Comparing Functional Paradigms for Exact Real-number Computation

Andrej Bauer[1,⋆], Martín Hötzel Escardó[2], and Alex Simpson[3,⋆⋆]

[1] IMFM, University of Ljubljana, Slovenia
[2] School of Computer Science, University of Birmingham, England
[3] LFCS, Division of Informatics, University of Edinburgh, Scotland

Abstract. We compare the definability of total functionals over the reals in two functional-programming approaches to exact real-number computation: the *extensional* approach, in which one has an abstract datatype of real numbers; and the *intensional* approach, in which one encodes real numbers using ordinary datatypes. We show that the type hierarchies coincide up to second-order types, and we relate this fact to an analogous comparison of type hierarchies over the *external* and *internal* real numbers in Dana Scott's category of equilogical spaces. We do not know whether similar coincidences hold at third-order types. However, we relate this question to a purely topological conjecture about the Kleene-Kreisel continuous functionals over the natural numbers. Finally, although it is known that, in the extensional approach, parallel primitives are necessary for programming total first-order functions, we demonstrate that, in the intensional approach, such primitives are not needed for second-order types and below.

1 Introduction

In functional programming, there are two main approaches to exact real-number computation. One is to use a specialist functional programming language that contains the real numbers as an abstract datatype. This approach is *extensional* in the sense that the data structures representating real numbers are hidden from view and one may only manipulate reals via representation-independent operations upon them. A second approach is to use an ordinary functional language, and to encode real numbers using standard infinite data structures, for example, streams. This approach is *intensional* in the sense that one has direct access to the encodings of reals, allowing the possibility of distinguishing between different representations of the same real number. In recent years, the extensional approach has been the subject of much theoretical investigation via the study of specialist languages, such as Di Gianantonio's **RL** [DiG93] and Escardó's **RealPCF** [Esc96]. On the other hand, the intensional approach is the one that is actually used when exact real-number computation is implemented in practice—see, for example, [GL01].

⋆ Research supported by the Slovene Ministry of Science grant Z1-3138-0101-01
⋆⋆ Research supported by an EPSRC Advanced Research Fellowship.

This paper presents preliminary results in a general investigation relating the two approaches. Specifically, we address the question of how the programmability of higher-type total functionals over the real numbers compares between the two approaches. To this end, we consider two type hierarchies built using function space and product over a single base type, real. The first hierarchy is constructed by interpreting each type σ as the set $[\sigma]_E$ of extensionally programmable total functionals of that type, and the second by interpreting σ as the set $[\sigma]_I$ of intensionally programmable total functionals. As our first main result, Theorem 1, we prove that for all second-order (and below) types σ, the sets $[\sigma]_E$ and $[\sigma]_I$ coincide, thus a second-order functional is extensionally programmable if and only if it is intensionally programmable. This result thus applies at the type level at which definite integration,

$$(f, a, b) \mapsto \int_a^b f(x)\, \mathrm{d}x \quad : \quad (\mathsf{real} \to \mathsf{real}) \times \mathsf{real} \times \mathsf{real} \to \mathsf{real} \ ,$$

resides; although, in the case of integration, our result gives no new information as it is already known to be programmable under both the extensional and intensional approaches, see [EE00] and [Sim98] respectively.

We prove Theorem 1 by relating it to an analogous question of the coincidence of type hierarchies in the setting of Dana Scott's category of equilogical spaces [Sco96,BBS02]. In that setting there is an *external* type hierarchy $(\sigma)_E$, built over Euclidean space, and there is an *internal* hierarchy $(\sigma)_I$, built over the object of real numbers as defined in the internal logic of the category. Again, we show that $(\sigma)_E$ and $(\sigma)_I$ coincide up to second-order types, Theorem 2.

It is of course natural to ask whether the above type hierarchies also coincide for third-order σ and above. We do not know the answer to this question, but a further contribution of this paper is to relate the agreement of the hierarchies at higher types to a purely topological conjecture about the Kleene-Kreisel continuous functionals [Kle59,Kre59] of second-order type, see Sect. 5.

Our methodology for studying the two approaches to exact real-number computation is to consider a paradigmatic programming language for each. For the extensional approach, we use Escardó's **RealPCF+**, which is **RealPCF** [Esc96] extended by a parallel existential operator—a language that enjoys the merit of being universal with respect to its domain-theoretic semantics [ES99]. For the intensional approach, we encode real numbers within Plotkin's **PCF++**, which is **PCF** extended by parallel-conditional and existential operators [Plo77]. Again, **PCF++** enjoys a universality property with respect to its denotational semantics [Plo77].

Admittedly, both **RealPCF+** and **PCF++** are idealized languages, distant from real-world functional languages such as Haskell [Has]. As such, they provide ideal vehicles for a theoretical investigation into programmability questions such as ours. Nevertheless, it is our desire that our results should relate to the actual practice of exact real-number computation. There is one main obstacle to such a transference of the results: the parallel features of **PCF++** do not appear in Haskell and related languages. We address this issue in Sect. 7, where we show that, again for second-order σ and below, the parallel features of **PCF++** are nowhere required to program functionals in $[\sigma]_I$, Theorem 3. Thus a second-order total functional over the reals is programmable in an ordinary sequential functional language if and only if it is programmable in the idealized, specialist and highly parallel language **RealPCF+**. Again, we do not know whether this result extends to third-order types and above.

