Memoryful Geometry of Interaction

From Coalgebraic Components to Algebraic Effects

Naohiko Hoshino
RIMS, Kyoto University
naophiko@kurims.kyoto-u.ac.jp

Koko Muroya Ichiro Hasuo
Dept. Computer Science, University of Tokyo
muroykk@is.s.u-tokyo.ac.jp ichiro@is.s.u-tokyo.ac.jp

Abstract

Girard's *Geometry of Interaction (GoI)* is interaction based semantics of linear logic proofs and, via suitable translations, of functional programs in general. Its mathematical cleanness identifies essential structures in computation; moreover its use as a compilation technique from programs to state machines—"GoI implementation," so to speak—has been worked out by Mackie, Ghica and others. In this paper, we develop Abramsky's idea of *resumption based GoI* systematically into a generic framework that accounts for computational effects (nondeterminism, probability, exception, global states, interactive I/O, etc.). The framework is categorical: Plotkin & Power's *algebraic operations* provide an interface to computational effects; the framework is built on the categorical axiomatization of GoI by Abramsky, Haghverdi and Scott; and, by use of the coalgebraic formalization of *component calculus*, we describe explicit construction of state machines as interpretations of functional programs. The resulting interpretation is shown to be sound with respect to equations between algebraic operations, as well as to Moggi's equations for the computational lambda calculus. We illustrate the construction by concrete examples.

Categories and Subject Descriptors D.3 [*Formal Definitions and Theory*]: Semantics; F.3 [*Semantics of Programming Languages*]: Algebraic approaches to semantics

General Terms Theory

Keywords Geometry of interaction, monad, algebraic effect

1. Introduction

Geometry of Interaction (GoI) is introduced by Girard [10] as semantics of proofs—i.e. programs, under the Curry-Howard correspondence—for the study of dynamics and invariants of the cut elimination process (i.e. program execution). Girard's original presentation of GoI is in the language of C^*-algebras; Mackie's alternative presentation [25] as *token machines* initiated another important application of GoI, namely as a *compilation technique*. There GoI provides translation of programs into state machines; and the machines' execution results are invariant under cut elimination. Dynamics in such machines can be understood as a mathematical counterpart of control flow in program execution, and in this way, GoI connects mathematics (denotational semantics), program evaluation (operational semantics) and state based computation (low-level languages/implementations). Applications of GoI are widespread: implementation of (imperative) functional programming languages [8, 9, 25]; relationship to Krivine abstract machines [7] and to defunctionalization [30]; optimal graph reduction for the lambda calculus [11]; and the design of a functional programming language for sublinear space [6].

CSL-LICS 2014, July 14–18, 2014, Vienna, Austria.

http://dx.doi.org/10.1145/2603088.2603124

Categorical GoI

This remarkable level of integration in GoI—of operational and denotational/structural semantics—is further exemplified by its categorical axiomatics (*categorical GoI*) developed by Abramsky, Haghverdi and Scott [2]. There a general construction is given from a *traced monoidal category*—together with additional constructs, altogether called a *GoI situation*—to a *combinatory algebra*. One can then apply the *realizability* construction (see e.g. [24]) that turns a combinatory algebra (an "untyped" model) into a categorical model of a typed calculus, from which one extracts realizers as concrete interpretations. The latter are sound by construction.

In a big picture, the current work is one of the attempts to instantiate this general methodology of categorical GoI to concrete situations. Our starting point is the previous work [14] where we extend the above workflow by a step prior to it. The extension comes from the following observation (a folklore result, see Lemma 4.3; see also [18]): many traced monoidal categories arise as a Kleisli category of a monad with a suitable order structure. The resulting extended workflow is as follows.

$$\begin{array}{l}\text{(a } \mathbf{Set}\text{-monad } T \text{ whose Kleisli category is } \mathbf{Cppo}\text{-enriched)} \\ \quad \overset{\text{Kleisli construction}}{\longmapsto} \quad \text{(a traced monoidal category)} \\ \quad \overset{\text{categorical GoI [2]}}{\longmapsto} \quad \text{(a combinatory algebra)} \\ \quad \overset{\text{realizability \& realizer extraction}}{\longmapsto} \quad \text{(a GoI interpretation of a typed calculus)}\end{array} \tag{1}$$

In [14] we pursued use of this extended general workflow that is parametrized by T: in order to interpret a calculus with a certain additional feature, we start with a monad T equipped with the same feature, and the generic constructions would yield a suitable GoI interpretation. In [14], specifically, we considered a calculus for quantum computation.

Effects and Resumption Based GoI

However, following this naive scenario turned out to be far from straightforward: in [14] we ended up using a complicated continuation monad that keeps track of all the measurement outcomes. In fact the same kind of difficulty is already with nondeterminism—one of the most basic computational effects—as we exhibit now. Here we shall speak on the intuitive level, using the game-theoretic terms of *queries* and *answers* instead of the categorical language for GoI.

Consider the call-by-value evaluation of the program

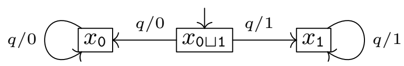
$$(\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\,(\mathtt{0} \sqcup \mathtt{1}) \tag{2}$$

where the subterm $\mathtt{0} \sqcup \mathtt{1}$ returns 0 or 1 nondeterministically. Then the whole program is expected to return 0 or 2. However, the usual GoI interpretation of (2) may return an unexpected value 1, as the result of the following interaction.

1. We ask the value of the left occurrence of x in $\mathtt{x} + \mathtt{x}$.
2. The subterm $\mathtt{0} \sqcup \mathtt{1}$ answers 0 or 1 nondeterministically.
3. We ask the value of the right occurrence of x in $\mathtt{x} + \mathtt{x}$.
4. The subterm $\mathtt{0} \sqcup \mathtt{1}$ answers 0 or 1 nondeterministically.
5. We add the values of the left x and the right x.

The difficulty is as follows. After the first query 2), the nondeterminism in the subterm $\mathtt{0} \sqcup \mathtt{1}$ is resolved, with the subterm reduced to 0 or 1. The second query 4) must stick to this choice; however, most instances of GoI—presented in terms of C^*-algebra, token machines or categories—lack such an explicit "memory" mechanism. The lack of memories in GoI causes similar difficulties with other computational effects. Even more, it seems to be also the reason why additive connectives are far less trivial to interpret in GoI: *additive slices*—a common tool for additive connectives in GoI [23, 33]—indeed have a strong flavor of memories. In the previous example (2) one may wonder if the call-by-value evaluation strategy is to blame. This is unlikely; see Remark 1.1.

The memory mechanism needed in the above example can be provided in the form of a *Mealy machine*, or a *(nondeterministic) transducer*. The term $\mathtt{0} \sqcup \mathtt{1}$ is now interpreted as



(3)

where the machine can initially respond to a query q with 0 or 1; however, after that the machine sticks to the same choice by remembering the state by means of its internal state. For more examples of Mealy-like machines, see Section 7.

The idea of using such memories (or states) in GoI is not new. In [1, 2], an instance of categorical GoI is given by the category of *resumptions*—roughly speaking, a resumption from a set A to B is a "stateful computation" from A to B; more precisely it is a suitable equivalence class (e.g. modulo bisimilarity) of transducers from A to B. In [2] it is also characterized as an element of a final coalgebra.

Remark 1.1. Lack of memories in effectful situations causes difficulties in GoI regardless of evaluation strategies. Consider the equation $(\mathtt{t} \sqcup \mathtt{s})\,\mathtt{u} = \mathtt{t}\,\mathtt{u} \sqcup \mathtt{s}\,\mathtt{u}$; we expect it to hold regardless of evaluation strategies. In the GoI interpretation of $(\mathtt{t} \sqcup \mathtt{s})\,\mathtt{u}$, when u receives a query from the t part in $\mathtt{t} \sqcup \mathtt{s}$, we must make sure that the response goes back to t, not to s. It is not clear how to enforce this without using memories, as one would see trying to interpret the program in (any presentation of) GoI.

Contributions—from Coalgebraic Components to Algebraic Effects

Building on Abramsky's idea of *resumption based GoI*, the current paper aims at a generic framework that yields GoI interpretations—or rather *GoI implementations*, since they are given concretely as state based transducers like in (3)—of calculi with various computational effects.

More specifically: we model an effect by a monad T following Moggi [26]; and as an syntactic interface we use *algebraic operations* like $\sqcup$ for the powerset monad $\mathcal{P}$, following Plotkin & Power [28]. Concrete interpretations are given by (state based) transducers with the same effect T, much like the nondeterministic one in (3). Assuming that T comes with a suitable $\mathbf{Cppo}$ structure—many effects qualify by the slight modification of adding partiality—we show that the category $\mathbf{Res}(T)$ of T-*resumptions* is traced monoidal. Then the general workflow in (1), starting from its second step, yields a GoI interpretation of a calculus, with transducers as realizers. The resulting interpretation is sound with respect to the algebraic axioms (e.g. associativity of $\sqcup$) as well as Moggi's equations for the computational lambda calculus.

This overall procedure—from a monad T and algebraic operations to a GoI interpretation in the form of T-transducers—we wish to call *memoryful GoI*, emphasizing the role of memories ("memories" here are the same as "internal states"; the choice is to distinguish from the global "state" monad). A novelty is the use of memories (in the form of internal states of transducers) that allows us to describe interpretations of algebraic effects in a generic yet concrete manner.

We describe the construction of transducers concretely in terms of *coalgebraic component calculus*. Component calculi are heavily studied in software engineering (see e.g. [4]) as means of composing *software components*. A major concern there is *compositionality*, much like in the study of process calculi (see e.g. [3]); and successful (co)algebraic techniques have been developed for the latter [21, 32] as well as for the former [5]. We rely on the categorical formalization of component calculi in [5, 15] where components are coalgebras; categorical genericity is needed since our transducers are parametrized by a monad T. In this way, the current work pursues "some convergence and unification"—suggested in the paper [1], from which we draw inspirations—of GoI and coalgebra, in program semantics.

Organization of This Paper In Section 2 we recall categorical GoI, and in Section 3 we summarize notations used in this paper. In Section 4 we introduce component calculus on transducers; in Section 5 we quotient components by behavioral equivalence and define a traced symmetric monoidal category of sets and resumptions. In Section 6, we sketch construction of categorical models of the computational lambda calculus based on categorical GoI with transducers; the resulting GoI interpretation is described in Section 7 with concrete examples. Due to space limitation, some proofs and argument in this paper are deferred to an extended version.

2. Categorical Geometry of Interaction

We recall a categorical formulation of GoI called *GoI situation* introduced by Abramsky, Haghverdi and Scott and pin down the obstacle explained in the introduction in mathematical terms. A crucial notion in their categorical axiomatics is that of *traced symmetric monoidal category*.

Definition 2.1. A *traced symmetric monoidal category* is a symmetric monoidal category $(\mathcal{C}, \otimes, \mathrm{I})$ with a family of maps

$$\mathrm{tr}^{C}_{A,B} \colon \mathcal{C}(A \otimes C, B \otimes C) \to \mathcal{C}(A, B)$$

subject to certain conditions (see [13, 20]).

Definition 2.2. Let $\mathcal{C}$ be a traced symmetric monoidal category. A *traced symmetric monoidal functor* $F \colon \mathcal{C} \to \mathcal{C}$ is a strong symmetric monoidal functor such that

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

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

For example, the category $(\mathbf{Rel}, +, \emptyset)$ of sets and relations forms a traced symmetric monoidal category: for a $\mathbf{Rel}$-morphism

$f\colon A+C \to B+C$, we define the trace operator $\mathrm{tr}^C_{A,B}(f)\colon A \to B$ by the *execution formula*

$$\mathrm{tr}^C_{A,B}(f) = f_{AB} \cup \bigcup_{n\geq 0} f_{CB} \circ f^n_{CC} \circ f_{AC}$$

where $f_{XY}\colon X \to Y$ is the restriction of f to a relation between X and Y.

Definition 2.3. A *GoI situation* is a list $(\mathcal{C}, U, F, \phi, \psi, u, v)$ consisting of a traced symmetric monoidal category $(\mathcal{C}, \otimes, \mathrm{I})$, a $\mathcal{C}$-object U and a traced symmetric monoidal functor $F\colon \mathcal{C} \to \mathcal{C}$ with retractions $\phi\colon U \otimes U \lhd U : \psi$ and $u\colon FU \lhd U : v$ together with the following retractions

$$n\colon \mathrm{I} \lhd U : n' \quad e_A\colon A \lhd FA : e'_A \quad d_A\colon FFA \lhd FA : d'_A$$
$$c_A\colon FA \otimes FA \lhd FA : c'_A \quad w_A\colon \mathrm{I} \lhd FA : w'_A$$

such that e_A, d_A, c_A and w_A are natural in A.

The retraction (ϕ, ψ) and (n, n') provides GoI interpretation of the multiplicative fragment of linear logic, and the traced symmetric monoidal functor F with the remaining retractions provide GoI interpretation of the exponential fragment of linear logic. In [2], a GoI situation is shown to yield a *linear combinatory algebra*; via the Girard translation, we obtain an *SK-algebra* that is a model of intuitionistic logic.

Proposition 2.4. *Let $(\mathcal{C}, U, F, \phi, \psi, u, v)$ be a GoI situation. The set $\mathcal{C}(U, U)$ with the binary application $a \bullet b$ on $\mathcal{C}(U, U)$ given by*

$$a \bullet b = \mathrm{tr}^U_{U,U}((U \otimes (u \circ Fb \circ v)) \circ \psi \circ a \circ \phi)$$

forms an SK-algebra*: there exist $S, K \in \mathcal{C}(U, U)$ such that*

$$S \bullet a \bullet b \bullet c = a \bullet c \bullet (b \bullet c), \qquad K \bullet a \bullet b = a$$

where we assume that the binary application is left associative.

On the categorical level, the obstacle in the introduction stems from the fact that the trace operator tr on $\mathbf{Rel}$ does not preserve the union of relations:

$$\mathrm{tr}^C_{A,B}(f \cup g) = \mathrm{tr}^C_{A,B}(f) \cup \mathrm{tr}^C_{A,B}(g) \cup g_{CB} \circ f_{AC} \cup \cdots \quad (4)$$

Failure of preservation of the trace results in failure of the equation:

$$(a \cup b) \bullet c \neq (a \bullet c) \cup (b \bullet c) \quad (5)$$

in the SK-algebra $(\mathbf{Rel}(U, U), \bullet)$ constructed from a GoI situation $(\mathbf{Rel}, U, F, \phi, \psi, u, v)$. The unexpected value returned by the program (2) appears in the extra summands in (4).

Remark 2.5. In the original definition of GoI situation in [2], the retractions (e, e'), (d, d'), (c, c') and (w, w') are required to be monoidal natural transformations. In this paper, we only require the injection side of retractions to be natural since this is enough to prove Proposition 2.4. This relaxation is needed in our concrete examples. We also note that since we do not require e, d, c and w to be monoidal, the retraction pairs (e, e'), (d, d'), (c, c') and (w, w') do not give rise to pointwise natural transformations. Pointwise naturality is required in [2] to construct weak linear category.

3. Notations

We summarize several notations used in this paper. Let $\mathbf{Set}$ be the category of sets and maps (i.e. functions). We write

$$A \xrightarrow{\mathrm{inl}_{A,B}} A + B \xleftarrow{\mathrm{inr}_{A,B}} B, \qquad A + A \xrightarrow{\gamma_A} A$$

for the injections and the codiagonal map. We write

$$A \times B \xrightarrow{\sigma_{A,B}} B \times A, \qquad A \times B + A \times C \xrightarrow{\delta_{A,B,C}} A \times (B + C)$$

for the canonical bijections. We write $\top_A\colon A \to 1$ and $\perp_A\colon \emptyset \to A$ for the unique maps. For sets I and A, we write A^I for the I-fold product of A.

Let (T, η, μ) be a monad on $\mathbf{Set}$. In order to distinguish maps from morphisms in the Kleisli category $\mathbf{Set}_T$, we write $f\colon A \to_T B$ when f is a $\mathbf{Set}_T$-morphism from A to B. Since T is a monad on $\mathbf{Set}$, we have tensorial strengths

$$TA \times B \xrightarrow{\mathrm{st}_{A,B}} T(A \times B), \quad A \times TB \xrightarrow{\mathrm{st}'_{A,B}} T(A \times B).$$

For $\mathbf{Set}_T$-morphisms $f\colon A \to_T B$ and $g\colon B \to_T C$, a map $h\colon A \to B$ and a set D, we define $\mathbf{Set}_T$-morphisms $g \circ_T f$, $f \otimes D$, $D \otimes f$ and h^* by

$$\begin{aligned}
g \circ_T f &= \mu_C \circ Tg \circ f\colon A \to_T C,\\
f \otimes D &= \mathrm{st}_{B,D} \circ (f \times D)\colon A \times D \to_T B \times D,\\
D \otimes f &= \mathrm{st}'_{D,B} \circ (D \times f)\colon D \times A \to_T D \times B,\\
h^* &= \eta_B \circ h\colon A \to_T B.
\end{aligned}$$

The first construction is the composition of Kleisli morphisms. The second and the third constructions are the *premonoidal products* in $\mathbf{Set}_T$. See [29] for premonoidal category, although further familiarity will not be needed. For $\mathbf{Set}_T$-morphisms $f\colon A \to_T B$ and $g\colon C \to_T D$ such that

$$(f \otimes D) \circ_T (A \otimes g) = (B \otimes g) \circ_T (f \otimes C),$$

we write $f \otimes g$ for $(f \otimes D) \circ_T (A \otimes g)$; this happens when T is a *commutative* monad. The last construction $(-)^*$ is the Kleisli inclusion from $\mathbf{Set}$ to $\mathbf{Set}_T$, which lifts the (finite) coproducts $(\emptyset, +, \mathrm{inl}, \mathrm{inr})$ of $\mathbf{Set}$ to (finite) coproducts $(\emptyset, +, \mathrm{inl}^*, \mathrm{inr}^*)$ of $\mathbf{Set}_T$.

For legibility, we omit some obvious isomorphisms in the remainder of this paper. For example, we write η_A for a map from $1 \times A$ to $T(1 \times A)$ obtained by composing η_A with obvious isomorphisms.

4. Transducers and a Component Calculus

4.1 Transducer

Transducers are "functions with internal states."

