Induction and recursion on the partial real line via biquotients of bifree algebras

Martín Hötzel Escardó
Department of Computing
Imperial College
180 Queen's Gate
London SW7 2BZ, United Kingdom
m.escardo@doc.ic.ac.uk

Thomas Streicher
Fachbereich Mathematik
Technische Hochschule Darmstadt
Schloßgartenstraße 7
64289 Darmstadt, Germany
streicher@mathematik.th-darmstadt.de

Abstract

The partial real line is the continuous domain of compact real intervals ordered by reverse inclusion. The idea is that singleton intervals represent total real numbers, and that the remaining intervals represent partial real numbers. The partial real line has been used to model exact real number computation in the framework of the programming language Real PCF.

We introduce induction principles and recursion schemes for the partial unit interval, which allow us to verify that Real PCF programs meet their specification. The theory is based on a domain-equation-like presentation of the partial unit interval, which we refer to as a biquotient of a bifree algebra.

1. Introduction

The *partial real line* is the continuous domain of compact real intervals ordered by reverse inclusion [20]. The idea is that singleton intervals represent *total real numbers*, and that the remaining intervals represent (properly) *partial real numbers*. This is justified by the fact that the singleton map $x \mapsto \{x\}$ is a topological embedding of the Euclidean real line into the partial real line endowed with its Scott topology. Moreover, the Lawson topology on the partial real line coincides with the Hausdorff topology considered in interval analysis [15], as it is shown in [9].

The partial real line has been used to model exact real number computation in the framework of the programming language Real PCF [6, 8], including computation of integrals [3].

We introduce induction principles and recursion schemes for the *partial unit interval* (the domain of closed subintervals of the unit interval with end-points 0 and 1), which allow us to verify that Real PCF programs meet their specification.

The induction principles and recursion schemes discussed in this paper resemble the so-called *Peano axioms* for natural numbers, and they abstractly characterize the partial unit interval up to isomorphism, without reference to real numbers or intervals. Essentially, we replace zero and the successor function by the linear maps $x \mapsto x/2$ and $x \mapsto (x+1)/2$, which play the rôle of partial real number constructors. This is related to binary expansions.

Preliminary ideas on recursion and induction on the real line appeared in [5], which considers uniform spaces. Our axioms are formulated in the ambient category of continuous Scott domains [1].

In domain theory one usually derives induction principles and recursion schemes from canonical solutions of domain equations [14, 24, 18]. Since the partial real line is not algebraic, it is not the canonical solution of any domain equation involving usual functors.

We establish new results about the notion of inductive retraction introduced in [8], which generalizes canonical solutions of domain equations by means of ideas similar to those of Freyd [11, 12]. In particular, we introduce the notion of a biquotient of a bifree algebra, and we show that the inductive retractions are the biquotients of the bifree algebras.

An interesting observation is that the Peano-like axioms discussed above consider only induction and recursion, but the inductive retraction induced by them gives rise to coinduction and corecursion.

The techniques discussed here in a more general

setting were also applied in conjunction with the technique introduced in [26] to establish universality of Real PCF extended with a certain computable existential quantifier [8].

2. The partial unit interval

In this paper a ***domain*** is a bounded complete continuous dcpo with bottom [1]. The domain $\mathcal{I} = \mathbf{I}[0,1]$ of closed subintervals of the unit interval $\mathbb{I} = [0,1]$, ordered by reverse inclusion, is referred to as the ***partial unit interval***. The elements of $\mathcal{I}$ are referred to as ***partial real numbers***, and a real number r is notationally identified with the singleton interval $\{r\}$ and referred to as a ***total real number***.

Any continuous function $f : \mathbb{I} \to \mathbb{I}$ has a Scott continuous ***canonical extension*** $\mathbf{I}f : \mathcal{I} \to \mathcal{I}$ defined by $\mathbf{I}f(x) = \{f(r) | r \in x\}$. If f is increasing w.r.t. to the natural order of real numbers then $\mathbf{I}f(x) = [f(\inf x), f(\sup x)]$. A function $f : \mathbb{I} \to \mathbb{I}$ is notationally identified with its canonical extension $\mathbf{I}f : \mathcal{I} \to \mathcal{I}$, and we often define a function $f : \mathcal{I} \to \mathcal{I}$ by first defining a function $f : \mathbb{I} \to \mathbb{I}$ and then implicitly taking its canonical extension.

3. Peano-like axioms for the partial unit interval

Full proofs of the result of this section can be found in [7]. Define $\mathrm{cons}_L, \mathrm{cons}_R : \mathcal{I} \to \mathcal{I}$ by

$$\mathrm{cons}_L(x) = x/2 \qquad \mathrm{cons}_R(x) = (x+1)/2.$$

That is, cons_L and cons_R are the unique increasing affine maps such that

$$\mathrm{cons}_L(\bot) = L \qquad \mathrm{cons}_R(\bot) = R,$$

where $L = [0, 1/2]$ and $R = [1/2, 1]$.

In this section we consider cons_L and cons_R as partial real number "constructors", in a similar fashion as zero and successor are considered as natural number constructors. An important difference is that the natural numbers together with zero and successor form a free algebra, whereas $\mathcal{I}$ together with cons_L and cons_R will form a kind of quotient of a free algebra with respect to some equations. The main such equation is

$$\mathrm{cons}_L(1) = \mathrm{cons}_R(0).$$

Another important difference is that natural numbers are constructed from zero by finitely many applications of the successor function, whereas the elements of $\mathcal{I}$ are constructed "from nothing" by infinitely many applications of cons_L and cons_R. For example, every *total* $x \in \mathcal{I}$ can be constructed as

$$x = \bigsqcup_n{}^{\uparrow} \mathrm{cons}_{a_1} \circ \cdots \circ \mathrm{cons}_{a_n}(\bot)$$

for some sequence $a_i \in \{L, R\}$ corresponding to a binary expansion of x. Partial real numbers are constructed by iterating cons_L and cons_R in a more elaborate way, as it is shown in §5.2.

