Contents lists available at SciVerse ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Relating computational effects by $\top\top$-lifting

Shin-ya Katsumata [*]

Research Institute for Mathematical Sciences, Kyoto University, Kitashirakawaoiwakecho, Sakyoku, Kyoto, 606-8502, Japan

ARTICLE INFO

Article history:
Available online 26 October 2012

Keywords:
Logical relation
Monad
Fibration
Computational effects
$\top\top$-lifting
Algebraic operation
Generic effect
Fresh name creation

ABSTRACT

We consider the problem of establishing a relationship between two interpretations of base type terms of a λ_c-calculus extended with algebraic operations. We show that the given relationship holds if it satisfies a set of natural conditions. We apply this result to 1) comparing two monadic semantics related by a strong monad morphism, and 2) comparing two monadic semantics of fresh name creation: Stark's new name creation monad and the global counter monad. We also consider the same problem, relating semantics of computational effects, in the presence of recursive functions. We apply this additional by extending the previous monad morphism comparison result to the recursive case.

1. Introduction

Suppose that two monadic semantics $\mathcal{A}_1, \mathcal{A}_2$ are given to a call-by-value functional language, and each semantics $\mathcal{A}_i$ $(i = 1, 2)$ interprets a base type b by a set $A_i b$ and computational effects by a monad $\mathcal{T}_i$ over the category **Set**. After comparing these semantics, we find a relationship $Vb \subseteq A_1 b \times A_2 b$ between base type values, and also a relationship $Cb \subseteq T_1 A_1 b \times T_2 A_2 b$ between base type computations. We then consider the following problem:

For any well-typed term $x_1 : b_1, \dots, x_n : b_n \vdash M : b$ and $(v_i, w_i) \in Vb_i$ $(1 \leqslant i \leqslant n)$, do we have $(\mathcal{A}_1[\![M]\!](v_1, \dots, v_n), \mathcal{A}_2[\![M]\!](w_1, \dots, w_n)) \in Cb$?

We name this problem the *effect simulation problem*. It subsumes various natural questions that arise when we have two monadic semantics of a call-by-value functional language. For instance:

- The standard interpretation of a functional language with a nondeterministic choice operator or is given by the monadic semantics $\mathcal{A}_1$ employing the powerset monad. On the other hand, we can use the list monad to list up the nondeterministic choices made by programs; let us call the monadic semantics using the list monad $\mathcal{A}_2$. These semantics interpret the same program in two different ways:
$$\mathcal{A}_1\left[\!\left[\mathrm{or}(2,3) + \mathrm{or}(3,4)\right]\!\right] = \{4,5,6\}, \qquad \mathcal{A}_2\left[\!\left[\mathrm{or}(2,3) + \mathrm{or}(3,4)\right]\!\right] = \langle 5,6,4,5\rangle.$$
From this, we expect that for all base type program M, $\mathcal{A}_1[\![M]\!]$ is the set of occurrences of elements in the list $\mathcal{A}_2[\![M]\!]$.
- From a monadic semantics $\mathcal{A}$ of a language by a monad T, we can build another monadic semantics $\mathcal{A}_{CPS}$ of the same language by the *continuation monad* $(- \Rightarrow TR) \Rightarrow TR$. The latter semantics is called *CPS semantics*. In [3], Filinski formally established a relationship between monadic semantics and CPS semantics.

[*] Fax: +81 75 753 7272.
E-mail address: sinya@kurims.kyoto-u.ac.jp.

0890-5401/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ic.2012.10.014

- The main theme of the recent work [6] by Filinski is the effect simulation problem; there, some relationships between two monadic semantics of a functional language with recursive types and computational effects are studied.

We tackle the effect simulation problem under the situation where 1) the functional language is the simply typed λ_c-calculus with products, sums, effect-free constants and *algebraic operations* [28], and 2) the underlying category of a semantics is a bi-cartesian closed category with a strong monad.

The main theorem (Theorem 7) of this paper gives a sufficient conditions to answer positively to the effect simulation problem; the answer to the effect simulation problem is "yes" if I) the product $\eta_1 \times \eta_2$ of the monad units maps the values related by V to the pure computations related by C, II) V is closed under the effect-free constants and III) C is closed under the algebraic operations in the λ_c-calculus. The point of the main theorem is the generality: it holds with any monad, algebraic operation and relation V and C.

The main theorem is helpful for establishing relationships between various monadic semantics of call-by-value functional languages. Section 10 shows a general comparison theorem of two monadic semantics related by strong monad morphisms. This result subsumes the comparison of the list monad and the powerset monad for nondeterminism, and Filinski's result comparing monadic semantics and CPS semantics. Section 11 compares two interpretations of new name creation: Stark's name creation monad [32], and a global state monad.

Synopsis. Section 2 presents the syntax of the λ_c-calculus extended with algebraic operations. After introducing some category-theoretic concepts used in this paper in Section 3, Section 4 gives the categorical semantics of the λ_c-calculus. Section 5 illustrates the *effect simulation problem* in a set-theoretic setting. To generalise the problem to a categorical settings, Section 6 introduces the fibrational category theory, which provides a concept of predicates on objects in a category. Section 7 generalises the set-theoretic effect simulation problem to *fibrational effect property problem*, and states the main theorem (Theorem 7) that gives sufficient conditions to answer positively to the fibrational effect property problem. Our proof hinges on a construction of logical relations for monads, and in this paper we employ a technique called *categorical* $\top\top$*-lifting* [14], which is a semantic formulation of the *leapfrog method* introduced by Lindley and Stark [18,19]. Section 8 reviews the categorical $\top\top$-lifting, then gives a characterisation of when the logical relations for monads constructed by $\top\top$-lifting are closed under a given set of algebraic operations. Section 9 is the proof of Theorem 7. Sections 10 and 11 apply Theorem 7 to establish relationships between monadic semantics of the $\lambda_c(\Sigma)$-calculus. Section 12 considers the effect simulation problem under the presence of recursive functions. In Section 13 is the conclusion of this paper and a comparison with related works.

This article is the journal version of [16]. We extend algebraic operations to have both an arity and a *co-arity* (which is a *parameter type* in [29,30]). We also correct the mistake in the interpretation of the recursive function term $\mu f x\,.\,M$ in [16]; the correct one is presented as Eq. (7).

Preliminary. Vector notation, such as $\vec{x}$, abbreviates a sequence $x_1, \ldots, x_n$. The length of the sequence is written by $|\vec{x}|$.

2. The λ_c-calculus with algebraic operations

We adopt the simply-typed λ_c-calculus with effect-free constants and algebraic operations as an idealised call-by-value functional language. In Section 12 we add the recursive function terms $\mu f x\,.\,M$ to the λ_c-calculus.

Let B be a set of base types. We use b (and its variants) to range over B. We define the set $\mathrm{Typ}(B)$ of types by the following BNF:

$$\mathrm{Typ}(B) \ni \rho ::= b \mid \prod(\rho, \ldots, \rho) \mid \coprod(\rho, \ldots, \rho) \mid \rho \Rightarrow \rho.$$

We write 1 for $\prod()$, and $\bar{n}$ ($n \in \mathbf{N}$) for $\coprod(\overbrace{1, \ldots, 1}^{n})$. Binary products and binary sums are denoted by the usual infix operators $\times$ and $+$. We define the subset $\mathrm{GTyp}(B) \subseteq \mathrm{Typ}(B)$ of ground types by

$$\mathrm{GTyp}(B) \ni \beta ::= b \mid \prod(\beta, \ldots, \beta) \mid \coprod(\beta, \ldots, \beta).$$

Definition 1. A λ_c-signature Σ is a tuple

$$\Sigma = \bigl(B, K, O, ar, car : K \cup O \to \mathrm{GTyp}(B)\bigr)$$

where B is a set of base types, K and O are respectively disjoint sets of symbols for effect-free constants and algebraic operations, and ar and car are respectively functions giving input and output types to effect-free constant symbols, and arities and co-arities to algebraic operation symbols.

In this paper, we usually consider a single signature Σ in a discussion. We thus use B, K, O, ar, car to refer to each component of the signature Σ in the context. We sometimes declare a λ_c-signature Σ by the following notation:

$$\Sigma = \big(B, \quad K = \{\ldots k_i : \beta_i \to \beta'_i, \ldots\}, \quad O = \{\ldots o_j : \delta_j \to \delta'_j \ldots\}\big).$$

The declared signature is such that the set of base types is B, and the other components are given by

$$K = \{\ldots k_i \ldots\}, \qquad O = \{\ldots o_j \ldots\}, \qquad ar(k_i) = \beta_i, \qquad car(k_i) = \beta'_i, \qquad ar(o_j) = \delta_j, \qquad car(o_j) = \delta'_j.$$

Let Σ be a λ_c-signature. We define the computational lambda calculus $\lambda_c(\Sigma)$. The set of raw terms is defined by the following BNF:

$$M ::= x \mid k\ M \mid o_\rho\ M \mid (M, \ldots, M) \mid \pi_i\ M \mid \iota_i\ M \mid \delta(M, x : \rho\ .\ M, \ldots, x : \rho\ .\ M) \mid \lambda x : \rho\ .\ M \mid MM$$

where k, o range over K, O respectively. The type system of $\lambda_c(\Sigma)$ extends the one for the simply typed lambda calculus with products and sums (see e.g. [22]). The typing rules for $k\ M$ and $o_\rho\ M$ are:

$$\frac{\Gamma \vdash M : ar(k)}{\Gamma \vdash k\ M : car(k)}, \qquad \frac{\Gamma \vdash M : (ar(o) \Rightarrow \rho) \times car(o)}{\Gamma \vdash o_\rho\ M : \rho}.$$

The following inference for a $(\beta, 1)$-ary algebraic operation o:

$$\frac{\Gamma \vdash M : \beta \Rightarrow \rho}{\Gamma \vdash o_\rho\ M : \rho}$$

is the syntactic sugar for the following derivation:

$$\dfrac{\dfrac{\Gamma \vdash M : \beta \Rightarrow \rho \quad \overline{\Gamma \vdash () : 1}}{\Gamma \vdash (M, ()) : (\beta \Rightarrow \rho) \times 1}}{\Gamma \vdash o_\rho(M, ()) : \rho}.$$

It is also convenient to treat an $(\bar{n}, 1)$-ary algebraic operation in the generalised sense as a simple n-ary algebraic operation. The inference:

$$\frac{\Gamma \vdash M_1 : \rho \quad \ldots \quad \Gamma \vdash M_n : \rho}{\Gamma \vdash o_\rho(M_1, \ldots, M_n) : \rho}$$

is the syntactic sugar for the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash M_1 : \rho}{\Gamma, x : \bar{n}, x_1 : 1 \vdash M_1 : \rho} \quad \ldots \quad \dfrac{\Gamma \vdash M_n : \rho}{\Gamma, x : \bar{n}, x_n : 1 \vdash M_n : \rho}}{\Gamma, x : \bar{n} \vdash \delta(x, x_1 : 1\ .\ M_1, \ldots, x_n : 1\ .\ M_n) : \rho}}{\Gamma \vdash \lambda x : \bar{n}\ .\ \delta(x, x_1 : 1\ .\ M_1, \ldots, x_n : 1\ .\ M_n)p : \bar{n} \Rightarrow \rho} \quad \overline{\Gamma \vdash () : 1}}{\Gamma \vdash (\lambda x : \bar{n}\ .\ \delta(x, x_1 : 1\ .\ M_1, \ldots, x_n : 1\ .\ M_n), ()) : (\bar{n} \Rightarrow \rho) \times 1}}{\Gamma \vdash o_\rho(\lambda x : \bar{n}\ .\ \delta(x, x_1 : 1\ .\ M_1, \ldots, x_n : 1\ .\ M_n), ()) : \rho}.$$

3. Categorical preliminary

We regard every set as a discrete category. When $\mathbb{C}$ is a category, by $I \in \mathbb{C}$ we mean that I is an object in $\mathbb{C}$. We fix a distinguished element $*$ and let $\{*\}$ be the terminal object (denoted by 1) in **Set**.

A *bi-cartesian closed* (*bi-CC*) *structure* on a category $\mathbb{C}$ consists of a terminal object 1, an initial object 0, and an assignment of a binary product $(I \times J, \langle -, - \rangle_{I,J}, (\pi_1)_{I,J}, (\pi_2)_{I,J})$, a binary coproduct $(I + J, [-, -]_{I,J}, (\iota_1)_{I,J}, (\iota_2)_{I,J})$ and an exponential $(I \Rightarrow J, \lambda_{I,J}(-), ev_{I,J})$ to each $I, J \in \mathbb{C}$. We usually omit annotating objects on the components of bi-CC structures. In this paper we define a *bi-CCC* to be a pair of a category $\mathbb{C}$ and a bi-CC structure on $\mathbb{C}$. We write $!_I : I \to 1$ ($?_I : 0 \to I$) for the unique morphism to the terminal (resp. from the initial) object. The inverse of the currying operator $\lambda(-)$ is given by $\lambda^{-1}(f) = ev \circ (f \times \mathrm{id})$. We fix a method to extend the terminal (initial) object and the binary (co)product to finite (co)products.