Although our investigation is one into questions of programmability (i.e. of definability) within **RealPCF+** and **PCF++**, we carry out the investigation purely at the denotational level, relying on known universality results to infer definability consequences from the semantic correspondences we establish. In doing so, there is one major way in which the results presented in this paper depart from the outline presented above. A full investigation would show that the computable (and hence definable) total functionals coincide between the denotational interpretations of the extensional and intensional approaches. Instead, we establish the coincidence of *arbitrary* total functionals, whether computable or not. Our reason for ignoring computability questions is that the results we establish already require significant technical machinery. Although it should be possible to give computability-sensitive versions of our results, this would unavoidably cause still further technical complications in the proofs. We leave this for future work. We remark that the results we prove, although computability free, do nonetheless have definability consequences relative to functional languages extended with oracles for all set-theoretic functions from $\mathbb{N}$ to $\mathbb{N}$, or equivalently relative to functional languages with programs given by infinite syntax trees.

The main question left open by this paper is whether our results extend to third-order types and beyond. Regarding this, we remark that we lack examples of genuinely interesting *total* functionals of type three to which such generalisations of our results would apply. However, another interesting development of our work would be to compare hierarchies of *partial* functionals over the reals.

This paper assumes some familiarity with domain theory, topology and category theory, for which our references are [AJ94,Dug89,Mac71] respectively. An important contribution of the paper is to show how mathematical tools from these subjects may be combined to attack seemingly innocuous questions that originate in functional programming.

Proofs. For lack of space, proofs are only outlined in this conference version of the paper. More detailed proofs can be found at:

`http://www.dcs.ed.ac.uk/home/als/Research/icalp_proofs.ps` .

2 Domains for Real-number Computation

We first fix terminology—see [AJ94] for definitions. We assume that a directed-complete partial order (dcpo), $(D, \sqsubseteq)$, has least element. We call a dcpo ω-*continuous* if it has a countable basis, and we write $x \ll y$ for the way-below relation on it. For us, a *domain* is an ω-continuous bounded-complete dcpo. We write $\omega\mathbf{BC}$ for the category of domains and (directed-)continuous functions, and we write $\omega\mathbf{L}$ for its full subcategory of ω-continuous lattices. Both categories are cartesian closed with exponentials given by the dcpo of all continuous functions.

Our main interest will be in two particular domains, one for each of the two approaches to exact real-number computation mentioned in the introduction. The *interval domain* $\mathcal{I}$ has underlying set $\{\mathbb{R}\} \cup \{[a,b] \mid a \leq b \in \mathbb{R}\}$, with its order defined by $\delta \sqsubseteq \delta'$ if and only if $\delta \supseteq \delta'$. This is indeed a domain.

The interval domain is intimately connected with the extensional approach to exact real-number computation. Indeed, the abstract datatype of real numbers in **RealPCF** [Esc96] is specifically designed to have $\mathcal{I}$ as its denotational interpretation. Furthermore, Escardó and Streicher [ES99] have established a universality result with respect to the domain-theoretic semantics: every computable element in the domain interpreting a **RealPCF** type is definable, by a term of that type, in the language **RealPCF+**, which is **RealPCF** extended with a parallel existential operator. In this paper, although we are motivated by definability questions, we do not wish to entangle ourselves in computability issues. Thus we remark on the following modified version of Escardó and Streicher's result. Every element (computable or not) in the domain interpreting a **RealPCF** type is definable in the language $\mathit{\Omega}$**RealPCF+**, which is **RealPCF+** extended with an oracle for every set-theoretic function from $\mathbb{N}$ to $\mathbb{N}$.

Under the intensional approach to exact real-number computation, one needs to select a computationally *admissible* representation of real numbers [WK87]. There are many equivalent choices. For simplicity, we use a mantissa-exponent representation, where the mantissa, a real number in the interval $[-1,1]$, is represented using *signed-binary* expansions. Specifically, a real number is represented by a pair (n, α) where the mantissa $\alpha \in \{-1,0,1\}^\omega$ represents the number $0.\alpha_0\alpha_1\alpha_2\ldots$, i.e. $\sum_{i=0}^{\infty} 2^{-(i+1)}\alpha_i$, and the exponent $n \in \mathbb{N}$ gives a multiplier of 2^n, thus the pair (n,α) represents the real number $\sum_{i=0}^{\infty} 2^{n-(i+1)}\alpha_i$.

To implement the above representation in a functional programming language, one would most conveniently encode a real number as a pair consisting of a natural number followed by a stream. However, in order to fix on as simple a language as possible, we use instead a direct implementation in Plotkin's **PCF** [Plo77] extended with product types. In **PCF**, the base type, nat, is interpreted as the flat domain $\mathbb{N}_\perp = \{\perp\} \cup \mathbb{N}$ with least element $\perp$. Function space and product are interpreted using the cartesian-closed structure of $\omega\mathbf{BC}$. As we are interested in definability, we mention Plotkin's universality result: every computable element in the domain interpreting a **PCF** type is definable in the language **PCF++**, which is **PCF** extended with parallel-conditional and existential operators. Again, there is a computability-free version of this result. Every element (computable or not) in the domain interpreting a **PCF** type is definable in the language $\mathit{\Omega}$**PCF++**, which is **PCF++** extended with an oracle for every set-theoretic function from $\mathbb{N}$ to $\mathbb{N}$.

