Theory Comput Syst (2008) 43: 394–409
DOI 10.1007/s00224-007-9017-6

Coinduction for Exact Real Number Computation

Ulrich Berger · Tie Hou

Published online: 27 July 2007

Abstract This paper studies coinductive representations of real numbers by signed digit streams and fast Cauchy sequences. It is shown how the associated coinductive principle can be used to give straightforward and easily implementable proofs of the equivalence of the two representations as well as the correctness of various corecursive exact real number algorithms. The basic framework is the classical theory of coinductive sets as greatest fixed points of monotone operators and hence is different from (though related to) the type theoretic approach by Ciaffaglione and Gianantonio.

Keywords Exact real number computation · Coinduction · Corecursion · Signed digit streams

1 Introduction

This paper explores the use of coinduction in exact real arithmetic. In exact real arithmetic one studies algorithms for performing operations on the reals whose results are guaranteed to be completely accurate. Complete accuracy can be achieved by representing real numbers as potentially infinite data structures such as streams. There are a number of different representations used for exact real arithmetic, such as integral base systems with positive and negative digits, base 2/3 with binary digits, nested sequences of rational intervals, Cauchy sequences [16], continued fractions [8], golden-ratio based systems with binary digits, as well as continued fractions and their generalisation, linear fractional (or Möbius) transformations [6, 7]. Many algorithms have

U. Berger (✉)
University of Wales Swansea, Swansea, SA2 8PP, Wales, UK
e-mail: u.berger@swansea.ac.uk

T. Hou
Informatics Institute, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
e-mail: hou@science.uva.nl

been proposed for real computations using these representations, however their correctness is rarely proved formally. Plume [14] developed algorithms for the basic arithmetic operations, transcendental functions and integration, for various representation, but gave only informal proofs of the correctness for some of the algorithms. A more formal approach was taken by Chirimar and Howe [5] as well as Schwichtenberg [16] who implemented and verified real number algorithms based on the Cauchy sequence representation in the systems Nuprl and Minlog, respectively.

Approaches to exact real arithmetic based on coinduction were undertaken by Ciaffaglione and Gianantonio [4], Bertot [1, 2] as well as Niqui [12], who formalised and implemented real numbers as a coinductive type of streams in the Coq proof assistant which is based on Martin-Löf Type theory. In the literature on coinduction in general, coinduction is usually defined as the principle expressing that equality is the largest bisimulation on streams (see e.g. [9, 15]). This form of coinduction is too restricted to express equality of real numbers given as infinite streams, due to the—for computability reasons inevitable—redundancy of real number representations. A more general form of coinduction, studied for example by Lenisa [11], is based on the fact that every monotone set operator has a greatest fixed point. In this paper, we follow the same approach: We define relations of the form "stream a represents real number r" as the greatest fixed points of monotone set operators, and use the corresponding coinduction proof principles to prove properties of these relations. More concretely, we study representations of real numbers by Cauchy sequences and signed digit streams and prove that both representations are equivalent. Furthermore, we prove the correctness of the implementations of addition, multiplication, division and limit with respect to signed digit streams given in [4]. For technical convenience, we restrict ourselves to the interval $[-1, 1]$ and hence study the average function instead of addition.

Coinduction is a method of growing importance for reasoning about infinite data structures, which occur not only in computable analysis, but also in, for example, lazy functional languages and process algebra. Moreover, proofs using coinduction seem to be well-suited for implementation in proof assistants (see, for example [1]). We will therefore, throughout this paper, keep an eye on aspects of formalisation and implementation of our notions and proofs.

2 Coinduction and Corecursion

In this section we introduce the concept of coinduction from a classical set-theoretic point of view. We prove a general lemma on closure properties of coinductive sets, which will be useful later. We also briefly discuss infinite streams and a simple corecursion scheme for stream producing functions.

