Available online at www.sciencedirect.com

ScienceDirect

ELSEVIER

Electronic Notes in Theoretical Computer Science 298 (2013) 383–402

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

A Stream Calculus of Bottomed Sequences for Real Number Computation

Kei Terayama[1] Hideki Tsuiki[2]

Graduate School of Human and Environmental Studies
Kyoto University
Kyoto, Japan

Abstract

A calculus XPCF of $1\bot$-sequences, which are infinite sequences of $\{0,1,\bot\}$ with at most one copy of bottom, is proposed and investigated. It has applications in real number computation in that the unit interval $\mathbb{I}$ is topologically embedded in the set $\Sigma^\omega_{\bot,1}$ of $1\bot$-sequences and a real function on $\mathbb{I}$ can be written as a program which inputs and outputs $1\bot$-sequences. In XPCF, one defines a function on $\Sigma^\omega_{\bot,1}$ only by specifying its behaviors for the cases that the first digit is 0 and 1. Then, its value for a sequence starting with a bottom is calculated by taking the meet of the values for the sequences obtained by filling the bottom with 0 and 1. The validity of the reduction rule of this calculus is justified by the adequacy theorem to a domain-theoretic semantics. Some example programs including addition and multiplication are shown. Expressive powers of XPCF and related languages are also investigated.

Keywords: Bottom, stream, real number computation, domain model, PCF, adequacy, parallel or

1 Introduction

Streams are a useful data structure used for expressing infinite sequences and one can implement real number computation with streams through signed digit expansion[1,2] or other expansions of real numbers[6]. However, since a stream can only be accessed one-way from left to right, if there is a bottom, i.e., a term whose evaluation does not terminate, in a stream, then a program get stuck when it tries to read in the value of the bottom cell and cannot input the rest of the sequence though it may contain valuable data.

Usually, a bottom is considered as a kind of programming error which should be avoided in a correct program. However, it is known that infinite sequences which may contain bottoms are useful in representing continuous topological spaces like

1 Email: terayama@i.h.kyoto-u.ac.jp

2 Email: tsuiki@i.h.kyoto-u.ac.jp

http://dx.doi.org/10.1016/j.entcs.2013.09.023

$\mathbb{R}$. Here, we call an infinite sequence of $\Sigma \cup \{\perp\}$ which may contain at most one copy of bottom a $1\perp$-sequence. It is shown in [8] and [15] that $\mathbb{R}$ and $\mathbb{I} = [0,1]$ can be topologically embedded in the space $\Sigma^{\omega}_{\perp,1}$ of $1\perp$-sequences of Σ for $\Sigma = \{0,1\}$ and this embedding is called the Gray embedding in [15]. The signed-digit expansion and other admissible representations of $\mathbb{R}$ turn out to be redundant in the sense that infinitely many reals each satisfy the property of being represented by infinitely many codes[4,17]. On the other hand, with the Gray embedding, a unique code can be assigned to each real number by extending the code space with at most one copy of $\perp$. This embedding result is extended in [16] to other topological spaces and it is shown that any n-dimensional separable metric space can be topologically embedded in the space $\Sigma^{\omega}_{\perp,n}$ of $n\perp$-sequences.

[8] expressed a $1\perp$-sequence as a function from $\mathsf{N}_\perp$ to $\{-1,1,\perp\}$ and used the parallel if operator pif to access $1\perp$-sequences and showed that real number algorithms can be expressed in PCF + pif. In order to evaluate $\mathsf{pif}\ L\ M\ N$, one need to evaluate L, M, and N in parallel. Therefore, pif operator causes explosion of parallel computations and it seems difficult to implement it efficiently. Martin Escardó proposed Real PCF[6] which is an extension of PCF with real numbers. It is based on interval domains and a kind of parallel conditional operator is used.

On the other hand, [15] restricted the number of $\perp$ to one and introduced an IM2-machine (Indeterministic Multiheads Type2 Machine) which enables extended stream access to $1\perp$-sequences. However, the behavior of an IM2-machine needs to be specified through a set of overlapping rules and therefore functions expressible with IM2-machines are multi-valued functions in general. Moreover, a program of an IM2-machine is complicated because one needs to express its behaviors for inputs from extra heads.

In this paper, we introduce a calculus XPCF of $1\perp$-sequences with which one can express extended stream accesses to them. It is an extension of PCF with a datatype S of $1\perp$-sequences and is based on the algebraic domain **BD** of $1\perp$-sequences[16]. The datatype S has, in addition to the constructors $\mathsf{0} : \mathsf{S} \to \mathsf{S}$ and $\mathsf{1} : \mathsf{S} \to \mathsf{S}$ to prepend a digit to a sequence, constructors $\bar{\mathsf{0}} : \mathsf{S} \to \mathsf{S}$ and $\bar{\mathsf{1}} : \mathsf{S} \to \mathsf{S}$ to insert a digit as the second element of a sequence. However, a function on S is defined by expressing its behaviors only for cases the argument has the form $\mathsf{0}N$ and $\mathsf{1}N$ with the expression $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1 \rangle$. It means a function on $\Sigma^{\omega}_{\perp,1}$ to apply $[\![\lambda x.M_0]\!]$ to s if the argument is $0s$, to apply $[\![\lambda x.M_1]\!]$ to s if the argument is $1s$, and apply both of them to s and take the meet of the results if the argument is $\perp s$. XPCF can be considered as an algebraic domain variant of Real PCF. This calculus has the computational adequacy property with respect to its domain-theoretic model.

We give some example programs of XPCF including addition and multiplication on $\mathbb{I}$ through the Gray embedding. We also studied the expressive power of this language and showed that XPCF has the same expressive power as PCF + pif on types which do not contain S, that all computable elements of **BD** are expressible on type S, and that if we extend XPCF with the $\exists$ operator, then all the computable elements in the semantic domains are expressible.

As [7] showed, any real number calculus which is adequate to the interval do-

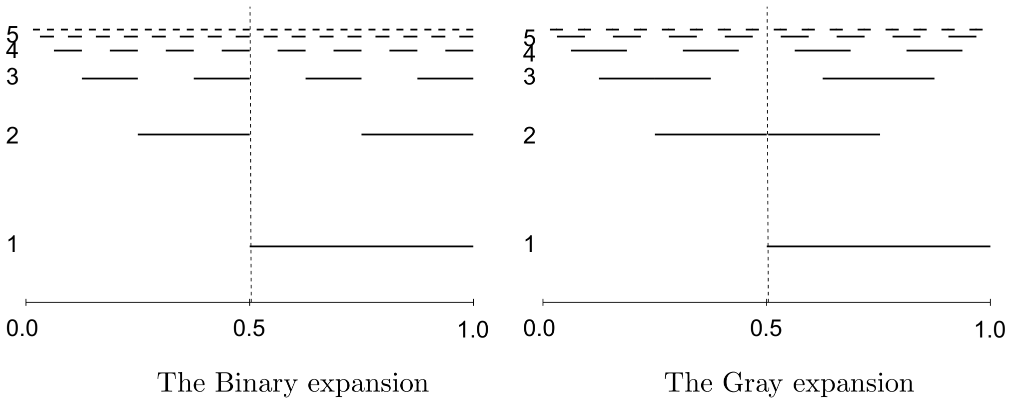

Fig. 1. The Binary and Gray expansion of $\mathbb{I}$

main model and in which the average function can be represented, does not have a sequential reduction strategy. Their proof also applies to our model with some modifications and thus any sequential reduction strategy of this calculus is not adequate. We designed a sequential reduction strategy of XPCF and implemented it with Haskell. Though it is not adequate and some of the terms cannot be reduced to their denotations, it sequentially evaluates many of the terms like addition and multiplication.

In the next section, we start with explaining the Gray embedding of $\mathbb{I}$ in $\Sigma^{\omega}_{\perp,1}$ and the domain **BD** of $1\perp$-sequences. In Section 3, we define the syntax and semantics of XPCF and, in Section 4, we show how real functions can be expressed in XPCF with some program examples. Then, we give reduction rules of XPCF in Section 5 and show the adequacy property in Section 6. In section 7, we study expressive powers of XPCF.

Notations: Recall that we fix $\Sigma = \{0, 1\}$. We denote by Γ^* the set of finite sequences of a character set Γ and by Γ^ω the set of infinite sequences of Γ. We define $\Gamma^\infty = \Gamma^* \cup \Gamma^\omega$, which is a Scott domain, i.e., a bounded complete ω-algebraic dcpo. Let $\Sigma_\perp = \Sigma \cup \{\perp\}$, and $\Sigma^\omega_\perp$ be the set of infinite sequences of $\Sigma_\perp$. $\Sigma_\perp$ has the order generated by $\perp \sqsubseteq 0$ and $\perp \sqsubseteq 1$. On $\Sigma^\omega_\perp$, we define the order $\sqsubseteq$ as $s \sqsubseteq t$ if $s(n) \sqsubseteq t(n)$ for every n. $(\Sigma^\omega_\perp, \sqsubseteq)$ is a Scott domain. We define $\Sigma^\omega_{\perp,1} = \{s \in \Sigma^\omega_\perp \mid s \text{ contains at most one } \perp\}$.

2 Real number computation and $1\perp$-sequences

2.1 Gray embedding

The Gray expansion is an expansion of $\mathbb{I}$ as infinite sequences of Σ which is different from the ordinary binary expansion [15]. It is based on Gray code[10], which is a coding of natural numbers with Σ different from the binary code. Figure 1 shows the binary and Gray expansion of $\mathbb{I}$. In the binary expansion of x, the head h of the expansion indicates whether x is in $[0,\ 1/2]$ or in $[1/2,\ 1]$ and the tail is the expansion of $f(x, h)$ for f the function defined as

$$f(x,h) = \begin{cases} 2x & (\text{if } h = 0) \\ 2x - 1 & (\text{if } h = 1) \end{cases}.$$

Thus, with the binary expansion, the tail of the expansion of $1/2$ depends on the choice of the head character h and $1/2$ has two expansions 1000... and 0111.... On the other hand, the head of the Gray expansion is the same as that of the binary expansion, whereas the tail is the expansion of $t(x)$ for t the so-called tent function:

$$t(x) = \begin{cases} 2x & (0 \leq x \leq 1/2) \\ 2(1 - x) & (1/2 < x \leq 1) \end{cases}.$$

Note that $t(x)$ is continuous on $x = 1/2$ and therefore the tail of the expansion does not depend on the choice of the first digit. Actually, the two expansions of $1/2$ are 01000... and 11000... which coincide from the second character. It means that the value is half not depending on the first character. Therefore, we leave the first character undefined ($\perp$) and define a new expansion of $1/2$ as $\perp 1000\ldots$. It is also the case for expansions of dyadic numbers (rational numbers of the form $m/2^k$) and therefore we assign codes of the form $p\perp 1000\ldots$ for $p \in \{0,1\}^*$ to those numbers. In this way, we have a mapping $\varphi : \mathbb{I} \to \Sigma^{\omega}_{\perp,1}$ called the Gray-embedding as follows.