We represent real numbers, in **PCF**, using the type $\mathsf{nat} \to \mathsf{nat}$ whose denotational interpretation is the *function domain* $\mathcal{J} = {\mathbb{N}_\perp}^{\mathbb{N}_\perp}$. We say that a function $f \in \mathcal{J}$ is *real representing* if $f(0) \neq \perp$ and if $f(x) \in \{0,1,2\}$ when $x > 0$. Any such real-representing f encodes the real number $\sum_{i=1}^{\infty} 2^{f(0)-i}(f(i)-1)$.

3 Two Type Hierarchies of Assemblies

Our goal is to investigate the type hierarchies of total functionals on reals programmable in the two approaches to exact real-number computation. We consider simple types over a base type of real numbers, with types given by:

$$\sigma ::= \mathsf{real} \mid \sigma \times \sigma' \mid \sigma \to \sigma' \ .$$

The *order* of a type is: $\mathrm{order}(\mathsf{real}) = 0$; $\mathrm{order}(\sigma \times \sigma') = \max(\mathrm{order}(\sigma), \mathrm{order}(\sigma'))$; and $\mathrm{order}(\sigma \to \sigma') = \max(1 + \mathrm{order}(\sigma), \mathrm{order}(\sigma'))$.

For the extensional approach, we study the total functionals on reals programmable in the language Ω**RealPCF+**. Because of the universality result mentioned above, such functionals are exactly those total functionals that arise in the type hierarchy over $\mathcal{I}$ in $\omega\mathbf{BC}$. However, the type hierarchy over $\mathcal{I}$ contains both superfluous elements and redundancies. For example, $\mathcal{I}$ itself contains "partial" real numbers (proper intervals) in addition to "total" reals (singleton intervals). At first-order types, such as $\mathcal{I}^{\mathcal{I}}$, there are elements that do not represent total functions on reals because they fail to preserve total reals. Furthermore, at the same type, it is possible to have two different functions $f, g \colon \mathcal{I} \to \mathcal{I}$ that represent the same total function on reals, because, although they behave identically on total reals, they differ in their behaviour on partial reals.

For the intensional approach, we study the functionals programmable in Ω**PCF++**, using the representation described in Sect. 2. Because of the universality result, such functionals are exactly those total functionals that arise in the type hierarchy over $\mathcal{J}$ in $\omega\mathbf{BC}$. Again, there is superfluity and redundancy. Within $\mathcal{J}$, we singled out the real-representing elements in Sect. 2, and in fact each real number has infinitely many different representations. Because of this, there are two ways that a function from $\mathcal{J}$ to $\mathcal{J}$ may fail to represent a function on real numbers: either it may map some real-representing element to a non-real-representing element; or it may map two different representations of the same real number to representations of different real numbers.

Assemblies offer a convenient way of identifying the elements of the hierarchies over $\mathcal{I}$ and $\mathcal{J}$ in $\omega\mathbf{BC}$ that represent total functionals on reals. An *assembly* is a triple $A = (|A|, \|A\|, \Vdash_A)$ where $|A|$ is a set, $\|A\|$ is a domain, and $\Vdash_A$ is a binary relation between $\|A\|$ and $|A|$ such that, for all $a \in |A|$, there exists $x \in \|A\|$ such that $x \Vdash_A a$. A morphism from one assembly A to another B is simply a function $f \colon |A| \to |B|$ for which there exists a continuous $g \colon \|A\| \to \|B\|$ such that $x \Vdash_A a$ implies $g(x) \Vdash_B f(a)$, in which case we say that g *tracks* f. We write $\mathbf{Asm}(\omega\mathbf{BC})$ for the category of assemblies over domains, and $\mathbf{Asm}(\omega\mathbf{L})$ for the full subcategory of assemblies over ω-continuous lattices. Again, both categories are cartesian closed, with the exponential B^A given by

$$\begin{aligned} |B^A| &= \{f \colon |A| \to |B| \bigm| f \text{ is a morphism from } A \text{ to } B\} \\ \|B^A\| &= \|B\|^{\|A\|} \text{ in } \omega\mathbf{BC} \\ g \Vdash_{B^A} f &\iff g \text{ tracks } f \ . \end{aligned}$$

We remark that $\mathbf{Asm}(\omega\mathbf{BC})$ is equivalent to the category $\mathbf{Asm}(\mathsf{U})$ of assemblies over the combinatory algebra given by U, a chosen universal domain [GS90]; and $\mathbf{Asm}(\omega\mathbf{L})$ is equivalent to $\mathbf{Asm}(\mathsf{P})$, for Scott's combinatory algebra $\mathsf{P} = \mathcal{P}\omega$ which is an ω-continuous lattice under $\subseteq$. Thus, up to equivalence, $\mathbf{Asm}(\omega\mathbf{BC})$ and $\mathbf{Asm}(\omega\mathbf{L})$ arise as the categories of double-negation separated objects in the realizability toposes $\mathbf{RT}(\mathsf{U})$ and $\mathbf{RT}(\mathsf{P})$ respectively.