We will use the following set-theoretic notation. If A is a set, then $\wp(A) := \{X | X \subseteq A\}$ is its power set. We will write "$X(a)$" interchangeably for "$a \in X$". If $f: A \to A$ is a function, then its iterations $f^n: A \to A$ $(n \in \mathbb{N})$ are defined by $f^0(a) := a$, $f^{n+1}(a) := f(f^n(a))$.

2.1 Coinductive Relations as Greatest Fixed Points

An operation $\Phi : \wp(A) \to \wp(A)$, is *monotone* iff $X \subseteq Y$ implies $\Phi(X) \subseteq \Phi(Y)$ for $X, Y \subseteq A$. It is well-known that any monotone operation Φ has a least and a greatest fixed point, X_Φ and X^Φ respectively, that is, $\Phi(X_\Phi) = X_\Phi$, $\Phi(X^\Phi) = X^\Phi$, and for any other fixed point $Y \subseteq A$ of Φ (i.e. $\Phi(Y) = Y$) we have $X_\Phi \subseteq Y \subseteq X^\Phi$. The sets X_Φ and X^Φ can be defined by

$$X_\Phi := \bigcap \{Y | Y \subseteq A, \Phi(Y) \subseteq Y\},$$

$$X^\Phi := \bigcup \{Y | Y \subseteq A, Y \subseteq \Phi(Y)\}.$$

It is easy to see that the monotonicity of Φ implies the required properties of X_Φ and X^Φ. In the following, we will concentrate on the greatest fixed point, X^Φ. By definition of X^Φ, we have for any set $Y \subseteq A$ that $Y \subseteq \Phi(Y)$ implies $Y \subseteq X^\Phi$. This principle is called *coinduction*.

In applications the operation Φ is usually described by a formula $F[X, a]$ as $\Phi(X) := \{a \in A | F[X, a]\}$. In this case the monotonicity of Φ can be guaranteed by the (sufficient but not necessary) condition that X does only occur positively in $F[X, a]$. For our purposes it will suffice to consider formulae of the form $F[X, a] := B(a) \wedge X\big(f(a)\big)$, where $f : A \to A$ is a fixed function and B is a fixed subset of A. Hence, $\Phi(X) := B \cap f^{-1}(X)$. Setting $R := X^\Phi$, the coinductive principle reads in this particular case

$$coind_{f,B}(Y) \quad \frac{\forall a \in A \,.\, Y(a) \Rightarrow B(a) \wedge Y(f(a))}{\forall a \in A \,.\, Y(a) \Rightarrow R(a)}.$$

The closure condition, $R \subseteq \Phi(R)$, then reads

$$(R) \quad \forall a \in A \,.\, R(a) \Rightarrow B(a) \wedge R(f(a)).$$

Because in fact $R = \Phi(R)$, the reverse of this implication holds as well. We will simply say that R is coinductively defined by B and f via the rule (R).

Lemma 1 *Let $R \subseteq A$ be coinductively defined by B and f. Then*

$$R(a) \Leftrightarrow \forall n \in \mathbb{N}\, B(f^n(a)).$$

Hence R is the set of all points whose f-orbit is contained in B.

Proof ($\Rightarrow$) Induction on n.

($\Leftarrow$) Set $Y(a) := \forall n\, B(f^n(a))$. Clearly, $Y(a)$ implies $B(a)$ and $Y(f(a))$. Hence $Y \subseteq R$, by coinduction. □

2.2 Closure Properties of Coinductive Relations

Consider $R \subseteq A$, coinductively defined by $B \subseteq A$ and $f : A \to A$. We are interested in the question under which conditions R is closed under a given function. The following lemma takes care of a slightly more general situation.

Lemma 2 *Let X be a set, $g : X \to X$, $h : X \to A$ functions, and $S \subseteq X$ such that for all $x \in X$, if $S(x)$, then*

1. $S(g(x))$,
2. $f(h(x)) = h(g(x))$,
3. $B(h(x))$.

Then $\forall x \in X(S(x) \Rightarrow R(h(x)))$.