Definition 2.1 ([15]) Let $P : \mathbb{I} \to \Sigma_{\perp}$ be the map

$$P(x) = \begin{cases} 0 & (x < 1/2) \\ \perp & (x = 1/2) \\ 1 & (x > 1/2) \end{cases}.$$

the Gray embedding φ is a function from $\mathbb{I}$ to $\Sigma^{\omega}_{\perp,1}$ defined as $\varphi(x)(n) = P(t^n(x))$ $(n = 0, 1, \ldots)$.

An embedding of $\mathbb{R}$ in $\{-1,1\}^{\omega}_{\perp,1}$ is defined in [8] independently by Gianantonio, and the Gray embedding is essentially the same as the restriction of his embedding to $\mathbb{I}$. We call the $1\perp$-sequence $\varphi(x)$ the modified Gray expansion of x. The Gray embedding φ is actually a topological embedding with the topology of $\Sigma^{\omega}_{\perp,1}$ the subspace topology of the Scott topology of $\Sigma^{\omega}_{\perp}$.

2.2 Domains of $1\perp$-sequences

We explain the domain **BD** of $1\perp$-sequences [16]. Let $\Sigma^*_{\perp,1}$ be the set of finite $1\perp$-sequences of Σ. Here, $p \in {\Sigma_{\perp}}^*$ is a finite $1\perp$-sequence of Σ if $\perp$ appears at most once in p and $\perp$ is not the final character of p. We have $\Sigma^*_{\perp,1} = \{\epsilon, 0, 1, \perp 0, \perp 1, ...\}$ with ϵ the empty sequence. We can regard $\Sigma^*_{\perp,1}$ as a subset of $\Sigma^{\omega}_{\perp}$ by identifying $p \in \Sigma^*_{\perp,1}$ with $p\perp^{\omega} \in \Sigma^{\omega}_{\perp}$. We define $\mathbf{BD} = \Sigma^*_{\perp,1} \cup \Sigma^{\omega}_{\perp,1}$, which is a Scott subdomain of $\Sigma^{\omega}_{\perp}$ with the least element $\perp_{\mathbf{BD}} = \epsilon$ as Figure 2 shows. For $c \in \Sigma$, we also denote

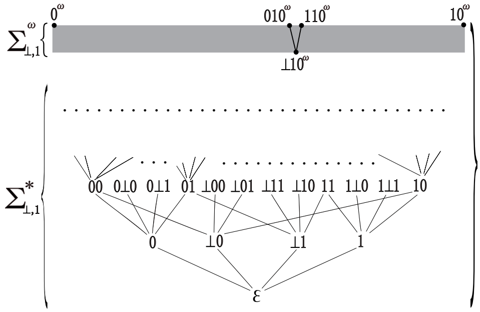

Fig. 2. The domain **BD**

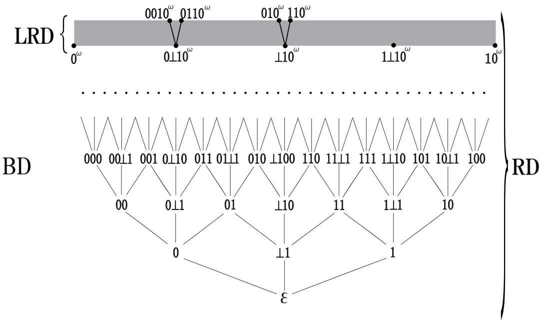

Fig. 3. The domain **RD**

by c the continuous function from **BD** to **BD** to prepend c and denote by $\overline{c}$ the continuous function from **BD** to **BD** to insert c as the second character, where $\overline{c}(\epsilon)$ is defined as $\perp c$. We have the equation $\overline{b} \circ c = c \circ b$ for $b, c \in \Sigma$.

We regard that each finite sequence $s = d_0 d_1 \ldots d_{n-1}$ of $\{0, 1, \overline{0}, \overline{1}\}$ represents the element $d_0(d_1(\ldots(d_{n-1}(\epsilon))))$ of $\Sigma^*_{\perp,1}$ and each infinite sequence $s = d_0 d_1 \ldots$ of $\{0, 1, \overline{0}, \overline{1}\}$ represents the limit of the infinite increasing sequence $(s_n)_{n=0,1,\ldots}$ in **BD** for $s_n = d_0(d_1(\ldots(d_{n-1}(\epsilon))))$. Note that this limit exists in $\Sigma^\omega_{\perp,1}$. In particular, the sequence $b_0 b_1 \ldots b_{m-1} \overline{c_0} \overline{c_1} \ldots \overline{c_{n-1}}$ represents $b_0 b_1 \ldots b_{m-1} \perp c_0 c_1 \ldots c_{n-1} \in \Sigma^*_{\perp,1}$ (or $b_0 b_1 \ldots b_{m-1}$ if $n = 0$), and the infinite sequence $b_0 b_1 \ldots b_{m-1} \overline{c_0} \overline{c_1} \ldots$ represents $b_0 b_1 \ldots b_{m-1} \perp c_0 c_1 \ldots \in \Sigma^\omega_{\perp,1}$.

Since **BD** is a Scott domain, the meet (i.e., the greatest lower bound) exists for any subset of **BD**. We show it explicitly, because it plays an important role in the semantics of XPCF. First, the meet on $\Sigma_\perp = \{0, 1, \perp\}$ is obviously defined. It is naturally extended to the meet $s \sqcap_{\Sigma^\omega_\perp} t$ in $\Sigma^\omega_\perp$ as $(s \sqcap_{\Sigma^\omega_\perp} t)(n) = s(n) \sqcap t(n)$. Let trunc be the function from $\Sigma^\omega_\perp$ to **BD** to truncate the sequence after the second $\perp$ to form a finite $1\perp$-sequence if it contains more than one copies of $\perp$, and returns itself if it does not.

Proposition 2.2 *The meet $s \sqcap t$ of $s, t \in$ **BD** is equal to* $\mathrm{trunc}(s \sqcap_{\Sigma^\omega_\perp} t)$.

We define the subdomain **RD** of **BD** which is used for expressing $\mathbb{I}$ through the Gray representation. We define

$$\mathbf{RD} = \{p\perp 10^n \ : \ p \in \Sigma^*, n \in \{0, 1, \ldots, \omega\}\} \cup \Sigma^\infty.$$

It is a Scott domain. Let **LRD** be the subset $\{p\perp 10^\omega : p \in \Sigma^*\} \cup \Sigma^\omega$ of $\Sigma^\omega_{\perp,1}$. **LRD** is the set of limit (i.e., non-compact) elements of **RD** as Figure 3 shows. **LRD** consists of $\varphi(\mathbb{I})$ and those sequences obtained by filling a bottom of $s \in \varphi(\mathbb{I})$ with 0 and 1. One can see that $\mathbb{I}$ is a retract of **LRD** and $\mathbb{I}$ is homeomorphic to the set of minimal elements of **LRD** with the retract map $\delta : \mathbf{LRD} \to \mathbb{I}$ defined as $\delta(s) = x$ if $\varphi(x) \sqsubseteq s$. One can see that the triple $(\mathbf{RD}, \mathbf{LRD}, \delta)$ is a retract domain representation of $\mathbb{I}$ in the sense of [3] and we call the map $\delta : \mathbf{LRD} \to \mathbb{I}$ the Gray representation.

We can consider two codings of $\mathbb{I}$ based on the Gray embedding. The first one is obtained by identifying x with $\varphi(x)$ through the embedding and the other one is the Gray representation δ. For example, for $1/2 \in \mathbb{I}$, $\perp 10^\omega$ is the unique codes with φ. On the other hand, there are three codes $\perp 10^\omega, 010^\omega$ and 110^ω for $1/2$ with respect to δ. Based on these codings, we have two notions that a function on **BD** realize a function on $\mathbb{I}$.

Definition 2.3 Let F be a function from $\mathbf{BD}^n$ to **BD** and f be a (partial) real function from $\mathbb{I}^n$ to $\mathbb{I}$.

(1) F *exactly realizes* f if, for every $(x_1, \ldots, x_n) \in \mathrm{dom}(f)$,

$$F(\varphi(x_1), \ldots, \varphi(x_n)) = \varphi(f(x_1, \ldots, x_n)).$$

(2) F *realizes* f if, for every $(p_1, \ldots, p_n) \in (\delta^n)^{-1}(\mathrm{dom}(f))$,

$$\delta(F(p_1, \ldots, p_n)) = f(\delta(p_1), \ldots, \delta(p_n)).$$

3 Syntax and denotational semantics of XPCF

Throughout this paper, we write types and constants of syntactic entities in Sans-serif font and program names and the names of semantic domains in **Bold** font.

3.1 PCF

We review the syntax, the semantics, and the reduction rules of the language PCF in Table 1. See [11] or some textbooks like [14] for the details of PCF. PCF has ground types B for boolean values and N for integers. For a term M, $FV(M)$ denotes the free variables of M and M is *closed* if $FV(M)$ is empty. A program is a closed term of a ground type. An *environment* ρ is a type-respecting map from the set of variables to $\bigcup\{D_\sigma | \sigma\,\text{type}\}$ and, for $a \in D_\sigma$, $\rho[a/x^\sigma]$ is the environment which maps x^σ to a and any other variable y^σ to $\rho(y^\sigma)$. If M is a closed term, then $[\![M]\!](\rho)$ does not depend on ρ and we write $[\![M]\!]$ for $[\![M]\!](\rho)$.

The operational semantics of PCF is given by the *immediate reduction relation* in Table 1. The result of evaluation of a program M is a constant c defined as

$$\mathrm{Eval}_{\mathrm{PCF}}(M) = c \;\text{ iff }\; M \rhd^* c.$$

The following theorem is often referred to as the *Adequacy Property* of PCF. It asserts that the operational and denotational semantics coincide.