Definition 4.1. Let T be a monad on $\mathbf{Set}$. For sets A and B, a *T-transducer from A to B* is a pair (X, c) consisting of a set X together with a $\mathbf{Set}_T$-morphism $c\colon X \times A \to_T X \times B$. A *pointed* T-transducer is a triple (X, c, x) consisting of a T-transducer (X, c) and a map $x\colon 1 \to X$. We often drop the word 'pointed' in 'pointed T-transducer.' When (X, c, x) is a T-transducer from A to B, we write $(X, c, x)\colon A \twoheadrightarrow B$.

A T-transducer (X, c, x) is a machine consisting of a set of (internal) states X, an initial state x and a transition rule c. For example, when T is the identity functor, an equation $c(y, a) = (y', a')$ means that if an input is a and the current internal state of the machine is y, then the machine outputs a' and the next internal state is y'. The monad T enables us to consider various *effects* of transition rules: when T is the powerset monad, transition rules are nondeterministic.

Requirement 4.2. Throughout the paper we require that the symmetric monoidal category $(\mathbf{Set}_T, +, \emptyset)$ has a trace operator tr that satisfies the following restricted uniformity [12]: for all $h\colon C \to D$, $f\colon A + C \to_T B + C$ and $g\colon A + D \to_T B + D$, if $(B + h^*) \circ_T f = g \circ_T (A + h^*)$, then $\mathrm{tr}^C_{A,B}(f) = \mathrm{tr}^D_{A,B}(g)$.

It is typical in *particle style* GoI [2] that the underlying traced symmetric monoidal category of a GoI situation is a Kleisli category with a trace operator that is uniform in the above restricted sense. The next lemma is useful for checking Requirement 4.2. We write $\mathbf{Cppo}$ for the category of pointed complete posets (cpo) and continuous maps. We consider $\mathbf{Cppo}$-enrichment with respect to the symmetric monoidal structure given by the finite products. A $\mathbf{Cppo}$-enriched cocartesian category $\mathcal{C}$ is a $\mathbf{Cppo}$-enriched category whose underlying category has finite coproducts such that the coproduct $f + g\colon A + B \to C + D$ is continuous on f and g.

Lemma 4.3. *If the Kleisli category $\mathbf{Set}_T$ is a $\mathbf{Cppo}$-enriched cocartesian category such that the bottom morphisms $\perp_{A,B}\colon A \to_T B$ satisfy the following conditions:*

- $f \circ_T \perp_{A,B} = \perp_{A,B'}$ *for all* $f\colon B \to_T B'$
- $\perp_{A,B} \circ_T g^* = \perp_{A',B}$ *for all* $g\colon A' \to A$

then $(\mathbf{Set}_T, +, \emptyset)$ *satisfies Requirement 4.2.*

In the following examples, we can use Lemma 4.3 to check Requirement 4.2: we combine partiality with the standard definitions of monads so that the Kleisli categories are enriched over $\mathbf{Cppo}$.

Example 4.4. We give leading examples of monads that satisfy Requirement 4.2.

- The *lift* monad $\mathcal{L}A = 1 + A$.
- The *(full) powerset* monad $\mathcal{P}A = 2^A$ and the *countable powerset* monad $\mathcal{P}_\omega A = \{a \subseteq A \mid a \text{ is countable}\}$.
- The *probabilistic subdistribution* monad
$$\mathcal{D}A = \left\{d\colon A \to [0,1] \mid \textstyle\sum_{a \in A} da \le 1\right\}$$
where $[0,1]$ is the unit interval.
- A *global state* monad $\mathcal{S}A = (1 + A \times V^L)^{V^L}$ where V and L are (countable) sets.
- A *writer* monad $TA = 1 + M \times A$ where M is a monoid.
- An *exception* monad $\mathcal{E}A = 1 + E + A$ where E is a set.
- A *continuation* monad $TA = R^A \Rightarrow R$ where R is a pointed cpo and $R^A \Rightarrow R$ is the set of continuous maps from the A-fold product of R to R.
- An *I/O* monad $TA = \nu D.\,(O \times D + D^I + A)_\perp$ where O and I are (countable) sets.

In the last example, we regard a set as a cpo with the discrete order, and $D_\perp$ is the pointed cpo obtained by adding a bottom element to a cpo D. For an endo-functor F on $\mathbf{Cppo}$, the fixed point $\nu D.\,FD$ denotes a final F-coalgebra in $\mathbf{Cppo}$.

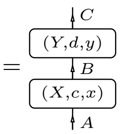
4.2 A Component Calculus

We shall extend some constructions on $\mathbf{Set}_T$ to constructions on T-transducers, namely the sequential composition $f \circ_T g$, the traced symmetric monoidal structure $(\mathbf{Set}_T, +, \emptyset, \mathrm{tr})$ and algebraic operations on T. These extensions will organize T-transducers into a "traced symmetric monoidal category," on which we will define a "GoI situation." Here the quotation marks ("so to speak") are because the equational axioms of traced symmetric monoidal category and GoI situation hold only up-to suitable equivalences between T-transducers. In Section 5, we will (properly) introduce a traced symmetric monoidal category and a GoI situation as quotients of the "traced symmetric monoidal category" and the "GoI situation" in this section.

4.2.1 Identity and Composition

For a set A, we define an "identity" on A to be the obvious one-state T-transducer
$$(1, \eta_A, \mathrm{id}_1)\colon A \twoheadrightarrow A.$$
Generalizing the above "identity," we have a construction J from a $\mathbf{Set}_T$-morphism $f\colon A \to_T B$ to a T-transducer $Jf\colon A \twoheadrightarrow B$ defined by $(1, f, \mathrm{id}_1)$. For a map $g\colon A \to B$, we define a T-transducer $J_0 g\colon A \twoheadrightarrow B$ as the composition of J and the Kleisli inclusion, namely $(1, g^*, \mathrm{id}_1)$.

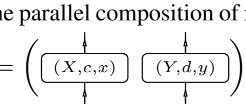
For T-transducers $(X, c, x)\colon A \twoheadrightarrow B$ and $(Y, d, y)\colon B \twoheadrightarrow C$, we define a "composition"
$$(Y, d, y) \circ (X, c, x)\colon A \twoheadrightarrow C$$
to be $(X \times Y, e, x \times y)$ where e is a $\mathbf{Set}_T$-morphism from $X \times Y \times A$ to $X \times Y \times C$ given by
$$(X \otimes d) \circ_T (X \otimes \sigma^*_{B,Y}) \circ_T (c \otimes Y) \circ_T (X \otimes \sigma^*_{Y,A}).$$
This is the sequential composition of machines:

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

We note that this is an intuitive representation of the composition and is inept for rigorous reasoning. For example, the composition of T-transducers fails to be associative in the strict sense.

4.2.2 Monoidal Product

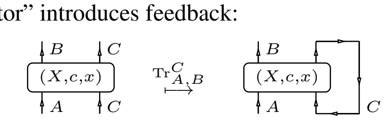
We define a "monoidal product"
$$(X, c, x) \boxplus (Y, d, y)\colon A + C \twoheadrightarrow B + D$$
of T-transducers $(X, c, x)\colon A \twoheadrightarrow B$ and $(Y, d, y)\colon C \twoheadrightarrow D$ to be $(X \times Y, e, x \times y)$ where
$$e\colon X \times Y \times (A + C) \to_T X \times Y \times (B + D)$$
is a unique $\mathbf{Set}_T$-morphism such that
$$\begin{aligned} e \circ_T (X \otimes Y \otimes \mathrm{inl}^*_{A,C}) &= (\sigma^*_{Y,X} \otimes \mathrm{inl}^*_{B,D}) \\ &\qquad \circ_T (Y \otimes c) \circ_T (\sigma^*_{X,Y} \otimes A), \\ e \circ_T (X \otimes Y \otimes \mathrm{inr}^*_{A,C}) &= (X \otimes Y \otimes \mathrm{inr}^*_{B,D}) \circ_T (X \otimes d). \end{aligned}$$
The "monoidal product" $\boxplus$ is the parallel composition of machines:

$(X, c, x) \boxplus (Y, d, y) =$ 

The T-transducers (X, c, x) and (Y, d, y) behave independently following their own internal states.

4.2.3 Trace

For a T-transducer $(X, c, x)\colon A + C \twoheadrightarrow B + C$, we define a T-transducer $\mathrm{Tr}^C_{A,B}(X, c, x)\colon A \twoheadrightarrow B$ to be
$$\left(X, \mathrm{tr}^{X \times C}_{X \times A, X \times B}((\delta^*_{X,B,C})^{-1} \circ_T c \circ_T \delta^*_{X,A,C}), x\right).$$
Here δ is the bijection in Section 3 and tr is the trace operator of $\mathbf{Set}_T$. The operator Tr is a "trace operator" with respect to the "symmetric monoidal structure" $(\boxplus, \emptyset)$. Checking that trace axioms are indeed "satisfied" is laborious but doable; see [15]. The "trace operator" introduces feedback:



The internal states enable us to memorize history of feedbacking. Let $\mathcal{L}$ be the lifting monad given in Example 4.4. Then the internal states of an $\mathcal{L}$-transducer
$$(\{0, 1, \cdots, n\}, c, 0)\colon \{0\} + \{0\} \twoheadrightarrow \{0\} + \{0\}$$
given by
$$c(i, \mathrm{inl}_{\{0\}+\{0\}}(0)) = \begin{cases} (0, \mathrm{inr}_{\{0\}+\{0\}}(0)) & (i < n) \\ (0, \mathrm{inl}_{\{0\}+\{0\}}(0)) & (i = n) \end{cases}$$
$$c(i, \mathrm{inr}_{\{0\}+\{0\}}(0)) = \begin{cases} (i+1, \mathrm{inr}_{\{0\}+\{0\}}(0)) & (i < n) \\ (0, \mathrm{inl}_{\{0\}+\{0\}}(0)) & (i = n) \end{cases}$$

memorizes number of feedback loops.

4.2.4 GoI Situation

Let $\mathbb{N}$ be the set of natural numbers. We define maps

$$\kappa_n : 1 \to \mathbb{N}, \quad \varpi_{n,X} : X^{\mathbb{N}} \to X^{\mathbb{N}} \times X, \quad f^{\mathbb{N}} : A^{\mathbb{N}} \to B^{\mathbb{N}}$$

to be the constant map $\kappa_n(*) = n$, the permutation that picks the n-th element, and the $\mathbb{N}$-fold product of a map $f : A \to B$.

For a set A, we define a set FA to be $\mathbb{N} \times A$, and for a T-transducer $(X, c, x) : A \twoheadrightarrow B$, we define a T-transducer

$$F(X, c, x) : FA \twoheadrightarrow FB$$

to be $(X^{\mathbb{N}}, c', x^{\mathbb{N}})$ whose transition map

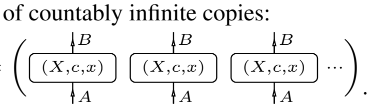
$$c' : X^{\mathbb{N}} \times \mathbb{N} \times A \to_T X^{\mathbb{N}} \times \mathbb{N} \times B$$

is a unique $\mathbf{Set}_T$-morphism such that

$$c'(-, n, -) : X^{\mathbb{N}} \times A \to_T X^{\mathbb{N}} \times \mathbb{N} \times B = ((\varpi^*_{n,X})^{-1} \otimes \kappa^*_n \otimes B) \circ_T (X^{\mathbb{N}} \otimes c) \circ_T (\varpi^*_{n,X} \otimes A)$$

for all natural numbers n. The construction F, which corresponds to the linear exponential comonad of linear logic, introduces a parallel composition of countably infinite copies:

$$F(X, c, x) =$$



Each (X, c, x) in $F(X, c, x)$ behaves independently.

We choose bijections $\phi : \mathbb{N} + \mathbb{N} \cong \mathbb{N} : \psi$ and $u : F\mathbb{N} \cong \mathbb{N} : v$ in $\mathbf{Set}$, which induce the following "retractions"

$$J_0\phi : \mathbb{N} + \mathbb{N} \cong \mathbb{N} : J_0\psi, \qquad J_0u : F\mathbb{N} \cong \mathbb{N} : J_0v.$$

The list $(\mathbb{N}, F, J_0\phi, J_0\psi, J_0u, J_0v)$ forms a "GoI situation." In fact, we have the following "retractions"

$$J_0\bot_{\mathbb{N}} : \emptyset \lhd \mathbb{N} : J(\mathrm{tr}^{\mathbb{N}}_{\mathbb{N},\emptyset}(\gamma^*_{\mathbb{N}}))$$

$$J_0(\kappa_1 \times A) : A \lhd FA : J_0(\top_{\mathbb{N}} \times A) \qquad \text{(dereliction)}$$

$$J_0(u \times A) : FFA \cong FA : J_0(v \times A) \qquad \text{(digging)}$$

$$J_0\bot_{FA} : \emptyset \lhd FA : J(\mathrm{tr}^{FA}_{FA,\emptyset}(\gamma^*_{FA})) \qquad \text{(weakening)}$$

$$J_0(\phi \times A) : FA + FA \cong FA : J_0(\psi \times A) \qquad \text{(contraction)}$$

where we omit several obvious "isomorphisms."

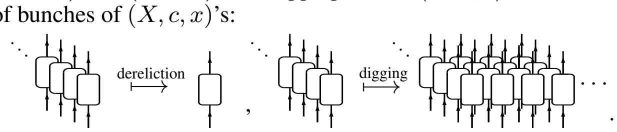
We illustrate how these "retractions" act on T-transducers. We note that a pair of T-transducers

$$(Y, d, y) : A' \rightleftarrows A : (Y', d', y')$$

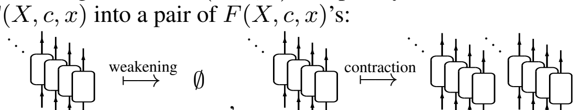
induces a translation of T-transducers:

$$(X, c, x) : A \twoheadrightarrow A \longmapsto (Y', d', y') \circ (X, c, x) \circ (Y, d, y) : A' \twoheadrightarrow A'.$$

For a T-transducer $(X, c, x) : A \twoheadrightarrow A$, dereliction pulls out the first (X, c, x) in $F(X, c, x)$, and digging sorts $F(X, c, x)$ into a bunch of bunches of (X, c, x)'s:



Weakening discards $F(X, c, x)$ completely, and contraction sorts $F(X, c, x)$ into a pair of $F(X, c, x)$'s:



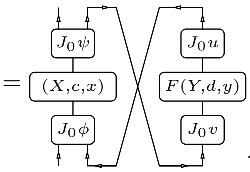
The construction of GoI situation is essentially equivalent to those in [2]. For operational description of GoI situation, see [16].

For T-transducers $(X, c, x), (Y, d, y) : \mathbb{N} \twoheadrightarrow \mathbb{N}$, we define a T-transducer $(X, c, x) \bullet (Y, d, y) : \mathbb{N} \twoheadrightarrow \mathbb{N}$ to be

$$\mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((\mathbb{N} \boxplus (J_0u \circ F(Y, d, y) \circ J_0v)) \circ J_0\psi \circ (X, c, x) \circ J_0\phi).$$

Since $(\mathbb{N}, F, J_0\phi, J_0\psi, J_0u, J_0v)$ is a "GoI situation," the set of T-transducers with the binary application $(-) \bullet (-)$ forms an "SK-algebra" by Proposition 2.4. The binary application consists of *parallel composition plus hiding*:

$$(X, c, x) \bullet (Y, d, y) =$$



Hiding means that we can not observe interaction between $J_0u \circ F(Y, d, y) \circ J_0v$ and $J_0\psi \circ (X, c, x) \circ J_0\phi$ from outside.

Remark 4.5. Since we work in untyped setting (at the transducer level), it is straightforward to extend our results to model polymorphic calculi. For modeling computational lambda calculi without polymorphism, you can choose typed approach, which will simplify our framework.

4.2.5 Algebraic Operation

We extend algebraic operations on the monad T to operations on T-transducers. We first recall the definition of algebraic operation, which is a mathematical interface to computational effects.

Definition 4.6 ([28])**.** Let T be a strong monad on a cartesian closed category $(\mathcal{C}, 1, \times, \Rightarrow)$ with countable products, and let I be a countable set. An *I-ary algebraic operation on T* is a family of $\mathcal{C}$-morphisms

$$\{\alpha_{A,B} : (A \Rightarrow TB)^I \to (A \Rightarrow TB)\}_{A,B \in \mathcal{C}}$$

such that

$$\alpha_{A',B'} \circ \mathrm{cp}^I \circ \Delta = \mathrm{cp} \circ ((B \Rightarrow TB') \times \alpha_{A,B} \times (A' \Rightarrow A))$$

where

$$\mathrm{cp} : (B \Rightarrow TB') \times (A \Rightarrow TB) \times (A' \Rightarrow A) \to (A' \Rightarrow TB')$$

is the (Kleisli) composition, and

$$\Delta : (B \Rightarrow TB') \times (A \Rightarrow TB)^I \times (A' \Rightarrow A) \to ((B \Rightarrow TB') \times (A \Rightarrow TB) \times (A' \Rightarrow A))^I$$

is the $\mathcal{C}$-morphism that is diagonal in the first argument and the third argument. We write $\mathrm{ary}(\alpha)$ for I.

We define $\mathbf{AlgOp}_T$ to be the category of algebraic operations on T: an object is a countable set, and a morphism from I to I' is a family of $\mathcal{C}$-morphisms

$$\{\alpha_{A,B} : (A \Rightarrow TB)^I \to (A \Rightarrow TB)^{I'}\}_{A,B \in \mathcal{C}}$$

such that the family $\{\pi_{j\,A,B} \circ \alpha_{A,B}\}_{A,B \in \mathcal{C}}$ is an I-ary algebraic operation for all $j \in I'$ where $\pi_{j\,A,B}$ is the j-th projection from $(A \Rightarrow TB)^{I'}$ to $A \Rightarrow TB$. The category $\mathbf{AlgOp}_T$ has countable products given by the disjoint sum.

Example 4.7. We give examples of algebraic operations.

- A binary algebraic operation $\oplus$ on $\mathcal{P}$ given by

  $$(f \oplus_{A,B} g)(a) = f(a) \cup g(a)$$

  where we use an infix notation. We will use the operation $\oplus$ to interpret the $\sqcup$ construct in the introduction.