Let $F : \mathbb{C} \to \mathbb{D}$ be a functor between bi-CCCs $\mathbb{C}$ and $\mathbb{D}$. The functor F *preserves finite products* if $!_{F1} : F1 \to 1$ and $l_{I,J} = \langle F\pi_1, F\pi_2 \rangle : F(I \times J) \to FI \times FJ$ are isomorphisms. The functor F *strictly preserves the bi-CC structure* if $!_1, l_{I,J}$ and $?_{F0} : 0 \to F0$ and $m_{I,J} = [F\iota_1, F\iota_2] : FI + FJ \to F(I + J)$ are the identity morphisms, and moreover $n_{I,J} = \lambda(F(ev) \circ l^{-1}_{I \Rightarrow J, I}) : F(I \Rightarrow J) \to (FI \Rightarrow FJ)$ is also the identity morphism. One can easily show that F strictly preserves the bi-CC structure if and only if F commutes with the components of the bi-CC structure, that is, on objects:

$$F1 = 1, \qquad F0 = 0, \qquad F(I \mathbin{\Box} J) = FI \mathbin{\Box} FJ \quad \big(\Box \in \{\times, +, \Rightarrow\}\big)$$

and on morphisms:

$$F\langle f,g\rangle = \langle Ff, Fg\rangle, \qquad F[f,g] = [Ff, Fg], \qquad F\big(\lambda(f)\big) = \lambda(Ff),$$
$$F!_I = !_{FI}, \qquad F?_I = ?_{FI}, \qquad F\pi_i = \pi_i, \qquad F\iota_i = \iota_i, \qquad F(ev) = ev \quad (i = 1, 2).$$

Let $\mathbb{C}$ be a bi-CCC. A *strong monad* $\mathcal{T}$ on $\mathbb{C}$ is a pair of a monad (T, η, μ) and a natural transformation $\theta_{I,J} : I \times TJ \to T(I \times J)$ called *strength* such that

$$\begin{tikzcd}
1 \times TI \arrow[r, "\theta_{1,I}"] \arrow[dr, "\pi_2"'] & T(1 \times I) \arrow[d, "T\pi_2"] \\
& TI
\end{tikzcd}
\qquad
\begin{tikzcd}
(I \times J) \times TK \arrow[rr, "\theta_{I\times J,K}"] \arrow[d, "\mathsf{assoc}_{I,J,TK}"] & & T((I \times J) \times K) \arrow[d, "T(\mathsf{assoc}_{I,J,K})"] \\
I \times (J \times TK) \arrow[r, "I\times\theta_{J,K}"'] & I \times T(J \times K) \arrow[r, "\theta_{I,J\times K}"'] & T(I \times (J \times K))
\end{tikzcd}$$

$$\begin{tikzcd}
I \times J \arrow[r, "I\times\eta_J"] \arrow[dr, "\eta_{I\times J}"'] & I \times TJ \arrow[d, "\theta_{I,J}"] \\
& T(I \times J)
\end{tikzcd}
\qquad
\begin{tikzcd}
I \times T^2 J \arrow[rr, "I\times\mu_J"] \arrow[d, "\theta_{I,TJ}"'] & & I \times TJ \arrow[d, "\theta_{I,J}"] \\
T(I \times TJ) \arrow[r, "T\theta_{I,J}"'] & T^2(I \times J) \arrow[r, "\mu_{I\times J}"'] & T(I \times J)
\end{tikzcd}$$

here $\mathsf{assoc}_{I,J,K} : (I \times J) \times K \to I \times (J \times K)$ is the associativity morphism. We call the following function *Kleisli lifting* associated to the monad $\mathcal{T}$.

$$(-)^\# : \mathbb{C}(I, TJ) \to \mathbb{C}(TI, TJ), \quad f^\# = \mu_J \circ Tf.$$

Let $\mathcal{T} = (T, \eta, \mu, \theta)$ be a strong monad on $\mathbb{C}$. In [28,29], Plotkin and Power presented two forms of the operations that manipulate computational effects: *generic effects* and *algebraic operations*. Let $D, C \in \mathbb{C}$. A (D, C)*-ary generic effect* for $\mathcal{T}$ is just a morphism $e : C \to TD$ in $\mathbb{C}$. Next, a (D, C)*-ary algebraic operation* for $\mathcal{T}$ is a natural transformation

$$\alpha_I : (D \Rightarrow TI) \to (C \Rightarrow TI)$$

such that the following diagrams commute (Proposition 1, [29]):

$$\begin{tikzcd}
I \times (D \Rightarrow TJ) \arrow[r, "\mathsf{st}^D_{I,TJ}"] \arrow[d, "I\times\alpha_J"'] & D \Rightarrow I \times TJ \arrow[r, "D\Rightarrow\theta_{I,J}"] & D \Rightarrow T(I \times J) \arrow[d, "\alpha_{I\times J}"] \\
I \times (C \Rightarrow TJ) \arrow[r, "\mathsf{st}^C_{I,TJ}"'] & C \Rightarrow I \times TJ \arrow[r, "C\Rightarrow\theta_{I,J}"'] & C \Rightarrow T(I \times J)
\end{tikzcd}
\qquad
\begin{tikzcd}
D \Rightarrow T^2 I \arrow[r, "D\Rightarrow\mu_I"] \arrow[d, "\alpha_{TI}"'] & D \Rightarrow TI \arrow[d, "\alpha_I"] \\
C \Rightarrow T^2 I \arrow[r, "C\Rightarrow\mu_I"'] & C \Rightarrow TI
\end{tikzcd}$$

here, $\mathsf{st}^K_{I,J} : I \times (K \Rightarrow J) \to (K \Rightarrow I \times J)$ is the morphism defined by $\mathsf{st}^K_{I,J} = \lambda(I \times ev \circ \mathsf{assoc}_{I,K\Rightarrow J,K})$. We call D and C the *arity* and the *co-arity* of α, respectively. The terminology "co-arity" represents the variance that is the opposite of an arity; in [29,30] *parameter type* is also used. We write $\mathrm{Alg}(\mathcal{T}, D, C)$ for the collection of (D, C)-ary algebraic operations for $\mathcal{T}$. The definition of algebraic operations with respect to a pair of objects is due to [29], and generalises the arity from natural numbers [28]. An n-ary algebraic operation ($n \in \mathbf{N}$) bijectively corresponds to a $(\coprod_{i=1}^n 1, 1)$-ary algebraic operation.

We review the correspondence between generic effects and algebraic operations studied in [29]. We introduce some auxiliary functions and morphisms. The function

$$\mathsf{sw}^{I,J}_K : \mathbb{C}(I, J \Rightarrow K) \to \mathbb{C}(J, I \Rightarrow K), \quad \mathsf{sw}^{I,J}_K(f) = \lambda\big(\lambda^{-1}(f) \circ \langle \pi_2, \pi_1\rangle\big)$$

swaps the order of arguments of a morphism. Doing this twice yields the original morphism, that is, $\mathsf{sw}^{J,I}_K(\mathsf{sw}^{I,J}_K(f)) = f$. Next, the function

$$\mathsf{nm}_{I,J} : \mathbb{C}(I, J) \to \mathbb{C}(1, I \Rightarrow J), \quad \mathsf{nm}_{I,J} = \lambda(f \circ \pi_2)$$

converts a given morphism $f : I \to J$ to a point (which we call in this paper its *name*) in $I \Rightarrow J$. This function is bijective. The morphism

$$\mathsf{kl}^{\mathcal{T}}_{I,J} : (I \Rightarrow TJ) \to (TI \Rightarrow TJ), \quad \mathsf{kl}^{\mathcal{T}}_{I,J} = \lambda\big(ev^\# \circ \theta_{I\Rightarrow TJ,I}\big)$$

internalises the Kleisli lifting. By swapping its arguments, we obtain

$$\sigma^{\mathcal{T},J}_I : TI \to \big((I \Rightarrow TJ) \Rightarrow TJ\big), \quad \sigma^{\mathcal{T},J}_I = \mathsf{sw}^{I\Rightarrow TJ,TI}_{TJ}\big(\mathsf{kl}^{\mathcal{T}}_{I,J}\big),$$

which corresponds to the *bind* operator in Haskell.

The correspondence between generic effects and algebraic operations is given by the following assignment:

$$\mathrm{Aop}^{\mathcal{T}} : \mathbb{C}(C, TD) \to \mathrm{Alg}(\mathcal{T}, D, C), \quad \mathrm{Aop}^{\mathcal{T}}(e)_I = \mathrm{sw}_{TI}^{C, D \Rightarrow TI}\big(\sigma_D^{\mathcal{T}, I} \circ e\big).$$

It has the inverse $\mathrm{Gef}^{\mathcal{T}} : \mathrm{Alg}(\mathcal{T}, D, C) \to \mathbb{C}(C, TD)$ given by

$$\mathrm{Gef}^{\mathcal{T}}(\alpha) = \mathrm{nm}_{C,TD}^{-1}\big(\alpha_D \circ \mathrm{nm}_{D,TD}(\eta_D)\big).$$

Let $\mathcal{T}' = (T', \eta', \mu', \theta')$ be another strong monad on $\mathbb{C}$. A *strong monad morphism* from $\mathcal{T}$ to $\mathcal{T}'$ is a natural transformation $\gamma_I : TI \to T'I$ such that

$$\begin{tikzcd} I \arrow[r, "\eta_I"] \arrow[dr, "\eta'_I"'] & TI \arrow[d, "\gamma_I"] \\ & T'I \end{tikzcd} \qquad \begin{tikzcd} T^2 I \arrow[rr, "\mu_I"] \arrow[d, "T\gamma_I"'] & & TI \arrow[d, "\gamma_I"] \\ T(T'I) \arrow[r, "\gamma_{T'I}"'] & (T')^2 I \arrow[r, "\mu'_I"'] & T'I \end{tikzcd} \qquad \begin{tikzcd} I \times TJ \arrow[r, "\theta_{I,J}"] \arrow[d, "I \times \gamma_J"'] & T(I \times J) \arrow[d, "\gamma_{I \times J}"] \\ I \times T'J \arrow[r, "\theta'_{I,J}"'] & T'(I \times J) \end{tikzcd}$$

It determines the following function $\mathrm{Alg}(\gamma, D, C) : \mathrm{Alg}(\mathcal{T}, D, C) \to \mathrm{Alg}(\mathcal{T}', D, C)$:

$$\mathrm{Alg}(\gamma, D, C)(\alpha) = \mathrm{Aop}^{\mathcal{T}'}\big(\gamma_D \circ \mathrm{Gef}^{\mathcal{T}}(\alpha)\big).$$

4. The semantics of the λ_c-calculus

In this paper we consider the semantics of the $\lambda_c(\Sigma)$-calculus in a bi-CCC with a strong monad.

Definition 2. A *bi-cartesian closed* $\lambda_c(\Sigma)$*-structure* is a tuple $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ where $\mathbb{C}$ is a bi-CCC, $\mathcal{T}$ is a strong monad on $\mathbb{C}$, A is a functor of type $B \to \mathbb{C}$; before continuing further, we extend A to the functor $\mathcal{A}[\![-]\!] : \mathrm{GTyp}(B) \to \mathbb{C}$ by

$$\mathcal{A}[\![b]\!] = Ab, \qquad \mathcal{A}\Big[\!\!\Big[\prod(\vec{\beta})\Big]\!\!\Big] = \prod_{i=1}^{|\vec{\beta}|} \mathcal{A}[\![\beta_i]\!], \qquad \mathcal{A}\Big[\!\!\Big[\coprod(\vec{\beta})\Big]\!\!\Big] = \coprod_{i=1}^{|\vec{\beta}|} \mathcal{A}[\![\rho_i]\!].$$

Continuation of the definition is as follows: a assigns to each $k \in K$ a morphism $a(k) : \mathcal{A}[\![ar(k)]\!] \to \mathcal{A}[\![car(k)]\!]$, and to each $o \in O$ an algebraic operation $a(o) \in \mathrm{Alg}(\mathcal{T}, \mathcal{A}[\![ar(o)]\!], \mathcal{A}[\![car(o)]\!])$.

In this definition, we require that $\mathbb{C}$ has *all* exponentials, but this requirement is stronger than what is needed to interpret the $\lambda_c(\Sigma)$-calculus. In fact it is sufficient for $\mathbb{C}$ to have *Kleisli exponentials*, which are representations of $\mathbb{C}(- \times I, TJ) : \mathbb{C}^{op} \to \mathbf{Set}$ [25]. The reason for this requirement is a technical one: at this moment we do not know how to perform the *categorical* $\top\top$*-lifting* [14] over the categories with only Kleisli exponentials. Still, many natural concrete semantics of the $\lambda_c(\Sigma)$-calculus are covered by bi-cartesian closed $\lambda_c(\Sigma)$-structures. Below, we simply use $\lambda_c(\Sigma)$-structure to mean a bi-cartesian closed $\lambda_c(\Sigma)$-structure.

We write $\mathcal{A}_1 \times \mathcal{A}_2$ for the evident $\lambda_c(\Sigma)$-structure such that its category is given by the product of the categories of $\mathcal{A}_1$ and $\mathcal{A}_2$. Each $\lambda_c(\Sigma)$-structure $\mathcal{A}$ determines a natural interpretation $\mathcal{A}[\![-]\!]$ of well-typed $\lambda_c(\Sigma)$-terms in the category of $\mathcal{A}$ (see e.g. [2]). Effect-free constants and algebraic operations are interpreted as follows:

$$\mathcal{A}\big[\!\big[k(M)\big]\!\big] = T\big(a(k)\big) \circ \mathcal{A}[\![M]\!],$$
$$\mathcal{A}\big[\!\big[o_\rho(M)\big]\!\big] = \big(\lambda^{-1}\big(a(o)_{\mathcal{A}[\![\rho]\!]}\big)\big)^{\#} \circ \mathcal{A}[\![M]\!].$$

5. Effect simulation problem

The main problem we consider is the *effect simulation problem*. We first introduce a set-theoretic version of it. Let Σ be a λ_c-signature and $\mathcal{A}_1$ and $\mathcal{A}_2$ be $\lambda_c(\Sigma)$-structures over **Set**. A *simulation* between $\mathcal{A}_1$ and $\mathcal{A}_2$ is a pair (V, C) where V and C are B-indexed families of binary relations such that $Vb \subseteq \mathcal{A}_1[\![b]\!] \times \mathcal{A}_2[\![b]\!]$ and $Cb \subseteq T_1\mathcal{A}_1[\![b]\!] \times T_2\mathcal{A}_2[\![b]\!]$. The effect simulation problem is the following:

Suppose that a simulation (V, C) between $\mathcal{A}_1$ and $\mathcal{A}_2$ is given. Then for any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$ and $(v_i, w_i) \in Vb_i$ $(1 \leqslant i \leqslant n)$, do we have $(\mathcal{A}_1[\![M]\!](\vec{v}), \mathcal{A}_2[\![M]\!](\vec{w})) \in Cb$?

Example 1. Let Σ be a λ_c-signature such that $O = \{\mathsf{null} : \bar{0} \to 1, \mathsf{or} : \bar{2} \to 1\}$; we leave the set B of base types and K of effect-free constant symbols unspecified because they are irrelevant in this example. These algebraic operations enable the following inferences (see Section 2):

$$\frac{}{\Gamma \vdash \mathsf{null}_\rho() : \rho} \qquad \frac{\Gamma \vdash M_1 : \rho \quad \Gamma \vdash M_2 : \rho}{\Gamma \vdash \mathsf{or}_\rho(M_1, M_2) : \rho}.$$

We regard them as the primitives for nondeterministic computation.

The standard semantics of the $\lambda_c(\Sigma)$-calculus is given by the $\lambda_c(\Sigma)$-structure $\mathcal{A}_1 = (\mathbf{Set}, \mathcal{T}_p, A, a_1)$, where $\mathcal{T}_p$ is the finite powerset monad, $A : B \to \mathbf{Set}$ is any functor, and a_1 assigns a function to each effect-free constant symbol $k \in K$ and the following algebraic operations to null and or:

$$a_1(\mathsf{null})_I(f)(*) = \emptyset, \qquad a_1(\mathsf{or})_I(f)(*) = f\big(\iota_1(*)\big) \cup f\big(\iota_2(*)\big).$$

On the other hand, we may represent nondeterministic choices by finite lists instead of finite sets. This representation corresponds to the semantics of the $\lambda_c(\Sigma)$-calculus by the $\lambda_c(\Sigma)$-structure $\mathcal{A}_2 = (\mathbf{Set}, \mathcal{T}_m, A, a_2)$, where $\mathcal{T}_m$ is the free monoid monad, and a_2 is the assignment such that $a_2(k) = a_1(k)$ for all $k \in K$, and it assigns the following algebraic operations to null and or:

$$a_2(\mathsf{null})_I(f)(*) = \epsilon, \qquad a_2(\mathsf{or})_I(f)(*) = f\big(\iota_1(*)\big) \cdot f\big(\iota_2(*)\big).$$

We expect that for any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \dots, x_n : b_n \vdash M : b$ and $v_i \in Ab_i$, the denotation $\mathcal{A}_1[\![M]\!](v_1, \dots, v_n)$ gives the set of possible return values listed up in $\mathcal{A}_2[\![M]\!](v_1, \dots, v_n)$. That is, we expect that the answer to the effect simulation problem with the simulation (V, C) defined below is yes (b ranges over B).

$$Vb = \big\{(v, v) \;\big|\; v \in Ab\big\},$$
$$Cb = \big\{(X, l) \in T_p(Ab) \times T_m(Ab) \;\big|\; X = \text{the set of elements in } l\big\}.$$

6. Fibrations for logical relations

We consider the general situation where the underlying categories of λ_c-structures are general bi-CCCs. To formulate the concept of a relation between two objects from different categories, we first formulate the concept of a predicate over an object in an arbitrary category in terms of *fibrational category theory*, then derive the concept of a relation as a predicate over a product category.