We use $\mathbf{Asm}(\omega\mathbf{BC})$ to define the two type hierarchies of total functionals we are interested in. For the extensional approach, we define an assembly $[\![\sigma]\!]_E$ for each type σ. For the base type, real, this is given by:

$$|[\![\mathsf{real}]\!]_E| = \mathbb{R} \qquad \|[\![\mathsf{real}]\!]_E\| = \mathcal{I} \qquad \delta \Vdash_{[\![\mathsf{real}]\!]_E} x \iff \delta = \{x\}$$

from which $[\![\sigma]\!]_E$ is defined using the cartesian-closed structure of $\mathbf{Asm}(\omega\mathbf{BC})$.

For each type σ, the set $|[\![\sigma]\!]_E|$, for which we henceforth use the less cluttered $[\sigma]_E$, is a set of total functionals over the reals. By the universality of $\mathit{\Omega}\mathbf{RealPCF+}$, the functionals $f \in [\sigma]_E$ are exactly those for which there exists an $\mathit{\Omega}\mathbf{RealPCF+}$ program P of type σ such that P computes f, as witnessed by the relation $[\![P]\!] \Vdash_{[\![\sigma]\!]_E} f$, where $[\![P]\!]$ is the denotational interpretation of P.

Similarly, for the intensional approach, the assembly $[\![\mathsf{real}]\!]_I$ is defined by:

$$|[\![\mathsf{real}]\!]_I| = \mathbb{R} \quad \|[\![\mathsf{real}]\!]_I\| = \mathcal{J} \quad f \Vdash_{[\![\mathsf{real}]\!]_I} x \iff f \text{ is real representing and } x = \textstyle\sum_{i=1}^{\infty} 2^{f(0)-i} \cdot (f(i) - 1)$$

and again $[\![\sigma]\!]_I$ is induced for arbitrary σ using the cartesian-closed structure of $\mathbf{Asm}(\omega\mathbf{BC})$. This time the set $|[\![\sigma]\!]_I|$, for which we henceforth write $[\sigma]_I$, is the set of those total functionals f for which there exists an $\mathit{\Omega}\mathbf{PCF++}$ program P of type σ^* (where $\mathsf{real}^* = \mathsf{nat} \to \mathsf{nat}$ and $(\cdot)^*$ commutes with function space and product) such that P computes f, as witnessed by the relation $[\![P]\!] \Vdash_{[\![\sigma]\!]_I} f$.

Theorem 1. *For any type σ with $\mathrm{order}(\sigma) \leq 2$, it holds that $[\sigma]_E = [\sigma]_I$.*

4 Two Type Hierarchies of Equilogical Spaces

Although Theorem 1 can be proved directly, we find it illuminating to approach it by looking at another situation in which there are competing hierarchies of total functionals over the reals. Our second example of a pair of hierarchies of real functionals arises in Dana Scott's category of *equilogical spaces* [Sco96,BBS02], a cartesian-closed extension of the category of topological spaces.

In the present paper we only consider countably-based equilogical spaces, and we do not impose Scott's T_0 requirement. For our purposes then, an *equilogical space* is a triple $X = (|X|, \|X\|, q_X)$ where $|X|$ is a set, $\|X\|$ is a countably-based topological space and $q_X : \|X\| \to |X|$ is a surjective function. A morphism from one equilogical space X to another Y is simply a function $f : |X| \to |Y|$ for which there exists a continuous $g : \|X\| \to \|Y\|$ such that $q_Y \circ g = f \circ q_X$. Again we say that *$g$ tracks f*. We write $\omega\mathbf{Equ}$ for the category of equilogical spaces.

We write $\omega\mathbf{Top}$ for the category of countably-based topological spaces. There is a full and faithful functor from $\omega\mathbf{Top}$ to $\omega\mathbf{Equ}$, mapping a countably-based space S to (S, S, id_S). A remarkable fact is that is $\omega\mathbf{Equ}$ is equivalent to the category $\mathbf{Asm}(\omega\mathbf{L})$ [BBS02]. Thus $\omega\mathbf{Equ}$ is cartesian closed.

There are two non-isomorphic equilogical spaces, each with good claims to be *the* equilogical space of real numbers. The *external* reals, $\mathsf{R_E}$, is the inclusion of the topological Euclidean reals as the object $(\mathbb{R}, \mathbb{R}, \mathsf{id}_{\mathbb{R}})$. The *internal* reals, $\mathsf{R_I}$, is the object $(\mathbb{N} \times \mathsf{3}^\omega, \mathbb{R}, \mathsf{r})$, where $\mathsf{3} = \{-1, 0, 1\}$ with the discrete topology, both $\mathsf{3}^\omega$ and $\mathbb{N} \times \mathsf{3}^\omega$ are given the product topologies, and r is:

$$\mathsf{r}(n, \alpha) = \textstyle\sum_{i=0}^{\infty} 2^{n-(i+1)} \alpha_i \ . \tag{1}$$

Thus, the internal reals are again based on the intensional signed-digit notation.

The next proposition, which says that $\mathsf{R_I}$ is the object of reals as described in the internal logic of the topos $\mathbf{RT}(\mathsf{P})$, explains our nomenclature for $\mathsf{R_I}$.

