Theory Comput. Systems OF1–OF57 (2006)
DOI: 10.1007/s00224-006-1339-2

Theory of Computing Systems

SHRAD: A Language for Sequential Real Number Computation

Amin Farjudian

Department of Mathematical Sciences, Sharif University of Technology,
Azadi St., Tehran, Iran
amin_farjudian@sina.sharif.ir

Abstract. Since Di Gianantonio [1993] introduced his semantics for exact real number computation, there has always been a struggle to maintain data abstraction and efficiency as much as possible. The interval domain model—or its variations—can be regarded as the standard setting to obtain maximum data abstraction. As for efficiency there has been much focus on sequentiality to the extent that these two terms have become almost synonymous. Escardó et al. [1998, 2004] demonstrated that there is not much one can get by sequential computation in the interval domain model. In Farjudian [2004a, 2003] we reinforced this result by exposing the limited power of (some extensions of) the sequential fragment of Real-PCF.

The previous argument suggests some sort of compromise in the beauty of the model in order to keep efficiency. One way forward is to try to sacrifice *single-valuedness over partial real numbers*. This is exactly what we will see in designing SHRAD,[1] where we succeed in presenting a framework for exact real number computation which satisfies the following all at the same time:

1. It is sequential.
2. Multi-valuedness over total real numbers is carefully avoided.
3. All the computable first-order functions are defined in the language (expressivity).

1. Comparison

Although exact real number computation has not yet found its way to our pocket calculators, there is a fairly rich literature available on the subject containing various approaches.

[1] Which originally comes from Farjudian [2004b].

Here we are about to make an attempt to find a middle-ground between *efficiency* and *data-abstraction*, thereby presenting a new framework in which major problems with previous works are avoided. This in turn necessitates an overview of the literature, with the intention of highlighting how our framework differs from all the previous ones. Thus we hand-pick a number of previous frameworks and briefly go through their descriptions. Again we emphasize that the literature is too vast to let our modest overview get anywhere near being comprehensive.

Böhm et al. [1986] present the implementations of the four basic arithmetic operators—addition, subtraction, multiplication and division—based on the representation of real numbers via redundant sequences of b-adic numbers. Ménissier-Morain [1996] takes this matter further and extends the implementations to more complicated functions. Böhm and Cartwright's work is regarded as seminal though in retrospect the lack of any kind of semantics makes it extremely hard to verify the correctness of any of the algorithms presented in Ménissier-Morain [1996].

Di Gianantonio [1993] was the first to introduce a framework for exact real number computation which included a representation-independent denotational semantics versus a representation-dependent operational one [Di Gianantonio, 1993, 1999]. Yet the framework suffers from the following deficiencies:

1. Existence of parallel operators.
2. Multi-valuedness is allowed in the language.
3. The domain which models the data type for real numbers—as it is—cannot have the real line as its space of maximal elements.

Real-PCF was put forward by Escardó [1997] who used interval domain to model the data type of real numbers. Though we do not discuss this approach in any detail here, it is useful to mention how Real-PCF solved the problem with Di Gianantonio's approach. The interval domain is a continuous domain which contains the real line as its subspace of maximal elements. *Extensionality* is kept over both partial and total real numbers. In fact this has made Real-PCF unique regarding data abstraction. Universality at first-order is already given by the language while adding a constant for the *existential quantifier* results in universality at all types [Escardó, 1996]. Yet the presence of parallel operators makes it extremely impractical to use Real-PCF even for computations involving basic operations over real numbers.

LFT Approach was introduced by Edalat and Potts [2000] and incorporated the (familiar) idea of representing real numbers as shrinking sequences of rational intervals. Later Edalat and Heckmann [2002] amalgamated signed-digit binary representation of real numbers and the LFT implementation of operations over them into a unified framework. Over non-negative real numbers this rather elegant approach maintains the single-valuedness property [Potts et al., 1997], though once extended over to the whole real line the presence of *redundant-if* allows multi-valuedness [Edalat et al., 1998]. However, once again the major problem lies in the inefficiency of the framework, this time regarding *space*. This matter has been studied from certain angles by Heckmann [1998, 2000a] who has also carried out further investigation into the LFT approach [Heck-

mann, 2000b, 2001, 2002]. Looking at the issue from another angle, one can take the definition of the *tangent* function as presented in Edalat et al. [1998, Section 3.3]. It is obvious that the input stream has to be kept in its entirety throughout the computation process as even the first element is not consumed at any stage. Adding to the previous analyses—bearing in mind the limits we face in practice regarding space—Konečný [2000, 2002] proved that any function computable by a finite transducer using LFT-representation is essentially affine.

Real-RAM: Müller [2000] implemented basic arithmetic operations together with the limit operation over real numbers inside C++. This leads to a relatively fast implementation which is inhabited side-by-side with *most* other imperative features of the language C++, but *not all*. This framework is neither single-valued nor meant to be so. In fact it closely follows the ideas of Brattka [1996] where corresponding to μ-recursive *functions* over natural numbers, a class of recursive *relations* over real numbers have been introduced that can be generated from a class of basic recursive relations—e.g. addition, multiplication, limit, etc. However, no formal relation between the two frameworks has been presented yet.

Marcial-Romero and Escardó [2004] present a framework the syntax of which is essentially the same as Real-PCF, with ordinary sequential *if* and *rtest*[2] replacing *pif* and *head*, respectively.[3] This way the inefficiency of the parallel-if present in Real-PCF is avoided at the cost of losing single-valuedness over *both* partial and total real numbers. Hence the original interval domain model is of no help anymore and a special category of *power domains* is employed as the model.

Our Approach. We present the implementation of an **abstract data type** for real numbers in a **sequential** setting, which is at least **expressive enough** to give us all computable[4] first-order functions on real numbers. Moreover, we **syntactically prevent multi-valuedness** over *total* real numbers. These properties cannot be found *all at the same time* in any of the previous works available in the literature.

Of course, Escardó et al. [1998, 2004] demonstrated that sequentiality and extensionality are impossible to co-exist in a framework based on the interval domain or any of its variants. Thus we choose to relax extensionality over *partial* real numbers in order to maintain sequentiality. However, preventing multi-valuedness over total real numbers syntactically is an achievement which we put more emphasis on. These terms will be explained in detail in due course.

2. The Syntax of SHRAD

Definition 2.1 (Types of SHRAD). The set $\mathbb{T}_{\mathcal{SH}}$ is defined by the following grammar:

$$\sigma ::= \mathit{nat} \mid \mathit{bool} \mid r \mid \sigma \to \sigma .$$

[2] Redundant-test.
[3] See *redundant-if* construct studied by Edalat et al. [1998].
[4] For example, in the TTE sense [Weihrauch, 2000].

Definition 2.2 (SHRAD). The language SHRAD[5] is the extension of PCF with a ground type r together with the set $\mathcal{P}_{SH}$ of the following constants:

1. $L\colon r \to r$, $M\colon r \to r$, $R\colon r \to r$.
2. $L^{-1}\colon r \to r$, $M^{-1}\colon r \to r$, $R^{-1}\colon r \to r$.
3. $qtoR\colon nat \to nat \to nat \to r$.
4. $avg\colon r \to r \to r$.
5. $mult\colon r \to r \to r$.
6. $abs\colon r \to r$.
7. $isNeg\colon r \to bool$.
8. $limit\colon (nat \to r) \to r$.
9. $Y_\sigma\colon (\sigma \to \sigma) \to \sigma$ (for each SHRAD-type $\sigma \in \mathbb{T}_{\mathcal{SH}}$).

3. Denotational Semantics

The PCF fragment of SHRAD is interpreted in the standard fashion as done by Plotkin [1977]. As regards the type r we need to clarify what our objectives are. The constants L, M and R are going to be used as *digits*, and following tradition we aim for reducing programs of type r to sequences of digits. Hence we reserve symbol $\mathcal{D}$ and define:

Definition 3.1 (The Set $\mathcal{D}$ of Digits).

$$\mathcal{D} := \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}.$$

Notation 3.2 (Important). In Definition 3.1 we have already made a not-so-bold typographical distinction between the syntactic entities L, M, R—written in ***italic***—and their semantic counterparts L, M, R—written in **sans serif** font. We maintain this distinction throughout the document. Thus, the denotation of the constant *avg* will be written as avg, and so on.

Definition 3.3 (X^*, X^ω, X^∞). Let X be an arbitrary set:

1. By X^* we mean the set of all *finite* sequences over X.
2. By X^ω we mean the set of all *infinite* sequences over X.
3. By X^∞ we mean the union of the previous two, i.e.

$$X^\infty := X^* \cup X^\omega.$$

[5] It might seem helpful right here to mention that:

- **S** stands for ***S**equential*.
- **H** stands for ***H**igher order*, due to the presence of the second-order constant *limit*.
- **RAD** stands for ***R**eal numbers as an **A**bstract **D**ata type*.

Definition 3.4 ($\mathcal{D}^\infty$). $(\mathcal{D}^\infty, \sqsubseteq_{\mathcal{D}^\infty})$ is the algebraic domain of finite and infinite sequences over the alphabet $\mathcal{D}$, i.e.

$$\mathcal{D}^\infty := \mathcal{D}^* \cup \mathcal{D}^\omega$$

partially ordered by

$$\forall x, y \in \mathcal{D}^\infty, \qquad x \sqsubseteq_{\mathcal{D}^\infty} y \quad \Longleftrightarrow \quad x \text{ is an initial segment of } y,$$

which we simply write as $\mathcal{D}^\infty$.

The flow of material already reveals that we have the *signed-digit binary* representation in the back of our minds. In fact, to bring it to the surface, let us define an interval domain $\mathcal{J}$ which is in all aspects isomorphic to the unit interval domain $\mathcal{I}$ except that it is built up over $[-1, 1]$:

Definition 3.5 (The Interval Domain $\mathcal{J}$). The interval domain $\mathcal{J}$ is the cpo $(\mathcal{J}, \sqsubseteq)$ defined by:

- $\mathcal{J} = \{[r, s] \mid r, s \in \mathbb{R}, -1 \leq r \leq s \leq 1\}$.
- $\forall a, b \in \mathcal{I}$: $a \sqsubseteq b \Longleftrightarrow a \supseteq b$.

Note 3.6. *In particular, we can have functions*

1. $\mathsf{Cons}_a, \mathsf{Tail}_a \colon \mathcal{J} \to \mathcal{J}$,
2. $\mathsf{head}_r \colon \mathcal{J} \to \mathbb{B}_\perp$,

where a ranges over the set:

- $\{[p, q] \mid -1 \leq p \leq q \leq 1\}$, *as a subscript to* Cons,
- $\{[p, q] \mid -1 \leq p < q \leq 1\}$, *as a subscript to* Tail,

and r over that of $(-1, 1)$.

Note 3.7. *Consider the embedding e: $[-1, 1] \hookrightarrow \mathcal{J}$ defined by*

$$\forall x \in [-1, 1], \qquad e(x) = [x, x].$$

We freely talk about "the maximal element $x \in \mathcal{J}$" instead of $[x, x]$, unless the difference happens to be too crucial to ignore.

It is straightforward to interpret elements of $\mathcal{D}^\infty$ inside $\mathcal{J}$ using the following recursive definition:

Definition 3.8 ($to_{\mathcal{J}}\colon \mathcal{D}^\infty \to \mathcal{J}$).

$$to_{\mathcal{J}}\ ([\,]) = [-1, 1],$$
$$to_{\mathcal{J}}\ (\mathsf{L} : x) = \mathsf{Cons}_{[-1,0]}\ (to_{\mathcal{J}}\ x),$$
$$to_{\mathcal{J}}\ (\mathsf{M} : x) = \mathsf{Cons}_{[-1/2,1/2]}\ (to_{\mathcal{J}}\ x),$$
$$to_{\mathcal{J}}\ (\mathsf{R} : x) = \mathsf{Cons}_{[0,1]}\ (to_{\mathcal{J}}\ x).$$

An element $d \in \mathcal{D}^\infty$ is said to *represent*—be a *representation* of—an interval $a \in \mathcal{J}$ iff $to_{\mathcal{J}}(d) = a$. In particular, following the Note 3.7 above, we say that the infinite sequence $d_x \in \mathcal{D}^\omega$ is a *representation* of the real number $x \in [-1, 1]$, iff

$$to_{\mathcal{J}}(d_x) = [x, x] = e(x).$$

Plotkin's terminology (see Plotkin [1977]) is adopted as follows:

Definition 3.9. The collection $\mathbb{D}$ of domains for SHRAD is the set

$$\mathbb{D} := \{\mathbb{D}_\sigma \mid \sigma \text{ a SHRAD-type}\}$$

built recursively by:

- $\mathbb{D}_{bool} = \mathbb{B}_\perp$, $\mathbb{D}_{nat} = \mathbb{N}_\perp$, $\mathbb{D}_r = \mathcal{D}^\infty$,
- $\mathbb{D}_{\sigma\to\tau} = [\mathbb{D}_\sigma \to \mathbb{D}_\tau]$, i.e. the domain-theoretic function space.

Definition 3.10 (Interpreting Types). The interpretation of any type σ—written as $[\![\sigma]\!]$—is $\mathbb{D}_\sigma$, i.e.

$$[\![\sigma]\!] := \mathbb{D}_\sigma.$$

Definition 3.11 (Interpreting Constants). We interpret the elements of $\mathcal{P}_{SH}$ via the function

$$\mathcal{A}_{\mathcal{SH}}\colon \mathcal{P}_{SH} \to \cup\{\mathbb{D}_\sigma \mid \sigma \text{ a SHRAD-type}\},$$

which is type-respecting, i.e.

$$\forall c \in \mathcal{P}_{SH}, \qquad c\colon \sigma \ \Rightarrow \mathcal{A}_{\mathcal{SH}}(c) \in \mathbb{D}_\sigma.$$

The fix-point constants are dealt with as usual:

$$\forall f \in \mathbb{D}_{\sigma\to\sigma}, \qquad \mathcal{A}_{\mathcal{SH}}(Y_\sigma)(f) = \sqcup\{f^n(\perp) \mid n \geq 0\}.$$

For other constants $c \in \mathcal{P}_{SH}$, Table 1 shows where the reader can find the exact definition of $\mathcal{A}_{\mathcal{SH}}(c)$ together with a discussion of its correctness. In order to simplify the definition

Table 1. Where to find the interpretation of SHRAD constants.

Constant	Implemented as	Definition
L, M, R	$\mathsf{Cons_{L,M,R}}$	4.4
L^{-1}, M^{-1}, R^{-1}	$\mathsf{L^{-1}}$, $\mathsf{M^{-1}}$, $\mathsf{R^{-1}}$	4.16
qtoR	qtoR	4.22
avg	avg	4.28
mult	mult	4.37
abs	abs	4.45
isNeg	isNeg	4.51
limit	limit	4.68

of $\mathcal{A}_{\mathcal{SH}}$, we remind the reader of a similar interpretation function $\mathcal{A}$ over PCF constants—i.e. $\mathcal{C}_{\mathrm{A}}$—as defined by Plotkin [1977].

Definition 3.12. $\mathcal{A}\colon \mathcal{C}_{\mathrm{A}} \to \cup\{\mathbb{D}_\sigma \mid \sigma \text{ a PCF type}\}$ is defined as in Table 2.

Definition 3.13 (Denotational Semantics). Relative to each environment ρ, the denotational semantics of SHRAD-terms is defined recursively as follows:

1. $[\![x]\!]_\rho := \rho(x)$, where x is a variable.
2. $[\![c]\!]_\rho := \mathcal{A}(c)$ where c is a constant of PCF and $\mathcal{A}$ is the function defined in Table 2.
3. $[\![c]\!]_\rho := \mathcal{A}_{\mathcal{SH}}(c)$, where $c \in \mathcal{P}_{SH}$ is a proper SHRAD-constant and $\mathcal{A}_{\mathcal{SH}}$ is the function defined in Definition 3.11.
4. $[\![MN]\!]_\rho := [\![M]\!]_\rho[\![N]\!]_\rho$.
5. $\forall a \in \mathbb{D}_\sigma$, $[\![\lambda x\colon \sigma\,.\,M]\!]_\rho(a) := [\![M]\!]_{\rho_{x\mapsto a}}$.

Table 2. The function $\mathcal{A}$.

$$
\begin{aligned}
\mathcal{A}(\mathit{true}) &= \mathit{tt}\\
\mathcal{A}(\mathit{false}) &= \mathit{ff}\\
\mathcal{A}(\overline{n}) &= n \quad (n \geq 0)\\
\mathcal{A}(\mathit{if}_\sigma)(p)(x)(y) &= \begin{cases} x & (\text{if } p = \mathit{tt})\\ y & (\text{if } p = \mathit{ff})\\ \bot & (\text{if } p = \bot)\end{cases} \quad (p \in \mathbb{B}_\bot,\ x, y \in \mathbb{D}_\sigma \text{ and } \sigma \text{ ground})\\
\mathcal{A}(\mathit{succ})(x) &= \begin{cases} x+1 & (x \geq 0)\\ \bot & (x = \bot)\end{cases} \quad (x \in \mathbb{N}_\bot)\\
\mathcal{A}(\mathit{pred})(x) &= \begin{cases} x-1 & (x \geq 1)\\ \bot & (x \in \{\bot, 0\})\end{cases} \quad (x \in \mathbb{N}_\bot)\\
\mathcal{A}(\mathit{Zero})(x) &= \begin{cases} \mathit{tt} & (x = 0)\\ \mathit{ff} & (x > 0)\\ \bot & (x = \bot)\end{cases} \quad (x \in \mathbb{N}_\bot)\\
\mathcal{A}(Y_\sigma)(f) &= \sqcup\{f^n(\bot) \mid n \geq 0\} \quad (\forall f \in \mathbb{D}_{\sigma\to\sigma})
\end{aligned}
$$

4. Interpreting Constants

We already know how to interpret a program $P\colon r$ as a (partial) real number, all we need to do is to consider $to_{\mathcal{J}}\,[\![P]\!]$. Now let us move up to first-order and consider a term $P\colon r^m \to r$ ($m \geq 1$). Perhaps the natural way of observing this term's behaviour is to define some

$$to_{\mathcal{J}^m\to\mathcal{J}}\colon [(\mathcal{D}^\infty)^m \to \mathcal{D}^\infty] \to (\mathcal{J}^m \to \mathcal{J}),$$

where, given a function $f \in [(\mathcal{D}^\infty)^m \to \mathcal{D}^\infty]$ and a vector $\boldsymbol{a} = (a_1, \ldots, a_m) \in \mathcal{J}^m$,

$$to_{\mathcal{J}^m\to\mathcal{J}}\ f\ \boldsymbol{a} = \begin{cases} to_{\mathcal{J}}(f\ d_1 \ldots d_m) & \text{if} \quad \forall i \leq m, \quad to_{\mathcal{J}}\ d_i = a_i, \\ \text{undefined} & \text{if no such } \boldsymbol{d} \text{ exists.} \end{cases} \tag{1}$$

For a SHRAD-definable f, the question is whether $to_{\mathcal{J}^m\to\mathcal{J}}\ f$ is a well-defined *function* or not? In other words, for such a function, does the specific choice of $\boldsymbol{d}$ in (1) above really matter? The answer to this last question is—unfortunately—"yes", and a counter-example is readily available.