We give a brief definition of fibrations and related concepts; see [12] for the full details. Let $p : \mathbb{E} \to \mathbb{C}$ be a functor. We call its domain ($\mathbb{E}$) *total category*. We say that $X \in \mathbb{E}$ is *above* $I \in \mathbb{C}$ if $pX = I$, and similarly for morphisms. The *fibre category* over $I \in \mathbb{C}$ is the subcategory $\mathbb{E}_I$ of $\mathbb{E}$ consisting of objects above $I \in \mathbb{C}$ and morphisms above id_I.

We next assume that p is faithful. Then each fibre category $\mathbb{E}_I$ is a preorder. In this situation, we regard the objects above I as predicates on I, and $\mathbb{E}_I$ as the preorder of predicates on I. We now introduce a notation frequently used in this paper. For $X, Y \in \mathbb{E}$ and $f : pX \to pY$, we define $f : X \dot\to Y$ to be the following proposition:

$$\exists \dot f : X \to Y \,.\, p(\dot f) = f.$$

Its intuitive meaning is "the f-image of any element satisfying X satisfies Y". When $f : X \dot\to Y$ holds, the morphism $\dot f$ that exists above f is unique. We call it the *witness* of $f : X \dot\to Y$. We note that $X \leqslant Y$ holds in the fibre preorder $\mathbb{E}_I$ if and only if $\mathrm{id}_I : X \dot\to Y$.

Definition 3. A *partial order bifibration with fibrewise small products* is a faithful functor $p : \mathbb{E} \to \mathbb{C}$ such that:

(**Partial Order**) Each fibre category is a partial order.

(**Fibration**)[1] For any $I \in \mathbb{C}$, $Y \in \mathbb{E}$ and $f : I \to pY$, there exists $X \in \mathbb{E}$ above I such that $f : X \dot\to Y$ and the following property holds: for any $Z \in \mathbb{E}$ and $g : pZ \to I$, $f \circ g : Z \dot\to Y$ implies $g : Z \dot\to X$.
This property and $\mathbb{E}_I$ being a partial order imply that X is unique; hence we write f^*Y for X, and $\overline{f}Y$ for the witness of $f : f^*Y \dot\to Y$. Furthermore, for any $f : I \to J$ in $\mathbb{C}$, the mapping $Y \in \mathbb{E}_J \mapsto f^*Y \in \mathbb{E}_I$ extends to a functor $f^* : \mathbb{E}_J \to \mathbb{E}_I$. We call it the *inverse image functor* (along f). Intuitively, f^*Y corresponds to the predicate $\{i \in I \mid f(i) \in Y\}$ on I.

(**Bi-**) Each inverse image functor f^* has a left adjoint called the *direct image functor* (along f), denoted by f_*. Intuitively, for $X \in \mathbb{E}$ and $f : pX \to J$ in $\mathbb{C}$, f_*X corresponds to the predicate $\{f(x) \mid x \in X\}$ on J.

(**Fibrewise Small Products**) Each fibre category has small products, denoted by $\bigwedge$, and the inverse image functors (necessarily) preserve them.

We show a couple of properties of partial order bifibrations $p : \mathbb{E} \to \mathbb{C}$ with fibrewise small products.

[1] This definition of fibration exploits the assumption that p is faithful. The concept of a fibration is defined differently for an arbitrary functor [12, Definition 1.1.3].

Lemma 1. *For any $Y, Z \in \mathbb{E}$, $I \in \mathbb{C}$ and morphisms $pZ \xrightarrow{g} I \xrightarrow{f} pY$ in $\mathbb{C}$, we have $f \circ g : Z \dot{\to} Y \Leftrightarrow g : Z \dot{\to} f^*Y$.*

Proof. ($\Rightarrow$) Immediate from the definition of partial order bifibration. ($\Leftarrow$) As $f : f^*Y \dot{\to} Y$, we have $f \circ g : Z \dot{\to} Y$. $\square$

By letting $pZ = I$ and $g = \mathrm{id}_I$, we obtain $f : Z \dot{\to} Y \Leftrightarrow Z \leqslant f^*Y$ as a corollary of this lemma. Note that in the fibration $\pi : \mathbf{Pred} \to \mathbf{Set}$, this property translates to "the f-image of elements satisfying X satisfies Y iff X is a subset of the inverse image $f^{-1}Y$". We also obtain $(g \circ f)^* = f^* \circ g^*$, because for all $Z, Y \in \mathbb{E}$, we have

$$Z \leqslant (g \circ f)^*Y \quad \Leftrightarrow \quad g \circ f : Z \dot{\to} Y \quad \Leftrightarrow \quad f : Z \dot{\to} g^*Y \quad \Leftrightarrow \quad Z \leqslant f^*\big(g^*Y\big),$$

hence by letting Z be $(g \circ f)^*Y$ and $f^*(g^*Y)$ we obtain the equation.

Lemma 2. *For any $Y, Z \in \mathbb{E}$, $I \in \mathbb{C}$ and morphisms $pZ \xrightarrow{g} I \xrightarrow{f} pY$ in $\mathbb{C}$, we have $f \circ g : Z \dot{\to} Y \Leftrightarrow g : f_*Z \dot{\to} Y$.*

Proof. Immediate from the assumption that f_* is the left adjoint to f^*. $\square$

Lemma 3. *For any $f : I \to J$ in $\mathbb{C}$ and a set-indexed family $\{Y_i\}_{i \in \Lambda}$ of objects in $\mathbb{E}_J$, we have $(\forall i \in \Lambda \,.\, f : X \dot{\to} Y_i) \Leftrightarrow f : X \dot{\to} \bigwedge_{i \in \Lambda} Y_i$.*

Proof. We use Lemma 1 and the preservation of $\bigwedge$ by the inverse image functor f^*.

$$\begin{aligned}\forall i \in \Lambda \,.\, f : X \dot{\to} Y_i \quad &\Leftrightarrow \quad \forall i \in \Lambda \,.\, X \leqslant f^*Y_i \\ &\Leftrightarrow \quad X \leqslant \bigwedge_{i \in \Lambda} f^*Y_i = f^*\Big(\bigwedge_{i \in \Lambda} Y_i\Big) \\ &\Leftrightarrow \quad f : X \dot{\to} \bigwedge_{i \in \Lambda} Y_i. \qquad \square\end{aligned}$$

In the categorical semantics of type theories and programming languages, the concept of a logical relation for a type constructor, such as a sum or a function space, is often provided by the categorical structure corresponding to it, such as a coproduct or an exponential, in the category $\mathbb{E}$ of predicates, and moreover the corresponding categorical structure is *strictly preserved* by a functor $p : \mathbb{E} \to \mathbb{C}$. This is the central observation in [20,23], and Hermida showed that a bi-CC structure on the total category $\mathbb{E}$ of a fibration $p : \mathbb{E} \to \mathbb{C}$ can be constructed when p is a (bi)fibration with certain structures [11].

When solving the effect simulation problem, we will use logical relations for products, sums and arrow types. We thus consider the fibrations whose total categories have bi-CC structures that are strictly preserved by the fibrations. We call them *fibrations for logical relations* in this paper.

Definition 4. A *fibration for logical relations* over a bi-CCC $\mathbb{C}$ is a partial order bifibration $p : \mathbb{E} \to \mathbb{C}$ with fibrewise small products[2] such that $\mathbb{E}$ is a bi-CCC and p strictly preserves the bi-CC structure.

Notational convention: we write the components of the bi-CC structure on $\mathbb{E}$ with dots on top, like $\dot{1}, \dot{!}, \dot{0}, \dot{?}, \dot{\times}, \dot{\langle} -, - \dot{\rangle}, \dot{\pi}_i, \dot{+}, \dot{[} -, - \dot{]}, \dot{\iota}_i, \dot{\Rightarrow}, \dot{ev}, \dot{\lambda}(-)$.

In [12, Section 9.2], constructing a bi-CC structure on the total category of a fibration is discussed. The following proposition summarises the relevant constructions presented there into one:

Proposition 4. *Let $\mathbb{C}$ be a bi-CCC and $p : \mathbb{E} \to \mathbb{C}$ be a partial order bifibration such that each fibre partial order category $\mathbb{E}_I$ is a bi-CCC with small products, every reindexing functor f^* (strictly) preserves the bi-CC structure, and p has simple products, that is, the inverse image functor along the first projection $\pi_1 : I \times J \to I$ has a right adjoint $\forall_{I,J} : \mathbb{E}_{I \times J} \to \mathbb{E}_I$ satisfying the Beck–Chevalley condition. Then p is a fibration for logical relations.*

Proof. We write $(\top, \wedge, \bot, \vee, \supset)$ for the bi-CC structure on each fibre category. We define the bi-CC structure on $\mathbb{E}$ as follows:

$$\begin{aligned}&\dot{1} = \top_1, \\ &\dot{0} = \bot_0, \\ &X \mathbin{\dot{\times}} Y = \pi_1^*X \wedge \pi_2^*Y,\end{aligned}$$

[2] We actually only use fibrewise products up to the cardinality of a set B of base types.

$$X \dot{+} Y = (\iota_1)_* X \vee (\iota_2)_* Y,$$
$$X \dot{\Rightarrow} Y = \forall_{pX \Rightarrow pY, pX}\big(\pi_2^* X \supset ev^* Y\big).$$

For the verification of the bi-CC structure, consult [12, Section 9.2]. □

Corollary 5. *The subobject fibration of the presheaf category* $[\mathbb{C}, \mathbf{Set}]$ *over a small category* $\mathbb{C}$ *is a fibration for logical relations.*

Proof. The fibration satisfy all the conditions mentioned in Proposition 4; see [12]. □

Example 2. (See [12, Chapter 0].) We define the category **Pred** by the following data: an object in **Pred** is a pair (X, I) where X is a subset of I, and a morphism from (X, I) to (Y, J) is a function $f : I \to J$ such that for any $i \in X$, $f(i) \in Y$. This category is isomorphic to the category of subobjects of **Set** [12, Section 1.3]. We now consider the functor $\pi : \mathbf{Pred} \to \mathbf{Set}$ defined by $\pi(X, I) = I$ and $\pi(f) = f$. This is faithful, and the fibre category $\mathbf{Pred}_I$ is isomorphic to the poset $(2^I, \subseteq)$. One can easily check that the following gives the inverse image functor along a function $f : I \to J$ and its left adjoint, the direct image functor:

$$f^*(Y, J) = \big(\{x \mid f(x) \in Y\}, I\big), \qquad f_*(X, I) = \big(\{f(x) \mid x \in X\}, J\big).$$

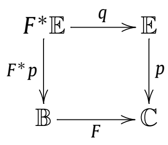
Finally, every fibre category has small products given by the intersection. Therefore π is a partial order bifibration with fibrewise small products.

From Corollary 5, we obtain the bi-CC structure over **Pred** that is strictly preserved by π; see also [12, Exercise 9.2.1]. The object part of the bi-CC structure is given as follows:

$$\dot{1} = (1, 1),$$
$$\dot{0} = (0, 0),$$
$$(X, I) \dot{\times} (Y, J) = \big(\{v \mid \pi_1(v) \in X,\ \pi_2(v) \in Y\}, I \times J\big),$$
$$(X, I) \dot{+} (Y, J) = \big(\{\iota_1(x) \mid x \in X\} \cup \{\iota_2(y) \mid y \in Y\}, I + J\big),$$
$$(X, I) \dot{\Rightarrow} (Y, J) = \big(\{f \mid \forall x \in X\ .\ f(x) \in Y\}, I \Rightarrow J\big).$$

To summarise, π is a fibration for logical relations.

A useful construction of fibrations for logical relations is by *pullback* (*change-of-base*) along finite-product preserving functors. In this paper, we specify the pullback $(F^*\mathbb{E}, F^*p, q)$ of a faithful functor $p : \mathbb{E} \to \mathbb{C}$ along $F : \mathbb{B} \to \mathbb{C}$ (see diagram below)


$$\begin{tikzcd} F^*\mathbb{E} \arrow[r, "q"] \arrow[d, "F^*p"'] & \mathbb{E} \arrow[d, "p"] \\ \mathbb{B} \arrow[r, "F"'] & \mathbb{C} \end{tikzcd}$$

by the following data: an object in $F^*\mathbb{E}$ is a pair (X, I) such that $X \in \mathbb{E}$, $I \in \mathbb{B}$ and X is above FI, and a morphism from (X, I) to (Y, J) is a morphism $f : I \to J$ such that $Ff : X \dot{\to} Y$. The functor F^*p sends (X, I) to I, and f to itself, while q sends (X, I) to X and f to its witness $\dot{f}$. We note that F^*p is again faithful, and the fibre category of $(F^*\mathbb{E})_I$ is isomorphic to $\mathbb{E}_{FI}$.