- A binary algebraic operation $\oplus^p$ on $\mathcal{D}$ given by
$$(f \oplus^p_{A,B} g)(a) = p \cdot f(a) + (1-p) \cdot g(a)$$
where p is a real number in the unit interval $[0,1]$.
- A state monad $\mathcal{S}X = (1 + X \times V^L)^{V^L}$ for a countable set of *locations* L and a countable set of *values* V has the following algebraic operations
$$\mathrm{lookup}_{\ell,A,B} \colon \mathbf{Set}_{\mathcal{S}}(A,B)^V \to \mathbf{Set}_{\mathcal{S}}(A,B),$$
$$\mathrm{update}_{v,\ell,A,B} \colon \mathbf{Set}_{\mathcal{S}}(A,B) \to \mathbf{Set}_{\mathcal{S}}(A,B)$$
for each $\ell \in L$ and $v \in V$ given by
$$((\mathrm{lookup}_{\ell,A,B}(f))(a))(s) = (f_{s(\ell)}(a))(s),$$
$$((\mathrm{update}_{v,\ell,A,B}(f))(a))(s) = (fa)(s[v/\ell])$$
where the state $s[v/\ell] \in V^L$ is given by $s[v/\ell](\ell') = s(\ell')$ for $\ell \neq \ell'$ and $s[v/\ell](\ell) = v$.

For other examples of algebraic operations, see [28].

For a monad T on $\mathbf{Set}$, an I-ary algebraic operation α on T and a family of T-transducers $\{(X_i, c_i, x_i) \colon A \twoheadrightarrow B\}_{i \in I}$, we define
$$\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i \in I} \colon A \twoheadrightarrow B$$
to be a T-transducer $(1 + Y, d, \mathrm{inl}_{1,Y})$ consisting of a coproduct $Y = \coprod_{i \in I} X_i \xleftarrow{\mathrm{inj}_i} X_i$ and a unique $\mathbf{Set}_T$-morphism d from $(1+Y) \times A$ to $(1+Y) \times B$ satisfying
$$d \circ_T (\mathrm{inl}^*_{1,Y} \otimes A) = \alpha_{A,(1+Y)\times B}\{c'_i\}_{i \in I}$$
$$d \circ_T ((\mathrm{inr}^*_{1,Y} \circ_T \mathrm{inj}^*_i) \otimes A) = ((\mathrm{inr}^*_{1,Y} \circ_T \mathrm{inj}^*_i) \otimes B) \circ_T c_i$$
where c'_i is a $\mathbf{Set}_T$-morphism from A to $(1+Y) \times B$ given by $(\mathrm{inr}^*_{1,Y} \otimes B) \circ_T (\mathrm{inj}^*_i \otimes B) \circ_T c_i \circ_T (x^*_i \otimes A)$.

Intuitively, the construction $\overline{\alpha}_{A,B}$ introduces branching at the (fresh) initial state. A T-transducer $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i \in I}$ memorizes the first branching information using its internal states, and after the first branching, the behavior of $\overline{\alpha}_{A,B}\{(X_i, c_i, x_i)\}_{i \in I}$ in the i-th branching follows the i-th T-transducer (X_i, c_i, x_i) for each $i \in I$. For example, the nondeterministic Mealy machine in (3) is the same as the following $\mathcal{P}$-transducer
$$(\{x_0\}, c_0, x_0) \overline{\oplus}_{\{q\},\{0,1\}} (\{x_1\}, c_1, x_1) \colon \{q\} \to \{0,1\}$$
where $(\{x_i\}, c_i, x_i) \colon \{q\} \twoheadrightarrow \{0,1\}$ are $\mathcal{P}$-transducers given by $c_i(x_i, q) = (x_i, i)$ for $i = 0, 1$.

5. Behavioral Equivalence

We have presented "GoI situation" on the "traced symmetric monoidal category" of sets and T-transducers. Precisely speaking, they are not so in a strict sense: in order to satisfy the equational axioms of traced symmetric monoidal category and GoI situation, T-transducers must be suitably quotiented. For example, $(X, c, x) \circ (1, \eta_A, \mathrm{id}_1)$ is not equal to (X, c, x), and we need to identify them. In this paper, we use *behavioral equivalence*, a notion common in coalgebra [19]. Intuitively, two (pointed) T-transducers are behaviorally equivalent if the initial states are connected by a zigzag of homomorphisms.

Definition 5.1. Let (X, c, x) and (Y, d, y) be T-transducers from A to B. A *homomorphism from* (X, c, x) *to* (Y, d, y) is a map $h \colon X \to Y$ such that $(h^* \otimes B) \circ_T c = d \circ_T (h^* \otimes A)$ and $h \circ x = y$.

Definition 5.2. For T-transducers (X, c, x) and (Y, d, y) from A to B, we say that (X, c, x) is *behaviorally equivalent to* (Y, d, y) if there is a T-transducer $(Z, e, z) \colon A \twoheadrightarrow B$ and homomorphisms from (X, c, x) to (Z, e, z) and from (Y, d, y) to (Z, e, z). When (X, c, x) is behaviorally equivalent to (Y, d, y), we write $(X, c, x) \simeq^T_{A,B} (Y, d, y)$.

A zigzag of homomorphisms can be reduced to a *cospan* used in the previous definition, since the category of T-transducers (identified as coalgebras) has pushouts. See [19].

Up to the behavioral equivalence, we can drop the quotation marks in Section 4.2. It is easy to check that constructions $\circ, \boxplus, \mathrm{Tr}, F, \bullet$ and $\overline{\alpha}$ are compatible with the behavioral equivalence. Below we abuse notations: we use $\boxplus, \mathrm{Tr}, F$ and $\bullet$ for operators on T-transducers as well as those on equivalence classes of T-transducers.

We define a category $\mathbf{Res}(T)$ by

- Objects are sets.
- Morphisms from A to B are $\simeq^T_{A,B}$-equivalence classes of T-transducers from A to B.

For a T-transducer $(X, c, x) \colon A \twoheadrightarrow B$, we write $[(X, c, x)]$ for the $\mathbf{Res}(T)$-morphism from A to B represented by (X, c, x). The identity on A is $[(1, \eta_A, \mathrm{id}_1)]$, and the composition of a $\mathbf{Res}(T)$-morphism $[(X, c, x)]$ from A to B and a $\mathbf{Res}(T)$-morphism $[(Y, d, y)]$ from B to C is $[(Y, d, y) \circ (X, c, x)]$.

The category $\mathbf{Res}(T)$ with $(\boxplus, \emptyset, \mathrm{Tr})$ is a traced symmetric monoidal category. The coherence isomorphisms of the symmetric monoidal category $(\mathbf{Set}, +, \emptyset)$ induce coherence isomorphisms of $(\mathbf{Res}(T), \boxplus, \emptyset)$. The following list
$$(\mathbf{Res}(T), \mathbb{N}, F, [J_0\phi], [J_0\psi], [J_0u], [J_0v])$$
is a GoI situation, and $(\mathbf{Res}(T)(\mathbb{N}, \mathbb{N}), \bullet)$ is an SK-algebra.

Theorem 5.3. *Let α be an I-ary algebraic operation on T. The operator $\overline{\alpha}$ is natural, and* Tr *is distributive over $\overline{\alpha}$ modulo the behavioral equivalence:*

- *For each T-transducer $(X, c, x) \colon B \twoheadrightarrow B'$, for each family of T-transducers $\{(Y_i, d_i, y_i) \colon A \twoheadrightarrow B\}_{i \in I}$ and for each map $h \colon A' \to A$, it holds that*
$$(X, c, x) \circ \overline{\alpha}_{A,B}\{(Y_i, d_i, y_i)\}_{i \in I} \circ J_0 h$$
$$\simeq^T_{A',B'} \overline{\alpha}_{A',B'}\{(X, c, x) \circ (Y_i, d_i, y_i) \circ J_0 h\}_{i \in I}.$$
- *For each family of T-transducers $\{(X_i, c_i, x_i)\}_{i \in I}$ from $A + C$ to $B + C$, it holds that*
$$\mathrm{Tr}^C_{A,B}(\overline{\alpha}_{A+C,B+C}\{(X_i, c_i, x_i)\}_{i \in I})$$
$$\simeq^T_{A,B} \overline{\alpha}_{A,B}\{\mathrm{Tr}^C_{A,B}(X_i, c_i, x_i)\}_{i \in I}. \quad (6)$$

Let α be an I-ary algebraic operation on T. The following behavioral equivalence is a consequence of Theorem 5.3:
$$\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i, c_i, x_i)\}_{i \in I} \bullet (Y, d, y)$$
$$\simeq^T_{\mathbb{N},\mathbb{N}} \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i, c_i, x_i) \bullet (Y, d, y)\}_{i \in I} \quad (7)$$
where $\{(X_i, c_i, x_i)\}_{i \in I}$ and (Y, d, y) are T-transducers from $\mathbb{N}$ to $\mathbb{N}$. In fact, we have
$$\begin{aligned}
&\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i, c_i, x_i)\}_{i \in I} \bullet (Y, d, y) \\
&= \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((Z, e, z) \circ J_0\psi \circ \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i, c_i, x_i)\}_{i \in I} \circ J_0\phi) \\
&\simeq^T_{\mathbb{N},\mathbb{N}} \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}(\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(Z, e, z) \circ J_0\psi \circ (X_i, c_i, x_i) \circ J_0\phi\}_{i \in I}) \\
&\simeq^T_{\mathbb{N},\mathbb{N}} \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((Z, e, z) \circ J_0\psi \circ (X_i, c_i, x_i) \circ J_0\phi)\}_{i \in I} \\
&= \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{(X_i, c_i, x_i) \bullet (Y, d, y)\}_{i \in I}
\end{aligned}$$
where we write (Z, e, z) for $\mathbb{N} \boxplus (J_0 u \circ F(Y, d, y) \circ J_0 v)$. The first equivalence follows from naturality of $\overline{\alpha}$, and the second equivalence follows from distributivity of Tr over $\overline{\alpha}$. The behavioral equivalences (6) and (7) are the equivalences that we wish to hold as we observed at the end of Section 2.

6. Realizability and Categorical Models

In the next section, we exemplify GoI interpretation of algebraic effects. The purpose of this section is to sketch how to derive them: we use realizability technique. For details of arguments and proofs in this section are deferred to an extended version.

From the SK-algebra $(\mathbf{Res}(T)(\mathbb{N}, \mathbb{N}), \bullet)$, we can utilize the realizability construction and constructs a cartesian closed category $\mathbf{Per}(T)$ consisting of *partial equivalence relations* on the SK-algebra $\mathbf{Res}(T)(\mathbb{N}, \mathbb{N})$ and realizable maps. See [17] for a precise definition of the category of partial equivalence relations. Since the category $\mathbf{Per}(T)$ has countable products, we can consider algebraic operations with countable arities on monads on $\mathbf{Per}(T)$.

The next theorem is our main theorem, from which soundness of GoI interpretation that we are going to give follows.

Theorem 6.1. *The cartesian closed category $\mathbf{Per}(T)$ has a strong monad Φ and an identity-on-object countable-product-preserving faithful functor $(-)^\dagger \colon \mathbf{AlgOp}_T \to \mathbf{AlgOp}_\Phi$.*

We only give a definition of ΦR for R in $\mathbf{Per}(T)$. Let h be a map from $\mathbb{N}$ to $\mathbb{N}$ given by

$$\phi \circ (\phi + \mathbb{N}) \circ (\mathbb{N} + \varsigma) \circ (\psi + \mathbb{N}) \circ \varsigma \circ (\phi + \mathbb{N}) \circ (\mathbb{N} + \varsigma) \circ (\psi + \mathbb{N}) \circ \psi$$

where $\varsigma \colon \mathbb{N} + \mathbb{N} \to \mathbb{N} + \mathbb{N}$ is the swapping. We derived h using *combinatory completeness*: the map h represents a term $\lambda x.\, \lambda k.\, k\, x$ of the untyped linear lambda calculus [31]. We say that an object R in $\mathbf{Per}(T)$ is *closed* when $(\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{a_i\}_{i \in \mathrm{ary}(\alpha)}, \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{a'_i\}_{i \in \mathrm{ary}(\alpha)})$ is in R for each $\{(a_i, a'_i) \in R\}_{i \in \mathrm{ary}(\alpha)}$ and for each algebraic operation α on T. We define ΦR by

$$\Phi R = \bigcap \{S \in \mathbf{Per}(T) \mid R' \subseteq S \text{ and } S \text{ is closed}\}$$

where $R' = \{([J_0 h] \bullet a, [J_0 h] \bullet a') \mid (a, a') \in R\}$.

By Theorem 6.1, the Kleisli category $\mathbf{Per}(T)_\Phi$ is a categorical model of the computational lambda calculus, i.e., there is a canonical interpretation of the computational lambda calculus in $\mathbf{Per}(T)_\Phi$. The interpretation, which we call *categorical interpretation*, is sound with respect to the standard equational theory of the computational lambda calculus [26]. We can interpret algebraic effects using algebraic operations on Φ induced by algebraic operations on T via $(-)^\dagger$. For example, when we need nondeterminism, we can start from the powerset monad; when we need global states, we can start from a global state monad.

We sketch extraction of GoI interpretation—i.e. extraction of concrete T-transducers as realizers—from the categorical interpretation of the computational lambda calculus extended with algebraic effects and a base type `nat`. For simplicity, we only consider closed terms.

1. We choose a monad T on $\mathbf{Set}$ that satisfies Requirement 4.2.
2. We interpret the computational lambda calculus in the Kleisli category $\mathbf{Per}(T)_\Phi$ as in [26, 28] where we interpret algebraic effects by algebraic operations on Φ derived from algebraic operations on T via $(-)^\dagger$, and we interpret `nat` by a natural number object of $\mathbf{Per}(T)$.
3. The categorical interpretation of a closed term `t` of a type τ bijectively corresponds to an equivalence class of a partial equivalence relation $\Phi[\![\tau]\!]$ where $[\![\tau]\!]$ is the categorical interpretation of the type τ. We choose a $\mathbf{Res}(T)$-morphism on $\mathbb{N}$ that represents the equivalence class, and then, we extract a T-transducer $(\!|\mathtt{t}|\!) \colon \mathbb{N} \twoheadrightarrow \mathbb{N}$ that represents the $\mathbf{Res}(T)$-morphism on $\mathbb{N}$.

We call the T-transducer $(\!|\mathtt{t}|\!)$ *GoI interpretation* of a term `t`.

We extracted GoI interpretation so that the next theorem holds.

Theorem 6.2 (Soundness)**.** *For closed terms* `t` *and* `s` *of type τ,*

- *If* $\mathtt{t} \approx \mathtt{s}$, *then* $([(\!|\mathtt{t}|\!)], [(\!|\mathtt{s}|\!)]) \in \Phi[\![\tau]\!]$.
- *If* $\mathtt{t} \approx \mathtt{s}$ *and τ is the base type* `nat`, *then* $(\!|\mathtt{t}|\!) \simeq^T_{\mathbb{N},\mathbb{N}} (\!|\mathtt{s}|\!)$.

where $[(\!|\mathtt{t}|\!)]$ is the $\mathbf{Res}(T)$-morphism represented by $(\!|\mathtt{t}|\!)$, and we write $\mathtt{t} \approx \mathtt{s}$ *when the equation holds in the extension of the computational lambda calculus. For example, we have*

$$\mathtt{v}\,(3 \sqcup 5) \approx \mathtt{v}\,3 \sqcup \mathtt{v}\,5, \qquad 3 \sqcup 5 \sqcup 3 \approx 3 \sqcup 5 \approx 5 \sqcup 3$$

for any value `v` *when the extension of the computational lambda calculus has nondeterminism.*

7. GoI Interpretation of Algebraic Effects

7.1 Memoryless GoI Interpretation

For comparison, we first present (memoryless) GoI interpretation of the following programs:

$$(\lambda \mathtt{xy} : \mathtt{nat}.\, \mathtt{x} + \mathtt{y})\, 5\, 3 \qquad (\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, 3.$$

We write $\boldsymbol{g}$ for $\phi \circ \mathrm{inl}_{\mathbb{N},\mathbb{N}}$, $\boldsymbol{d}$ for $\phi \circ \mathrm{inr}_{\mathbb{N},\mathbb{N}}$ like [23] and $\langle n, m\rangle$ for $u(n, m)$. For $i \in \mathbb{N}$, we define a map $\mathbf{k}_i \colon \mathbb{N} \to \mathbb{N}$ by

$$\mathbf{k}_i \langle m, n\rangle = \langle m, i\rangle,$$

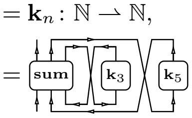
and we define maps $\mathbf{sum}, \mathbf{cpy} \colon \mathbb{N} + \mathbb{N} + \mathbb{N} \to \mathbb{N} + \mathbb{N} + \mathbb{N}$ by

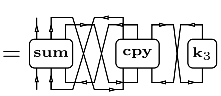
$$\begin{aligned}
\mathbf{sum}(\mathrm{inj}_1(n)) &= \mathrm{inj}_2(n) \\
\mathbf{sum}(\mathrm{inj}_2(n)) &= \mathrm{inj}_3\langle n, 0\rangle \\
\mathbf{sum}(\mathrm{inj}_3\langle\langle n, m\rangle, l\rangle) &= \mathrm{inj}_1\langle n, m + l\rangle \\
\mathbf{cpy}(\mathrm{inj}_1\langle n, m\rangle) &= \mathrm{inj}_3\langle \boldsymbol{g}n, m\rangle \\
\mathbf{cpy}(\mathrm{inj}_2\langle n, m\rangle) &= \mathrm{inj}_3\langle \boldsymbol{d}n, m\rangle \\
\mathbf{cpy}(\mathrm{inj}_3\langle \boldsymbol{g}n, m\rangle) &= \mathrm{inj}_1\langle n, m\rangle \\
\mathbf{cpy}(\mathrm{inj}_3\langle \boldsymbol{d}n, m\rangle) &= \mathrm{inj}_2\langle n, m\rangle
\end{aligned}$$

where $\mathrm{inj}_i \colon \mathbb{N} \to \mathbb{N} + \mathbb{N} + \mathbb{N}$ is the i-th injection. The map $\mathbf{cpy}$ is from contraction in the GoI situation.