Take the constant $avg\colon r \to r \to r$ and consider its denotation $\mathsf{avg}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$ as in Definition 4.28. Also consider the intervals

$$I_1 = [-\tfrac{1}{2}, 0], \qquad I_2 = [-\tfrac{1}{2}, \tfrac{1}{2}].$$

It is easy to verify that

$$to_{\mathcal{J}}(\mathsf{L} : \mathsf{R} : [\,]) = to_{\mathcal{J}}(\mathsf{M}\colon \mathsf{L}\colon [\,]) = I_1,$$
$$to_{\mathcal{J}}(\mathsf{M} : [\,]) = I_2.$$

Yet we have

$$\begin{cases} \mathsf{avg}\ (\mathsf{L}\colon \mathsf{R}\colon [\,])\ (\mathsf{M}\colon [\,]) = [\,] \\ \mathsf{avg}\ (\mathsf{M}\colon \mathsf{L}\colon [\,])\ (\mathsf{M}\colon [\,]) = \mathsf{M}\colon [\,] \end{cases}$$

which leads to

$$\begin{cases} to_{\mathcal{J}}(\mathsf{avg}(\mathsf{L}\colon \mathsf{R}\colon [\,])\ (\mathsf{M}\colon [\,])) = [-1, 1], \\ to_{\mathcal{J}}(\mathsf{avg}(\mathsf{M}\colon \mathsf{L}\colon [\,])\ (\mathsf{M}\colon [\,])) = [-\tfrac{1}{2}, \tfrac{1}{2}] \end{cases} \tag{2}$$

Thus, in general $to_{\mathcal{J}^m\to\mathcal{J}}[\![P]\!]$ is a *relation* rather than a function. In the special case of the above counter-example, we say that $to_{\mathcal{J}^2\to\mathcal{J}}\ \mathsf{avg}\colon \mathcal{J}^2 \to \mathcal{J}$—or by an abuse of language $\mathsf{avg}\colon (\mathcal{D}^\infty)^2 \to \mathcal{D}^\infty$ for that matter—is *non-extensional* at $([-\tfrac{1}{2}, 0], [-\tfrac{1}{2}, \tfrac{1}{2}])$.

How bad is the situation? Looking at it from another angle, we should remember that it is the behaviour of the function over *total* real numbers that ultimately matters. Therefore we need to know whether for any term $P\colon r^m \to r$ it is possible that $to_{\mathcal{J}^m\to\mathcal{J}}[\![P]\!]$ is non-extensional over any maximal $\boldsymbol{a} \in \mathcal{J}^m$.

For the rest of this section we go through the denotation of each individual constant and demonstrate that for each such constant, say $c \in \mathcal{P}_{SH}$, $[\![c]\!]$ preserves the partial equivalence relation of Definition 4.1 below, *except for limit*. Then in Section 6.1 we

weaken the extensionality criteria by defining a binary logical relation X which is indeed preserved by any term of the language.[6]

Definition 4.1 ($\simeq_{\mathcal{D}^\infty} \subseteq \mathcal{D}^\infty \times \mathcal{D}^\infty$). $x, y \in \mathcal{D}^\infty$ are equivalent if and only if they represent the same *total* real number, i.e.

$$x \simeq_{\mathcal{D}^\infty} y \iff (length(x) \notin \mathbb{N}) \wedge (to_{\mathcal{J}}\, x = to_{\mathcal{J}}\, y)$$

At the same time, we embark on proving the correctness of each interpretation, for which we present the following definition:

Notation 4.2 ($\mathcal{J}_{\max}$). By $\mathcal{J}_{\max}$ we mean the set $[-1, 1]$, the image of which under the embedding e of Note 3.7 is the set of maximal elements of $\mathcal{J}$.

Definition 4.3 ($to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}}$). For each $m \geq 1$, let

$$to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}} \colon [(\mathcal{D}^\infty)^m \to \mathcal{D}^\infty] \rightarrowtail (\mathcal{J}_{\max}^m \rightharpoonup \mathcal{J}_{\max})$$

be the *partial* function where given any $f \in [(\mathcal{D}^\infty)^m \to \mathcal{D}^\infty]$, the function

$$to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}}\,(f)$$

1. is *undefined* if for some $\boldsymbol{a} \in \mathcal{J}_{\max}^m$, there are two representations $\boldsymbol{d_1}, \boldsymbol{d_2} \in (\mathcal{D}^\infty)^m$ such that

   $to_{\mathcal{J}}\,(f\,(\boldsymbol{d_1}))\cap to_{\mathcal{J}}\,(f\,(\boldsymbol{d_2})) = \emptyset$

2. otherwise is the (partial) function $\hat{f}\colon \mathcal{J}_{\max}^m \rightharpoonup \mathcal{J}_{\max}$ for which:

   (a) $dom(\hat{f})$ is the set of all $\boldsymbol{x} \in \mathcal{J}_{\max}^m$ such that for any two $\boldsymbol{d_1}, \boldsymbol{d_2} \in (\mathcal{D}^\infty)^m$ representing $\boldsymbol{x}$,

   $(to_{\mathcal{J}}(f(\boldsymbol{d_1})) \in \mathcal{J}_{\max}) \wedge (to_{\mathcal{J}}(f(\boldsymbol{d_1})) = to_{\mathcal{J}}(f(\boldsymbol{d_2})))$,

   (b) $\forall \boldsymbol{a} \in dom(\hat{f})\,,\ \hat{f}(\boldsymbol{a}) = to_{\mathcal{J}}\,(f\,(\boldsymbol{d}))$ where

   $\boldsymbol{d} = (d_1, \ldots, d_m) \in (\mathcal{D}^\infty)^m$

   is some representation of $\boldsymbol{a}$.

For instance, in Corollary 4.33 we show that:

$$\forall x, y \in [-1, 1], \qquad (to_{\mathcal{J}_{\max}^2 \to \mathcal{J}_{\max}}\,(\mathsf{avg}))\,x\,y = \frac{x+y}{2}.$$

After discussing each individual constant, we will have a further task to do and that is demonstrating that the extensionality property is preserved throughout the construction

[6] See Conjecture 10.2.

of more complicated terms out of the basic ones. Of course one has to bear in mind that with the introduction of *isNeg* and *limit*, we will be bound to get partial functions over real numbers. This is yet another motivation for a reformulation of extensionality by the logical relation X as we do in Section 6. Thereafter, we are assured that in fact for *any* SHRAD-definable

$$f\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$$

one gets a proper (partial) function

$$\hat{f}\colon [-1,1]^m \rightharpoonup [-1,1].$$

4.1. $\mathsf{Cons_L}, \mathsf{Cons_M}, \mathsf{Cons_R}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty$

We begin by defining:

Definition 4.4 ($\mathsf{Cons_L}, \mathsf{Cons_M}, \mathsf{Cons_R}$).

$$\begin{cases} \mathsf{Cons_L}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{Cons_L}(x) = \mathsf{L} : x. \end{cases} \qquad \begin{cases} \mathsf{Cons_M}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{Cons_M}(x) = \mathsf{M} : x. \end{cases} \qquad \begin{cases} \mathsf{Cons_R}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{Cons_R}(x) = \mathsf{R} : x. \end{cases}$$

Next we set

$$\mathcal{A}_{\mathcal{SH}}(L) := \mathsf{Cons_L}, \qquad \mathcal{A}_{\mathcal{SH}}(M) := \mathsf{Cons_M}, \qquad \mathcal{A}_{\mathcal{SH}}(R) := \mathsf{Cons_R}.$$

For each $d \in \mathcal{D}^\infty$, let $[a_d, b_d] = to_{\mathcal{J}}\, d$. We have

$$\begin{aligned} to_{\mathcal{J}}(\mathsf{Cons_L}\ d) &= to_{\mathcal{J}}(\mathsf{L} : d) \\ (\text{definition of } to_{\mathcal{J}}) &= \mathsf{Cons}_{[-1,0]}(to_{\mathcal{J}}\, d) \\ &= \mathsf{Cons}_{[-1,0]}\ [a_d, b_d] \\ (\text{definition of } \mathsf{Cons}_{[-1,0]}\colon \mathcal{J} \to \mathcal{J})^{7} &= [(a_d - 1)/2\ ,\ (b_d - 1)/2]. \end{aligned}$$

Adding similar arguments, we get:

Proposition 4.5. *For each $d \in \mathcal{D}^\infty$, let $[a_d, b_d] = to_{\mathcal{J}}\, d$. We have*:

1. $to_{\mathcal{J}}(\mathsf{Cons_L}\ d) = [(a_d - 1)/2\ ,\ (b_d - 1)/2]$.
2. $to_{\mathcal{J}}(\mathsf{Cons_M}\ d) = [a_d/2\ ,\ b_d/2]$.
3. $to_{\mathcal{J}}(\mathsf{Cons_R}\ d) = [(a_d + 1)/2\ ,\ (b_d + 1)/2]$.

Lemma 4.6. *For any $d_x, e_x \in \mathcal{D}^\infty$ such that $d_x \simeq_{\mathcal{D}^\infty} e_x$, we have*:

1. $\mathsf{Cons_L}\ d_x \simeq_{\mathcal{D}^\infty} \mathsf{Cons_L}\ e_x$.
2. $\mathsf{Cons_M}\ d_x \simeq_{\mathcal{D}^\infty} \mathsf{Cons_M}\ e_x$.
3. $\mathsf{Cons_R}\ d_x \simeq_{\mathcal{D}^\infty} \mathsf{Cons_R}\ e_x$.

[7] See Escardó [1997].

Proof. Let $d_x, e_x \in \mathcal{D}^\omega$ be representations of the same total real number, say, $x \in [-1, 1]$, i.e.

$$x = to_{\mathcal{J}}\, d_x = to_{\mathcal{J}}\, e_x.$$

Now by Proposition 4.5 we have

$$\begin{aligned} to_{\mathcal{J}}(\mathsf{Cons_L}\ d_x) &= (x-1)/2 \\ &= to_{\mathcal{J}}(\mathsf{Cons_L}\ e_x). \end{aligned}$$

Therefore,

$$\mathsf{Cons_L}\ d_x \simeq_{\mathcal{D}^\infty} \mathsf{Cons_L}\ e_x.$$

Similar arguments apply to both $\mathsf{Cons_M}$ and $\mathsf{Cons_R}$. □

Combining Proposition 4.5 and Lemma 4.6, one gets:

Corollary 4.7 (Extensionality). $\mathsf{Cons_L}$, $\mathsf{Cons_M}$ *and* $\mathsf{Cons_R}$ *are extensional over total real numbers and hence are interpreted as the following functions over* $[-1, 1]$:

$$\begin{cases} to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_L}\colon [-1,1] \to [-1,1], \\ (to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_L})(x) = (x-1)/2. \end{cases}$$

$$\begin{cases} to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_M}\colon [-1,1] \to [-1,1], \\ (to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_M})(x) = x/2. \end{cases}$$

$$\begin{cases} to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_R}\colon [-1,1] \to [-1,1], \\ (to_{\mathcal{J}_{\max}\to\mathcal{J}_{\max}}\ \mathsf{Cons_R})(x) = (x+1)/2. \end{cases}$$

Notation 4.8. Although we gave a proper name $\mathsf{Cons_L}$ to $\mathcal{A}_{\mathcal{SH}}(L)$, we will freely use the (Haskell style) notation $\mathsf{L} : x$ instead of the cumbersome $\mathsf{Cons_L}\ x$ wherever convenient. The same applies to M: and R:.

4.2. $\mathsf{L}^{-1}, \mathsf{M}^{-1}, \mathsf{R}^{-1}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty$

For any real number $x \in (-1, 1)$, infinitely many representations are available inside $\mathcal{D}^\infty$. For instance, the following three sequences all represent the real number *zero*:

$$d_0 = \mathsf{L} : \mathsf{R} : \mathsf{R} : \mathsf{R} \cdots,$$

$$d_0' = \mathsf{M} : \mathsf{M} : \mathsf{M} : \mathsf{M} \cdots,$$

$$d_0'' = \mathsf{R} : \mathsf{L} : \mathsf{L} : \mathsf{L} \cdots.$$

Now suppose that at some stage we get a sequence $d_s = \mathsf{L}\colon x$ representing a real number $s \in [-1, 1]$, and for some reason we need to rewrite the sequence in such a way that it

starts with, say, M, and, of course, the resulting sequence $e_s = \mathsf{M} : x'$ also represents s. Well, first, this is not possible at all unless $s \in [-\frac{1}{2}, \frac{1}{2}]$, in which case, of course, there is a straightforward way of getting e_s from d_s. However, even in case s lies outside $[-\frac{1}{2}, \frac{1}{2}]$, we can still produce some $e_{s'} = \mathsf{M}{:}\, x'$ which represents one of the two values $\{-\frac{1}{2}, \frac{1}{2}\}$, i.e. the one closer to s. So let us start implementing M^{-1}.

The implementation is a bit involved, hence we break it into different stages. We need a few definitions to begin with.

Definition 4.9 ($d_{-1}, d_1 \in \mathcal{D}^\infty$). We let $d_{-1}, d_1 \in \mathcal{D}^\infty$ denote the representations of the real numbers -1 and 1, respectively, i.e.

$$d_{-1} = \mathsf{L} : \mathsf{L} : \mathsf{L} : \cdots,$$
$$d_1 = \mathsf{R} : \mathsf{R} : \mathsf{R} : \cdots.$$

Next we define (versions of) (-1) and $(+1)$ functions over $\mathcal{D}^\infty$:

Definition 4.10 ($((-1)_{\mathcal{D}^\infty}, (+1)_{\mathcal{D}^\infty}$).

$$\begin{cases} (-1)_{\mathcal{D}^\infty}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ (-1)_{\mathcal{D}^\infty}(\mathsf{L} : x) = d_{-1}, \\ (-1)_{\mathcal{D}^\infty}(\mathsf{M} : x) = \mathsf{L} : ((-1)_{\mathcal{D}^\infty}\, x), \\ (-1)_{\mathcal{D}^\infty}(\mathsf{R} : x) = \mathsf{L} : x. \end{cases} \qquad \begin{cases} (+1)_{\mathcal{D}^\infty}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ (+1)_{\mathcal{D}^\infty}(\mathsf{L}{:}\, x) = \mathsf{R}{:}\, x, \\ (+1)_{\mathcal{D}^\infty}(\mathsf{M}{:}\, x) = \mathsf{R}{:}\, ((+1)_{\mathcal{D}^\infty}\, x), \\ (+1)_{\mathcal{D}^\infty}(\mathsf{R}{:}\, x) = d_1. \end{cases}$$

Proposition 4.11 (Totality of $(-1)_{\mathcal{D}^\infty}$ and $(+1)_{\mathcal{D}^\infty}$).

$$\forall x \in \mathcal{D}^\omega, \qquad \begin{cases} ((-1)_{\mathcal{D}^\infty}\, x) \in \mathcal{D}^\omega, \\ ((+1)_{\mathcal{D}^\infty}\, x) \in \mathcal{D}^\omega. \end{cases}$$

Proof. Left to the reader. □

We also define the counterpart (-1) and $(+1)$ functions over $\mathcal{J}$:

Definition 4.12 ($((-1)_{\mathcal{J}}, (+1)_{\mathcal{J}}$).

$$\begin{cases} (-1)_{\mathcal{J}}\colon \mathcal{J} \to \mathcal{J}, \\ (-1)_{\mathcal{J}}[a, b] = [\max\{-1, a-1\}, \max\{-1, b-1\}]. \end{cases}$$

$$\begin{cases} (+1)_{\mathcal{J}}\colon \mathcal{J} \to \mathcal{J}, \\ (+1)_{\mathcal{J}}[a, b] = [\min\{1, a+1\}, \min\{1, b+1\}]. \end{cases}$$

Proposition 4.13.

$$\forall x \in \mathcal{D}^\infty, \qquad \begin{cases} to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ x) \sqsubseteq (-1)_{\mathcal{J}}(to_{\mathcal{J}}\ x), \\ to_{\mathcal{J}}((+1)_{\mathcal{D}^\infty}\ x) \sqsubseteq (+1)_{\mathcal{J}}(to_{\mathcal{J}}\ x). \end{cases}$$

Proof. We just consider the case of $(-1)_{\mathcal{D}^\infty}$. The proof for $(+1)_{\mathcal{D}^\infty}$ is similar. In any case, the proof is done by induction over x:

1. Case $x = [\ ]$: we have

$$\begin{aligned} to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ x) &= to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ [\ ]) \\ \text{(definition of } (-1)_{\mathcal{D}^\infty}) &= to_{\mathcal{J}}([\ ]) \\ \text{(definition of } to_{\mathcal{J}}) &= [-1, 1] \\ &\sqsubseteq [-1, 0] \\ \text{(definitions of } (-1)_{\mathcal{J}} \text{ and } to_{\mathcal{J}}) &= (-1)_{\mathcal{J}}(to_{\mathcal{J}}[\ ]) \\ &= (-1)_{\mathcal{J}}(to_{\mathcal{J}}\ x). \end{aligned}$$

2. Case $x = \mathsf{L} : x'$: we have

$$\begin{aligned} to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ x) &= to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ (\mathsf{L} : x')) \\ \text{(definition of } (-1)_{\mathcal{D}^\infty}) &= to_{\mathcal{J}}(d_{-1}) \\ \text{(definition of } d_{-1}) &= [-1, -1] \\ \text{(easy calculations)} &= (-1)_{\mathcal{J}}\mathsf{Cons}_{[-1,0]}(to_{\mathcal{J}}x') \\ \text{(definition of } to_{\mathcal{J}}) &= (-1)_{\mathcal{J}}(to_{\mathcal{J}}\ (\mathsf{L} : x')) \\ &= (-1)_{\mathcal{J}}\ (to_{\mathcal{J}}\ x). \end{aligned}$$

3. Case $x = \mathsf{M} : x'$: we have

$$\begin{aligned} to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ x) &= to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ (\mathsf{M} : x')) \\ \text{(definition of } (-1)_{\mathcal{D}^\infty}) &= to_{\mathcal{J}}(\mathsf{L} : ((-1)_{\mathcal{D}^\infty}\ x')) \\ \text{(definition of } to_{\mathcal{J}}) &= \mathsf{Cons}_{[-1,0]}\ (to_{\mathcal{J}}\ ((-1)_{\mathcal{D}^\infty}\ x')) \\ \text{(induction hypothesis)} &\sqsubseteq \mathsf{Cons}_{[-1,0]}\ ((-1)_{\mathcal{J}}\ (to_{\mathcal{J}}\ x')) \\ \text{(cumbersome calculations)} &= (-1)_{\mathcal{J}}\mathsf{Cons}_{[-1/2,1/2]}(to_{\mathcal{J}}\ x') \\ \text{(definition of } to_{\mathcal{J}}) &= (-1)_{\mathcal{J}}(to_{\mathcal{J}}\ (\mathsf{M} : x')) \\ &= (-1)_{\mathcal{J}}\ (to_{\mathcal{J}}\ x). \end{aligned}$$

4. Case $x = \mathsf{R} : x'$: we have

$$to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ x) = to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\ (\mathsf{R} : x'))$$

$$\begin{aligned}
(\text{definition of } (-1)_{\mathcal{D}^\infty}) &= to_{\mathcal{J}}(\mathsf{L} : x') \\
(\text{definition of } to_{\mathcal{J}}) &= \mathsf{Cons}_{[-1,0]}\,(to_{\mathcal{J}}\, x') \\
(\text{easy calculations}) &= (-1)_{\mathcal{J}}\mathsf{Cons}_{[0,1]}(to_{\mathcal{J}}\, x') \\
(\text{definition of } to_{\mathcal{J}}) &= (-1)_{\mathcal{J}}(to_{\mathcal{J}}\,(\mathsf{R} : x')) \\
&= (-1)_{\mathcal{J}}\,(to_{\mathcal{J}}\, x).
\end{aligned}$$

□

Combining Propositions 4.11 and 4.13 we get:

Corollary 4.14 (Correctness of $(-1)_{\mathcal{D}^\infty}$ and $(+1)_{\mathcal{D}^\infty}$). *For any $d_x \in \mathcal{D}^\omega$ representing a real number $x \in [-1, 1]$:*

$$\begin{cases} to_{\mathcal{J}}((-1)_{\mathcal{D}^\infty}\, d_x) = \max\{-1, x - 1\}, \\ to_{\mathcal{J}}((+1)_{\mathcal{D}^\infty}\, d_x) = \min\{1, x + 1\}, \end{cases}$$

which leads to:

Corollary 4.15 (Extensionality of $(-1)_{\mathcal{D}^\infty}$ and $(+1)_{\mathcal{D}^\infty}$).

$$\forall d_x, e_x \in \mathcal{D}^\infty, \qquad d_x \simeq_{\mathcal{D}^\infty} e_x \Rightarrow \begin{cases} (-1)_{\mathcal{D}^\infty}\, d_x \simeq_{\mathcal{D}^\infty} (-1)_{\mathcal{D}^\infty}\, e_x, \\ (+1)_{\mathcal{D}^\infty}\, d_x \simeq_{\mathcal{D}^\infty} (+1)_{\mathcal{D}^\infty}\, e_x. \end{cases}$$

We are now in a stage where we can define our main functions:

Definition 4.16 ($\mathsf{L}^{-1}, \mathsf{M}^{-1}, \mathsf{R}^{-1}: \mathcal{D}^\infty \to \mathcal{D}^\infty$).