Proposition 6. *Let* $\mathbb{B}, \mathbb{C}$ *be bi-CCCs,* $p : \mathbb{E} \to \mathbb{C}$ *be a fibration for logical relations and* $F : \mathbb{B} \to \mathbb{C}$ *be a finite-product preserving functor. Then* $F^*p : F^*\mathbb{E} \to \mathbb{B}$ *is a fibration for logical relations.*

Proof. The following gives the inverse image functor and the direct image functor along $f : I \to J$ in $\mathbb{B}$:

$$f^*(Y, J) = \big(f^*Y, J\big), \qquad f_*(X, I) = (f_*X, I).$$

Each fibre category $(F^*\mathbb{E})_I$ has small products as it is isomorphic to the partial order category $\mathbb{E}_{FI}$. We only present the object part of the bi-CC structure in $F^*\mathbb{E}$.

$$\ddot{1} = \big((!_{F1})^*\dot{1}, 1\big),$$
$$\ddot{0} = \big((!_{F0})_*\dot{0}, 0\big),$$
$$(X, I) \ddot{\times} (Y, J) = \big(l^*(X \dot{\times} Y), I \times J\big),$$
$$(X, I) \ddot{+} (Y, J) = \big(m_*(X \dot{+} Y), I + J\big),$$
$$(X, I) \ddot{\Rightarrow} (Y, J) = \big(n^*(X \dot{\Rightarrow} Y), I \Rightarrow J\big).$$

Here, l, m, n are the canonical morphisms from Section 3. □

Example 3. Three more examples are depicted below.

$$\begin{tikzcd}
\mathbf{SubSc}(\mathbb{C}) \arrow[r] \arrow[d, "p"'] \arrow[dr, phantom, "(1)"] & \mathbf{Pred} \arrow[d, "\pi"] & \mathbf{BRel} \arrow[r] \arrow[d, "p"'] \arrow[dr, phantom, "(2)"] & \mathbf{Pred} \arrow[d, "\pi"] & \mathbb{K}_F \arrow[r] \arrow[d, "p"'] \arrow[dr, phantom, "(3)"] & \mathbf{Sub}([\mathbb{B}, \mathbf{Set}]) \arrow[d] \\
\mathbb{C} \arrow[r, "\mathbb{C}(1,-)"'] & \mathbf{Set} & \mathbf{Set}^2 \arrow[r, "\times"'] & \mathbf{Set} & \mathbb{C} \arrow[r, "F"'] & {[\mathbb{B}, \mathbf{Set}]}
\end{tikzcd}$$

1. The *subscone* [23] over a bi-CCC $\mathbb{C}$ is the category $\mathbf{SubSc}(\mathbb{C})$ at the vertex of the pullback of $\pi : \mathbf{Pred} \to \mathbf{Set}$ along the global element functor $\mathbb{C}(1,-) : \mathbb{C} \to \mathbf{Set}$. The leg $p : \mathbf{SubSc}(\mathbb{C}) \to \mathbb{C}$ is a fibration for logical relations.
2. We pullback π along the binary product functor $\times : \mathbf{Set}^2 \to \mathbf{Set}$. The category $\mathbf{BRel}$ at the vertex of the pullback is explicitly described as follows: an object is a tuple (X, I, I') where $X \subseteq I \times I'$, and a morphism from (X, I, I') to (Y, J, J') is a pair (f, g) of functions $f : I \to J$ and $g : I' \to J'$ such that $f \times g : X \dot{\rightarrow} Y$. The leg $p : \mathbf{BRel} \to \mathbf{Set}^2$ is a fibration for logical relations.
3. By pulling back the subobject fibration of a presheaf category along a finite-product preserving functor $F : \mathbb{C} \to [\mathbb{B}, \mathbf{Set}]$, we obtain a fibration $p : \mathbb{K}_F \to \mathbb{C}$ for logical relations. This construction will be employed in Sections 10 and 11. An object in $\mathbb{K}_F$ is a pair (X, I) of an object I in $\mathbb{C}$ and a subpresheaf X of FI, and a morphism $f : (X, I) \to (Y, J)$ in $\mathbb{K}_F$ is a $\mathbb{C}$-morphism such that $Ff : X \dot{\rightarrow} Y$; this is equivalent to that for any $H \in \mathbb{B}$ and $x \in XH$, $Ff(x) \in YH$.
   The object part of the bi-CC structure on $\mathbb{K}_F$ is given as follows (here we use a meta-lambda notation $\lambda H \in \mathbb{B} . \{\ldots\}$ to define the presheaves on objects):

$$\begin{aligned}
&\dot{1} = \big(\lambda H \in \mathbb{B} . \{*\}, 1\big),\\
&\dot{0} = (\lambda H \in \mathbb{B} . \emptyset, 0),\\
&(X, I) \mathbin{\dot{\times}} (Y, J) = \big(\lambda H \in \mathbb{B} . \big\{v \bigm| F\pi_1(v) \in XH,\ F\pi_2(v) \in YH\big\}, I \times J\big),\\
&(X, I) \mathbin{\dot{+}} (Y, J) = \big(\lambda H \in \mathbb{B} . \big\{F\iota_1(v) \bigm| v \in XH\big\} \cup \big\{F\iota_2(v) \bigm| v \in YH\big\}, I + J\big),\\
&(X, I) \mathbin{\dot{\Rightarrow}} (Y, J) = \big(\lambda H \in \mathbb{B} . \big\{f \bigm| \forall H' \in \mathbb{B},\ h : H \to H',\ x \in XH' .\\
&\qquad\qquad F(ev)_{H'} \circ \big(l^{-1}_{I \Rightarrow J, I}\big)_{H'}\big(F(I \Rightarrow J)(h)(f), x\big) \in YH'\big\}, I \Rightarrow J\big).
\end{aligned}$$

   This bi-CC structure is a general form of Kripke logical relations with varying arity [13].

7. Fibrational effect property problem

Having abstracted the concept of a predicate in terms of fibrational category theory, we consider the *fibrational effect property problem*, which subsumes the effect simulation problem. Let $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ be a $\lambda_c(\Sigma)$-structure and $p : \mathbb{E} \to \mathbb{C}$ be a fibration for logical relations. A *property* over $\mathcal{A}$ is a pair (V, C) of functors $V, C : B \to \mathbb{E}$ such that $p \circ V = A$ and $p \circ C = T \circ A$, that is, for any $b \in B$, Vb is above Ab and Cb is above $T(Ab)$. The fibrational effect property problem is:

> Suppose that a property (V, C) over $\mathcal{A}$ is given. Then does any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$ satisfy $\mathcal{A}[\![M]\!] : \dot{\prod}_{i=1}^{n} Vb_i \dot{\rightarrow} Cb$?

We then define the *fibrational effect simulation problem* between two $\lambda_c(\Sigma)$-structures $\mathcal{A}_1$ and $\mathcal{A}_2$ to be the fibrational effect property problem over $\mathcal{A}_1 \times \mathcal{A}_2$. We also call the property over $\mathcal{A}_1 \times \mathcal{A}_2$ *simulation* between $\mathcal{A}_1$ and $\mathcal{A}_2$. The set-theoretic effect simulation problem in the beginning of Section 5 uses $q : \mathbf{BRel} \to \mathbf{Set}^2$ in Example 3 as the fibration for logical relations.

The choice of a fibration for logical relations depends on the property we would like to establish on the semantics of $\lambda_c(\Sigma)$. For instance, when the property is parametrised by a category $\mathbb{B}$, the concept of a predicate with which we can naturally express the property would be provided by the pullback calculated in Example 3.3. This is indeed the case when comparing two monadic semantics of the ν-calculus in Section 11. At this moment, however, there seems no general guideline for choosing the fibration for logical relations.

When a property (V, C) is given, we extend V to the functor of type $\mathcal{V}[\![-]\!] : \mathrm{GTyp}(B) \to \mathbb{E}$ in the same way as we did in Definition 2. We note that for all $\rho \in \mathrm{GTyp}(B)$, $\mathcal{V}[\![\rho]\!]$ is above $\mathcal{A}[\![\rho]\!]$.

Theorem 7. *Let Σ be a λ_c-signature, $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ be a $\lambda_c(\Sigma)$-structure, $p : \mathbb{E} \to \mathbb{C}$ be a fibration for logical relations and (V, C) be a property over $\mathcal{A}$. If the property satisfies the following three conditions*:

(I) *for all $b \in B$, $\eta_{Ab} : Vb \dot{\rightarrow} Cb$,*
(C1) *for all $k \in K$, $a(k) : \mathcal{V}[\![ar(k)]\!] \dot{\rightarrow} \mathcal{V}[\![car(k)]\!]$, and*
(C2) *for all $o \in O$ and $b \in B$, $a(o)_{Ab} : (\mathcal{V}[\![ar(o)]\!] \mathbin{\dot{\Rightarrow}} Cb) \dot{\rightarrow} (\mathcal{V}[\![car(o)]\!] \mathbin{\dot{\Rightarrow}} Cb)$,*

then the answer of the effect property problem for $\lambda_c(\Sigma)$ is yes: for all well-typed $\lambda_c(\Sigma)$-terms $x_1 : b_1, \dots, x_n : b_n \vdash M : b$, we have $\mathcal{A}[\![M]\!] : \prod_{i=1}^{n} Vb_i \dot{\to} Cb$.

Our proof of this theorem hinges on the technique called *categorical* $\top\top$*-lifting* [14], which is a semantic formulation of Lindley and Stark's leapfrog method [18,19]. After reviewing this technique in the next section, we prove Theorem 7 in Section 9.

8. Categorical $\top\top$-lifting

Let $\mathbb{C}$ be a bi-CCC, $\mathcal{T}$ be a strong monad over $\mathbb{C}$ and $p : \mathbb{E} \to \mathbb{C}$ be a fibration for logical relations. We formulate the concept of a logical relation for $\mathcal{T}$ as a strong monad $\dot{\mathcal{T}} = (\dot{T}, \dot{\eta}, \dot{\mu}, \dot{\theta})$ over $\mathbb{E}$ such that

$$p(\dot{T}A) = T(pA), \qquad p\dot{\eta}_X = \eta_{pX}, \qquad p\dot{\mu}_X = \mu_{pX}, \qquad p\dot{\theta}_{X,Y} = \theta_{pX,pY}.$$

We call such $\dot{\mathcal{T}}$ a *lifting* of $\mathcal{T}$.

The main technical tool we use is the *categorical* $\top\top$*-lifting* in [14], which we review below. It is a method to construct a lifting of $\mathcal{T}$, and it requires as a parameter a pair $\mathcal{R} = (R, S)$ of functors $R : \Lambda \to \mathbb{C}$, $S : \Lambda \to \mathbb{E}$ from a set Λ such that $T \circ R = p \circ S$. In other words, the parameter is a family of objects $\{(Ri, Si)\}_{i \in \Lambda}$ such that Si is above $T(Ri)$. The $\top\top$-lifting of $\mathcal{T}$ with respect to $\mathcal{R}$ is defined as follows. Let $X \in \mathbb{E}$ be above $I \in \mathbb{C}$. We first define $X^{\top\top(i)} \in \mathbb{E}$ above TI to be the inverse image of $(X \dot{\Rightarrow} Si) \dot{\Rightarrow} Si$ along $\sigma_I^{\mathcal{T},Ri}$:

$$\begin{tikzcd}
X^{\top\top(i)} \arrow[r, dotted] & (X \dot{\Rightarrow} Si) \dot{\Rightarrow} Si & \mathbb{E} \arrow[d, "p"] \\
TI \arrow[r, "\sigma_I^{\mathcal{T},Ri}"'] & (I \Rightarrow T(Ri)) \Rightarrow T(Ri) & \mathbb{C}
\end{tikzcd}$$

We then define $\dot{T}X$ by the following fibrewise product:

$$\dot{T}X = \bigwedge_{i \in \Lambda} X^{\top\top(i)}. \tag{1}$$

Proposition 8. *(See [14].) Let $X, Y \in \mathbb{E}$.*

1. *For all morphism f in $\mathbb{C}$, $f : X \dot{\to} Y$ implies $Tf : \dot{T}X \dot{\to} \dot{T}Y$.*
2. *We have $\eta_{pX} : X \dot{\to} \dot{T}X$, $\mu_{pX} : \dot{T}\dot{T}X \dot{\to} \dot{T}X$ and $\theta_{pX,pY} : X \dot{\times} \dot{T}Y \dot{\to} \dot{T}(X \dot{\times} Y)$.*

From Proposition 8.1, for a morphism $\dot{f} : X \to Y$ in $\mathbb{E}$, we define $\dot{T}\dot{f}$ to be the witness of $T(p\dot{f}) : \dot{T}X \dot{\to} \dot{T}Y$. We similarly define morphisms $\dot{\eta}_X, \dot{\mu}_X, \dot{\theta}_{X,Y}$ in $\mathbb{E}$ to be the witnesses of the statements in Proposition 8.2. The fact that p is faithful and T is a monad implies immediately that all the monad laws hold for $\dot{T}$.

Proposition 9. *(See [14].) The assignment $X \mapsto \dot{T}X$ defined by (1) uniquely extends to a lifting $\dot{\mathcal{T}} = (\dot{T}, \dot{\eta}, \dot{\mu}, \dot{\theta})$ of $\mathcal{T}$.*

Below we show some properties about $\dot{\mathcal{T}}$ that are essential to our proof of Theorem 7. The first proposition gives a sufficient condition for $\dot{T}X$ to be included in Si.

Proposition 10. *For any $i \in \Lambda$ and $X \in \mathbb{E}$ above $Ri \in \mathbb{C}$, $\eta_{Ri} : X \dot{\to} Si$ implies $\dot{T}X \leqslant Si$.*

Proof. As $\dot{T}X = \bigwedge_{i \in \Lambda} T^{\top\top(i)}X$, it is sufficient to show $T^{\top\top(i)}X \leqslant Si$. Let us write $\dot{\eta} : X \to Si$ for the witness of $\eta_{Ri} : X \dot{\to} Si$. Then in $\mathbb{E}$ we obtain a morphism

$$\begin{tikzcd}[column sep=huge]
T^{\top\top(i)}X \arrow[r, "\overline{\sigma_{Ri}^{\mathcal{T},Ri}}((X \dot{\Rightarrow} Si) \dot{\Rightarrow} Si)"] & (X \dot{\Rightarrow} Si) \dot{\Rightarrow} Si \arrow[r, "\dot{ev} \circ \langle \mathrm{id}, \mathrm{nm}(\dot{\eta}) \circ \dot{!} \rangle"] & Si
\end{tikzcd}$$

which is above $ev \circ \langle \mathrm{id}, \mathrm{nm}(\eta_{Ri}) \circ ! \rangle \circ \sigma_{Ri}^{\mathcal{T},Ri} = \mathrm{id}_{TRi}$. Therefore $T^{\top\top(i)}X \leqslant Si$. □

The next theorem characterises when we can lift algebraic operations / generic effects for $\mathcal{T}$ to the $\top\top$-lifted monad $\dot{\mathcal{T}}$.

Theorem 11. *Let $p : \mathbb{E} \to \mathbb{C}$ be a fibration for logical relations, $\mathcal{T}$ be a strong monad on $\mathbb{C}$ and $\mathcal{R} = (R : \Lambda \to \mathbb{C}, S : \Lambda \to \mathbb{E})$ be a parameter for $\top\top$-lifting. For any $D, C \in \mathbb{E}$ and $\alpha \in \mathrm{Alg}(\mathcal{T}, pD, pC)$, the following are equivalent.*

1. *For all $i \in \Lambda$, $\alpha_{Ri} : (D \dot{\Rightarrow} Si) \dot{\to} (C \dot{\Rightarrow} Si)$.*
2. $\mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \dot{T} D$.
3. *There is an algebraic operation $\dot{\alpha} \in \mathrm{Alg}(\dot{\mathcal{T}}, D, C)$ such that $\dot{\alpha}_X$ is above α_{pX} (such $\dot{\alpha}$ is unique from the faithfulness of p).*
4. *For all $X \in \mathbb{E}$, $\alpha_{pX} : (D \dot{\Rightarrow} \dot{T} X) \dot{\to} (C \dot{\Rightarrow} \dot{T} X)$.*

This theorem says that in order to check that an algebraic operation has a lifting (Theorem 11.3), or the corresponding generic effect is related by $\dot{T}$ (Theorem 11.2), it is necessary and sufficient to check that the parameter (R, S) of the $\top\top$-lifting is closed under algebraic effects (Theorem 11.1).