Proof Assume the hypotheses of the lemma and assume $S(x)$. According to Lemma 1, it suffices to show $\forall n\ B(f^n(h(x)))$. From the first hypothesis it follows $S(g^n(x))$, by induction on n. Hence $B(h(g^n(x)))$, by the third hypothesis. But from the second hypothesis, using induction again, we obtain that $h(g^n(x)) = f^n(h(x))$. □

Remark Of course, this lemma can also be proven by coinduction directly, setting $Y(a) := \exists x \in X(S(x) \wedge a = h(x) \wedge R(a))$, without using Lemma 1 and without induction on natural numbers.

2.3 Infinite Streams and Corecursion

If X is a set, then X^ω denotes the set of *infinite streams* (or infinite lists) of elements in X (from a mathematical point of view, infinite streams are the same as functions from the natural numbers to X, but in lazy functional programming languages infinite streams and functions are implemented differently). If $x \in X^\omega$ is an infinite stream, then $\mathrm{head}(x) \in X$ is its head and $\mathrm{tail}(x) \in X^\omega$ its tail, that is $x = \mathrm{head}(x) : \mathrm{tail}(x)$ where ":" is the "cons" operation. We set $x_n := \mathrm{head}(\mathrm{tail}^n(x))$, and, for notational convenience, also $x' := \mathrm{tail}(x)$. Hence $x_0 = \mathrm{head}(x)$, $x_{n+1} = (x')_n$ and $x = x_0 : x_1 : x_2 : \ldots$.

We say a function $\phi : Y \to X^\omega$ is defined by *corecursion* from $H : Y \to X$ and $T : Y \to Y$ if

$$\phi(y) = H(y) : \phi(T(y)) \qquad (*)$$

for all $y \in Y$. In the literature this scheme is often also called *coiteration* [11], or *coinduction* [9]. It is easy to see that, given H and T, the function ϕ always exists and is uniquely determined by $(*)$. In fact, $\phi(y) = x_0 : x_1 : x_2 : \ldots$ where $x_n = H(T^n(y))$. This also shows that ϕ is computable whenever H and T are (assuming a notion of computability is defined for functions on X and Y). In a lazy functional programming language $(*)$ can also be used directly for computing $\phi(y)$. The category-theoretic and type-theoretic status of corecursion and its relation the proof principle of coinduction is explained, for example, in [9] and [4].

All the recursive definitions of stream producing functions in this paper will be instances of the simple corecursion scheme $(*)$. More general schemes have been discussed, for example, by Telford and Turner [18], and Buchholz [3]. Important examples of recursive definitions of stream functions related to exact real number computation that do not fit into the recursion scheme $(*)$ can be found in [6, 7, 12].

3 Coinductive Representations of Real Numbers by Signed Digit Streams and Cauchy Sequences

In this section we show how to represent real numbers by streams of signed digits and Cauchy sequences of rational numbers (given as infinite streams) using coinductively defined representation relations. We will prove that these representations are equivalent to the usual ones involving the notion of infinite sum and limits from analysis.

Throughout the rest of this paper (except for Lemma 10) we will adhere to the following naming conventions:

r, p, q	$\in \mathbb{R}$	real numbers
u, v, w	$\in \mathbb{Q}$	rational numbers
n, m	$\in \mathbb{N}$	natural numbers $(0, 1, 2, \ldots)$
i, j, k, l	$\in \mathbb{Z}$	integers $(\ldots, -2, -1, 0, 1, 2, \ldots)$
δ	$\in \mathrm{SD}$	signed digits $(-1, 0, 1)$
a, b, c, d, e	$\in \mathrm{SD}^\omega$	streams of signed digits
x, y, z	$\in \mathbb{Q}^\omega$	streams of rational numbers

3.1 Coinductive Representation by Signed Digit Streams

Let $\mathrm{SD} := \{-1, 0, 1\}$ be the set of *signed digits*. If $a = a_0 : a_1 : a_2 : \ldots \in \mathrm{SD}^\omega$ is a signed digit stream, then

$$\sigma(a) := \sum_{n=0}^{\infty} a_n 2^{-(n+1)}$$

is called the real number represented by a. We clearly have:

