RESUMPTION-BASED CATEGORICAL GEOMETRY OF INTERACTION FOR EFFECTS
計算効果のための、Resumption に基づいた圏論的 Geometry of Interaction

by

Koko Muroya
室屋晃子

A Senior Thesis
卒業論文

Submitted to
the Department of Information Science
the Faculty of Science, the University of Tokyo
on February 4, 2014
in Partial Fulfillment of the Requirements
for the Degree of Bachelor of Science

Thesis Supervisor: Ichiro Hasuo　蓮尾一郎
Lecturer of Information Science

ABSTRACT

Girard's Geometry of Interaction (GoI) gives a model of linear logic and is applied to give semantics of programming languages. In this paper the base steps of an approach to obtain resumption-based GoI interpretation of effects, namely resumption-based categorical GoI, are described. In our approach, categorical GoI — a categorical axiomatization of GoI introduced by Abramsky, Haghverdi and Scott — are used. The usage of resumptions for categorical GoI is also pointed out by them. Since resumptions are constructed by a kind of state machine called transducer, we give the concrete constructions of transducers that are used in categorical GoI. We also give the construction that represent algebraic operations. The ability of algebraic operations to represent effects is studied by Power and Plotkin. Linear combinatory algebras are also studied in this paper. It is described by Simpson that linear combinatory algebras can represent a kind of λ-calculus named linear λ-calculus. We defined a model of linear λ-calculus and proved the linear version of the Meyer-Scott theorem: it was found that linear combinatory algebras need extra combinators to form a model of linear λ-calculus.

論文要旨

Girard による Geometry of Interaction(GoI) は線形論理のモデルを与えるものであり、プログラミング言語の意味論を与えるために応用されている。本論文では、計算効果の GoI による解釈を得るための resumption に基づいたアプローチについて、その基本となるステップを説明する。このアプローチでは categorical GoI と呼ばれる、Abramsky, Haghverdi, Scott によって与えられた GoI の圏論的な定式化を用いた。彼らは圏論的 GoI における Resumption の利用についても指摘している。Resumption は transducer という一種のステートマシンによって構成されるので、我々は categorical GoI で利用される transducer の具体的な構成を与えた。また algebraic operation を表現するような構成も与えた。Algebraic operation の計算効果に対する表現能力は Plotkin,Power によって研究されている。また本論文では線形コンビネータ代数についても考察する。Simpson によって線形コンビネータ代数が線形ラムダ計算と呼ばれる一種のラムダ計算を表現することが示されたが、本論文では線形ラムダ計算のモデルを定義し、Meyer-Scott の定理の線形版が成り立つこと、つまり線形コンビネータ代数が線形ラムダ計算のモデルとなるためには追加のコンビネータが必要であることを示した。

Acknowledgements

I am deeply grateful to my supervisor Ichiro Hasuo for helpful advice. I thank Naohiko Hoshino for motivating this study. I appreciate useful discussions with the members of Hasuo Laboratory.

Contents

Chapter 1

Introduction

Girard introduced *Geometry of Interaction (GoI)* in [3] that gives semantics of proofs of linear logic. GoI is applied to give semantics of programming languages in various ways. Mackie gave semantics of PCF as token machines in [11]. In [2] followed by several papers, Ghica gave game-theoretic semantics of functional languages by translating programs into logical circuits. These results led to implementations such that compilers. This is one of the advantages of using GoI: semantics given by using GoI inherits the feature of GoI and enables us to observe dynamics of computations. Hasuo and Hoshino adopted another approach in [5]: they used a categorical axiomatization of GoI called *categorical GoI* together with the *realizability* technique.

Categorical GoI is given by Abramsky, Haghverdi and Scott [1]. From a category equipped with certain structures, called *GoI situation*, it extracts a *linear combinatory algebra* that is a combinatory algebra equipped with the ! modality. The realizability technique is introduced by Kleene in [9]. By the realizability technique, a category that gives a model of typed λ-calculus can be obtained from a linear combinatory algebra. This category gives game-theoretic interpretation of λ-calculus called *GoI interpretation*: the application of a term over another is interpreted as interactions of these terms.

The GoI interpretation of a term is represented by an arrow in the source category $\mathbb{C}$ of categorical GoI: in particular it is represented by a function if a category of sets and functions is used as $\mathbb{C}$. This leads to an idea that models of λ-calculus with effects — for example nondeterminism and probability — can be obtained by embedding the effect into arrows in $\mathbb{C}$. This idea indeed works: Hasuo and Hoshino [5] showed that some Kleisli categories can be the source of categorical GoI and extracted a model of quantum computation. Following Moggi [12], they modeled effects by monads and embedded effects into Kleisli arrows.

However, it is found that the construction is not so straightforward as expected. Consider the call-by-value evaluation of the following λ-term

$$(\lambda x.\, x + x)(1 \sqcup 2)$$

where the subterm $1 \sqcup 2$ returns 1 or 2 nondeterministically. Though we expect that this term returns 2 or 4, the GoI interpretation of this term can return 3 as the result of the following sequence of queries and answers.

1. The term $\lambda x.\, x + x$ asks the value of x that occurs in left.
2. The term $1 \sqcup 2$ returns a value 1.
3. The term $\lambda x.\, x + x$ asks the value of x that occurs in right.

4. The term $1 \sqcup 2$ returns a value 2.

5. The term $\lambda x.\, x + x$ adds 2 to 1 and returns a value 3.

In the call-by-value evaluation we expect that after the term $1 \sqcup 2$ chooses its value it follows its choice permanently. In fact this problem is not due to the evaluation strategy: consider the following term

$$(h \sqcup k)y \ .$$

The term $h \sqcup k$ is also needed to follow its choice, otherwise the answer of y to the request from h can be received by k. Indeed there is a case that the expected equation

$$(h \sqcup k)y = hy \sqcup ky$$

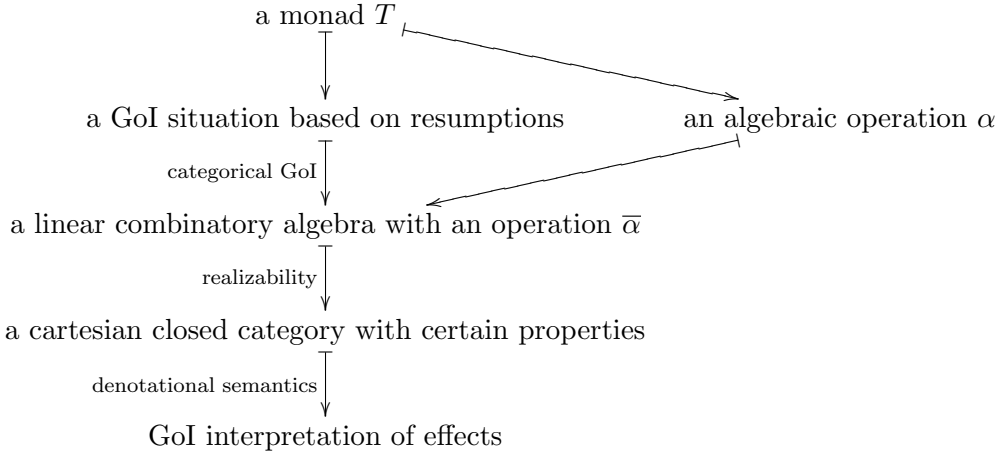
does not hold: the GoI interpretation of the term

$$((\lambda f.\, f\ 0) \sqcup (\lambda f.\, (f\ 1) + 1))(\lambda x.\, x)$$

can return an unexpected value 1 as well as expected values 0 and 2. The problem here is that the term $M \sqcup N$ cannot remember its choice. To interpret effects, it seems that GoI interpretations need to be equipped with "memories".

In [5], the lack of "memories" was dealt with by using the continuation monad on the category that is obtained by the realizability technique. There is another approach to deal with this problem: to embed "memories" as well as effects into arrows in the source category $\mathbb{C}$. One way to embed "memories" and effects into arrows is to use *resumptions*. Roughly speaking a resumption is a state machine: given an input, it decides outputs and also the next state according to its current internal state. This decision is affected by the embedded effect. Precisely a resumption is an equivalence class of transducers modulo an appropriate equivalence relation. "Memories" are embedded into resumptions as internal states and an effect is embedded into resumptions by a monad following Moggi [12].

Our motivation is to obtain a GoI interpretation of effects. The expected workflow based on resumptions is as below.



In Chapter 2 we study the first two step of this workflow: we use a monad T and an algebraic operation α to extract a linear combinatory algebra with operator $\overline{\alpha}$. We call this procedure *resumption-based categorical GoI*. In fact it is described in [1] that a category of resumptions can be the source category of categorical

GoI. We give a GoI situation based on resumptions by constructing concrete transducers.

The idea to use an algebraic operation follows Plotkin and Power [14, 13]. They showed in [14] that algebraic operations give adequate semantics for algebraic effects. We expect that this result will help us to obtain GoI interpretation of effects. In resumption-based categorical GoI we use an algebraic operation α and define an operator $\overline{\alpha}$ that constructs concrete transducer. We find that this operator enables a linear combinatory algebra to represent some interfaces to effects.

Additionally in Chapter 3 we study linear combinatory algebras. It is described by Simpson [16] that linear combinatory algebras can represent a kind of λ-calculus by its property called linear combinatory completeness. This λ-calculus is called *linear λ-calculus*. It inherits the ! modality of linear logic and explicitly tracks copying of data in computation. We define a model of untyped linear λ-calculus named *linear λ-model* and prove that linear combinatory algebras need extra power to form a linear λ-model. This result is the linear counterpart of the Meyer-Scott theorem.

The results in Chapter 2 forms part of the paper [7].

Chapter 2

Resumption-Based Categorical GoI

2.1 Categorical GoI

In this section we describe the source and outcome of categorical GoI. What play an important role in categorical GoI are the symmetric monoidal structure of a category and a *trace*. The description of the symmetric monoidal structure is in [10]. The following axiomatization of a trace is from [4].

Definition 2.1.1 (trace)**.** A *trace* on a symmetric monoidal category $(\mathbb{C}, I, \otimes)$ is a family of functions $\{\mathrm{tr}^X_{A,B} \colon \mathbb{C}(A \otimes X, B \otimes X) \to \mathbb{C}(A, B)\}_{X,A,B \in \mathbb{C}}$ that satisfies the following conditions:

- *tightening (naturality)*: $\mathrm{tr}^X_{A',B'}((k \otimes \mathrm{id}_X) \circ f \circ (h \otimes \mathrm{id}_X)) = k \circ \mathrm{tr}^X_{A,B}(f) \circ h$ for all $f \colon A \otimes X \to B \otimes X$, $h \colon A' \to A$ and $k \colon B \to B'$
- *sliding (dinaturality)*: $\mathrm{tr}^Y_{A,B}(f \circ (\mathrm{id}_A \otimes g)) = \mathrm{tr}^X_{A,B}((\mathrm{id}_B \otimes g) \circ f)$ for all $f \colon A \otimes X \to B \otimes Y$ and $g \colon Y \to X$
- *vanishing*: $\mathrm{tr}^I_{A,B}(f) = f$ and $\mathrm{tr}^{X \otimes Y}_{A,B}(g) = \mathrm{tr}^X_{A,B}(\mathrm{tr}^Y_{A \otimes X, B \otimes X}(g))$ for all $f \colon A \to B$ and $g \colon A \otimes X \otimes Y \to B \otimes X \otimes Y$
- *superposing*: $\mathrm{tr}^X_{C \otimes A, C \otimes B}(\mathrm{id}_C \otimes f) = \mathrm{id}_C \otimes \mathrm{tr}^X_{A,B}(f)$ for all $f \colon A \otimes X \to B \otimes X$
- *yanking*: $\mathrm{tr}^X_{X,X}(c_{X,X}) = \mathrm{id}_X$ where $c_{Y,Z} \colon Y \otimes Z \to Z \otimes Y$ is the symmetry in $\mathbb{C}$.

The trace $\mathrm{tr}^X_{A,B}(f)$ can be seen as the "iteration" of f: given an input in A, f is applied repeatedly until an output in B is obtained. A *traced symmetric monoidal category* is a symmetric monoidal category that has a trace. When a category $(\mathbb{C}, I, \otimes)$ is a traced symmetric monoidal category, a symmetric monoidal functor $F \colon \mathbb{C} \to \mathbb{C}$ is *traced* if it satisfies the following equation

$$\mathrm{tr}^{FX}_{FA,FB}(m^{-1}_{B,X} \circ Ff \circ m_{A,X}) = F(\mathrm{tr}^X_{A,B}(f))$$

for all $f \colon A \otimes X \to B \otimes X$ where $m_{A,B} \colon FA \otimes FB \to F(A \otimes B)$ is the coherence isomorphism of F.

We use string diagrams to represent arrows in a traced symmetric monoidal category $(\mathbb{C}, I, \otimes)$. An arrow $f \colon X \to Y$ in $\mathbb{C}$ is represented by a box with inputs and outputs. The box, inputs and outputs are labeled by f, X and Y respectively. The symmetric monoidal structure is expressed by parallel alignment and traces are expressed by "feedback". Figure 2.1 shows examples of string diagrams: the first one is of $f \colon X \otimes Z \to Y \otimes Z$, the second one is of $g \otimes h \colon X \otimes Z \to Y \otimes W$ and the third one is of $\mathrm{tr}^Z_{X,Y}(f) \colon X \to Y$.

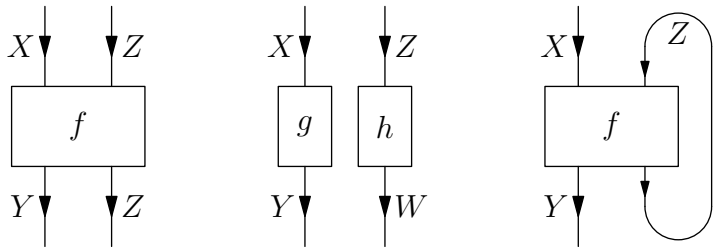


Figure 2.1: string diagrams in $\mathbb{C}$

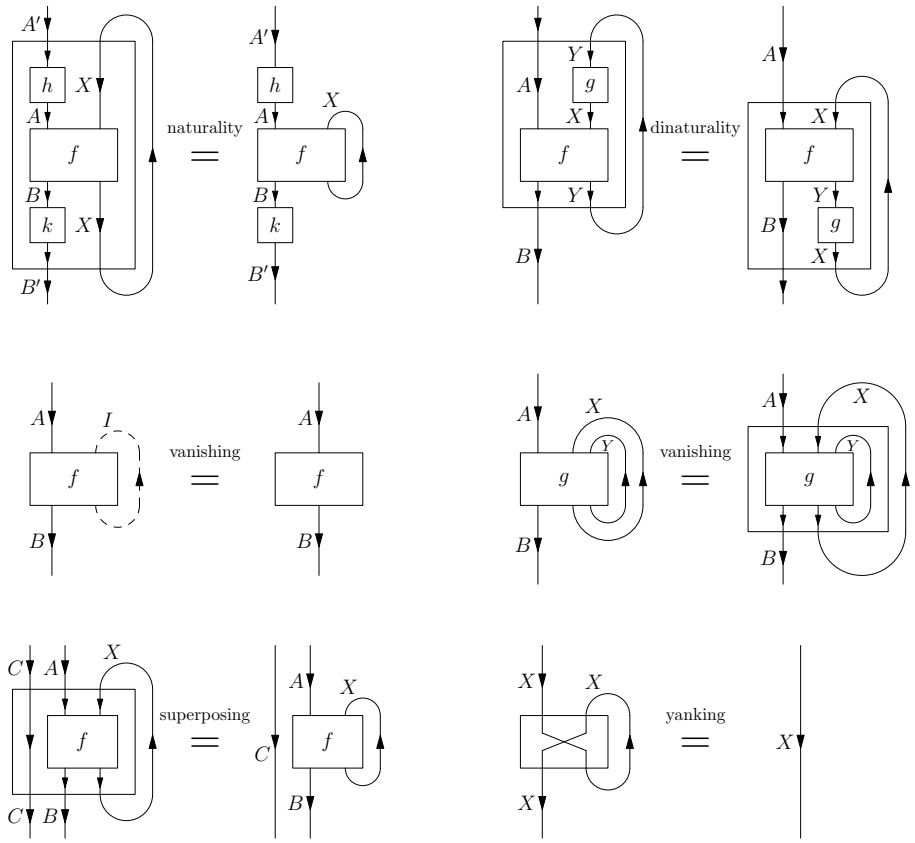


Figure 2.2: trace axioms in $\mathbb{C}$

The axioms of trace in Definition 2.1.1 are expressed by string diagrams in Figure 2.2.

On a traced symmetric monoidal category, we can construct a *GoI situation*.

Definition 2.1.2 (retraction)**.** A *retraction* $f : X \lhd Y : g$ in a category $\mathbb{C}$ is a pair of arrows $f\colon X \to Y$ and $g\colon Y \to X$ in $\mathbb{C}$ that satisfies

$$g \circ f = \mathrm{id}_X \ .$$

If $f \circ g = \mathrm{id}_Y$ also holds then we write $f : X \cong Y : g$.