Proof of Theorem 11. First, as p strictly preserves the bi-CC structure and $\dot{\mathcal{T}}$ is a lifting of $\mathcal{T}$, we have

$$p\big(\mathrm{sw}(f)\big) = \mathrm{sw}(pf), \qquad p\big(\mathrm{nm}(f)\big) = \mathrm{nm}(pf), \qquad p\big(\mathrm{Aop}^{\dot{\mathcal{T}}}(e)_X\big) = \mathrm{Aop}^{\mathcal{T}}(pe)_{pX}.$$

$(1 \Leftrightarrow 2)$ We have:

$$\begin{aligned}
&\forall i \in \Lambda \,.\, \alpha_{Ri} : (D \dot{\Rightarrow} Si) \dot{\to} (C \dot{\Rightarrow} Si) \\
\Leftrightarrow\quad &\forall i \in \Lambda \,.\, \sigma^{\mathcal{T},Ri}_{pD} \circ \mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \big((D \dot{\Rightarrow} Si) \dot{\Rightarrow} Si\big) \\
\Leftrightarrow\quad &\forall i \in \Lambda \,.\, \mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \big(\sigma^{\mathcal{T},Ri}_{pD}\big)^{*}\big((D \dot{\Rightarrow} Si) \dot{\Rightarrow} Si\big) \\
\Leftrightarrow\quad &\mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \bigwedge_{i \in \Lambda} \big(\sigma^{\mathcal{T},Ri}_{pD}\big)^{*}\big((D \dot{\Rightarrow} Si) \dot{\Rightarrow} Si\big) \\
\Leftrightarrow\quad &\mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \dot{T} D.
\end{aligned}$$

$(2 \Rightarrow 3)$ Let $\dot{e} : C \to \dot{T} D$ in $\mathbb{E}$ be the witness of $\mathrm{Gef}^{\mathcal{T}}(\alpha) : C \dot{\to} \dot{T} D$. Then we have $p(\mathrm{Aop}^{\dot{\mathcal{T}}}(\dot{e})_X) = \mathrm{Aop}^{\mathcal{T}}(\mathrm{Gef}^{\mathcal{T}}(\alpha))_{pX} = \alpha_{pX}$, meaning that $\mathrm{Aop}^{\dot{\mathcal{T}}}(\dot{e})$ is the one in question. $(3 \Rightarrow 4)$ Immediate. $(4 \Rightarrow 2)$ Let $\dot{\alpha}_D$ be the witness of $\alpha_{pD} : (D \dot{\Rightarrow} \dot{T} D) \dot{\to} (C \dot{\Rightarrow} \dot{T} D)$. Then the morphism $\mathrm{nm}^{-1}_{C,\dot{T}D}(\dot{\alpha}_D \circ \mathrm{nm}_{D,\dot{T}D}(\dot{\eta}_D)) : C \dot{\to} \dot{T} D$ in $\mathbb{E}$ is above $\mathrm{Gef}^{\mathcal{T}}(\alpha)$. □

9. The proof of Theorem 7

We are ready to prove Theorem 7. We consider the $\top\top$-lifting of $\mathcal{T}$ with the pair $\mathcal{R} = (A : B \to \mathbb{C}, C : B \to \mathbb{E})$ of functors from the set B of base types as the parameter. From Proposition 8, we obtain a strong monad $\dot{\mathcal{T}}$ over $\mathbb{E}$ which is a lifting of $\mathcal{T}$. We now define an assignment $\dot{a}$ as follows:

1. For each $k \in K$, we define $\dot{a}(k)$ to be the witness of $a(k) : \mathcal{V}[\![ar(k)]\!] \dot{\to} \mathcal{V}[\![car(k)]\!]$ that exists from the assumption (C1).
2. For each $o \in O$, we define $\dot{a}(o)$ to be the unique $(\mathcal{V}[\![ar(o)]\!], \mathcal{V}[\![car(o)]\!])$-ary algebraic operation for $\dot{\mathcal{T}}$ above $a(o)$ that exists from the assumption (C2) and Theorem 11.3.

Therefore the tuple $\mathcal{V} = (\mathbb{E}, \dot{\mathcal{T}}, V, \dot{a})$ is a $\lambda_c(\Sigma)$-structure. As p preserves all relevant categorical structures, induction on terms shows that for all well-typed $\lambda_c(\Sigma)$-terms $x_1 : \rho_1, \ldots, x_n : \rho_n \vdash M : \rho$, we have

$$\mathcal{A}[\![M]\!] : \dot{\prod}_{i=1}^{n} \mathcal{V}[\![\rho_i]\!] \dot{\to} \dot{T}\mathcal{V}[\![\rho]\!];$$

its witness is $\mathcal{V}[\![M]\!]$. Particularly, when all types are base types, we obtain

$$\mathcal{A}[\![M]\!] : \dot{\prod}_{i=1}^{n} V b_i \dot{\to} \dot{T}(V b) \leqslant C b.$$

The last inequality is from the assumption (I) and Proposition 10.

10. Effect simulation by a monad morphism

Monadic semantics are often related by *strong monad morphisms.*

Definition 5. Let Σ be a λ_c-signature, $\mathcal{A}_1 = (\mathbb{C}, \mathcal{T}_1, A, a)$ be a $\lambda_c(\Sigma)$-structure, $\mathcal{T}_2$ be a strong monad on $\mathbb{C}$ and $\gamma : \mathcal{T}_1 \to \mathcal{T}_2$ be a strong monad morphism. We define the *image* of $\mathcal{A}_1$ along γ to be the $\lambda_c(\Sigma)$-structure $\gamma\mathcal{A}_1 = (\mathbb{C}, \mathcal{T}_2, A, \gamma(a))$, where $\gamma(a)$ is the following assignment:

$$\begin{aligned}
&\gamma(a)(k) = a(k) \quad (k \in K), \\
&\gamma(a)(o) = \mathrm{Alg}\big(\gamma, \mathcal{A}[\![ar(o)]\!], \mathcal{A}[\![car(o)]\!]\big)\big(a(o)\big) \quad (o \in O).
\end{aligned}$$

Theorem 12. *Let Σ be a λ_c-signature, $\mathcal{A} = (\mathbb{C}, \mathcal{T}_1, A, a)$ be a $\lambda_c(\Sigma)$-structure, $\mathcal{T}_2$ be a strong monad on $\mathbb{C}$ and $\gamma : \mathcal{T}_1 \to \mathcal{T}_2$ be a strong monad morphism. Then for any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \dots, x_n : b_n \vdash M : b$, we have $\gamma_{Ab} \circ \mathcal{A}[\![M]\!] = (\gamma\mathcal{A})[\![M]\!]$.*

Proof. We assume that $\mathbb{C}$ is small without loss of generality; if not, we confine ourselves to a small full subcategory of $\mathbb{C}$ that contains $Ab, 1, 0$ and is closed under $\times, +, \Rightarrow, T_1, T_2$.

We pullback the subobject fibration $\mathbf{Sub}([\mathbb{C}^{op}, \mathbf{Set}]) \to [\mathbb{C}^{op}, \mathbf{Set}]$ along the finite-product preserving functor $D : \mathbb{C}^2 \to [\mathbb{C}^{op}, \mathbf{Set}]$ defined by $D(I, J) = \mathbf{y}I \times \mathbf{y}J$; here $\mathbf{y} : \mathbb{C} \to [\mathbb{C}^{op}, \mathbf{Set}]$ is the Yoneda embedding.

$$\begin{tikzcd}
\mathbb{K} \arrow[r] \arrow[d, "q"'] & \mathbf{Sub}([\mathbb{C}^{op}, \mathbf{Set}]) \arrow[d] \\
\mathbb{C}^2 \arrow[r, "D"'] & {[\mathbb{C}^{op}, \mathbf{Set}]}
\end{tikzcd}$$

From Proposition 6, the leg $q : \mathbb{K} \to \mathbb{C}^2$ in the above diagram is a fibration for logical relations. An object in $\mathbb{K}$ is a triple (X, I, I') where $I, I' \in \mathbb{C}$ and X is a subpresheaf of $\mathbf{y}I \times \mathbf{y}I'$. A morphism from (X, I, I') to (Y, J, J') is a pair (f, g) of $f : I \to J$ and $g : I' \to J'$ in $\mathbb{C}$ such that for any $H \in \mathbb{C}$ and $(x_1, x_2) \in XH$, we have $(f \circ x_1, g \circ x_2) \in YH$. The exponential in $\mathbb{K}$ is given as follows (see Example 3.3):

$$\begin{aligned}
(X, I, I') \dot{\Rightarrow} (Y, J, J') = \bigl(\lambda H \in \mathbb{C}\,.\,\bigl\{(f, g) \in \mathbb{C}(H, I \Rightarrow J) \times \mathbb{C}(H, I' \Rightarrow J') \bigm| \\
\forall H' \in \mathbb{C},\ h : H' \to H,\ (x, y) \in XH'\,.\,\bigl(\mathrm{ev} \circ \langle f \circ h, x\rangle, \mathrm{ev} \circ \langle g \circ h, y\rangle\bigr) \in YH'\bigr\}, \\
I \Rightarrow J, I' \Rightarrow J'\bigr).
\end{aligned}$$

We define a simulation (V, C) between $\mathcal{A}$ and $\gamma\mathcal{A}$ by

$$\begin{aligned}
Vb &= \bigl(\lambda H \in \mathbb{C}\,.\,\bigl\{(f, f) \in \mathbb{C}(H, Ab) \times \mathbb{C}(H, Ab) \bigm| f : H \to Ab\bigr\}, Ab, Ab\bigr), \\
Cb &= \bigl(\lambda H \in \mathbb{C}\,.\,\bigl\{(f, \gamma_{Ab} \circ f) \in \mathbb{C}\bigl(H, T_1(Ab)\bigr) \times \mathbb{C}\bigl(H, T_2(Ab)\bigr) \bigm| f : H \to T_1(Ab)\bigr\}, T_1(Ab), T_2(Ab)\bigr).
\end{aligned}$$

They are objects in $\mathbb{K}$, and satisfy (I) and (C1). For (C2), below we let $d = ar(o)$ and $c = car(o)$. We define an auxiliary object $Cd \in \mathbb{K}$ by

$$Cd = \bigl(\lambda H \in \mathbb{C}\,.\,\bigl\{\bigl(f, \gamma_{\mathcal{A}[\![d]\!]} \circ f\bigr) \in \mathbb{C}\bigl(H, T_1\mathcal{A}[\![d]\!]\bigr) \times \mathbb{C}\bigl(H, T_2\mathcal{A}[\![d]\!]\bigr) \bigm| f : H \to T_1\mathcal{A}[\![d]\!]\bigr\}, T_1\mathcal{A}[\![d]\!], T_2\mathcal{A}[\![d]\!]\bigr).$$

We easily see that for any $\alpha \in \mathrm{Alg}(\mathcal{T}, \mathcal{A}[\![d]\!], \mathcal{A}[\![c]\!])$, we have

$$\bigl(\mathrm{Gef}^{\mathcal{T}_1}(\alpha), \gamma_{\mathcal{A}[\![d]\!]} \circ \mathrm{Gef}^{\mathcal{T}_1}(\alpha)\bigr) : \mathcal{V}[\![c]\!] \dot{\to} Cd.$$

We next show (below annotations of objects are omitted)

$$\bigl(ev^{\#_1} \circ (\theta_1), ev^{\#_2} \circ (\theta_2)\bigr) : \bigl(\mathcal{V}[\![d]\!] \dot{\Rightarrow} Cb\bigr) \dot{\times} Cd \dot{\to} Cb. \tag{2}$$

Here $(-)^{\#_i}$ is the Kleisli lifting of $\mathcal{T}_i$ $(i = 1, 2)$. This is sufficient to derive (C2), because

$$\begin{aligned}
&\bigl(ev^{\#_1} \circ \theta_1, ev^{\#_2} \circ \theta_2\bigr) : \bigl(\mathcal{V}[\![d]\!] \dot{\Rightarrow} Cb\bigr) \dot{\times} Cd \dot{\to} Cb \\
\Leftrightarrow\quad &\bigl(\sigma^{\mathcal{T}_1, Ab}, \sigma^{\mathcal{T}_2, Ab}\bigr) : Cd \dot{\to} \bigl(\bigl(\mathcal{V}[\![d]\!] \dot{\Rightarrow} Cb\bigr) \dot{\Rightarrow} Cb\bigr) \\
\Rightarrow\quad &\bigl(\sigma^{\mathcal{T}_1, Ab} \circ \mathrm{Gef}^{\mathcal{T}_1}(\alpha), \sigma^{\mathcal{T}_2, Ab} \circ \gamma \circ \mathrm{Gef}^{\mathcal{T}_1}(\alpha)\bigr) : \mathcal{V}[\![c]\!] \dot{\to} \bigl(\bigl(\mathcal{V}[\![d]\!] \dot{\Rightarrow} Cb\bigr) \dot{\Rightarrow} Cb\bigr) \\
\Leftrightarrow\quad &\bigl(\alpha_{Ab}, \mathrm{Alg}\bigl(\gamma, \mathcal{A}[\![d]\!], \mathcal{A}[\![c]\!]\bigr)(\alpha)_{Ab}\bigr) : \bigl(\mathcal{V}[\![d]\!] \dot{\Rightarrow} Cb\bigr) \dot{\to} \bigl(\mathcal{V}[\![c]\!] \dot{\Rightarrow} Cb\bigr).
\end{aligned}$$

By unfolding definitions, (2) becomes the following proposition: for any $H \in \mathbb{C}$, $f_1 : H \to (\mathcal{A}[\![d]\!] \Rightarrow T_1 Ab)$, $f_2 : H \to (\mathcal{A}[\![d]\!] \Rightarrow T_2 Ab)$ and $x : H \to T_1\mathcal{A}[\![d]\!]$, the equation

$$\gamma \circ ev^{\#_1} \circ \theta_1 \circ \langle f_1, x\rangle = ev^{\#_2} \circ \theta_2 \circ \langle f_2, \gamma \circ x\rangle \tag{3}$$

holds under the following assumption:

$$\forall H' \in \mathbb{C},\ h : H' \to H,\ \ y : H' \to \mathcal{A}[\![d]\!]\,.\,ev \circ \langle f_2 \circ h, y\rangle = \gamma \circ ev \circ \langle f_1 \circ h, y\rangle.$$

We thus prove this proposition. By instantiating the above assumption with $H' = H \times \mathcal{A}[\![d]\!]$, $h = \pi_1$, $y = \pi_2$, we obtain $f_2 = (\mathcal{A}[\![d]\!] \Rightarrow \gamma) \circ f_1$. Thus the right hand side of (3) is equal to $ev^{\#_2} \circ \theta_2 \circ ((\mathcal{A}[\![d]\!] \Rightarrow \gamma) \times \gamma) \circ \langle f_1, x\rangle$, which is equal to the left hand side of (3). Therefore (2) is proved. □

Table 1
Definition of two ν-calculus structures.