Lemma 3 *$|\sigma(a)| \leq 1$ and $\sigma(a') = 2\sigma(a) - a_0$.*

This suggests the following coinductive definition of a relation $\sim \subseteq \mathrm{SD}^\omega \times \mathbb{R}$ by $f(a, r) := (a', 2r - a_0)$ and $B(a, r) := |r| \leq 1$ with the intended meaning $a \sim r \Leftrightarrow \sigma(a) = r$:

Definition 4 (Coinductive definition of $a \sim r$)

$$(\sim) \quad a \sim r \Rightarrow |r| \leq 1 \wedge a' \sim 2r - a_0.$$

The advantage of using $\sim$ instead of the function σ is that $\sim$ does not use the (logically complicated) notion of limit (in the form of an infinite sum).

Lemma 5 (Correctness of $\sim$) *$a \sim r \Leftrightarrow \sigma(a) = r$.*

Proof ($\Longrightarrow$) We show, by induction on n,

$$\forall n \in \mathbb{N} \forall a, r\, (a \sim r \Rightarrow |\sigma(a) - r| \leq 2^{1-n}).$$

$n = 0$: By Definition 4, $a \sim r$ implies $|r| \leq 1$. Since also $|\sigma(a)| \leq 1$, we have $|\sigma(a) - r| \leq 2$.

$n + 1$: Assume $a \sim r$. We need to show $|\sigma(a) - r| \leq 2^{1-(n+1)}$. By Definition 4, we have $a' \sim 2r - a_0$. By I.H., $|\sigma(a') - (2r - a_0)| \leq 2^{1-n}$. Hence, by Lemma 3,

$$|\sigma(a) - r| = \left|\frac{a_0 + \sigma(a')}{2} - r\right| = \frac{|\sigma(a') + a_0 - 2r|}{2} \leq \frac{2^{1-n}}{2} = 2^{1-(n+1)}.$$

($\Longleftarrow$) Set $Y := \{(a, r) | \sigma(a) = r\}$. By Lemma 3, $Y(a, r)$ implies $|r| \leq 1$ and $Y(a', 2r - a_0)$. Hence $Y \subseteq \sim$, by coinduction. □

3.2 Coinductive Representation by Cauchy Sequences

We call a stream of rational numbers, $x \in \mathbb{Q}^\omega$, an *l-Cauchy sequence* ($l \in \mathbb{Z}$) if

$$\forall n \forall m \geq n . |x_n - x_m| \leq 2^{l-n}.$$

We set $\mathrm{CR} = \mathbb{Q}^\omega \times \mathbb{Z}$, and define coinductively a relation $\sim^{\mathrm{c}} \subseteq \mathrm{CR} \times \mathbb{R}$ by $f((x, l), r) = ((x', l - 1), r)$ and $B((x, l), r) = |x_0 - r| \leq 2^l$ with the intended meaning: $(x, l) \sim^{\mathrm{c}} r \Leftrightarrow$ *x is an l-Cauchy sequence converging to r*.

Definition 6 (Coinductive definition of $(x, l) \sim^{\mathrm{c}} r$)

$$(\sim^{\mathrm{c}}) \quad (x, l) \sim^{\mathrm{c}} r \Rightarrow |x_0 - r| \leq 2^l \wedge (x', l - 1) \sim^{\mathrm{c}} r.$$

Lemma 7 $(x, l) \sim^{\mathrm{c}} r \Leftrightarrow \forall n \in \mathbb{N} . |x_n - r| \leq 2^{l-n}$.

Proof Immediate, by Lemma 1. □

4 Adequacy of the Signed Digit Stream Representation

It is common to regard the Cauchy sequence representation of real numbers as the standard one. We call any other representation adequate if there are computable back-and-forth translations between it and the Cauchy representation.