$$\begin{cases} \mathsf{L}^{-1}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{L}^{-1}(\mathsf{L} : x) = x, \\ \mathsf{L}^{-1}(\mathsf{M} : x) = (+1)_{\mathcal{D}^\infty}\, x, \\ \mathsf{L}^{-1}(\mathsf{R} : x) = d_1. \end{cases} \qquad \begin{cases} \mathsf{M}^{-1} : \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{M}^{-1}(\mathsf{L} : x) = (-1)_{\mathcal{D}^\infty}\, x, \\ \mathsf{M}^{-1}(\mathsf{M} : x) = x \\ \mathsf{M}^{-1}(\mathsf{R} : x) = (+1)_{\mathcal{D}^\infty}\, x. \end{cases}$$

$$\begin{cases} \mathsf{R}^{-1}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty \\ \mathsf{R}^{-1}(\mathsf{L} : x) = d_{-1}, \\ \mathsf{R}^{-1}(\mathsf{M} : x) = (-1)_{\mathcal{D}^\infty}\, x, \\ \mathsf{R}^{-1}(\mathsf{R} : x) = x. \end{cases},$$

Lemma 4.17 (Totality of L^{-1}, M^{-1} and R^{-1}).

$$\forall x \in \mathcal{D}^\omega, \qquad \begin{cases} \mathsf{L}^{-1}x \in \mathcal{D}^\omega, \\ \mathsf{M}^{-1}x \in \mathcal{D}^\omega, \\ \mathsf{R}^{-1}x \in \mathcal{D}^\omega. \end{cases}$$

Proof. Immediate from the definition using Proposition 4.11. □

Lemma 4.18. *For all* $x \in \mathcal{D}^\infty$,

$$\begin{cases} to_{\mathcal{J}}(\mathsf{L}^{-1}x) \sqsubseteq \mathsf{Tail}_{[-1,0]}\ (to_{\mathcal{J}}(x)), \\ to_{\mathcal{J}}(\mathsf{M}^{-1}x) \sqsubseteq \mathsf{Tail}_{[-1/2,1/2]}\ (to_{\mathcal{J}}(x)), \\ to_{\mathcal{J}}(\mathsf{R}^{-1}x) \sqsubseteq \mathsf{Tail}_{[0,1]}\ (to_{\mathcal{J}}(x)). \end{cases}$$

Proof. The proof can be easily done by induction over x, hence we omit it here. □

Combining Lemmata 4.17 and 4.18 we get:

Corollary 4.19. *For any* $d_x \in \mathcal{D}^\omega$ *representing a real number* $x \in [-1, 1]$,

$$to_{\mathcal{J}}(\mathsf{L}^{-1}\ d_x) = \begin{cases} 2x+1 & \text{if} \quad x \in [-1, 0], \\ 1 & \text{if} \quad x \in [0, 1], \end{cases}$$

$$to_{\mathcal{J}}(\mathsf{M}^{-1}\ d_x) = \begin{cases} -1 & \text{if} \quad x \in [-1, -\frac{1}{2}], \\ 2x & \text{if} \quad x \in [-\frac{1}{2}, \frac{1}{2}], \\ 1 & \text{if} \quad x \in [\frac{1}{2}, 1], \end{cases}$$

$$to_{\mathcal{J}}(\mathsf{R}^{-1}\ d_x) = \begin{cases} -1 & \text{if} \quad x \in [-1, 0], \\ 2x-1 & \text{if} \quad x \in [0, 1], \end{cases}$$

which leads to:

Corollary 4.20 (Extensionality of L^{-1}, M^{-1} and R^{-1}).

$$\forall d_x, e_x \in \mathcal{D}^\infty, \qquad d_x \simeq_{\mathcal{D}^\infty} e_x \Rightarrow \begin{cases} \mathsf{L}^{-1}d_x \simeq_{\mathcal{D}^\infty} \mathsf{L}^{-1}e_x, \\ \mathsf{M}^{-1}d_x \simeq_{\mathcal{D}^\infty} \mathsf{M}^{-1}e_x, \\ \mathsf{R}^{-1}d_x \simeq_{\mathcal{D}^\infty} \mathsf{R}^{-1}e_x. \end{cases}$$

4.3. $\mathsf{qtoR}\colon \mathbb{N}_\perp \to \mathbb{N}_\perp \to \mathbb{N}_\perp \to \mathcal{D}^\infty$

We need an operator over triplets of natural numbers such that on any

$$(p, m, n) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

satisfying

1. $n \neq 0$,
2. $m/n \leq 1$,

returns some maximal element of $\mathcal{D}^\infty$ representing the rational number:

1. m/n if $p > 0$.
2. $-m/n$ if $p = 0$.

First assume that

$$\widehat{\mathsf{qtoR}}\colon \mathbb{Q}\cap[-1,1]\to\mathcal{D}^\infty$$

is some (effective or non-effective) function which does the same thing but acts directly on rational numbers, i.e. $\widehat{\mathsf{qtoR}}$ picks a canonical representation for each rational number in its domain. Therefore,

$$\forall q\in\mathbb{Q}\cap[-1,1],\qquad to_{\mathcal{J}}(\widehat{\mathsf{qtoR}}\ q)=q.$$

It is easy to verify that:

1. $\forall q\in\mathbb{Q}\cap[0,1],\ q=to_{\mathcal{J}}(\mathsf{R}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[0,1]}\ q)))$.
2. $\forall q\in\mathbb{Q}\cap[-1,0],\ q=to_{\mathcal{J}}(\mathsf{L}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[-1,0]}\ q)))$.
3. $\forall q\in\mathbb{Q}\cap[-\frac{1}{2},\frac{1}{2}],\ q=to_{\mathcal{J}}(\mathsf{M}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[-1/2,1/2]}\ q)))$.

In fact, we can *effectively* define such a function:

Definition 4.21 ($\widehat{\mathsf{qtoR}}$).

$$\widehat{\mathsf{qtoR}}\colon \mathbb{Q}\cap[-1,1]\to\mathcal{D}^\infty,$$

$$\widehat{\mathsf{qtoR}}\ q=\begin{cases}\mathsf{R}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[0,1]}\ q))) & \text{if}\quad q>0,\\ \mathsf{L}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[-1,0]}\ q))) & \text{if}\quad q<0,\\ \mathsf{M}\colon\ (\widehat{\mathsf{qtoR}}(\mathsf{Tail}_{[-1/2,1/2]}\ q))) & \text{otherwise.}\end{cases}$$

Going back to the triplets of natural numbers, we define our main function as:

Definition 4.22 (qtoR). The function $\mathsf{qtoR}\colon\mathbb{N}_\perp\to\mathbb{N}_\perp\to\mathbb{N}_\perp\to\mathcal{D}^\infty$ is defined as follows:

$$\mathsf{qtoR}\ p\ m\ n=\begin{cases}[\,] & \text{if}\quad n=0,\\ \widehat{\mathsf{qtoR}}\ 1 & \text{if}\quad (m/n>1)\wedge(p>0),\\ \widehat{\mathsf{qtoR}}\ (-1) & \text{if}\quad (m/n>1)\wedge(p=0),\\ \widehat{\mathsf{qtoR}}\ (m/n) & \text{if}\quad (m/n\le 1)\wedge(p>0),\\ \widehat{\mathsf{qtoR}}\ (-m/n) & \text{if}\quad (m/n\le 1)\wedge(p=0).\end{cases}$$

The following lemmata are now immediate:

Lemma 4.23 (Correctness of qtoR).

$$\forall p,m,n\in\mathbb{N},\qquad to_{\mathcal{J}}(\mathsf{qtoR}\ p\ m\ n)=\begin{cases}[-1,1] & \textit{if}\quad n=0,\\ 1 & \textit{if}\quad (m/n>1)\wedge(p>0),\\ -1 & \textit{if}\quad (m/n>1)\wedge(p=0),\\ m/n & \textit{if}\quad (m/n\le 1)\wedge(p>0),\\ -m/n & \textit{if}\quad (m/n\le 1)\wedge(p=0).\end{cases}$$

Notation 4.24. We say that a triplet $(p, m, n) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ *represents* the rational number $q \in \mathbb{Q} \cap [-1, 1]$ if:

1. $n \neq 0$,
2. $\begin{cases} (p > 0) \wedge (m/n > 1) & \text{if} \quad q = 1, \\ (p = 0) \wedge (m/n > 1) & \text{if} \quad q = -1, \\ (p > 0) \wedge (m/n = q) & \text{if} \quad 0 \leq q \leq 1, \\ (p = 0) \wedge (-m/n = q) & \text{if} \quad -1 \leq q \leq 0. \end{cases}$

Lemma 4.25 (Totality of qtoR). *On triplets (p, m, n) representing rational numbers, i.e. $n \neq 0$, (qtoR p m n) is an infinite sequence of digits*:

$$\forall (p, m, n) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}, \qquad n \neq 0 \Rightarrow \mathsf{qtoR}\ p\ m\ n \in \mathcal{D}^\omega.$$

4.4. avg: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$

We are going to present an implementation of the *average* function over $\mathcal{D}^\infty$, written in Haskell style to emphasize the sequential nature of the implementation. We essentially define the function over a few cases and then using commutativity law and the negation function, refer other cases to these few as well. However, it is useful to emphasize that the definition could be done in such a way that no case was converted to any other case, only it would be much longer and more cumbersome than it is already.

Let us first define the auxiliary function neg:

Definition 4.26 (neg: $\mathcal{D}^\infty \to \mathcal{D}^\infty$).

$$\begin{aligned} \mathsf{neg}[\,] &= [\,], \\ \mathsf{neg}(\mathsf{L} : x) &= \mathsf{R} : (\mathsf{neg}(x)), \\ \mathsf{neg}(\mathsf{M} : x) &= \mathsf{M} : (\mathsf{neg}(x)), \\ \mathsf{neg}(\mathsf{R} : x) &= \mathsf{L} : (\mathsf{neg}(x)). \end{aligned}$$

Proposition 4.27. neg *satisfies the following properties*:

1. (*totality*): $\forall d \in \mathcal{D}^\omega, \mathsf{neg}(d) \in \mathcal{D}^\omega$.
2. (*extensionality*) : $\forall d_x, d_y \in \mathcal{D}^\infty, d_x \simeq_{\mathcal{D}^\infty} d_y \Rightarrow \mathsf{neg}(d_x) \simeq_{\mathcal{D}^\infty} \mathsf{neg}(d_y)$.
3. (*correctness*): *For any $d \in \mathcal{D}^\infty$, let $[a_d, b_d] = to_{\mathcal{J}}(d)$. Therefore we have*

   $to_{\mathcal{J}}(\mathsf{neg}(d)) = [-b_d, -a_d]$

   and in particular

   $\forall d \in \mathcal{D}^\omega, \qquad to_{\mathcal{J}}(\mathsf{neg}(d)) = -to_{\mathcal{J}}(d)$.

Proof. Left to the reader. □

Definition 4.28 (avg: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$). The function avg is defined as in Table 3.

Table 3. The function avg: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$.

avg	(L: x)	(L: y)	=	L: (avg x y)
avg	(L: x)	(R: y)	=	M: (avg x y)
avg	(L: L: x)	(M: L: y)	=	L: M: (avg x y)
avg	(L: L: x)	(M: M: y)	=	L: (avg (M: x) (R: y))
avg	(L: L: x)	(M: R: y)	=	L: R: (avg x y)
avg	(L: M: x)	(M: L: y)	=	L: (avg (M: x) (R: y))
avg	(L: M: x)	(M: M: y)	=	L: R: (avg x y)
avg	(L: M: x)	(M: R: y)	=	M: (avg (L: x) (M: y))
avg	(L: R: x)	(M: L: y)	=	L: R: (avg x y)
avg	(L: R: x)	(M: M: y)	=	M: (avg (M: x) (L: y))
avg	(L: R: x)	(M: R: y)	=	M: M: (avg x y)
avg	(M: x)	(M: y)	=	M: (avg x y)
avg	(M: x)	(L: y)	=	avg(L: y)(M: x)
avg	(M: x)	(R: y)	=	neg(avg (M: (neg(x))) (L: (neg(y))))
avg	(R: x)	(R: y)	=	R: (avg x y)
avg	(R: x)	(L: y)	=	M: (avg x y)
avg	(R: x)	(M: y)	=	avg (M: y) (R: x)

Lemma 4.29 (Continuity of avg). avg: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$ *is continuous*.

Proof. Immediate from the definition. □

Lemma 4.30 (avg is total). *The result of applying* avg *to representations of* total *real numbers is the representation of a* total *real number, i.e.*

$$\forall x, y \in \mathcal{D}^\omega, \qquad \mathsf{avg}\ x\ y\ \in \mathcal{D}^\omega.$$

Proof (sketch). At each stage, avg only needs at most two digits from each of its arguments to produce one digit of the output. □

To ensure that avg, when interpreted as an operation over real numbers, correctly calculates the *average* of its arguments, we introduce the following definition:

Definition 4.31 ($\oplus$: $\mathcal{J} \to \mathcal{J} \to \mathcal{J}$).

$$\forall [x_1, y_1], [x_2, y_2] \in \mathcal{J}, \qquad [x_1, y_1] \oplus [x_2, y_2] := [(x_1 + x_2)/2, (y_1 + y_2)/2].$$

Lemma 4.32.

$$\forall a, b \in \mathcal{D}^\infty, \qquad to_{\mathcal{J}}(\mathsf{avg}\ a\ b) \sqsubseteq_{\mathcal{J}} (to_{\mathcal{J}}\ a) \oplus (to_{\mathcal{J}}\ b).$$

Proof. We start by proving the lemma for elements of $\mathcal{D}^*$, i.e. finite sequences of digits. This we can achieve using induction over length of a and b. So suppose $a, b \in \mathcal{D}^*$:

Base case $a = b = [\,]$. In this case the lemma holds because $\mathsf{avg}\ [\,]\ [\,] = [\,]$, therefore

$$\begin{aligned} to_{\mathcal{J}}(\mathsf{avg}\ [\,]\ [\,]) &= to_{\mathcal{J}}([\,]) \\ &= [-1, 1] \\ &\sqsubseteq [-1, 1] \oplus [-1, 1] \\ &= to_{\mathcal{J}}([\,]) \oplus to_{\mathcal{J}}([\,]). \end{aligned}$$

Induction case. Let $a = a_0 \cdots a_n$, $b = b_0 \cdots b_m$ and suppose that the lemma holds for sequences of digits c with $\mathit{length}\, c < \max\{m, n\}$. We really need to check each individual case in Table 3 for avg. However, here we merely content ourselves with verifying one case, leaving the others to the ambitious reader as they are similarly straightforward. As an example, let

$$a = \mathsf{L}{:}\,\mathsf{L}{:}\, a_2 \cdots a_n,$$

$$b = \mathsf{M}{:}\,\mathsf{L}{:}\, b_2 \ldots b_m.$$

In other words,

$$a_0 = a_1 = \mathsf{L}$$

and

$$b_0 = \mathsf{M}, \qquad b_1 = \mathsf{L}.$$

We also let $a'' = a_2 \cdots a_n$ and $b'' = b_2 \cdots b_m$. Thus we can write

$$\begin{aligned} to_{\mathcal{J}}(\mathsf{avg}\ a\ b) &= to_{\mathcal{J}}(\mathsf{avg}(\mathsf{L}{:}\,\mathsf{L}{:}\, a'')(\mathsf{M}{:}\,\mathsf{L}{:}\, b'')) \\ \text{(definition of avg)} &= to_{\mathcal{J}}(\mathsf{L}{:}\,\mathsf{M}{:}\,(\mathsf{avg}\ a''\ b'')) \\ \text{(definition of } to_{\mathcal{J}}) &= \mathsf{Cons}_{[-1,0]}\ \mathsf{Cons}_{[-1/2,1/2]}\ to_{\mathcal{J}}(\mathsf{avg}\ a''\ b'') \\ \text{(induction hypothesis)} &\sqsubseteq \mathsf{Cons}_{[-1,0]}\ \mathsf{Cons}_{[-1/2,1/2]}((to_{\mathcal{J}}\ a'') \oplus (to_{\mathcal{J}}\ b'')) \\ \text{(a little bit of calculation)} &= (\mathsf{Cons}_{[-1,0]}\mathsf{Cons}_{[-1,0]}(to_{\mathcal{J}}\ a'')) \\ &\quad \oplus (\mathsf{Cons}_{[-1/2,1/2]}\mathsf{Cons}_{[-1,0]}(to_{\mathcal{J}}\ b'')) \\ \text{(definition of } to_{\mathcal{J}}) &= (to_{\mathcal{J}}\ (\mathsf{L}{:}\,\mathsf{L}{:}\, a'')) \oplus (to_{\mathcal{J}}\ (\mathsf{M}{:}\,\mathsf{L}{:}\, b'')) \\ &= (to_{\mathcal{J}}\ a) \oplus (to_{\mathcal{J}}\ b). \end{aligned}$$

Now we extend the result to the whole $\mathcal{D}^\infty$. For any $x \in \mathcal{D}^\infty$, we let $x_{<n}$ denote the initial segment of x of length not more than n. So, in particular,

$$\mathit{length}\, x < n \Rightarrow x_{<n} = x.$$

For any $x \in \mathcal{D}^\infty$, we have

$$x = \bigsqcup_{n \in \mathbb{N}} x_{<n}.$$

Therefore one can write:

$$\begin{aligned}
to_{\mathcal{J}}(\mathsf{avg}\ a\ b) &= to_{\mathcal{J}}\left(\mathsf{avg}\left(\bigsqcup_{n\in\mathbb{N}} a_{<n}\right)\left(\bigsqcup_{n\in\mathbb{N}} b_{<n}\right)\right)\\
\text{(continuity of avg)} &= to_{\mathcal{J}}\left(\bigsqcup_{m,n\in\mathbb{N}} \mathsf{avg}\ a_{<m}\ b_{<n}\right)\\
\text{(continuity of } to_{\mathcal{J}}) &= \bigsqcup_{m,n\in\mathbb{N}} (to_{\mathcal{J}}\ (\ \mathsf{avg}\ a_{<m}\ b_{<n}\ ))\\
\text{(by first part of the proof)} &\sqsubseteq \bigsqcup_{m,n\in\mathbb{N}} ((to_{\mathcal{J}}\ a_{<m}) \oplus (to_{\mathcal{J}}\ b_{<n}))\\
\text{(continuity of } \oplus) &= \left(\bigsqcup_{m\in\mathbb{N}} to_{\mathcal{J}}\ a_{<m}\right) \oplus \left(\bigsqcup_{n\in\mathbb{N}} to_{\mathcal{J}}\ b_{<n}\right)\\
\text{(continuity of } to_{\mathcal{J}}) &= to_{\mathcal{J}}\left(\bigsqcup_{n\in\mathbb{N}} a_{<n}\right) \oplus to_{\mathcal{J}}\left(\bigsqcup_{n\in\mathbb{N}} b_{<n}\right)\\
&= to_{\mathcal{J}}(a) \oplus to_{\mathcal{J}}(b).
\end{aligned}$$

□

Combining Lemmata 4.30 and 4.32 results in:

Corollary 4.33 (Correctness of avg). *For elements $a, b \in \mathcal{D}^\omega$, i.e. infinite sequences of digits*,

$$to_{\mathcal{J}}(\mathsf{avg}\ a\ b) = (to_{\mathcal{J}}\ a) \oplus (to_{\mathcal{J}}\ b).$$

Apart from correctness, the above corollary demonstrates the extensionality of avg at total real numbers as well. We already discussed that avg is not necessarily extensional at all partial real numbers (see equation (2)).