Theorem 3.1 ([11, Theorem 3.1]) *For any PCF program M and constant c,*

$$\mathrm{Eval}_{\mathrm{PCF}}(M) = c \;\text{ iff }\; [\![M]\!] = [\![c]\!].$$

3.2 Syntax and semantics of XPCF

The syntax and denotational semantics of XPCF is listed in Table 2. We list only the differences compared with the PCF specification. It has a ground type S such

Syntax of PCF

Types: $\sigma ::= \mathsf{B} \,|\, \mathsf{N} \,|\, \sigma \to \sigma$

Variables(of type σ): $x^\sigma ::= x^\sigma, y^\sigma, z^\sigma, \ldots$

Constants: $c ::= \mathsf{tt}, \mathsf{ff}, \mathsf{if}_\sigma, \mathsf{Y}_\tau, \mathsf{k}_n, \mathsf{inc}, \mathsf{dec}, \mathsf{zero}$ (σ:ground type, τ:type, $n \in \mathbb{N}$)

Terms: $M ::= x^\sigma \,|\, c \,|\, (MM) \,|\, (\lambda x^\sigma.M)$

Typing Rules:

$$x^\sigma : \sigma \qquad \mathsf{tt} : \mathsf{B} \qquad \mathsf{ff} : \mathsf{B} \qquad \mathsf{k}_n : \mathsf{N} \qquad \mathsf{inc} : \mathsf{N} \to \mathsf{N}$$

$$\mathsf{dec} : \mathsf{N} \to \mathsf{N} \qquad \mathsf{zero} : \mathsf{N} \to \mathsf{B} \qquad \mathsf{if}_\sigma : \mathsf{B} \to \sigma \to \sigma \to \sigma$$

$$\mathsf{Y}_\sigma : (\sigma \to \sigma) \to \sigma \qquad \frac{M : \tau}{\lambda x^\sigma.M : \sigma \to \tau} \qquad \frac{M : \sigma \to \tau \quad N : \sigma}{MN : \tau}$$

Denotational semantics of PCF

Domains: D_B :the flat domain $\{\perp_\mathsf{B}, tt, ff\}$ of truth values.

D_N :the flat domain $\{\perp_\mathsf{N}, 0, 1, \ldots\}$ of natural numbers.

$D_{\sigma\to\tau}$:the domain $[D_\sigma \to D_\tau]$ of continuous functions from D_σ to D_τ, with the least element denoted by $\perp_{\sigma\to\tau}$.

Interpretation of constants:

$$[\![\mathsf{tt}]\!] = tt \qquad [\![\mathsf{ff}]\!] = ff \qquad [\![\mathsf{k}_n]\!] = n$$

$$[\![\mathsf{inc}]\!] = \lambda n \in D_\mathsf{N}. \begin{cases} n+1 & (n \neq \perp_\mathsf{N}) \\ \perp_\mathsf{N} & (n = \perp_\mathsf{N}) \end{cases} \qquad [\![\mathsf{dec}]\!] = \lambda n \in D_\mathsf{N}. \begin{cases} n-1 & (n \geq 1) \\ \perp_\mathsf{N} & (n \in \{\perp_\mathsf{N}, 0\}) \end{cases}$$

$$[\![\mathsf{zero}]\!] = \lambda n \in D_\mathsf{N}. \begin{cases} tt & (n = 0) \\ ff & (n > 0) \\ \perp_\mathsf{B} & (n = \perp_\mathsf{N}) \end{cases} \qquad [\![\mathsf{Y}_\sigma]\!] = \lambda F \in D_{\sigma\to\sigma}. \bigsqcup_{n\in\mathsf{N}} F^n(\perp_\sigma)$$

$$[\![\mathsf{if}_\sigma]\!] = \lambda b \in D_\mathsf{B}.\lambda x \in D_\sigma.\lambda y \in D_\sigma. \begin{cases} x & (b = tt) \\ y & (b = ff) \\ \perp_\sigma & (b = \perp_\mathsf{B}) \end{cases}$$

Denotational semantics:

(i) $[\![x^\sigma]\!](\rho) = \rho(x^\sigma)$ (ii) $[\![c]\!](\rho) = [\![c]\!]$

(iii) $[\![MN]\!](\rho) = [\![M]\!](\rho)([\![N]\!](\rho))$ (iv) $[\![\lambda x^\sigma.M]\!](\rho) = \lambda a \in D_\sigma.[\![M]\!](\rho[a/x^\sigma])$

Operational semantics of PCF

Reduction rules:

$$(\lambda x^\sigma.M)N \rhd M[N/x^\sigma] \qquad \mathsf{Y}_\sigma\, M \rhd M(\mathsf{Y}_\sigma\, M) \qquad \mathsf{inc}\,\mathsf{k}_n \rhd \mathsf{k}_{n+1} \qquad \mathsf{dec}\,\mathsf{k}_{n+1} \rhd \mathsf{k}_n$$

$$\mathsf{zero}\,\mathsf{k}_0 \rhd \mathsf{tt} \qquad \mathsf{zero}\,\mathsf{k}_{n+1} \rhd \mathsf{ff} \qquad \mathsf{if}_\sigma\,\mathsf{tt}\,M\,N \rhd M \qquad \mathsf{if}_\sigma\,\mathsf{ff}\,M\,N \rhd N$$

$$\frac{M \rhd M'}{M\,N \rhd M'\,N} \qquad \frac{N \rhd N'}{M\,N \rhd M\,N'} \text{ (if } M \text{ is } \mathsf{if}_\sigma, \mathsf{inc}, \mathsf{dec} \text{ or } \mathsf{zero})$$

Table 1
Syntax and semantics of PCF

Syntax of XPCF Syntax of PCF extended with the followings.

Types: S

Constants: $\mathtt{0}, \mathtt{1}, \overline{\mathtt{0}}, \overline{\mathtt{1}}$

Terms: $\langle \mathtt{0}x^{\mathsf{S}} \to M; \mathtt{1}x^{\mathsf{S}} \to M \rangle \mid \langle\!\langle \mathtt{0}x^{\mathsf{S}} \to M; \mathtt{1}x^{\mathsf{S}} \to M \rangle\!\rangle$

Typing Rules:

$$\mathtt{0} : \mathsf{S} \to \mathsf{S} \qquad \mathtt{1} : \mathsf{S} \to \mathsf{S} \qquad \overline{\mathtt{0}} : \mathsf{S} \to \mathsf{S} \qquad \mathtt{1} : \mathsf{S} \to \mathsf{S}$$

$$\frac{M_0 : \sigma \quad M_1 : \sigma}{\langle \mathtt{0}x^{\mathsf{S}} \to M_0; \mathtt{1}x^{\mathsf{S}} \to M_1 \rangle : \mathsf{S} \to \sigma} \qquad \frac{M_0 : \sigma \quad M_1 : \sigma}{\langle\!\langle \mathtt{0}x^{\mathsf{S}} \to M_0; \mathtt{1}x^{\mathsf{S}} \to M_1 \rangle\!\rangle : \mathsf{S} \to \sigma}$$

Denotational semantics of XPCF Semantics of PCF extended with the followings.

Domains: $D_{\mathsf{S}} = \mathbf{BD}$

Interpretation of constants:

$[\![\mathtt{0}]\!] = 0 \in D_{\mathsf{S}\to\mathsf{S}}$ (where $0(s) = 0s$ for $s \in D_{\mathsf{S}}$)

$[\![\mathtt{1}]\!] = 1 \in D_{\mathsf{S}\to\mathsf{S}}$ (where $1(s) = 1s$ for $s \in D_{\mathsf{S}}$)

$[\![\overline{\mathtt{0}}]\!] = \overline{0} \in D_{\mathsf{S}\to\mathsf{S}}$ (where $\overline{0}(as) = a0s$ for $a \in \Sigma_\perp$ and $s \in D_{\mathsf{S}}$, $\overline{0}(\epsilon) = \perp 0$)

$[\![\overline{\mathtt{1}}]\!] = \overline{1} \in D_{\mathsf{S}\to\mathsf{S}}$ (where $\overline{1}(as) = a1s$ for $a \in \Sigma_\perp$ and $s \in D_{\mathsf{S}}$, $\overline{1}(\epsilon) = \perp 1$)

Denotational semantics:

(v) $[\![\langle \mathtt{0}x^{\mathsf{S}} \to M_0; \mathtt{1}x^{\mathsf{S}} \to M_1 \rangle]\!](\rho) =$

$$\lambda s \in D_{\mathsf{S}}. \begin{cases} \perp_\sigma & (\text{if } s = \epsilon) \\ [\![M_0]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = 0s') \\ [\![M_1]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = 1s') \\ [\![M_0]\!](\rho[s'/x^{\mathsf{S}}]) \sqcap [\![M_1]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = \perp s') \end{cases}$$

(vi) $[\![\langle\!\langle \mathtt{0}x^{\mathsf{S}} \to M_0; \mathtt{1}x^{\mathsf{S}} \to M_1 \rangle\!\rangle]\!](\rho) =$

$$\lambda s \in D_{\mathsf{S}}. \begin{cases} [\![M_0]\!](\rho[\epsilon/x^{\mathsf{S}}]) \sqcap [\![M_1]\!](\rho[\epsilon/x^{\mathsf{S}}]) & (\text{if } s = \epsilon) \\ [\![M_0]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = 0s') \\ [\![M_1]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = 1s') \\ [\![M_0]\!](\rho[s'/x^{\mathsf{S}}]) \sqcap [\![M_1]\!](\rho[s'/x^{\mathsf{S}}]) & (\text{if } s = \perp s') \end{cases}$$

Table 2
Syntax and denotational semantics of XPCF

that $D_{\mathsf{S}} = \mathbf{BD}$ with constants $\mathtt{0}, \mathtt{1}, \overline{\mathtt{0}}, \overline{\mathtt{1}}$ of type $\mathsf{S} \to \mathsf{S}$ which denote the functions $0, 1, \overline{0}, \overline{1}$, respectively. For a variable of type S, we omit the type and write x for x^{S}, for simplicity. We have function terms $\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle$ and $\langle\!\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle\!\rangle$ of type $\mathsf{S} \to \sigma$ for M_0 and M_1 terms of type σ. The variable x is a bound variable of $\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle$ and $\langle\!\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle\!\rangle$.

We call $\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle$ and $\langle\!\langle \mathtt{0}x \to M_0; \mathtt{1}x \to M_1 \rangle\!\rangle$ *extended conditional terms*. For the two functions $f_0 = [\![\lambda x^{\mathsf{S}}.M_0]\!]$ and $f_1 = [\![\lambda x^{\mathsf{S}}.M_1]\!]$ from D_{S} to D_σ, the function $f = [\![\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle]\!] : D_{\mathsf{S}} \to D_\sigma$ returns $f_0(s)$ if the argument is $0s$ and $f_1(s)$ if the argument is $1s$. For the case the argument starts with $\perp$, we define $f(\perp s) = f_0(s) \sqcap f_1(s)$, which is the meet of $f_0(s)$ and $f_1(s)$ in D_σ. Here, meets on D_{N} and D_{B} are obviously defined, meets on D_{S} are explained in Section 2.2, and the meet of two functions $g, h \in D_{\sigma\to\tau}$ is the pointwise meet function $(g \sqcap h)(x) = g(x) \sqcap h(x)$. We define $f(\epsilon) = \perp_\sigma$. Thus, f is a strict function. It means that we adopt call by value semantics to an application of $\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle$.