The predecessor function, undefined or arbitrarily defined at zero, is a left inverse of the successor function. Similarly, cons_a has a left inverse tail_a defined by

$$\mathrm{tail}_L(x) = \min(2x, 1), \qquad \mathrm{tail}_R(x) = \max(0, 2x-1).$$

The fact that every number is either zero or a successor can be expressed by the equation

$$n = \text{if } n = 0 \text{ then } 0 \text{ else } \mathrm{succ}(\mathrm{pred}(n)).$$

Let $\mathcal{B}$ be the flat domain of truth values, define $\mathrm{head} : \mathcal{I} \to \mathcal{B}$ by

$$\mathrm{head}(x) = \begin{cases} \mathsf{tt} & \text{if } \sup x < 1/2, \\ \mathsf{ff} & \text{if } \inf x > 1/2, \\ \bot & \text{otherwise} \end{cases}$$

and the ***parallel conditional*** by

$$\text{pif } p \text{ then } x \text{ else } y = \begin{cases} x & \text{if } p = \mathsf{tt}, \\ y & \text{if } p = \mathsf{ff}, \\ x \sqcap y & \text{if } p = \bot. \end{cases}$$

Lemma 3.1 (Elementary axioms)

$$\begin{aligned} \mathrm{tail}_L(\mathrm{cons}_L(x)) &= x \\ \mathrm{tail}_L(\mathrm{cons}_R(y)) &= 1 \\ \mathrm{tail}_L(\mathrm{cons}_L(x) \sqcap \mathrm{cons}_R(y)) &= x \sqcap 1 \\ \mathrm{tail}_R(\mathrm{cons}_L(x)) &= 0 \\ \mathrm{tail}_R(\mathrm{cons}_R(y)) &= y \\ \mathrm{tail}_R(\mathrm{cons}_L(x) \sqcap \mathrm{cons}_R(y)) &= 0 \sqcap y \end{aligned}$$

$$\begin{aligned} \mathrm{head}(\mathrm{cons}_L(x)) &\sqsubseteq \mathsf{tt} \\ \mathrm{head}(\mathrm{cons}_L(x)) &= \bot \quad \textit{iff} \quad x \sqsubseteq 1 \\ \mathrm{head}(\mathrm{cons}_R(y)) &\sqsubseteq \mathsf{ff} \\ \mathrm{head}(\mathrm{cons}_R(y)) &= \bot \quad \textit{iff} \quad y \sqsubseteq 0 \end{aligned}$$

$$\begin{aligned} x = \ & \text{pif } \mathrm{head}(x) \\ & \text{then } \mathrm{cons}_L(\mathrm{tail}_L(x)) \text{ else } \mathrm{cons}_R(\mathrm{tail}_R(x)). \end{aligned}$$

Given a set X, an element $x \in X$ and a function $g : \mathbb{N} \to X$, there is a unique function $f : \mathbb{N} \to X$ such that $f(0) = x$ and $f(n+1) = g(n)$. A similar fact holds for $\mathcal{I}$ equipped with cons_L and cons_R, but we have to take into account the equation $\mathrm{cons}_L(1) = \mathrm{cons}_R(0)$.

Lemma 3.2 (Definition by cases) *Let D be a domain and $g_L, g_R : \mathcal{I} \to D$ be continuous maps such that*

$$g_L(1) = g_R(0).$$

Then there is a unique continuous map $f : \mathcal{I} \to D$ such that

$$\begin{array}{rcl} f(\mathrm{cons}_L(x)) & = & g_L(x) \\ f(\mathrm{cons}_R(y)) & = & g_R(y) \\ f(\mathrm{cons}_L(x) \sqcap \mathrm{cons}_R(y)) & = & g_L(x) \sqcap g_R(y) \\ & & \quad \textit{if } x \sqsubseteq 1 \textit{ and } y \sqsubseteq 0, \end{array}$$

namely the function f defined by

$$\begin{array}{rcl} f(x) & = & \mathrm{pif}\ \mathrm{head}(x) \\ & & \mathrm{then}\, g_L(\mathrm{tail}_L(x))\, \mathrm{else}\, g_R(\mathrm{tail}_R(x)). \end{array}$$

The natural numbers enjoy an induction principle, which can be expressed by saying that if a set of natural numbers contains zero and is closed under the successor operation, then it contains all natural numbers. A similar principle is enjoyed by the partial unit interval endowed with the operations cons_L and cons_R.

A subset of a domain D is called ***inductive*** if it closed under the formation of least upper bounds of directed subsets.

Lemma 3.3 (Dyadic induction) *Let $A \subseteq \mathcal{I}$ be inductive, and assume that the following conditions hold:*

1. *(Base case) $\bot \in A$.*
2. *(Inductive step) $x \in A$ and $y \in A$ imply*
   - *(a)* $\mathrm{cons}_L(x) \in A$,
   - *(b)* $\mathrm{cons}_R(y) \in A$,
   - *(c)* $\mathrm{cons}_L(x) \sqcap \mathrm{cons}_R(y) \in A$ *if $x \sqsubseteq 1$ and $y \sqsubseteq 0$.*

Then $A = \mathcal{I}$.

We can define functions on natural numbers by iteration. If X is a set, x is an element of X and $g : X \to X$ is a function, then there is a unique function $f : \mathbb{N} \to X$ such that $f(0) = x$ and $f(n+1) = g(f(n))$. A similar fact holds for the partial unit interval equipped with cons_L and cons_L. We first need a lemma:

Lemma 3.4 (Dyadic recursion) *Let D be a domain, $g_L, g_R : D \to D$ be continuous maps, and $f : \mathcal{I} \to D$ be any continuous solution to the functional equation*

$$\begin{array}{rcl} f(x) & = & \mathrm{pif}\ \mathrm{head}(x) \\ & & \mathrm{then}\, g_L(f(\mathrm{tail}_L(x)))\, \mathrm{else}\, g_R(f(\mathrm{tail}_R(x))). \end{array}$$

Then the following statements hold:

1. *$f(0)$, $f(1)$ and $f(\bot)$ are fixed points of g_L, g_R and $g_L \sqcap g_R$ respectively.*
2. *f is uniquely determined by the values that it assumes at 0, 1, and $\bot$.*
3. *f is the least solution iff $f(0) = \mathrm{fix}\ g_L$, $f(1) = \mathrm{fix}\ g_L$, and $f(\bot) = \mathrm{fix}\ (g_L \sqcap g_R)$.*

In particular, if g_L, g_R, and $g_L \sqcap g_R$ have unique fixed points then f is the unique solution.

Lemma 3.5 (Dyadic iteration) *Let D be a domain, and $g_L, g_R : D \to D$ be continuous maps such that*

$$g_L(\mathrm{fix}\ g_R) = g_R(\mathrm{fix}\ g_L).$$

Then there is a unique continuous map $f : \mathcal{I} \to D$ satisfying the equations

(Base case)

$$\begin{array}{rcl} f(0) & = & \mathrm{fix}\ g_L \\ f(1) & = & \mathrm{fix}\ g_R \\ f(\bot) & = & \mathrm{fix}\ (g_L \sqcap g_R) \end{array}$$

(Iteration step)

$$\begin{array}{rcl} f(\mathrm{cons}_L(x)) & = & g_L(f(x)) \\ f(\mathrm{cons}_R(y)) & = & g_R(f(y)) \\ f(\mathrm{cons}_R(x) \sqcap \mathrm{cons}_R(y)) & = & g_L(f(x)) \sqcap g_R(f(y)) \\ & & \quad \textit{if } x \sqsubseteq 1 \textit{ and } y \sqsubseteq 0, \end{array}$$

namely the least continuous solution to the equation of Lemma 3.4.