In (memoryless) GoI interpretation, we interpret a closed term as a partial map from $\mathbb{N}$ to $\mathbb{N}$. The following diagrams present GoI interpretation of programs:

$$(\!|n|\!) = \mathbf{k}_n \colon \mathbb{N} \rightharpoonup \mathbb{N},$$

$(\!|(\lambda \mathtt{xy} : \mathtt{nat}.\, \mathtt{x} + \mathtt{y})\, 5\, 3|\!) =$  $: \mathbb{N} \rightharpoonup \mathbb{N},$

$(\!|(\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, 3|\!) =$  $: \mathbb{N} \rightharpoonup \mathbb{N}.$

If we input $\langle n, m\rangle$ to $(\!|(\lambda \mathtt{xy} : \mathtt{nat}.\, \mathtt{x} + \mathtt{y})\, 5\, 3|\!)$, then we get an output $\langle n, 8\rangle$ as a result of the following interactive computation between $\mathbf{sum}$, $\mathbf{k}_3$ and $\mathbf{k}_5$.

1. $\mathbf{sum}$ receives an input $\langle n, m\rangle$ from the leftmost port and outputs $\langle n, m\rangle$ from the middle port to ask a value of `x`.
2. $\mathbf{k}_5$ answers $\langle n, 5\rangle$ to $\mathbf{sum}$.
3. $\mathbf{sum}$ receives an input $\langle n, 5\rangle$ from the middle port and outputs $\langle\langle n, 5\rangle, 0\rangle$ from the rightmost port to ask a value of `y`.
4. $\mathbf{k}_3$ answers $\langle\langle n, 5\rangle, 3\rangle$ to $\mathbf{sum}$.
5. $\mathbf{sum}$ outputs $\langle n, 8\rangle$ from the leftmost port.

As a whole, GoI interpretation is sound with respect to β-equality: $(\!|(\lambda \mathtt{xy} : \mathtt{nat}.\, \mathtt{x} + \mathtt{y})\, 5\, 3|\!)$ is equal to $\mathbf{k}_8$. Similarly, we can check that the GoI interpretation $(\!|(\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, 3|\!)$ is equal to $\mathbf{k}_6$. The interactive computation illustrates how $\mathbf{sum}$ and $\mathbf{cpy}$ work: $\mathbf{sum}$ computes sum, and $\mathbf{cpy}$ copies data.

7.2 Memoryful GoI Interpretation of Global State

GoI interpretation of the computational lambda calculus (i.e. call-by-value calculus) in Section 7.2 and Section 7.3 follows from the general scheme that we developed in this paper. In this section, we present GoI interpretation of the computational lambda calculus extended with global states. We have a countably infinite set Loc of location names, and each location stores a natural number. Existence of global states enables us to fetch a natural number stored at a location $\ell \in \mathrm{Loc}$ and update a value stored at a location ℓ:

$$!\,\ell : \mathtt{nat}, \qquad \ell := 3 : \mathtt{unit}.$$

We extract GoI interpretation of global states from the categorical interpretation in $\mathbf{Per}(\mathcal{S})_\Phi$ where Φ is the monad in Theorem 6.1 for $T = \mathcal{S}$ given by $\mathcal{S}A = (1 + A \times \mathbb{N}^{\mathrm{Loc}})^{\mathbb{N}^{\mathrm{Loc}}}$. This is a global state monad in Example 4.4 where $V = \mathbb{N}$ and $L = \mathrm{Loc}$. In this section and the next section, we often confuse a map $f : \mathbb{N} \to \mathbb{N}$ with a T-transducer $J_0 f : \mathbb{N} \multimap \mathbb{N}$ and a $\mathbf{Set}_T$-morphism $g : \mathbb{N} \to_T \mathbb{N}$ with a T-transducer $Jg : \mathbb{N} \multimap \mathbb{N}$.

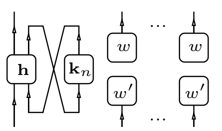
GoI Interpretation of core fragment For a type judgment

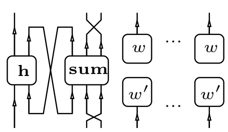
$$\Gamma \vdash \mathtt{t} : \tau \qquad (\Gamma = x_1 : \tau_1, \cdots, x_n : \tau_n),$$

we inductively define an $\mathcal{S}$-transducer

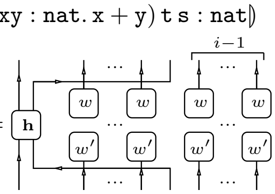
$$(\!|\Gamma \vdash \mathtt{t} : \tau|\!) : \mathbb{N}^{\otimes(n+1)} \multimap \mathbb{N}^{\otimes(n+1)}$$

where $\mathbb{N}^{\otimes(n+1)}$ is the $(n+1)$-fold tensor product of $\mathbb{N}$, i.e., $(n+1)$-fold direct sum of $\mathbb{N}$. First, we give GoI interpretation of constants, variables, the term application and the lambda abstraction. We interpret natural numbers, summation and variables as follows:

$(\!|\Gamma \vdash n : \mathtt{nat}|\!) =$ 

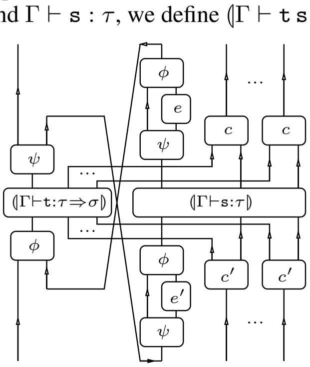
$(\!|\Gamma, \mathtt{x} : \mathtt{nat}, \mathtt{y} : \mathtt{nat} \vdash \mathtt{x} + \mathtt{y} : \mathtt{nat}|\!) =$ 

$$(\!|\Gamma \vdash \mathtt{t} + \mathtt{s} : \mathtt{nat}|\!) = (\!|\Gamma \vdash (\lambda \mathtt{xy} : \mathtt{nat}.\, \mathtt{x} + \mathtt{y})\, \mathtt{t}\, \mathtt{s} : \mathtt{nat}|\!)$$

$(\!|\mathtt{x}_1 : \tau_1, \cdots, \mathtt{x}_n : \tau_n \vdash \mathtt{x}_i : \tau_i|\!) =$ 

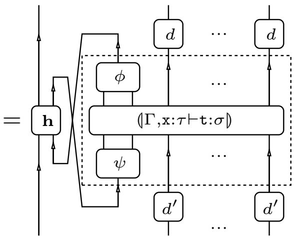
where $w' : \mathbb{N} \to_{\mathcal{S}} \emptyset$ and $w : \emptyset \to_{\mathcal{S}} \mathbb{N}$ are unique $\mathbf{Set}_{\mathcal{S}}$-morphisms, and $\mathbf{h} : \mathbb{N} + \mathbb{N} \to \mathbb{N} + \mathbb{N}$ is $\psi \circ h \circ \phi$. Here h is from Section 6, and w and w' are from weakening of the GoI situation in Section 4.2.4. The combinator $\mathbf{h}$ corresponds to the unit of the monad Φ.

For $\Gamma \vdash \mathtt{t} : \tau \Rightarrow \sigma$ and $\Gamma \vdash \mathtt{s} : \tau$, we define $(\!|\Gamma \vdash \mathtt{t}\, \mathtt{s} : \sigma|\!)$ by

$(\!|\Gamma \vdash \mathtt{t}\,\mathtt{s} : \sigma|\!) =$ 

where we write e for $u \circ (\kappa_1 \times \mathbb{N})$, e' for $(\top_{\mathbb{N}} \times \mathbb{N}) \circ v$, c for $u \circ (\phi \times \mathbb{N}) \circ (v + v)$ and c' for $(u + u) \circ (\psi \times \mathbb{N}) \circ v$. Here we omit some obvious bijections like $\mathbb{N} \times (\mathbb{N} + \mathbb{N}) \cong \mathbb{N} \times \mathbb{N} + \mathbb{N} \times \mathbb{N}$. These $\mathcal{S}$-transducers are from dereliction and contraction of the GoI situation given in Section 4.2.4. We note that the component **cpy** consists of c and c'.

For $\Gamma, \mathtt{x} : \tau \vdash \mathtt{t} : \sigma$, we define $(\!|\Gamma \vdash \lambda \mathtt{x} : \tau.\, \mathtt{t} : \tau \Rightarrow \sigma|\!)$ by

$(\!|\Gamma \vdash \lambda \mathtt{x} : \tau.\, \mathtt{t} : \tau \Rightarrow \sigma|\!) =$ 

where d and d' are $\mathcal{S}$-transducers from $\mathbb{N}$ to $\mathbb{N}$ given by $d = u \circ (u \times \mathbb{N}) \circ Fv \circ v$ and $d' = u \circ Fu \circ (v \times \mathbb{N}) \circ v$. They are from digging of the GoI situation in Section 4.2.4. The dashed line box is an application of the "strong monoidal" F given in Section 4.2.4 followed by composition of the following isomorphism

$$F(\mathbb{N} \otimes \cdots \otimes \mathbb{N}) \xrightarrow{\cong} F\mathbb{N} \otimes \cdots \otimes F\mathbb{N} \xrightarrow{u \otimes \cdots \otimes u} \mathbb{N} \otimes \cdots \otimes \mathbb{N}.$$

GoI interpretation of algebraic effects Let $\mathbf{Per}(T)$-objects L and N be countably infinite coproducts of terminal object 1. The algebraic operations in Example 4.7 induce algebraic operations on Φ, which induce the following $\mathbf{Per}(\mathcal{S})_\Phi$-morphisms

$$\mathrm{drf} : L \to_\Phi N \qquad\qquad \mathrm{asg} : L \times N \to_\Phi 1$$

called *generic effects* in [28]. Interpretation of dereferencing $!\,\ell$ and assignment $\ell := n$ are derived from drf and asg respectively. For simplicity, we give GoI interpretation of $!\,\ell$ and $\ell := n$ for a fixed location ℓ and a fixed value n.

We interpret $- \vdash !\,\ell : \mathtt{nat}$ by an $\mathcal{S}$-transducer

$$\mathbf{drf}_\ell = (\{x_\ell, x_1, x_2, \cdots\}, c, x_\ell) : \mathbb{N} \multimap \mathbb{N}$$

where the $\mathbf{Set}_{\mathcal{S}}$-morphism c on $\{x_\ell, x_1, x_2, \cdots\} \times \mathbb{N}$ is given by

$$(c(x_\ell, n))(s) = (x_{s(\ell)}, (\!|s(\ell)|\!)(n), s),$$
$$(c(x_m, n))(s) = (x_m, (\!|m|\!)(n), s).$$

Initially, the $\mathcal{S}$-transducer $\mathbf{drf}_\ell$ looks up the global state s and behaves as an $\mathcal{S}$-transducer $(\!|s(\ell)|\!)$. At the same time, the $\mathcal{S}$-transducer $\mathbf{drf}_\ell$ stores the value $s(\ell)$ locally using its internal state. Thereafter, $\mathbf{drf}_\ell$ looks up its internal state: if an internal state is x_n, then $\mathbf{drf}_\ell$ behaves following $(\!|n|\!)$ without referring to global states.

We interpret $- \vdash \ell := n : \mathtt{unit}$ by an $\mathcal{S}$-transducer

$$\mathbf{asg}_{\ell,n} = (\{x_{\mathrm{run}}, x_{\mathrm{done}}\}, c', x_{\mathrm{run}}) : \mathbb{N} \multimap \mathbb{N}$$

where the $\mathbf{Set}_{\mathcal{S}}$-morphism c' on $\{x_{\mathrm{run}}, x_{\mathrm{done}}\} \times \mathbb{N}$ is given by

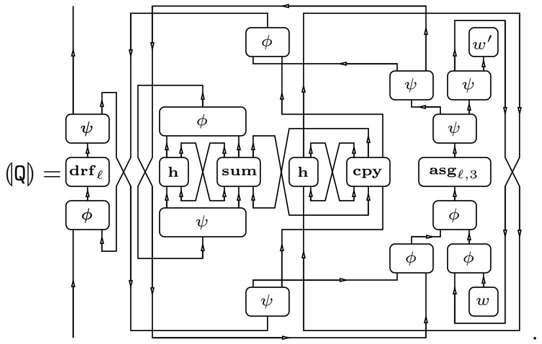
$$(c'(x_{\mathrm{run}}, m))(s) = (x_{\mathrm{done}}, (\!|\mathtt{1}|\!)(m), s[n/\ell])$$
$$(c'(x_{\mathrm{done}}, m))(s) = (x_{\mathrm{done}}, (\!|\mathtt{1}|\!)(m), s).$$

Initially, $(\{x_{\mathrm{run}}, x_{\mathrm{done}}\}, c', x_{\mathrm{run}})$ updates a global state, and thereafter, $(\{x_{\mathrm{run}}, x_{\mathrm{done}}\}, c', x_{\mathrm{run}})$ does nothing. We note that $(\!|\mathtt{1}|\!)$ in the right hand side can be any GoI interpretation of a constant.

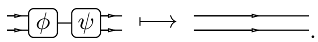
For example, the GoI interpretation of the following program

$$\mathtt{Q} = (\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + (\ell := 3); \mathtt{x})\, (!\,\ell) : \mathtt{nat}$$

where $\mathtt{t};\mathtt{s}$ is an abbreviation of $(\lambda \mathtt{x} : \mathtt{unit}.\, \mathtt{s})\, \mathtt{t}$ is



Here we simplified the canonically derived GoI interpretation. For example, since ϕ is the inverse of ψ, we can apply the following reduction to GoI interpretation:



These diagrams represent the identity on $\mathbb{N} + \mathbb{N}$, and the reduction does not affect the execution result of GoI interpretation. Correctness of the simplification can be checked by the realizability interpretation. Automatic simplification is future work.

For an input $\boldsymbol{dd}\langle n, m\rangle$ and a global state s such that $s(\ell) = 2$, the $\mathcal{S}$-transducer behaves as follows:

1. $\mathbf{drf}_\ell$ refers to the global state s and memorizes the value $s(\ell) = 2$ by means of its internal state.
2. $\mathbf{asg}_{\ell,3}$ assigns 3 to ℓ changing its internal state to x_{done}.
3. $\mathbf{sum}$ asks a value of the right occurrence of $\mathtt{x}$ in $\mathtt{x} + \mathtt{x}$.
4. $\mathbf{cpy}$ passes the query from $\mathbf{sum}$ to $\mathbf{drf}_\ell$.
5. $\mathbf{drf}_\ell$ answers 2 to the query following its internal state. In this step, $\mathbf{drf}_\ell$ does not refer to the global state $s[3/\ell]$.
6. $\mathbf{cpy}$ passes the answer from $\mathbf{drf}_\ell$ to $\mathbf{sum}$.
7. $\mathbf{sum}$ asks a value of the left occurrence of $\mathtt{x}$ in $\mathtt{x} + \mathtt{x}$.
8. $\mathbf{cpy}$ passes the query from $\mathbf{sum}$ to $\mathbf{drf}_\ell$.
9. $\mathbf{drf}_\ell$ answers 2 to the query following its internal state. In this step, $\mathbf{drf}_\ell$ does not refer to the global state $s[3/\ell]$.
10. $\mathbf{cpy}$ passes the answer from $\mathbf{drf}_\ell$ to $\mathbf{sum}$.
11. $\mathbf{sum}$ outputs $4 = 2 + 2$.

We note that without internal states, $\mathbf{drf}_\ell$ can not but refer to a global state at 5) and 9), which results in a wrong output. We only sketched computation process for lack of space. For example, we omit some access to $\mathbf{asg}_{\ell,3}$.

As a whole, the $\mathcal{S}$-transducer

$$(\!|(\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + (\ell := 3); \mathtt{x})\, (!\,\ell)|\!) \colon \mathbb{N} \rightharpoonup \mathbb{N}$$

is behaviorally equivalent to an $\mathcal{S}$-transducer

$$(\{x_\ell, x_1, x_2, \cdots\}, d, x_\ell) \colon \mathbb{N} \rightharpoonup \mathbb{N}$$

where the $\mathbf{Set}_{\mathcal{S}}$-morphism

$$d \colon \{x_\ell, x_1, x_2, \cdots\} \times \mathbb{N} \to \{x_\ell, x_1, x_2, \cdots\} \times \mathbb{N}$$

is given by

$$\begin{aligned}(d(x_\ell, n))(s) &= (x_{s(\ell)}, (\!|s(\ell) + s(\ell)|\!)(n), s[3/\ell]),\\ (d(x_m, n))(s) &= (x_m, (\!|m + m|\!)(n), s).\end{aligned}$$

We can observe that the GoI interpretation of the following program

$$(\lambda \mathtt{x} : \mathtt{nat}.\, (\ell := 3); (\mathtt{x} + \mathtt{x}))\, (!\,\ell) \colon \mathtt{nat}$$

is also behaviorally equivalent to $(\{x_\ell, x_1, x_2, \cdots\}, d, x_\ell)$.

Remark 7.1. Some readers may notice symmetries in diagrams in this paper: the top half of diagrams are mirror images of the bottom half of diagrams. This phenomenon stems from $\mathbf{Int}$-construction in the GoI workflow [2, 20] and is also observed in [22].

7.3 Memoryful GoI Interpretation of Nondeterminism

In this section, we consider nondeterminism. We can extract GoI interpretation for nondeterminism from the categorical interpretation in $\mathbf{Per}(\mathcal{P})_\Phi$ where Φ is the monad in Theorem 6.1 for $T = \mathcal{P}$. We interpret the computational lambda calculus as in Section 7.2. Interpretation of the nondeterministic choice operator $\sqcup$ is derived from the algebraic operation $\oplus^\dagger$ on Φ.

We give two examples of GoI interpretation. The first one is GoI interpretation of the nondeterministic choice $\mathsf{3} \sqcup \mathsf{5}$. The GoI interpretation $(\!|- \vdash \mathtt{3} \sqcup \mathtt{5} : \mathtt{nat}|\!) \colon \mathbb{N} \rightharpoonup \mathbb{N}$ is a $\mathcal{P}$-transducer $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!) = (\{x_{3\sqcup 5}, x_3, x_5\}, c, x_{3\sqcup 5})$ given by

$$\begin{aligned}c(x_{3\sqcup 5}, n) &= \{(x_3, (\!|\mathsf{3}|\!)(n)), (x_5, (\!|\mathsf{5}|\!)(n))\}\\ c(x_3, n) &= \{(x_3, (\!|\mathsf{3}|\!)(n))\}\\ c(x_5, n) &= \{(x_5, (\!|\mathsf{5}|\!)(n))\}.\end{aligned}$$

The $\mathcal{P}$-transducer $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ behaves like the nondeterministic Mealy machine (3): initially, the $\mathcal{P}$-transducer $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ nondeterministically chooses $(\!|\mathsf{3}|\!)$ or $(\!|\mathsf{5}|\!)$; thereafter $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ sticks to the same choice referring to its internal state.