The concept of computability on infinite streams can be explained by means of 'Oracle Turing machines' (Turing [19]). More recent accounts of the computability and complexity of stream functions are studied by e.g. Ko [10] and Weihrauch [20]. All the stream functions defined in this paper will be defined either explicitly from previously defined functions or by the corecursion scheme discussed in Sect. 2.3. Hence they are clearly computable.

In order to show that the signed digit stream representation is adequate, we need to provide computable functions $\varphi : \mathrm{SD}^\omega \to \mathrm{CR}$ and $\psi : \mathrm{CR} \to \mathrm{SD}^\omega$, such that for all $r \in \mathbb{R}$,

1. $\forall a \in \mathrm{SD}^\omega\ (a \sim r \Rightarrow \varphi(a) \sim^{\mathrm{c}} r)$,
2. $\forall (x,l) \in \mathrm{CR}\ ((x,l) \sim^{\mathrm{c}} r \Rightarrow \psi(x,l) \sim r)$.

4.1 From Signed Digit Streams to Cauchy Sequences

The translation from signed digit streams to Cauchy sequences is straightforward.

Definition 8 We define $\varphi_{\mathrm{aux}} : \mathbb{Z} \times \mathbb{Q} \times \mathrm{SD}^\omega \to \mathbb{Q}^\omega$ by

$$\varphi_{\mathrm{aux}}(k,u,a) = (u + 2^{k-1}a_0)\ :\ \varphi_{\mathrm{aux}}(k-1, u + 2^{k-1}a_0, a')$$

Note that this definition of φ_{aux} is an instance of the corecursion scheme ($*$) in Sect. 2.3.

Lemma 9 (Converting signed digit streams into Cauchy sequences) *If $a \sim r$, then* $(\varphi_{\mathrm{aux}}(k,u,a), k-1) \sim^{\mathrm{c}} u + 2^k r$.

Proof We use Lemma 2 with the following values taken for S, g, h, B, f.

$$\begin{aligned}
&S(k,u,a,r) := a \sim r,\\
&g(k,u,a,r) := (k-1, u + 2^{k-1}a_0, a', 2r - a_0),\\
&h(k,u,a,r) := ((\varphi_{\mathrm{aux}}(k,u,a), k-1), u + 2^k r),\\
&B((x,l),r) := |x_0 - r| \le 2^l,\\
&f((x,l),r) := ((x', l-1), r).
\end{aligned}$$

Assume $S(k,u,a,r)$, that is, $a \sim r$. We need to show:

1. $S(g(k,u,a,r))$, that is, $a' \sim 2r - a_0$.
2. $f(h(k,u,a,r)) = h(g(k,u,a,r))$.
3. $B(h(k,u,a,r))$.

Condition 1 holds, by the definition of $\sim$. Condition 2 holds because

$$f(h(k,u,a,r)) = ((\varphi_{\mathrm{aux}}(k-1, u + 2^{k-1}a_0, a'), k-2), u + 2^k r) = h(g(k,u,a,r)).$$

Finally, condition 3 holds because from $a \sim r$ we get $|2r - a_0| \le 1$ and therefore

$$|u + 2^{k-1}a_0 - (u + 2^k r)| = |2^{k-1}a_0 - 2^k r| \le 2^{k-1}. \qquad \square$$

4.2 From Cauchy Sequences to Signed Digit Streams

Our next task is to define a translation from the Cauchy representation to the signed digit representation. First, we restate a special instance of Lemma 2 (where $R = \sim$) in a form that is convenient to use.

Lemma 10 *Let X be a set, $g : X \to X$, $h : X \to \mathrm{SD}^\omega \times \mathbb{R}$ functions, and $S \subseteq X$ such that for all $x \in S$:*

1. $S(g(x))$,
2. *If* $h(x) = (\delta : a, r)$, *then* $|r| \leq 1$ *and* $h(g(x)) = (a, 2r - \delta)$.

Then for all $x \in S$, *if* $h(x) = (a, r)$, *then* $a \sim r$.