Proposition 1. *The object* $\mathsf{R_I}$*, when transported along*

$$\omega\mathbf{Equ} \xrightarrow{\simeq} \mathbf{Asm}(\omega\mathbf{L}) \xrightarrow{\simeq} \mathbf{Asm}(\mathsf{P}) \longrightarrow \mathbf{RT}(\mathsf{P}) \ \ ,$$

gives the object of Dedekind (equivalently Cauchy) reals in $\mathbf{RT}(\mathsf{P})$*.*

We use the cartesian-closed structure to determine two type hierarchies, the *external* $(\![\sigma]\!)_E$, and the *internal* $(\![\sigma]\!)_I$ in $\omega\mathbf{Equ}$, defined at base type by:

$$(\![\mathsf{real}]\!)_E = \mathsf{R_E} \qquad\qquad (\![\mathsf{real}]\!)_I = \mathsf{R_I} \ .$$

We write $(\sigma)_E$ as an abbreviation for $|(\![\sigma]\!)_E|$, and $(\sigma)_I$ for $|(\![\sigma]\!)_I|$.

Theorem 2. *For any type σ with* $\mathrm{order}(\sigma) \leq 2$*, it holds that* $(\sigma)_E = (\sigma)_I$*.*

5 Proofs of Theorems 1 and 2

In this section, we outline the proof of Theorem 2 and the derivation of Theorem 1 from Theorem 2.

We shall need to consider various types of topological spaces. A space is said to be *zero-dimensional* if every neighbourhood of a point has a clopen subneighbourhood, where a *clopen* set is one that is both open and closed.

In a topological space T, an infinite sequence $(x_i)_{i \geq 0}$ *converges* to a point x, notation $(x_i) \to x$, if, for all neighbourhoods $U \ni x$, the sequence (x_i) is eventually in U (i.e., there exists $l \geq 0$ such that $x_j \in U$ for all $j \geq l$). A subset $X \subseteq T$ is *sequentially open* if, whenever $(x_i) \to x \in X$, it holds that (x_i) is eventually in X. Every open set is sequentially open. A space T is said to be *sequential* if every sequentially open subset is open. We write **Seq** for the category of sequential spaces. This category is known to be cartesian closed. If S and T are sequential then the exponential T^S is given by the set of all continuous functions endowed with the unique sequential topology that induces the convergence relation $(f_i) \to f$ if and only if, whenever $(x_i) \to x$ in S, it holds that $(f_i(x_i)) \to f(x)$ in T.

We write $\omega\mathbf{qTop}$ for the category of all quotient spaces of countably-based spaces, i.e. a topological space T is an object of $\omega\mathbf{qTop}$ if and only if there exists a countably-based space S with a topological quotient $q\colon S \twoheadrightarrow T$. There are subcategory inclusions $\omega\mathbf{Top} \hookrightarrow \omega\mathbf{qTop} \hookrightarrow \mathbf{Seq}$. Importantly, the category $\omega\mathbf{qTop}$ is cartesian closed with its cartesian-closed structure inherited from $\mathbf{Seq}$ [MS02].

A topological space is said to be *hereditarily Lindelöf* if, for every family $\{U_i\}_{i\in I}$ of open sets, there is a countable subfamily $\{U_j\}_{j\in J}$ (i.e. where $J \subseteq I$ is countable) such that $\bigcup_{j\in J} U_j = \bigcup_{i\in I} U_i$. It is easily shown that every space in $\omega\mathbf{qTop}$ is hereditarily Lindelöf.

The next proposition relates the above notions to an important property of the function $\mathsf{r}\colon \mathbb{N} \times 3^\omega \to \mathbb{R}$, defined in (1), which is a topological quotient. We first introduce terminology that makes sense in an arbitrary category. Given an object Z and a morphism $g\colon X \longrightarrow Y$ we say that Z is *g-projective*, or equivalently that *g projects* Z, if, for every $f\colon Z \longrightarrow Y$, there exists $\overline{f}\colon Z \to X$ such that the left-hand diagram below commutes. Dually, we say that Z is *g-injective*, or equivalently that *g injects* Z, if, for every $f\colon X \longrightarrow Z$, there exists $\overline{f}\colon Y \to Z$ such that the right-hand diagram commutes.

$$\begin{tikzcd} Z \arrow[r, "\overline{f}"] \arrow[dr, "f"'] & X \arrow[d, "g"] \\ & Y \end{tikzcd} \qquad\qquad \begin{tikzcd} Y \arrow[r, "\overline{f}"] & Z \\ X \arrow[u, "g"] \arrow[ur, "f"'] & \end{tikzcd}$$

Proposition 2. *Zero-dimensional hereditarily Lindelöf spaces are r-projective.*

Consider the full subcategory $\omega\mathbf{0Equ}$ of $\omega\mathbf{Equ}$ consisting of those equilogical spaces that are isomorphic to one X for which $\|X\|$ is zero-dimensional. Easily, $\omega\mathbf{0Equ}$ is closed under finite products, and it contains every countably-based zero-dimensional space under the inclusion of $\omega\mathbf{Top}$ in $\omega\mathbf{Equ}$. Moreover, using Proposition 2, one can prove that $\omega\mathbf{0Equ}$ contains the objects $(\![\sigma]\!)_I$ for σ with $\mathrm{order}(\sigma) \leq 1$.

We say that a morphism $e\colon X \to Y$ in $\omega\mathbf{Equ}$ is *tight* if it is mono and it projects every space in $\omega\mathbf{0Equ}$. Every tight morphism is also epi. We say that an equilogical space is *tight-injective* if it is injective with respect to every tight map. It can be proved that the full subcategory $\omega\mathbf{Equ}_{\mathbf{ti}}$ of tight-injective objects in $\omega\mathbf{Equ}$ is cartesian closed and contains every countably-based space. Thus every object $(\![\sigma]\!)_E$ is tight-injective.