Finally, the set of natural numbers is uniquely specified, up to isomorphism, by the so called Peano axioms, which are essentially the properties that we informally considered above for the sake of motivation. This idea is made formal in e.g. Stoll [25], where *unary systems* are used as a tool (a unary system is a set X together with an element $x \in X$ and a function $s : X \to X$).

In the following definition, the domain D generalizes the partial unit interval and the maps a_L and a_R generalize the constructor maps cons_L and cons_R respectively.

Definition 3.6 A ***binary system*** is a domain D equipped with a pair of continuous maps $a_L, a_R : D \to D$ such that

$$a_L(1) = a_R(0),$$

where $0 \stackrel{\mathrm{def}}{=} \mathrm{fix}\ a_L$ and $1 \stackrel{\mathrm{def}}{=} \mathrm{fix}\ a_R$.

We also impose the technical condition $\mathrm{fix}\,(a_L \sqcap a_R) = \bot$, which ensures that homomorphisms defined below, as Lemma 3.5 suggests, make binary systems into a category under ordinary function composition.

A ***homomorphism*** from a binary system (D, a_L, a_R) to a binary system (E, b_L, b_R) is a continuous map $f : D \to E$ such that

$$\begin{aligned} f(0) &= 0 \\ f(1) &= 1 \\ f(\bot) &= \bot \\ f(a_L(x)) &= b_L(f(x)) \\ f(a_R(y)) &= b_R(f(y)) \\ f(a_L(x) \sqcap a_R(y)) &= b_L(f(x)) \sqcap b_R(f(y)) \\ &\quad \text{if } x \sqsubseteq 1 \text{ and } y \sqsubseteq 0. \end{aligned}$$ □

Binary systems were introduced and investigated in the context of uniform spaces in the extended abstract [5]. Lemma 3.5 can be formulated as

Theorem 3.7 *$(\mathcal{I}, \mathrm{cons}_L, \mathrm{cons}_R)$ is an initial object in the category of binary systems.*

Compare the following lemma to Lemmas 3.1 and 3.2, where c_L, c_R, h, t_L, t_R play the rôle of cons_L, cons_R, head, tail_L, and tail_R respectively:

Lemma 3.8 *Let $D = (D, c_L, c_R)$ be a binary system. Then there is at most one triple of continuous maps $h : D \to \mathcal{B}$ and $t_L, t_R : D \to D$ such that*

$$\begin{aligned} t_L(c_L(x)) &= x \\ t_L(c_R(y)) &= 1 \\ t_L(c_L(x) \sqcap c_R(y)) &= x \sqcap 1 \\ t_R(c_L(x)) &= 0 \\ t_R(c_R(y)) &= y \\ t_R(c_L(x) \sqcap c_R(y)) &= 0 \sqcap y \\ h(c_L(x)) &\sqsubseteq \mathsf{tt} \\ h(c_L(x)) &= \bot \quad \textit{iff} \quad x \sqsubseteq 1 \\ h(c_R(y)) &\sqsubseteq \mathsf{ff} \\ h(c_R(y)) &= \bot \quad \textit{iff} \quad y \sqsubseteq 0 \end{aligned}$$

$$x = \text{pif } h(x) \text{ then } c_L(t_L(x)) \text{ else } c_R(t_R(x)).$$

Definition 3.9 If such maps exist then they are called the ***destructors*** of (D, c_L, c_R). □

Definition 3.10 A binary system (D, c_L, c_R) satisfies the ***dyadic induction principle*** if for any inductive set $A \subseteq D$ the conditions

1. (Base case) $\bot \in A$,
2. (Inductive step) $x \in A$ and $y \in A$ imply
   (a) $c_L(x) \in A$,
   (b) $c_R(y) \in A$,
   (c) $c_L(x) \sqcap c_R(y) \in A$ if $x \sqsubseteq 1$ and $y \sqsubseteq 0$

together entail that $A = D$. □

Definition 3.11 A binary system (D, c_L, c_R) is ***inductive*** if

1. 0 and 1 are the unique fixed points of c_L and c_R.
2. $0 \neq 1$ and $0 \sqcap 1 = \bot$.
3. It has destructors.
4. It satisfies the dyadic induction principle. □

The following theorem axiomatically characterizes the binary system $(\mathcal{I}, \mathrm{cons}_L, \mathrm{cons}_R)$ up to isomorphism, without explicit reference to real numbers or intervals.

Theorem 3.12 *A binary system is inductive iff it is initial. In particular, any two inductive binary systems are isomorphic.*

4. Bifree algebras generalized

In this section we introduce a new technique for defining structural recursion schemes based on the theory of domain equations. The basic idea is to consider not a distinguished domain D such that $D \cong \mathbf{F}D$, but instead a domain D such that D is a retract of $\mathbf{F}D$ in a special way. We refer to such a domain D as an ***$\mathbf{F}$-inductive retract***. It turns out that $\mathbf{F}$-inductive retracts generalize special $\mathbf{F}$-invariant objects in the sense of Freyd [11, 12]. We therefore could have adopted the (rather long) terminology ***special semi-invariant object***.

4.1. Inductive retractions

In the remaining of this section $\mathbf{X}$ is any category and $\mathbf{F}$ is an endofunctor of $\mathbf{X}$.

We first recall a concept due to Freyd [12]:

Definition 4.1 A ***bifree* $\mathbf{F}$*-algebra*** is an initial algebra $i : \mathbf{F}C \to C$ such that its inverse $i^{-1} : C \to \mathbf{F}C$ is a final coalgebra. □

Definition 4.2 An $\mathbf{F}$***-inductive retraction*** is a pair of arrows

$$X \underset{\beta}{\overset{\alpha}{\leftrightarrows}} \mathbf{F}X$$

such that $f = \alpha \circ \mathbf{F}f \circ \beta$ iff $f = \mathrm{id}_X$. □

The right-to-left implication shows that

$$\alpha \circ \beta = \mathrm{id}_X$$

and hence X is a retract of $\mathbf{F}X$. Also, notice that if $\langle \alpha, \beta \rangle$ is an $\mathbf{F}$-inductive retraction in $\mathbf{X}$, then $\langle \beta, \alpha \rangle$ is an $\mathbf{F}$-inductive retraction in $\mathbf{X}^{\mathrm{op}}$.