The following generalised sign functions $\mathrm{sg}_\epsilon : \mathbb{Q} \to \mathrm{SD}$, for rational numbers $\epsilon \geq 0$, will be useful as well:

$$\mathrm{sg}_\epsilon(u) = \begin{cases} 1 & \text{if } u > \epsilon, \\ 0 & \text{if } |u| \leq \epsilon, \\ -1 & \text{if } u < -\epsilon. \end{cases}$$

Lemma 11 (a) *If* $|r| \leq 1$ *and* $|r - u| \leq 1/4$, *then* $|2r - \mathrm{sg}_{1/4}(u)| \leq 1$.
(b) *If* $|u| \leq 6$, *then* $|u - 4\mathrm{sg}_2(u)| \leq 2$.

Proof Easy case analysis. □

The following function ψ_{aux} solves a generalisation of the problem of translating l-Cauchy sequences to signed digit streams. It represents the mapping $(u, r, n) \mapsto 2^n(r - u)$ where the r in the input is given as an l-Cauchy sequence and the output is produced as a stream of signed digits. Intuitively, the reason for the extra argument n is that scaling is necessary as one progresses into the input stream. The role of u can be explained as follows: In order to determine the first digit of the output one needs to locate $2^n(r - u)$ in one of the intervals $[-1, 0]$, $[-1/2, 1/2]$, or $[0, 1]$. This is done by using the sign function and Lemma 11. The argument u accumulates the weighted digits produced so far. Most of the corecursive definitions later in the paper are based on similar ideas.

Definition 12

$$\begin{aligned} \psi_{\mathrm{aux}} : \quad & \mathbb{N} \times \mathbb{Q} \times \mathbb{Q}^\omega \to \mathrm{SD}^\omega \\ \psi_{\mathrm{aux}} \quad & (n, u, x) = \delta : \psi_{\mathrm{aux}}(n + 1, v, x'), \\ & \text{where} \quad \delta := \mathrm{sg}_{1/4}(2^n(x_0 - u)), \ v := u + 2^{-(n+1)}\delta. \end{aligned}$$

We set $\psi(x, l) := \psi_{\mathrm{aux}}(0, 0, \mathrm{tail}^m(x))$ where $m := \max(0, l + 2)$.

Lemma 13 (Converting Cauchy sequences into signed digit streams) *If* $(x, -(n + 2)) \sim^{\mathrm{c}} r$ *and* $|2^n(r - u)| \leq 1$, *then* $\psi_{\mathrm{aux}}(n, u, x) \sim 2^n(r - u)$ (*where* $n \in \mathbb{N}$).

Proof Fixing $r \in \mathbb{R}$, we apply Lemma 10 with

$$\begin{aligned} S(u, x, n) &:= (x, -(n + 2)) \sim^{\mathrm{c}} r \wedge |2^n(r - u)| \leq 1, \\ h(u, x, n) &:= (\psi_{\mathrm{aux}}(n, u, x), 2^n(r - u)), \\ g(u, x, n) &:= (v, x', n + 1) \quad \text{with } v \text{ as in the definition of } \psi_{\mathrm{aux}}. \end{aligned}$$

Assume $S(u, x, n)$.

1. $S(v, x', n+1)$: $(x', -(n+3)) \sim^{\mathrm{c}} r$ follows from $(x, -(n+2)) \sim^{\mathrm{c}} r$. Furthermore, since $(x, -(n+2)) \sim^{\mathrm{c}} r$, we have $|x_0 - r| \leq 2^{-(n+2)}$ and therefore $|2^n(r-u) - 2^n(x_0 - u)| \leq 1/4$. Since we also have $|2^n(r-u)| \leq 1$, it follows with Lemma 11(a) that $|2^{n+1}(r-v)| = |2^{n+1}(r-u) - \delta| \leq 1$.
2. $h(u, x, n) = (\delta : \psi_{\mathrm{aux}}(n+1, v, x'), 2^n(r-u))$ where $|2^n(r-u)| \leq 1$. Furthermore, $h(g(u, x, n)) = (\psi_{\mathrm{aux}}(n+1, v, x'), 2^{n+1}(r-v))$, which is as required since $2 \cdot 2^n(r-u) - \delta = 2^{n+1}(r-v)$. □

Theorem 1 (Adequacy of the signed digit representation) (1) *If* $a \sim r$, *then* $\varphi(a) \sim^{\mathrm{c}} r$.