	$\mathcal{A}_1$	$\mathcal{A}_2$
$\mathbb{C}$	$[\mathbf{I}, \mathbf{Set}]$	$\mathbf{Set}$
T	$T_1 F = \mathrm{colim}_{Q \in \mathbf{I}} F(- + Q)$	$T_2 I = \mathbf{N} \Rightarrow I \times \mathbf{N}$
A	$A_1 \mathrm{nam} = N : \mathbf{I} \hookrightarrow \mathbf{Set}$	$A_2 \mathrm{nam} = \mathbf{N}$
a	$(a_1(\mathrm{eq}))_P(i, j) = \begin{cases} \iota_1(*) & (i = j) \\ \iota_2(*) & (i \neq j) \end{cases}$	$a_2(\mathrm{eq})(i, j) = \begin{cases} \iota_1(*) & (i = j) \\ \iota_2(*) & (i \neq j) \end{cases}$
	See (4) below	$a_2(\nu) = \lambda f x\ .\ f(x)(x+1)$

Example 4 *(Continued from Example 1).* There is a strong monad morphism γ from $\mathcal{T}_m$ to $\mathcal{T}_p$ mapping a list $l \in T_m I$ to the set $\gamma_I(l) \in T_p I$ of elements occurring in l. Furthermore $\mathcal{A}_1$ is the image of $\mathcal{A}_2$ along γ. Thus from Theorem 12, for any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$ and value $v_i \in Ab_i$, $\mathcal{A}_1[\![M]\!](\vec{v})$ is the set of elements occurring in $\mathcal{A}_2[\![M]\!](\vec{v})$.

Example 5. Let $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ be a $\lambda_c(\Sigma)$-structure, and $R \in \mathbb{C}$. We write $\mathcal{C}^{TR}$ for the *continuation monad*, whose functor part is given by $(- \Rightarrow TR) \Rightarrow TR$ [2]. The morphism $\sigma_I^{\mathcal{T},R} : TI \to ((I \Rightarrow TR) \Rightarrow TR)$ is actually a strong monad morphism from $\mathcal{T}$ to $\mathcal{C}^{TR}$. Therefore from Theorem 12, for any well-typed $\lambda_c(\Sigma)$-term $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$, we have

$$\sigma_{Ab}^{\mathcal{T},R} \circ \mathcal{A}[\![M]\!] = \sigma^{\mathcal{T},R} \mathcal{A}[\![M]\!].$$

The right hand side of this equation is the *CPS semantics* [3] of $\lambda_c(\Sigma)$, while the left hand side roughly corresponds to the mapping $\rho \mapsto \lambda k\ .\ k^{\#}(\mathcal{A}[\![M]\!]\rho)$. This equation is indeed the *monadic congruence result* [3] at base types. The above equation takes algebraic operations into account, and holds in any bi-CCC with a strong monad, including freely generated ones.

11. Comparing two monadic semantics of the ν-calculus

We fix a countably infinite set including the unique element $*$ of $1 \in \mathbf{Set}$, and call the elements in this infinite set *names*. We define $\mathbf{I}$ to be the category whose objects are finite subsets of names and whose morphisms are injections between them.

Dynamic name creation, such as the one in the π-calculus, is often categorically modelled in the presheaf category over $\mathbf{I}$ [32,33]. On the other hand, in practical implementations of programming languages, names are represented by natural numbers, and dynamic name creation is implemented by a hidden global counter that keeps track of the next fresh name. In this section, we consider Stark's ν-*calculus* [32] and discuss an effect simulation problem between presheaf semantics and global counter semantics of name creation.

The ν-calculus is specified by the λ_c-signature

$$\Sigma = \big(\{\mathrm{nam}\}, \quad K = \{\mathrm{eq} : \mathrm{nam} \times \mathrm{nam} \to 1 + 1\}, \quad O = \{\nu : \mathrm{nam} \to 1\}\big).$$

The base type nam is for *names*, the effect-free constant eq is for checking name equality, and the algebraic operation ν is to allocate a fresh name and bind it to x, like the one in π-calculus. The ν-calculus is then defined to be $\lambda_c(\Sigma)$. Below we call a $\lambda_c(\Sigma)$-structure a ν-*calculus structure*. We abbreviate $\nu(\lambda x : \mathrm{nam}\ .\ M)$ to $\nu(x.M)$.

In Table 1 we present two ν-calculus structures for which we consider an effect simulation problem. The ν-calculus structure $\mathcal{A}_1$ extracts the ingredients that are used in the categorical semantics of the ν-calculus in [32]. The monad $\mathcal{T}_1$ is Stark's *dynamic name creation monad*:

$$T_1 F P = \mathrm{colim}_{Q \in \mathbf{I}} F(P + Q) = \big\{(Q, x)\ \big|\ Q \in \mathbf{I},\ x \in F(P + Q)\big\}/\sim$$

where $\sim$ is the equivalence relation defined by: $(Q, x) \sim (R, y)$ if there are $S \in \mathbf{I}$ and two injections $l : Q \rightarrowtail S$, $m : R \rightarrowtail S$ such that $F(P + l)(x) = F(P + m)(y)$. The object for the name type is the inclusion functor $N : \mathbf{I} \hookrightarrow \mathbf{Set}$. This is the standard choice for representing names. We note that $T_1 N P \simeq P + 1$. Therefore we redefine T_1 to be the monad that is naturally isomorphic to the original T_1 and behaves as the functor $P \mapsto P + 1$ when applied to N.

We give exponentials in $[\mathbf{I}, \mathbf{Set}]$ in the standard way:

$$(F \Rightarrow G)P = [\mathbf{I}, \mathbf{Set}](\mathbf{y}P \times F, G).$$

The behaviour of the name equality predicate at a finite set P is given in Table 1; there, i and j are elements in P. The algebraic operation $a_1(\nu)$ for name creation is defined by (below $F \in [\mathbf{I}, \mathbf{Set}]$, $P \in \mathbf{I}$ and $\beta \in (N \Rightarrow T_1 F)P$):

$$\cfrac{\cfrac{(a_1(\nu))_F : (N \Rightarrow T_1 F) \to T_1 F}{(a_1(\nu)_F)_P : [\mathbf{I}, \mathbf{Set}](\mathbf{y}P \times N, T_1 F) \to T_1 F P}}{(a_1(\nu)_F)_P(\beta) = [1 + Q, F(\mathrm{assoc})(y)]_\sim : T_1 F P.} \tag{4}$$

In the right hand side of the defining equation of $a_1(\nu)$, $Q \in \mathbf{I}$ and $y \in F((P+1)+Q)$ are components of a pair (Q, y) in the equivalence class $\beta_{P+1}(\iota_1, \iota_2(*)) \in T_1F(P+1)$, and $\mathrm{assoc} : (P+1)+Q \to P+(1+Q)$ is the associativity morphism for the disjoint union.

The ν-calculus structure $\mathcal{A}_2$ is a semantic analogue of the dynamic name creation by a global state. We note that the interpretation $\mathcal{A}_2[\![-]\!]$ is not sound with respect to the ν-calculus axioms in [32].

We compare the denotation of a well-typed ν-calculus term $x_1 : \mathrm{nam}, \dots, x_n : \mathrm{nam} \vdash M : \mathrm{nam}$ in each ν-calculus structure. Suppose that p names have been allocated, and some of them are supplied to the free variables of M. Then M returns either one of the allocated names passed to it: $x_1, \dots, x_n$, or M allocates a new name and returns it. This behaviour is expressed differently in each ν-calculus structure:

- (in $\mathcal{A}_1$) Let P be the finite set consisting of p allocated names. We feed n names $i_1, \dots, i_n \in P$ to the free variables of M. When M returns a previously allocated name, the denotation $\mathcal{A}_1[\![M]\!]_P(\overrightarrow{i}) \in T_1NP = P+1$ is $\iota_1(i)$ with some $i \in P$. Otherwise, M returns a new name and the denotation is $\iota_2(*)$.
- (in $\mathcal{A}_2$) Natural numbers $0, \dots, p-1$ correspond to the allocated names. We thus feed $0 \leqslant i_1 \dots i_n \leqslant p-1$ to the free variables of M. The global counter pointing to the next fresh name is now p, so the name that M returns is given by $i = \pi_1(\mathcal{A}_2[\![M]\!](\overrightarrow{i})(p))$. When M returns an allocated name, $0 \leqslant i \leqslant p-1$; otherwise $i \geqslant p$. In fact, this behaviour of M remains the same even when the counter is increased from p. Therefore when M returns an allocated name i, for any $k \geqslant p$ we have $\pi_1(\mathcal{A}_2[\![M]\!](\overrightarrow{i})(k)) = i$; otherwise for any $k \geqslant p$ we have $\pi_1(\mathcal{A}_2[\![M]\!](\overrightarrow{i})(k)) \geqslant k$.

Based on this analysis, we establish a correspondence between the denotation of M in each ν-calculus structure. As names are represented differently in each structure, this relationship is parametrised by bijective correspondences between allocated names and natural numbers. Below, for a finite set P, we define $\#P$ to be its cardinality, and $\underline{P}$ to be $\{0, \dots, \#P-1\}$ when $P \neq \emptyset$; otherwise $\emptyset$. A *name enumeration* is a bijection $\sigma : P \to \underline{P}$.

Theorem 13. *Let $x_1 : \mathrm{nam}, \dots, x_n : \mathrm{nam} \vdash M : \mathrm{nam}$ be a ν-calculus term. For any finite set P, element $i_1, \dots, i_n \in P$ and a name enumeration $\sigma : P \to \underline{P}$, either*

- *there is some $j \in P$ such that $\mathcal{A}_1[\![M]\!]_P(\overrightarrow{i}) = \iota_1(j)$ and $\pi_1(\mathcal{A}_2[\![M]\!](\sigma(\overrightarrow{i}))(k)) = \sigma(j)$ for all $k \geqslant \#P$, or*
- *$\mathcal{A}_1[\![M]\!]_P(\overrightarrow{i}) = \iota_2(*)$ and $\pi_1(\mathcal{A}_2[\![M]\!](\sigma(\overrightarrow{i}))(k)) \geqslant k$ for all $k \geqslant \#P$.*

The rest of this section is the proof of this theorem. We construct a suitable fibration for logical relations over $[\mathbf{I}, \mathbf{Set}] \times \mathbf{Set}$, and give a simulation (V, C) between $\mathcal{A}_1$ and $\mathcal{A}_2$ that implies the goal of the theorem. We then check that it satisfies (I), (C1) and (C2).

We first define the category $\mathbf{E}$ of name enumerations by the following data: an object is a name enumeration, and a morphism $h : (\sigma : P \to \underline{P}) \to (\tau : Q \to \underline{Q})$ is a (necessarily unique) injection $h : P \to Q$ such that $\sigma = \tau \circ h$. We note that $\mathbf{E}$ is actually equivalent to $(\mathbf{N}, \leqslant)$. We write $\pi : \mathbf{E} \to \mathbf{I}$ for the functor defined by $\pi(\sigma : P \to \underline{P}) = P$ and $\pi(f) = f$.

We next pullback the subobject fibration $\mathbf{Sub}([\mathbf{E}, \mathbf{Set}]) \to [\mathbf{E}, \mathbf{Set}]$ along the following finite-product preserving functor $D : [\mathbf{I}, \mathbf{Set}] \times \mathbf{Set} \to [\mathbf{E}, \mathbf{Set}]$:

$$D(F, I) = (F \circ \pi) \times \Delta I;$$

here $\Delta : \mathbf{Set} \to [\mathbf{E}, \mathbf{Set}]$ is the diagonal functor.

$$\begin{tikzcd}
\mathbf{ERel} \arrow[r] \arrow[d, "q"'] & \mathbf{Sub}([\mathbf{E}, \mathbf{Set}]) \arrow[d] \\
{[\mathbf{I}, \mathbf{Set}] \times \mathbf{Set}} \arrow[r, "D"'] & {[\mathbf{E}, \mathbf{Set}]}
\end{tikzcd}$$

We obtain the fibration $q : \mathbf{ERel} \to [\mathbf{I}, \mathbf{Set}] \times \mathbf{Set}$ for logical relations by Proposition 6. The explicit definition of $\mathbf{ERel}$ is the following: an object is a triple (X, F, I) where $F \in [\mathbf{I}, \mathbf{Set}]$, $I \in \mathbf{Set}$ and X assigns a binary relation $X\sigma \subseteq FP \times I$ to each name enumeration $\sigma : P \to \underline{P}$. This assignment X satisfies: for any $h : \sigma \to \tau$ and $(x, y) \in X\sigma$, we have $(Fhx, y) \in X\tau$. A morphism from (X, F, I) to (Y, G, J) is a pair (ϕ, f) such that $\phi : F \to G$ is a morphism in $[\mathbf{E}, \mathbf{Set}]$, $f : I \to J$ is a function and for any name enumeration $\sigma : P \to \underline{P}$ and $(x, y) \in X\sigma$, we have $(\phi_P(x), f(y)) \in Y\sigma$.

We give a simulation (V, C) between $\mathcal{A}_1$ and $\mathcal{A}_2$ that entails Theorem 13. We define two binary relations $X\sigma$ and $Y\sigma$ for each name enumeration $\sigma : P \to \underline{P}$ by

$$\begin{aligned}
X\sigma &= \bigl\{\bigl(i, \sigma(i)\bigr) \in P \times \underline{P} \bigm| i \in P\bigr\}, \\
Y\sigma &= \bigl\{\bigl(\iota_1(i), f\bigr) \in T_1NP \times T_2\mathbf{N} \bigm| i \in P \wedge \forall k \geqslant \#P \,.\, \pi_1 \circ f(k) = \sigma(i)\bigr\} \\
&\quad \cup \bigl\{\bigl(\iota_2(*), f\bigr) \in T_1NP \times T_2\mathbf{N} \bigm| \forall k \geqslant \#P \,.\, \pi_1 \circ f(k) \geqslant k\bigr\}.
\end{aligned}$$

Proposition 14. *The pair $V, C : \{\mathrm{nam}\} \to \mathbf{ERel}$ of functors defined by*

$$V(\mathrm{nam}) = (X, N, \mathbf{N}), \qquad C(\mathrm{nam}) = (Y, T_1 N, T_2 \mathbf{N})$$

forms a simulation (V, C) between $\mathcal{A}_1$ and $\mathcal{A}_2$.

Proof. What we actually check here is that $V(\mathrm{nam})$ and $C(\mathrm{nam})$ belong to **ERel**. The case of $V(\mathrm{nam})$ is easy, so we move on to the case of $C(\mathrm{nam})$. Let $\sigma : P \to \underline{P}$, $\tau : Q \to \underline{Q}$ be name enumerations, $h : \sigma \to \tau$ be a morphism in **E** and $(x, f) \in Y\sigma$. We show $((h+1)(x), f) \in Y\tau$.

- Case $x = \iota_1(i)$ with some $i \in P$ such that $\pi_1 \circ f(k) = \sigma(i)$ for all $k \geqslant \#P$. Then for all $k \geqslant \#Q \geqslant \#P$, we have $\pi_1 \circ f(k) = \sigma(i) = \tau(h(i))$. Therefore $(\iota_1(h(i)), f) \in Y\tau$.
- Case $x = \iota_2(*)$ and $\forall k \geqslant \#P \,.\, \pi_1 \circ f(k) \geqslant k$. Then we have $((h+1)(\iota_2(*)), f) = (\iota_2(*), f)$, and this is included in $Y\tau$ as $\#Q \geqslant \#P$. □

We then prove the following proposition, from which we immediately obtain Theorem 13 by Theorem 7.

Proposition 15. *The simulation (V, C) defined in Proposition 14 satisfies* (I), (C1) *and* (C2).

Proof. We omit the proof of (V, C) satisfying (I) and (C1). Below we show that it satisfies (C2). As there is only one base type nam and one algebraic operation ν, it suffices to prove $(a_1(\nu)_N, a_2(\nu)_{\mathbf{N}}) : (V(\mathrm{nam}) \dot{\Rightarrow} C(\mathrm{nam})) \dot{\to} C(\mathrm{nam})$.