The meaning of the term $\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle$ is different from that of $\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle$ only for the case of ϵ, and $[\![\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle]\!]$ is not a strict function. Note that if we identify ϵ and $\perp^\omega$ and match $\perp s'$ with $\perp^\omega$ for $s' = \epsilon$, then the last case of the semantics of $\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle$ subsumes the first case. Note that both functions $[\![\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle]\!]$ and $[\![\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle]\!]$ are continuous. Our intention in introducing two kinds of extended conditional terms is that $\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle$ is used in writing a program and $\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle$ is used only in reduction steps, which we explain in Section 5. We call a closed ground type term a program if it does not contain extended conditional terms of the form $\langle\!\langle 0x^{\mathsf{S}} \to M_0; 1x^{\mathsf{S}} \to M_1 \rangle\!\rangle$ as subterms.

4 Program examples of XPCF

The function **nh** to invert the first digit is written as

$$\mathbf{nh} = \langle 0x \to 1x; 1x \to 0x \rangle.$$

Note that $[\![\mathbf{nh}]\!](\perp s) = 0s \sqcap 1s = \perp s$ for $s \in \Sigma^\omega$.

The function **ns** to invert the second digit is written as

$$\mathbf{ns} = \langle 0x \to 0(\mathbf{nh}\ x); 1x \to 1(\mathbf{nh}\ x) \rangle.$$

The following terms **head** : $\mathsf{S} \to \mathsf{B}$ and **tail** : $\mathsf{S} \to \mathsf{S}$ are the head and the tail function on D_{S}.

$$\mathbf{head} = \langle 0x \to \mathsf{ff}; 1x \to \mathsf{tt} \rangle,$$
$$\mathbf{tail} = \langle 0x \to x; 1x \to x \rangle.$$

Here, we identify $0, 1, \perp \in \Sigma_\perp$ with *ff*, *tt*, $\perp_{\mathsf{B}} \in D_{\mathsf{B}}$, respectively. Note that there is no cons function: $\mathsf{B} \to \mathsf{S} \to \mathsf{S}$ because if we prepend $\perp$ to a $1\perp$-sequence, then the result may not be a $1\perp$-sequence. The function **inv** to invert all the digits is written as

$$\mathbf{inv} = \mathsf{Y}_{\mathsf{S}\to\mathsf{S}}(\lambda f^{\mathsf{S}\to\mathsf{S}}.\langle 0x \to 1(f\ x); 1x \to 0(f\ x) \rangle).$$

For simplicity, we use the recursive definition notation to abbreviate a term defined with the Y operator. For example, **inv** is written as

$$\mathbf{inv} = \langle 0x \to 1(\mathbf{inv}\ x); 1x \to 0(\mathbf{inv}\ x) \rangle.$$

We show how real numbers and real functions are expressed in XPCF. Since $\varphi(0) = 0^\omega$, $\varphi(1) = 10^\omega$ and $\varphi(1/2) = \perp 10^\omega$, we can express these numbers as

$$\mathbf{0} = \mathsf{Y}_{\mathsf{S}}\ 0,$$

$\mathbf{1} = 1(\mathsf{Y_S}\ 0)$,
$\mathbf{1/2} = \bar{1}(\mathsf{Y_S}\ \bar{0})$.

In Section 2.1, we defined notions that a function on **BD** (exactly) realizes a function on $\mathbb{I}$. We say that a closed XPCF term (exactly) realizes a real function if it denotes a function which (exactly) realizes the function. The program

$$\mathbf{div2} = \lambda x^{\mathsf{S}}.0x.$$

realizes the function $\mathrm{div2}(x) = x/2$ but does not exactly realize it because $[\![\mathbf{div2}]\!](10^\omega) = 010^\omega$ whereas $\varphi(1/2) = \perp 10^\omega$. There is also a program which exactly realizes div2, which is given later. Since the complement function $\mathrm{comp}(x) = 1 - x$ is realized by the function to invert the first digit, comp is exactly realized by the program **nh**. The tent function t is exactly realized by **tail**.

Programs which realize addition (average) **av**, subtraction **sub**, multiplication **mult** and a program **div2b** which exactly realizes div2 can be written as follows.

$$\begin{aligned}\mathbf{av} = \langle & 0x \to \langle 0y \to 0(\mathbf{av}\,x\,y); 1y \to \bar{1}(\mathbf{ns}(\mathbf{av}\,x\,(\mathbf{nh}\,y)))\rangle;\\ & 1x \to \langle 0y \to \bar{1}(\mathbf{ns}(\mathbf{av}\,(\mathbf{nh}\,x)\,y)); 1y \to 1(\mathbf{av}\,x\,y)\rangle\rangle\end{aligned}$$

$$\begin{aligned}\mathbf{sub} = \langle & 0x \to \langle 0y \to 0(\mathbf{sub}\,x\,y); 1y \to \mathsf{Y_S}0\rangle;\\ & 1x \to \langle 0y \to \mathbf{nh}(\mathbf{av}\,x\,y); 1y \to 0(\mathbf{sub}\,y\,x)\rangle\rangle\end{aligned}$$

$$\begin{aligned}\mathbf{mult} = \langle & 0x \to \langle 0y \to 0(0(\mathbf{mult}\,x\,y)); 1y \to 0(\mathbf{mult}\,x\,(1y))\rangle;\\ & 1x \to \langle 0y \to 0(\mathbf{mult}\,(1x)\,y);\\ & \qquad 1y \to \mathbf{av}\,(\mathbf{nh}(\mathbf{av}\,x\,y))\,(1(\mathbf{nh}(\mathbf{mult}\,(\mathbf{nh}\,x)\,(\mathbf{nh}\,y))))\rangle\rangle\end{aligned}$$

$$\mathbf{div2b} = \langle 0x \to 0(0x); 1x \to \bar{1}(\mathbf{f}x)\rangle$$
$$\mathbf{f} = \langle 0x \to \bar{0}(\mathbf{f}x); 1x \to 0(1x)\rangle$$

Here, $[\![\mathbf{f}]\!]$ is a function which satisfies $[\![\mathbf{f}]\!](0^\omega) = \perp 0^\omega$ and $[\![\mathbf{f}]\!](x) = 0x$ if x contains the character 1.

5 Operational semantics of XPCF

5.1 Operational semantics of XPCF

Table 3 shows the reduction rule of XPCF. For $\mathsf{d} \in \{0, 1, \bar{0}, \bar{1}\}$, we say that a term M of type S outputs d if M is reduced to $\mathsf{d}M'$.

We explain how the reduction of a term $\langle 0x \to M_0; 1x \to M_1\rangle\ N$ proceeds. The first lines of rules (COND 0), (COND 1), (COND $\bar{0}$), and (COND $\bar{1}$) are for the reduction of an application term $\langle 0x \to M_0; 1x \to M_1\rangle\ N$. Note that a closed term N is reduced by (APP-R) to one of these four forms if $[\![N]\!]$ is not $\perp$ by the adequacy theorem in the next section. If N has the form $\bar{0}N'$, (COND $\bar{0}$) is applied and then we have a term $M_0[0x/x]$ and $M_1[1x/x]$. After that, M_0 and M_1 are evaluated by (LEFT) and (RIGHT) only with the additional information that the first character of x is 0. Note that if M_0 contains x, then $M_0[0x/x]$ also contains x and therefore it is expected that this evaluation terminates when it requires the

Reduction rule of XPCF

In addition to the reduction rule of PCF, we have the following rules.

(COND 0) $\langle 0x \to M_0; 1x \to M_1\rangle(0N) \rhd M_0[N/x]$
$\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle(0N) \rhd M_0[N/x]$

(COND 1) $\langle 0x \to M_0; 1x \to M_1\rangle(1N) \rhd M_1[N/x]$
$\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle(1N) \rhd M_1[N/x]$

(COND $\overline{0}$) $\langle 0x \to M_0; 1x \to M_1\rangle(\overline{0}N) \rhd \langle\!\langle 0x \to M_0[0x/x]; 1x \to M_1[0x/x]\rangle\!\rangle N$
$\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle(\overline{0}N) \rhd \langle\!\langle 0x \to M_0[0x/x]; 1x \to M_1[0x/x]\rangle\!\rangle N$

(COND $\overline{1}$) $\langle 0x \to M_0; 1x \to M_1\rangle(\overline{1}N) \rhd \langle\!\langle 0x \to M_0[1x/x]; 1x \to M_1[1x/x]\rangle\!\rangle N$
$\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle(\overline{1}N) \rhd \langle\!\langle 0x \to M_0[1x/x]; 1x \to M_1[1x/x]\rangle\!\rangle N$

(OUT 1) $\langle\!\langle 0x \to \mathsf{d}M_0; 1x \to \mathsf{d}M_1\rangle\!\rangle N \rhd \mathsf{d}(\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle N)$ $(\mathsf{d} \in \{0, 1, \overline{0}, \overline{1}\})$

(OUT 2) $\langle\!\langle 0x \to \mathsf{b}(\mathsf{c}M_0); 1x \to \mathsf{b}'(\mathsf{c}M_1)\rangle\!\rangle N \rhd \overline{\mathsf{c}}(\langle\!\langle 0x \to \mathsf{b}M_0; 1x \to \mathsf{b}'M_1\rangle\!\rangle N)$
$(\mathsf{b}, \mathsf{b}' \in \{0, 1\}$ and $\mathsf{b} \neq \mathsf{b}')$

(OUT 3) $\langle\!\langle 0x \to \overline{\mathsf{b}}M_0; 1x \to \mathsf{c}(\mathsf{b}M_1)\rangle\!\rangle N \rhd \overline{\mathsf{b}}(\langle\!\langle 0x \to M_0; 1x \to \mathsf{c}M_1\rangle\!\rangle N)$ $(\mathsf{b}, \mathsf{c} \in \{0, 1\})$

(OUT 4) $\langle\!\langle 0x \to \mathsf{c}(\mathsf{b}M_0); 1x \to \overline{\mathsf{b}}M_1\rangle\!\rangle N \rhd \overline{\mathsf{b}}(\langle\!\langle 0x \to \mathsf{c}M_0; 1x \to M_1\rangle\!\rangle N)$ $(\mathsf{b}, \mathsf{c} \in \{0, 1\})$

(OUT 5) $\langle\!\langle 0x \to c; 1x \to c\rangle\!\rangle \rhd c$ $(c \in \{\mathsf{tt}, \mathsf{ff}, \mathsf{k}_n\})$

(BAR) $\overline{\mathsf{b}}(\mathsf{c}M) \rhd \mathsf{c}(\mathsf{b}M)$ $(\mathsf{b}, \mathsf{c} \in \{0, 1\})$

(PERM) $\langle 0x \to M_0; 1x \to M_1\rangle NL \rhd \langle 0x \to M_0L; 1x \to M_1L\rangle N$
$\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle NL \rhd \langle\!\langle 0x \to M_0L; 1x \to M_1L\rangle\!\rangle N$
(If $x \in FV(L)$, then rename the bound variable x to avoid variable collision.)