The following proposition shows that inductive retractions generalize bifree algebras:

Proposition 4.3 *Let $\beta : X \leftrightarrows \mathbf{F}X : \alpha$ be an $\mathbf{F}$-inductive isomorphism. If $\mathbf{F}$ has a bifree algebra then it is isomorphic to α.*

Proof Let $i : \mathbf{F}C \to C$ be a bifree algebra, $r : i \to \alpha$ be the unique algebra homomorphism and $s : \beta \to i^{-1}$ be the unique coalgebra homomorphism. This means that $r \circ i = \alpha \circ \mathbf{F}r$ and $i^{-1} \circ s = \mathbf{F}s \circ \beta$. Hence

$$r \circ s = r \circ i \circ i^{-1} \circ s = \alpha \circ \mathbf{F}r \circ \mathbf{F}s \circ \beta = \alpha \circ \mathbf{F}(r \circ s) \circ \beta.$$

By inductivity, $r \circ s = \mathrm{id}_X$. Since $s = i \circ \mathbf{F}s \circ \beta$, we have that

$$s \circ r \circ i = i \circ \mathbf{F}s \circ \beta \circ \alpha \circ \mathbf{F}r = i \circ \mathbf{F}(s \circ r).$$

Hence $s \circ r : i \to i$ and therefore $s \circ r = \mathrm{id}_C$. □

In the remaining of this section, $i : \mathbf{F}C \to C$ is a bifree algebra and $\beta : X \leftrightarrows \mathbf{F}X : \alpha$ is an $\mathbf{F}$-inductive retraction.

The first part of the proof of Proposition 4.3 shows that every inductive retract is a retract of the bifree algebra, in a canonical way:

Lemma 4.4 *If $r : i \to \alpha$ and $s : \beta \to i^{-1}$ are the unique (co)algebra homomorphisms then $s : X \leftrightarrows C : r$ is a retraction with $r \circ s = \mathrm{id}_X$.*

4.2. Structural recursion

Proposition 4.5 *Let $r : i \to \alpha$, $s : \beta \to i^{-1}$, $e = s \circ r : C \to C$, $h : i \to a$, and $k : b \to i^{-1}$.*

1. *For any algebra $a : \mathbf{F}A \to A$, there is a homomorphism $f : \alpha \to a$ iff $h = h \circ e$, and in this case $f = h \circ s$.*

2. *For any coalgebra $b : B \to \mathbf{F}B$, there is a homomorphism $g : b \to \beta$ iff $k = e \circ k$, and in this case $g = r \circ k$.*

Proof (1): If $f : \alpha \to a$ then $f \circ r = h$ because $r : i \to \alpha$. Therefore $f = h \circ s$ and $h = h \circ s \circ r$. Conversely, if $f \circ r : i \to a$ then $f : \alpha \to a$ because

$$\begin{aligned} f \circ \alpha &= f \circ \alpha \circ \mathbf{F}(r \circ s) = f \circ \alpha \circ \mathbf{F}r \circ \mathbf{F}s \\ &= f \circ r \circ i \circ \mathbf{F}s = a \circ \mathbf{F}(f \circ r) \circ \mathbf{F}s \\ &= a \circ \mathbf{F}f \circ \mathbf{F}(r \circ s) = a \circ \mathbf{F}f. \end{aligned}$$

If $h \circ s \circ r = h$ then this holds in particular for $f = h \circ s$.
(2): Dual to (1). □

Roughly, condition (1) means that h respects the congruence on C induced by the idempotent $e = s \circ r$, and that f is the restriction of h to X via s. Dually, condition (2) means that the image of k is contained in image of e and that g is the corestriction of k to X via r.

Corollary 4.6

1. *For any algebra $a : \mathbf{F}A \to A$ there is at most one homomorphism $f : \alpha \to a$.*

2. *For any coalgebra $b : B \to \mathbf{F}B$ there is at most one homomorphism $g : b \to \beta$.*

Only for the last result of this subsection, we assume that our base category $\mathbf{X}$ is the category **SDom** of domains and strict continuous maps.

Proposition 4.7 *Let $\mathbf{F} : \mathbf{SDom} \to \mathbf{SDom}$ be locally continuous.*

1. *If there is a homomorphism $f : \alpha \to a$ for a given algebra $a : \mathbf{F}A \to A$ then it is the least $f' : X \to A$ such that $f' = a \circ \mathbf{F}f' \circ \beta$.*

2. *If there is a homomorphism $g : b \to \beta$ for a given coalgebra $b : B \to \mathbf{F}B$ then it is the least $g' : B \to X$ such that $g' = \alpha \circ \mathbf{F}g' \circ b$.*

Proof (1): The least solution of the above equation is $f' = \bigsqcup_n f_n$, where the sequence f_n is inductively defined by $f_0 = \bot$ and $f_{n+1} = a \circ \mathbf{F}f_n \circ \beta$. Define $\mathrm{id}_n : X \to X$ by $\mathrm{id}_0 = \bot$ and $\mathrm{id}_{n+1} = \alpha \circ \mathbf{F}\mathrm{id}_n \circ \beta$. By local continuity of $\mathbf{F}$,

$$\begin{aligned} \bigsqcup_n \mathrm{id}_n &= \bigsqcup_n \mathrm{id}_{n+1} = \bigsqcup_n (\alpha \circ \mathbf{F}\mathrm{id}_n \circ \beta) \\ &= \alpha \circ \mathbf{F}\left(\bigsqcup_n \mathrm{id}_n\right) \circ \beta. \end{aligned}$$

Hence $\bigsqcup_n \mathrm{id}_n = \mathrm{id}_X$ by inductivity. Since f is strict, we have that $f_0 = f \circ \mathrm{id}_0$. Assuming that $f_n = f \circ \mathrm{id}_n$ we deduce that

$$\begin{aligned} f_{n+1} &= a \circ \mathbf{F}f_n \circ \beta = a \circ \mathbf{F}f \circ \mathbf{F}\mathrm{id}_n \circ \beta \\ &= f \circ \alpha \circ \mathbf{F}\mathrm{id}_n \circ \beta = f \circ \mathrm{id}_{n+1}. \end{aligned}$$

Hence $f_n = f \circ \mathrm{id}_n$ for every n. Therefore

$$f' = \bigsqcup_n f_n = \bigsqcup_n (f \circ \mathrm{id}_n) = f \circ \bigsqcup_n \mathrm{id}_n = f \circ \mathrm{id}_X = f.$$

(2): Dual to (1). □

Thus, in order to find a recursive definition of a function $f : X \to A$ we can try to find an algebra a such that $f : \alpha \to a$ is a homomorphism, and in order to find a recursive definition of a function $g : B \to X$ we can try to find a coalgebra b such that $g : b \to \beta$ is a homomorphism. If we succeed in finding such algebra a and coalgebra b, then we obtain a definition of f by ***structural recursion*** and a definition of g by ***structural corecursion***.