We first calculate $V(\mathrm{nam}) \dot{\Rightarrow} C(\mathrm{nam})$. It is a triple $(Z, N \Rightarrow T_1 N, \mathbf{N} \Rightarrow T_2 \mathbf{N})$ such that Z assigns the following binary relation to each name enumeration $\sigma : P \to \underline{P}$:

$$\begin{aligned} Z\sigma = \big\{ &(\beta, f) \in (N \Rightarrow T_1 N)P \times (\mathbf{N} \Rightarrow T_2 \mathbf{N}) \,\big|\, \\ &\forall \tau : Q \to \underline{Q},\ h : \sigma \to \tau,\ x \in Q \,.\, \big(\beta_Q\big(\pi(h), x\big), f\big(\tau(x)\big)\big) \in Y\tau \big\}. \end{aligned}$$

Lemma 16. *Let $h : (\sigma : P \to \underline{P}) \to (\tau : Q \to \underline{Q})$ be a morphism in* **E**, *$(\beta, f) \in Z\sigma$ and $x \in Q \setminus \mathrm{Im}(h)$. Then we have*

$$\big(\big([h, x] + 1\big)\big(\beta_{P+1}\big(\iota_1, \iota_2(*)\big)\big), f\big(\tau(x)\big)\big) \in Y\tau;$$

here $x \in Q$ is identified as a morphism $x : 1 \to Q$ in **I**.

Proof. From the definition of Z, we have $(\beta_Q(h, x), f(\tau(x))) \in Y\tau$. As $x \in Q \setminus \mathrm{Im}(h)$, $[h, x] : P + 1 \to Q$ is an injection, and from the naturality of β:

$$\begin{tikzcd} \mathbf{I}(P, P+1) \times (P+1) \arrow[r, "\beta_{P+1}"] \arrow[d, "([h,x]\circ -) \times [h,x]"'] & (P+1)+1 \arrow[d, "{[h,x]+1}"] \\ \mathbf{I}(P, Q) \times Q \arrow[r, "\beta_Q"'] & Q+1 \end{tikzcd}$$

we obtain $\beta_Q(h, x) = ([h, x] + 1)(\beta_{P+1}(\iota_1, \iota_2(*)))$. □

Let $\sigma : P \to \underline{P}$ and $(\beta, f) \in Z\sigma$. We show $((a_1(\nu)_N)_P(\beta), a_2(\nu)_{\mathbf{N}}(f)) \in Y\sigma$. The first component of this pair can be computed by the auxiliary function $\mu_P : (P+1)+1 \to P+1$ defined by

$$\mu_P(x) = \begin{cases} \iota_1(i) & (x = \iota_1(\iota_1(i)),\ i \in P), \\ \iota_2(*) & (\text{otherwise}). \end{cases}$$

Then $(a_1(\nu)_N)_P(\beta) = \mu_P(\beta_{P+1}(\iota_1, \iota_2(*)))$. Thus the goal is rewritten to

$$\big(\mu_P\big(\beta_{P+1}\big(\iota_1, \iota_2(*)\big)\big), \lambda k \,.\, f(k)(k+1)\big) \in Y\sigma, \tag{5}$$

and we prove it by the case analysis on $\beta_{P+1}(\iota_1, \iota_2(*))$.

- Either $\beta_{P+1}(\iota_1, \iota_2(*)) = \iota_2(*)$ or $\beta_{P+1}(\iota_1, \iota_2(*)) = \iota_1(\iota_2(*))$. In these cases, the value of $\mu_P(\beta_{P+1}(\iota_1, \iota_2(*)))$ is $\iota_2(*)$. The goal (5) now becomes:

  $$\forall k \geqslant \#P \,.\, \pi_1\big(f(k)(k+1)\big) \geqslant k.$$

  Let $k \geqslant \#P$. We take a finite set Q' containing $k - \#P + 1$ elements, and let Q be $P + Q'$, $\tau : Q \to \underline{Q}$ be an extension of σ to Q, and $x \in Q'$ be an element such that $\tau(x) = k = \#Q - 1$. The left injection $\iota_1 : P \to Q$ is a morphism from σ to τ in **E**, and $x \notin \mathrm{Im}(\iota_1)$.

We instantiate Lemma 16 with $\iota_1 : \sigma \to \tau$, (β, f) and x, and obtain $(\iota_2(*), f(k)) \in Y\tau$. There are then two sub-cases:

– Sub-case $\beta_{P+1}(\iota_1, \iota_2(*)) = \iota_2(*)$. From the definition of $Y\tau$, we obtain: for any $k' \geqslant \#Q$, $\pi_1(f(k)(k')) \geqslant k'$. We instantiate k' with $\#Q = k+1$, and obtain $\pi_1(f(k)(k+1)) \geqslant k+1 \geqslant k$.

– Sub-case $\beta_{P+1}(\iota_1, \iota_2(*)) = \iota_1(\iota_2(*))$. From the definition of $Y\tau$, we obtain: for any $k' \geqslant \#Q$, $\pi_1(f(k)(k')) = k$. By letting $k' = k+1$, we obtain $\pi_1(f(k)(k+1)) = k$.

- $\beta_{P+1}(\iota_1, \iota_2(*)) = \iota_1(\iota_1(i))$ for some $i \in P$. The value of $\mu_P(\beta_{P+1}(\iota_1, \iota_2(*)))$ is $\iota_1(i)$. The goal (5) now becomes

$$\forall k \geqslant \#P \,.\, \pi_1\big(f(k)(k+1)\big) = \sigma(i).$$

We take Q', Q, τ, x as before; so we have $Q = P + Q'$, $\#Q = k+1$, $\iota_1 : \sigma \to \tau$ and $x \notin \mathrm{Im}(\iota_1)$.
We instantiate Lemma 16 with $\iota_1 : \sigma \to \tau$, (β, f) and x, and obtain $(\iota_1(i), f(k)) \in Y\tau$, that is, for any $k' \geqslant \#Q$, $\pi_1(f(k)(k')) = \tau(\iota_1(i)) = \sigma(i)$. By letting $k' = k+1 = \#Q$, we obtain $\pi_1(f(k)(k+1)) = \sigma(i)$.

The advantage of the proof method by the $\top\top$-lifting is that, although the actual contents of $\dot{T}X$ is unclear, it gives a well-defined logical relation for the monad $\mathcal{T}$ that is sufficient to establish a relationship between two monadic semantics of a λ_c-calculus. The $\top\top$-lifting also reduces the problem to its essence: the closure property of the simulation under the algebraic operations in the calculus. Solving the effect simulation problem without the $\top\top$-lifting would require hand-crafting a suitable logical relation, but doing so would quickly become difficult, as the concept of the predicate is involved and the simulation relation we would like to establish is non-trivial.

12. Extending $\lambda_c(\Sigma)$ with recursive functions

We next add the recursive function term $\mu f x.M$ to $\lambda_c(\Sigma)$. This term creates a closure that may recursively call itself inside M; its typing rule is

$$\frac{\Gamma, f : \rho \to \sigma, x : \rho \vdash M : \sigma}{\Gamma \vdash \mu f x.M : \rho \to \sigma} \quad .$$

We call the extended calculus $\lambda_c^{\mathbf{fix}}(\Sigma)$.

We consider the effect property problem for the class of λ_c-structures where the underlying category is enriched over $\omega\mathbf{CPO}$, the category of ω-complete partial orders (which may not have least elements) and continuous functions between them.

An *ω***CPO***-enriched bi-CCC* is a bi-CCC $\mathbb{C}$ such that each homset is an $\omega\mathbf{CPO}$, and the composition, tupling $\langle -,- \rangle$, cotupling $[-,-]$ and currying $\lambda(-)$ of the bi-CC structure on $\mathbb{C}$ are all monotone and continuous. We write $\mathbb{C}_0$ for the underlying ordinary bi-CCC of $\mathbb{C}$ (that is, we forget the $\omega\mathbf{CPO}$-structure on each homset).

Let $\mathbb{C}$ be an $\omega\mathbf{CPO}$-enriched bi-CCC. A *pseudo-lifting strong monad* over $\mathbb{C}$ is an ordinary strong monad $\mathcal{T}$ over $\mathbb{C}_0$ such that it has a $(0,1)$-ary algebraic operation bt, and its component $\mathsf{bt}_I \in \mathbb{C}(0 \Rightarrow TI, 1 \Rightarrow TI) \simeq \mathbb{C}(1, TI)$ at object $I \in \mathbb{C}$ is the least morphism. We write $\bot_I : 1 \to TI$ for the morphism corresponding to bt_I via the above bijection. We note that $\mathrm{Gef}^{\mathcal{T}}(\mathsf{bt}) = \bot_0$ and $\bot_I = T?_I \circ \bot_0$.

Definition 6. An $\omega\mathbf{CPO}$-enriched $\lambda_c(\Sigma)$-structure is a tuple $(\mathbb{C}, \mathcal{T}, A, a)$ such that $\mathbb{C}$ is an $\omega\mathbf{CPO}$-enriched bi-CCC, $\mathcal{T}$ is a pseudo-lifting monad over $\mathbb{C}$ and $(\mathbb{C}_0, \mathcal{T}, A, a)$ is a $\lambda_c(\Sigma)$-structure.

Let $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ be an $\omega\mathbf{CPO}$-enriched $\lambda_c(\Sigma)$-structure. We define the fixpoint operator $\mathrm{fix}_I : \mathbb{C}(TI, TI) \to \mathbb{C}(1, TI)$ restricted to TI by

$$\mathrm{fix}_I(f) = \bigsqcup_{i=0}^{\infty} f^{(i)} \circ \bot_I. \tag{6}$$

This satisfies the axioms of the *uniform T-fixpoint operator* by Simpson and Plotkin [31]. As $\mathbb{C}$ is a bi-CCC, we can parametrise it as $\mathrm{fix}_I^J : \mathbb{C}(J \times TI, TI) \to \mathbb{C}(J, TI)$; see [10] for the details.

Hasegawa and Kakutani axiomatise the behaviour of the recursive function term $\mu f x \,.\, M$ in call-by-value functional languages as a *stable uniform call-by-value fixpoint operator* [10]. They bijectively correspond to uniform T-fixpoint operators. Therefore fix gives adequate semantics to the recursive function term:

$$\mathcal{A}[\![\Gamma \vdash \mu f x.M : \rho \Rightarrow \sigma]\!] = \eta \circ \mathrm{al} \circ \mathrm{fix}_{\mathcal{A}[\![\rho \Rightarrow \sigma]\!]}^{\mathcal{A}[\![\Gamma]\!]}\big(\mathcal{A}\,[\![\lambda x \,.\, M]\!] \circ \mathrm{id} \times \mathrm{al}\big). \tag{7}$$

The right hand side is the expansion of the stable uniform call-by-value fixpoint operator derived from the uniform T-fixpoint operator (6). The morphism $\mathrm{al}_{I,J} : T(I \Rightarrow TJ) \to (I \Rightarrow TJ)$ is the canonical T-algebra structure over $I \Rightarrow TJ$, and it is defined by

$$\mathrm{al}_{I,J} = \lambda\big(ev^{\#} \circ \theta'_{I \Rightarrow TJ, I}\big);$$

here $\theta'_{I,J} : TI \times J \to T(I \times J)$ is the symmetric version of the strength $\theta_{I,J}$.

Next, let $p : \mathbb{E} \to \mathbb{C}_0$ be a fibration for logical relations. We formulate the concept of admissible predicates in the fibrational setting.

Definition 7. We call $X \in \mathbb{E}$ above $TI \in \mathbb{C}_0$ *admissible* if 1) $\bot_{pX} : \dot{1} \dot{\to} X$ and 2) for all $Y \in \mathbb{E}$ and ω-chain $f_i \in \mathbb{C}(pY, pX)$ such that $f_i : Y \dot{\to} X$, we have $\bigsqcup_{i=0}^{\infty} f_i : Y \dot{\to} X$.

Theorem 17. *Let Σ be a signature, $\mathcal{A} = (\mathbb{C}, \mathcal{T}, A, a)$ be an ω**CPO**-enriched $\lambda_c(\Sigma)$-structure, $p : \mathbb{E} \to \mathbb{C}_0$ be a fibration for logical relations and (V, C) be a property over $\mathcal{A}$. If the property satisfies the conditions* (I), (C1) *and* (C2) *in Theorem 7 and*

(A) *for all $b \in B$, Cb is admissible,*

then the answer of the effect property problem for $\lambda_c^{\mathbf{fix}}(\Sigma)$ is yes: for all well-typed $\lambda_c^{\mathbf{fix}}(\Sigma)$-terms $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$, we have $\mathcal{A}[\![M]\!] : \dot{\prod}_{i=1}^{n} Vb_i \dot{\to} Cb$.

Proof. From the same argument as the proof of Theorem 7, we obtain a $\lambda_c(\Sigma)$-structure $\mathcal{V} = (\mathbb{E}, \dot{\mathcal{T}}, V, \dot{a})$. To prove the basic lemma, it suffices to show that for any $X \in \mathbb{E}$, $f : \dot{T}X \dot{\to} \dot{T}X$ implies $\mathrm{fix} f : \dot{1} \dot{\to} \dot{T}X$. Let f be such a morphism. Then

$$\begin{aligned}
\text{(A)} \quad &\Rightarrow \quad \forall b \in B \,.\, \bot_{Ab} : \dot{1} \dot{\to} Cb \\
&\Leftrightarrow \quad \forall X \in \mathbb{E} \,.\, \bot_{pX} : \dot{1} \dot{\to} \dot{T}X \\
&\Rightarrow \quad \forall X \in \mathbb{E},\ i \in \mathbf{N} \,.\, f^{(i)} \circ \bot_{pX} : \dot{1} \dot{\to} \dot{T}X \\
&\Leftrightarrow \quad \forall b \in B,\ X \in \mathbb{E},\ i \in \mathbf{N} \,.\, \sigma_{pX}^{\mathcal{T},Ab} \circ f^{(i)} \circ \bot_{pX} : \dot{1} \dot{\to} \big((X \dot{\Rightarrow} Cb) \dot{\Rightarrow} Cb\big) \\
&\Leftrightarrow \quad \forall b \in B,\ X \in \mathbb{E},\ i \in \mathbf{N} \,.\, \mathrm{nm}^{-1}\big(\sigma_{pX}^{\mathcal{T},Ab} \circ f^{(i)} \circ \bot_{pX}\big) : (X \dot{\Rightarrow} Cb) \dot{\to} Cb.
\end{aligned}$$

Since Cb is admissible, we have

$$\bigsqcup_{i=0}^{\infty} \mathrm{nm}^{-1}\big(\sigma_{pX}^{\mathcal{T},Ab} \circ f^{(i)} \circ \bot_{pX}\big) = \mathrm{nm}^{-1}\big(\sigma_{pX}^{\mathcal{T},Ab} \circ \mathrm{fix} f\big) : (X \dot{\Rightarrow} Cb) \dot{\to} Cb.$$