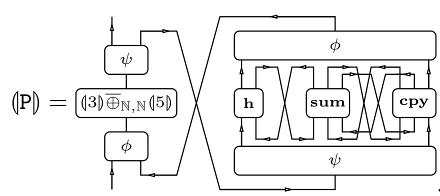
We next consider the following program:

$$\mathtt{P} = (\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, (\mathtt{3} \sqcup \mathtt{5}) \colon \mathtt{nat}.$$

We have the following equations:

$$\begin{aligned}\mathtt{P} &= ((\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, \mathtt{3}) \sqcup ((\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\, \mathtt{5})\\ &= (\mathtt{3} + \mathtt{3}) \sqcup (\mathtt{5} + \mathtt{5}) = \mathtt{6} \sqcup \mathtt{10}.\end{aligned}$$

GoI interpretation of the program $\mathtt{P}$ is



Here we also simplified canonically derived GoI interpretation.

The $\mathcal{P}$-transducer $(\!|\mathtt{P}|\!)$ behaves as follows:

1. We input $\boldsymbol{dd}\langle n, m\rangle$.
2. $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ receives an input $\boldsymbol{gdd}\langle n, m\rangle$, and the internal state nondeterministically changes to x_3 or x_5. Here we assume that $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ chooses x_3. Then $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ outputs $\boldsymbol{dgdd}\langle n, m\rangle$.
3. $\mathbf{sum}$ receives $\langle n, m\rangle$ from the leftmost input port and outputs $\langle n, m\rangle$ from the middle port to ask a value of the right occurrence of $\mathtt{x}$ in $\mathtt{x} + \mathtt{x}$.
4. $\mathbf{cpy}$ receives an input $\langle n, m\rangle$ from the leftmost port and outputs $\langle \boldsymbol{g}n, m\rangle$ from the rightmost port to get $\mathtt{x}$.
5. $(\!|\mathsf{3}|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|\mathsf{5}|\!)$ receives $\boldsymbol{dd}\langle \boldsymbol{g}n, m\rangle$. Since the internal state is x_3, it answers $\boldsymbol{dd}\langle \boldsymbol{g}n, 3\rangle$
6. $\mathbf{cpy}$ receives $\langle \boldsymbol{g}n, 3\rangle$ from the rightmost port and answers $\langle n, 3\rangle$ to $\mathbf{sum}$ via the leftmost port.
7. $\mathbf{sum}$ receives $\langle n, 3\rangle$ from the middle port and asks a value of the left occurrence of $\mathtt{x}$ in $\mathtt{x} + \mathtt{x}$ by outputting $\langle\langle n, 3\rangle, 0\rangle$ from the rightmost port.

8. **cpy** receives $\langle\langle n, 3\rangle, 0\rangle$ from the middle port and outputs $\langle \boldsymbol{d}\langle n, 3\rangle, 0\rangle$ from the rightmost port to get $\mathtt{x}$.
9. $(\!|3|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|5|\!)$ receives $\boldsymbol{dd}\langle \boldsymbol{d}\langle n, 3\rangle, 0\rangle$. Since the internal state is x_3, it answers $\boldsymbol{dd}\langle \boldsymbol{d}\langle n, 3\rangle, 3\rangle$.
10. **cpy** receives $\langle \boldsymbol{d}\langle n, 3\rangle, 3\rangle$ and answers $\langle\langle n, 3\rangle, 3\rangle$ to **sum** via the middle port.
11. **sum** receives $\langle\langle n, 3\rangle, 3\rangle$ from the rightmost port and outputs $\langle n, 6\rangle$ from the leftmost port.
12. We get an output $\boldsymbol{dd}\langle n, 6\rangle$.

In the computation, the component $\mathbf{h}$ controls interaction between $(\!|3|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|5|\!)$ and the **sum**-**cpy** fragment.

Similarly, if $(\!|3|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|5|\!)$ chooses x_5 at the first step, then we get $\boldsymbol{dd}\langle n, 10\rangle$ as an output. As a whole, we have the following behavioral equivalence:

$$(\!|(\lambda \mathtt{x} : \mathtt{nat}.\, \mathtt{x} + \mathtt{x})\,(3 \sqcup 5)|\!) \simeq^{\mathcal{P}}_{\mathbb{N},\mathbb{N}} (\!|6|\!)\overline{\oplus}_{\mathbb{N},\mathbb{N}}(\!|10|\!) = (\!|6 \sqcup 10|\!).$$

8. Conclusion

We gave a general GoI/realizability workflow that interprets the computational lambda calculus with algebraic effects as concrete state machines. In other words, our framework equips token machines with internal memories and it allows to handle generic algebraic effects. Our result provides a systematic approach to categorical GoI for algebraic effects. It would be interesting to apply our results to compiler construction (initial steps are made in [27]), GoI for additives and quantum lambda calculi.

Acknowledgments

We thank Dan Ghica, Toshiki Kataoka and anonymous reviewers for useful comments and discussions. The first author is supported by Grants-in-Aid for Young Scientists (B) No. 26730004, JSPS. The second and third authors are supported by Grants-in-Aid for Young Scientists (A) No. 24680001, JSPS, and by Aihara Innovative Mathematical Modeling Project, FIRST Program, JSPS/CSTP.

References

[1] S. Abramsky. Retracing some paths in process algebra. In *CONCUR*, pages 1–17, 1996.

[2] S. Abramsky, E. Haghverdi, and P. Scott. Geometry of interaction and linear combinatory algebras. *Math. Struct. in Comp. Sci.*, 12:625–665, 2002.

[3] L. Aceto, W. Fokkink, and C. Verhoef. Structural operational semantics. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier, 2001.

[4] C. Baier, M. Sirjani, F. Arbab, and J. J. M. M. Rutten. Modeling component connectors in Reo by constraint automata. *Science of Computer Programming*, 61(2):75–113, 2006.

[5] L. S. Barbosa. Towards a calculus of state-based software components. *Journal of Universal Computer Science*, 9(8):891–909, 2003.

[6] U. Dal Lago and U. Schöpp. Functional programming in sublinear space. In *Programming Languages and Systems*, volume 6012 of *LNCS*, pages 205–225. Springer Berlin Heidelberg, 2010.

[7] V. Danos and L. Regnier. Reversible, irreversible and optimal λ-machines: Extended abstract. *ENTCS*, 3(0):40–60, 1996.

[8] O. Fredriksson and D. R. Ghica. Seamless distributed computing from the geometry of interaction. In C. Palamidessi and M. D. Ryan, editors, *Trustworthy Global Computing*, volume 8191 of *Lecture Notes in Computer Science*, pages 34–48. Springer Berlin Heidelberg, 2013.

[9] D. R. Ghica. Geometry of synthesis: a structured approach to VLSI design. In M. Hofmann and M. Felleisen, editors, *POPL*, pages 363–375. ACM, 2007. ISBN 1-59593-575-4.

[10] J.-Y. Girard. Geometry of interaction 1: Interpretation of system F. In S. V. 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.

[11] G. Gonthier, M. Abadi, and J.-J. Lévy. The geometry of optimal lambda reduction. In R. Sethi, editor, *POPL*, pages 15–26. ACM Press, 1992. ISBN 0-89791-453-8.

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

[13] M. Hasegawa. On traced monoidal closed categories. *Math. Struct. in Comp. Sci.*, 19:217–244, 4 2009. ISSN 1469-8072.

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

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

[16] N. Hoshino. A modified GoI interpretation for a linear functional programming language and its adequacy. In M. Hofmann, editor, *FOSSACS*, volume 6604 of *Lecture Notes in Computer Science*, pages 320–334. Springer, 2011.

[17] B. Jacobs, editor. *Categorical Logic and Type Theory*, volume 141 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 19999.

[18] B. Jacobs. From coalgebraic to monoidal traces. *ENTCS*, 264(2):125–140, 2010.

[19] B. Jacobs. Introduction to coalgebra. towards mathematics of states and observations, Version 2.0 2012.

[20] A. Joyal, R. Street, and D. Verity. Traced monoidal categories. *Mathematical Proceedings of the Cambridge Philosophical Society*, 119: 447–468, 4 1996.

[21] B. Klin. Bialgebraic methods and modal logic in structural operational semantics. *Information and Computation*, 207(2):237–257, 2009.

[22] Y. Lafont. Interaction combinators. *Information and Computation*, 137(1):69–101, 1997. .

[23] O. Laurent. A token machine for full geometry of interaction. In *TLCA*, pages 283–297, 2001.

[24] J. Longley. *Realizability Toposes and Language Semantics*. PhD thesis, University of Edinburgh, 1994.

[25] I. Mackie. The geometry of interaction machine. In R. K. Cytron and P. Lee, editors, *POPL*, pages 198–208. ACM Press, 1995. ISBN 0-89791-692-1.

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

[27] K. Muroya, T. Kataoka, I. Hasuo, and N. Hoshino. Compiling effectful terms to transducers: Prototype implementation of memoryful geometry of interaction. Submitted, 2014.

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

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

[30] U. Schöpp. On interaction, continuations and defunctionalization. In *TLCA*, volume 7941 of *LNCS*, pages 205–220. Springer Berlin Heidelberg, 2013.

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

[32] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *LICS*, pages 280–291. IEEE Computer Society, 1997.

[33] A. Yoshimizu, I. Hasuo, C. Faggian, and U. Dal Lago. Measurements in proof nets as higher-order quantum circuits. In *ESOP*, 2014. To appear.

A. A Proof of Lemma 4.3

A $\mathbf{Cppo}$-enriched cartesian category $\mathcal{C}$ is a $\mathbf{Cppo}$-enriched category $\mathcal{C}$ such that the underlying category has finite products and the product functor $f \times g \colon A \times C \to B \times D$ is continuous on $f \colon A \to B$ and $g \colon C \to D$. For $\mathcal{C}$-morphisms $\{f_i \colon A \to B_i\}_{i=1,2,\cdots,n}$, we write $\langle f_1, \cdots, f_n \rangle \colon A \to B_1 \times \cdots \times B_n$ for the tupling of $\{f_i\}_{i=1,2,\cdots,n}$. We write $\pi_{A,B} \colon A \times B \to A$ for the first projection, and we write $\pi'_{A,B} \colon A \times B \to B$ for the second projection. When $\mathcal{C}$ is a $\mathbf{Cppo}$-enriched cartesian category, we write $\perp_{A,B} \colon A \to B$ for the bottom element in $\mathcal{C}(A, B)$.

Lemma A.1. *If $(\mathcal{C}, \times, 1)$ is a $\mathbf{Cppo}$-enriched cartesian category and the bottom morphisms $\perp_{A,B} \colon A \to B$ satisfy the following condition:*

- *$\perp_{A,B} \circ f = \perp_{A',B}$ for all $\mathcal{C}$-morphisms $f \colon A' \to A$*

then $(\mathcal{C}, \times, 1)$ has a Conway operator.

Proof. For a $\mathcal{C}$-morphism $f \colon A \times B \to B$, we define a $\mathcal{C}$-morphism $\mathrm{fix}_{A,B}(f) \colon A \to B$ by

$$\mathrm{fix}_{A,B}(f) = \bigvee_{n \geq 1} \mathrm{fix}^{(n)}_{A,B}(f)$$

where

$$\mathrm{fix}^{(n)}_{A,B}(f) = \perp_{A,B}, \quad \mathrm{fix}^{(n+1)}_{A,B}(f) = f \circ \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(f) \rangle.$$

The operator fix is a fixed point operator:

$$\begin{aligned} f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(f) \rangle &= \bigvee_{n \geq 1} f \circ \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(f) \rangle \\ &= \bigvee_{n \geq 2} \mathrm{fix}^{(n)}_{A,B}(f) \\ &= \mathrm{fix}_{A,B}(f). \end{aligned}$$

For a $\mathcal{C}$-morphism $f \colon A \times B \to B$ and a $\mathcal{C}$-morphism $g \colon A' \to A'$, we can show that

$$\mathrm{fix}^{(n)}_{A,B}(f) \circ g = \mathrm{fix}^{(n)}_{A',B}(f \circ (g \times B))$$

by induction on n. Hence, fix is natural:

$$\mathrm{fix}_{A,B}(f) \circ g = \mathrm{fix}_{A',B}(f \circ (g \times B)).$$

For $\mathcal{C}$-morphisms $f \colon A \times B \to C$ and $g \colon A \times C \to B$, we have

$$\begin{aligned} & f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \\ &= \bigvee_{n \geq 1} f \circ \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \\ &= \bigvee_{n \geq 1} f \circ \langle \pi_{A,C}, g \rangle \circ \langle \pi_{A,B}, f \rangle \circ \\ & \qquad \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \\ &= \bigvee_{n \geq 1} f \circ \langle \pi_{A,C}, g \rangle \circ \langle \mathrm{id}_A, f \circ \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \rangle \\ &= f \circ \langle \pi_{A,C}, g \rangle \circ \langle \mathrm{id}_A, f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \rangle. \end{aligned}$$

Therefore,

$$f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \geq \mathrm{fix}_{A,C}(f \circ \langle \pi_{A,C}, g \rangle)).$$

On the other hand, we can show that

$$f \circ \langle \mathrm{id}_A, \mathrm{fix}^{(n)}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle \leq \mathrm{fix}^{(n+1)}_{A,C}(f \circ \langle \pi_{A,C}, g \rangle))$$

by induction on n. Hence, the operator fix is dinatural:

$$f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(g \circ \langle \pi_{A,B}, f \rangle) \rangle = \mathrm{fix}_{A,C}(f \circ \langle \pi_{A,C}, g \rangle)).$$

For a $\mathcal{C}$-morphism $f \colon A \times B \times B \to B$,

$$\begin{aligned} & f \circ \langle \pi_{A,B}, \pi'_{A,B}, \pi'_{A,B} \rangle \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)) \rangle \\ &= f \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)), \\ & \qquad \mathrm{fix}_{A \times B,B}(f) \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)) \rangle \rangle \\ &= f \circ \langle \mathrm{id}_{A \times B}, \mathrm{fix}_{A \times B,B}(f) \rangle \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)) \rangle \\ &= \mathrm{fix}_{A \times B,B}(f) \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)) \rangle \\ &= \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)). \end{aligned}$$

Therefore,

$$\mathrm{fix}_{A,B}(f \circ \langle \pi_{A,B}, \pi'_{A,B}, \pi'_{A,B} \rangle) \leq \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)).$$

On the other hand, we can show that

$$\mathrm{fix}^{(n)}_{A \times B,B}(f) \circ \langle \mathrm{id}_A, \mathrm{fix}_{A,B}(f \circ \langle \pi_{A,B}, \pi'_{A,B}, \pi'_{A,B} \rangle) \rangle \leq \mathrm{fix}_{A,B}(f \circ \langle \pi_{A,B}, \pi'_{A,B}, \pi'_{A,B} \rangle)$$

by induction on n. Hence, fix satisfies diagonal property:

$$\mathrm{fix}_{A,B}(f \circ \langle \pi_{A,B}, \pi'_{A,B}, \pi'_{A,B} \rangle) = \mathrm{fix}_{A,B}(\mathrm{fix}_{A \times B,B}(f)).$$

We used left strictness in naturality and diagonal property. □

Lemma A.2 (Lemma 4.3)**.** *If the Kleisli category $\mathbf{Set}_T$ is a $\mathbf{Cppo}$-enriched cocartesian category and the bottom morphisms $\perp_{A,B} \colon A \to_T B$ satisfy the following conditions:*

- *$f \circ_T \perp_{A,B} = \perp_{A,B'}$ for all $f \colon B \to_T B'$*
- *$\perp_{A,B} \circ_T g^* = \perp_{A',B}$ for all $g \colon A' \to A$*

then $(\mathbf{Set}_T, +, \emptyset)$ satisfies Requirement 4.2.

Proof. By the dual of Lemma A.1 and the bijective correspondence between Conway operators and trace operators, we see that $\mathbf{Set}_T$ has a trace operator: for $f \colon A + C \to_T B + C$, we define $\mathrm{tr}^C_{A,B}(f) \colon A \to_T B$ by

$$\gamma^*_B \circ_T (B + \mathrm{iter}_{C,B}(f)) \circ_T f \circ_T \mathrm{inl}^*_{A,C}$$

where $\mathrm{iter}_{B,C}(f) \colon C \to_T B$ is given by

$$\begin{aligned} \mathrm{iter}_{C,B}(f) &= \bigvee_{n \geq 1} \mathrm{iter}^{(n)}_{C,B}(f) \\ \mathrm{iter}^{(1)}_{C,B}(f) &= \perp_{C,B} \\ \mathrm{iter}^{(n+1)}_{C,B}(f) &= \gamma^*_B \circ_T (B + \mathrm{iter}^{(n)}_{C,B}(f)) \circ_T f \circ_T \mathrm{inr}^*_{A,C}. \end{aligned}$$

For $f \colon A + C \to_T B + C$, $g \colon A + D \to_T B + D$ and $h \colon C \to D$, if $(B + h^*) \circ_T f = g \circ_T (A + h^*)$, then we can show

$$\mathrm{iter}_{D,B}(g) \circ_T h^* = \mathrm{iter}_{C,B}(f)$$

by induction on n. Then

$$\begin{aligned} & \mathrm{tr}^C_{A,B}(f) \\ &= \gamma^*_B \circ_T (B + \mathrm{iter}_{C,B}(f)) \circ_T f \circ_T \mathrm{inl}^*_{A,C} \\ &= \gamma^*_B \circ_T (B + (\mathrm{iter}_{D,B}(g) \circ_T h^*)) \circ_T f \circ_T \mathrm{inl}^*_{A,C} \\ &= \mathrm{iter}_{D,B}(g) \circ_T g \circ_T \mathrm{inl}^*_{A,D} \\ &= \mathrm{tr}^D_{A,B}(g). \end{aligned}$$

Hence, $(\mathbf{Set}_T, +, \emptyset)$ satisfies Requirement 4.2. □

B. Proofs in Section 5

Lemma B.1. *For a $\mathbf{Set}_T$-morphism $f \colon A + C \to_T B + C$ and a set D, it holds that*

$$\mathrm{tr}^{D \times C}_{D \times A, D \times B}(g) = D \otimes \mathrm{tr}^C_{A,B}(f)$$

where $g = (\delta^*_{D,B,C})^{-1} \circ_T (D \otimes f) \circ_T \delta^*_{D,A,C}$.