4.3. Biquotients of bifree algebras

We have seen that any $\mathbf{F}$-inductive retraction $\beta : \mathbf{X} \leftrightarrows \mathbf{FX} : \alpha$ appears as a retract of the bifree $\mathbf{F}$-algebra $i : \mathbf{F}C \to C$ via $r : i \to \alpha$ and $s : \beta \to i^{-1}$ with $r \circ s = \mathrm{id}_X$. We now characterize for a bifree algebra $i : \mathbf{F}C \to C$ the idempotents $e : C \to C$ which admit a splitting $e = s \circ r$ of the kind just described. Recall that any idempotent in **SDom** splits through its image [1]. But notice that we are still working in an arbitrary category $\mathbf{X}$.

Definition 4.8 Let $e : C \to C$ be an idempotent and define

$$C \underset{b}{\overset{a}{\leftrightarrows}} \mathbf{F}C$$

by

$$a = e \circ i \qquad\qquad b = i^{-1} \circ e.$$

We say that e is a ***biquotient*** of the bifree algebra $i : \mathbf{F}C \to C$ if the following conditions hold:

(i) $e : i \to a$

(ii) $e : b \to i^{-1}$

(iii) $h = a \circ \mathbf{F}h \circ b$ iff $h = e$. □

Theorem 4.9

1. *If $\beta : X \rightleftarrows \mathbf{F}X : \alpha$ is an $\mathbf{F}$-inductive retraction, $r : i \to \alpha$ and $s : \beta \to i^{-1}$, then $e \overset{\text{def}}{=} s \circ r$ is a biquotient of i. Moreover, α and β can be recovered from r and s as*

$$\alpha = r \circ i \circ \mathbf{F}s \qquad \beta = \mathbf{F}r \circ i^{-1} \circ s.$$

2. *If $e : C \to C$ is a biquotient of i and $e = s \circ r$ with $r \circ s = id_X$, then the maps*

$$\begin{aligned} \alpha &\overset{\text{def}}{=} r \circ i \circ \mathbf{F}s : \mathbf{F}X \to X \\ \beta &\overset{\text{def}}{=} \mathbf{F}r \circ i^{-1} \circ s : X \to \mathbf{F}X \end{aligned}$$

constitute an $\mathbf{F}$-inductive retraction. Moreover, we have $r : i \to \alpha$ and $s : \beta \to i^{-1}$.

Proof (1): Conditions (i) and (ii) hold by the following equational reasoning:

$$\begin{aligned} e \circ i \circ \mathbf{F}e &= s \circ r \circ i \circ \mathbf{F}s \circ \mathbf{F}r \\ &= s \circ \alpha \circ \mathbf{F}r \circ \mathbf{F}s \circ \mathbf{F}r \\ &= s \circ \alpha \circ \mathbf{F}r = s \circ r \circ i = e \circ i, \\ \mathbf{F}e \circ i^{-1} \circ e &= \mathbf{F}s \circ \mathbf{F}r \circ i^{-1} \circ s \circ r \\ &= \mathbf{F}s \circ \mathbf{F}r \circ \mathbf{F}s \circ \beta \circ r \\ &= \mathbf{F}s \circ \beta \circ r = i^{-1} \circ s \circ r = i^{-1} \circ e. \end{aligned}$$

From this we get immediately that

$$e \circ i \circ \mathbf{F}e \circ i^{-1} \circ e = e \circ i \circ i^{-1} \circ e = e \circ e = e.$$

For the other implication of condition (iii), let $h : C \to C$ with $e \circ i \circ \mathbf{F}h \circ i^{-1} \circ e = h$. It follows that

$$r \circ i \circ \mathbf{F}h \circ i^{-1} \circ s = r \circ h \circ s.$$

Hence

$$\begin{aligned} r \circ h \circ s &= r \circ i \circ \mathbf{F}h \circ i^{-1} \circ s \\ &= \alpha \circ \mathbf{F}r \circ \mathbf{F}h \circ \mathbf{F}s \circ \beta \\ &= \alpha \circ \mathbf{F}(r \circ h \circ s) \circ \beta, \end{aligned}$$

which entails $r \circ h \circ s = \mathrm{id}_X$ as α and β form an $\mathbf{F}$-inductive retract. Thus we get

$$h = e \circ h \circ e = s \circ r \circ h \circ s \circ r = s \circ r = e.$$

The proposed reconstruction of α and β from r and s can be seen as follows:

$$\begin{aligned} r \circ i \circ \mathbf{F}s &= \alpha \circ \mathbf{F}r \circ \mathbf{F}s = \alpha, \\ \mathbf{F}r \circ i^{-1} \circ s &= \mathbf{F}r \circ \mathbf{F}s \circ \beta = \beta. \end{aligned}$$

(2): We have that

(a) $r \circ i = r \circ i \circ \mathbf{F}(s \circ r)$,
(b) $i^{-1} \circ s = \mathbf{F}(s \circ r) \circ i^{-1} \circ s$,
(c) $s \circ r \circ i \circ \mathbf{F}h \circ i^{-1} \circ s \circ r = h$ iff $h = e$,

and hence that

$$\begin{aligned} \alpha \circ \beta &= r \circ i \circ \mathbf{F}s \circ \mathbf{F}r \circ i^{-1} \circ s \\ &= r \circ i \circ \mathbf{F}(s \circ r) \circ i^{-1} \circ s \\ &= r \circ i \circ i^{-1} \circ s = r \circ s = \mathrm{id}_X. \end{aligned}$$

Let $f : X \to X$ with $f = \alpha \circ \mathbf{F}f \circ \beta$. As

$$\alpha \circ \mathbf{F}f \circ \beta = r \circ i \circ \mathbf{F}(s \circ f \circ r) \circ i^{-1} \circ s,$$

for $h \stackrel{\text{def}}{=} s \circ f \circ r$ we get

$$h = s \circ r \circ i \circ \mathbf{F}h \circ i^{-1} \circ s \circ r,$$

from which we get by (c) that $h = e$. But then

$$f = r \circ s \circ f \circ s \circ r = r \circ h \circ s = r \circ e \circ s = \mathrm{id}_X$$

as desired. Finally, $r : i \to \alpha$ and $s : \beta \to i^{-1}$ because

$$\begin{aligned} \alpha \circ \mathbf{F}r &= r \circ i \circ \mathbf{F}s \circ \mathbf{F}r = r \circ i \text{ by (a)}, \\ \mathbf{F}s \circ \beta &= \mathbf{F}s \circ \mathbf{F}r \circ i^{-1} \circ s = i^{-1} \circ s \text{ by (b)}. \end{aligned}$$ □