Now assume $d_x, e_x \in \mathcal{D}^\omega$ are two representations of the same number $x \in [-1, 1]$, and $d_y, e_y \in \mathcal{D}^\omega$ that of $y \in [-1, 1]$. In other words,

$$to_{\mathcal{J}}\ d_x = to_{\mathcal{J}}\ e_x = x,$$

$$to_{\mathcal{J}}\ d_y = to_{\mathcal{J}}\ e_y = y.$$

By Corollary 4.33, we have

$$\begin{aligned}
to_{\mathcal{J}}(\mathsf{avg}\ d_x\ d_y) &= (to_{\mathcal{J}}\ d_x) \oplus (to_{\mathcal{J}}\ d_y)\\
&= x \oplus y\\
&= (to_{\mathcal{J}}\ e_x) \oplus (to_{\mathcal{J}}\ e_y)\\
&= to_{\mathcal{J}}(\mathsf{avg}\ e_x\ e_y).
\end{aligned}$$

Corollary 4.34 (Extensionality of avg). avg *preserves the equivalence relation $\simeq_{\mathcal{D}^\infty}$, i.e. for all $d_x, e_x, d_y, e_y \in \mathcal{D}^\infty$,*

$$\left.\begin{array}{l} d_x \simeq_{\mathcal{D}^\infty} e_x \\ d_y \simeq_{\mathcal{D}^\infty} e_y \end{array}\right\} \quad \Rightarrow \quad (\mathsf{avg}\ d_x\ d_y) \simeq_{\mathcal{D}^\infty} (\mathsf{avg}\ e_x\ e_y).$$

4.5. mult: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$

Picking up pen and paper, trying to test one's own skills at implementing operations on real numbers in signed-binary format, the average operation should cause no trouble unless the lengthy mess of checking too many cases appears cumbersome, otherwise it is straightforward. As for *multiplication*, it is a rather different story. The implementation is not at all trivial. The one we present here was worked out by Plume [1998].

Again, we need a few definitions first:

Definition 4.35 (dp). Consider the set $\mathcal{D} = \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ of digits and the following pair of operations:

$$
\begin{aligned}
&e\colon \mathcal{D} \to \mathbb{Z}, && r\colon \mathbb{Z} \to \mathcal{D},\\
&e(\mathsf{L}) = -1, && \\
&e(\mathsf{M}) = 0, && r(x) = \begin{cases} \mathsf{L} & \text{if} \quad x < 0,\\ \mathsf{M} & \text{if} \quad x = 0,\\ \mathsf{R} & \text{if} \quad x > 0. \end{cases}\\
&e(\mathsf{R}) = 1, &&
\end{aligned}
$$

The product $\mathsf{dp}\colon \mathcal{D} \to \mathcal{D} \to \mathcal{D}$ over the digits is defined by

$$\forall d_1, d_2 \in \mathcal{D}, \qquad \mathsf{dp}\ d_1\ d_2 = r(e(d_1) \times e(d_2)),$$

where $\times$ is the ordinary product over the integers. This can be summarized in

dp	L	M	R
L	R	M	L
M	M	M	M
R	L	M	R

Definition 4.36 (dsp). A special product of digits and sequences is defined by

$$
\begin{aligned}
&\mathsf{dsp}\colon \mathcal{D} \to \mathcal{D}^\infty \to \mathcal{D}^\infty\ ,\\
&\mathsf{dsp}\ \mathsf{L}\ d = \mathsf{neg}(d),\\
&\mathsf{dsp}\ \mathsf{M}\ d = d_0,\\
&\mathsf{dsp}\ \mathsf{R}\ d = d,
\end{aligned}
$$

where $d_0 \in \mathcal{D}^\infty$ is some representation of the real number *zero*, say

$$d_0 = \mathsf{M}\colon \mathsf{M}\colon \mathsf{M}\colon \cdots.$$

Now we are ready to present the definition of mult (again) in Haskell style:

Definition 4.37 (mult). The function $\mathsf{mult}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$ is defined by

$$\mathsf{mult}\ (a_0 \colon a_1 \colon x)\ (b_0 \colon b_1 \colon y) = \mathsf{avg}\ p\ q,$$

where

$$\begin{cases} p = \mathsf{avg}\ p_1\ p_2, \\ p_1 = (\mathsf{dp}\ a_0\ b_1) \colon (\mathsf{avg}\ p_{11}\ p_{12}), \\ p_{11} = \mathsf{dsp}\ b_1\ x, \\ p_{12} = \mathsf{dsp}\ a_1\ y, \\ p_2 = \mathsf{avg}\ p_{21}\ p_{22}, \\ p_{21} = \mathsf{dsp}\ b_0\ x, \\ p_{22} = \mathsf{dsp}\ a_0\ y \end{cases}$$

and

$$\begin{cases} q = c_0 \colon c_1 \colon c_2 \colon (\mathsf{mult}\ x\ y), \\ c_0 = \mathsf{dp}\ a_0\ b_0, \\ c_1 = \mathsf{dp}\ a_1\ b_0, \\ c_2 = \mathsf{dp}\ a_1\ b_1. \end{cases}$$

Again—much like the case of avg—continuity is immediate:

Lemma 4.38 (Continuity of mult). $\mathsf{mult}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$ *is continuous*.

The difficulty with implementing multiplication is that one has to make sure the algorithm produces output once it is provided with enough input digits. The obvious implementations do not satisfy this property, whereas Plume's algorithm does. Let us consider the following two infinite sequences:

$$a = a_0 \colon a_1 \colon x \quad (x \in \mathcal{D}^\omega),$$

$$b = b_0 \colon b_1 \colon y \quad (y \in \mathcal{D}^\omega).$$

According to the definition of mult,

$$\mathsf{mult}\ a\ b\ =\ \mathsf{avg}\ p\ q,$$

where p is readily available as it makes no recursive calls to mult itself, and q is of the form

$$q = c_0 \colon c_1 \colon c_2 \colon (\mathsf{mult}\ x\ y),$$

in which c_0, c_1 and c_2 are expressed in terms of a_i's and b_j's ($i, j \in \{0, 1\}$), i.e. the first three digits of q can be worked out without any recursive calls made to mult.

As avg needs at most two digits from each of its arguments before generating at least one digit of the output, the above argument demonstrates that mult a b produces at least one digit of the output before making any recursive calls to mult again. This leads to:

Lemma 4.39 (mult is total). *The result of applying* mult *to representations of* total *real numbers is the representation of a* total *real number, i.e.*

$$\forall x, y \in \mathcal{D}^\omega, \qquad \mathsf{mult}\ x\ y\ \in \mathcal{D}^\omega.$$

The correctness of mult can be proved following the same line as that of avg. Therefore, we omit the details and merely present the following:

Definition 4.40 ($\otimes\colon \mathcal{J} \to \mathcal{J} \to \mathcal{J}$).

$$\forall [x_1, y_1], [x_2, y_2] \in \mathcal{J}, \qquad [x_1, y_1] \otimes [x_2, y_2] := [\min \mathrm{EP}, \max \mathrm{EP}],$$

where

$$\mathrm{EP} = \{x_1x_2, x_1y_2, y_1x_2, y_1y_2\}.$$

Lemma 4.41.

$$\forall a, b \in \mathcal{D}^\infty, \qquad to_{\mathcal{J}}(\mathsf{mult}\ a\ b) \sqsubseteq_{\mathcal{J}} (to_{\mathcal{J}}\ a) \otimes (to_{\mathcal{J}}\ b).$$

Combining Lemmata 4.39 and 4.41 one gets:

Corollary 4.42 (Correctness of mult). *For elements $a, b \in \mathcal{D}^\omega$, i.e. infinite sequences of digits*:

$$to_{\mathcal{J}}(\mathsf{mult}\ a\ b) = (to_{\mathcal{J}}\ a) \otimes (to_{\mathcal{J}}\ b).$$

Corollary 4.43 (Extensionality of mult). mult *preserves the equivalence relation $\simeq_{\mathcal{D}^\infty}$ of Definition* 4.1, *i.e. for all $d_x, e_x, d_y, e_y \in \mathcal{D}^\infty$,*

$$\left.\begin{array}{l} d_x \simeq_{\mathcal{D}^\infty} e_x \\ d_y \simeq_{\mathcal{D}^\infty} e_y \end{array}\right\} \quad \Rightarrow \quad (\mathsf{mult}\ d_x\ d_y) \simeq_{\mathcal{D}^\infty} (\mathsf{mult}\ e_x\ e_y).$$

Remark 4.44. We have chosen the implementation of multiplication worked out by Plume [1998] while there are other alternatives available as well. In fact, much earlier, Trivedi and Ercegovac [1977] presented implementations of multiplication and division in a slightly more general setting. However, it is interesting to note that their division algorithm fails for *signed-digit binary* format, though once the base is raised to anything above 2, it works fine.

4.6. abs: $\mathcal{D}^\infty \to \mathcal{D}^\infty$

Here we include a function to help us with the *absolute value* operation over real numbers. The reason we do this is merely to simplify Definition 4.65 of the is_Str_Cau function which will be part of the implementation of the limit function (Section 4.8). Of course,

limit could be implemented without explicit resort to abs, in which case—as we will see in Section 7—abs is definable from other constants of SHRAD. In that respect, abs is not an essential part of SHRAD, and we are not going to dwell on it too much. Needless to say that the absolute value operation is a much-used popular operation over real numbers—one prominent use of which is to get the distance between pairs of reals—thus we saw no harm in including it as a basic constant of the language.

We define the denotation of the constant *abs*—i.e. $\mathcal{A}_{\mathcal{SH}}(abs)$—as:

Definition 4.45 (abs: $\mathcal{D}^\infty \to \mathcal{D}^\infty$).

$$\begin{aligned}\mathsf{abs}\,(R\!:\!x) &= R\!:\!x,\\ \mathsf{abs}\,(M\!:\!x) &= M\!:\!(\mathsf{abs}\,(x)),\\ \mathsf{abs}\,(L\!:\!x) &= R\!:\!(\mathsf{neg}(x)),\end{aligned}$$

where neg: $\mathcal{D}^\infty \to \mathcal{D}^\infty$ is the negation function defined in Definition 4.26.

Lemma 4.46 (abs is total).

$$\forall x \in \mathcal{D}^\omega, \qquad \mathsf{abs}\,(x) \in \mathcal{D}^\omega.$$

Proof. Left to the reader. □

Lemma 4.47 (Correctness of abs). *For each $d \in \mathcal{D}^\infty$, let $[a_d.b_d] = to_{\mathcal{J}}(d)$. Then*

$$to_{\mathcal{J}}(\mathsf{abs}\,(d)) \sqsubseteq [r, s],$$

where

$$[r, s] = \begin{cases} [a_d, b_d] & \text{if} \quad 0 \le a_d,\\ [-b_d, -a_d] & \text{if} \quad b_d \le 0,\\ [0, b_d] & \text{if} \quad a_d \le 0 \le b_d.\end{cases}$$

Proof. Left to the reader. □

Corollary 4.48 (Extensionality of abs).

$$\forall x, y \in \mathcal{D}^\infty, \qquad x \simeq_{\mathcal{D}^\infty} y \Rightarrow \mathsf{abs}\,(x) \simeq_{\mathcal{D}^\infty} \mathsf{abs}\,(y).$$

4.7. isNeg: $\mathcal{D}^\infty \to \mathbb{B}_\perp$

We present the denotation of the only constant which is the link *from* continuous types *to* discrete ones. The major role this predicate is going to play for us—much like the function abs—is in simplifying Definition 4.65 of is_Str_Cau which is in turn part of the implementation of limit (Section 4.8). This time however, *isNeg* is an essential constant of SHRAD, as it would not be possible to define isNeg from other constants. Moreover,

although in our framework there is no possibility for defining *total* functions over real numbers by cases—as opposed to using pif_I in Real-PCF—it is still useful to have such a constant as part of the language. After all, now and again the user would like to compare the result of his calculations with some other value. Thus, for instance, to see if x is less than some value y, one can run the test

$$\mathsf{isNeg}\,((x - y)).$$

Therefore, we include this constant in our language.

Before coming to the denotation of *isNeg*, i.e. isNeg, we define two simple predicates. The first one checks to see if all the digits of its input are L, in which case it runs forever. Otherwise it terminates and returns *ff*:

Definition 4.49 (is_all_L: $\mathcal{D}^\infty \to \mathbb{B}_\perp$).

$$\mathsf{is\_all\_L}\,(\mathsf{L} : x) = \mathsf{is\_all\_L}\,(x),$$
$$\mathsf{is\_all\_L}\,(\mathsf{M} : x) = \mathit{ff},$$
$$\mathsf{is\_all\_L}\,(\mathsf{R} : x) = \mathit{ff}.$$

The second predicate is some sort of dual to the previous one, in that it runs forever if all the digits are R, otherwise terminates with value *tt*:

Definition 4.50 (not_all_R: $\mathcal{D}^\infty \to \mathbb{B}_\perp$).

$$\mathsf{not\_all\_R}\,(\mathsf{L}: x) = \mathit{tt},$$
$$\mathsf{not\_all\_R}\,(\mathsf{M}: x) = \mathit{tt},$$
$$\mathsf{not\_all\_R}\,(\mathsf{R}: x) = \mathsf{not\_all\_R}\,(x).$$

Having defined these two predicates, we present isNeg as follows:

Definition 4.51 (isNeg): $\mathcal{D}^\infty \to \mathbb{B}_\perp$.

$$\mathsf{isNeg}\,(\mathsf{L}: x) = \mathsf{not\_all\_R}\,(x),$$
$$\mathsf{isNeg}\,(\mathsf{M}: x) = \mathsf{isNeg}\,(x),$$
$$\mathsf{isNeg}\,(\mathsf{R}: x) = \mathsf{is\_all\_L}\,(x).$$

Unlike previous functions, isNeg *is not a total one*, and the sources of partiality are representations of 0. Thus, by abusing notation, we can formulate the following lemma:

Lemma 4.52 (isNeg is total over $[-1, 1]\backslash\{0\}$).

$$\forall x \in \mathcal{D}^\omega, \qquad \mathsf{isNeg}\,(x) \in \{\mathit{tt}, \mathit{ff}\} \iff \mathit{to}_{\mathcal{J}}(x) \neq 0.$$

Proof. Left to the reader. □

Lemma 4.53 (Correctness of isNeg). *For any $d \in \mathcal{D}^\infty$, let $[a_d, b_d] = to_{\mathcal{I}}(d)$. Thus,*

$$\mathsf{isNeg}\,(d) = \begin{cases} tt & \text{if} \quad b_d < 0, \\ ff & \text{if} \quad 0 < a_d, \\ \bot & \text{if} \quad a_d \leq 0 \leq b_d. \end{cases}$$

Proof. Left to the reader. □

Corollary 4.54 (Extensionality of isNeg).

$$\forall x, y \in \mathcal{D}^\infty, \qquad x \simeq_{\mathcal{D}^\infty} y \Rightarrow \mathsf{isNeg}\,(x) = \mathsf{isNeg}\,(y).$$

4.8. limit: $[\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathcal{D}^\infty$

In the realm of (classical) mathematical analysis, the class of all sequences over (say) $\mathbb{R}$ is partitioned into two subclasses: *divergent* and *convergent*, with the latter considered to be the relatively more tame and easy-to-handle one. Enter *computable* (or *intuitionistic*) analysis, and even this one appears to be too wild to handle.

Assume that we are to implement a general limit operator,

$$\mathsf{gen\_limit}\colon \mathbb{R}^{\mathbb{N}} \rightharpoonup \mathbb{R},$$

such that given *any* convergent sequence $s\colon \mathbb{N} \to \mathbb{R}$ we get

$$\mathsf{gen\_limit}\; s = \lim_{n\to\infty} s(n).$$

For each $k \in \mathbb{N}$, let $u_k\colon \mathbb{N} \to \mathbb{R}$ be the sequence defined by

$$\forall i \in \mathbb{N}, \qquad u_k(i) := \begin{cases} 1 & \text{if} \quad i \leq k, \\ 0 & \text{if} \quad i > k, \end{cases}$$

and define the sequence $u\colon \mathbb{N} \to \mathbb{R}$ by

$$\forall i \in \mathbb{N}, \qquad u(i) := 1.$$

Obviously, u together with all u_k's are convergent with

$$\forall k \in \mathbb{N}, \qquad \lim_{n\to\infty} u_k(n) = 0$$

and

$$\lim_{n\to\infty} u(n) = 1.$$

Hence, for gen_limit to work correctly, it has to output correct results over these sequences. If one fixes an approximation range, say

$$\varepsilon = \tfrac{1}{4}$$

and applies gen_limit over u, it must be possible to get an interval a_u of length not more than ε, which contains $\lim_{n\to\infty} u(n)$, i.e.

$$1 \in a_u$$

and the whole process must end in some finite time.

Assume that we are in the middle of the computation of (gen_limit u). At any stage, say after n steps, at most n terms of u are evaluated by gen_limit. Let k_0 be the greatest subscript of these terms. If no such term exists[8] then let $k_0 = 0$. The problem gen_limit faces at this stage is that there is no way in which it can be assured of the sequence it is dealing with—i.e. u—not being (say) u_{k_0}, whose immediate next term is 0, and whose limit is also 0. Thus, gen_limit is unable to output the expected interval a_u.

Remark 4.55. The tacit assumption that the flow of input/output information in time is *incremental* plays a crucial role in the previous argument. This means that once an interval is produced as a partial result, not only is the final result inside it, but all further refinements of the result also occur there.

The way out of this is, of course, to narrow the class of *all* convergent sequences to some smaller manageable subclass. There are different choices at hand, but the relevant ones all prove to be *equivalent*, in the sense that the model of the real numbers $\mathbb{R}$ and the function space $C(\mathbb{R})$ arising from each choice is the same.

Here we pick the set Cau_{2^n} of all the Cauchy sequences $s\colon \mathbb{N} \to \mathbb{R}$ such that

$$\forall m, n \in \mathbb{N}, \qquad n \leq m \Rightarrow |s(n) - s(m)| < 2^{-n}$$

and as up to now we have restricted the discussion to real numbers in $[-1, 1]$, the following definition makes the statement of some of the results more concise:

Definition 4.56 (Strong Cauchy Sequences: $\mathrm{Cau}_{2^n}(C)$). For any subset C of the real numbers $\mathbb{R}$ we let $\mathrm{Cau}_{2^n}(C)$ be the set of all Cauchy sequences $(r_n)_{n\in\mathbb{N}}$ such that:

1. $\forall n \in \mathbb{N}, r_n \in C$.
2. $\forall m, n \in \mathbb{N}, n \leq m \Rightarrow |r_n - r_m| < 2^{-n}$.

In order to demonstrate that $\mathrm{Cau}_{2^n}([-1, 1])$ is in fact the right choice, we embark on presenting the implementation of the function

$$\mathsf{limit}\colon [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathcal{D}^\infty,$$

where given any $(s_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1, 1])$ and any $d_s\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$ representing it, i.e.

$$\forall i \in \mathbb{N}, \qquad to_{\mathcal{J}}(d_s(i)) = s_i,$$

we get

$$to_{\mathcal{J}}(\mathsf{limit}\, d_s) = \lim_{n\to\infty} s_n.$$

As usual, we need a few definitions first:

Definition 4.57 ($\widetilde{\ }\colon \{\mathsf{L}, \mathsf{M}, \mathsf{R}\} \to \mathcal{J}$). For each digit $d \in \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ we define

$$\widetilde{d} = to_{\mathcal{J}}([d]).$$

[8] That is, gen_limit has not evaluated any of the terms of u so far.

Thus, we have

$$\widetilde{\mathsf{L}} = [-1, 0], \qquad \widetilde{\mathsf{M}} = [-\tfrac{1}{2}, \tfrac{1}{2}], \qquad \widetilde{\mathsf{R}} = [0, 1].$$

Definition 4.58 ($x_{<n}$). Let X be a set and let $x \in X^\infty$ be a (finite or infinite) sequence over it. For each natural number $n \in \mathbb{N}$, by $x_{<n}$ we mean the initial segment of x of length not more than n. In particular,

$$\mathit{length}(x) \leq n \iff x_{<n} = x.$$

Definition 4.59 ($\underline{x}, \overline{x}, m(x), \mu(x)$). Whenever x denotes an interval, by $\underline{x}, m(x), \overline{x}$ and $\mu(x)$ we mean the left endpoint, the middle point, the right endpoint and the length of that interval, respectively. In other words, we presume

$$\begin{aligned} \underline{x} &= \inf\{y \mid y \in x\}, \\ \overline{x} &= \sup\{y \mid y \in x\}, \\ m(x) &= (\underline{x} + \overline{x})/2, \\ \mu(x) &= \overline{x} - \underline{x}. \end{aligned}$$

Definition 4.60 ($\mathit{stretch}_{\mathcal{J}} \colon \mathcal{J} \to \mathbb{R} \to \mathcal{J}$). For any interval $a = [\underline{a}, \overline{a}] \in \mathcal{J}$ and any real number $r \in \mathbb{R}$ we define

$$\mathit{stretch}_{\mathcal{J}}\ a\ r := \begin{cases} [\max\{-1, \underline{a} - r\}, \min\{\overline{a} + r, 1\}] & \text{if} \quad 0 \leq r, \\ a & \mathit{otherwise}. \end{cases}$$

Proposition 4.61. *For any $d \in \mathcal{D}^\infty$:*

1. $\mu(\mathit{to}_{\mathcal{J}}(d)) > 0 \iff d \in \mathcal{D}^* \iff \mathit{length}(d) \in \mathbb{N}$, *in which case*
   $\mu(\mathit{to}_{\mathcal{J}}(d)) = 2^{1-\mathit{length}(d)}$,
   where $\mu(a)$ is the length of the interval a, as in Definition 4.59 *above.*
2. $\mu(\mathit{to}_{\mathcal{J}}\, d) = 0 \iff d \in \mathcal{D}^\omega \iff \mathit{length}(d) \notin \mathbb{N}$.