Definition 2.1.3 (GoI situation)**.** A *GoI situation* is a triple $(\mathbb{C}, U, F)$ that satisfies the following properties.

- The triple consists of a traced symmetric monoidal category $(\mathbb{C}, I, \otimes)$, an object U in $\mathbb{C}$ and a traced symmetric monoidal functor $F\colon \mathbb{C} \to \mathbb{C}$.

- There exist retractions:

$$\begin{array}{lll} \phi : U \otimes U \lhd U : \psi & u : FU \lhd U : v & n : I \lhd U : n' \\ e_A : A \lhd FA : e'_A & d_A : FFA \lhd FA : d'_A & \\ c_A : FA \otimes FA \lhd FA : c'_A & w_A : I \lhd FA : w'_A & \end{array}$$

  such that e, d, c and w are monoidal natural transformations.

From a GoI situation, a *linear combinatory algebra* is extracted via categorical GoI.

Definition 2.1.4 (linear combinatory algebra)**.** A *linear combinatory algebra* is a triple $(D, \cdot, !)$ that satisfies the following properties.

- The triple consists of a set D, a binary operation $\cdot \colon D \times D \to D$ (called *application*) and a unary operation $! \colon D \to D$.
- The set D has elements $\mathsf{I}, \mathsf{B}, \mathsf{C}, \mathsf{K}, \mathsf{W}, \mathsf{D}, \delta, \mathsf{F}$ (called *combinators*) that satisfy the following equations for all a, b, c in D:

$$\begin{array}{rr} \mathsf{I}a = a & \mathsf{B}abc = a(bc) \\ \mathsf{C}abc = acb & \mathsf{K}x\,!\,y = x \\ \mathsf{W}x\,!\,y = x\,!\,y\,!\,y & \mathsf{D}\,!\,x = x \\ \delta\,!\,x = !\,!\,x & \mathsf{F}\,!\,x\,!\,y = !(xy) \ . \end{array}$$

The application $\cdot$ in linear combinatory algebras is regarded as left-associative.

Proposition 2.1.5 (categorical GoI [1])**.** *Let $(\mathbb{C}, U, F)$ be a GoI situation. A triple $(\mathbb{C}(U,U), \cdot, !)$ forms a linear combinatory algebra where a binary operation $\cdot$ and a unary operation $!$ are defined as*

$$\begin{aligned} f \cdot g &:= \mathrm{tr}^U_{U,U}((\mathrm{id}_U \otimes g) \circ \psi \circ f \circ \phi) \\ !\,f &:= u \circ F(f) \circ v \ . \end{aligned}$$

Figure 2.3 shows string diagrams of arrows $f \cdot g$ and $!\,f$. The application $\cdot$ is represented by an interaction of the diagrams of f and g. The functor F is used to represent the ! modality.

2.2 Monads on Set

Let **Set** be a category of sets and functions. It is known that **Set** has finite coproducts $(\emptyset, +)$ and is a cartesian closed category i.e. a category with finite products $(1, \times)$ and exponents $\Rightarrow$. Here 1 represents an singleton $\{*\}$. We denote the injections, codiagonal maps, swappings of products and distributions as follows:

$$\begin{array}{cc} X \xrightarrow{\mathrm{inl}_{X,Y}} X + Y \xleftarrow{\mathrm{inr}_{X,Y}} Y & X + X \xrightarrow{\gamma_X} X \\ X \times Y \xrightarrow[\cong]{\sigma_{X,Y}} Y \times X & X \times Y + X \times Z \xrightarrow[\cong]{\delta_{X,Y,Z}} X \times (Y + Z) \ . \end{array}$$

Let $T \colon \mathbf{Set} \to \mathbf{Set}$ to be a monad. Since monads on **Set** are strong, T has the tensorial strengths. We denote the unit, multiplication and strengths of T as follows:

$$\begin{array}{cc} X \xrightarrow{\eta_X} TX & T^2X \xrightarrow{\mu_X} TX \\ TX \times Y \xrightarrow{\mathrm{st}_{X,Y}} T(X \times Y) & X \times TY \xrightarrow{\mathrm{st}'_{X,Y}} T(X \times Y) \ . \end{array}$$

Following [12], we use a monad T to model effects in computation.

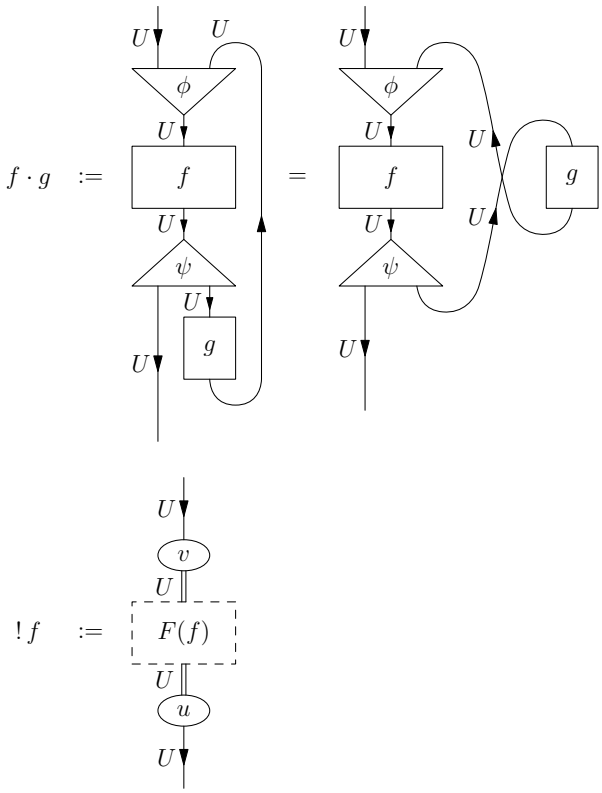


Figure 2.3: operators of a linear combinatory algebra

Example 2.2.1. We give examples of $T\colon \mathbf{Set} \to \mathbf{Set}$ that models an effect.

- The *list monad* $\mathcal{L}X = 1 + X$ that models choices between termination or continuation of computation.
- The *powerset monad* $\mathcal{P}X = 2^X$ that models nondeterministic choices of computation.
- The *subdistribution monad* $\mathcal{D}X = \{d\colon X \to [0,1] \mid \sum_{x \in X} d(x) \leq 1\}$ that models probabilistic choices of computation.

We write $\mathbf{Set}_T$ for the Kleisli category of T. Kleisli compositions are defined as

$$g \circ_T f = \mu_Z \circ T(g) \circ f\colon X \to_T Z$$

for all $f\colon X \to_T Y$ and $g\colon Y \to_T Z$ in $\mathbf{Set}_T$. Kleisli compositions and Kleisli arrows are denoted as $\circ_T$ and $\to_T$ respectively, while compositions and arrows in the base category **Set** are denoted as $\circ$ and $\to$ respectively.

The constructor $(-)^*$ is defined as

$$f^* := \eta_Y \circ f\colon X \to_T Y$$

for all $f\colon X \to Y$ in **Set**. It lifts arrows in **Set** to those in $\mathbf{Set}_T$, preserving identities and compositions. It also lifts coproducts by lifting coprojections. The operation $\otimes$ on $\mathbf{Set}_T$ is defined as

$$g \otimes D := \mathrm{st}_{X,D} \circ (g \times \mathrm{id}_D)\colon X \times D \to_T Y \times D$$
$$D \otimes g := \mathrm{st}'_{D,Y} \circ (\mathrm{id}_D \times g)\colon D \times X \to_T D \times Y$$

for all sets D and $g\colon X \to_T Y$ in $\mathbf{Set}_T$. It gives an premonoidal structure of $\mathbf{Set}_T$ [15]. For $f\colon X \to_T Y$ and $g\colon Z \to_T W$ in $\mathbf{Set}_T$, if the equation

$$(f \otimes W) \circ_T (X \otimes g) = (Y \otimes g) \circ_T (f \otimes Z)$$

holds, then we write $f \otimes g$ for $(f \otimes W) \circ_T (X \otimes g)$. It is easy to show that the constructor $(-)^*$ and the operator $\otimes$ satisfies the following equations

$$f^* \otimes g^* = (f \times g)^*$$
$$f^* \otimes A = f^* \otimes \mathrm{id}_A^*$$
$$A \otimes f^* = \mathrm{id}_A^* \otimes f^*$$

for all $f\colon X \to X'$ and $g\colon Y \to Y'$ in $\mathbf{Set}$ and for all sets A.

Arrows in $\mathbf{Set}_T$ can be seen as arrows in $\mathbf{Set}$ (i.e. functions) that are equipped with effects. Later we use them and form a GoI situation. To ensure that a GoI situation is formed, we require the following condition.

Requirement 2.2.2. In the whole paper, we require that $T\colon \mathbf{Set} \to \mathbf{Set}$ is a monad and it makes the symmetric monoidal category $(\mathbf{Set}_T, \emptyset, +)$ have restricted uniform trace operator tr [4]: for all $f\colon A+X \to_T B+X$ and $g\colon A+Y \to_T B+Y$ in $\mathbf{Set}_T$ and $h\colon X \to Y$ in $\mathbf{Set}$, if $(\mathrm{id}_B^* + h^*) \circ_T f = g \circ_T (\mathrm{id}_A^* + h^*)$ then $\mathrm{tr}^X_{A,B}(f) = \mathrm{tr}^Y_{A,B}(g)$.

The following lemma shows the useful sufficient condition of Requirement 2.2.2. All monads in Example 2.2.1 satisfies this condition. We give a detailed proof of this lemma in the remainder of this section.

Lemma 2.2.3. *If $\mathbf{Set}_T$ is a* **Cppo***-enriched cocartesian category and the bottom maps $\{\bot_{A,B}\colon A \to_T B\}_{A,B \in \mathbf{Set}_T}$ satisfies the following equations*

$$f \circ_T \bot_{A,B} = \bot_{A,B'}$$
$$\bot_{A,B} \circ_T g^* = \bot_{A',B}$$

for all $f\colon B \to_T B'$ in $\mathbf{Set}_T$ and $g\colon A' \to A$ in $\mathbf{Set}$, then T satisfies Requirement 2.2.2.

Proof. For all arrows h in $\mathbf{Set}$, we write $\mathcal{S}$ for the collection of lifted arrows h^* in $\mathbf{Set}_T$. We want to show that $\mathbf{Set}_T$ has a uniform trace operator for $\mathcal{S}$.

The dual category of $\mathbf{Set}_T$, namely $\mathbf{Set}_T^{\mathrm{op}}$, is a **Cppo**-enriched cartesian category since coproducts in one category are products in its dual category. As the special case of the result in [4], on $\mathbf{Set}_T^{\mathrm{op}}$, uniform Conway operators for $\mathcal{S}$ and uniform trace operators for $\mathcal{S}$ coincide. In addition, it is easy to show that uniform trace operators for $\mathcal{S}$ on $\mathbf{Set}_T$ and those for $\mathcal{S}$ on $\mathbf{Set}_T^{\mathrm{op}}$ correspond. Therefore the existence of a uniform trace operator for $\mathcal{S}$ on $\mathbf{Set}_T$ is equivalent to the existence of a uniform Conway operator for $\mathcal{S}$ on $\mathbf{Set}_T^{\mathrm{op}}$.

$$\cfrac{\text{a uniform Conway operator for } \mathcal{S} \text{ on } \mathbf{Set}_T^{\mathrm{op}}}{\cfrac{\text{a uniform trace operator for } \mathcal{S} \text{ on } \mathbf{Set}_T^{\mathrm{op}}}{\text{a uniform trace operator for } \mathcal{S} \text{ on } \mathbf{Set}_T}}$$

Here we define an operator $(-)^\dagger$ on $\mathbf{Set}_T$ and show that the corresponding operator of $(-)^\dagger$ on $\mathbf{Set}_T^{\mathrm{op}}$ is a uniform Conway operator for $\mathcal{S}$.

First, we explain that $(-)^\dagger$ can be defined by using the least fixpoint operator fix on a cppo i.e. a pointed complete partial order. Details of the least fixpoint

operator on cppo's and the fixpoint theorem are in [17]. For a map $f\colon X \to_T A + X$, a function $F_f\colon \mathbf{Set}_T(X,A) \to \mathbf{Set}_T(X,A)$ can be defined as

$$F_f(g) := [\mathrm{id}_A^*, g] \circ_T f\colon X \to_T A$$

for all $g\colon X \to_T A$. This function F_f is continuous for all f, i.e. F_f satisfies the following properties.

- For each $g, g'\colon X \to_T A$, if $g \sqsubseteq g'$ then $F_f(g) \sqsubseteq F_f(g')$ holds because of the continuity of the composition $\circ_T$ and cotupling $[-,-]$.
- For all ω-chains $g_0 \sqsubseteq g_1 \sqsubseteq \ldots \sqsubseteq g_n \sqsubseteq \ldots$, its least upper bound is preserved by F_f:

$$\begin{aligned}
\bigsqcup_n F_f(g_n) &= \bigsqcup_n ([\mathrm{id}_A^*, g_n] \circ_T f) \\
&= \bigsqcup_n [\mathrm{id}_A^*, g_n] \circ_T f \quad \text{(continuity of composition)} \\
&= [\mathrm{id}_A^*, \bigsqcup_n g_n] \circ_T f \quad \text{(continuity of cotupling)} \\
&= F_f(\bigsqcup_n g_n) \ .
\end{aligned}$$

Therefore we can take the least fixpoint of F_f by the fixpoint theorem and define $f^\dagger$ as

$$\begin{aligned}
f^\dagger &:= \mathrm{fix}(F_f) \\
&= \bigsqcup_n F_f^n(\bot_{X,A})\colon X \to_T A \ .
\end{aligned}$$

Second, we show that the operator $(-)^\dagger$ on $\mathbf{Set}_T$ gives rise to a uniform Conway operator for $\mathcal{S}$ on $\mathbf{Set}_T^{\mathrm{op}}$. We overload the operator $(-)^\dagger$ as below:

$$\frac{\dfrac{\dfrac{f\colon A \times X \to X \ \text{ in } \mathbf{Set}_T^{\mathrm{op}}}{f\colon X \to A + X \ \text{ in } \mathbf{Set}_T}}{f^\dagger\colon X \to A \ \text{ in } \mathbf{Set}_T}}{f^\dagger\colon A \to X \ \text{ in } \mathbf{Set}_T^{\mathrm{op}}}$$

for each map $f\colon A \times X \to X$ in $\mathbf{Set}_T^{\mathrm{op}}$. Obviously, the operator $(-)^\dagger$ on $\mathbf{Set}_T^{\mathrm{op}}$ is a uniform Conway operator for $\mathcal{S}$ if and only if the operator $(-)^\dagger$ on $\mathbf{Set}_T$ satisfies the following five conditions:

1. $((g + \mathrm{id}_X^*) \circ_T f)^\dagger = g \circ_T f^\dagger$ for all $f\colon X \to_T A + X$ and $g\colon A \to_T A'$
2. $f^\dagger = [\mathrm{id}_A^*, f^\dagger] \circ_T f$ for all $f\colon X \to_T A + X$
3. $([\mathrm{inl}_{A,X}^*, g] \circ_T f)^\dagger = [\mathrm{id}_A^*, ([\mathrm{inl}_{A,Y}^*, f] \circ_T g)^\dagger] \circ_T f$ for all $f\colon X \to_T A + Y$ and $g\colon Y \to_T A + X$
4. $((\mathrm{id}_A^* + \nabla_X) \circ_T f)^\dagger = (f^\dagger)^\dagger$ for all $f\colon X \to_T A + X + X$, where $\nabla_X := [\mathrm{id}_X^*, \mathrm{id}_X^*]\colon X + X \to_T X$ is the codiagonal map
5. $f \circ_T h^* = (\mathrm{id}_A^* + h^*) \circ_T g$ for all $f\colon X \to_T A + X$ and $g\colon Y \to_T A + Y$ in $\mathbf{Set}_T$ and $h\colon Y \to X$ in $\mathbf{Set}$.

These conditions are the dual version of the definition of a uniform Conway operator explained in [4]. We show how these five conditions hold in the following.