5. Structural recursion on the partial unit interval

In order to obtain an inductive retraction for the partial unit interval, we put the real number constructors (resp. destructors) together. Define $\mathbf{T} : \mathbf{SDom} \to \mathbf{SDom}$ by

$$\mathbf{T}D = \mathcal{B} \times D \times D,$$

and define $\mathcal{I} \underset{\text{destr}}{\overset{\text{cons}}{\leftrightarrows}} \mathbf{T}\mathcal{I}$ by

$$\begin{aligned} \text{cons} &= \text{pif} \circ (\mathrm{id} \times \text{cons}_L \times \text{cons}_R), \\ \text{destr} &= \langle \text{head}, \text{tail}_L, \text{tail}_R \rangle. \end{aligned}$$

Theorem 5.1 cons *and* destr *form a* $\mathbf{T}$*-inductive retraction.*

Proof The equation $f = \text{cons} \circ \mathbf{T}f \circ \text{destr}$ is equivalent to the equation

$$\begin{aligned} f(x) = \ & \text{pif head}(x) \\ & \text{then cons}_L(f(\text{tail}_L(x))) \\ & \text{else cons}_R(f(\text{tail}_R(x))). \end{aligned}$$

By Lemma 3.1, $f = \mathrm{id}$ is a solution. But cons_L and cons_R and $\text{cons}_R \sqcap \text{cons}_L$ have unique fixed-points. Therefore $f = \mathrm{id}$ is the unique solution by virtue of Lemma 3.4. □

5.1. Binary T-algebras

We now relate the algebra $\text{cons} : \mathbf{T}\mathcal{I} \to \mathcal{I}$ to the binary system $(\mathcal{I}, \text{cons}_L, \text{cons}_R)$.

Definition 5.2 A ***binary* T*-algebra*** is a $\mathbf{T}$-algebra $a : \mathbf{T}D \to D$ of the form

$$\text{pif} \circ (\mathrm{id} \times a_L \times a_R)$$

for (necessarily unique) $a_L, a_R : D \to D$. □

Compare the following proposition to Lemma 3.5 and Definition 3.6:

Proposition 5.3 *Let* $a : \mathbf{T}D \to D$ *be a binary* $\mathbf{T}$*-algebra. Then a strict continuous map* $f : \mathcal{I} \to D$ *is a homomorphism from* cons *to a iff*

$$\begin{aligned} f(\text{cons}_L(x)) &= a_L(f(x)) \\ f(\text{cons}_R(y)) &= a_R(f(y)) \\ f(\text{cons}_R(x) \sqcap \text{cons}_R(y)) &= a_L(f(x)) \sqcap a_R(f(y)). \end{aligned}$$

Compare the following proposition to Lemmas 3.4 and 3.5:

Proposition 5.4 *If there is a homomorphism from* cons *to a binary* $\mathbf{T}$*-algebra* $a : \mathbf{T}D \to D$ *then it is the least continuous map* $f : \mathcal{I} \to D$ *such that*

$$\begin{aligned} f(x) = \ & \text{pif head}(x) \\ & \text{then } a_L(f(\text{tail}_L(x))) \text{ else } a_R(f(\text{tail}_R(x)). \end{aligned}$$

Proof By Proposition 4.7 we know that if there is a homomorphism from cons to a, then it is the least continuous function f such that

$$f = a \circ \mathbf{T}f \circ \text{destr},$$

which is equivalent to the above equation. □

5.2. Bifurcated binary expansions

The canonical solution of the domain equation $D \cong \mathbf{T}D$ is the domain $\mathcal{B}\mathrm{Tree}$ of infinite binary trees with nodes labeled by truth values, ordered nodewise, together with the bifree algebra

$$\mathrm{mktree} : \mathbf{T}\mathcal{B}\mathrm{Tree} \to \mathcal{B}\mathrm{Tree}$$

which maps a list $\langle p, s, t\rangle$ to the tree with root labeled by p and with left and right subtrees s and t respectively [18]. Let

$$\begin{array}{rcccl} \mathrm{num} & : & \mathrm{mktree} \to \mathrm{cons} & : & \mathcal{B}\mathrm{Tree} \to \mathcal{I} \\ \mathrm{bin} & : & \mathrm{destr} \to \mathrm{mktree}^{-1} & : & \mathcal{I} \to \mathcal{B}\mathrm{Tree} \end{array}$$

be the unique (co)algebra homomorphisms. By Lemma 4.4, $\mathrm{num} \circ \mathrm{bin} = \mathrm{id}$, so that $\mathcal{I}$ is a retract of $\mathcal{B}\mathrm{Tree}$. The tree $\mathrm{bin}\,(x)$ is referred to as the ***bifurcated binary expansion*** of the partial number x in [7].

5.3. Coinduction

Dana Scott suggested that we should also consider a characterization of the partial unit interval via "co-Peano axioms" based on coinduction and coiteration. Although we don't have such a characterization yet, we have a coinduction principle related to the ideas of Smyth [23] and Fiore [10].

Definition 5.5 A ***bisimulation*** on the partial unit interval is a binary relation $\sim \subseteq \mathcal{I} \times \mathcal{I}$ such that

$$\begin{array}{l} x \sim y \text{ implies that } \mathrm{head}(x) = \mathrm{head}(y) \text{ and} \\ \mathrm{tail}_a(x) \sim \mathrm{tail}_a(y) \text{ for } a \in \{R, L\}. \end{array}$$

We say that x and y are ***bisimilar*** if they are related by some bisimulation. □

Theorem 5.6 (Coinduction) *If $x, y \in \mathcal{I}$ are bisimilar then $x = y$.*

Proof Let x and y be bisimilar partial numbers. Then $\mathrm{bin}\,(x)$ and $\mathrm{bin}\,(y)$ are bisimilar trees. Hence $\mathrm{bin}\,(x) = \mathrm{bin}\,(y)$. Therefore $x = y$ because bin is split mono. □

5.4. Examples

Proposition 5.7 *The complement map* $\mathrm{compl} : \mathcal{I} \to \mathcal{I}$ *defined by* $\mathrm{compl}(x) = 1 - x$ *can be recursively defined by*

$$\begin{array}{rcl} \mathrm{compl}(x) & = & \mathrm{pif}\ \mathrm{head}(x) \\ & & \mathrm{then}\ \mathrm{cons}_R(\mathrm{compl}(\mathrm{tail}_L(x))) \\ & & \mathrm{else}\ \ \mathrm{cons}_L(\mathrm{compl}(\mathrm{tail}_R(x))). \end{array}$$