Therefore by allowing $2^{-\infty} = 0$, we can write

$$\forall d \in \mathcal{D}^\infty, \qquad \mu(\mathit{to}_{\mathcal{J}}(d)) = 2^{1-\mathit{length}(d)}.$$

Proof. Left to the reader. □

Consider any interval $a = [\underline{a}, \overline{a}] \in \mathcal{J}$ with $\mu(a) \leq \frac{1}{2}$. One can easily argue that:

1. $0 \leq \underline{a} \Rightarrow [0, 1] \sqsubseteq_{\mathcal{J}} a$.
2. $\overline{a} \leq 0 \Rightarrow [-1, 0] \sqsubseteq_{\mathcal{J}} a$.
3. otherwise $[-\frac{1}{2}, \frac{1}{2}] \sqsubseteq_{\mathcal{J}} a$.

In other words,

$$\forall a \in \mathcal{J}, \qquad \mu(a) \leq \tfrac{1}{2} \quad \Rightarrow \quad \exists X \in \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}, \qquad to_{\mathcal{J}}([X]) \sqsubseteq_{\mathcal{J}} a.$$

A generalization of this fact (to be expressed in the following proposition) is the essence of generating successive digits of the limit. First, we fix a notation:

Notation 4.62. Let X be a set, let $w \in X^*$ and $v \in X^\infty$ be two sequences over X. By

$$w +\!\!+ v$$

we mean the *concatenation* of w and v.

Proposition 4.63. *If $a \in \mathcal{J}$ and $d \in \mathcal{D}^*$ satisfy*

1. $(to_{\mathcal{J}}\, d) \sqsubseteq_{\mathcal{J}} a$,
2. $\mu(a) \leq 2^{-length(d)-1}$,

then,

$$\exists X \in \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}, \qquad to_{\mathcal{J}}\,(d +\!\!+ [X]) \sqsubseteq_{\mathcal{J}} a.$$

Proof. By induction over the length of d:

Base case $d = [\,]$. In this case $\mu(a) \leq 2^{-1}$ and we have already discussed this case in the argument preceding this proposition.

Inductive case. Suppose $d = d_0{:}\,d'$. We first observe that as $to_{\mathcal{J}}(d) \sqsubseteq a$ (and consequently $to_{\mathcal{J}}(d_0) \sqsubseteq a$) we get

$$a = \mathsf{Cons}^{\sim}_{d_0}\, \mathsf{Tail}^{\sim}_{d_0}\, a. \tag{3}$$

Now we let $a' = \mathsf{Tail}^{\sim}_{d_0}\, a$ and we see that

$$\begin{aligned} to_{\mathcal{J}}(d') &= to_{\mathcal{J}}({d_0}^{-1} d) \\ \text{(by Lemma 4.18)} \quad &\sqsubseteq \mathsf{Tail}^{\sim}_{d_0}(to_{\mathcal{J}}(d)) \\ (\mathsf{Tail}^{\sim}_{d_0} \text{ is monotone}) \quad &\sqsubseteq \mathsf{Tail}^{\sim}_{d_0}\, a \\ &= a'. \end{aligned}$$

On the other hand,

$$\begin{aligned} \mu(a') &= \mu(\mathsf{Tail}^{\sim}_{d_0}\, a) \\ &\leq 2 \times \mu(a) \\ &\leq 2 \times 2^{-length(d)-1} \\ &= 2^{-length(d)} \\ &= 2^{-(length(d')+1)} \\ &= 2^{-length(d')-1}. \end{aligned}$$

Now we are in the right position to apply the induction hypothesis to d' and a' in order to get the digit X which makes

$$to_{\mathcal{J}}(d' +\!\!+ [X]) \sqsubseteq a'.$$

Finally we see that

$$\begin{aligned}
to_{\mathcal{J}}(d +\!\!+ [X]) &= to_{\mathcal{J}}((d_0 \colon d') +\!\!+ [X]) \\
&= to_{\mathcal{J}}(d_0 \colon (d' +\!\!+ [X])) \\
\text{(definition of } to_{\mathcal{J}}) &= \mathsf{Cons}^{\sim}_{d_0}(to_{\mathcal{J}}(d' +\!\!+ [X])) \\
(\mathsf{Cons}^{\sim}_{d_0} \text{ is monotone}) &\sqsubseteq \mathsf{Cons}^{\sim}_{d_0}(a') \\
\text{(definition of } a') &= \mathsf{Cons}^{\sim}_{d_0}.\mathsf{Tail}^{\sim}_{d_0}(a) \\
\text{(equation (3))} &= a.
\end{aligned}$$

□

Take any $(r_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1,1])$ with $r = \lim_{n\to\infty} r_n$. As $[-1,1]$ is a closed subset of $\mathbb{R}$, we get $r \in [-1,1]$. Let $f \colon \mathbb{N}_\perp \to \mathcal{D}^\omega$ be *any* representation of $(r_n)_{n\in\mathbb{N}}$. We want to outline the process which leads to the computation of some $f_r \in \mathcal{D}^\omega$ representing the real number $r \in [-1,1]$.

To produce the first digit of f_r, by Proposition 4.63, we should look for an interval $a \in \mathcal{J}$ such that

1. $r \in a$,
2. $\mu(a) \leq \frac{1}{2}$,

and of course, all the information we have is the sequence f and the fact that $(r_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1,1])$. Yet, this information is enough to show that

1. $|r - r_3| \leq \frac{1}{8}$,
2. $r_3 \in to_{\mathcal{J}}\,(f(3)_{<3})$,
3. $\mu(to_{\mathcal{J}}\,(f(3)_{<3})) = 2^{1-3} = \frac{1}{4}$,

which leads to

$$r \in stretch_{\mathcal{J}}(to_{\mathcal{J}}(f(3)_{<3}))\tfrac{1}{8}$$

So, by considering $a_1 = stretch_{\mathcal{J}}(to_{\mathcal{J}}(f(3)_{<3}))\frac{1}{8}$ we get the interval we were looking for, as

$$(r \in a_1) \wedge (\mu(a_1) \leq (\tfrac{1}{4} + \tfrac{1}{8} + \tfrac{1}{8}) = \tfrac{1}{2})$$

This way, we can produce the first digit of f_r. The way to proceed follows the same basic method implemented recursively. So assume that at some stage, we have produced the first n digits of f_r, which we write as

$$(f_r)_{<n} = d_0 \cdots d_{n-1}.$$

To get the next digit d_n, all we need to do is to consider the interval $a_n \in \mathcal{J}$ defined by

$$a_n := b_n \cap (to_{\mathcal{J}}((f_r)_{<n})), \tag{4}$$

where

$$b_n = stretch_{\mathcal{J}}(to_{\mathcal{J}}(f(n+3)_{<n+3}))2^{-(n+3)},$$

as we have

1. $r \in to_{\mathcal{J}}((f_r)_{<n})$,
2. $|r - r_{n+3}| \leq 2^{-(n+3)}$,
3. $r_{n+3} \in to_{\mathcal{J}}\,(f(n+3)_{<n+3})$,
4. $\mu(to_{\mathcal{J}}\,(f(n+3)_{<n+3})) = 2^{1-(n+3)} = 2^{-(n+2)}$,

therefore, by 1–3,

$$r \in a_n$$

and, by 4,

$$\begin{aligned}\mu(a_n) &\leq \mu(b_n)\\ &= 2^{-(n+2)} + 2 \times 2^{-(n+3)}\\ &< 2^{-(n+1)}\\ &= 2^{-length((f_r)_{<n})-1}.\end{aligned}$$

Hence, by Proposition 4.63 we are able to produce the next digit $d_n \in \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ such that

$$to_{\mathcal{J}}\,((f_r)_{<n+1}) = to_{\mathcal{J}}\,(d_0 \cdots d_{n-1} d_n) \sqsubseteq_{\mathcal{J}} a_n.$$

Combined with the fact that $r \in a_n$ we get

$$r \in to_{\mathcal{J}}\,((f_r)_{<n+1}).$$

Thus we have gone through the algorithm which we are about to apply in defining the function limit: $[\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathcal{D}^\infty$ promised before. The representations of strong Cauchy sequences are just a fraction of the whole objects in the domain $[\mathbb{N}_\perp \to \mathcal{D}^\infty]$. Hence in implementing limit, alongside generating output digits, we run a *test* over successive terms of the input in order to ensure that the sequence is legitimate. The unfortunate fact is that this test is quite expensive; nonetheless, without it we lose one of the main characteristics of our framework, i.e. *extensionality*. In Example 6.7 we illustrate why this expensive test is inevitable and there we discuss the whole issue further.

According to the algorithm above, for each $n \geq 3$, the $(n-2)$nd digit of the limit—i.e. d_{n-3}—is generated based on the nth term of the sequence—i.e. f_n. Now for each $n > 3$, we first check

$$\begin{aligned}&| r_3 - r_n | < 2^{-3}\\ &\quad\vdots \qquad\quad \vdots\\ &| r_{n-1} - r_n | < 2^{-(n-1)}\end{aligned}$$

and embark on generating the digit d_{n-3}, only if all these $n-3$ inequalities hold.[9] For this we define the predicate

$$\mathsf{is\_Str\_Cau}\colon [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathbb{N}_\perp \to \mathbb{B}_\perp$$

such that given any function $f\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$ representing the sequence $(r_n)_{n\in\mathbb{N}}$, and any $n > 3$,

$$\mathsf{is\_Str\_Cau}\,(f)\,(n) = tt \iff \forall k \in \{3, \ldots, n-1\}, \quad |r_k - r_n| < 2^{-k},$$

which in turn uses the predicate

$$\mathsf{cau\_Test}\,(x, y, k) = |x - y| < 2^{-k},$$

the implementation of which necessitated the introduction of the constant *abs* in the language:

Definition 4.64 ($\mathsf{cau\_Test}\colon (\mathcal{D}^\infty, \mathcal{D}^\infty, \mathbb{N}_\perp) \to \mathbb{B}_\perp$). For all $d_1, d_2 \in \mathcal{D}^\infty$ and $k \in \mathbb{N}_\perp$,

$$\mathsf{cau\_Test}\,(d_1, d_2, k) = \mathsf{isNeg}\,(\mathsf{sub}\,(\mathsf{abs}\,(\mathsf{sub}\,(d_1)\,(d_2)))\,(\mathsf{qtoR}\,(1, 1, 2^k))),$$

where

$$\begin{cases} \mathsf{sub}\colon \mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty, \\ \mathsf{sub}\,(x)\,(y) = \mathsf{M}^{-1}\,(\mathsf{avg}\ x\ (\mathsf{mult}\ d_{-1}\ y)) \end{cases}$$

and $to_{\mathcal{J}}(d_{-1}) = -1$.

Definition 4.65 ($\mathsf{is\_Str\_Cau}\colon [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathbb{N}_\perp \to \mathbb{B}_\perp$).

$$\mathsf{is\_Str\_Cau}\,(f)\,(n) = \begin{cases} tt & \text{if} \quad n \leq 3, \\ \mathsf{aux\_Is\_Str\_Cau}\,(f)\,(3)\,(n) & \text{if} \quad n > 3, \end{cases}$$

where $\mathsf{aux\_Is\_Str\_Cau}\,(f)\,(k)\,(n)$ is

$$\begin{cases} tt & \text{if} \quad k = n, \\ \mathsf{aux\_Is\_Str\_Cau}\,(f)\,(k+1)\,(n) & \text{if} \quad \mathsf{cau\_Test}\,(f(k), f(n), k) = tt, \\ ff & \text{if} \quad \mathsf{cau\_Test}\,(f(k), f(n), k) = ff, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Going back to the implementation of limit, we first define a function $\mathsf{bcna}\colon \mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ such that given any $d \in \mathcal{D}^\infty$ for which a digit $X \in \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ exists satisfying

$$to_{\mathcal{J}}(X) \sqsubseteq stretch_{\mathcal{J}}\, to_{\mathcal{J}}(d)\tfrac{1}{8}$$

[9] It is worth emphasizing here that *inequality* over real numbers is semi-decidable, as opposed to *equality*.

we get

$$\mathsf{bcna}(d) = X.$$

One tentative definition could be

$$\mathsf{bcna}\colon \mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\},$$

$$\mathsf{bcna}\ d = \begin{cases} \mathsf{R} & \text{if} \quad \underline{a} \geq 0, \\ \mathsf{L} & \text{if} \quad \overline{a} \leq 0, \\ \mathsf{M} & \text{if} \quad (-\tfrac{1}{2} \leq \underline{a}) \wedge (\overline{a} \leq \tfrac{1}{2}), \\ \text{undefined} & \text{otherwise.} \end{cases}$$

where $[\underline{a}, \overline{a}] = \mathit{stretch}_{\mathcal{J}}(\mathit{to}_{\mathcal{J}}\ (d_{<3}))\frac{1}{8}$.

Although the definition works fine, we would rather not resort to rational numbers via $\mathit{to}_{\mathcal{J}}$ and $\mathit{stretch}_{\mathcal{J}}$. Instead we pretend to be fussy and present a purely symbolic definition for bcna using pattern matching, although it requires a long list of cases to be checked:

Definition 4.66 (bcna: $\mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$). See Table 4.

Next we define the function cna: $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$ such that given inner $\in \mathcal{D}^\infty$ and outer $:= d_0\, d_1 \cdots d_n \in \mathcal{D}^*$ with

1. $\mathit{length}(\mathsf{outer}) + 3 \leq \mathit{length}(\mathsf{inner})$,
2. $\mathit{to}_{\mathcal{J}}(\mathsf{outer}) \sqsubseteq \mathit{to}_{\mathcal{J}}(\mathsf{inner})$,

Table 4. The function bcna: $\mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$.

bcna	(L: L: x)	=	L
bcna	(L: M: x)	=	L
bcna	(L: R: L: x)	=	L
bcna	(L: R: M: x)	=	M
bcna	(L: R: R: x)	=	M
bcna	(M: L: L: x)	=	L
bcna	(M: L: M: x)	=	M
bcna	(M: L: R: x)	=	M
bcna	(M: M: x)	=	M
bcna	(M: R: L: x)	=	M
bcna	(M: R: M: x)	=	R
bcna	(M: R: R: x)	=	R
bcna	(R: L: L: x)	=	M
bcna	(R: L: M: x)	=	R
bcna	(R: L: R: x)	=	R
bcna	(R: M: x)	=	R
bcna	(R: R: x)	=	R

we get

$$to_{\mathcal{J}}(\mathsf{outer} +\!\!+ [X]) \sqsubseteq stretch_{\mathcal{J}}(to_{\mathcal{J}}(\mathsf{inner}))2^{-(n+3)},$$

where

1. $X = \mathsf{cna\ inner\ outer}$,
2. $n = length(\mathsf{outer})$.

Of course for practical purposes, we have to relax the condition:

$$to_{\mathcal{J}}(\mathsf{outer}) \sqsubseteq to_{\mathcal{J}}(\mathsf{inner})$$

and only stick to the first condition:

$$length(\mathsf{outer}) + 3 \leq length(\mathsf{inner}).$$

The reason is best observed by a second look at the definition of a_n (equation (4)) where parts of the interval

$$b_n = stretch_{\mathcal{J}}(to_{\mathcal{J}}(f(n+3)_{<n+3}))2^{-(n+3)}$$

might well lie outside the (nonetheless *overlapping*) interval

$$to_{\mathcal{J}}((f_r)_{<n}).$$

However, we are only interested in the overlapping segment containing the limit, i.e. r.

Definition 4.67 ($\mathsf{cna}: \mathcal{D}^\infty \to \mathcal{D}^\infty \to \{\mathsf{L}, \mathsf{M}, \mathsf{R}\}$).

$$\mathsf{cna\ inner\ outer} = \begin{cases} \mathsf{bcna\ inner} & \text{if} \quad length(\mathsf{outer}) = 0, \\ \mathsf{undefined} & \text{if} \quad length(\mathsf{inner}) = 0, \\ \mathsf{cna}(d_0^{-1}(\mathsf{inner}))\ \mathsf{outer}' & \text{otherwise}, \end{cases}$$

where $\mathsf{outer} = d_0 : \mathsf{outer}'$.

Now we can present our main function:

Definition 4.68 (limit).

$$\begin{cases} \mathsf{limit}: [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathcal{D}^\infty, \\ \mathsf{limit}\ f = \mathsf{aux\_limit}\ f 3[\,], \end{cases}$$

where the auxiliary function aux_limit is defined by

$$\mathsf{aux\_limit}: [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathbb{N}_\perp \to \mathcal{D}^\infty \to \mathcal{D}^\infty,$$

$$\mathsf{aux\_limit}\ f n a = \begin{cases} d: \ (\mathsf{aux\_limit}\ f(n+1)(a +\!\!+ [d])) & \text{if} \quad \mathsf{is\_Str\_Cau}\,(f)\,(n), \\ [\,] & \text{otherwise} \end{cases}$$

where $d = \mathsf{cna}(f(n)_{<n})a$.

We have already gone through the algorithm for limit, implicit in which is the proofs for the following lemmas:

Lemma 4.69 (limit is Total over Strong Cauchy Sequences). *For any* $f\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$,

$$\boxed{\mathsf{limit}\ f \in \mathcal{D}^\omega \iff f \text{ represents some strong Cauchy sequence.}}$$

Lemma 4.70. *Let* $f\colon \mathbb{N}_\perp \to \mathcal{D}^\omega$ *represent the strong Cauchy sequence*

$$(r_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1,1])$$

with $r = \lim_{n\to\infty} r_n \in [-1,1]$. *Then*

$$\forall n \in \mathbb{N}, \qquad r \in to_{\mathcal{J}}((\mathsf{limit}\ f)_{<n}).$$

Combining Lemmata 4.69 and 4.70 we get:

Corollary 4.71 (Correctness of limit). *Let* $f\colon \mathbb{N}_\perp \to \mathcal{D}^\omega$ *represent the strong Cauchy sequence* $(r_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1,1])$. *Then* (limit f) *represents the total real number* $\lim_{n\to\infty} r_n \in [-1,1]$, *i.e.*

$$(\forall i \in \mathbb{N}, \qquad to_{\mathcal{J}}(f(i)) = r_i) \Rightarrow \left(to_{\mathcal{J}}(\mathsf{limit}\ f) = \lim_{n\to\infty} r_n\right).$$

Extensionality of limit over strong Cauchy sequences is also guaranteed by Corollary 4.71:

Corollary 4.72 (Extensionality of limit over Strong Cauchy Sequences). *Let* $f, g\colon \mathbb{N} \to \mathcal{D}^\omega$ *be representations of the same strong Cauchy sequence*

$$(r_n)_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}([-1,1]).$$

Then

$$\mathsf{limit}\ f \simeq_{\mathcal{D}^\infty} \mathsf{limit}\ g.$$

5. Operational Semantics

We continue with our style of defining operational semantics via *immediate* reduction rules. The PCF fragment of SHRAD follows that of standard PCF as in Plotkin [1977]. As for the proper SHRAD terms, one needs to know how to recover digits by reduction. However, this can already be extracted from the denotational semantics in a straightforward

way. Take the constant $avg\colon\ r \to r \to r$ for instance. We have

$$[\![avg]\!] = \mathsf{avg}.$$

From Table 3 we learn

$$\mathsf{avg}(\mathsf{L} : x)\ (\mathsf{L} : y) = \mathsf{L} : (\mathsf{avg}\ x\ y).$$

Therefore one can postulate

$$avg(L{:}\, x)\ (L{:}\, y) \to L{:}\, (avg\ x\ y).$$

Other constants follow suit, though at times not as easy as that of *avg*. For instance we already know what a lengthy path one has to take in order to generate one digit out of a limit computation. Nonetheless, the principle remains the same and we skip the details.