1. For all $f\colon X \to_T A + X$ and $g\colon A \to_T A'$, the equation
$$F^n_{(g+\mathrm{id}^*_X)\circ_T f}(\bot_{X,A'}) = g \circ_T F^n_f(\bot_{X,A})$$
holds for all $n \in \omega$. This can be proved by the induction on n:
   - If $n = 0$ then $\bot_{X,A'} = g \circ_T \bot_{X,A}$.
   - If the equation holds for n then
$$\begin{aligned}
&F^{n+1}_{(g+\mathrm{id}^*_X)\circ_T f}(\bot_{X,A'}) \\
&= [\mathrm{id}^*_A, F^n_{(g+\mathrm{id}^*_X)\circ_T f}(\bot_{X,A'})] \circ_T (g + \mathrm{id}^*_X) \circ_T f \\
&= [\mathrm{id}^*_A, g \circ_T F^n_f(\bot_{X,A})] \circ_T (g + \mathrm{id}^*_X) \circ_T f \ \text{(induction hypothesis)} \\
&= [g, g \circ_T F^n_f(\bot_{X,A})] \circ_T f \\
&= g \circ_T [\mathrm{id}^*_A, F^n_f(\bot_{X,A})] \circ_T f \\
&= g \circ_T F^{n+1}_f(\bot_{X,A}) \ .
\end{aligned}$$
   Therefore it holds that
$$\begin{aligned}
((g + \mathrm{id}^*_X) \circ_T f)^\dagger &= \bigsqcup_n F^n_{(g+\mathrm{id}^*_X)\circ_T f}(\bot_{X,A'}) \\
&= \bigsqcup_n (g \circ_T F^n_f(\bot_{X,A})) \\
&= g \circ_T \bigsqcup_n F^n_f(\bot_{X,A}) \\
&= g \circ_T f^\dagger \ .
\end{aligned}$$

2. Because fix is the least fixpoint operator, it holds that
$$\begin{aligned}
f^\dagger &= \mathrm{fix}(F_f) \\
&= F_f(\mathrm{fix}(F_f)) \\
&= [\mathrm{id}^*_A, \mathrm{fix}(F_f)] \circ_T f \\
&= [\mathrm{id}^*_A, f^\dagger] \circ_T f \ .
\end{aligned}$$
   for all $f\colon X \to_T A + X$.

3. For all $f\colon X \to_T A + Y$ and $g\colon Y \to_T A + X$, we need to confirm the following equation
$$\mathrm{fix}(F_{[\mathrm{inl}^*_{A,X}, g]\circ_T f}) = [\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f \ .$$
   This equation can be confirmed by proving that $[\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f$ is the least fixpoint of $F_{[\mathrm{inl}^*_{A,X}, g]\circ_T f}$. First, $[\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f$ is a fixpoint of $F_{[\mathrm{inl}^*_{A,X}, g]\circ_T f}$:
$$\begin{aligned}
&F_{[\mathrm{inl}^*_{A,X}, g]\circ_T f}([\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f) \\
&= [\mathrm{id}^*_A, [\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f] \circ_T [\mathrm{inl}^*_{A,X}, g] \circ_T f \\
&= [\mathrm{id}^*_A, [\mathrm{id}^*_A, [\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f] \circ_T g] \circ_T f \\
&= [\mathrm{id}^*_A, [\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T [\mathrm{inl}^*_{A,Y}, f] \circ_T g] \circ_T f \\
&= [\mathrm{id}^*_A, F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g}(\mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g}))] \circ_T f \\
&= [\mathrm{id}^*_A, \mathrm{fix}(F_{[\mathrm{inl}^*_{A,Y}, f]\circ_T g})] \circ_T f \ .
\end{aligned}$$

Second, $[\mathrm{id}_A^*, \mathrm{fix}(F_{[\mathrm{inl}_{A,Y}^*, f]\circ_T g})] \circ_T f$ is less than any fixpoint of $F_{[\mathrm{inl}_{A,X}^*, g]\circ_T f}$: if $h\colon X \to_T A$ is a fixpoint of $F_{[\mathrm{inl}_{A,X}^*, g]\circ_T f}$, then $[\mathrm{id}_A^*, h] \circ_T g$ is a fixpoint of $F_{[\mathrm{inl}_{A,Y}^*, f]\circ_T g}$, as shown below

$$\begin{aligned}
F_{[\mathrm{inl}_{A,Y}^*, f]\circ_T g}([\mathrm{id}_A^*, h] \circ_T g) &= [\mathrm{id}_A^*, [\mathrm{id}_A^*, h] \circ_T g] \circ_T [\mathrm{inl}_{A,Y}^*, f] \circ_T g \\
&= [\mathrm{id}_A^*, [\mathrm{id}_A^*, [\mathrm{id}_A^*, h] \circ_T g] \circ_T f] \circ_T g \\
&= [\mathrm{id}_A^*, [\mathrm{id}_A^*, h] \circ_T [\mathrm{inl}_{A,X}^*, g] \circ_T f] \circ_T g \\
&= [\mathrm{id}_A^*, F_{[\mathrm{inl}_{A,X}^*, g]\circ_T f}(h)] \circ_T g \\
&= [\mathrm{id}_A^*, h] \circ_T g \ ,
\end{aligned}$$

therefore it holds that

$$\begin{aligned}
[\mathrm{id}_A^*, \mathrm{fix}(F_{[\mathrm{inl}_{A,Y}^*, f]\circ_T g})] \circ_T f &\sqsubseteq [\mathrm{id}_A^*, [\mathrm{id}_A^*, h] \circ_T g] \circ_T f \\
&= [\mathrm{id}_A^*, h] \circ_T [\mathrm{inl}_{A,X}^*, g] \circ_T f \\
&= F_{[\mathrm{inl}_{A,X}^*, g]\circ_T f}(h) \\
&= h \ .
\end{aligned}$$

4. For all $f\colon X \to_T A + X + X$, by showing that $F_{(\mathrm{id}_A^* + \nabla_X)\circ_T f}$ is a fixpoint of $F_{\mathrm{fix}(F_f)}$ and vise versa, we can prove the equation

$$\mathrm{fix}(F_{(\mathrm{id}_A^* + \nabla_X)\circ_T f}) = \mathrm{fix}(F_{\mathrm{fix}(F_f)}) \ .$$

First, if $h\colon X \to_T A$ is a fixpoint of $F_{\mathrm{fix}(F_f)}$ then h is also a fixpoint of $F_{(\mathrm{id}_A^* + \nabla_X)\circ_T f}$:

$$\begin{aligned}
F_{(\mathrm{id}_A^* + \nabla_X)\circ_T f}(h) &= [\mathrm{id}_A^*, h] \circ_T (\mathrm{id}_A^* + \nabla_X) \circ_T f \\
&= [\mathrm{id}_A^*, h \circ_T \nabla_X] \circ_T f \\
&= [\mathrm{id}_A^*, [h, h]] \circ_T f \\
&= [[\mathrm{id}_A^*, h], h] \circ_T f \ \ \text{(associative law of } +) \\
&= [[\mathrm{id}_A^*, h], [\mathrm{id}_A^*, h] \circ_T \mathrm{fix}(F_f)] \circ_T f \\
&= [\mathrm{id}_A^*, h] \circ_T [\mathrm{id}_A^*, \mathrm{fix}(F_f)] \circ_T f \\
&= [\mathrm{id}_A^*, h] \circ_T \mathrm{fix}(F_f) \\
&= h \ .
\end{aligned}$$

Second, if $h\colon X \to_T A$ is a fixpoint of $F_{(\mathrm{id}_A^* + \nabla_X)\circ_T f}$ then h satisfies the inequation

$$\begin{aligned}
\mathrm{fix}(F_{\mathrm{fix}(F_f)}) &= \bigsqcup_n F^n_{\bigsqcup_m F_f^m(\bot_{X,A+X})}(\bot_{X,A}) \\
&\sqsubseteq h \ .
\end{aligned}$$

This can be confirmed by proving that the inequation

$$F^n_{\bigsqcup_m F_f^m(\bot_{X,A+X})}(\bot_{X,A}) \sqsubseteq h$$

holds for all $n \in \omega$. This inequation is proved by induction on n:

- If $n = 0$ then $\bot_{X,A} \sqsubseteq h$.

- Here we use g to represent $F^n_{\bigsqcup_m F^m_f(\bot_{X,A+X})}(\bot_{X,A})$. Assume that the inequation holds for n, namely $g \sqsubseteq h$. We want to confirm the following inequation

$$\begin{aligned} F^{n+1}_{\bigsqcup_m F^m_f(\bot_{X,A+X})}(\bot_{X,A}) &= F_{\bigsqcup_m F^m_f(\bot_{X,A+X})}(g) \\ &= [\mathrm{id}^*_A, g] \circ_T \bigsqcup_m F^m_f(\bot_{X,A+X}) \\ &= \bigsqcup_m ([\mathrm{id}^*_A, g] \circ_T F^m_f(\bot_{X,A+X})) \\ &= \bigsqcup_m F_{F^m_f(\bot_{X,A+X})}(g) \\ &\sqsubseteq h \ . \end{aligned}$$

  This inequation can be proved by confirming that the inequation

$$[\mathrm{id}^*_A, g] \circ_T F^m_f(\bot_{X,A+X}) = F_{F^m_f(\bot_{X,A+X})}(g) \sqsubseteq h$$

  holds for all $m \in \omega$. We prove this by induction on m:
  - If $m = 0$ then $[\mathrm{id}^*_A, g] \circ_T \bot_{X,A+X} = \bot_{X,A} \sqsubseteq h$.
  - If the target inequation holds for m, then it holds that

$$\begin{aligned} [\mathrm{id}^*_A, g] \circ_T F^{m+1}_f(\bot_{X,A+X}) &= [\mathrm{id}^*_A, g] \circ_T [\mathrm{id}^*_{A+X}, F^m_f(\bot_{X,A+X})] \circ_T f \\ &= [[\mathrm{id}^*_A, g], [\mathrm{id}^*_A, g] \circ_T F^m_f(\bot_{X,A+X})] \circ_T f \\ &= [[\mathrm{id}^*_A, g], F_{F^m_f(\bot_{X,A+X})}(g)] \circ_T f \\ &= [\mathrm{id}^*_A, [g, F_{F^m_f(\bot_{X,A+X})}(g)]] \circ_T f \\ &\sqsubseteq [\mathrm{id}^*_A, [h, h]] \circ_T f \ \text{ (induction hypotheses)} \\ &= [\mathrm{id}^*_A, h] \circ_T (\mathrm{id}^*_A + \nabla_X) \circ_T f \\ &= F_{(\mathrm{id}^*_A + \nabla_X) \circ_T f}(h) \\ &= h \ . \end{aligned}$$

5. For all $f \colon X \to_T A + X$ and $g \colon Y \to_T A + Y$ in $\mathbf{Set}_T$ and $h \colon Y \to_T X$ in $\mathbf{Set}$, we want to prove the equation

$$\mathrm{fix}(F_g) = \mathrm{fix}(F_f) \circ_T h^*$$

   with the assumption $f \circ_T h^* = (\mathrm{id}^*_A + h^*) \circ_T g$. Here we show that $\mathrm{fix}(F_f) \circ_T h^*$ is the least fixpoint of F_g. First, $\mathrm{fix}(F_f) \circ_T h^*$ is a fixpoint of F_g:

$$\begin{aligned} F_g(\mathrm{fix}(F_f) \circ_T h^*) &= [\mathrm{id}^*_A, \mathrm{fix}(F_f) \circ_T h^*] \circ_T g \\ &= [\mathrm{id}^*_A, \mathrm{fix}(F_f)] \circ_T (\mathrm{id}^*_A + h^*) \circ_T g \\ &= [\mathrm{id}^*_A, \mathrm{fix}(F_f)] \circ_T f \circ_T h^* \\ &= F_f(\mathrm{fix}(F_f)) \circ_T h^* \\ &= \mathrm{fix}(F_f) \circ_T h^* \ . \end{aligned}$$

   Second, $\mathrm{fix}(F_f) \circ_T h^*$ is less than any fixpoint of F_g: if $k \colon Y \to_T A$ is a fixpoint of F_g, it holds that

$$\begin{aligned} \mathrm{fix}(F_f) \circ_T h^* &= \bigsqcup_n F^n_f(\bot_{X,A}) \circ_T h^* \\ &= \bigsqcup_n (F^n_f(\bot_{X,A}) \circ_T h^*) \\ &\sqsubseteq k \ . \end{aligned}$$

This is the consequence of the inequation $F_f^n(\bot_{X,A}) \circ_T h^* \sqsubseteq k$ that holds for all $n \in \omega$. This can be proved by induction on n:

- If $n = 0$ then $\bot_{X,A} \circ_T h^* = \bot_{Y,A} \sqsubseteq k$.
- If the inequation holds for n, it holds that

$$\begin{aligned} F_f^{n+1}(\bot_{X,A}) \circ_T h^* &= [\mathrm{id}_A^*, F_f^n(\bot_{X,A})] \circ_T f \circ_T h^* \\ &= [\mathrm{id}_A^*, F_f^n(\bot_{X,A})] \circ_T (\mathrm{id}_A^* + h^*) \circ_T g \\ &= [\mathrm{id}_A^*, F_f^n(\bot_{X,A}) \circ_T h^*] \circ_T g \\ &\sqsubseteq [\mathrm{id}_A^*, k] \circ_T g \ \text{(induction hypothesis)} \\ &= F_g(k) \\ &= k \ . \end{aligned}$$

□

2.3 Algebraic Operations

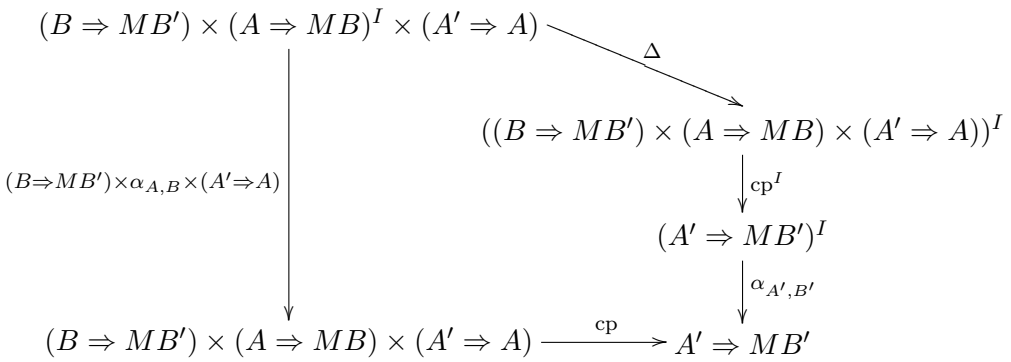
In [14] Plotkin and Power gives adequate denotational semantics for algebraic effects by using *algebraic operations*. Following this result, we utilize algebraic operations in this paper with the hope that these enable resumptions to represent algebraic effects. In this section we study algebraic operations.

First we recall the definition of algebraic operations in [13].

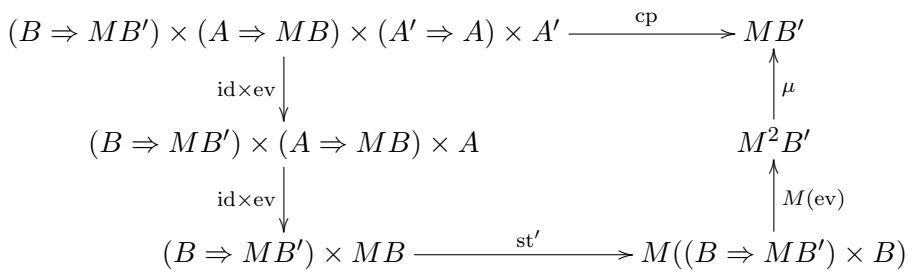
Definition 2.3.1 (algebraic operation [13])**.** Let $(\mathbb{C}, 1, \times, \Rightarrow)$ be a cartesian closed category with countable products $\prod$ and $M\colon \mathbb{C} \to \mathbb{C}$ be a strong monad. For a countable set I, an *I-ary algebraic operation on M* is a family of arrows in $\mathbb{C}$

$$\{\alpha_{A,B}\colon (A \Rightarrow MB)^I \to (A \Rightarrow MB)\}_{A,B \in \mathbb{C}}$$

that is natural: for each A, A', B and B' in $\mathbb{C}$, the following diagram commutes



where the arrow Δ is diagonal for the first and third arguments and cp is defined as the bijective correspondent of the arrow cp in the following commutative diagram.



The arrow cp can be seen as the Kleisli compositions. Since there is an isomorphism between an exponent object $A \Rightarrow MB$ in **Set** and a homset $\mathbf{Set}_M(A, B)$, if $\mathbb{C}$ is **Set** then the naturality of algebraic operations can be also written as below:

$$\alpha_{A',B'}\{k \circ_M f_i \circ_M h^*\}_{i\in I} = k \circ_M \alpha_{A,B}\{f_i\}_{i\in I} \circ_M h^*$$

for all $h\colon A' \to A$ in **Set**, $k\colon B \to_M B'$ in $\mathbf{Set}_M$ and $\{f_i\colon A \to_M B\}_{i\in I}$ in $\mathbf{Set}_M$.

We denote the projections in $\mathbb{C}$ by

$$\prod_{j\in J} X_j \xrightarrow{\pi_j} X_j$$

for a countable set J. Obviously these projections π_j are J-ary algebraic operations on all strong monads M on $\mathbb{C}$. We give an example of algebraic operations on T, namely that on $\mathcal{P}$ in Example 2.2.1.