Proof This follows from Proposition 5.4, because compl is an algebra homomorphism from cons to $\mathrm{pif} \circ (\mathrm{id} \times \mathrm{cons}_R \times \mathrm{cons}_L)$. □

Proposition 5.8 *The map* $\exp : \mathbf{I}[0,1] \to \mathbf{I}[1,2]$ *defined by* $\exp(x) = 2^x$ *can be recursively defined by*

$$\begin{array}{rcl} \exp(x) & = & \mathrm{pif}\ \mathrm{head}(x) \\ & & \mathrm{then}\ \sqrt{\exp(\mathrm{tail}_L(x))} \\ & & \mathrm{else}\ \ \sqrt{2\exp(\mathrm{tail}_L(x))}. \end{array}$$

Proof Define $a_L, a_R : \mathbf{I}[1,2] \to \mathbf{I}[1,2]$ by $a_L(x) = \sqrt{x}$ and $a_R(x) = \sqrt{2x}$. Then exp is an algebra homomorphism from cons to $\mathrm{pif} \circ (\mathrm{id} \times a_L \times a_R)$, and the result again follows from Proposition 5.4. □

More examples can be found in [6, 7], and a recursive definition of Riemann integration can be found in [3].

6. Applications to Real PCF

Real PCF [6] is an extension of PCF [21, 16] with a ground type for the partial real line and some primitive functions, which include the real number constructors and destructors discussed in Section 3. The remaining primitives are necessary to obtain the operational semantics of Real PCF, but they are not needed to obtain the results discussed below, so that we don't pause to introduce them. For simplicity, we only discuss the partial unit interval. A treatment of the partial real line can be found in [7].

Definition 6.1 A programming language $\mathcal{L}$ is ***universal*** if every computable element of the universe of discourse of $\mathcal{L}$ is definable in $\mathcal{L}$. □

This depends on a notion of computability in the universe of discourse. In domain theory this is achieved via the notion of effective presentation (see [4, 18] for the algebraic case and [22] for the general continuous case).

Before tackling Real PCF, we recall some basic facts about PCF proved by Plotkin [16]. It is easy too see that all partial recursive functions [19] are PCF definable. However, simple computable functions such as the parallel conditional and an existential quantifier $\exists : [\mathcal{N} \to \mathcal{B}] \to \mathcal{B}$ fail to be PCF definable. Plotkin showed that if we extend PCF with the parallel conditional then all computable first order-functions become definable, and that if we further extend PCF with the existential quantifier then all computable functions of all orders become definable.

Streicher [26] generalized this result to an extension of PCF with recursive types, parallel-or and $\exists$ [18], and

Escardó [8] generalized it to Real PCF. We now briefly consider Real PCF extended with recursive types.

It is straightforward to show that there exists an effective presentation of the partial real line which makes the primitive constructors and destructors computable; for example, any standard enumeration of the rational basis gives such an effective presentation. Let's fix any such effective presentation and call it the ***standard effective presentation***. One then may wonder if a cleverer choice of an effective presentation would change the induced set of computable elements and functions, and this is indeed the case in general [13]. We show in Section 6.1 below that this is *not* the case in our application.

6.1. Absoluteness of the standard effective presentation

Definition 6.2 Two effective presentations b and b' of a domain D are ***equivalent*** if $\mathrm{id}_D : D \to D$ is computable both as a map $(D, b) \to (D, b')$ and as map $(D, b') \to (D, b)$. □

Notice that this is the notion of equivalence of objects in concrete categories discussed in [2], specialized to the category of effectively given domains and computable maps considered as concrete over the category of domains and continuous functions, via the forgetful functor which forgets effective presentations.

Theorem 6.3 *Any two effective presentations of $\mathcal{I}$ which make* cons $: \mathbf{T}\mathcal{I} \to \mathcal{I}$ *and* destr $: \mathcal{I} \to \mathbf{T}\mathcal{I}$ *computable are equivalent.*

Proof Let b' and b'' be two such effective presentations, and let $\mathcal{I}'$ and $\mathcal{I}''$ denote $\mathcal{I}$ endowed with b' and b''. By Corollary 4.6, $\mathrm{id}_{\mathcal{I}} : \mathcal{I} \to \mathcal{I}$ is the unique algebra homomorphism cons $\to$ cons, and by Proposition 4.7, $\mathrm{id}_{\mathcal{I}}$ is the least fixed point of functional

$$F : [\mathcal{I} \to \mathcal{I}] \to [\mathcal{I} \to \mathcal{I}]$$

defined by

$$F(f) = \mathrm{cons} \circ \mathbf{T}f \circ \mathrm{destr}.$$

By hypothesis, cons is computable both as as map $(1')$ $\mathbf{T}\mathcal{I}' \to \mathcal{I}'$ and as a map $(1'')$ $\mathbf{T}\mathcal{I}'' \to \mathcal{I}''$, and destr is computable both as a map $(2')$ $\mathcal{I}' \to \mathbf{T}\mathcal{I}'$ and as a map $(2'')$ $\mathbf{T}\mathcal{I}'' \to \mathcal{I}''$. By $(1'')$ and $(2')$ we conclude that F is computable as a map $[\mathcal{I}' \to \mathcal{I}''] \to [\mathcal{I}' \to \mathcal{I}'']$, which shows that $\mathrm{id}_{\mathcal{I}}$ is computable as a map $\mathcal{I}' \to \mathcal{I}''$. Similarly, by $(1')$ and $(2'')$ we conclude that it is also computable as a map $\mathcal{I}'' \to \mathcal{I}'$. □

6.2. Universality of Real PCF

We prove that Real PCF extended recursive types and $\exists$ is computationally complete by means of the technique introduced in [26]. Here are the main steps of the technique:

1. Take a universal domain $\mathcal{U}$ of PCF, for example $[\mathcal{N} \to \mathcal{B}]$ (see [17]).

2. Show that for every domain D in the extended language there is a definable retraction

$$D \underset{s_D}{\overset{r_D}{\leftrightarrows}} \mathcal{U}$$

with $r_D \circ s_D = \mathrm{id}_D$.

3. Given $d \in D$ computable, $s_D(d) \in \mathcal{U}$ is computable because s_D is computable.

4. Since PCF extended with parallel-or and $\exists$ is universal and $\mathcal{U}$ is a PCF domain, $s_D(d)$ is definable.

5. Hence d is definable as $d = r_D(s_D(d))$, and r_D and $s_D(d)$ are definable.

6. Therefore every computable element is definable.

The crucial step consists in showing that D is a definable retract of $\mathcal{U}$, and this is not so simple in the presence of recursive types. But by the general results of [26], it suffices to show that every ground type is a definable retract of $\mathcal{U}$. This has been done for the PCF ground types, so that we only need to do it for our new ground type.

