)) \\
&= [\![H]\!](\mathrm{arr}(u)) \\
&= \mathrm{arr}(u) \ggg [\![P]\!]
\end{aligned}$$

and

$$\begin{aligned}
(\mathbf{handle}\ R\ \mathbf{with}\ H)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] &= \mathbf{handle}\ R'\ \mathbf{with}\ H \\
&\to^* \mathbf{handle} \lfloor U \rfloor\ \mathbf{with}\ H \\
&\to P[U/x].
\end{aligned}$$

Therefore, by Lemma 5.14(3), we obtain

$$\begin{aligned}
&\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\mathbf{handle}\ R\ \mathbf{with}\ H]\!](v_1, \ldots, v_m) \\
&\blacktriangleleft_D (\mathbf{handle}\ R\ \mathbf{with}\ H)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n]
\end{aligned}$$

Case $R' \to^* \mathscr{F}[\mathrm{op}(U)]$ for a value U and $\mathrm{op} : \gamma \to \delta \in \Sigma$ and there exist $u \in [\![\gamma]\!]$ with $u \lhd_\gamma U$ and $b \in \mathscr{A}_\Sigma([\![\delta]\!], [\![C]\!])$ such that $\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![R]\!](v_1, \ldots, v_m) = \mathrm{arr}(u) \ggg \mathrm{op} \ggg b$ and $w \lhd_\delta W \implies \mathrm{arr}(w) \ggg b \blacktriangleleft_C \mathscr{F}[\lfloor W \rfloor]$ for any $w \in [\![\delta]\!]$ and $\diamond \vdash W : \delta$. Let $Q'_{\mathrm{op}} = Q_{\mathrm{op}}[U/z, (\lambda^\bullet y.\ \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H)/k]$. We have

$$\begin{aligned}
(\mathbf{handle}\ R\ \mathbf{with}\ H)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n] &= \mathbf{handle}\ R'\ \mathbf{with}\ H \\
&\to^* \mathbf{handle}\ \mathscr{F}[\mathrm{op}(U)]\ \mathbf{with}\ H \\
&\to Q'_{\mathrm{op}}
\end{aligned}$$

By the induction hypothesis and $u \lhd_\gamma U$, we have $\mathrm{arr}(u) \ggg [\![Q_{\mathrm{op}}]\!](\kappa) \blacktriangleleft_D Q'_{\mathrm{op}}$ for any $\kappa \in [\![\delta \rightsquigarrow D]\!]$ with $\kappa \lhd_{\delta \rightsquigarrow D} (\lambda^\bullet y.\ \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H)$. We can show $[\![H]\!](b) \lhd_{\delta \rightsquigarrow D} \lambda^\bullet y.\ \mathbf{handle}\ \mathscr{F}[\lfloor y \rfloor]\ \mathbf{with}\ H$ from $\forall w.\ \forall W.\ w \lhd_\delta W \implies \mathrm{arr}(w) \ggg b \blacktriangleleft_C \mathscr{F}[\lfloor W \rfloor]$. Thus, we have

$$\mathrm{arr}(u) \ggg [\![Q_{\mathrm{op}}]\!]([\![H]\!](b)) \blacktriangleleft_D Q'_{\mathrm{op}}$$

By Lemma 5.14(3), we obtain

$$\begin{aligned}
&\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![\mathbf{handle}\ R\ \mathbf{with}\ H]\!](v_1, \ldots, v_m) \\
&= \mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![H]\!]([\![R]\!](v_1, \ldots, v_m)) \\
&= [\![H]\!](\mathrm{arr}(\langle w_1, \ldots, w_n \rangle) \ggg [\![R]\!](v_1, \ldots, v_m)) \\
&= [\![H]\!](\mathrm{arr}(u) \ggg \mathrm{op} \ggg b) \\
&= \mathrm{arr}(u) \ggg [\![Q_{\mathrm{op}}]\!]([\![H]\!](b)) \\
&\blacktriangleleft_D (\mathbf{handle}\ R\ \mathbf{with}\ H)[V_1/x_1, \ldots, V_m/x_m, W_1/y_1, \ldots, W_n/y_n].
\end{aligned}$$

$\blacksquare$

5 Proofs for Section 6 (Related work)

Proposition 6.1. *There is a map*

$$\Xi \colon \int_A \int_B \mathbf{Set}\left((\mathrm{I}_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A})(A, B), \mathscr{A}(A, B)\right)$$
$$\to \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \mathbf{Set}(\mathscr{A}(\llbracket\delta\rrbracket, B), \mathscr{A}(\llbracket\gamma\rrbracket, B))$$

for any set B.