6. Extensionality

By now, not only have we given the precise definition of what it means for a function over real numbers to be represented by a SHRAD-definable object via $to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}}$ of Definition 4.3, but also we have a good intuition of the concept as a result of Section 4. Here, as we will be dealing with the subject exclusively, perhaps a more handy terminology and notation is not a bad idea. Thus we define:

Definition 6.1.

1. A partial function $g\colon [-1, 1]^m \rightharpoonup [-1, 1]$ is said to be SHRAD-*definable* if it is representable by some SHRAD-definable $\tilde{g}\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$, i.e.

   $$to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}}(\tilde{g}) = g.$$

2. A function $f\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$ is said to *represent* a function over real numbers if some $\hat{f}\colon [-1, 1]^m \rightharpoonup [-1, 1]$ exists for which

   $$to_{\mathcal{J}_{\max}^m \to \mathcal{J}_{\max}}(f) = \hat{f}.$$

Take any SHRAD-definable $f\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$. We demonstrate that for any two $\boldsymbol{d}, \boldsymbol{e} \in (\mathcal{D}^\omega)^m$ representing the same m-tuple of real numbers, i.e.

$$\forall i \in \{1, \ldots, m\}, \qquad to_{\mathcal{J}}(d_i) = to_{\mathcal{J}}(e_i),$$

it is *impossible* for all the following to hold at the same time:

(i) $f(\boldsymbol{d}) \in \mathcal{D}^\omega$.
(ii) $f(\boldsymbol{e}) \in \mathcal{D}^\omega$.
(iii) $to_{\mathcal{J}}(f(\boldsymbol{d})) \neq to_{\mathcal{J}}(f(\boldsymbol{d}))$.

We simply refer to this property of the (functions definable in the) language as *extensionality*.

To prove extensionality for SHRAD, we resort to logical relations.

6.1. *The Logical Relation $\mathcal{X}$*

Definition 6.2 (Binary Logical Relation $\mathcal{X}$). Let $\bigcup\{\mathbb{D}_\sigma \mid \sigma \text{ a SHRAD-type}\}$ be the collection of domains for SHRAD as in Definition 3.9. The binary logical relation X is defined over the pairs of objects in this model by:

1. For $o \in \{bool, nat\}$,

   $$\forall x, y \in \mathbb{D}_o, \qquad x \,\mathcal{X}^o\, y \iff x \uparrow y,$$

2. $\forall d, e \in \mathcal{D}^\infty, d \,\mathcal{X}^r\, e \iff to_{\mathcal{J}}(d) \uparrow to_{\mathcal{J}}(e)$,

where $a \uparrow b$ stands for "a and b are consistent". The logical relation X is built upon the above ground type cases.

Proposition 6.3. *Over the ground types $o \in \{bool, nat\}$, we can rephrase the logical relation as*:

1. $\forall x, y \in \mathbb{D}_o, \qquad x \,\mathcal{X}^o\, y \iff (x \sqsubseteq y) \vee (y \sqsubseteq x)$,
2. $\forall x, y \in \mathbb{D}_o, \qquad x \,\mathcal{X}^o\, y \iff (x = y) \vee (x = \bot) \vee (y = \bot)$.

Proof. Easy. □

Proposition 6.4. *Over the ground type r*:

$$\forall x, y \in \mathcal{D}^\infty, \qquad x \,\mathcal{X}^r\, y \Longleftrightarrow to_{\mathcal{J}}(x) \cap to_{\mathcal{J}}(y) \neq \emptyset.$$

Proof. Easy. □

Thus we freely use any of the equivalent reformulations of X over ground types as expressed in the previous couple of propositions.

Next we demonstrate that X is a C-logical relation, where C is the set of SHRAD-constants:

1. For constants inherited from PCF it is straightforward to show that they preserve X. Hence we skip the details.
2. For proper SHRAD-constants (except *limit* and Y_σ's) we have already established the property in Section 4. For instance, by Lemma 4.32, *avg* preserves consistency. The cases for other such constants are similarly dealt with, which we summarize in Table 5.
3. The fix-point constants Y_σ's and *limit* are the ones that require further attention. Thus in what follows we focus on these constants.

6.1.1. *The Fix-Point Constants*. In Farjudian [2004a] we show how fix-point constants preserve certain logical relations. The technique is similar here and we do not got into the details. Again the interesting bit is to demonstrate that Y_r is invariant under X, which in turn boils down to:

Table 5. $\mathcal{X}$ and SHRAD-constants.

Constant	Corresponding statement
L, M, R	Proposition 4.5
L^{-1}, M^{-1}, R^{-1}	Lemma 4.18
qtoR	Lemma 4.23
avg	Lemma 4.32
mult	Lemma 4.41
abs	Lemma 4.47
isNeg	Lemma 4.53

Proposition 6.5. *The set of $\mathcal{X}^r$ invariant elements forms an* inclusive predicate *in $\mathcal{D}^\infty \times \mathcal{D}^\infty$.*

Proof.

1. $[-1,1] \cap [-1,1] \neq \emptyset \quad \Rightarrow to_{\mathcal{J}}([\,]) \uparrow to_{\mathcal{J}}([\,])$
   $\Rightarrow \bot_{\mathcal{D}^\infty} \mathcal{X}^r \bot_{\mathcal{D}^\infty}$.
2. Let $\{d_i \mid i \in \mathbb{N}\}$ and $\{e_j \mid j \in \mathbb{N}\}$ be two ascending chains in $\mathcal{D}^\infty$ such that

   $\forall i \in \mathbb{N}, \qquad d_i \,\mathcal{X}^r\, e_i$.

   Note that as $to_{\mathcal{J}}$ is monotone, $\{to_{\mathcal{J}}(d_i) \mid i \in \mathbb{N}\}$ and $\{to_{\mathcal{J}}(e_i) \mid i \in \mathbb{N}\}$ are two ascending chains in $\mathcal{J}$ as well. Fixing any $i \in \mathbb{N}$ we observe that:
   (a) As $\{to_{\mathcal{J}}(e_j) \mid j \in \mathbb{N}\}$ is ascending:

   $\forall j \leq i, \qquad to_{\mathcal{J}}(e_j) \sqsubseteq to_{\mathcal{J}}(e_i)$,

   which combined with the fact that $d_i \,\mathcal{X}^r\, e_i$ results in

   $\forall j \leq i, \qquad d_i \,\mathcal{X}^r\, e_j$.

   (b) As $\{to_{\mathcal{J}}(d_j) \mid j \in \mathbb{N}\}$ is ascending:

   $\forall j \geq i, \qquad to_{\mathcal{J}}(d_i) \sqsubseteq to_{\mathcal{J}}(d_j)$,

   which combined with the fact that $d_j \,\mathcal{X}^r\, e_j$ results in

   $\forall j \geq i, \qquad d_i \,\mathcal{X}^r\, e_j$.

   From (a) and (b) it follows that for each $i \in \mathbb{N}$,

   $\{to_{\mathcal{J}}(d_i) \cap to_{\mathcal{J}}(e_j) \mid j \in \mathbb{N}\}$

   forms an ascending chain of non-empty compact intervals in $\mathcal{J}$, which has to have a non-empty intersection. Thus,

   $$to_{\mathcal{J}}(d_i) \cap (\cap\{to_{\mathcal{J}}(e_j) \mid j \in \mathbb{N}\}) = \cap\{to_{\mathcal{J}}(d_i) \cap to_{\mathcal{J}}(e_j) \mid j \in \mathbb{N}\} \neq \emptyset.$$

   Now let us fix the non-empty interval

   $a = \bigcap\{to_{\mathcal{J}}(e_j) \mid j \in \mathbb{N}\}$.

By the preceding argument, for any $i \in N$,

$to_{\mathcal{J}}(d_i) \cap a$

is non-empty. As a result $\{to_{\mathcal{J}}(d_i) \cap a \mid i \in \mathbb{N}\}$ forms an ascending chain of non-empty compact intervals in $\mathcal{J}$, which again has to have a non-empty intersection. Therefore

$$\begin{aligned} to_{\mathcal{J}}(\sqcup\{d_i \mid i \in \mathbb{N}\}) \cap to_{\mathcal{J}}(\sqcup\{e_j \mid j \in \mathbb{N}\}) &= (\sqcup\{to_{\mathcal{J}}(d_i) \mid i \in \mathbb{N}\}) \\ &\quad \cap to_{\mathcal{J}}(\sqcup\{e_j \mid j \in \mathbb{N}\}) \\ (\text{as } a = to_{\mathcal{J}}(\sqcup\{e_j \mid j \in \mathbb{N}\})) &= \sqcup\{to_{\mathcal{J}}(d_i) \cap a \mid i \in \mathbb{N}\} \\ (\text{preceding argument}) &\neq \emptyset. \end{aligned}$$

□

6.1.2. *The Constant limit*. Here we demonstrate how *limit* preserves the logical relation X. The argument we follow is based on the algorithm presented in Section 4.8 and we assume familiarity with that material throughout, without repeated cross-referencing.

Let $f, g\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$ satisfy

$f\ \mathcal{X}^{nat\to r}\ g,$

which in particular implies

$\forall n \in \mathbb{N}\colon \quad to_{\mathcal{J}}(f(n)) \cap to_{\mathcal{J}}(g(n)) \neq \emptyset.$

We must prove that $(\mathsf{limit}\, f)\ \mathcal{X}^r\ (\mathsf{limit}\, g)$. For this we prove the following statement:

$\forall i, j \in \mathbb{N}, \qquad to_{\mathcal{J}}((\mathsf{limit}\, f)_{<i}) \cap to_{\mathcal{J}}((\mathsf{limit}\, g)_{<j}) \neq \emptyset,$

which suffices to lead to

$to_{\mathcal{J}}(\mathsf{limit}\, f) \cap to_{\mathcal{J}}(\mathsf{limit}\, g) \neq \emptyset.$

Let us first fix a couple of notations and assume

$\mathsf{limit}\, f = a_0 : a_1 : a_2 : \cdots,$
$\mathsf{limit}\, g = b_0 : b_1 : b_2 : \cdots.$

Now in case either of $\mathsf{limit}\, f$ or $\mathsf{limit}\, g$ produces no digits, the result follows trivially. Therefore we assume that a_0 and b_0 have been output. We know that:

(i) $to_{\mathcal{J}}([a_0]) \sqsubseteq to_{\mathcal{J}}(f(3))$.
(ii) $to_{\mathcal{J}}([b_0]) \sqsubseteq to_{\mathcal{J}}(g(3))$.
(iii) $to_{\mathcal{J}}(f(3)) \cap to_{\mathcal{J}}(g(3)) \neq \emptyset$.

If both $\mathsf{limit}\, f$ and $\mathsf{limit}\, g$ stop generating any further digits we have nothing more to talk about. On the other hand (without loss of generality) let $\mathsf{limit}\, g$ generate a further digit b_1 (Figure 1). This necessitates the following condition has to hold:

$\mathsf{is\_Str\_Cau}\ (g)\ (4) = tt,$

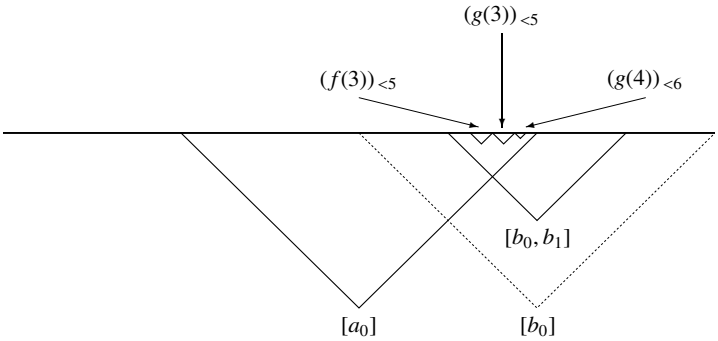

Fig. 1. Extensionality of the limit. (Note: in order to save space in this figure by $d \in \mathcal{D}^\infty$ we mean $to_{\mathcal{J}}(d) \in \mathcal{J}$.)

which in turn implies

$$\mu\,(to_{\mathcal{J}}(g(3)) \sqcap to_{\mathcal{J}}(g(4))) < \tfrac{1}{8} \tag{5}$$

(μ is defined in Definition 4.59).

Equation (5) above simply says that $to_{\mathcal{J}}(g(3))$ and $to_{\mathcal{J}}(g(4))$ both lie inside an open interval of length $\frac{1}{8}$. This in particular requires $g(3)$ (and $g(4)$ alike) to be defined up to at least five terms.

Going back to cna[10] of Definition 4.67 we know that a_0 is produced in such a way that the interval $to_{\mathcal{J}}([a_0])$ contains not only $to_{\mathcal{J}}(f(3))$, but also a $\frac{1}{8}$ neighbourhood of it too. This, combined with

(a) $to_{\mathcal{J}}(f(3)) \cap to_{\mathcal{J}}(g(3)) \neq \emptyset$,
(b) $\mu(to_{\mathcal{J}}(g(3)) \sqcap to_{\mathcal{J}}(g(4))) < \frac{1}{8}$,

implies

$$to_{\mathcal{J}}([a_0]) \sqsubseteq to_{\mathcal{J}}(g(4)). \tag{6}$$

Moreover, once the test

$$\mathsf{is\_Str\_Cau}\,(g)\,(4) = tt$$

is passed, b_1 is output in such a way that

$$to_{\mathcal{J}}([b_0, b_1]) \sqsubseteq to_{\mathcal{J}}(g(4)). \tag{7}$$

Equations (6) and (7) above imply

$$to_{\mathcal{J}}([a_0]) \cap to_{\mathcal{J}}([b_0, b_1]) \neq \emptyset,$$

hence $(\mathsf{limit}\, f)_{<1}\, \mathcal{X}^r\, (\mathsf{limit}\, g)_{<2}$.

The preceding argument can be extended to the general case in a straightforward manner:

[10] In this particular case in fact bcna of Definition 4.66.

Proposition 6.6. *Let* $f, g\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$ *satisfy* $f\ \mathcal{X}^{nat\to r}\ g$. *Then*

$$\forall i, j \in \mathbb{N}, \qquad to_{\mathcal{J}}\ ((\mathsf{limit}\ f)_{<i}) \cap to_{\mathcal{J}}\left((\mathsf{limit}\ g)_{<j}\right) \neq \emptyset.$$

Proof (Induction). We have already established the case for $i + j = 3$. Now assume that for some $i, j \in \mathbb{N}$ we have

$$to_{\mathcal{J}}((\mathsf{limit}\ f)_{<i}) \cap to_{\mathcal{J}}((\mathsf{limit}\ g)_{<j}) \neq \emptyset.$$

If both limit f and limit g stop generating any further digits then we are done. Otherwise (without loss of generality) let limit g output the next digit b_j. This requires

$$\mathsf{is\_Str\_Cau}\ (g)\ (j+3) = tt$$

to hold. Let a_{i_0} be the last digit of $(\mathsf{limit}\ f)_{<i}$.[11] Then we have:

1. $f\ \mathcal{X}^{nat\to r}\ g \Rightarrow f(i_0+3)\ \mathcal{X}^r\ g(i_0+3)$
   $\Rightarrow to_{\mathcal{J}}(f(i_0+3)) \cap to_{\mathcal{J}}(g(i_0+3)) \neq \emptyset.$
2. $\mathsf{is\_Str\_Cau}\ (g)\ (j+3) = tt$
   $\Rightarrow\ \mu\ (to_{\mathcal{J}}(g(i_0+3)) \sqcap to_{\mathcal{J}}(g(j+3))) < 2^{-(i_0+3)}.$

Combined with the fact that $to_{\mathcal{J}}\ ((\mathsf{limit}\ f)_{<i})$ contains not only $to_{\mathcal{J}}(f(i_0+3))$ but also the $2^{-(i_0+3)}$ neighbourhood of it as well, we get

$$to_{\mathcal{J}}((\mathsf{limit}\ f)_{<i}) \sqsubseteq to_{\mathcal{J}}(g(j+3)). \tag{8}$$

On the other hand once b_j is generated, we are assured of

$$to_{\mathcal{J}}((\mathsf{limit}\ g)_{<j+1}) \sqsubseteq to_{\mathcal{J}}(g(j+3)). \tag{9}$$

Equations (8) and (9) result in

$$to_{\mathcal{J}}\ ((\mathsf{limit}\ f)_{<i}) \cap to_{\mathcal{J}}\left((\mathsf{limit}\ g)_{<j+1}\right) \neq \emptyset. \qquad \square$$

6.1.3. *Extensionality and the Necessity of* is_Str_Cau. In retaining the extensionality of the whole framework, we had to hang on to the expensive is_Str_Cau predicate which definitely takes its toll on the efficiency of the computations. For the moment assume that there were no such predicate intermingled into the implementation of limit. So let limit′ be the variant of limit in which the expensive predicate is not employed. To obtain limit′ all one needs to do is to replace aux_limit of Definition 4.68 by aux_limit′ defined as follows:

$$\mathsf{aux\_limit'}\colon [\mathbb{N}_\perp \to \mathcal{D}^\infty] \to \mathbb{N}_\perp \to \mathcal{D}^\infty \to \mathcal{D}^\infty,$$
$$\mathsf{aux\_limit'}\ f\ n\ a = d\colon (\mathsf{aux\_limit'}\ f\ (n+1)\ (a +\!\!+ [d]))$$
$$\text{where} \quad d = \mathsf{cna}\ (f(n)_{<n})\ a.$$

[11] Note that if $(\mathsf{limit}\ f)_{<i}$ has not stopped outputting digits yet then $i_0 = i - 1$.

Under this condition, we manufacture a computation scenario which results in non-extensionality. Remember from Definition 4.9 that d_{-1} and d_1 are representation of -1 and 1, respectively.

Example 6.7. Consider $f, g\colon \mathbb{N}_\perp \to \mathcal{D}^\infty$ defined by:

1. $\forall n \leq 3$,
   - $f(n) = \mathsf{M} : \mathsf{M} : d_{-1}$,
   - $g(n) = \mathsf{L} : \mathsf{M} : d_1$.
2. $\forall n > 3$, $f(n) = g(n) = \mathsf{M} : \mathsf{M} : d_1$.

Note that

$$\begin{aligned} &\forall n \leq 3 \qquad to_{\mathcal{J}}(f(n)) = to_{\mathcal{J}}(g(n)) = -\tfrac{1}{4},\\ &\forall n > 3 \qquad to_{\mathcal{J}}(f(n)) = to_{\mathcal{J}}(g(n)) = \tfrac{1}{4}. \end{aligned}$$

Thus in particular $f\ \mathcal{X}^{nat\to r}\ g$. On the other hand it is not difficult to check that without is_Str_Cau, we get

$$\begin{cases} \mathsf{limit}'\ f = \mathsf{M} : \mathsf{R} : d_0,\\ \mathsf{limit}'\ g = \mathsf{L} : d_1, \end{cases}$$

where $d_0 = \mathsf{M} : \mathsf{M} : \mathsf{M} : \cdots$, which means

$$\begin{cases} to_{\mathcal{J}}(\mathsf{limit}'\ f) = \tfrac{1}{4},\\ to_{\mathcal{J}}(\mathsf{limit}'\ g) = 0. \end{cases}$$

Therefore without is_Str_Cau the extensionality is lost. In this particular case, if we apply limit instead of limit′ we get

$$\begin{cases} \mathsf{limit}\ f = [\mathsf{M}],\\ \mathsf{limit}\ g = [\mathsf{L}], \end{cases}$$

hence

$$\mathsf{limit}\ f\ \mathcal{X}^r\ \mathsf{limit}\ g.$$

The reason is that neither f nor g represent a legitimate strongly Cauchy sequence as we have

$$\begin{cases} |to_{\mathcal{J}}(f(3)) - to_{\mathcal{J}}(f(4))| > 2^{-2},\\ |to_{\mathcal{J}}(g(3)) - to_{\mathcal{J}}(g(4))| > 2^{-2}, \end{cases}$$

so according to our implementation of limit, after the first digit of the limit is generated, f and g fail to pass the strong Cauchyness test, i.e. is_Str_Cau, and the process comes to a halt.