Theorem 6.4 *Real PCF extended with recursive types, the parallel conditional and $\exists$ is a universal programming laguage.*

Proof By Section 5.2, we know that

$$\begin{array}{rcccl} \mathrm{num} & : & \mathrm{mktree} \to \mathrm{cons} & : & \mathcal{B}\mathrm{Tree} \to \mathcal{I} \\ \mathrm{bin} & : & \mathrm{destr} \to \mathrm{mktree}^{-1} & : & \mathcal{I} \to \mathcal{B}\mathrm{Tree} \end{array}$$

form a retraction with $\mathrm{num} \circ \mathrm{bin} = \mathrm{id}_{\mathcal{I}}$. Since $\mathcal{B}$Tree is a recursive type, and since num and bin are definable, we see that $\mathcal{I}$ is a definable retract of $\mathcal{B}$Tree. But we already know that $\mathcal{B}$Tree is a definable retract of $\mathcal{U}$. Since definable retracts compose, $\mathcal{I}$ is a definable retract of $\mathcal{U}$. □

This general result does not tell the full story about definability of computable first order functions (over the partial real interval $\mathcal{I}$). By means of a more direct method of proof similar to that of [16], in [7] it is shown that the existential quantifier is not needed to obtain the definability result at first-order types.

7. Conclusions

We have studied a notion of the most well-behaved quotients of bifree algebras, namely the so-called biquotients. We have applied this notion exclusively to the study of the partial real line and its recursion and induction principles.

It might be worthwhile to look at other applications of this notion. Typically one would like to have a characterisation of those equational theories $\mathcal{E}$ over a signature Σ such that the quotient of $T_\infty(\Sigma)$ by $\mathcal{E}$ is a biquotient.

This might be interesting especially for the case of stream domains extending the work on partially commutative monoids in trace theory towards infinite behaviours.

8. Acknowledgements

The first author was supported by the Brazilian agency CNPq, an EPSRC project "Programming Languages for Real Number Computation: Theory and Implementation", and an ARC project "A Computational Approach to Measure and Integration Theory".

References

[1] S. Abramsky and A. Jung. Domain theory. In S. Abramsky, D.M. Gabbay, and T.S.E Maibaum, editors, *Handbook of Logic in Computer Science*, volume 3, pages 1–168. Clarendon Press, Oxford, 1994.

[2] J Adamek, H Herrlich, and G.E. Strecker. *Abstract and Concrete Categories*. John Wiley & Sons, Inc., 1990.

[3] A. Edalat and M.H. Escardó. Integration in Real PCF (extended abstract). In *Proceedings of the Eleventh Annual IEEE Symposium on Logic In Computer Science*, New Brunswick, New Jersey, USA, July 1996.

[4] H. Egli and R.L. Constable. Computability concepts for programming languages. *Theoretical Computer Science*, 2:133–145, 1976.

[5] M.H. Escardó. Induction and recursion on the real line. In C. Hankin, I. Mackie, and R. Nagarajan, editors, *Theory and Formal Methods 1994: Proceedings of the Second Imperial College Department of Computing Workshop on Theory and Formal Methods*, pages 259–282, Møller Centre, Cambridge, 11–14 September 1994. IC Press. 1995.

[6] M.H. Escardó. PCF extended with real numbers. *Theoretical Computer Science*, 162(1):79–115, August 1996.

[7] M.H. Escardó. *PCF extended with real numbers: A domain-theoretic approach to higher-order exact real number computation.* PhD thesis, Imperial College, Department of Computing, November 1996.

[8] M.H. Escardó. Real PCF extended with $\exists$ is universal. In A. Edalat, S. Jourdan, and G. McCusker, editors, *Advances in Theory and Formal Methods of Computing: Proceedings of the Third Imperial College Workshop, April 1996*, pages 13–24, Christ Church, Oxford, 1996. IC Press.

[9] M.H. Escardó and D.M. Claudio. Scott domain theory as a foundation for interval analysis. Technical Report 218, UFRGS/II, Porto Alegre, Brazil, 1993.

[10] M.P. Fiore. A coinduction principle for recursive data types based on bisimulation. *Information and Computation*, 127(2):186–198, 1996.

[11] P. J. Freyd. Algebraically complete categories. In A. Carboni et al., editors, *Proc. 1990 Como Category Theory Conference*, pages 95–104, Berlin, 1991. Springer-Verlag. Lecture Notes in Mathematics Vol. 1488.

[12] P. J. Freyd. Remarks on algebraically compact categories. In M. P. Fourman, P. T. Johnstone, and A. M. Pitts, editors, *Applications of Categories in Computer Science: Proceedings of the LMS Symposium, Durham, 1991.* Cambridge University Press, 1992. LMS Lecture Notes Series, 177.

[13] A. Kanda and D. Park. When are two effectively given domains identical? In K. Weihrauch, editor, *Theoretical Computer Science* 4^{th} *GI Conference*, LNCS, 1979.

[14] D.J. Lehmann and M.B. Smyth. Algebraic specification of data types: a synthetic approach. *Math. Syst. Theory*, 14:97–139, 1981.

[15] R.E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, 1966.

[16] G. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5(1):223–255, 1977.

[17] G. Plotkin. $\mathcal{B}^\omega$ as a universal domain. *Journal of Computer and System Sciences*, 17:209–236, 1978.

[18] G. Plotkin. Domains. Post-graduate Lectures in advanced domain theory, University of Edinburgh, Department of Computer Science. http:// ida.dcs.qmw.ac.uk/ sites/ other/ domain.notes.other, 1980.

[19] H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967.

[20] D. S. Scott. Lattice theory, data types and semantics. In *Formal semantics of programming languages*, pages 66–106, Englewood Cliffs, 1972. Prentice-Hall.

[21] D. S. Scott. A type-theoretical alternative to CUCH, ISWIM and OWHY. *Theoretical Computer Science*, 121:411–440, 1993. Reprint of a manuscript produced in 1969.

[22] M.B. Smyth. Effectively given domains. *Theoretical Computer Science*, 5(1):256–274, 1977.

[23] M.B. Smyth. I-categories and duality. In M.P. Fourman, P.T. Johnstone, and Pitts A.M., editors, *Applications of Categories in Computer Science*, pages 270–287, Cambridge, 1992. Cambridge University Press. London Mathematical Society Lecture Notes Series 177.

[24] M.B. Smyth and G. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM Journal of Computing*, 11(4):761–783, 1982.

[25] R. Stoll. *Set Theory and Logic*. W.H. Freeman and Company, San Fransisco, 1966.

[26] T. Streicher. A universality theorem for PCF with recursive types, parallel-or and $\exists$. *Mathematical Structures for Computing Science*, 4(1):111 – 115, 1994.