(LEFT) $\dfrac{M_0 \rhd M_0'}{\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle \rhd \langle\!\langle 0x \to M_0'; 1x \to M_1\rangle\!\rangle}$

(RIGHT) $\dfrac{M_1 \rhd M_1'}{\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle \rhd \langle\!\langle 0x \to M_0; 1x \to M_1'\rangle\!\rangle}$

(APP-R) $\dfrac{N \rhd N'}{MN \rhd MN'}$ (if M is $0, 1, \overline{0}, \overline{1}, \langle 0x \to M_0; 1x \to M_1\rangle$ or $\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle$)

Table 3
Operational semantics of XPCF

value of x. Then, (BAR) rule is used to arrange outputs of $M_0[0x/x]$ and $M_1[0x/x]$ to the form $\mathsf{b}_0\mathsf{b}_1 \ldots \mathsf{b}_k\overline{\mathsf{c}_0}\overline{\mathsf{c}_1} \ldots \overline{\mathsf{c}_j}$ for $\mathsf{b}_i, \mathsf{c}_i \in \{0, 1\}$. After that, if they coincide on the first or the second digit, then it makes an output with rules (OUT 1) to (OUT 5) and repeat it until no more output is possible. Thus, we obtain a term of the form $\mathsf{d}_0\mathsf{d}_1 \ldots \mathsf{d}_i(\langle\!\langle 0x \to M_0'; 1x \to M_1'\rangle\!\rangle\ N')$ and we can continue this process to the subterm $\langle\!\langle 0x \to M_0'; 1x \to M_1'\rangle\!\rangle\ N'$ with (APP-R) since all the rules applicable to $\langle 0x \to M_0; 1x \to M_1\rangle\ N$ are also applicable to $\langle\!\langle 0x \to M_0; 1x \to M_1\rangle\!\rangle\ N$.

One can see that the above reduction procedure fails to reduce $\langle 0x \to M_0; 1x \to M_1\rangle\ N\ L$ for M_0 and M_1 function type terms and $[\![N]\!] = \perp s$ because the output of L cannot be fed to function terms M_0 and M_1. For the evaluation of this term, we need the (PERM) rule. Suppose that $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle : \mathsf{S} \to \mathsf{S} \to \mathsf{S}$ and M_0 and M_1 are extended conditional terms of the form $\langle \mathsf{0}y \to ...; \mathsf{1}y \to ...\rangle$. We first reduce the term $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle\ N\ L$ to $\langle \mathsf{0}x \to (M_0\ L); \mathsf{1}x \to (M_1\ L)\rangle\ N$ with the (PERM) rule and then reduce it as we explained. The (PERM) rule corresponds to reducing the lambda term $(\lambda x.M)\ N\ L$ to $(\lambda x.(M\ L))\ N$, and it is similar to the permutative conversion rule used for proof normalization in proof theory [12].

One may wonder why we distinguish $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle$ with $\langle\!\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle\!\rangle$ because we can make the same reduction if we replace the former with the latter. However, strictness of $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle$ plays an important role in writing recursively defined functions. Many of the functions on S are defined with the Y operator as $F = \mathsf{Y}_{\mathsf{S}\to\mathsf{S}}(\lambda f.M)$ with $M = \langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle$ and it is reduced to $M[\mathsf{Y}_{\mathsf{S}\to\mathsf{S}}(\lambda f.M)/f]$ which cannot be reduced any more. If $M = \langle\!\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle\!\rangle$ instead, then copies of $\mathsf{Y}_{\mathsf{S}\to\mathsf{S}}(\lambda f.M)$ in M_0 and M_1 can be reduced with the rules (LEFT) and (RIGHT) and therefore it causes an infinite computation even if no argument is given to the function term F.

5.2 A sequential strategy of XPCF

Though one needs to evaluate M_0, M_1, and N in parallel for the evaluation of $M = \langle\!\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle\!\rangle\ N$, the procedure we mentioned above is almost sequential in that the evaluations of M_0 and M_1 are expected to terminate because they contain the free variable x in many cases. There are some cases that the evaluation of M_0 does not terminate and it outputs infinitely many digits. However, if M_0 has the form $\mathsf{d}_0\mathsf{d}_1M'$, then, from the forms of (OUT 1) to (OUT 4), one can consider that M_0 has enough outputs for M to make an output and terminate its reduction and proceed to the evaluation of M_1. We also need to take care of the case M_0 has the form $\langle\!\langle \mathsf{0}y \to M_{00}; \mathsf{1}y \to M_{11}\rangle\!\rangle\ L$. In this case, if we reduce M according to the procedure we mentioned above, and L is reduced to $\overline{\mathsf{0}}L_1 \rhd^* \overline{\mathsf{0}}^2 L_2 \rhd^* \cdots \rhd^* \overline{\mathsf{0}}^n L_n \rhd^* \cdots$, for example, then one repeats the application of (COND $\overline{\mathsf{0}}$) without instantiating the outputs of N to x. However, we can handle many of the cases by defining that $\langle\!\langle \mathsf{0}y \to M_{00}; \mathsf{1}y \to M_{11}\rangle\!\rangle\ L$ cannot be reduced if M_{00} and M_{11} cannot be reduced and all the appearances of y in M_{00} and M_{11} have the form $\mathsf{c}_0\mathsf{c}_1 \ldots \mathsf{c}_k y$ for $k > 1$ and $\mathsf{c}_i \in \{\mathsf{0}, \mathsf{1}\}$. Note that, in this case, further digits of y do not change the situation that M_{00} cannot be reduced. In this way, we designed a sequential reduction strategy of XPCF. We implemented it with Haskell. As it is proved in [7], in an interval domain model, an adequate real number calculus in which average function is definable does not have a sequential reduction strategy. It is also the case in our model and this sequential strategy is not adequate. Therefore, it does not evaluate all the terms to their denotations. However, we observed that applications of terms in Section 4 are reduced with our implementation and we expect that it evaluates many of the "meaningful" terms to their semantics.

6 Computational adequacy of XPCF

We show the soundness and completeness properties of XPCF. We first show that two kinds of substitutions in the reduction rule of XPCF preserve meanings.

Lemma 6.1 (i) *For terms $M : \tau$ and $N : \sigma$, a variable x^σ, and an environment ρ, $[\![M[N/x^\sigma]]\!](\rho) = [\![M]\!](\rho[[\![N]\!]/x^\sigma])$.*

(ii) *For a term M and $\mathsf{b} \in \{\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}\}$, $[\![M[\mathsf{b}x/x]]\!](\rho) = [\![M]\!](\rho[[\![\mathsf{b}]\!]\rho(x)/x])$.*

Proof. By structural induction on M. □

From Lemma 6.1, the following proposition holds.

Proposition 6.2 *For XPCF terms M, N and an environment ρ, if $M \triangleright N$ then $[\![M]\!](\rho) = [\![N]\!](\rho)$.*

Proof. It is proved by showing that the denotational semantics of the left side and the right side coincide for every reduction rule. □

In PCF, the result of evaluation of a program M of type σ is a constant of type σ if it exists. On the other hand, in XPCF, we consider non-terminating computations which output digits in $\{\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}\}$ as $M \triangleright \ldots \triangleright \mathsf{d}_0(\mathsf{d}_1 \cdots (\mathsf{d}_{n-1}M')) \triangleright \ldots$. Note that the sequence $\mathsf{d}_0, \mathsf{d}_1, \cdots$ is not determined uniquely by M. For example, the term $M = \langle\!\langle \mathsf{0}x \to \mathsf{0}(\mathsf{Y0}); \mathsf{1}x \to \mathsf{1}(\mathsf{Y0}) \rangle\!\rangle(\mathsf{1}\Omega_\mathsf{S})$ for $\Omega_\mathsf{S} = \mathsf{Y}_{\mathsf{S}\to\mathsf{S}}(\lambda x^\mathsf{S}.x^\mathsf{S})$ is reduced to terms of the forms $\mathsf{10}^n M'$ and $\overline{\mathsf{0}}^n N'$ for every n. However, from Proposition 6.2, if $M \triangleright^* \mathsf{d}_0(\mathsf{d}_1 \cdots (\mathsf{d}_{n-1}M'))$, then we have $d_0(d_1 ... (d_{n-1}\epsilon)) \sqsubseteq d_0(d_1 ... (d_{n-1}[\![M']\!])) = [\![M]\!]$ and thus the outputs are bounded by the denotation $[\![M]\!]$ of M and have the least upper bound. Therefore, we define an evaluation function **Eval** from XPCF programs of type σ to elements of D_σ as follows.

Definition 6.3 (i) For an XPCF program M of type N or B, we define

$$\mathbf{Eval}(M) = \begin{cases} [\![\mathrm{Eval}_{\mathrm{PCF}}(M)]\!] & \text{if } \mathrm{Eval}_{\mathrm{PCF}}(M) \text{ exists,} \\ \bot & \text{otherwise.} \end{cases}$$

(ii) For an XPCF program M of type S, we define

$$\mathbf{Eval}(M) = \bigsqcup \{ d_0(d_1 ... (d_{n-1}(\epsilon))) \mid M \triangleright^* \mathsf{d}_0(\mathsf{d}_1 \cdots (\mathsf{d}_{n-1}M')) \,\text{for some}\, M' \}$$

with $\mathsf{d}_i \in \{\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}\}$ and $d_i = [\![\mathsf{d}_i]\!]$ for $0 \leq i < n$.

The soundness of XPCF is derived from Proposition 6.2 immediately.

Theorem 6.4 (Soundness of XPCF) *For a program M, $\mathbf{Eval}(M) \sqsubseteq [\![M]\!]$.*

To show the completeness, we use the computability method (see [11]). That is, define the set Comp_σ of computable terms of type σ for each type σ and then show that all the XPCF terms are computable.

Definition 6.5 We define the predicate Comp_σ for each type σ by induction on types.