Example 2.3.2. For all arrows f and g in $\mathbf{Set}_\mathcal{P}$, we define a family of maps $\{\oplus_{A,B}\colon \mathbf{Set}_T(A, B)^2 \to \mathbf{Set}_T(A, B)\}_{A,B\in\mathbf{Set}}$ as

$$(f \oplus g)(x) := f(x) \cup g(x) \ .$$

This is a 2-ary algebraic operation on $\mathcal{P}$. Its naturality can be confirmed as below: for all a' in A', $h\colon A' \to A$ in **Set**, $k\colon B \to_\mathcal{P} B'$ in $\mathbf{Set}_\mathcal{P}$ and $\{f_i\colon A \to_\mathcal{P} B\}_{i\in I}$ in $\mathbf{Set}_\mathcal{P}$, it holds that

$$\begin{aligned}
&((k \circ_\mathcal{P} f_1 \circ_\mathcal{P} h^*) \oplus_{A',B'} (k \circ_\mathcal{P} f_2 \circ_\mathcal{P} h^*))(a') \\
&= (k \circ_\mathcal{P} f_1 \circ_\mathcal{P} h^*)(a') \cup (k \circ_\mathcal{P} f_2 \circ_\mathcal{P} h^*)(a') \\
&= (k \circ_\mathcal{P} f_1)(h(a')) \cup (k \circ_\mathcal{P} f_2)(h(a')) \\
&= \{k(b) \mid b \in f_1(h(a'))\} \cup \{k(b) \mid b \in f_2(h(a'))\} \\
&= \{k(b) \mid b \in (f_1(h(a')) \cup f_2(h(a')))\} \\
&= \{k(b) \mid b \in (f_1 \oplus_{A,B} f_2)(h(a'))\} \\
&= (k \circ_\mathcal{P} (f_1 \oplus_{A,B} f_2))(h(a')) \\
&= (k \circ_\mathcal{P} (f_1 \oplus_{A,B} f_2) \circ_\mathcal{P} h^*)(a') \ .
\end{aligned}$$

A category of algebraic operations on M can be defined as below.

- Objects are countable sets.
- An arrow from I to J is a family of arrows $\{\alpha_{A,B}\colon (A \Rightarrow MB)^I \to (A \Rightarrow MB)^J\}_{A,B\in\mathbb{C}}$ in $\mathbb{C}$ such that $\pi_j \circ \alpha$ is an algebraic operation on M for all j in J.
- The identity arrow of I is $\{\mathrm{id}_{(A\Rightarrow MB)}\colon (A \Rightarrow MB)^I \to (A \Rightarrow MB)^I\}_{A,B\in\mathbb{C}}$.
- The composition of arrows $\{\alpha_{A,B}\colon (A \Rightarrow MB)^I \to (A \Rightarrow MB)^{I'}\}_{A,B\in\mathbb{C}}$ from I to I' and $\{\beta_{A,B}\colon (A \Rightarrow MB)^{I'} \to (A \Rightarrow MB)^J\}_{A,B\in\mathbb{C}}$ from I' to J is $\{\beta_{A,B} \circ \alpha_{A,B}\colon (A \Rightarrow MB)^I \to (A \Rightarrow MB)^J\}_{A,B\in\mathbb{C}}$ from I to J.

It is straightforward to confirm that the compositions indeed give algebraic operations because the projections π_j are algebraic operations on M. We write $\mathbf{AlgOp}_M$ for this category.

The category $\mathbf{AlgOp}_M$ has countable products given by the countable disjoint union $\biguplus_{j\in J} I_j$ of countable sets. This can be shown by the following bijective correspondences.

$$\dfrac{\dfrac{\dfrac{\{(A \Rightarrow MB)^I \to (A \Rightarrow MB)^{I_j}\}_{j\in J}}{\{(A \Rightarrow MB)^I \xrightarrow{f_{j,k}} (A \Rightarrow MB)\}_{j\in J, k\in I_j}}}{\{(A \Rightarrow MB)^I \xrightarrow{f_{j,k}} (A \Rightarrow MB)\}_{(j,k)\in \biguplus_{j\in J} I_j}}}{(A \Rightarrow MB)^I \to (A \Rightarrow MB)^{\biguplus_{j\in J} I_j}}$$

Due to this property, new algebraic operations can be constructed of existing ones: for example, $(x \oplus y) \oplus z$ constructed of $\oplus$ in Example 2.3.2 is a 3-ary algebraic operation on $\mathcal{P}$. The category $\mathbf{AlgOp}_M$ is indeed a Lawvere theory of algebraic operations (see e.g. [8]): it represents algebraic operations as arrows and the equational theory of algebraic operations as commutative diagrams.

2.4 Transducers

We embed "memories" as well as effects into arrows in **Set** (i.e. functions) and obtain *T-transducers*.

Definition 2.4.1 (T-transducer)**.** For sets A and B, a *T-transducer from A to B* is a triple (X, c, x) consisting of a set X, an arrow $c\colon X \times A \to_T T(X \times B)$ in $\mathbf{Set}_T$ and an arrow $x\colon 1 \to X$ in **Set**.

T-transducers (X, c, x) from A to B are machines that have inputs A, outputs B and internal states X. The arrow x indicates the initial state and the arrow c is the transition function: given an input and the current state it decides the output(s) and the next state(s). This decision is affected by the effect that is represented by T. We write $(X, c, x)\colon A \nrightarrow B$ for T-transducers from A to B.

T-transducers $(1, c, \mathrm{id}_1)\colon A \nrightarrow B$ with one internal state behave as functions from A to B and can be seen as "memoryless" transducers. Two constructors J and J_0 that lifts functions to T-transducers can be defined as

$$\begin{aligned} Jf &:= (1, f, \mathrm{id}_1)\colon A \nrightarrow B \text{ for all } f\colon A \to_T B \text{ in } \mathbf{Set}_T \\ J_0 g &:= Jg^* = (1, g^*, \mathrm{id}_1)\colon A \nrightarrow B \text{ for all } g\colon A \to B \text{ in } \mathbf{Set}. \end{aligned}$$

In the remainder of this section, we introduce some notions on T-transducers. Here we use quotation marks (" ") to represent each notion because in fact these notions do not always satisfy corresponding axiomatizations. The main reason for this problem is that T-transducers of the same behavior, i.e. T-transducers that return the same output(s) to the same input, are sometimes distinguished due to their unequal state spaces. To fix this problem and form a GoI situation based on T-transducers, we introduce an appropriate equivalence relation called *behavioral equivalence* in the next section.

The "composition" of two T-transducers $(X, c, x)\colon A \nrightarrow B$ and $(Y, d, y)\colon B \nrightarrow C$ is defined as

$$(Y, d, y) \circ (X, c, x) := (X \times Y, e, x \times y)\colon A \nrightarrow C$$

where $e\colon X \times Y \times A \to_T X \times Y \times C$ is defined as

$$e := (X \otimes d) \circ_T (X \otimes \sigma^*_{B,Y}) \circ_T (c \otimes Y) \circ_T (X \otimes \sigma^*_{Y,A}) \ .$$

One transition of this "composition" is internally two sequential transitions: the first transition is according to c and the second one is according to d.

For each set A we define the "identity" of A as

$$J\eta_A = (1, \eta_A, \mathrm{id}_1) \colon A \nrightarrow A \ .$$

This "identity" is an example of the notion that does not satisfy the corresponding axiomatizations: the "identity" is not actually neutral for "composition" because T-transducers with unequal state spaces are distinguished. The "composition" of a T-transducer with state space X and the "identity" of A has the state space $X \times 1$ or $1 \times X$ that is isomorphic but unequal to the original state space X.

For two T-transducers $(X, c, x) \colon A \nrightarrow B$ and $(Y, d, y) \colon C \nrightarrow D$, the "monoidal product" is defined as

$$(X, c, x) \boxplus (Y, d, y) := (X \times Y, e, x \times y) \colon A + C \nrightarrow B + D$$

where $e \colon X \times Y \times (A + C) \to_T X \times Y \times (B + D)$ is defined as the unique arrow such that

$$\begin{aligned} e \circ_T (X \otimes Y \otimes \mathrm{inl}^*_{A,C}) &= (\sigma^*_{Y,X} \otimes \mathrm{inl}^*_{B,D}) \circ_T (Y \otimes c) \circ_T (\sigma^*_{X,Y} \otimes A) \\ e \circ_T (X \otimes Y \otimes \mathrm{inr}^*_{A,C}) &= (X \otimes Y \otimes \mathrm{inr}^*_{B,D}) \circ_T (X \otimes d) \ . \end{aligned}$$

The "unit" of the "monoidal product" is defined as the unique T-transducer

$$J\eta_\emptyset = (1, \eta_\emptyset, \mathrm{id}_1) \colon \emptyset \nrightarrow \emptyset \ .$$

The "monoidal product" of T-transducers has two parallel T-transducers internally. Each time when an input comes, it chooses which T-transducer to run according to the input.

The "trace" of a T-transducer $(X, c, x) \colon A + C \nrightarrow B + C$ is defined as below by using the trace operator tr on $\mathbf{Set}_T$:

$$\mathrm{Tr}^C_{A,B}(X, c, x) := (X, e, x) \colon A \nrightarrow B$$

where $e \colon X \times A \to_T X \times B$ is defined as

$$e := \mathrm{tr}^{X \times C}_{X \times A, X \times B}((\delta^*_{X,B,C})^{-1} \circ_T c \circ_T \delta^*_{X,A,C}) \ .$$

We expect this "trace" to be a trace in Definition 2.1.1 with respect to the "symmetric monoidal structure" $(\boxplus, \emptyset)$.

Let $\mathbb{N}$ be the set of natural numbers. For each n in $\mathbb{N}$, set X and $f \colon A \to B$ in $\mathbf{Set}$, we define the following three arrows in $\mathbf{Set}$:

$$\begin{aligned} \kappa_n &\colon 1 \to \mathbb{N} \\ &\quad * \mapsto n \\ \omega_{n,X} &\colon X^{\mathbb{N}} \to X^{\mathbb{N}} \times X \\ &\quad (x_0, x_1, \ldots, x_n, \ldots) \mapsto ((x_0, x_1, \ldots, x_{n-1}, x_{n+1}, \ldots), x_n) \\ f^{\mathbb{N}} &\colon A^{\mathbb{N}} \to B^{\mathbb{N}} \\ &\quad (a_0, a_1, \ldots) \mapsto (f(a_0), f(a_1), \ldots) \ . \end{aligned}$$

The first arrow is the constant function that returns n. The second one extracts the n-th object from a list. The last one applies f to each object in a list.

We define an operator F on both sets and T-transducers. For each set A, the set FA is defined as

$$FA := \mathbb{N} \times A \ .$$

For each T-transducer $(X, c, x)\colon A \nrightarrow B$, the T-transducer $F(X, c, x)\colon FA \nrightarrow FB$ is defined as

$$F(X, c, x) := (X^{\mathbb{N}}, e, x^{\mathbb{N}})$$

where $e\colon X^{\mathbb{N}} \times FA \to_T X^{\mathbb{N}} \times FB$ is defined as the unique arrow such that

$$e \circ_T (X^{\mathbb{N}} \otimes \kappa_n^* \otimes A) = ((\omega_{n,X}^*)^{-1} \otimes \kappa_n^* \otimes B) \circ_T (X^{\mathbb{N}} \otimes c) \circ_T (\omega_{n,X}^* \otimes A)$$

holds for all n in $\mathbb{N}$. The operator F on T-transducers makes the countable infinite copies of the given T-transducer (X, c, x). These copies runs independently: each input to the T-transducer $F(X, c, x)$ is read by one corresponding T-transducer (X, c, x) and an output is made by this T-transducer (X, c, x).

Let $\phi : \mathbb{N} + \mathbb{N} \cong \mathbb{N} : \psi$ and $u : F\mathbb{N} \cong \mathbb{N} : v$ be bijections in **Set**. We choose pairs of T-transducers that expected to be "retractions":

$$\begin{aligned}
J_0\phi &: \mathbb{N} + \mathbb{N} \cong \mathbb{N} : J_0\psi \\
J_0 u &: F\mathbb{N} \cong \mathbb{N} : J_0 v \\
J_0\bot_{\mathbb{N}} &: \emptyset \lhd \mathbb{N} : J(\mathrm{tr}^{\mathbb{N}}_{\mathbb{N},\emptyset}(\gamma^*_{\mathbb{N}})) \\
J_0(\kappa_1 \times A) &: A \lhd FA : J_0(\top_{\mathbb{N}} \times A) && \text{(dereliction)} \\
J_0(u \times A) &: FFA \cong FA : J_0(v \times A) && \text{(digging)} \\
J_0\bot_{FA} &: \emptyset \lhd FA : J(\mathrm{tr}^{FA}_{FA,\emptyset}(\gamma^*_{FA})) && \text{(weakening)} \\
J_0(\phi \times A) &: FA + FA \cong FA : J_0(\psi \times A) && \text{(contraction)}
\end{aligned}$$

where $\bot_X\colon \emptyset \to X$ is the unique arrow from the initial object $\emptyset$ and $\top_X\colon X \to 1$ is the unique arrow to the final object 1 in **Set**. We expect that the operator F and these "retraction" forms a "GoI situation". These "retractions" are used to construct T-transducers of existing ones: for a T-transducer $(X, c, x)\colon A \nrightarrow A$ and a retraction $(Y, d, y) : A' \lhd A : (Y', d', y')$, a T-transducer $(X', c', x')\colon A' \nrightarrow A'$ is constructed as

$$(X', c', x') := (Y', d', y') \circ (X, c, x) \circ (Y, d, y) \ .$$

Figure 2.4 illustrates the constructions induced by "retractions". Given a branch of T-transducers, dereliction merges all T-transducers in the branch into one, digging sorts T-transducers and makes countably infinite branches, weakening discards the branch completely and contraction divides the branch into two branches.

We define a constructor $\overline{(-)}$ that makes an operator on T-transducers from an algebraic operation on T. For an I-ary algebraic operation α on T and a family of T-transducers $\{(X_i, c_i, x_i)\colon A \nrightarrow B\}_{i\in I}$, an operator $\overline{\alpha}$ is defined as

$$\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i\in I} := (1 + \coprod_{i\in I} X_i, d, \mathrm{inl}^*_{1,\coprod_{i\in I}})$$

where $d\colon (1 + \coprod_{i\in I} X_i) \times A \to_T (1 + \coprod_{i\in I} X_i) \times B$ is the unique arrow such that

$$\begin{aligned}
d \circ_T \mathrm{inl}^*_{1,\coprod_{i\in I} X_i} &= \alpha_{A,B}\{\widetilde{c_i}\}_{i\in I} \\
d \circ_T ((\mathrm{inr}^*_{1,\coprod_{i\in I} X_i} \circ_T \mathrm{inj}_i^*) \otimes A) &= ((\mathrm{inr}^*_{1,\coprod_{i\in I} X_i} \circ_T \mathrm{inj}_i^*) \otimes B) \circ_T c_i \ .
\end{aligned}$$

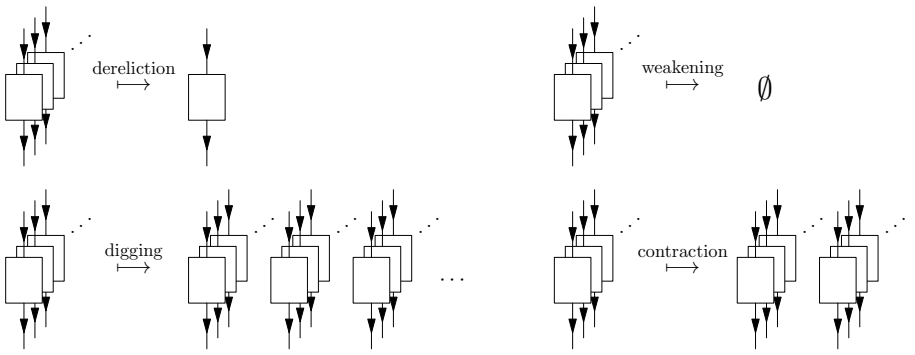


Figure 2.4: constructions induced by "retractions"

We denote the countable coproducts in **Set** by $\coprod_{i\in I} X_i \xleftarrow{\mathrm{inj}_i} X_i$. For each arrow $c_i \colon X_i \times A \to_T X_i \times B$, an arrow $\widetilde{c_i} \colon A \to_T (1 + \coprod_{i\in I} X_i) \times B$ is defined as

$$\widetilde{c_i} := ((\mathrm{inr}^*_{1,\coprod_{i\in I} X_i} \circ_T \mathrm{inj}_i^*) \otimes B) \circ_T c_i \circ_T (x_i^* \otimes A) \ .$$

The arrow $\widetilde{c_i}$ executes the initial transition of the T-transducer (X_i, c_i, x_i). The T-transducer $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i\in I}$ has a flesh initial state. When it receives an input for the first time it chooses which T-transducer to run by using the algebraic operation α. After an i-th T-transducer (X_i, c_i, x_i) is chosen the T-transducer $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i\in I}$ remembers its choice by using internal states and keeps behaving as (X_i, c_i, x_i).

2.5 Behavioral Equivalence and Resumptions

As pointed out in Section 2.4, we need an appropriate equivalence relation to form a GoI situation based on T-transducers. Here we use the *behavioral equivalence*.