From this, we obtain $\mathrm{fix} f : \dot{1} \dot{\to} \dot{T}X$. □

Lemma 18. *Let $\mathbb{C}$ be an ω**CPO**-enriched bi-CCC, $\mathcal{T}_1$ be a pseudo-lifting strong monad over $\mathbb{C}$, $\mathcal{T}_2$ be a strong monad over $\mathbb{C}_0$ and $\gamma : \mathcal{T}_1 \to \mathcal{T}_2$ be a strong monad morphism. Then $\mathcal{T}_2$ is also a pseudo-lifting strong monad over $\mathbb{C}$.*

Proof. Let $\mathsf{bt} \in \mathrm{Alg}(\mathcal{T}_1, 0, 1)$ be the algebraic operation that exists from the definition of the pseudo-lifting strong monad, and $(\bot_1)_I : 1 \to T_1 I$ be the least morphism corresponding to the component bt_I. We show that each component of $\mathsf{bt}' = \mathrm{Alg}(\gamma, 0, 1)(\mathsf{bt})$ is the least morphism. Let us write $(\bot_2)_I : 1 \to T_2 I$ for the morphism corresponding to the component bt'_I. Then we have

$$(\bot_2)_I = T_2 ?_I \circ \mathrm{Gef}^{\mathcal{T}_2}\big(\mathsf{bt}'\big) = T_2 ?_I \circ \gamma_0 \circ (\bot_2)_0 = \gamma_I \circ (\bot_1)_I.$$

We show that $(\bot_2)_I$ is the least morphism. Let $x : 1 \to T_2 I$ be any morphism. Then

$$\begin{aligned}
x = (\mu_2)_I \circ \gamma_{T_2 I} \circ (\eta_1)_{T_2 I} \circ x &\geqslant (\mu_2)_I \circ \gamma_{T_2 I} \circ (\bot_1)_{T_2 I} \\
&= (\mu_2)_I \circ \gamma_{T_2 I} \circ T_1\big((\eta_2)_I\big) \circ (\bot_1)_I = (\bot_2)_I. \qquad \square
\end{aligned}$$

Theorem 19. *Let $\mathcal{A} = (\mathbb{C}, \mathcal{T}_1, A, a)$ be an ω**CPO**-enriched $\lambda_c(\Sigma)$-structure, $\mathcal{T}_2$ be a strong monad over $\mathbb{C}$ and $\gamma : \mathcal{T}_1 \to \mathcal{T}_2$ be a strong monad morphism. Then for any well-typed $\lambda_c^{\mathbf{fix}}(\Sigma)$-term $x_1 : b_1, \ldots, x_n : b_n \vdash M : b$, we have $\gamma_{Ab} \circ \mathcal{A}[\![M]\!] = (\gamma\mathcal{A})[\![M]\!]$.*

Proof. The proof is the same as that of Theorem 12. To apply Theorem 17, it suffices to check that the property C satisfies 1) for any $b \in B$, $((\bot_1)_{Ab}, (\bot_2)_{Ab}) : \dot{1} \dot{\to} Cb$ and 2) for any $b \in B$, $(X, I, J) \in \mathbb{K}$ and ω-chains of morphisms $f_i : I \to T_1 Ab$ and $g_i : J \to T_2 Ab$ in $\mathbb{C}$ such that $(f_i, g_i) : (X, I, J) \dot{\to} Cb$, we have $(\bigsqcup_{i=0}^{\infty} f_i, \bigsqcup_{i=0}^{\infty} g_i) : (X, I, J) \dot{\to} Cb$. 1) is immediate. 2) Let $H \in \mathbb{C}$ and $(a, b) \in XH$. First, we have $g_i \circ b = \gamma_{Ab} \circ f_i \circ a$ for any natural number i. From the continuity of the composition of $\mathbb{C}$, we obtain

$$\left(\bigsqcup_{i=0}^{\infty} g_i\right) \circ b = \bigsqcup_{i=0}^{\infty} (g_i \circ b) = \bigsqcup_{i=0}^{\infty} (\gamma_{Ab} \circ f_i \circ a) = \gamma_{Ab} \circ \left(\bigsqcup_{i=0}^{\infty} f_i\right) \circ a.$$

From this, we obtain $(\bigsqcup_{i=0}^{\infty} f_i, \bigsqcup_{i=0}^{\infty} g_i) : (X, I, J) \dot{\to} Cb$. □

Table 2
Effect property problems.

Signature	Algebraic Operations	Property ($b \in B$)
Σ_1	$O = \emptyset$	$C_1 b = (\{\{x\} \mid x \in Ab\}, T_p Ab)$
Σ_2	$O = \{\mathrm{null} : \bar{0} \to 1\}$	$C_2 b = (\{\{x\} \mid x \in Ab\} \cup \{\emptyset\}, T_p Ab)$
Σ_3	$O = \{\mathrm{or} : \bar{2} \to 1\}$	$C_3 b = (\{v \mid v \subseteq Ab,\ v \neq \emptyset\}, T_p Ab)$

13. Related work

Filinski is one of the pioneers in logical relations for monads [4], and developed various techniques to establish relationships between semantics of higher-order languages with effects [3–7]. In [7, Proposition 3.7] three methods to construct logical relations for monads are mentioned: 1) When T is a monad constructed from the standard type constructors, such as state monad and continuation monad, then define a logical relation $\dot{T}$ for T in the same way as T is constructed. 2) For a logical relation $\dot{T}$ for a monad T and a strong monad morphism $\sigma : S \to T$, the inverse image $\sigma^* \dot{T}$ is a logical relation for S. 3) For a family of logical relations $\dot{T}_i$ for a monad T, the intersection $\bigwedge_i \dot{T}_i$ is again a logical relation for T.

The categorical $\top\top$-lifting is technically a particular combination of the methods 1–3 above (in fibrational category theory). However, what is new about $\top\top$-lifting is that we *use* the simulation / property we would like to establish on computational effects to define the logical relation $\dot{T}$. This idea is a secret recipe in the proofs of various results by the precursors of categorical $\top\top$-lifting, such as biorthogonality [9,21,26], $\top\top$-closure [27] and leapfrog method [18,19]; see also [15].

Under the presence of parametric polymorphism, Møgelberg and Simpson give a uniform method to lift the computation type constructor ! to binary relations [24]. Although this lifting is computed inside a logic for parametric polymorphism, it seems to share the basic idea with $\top\top$-lifting. In a recent work [1], Atkey et al. study the induction principle for the inductive types involving computational effects. They consider a lifting of monads to the total category of a general type-theoretic fibration using the direct image functors and comprehension. This technique is based on the method to lift endofunctors across fibrations [8].

Larrecq et al. propose a method to lift a monad on $\mathbb{C}$ to the category of subscones over $\mathbb{C}$ using mono-factorisation systems [17]. They also discuss a relational lifting of monads, and demonstrate that various notions of logical relations for monads are derivable by their method. We look at how their method lifts a monad $\mathcal{T}$ on **Set** to the one on **Pred**, the trivial subscone over **Set**. Their method yields the monad $\tilde{\mathcal{T}}$ on **Pred** whose functor part is given by $\tilde{T}(X, I) = (\mathrm{Im}(T\iota), TI)$; here $\iota : X \hookrightarrow I$ is the inclusion function and $\mathrm{Im}(T\iota)$ is the image of $T\iota : TX \to TI$. For instance, the lifting $\tilde{\mathcal{T}_p}$ of the powerset monad $\mathcal{T}_p$ is given by $\tilde{T}_p(X, I) = (T_p X, T_p I)$.

However, simply lifting the monad by their method is insufficient to solve general effect property problems. Consider three λ_c-signatures $\Sigma_1, \Sigma_2, \Sigma_3$ whose algebraic operations are declared in the middle of Table 2. We restrict the $\lambda_c(\Sigma)$-structure $\mathcal{A}_1$ from Example 1 to each Σ_i, and call the resulting $\lambda_c(\Sigma_i)$-structure $\mathcal{B}_i$. We also define a property (V_i, C_i) over $\mathcal{B}_i$ by $V_i b = Ab$ $(b \in B)$ and the defining equations on the right of Table 2. We note that the answer of the effect property problem with respect to each (V_i, C_i) is yes. Let us see what happens if we use $\tilde{\mathcal{T}_p}$ in the proof of Theorem 7 instead of the $\top\top$-lifting of $\mathcal{T}_p$. As $\tilde{\mathcal{T}_p}$ admits the lifting of *both* algebraic operations $a_1(\mathrm{null})$ and $a_1(\mathrm{or})$, we can successfully construct $\dot{a}$ in the proof of the theorem. Unfortunately, we fail at the very last step: $\tilde{T}_p V b = T_p Ab \not\subseteq C_i b$.

In the case of the lifting of monads along $\pi : \mathbf{Pred} \to \mathbf{Set}$, their method yields the monad lifting that admits the lifting of *all* $(\coprod_{i=1}^{n} 1, 1)$-algebraic operation as $(\dot{\coprod}_{i=1}^{n} \dot{1}, \dot{1})$-algebraic operation. On the other hand, the property we would like to establish on the computational effects caused by $\lambda_c(\Sigma)$-programs may not always be closed under every such algebraic operation, as we have seen in the above example. It remains to be seen whether lifting by factorisation systems can be used differently to solve the effect problem in general.

The advantage of logical relations for monads by $\top\top$-lifting is that it does not limit the form of simulation / property we would like to establish on computational effects. Furthermore, Theorem 11 gives a good characterisation of when algebraic operations are related by the logical relations given by $\top\top$-lifting. On the other hand, it is rather difficult to check whether non-algebraic operations that manipulate computational effects, such as Felleisen's $\mathcal{C}$-operator, are related by the logical relations given by $\top\top$-lifting; this shall be discussed in a separate paper. Extending our results with recursive types and handlers for algebraic effects [30] is also a future work.

Acknowledgments

The author is grateful to the anonymous reviewers for their detailed comments and constructive criticism that have contributed to this paper. The proof of Theorem 11 has improved significantly thanks to the reviewers' suggestions. He also thanks to Masahito Hasegawa, Susumu Nishimura and Naohiko Hoshino for fruitful discussions. This research was partly supported by Grant-in-Aid for Young Scientists (B) 20700012.

References

[1] R. Atkey, N. Ghani, B. Jacobs, P. Johann, Fibrational induction meets effects, in: L. Birkedal (Ed.), Foundations of Software Science and Computational Structures, in: Lecture Notes in Computer Science, vol. 7213, Springer, 2012, pp. 42–57.
[2] N. Benton, J. Hughes, E. Moggi, Monads and effects, in: Proc. APPSEM 2000, in: Lecture Notes in Computer Science, vol. 2395, Springer, 2002, pp. 42–122.
[3] A. Filinski, Representing monads, in: Proc. POPL 1994, pp. 446–457.
[4] A. Filinski, Controlling effects, PhD thesis, Carnegie Mellon University, 1996.
[5] A. Filinski, Representing layered monads, in: Proc. POPL 1999, pp. 175–188.
[6] A. Filinski, On the relations between monadic semantics, Theoretical Computer Science 375 (2007) 41–75.
[7] A. Filinski, K. Støvring, Inductive reasoning about effectful data types, in: R. Hinze, N. Ramsey (Eds.), ICFP, ACM, 2007, pp. 97–110.
[8] N. Ghani, P. Johann, C. Fumex, Fibrational induction rules for initial algebras, in: A. Dawar, H. Veith (Eds.), CSL, in: Lecture Notes in Computer Science, vol. 6247, Springer, 2010, pp. 336–350.
[9] J.Y. Girard, Linear logic, Theoretical Computer Science 50 (1987) 1–102.
[10] M. Hasegawa, Y. Kakutani, Axioms for recursion in call-by-value, Higher-Order and Symbolic Computation 15 (2002) 235–264.
[11] C. Hermida, Fibrations, logical predicates and indeterminants, PhD thesis, University of Edinburgh, 1993.
[12] B. Jacobs, Categorical Logic and Type Theory, Elsevier, 1999.
[13] A. Jung, J. Tiuryn, A new characterization of lambda definability, in: Proc. TLCA, in: Lecture Notes in Computer Science, vol. 664, Springer, 1993, pp. 245–257.
[14] S. Katsumata, A semantic formulation of $\top\top$-lifting and logical predicates for computational metalanguage, in: Proc. CSL 2005, in: Lecture Notes in Computer Science, vol. 3634, Springer, 2005, pp. 87–102.
[15] S. Katsumata, A characterisation of lambda definability with sums via $\top\top$-closure operators, in: Proc. CSL 2008, in: Lecture Notes in Computer Science, vol. 5213, Springer, 2008, pp. 278–292.
[16] S. Katsumata, Relating computational effects by $\top\top$-lifting, in: L. Aceto, M. Henzinger, J. Sgall (Eds.), ICALP (2), in: Lecture Notes in Computer Science, vol. 6756, Springer, 2011, pp. 174–185.
[17] J.G. Larrecq, S. Lasota, D. Nowak, Logical relations for monadic types, Mathematical Structures in Computer Science 18 (2008) 1169–1217.
[18] S. Lindley, Normalisation by evaluation in the compilation of typed functional programming languages, PhD thesis, University of Edinburgh, 2004.
[19] S. Lindley, I. Stark, Reducibility and $\top\top$-lifting for computation types, in: Proc. TLCA, pp. 262–277.
[20] Q. Ma, J. Reynolds, Types, abstractions, and parametric polymorphism, part 2, in: Proc. MFPS 1991, in: Lecture Notes in Computer Science, vol. 598, Springer, 1992, pp. 1–40.
[21] P.A. Melliès, J. Vouillon, Recursive polymorphic types and parametricity in an operational framework, in: LICS, IEEE Computer Society, 2005, pp. 82–91.
[22] J. Mitchell, Foundations for Programming Languages, MIT Press, 1996.
[23] J. Mitchell, A. Scedrov, Notes on sconing and relators, in: CSL, in: Lecture Notes in Computer Science, vol. 702, Springer, 1992, pp. 352–378.
[24] R.E. Møgelberg, A. Simpson, Relational parametricity for computational effects, Logical Methods in Computer Science 5 (2009).
[25] E. Moggi, Computational lambda-calculus and monads, in: LICS, IEEE Computer Society, 1989, pp. 14–23.
[26] M. Parigot, Proofs of strong normalisation for second order classical natural deduction, Journal of Symbolic Logic 62 (1997) 1461–1479.
[27] A. Pitts, Parametric polymorphism and operational equivalence, Mathematical Structures in Computer Science 10 (2000) 321–359.
[28] G. Plotkin, J. Power, Semantics for algebraic operations, Electronic Notes in Theoretical Computer Science 45 (2001).
[29] G. Plotkin, J. Power, Algebraic operations and generic effects, Applied Categorical Structures 11 (2003) 69–94.
[30] G. Plotkin, M. Pretnar, Handlers of algebraic effects, in: G. Castagna (Ed.), ESOP, in: Lecture Notes in Computer Science, vol. 5502, Springer, 2009, pp. 80–94.
[31] A. Simpson, G. Plotkin, Complete axioms for categorical fixed-point operators, in: LICS, pp. 30–41.
[32] I. Stark, Categorical models for local names, Lisp and Symbolic Computation 9 (1996) 77–107.
[33] I. Stark, A fully abstract domain model for the π-calculus, in: Proc. LICS 1996, IEEE, 1996, pp. 36–42.