Proof. For all maps $d\colon 1 \to D$, we have

$$g \circ (d \times A + d \times C) = T(d \times B + d \times C) \circ f.$$

By uniformity, we obtain

$$\mathrm{tr}^{D\times C}_{D\times A, D\times B}(g) \circ (d \times A) = (D \otimes \mathrm{tr}^{C}_{A,B}(f)) \circ (d \times A).$$

Since $\mathbf{Set}$ is well-pointed, $\mathrm{tr}^{D\times C}_{D\times A, D\times B}(g)$ is equal to $D \otimes \mathrm{tr}^{C}_{A,B}(f)$. □

Proposition B.2. *The traced symmetric monoidal category* $(\mathbf{Res}(T), \boxplus, \emptyset, \mathrm{Tr})$ *and the GoI situation*

$$(\mathbf{Res}(T), \mathbb{N}, F, J_0\phi, J_0\psi, J_0u, J_0v)$$

are well-defined.

Proof. It is straightforward to check that the composition $\circ$, the monoidal product $\boxplus$, the functor F are compatible with the behavioral equivalence. We can show that Tr is compatible with the behavioral equivalence by uniformity of the trace operator tr. It is easy to check that $\mathbf{Res}(T)$ forms a symmetric monoidal category. Yanking, exchange and superposing of Tr follow from these of tr. Tightening follows from tightening of tr and Lemma B.1. It is straightforward to check naturality of retractions in Section 4.2.4. It follows from Lemma B.1 and uniformity of tr that F is a traced symmetric monoidal functor. □

Theorem B.3 (Theorem 5.3)**.** *Let α be an I-ary algebraic operation on T. The operator $\overline{\alpha}$ is natural, and* tr *distributes over $\overline{\alpha}$ modulo the behavioral equivalence:*

- *For all T-transducers $(X, c, x)\colon B \twoheadrightarrow B'$, for all families of T-transducers $\{(Y_i, d_i, y_i)\colon A \twoheadrightarrow B\}_{i\in I}$ and for all maps $h\colon A' \to A$, it holds that*

$$\begin{aligned}&(X, c, x) \circ \overline{\alpha}_{A,B}\{(Y_i, d_i, y_i)\}_{i\in I} \circ J_0 h \\ &\quad \simeq^{T}_{A',B'} \overline{\alpha}_{A',B'}\{(X, c, x) \circ (Y_i, d_i, y_i) \circ J_0 h\}_{i\in I}.\end{aligned}$$

- *For all families of T-transducers $\{(X_i, c_i, x_i)\}_{i\in I}$ from $A+C$ to $B+C$, it holds that*

$$\begin{aligned}&\mathrm{Tr}^{C}_{A,B}(\overline{\alpha}_{A+C,B+C}\{(X_i, c_i, x_i)\}_{i\in I}) \\ &\qquad\qquad \simeq^{T}_{A,B} \overline{\alpha}_{A,B}\{\mathrm{Tr}^{C}_{A,B}(X_i, c_i, x_i)\}_{i\in I}.\end{aligned}$$

Proof. Let α be an I-ary algebraic operation on T. It is easy to check that a map

$$x + X \times \coprod_{i\in I} Y_i \colon 1 + X \times \coprod_{i\in I} Y_i \to X \times \left(1 + \coprod_{i\in I} Y_i\right)$$

is a homomorphism from $\overline{\alpha}_{A',B'}\{(X, c, x) \circ (Y_i, d_i, y_i) \circ J_0 h\}_{i\in I}$ to $(X, c, x) \circ \overline{\alpha}_{A,B}\{(Y_i, d_i, y_i)\}_{i\in I} \circ J_0 h$. We prove the second equivalence using string diagrams in the traced symmetric monoidal category $(\mathbf{Set}_T, +, \emptyset, \mathrm{tr})$. For a family of T-transducers $\{(X_i, c_i, x_i)\colon A + C \twoheadrightarrow B + C\}_{i\in I}$, the T-transducer

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

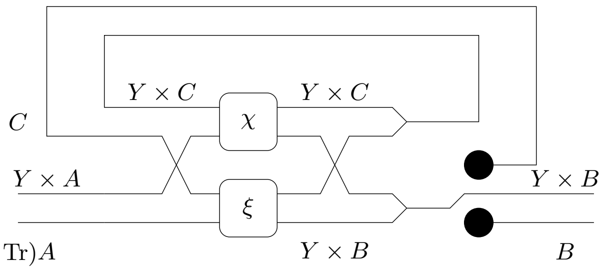
is given by

$$(1 + Y, d, \mathrm{inl}_{1,Y}) : A \twoheadrightarrow B$$

where Y is the coproduct

$$Y = \coprod_{i\in I} X_i \xleftarrow{\mathrm{inj}_i} X_i$$

and d is given by the following string diagram:



where

- The links of the form ≻— are the codiagonal morphisms.
- The black circles are unique $\mathbf{Set}_T$-morphisms from the empty set.
- $\xi\colon A + C \to_T Y \times B + Y \times C$ is

$$\begin{aligned}&(\delta^{*}_{Y,B,C})^{-1} \circ_T \alpha_{A+C, Y\times(B+C)} \\ &\quad \{(\mathrm{inj}^{*}_{i} \otimes (B + C)) \circ_T c_i \circ_T (x^{*}_{i} \otimes (A + C))\}_{i\in I}.\end{aligned}$$

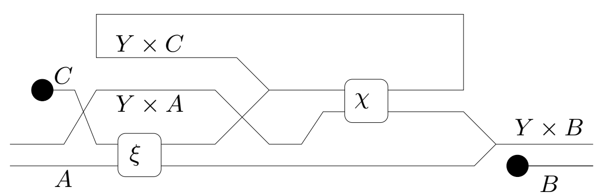
- $\chi\colon Y \times A + Y \times C \to_T Y \times B + Y \times C$ is

$$\theta_{B,C} \circ_T \left(\coprod_{1\le i\le n} c_i\right) \circ_T \theta^{-1}_{A,C}$$

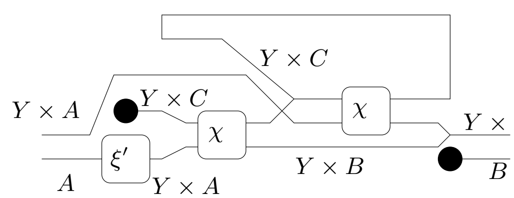
  where $\theta_{A,B}$ is the following canonical isomorphism

$$\theta_{A,B}\colon \coprod_{i\in I}(X_i \times (A + B)) \xrightarrow{\cong}_T Y \times A + Y \times B.$$

By the axioms of trace operator, d is equal to



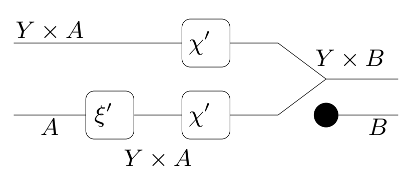
By naturality of the algebraic operation α, this is equal to



where $\xi' : A \to_T Y \times A$ is

$$\alpha_{A, Y\times A}\{(\mathrm{inj}^{*}_{i} \circ_T x^{*}_{i}) \otimes A\}_{i\in I}.$$

By the axioms of trace operators, this is equal to



where $\chi' = \mathrm{tr}^{Y\times C}_{Y\times A, Y\times B}(\chi)$. By naturality of α and uniformity of tr, it follows that the T-transducer $(1 + Y, d, \mathrm{inl}_{1,Y})$ is equal to

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

In particular, the trace operator distributes over $\overline{\alpha}$ modulo the behavioral equivalence. □

C. Realizability and Categorical Models

C.1 Partial Equivalence Relation

The set $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ has another binary operation given by

$$a \cdot b = \mathrm{Tr}^{\mathbb{N}}_{\mathbb{N},\mathbb{N}}((\mathbb{N} \boxplus b) \circ J_0\psi \circ a \circ J_0\phi).$$

For all algebraic operations α on T, the binary application $\cdot$ is also right distributive over $\overline{\alpha}_{\mathbb{N},\mathbb{N}}$. The following lemmas are basic in realizability argument. For proofs of Lemma C.1 and Lemma C.3, see [2] and [31].

Lemma C.1 (Combinatory Completeness)**.** *For every formal expression $e(x, x_1, x_2, \cdots, x_n)$ generated by*

- *variables $x, x_1, x_2, \cdots, x_n$*
- *elements in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$*
- *binary application symbols $(-) \bullet (-)$ and $(-) \cdot (-)$,*

there exists a formal expression $\lambda x.\, e(x, x_1, \cdots, x_n)$ generated by

- *variables $x_1, x_2, \cdots, x_n$*
- *elements in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$*
- *binary application symbols $(-) \bullet (-)$ and $(-) \cdot (-)$,*

such that

$$((\lambda x.\, e(x, x_1, \cdots, x_n))[a_1/x_1, \cdots, a_n/x_n]) \bullet a = e(a, a_1, \cdots, a_n)$$

for all $a, a_1, \cdots, a_n \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ where $e(a, a_1, \cdots, a_n)$ is an element in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ obtained by replacing all x_i in e by a_i. We write $\lambda xy \cdots z.\, e$ for $\lambda x.\, (\lambda y.\, \cdots (\lambda z.\, e) \cdots)$

Definition C.2. Let $e(x, x_1, x_2, \cdots, x_n)$ be a formal expression generated by

- variables $x, x_1, x_2, \cdots, x_n$
- elements in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$
- binary application symbols $\bullet$ and $\cdot$

we say that x appears *linearly* in e when x occurs exactly once in e and e is not of the form $(\cdots \bullet (\cdots x \cdots))$.

Lemma C.3 (Linear Combinatory Completeness)**.** *For every formal expression $e(x, x_1, x_2, \cdots, x_n)$ generated by*

- *variables $x, x_1, x_2, \cdots, x_n$*
- *elements in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$*
- *binary application symbols $(-) \bullet (-)$ and $(-) \cdot (-)$,*

if x appears linearly in e, then there exists a formal expression $\lambda^\circ x.\, e(x, x_1, \cdots, x_n)$ generated by

- *variables $x_1, x_2, \cdots, x_n$*
- *elements in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$*
- *binary application symbols $(-) \bullet (-)$ and $(-) \cdot (-)$,*

such that

$$((\lambda^\circ x.\, e(x, x_1, \cdots, x_n))[a_1/x_1, \cdots, a_n/x_n]) \cdot a = e(a, a_1, \cdots, a_n)$$

for all $a, a_1, \cdots, a_n \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ where $e(a, a_1, \cdots, a_n)$ is an element in $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ obtained by replacing all x_i in e by a_i.

We define a category $\mathbf{Per}(T)$ by:

- An object is a partial equivalence relation (per) on $\mathbf{Res}(T)(\mathbb{N},\mathbb{N})$.
- A $\mathbf{Per}(T)$-morphism from R to S is an equivalence class of the following per:
$$R \Rightarrow S = \{(a, a') \mid \forall (b, b') \in R.\, (a \bullet b, a' \bullet b') \in S\}.$$

For a $\mathbf{Per}(T)$-morphism $f\colon R \to S$, a *realizer* of f is a representative of f. If r is a realizer of f, then we say that r *realizes* f. When r realizes a $\mathbf{Per}(T)$-morphism from R to S, we write $[r]\colon R \to S$ for the morphism from R to S realized by r.

Proposition C.4 ([24])**.** *$\mathbf{Per}(T)$ is a bicartesian closed category with finite limits.*

We describe the bicartesian closed structure of $\mathbf{Per}(T)$:

- The terminal object is $\{(\lambda x.\, x, \lambda x.\, x)\}$.
- The initial object is the empty set.
- The cartesian product $R \times S$ is
$$\{(a, a') \mid (a \bullet \mathsf{T}, a' \bullet \mathsf{T}) \in R \wedge (b \bullet \mathsf{F}, b' \bullet \mathsf{F}) \in S\}$$
where $\mathsf{T} = \lambda xy.\, x$ and $\mathsf{F} = \lambda xy.\, y$.
- The exponential from R to S is $R \Rightarrow S$.
- The coproduct $R + S$ is
$$\{(a, a') \mid a \bullet \mathsf{T} = a' \bullet \mathsf{T} = \mathsf{T} \wedge (a \bullet \mathsf{F}, a' \bullet \mathsf{F}) \in R\} \cup \{(b, b') \mid b \bullet \mathsf{T} = b' \bullet \mathsf{T} = \mathsf{F} \wedge (b \bullet \mathsf{F}, b' \bullet \mathsf{F}) \in S\}.$$
- The equalizer of $\mathbf{Per}(T)$-morphisms $f, f'\colon R \to S$ is
$$m\colon \{(a, a') \in R \mid (r \bullet a, r' \bullet a') \in S\} \to S$$
where r is a realizer of f and r' is a realizer of f'. The $\mathbf{Per}(T)$-morphism m is realized by $\lambda x.\, x$.

Proposition C.5. *$\mathbf{Per}(T)$ has countable products and countable coproducts.*

Proof. Since $\mathbf{Per}(T)$ has finite limits and colimits, we give countably infinite products and coproducts. Let

$$\{(X_i, c_i, x_i)\}_{i \in \mathbb{N}}$$

be a family of T-transducers from $\mathbb{N}$ to $\mathbb{N}$. For $n \in \mathbb{N}$, we write ξ_n for the following canonical bijection

$$\xi_n\colon \prod_{n \in \mathbb{N}} X_n \to \left(\prod_{i \in \mathbb{N}\setminus\{n\}} X_i\right) \times X_n.$$

We define a T-transducer $\langle X_i, c_i, x_i\rangle_{i \in \mathbb{N}}\colon \mathbb{N} \multimap \mathbb{N}$ to be

$$\left(\prod_{i \in \mathbb{N}} X_i, d, \prod_{i \in \mathbb{N}} x_i\right)$$

where $d\colon \left(\prod_{i\in\mathbb{N}} X_i\right) \times \mathbb{N} \to_T \left(\prod_{i\in\mathbb{N}} X_i\right) \times \mathbb{N}$ is a unique morphism such that $d \circ_T \left(\left(\prod_{i\in\mathbb{N}} X_i\right) \otimes \kappa^*_{\langle m,n\rangle}\right)$ is equal to

$$((\xi^*_m)^{-1} \otimes (u^* \circ_T (\kappa^*_m \otimes \mathbb{N}))) \circ_T \left(\left(\prod_{i\in\mathbb{N}\setminus\{m\}} X_i\right) \otimes c_m\right) \circ_T (\xi^*_m \otimes \kappa^*_n).$$

The construction

$$\{(X_i, c_i, x_i)\colon \mathbb{N} \multimap \mathbb{N}\}_{i\in\mathbb{N}} \mapsto \langle X_i, c_i, x_i\rangle_{i\in\mathbb{N}}\colon \mathbb{N} \multimap \mathbb{N}$$

is compatible with the behavioral equivalence. We define

$$\mathsf{M}\colon \mathbf{Res}(T)(\mathbb{N},\mathbb{N})^{\mathbb{N}} \to \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$$

to be the induced map. For $n \in \mathbb{N}$, we define a map $p_n\colon \mathbb{N} + \mathbb{N} \to \mathbb{N} + \mathbb{N}$ by

$$p_n(\mathrm{inl}_{\mathbb{N},\mathbb{N}}(m)) = \mathrm{inr}_{\mathbb{N},\mathbb{N}}(\langle 0, \langle n, m\rangle\rangle)$$
$$p_n(\mathrm{inr}_{\mathbb{N},\mathbb{N}}(\langle k, \langle l, m\rangle\rangle)) = \mathrm{inl}_{\mathbb{N},\mathbb{N}}(m).$$

We define $\mathsf{P}_n \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ to be the equivalence class of $J_0(\phi \circ p_n \circ \psi)$. It is straightforward to check that P_n satisfies

$$\mathsf{P}_n \bullet \mathsf{M}\{a_i\}_{i\in\mathbb{N}} = a_n.$$

For a family of $\mathbf{Per}(T)$-objects $\{R_i\}_{i\in\mathbb{N}}$, we define a $\mathbf{Per}(T)$-object $\prod_{i\in\mathbb{N}} R_i$ to be

$$\{(a,a') \mid \forall i \in \mathbb{N}.\, (a \bullet \mathsf{P}_i, a' \bullet \mathsf{P}_i) \in R_i\}.$$

The object $\prod_{i\in\mathbb{N}} R_i$ together with projections

$$\left\{[\lambda x.\, x \bullet \mathsf{P}_n] \colon \prod_{i\in\mathbb{N}} R_i \to R_n\right\}_{n\in\mathbb{N}}$$

forms a product of the family $\{R_i\}_{i\in\mathbb{N}}$. Given a family of $\mathbf{Per}(T)$-morphisms $\{f_i \colon S \to R_i\}_{i\in\mathbb{N}}$, the tupling $g\colon S \to \prod_{i\in\mathbb{N}} R_i$ is the morphism realized by

$$\lambda xk.\, k \bullet (\mathsf{M}\{r_i\}_{i\in\mathbb{N}}) \bullet x$$

where r_i is a realizer of f_i. We define a $\mathbf{Per}(T)$-object $\coprod_{i\in\mathbb{N}} R_i$ to be

$$\bigcup_{i\in\mathbb{N}} \{(a,a') \mid a \bullet \mathsf{T} = a' \bullet \mathsf{T} = \mathsf{P}_i \wedge (a \bullet \mathsf{F}, a' \bullet \mathsf{F}) \in R_i\}.$$

The object $\coprod_{i\in\mathbb{N}} R_i$ together with injections

$$\left\{[\lambda xk.\, k \bullet \mathsf{P}_n \bullet x] \colon R_n \to \coprod_{i\in\mathbb{N}} R_i\right\}_{n\in\mathbb{N}}$$

forms a coproduct of the family $\{R_i\}_{i\in\mathbb{N}}$. Let $\{f_i \colon R_i \to S\}_{i\in\mathbb{N}}$ be a family of $\mathbf{Per}(T)$-morphisms. The cotupling $g\colon \coprod_{i\in\mathbb{N}} R_i \to S$ is the morphism realized by

$$\lambda x.\, x \bullet \mathsf{T} \bullet (\mathsf{M}\{r_i\}_{i\in\mathbb{N}}) \bullet (x \bullet \mathsf{F})$$

where r_i is a realizer of f_i. □

Since $\mathbf{Per}(T)$ has countable coproducts, $\mathbf{Per}(T)$ has a natural number object. We explicitly describe a natural number object in $\mathbf{Per}(T)$. For a natural number n, we define $\mathsf{K}_n \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ to be the equivalence class of $J_0 k_n$ where the map $k_n \colon \mathbb{N} \to \mathbb{N}$ is given by $k_n(m) = n$.