Of course it all comes back to careless programming. So in case the programmer is sensible enough to implement *only* strongly Cauchy sequences, this expensive test can be omitted from the actual implementation of SHRAD in order to raise the efficiency.

6.2. *Extensionality at First-Order*

The material presented in Section 6.1 can be summarized in:

Proposition 6.8. *X is a C-logical relation, where C is the set of* SHRAD*-constants.*

Combined with the fundamental lemma of logical relations one gets:

Corollary 6.9. *For any* SHRAD*-term* $M : \sigma$,

$[\![M]\!]\, \mathcal{X}^\sigma\, [\![M]\!]$.

The question arises:

What has this property got to do with extensionality?

Assume that a function $f\colon \mathcal{D}^\infty \to \mathcal{D}^\infty$ is defined in SHRAD, representing some $\hat{f}\colon [-1, 1] \rightharpoonup [-1, 1]$. Then take some $x \in [-1, 1]$ together with one of its representations $d_x \in \mathcal{D}^\omega$ and finally suppose that $f(d_x)$ represents a *total* real number, i.e. $f(d_x)$ is an infinite sequence. Now in case $e_x \in \mathcal{D}^\omega$ is any (other) representation of x, by Corollary 6.9 above we are assured that:

1. If $f(e_x)$ is an *infinite* sequence, then it represents the same total real number as $f(d_x)$ does, i.e.

   $to_{\mathcal{J}}(f(e_x)) = to_{\mathcal{J}}(f(d_x))$.

2. If $f(e_x)$ is only a *finite* sequence, then

   $to_{\mathcal{J}}(f(e_x)) \sqsubseteq to_{\mathcal{J}}(f(d_x))$.

As a result:

> It is *impossible* for *different* representations of the same total real number as input to result in representations of *different* total real numbers as output.

However, does the fact that $f(d_x)$ is an infinite sequence automatically guarantee $f(e_x)$ to be infinite too? The answer is "no". It is still possible to write terms (say) f violating this property, in which case—by definition of $to_{\mathcal{J}^m_{\max} \to \mathcal{J}_{\max}}$ (Definition 4.3)—we rule out x as being an element of $dom(\hat{f})$. This is because such a case arises when f is "carelessly" implemented. To clarify this issue, we make the following further distinction:

Definition 6.10 (Strong versus Weak Extensionality).

- Let $m \geq 1$ and $f\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$. Then $\hat{f}\colon [-1, 1]^m \rightharpoonup [-1, 1]$ (or f itself for that matter) is said to be
  1. *strongly extensional* at $\boldsymbol{x} \in [-1, 1]^m$ if for any two $\boldsymbol{d_x}, \boldsymbol{e_x} \in (\mathcal{D}^\omega)^m$ representing $\boldsymbol{x}$,

     $to_{\mathcal{J}}(f(\boldsymbol{d_x})) = to_{\mathcal{J}}(f(\boldsymbol{e_x})) \in [-1, 1]$,

2. *weakly extensional* at $\boldsymbol{x} \in [-1, 1]^m$ if for any two $\boldsymbol{d_x}, \boldsymbol{e_x} \in (\mathcal{D}^\omega)^m$ representing $\boldsymbol{x}$,

   $to_{\mathcal{J}}(f(\boldsymbol{d_x})) \cap to_{\mathcal{J}}(f(\boldsymbol{e_x})) \neq \emptyset$.

- $\hat{f}$ (or f) is *strongly* (*weakly*) *extensional* if it is strongly (weakly) extensional over all $\boldsymbol{x} \in [-1, 1]^m$.

Proposition 6.11. *Strong extensionality implies the weak one* (*pointwise and overall alike*).

Proof. Follows from the fact that

$$to_{\mathcal{J}}(f(\boldsymbol{d_x})) = to_{\mathcal{J}}(f(\boldsymbol{e_x})) \quad \Rightarrow \quad to_{\mathcal{J}}(f(\boldsymbol{d_x})) \cap to_{\mathcal{J}}(f(\boldsymbol{e_x})) \neq \emptyset. \qquad \square$$

Definition 6.12. For any type σ and each $m \geq 0$, the type expression $\sigma^m \to \sigma$ is defined recursively as

$$\sigma^0 \to \sigma = \sigma,$$
$$\sigma^{(m+1)} \to \sigma = \sigma \to (\sigma^m \to \sigma).$$

As a direct consequence of Corollary 6.9, the main extensionality property of SHRAD is stated as:

Theorem 6.13 (Weak Extensionality). *Let $m \geq 1$ and $f\colon r^m \to r$ be any term written in* SHRAD. *Then $[\![f]\!]\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$ is weakly extensional.*

In Section 4 we demonstrated that all constants preserve $\simeq_{\mathcal{D}^\infty}$ of Definition 4.1, except for *limit* and the Y_σ's. In Section 8 we will introduce primitive recursion which for our purposes can be used instead of fix-point constants. On the other hand in Corollary 4.72 we showed that in case the input to *limit* represents a strongly Cauchy sequence even *limit* preserves $\simeq_{\mathcal{D}^\infty}$. Therefore as long as this last condition is cautiously cared for, strong extensionality is guaranteed. Of course, the problem here is that this property does not fully characterize strong extensionality and, after all, *it is not a syntactic characterization*. Yet from a certain viewpoint it can make sense. In Section 9 we will explain how to write a function over real numbers by virtue of the *effective Weierstraß theorem*.[12] According to that procedure, each function is written either as a polynomial or the limit of a (suitably fast converging) sequence of polynomials. Once a term (say) $f\colon r^m \to r$ is written according to that procedure, strong extensionality (of $[\![f]\!]\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$) is guaranteed.

Practice aside, the importance of weak extensionality is best seen in the theoretical side. In designing the language, by denying the user access to individual terms of the

[12] Theorem 7.8.

sequences (in $\mathcal{D}^\infty$), we have succeeded in obtaining weak extensionality. This is in contrast to other works based on (variations of) signed-digit binary representation available in the literature, such as Di Gianantonio [1993], Ménissier-Morain [1996] and Böhm et al. [1986].

7. Expressivity

In designing a framework for computation involving a language and its model, expressivity is one of the most important features. Naturally we like to be able to define more and more objects. Yet it should come as no surprise that more does not always mean better. For instance, take PCF as in Plotkin [1977]. The nagging problem of full abstraction leads to the augmentation of the language with parallel-or. Thus the cpo model becomes fully abstract with respect to PCF + {*por*} which is more expressive than PCF, nonetheless at the cost of embracing parallelism into the (up-to-then) sequential setting.

Now consider Weihrauch's approach [Weihrauch, 2000] to computability over real numbers, the so-called *Type*-2 *Theory of Effectivity* (TTE). We take TTE as the standard framework for computability over real numbers as—relative to other approaches—it scores high on both *expressivity* and *effectivity*. In fact a Type-2 machine has been realized [Weihrauch, 2000, Figure 2.2, page 16], confirming the realistic nature of TTE. Here we do not got into the details of the subject and content ourselves with the following brief description.

Although the framework admits various representations of real numbers (resulting in different notions of computability), for our purposes it suffices to consider the one with real numbers being represented by shrinking infinite sequences of intervals with rational endpoints (rational intervals). Now assume that for some suitable alphabet Σ and with some intermediate encoding, the set Σ^ω of all infinite sequences over Σ represents the set of all shrinking sequences of rational intervals which in turn represent real numbers. Thus one can define a representation $\nu\colon\ \Sigma^\omega \to \mathbb{R}$. This way we have functions over $\mathbb{R}$ on one hand, and *Type*-2 *machines* dealing with Σ^ω on the other hand. We do not present the definition of these machines here[13] as a mere intuition of computable string functions would suffice. The basic idea is that $f\colon\ \mathbb{R} \to \mathbb{R}$ is computable iff a string function F exists such that

$$\begin{array}{ccc} \Sigma^\omega & \xrightarrow{F} & \Sigma^\omega \\ {\scriptstyle\nu}\downarrow & & \downarrow{\scriptstyle\nu} \\ \mathbb{R} & \xrightarrow[f]{} & \mathbb{R} \end{array}$$

commutes. The definition for $f\colon\ \mathbb{R}^n \to \mathbb{R}$ with $(n > 1)$ is similar. This way a rich library of functions over real numbers is obtained containing all the "standard" computable functions such as addition, multiplication, division, exponentiation, etc.

[13] See Section 2.1 of Weihrauch [2000].

The problem with this framework is that there are string functions which do not realize any functions over real numbers. Moreover, TTE operates at the lowest possible level and offers no language at all. Therefore, in this document, we thought of selecting a number of functions as primitives and aiming to control somewhat the set of string functions definable while retaining a good deal of expressivity. We have already shown which primitives are selected (Definition 2.2) and have presented weak extensionality (Theorem 6.13) as the measure of control over definable string functions. Now, let us see how much expressivity we have managed to retain.

7.1. *Effective Weierstraß Theorem*

Notation 7.1. For any $x \in \mathbb{R}$, by $|x|$ we mean the *absolute value* of x, i.e.

$$|x| = \begin{cases} x & \text{if} \quad x \geq 0, \\ -x & \text{if} \quad x \leq 0. \end{cases}$$

Definition 7.2 (Maximum Norm). The function $\| \cdot \|_n \colon \mathbb{R}^n \to \mathbb{R}$ is defined by

$$\forall (x_1, \ldots, x_n) \in \mathbb{R}^n, \qquad \| (x_1, \ldots, x_n) \|_n = \max\{|x_1|, \ldots, |x_n|\}.$$

The subscript n may be dropped when it is easily understood from the context.

Let $n \geq 1$ and take some compact $X \subseteq \mathbb{R}^n$. It is true—both classically and constructively—that any continuous $f : X \to \mathbb{R}$ is uniformly continuous, i.e.

$$\forall \varepsilon > 0, \quad \exists \delta > 0, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \qquad \| \boldsymbol{x} - \boldsymbol{y} \| < \delta \Rightarrow |f(\boldsymbol{x}) - f(\boldsymbol{y})| < \varepsilon.$$

Thus whenever we talk about a continuous $f \colon X \to \mathbb{R}$, uniformity is implicitly implied as well.

Definition 7.3 ($C(X, Y)$). Let $X \subseteq \mathbb{R}^n$ be compact and let $Y \subseteq \mathbb{R}$ be arbitrary. We define

$$C(X, Y) = \{f \colon X \to Y \mid f \text{ is continuous}\}.$$

As we mostly deal with $[-1, 1]$, whenever $Y = [-1, 1]$ we simply drop it. Thus in particular, $C([-1, 1])$ is the set of all continuous $f \colon [-1, 1] \to [-1, 1]$.

For any compact $X \subseteq \mathbb{R}^n$, the set $C(X, \mathbb{R})$ contains functions of different shape. However, the whole set is somewhat under control, and that is due to the presence of a dense subset consisting of "well-behaved" functions called polynomials. This is best expressed by the so-called Weierstraß theorem which holds both in classical.[14] and constructive analysis.[15]

[14] See Rudin [1976, page 159] or [1973, page 115] for classical versions.
[15] See Bridges [1979, page 94] or Bishop and Bridges [1985, page 109] for constructive versions.

Theorem 7.4 (Weierstraß Approximation Theorem [Bridges, 1979, page 94]. *Let X be a compact subset of $\mathbb{R}^n$, let $f \in C(X, \mathbb{R})$ and let $\varepsilon > 0$. Then there exists a polynomial $p\colon \mathbb{R}^n \to \mathbb{R}$ such that*

$$\forall x \in X, \qquad |f(x) - p(x)| < \varepsilon.$$

This subject was taken even further independently by Hauck [1976] and Caldwell and Pour-El [1975] where it was proven that the theorem above holds even in a computable setting. Weihrauch [2000, pages 159–161] adopted the idea, in defining a representation for functions over (suitable compact intervals of) real numbers, of using fast converging Cauchy sequences of polynomials with rational coefficients.[16]

Definition 7.5 (Strong Cauchy Sequences of Functions). Let X be any set and let $Y \subseteq \mathbb{R}$. By $\mathrm{Cauf}_{2^n}(X, Y)$ we mean the set of all sequences $\{f_n\colon X \to Y \mid n \in \mathbb{N}\}$ such that for any $x \in X$, the sequence $(f_n(x))_{n\in\mathbb{N}}$ is strongly Cauchy in Y (see Definition 4.56), i.e.

$$\forall x \in X, \qquad (f_n(x))_{n\in\mathbb{N}} \in \mathrm{Cau}_{2^n}(Y).$$

Definition 7.6 (Cauchy Representation of Functions). Let $X \subseteq \mathbb{R}^n$ be arbitrary. We say that the sequence

$$\{p_n\colon \mathbb{R}^n \to \mathbb{R} \mid n \in \mathbb{N}\}$$

of polynomials with rational coefficients *represents* $f\colon X \to \mathbb{R}$ if and only if

1. $(p_n)_{n\in\mathbb{N}} \in \mathrm{Cauf}_{2^n}(X, \mathbb{R})$,
2. $\forall x \in X,\ f(x) = \lim_{n\to\infty} p_n(x)$.

In case the sequence can be effectively generated, we say that $(p_n)_{n\in\mathbb{N}}$ *effectively* represents f.

The following definition is derived from Weihrauch [2000]:

Definition 7.7 (Computable Cube). The cube $[a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n$ is called *computable* if and only if each a_i and b_i $(1 \leq i \leq n)$ is a TTE-computable real number.

Now we are in the right position to present the effective Weierstraß theorem. Of course, the version we present here is adjusted to suit our own framework, otherwise it is equivalent to the one in Weihrauch [2000, Lemma 6.1.10, page 160]:

[16] In Weihrauch [2000, page 159] the *rational polygons* are originally used, but later (on page 160 of the same book) it is mentioned that rational polygons can be substituted by *rational polynomials*.

Theorem 7.8 (Effective Weierstraß Theorem). *Let $n \geq 1$ and let $X \subseteq \mathbb{R}^n$ be a computable cube. Then for any $f\colon X \to \mathbb{R}$ the following are equivalent*:

1. *f is TTE-computable.*
2. *There exists a sequence $(p_n)_{n\in\mathbb{N}}$ of polynomials with rational coefficients effectively representing f.*

In light of the theorem above, we embark on demonstrating how to define functions over real numbers using SHRAD.

8. Primitive Recursion

As we need to define effective sequences of polynomials, perhaps it is more convenient to define *primitive recursion* exclusively.

Definition 8.1 (rec_σ, A_{rec_σ}). For each type σ, i.e. including PCF-types, we define a term

$$A_{rec_\sigma}\colon \gamma \to \gamma,$$

where

$$\gamma := \sigma \to (nat \to \sigma \to \sigma) \to (nat \to \sigma),$$

as follows:

$$A_{rec_\sigma} := \lambda T a_0 f n. \qquad \mathit{if} \quad Zero(n) a_0 (f n (T\, a_0 f (n-1))), \tag{10}$$

where:

1. $T\colon \sigma \to (nat \to \sigma \to \sigma) \to (nat \to \sigma)$,
2. $a_0\colon \sigma$,
3. $f\colon nat \to \sigma \to \sigma$,
4. $n\colon nat$,

based on which the constant rec_σ is defined in terms of Y_γ by

$$rec_\sigma = Y_\gamma\ A_{rec_\sigma}. \tag{11}$$

Bearing in mind the standard interpretation of the fix-point constant we have:

Definition 8.2 (rec_σ).

$$\mathcal{A}_{\mathcal{SH}}(rec_\sigma) := \mathsf{rec}_\sigma = \sqcup\{[\![A_{rec_\sigma}]\!]^n(\bot) \mid n \in \mathbb{N}\}.$$

Let us fix a type σ and consider a starting point $a_0 \in \mathbb{D}_\sigma$ together with a generating function $f\colon \mathbb{N}_\perp \to \mathbb{D}_\sigma \to \mathbb{D}_\sigma$. We define a sequence $(x_n)_{n\in\mathbb{N}}$ of elements of $\mathbb{D}_\sigma$ by

$$\begin{aligned} x_0 &= a_0, \\ x_1 &= f\,1\,x_0, \\ &\vdots \\ x_{n+1} &= f\,(n+1)\,x_n, \\ &\vdots \end{aligned}$$

and a sequence of functions $(g_n)_{n\in\mathbb{N}} \subseteq [\mathbb{N}_\perp \to \mathbb{D}_\sigma]$ by

$$\forall n \in \mathbb{N}, \qquad g_n := (\llbracket A_{rec_\sigma} \rrbracket^n(\perp))\,a_0\,f.$$

Proposition 8.3.

$$\forall n \geq 1, \quad \forall m < n, \qquad g_n(m) = x_m.$$

Proof. Easy induction:

Base case $(n = 1, m = 0)$:

$$\begin{aligned} g_1(0) &= (\llbracket A_{rec_\sigma} \rrbracket^n(\perp))\,a_0\,f\,0 \\ \text{(definition of } A_{rec_\sigma}) &= a_0. \end{aligned}$$

Inductive case $(n > 1)$:

$$\begin{aligned} g_n(m) &= (\llbracket A_{rec_\sigma} \rrbracket^n(\perp))\,a_0\,f\,m \\ &= ((\llbracket A_{rec_\sigma} \rrbracket \llbracket A_{rec_\sigma} \rrbracket^{n-1})(\perp))\,a_0\,f\,m \\ &= \llbracket A_{rec_\sigma} \rrbracket(\llbracket A_{rec_\sigma} \rrbracket^{n-1}(\perp))\,a_0\,f\,m \\ \text{(definition of } A_{rec_\sigma}) &= \begin{cases} a_0 & \text{if} \quad m = 0, \\ f\,m(\llbracket A_{rec_\sigma} \rrbracket^{n-1}(\perp)\,a_0\,f\,(m-1)) & \text{if} \quad m > 0, \end{cases} \\ \text{(definition of } (g_n)_{n\in\mathbb{N}}) &= \begin{cases} a_0 & \text{if} \quad m = 0, \\ f\ m\ (g_n(m-1)) & \text{if} \quad m > 0, \end{cases} \\ \text{(ind. hyp.)} &= \begin{cases} a_0 & \text{if} \quad m = 0, \\ f\ m\ x_{m-1} & \text{if} \quad m > 0, \end{cases} \\ \text{(definition of } (x_n)_{n\in\mathbb{N}}) &= \begin{cases} a_0 & \text{if} \quad m = 0, \\ x_m & \text{if} \quad m > 0, \end{cases} \\ (x_0 = a_0) &= \begin{cases} x_0 & \text{if} \quad m = 0, \\ x_m & \text{if} \quad m > 0, \end{cases} \\ &= x_m. \end{aligned}$$

□

On the other hand we have

$$\begin{aligned}\mathsf{rec}_\sigma a_0 f &= (\sqcup\{[\![A_{rec_\sigma}]\!]^n(\bot) \mid n \in \mathbb{N}\})a_0 f\\ &= \sqcup\{[\![A_{rec_\sigma}]\!]^n(\bot)a_0 f \mid n \in \mathbb{N}\}\\ \text{(definition of } (g_n)_{n\in\mathbb{N}}) &= \sqcup\{g_n \mid n \in \mathbb{N}\}.\end{aligned}$$

Therefore, we can summarize our observation of the behaviour of rec_σ in the following lemma:

Lemma 8.4. *Let σ be any type and consider $a_0 \in \mathbb{D}_\sigma$ and $f \in [\mathbb{N}_\bot \to \mathbb{D}_\sigma \to \mathbb{D}_\sigma]$. Define the sequence $(x_n)_{n\in\mathbb{N}}$ by*

$$\begin{aligned}x_0 &= a_0,\\ x_1 &= f1x_0,\\ &\vdots\\ x_{n+1} &= f(n+1)x_n,\\ &\vdots\end{aligned}$$

then the following holds:

$$\forall n \in \mathbb{N}, \qquad \mathsf{rec}_\sigma a_0 f n = x_n.$$

8.1. *Primitive Recursion and Strong Extensionality*

Primitive recursion obviously preserves $\simeq_{\mathcal{D}^\infty}$ of Definition 4.1. Moreover, in Section 4 we demonstrated that all constants except *limit* and Y_σ's preserve $\simeq_{\mathcal{D}^\infty}$ too. Therefore:

Theorem 8.5 (Strong Extensionality). *Assume that $\mathcal{L}$ is the language derived from* SHRAD *by omitting limit and replacing Y_σ's by rec_σ's at each type σ. Let $m \geq 1$ and $f\colon r^m \to r$ be any term written in $\mathcal{L}$. Then $[\![f]\!]\colon (\mathcal{D}^\infty)^m \to \mathcal{D}^\infty$ is strongly extensional.*

9. How to Define a Function

In our setting—by virtue of the effective Weierstraß theorem—the procedure for defining a function starts with defining a suitable effective sequence of polynomials with rational coefficients, which we simply call *rational polynomials*. As we are merely sketching the procedure, we focus on functions with one argument over real numbers. Also, for now suppose that everything we talk about happens inside $[-1, 1]$. We will later show how this can be extended to cover the general case.