Definition 2.5.1 (homomorphism)**.** For T-transducers $(X, c, x), (Y, d, y) \colon A \nrightarrow B$, an arrow $h \colon X \to Y$ in **Set** is a *homomorphism from* (X, c, x) *to* (Y, d, y) if it holds that

$$(h^* \otimes B) \circ_T c = d \circ_T (h^* \otimes A) \ .$$

Definition 2.5.2 (behavioral equivalence)**.** Two T-transducers $(X, c, x) \colon A \nrightarrow B$ and $(Y, d, y) \colon A \nrightarrow B$ are *behavioral equivalent* if there exists a T-transducer $(Z, e, z) \colon A \nrightarrow B$ such that there are homomorphisms from (X, c, x) to (Z, e, z) and from (Y, d, y) to (Z, e, z).

This relation $\simeq^T$ is indeed an equivalence relation on T-transducers: reflexivity and symmetry are obvious and transitivity can be confirmed by using pushouts in **Set**.

We write $(X, c, x) \xrightarrow{h} (Z, e, z)$ if h is a homomorphism from (X, c, x) to (Z, e, z) and $(X, c, x) \simeq^T_{A,B} (Y, d, y)$ if (X, c, x) and (Y, d, y) are behavioral equivalent. Since an identity arrow $\mathrm{id}_X \colon X \to X$ is a homomorphism from (X, c, x) to (X, c, x), the existence of a homomorphism from (X, c, x) to (Y, d, y) implies the behavioral equivalence $(X, c, x) \simeq^T_{A,B} (Y, d, y)$.

We write $[(X, c, x)] \colon A \nrightarrow B$ for the $\simeq^T$-equivalence class of a T-transducer $(X, c, x) \colon A \nrightarrow B$ and call this *resumption*. A category of resumptions can be defined as below.

- Objects are sets.
- An arrow from A to B is a resumption $[(X, c, x)]\colon A \nrightarrow B$.
- The identity arrow of A is $J\eta_A\colon A \nrightarrow A$.
- The composition is given by $\circ$.

We write $\mathbf{Res}(T)$ for this category. In the remainder of this section it is proved that all notions on T-transducers introduced in Section 2.4 can be extended to resumptions and that a GoI situation can be obtained.

Proposition 2.5.3. *Let α be an I-ary algebraic operation on T. The operator $\overline{\alpha}$ defined in Section 2.4 is well-defined modulo the behavioral equivalence.*

Proof. Let $\{(X_i, c_i, x_i)\colon A \nrightarrow B\}_{i\in I}$ and $\{(X'_i, c'_i, x'_i)\colon A \nrightarrow B\}_{i\in I}$ be families of T-transducers. We show that if there exist homomorphisms $\{(X_i, c_i, x_i) \xrightarrow{h_i} (X'_i, c'_i, x'_i)\}_{i\in I}$ then an arrow $1 + \coprod_{i\in I} h_i\colon 1 + \coprod_{i\in I} X_i \to 1 + \coprod_{i\in I} X'_i$ is a homomorphism from $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i\in I}$ to $\overline{\alpha}_{A,B}\{(X'_i, c'_i, x'_i)\}_{i\in I}$ i.e. the following diagram in $\mathbf{Set}_T$ commutes.

$$\begin{array}{ccc} (1 + \coprod_{i\in I} X_i) \times A & \xrightarrow{e} & (1 + \coprod_{i\in I} X_i) \times B \\ \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes A} & & \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes B} \\ (1 + \coprod_{i\in I} X'_i) \times A & \xrightarrow{e'} & (1 + \coprod_{i\in I} X'_i) \times B \end{array}$$

We write e for the transition function of $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i\in I}$ and e' for that of $\overline{\alpha}_{A,B}\{(X'_i, c'_i, x'_i)\}_{i\in I}$.

Since $(1 + \coprod_{i\in I} X_i) \times A$ is isomorphic to $1 \times A + \coprod_{i\in I}(X_i \times A)$, it suffices to show that the following two diagrams in $\mathbf{Set}_T$ commutes for all i in I.

$$\begin{array}{ccccc} 1 \times A & \xrightarrow{\mathrm{inl}^*_{1,\coprod_{i\in I} X_i}\otimes A} & (1 + \coprod_{i\in I} X_i) \times A & \xrightarrow{e} & (1 + \coprod_{i\in I} X_i) \times B \\ & & \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes A} & & \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes B} \\ & & (1 + \coprod_{i\in I} X'_i) \times A & \xrightarrow{e'} & (1 + \coprod_{i\in I} X'_i) \times B \end{array}$$

$$\begin{array}{ccccc} X_i \times A & & & & \\ \downarrow{\scriptstyle \mathrm{inj}^*_i\otimes A} & & & & \\ \coprod_{i\in I} X_i \times A & \xrightarrow{\mathrm{inr}^*_{1,\coprod_{i\in I} X_i}\otimes A} & (1 + \coprod_{i\in I} X_i) \times A & \xrightarrow{e} & (1 + \coprod_{i\in I} X_i) \times B \\ & & \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes A} & & \downarrow{\scriptstyle (1+\coprod_{i\in I} h_i)^*\otimes B} \\ & & (1 + \coprod_{i\in I} X'_i) \times A & \xrightarrow{e'} & (1 + \coprod_{i\in I} X'_i) \times B \end{array}$$

The first diagram commutes because the following equation holds

$$((1 + \coprod_{i\in I} h_i)^* \otimes B) \circ_T \widetilde{c_i} = \widetilde{c'_i} \circ_T \mathrm{id}^*_A$$

for all i in I and leads to the equation

$$((1 + \coprod_{i\in I} h_i)^* \otimes B) \circ_T \alpha_{A,(1+\coprod_{i\in I} X_i)\times B}\{\widetilde{c_i}\}_{i\in I} = \alpha_{A,(1+\coprod_{i\in I} X'_i)\times B}\{\widetilde{c'_i}\}_{i\in I}$$

by naturality of the algebraic operation α. It is straightforward to confirm that the second diagram commutes. □

Proposition 2.5.4. *The category* $(\mathbf{Res}(T), \emptyset, \boxplus, \mathrm{Tr})$ *forms a traced symmetric monoidal category.*

Proof. First, $\mathbf{Res}(T)$ indeed forms a category. The composition $\circ$ is well-defined modulo the behavioral equivalence since it holds that

$$\begin{aligned}&(X,c,x) \xrightarrow{h} (X',c',x') \wedge (Y,d,y) \xrightarrow{k} (Y',d',y')\\ &\quad\Longrightarrow (Y,d,y)\circ(X,c,x) \xrightarrow{h\times k} (Y',d',y')\circ(X',c',x')\ .\end{aligned}$$

The identity is neutral for the composition because there are homomorphisms

$$\begin{aligned}(X,c,x)\circ(1,\eta_A,\mathrm{id}_1) &\xrightarrow{\mathrm{id}_X} (X,c,x)\\ (1,\eta_B,\mathrm{id}_1)\circ(X,c,x) &\xrightarrow{\mathrm{id}_X} (X,c,x)\end{aligned}$$

for each T-transducer $(X,c,x)\colon A \nrightarrow B$.

Second, $(\mathbf{Res}(T), \emptyset, \boxplus)$ gives a symmetric monoidal structure. The monoidal product $\boxplus$ is well-defined modulo the behavioral equivalence since it holds that

$$\begin{aligned}&(X,c,x) \xrightarrow{h} (X',c',x') \wedge (Y,d,y) \xrightarrow{k} (Y',d',y')\\ &\quad\Longrightarrow (X,c,x)\boxplus(Y,d,y) \xrightarrow{h\times k} (X',c',x')\boxplus(Y',d',y')\ .\end{aligned}$$

The T-transducer $(1,\eta_\emptyset,\mathrm{id}_1)\colon \emptyset \nrightarrow \emptyset$ is indeed the unit of the monoidal product because there are homomorphisms

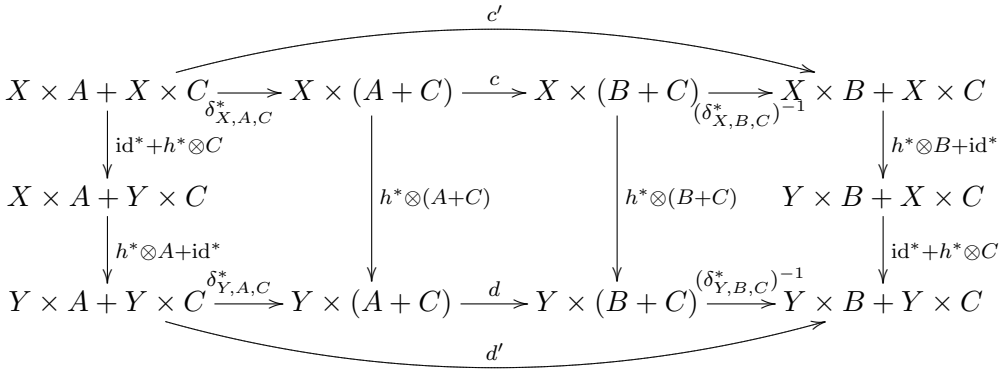
$$\begin{aligned}(X,c,x)\boxplus(1,\eta_\emptyset,\mathrm{id}_1) &\xrightarrow{\mathrm{id}_X} (X,c,x)\\ (1,\eta_\emptyset,\mathrm{id}_1)\boxplus(X,c,x) &\xrightarrow{\mathrm{id}_X} (X,c,x)\end{aligned}$$

for each T-transducer $(X,c,x)\colon A \nrightarrow B$. The symmetric monoidal structure of $(\mathbf{Res}(T), \emptyset, \boxplus)$ is lifted from that of $(\mathbf{Set}, \emptyset, +)$ by the constructor J_0.

Finally, Tr is a trace on the symmetric monoidal category $(\mathbf{Res}(T), \emptyset, \boxplus)$. For all T-transducers $(X,c,x)\colon A+C \nrightarrow B+C$ and $(Y,d,y)\colon A+C \nrightarrow B+C$, assume that $h\colon X \to Y$ is a homomorphism from (X,c,x) to (Y,d,y). We define arrows c' and d' in $\mathbf{Set}_T$ as

$$\begin{aligned}c' &:= (\delta^*_{X,B,C})^{-1} \circ_T c \circ_T \delta^*_{X,A,C}\\ d' &:= (\delta^*_{Y,B,C})^{-1} \circ_T d \circ_T \delta^*_{Y,A,C}\ .\end{aligned}$$

Since the following diagram in $\mathbf{Set}_T$ commutes,



the following equation holds by the restricted uniformity of tr.

$$\mathrm{tr}^{X\times C}_{X\times A, Y\times B}((h^*\otimes B + \mathrm{id}^*_{X\times C})\circ_T c') = \mathrm{tr}^{Y\times C}_{X\times A, Y\times B}(d'\circ_T (h^*\otimes A + \mathrm{id}^*_{Y\times C}))$$

By naturality of tr, it holds that

$$(h^* \otimes B) \circ_T \mathrm{tr}^{X\times C}_{X\times A, X\times B}(c') = \mathrm{tr}^{Y\times C}_{Y\times A, Y\times B}(d') \circ_T (h^* \otimes A)$$

and h is a homomorphism from $\mathrm{Tr}^C_{A,B}(X,c,x)$ to $\mathrm{Tr}^C_{A,B}(Y,d,y)$. It is proved in [6] that Tr satisfies the axioms in Definition 2.1.1 and is indeed a trace. □

Proposition 2.5.5. *The triple* $(\mathbf{Res}(T), F, \mathbb{N})$ *forms a GoI situation.*

Proof. First, F is a symmetric monoidal functor on $\mathbf{Res}(T)$. The composition $\circ$ is preserved by F because an isomorphism $(X \times Y)^{\mathbb{N}} \xrightarrow{\cong} X^{\mathbb{N}} \times Y^{\mathbb{N}}$ in **Set** is an homomorphism from $F((Y,d,y) \circ (X,d,x))$ to $F(Y,d,y) \circ F(X,c,x)$ for all T-transducers $(X,d,x)\colon A \nrightarrow B$ and $(Y,d,y)\colon B \nrightarrow C$. The key is the following commutative diagram in **Set**:

$$\begin{array}{ccc} (X\times Y)^{\mathbb{N}} & \xrightarrow{\cong} & X^{\mathbb{N}} \times Y^{\mathbb{N}} \\ \downarrow{\scriptstyle \omega_{n,X\times Y}} & & \downarrow{\scriptstyle \omega_{n,X}\times\omega_{n,Y}} \\ (X\times Y)^{\mathbb{N}} \times X \times Y & \xrightarrow{\cong} & X^{\mathbb{N}} \times X \times Y^{\mathbb{N}} \times Y \end{array}$$

where n is an arbitrary element of $\mathbb{N}$. The coherence isomorphism $FA + FB \xrightarrow{m} F(A+B)$ is lifted from an isomorphism $FA + FB \to F(A+B)$ in **Set** by the constructor J_0.

Second, F is traced i.e. F satisfies the following equation

$$\mathrm{Tr}^{FC}_{FA,FB}(J_0 m^{-1} \circ F(X,c,x) \circ J_0 m) = F(\mathrm{Tr}^C_{A,B}(X,c,x)) \tag{2.1}$$

for all T-transducers $(X,c,x)\colon A + C \nrightarrow B + C$. We write $\mathrm{tr}^{X\times C}_{X\times A, X\times B}(c')$ and d' respectively for the transition function of T-transducers $\mathrm{Tr}^C_{A,B}(X,c,x)$ and $\mathrm{Tr}^{FC}_{FA,FB}(J_0 m^{-1} \circ F(X,c,x) \circ J_0 m)$. By the restricted uniformity and naturality of tr, it holds that

$$\begin{aligned} &\mathrm{tr}^{X^{\mathbb{N}}\times FC}_{X^{\mathbb{N}}\times FA, X^{\mathbb{N}}\times FB}(d') \circ_T (X^{\mathbb{N}} \otimes \kappa_n^* \otimes A) \\ &= ((\omega^*_{n,X})^{-1} \otimes \kappa_n^* \otimes B) \circ_T \mathrm{tr}^{X^{\mathbb{N}}\times X\times C}_{X^{\mathbb{N}}\times X\times A, X^{\mathbb{N}}\times X\times B}((\delta^*_{X^{\mathbb{N}}, X\times B, X\times C})^{-1} \\ &\quad \circ_T (X^{\mathbb{N}} \otimes c') \circ_T \delta^*_{X^{\mathbb{N}}, X\times A, X\times C}) \circ_T (\omega^*_{n,X} \otimes A) \\ &= ((\omega^*_{n,X})^{-1} \otimes \kappa_n^* \otimes B) \circ_T (X^{\mathbb{N}} \otimes \mathrm{tr}^{X\times C}_{X\times A, X\times B}(c')) \circ_T (\omega^*_{n,X} \otimes A) \end{aligned}$$

for each n in $\mathbb{N}$. This equation indicates that (2.1) holds: the two T-transducers $\mathrm{Tr}^{FC}_{FA,FB}(J_0 m^{-1} \circ F(X,c,x) \circ J_0 m)$ and $F(\mathrm{Tr}^C_{A,B}(X,c,x))$ are exactly equal.

Finally, the pairs of resumptions given in Section 2.4 are retractions and satisfy the properties described in Definition 2.1.3. It is straightforward to confirm this. □

2.6 Resumption-Based Categorical GoI for Effects

In this section we briefly observe how can resumptions benefit us via categorical GoI. From the GoI situation $(\mathbf{Res}(T), \mathbb{N}, F)$ based on resumptions, a linear combinatory algebra $(\mathbf{Res}(T)(\mathbb{N},\mathbb{N}), \cdot, !)$ can be extracted via categorical GoI (Proposition 2.1.5). The operators $\cdot$ and $!$ are given by

$$\begin{aligned} f \cdot g &:= \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((J_0 \eta_{\mathbb{N}} \boxplus g) \circ J_0 \psi \circ f \circ J_0 \phi) \\ !f &:= J_0 u \circ F(f) \circ J_0 v \end{aligned}$$

for all f and g in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$. By Proposition 2.5.3 this linear combinatory algebra $(\mathbf{Res}(T)(\mathbb{N},\mathbb{N}),\cdot,!)$ has an I-ary operator $\overline{\alpha}$ for each I-ary algebraic operation α on T. This operator $\overline{\alpha}$ satisfies certain properties described in the following theorem.