We prove Theorem 2 by constructing tight morphisms $(\![\sigma]\!)_I \to (\![\sigma]\!)_E$ when $\mathrm{order}(\sigma) \leq 2$. The first lemma gives the crucial construction for function types.

Lemma 1. *Given tight maps $e\colon X \to Z$ and $f\colon Y \to W$, where Y is in $\omega\mathbf{0Equ}$ and Z, W are in $\omega\mathbf{Equ}_{\mathbf{ti}}$, then the function $f^{e^{-1}}\colon |Y|^{|X|} \to |W|^{|Z|}$ restricts to a function $g\colon |Y^X| \to |W^Z|$ giving a tight morphism $g\colon Y^X \to W^Z$.*

Lemma 2. *If $\mathrm{order}(\sigma) \leq 2$ then $(\sigma)_I = (\sigma)_E$ and the identity function gives a tight morphism $(\![\sigma]\!)_I \to (\![\sigma]\!)_E$.*

Theorem 2 is an immediate consequence.

We next consider how Theorem 2 might be extended to higher types. Certainly, the proof above does not extend directly, because one can show that $(\![(\mathsf{real} \to \mathsf{real}) \to \mathsf{real}]\!)_I$ is not in $\omega\mathbf{0Equ}$. However, this leaves open the possibility of replacing the use of $\omega\mathbf{0Equ}$ with that of another category.

Proposition 3. *Suppose there exists a full subcategory of $\omega\mathbf{Equ}$ satisfying four conditions: (i) it is closed under finite products; (ii) it contains the 1-point compactification of $\mathbb{N}$; (iii) it contains every object $(\![\sigma]\!)_I$; (iv) every object in the subcategory is projective with respect to the "identity" $\mathsf{R_I} \to \mathsf{R_E}$. Then $(\sigma)_E = (\sigma)_I$ for all types σ.*

We do not know whether such a subcategory exists. The difficult conditions to reconcile are (iii) and (iv). Let us pinpoint our ignorance more exactly by considering the "pure" second- and third-order types:

$$\mathsf{real}_2 \equiv (\mathsf{real} \to \mathsf{real}) \to \mathsf{real} \qquad\qquad \mathsf{real}_3 \equiv \mathsf{real}_2 \to \mathsf{real}$$

Proposition 4.

1. $(\mathsf{real}_3)_E \supseteq (\mathsf{real}_3)_I$.
2. $(\mathsf{real}_3)_E = (\mathsf{real}_3)_I$ *if and only if the object* $(\![\mathsf{real}_2]\!)_I$ *is projective with respect to the identity* $\mathsf{R_I} \to \mathsf{R_E}$ *(tracked by* r*).*

We have not succeeded in establishing whether $(\![\mathsf{real}_2]\!)_I$ is projective with respect to the identity $\mathsf{R_I} \to \mathsf{R_E}$. However, we have managed to reduce this condition to a conjecture concerning the topology of the Kleene-Kreisel continuous functionals over $\mathbb{N}$ [Kle59,Kre59]. Many presentations of the continuous functionals are known, but, for our conjecture, the simplest description is as the hierarchy of simple types over $\mathbb{N}$ in the cartesian-closed category $\omega\mathbf{qTop}$, or equivalently in $\mathbf{Seq}$, or equivalently in the cartesian-closed category of compactly-generated Hausdorff spaces, see [Nor80].

Conjecture 1. The sequential space $\mathbb{N}^{\mathbb{B}}$, where $\mathbb{B} = \mathbb{N}^{\mathbb{N}}$, is zero dimensional.

Proposition 5. *If Conjecture 1 holds then* $(\![\mathsf{real}_2]\!)_I$ *is projective with respect to the identity* $\mathsf{R_I} \to \mathsf{R_E}$ *and hence* $(\mathsf{real}_3)_E = (\mathsf{real}_3)_I$.

We prove Theorem 1 by reducing it to Theorem 2. This requires some work. Although we have stated that $\omega\mathbf{Equ}$ is equivalent to the full subcategory $\mathbf{Asm}(\omega\mathbf{L})$ of $\mathbf{Asm}(\omega\mathbf{BC})$, this is of no immediate help because neither $[\![\mathsf{real}]\!]_E$ nor $[\![\mathsf{real}]\!]_I$ resides in this subcategory. However, following [Bau00], there is a second way of viewing $\mathbf{Asm}(\omega\mathbf{L})$, and hence $\omega\mathbf{Equ}$, as (equivalent to) a full subcategory of $\mathbf{Asm}(\omega\mathbf{BC})$, under which $[\![\mathsf{real}]\!]_E$ and $[\![\mathsf{real}]\!]_I$ are included.

We say that an assembly A in $\mathbf{Asm}(\omega\mathbf{BC})$ is *dense* if $\mathrm{supp}(A)$ is dense in $\|A\|$ under the Scott topology, where:

$$\mathrm{supp}(A) = \{x \in \|A\| \mid \text{there exists } a \in |A| \text{ such that } x \Vdash a\}\ .$$

An essembly is *essentially dense* if it is isomorphic to a dense assembly. It holds that $\mathbf{Asm_{ed}}(\omega\mathbf{BC}) \simeq \omega\mathbf{Equ}$, where $\mathbf{Asm_{ed}}(\omega\mathbf{BC})$ is the full subcategory of essentially dense assemblies in $\mathbf{Asm}(\omega\mathbf{BC})$ [BBS02,Bau00].