9.1. *Polynomials*

First things first. In order to define the polynomial

$$p = \sum_{i=0}^{n} a_i x^i$$

one needs:

Multiplication. We already have a primitive for this operation in the language.

Addition. Well, we are half-way through as we have average as a primitive. Thus we define

$add\colon r \to r \to r,$
$add = \lambda xy.\, M^{-1}(avg\, x\, y).$

Now if we allow the infix binary operator $\boxplus\colon [-1, 1] \to [-1, 1] \to [-1, 1]$ to be

$\boxplus := to_{\mathcal{J}_{\max}^2 \to \mathcal{J}_{\max}}[\![add]\!]$

it should not be difficult to see that over the total real numbers in $[-1, 1]$

$\forall x, y \in [-1, 1], \qquad x \boxplus y = \max\{-1, \min\{x + y, 1\}\}.$

Rational coefficients. Rational numbers in $[-1, 1]$ are provided via *qtoR*, but we must be careful as even when we are approximating some $f\colon [-1, 1] \to [-1, 1]$ with rational polynomials, the coefficients need not be restricted to $[-1, 1]$. Thus, we implement an operation:

$\text{qsplit}\colon \mathbb{Q} \to \mathbb{N} \times \mathbb{Q},$
$\qquad\qquad q \mapsto (n, q'),$

where $q' \in [-1, 1]$ and $2^n \times q' = q$. The implementation (in PCF) is fairly straightforward and we leave it to the reader. The only thing to remember is that we represented rational numbers as *triplets of natural numbers* (see Section 4.3) and as we do not have product types in SHRAD (Definition 2.1) a proper implementation requires *four* components (say):

$qsplit_i\colon \underbrace{nat \to nat \to nat}_{\mathbb{Q}} \to nat$

for $i \in \{1, 2, 3, 4\}$.

Notation 9.1. The type $nat \to nat \to nat$ is abbreviated by *rat* (for *rational*).

Now let us put the pieces together. As the rational numbers ended up being decomposed into pairs of natural and rational numbers, accordingly we need to equip ourselves with new operations allowing us to multiply out-of-range rational numbers with real numbers in $[-1, 1]$, the same for addition. However, these are common-sense practices in programming. For example, one can implement the following operation:

$\text{qmult}\colon \mathbb{Q} \times [-1, 1] \to [-1, 1],$
$\qquad\qquad (q, x) \mapsto \max\{-1, \min\{s, 1\}\},$

where $s = 2^n\,(q'x)$ and $(n, q') = \mathrm{qsplit}\,q$ by

$$\begin{aligned} &qmult\colon\ rat \to r \to r,\\ &qmult = \lambda q x.\,(M^{-1})^n\,(mult\ qtoR(q')\ x), \qquad \text{where } (n, q') = \mathrm{qsplit}\,q. \end{aligned}$$

For adding (out-of-range) rational numbers to real numbers in $[-1, 1]$ one may define

$$\begin{aligned} &qadd\colon\ rat \to r \to r,\\ &qadd = \lambda q x.\,(M^{-1})^n\,(add\ qtoR(q')\ M^n(x)), \end{aligned}$$

where $(n, q') = \mathrm{qsplit}\,q$.

9.2. *Out-of-Range Functions*

Our framework deals only with real numbers in $[-1, 1]$. Hence, any function working outside this range must be scaled and moved inside this interval.

Convention 9.2. *From now on*, *by* computable *we mean* TTE-computable.

Assume that for some computable $r, s \in \mathbb{R}$, a computable $f\colon [r, s] \to \mathbb{R}$ has caught our eye and we want to implement it in SHRAD. As computability implies continuity, the range of f must be some compact subset of $\mathbb{R}$—in fact some compact interval (say) $[t, u]$. In case both $dom(f)$ and $range\,(f)$ happen to be subsets of $[-1, 1]$, no scaling is needed and we can proceed directly to the implementation of f in SHRAD. If, on the other hand, any of them goes beyond the limit, we must engage in some scaling. For that, let k be—say, the least—natural number such that

$$\forall x \in dom(f) \cup range\,(f)\,, \qquad |x| < 2^k.$$

If we define $g\colon \mathbb{R} \rightharpoonup \mathbb{R}$ by

$$g(x) = \frac{f(2^k x)}{2^k},$$

it can easily be verified that

$$\begin{aligned} dom(g) &= [r/2^k, s/2^k],\\ range\,([]\ g] &= [t/2^k, u/2^k], \end{aligned}$$

hence both $dom(g)$ and $range\,([]\ g]$ are subsets of $[-1, 1]$ (Figure 2) and as computability of g easily follows that of f, we can proceed to implement g in SHRAD. Once this has been done, f can be calculated by re-scaling, i.e.

$$\forall x \in [r, s], \qquad f(x) = 2^k g(x/2^k).$$

It is easily seen that this procedure can be generalized to the case of functions with more than one argument.

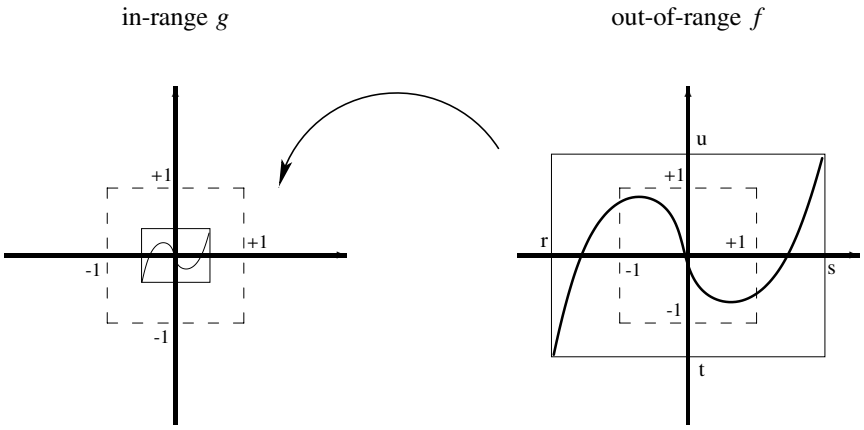

Fig. 2. Handling out-of-range functions.

9.3. *Test of Practice*

The procedure outlined so far is theoretically appealing. However, how feasible is the whole process in specific concrete cases? The first example that we discuss is the case of *exponential function* over $[-1, 1]$. As the range stretches beyond $[-1, 1]$, we instead consider $f\colon [-1, 1] \to [-1, 1]$ defined by

$$f(x) = \tfrac{1}{4}\exp(x).$$

This f is "in-the-range", i.e.

$$\mathit{range}\,(f) \subseteq [-1, 1].$$

From Weihrauch [2000, Example 4.3.3, page 111] we learn that exp is computable, hence there exists an effective representation of f with rational polynomials. In fact we may just resort to the *Taylor* series

$$f(x) = \frac{1}{4}\sum_{n=0}^{\infty}\frac{x^n}{n!}$$

and define the sequence $(p_n)_{n\in\mathbb{N}}$ by

$$\forall n \in \mathbb{N}, \qquad p_n(x) = \frac{1}{4}\sum_{i=0}^{n}\frac{x^i}{i!}.$$

Incidentally $(p_n)_{n\in\mathbb{N}}$ is a strongly Cauchy sequence of polynomials over $[-1, 1]$, so all we need to do is to apply *rec* of Section 8 combined with the procedure outlined in Section 9.1 for implementing rational polynomials to get $(p_n)_{n\in\mathbb{N}}$. As the last step, we apply limit and we are done.

Even *sine* and *cosine* have Taylor series whose coefficients diminish factorially which makes them easily implementable in SHRAD:

$$\sin(x) = \sum_{n=0}^{\infty}(-1)^n\frac{x^{2n+1}}{(2n+1)!},$$

$$\cos(x) = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!}.$$

Yet there are important functions with rather *slow* converging Taylor series. After all, we admit that Taylor series were in no way tailor made for SHRAD. Take *natural logarithm* for instance. For $|x| < 1$ we have

$$\log(1+x) = \sum_{j=1}^{\infty} (-1)^{j+1} \frac{x^j}{j}.$$

Also notice that $\log(1+x)$ tends to $-\infty$ as $x \to -1$. So each time we need to content ourselves with some interval

$$\left[-1 + \frac{1}{2^n}, 1\right] \qquad (\text{for some } n \in \mathbb{N})$$

and scale the range of $\log(1+x)$ on this domain down to $[-1, 1]$ and then try to come up with a sequence of rational polynomials which effectively represents $\log(1+x)$ on this domain. The case for the function

$$x \mapsto \frac{1}{x}$$

is more or less the same.

10. Future Work

In this paper we presented the language SHRAD and demonstrated some of its key properties. Yet it should (hopefully) only be the beginning. The future directions may be categorized as follows:

Theory. On the more theoretical side, we draw attention to the emergence of a new family of mathematical objects used as models of data types. We modelled SHRAD inside a *category of cpo's equipped with a special logical relation* and demonstrated that all SHRAD-definable objects preserve the logical relation. Looking from the opposite angle, we managed to purge the original category of cpo's (without the logical relation) from those unwanted objects which do not preserve the logical relation. However, then the question of universality comes up:

Problem 10.1. *How is it possible to extend* SHRAD *to a language which is universal with respect to the specific category of cpo's* $\{\mathbb{D}_\sigma \mid \sigma \in \mathbb{T}_{\mathcal{SH}}\}$ *of Definition* 3.9 *together with the logical relation* $\mathcal{X}$ *of Definition* 6.2.

This in turn necessitates a more comprehensive study of this model which is as of now not so well known.

We proved the weak-extensionality of the setting in Theorem 6.13. Nevertheless one may wonder if this can be strengthened to strong extensionality by e.g. coming up

with a new implementation of limit operator and replacing fix-point constants with their weaker counterparts, i.e. primitive recursion.

We can push this matter even further. In Section 4 we demonstrated that avg is extensional over the *total* real numbers while it is not extensional over the *partial* ones.[17] Now it is perfectly legitimate to ask whether based on the same model as ours it is possible to come up with other forms of implementations for average, multiplication, etc which are extensional over *both* partial and total real numbers. Our feeling is negative though we do not yet have a proof. Thus we propose:

Conjecture 10.2. *There exists no function* avg': $\mathcal{D}^\infty \to \mathcal{D}^\infty \to \mathcal{D}^\infty$ *such that*:

1. $\forall x, y \in \mathcal{D}^\omega, \quad to_{\mathcal{J}}(\mathsf{avg}'\ x\ y) = (to_{\mathcal{J}}(x) + to_{\mathcal{J}}(y))/2,$
2. $\forall x_1, x_2, y_1, y_2 \in \mathcal{D}^\infty, \quad [(to_{\mathcal{J}}(x_1) = to_{\mathcal{J}}(x_2)) \wedge (to_{\mathcal{J}}(y_1) = to_{\mathcal{J}}(y_2))] \Rightarrow (to_{\mathcal{J}}(\mathsf{avg}'\ x_1\ y_1) = to_{\mathcal{J}}(\mathsf{avg}'\ x_2\ y_2)).$

If the previous conjecture turns out to be true, then unfortunately, objects like the *computable zero-finding functional* are highly unlikely to be definable in SHRAD. The reason is that for such objects, the extensionality over partial real numbers cannot be avoided. Thus, in case such an object is definable in the language, it must be defined in the fragment consisting of the primitives that are extensional over partial real numbers.

On the other hand, one may think of studying variations of SHRAD obtained by adding/replacing the primitives. One immediate choice is to add a constant for integration and then employ the implementation by Longley [1998, 1999].

Another interesting primitive is cases by Escardó [2000] which can be used for definition by cases over reals in a signed-digit binary setting. A way forward is to study the languages obtained by:

1. Adding cases to SHRAD.
2. Adding cases to SHRAD while dropping *limit*.

Practice. On a more practical side we need to analyse the efficiency of the framework. Although parallelism is avoided, the complexity issue needs to be taken care of. It would be interesting to know the complexity of a few basic operations such as limit, exponential, etc.

Acknowledgements

My thanks go—first and foremost—to my supervisor Professor Achim Jung. He helped me have the opportunity to do Ph.D. under his supervision in the first place, and I benefited from his expertise throughout all the stages of my studies. I am also grateful for his invaluable comments right from the beginning of the writing of my thesis Farjudian [2004b]—out of which comes this paper—to the end.

[17] See (2) and Corollary 4.34.

I also thank the other members of my thesis group, Prof. Uday Reddy and Dr. Martín Escardó.

The thesis could not be regarded complete without the assurance of the approvals of the external and the internal examiners—Prof. Abbas Edalat and Dr. Martín Escardó, respectively. Their careful reading of the thesis and precious suggestions were—and definitely will be—heeded.

Finally, I express my special thanks to Prof. Edalat on his great and generous help to me so as to not lose the opportunity of participating in the CiE2005 conference.

References

Errett Bishop and Douglas S. Bridges. *Constructive Analysis*. Volume 279 of Grundlehren der Mathematischen Wissenschaften. Springer, Berlin, 1985.

H.-J. Böhm, R. Cartwright, M. J. O'Donnell, and M. Riggle. Exact real arithmetic: a case study in higher order programming. In *ACM Symposium on Lisp and Functional Programming*, 1986.

Vasco Brattka. Recursive characterization of computable real-valued functions and relations. *Theoretical Computer Science*, 162:45–77, 1996.

Douglas S. Bridges. *Constructive Functional Analysis*. Number 28 in Research Notes in Mathematics. Pitman, London, 1979.

J. Caldwell and Marian Boykan Pour-El. On a simple definition of computable functions of a real variable—with applications to functions of a complex variable. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 21:1–19, 1975.

Pietro Di Gianantonio. A Functional Approach to Computability on Real Numbers. Ph.D. thesis, Università di Pisa-Genova-Udine, 1993.

Pietro Di Gianantonio. An abstract data type for real numbers. *Theoretical Computer Science*, 221:295–326, 1999.

Abbas Edalat and R. Heckmann. Computing with real numbers: (i) LFT approach to real computation, (ii) domain-theoretic model of computational geometry. In G. Barthe, P. Dybjer, L. Pinto, and J. Saraiva, editors, *Applied Semantics*: *Advanced Lectures*, pages 193–267. Volume 2395 of Lecture Notes in Computer Science. Springer Verlag, 2002.

Abbas Edalat and Peter John Potts. A new representation for exact real numbers. In S. Brookes and M. Mislove, editors, *Electronic Notes in Theoretical Computer Science*, volume 6. Elsevier, Amsterdam, 2000.

Abbas Edalat, Peter John Potts, and Ph. Sünderhauf. Lazy computation with exact real numbers. In *Proceedings of the Third ACM SIGPLAN International Conference on Functional Programming*, pages 185–194, 1998.

Martín Hötzel Escardó. Real-PCF extended with ∃ 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*, pages 13–24, Christ Church, Oxford, April 1996. IC Press.

Martín Hötzel Escardó. PCF Extended with Real Numbers: A Domain-Theoretic Approach to Higher-Order Exact real Number Computation. Ph.D. thesis, Department of Computer Science, The University of Edinburgh, November 1997.

Martín Hötzel Escardó. Effective and sequential definition by cases on the reals via infinite signed-digit numerals. In Abbas Edalat, Achim Jung, Klaus Keimel, and Marta Kwiatkowska, editors, *Electronic Notes in Theoretical Computer Science*, volume 13. Elsevier, Amsterdam, 2000.

Martín Hötzel Escardó, M. Hofmann, and T. Streicher. Mediation is inherently parallel. EPSRC report for project GR/M64840. University of Edinburgh, Laboratory for Foundations of Computer Science, 1998.

Martín Hötzel Escardó, M. Hofmann, and T. Streicher. On the non-sequential nature of the interval-domain model of exact real-number computation. *Mathematical Structures in Computer Science*, 2004. Accepted for publication.

Amin Farjudian. Conservativity of wRPCF over PCF. Unpublished Manuscript, 2003.

Amin Farjudian. Sequentiality and piecewise-affinity in segments of Real-PCF. *Electronic Notes in Theoretical Computer Science*, 73:3–43, 2004a.

Amin Farjudian. Sequentiality in Real Number Computation. Ph.D. thesis, School of Computer Science, University of Birmingham, 2004b.

Jürgen Hauck. Berechenbare reelle Funktionenfolgen. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 22:265–282, 1976.

Reinhold Heckmann. The appearance of big integers in exact real arithmetic based on linear fractional transformations. In *Foundations of Software Science and Computation Structures*, volume 1378, pages 172–188. Springer, Berlin, 1998.

Reinhold Heckmann. How many argument digits are needed to produce *n* result digits? In Abbas Edalat, David Matula, and Philipp Sünderhauf, editors, *Electronic Notes in Theoretical Computer Science*, volume 24. Elsevier, Amsterdam, 2000a.

Reinhold Heckmann. Big integers and complexity issues in exact real arithmetic. In Abbas Edalat, Achim Jung, Klaus Keimel, and Marta Kwiatkowska, editors, *Electronic Notes in Theoretical Computer Science*, volume 13. Elsevier, Amsterdam, 2000b.

Reinhold Heckmann. Translation of Taylor series into LFT expansions. *Symbolic Algebraic Methods and Verification Methods*, 2001.

Reinhold Heckmann. Contractivity of linear fractional transformations. *Theoretical Computer Science*, 279(1):65–82, 2002.

Michal Konečný. Many-valued real functions computable by finite transducers using IFS-representations. Ph.D. thesis, School of Computer Science, The University of Birmingham, 2000.

Michal Konečný. Real functions computable by finite automatons using affine representations. *Theoretical Computer Science*, 284(2):373–396, July 2002.

John Longley. When is a functional program not a functional program? A walkthrough introduction to the sequentially realizable functionals, 1998. ML source file, available from `http://www.dcs.ed.ac.uk/hom/jrl/`.

John Longley. When is a functional program not a functional program? In *Proceedings of the Fourth ACM SIGPLAN International Conference on Functional Programming*, pages 1–7, New York, USA, September 1999. ACM Press, New York, 1999.

J. R. Marcial-Romero and Martín Hötzel Escardó. Semantics of a sequential language for exact real-number computation. To appear in LICS, 2004.

Valérie Ménissier-Morain. Arbitrary precision real arithmetic: design and algorithms. Submitted to the *Journal of Symbolic Computation*, September 1996.

Norbert Müller. The iRRAM: exact arithmetic in C++. In *Workshop on Constructivity and Complexity in Analysis*, Swansea, 2000.

G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

Dave Plume. A calculator for exact real number computation. 4th Year Project Report, Department of Computer Science and Artificial Intelligence, University of Edinburgh, 1998.

Peter Potts, Abbas Edalat, and Martín Hötzel Escardó. Semantics of exact real arithmetic. In *Twelfth Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, Los Alamitos, CA, 1997.

Walter Rudin. *Functional Analysis*. McGraw Hill Series in Higher Mathematics. McGraw Hill, New York, 1973.

Walter Rudin. *Principles of Mathematical Analysis*, third edition. McGraw Hill, Auckland, 1976.

Kishor S. Trivedi and Miloš D. Ercegovac. On-line algorithms for division and multiplication. *IEEE Transactions on Computers*, C-26(7):681–687, July 1977.

Klaus Weihrauch. *Computable Analysis*. Springer, Berlin, 2000.

Received September 15, 2005, *and in revised form January* 18, 2006, *and in final form May* 9, 2006.