Lemma C.6. *A $\mathbf{Per}(T)$-object $N = \{(\mathsf{K}_n, \mathsf{K}_n) \mid n \in \mathbb{N}\}$ is a natural number object.*

Proof. As shown in [24], the following $\mathbf{Per}(T)$-object is a natural number object:

$$N' = \{(\mathsf{K}'_n, \mathsf{K}'_n) \mid n \in \mathbb{N}\}$$

where

$$\mathsf{K}'_n = \lambda f x.\, \overbrace{f \bullet (\cdots (f}^{n} \bullet x) \cdots)).$$

We define $s\colon \mathbb{N} + \mathbb{N} \to \mathbb{N} + \mathbb{N}$ by

$$s(\mathrm{inl}_{\mathbb{N},\mathbb{N}}(n)) = \mathrm{inr}_{\mathbb{N},\mathbb{N}}\langle 0, n\rangle$$
$$s(\mathrm{inr}_{\mathbb{N},\mathbb{N}}\langle n, m\rangle) = \mathrm{inl}_{\mathbb{N},\mathbb{N}}(m+1).$$

Let $\mathsf{succ} \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ be the equivalence class of $J_0(\phi\circ s\circ\psi)$. Then $\mathsf{K}'_n \bullet \mathsf{succ} \bullet \mathsf{K}_0$ is equal to K_n. Hence, $\lambda n.\, n \bullet \mathsf{succ} \bullet \mathsf{K}_0$ realizes a $\mathbf{Per}(T)$-morphism from N' to N. Let

$$(X_n, c_n, x_n) \colon \mathbb{N} \rightharpoonup \mathbb{N}$$

be a T-transducer that represents K'_n. By the proof of combinatory completeness and by the definition of the GoI situation

$$(\mathbf{Res}(T), \mathbb{N}, F, [J_0\phi], [J_0\psi], [J_0 u], [J_0 v]),$$

we can assume that $X_n = \{*\}$ and $x_n = *$. Let $c'_n \colon \mathbb{N} \to \mathbb{N}$ be a map such that $(*, c'_n(m)) = c_n(*, m)$. We define a map $d\colon \mathbb{N} \rightharpoonup \mathbb{N}$ by

$$d(\boldsymbol{g}n) = \boldsymbol{d}\langle n, 0\rangle$$
$$d(\boldsymbol{d}\langle n, m\rangle) = \boldsymbol{g}(c'_m(n)).$$

Then $[J_0 d]$ realizes a $\mathbf{Per}(T)$-morphism from N to N'. It is easy to check that $[J_0 d]$ is the inverse of the $\mathbf{Per}(T)$-morphism realized by $\lambda n.\, n \bullet \mathsf{succ} \bullet \mathsf{K}_0$. □

C.2 Strong Monad on $\mathbf{Per}(T)$

Definition C.7. We say that a $\mathbf{Per}(T)$-object R is *closed* when the following statement

$$\forall \alpha : \text{algebraic operation on } T.\, \forall\{(a_i, a'_i) \in R\}_{i\in\mathrm{ary}(\alpha)}.$$
$$(\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{a_i\}_{i\in\mathrm{ary}(\alpha)}, \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{a'_i\}_{i\in\mathrm{ary}(\alpha)}) \in R$$

is true.

Let R be a closed $\mathbf{Per}(T)$-object, and let α be an I-ary algebraic operation on T. Then $\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}$ realizes a $\mathbf{Per}(T)$-morphism

$$[\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}] \colon R^I \to R$$

where $\mathsf{Q}_{i,I}$ is a realizer of the i-th projection from the I-fold product R^I to R. The $\mathbf{Per}(T)$-morphism $[\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}]$ is independent of our choice of realizers $\mathsf{Q}_{i,I}$.

Definition C.8. For closed $\mathbf{Per}(T)$-objects R and S, a $\mathbf{Per}(T)$-morphism $f\colon R \to S$ is *linear* when the following diagram commutes for all algebraic operations α on T:

$$\begin{array}{ccc} R^I & \xrightarrow{\quad f^I \quad} & S^I \\ {\scriptstyle [\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}]}\downarrow & & \downarrow{\scriptstyle [\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}]} \\ R & \xrightarrow[\quad f \quad]{} & S \end{array}$$

where I is the arity of α.

We define $\mathbf{CPer}(T)$ to be the $\mathbf{Per}(T)$-enriched category consisting of closed $\mathbf{Per}(T)$-objects and hom-objects

$$R \multimap S = \{(a,b) \in R \Rightarrow S \mid [a] \colon R \to S \text{ is linear}\}.$$

Proposition C.9. *The inclusion functor $U\colon \mathbf{CPer}(T) \to \mathbf{Per}(T)$ is a $\mathbf{Per}(T)$-enriched right adjunction.*

Proof. We define $\mathsf{L} \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})$ to be $\lambda x.\, \lambda^\circ k.\, k \bullet x$. It is easy to see that the set of $\mathbf{CPer}(T)$-objects is closed under small intersection. For a $\mathbf{Per}(T)$-object R, we define a $\mathbf{CPer}(T)$-object ΦR to be the least $\mathbf{CPer}(T)$-object that includes the following $\mathbf{Per}(T)$-object:

$$R' = \{(\mathsf{L} \bullet a, \mathsf{L} \bullet a') \mid (a,a') \in R\}.$$

Since ΦR includes R', the combinator L realizes a $\mathbf{Per}(T)$-morphism from R to ΦR. We show that $[\mathsf{L}] \colon R \to \Phi R$ is a unit of the right adjoint functor U. Let $[r] \colon R \to S$ be a $\mathbf{Per}(T)$-morphism from R to a $\mathbf{CPer}(T)$-object S. By the right distributivity of $\cdot$ over $\overline{\alpha}_{\mathbb{N},\mathbb{N}}$ for all algebraic operations α, the following $\mathbf{Per}(T)$-object

$$\{(a,a') \mid (a \cdot r, a' \cdot r) \in S\}$$

is closed and includes R'. Therefore, by the definition of ΦR, we see that $\lambda x.\, x \cdot r$ realizes a $\mathbf{Per}(T)$-morphism from ΦR to S. Linearity of $[\lambda x.\, x \cdot r] \colon \Phi R \to S$ follows from the right distributivity of $\cdot$ over $\overline{\alpha}_{\mathbb{N},\mathbb{N}}$ for all algebraic operations α. Let $f, g\colon \Phi R \to S$ be $\mathbf{CPer}(T)$-morphisms such that

$$R \xrightarrow{[\mathsf{L}]} \Phi R \xrightarrow{f} S = R \xrightarrow{[\mathsf{L}]} \Phi R \xrightarrow{g} S.$$

Let $e\colon R'' \to \Phi R$ be an equalizer of f and g. We can assume that e is realized by $\lambda x.\, x$ and that R'' is a subset of ΦR. For any I-ary algebraic operation α on T, we have the following diagram:

$$\begin{array}{ccccc} R''^I & \xrightarrow{e^I} & (\Phi R)^I & \overset{f^I}{\underset{g^I}{\rightrightarrows}} & S^I \\ {\scriptstyle h}\,\vdots & & \big\downarrow {\scriptstyle [\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}]} & & \big\downarrow {\scriptstyle [\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}]} \\ R'' & \xrightarrow[e]{} & \Phi R & \overset{f}{\underset{g}{\rightrightarrows}} & S. \end{array}$$

Since $[\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}] \circ e^I$ is realized by $\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}$, there exists a unique $\mathbf{Per}(T)$-morphism $h\colon R''^I \to R''$ realized by $\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}$. Hence, e is linear. By the definition of ΦR, we see that $R'' = \Phi R$. Since e equalizes f and g, we obtain $f = g$. We have shown that the $\mathbf{Per}(T)$-object $R \Rightarrow S$ is isomorphic to $\Phi R \multimap S$, and the isomorphism is natural in S. Hence, U is $\mathbf{Per}(T)$-enriched right adjunction. □

Since U is a $\mathbf{Per}(T)$-enriched right adjoint functor on $\mathbf{Per}(T)$, the adjunction induces a strong monad Φ on $\mathbf{Per}(T)$.

Lemma C.10. *Let α be an I-ary algebraic operation on T. The family of maps*

$$\alpha^\dagger_{S,R}\colon (S \Rightarrow \Phi R)^I \to S \Rightarrow \Phi R$$

realized by $\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{\mathsf{Q}_{i,I}\}_{i\in I}$ is an algebraic operation on Φ. We note that $S \Rightarrow \Phi R$ is closed.

Proof. For any $\mathbf{Per}(T)$-morphisms $f\colon S' \to S$ and $g\colon R \to \Phi R'$, the following $\mathbf{Per}(T)$-morphism

$$f \Rightarrow g'\colon S \Rightarrow \Phi R \to S' \Rightarrow \Phi R'$$

is linear where $g'\colon \Phi R \to \Phi R'$ is the Kleisli lifting of g. Since $\mathbf{Per}(T)$ is well-pointed, $\alpha^\dagger$ is an I-ary algebraic operation on Φ. □

Lemma C.11. *Let $\mathrm{proj}_{i\,A,B}\colon (A \Rightarrow TB)^I \to (A \Rightarrow TB)$ be the i-th projection. Then $\mathrm{proj}_{i\,R,S}^\dagger$ is the i-th projection.*

Proof. Let $\{(X_i, c_i, x_i)\colon A \twoheadrightarrow B\}_{i\in I}$ be a family of T-transducers. We write $\mathrm{inj}_i\colon X_i \to \coprod_{i\in I} X_i$ for the i-th injection. The injection

$$1 + \mathrm{inj}_i\colon 1 + X_i \to 1 + \coprod_{i\in I} X_i$$

is a homomorphism from $(1 + X_i, c'_i, \mathrm{inl}_{1,X_i})$ to

$$\overline{\mathrm{proj}}_{i\,A,B}\{(X_i, c_i, x_i)\}_{i\in I}$$

where c'_i is a unique $\mathbf{Set}_T$-morphism from $(1 + X_i) \times A$ to $(1 + X_i) \times B$ such that

$$c'_i \circ_T (\mathrm{inl}^*_{1,X_i} \otimes A) = (\mathrm{inr}^*_{1,X_i} \otimes B) \circ_T c_i \circ_T (x^*_i \otimes A)$$
$$c'_i \circ_T (\mathrm{inr}^*_{1,X_i} \otimes A) = (\mathrm{inr}^*_{1,X_i} \otimes B) \circ_T c_i.$$

On the other hand, since the cotupling $[x_i, \mathrm{id}_{X_i}]\colon 1 + X_i \to X_i$ is a homomorphism from $(1 + X_i, c'_i, \mathrm{inl}_{1,X_i})$ to (X_i, c_i, x_i), the T-transducer

$$\overline{\mathrm{proj}}_{i\,A,B}\{(X_i, c_i, x_i)\}_{i\in I}$$

is behaviorally equivalent to (X_i, c_i, x_i). Hence, $\mathrm{proj}_{i\,R,S}^\dagger$ is the morphism realized by $\mathsf{Q}_{i,I}$, which realizes the i-th projection. □

Lemma C.12. *For an I-ary algebraic operation α and a family of J-ary algebraic operations $\{\beta_i\}_{i\in I}$ on T, we define a J-ary algebraic operation γ on T by*

$$\gamma\{f_j\}_{j\in J} = \alpha_{A,B}\Big\{\beta_{i\,A,B}\{f_j\}_{j\in J}\Big\}_{i\in I}$$

where f_j are $\mathbf{Set}_T$-morphisms from A to TB. Then it holds that

$$\gamma^\dagger_{R,S} = \alpha^\dagger_{R,S} \circ \langle \beta_{i\,R,S}^\dagger \rangle_{i\in I}$$

where $\langle \beta_{i\,R,S}^\dagger \rangle_{i\in I}\colon (R \Rightarrow \Phi S)^J \to (R \Rightarrow \Phi S)^I$ is a unique $\mathbf{Per}(T)$-morphism whose i-th projection is equal to $\beta_{i\,R,S}^\dagger$.

Proof. Let $\{(X_j, c_j, x_j)\}_{j\in J}$ be a family of T-transducers from A to B. For $i \in I$, we define a T-transducer

$$(1 + Y, d_i, \mathrm{inl}_{1,Y})\colon A \twoheadrightarrow B$$

to be $\overline{\beta}_{i\,A,B}\{(X_j, c_j, x_j)\}_{j\in J}$ where

$$Y = \coprod_{j\in J} X_j.$$

We define a T-transducer

$$(1 + Z, e, \mathrm{inl}_{1,Z})\colon A \twoheadrightarrow B$$

to be $\overline{\alpha}_{A,B}\{(1 + Y, d_i, \mathrm{inl}_{1,Y})\}_{i\in I}$ where

$$Z = \coprod_{i\in I}(1 + Y) = \coprod_{i\in I}\left(1 + \coprod_{j\in J} X_j\right).$$

We define a T-transducer

$$(1 + W, f, \mathrm{inl}_{1,W})\colon A \twoheadrightarrow B$$

to be $\overline{\gamma}_{A,B}\{(X_j, c_j, x_j)\}_{j\in J}$ where

$$W = \coprod_{j\in J} X_j.$$

We define a T-transducer

$$(1 + V, g, \mathrm{inl}_{1,V})\colon A \twoheadrightarrow B$$

by

$$V = \coprod_{i\in I}\coprod_{j\in J} X_j \xleftarrow{\mathrm{inj}_{i,j}} X_j$$

where $g\colon (1+V)\otimes A \to_T (1+V)\otimes B$ is a unique $\mathbf{Set}_T$-morphism such that

$$g \circ_T (\mathrm{inl}^*_{1,V} \otimes A) = \alpha_{A,(1+V)\otimes B}\{(\mathrm{inr}^*_{1,V} \otimes B) \circ_T b_i\}_{i\in I}$$

$$g \circ_T ((\mathrm{inl}^*_{1,V} \circ_T \mathrm{inj}^*_{i,j}) \otimes A) = ((\mathrm{inl}^*_{1,V} \circ_T \mathrm{inj}^*_{i,j}) \otimes A) \circ_T c_j$$

where $b_i\colon A \to_T V \otimes B$ is

$$\beta_{i\,A,V\otimes B}\{(\mathrm{inj}^*_{i,j} \otimes B) \circ_T c_j \circ_T (x^*_j \otimes A)\}_{j\in J}.$$

The injection

$$1 + \coprod_{i\in I} \mathrm{inr}_{1,Y}\colon 1 + V \to 1 + Z$$

is a homomorphism from $(1 + V, g, \mathrm{inl}_{1,V})$ to $(1 + Z, e, \mathrm{inl}_{1,Z})$, and the map $h\colon 1 + V \to 1 + W$ that is codiagonal on the right summand $\coprod_{j=1}^m X_j$ is a homomorphism from $(1 + V, g, \mathrm{inl}_{1,V})$ to $(1 + W, f, \mathrm{inl}_{1,W})$. Hence, we obtain

$$\overline{\gamma}_{A,B}\{(X_j, c_j, x_j)\}_{j\in J} \simeq^T_{A,B} \overline{\alpha}_{A,B}\Big\{\overline{\beta}_{i\,A,B}\{(X_j, c_j, x_j)\}_{j\in J}\Big\}_{i\in I}.$$

Then it is easy to check $\gamma^\dagger_{R,S} \circ x = \alpha^\dagger_{R,S} \circ \langle {\beta_i}^\dagger_{R,S}\rangle_{i\in I} \circ x$ for all $\mathbf{Per}(T)$-morphism $x\colon 1 \to (R \Rightarrow \Phi S)^I$. Since $\mathbf{Per}(T)$ is well-pointed, we obtain the statement. □

Theorem C.13. *The operation $(-)^\dagger$ is an identity-on-object countable-product-preserving faithful functor from $\mathbf{AlgOp}_T$ to $\mathbf{AlgOp}_\Phi$.*

Proof. By Lemma C.11 and Lemma C.12, we can extend $(-)^\dagger$ to an identity on object countable product preserving functor from $\mathbf{AlgOp}_T$ to $\mathbf{AlgOp}_\Phi$. For I-ary algebraic operations α and β on T, we suppose that $\alpha^\dagger$ is equal to $\beta^\dagger$. Let Δ be a $\mathbf{Per}(T)$-object given by

$$\Delta = \{(a,a) \mid a \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})\},$$

which is closed. For any family $\{a_i \in \mathbf{Res}(T)(\mathbb{N},\mathbb{N})\}_{i\in I}$,

$$1 \xrightarrow{\prod_{i\in I}[\lambda x.\, a_i]} \Delta^I \xrightarrow{[\mathrm{L}]^I} (\Phi\Delta)^I \xrightarrow{\alpha^\dagger_{1,\Delta}} \Phi\Delta \xrightarrow{[\lambda y.\, y\cdot(\lambda x.\, x)]} \Delta$$

is equal to

$$1 \xrightarrow{\prod_{i\in I}[\lambda x.\, a_i]} \Delta^I \xrightarrow{[\mathrm{L}]^I} (\Phi\Delta)^I \xrightarrow{\beta^\dagger_{1,\Delta}} \Phi\Delta \xrightarrow{[\lambda y.\, y\cdot(\lambda x.\, x)]} \Delta.$$

This equality means that

$$\alpha_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\}_{i\in I} \simeq^T_{\mathbb{N},\mathbb{N}} \beta_{\mathbb{N},\mathbb{N}}\{(X_i,c_i,x_i)\}_{i\in I}$$

for each family of T-transducers $\{(X_i,c_i,x_i)\colon \mathbb{N} \twoheadrightarrow \mathbb{N}\}_{i\in I}$. In particular, for each family of $\mathbf{Set}_T$-morphisms $\{f_i\colon \mathbb{N} \to_T \mathbb{N}\}_{i\in I}$, we have

$$\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I} \simeq^T_{\mathbb{N},\mathbb{N}} \overline{\beta}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I}.$$

Therefore, there exists a T-transducer $(Y,d,y)\colon \mathbb{N} \twoheadrightarrow \mathbb{N}$ and homomorphisms

$$h\colon \overline{\alpha}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I} \to (Y,d,y),$$
$$k\colon \overline{\beta}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I} \to (Y,d,y).$$

We write (Z,e,z) for $\overline{\alpha}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I}$ and (Z',e',z') for $\overline{\beta}_{\mathbb{N},\mathbb{N}}\{Jf_i\}_{i\in I}$. By the definition of $\overline{\alpha}$, Z and Z' are equal to $1+I$, and z and z' are the first injections. The $\mathbf{Set}_T$-morphisms e and e' satisfy the following equations:

$$(\top^*_{1+I} \otimes \mathbb{N}) \circ_T e \circ_T (z^* \otimes \mathbb{N}) = \alpha_{\mathbb{N},\mathbb{N}}\{f_i\},$$
$$(\top^*_{1+I} \otimes \mathbb{N}) \circ_T e' \circ_T (z^* \otimes \mathbb{N}) = \beta_{\mathbb{N},\mathbb{N}}\{f_i\}.$$

Since

$$\begin{aligned}
&(\top^*_{1+I} \otimes \mathbb{N}) \circ_T e \circ_T (z^* \otimes \mathbb{N})\\
&= (\top^*_{Y} \otimes \mathbb{N}) \circ_T (h^* \otimes \mathbb{N}) \circ_T e \circ_T (z^* \otimes \mathbb{N})\\
&= (\top^*_{Y} \otimes \mathbb{N}) \circ_T d \circ_T (h^* \otimes \mathbb{N}) \circ_T (z^* \otimes \mathbb{N})\\
&= (\top^*_{Y} \otimes \mathbb{N}) \circ_T d \circ_T (y^* \otimes \mathbb{N})\\
&= (\top^*_{Y} \otimes \mathbb{N}) \circ_T d \circ_T (k^* \otimes \mathbb{N}) \circ_T (z'^* \otimes \mathbb{N})\\
&= (\top^*_{Y} \otimes \mathbb{N}) \circ_T (k^* \otimes \mathbb{N}) \circ_T e' \circ_T (z'^* \otimes \mathbb{N})\\
&= (\top^*_{1+I} \otimes \mathbb{N}) \circ_T e' \circ_T (z'^* \otimes \mathbb{N}),
\end{aligned}$$

we obtain $\alpha_{\mathbb{N},\mathbb{N}}\{f_i\} = \beta_{\mathbb{N},\mathbb{N}}\{f_i\}$. By naturality of α and β, we have $\alpha_{1,\mathrm{ary}(\alpha)} = \beta_{1,\mathrm{ary}(\alpha)}$. Hence, $\alpha = \beta$. □

C.3 A Proof of Theorem 6.2

We first give the syntax and equational theory of an extension $\mathcal{L}$ of the computational lambda calculus to algebraic effects and a base type `nat` of natural numbers. For treatment of algebraic effects, we employ generic effects.