Proposition 6. *For any type σ,*

1. $[\![\sigma]\!]_E$ *is a dense assembly, and*
2. *if* $\mathrm{order}(\sigma) \leq 1$ *then* $[\![\sigma]\!]_I$ *is an essentially dense assembly.*

Statement 1 follows from Normann's density theorem for an ω-algebraic variant of the interval domain [Nor00b]. Statement 2 will be addressed in Sect. 6.

Lemma 3. *For any type σ,*

1. $[\sigma]_E = (\sigma)_E$*, and*
2. *if* $\mathrm{order}(\sigma) \leq 2$ *then* $[\sigma]_I = (\sigma)_I$.

Theorem 1 follows immediately from Lemma 3 and Theorem 2.

6 Extensionalization

In this section, we prove Proposition 6.2. Our proof establishes a property of the domains underlying $[\![\sigma]\!]_I$, for first-order σ, that we call *extensionalization*. This property is of interest independent of its application to Proposition 6.2.

Following [Ber93], we define the set of *total elements* $\mathcal{T}_\tau \subseteq \|\tau\|$, where $\|\tau\|$ is the domain interpreting a **PCF** type τ. This is by:

$$\begin{aligned} \mathcal{T}_{\mathsf{nat}} &= \mathbb{N} \quad \subseteq \mathbb{N}_\perp \\ \mathcal{T}_{\tau_1 \times \tau_2} &= \mathcal{T}_{\tau_1} \times \mathcal{T}_{\tau_2} \\ \mathcal{T}_{\tau_1 \to \tau_2} &= \{ f \in \|\tau_1 \to \tau_2\| \mid \text{for all } x \in \mathcal{T}_{\tau_1},\ f(x) \in \mathcal{T}_{\tau_2} \} \end{aligned}$$

Recall also that, for a type σ over real, $\|[\![\sigma]\!]_I\| = \|\sigma^*\|$, where $(\cdot)^*$ is the translation to **PCF** types from Sect. 3. The proof of the next proposition uses Berger's generalization of the "KLS Theorem" [Ber93] together with a purely topological lemma.

Lemma 4. *If S is a nonempty closed subspace of a countably-based zero-dimensional space T then S is a retract of T.*

Proposition 7 (Extensionalization). *For any σ with* $\mathrm{order}(\sigma) \leq 1$*, the identity function on $[\![\sigma]\!]_I$ is tracked by a function $i : \|[\![\sigma]\!]_I\| \to \|[\![\sigma]\!]_I\|$ with the property that, for all $x \in \mathcal{T}_{\sigma^*}$,* $i(x) \in \mathcal{T}_{\sigma^*} \cap \mathrm{supp}([\![\sigma]\!]_I)$.

We call this result extensionalization for the following reason. As in Sect. 3, in order for an element $f \in \|[\![\mathsf{real} \to \mathsf{real}]\!]_I\|$ to track a morphism from $[\![\mathsf{real}]\!]_I$ to $[\![\mathsf{real}]\!]_I$ it must both preserve real-representing elements (i.e. it must preserve $\mathrm{supp}([\![\mathsf{real}]\!]_I)$) and it must also preserve the equivalence between such representations; thus one might say that it must behave "extensionally". The proposition relates such "extensional" elements of $\|[\![\mathsf{real} \to \mathsf{real}]\!]_I\|$ to total ones. Firstly, because i tracks the identity, it maps every extensional element f to an equivalent total extensional one. Secondly, every non-extensional but total f is mapped to an arbitrary extensional and still total element. Thus the total elements of $\|[\![\mathsf{real} \to \mathsf{real}]\!]_I\|$ are all "extensionalized" by i. Again we do not know whether such a process of extensionalization is also available for second-order σ and above.

Corollary 1. *If* $\mathrm{order}(\sigma) \leq 1$ *then the identity on* $[\![\sigma]\!]_I$ *is tracked by a function* i *such that, for all* $x \in \|[\![\sigma]\!]_I\|$, $i(x)$ *is in the Scott-closure of* $\mathrm{supp}([\![\sigma]\!]_I)$.

Proposition 6.2 follows, as the property stated in the corollary is easily seen to be sufficient to establish that $[\![\sigma]\!]_I$ is essentially dense.

7 Eliminating Parallelism

To conclude the paper, we return to our original motivation for studying the $[\![\sigma]\!]_E$ and $[\![\sigma]\!]_I$ hierarchies, namely that they correspond to the total functionals on reals definable in the two approaches to exact real-number computation. As discussed in Sect. 3, the $[\![\sigma]\!]_E$ functionals are exactly those programmable in Ω**RealPCF+**, and the functionals in $[\![\sigma]\!]_I$ are those programmable in Ω**PCF++**. Both these languages contain parallel primitives.