Theorem 2.6.1. *Let α be an I-ary algebraic operation on T. The operator $\overline{\alpha}$ is natural and the trace* Tr *distributes over $\overline{\alpha}$ modulo the behavioral equivalence:*

- *For each arrow $h\colon A' \to A$ in* **Set***, family of T-transducers $\{(Y_i,d_i,y_i)\colon A \nrightarrow B\}$ and T-transducer $(X,c,x)\colon B \nrightarrow B'$, it holds that*

$$(X,c,x)\circ\overline{\alpha}_{A,B}\{(Y_i,d_i,y_i)\}_{i\in I}\circ J_0h \simeq^T_{A,B} \overline{\alpha}_{A',B'}\{(X,c,x)\circ(Y_i,d_i,y_i)\circ J_0h\}_{i\in I} \ .$$

- *For each family of T-transducers $(X_i,c_i,x_i)\colon A+C \nrightarrow B+C$ it holds that*

$$\mathrm{Tr}^C_{A,B}(\overline{\alpha}_{A+C,B+C}\{(X_i,c_i,x_i)\}_{i\in I}) \simeq^T_{A,B} \overline{\alpha}_{A,B}\{\mathrm{Tr}^C_{A,B}(X_i,c_i,x_i)\}_{i\in I} \ .$$

Proof. The first behavioral equivalence can be confirmed by showing that the arrow

$$\begin{array}{ccc} 1\times(1+\coprod_{i\in I}Y_i)\times X & & 1+\coprod_{i\in I}(1\times Y_i\times X) \\ \cong\downarrow & & \downarrow\cong \\ X+\coprod_{i\in I}(Y_i\times X) & \xleftarrow{x+\coprod_{i\in I}\mathrm{id}_{Y_i\times X}} & 1+\coprod_{i\in I}(Y_i\times X) \end{array}$$

is an homomorphism between the two T-transducers in the left-hand side and the right-hand side of the first equation. This can be proved by using naturality of the algebraic operation α.

We confirm the second behavioral equivalence by using string diagrams in $\mathbf{Res}(T)$. The proof is shown in Figure 2.5. The unit where two strings are merged represents the codiagonal map. Strings with black circles as the initial points represent arrows from the initial object $\emptyset$.

□

As a consequence of this theorem, the operator $\overline{\alpha}$ satisfies the following behavioral equivalence

$$\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\}_{i\in I}\cdot(Y,d,y) \simeq^T_{\mathbb{N},\mathbb{N}} \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\cdot(Y,d,y)\}_{i\in I} \tag{2.2}$$

for all T-transducers $\{(X_i,c_i,x_i)\}_{i\in I}\colon \mathbb{N} \nrightarrow \mathbb{N}$ and $(Y,d,y)\colon \mathbb{N} \nrightarrow \mathbb{N}$. In detail it holds that

$$\begin{aligned} &\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\}_{i\in I}\cdot(Y,d,y) \\ &= \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((J_0\eta_{\mathbb{N}}\boxplus(Y,d,y))\circ J_0\psi\circ\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\}_{i\in I}\circ J_0\phi) \\ &\simeq_{\mathbb{N},\mathbb{N}} \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}(\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(J_0\eta_{\mathbb{N}}\boxplus(Y,d,y))\circ J_0\psi\circ(X_i,c_i,x_i)\circ J_0\phi\}_{i\in I}) \\ &\simeq_{\mathbb{N},\mathbb{N}} \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((J_0\eta_{\mathbb{N}}\boxplus(Y,d,y))\circ J_0\psi\circ(X_i,c_i,x_i)\circ J_0\phi)\}_{i\in I} \\ &\simeq^T_{\mathbb{N},\mathbb{N}} \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\cdot(Y,d,y)\}_{i\in I} \ . \end{aligned}$$

This behavioral equivalence indicates that the problem described in Section 1 can be solved by resumptions. It is known that a linear combinatory algebra can represent abstractions and applications of linear λ-calculus via a term τ on it. We recall this later in Chapter 3. On the linear combinatory algebra $(\mathbf{Res}(T)(\mathbb{N},\mathbb{N}),\cdot,!)$ we can extend terms with the operator $\overline{\alpha}$ and define a term

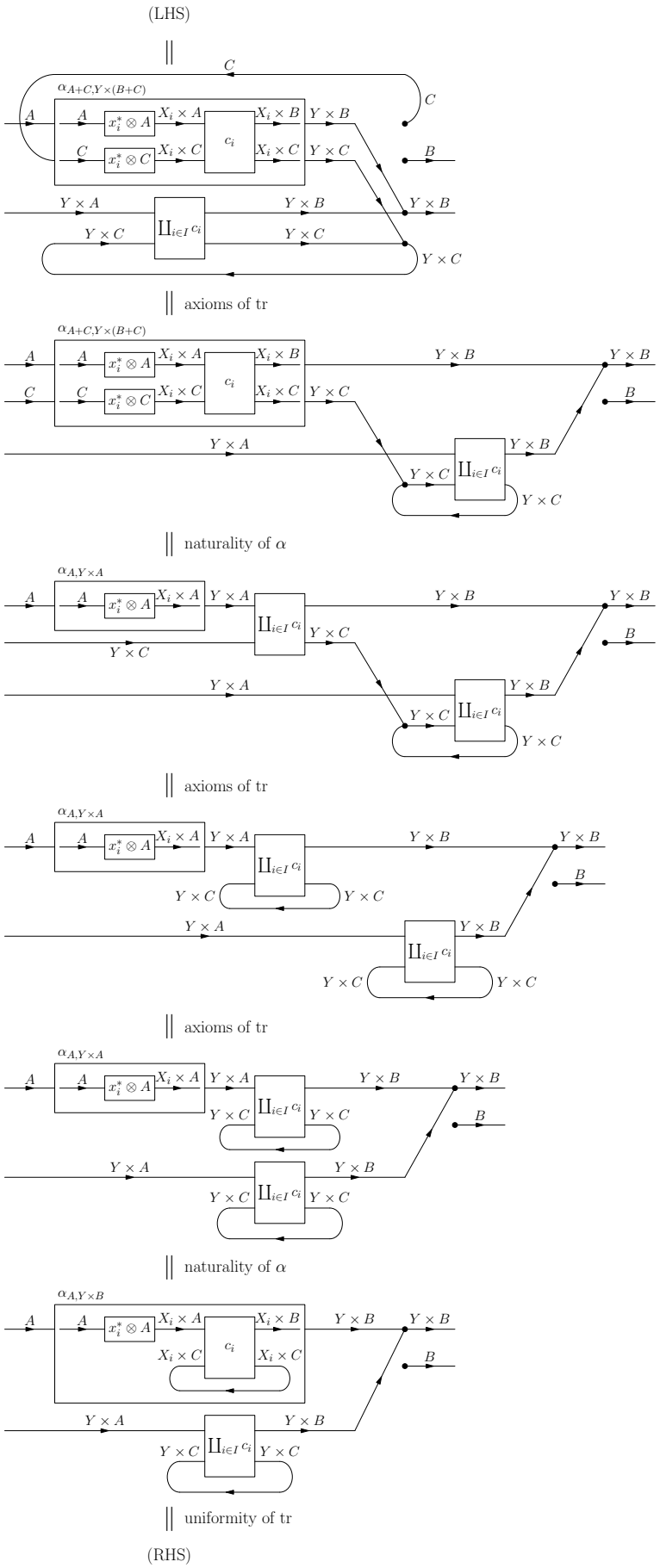


Figure 2.5: proof of the second behavioral equivalence in Theorem 2.6.1

$\overline{\alpha}\{\tau_i\}_{i\in I}$. This term $\overline{\alpha}\{\tau_i\}_{i\in I}$ can be seen as an representation of algebraic effects in linear λ-calculus. The behavioral equivalence (2.2) induces the equation

$$\overline{\alpha}\{e_i\}_{i\in I} \cdot d = \overline{\alpha}\{e_i \cdot d\}_{i\in I}$$

of terms $\{e_i\}_{i\in I}$ and d on $(\mathbf{Res}(T)(\mathbb{N},\mathbb{N}), \cdot, !)$. This equation can be seen as the representation of the equation in linear λ-calculus, for example

$$(M_1 \sqcup M_2)N = (M_1 N) \sqcup (M_2 N)$$

where $\sqcup$ is an effect described in Chapter 1.

Chapter 3

Model of Linear λ-Calculus

3.1 Linear λ-Model

Linear λ-calculus [16] is an extended λ-calculus with the ! modality that tracks copying of data. It has two abstractions: linear abstractions $\lambda x.\, M$ and non-linear abstractions $\lambda !x.\, M$. The argument x in linear abstractions $\lambda x.\, M$ must be used once without copying, where it may be copied or discarded in non-linear abstractions $\lambda !x.\, M$. The operator ! represents copying of terms.

Precisely, terms of linear λ-calculus is defined by the following rules

$$\frac{}{\vec{x} \vdash x_i}\ (\text{var}) \qquad \frac{\vec{x} \vdash M}{\vec{x} \vdash !\, M}\ (\text{bang})$$

$$\frac{\vec{x}, x_{n+1} \vdash M}{\vec{x} \vdash \lambda x_{n+1}.\, M}\ (\text{abs})\ \ (\text{if } x_{n+1} \text{ is linear in } M)$$

$$\frac{\vec{x}, x_{n+1} \vdash M}{\vec{x} \vdash \lambda !x_{n+1}.\, M}\ (\text{abs!}) \qquad \frac{\vec{x} \vdash M \quad \vec{x} \vdash N}{\vec{x} \vdash MN}\ (\text{app})$$

where $\vec{x}$ denotes $x_1, \ldots, x_n$ and each x_i is in a set of variables **Var**. We write $M \equiv N$ if M and N are syntactically equal. A variable x is *linear* in a term M if x is free and occurs exactly once in M except in the scope of the operator !. We denote sets of free variables and linear variables of a term M by $\mathrm{FV}(M)$ and $\mathrm{LV}(M)$ respectively. For a variable x and terms M and N, the substitution $M[N/x]$ is defined inductively as

$$\begin{aligned}
x[N/x] &= N \\
y[N/x] &= y \ \ (\text{if } x \not\equiv y) \\
(\lambda y.\, M)[N/x] &= \lambda y.\, M[N/x] \ \ (\text{if } x \not\equiv y \text{ and } y \notin \mathrm{FV}(N)) \\
(\lambda !y.\, M)[N/x] &= \lambda !y.\, M[N/x] \ \ (\text{if } x \not\equiv y \text{ and } y \notin \mathrm{FV}(N)) \\
(M_1 M_2)[N/x] &= (M_1[N/x])(M_2[N/x]) \\
(!\, M)[N/x] &= !(M[N/x]) \ .
\end{aligned}$$

The β-reduction rules for linear λ-calculus are defined as

$$\begin{aligned}
(\lambda x.\, M)N &\to_\beta M[N/x] \\
(\lambda !x.\, M)(!\, N) &\to_\beta M[N/x] \ .
\end{aligned}$$

The application to the non-linear abstraction $(\lambda !x.\, M)N'$ is reduced only if the applied term N' is copied by the operator !.

We extend a model of untyped λ-calculus, namely a *λ-model* described in [18] and define the *linear λ-model* that models untyped linear λ-calculus. One important feature of λ-calculus is that abstractions of λ-terms $\lambda x.\, M$ — that

behave as "functions" — are also treated as λ-terms. Models of λ-calculus need to express this feature, in other words they need the correspondence of "functions" and "elements". A λ-model is a model of untyped λ-calculus based on a cartesian closed category $(\mathbb{C}, 1, \times, \Rightarrow)$. It utilizes a retraction $m' : (D \Rightarrow D) \lhd D : e'$ in $\mathbb{C}$ on a object D (called a *reflexive object*) in $\mathbb{C}$ that expresses the correspondence of functions and elements, and gives sound denotational semantics of untyped λ-calculus [18, Proposition 3.1.1].

We define a linear λ-model as the linear version of a λ-model. A linear λ-model is also based on a cartesian closed category $(\mathbb{C}, 1, \times, \Rightarrow)$. The category **Set** of sets and functions is an example of a cartesian closed category.

Definition 3.1.1 (linear λ-model)**.** Let $(\mathbb{C}, 1, \times, \Rightarrow)$ be a cartesian closed category. A data $(D, e, m_1, m_2, !, [\![-]\!])$ is a *linear λ-model* if it satisfies the following properties.

- The data consists of an object D in $\mathbb{C}$ and arrows $e\colon D \to (D \Rightarrow D)$, $m_1, m_2\colon (D \Rightarrow D) \to D$, $!\colon D \to D$ and $[\![\vec{x} \vdash M]\!]\colon D^n \to D$ in $\mathbb{C}$.
- The arrows in the data satisfies following equations

$$[\![\vec{x} \vdash x_i]\!] = \pi_i \tag{3.1}$$
$$[\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!] = m_1 \circ [\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge \ \ (\text{if } x_{n+1} \in \mathrm{LV}(M)) \tag{3.2}$$
$$[\![\vec{x} \vdash \lambda !x_{n+1}.\, M]\!] = m_2 \circ [\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge \tag{3.3}$$
$$[\![\vec{x} \vdash M_1 M_2]\!] = \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x} \vdash M_1]\!] \times [\![\vec{x} \vdash M_2]\!]) \circ \Delta_{D^n} \tag{3.4}$$
$$[\![\vec{x} \vdash !\, M]\!] = ! \circ [\![\vec{x} \vdash M]\!] \tag{3.5}$$
$$[\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge = e \circ m_1 \circ [\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge \ \ (\text{if } x_{n+1} \in \mathrm{LV}(M)) \tag{3.6}$$
$$\begin{aligned} &\mathrm{ev} \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge \times \mathrm{id}_D) \\ &\quad = \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ (m_2 \times \mathrm{id}_D) \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge \times \mathrm{id}_D) \circ (\mathrm{id}_{D^n} \times !) \end{aligned} \tag{3.7}$$

  where $\Delta_X\colon X \to X \times X$ is the diagonal arrow in $\mathbb{C}$, $(-)^\wedge\colon \mathbb{C}(X \times D, Y) \to \mathbb{C}(X, D \Rightarrow Y)$ is the bijective correspondence of the adjunction $(-) \times D \dashv (D \Rightarrow (-))$ and $\mathrm{ev}\colon (D \Rightarrow Y) \times D \to Y$ is the counit of that adjunction.

A linear λ-model utilizes a retraction $m_1 : (D \Rightarrow D) \lhd D : e$ that models linear abstractions, and an arrow $m_2\colon (D \Rightarrow D) \to D$ that models non-linear abstractions with the arrow e. The pair of arrows m_2 and e can be seen as a limited retraction: for any arrows $d\colon 1 \to (D \Rightarrow D)$ and $d'\colon 1 \to D$ it holds that $ev \circ (e \circ m_2 \circ d, !\, d') = ev \circ (d, !\, d')$. This corresponds to the β-reduction rule that indicates a term $(\lambda !x.\, M)N$ is not always reduced.

A linear λ-model gives sound denotational semantics of untyped linear λ-calculus: the linear version of [18, Proposition 3.1.1] holds. We give a detailed proof in the remainder of this section.

Lemma 3.1.2. *Let $(D, e, m_1, m_2, !, [\![-]\!])$ be a linear λ-model. For substitutions of linear λ-terms it holds that*

$$[\![\vec{x} \vdash M[N/x_{n+1}]]\!] = [\![\vec{x}, x_{n+1} \vdash M]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n}$$

Proof. By induction on M.