(i) Let σ be B or N. A program $M : \sigma$ has property Comp_σ if $[\![M]\!] = \mathbf{Eval}(M)$.

(ii) A program $M : \mathsf{S}$ has property Comp_S if $[\![M]\!] \sqsubseteq \mathbf{Eval}(M)$. That is, for any $p \in \Sigma^*_{\perp,1}$ with $p \sqsubseteq [\![M]\!]$, $p \sqsubseteq \mathbf{Eval}(M)$ holds.

(iii) A closed term $M : \sigma \to \tau$ has property $\mathrm{Comp}_{\sigma\to\tau}$ if whenever $N : \sigma$ is a closed term with property Comp_σ then MN is a term with property Comp_τ.

(iv) An open term $M : \sigma$ with free variables $x_1^{\sigma_1}, ..., x_n^{\sigma_n}$ has property Comp_σ if $M[N_1/x_1^{\sigma_1}]\cdots[N_n/x_n^{\sigma_n}]$ has property Comp_σ whenever $N_1, ..., N_n$ are closed terms having properties Comp_{σ_1},...,Comp_{σ_n} respectively.

We say that a term of type σ is *computable* if it has property Comp_σ.

It is immediate to show the followings. (1) If $M : \sigma \to \tau$ and $N : \sigma$ are closed computable terms, so is MN. (2) For a ground type τ, a term $M : \sigma_1 \to \cdots \to \sigma_n \to \tau$ is computable if and only if $\tilde{M} N_1 \cdots N_n$ is computable for all closed computable terms $N_1 : \sigma_1, ..., N_n : \sigma_n$ and closed instantiation $\tilde{M}$ of M by computable terms.

For $s \in \Sigma^*_{\perp,1}$, we define the context $\underline{s}[X]$ as follows,

$$\underline{s}[X] = \begin{cases} \mathsf{b}_0(\mathsf{b}_1 \cdots (\mathsf{b}_{n-1} X)) & \text{if } s = b_0 b_1 \cdots b_{n-1}, \\ \mathsf{b}_0(\mathsf{b}_1 \cdots (\mathsf{b}_{n-1}(\overline{\mathsf{c}_0}(\overline{\mathsf{c}_1} \cdots (\overline{\mathsf{c}_{m-1}} X))))) & \text{if } s = b_0 b_1 \cdots b_{n-1} \perp c_0 c_1 \cdots c_{m-1}. \end{cases}$$

Here, $\mathsf{b}_i, \mathsf{c}_j \in \{\mathsf{0}, \mathsf{1}\}$, $b_i = [\![\mathsf{b}_i]\!]$, and $c_j = [\![\mathsf{c}_j]\!]$ for $0 \le i < n$ and $0 \le j < m$. We say that a term M of type S *outputs* $s \in \Sigma^*_{\perp,1}$ if there is a reduction $M \rhd^* \underline{s}[M']$ for some M'.

Lemma 6.6 *Let $M : \mathsf{S}$ be a computable term such that x is the only free variable and let $s \in \Sigma^*$. For any $p \sqsubseteq [\![M]\!](\rho[s/x])$ with $p \in \Sigma^*_{\perp,1}$, $M[\underline{s}[x]/x]$ outputs $q \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq q$*

Proof. We have $[\![M]\!](\rho[\epsilon/x]) = [\![M[\Omega_\mathsf{S}/x]]\!]$ by structural induction on M. From the equation $[\![\underline{s}[\Omega_\mathsf{S}]]\!] = s$ and Lemma 6.1 (i), we have

$$[\![M]\!](\rho[s/x]) = [\![M]\!](\rho[[\![\underline{s}[\Omega_\mathsf{S}]]\!]/x]) = [\![M[\underline{s}[\Omega_\mathsf{S}]/x]]\!].$$

Since M and $\underline{s}[\Omega_\mathsf{S}]$ are computable, $M[\underline{s}[\Omega_\mathsf{S}]/x]$ is computable. Therefore, for any $p \in \Sigma^*_{\perp,1}$ with $p \sqsubseteq [\![M]\!](\rho[s/x])$, there exists a reduction $M[\underline{s}[\Omega_\mathsf{S}]/x] \rhd^* \underline{q}[M']$ with $q \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq q$. If there is a reduction sequence $M[\underline{s}[\Omega_\mathsf{S}]/x] \rhd^* \underline{q}[M']$, then there is a reduction sequence $M[\underline{s}[x]/x] \rhd^* \underline{q}[M'']$ by ignoring the reductions related to Ω_S. Therefore, $M[\underline{s}[x]/x]$ outputs q such that $p \sqsubseteq q$. □

Proposition 6.7 *Every XPCF term is computable.*

Proof. We prove it by structural induction on terms.

In order to prove the computability of Y_σ for an XPCF type σ, we use an extension of the syntactic information order in [11], which we omit here. We only explain the proof of the cases $\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}, \langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1 \rangle$ and $\langle\!\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1 \rangle\!\rangle$.

The case $\mathsf{d} \in \{\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}\}$. We show that for any computable term M of type S, $\mathsf{d}M$ is computable. Because the function $[\![\mathsf{d}]\!] : D_\mathsf{S} \to D_\mathsf{S}$ is continuous, for any $p \in \Sigma^*_{\perp,1}$ with $p \sqsubseteq [\![\mathsf{d}M]\!] = [\![\mathsf{d}]\!]([\![M]\!])$, there exists $q \in \Sigma^*_{\perp,1}$ with $q \sqsubseteq [\![M]\!]$ such that $p \sqsubseteq [\![\mathsf{d}]\!](q)$. Since M is computable, M outputs $q' \in \Sigma^*_{\perp,1}$ such that $q \sqsubseteq q'$. Therefore, $\mathsf{d}M$ outputs $[\![\mathsf{d}]\!](q')$ which satisfies $p \sqsubseteq [\![\mathsf{d}]\!](q) \sqsubseteq [\![\mathsf{d}]\!](q')$ and thus d is computable.

We show that if terms M_0 and M_1 of type σ are computable, so is the term $\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1\rangle$. It is enough to show that the term $\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 N_2 \cdots N_n$ of type a ground type τ is computable when $N_1 : \mathsf{S}, N_2, ..., N_n$ are closed computable terms and $\tilde{M_0}$ and $\tilde{M_1}$ are instantiations of all free variables, except x, of M_0 and M_1 by closed computable terms, respectively. We only show the case $\tau = \mathsf{S}$.

Case $[\![N_1]\!] = \epsilon$. From the reduction rule, we have the following equation:

$$\begin{aligned}
&[\![\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 N_2 \cdots N_n]\!] \\
&\qquad = [\![\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle]\!](\epsilon)([\![N_2]\!]) \cdots ([\![N_n]\!]) \\
&\qquad = \perp_\sigma([\![N_2]\!]) \cdots ([\![N_n]\!]) = \perp_\mathsf{S}.
\end{aligned}$$

Therefore, $\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 N_2 \cdots N_n$ is computable.

Case $[\![N_1]\!] = 0s$. For any $p \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq [\![\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 \cdots N_n]\!] = [\![\tilde{M_0}]\!]\rho([s/x])([\![N_2]\!]) \cdots ([\![N_n]\!])$, from the continuity, there exists $s' \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq [\![\tilde{M_0}]\!]\rho([s'/x])([\![N_2]\!]) \cdots ([\![N_n]\!])$ and $0s' \sqsubseteq [\![N_1]\!]$. From the computability of N_1, there exists $0s'' \in \Sigma^*_{\perp,1}$ such that N_1 outputs $0s''$ and $0s' \sqsubseteq 0s''$. Then, $p \sqsubseteq [\![\tilde{M_0}]\!]\rho([s''/x])([\![N_2]\!]) \cdots ([\![N_n]\!])$ holds. Since we have $[\![\tilde{M_0}[\underline{s''}[\Omega_\mathsf{S}]/x]N_2 \cdots N_n]\!] = [\![\tilde{M_0}]\!]\rho([s''/x])([\![N_2]\!]) \cdots ([\![N_n]\!])$ and $\underline{s''}[\Omega_\mathsf{S}]$ is computable, $\tilde{M_0}[\underline{s''}[\Omega_\mathsf{S}]/x]N_2 \cdots N_n$ is also computable and outputs $t \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq t$. Therefore, we have a reduction sequence $\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 \cdots N_n \rhd^* \langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle \underline{0s''}[N_1'] \cdots N_n \rhd^* \underline{t}[M'']$ such that $p \sqsubseteq t$.

Case $[\![N_1]\!] = 1s$. The proof is similar to the case $[\![N_1]\!] = 0s$.

Case $[\![N_1]\!] = \perp u$ with $u \in \Sigma^\infty \backslash \{\epsilon\}$. From the reduction rule, we have the following equation:

$$\begin{aligned}
&[\![\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 \cdots N_n]\!] \\
&\qquad = [\![\langle \mathsf{0}x \to \tilde{M_0} N_2 \cdots N_n; \mathsf{1}x \to \tilde{M_1} N_2 \cdots N_n\rangle N_1]\!] \\
&\qquad = [\![\langle \mathsf{0}x \to \tilde{M_0} N_2 \cdots N_n; \mathsf{1}x \to \tilde{M_1} N_2 \cdots N_n\rangle]\!](\perp u) \\
&\qquad = [\![\tilde{M_0} N_2 \cdots N_n]\!](\rho[u/x]) \sqcap [\![\tilde{M_1} N_2 \cdots N_n]\!](\rho[u/x]).
\end{aligned}$$

Because of the continuity of $[\![\tilde{M_0} N_2 \cdots N_n]\!]$ and $[\![\tilde{M_1} N_2 \cdots N_n]\!]$, for any $p \in \Sigma^*_{\perp,1}$ with $p \sqsubseteq [\![\langle \mathsf{0}x \to \tilde{M_0}; \mathsf{1}x \to \tilde{M_1}\rangle N_1 \cdots N_n]\!]$, there is $s \in \Sigma^*$ such that $s \sqsubseteq u$, $p \sqsubseteq [\![\tilde{M_0} N_2 \cdots N_n]\!](\rho[s/x])$, and $p \sqsubseteq [\![\tilde{M_1} N_2 \cdots N_n]\!](\rho[s/x])$. Since N_1 is computable, N_1 outputs $\perp s$. By Lemma 6.6, $(\tilde{M_0} N_2 \cdots N_n)[\underline{s}[x]/x]$ outputs $q_0 \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq q_0$ and $(\tilde{M_1} N_2 \cdots N_n)[\underline{s}[x]/x]$ outputs $q_1 \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq q_1$. Therefore,

$\langle 0x \to \tilde{M}_0; 1x \to \tilde{M}_1 \rangle N_1 \cdots N_n$ has the following reduction:

$$\begin{aligned}
&\langle 0x \to \tilde{M}_0; 1x \to \tilde{M}_1 \rangle N_1 \cdots N_n \\
&\quad \rhd^* \langle 0x \to \tilde{M}_0 N_2 \cdots N_n; 1x \to \tilde{M}_1 N_2 \cdots N_n \rangle \underline{\perp s}[N_1'] \\
&\quad \rhd^* \langle\!\langle 0x \to (\tilde{M}_0 N_2 \cdots N_n)[\underline{s}[x]/x]; 1x \to (\tilde{M}_1 N_2 \cdots N_n)[\underline{s}[x]/x] \rangle\!\rangle N_1' \\
&\quad \rhd^* \langle\!\langle 0x \to \underline{q_0}[M_0']; 1x \to \underline{q_1}[M_1'] \rangle\!\rangle N_1' \\
&\quad \rhd^* \underline{q}[M']
\end{aligned}$$

for some $q \in \Sigma^*_{\perp,1}$ such that $p \sqsubseteq q \sqsubseteq (q_0 \sqcap q_1)$.