We define types by the following BNF:

$$\tau := \mathtt{unit} \mid \mathtt{nat} \mid \tau+\tau \mid \tau\times\tau \mid \tau \Rightarrow \tau.$$

We define $\mathcal{A}$ for the subclass of types given by the following BNF:

$$\beta := \mathtt{unit} \mid \mathtt{nat} \mid \beta+\beta \mid \beta\times\beta.$$

We call types in $\mathcal{A}$ *arity types*.

A *signature* Σ is a set of triples

$$(\mathtt{gen}, \beta, \beta')$$

consisting of an operation symbol `gen` and arity types β and β'. We define terms `t`, values `v`, effect terms `e` and evaluation contexts `E` by the following BNF:

$$\begin{aligned}
\mathtt{t} :=\ & \mathtt{x} \in \mathrm{Var} \mid * \mid \mathtt{c}_n\ (n\in\mathbb{N}) \mid \mathtt{t}+\mathtt{t} \mid \mathtt{t}\,\mathtt{t} \mid \lambda\mathtt{x}:\tau.\,\mathtt{t}\\
& \mid \mathtt{fst}(\mathtt{t}) \mid \mathtt{snd}(\mathtt{t}) \mid \langle\mathtt{t},\mathtt{t}\rangle \mid \mathtt{inl}_{\tau,\sigma}\mathtt{t} \mid \mathtt{inr}_{\tau,\sigma}\mathtt{t}\\
& \mid \mathtt{case}(\mathtt{t},\mathtt{x}.\,\mathtt{t},\mathtt{y}.\,\mathtt{t}) \mid \mathtt{gen}(\mathtt{t})\\
\mathtt{e} :=\ & \mathtt{x} \in \mathrm{Var} \mid * \mid \mathtt{c}_n\ (n\in\mathbb{N}) \mid \mathtt{e}+\mathtt{e}\\
& \mid \mathtt{fst}(\mathtt{e}) \mid \mathtt{snd}(\mathtt{e}) \mid \langle\mathtt{e},\mathtt{e}\rangle \mid \mathtt{inl}_{\tau,\sigma}\mathtt{e} \mid \mathtt{inr}_{\tau,\sigma}\mathtt{e}\\
& \mid \mathtt{case}(\mathtt{e},\mathtt{x}.\,\mathtt{e},\mathtt{y}.\,\mathtt{e}) \mid \mathtt{gen}(\mathtt{e})\\
\mathtt{v} :=\ & \mathtt{x} \in \mathrm{Var} \mid * \mid \mathtt{c}_n\ (n\in\mathbb{N}) \mid \mathtt{v}+\mathtt{v} \mid \mathtt{fst}(\mathtt{v}) \mid \mathtt{snd}(\mathtt{v})\\
& \mid \langle\mathtt{v},\mathtt{v}\rangle \mid \lambda\mathtt{x}:\tau.\,\mathtt{t} \mid \mathtt{inl}_{\tau,\sigma}\,\mathtt{v} \mid \mathtt{inr}_{\tau,\sigma}\,\mathtt{v}\\
& \mid \mathtt{case}(\mathtt{v},\mathtt{x}.\,\mathtt{v},\mathtt{y}.\,\mathtt{v})\\
\mathtt{E} :=\ & [-] \mid \mathtt{E}+\mathtt{t} \mid \mathtt{v}+\mathtt{E} \mid \mathtt{v}\,\mathtt{E} \mid \mathtt{E}\,\mathtt{t} \mid \mathtt{fst}(\mathtt{E}) \mid \mathtt{snd}(\mathtt{E})\\
& \mid \langle\mathtt{E},\mathtt{t}\rangle \mid \langle\mathtt{v},\mathtt{E}\rangle \mid \mathtt{inl}_{\tau,\sigma}\,\mathtt{E} \mid \mathtt{inr}_{\tau,\sigma}\,\mathtt{E} \mid \mathtt{case}(\mathtt{E},\mathtt{x}.\,\mathtt{t},\mathtt{x}.\,\mathtt{t})\\
& \mid \mathtt{gen}(\mathtt{E})
\end{aligned}$$

where Var is a (countable) set of variables. We will use effect terms to define the equational theory of $\mathcal{L}$. We write $\mathrm{Fv}(\mathtt{t})$ for the set of free variables in `t` and $\mathrm{Fv}(\mathtt{E})$ for the set of free variables in `E` where the lambda abstraction and `case` introduce variable bindings. As usual, we identify terms modulo α-equivalence, and for terms `t` and `s`, we write $\mathtt{t}[\mathtt{s}/\mathtt{x}]$ for the term substitution in a capture-avoiding manner. We use let-notation as a syntactic sugar:

$$\mathtt{let}\ \mathtt{x}:\tau\ \mathtt{be}\ \mathtt{t}\ \mathtt{in}\ \mathtt{s} = (\lambda\mathtt{x}:\tau.\,\mathtt{s})\,\mathtt{t}.$$

Typing rules for the core fragment are standard. For $(\mathtt{gen},\beta,\beta') \in \Sigma$, we have the following typing rule:

$$\frac{\Gamma \vdash \mathtt{t} : \beta}{\Gamma \vdash \mathtt{gen}(\mathtt{t}) : \beta'}$$

where Γ is a term environment: Γ is a finite sequence consisting of pairs of a variable and a type.

A *theory* $\mathcal{T}$ is a set of lists $(\Gamma,\mathtt{t},\mathtt{s},\beta)$ consisting of effect terms $\Gamma \vdash \mathtt{e} : \beta$ and $\Gamma \vdash \mathtt{e}' : \beta$ such that types that appear in Γ, `e` and `e'` are arity types. When $(\Gamma,\mathtt{e},\mathtt{e}',\tau)$ is in $\mathcal{T}$, we write $\Gamma \vdash \mathtt{e} \sim_{\mathcal{T}} \mathtt{e}' : \tau$. In Figure 1, we give axioms of the equational theory where we implicitly assume that all terms are well-typed under the term environments Γ. The equational theory of $\mathcal{L}$ consists of the axioms in Figure 1 and rules stating that $\approx$ is a congruence.

Remark C.14. As observed in [28], We can regard the following term construction

$$\frac{\Gamma \vdash \mathtt{t} : \beta \qquad \Gamma, \mathtt{x} : \beta' \vdash \mathtt{s} : \tau}{\Gamma \vdash \mathtt{op}(\mathtt{t}, \mathtt{x} : \beta'.\,\mathtt{s}) : \tau}$$

as a syntactic sugar of $\mathtt{let}\ \mathtt{x} : \beta' = \mathtt{gen}\,\mathtt{t}\ \mathtt{in}\ \mathtt{s}$. On the other hand, the term construction $\mathtt{gen}(\mathtt{t})$ can be encoded as follows:

$$\mathtt{op}(\mathtt{t}, \mathtt{x} : \beta'.\,\mathtt{x}).$$

With respect to this correspondence, the syntax of $\mathcal{L}$ is the same as the one given in [28].

$$\Gamma \vdash (\lambda \mathtt{x} : \tau.\, \mathtt{t})\, \mathtt{v} \approx \mathtt{t}[\mathtt{v}/\mathtt{x}] : \sigma$$
$$\Gamma \vdash \mathtt{v} \approx * : \mathtt{unit}$$
$$\Gamma \vdash \lambda \mathtt{x} : \tau.\, \mathtt{v}\, \mathtt{x} \approx \mathtt{v} : \tau \Rightarrow \sigma \quad (\mathtt{x} \notin \mathrm{Fv}(\mathtt{v}))$$
$$\Gamma \vdash \mathtt{c}_n + \mathtt{c}_m \approx \mathtt{c}_{n+m} : \mathtt{nat}$$
$$\Gamma \vdash \mathtt{fst}(\langle \mathtt{v}, \mathtt{v}' \rangle) \approx \mathtt{v} : \tau$$
$$\Gamma \vdash \mathtt{snd}(\langle \mathtt{v}, \mathtt{v}' \rangle) \approx \mathtt{v}' : \sigma$$
$$\Gamma \vdash \langle \mathtt{fst}(\mathtt{v}), \mathtt{snd}(\mathtt{v}) \rangle \approx \mathtt{v} : \tau \times \sigma$$
$$\Gamma \vdash \mathtt{case}(\mathtt{inl}_{\tau,\sigma}\, \mathtt{v}, \mathtt{x}.\, \mathtt{s}, \mathtt{y}.\, \mathtt{s}') \approx \mathtt{s}[\mathtt{v}/\mathtt{x}] : \rho$$
$$\Gamma \vdash \mathtt{case}(\mathtt{inr}_{\tau,\sigma}\, \mathtt{v}, \mathtt{x}.\, \mathtt{s}, \mathtt{y}.\, \mathtt{s}') \approx \mathtt{s}'[\mathtt{v}/\mathtt{x}] : \rho$$
$$\Gamma \vdash \mathtt{case}(\mathtt{v}, \mathtt{x}.\, \mathtt{inl}_{\tau,\sigma}\, \mathtt{x}, \mathtt{y}.\, \mathtt{inr}_{\tau,\sigma}\, \mathtt{y}) \approx \mathtt{v} : \tau + \sigma$$
$$\Gamma \vdash \mathtt{let}\ \mathtt{x} : \tau\ \mathtt{be}\ \mathtt{t}\ \mathtt{in}\ \mathtt{E}[\mathtt{x}] \approx \mathtt{E}[\mathtt{t}] : \tau \quad (\mathtt{x} \notin \mathrm{Fv}(\mathtt{E}))$$
$$\frac{\Gamma \vdash \mathtt{e} \sim_{\mathcal{T}} \mathtt{e}' : \beta}{\Gamma \vdash \mathtt{e} \approx \mathtt{e}' : \beta}$$

Figure 1. Axioms of the Equational Theory of $\mathcal{L}$

Next, we give a categorical interpretation of $\mathcal{L}$. We inductively define a set $\underline{\beta}$ for an arity type β by

$$\underline{\mathtt{unit}} = \{*\}$$
$$\underline{\mathtt{nat}} = \mathbb{N}$$
$$\underline{\beta + \beta'} = \underline{\beta} + \underline{\beta'}$$
$$\underline{\beta \times \beta'} = \underline{\beta} \times \underline{\beta'}.$$

We assume that T is a monad on $\mathbf{Set}$ such that

- T satisfies Requirement 4.2.
- For each
$$(\mathtt{gen}, \beta, \beta') \in \Sigma,$$
there is a $\mathbf{Set}_T$-morphism
$$\underline{\mathtt{gen}} : \underline{\beta} \to_T \underline{\beta'}.$$
We naturally extend $\underline{(-)}$ to effect terms: for an effect term
$$\mathtt{x}_1 : \beta_1 \cdots \mathtt{x}_n : \beta_n \vdash \mathtt{e} : \beta,$$
we define a $\mathbf{Set}_T$-morphism
$$\underline{\mathtt{e}} : \underline{\beta_1 \times \cdots \times \beta_n} \to_T \underline{\beta}$$
by the categorical interpretation of effect terms in $\mathbf{Set}_T$.
- For effect terms $\Delta \vdash \mathtt{e} : \beta$ and $\Delta \vdash \mathtt{e}' : \beta$, if $\Delta \vdash \mathtt{e} \sim_{\mathcal{T}} \mathtt{e}' : \beta$, then $\underline{\mathtt{e}} = \underline{\mathtt{e}}$.

Let Φ be the strong monad in Theorem 6.1. For a term

$$\mathtt{x}_1 : \tau_1, \cdots, \mathtt{x}_n : \tau_n \vdash \mathtt{t} : \tau$$

in $\mathcal{L}$, we define a $\mathbf{Per}(T)_\Phi$-morphism

$$[\![\mathtt{t}]\!] : [\![\tau_1]\!] \times \cdots \times [\![\tau_n]\!] \to [\![\tau]\!]$$

to be the categorical interpretation of the term $\mathtt{t}$ in $\mathbf{Per}(T)_\Phi$ given as follows:

- We interpret the core fragment following [26].
- We interpret $\mathtt{nat}$ by the natural number object N of $\mathbf{Per}(T)$ (Lemma C.6).
- For $(\mathtt{gen}, \beta, \beta') \in \Sigma$, we interpret
$$\Gamma \vdash \mathtt{gen}(\mathtt{t}) : \beta'$$
by
$$[\![\Gamma]\!] \xrightarrow{[\![\mathtt{t}]\!]} \Phi[\![\beta]\!] \xrightarrow{a} \Phi[\![\beta']\!]$$
where the $\mathbf{Per}(T)$-morphism $a \colon \Phi[\![\beta]\!] \to \Phi[\![\beta']\!]$ is the Kleisli lifting of the $\mathbf{Per}(T)_\Phi$-morphism from $[\![\beta]\!]$ to $[\![\beta']\!]$ induced by $\underline{\mathtt{gen}} : \underline{\beta} \to_T \underline{\beta'}$.

Note that $\mathbf{AlgOp}(T)$ is isomorphic to the full subcategory of $\mathbf{Set}_T^{\mathrm{op}}$ consisting of countable sets, and $\mathbf{AlgOp}(\Phi)$ is equivalent to the full subcategory of $\mathbf{Per}(T)_\Phi^{\mathrm{op}}$ generated by 1 and countable products. The functor $(-)^\dagger$ and the equivalences induce a countable-coproducts-preserving faithful functor from the subcategory of $\mathbf{Set}_T$ to $\mathbf{Per}(T)_\Phi$ that maps $\underline{\beta}$ to $[\![\beta]\!]$.

Soundness of the categorical interpretation of $\mathcal{L}$ follows from Theorem 6.1.

Theorem C.15. *If $\Gamma \vdash \mathtt{t} \approx \mathtt{s} : \tau$, then $[\![\mathtt{t}]\!] = [\![\mathtt{s}]\!]$.*

For a closed term $\mathtt{t}$ of type τ, the interpretation of the term $\mathtt{t}$ is a $\mathbf{Per}(T)$-morphism from 1 to $\Phi[\![\tau]\!]$, and therefore, the interpretation of the term $\mathtt{t}$ bijectively corresponds to an equivalence class of $\Phi[\![\tau]\!]$. We define GoI interpretation $(\!|\mathtt{t}|\!)$ so that the $\mathbf{Res}(T)$-morphism represented by $(\!|\mathtt{t}|\!)$ represents the equivalence class of $\Phi[\![\tau]\!]$ that bijectively corresponds to $[\![\mathtt{t}]\!]$.

Theorem C.16 (Theorem 6.2)**.** *For closed terms $\mathtt{t}$ and $\mathtt{s}$ of type τ in $\mathcal{L}$,*

- *If $\mathtt{t} \approx \mathtt{s}$, then $([(\!|\mathtt{t}|\!)], [(\!|\mathtt{s}|\!)]) \in \Theta[\![\tau]\!]$.*
- *If $\mathtt{t} \approx \mathtt{s}$ and τ is the base type $\mathtt{nat}$, then $(\!|\mathtt{t}|\!) \simeq^T_{\mathbb{N},\mathbb{N}} (\!|\mathtt{s}|\!)$.*

where $[(\!|\mathtt{t}|\!)]$ is the $\mathbf{Res}(T)$-morphism represented by $(\!|\mathtt{t}|\!)$.

Proof. The first clause follows from soundness of the categorical interpretation. Since the natural number object N of $\mathbf{Per}(T)$ is a subset of the following closed per:

$$\Delta = \{(a, a) \mid a \in \mathbf{Res}(T)(\mathbb{N}, \mathbb{N})\},$$

the per ΦN is a subset of Δ. Hence, all equivalence classes of ΦN are singletons, and if $\mathtt{t} \approx \mathtt{s}$ and τ is the base type $\mathtt{nat}$, then $(\!|\mathtt{t}|\!) \simeq^T_{\mathbb{N},\mathbb{N}} (\!|\mathtt{s}|\!)$. □