If $M \equiv x_{n+1}$,

$$\begin{aligned} [\![\vec{x} \vdash x_{n+1}[N/x_{n+1}]]\!] &= [\![\vec{x} \vdash N]\!] \\ &= \pi_{n+1} \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \\ &= [\![\vec{x}, x_{n+1} \vdash x_{n+1}]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ . \end{aligned}$$

If $M \equiv x_i$,

$$\begin{aligned}
[\![\vec{x} \vdash x_i[N/x_{n+1}]]\!] &= [\![\vec{x} \vdash x_i]\!] \\
&= \pi_i \\
&= \pi_i \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \\
&= [\![\vec{x}, x_{n+1} \vdash x_i]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ .
\end{aligned}$$

If $M \equiv \lambda x_{n+2}.\, M'$,

$$\begin{aligned}
&[\![\vec{x} \vdash (\lambda x_{n+2}.\, M')[N/x_{n+1}]]\!] \\
&= [\![\vec{x} \vdash \lambda x_{n+2}.\, M'[N/x_{n+1}]]\!] \\
&= m_1 \circ [\![\vec{x}, x_{n+2} \vdash M'[N/x_{n+1}]]\!]^\wedge \\
&= m_1 \circ ([\![\vec{x}, x_{n+2}, x_{n+1} \vdash M']\!] \\
&\qquad \circ (\mathrm{id}_{D^{n+1}} \times [\![\vec{x}, x_{n+2} \vdash N]\!]) \circ \Delta_{D^{n+1}})^\wedge \ \text{(induction hypothesis)} \\
&= m_1 \circ ([\![\vec{x}, x_{n+1}, x_{n+2} \vdash M']\!] \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!] \times \mathrm{id}_D) \circ (\Delta_{D^n} \times \mathrm{id}_D))^\wedge \ \text{(since } x_{n+2} \notin \mathrm{FV}(N)\text{)} \\
&= m_1 \circ ([\![\vec{x}, x_{n+1}, x_{n+2} \vdash M']\!])^\wedge \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by naturality of } (-)^\wedge\text{)} \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda x_{n+2}.\, M']\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ .
\end{aligned}$$

If $M \equiv \lambda! x_{n+2}.\, M'$,

$$\begin{aligned}
&[\![\vec{x} \vdash (\lambda! x_{n+2}.\, M')[N/x_{n+1}]]\!] \\
&= [\![\vec{x} \vdash \lambda! x_{n+2}.\, M'[N/x_{n+1}]]\!] \\
&= m_2 \circ [\![\vec{x}, x_{n+2} \vdash M'[N/x_{n+1}]]\!]^\wedge \\
&= m_2 \circ ([\![\vec{x}, x_{n+2}, x_{n+1} \vdash M']\!] \\
&\qquad \circ (\mathrm{id}_{D^{n+1}} \times [\![\vec{x}, x_{n+2} \vdash N]\!]) \circ \Delta_{D^{n+1}})^\wedge \ \text{(induction hypothesis)} \\
&= m_2 \circ ([\![\vec{x}, x_{n+1}, x_{n+2} \vdash M']\!] \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!] \times \mathrm{id}_D) \circ (\Delta_{D^n} \times \mathrm{id}_D))^\wedge \ \text{(since } x_{n+2} \notin \mathrm{FV}(N)\text{)} \\
&= m_2 \circ ([\![\vec{x}, x_{n+1}, x_{n+2} \vdash M']\!])^\wedge \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by naturality of } (-)^\wedge\text{)} \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda! x_{n+2}.\, M']\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ .
\end{aligned}$$

If $M \equiv M_1 M_2$,

$$\begin{aligned}
&[\![\vec{x} \vdash (M_1 M_2)[N/x_{n+1}]]\!] \\
&= [\![\vec{x} \vdash (M_1[N/x_{n+1}])(M_2[N/x_{n+1}])]\!] \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x} \vdash M_1[N/x_{n+1}]]\!] \times [\![\vec{x} \vdash M_2[N/x_{n+1}]]\!]) \circ \Delta_{D^n} \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x}, x_{n+1} \vdash M_1]\!] \times [\![\vec{x}, x_{n+2} \vdash M_1]\!]) \\
&\qquad \circ ((\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \times (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!])) \\
&\qquad \circ \Delta_{D^n \times D^n} \circ \Delta_{D^n} \ \text{(induction hypothesis)} \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x}, x_{n+1} \vdash M_1]\!] \times [\![\vec{x}, x_{n+2} \vdash M_1]\!]) \\
&\qquad \circ \Delta_{D^{n+1}} \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \\
&= [\![\vec{x}, x_{n+1} \vdash M_1 M_2]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ .
\end{aligned}$$

If $M \equiv !\,M'$,

$$\begin{aligned}
&[\![\vec{x} \vdash (!\,M')[N/x_{n+1}]]\!] \\
&= [\![\vec{x} \vdash !(M'[N/x_{n+1}])]\!] \\
&= ! \circ [\![\vec{x} \vdash M'[N/x_{n+1}]]\!] \\
&= ! \circ [\![\vec{x}, x_{n+1} \vdash M']\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \quad \text{(induction hypothesis)} \\
&= [\![\vec{x}, x_{n+1} \vdash !\,M']\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ .
\end{aligned}$$

□

Proposition 3.1.3. *For an arbitrary set D,* (I) *and* (II) *below are equivalent.*

(I) *A data $(D, e, m_1, m_2, !, [\![-]\!])$ is a linear λ-model.*

(II) *A data $(D, \cdot, !, [\![-]\!])$ consisting of functions $\cdot\colon D \times D \to D$, $!\colon D \to D$ and $[\![\vec{x} \vdash M]\!]\colon D^n \to D$ satisfies the following properties for every $\vec{d}$ in D^n:*

$$[\![\vec{x} \vdash x_i]\!](\vec{d}) = d_i \tag{3.8}$$

$$\begin{aligned}
&(x \in \mathrm{LV}(M) \wedge \forall d_{n+1} \in D.\, [\![\vec{x}, x_{n+1} \vdash M_1]\!](\vec{d}, d_{n+1}) = [\![\vec{x}, x_{n+1} \vdash M_2]\!](\vec{d}, d_{n+1})) \\
&\quad \Longrightarrow [\![\vec{x} \vdash \lambda x_{n+1}.\, M_1]\!](\vec{d}) = [\![\vec{x} \vdash \lambda x_{n+1}.\, M_2]\!](\vec{d})
\end{aligned} \tag{3.9}$$

$$\begin{aligned}
&\forall d_{n+1} \in D.\, [\![\vec{x}, x_{n+1} \vdash M_1]\!](\vec{d}, d_{n+1}) = [\![\vec{x}, x_{n+1} \vdash M_2]\!](\vec{d}, d_{n+1}) \\
&\quad \Longrightarrow [\![\vec{x} \vdash \lambda !x_{n+1}.\, M_1]\!](\vec{d}) = [\![\vec{x} \vdash \lambda !x_{n+1}.\, M_2]\!](\vec{d})
\end{aligned} \tag{3.10}$$

$$[\![\vec{x} \vdash M_1 M_2]\!](\vec{d}) = [\![\vec{x} \vdash M_1]\!](\vec{d}) \cdot [\![\vec{x} \vdash M_2]\!](\vec{d}) \tag{3.11}$$

$$[\![\vec{x} \vdash !\,M]\!](\vec{d}) = ! [\![\vec{x} \vdash M]\!](\vec{d}) \tag{3.12}$$

for each i such that $x_i \in \mathrm{FV}(M)$,

$$d_i = d'_i \Longrightarrow [\![x_1, \ldots, x_n \vdash M]\!](d_1, \ldots, d_n) = [\![x_1, \ldots, x_m \vdash M]\!](d'_1, \ldots, d'_m) \tag{3.13}$$

$$M_1 =_\beta M_2 \Longrightarrow [\![\vec{x} \vdash M_1]\!](\vec{d}) = [\![\vec{x} \vdash M_2]\!](\vec{d}) \ . \tag{3.14}$$

Proof. (I) ⇒ (II) We define $\cdot\colon D \times D \to D$ as

$$\cdot := \mathrm{ev} \circ (e \times \mathrm{id}_D) \ .$$

(3.8), (3.11) and (3.12) are equivalent to (3.1), (3.4) and (3.5) respectively. (3.13) can be proved by induction on M.

(3.9) and (3.10) hold by (3.2) and (3.3) because it holds that

$$\begin{aligned}
&\forall d_{n+1} \in D.\, [\![\vec{x}, x_{n+1} \vdash M_1]\!](\vec{d}, d_{n+1}) = [\![\vec{x}, x_{n+1} \vdash M_2]\!](\vec{d}, d_{n+1}) \\
&\quad \Longleftrightarrow [\![\vec{x}, x_{n+1} \vdash M_1]\!]^\wedge = [\![\vec{x}, x_{n+1} \vdash M_2]\!]^\wedge \ .
\end{aligned}$$

We can inductively prove that (3.14) holds if it holds that

$$M_1 \to_\beta M_2 \Longrightarrow [\![\vec{x} \vdash M_1]\!](\vec{d}) = [\![\vec{x} \vdash M_2]\!](\vec{d}) \ .$$

By definition of β-reduction, it suffices to show that the following two equations hold.

$$\begin{aligned}
&[\![\vec{x} \vdash (\lambda x_{n+1}.\, M)N]\!] = [\![\vec{x} \vdash M[N/x_{n+1}]]\!] \ \text{(if } x_{n+1} \in \mathrm{LV}(M)) \\
&[\![\vec{x} \vdash (\lambda !x_{n+1}.\, M)\,!\,N]\!] = [\![\vec{x} \vdash M[N/x_{n+1}]]\!]
\end{aligned}$$

These equations indeed hold as below:

$$
\begin{aligned}
&[\![\vec{x} \vdash (\lambda x_{n+1}.\, M)N]\!] \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!] \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ (m_1 \times \mathrm{id}_D) \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge} \times \mathrm{id}_D) \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(since } x_{n+1} \in \mathrm{LV}(M)\text{)} \\
&= \mathrm{ev} \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge} \times \mathrm{id}_D) \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by (3.6))} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by definition of ev)} \\
&= [\![\vec{x} \vdash M[N/x_{n+1}]]\!] \ \text{(by Lemma 3.1.2)} \\
&[\![\vec{x} \vdash (\lambda !x_{n+1}.\, M)\, !\, N]\!] \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ ([\![\vec{x} \vdash \lambda !x_{n+1}.\, M]\!] \times [\![\vec{x} \vdash !\, N]\!]) \circ \Delta_{D^n} \\
&= \mathrm{ev} \circ (e \times \mathrm{id}_D) \circ (m_2 \times \mathrm{id}_D) \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge} \times \mathrm{id}_D) \circ (\mathrm{id}_{D^n} \times !) \\
&\qquad \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \\
&= \mathrm{ev} \circ ([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge} \times \mathrm{id}_D) \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by (3.7))} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!] \circ (\mathrm{id}_{D^n} \times [\![\vec{x} \vdash N]\!]) \circ \Delta_{D^n} \ \text{(by definition of ev)} \\
&= [\![\vec{x} \vdash M[N/x_{n+1}]]\!] \ . \ \text{(by Lemma 3.1.2)}
\end{aligned}
$$

(I) $\Leftarrow$ (II) First, we fix an arbitrary element a_0 of D and define e, m_1 and m_2 as

$$
\begin{aligned}
e(a)(d) &:= ad \ \text{(for all } d \text{ in } D\text{)} \\
m_1(f) &:= \begin{cases} [\![u \vdash \lambda x.\, ux]\!](a) \ \text{(if } \exists a \in D.\, \forall d \in D.\, f(d) = ad\text{)} \\ a_0 \ \text{(otherwise)} \end{cases} \\
m_2(f) &:= \begin{cases} [\![u \vdash \lambda !x.\, u\, !\, x]\!](a) \ \text{(if } \exists a \in D.\, \forall d \in D.\, f(d) = a\, !\, d\text{)} \\ a_0 \ \text{(otherwise)} \ . \end{cases}
\end{aligned}
$$

The well-definedness of m_1 and m_2 need to be confirmed. Assume that there exists two elements a and a' that satisfy $f(d') = ad' = a'd'$ for all d' in D. For all d in D, it holds that

$$
\begin{aligned}
[\![u, x \vdash ux]\!](a, d) &= [\![u, x \vdash u]\!](a, d) \cdot [\![u, x \vdash x]\!](a, d) \\
&= ad \\
&= a'd \\
&= [\![u, x \vdash u]\!](a', d) \cdot [\![u, x \vdash x]\!](a', d) \\
&= [\![u, x \vdash ux]\!](a', d) \ .
\end{aligned}
$$

Since x is a linear variable of the term x, the following equation holds by (3.9):

$$
[\![u \vdash \lambda x.\, ux]\!](a) = [\![u \vdash \lambda x.\, ux]\!](a') \ .
$$

Therefore m_1 is well-defined.

Assume that there exists two elements a and a' that satisfy $f(d') = a\, !\, d' = a'\, !\, d'$ for all d' in D. For all d in D, it holds that

$$
\begin{aligned}
[\![u, x \vdash u\, !\, x]\!](a, d) &= [\![u, x \vdash u]\!](a, d) \cdot [\![u, x \vdash !\, x]\!](a, d) \\
&= a\, !\, d \\
&= a'\, !\, d \\
&= [\![u, x \vdash u]\!](a', d) \cdot [\![u, x \vdash !\, x]\!](a', d) \\
&= [\![u, x \vdash u\, !\, x]\!](a', d) \ .
\end{aligned}
$$

The following equation holds by (3.10):

$$[\![u \vdash \lambda!x.\, u\,!\,x]\!](a) = [\![u \vdash \lambda!x.\, u\,!\,x]\!](a') \ .$$

Therefore m_2 is well-defined.

Second, we prove that $[\![-]\!]$ satisfies certain properties in Definition 3.1.1. Obviously, (3.1) is equivalent to (3.8), and (3.5) is equivalent to (3.12).

Assume that x_{n+1} is a linear variable of M. For every d_{n+1} in D, the equation below holds.

$$\begin{aligned}
&[\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})(d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!](\vec{d}, d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash (\lambda x_{n+1}.\, M)x_{n+1}]\!](\vec{d}, d_{n+1}) \ \text{(since } x_{n+1} \in \mathrm{LV}(M)) \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}, d_{n+1}) \cdot [\![\vec{x}, x_{n+1} \vdash x_{n+1}]\!](\vec{d}, d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}, d_{n+1}) \cdot d_{n+1} \\
&= [\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}) \cdot d_{n+1} \ \text{(since } x_{n+1} \notin \mathrm{FV}(\lambda x_{n+1}.\, M)) \qquad (3.15)
\end{aligned}$$

From this equation it follows that

$$\begin{aligned}
&m_1([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})) \\
&= [\![u \vdash \lambda x_{n+1}.\, ux_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d})) \ . \ \text{(by definition of } m_1) \qquad (3.16)
\end{aligned}$$

For every d_{n+1} in D, it also follows that

$$\begin{aligned}
&[\![u, x_{n+1} \vdash ux_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![u, x_{n+1} \vdash u]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&\qquad \cdot [\![u, x_{n+1} \vdash x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}) \cdot d_{n+1} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!](\vec{d}, d_{n+1}) \ . \ \text{(by (3.15))}
\end{aligned}$$

Therefore, by (3.9), (3.2) holds as below:

$$\begin{aligned}
m_1([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})) &= [\![u \vdash \lambda x_{n+1}.\, ux_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d})) \ \text{(by (3.16))} \\
&= [\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}) \ .
\end{aligned}$$

For every d_{n+1} in D, the equation below holds.

$$\begin{aligned}
&[\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})(d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!](\vec{d}, d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash (\lambda!x_{n+1}.\, M)\,!\,x_{n+1}]\!](\vec{d}, d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}, d_{n+1}) \cdot [\![\vec{x}, x_{n+1} \vdash !\,x_{n+1}]\!](\vec{d}, d_{n+1}) \\
&= [\![\vec{x}, x_{n+1} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}, d_{n+1}) \cdot !\,d_{n+1} \\
&= [\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}) \cdot !\,d_{n+1} \ \text{(since } x_{n+1} \notin \mathrm{FV}(\lambda!x_{n+1}.\, M)) \qquad (3.17)
\end{aligned}$$

From this equation it follows that

$$\begin{aligned}
&m_2([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})) \\
&= [\![u \vdash \lambda!x_{n+1}.\, u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d})) \ . \ \text{(by definition of } m_2) \qquad (3.18)
\end{aligned}$$

For every d_{n+1} in D, it also follows that

$$\begin{aligned}
&[\![u, x_{n+1} \vdash u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![u, x_{n+1} \vdash u]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&\qquad \cdot [\![u, x_{n+1} \vdash !\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}) \cdot !\, d_{n+1} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!](\vec{d}, d_{n+1}) \ . \ \text{(by (3.17))}
\end{aligned}$$

Therefore, by (3.10), (3.3) holds as below:

$$\begin{aligned}
&m_2([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})) \\
&= [\![u \vdash \lambda!x_{n+1}.\, u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d})) \ \text{(by (3.18))} \\
&= [\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}) \ .
\end{aligned}$$

(3.4) holds as below:

$$\begin{aligned}
[\![\vec{x} \vdash M_1 M_2]\!](\vec{d}) &= [\![\vec{x} \vdash M_1]\!](\vec{d}) \cdot [\![\vec{x} \vdash M_2]\!](\vec{d}) \\
&= e([\![\vec{x} \vdash M_1]\!](\vec{d}))([\![\vec{x} \vdash M_2]\!](\vec{d})) \ \text{(by definition of } e) \\
&= \mathrm{ev}(e([\![\vec{x} \vdash M_1]\!](\vec{d})), [\![\vec{x} \vdash M_2]\!](\vec{d})) \ .
\end{aligned}$$

Assume that x_{n+1} is a linear variable of M. For every d_{n+1} in D, (3.6) holds as below:

$$\begin{aligned}
&e(m_1([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})))(d_{n+1}) \\
&= e([\![u \vdash \lambda x_{n+1}.\, u x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d})))(d_{n+1}) \ \text{(by (3.16))} \\
&= [\![u \vdash \lambda x_{n+1}.\, u x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d})) \cdot d_{n+1} \ \text{(by definition of } e) \\
&= [\![u \vdash \lambda x_{n+1}.\, u x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d})) \cdot [\![x_{n+1} \vdash x_{n+1}]\!](d_{n+1}) \\
&= [\![u, x_{n+1} \vdash (\lambda x_{n+1}.\, u x_{n+1}) x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![u, x_{n+1} \vdash u x_{n+1}]\!]([\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![\vec{x} \vdash \lambda x_{n+1}.\, M]\!](\vec{d}) \cdot d_{n+1} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})(d_{n+1}) \ . \ \text{(by (3.15))}
\end{aligned}$$

For every d_{n+1} in D, (3.7) holds as below:

$$\begin{aligned}
&e(m_2([\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})))(!\, d_{n+1}) \\
&= e([\![u \vdash \lambda!x_{n+1}.\, u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d})))(!\, d_{n+1}) \ \text{(by (3.18))} \\
&= [\![u \vdash \lambda!x_{n+1}.\, u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d})) \cdot !\, d_{n+1} \ \text{(by definition of } e) \\
&= [\![u \vdash \lambda!x_{n+1}.\, u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d})) \cdot [\![x_{n+1} \vdash !\,x_{n+1}]\!](d_{n+1}) \\
&= [\![u, x_{n+1} \vdash (\lambda!x_{n+1}.\, u\,!\,x_{n+1})\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![u, x_{n+1} \vdash u\,!\,x_{n+1}]\!]([\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}), d_{n+1}) \\
&= [\![\vec{x} \vdash \lambda!x_{n+1}.\, M]\!](\vec{d}) \cdot !\, d_{n+1} \\
&= [\![\vec{x}, x_{n+1} \vdash M]\!]^{\wedge}(\vec{d})(d_{n+1}) \ . \ \text{(by (3.17))}
\end{aligned}$$

□

3.2 Linear Combinatory Algebra and Linear λ-Calculus

In this section we study the ability of linear combinatory algebras (see Definition 2.1.4) to model linear λ-calculus. As described in [16], linear combinatory algebras satisfy the property called *linear combinatory completeness*. We define a *term* on a linear combinatory algebra $(D, \cdot, !)$ by BNF as

$$\tau ::= u \in \mathbf{Var} \mid d \in D \mid \tau \cdot \tau \mid !\tau \ .$$

All elements of D can be seen as a term that includes no variables. A variable u is *linear* in a term τ if u occurs exactly once in τ except in the scope of the operator !.