The computability proof of $\langle\!\langle 0x \to \tilde{M}_0; 1x \to \tilde{M}_1 \rangle\!\rangle$ is that of $\langle 0x \to M_0; 1x \to M_1 \rangle$ without the case $[\![N_1]\!] = \epsilon$ and without restricting in the final case to $u \notin \epsilon$. □

Therefore, the completeness of XPCF holds.

Theorem 6.8 (Completeness of XPCF) *For a program M,* $\mathbf{Eval}(M) \sqsupseteq [\![M]\!]$.

Combining the soundness and completeness of XPCF, we have the computational adequacy of XPCF. That is, $\mathbf{Eval}(M) = [\![M]\!]$ for every program M.

7 Expressive power of XPCF

In this section, we often omit the type and write x for x^σ and if for if_σ when no confusion can arise.

We compare expressive powers of XPCF and PCF$^+$. Here, PCF$^+$ is the calculus PCF extended with the parallel conditional $\mathsf{pif}_\sigma : \mathsf{B} \to \sigma \to \sigma \to \sigma$ as a constant for each $\sigma \in \{\mathsf{B}, \mathsf{N}\}$. The interpretation of pif_σ is given as follows

$$[\![\mathsf{pif}_\sigma]\!] = \lambda b \in D_{\mathsf{B}}.\lambda x \in D_\sigma.\lambda y \in D_\sigma. \begin{cases} x & (b = tt) \\ y & (b = ff) \\ x & (b = \perp_{\mathsf{B}} \text{ and } x = y) \\ \perp_\sigma & (\text{otherwise}). \end{cases}$$

The operational semantics of PCF$^+$ is the operational semantics of PCF together with:

$$\mathsf{pif}_\sigma\, M\, c\, c \rhd c, \qquad \mathsf{pif}_\sigma\, \mathsf{tt}\, M\, N \rhd M, \qquad \mathsf{pif}_\sigma\, \mathsf{ff}\, M\, N \rhd N,$$

$$\frac{M \rhd M'}{\mathsf{pif}_\sigma\, M \rhd \mathsf{pif}_\sigma\, M'}, \quad \frac{N \rhd N'}{\mathsf{pif}_\sigma\, M\, N \rhd \mathsf{pif}_\sigma\, M\, N'}, \quad \frac{L \rhd L'}{\mathsf{pif}_\sigma\, M\, N\, L \rhd \mathsf{pif}_\sigma\, M\, N\, L'}.$$

Consider the following XPCF term $\mathbf{pif}'_\sigma$ of type $\mathsf{B} \to \sigma \to \sigma \to \sigma$ for $\sigma \in \{\mathsf{B}, \mathsf{N}\}$.

$$\mathbf{pif}'_\sigma = \lambda u^{\mathsf{B}}.\lambda y^\sigma.\lambda z^\sigma.\langle\!\langle 0x \to y; 1x \to z \rangle\!\rangle(\mathsf{if}_{\mathsf{S}}\, u\, (0\Omega_{\mathsf{S}})\, (1\Omega_{\mathsf{S}})).$$

It satisfies $[\![\mathbf{pif}'_\sigma]\!] = [\![\mathsf{pif}_\sigma]\!]$ and therefore it expresses the pif_σ operator. Note that one can also express it as an XPCF program

$$\lambda u^{\mathsf{B}}.\lambda y^\sigma.\lambda z^\sigma.\langle 0x \to y; 1x \to z \rangle \overline{0}(\mathsf{if}_{\mathsf{S}}\, u\, (0\Omega_{\mathsf{S}})\, (1\Omega_{\mathsf{S}}))$$

without using $\langle\!\langle ... \rangle\!\rangle$. Thus, PCF$^+$ terms can be translated into XPCF terms by replacing pif_σ with $\mathbf{pif}'_\sigma$.

Theorem 7.1 *For a PCF$^+$ term $M : \sigma$ and a PCF environment ρ, there exists an XPCF term $M' : \sigma$ such that $[\![M]\!](\rho) = [\![M']\!](\rho')$. Here, ρ' is any extension of ρ to an XPCF environment.*

On the other hand, there is an embedding-projection pair (e-p pair in short) (e, p) between the domains $D_{\mathsf{S}} = \mathbf{BD}$ and $D_{\mathsf{N}\to\mathsf{B}} \cong \Sigma_\perp^\omega$ where the projection p is the function trunc in Section 2.2. Here, a pair of continuous functions $e : X \to Y$ and $p : Y \to X$ is an e-p pair if they satisfy $p \circ e = id_X$ and $e \circ p \sqsubseteq id_Y$. Terms $\mathbf{e} : \mathsf{S} \to (\mathsf{N} \to \mathsf{B})$ and $\mathbf{p} : (\mathsf{N} \to \mathsf{B}) \to \mathsf{S}$ such that $[\![\mathbf{e}]\!] = e$ and $[\![\mathbf{p}]\!] = p$ can be written in XPCF as follows,

$$\begin{aligned}
\mathbf{e} :&= \mathsf{Y}_{\mathsf{S}\to\mathsf{N}\to\mathsf{B}}(\lambda f^{\mathsf{S}\to\mathsf{N}\to\mathsf{B}}.\lambda g^{\mathsf{S}}.\lambda n^{\mathsf{N}}.\mathsf{if}\,(\mathsf{zero}\,n)\,(\mathbf{head}\,g)\,(f\,(\mathbf{tail}\,g)\,(\mathsf{dec}\,n)))\\
\mathbf{p} :&= \mathsf{Y}_{\mathsf{N}\to(\mathsf{N}\to\mathsf{B})\to\mathsf{S}}(\lambda g^{\mathsf{N}\to(\mathsf{N}\to\mathsf{B})\to\mathsf{S}}.\lambda n^{\mathsf{N}}.\lambda f^{\mathsf{N}\to\mathsf{B}}.\langle 0x \to 0(g\,(\mathsf{inc}\,n)\,f); 1x \to 1(g\,(\mathsf{inc}\,n)\,f)\rangle\\
&\qquad\qquad\qquad \bar{0}(\mathsf{if}(f\,n)(0\Omega_{\mathsf{S}})(1\Omega_{\mathsf{S}})))\mathsf{k}_0
\end{aligned}$$

where $\mathbf{tail} = \langle 0x \to x; 1x \to x\rangle$ and $\mathbf{head} = \langle 0x \to \mathsf{tt}; 1x \to \mathsf{ff}\rangle$.

We can extend the e-p pair (e, p) to higher order types. We inductively define σ^t for every XPCF type σ as follows

$$\mathsf{B}^t = \mathsf{B}, \mathsf{N}^t = \mathsf{N}, \mathsf{S}^t = \mathsf{N} \to \mathsf{B}, \text{and } (\sigma \to \tau)^t = \sigma^t \to \tau^t.$$

We inductively define $\mathbf{e}_\sigma : \sigma \to \sigma^t$ and $\mathbf{p}_\sigma : \sigma^t \to \sigma$ for every XPCF type σ as follows,

$$\begin{aligned}
\mathbf{e}_{\mathsf{N}} = \mathbf{p}_{\mathsf{N}} = \lambda x^{\mathsf{N}}.x, \quad \mathbf{e}_{\mathsf{B}} &= \mathbf{p}_{\mathsf{B}} = \lambda x^{\mathsf{B}}.x, \quad \mathbf{e}_{\mathsf{S}} = \mathbf{e}, \quad \mathbf{p}_{\mathsf{S}} = \mathbf{p},\\
\mathbf{e}_{\sigma\to\tau} &= \lambda f^{\sigma\to\tau}.\lambda x^{\sigma^t}.\mathbf{e}_\tau(f(\mathbf{p}_\sigma(x))),\\
\mathbf{p}_{\sigma\to\tau} &= \lambda f^{\sigma^t\to\tau^t}.\lambda x^{\sigma}.\mathbf{p}_\tau(f(\mathbf{e}_\sigma(x))).
\end{aligned}$$

It is immediate to show that $([\![\mathbf{e}_\sigma]\!], [\![\mathbf{p}_\sigma]\!])$ is an e-p pair for every type σ.