Proof By the following calculation:

$$\int_A \int_B \mathbf{Set}\left((\mathrm{I}_{\mathbb{C}} + \vec{\Sigma} \circ \mathscr{A})(A, B), \mathscr{A}(A, B)\right)$$

$$\cong \int_A \int_B \mathbf{Set}\left(\mathbb{C}(A, B) + \int^C \vec{\Sigma}(A, C) \times \mathscr{A}(C, B), \mathscr{A}(A, B)\right)$$

$$\cong \int_A \int_B \mathbf{Set}\left(\mathbb{C}(A, B), \mathscr{A}(A, B)\right) \times \int_A \int_B \int_C \mathbf{Set}\left(\vec{\Sigma}(A, C) \times \mathscr{A}(C, B), \mathscr{A}(A, B)\right)$$

$$\cong \int_B \int_A \mathbf{Set}\left(\mathbb{C}(A, B), \mathscr{A}(A, B)\right) \times \int_{A,B,C} \mathbf{Set}\left(\vec{\Sigma}(A, C), \mathscr{A}(C, B) \Rightarrow \mathscr{A}(A, B)\right)$$

$$\cong \int_B \mathscr{A}(B, B) \times \int_{A,B,C} \mathbf{Set}\left(\vec{\Sigma}(A, C), \mathscr{A}(C, B) \Rightarrow \mathscr{A}(A, B)\right)$$

$$\cong \int_B \mathscr{A}(B, B) \times \int_{A,B,C} \prod_{\mathsf{op}\colon \gamma \to \delta} \mathbf{Set}\left(\mathbb{C}(A, \llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow C)), \mathscr{A}(C, B) \Rightarrow \mathscr{A}(A, B)\right)$$

$$\cong \int_B \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \int_{B,C} \int_A \mathbf{Set}\left(\mathbb{C}(A, \llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow C)), \mathscr{A}(C, B) \Rightarrow \mathscr{A}(A, B)\right)$$

$$\cong \int_B \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \int_{B,C} \mathscr{A}(C, B) \Rightarrow \mathscr{A}(\llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow C), B)$$

$$\cong \int_B \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \int_{B,C} \mathbf{Set}\left(\mathscr{A}(C, B), \mathscr{A}(\llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow C), B)\right)$$

$$\xrightarrow{\text{projection}} \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \mathbf{Set}(\mathscr{A}(\llbracket\delta\rrbracket, B), \mathscr{A}(\llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow \llbracket\delta\rrbracket), B))$$

$$\xrightarrow{\mathrm{id} \times \prod \mathbf{Set}(\mathscr{A}(\llbracket\delta\rrbracket, B), \phi)} \mathscr{A}(B, B) \times \prod_{\mathsf{op}\colon \gamma \to \delta} \mathbf{Set}(\mathscr{A}(\llbracket\delta\rrbracket, B), \mathscr{A}(\llbracket\gamma\rrbracket, B))$$

where $\phi \colon \mathscr{A}(\llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow \llbracket\delta\rrbracket), B) \to \mathscr{A}(\llbracket\gamma\rrbracket, B)$ is defined as

$$\phi(a) = \mu^{\mathscr{A}}_{\llbracket\gamma\rrbracket, \llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow \llbracket\delta\rrbracket), B}\left(\eta^{\mathscr{A}}_{\llbracket\gamma\rrbracket, \llbracket\gamma\rrbracket \times (\llbracket\delta\rrbracket \Rightarrow \llbracket\delta\rrbracket)}(\langle\mathrm{id}_{\llbracket\gamma\rrbracket}, \Lambda(\mathrm{id}_{\llbracket\delta\rrbracket})\rangle), a\right).$$

∎