Proposition 3.2.1 (linear combinatory completeness [16])**.** *Let $(D, \cdot, !)$ be a linear combinatory algebra. For every term e on D,* (I) *and* (II) *below holds:*

(I) *If u is a linear variable of e, then a term e^* on D exists and satisfies following properties:*

$$\mathrm{LV}(e^*) = \mathrm{LV}(e) \setminus \{u\}, \qquad \mathrm{FV}(e^*) = \mathrm{FV}(e) \setminus \{u\}, \qquad e = e^* u \ .$$

(II) *If u is a variable, then a term $e^{!*}$ on D exists and satisfies following properties:*

$$\mathrm{LV}(e^{!*}) = \mathrm{LV}(e) \setminus \{u\}, \qquad \mathrm{FV}(e^{!*}) = \mathrm{FV}(e) \setminus \{u\}, \qquad e = e^{!*} ! u \ .$$

It is known that linear combinatory completeness enables linear combinatory algebras to represent two abstractions of linear λ-calculus and β-reductions of closed linear λ-terms.

However, it is found that a linear combinatory algebra itself cannot form a linear λ-model i.e. give sound denotational semantics of linear λ-calculus, as stated in the following theorem. This theorem is the linear version of the Meyer-Scott theorem (see [18]) that is for a λ-model and SK-algebra. Here we show its proof in detail.

Theorem 3.2.2 (linear version of the Meyer-Scott theorem)**.** *For an arbitrary set D,* (I) *and* (II) *below are equivalent.*

(I) *A data $(D, e, m_1, m_2, !, [\![-]\!])$ is a linear λ-model.*

(II) *A triple $(D, \cdot, !)$ is a linear combinatory algebra with extra combinators* L *and* R *that satisfy the following properties: for every a, a', b in D,*

$$\begin{aligned} \mathsf{L}ab &= ab, & \forall d \in D.\, ad = a'd &\Longrightarrow \mathsf{L}a = \mathsf{L}a' \\ \mathsf{R}a\,!\,b &= a\,!\,b, & \forall d \in D.\, a\,!\,d = a'\,!\,d &\Longrightarrow \mathsf{R}a = \mathsf{R}a' \ . \end{aligned}$$

Proof. (I) $\Rightarrow$ (II) It suffices to prove that Proposition 3.1.3(II) $\Rightarrow$ (II), since (I) and Proposition 3.1.3(II) are equivalent by Proposition 3.1.3. We define combinators as

$$\begin{aligned} \mathsf{I} &:= [\![\vdash \lambda x.\, x]\!](*) & \mathsf{B} &:= [\![\vdash \lambda x.\, \lambda y.\, \lambda z.\, x(yz)]\!](*) \\ \mathsf{C} &:= [\![\vdash \lambda x.\, \lambda y.\, \lambda z.\, xzy]\!](*) & \mathsf{K} &:= [\![\vdash \lambda x.\, \lambda !y.\, x]\!](*) \\ \mathsf{W} &:= [\![\vdash \lambda x.\, \lambda !y.\, x\,!\,y\,!\,y]\!](*) & \mathsf{D} &:= [\![\vdash \lambda !x.\, x]\!](*) \\ \delta &:= [\![\vdash \lambda !x.\, !\,!\,x]\!](*) & \mathsf{F} &:= [\![\vdash \lambda !x.\, \lambda !y.\, !(xy)]\!](*) \\ \mathsf{L} &:= [\![\vdash \lambda x.\, \lambda y.\, xy]\!](*) & \mathsf{R} &:= [\![\vdash \lambda x.\, \lambda !y.\, x\,!\,y]\!](*) \end{aligned}$$

where $D^0 = \{*\}$. It is easy to show that combinators satisfy certain equations in Definition 2.1.4. The properties of combinators L and R need to be confirmed.

For all d in D, if two elements a and a' of D satisfy $ad = a'd$, then it holds that

$$\begin{aligned}
[\![x, y \vdash xy]\!](a, d) &= [\![x, y \vdash x]\!](a, d) \cdot [\![x, y \vdash y]\!](a, d) \\
&= ad \\
&= a'd \\
&= [\![x', y \vdash x']\!](a', d) \cdot [\![x', y \vdash y]\!](a', d) \\
&= [\![x', y \vdash x'y]\!](a', d) \ .
\end{aligned}$$

From this equation it follows that

$$\begin{aligned}
\mathsf{L}a &= [\![\vdash \lambda x.\, \lambda y.\, xy]\!](*) \cdot a \\
&= [\![\vdash \lambda x.\, \lambda y.\, xy]\!](*) \cdot [\![x \vdash x]\!](a) \\
&= [\![x \vdash (\lambda x.\, \lambda y.\, xy)x]\!](a) \\
&= [\![x \vdash \lambda y.\, xy]\!](a) \\
&= [\![x' \vdash \lambda y.\, x'y]\!](a') \ \text{(by (3.9))} \\
&= [\![x' \vdash (\lambda x'.\, \lambda y.\, x'y)x']\!](a') \\
&= [\![\vdash \lambda x'.\, \lambda y.\, x'y]\!](*) \cdot [\![x' \vdash x']\!](a') \\
&= [\![\vdash \lambda x'.\, \lambda y.\, x'y]\!](*) \cdot a' \\
&= \mathsf{L}a' \ .
\end{aligned}$$

For all d in D, if two elements a and a' of D satisfy $a\,!\,d = a'\,!\,d$, then it holds that

$$\begin{aligned}
[\![x, y \vdash x\,!\,y]\!](a, d) &= [\![x, y \vdash x]\!](a, d) \cdot [\![x, y \vdash\,!\,y]\!](a, d) \\
&= a\,!\,d \\
&= a'\,!\,d \\
&= [\![x', y \vdash x']\!](a', d) \cdot [\![x', y \vdash\,!\,y]\!](a', d) \\
&= [\![x', y \vdash x'\,!\,y]\!](a', d) \ .
\end{aligned}$$

From this equation it follows that

$$\begin{aligned}
\mathsf{R}a &= [\![\vdash \lambda x.\, \lambda !y.\, x\,!\,y]\!](*) \cdot a \\
&= [\![\vdash \lambda x.\, \lambda !y.\, x\,!\,y]\!](*) \cdot [\![x \vdash x]\!](a) \\
&= [\![x \vdash (\lambda x.\, \lambda !y.\, x\,!\,y)x]\!](a) \\
&= [\![x \vdash \lambda !y.\, x\,!\,y]\!](a) \\
&= [\![x' \vdash \lambda !y.\, x'\,!\,y]\!](a') \ \text{(by (3.10))} \\
&= [\![x' \vdash (\lambda x'.\, \lambda !y.\, x'\,!\,y)x']\!](a') \\
&= [\![\vdash \lambda x'.\, \lambda !y.\, x'\,!\,y]\!](*) \cdot [\![x' \vdash x']\!](a') \\
&= [\![\vdash \lambda x'.\, \lambda !y.\, x'\,!\,y]\!](*) \cdot a' \\
&= \mathsf{R}a' \ .
\end{aligned}$$

(I) $\Leftarrow$ (II) First, we fix an arbitrary element a_0 of D and define e, m_1 and m_2

as

$$e(a)(d) := ad \ \text{(for every } d \text{ in } D)$$
$$m_1(f) := \begin{cases} \mathsf{L}a \ (\text{if } \exists a \in D.\, \forall d \in D.\, f(d) = ad) \\ a_0 \ (\text{otherwise}) \end{cases}$$
$$m_2(f) := \begin{cases} \mathsf{R}a \ (\text{if } \exists a \in D.\, \forall d \in D.\, f(d) = a\,!\,d) \\ a_0 \ (\text{otherwise}) \ . \end{cases}$$

The well-definedness of m_1 and m_2 need to be confirmed. For all d in D, assume that there exists two elements a and a' of D that satisfy $f(d) = ad = a'd$. This leads to the following equation that means m_1 is well-defined:

$$\mathsf{L}a = \mathsf{L}a' \ .$$

For all d in D, assume that there exists two elements a and a' of D that satisfy $f(d) = a\,!\,d = a'\,!\,d$. This leads to the following equation that means m_2 is well-defined:

$$\mathsf{R}a = \mathsf{R}a' \ .$$

Second, we define $[\![-]\!]$ inductively by (3.1), (3.2), (3.3), (3.4) and (3.5).

Assume that x_{n+1} is a linear variable of M. When an arbitrary element d_{n+1} of D is regarded as a variable, it can be inductively proved that d_i is a linear variable of a term $[\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1})$ on D. Therefore, by Proposition 3.2.1, an element a of D that does not include d_{n+1} exists and it satisfies the following equation:

$$[\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1}) = ad_{n+1} \ . \tag{3.19}$$

In other words, for all d_{n+1} in D, an element a of D exists and it satisfies the equation (3.19). Therefore, for all d_{n+1} in D, (3.6) holds as below:

$$\begin{aligned} & e(m_1([\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})))(d_{n+1}) \\ & = e(\mathsf{L}a)(d_{n+1}) \ \text{(by (3.19) and definition of } m_1) \\ & = \mathsf{L}ad_{n+1} \ \text{(by definition of } e) \\ & = ad_{n+1} \\ & = [\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1}) \ . \ \text{(by (3.19))} \end{aligned}$$

For a term $[\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1})$ on D, when an arbitrary element d_{n+1} of D is regarded as a variable, it holds by Proposition 3.2.1 that an element a of D that does not include d_{n+1} exists and it satisfies the following equation:

$$[\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1}) = a\,!\,d_{n+1} \ . \tag{3.20}$$

In other words, for all d_{n+1} in D, an element a of D exists and it satisfies the equation (3.20). Therefore, for all d_{n+1} in D, (3.7) holds as below:

$$\begin{aligned} & e(m_2([\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})))(!\,d_{n+1}) \\ & = e(\mathsf{R}a)(!\,d_{n+1}) \ \text{(by (3.20) and definition of } m_2) \\ & = \mathsf{R}a\,!\,d_{n+1} \ \text{(by definition of } e) \\ & = a\,!\,d_{n+1} \\ & = [\![\vec{x}, x_{n+1} \vdash M]\!]^\wedge(\vec{d})(d_{n+1}) \ . \ \text{(by (3.20))} \end{aligned}$$

□

This theorem claims that a linear combinatory algebra needs extra combinators L and R to form a linear λ-model. These extra combinators are kinds of "classifiers". They classify the linear λ-terms that have the "same" behavior, i.e. that are β-equivalent when the same term is applied to them.

In this section we studied the ability of linear combinatory algebras to model linear λ-calculus. It was found that linear combinatory algebras themselves cannot give sound denotational semantics of linear λ-calculus although they can represent β-reductions. In terms of functional programming, we can say that linear combinatory algebras cannot classify the same functions although they can express evaluations of functions and applications.

Chapter 4

Conclusions and Future Work

We technically described the base steps of the workflow shown in Chapter 1, namely resumption-based categorical GoI. We gave a GoI situation based on resumptions by defining concrete constructions of transducers: compositions, monoidal products, traces, countable infinite copying and suitable retractions. We also gave an operator $\overline{\alpha}$ on resumptions by using an algebraic operation α. The operator $\overline{\alpha}$ constructs transducers that represents the algebraic operation α. It was found that the operator $\overline{\alpha}$ enables the extracted linear combinatory algebra to represent some interfaces to effects.

Additionally, we defined a model of untyped linear λ-calculus called a linear λ-model. A linear λ-model was proved to give sound denotational semantics. We explained the ability of linear combinatory algebras to represent linear λ-calculus by proving the linear version of the Meyer-Scott theorem. It was found that linear combinatory algebras need extra combinators that classify functions with respect to their behavior to form a linear λ-model. This is much like in the classical setting.

Given this technical result, however, it is not trivial that how the extra combinators of a linear combinatory algebra are obtained especially in the framework of categorical GoI. Since the extra combinators are a kind of classifiers, we guess that they might be produced by resumption-based categorical GoI with an appropriate equivalence relation on transducers.

To give a concrete GoI interpretation in terms of resumptions, we need to describe the rest of the workflow in Chapter 1. It needs to be explained that how effects are interpreted by an algebraic operation α and an induced operator $\overline{\alpha}$ on resumptions. Our approach to give a resumption-based GoI interpretation has several parameters: a monad that models effects, an algebraic operations that represents effects and an equivalence relation used to define resumptions. This is due to the generality of categorical framework. These parameters will affect what kind of effects are interpreted in our framework. For example, following Hasuo and Hoshino [5] a resumption-based GoI interpretation of quantum computations might be obtained if quantum effects are represented by algebraic operations. We are also interested in an equivalence relation. We use the behavioral equivalence in this paper but the trace equivalence is another example; we might indeed be able to give an axiomatization of equivalence relations such that resumption-based categorical GoI works.

As mentioned in Chapter 1, our approach takes advantage of the feature of GoI and enables us to observe dynamics of computations. That is, our approach will give concrete transducers as a GoI interpretation of programs. We expect that this enables us to implement compilers of programming languages that supports effects, like the results in [11] and [2].

References

[1] Samson Abramsky, Esfandiar Haghverdi, and Philip Scott. Geometry of interaction and linear combinatory algebras. *Mathematical Structures in Computer Science*, 12:625–665, 10 2002.

[2] Dan R. Ghica. Geometry of synthesis: a structured approach to VLSI design. In Martin Hofmann and Matthias Felleisen, editors, *POPL*, pages 363–375. ACM, 2007.

[3] Jean-Yves Girard. Geometry of interaction 1: Interpretation of system F. In S. Valentini R. Ferro, C. Bonotto and A. Zanardo, editors, *Logic Colloquium '88 Proceedings of the Colloquium held in Padova*, volume 127 of *Studies in Logic and the Foundations of Mathematics*, pages 221–260. Elsevier, 1989.

[4] Masahito Hasegawa. The uniformity principle on traced monoidal categories. *ENTCS*, 69(0):137–155, 2003. CTCS'02.

[5] Ichiro Hasuo and Naohiko Hoshino. Semantics of higher-order quantum computation via geometry of interaction. In *LICS*, pages 237–246, 2011.

[6] Ichiro Hasuo and Bart Jacobs. Traces for coalgebraic components. *Mathematical Structures in Computer Science*, 21:267–320, 4 2011.

[7] Naohiko Hoshino, Koko Muroya, and Ichiro Hasuo. Memoryful geometry of interaction: From coalgebraic components to algebraic effects. preprint.

[8] Martin Hyland and John Power. The category theoretic understanding of universal algebra: Lawvere theories and monads. 172:437–458, 2007.

[9] S. C. Kleene. On the interpretation of intuitionistic number theory. 10:109–124, 1945.

[10] S. Mac Lane. *Categories for the Working Mathematician*. Springer, Berlin, 2nd edition, 1998.

[11] Ian Mackie. The geometry of interaction machine. In Ron K. Cytron and Peter Lee, editors, *POPL*, pages 198–208. ACM Press, 1995.

[12] Eugenio Moggi. Computational lambda-calculus and monads. Technical Report ECS-LFCS-88-66, Laboratory for Foundations of Computer Science, 1988.

[13] Gordon Plotkin and John Power. Algebraic operations and generic effects. *Applied Categorical Structures*, 11(1):69–94, 2003.

[14] Gordon D. Plotkin and John Power. Adequacy for algebraic effects. In *FoSSaCS*, pages 1–24, 2001.

[15] John Power and Edmund Robinson. Premonoidal categories and notions of computation. *Mathematical Structures in Computer Science*, 7:453–468, 10 1997.

[16] Alex Simpson. Reduction in a linear lambda-calculus with applications to operational semantics. In Giesl Jürgen, editor, *RTA*, volume 3467 of *LNCS*, pages 219–234. Springer Berlin Heidelberg, 2005.

[17] Glynn Winskel. *The Formal Semantics of Programming Languages*. 1993.

[18] 高橋 正子. **計算論——計算可能性とラムダ計算**. 近代科学社, 1991.