We define a syntactical translation $(-)^t$ from XPCF terms to PCF$^+$ terms so that $M^t : \sigma^t$ for $M : \sigma$. Before that, we define a function $r : D_{\mathsf{N}\to\mathsf{B}} \to D_{\mathsf{N}\to\mathsf{B}}$ as $r = e \circ p$ and $r_\sigma : D_{\sigma^t} \to D_{\sigma^t}$ as $r_\sigma = e_\sigma \circ p_\sigma$. The function r satisfies

$$r(f)(n) = \begin{cases} tt & \text{if } f(n) = tt \text{ and } \perp \text{ appears at most once in } f(0), \cdots, f(n-1)\\ ff & \text{if } f(n) = ff \text{ and } \perp \text{ appears at most once in } f(0), \cdots, f(n-1)\\ \perp & \text{otherwise.} \end{cases}$$

for $f : D_{\mathsf{N}} \to D_{\mathsf{B}}$ and $n \in D_{\mathsf{N}}$. Let $\mathbf{r}$ be any PCF$^+$ term such that $[\![\mathbf{r}]\!] = r$. For every XPCF type σ, we inductively define a PCF$^+$ term $\mathbf{r}_\sigma : \sigma^t \to \sigma^t$ which satisfies $[\![\mathbf{r}_\sigma]\!] = r_\sigma$ as follows

$$\mathbf{r}_{\mathsf{B}} = \lambda x^{\mathsf{B}}.x^{\mathsf{B}}, \mathbf{r}_{\mathsf{N}} := \lambda x^{\mathsf{N}}.x^{\mathsf{N}}, \mathbf{r}_{\mathsf{S}} := \mathbf{r}, \text{and } \mathbf{r}_{\sigma\to\tau} = \lambda f^{\sigma^t\to\tau^t}.\lambda x^{(\sigma^t)}.\mathbf{r}_\tau(f(\mathbf{r}_\sigma x)).$$

For an XPCF term M, we inductively define $M^t : \sigma^t$ as follows,

$$
\begin{aligned}
&(x^\sigma)^t = \mathbf{r}_\sigma x^{(\sigma^t)}, \quad \mathsf{c}^t = \mathsf{c}, \quad \mathsf{if}^t_\sigma = \mathsf{if}_\sigma, \quad \mathsf{Y}^t_\sigma = \lambda f^{\sigma^t \to \sigma^t}.\mathsf{Y}_{\sigma^t}(\mathbf{r}_{\sigma\to\sigma} f),\\
&(\lambda x^\sigma.M)^t = \lambda x^{(\sigma^t)}.M^t, \qquad (MN)^t = (M^t N^t),\\
&\mathsf{0}^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\lambda x^{\mathsf{N}}.\mathsf{if}\,(\mathsf{zero}\,x)\,\mathsf{tt}\,((\mathbf{r}_\mathsf{S}\,f)\,(\mathsf{dec}\,x)),\\
&\mathsf{1}^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\lambda x^{\mathsf{N}}.\mathsf{if}\,(\mathsf{zero}\,x)\,\mathsf{ff}\,((\mathbf{r}_\mathsf{S}\,f)\,(\mathsf{dec}\,x)),\\
&\overline{\mathsf{0}}^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\lambda x^{\mathsf{N}}.\mathsf{if}\,(\mathsf{zero}\,x)\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_0)\\
&\qquad\qquad (\mathsf{if}\,(\mathsf{zero}\,(\mathsf{dec}\,x))\,\mathsf{tt}\,((\mathbf{r}_\mathsf{S}\,f)\,(\mathsf{dec}\,x))),\\
&\overline{\mathsf{1}}^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\lambda x^{\mathsf{N}}.\mathsf{if}\,(\mathsf{zero}\,x)\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_0)\\
&\qquad\qquad (\mathsf{if}\,(\mathsf{zero}\,(\mathsf{dec}\,x))\,\mathsf{ff}\,((\mathbf{r}_\mathsf{S}\,f)\,(\mathsf{dec}\,x))),\\
&\langle\!\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1 \rangle\!\rangle^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\mathsf{pif}\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_0)\,M_0^t[\lambda y^{\mathsf{N}}.(\mathbf{r}_\mathsf{S}\,f)(\mathsf{inc}\,y)/x^{\mathsf{N}\to\mathsf{B}}]\\
&\qquad\qquad M_1^t[\lambda y^{\mathsf{N}}.(\mathbf{r}_\mathsf{S}\,f)(\mathsf{inc}\,y)/x^{\mathsf{N}\to\mathsf{B}}],\\
&\langle \mathsf{0}x \to M_0; \mathsf{1}x \to M_1 \rangle^t = \lambda f^{\mathsf{N}\to\mathsf{B}}.\mathsf{if}(\mathsf{pif}\,(\mathsf{if}\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_0)\,\mathsf{tt}\,\mathsf{tt})\,\mathsf{tt}\,(\mathsf{if}\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_1)\,\mathsf{tt}\,\mathsf{tt}))\\
&\qquad\qquad (\mathsf{pif}\,((\mathbf{r}_\mathsf{S}\,f)\,\mathsf{k}_0)\,M_0^t[\lambda y^{\mathsf{N}}.(\mathbf{r}_\mathsf{S}\,f)(\mathsf{inc}\,y)/x^{\mathsf{N}\to\mathsf{B}}]\\
&\qquad\qquad M_1^t[\lambda y^{\mathsf{N}}.(\mathbf{r}_\mathsf{S}\,f)(\mathsf{inc}\,y)/x^{\mathsf{N}\to\mathsf{B}}])\,\Omega_\sigma
\end{aligned}
$$

where c is a constant other than $\mathsf{0}, \mathsf{1}, \overline{\mathsf{0}}, \overline{\mathsf{1}}, \mathsf{if}_\sigma$ or Y_σ and the type of terms M_0 and M_1 is σ. Here, we assume that the same XPCF variable does not appear in different types to prevent conflictions in the translation to PCF$^+$ terms.

We define a translation $(\text{-})^t$ of environments as $\rho^t(x^{\sigma^t}) = e_\sigma(\rho(x^\sigma))$.

Proposition 7.2 *For any term $M : \sigma$ in XPCF and environment ρ, $e_\sigma([\![M]\!](\rho)) = [\![M^t]\!](\rho^t)$ holds.*

Proof. By structural induction on M. □

It is known that in PCF$^+$ all compact elements and all computable first-order functions are definable [11]. Through the translation $(\text{-})^t$, we can derive the following results on expressive power of XPCF.

Theorem 7.3 (i) *XPCF and PCF$^+$ have the same expressive power on PCF types.*

(ii) *All computable elements of D_S are definable in XPCF.*

(iii) *The function exist is not definable in XPCF. Here, exist : $D_{\mathsf{N}\to\mathsf{B}} \to D_\mathsf{B}$ is the function*

$$
\lambda f \in D_{\mathsf{N}\to\mathsf{B}}. \begin{cases} ff & f(\perp) = ff \\ tt & \exists n \in \mathbb{N}. f(n) = tt \\ \perp & \textit{otherwise.} \end{cases}
$$

Proof. (i) Since e_σ is the identity function if σ does not contain S, Theorem 7.1 and Proposition 7.2 show that XPCF and PCF$^+$ have the same expressive power on PCF types.

(ii) For any computable element $x \in D_{\mathsf{S}}$, $e_{\mathsf{S}}(x) \in D_{\mathsf{N} \to \mathsf{B}}$ is a computable element because e_{S} is a computable function. Therefore, $e_{\mathsf{S}}(x)$ is definable in PCF$^+$ [11]. Since p_{S} is definable in XPCF, $p_{\mathsf{S}}(e_{\mathsf{S}}(x)) = x$ is definable in XPCF.

(iii) Suppose that there exists a closed XPCF term $M : (\mathsf{N} \to \mathsf{B}) \to \mathsf{B}$ such that $[\![M]\!] = exist$. Since e_σ is identity for a PCF type σ, we have $[\![M]\!] = e_{(\mathsf{N}\to\mathsf{B})\to\mathsf{B}}([\![M]\!]) = [\![M^t]\!]$ from Proposition 7.2. However, *exist* is not definable in PCF$^+$ [11] and this is a contradiction. Therefore, *exist* is not definable in XPCF.□

In [11], Plotkin introduced the language PCF^{++} which is an extension of PCF$^+$ by adding the existential quantifier $\exists : (\mathsf{N} \to \mathsf{B}) \to \mathsf{B}$ as a constant such that $[\![\exists]\!] = exist$. [5] showed that Real PCF extended with $\exists$ is universal, based on a technique due to Thomas Streicher [13] to establish that PCF extended with recursive types, parallel-or and $\exists$ is universal. We define a calculus XPCF$^\exists$, which is the extension of XPCF with the $\exists$ operator. XPCF$^\exists$ is universal in the following sense.

Theorem 7.4 *For every XPCF type σ, all computable elements of D_σ are definable in XPCF$^\exists$.*

Proof. For any XPCF type σ and computable element $x \in D_\sigma$, $e_\sigma(x) \in D_{\sigma^t}$ is a computable element because e_σ is a computable function. Therefore, $e_\sigma(x)$ is definable in PCF^{++} by [11]. Since p_σ is definable in XPCF, $p_\sigma(e_\sigma(x)) = x$ is definable in XPCF$^\exists$. □

Acknowledgement

The authors would like to thank to Takashi Sakuragawa for valuable comments. This work was partly supported by JSPS KAKENHI Grant Number 22500014.

References

[1] Berger, U., and Hou, T., *Coinduction for Exact Real Number Computation*. Theory of Computing Systems. **43** (2008), 394-409.

[2] Boehm, H.J., Cartwright, R., Riggle, M., and O'Donnell, M.J., "Exact real arithmetic: a case study in higher order programming", ACM Symposium on Lisp and Functional Programming, ACM, New York, 1986.

[3] Blanck, J., *Domain representations of topological spaces*. Theoretical Computer Science. **247** (2000), 229–255.

[4] Brattka, V., and Hertling, P., *Topological properties of real number representations*. Theoretical Computer Science. **284** (2002), 241-257.

[5] Escardó, M.H., *Real PCF extended with ∃ is universal*. Advances in Theory and Formal Methods of Computing:Proceedings of the Third Imperial Collage Workshop. (1996), 13-24.

[6] Escardó, M.H., *PCF extended with real numbers*. Theoretical Computer Science. **162** (1996), 79-115.

[7] Escardó, M.H., Hofmann, M., and Streicher, T., *On the non-sequential nature of the interval-domain model of real-number computation*, Math. Struct. in Comp. Science. **14**(6) (2004), 803-814.

[8] Di Gianantonio, P., *An abstract data type for real numbers*. Theoretical Computer Science. **221** (1999), 295-326.

[9] Gierz, G., Hofmann, K.H., Keimel, K., Lawson, J.D., Mislove, M., and Scott, D.S., "Continuous Lattices and Domains", Cambridge University Press, 2003.

[10] Gray, F., "Pulse code communication", March 17, 1953 (filed Nov. 1947). U.S. Patent 2,632,058.

[11] Plotkin, G., *LCF considered as a programming language*, Theoretical Computer Science. **5** (1977), 223-255.

[12] Schwichtenberg, H., and Wainer, S.S., *Proofs and Computations*, Cambridge University Press, 2012.

[13] Streicher, T., *A universality theorem for PCF with recursive types, parallel-or and* ∃. Mathematical Structures for Computing Science. **4**(1) (1994), 111-115.

[14] Streicher, T., "Domain-Theoretic Foundations of Functional Programming", World Scientific, 2006.

[15] Tsuiki, H., *Real number computation through gray code embedding.* Theoretical Computer Science. **284**(2) (2002), 467-485.

[16] Tsuiki, H., *Compact metric spaces as minimal-limit sets in domains of bottomed sequences.* Math. Struct. in Comp. Science. **14** (2004), 853-878.

[17] Weihrauch, K., "An Introduction to Computable Analysis", Springer, Berlin, 2000.