In the context of **PCF**, Normann has proved that the type hierarchies of total functionals over $\mathbb{N}$ programmable in **PCF** and **PCF++** are identical for arbitrary types [Nor00a]. By the same proof, the hierarchies of $\mathbb{N}$-functionals programmable in Ω**PCF** and Ω**PCF++** are identical. In other words, parallel primitives are unnecessary as far as programming total functionals over $\mathbb{N}$ is concerned. It is natural to ask whether a similar phenomenon of elimination of parallelism occurs also for total functionals over $\mathbb{R}$.

For the extensional approach, the situation is unsatisfactory. In [EHS99], it is proved that there is no sequential way of implementing even the first-order function of binary addition. For this reason, core **RealPCF** contains a primitive parallel-conditional operation. However, one may still question whether the parallel existential of **RealPCF+** is required for programming total functionals. The only known result is that all second-order functionals can be defined in languages strictly weaker than **RealPCF+** [Nor02].

Our final result is that, in the intensional approach, parallelism is eliminable up to type two. Recall, from Sect. 3, our notation for **PCF** and its semantics.

Theorem 3. *If* $\mathrm{order}(\sigma) \leq 2$ *then, for any* $f \in [\sigma]_I$, *there exists an* Ω**PCF** *program* P *of type* σ^* *such that* $[\![P]\!] \Vdash_{[\![\sigma]\!]_I} f$.

The proof uses extensionalization, Proposition 7, to reduce the result to Normann's result for third-order **PCF** types. The type restriction on Proposition 7 is the only obstacle to extending Theorem 3 to higher-order types.

To ease comparison with the results for the extensional case discussed above, we remark that we have also proved a version of Theorem 3 for the standard (oracle free) versions of **PCF** and **PCF++**. Specifically, a total functional on $\mathbb{R}$ is definable in **PCF** if and only if it is definable in **PCF++**. The proof involves writing **PCF** programs for the extensionalization functions i of Proposition 7, and thus uses methods quite different from those of this paper.

References

[AJ94] S. Abramsky and A. Jung. Domain theory. In *Handbook of Logic in Computer Science*, volume 3, pages 1–168. OUP, 1994.

[Bau00] A. Bauer. *The Realizability Approach to Computable Analysis and Topology*. PhD thesis, Carnegie Mellon University, 2000.

[BBS02] A. Bauer, L. Birkedal, and D.S. Scott. Equilogical spaces. *Theoretical Computer Science*, to appear, 2002.

[Ber93] U. Berger. Total sets and objects in domain theory. *Annals of Pure and Applied Logic*, 60:91–117, 1993.

[DiG93] P. Di Gianantonio. *A Functional Approach to Computability on Real Numbers*. PhD thesis, Università di Pisa, 1993.

[Dug89] J. Dugundji. *Topology*. Wm. C. Brown, 1989.

[EE00] M.H. Escardó and A. Edalat. Integration in real PCF. *Information and Computation*, 160:128–166, 2000.

[EHS99] M.H. Escardó, M. Hofmann, and Th. Streicher. *Mediation is inherently parallel*. Talk at Workshop on Domains V, Darmstadt, 1999.

[ES99] M.H Escardó and Th. Streicher. Induction and recursion on the partial real line with applications to real PCF. *Theoretical Computer Science*, 210:121–157, 1999.

[Esc96] M.H. Escardó. PCF extended with real numbers. *Theoretical Computer Science*, 162:79–115, 1996.

[GL01] P. Gowland and D. Lester. A survey of exact computer arithmetic. In *Computability and Complexity in Analysis*. Springer LNCS 2064, 2001.

[GS90] C.A. Gunter and D.S. Scott. Semantic domains. In *Handbook of Theoretical Computer Science*. Elsevier Science Publisher, 1990.

[Has] `http://www.haskell.org/`.

[Kle59] S.C. Kleene. Countable functionals. In *Constructivity in Mathematics*, pages 81–100. North-Holland, 1959.

[Kre59] G. Kreisel. Interpretation of analysis by means of functionals of finite type. In *Constructivity in Mathematics*, pages 101–128. North-Holland, 1959.

[Mac71] S. Mac Lane. *Categories for the Working Mathematician*. Springer-Verlag, 1971.

[MS02] M. Menni and A.K. Simpson. Topological and limit-space subcategories of countably-based equilogical spaces. *Mathematical Structures in Computer Science*, to appear, 2002.

[Nor80] D. Normann. *Recursion on the countable functionals*. Springer LNM 811, 1980.

[Nor00a] D. Normann. Computabilty over the partial continuous functionals. *Journal of Symbolic Logic*, 65:1133–1142, 2000.

[Nor00b] D. Normann. The continuous functionals of finite types over the reals. *Electronic Notes in Theoretical Computer Science*, 35, 2000.

[Nor02] D. Normann. Exact real number computations relative to hereditarily total functionals. *Theoretical Computer Science*, to appear, 2002.

[Plo77] G.D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5(3):223–255, 1977.

[Sco96] D.S. Scott. A new category? Unpublished Manuscript. Available at `http://www.cs.cmu.edu/Groups/LTC/`, 1996.

[Sim98] A.K. Simpson. Lazy functional algorithms for exact real functionals. In *Mathematical Foundations of Computer Science*, pages 456–464. Springer LNCS 1450, 1998.

[WK87] K. Weihrauch and C. Kreitz. Representations of the real numbers and of the open subsets of the set of real numbers. *Annals of Pure and Applied Logic*, 35(3):247–260, 1987.