(2) *If* $(x, l) \sim^{\mathrm{c}} r$ *and* $|r| \leq 1$, *then* $\psi(x, l) \sim r$.

Proof (1) By Lemma 9 with $u = 0$ and $k = 0$.

(2) By Lemma 13 with $u = 0$ and $n = 0$ using the fact that $(x, l) \sim^{\mathrm{c}} r$ implies $(\mathrm{tail}^m(x), -2) \sim^{\mathrm{c}} r$ for $m = \max(0, l+2)$. □

5 Arithmetic Operations on Signed Digit Streams

In this section we define corecursive operations on signed digit streams representing addition, multiplication, division, and limit of fast converging sequences, all in the interval $[-1, 1]$. Since $[-1, 1]$ is not closed under addition and division we replace addition by the average function and restrict division p/q to (streams representing) $q \in [1/4, 1]$ and $p \in [-q, q]$. The definitions are taken (with some modifications) from Ciaffaglione and Gianantonio [4].

The following functions will be useful.

Definition 14

$$\mathrm{appr}_n : \mathrm{SD}^\omega \to \mathbb{Z} \quad (n \in \mathbb{N})$$

$$\mathrm{appr}_n(a) = \sum_{m=0}^{n-1} 2^{n-1-m} a_m = 2^{n-1}a_0 + 2^{n-2}a_1 + \cdots + a_{n-1}$$

$$\mathrm{add}: \mathbb{Q} \times \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$

$$\mathrm{add}(u, a) = \delta : \mathrm{add}(2u + a_0 - \delta, a') \text{ where } \delta = \mathrm{sg}_{1/4}\left(u + \frac{1}{2}a_0 + \frac{1}{4}a_1\right)$$

$$\mathrm{scale}_n: \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$

$$\mathrm{scale}_n(a) = \mathrm{add}(\mathrm{appr}_n(a), \mathrm{tail}^n(a)) \quad (n \in \mathbb{N})$$

$$\mathrm{mdig} : \mathrm{SD} \times \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$

$$\mathrm{mdig}(\delta, a) = \delta a_0 : \mathrm{mdig}(\delta, a')$$

Lemma 15 *Assume $a \sim r$ and let $p \in \mathbb{R}$, $n \in \mathbb{N}$, $u \in \mathbb{Q}$ and $\delta \in \mathrm{SD}$.*

(a) $\mathrm{tail}^n(a) \sim (2^n r - \mathrm{appr}_n(a))$, *in particular* $|2^n r - \mathrm{appr}_n(a)| \leq 1$.
(b) *If* $|u + r| \leq 1$, *then* $\mathrm{add}(u, a) \sim u + r$.
(c) *If* $|2^n r| \leq 1$, *then* $\mathrm{scale}_n(a) \sim 2^n r$.
(d) $\mathrm{mdig}(\delta, a) \sim \delta r$.

Proof Part (a) holds since $f^n(a, r) = (\mathrm{tail}^n(a), 2^n r - \mathrm{appr}_n(a))$, where $f(a, r) = (a', 2r - a_0)$ is the step-function in the definition of $\sim$.

We prove (b) using Lemma 10 with $S(u, a, r) := a \sim r \wedge |u + r| \leq 1$, $h(u, a, r) := (\mathrm{add}(u, a), u + r)$ and $g(u, a, r) := (2u + a_0 - \delta, a', 2r - a_0)$ where $\delta = \mathrm{sg}_{1/4}(u + \frac{1}{2}a_0 + \frac{1}{4}a_1)$. Assume $S(u, a, r)$.

1. We need to show $S(g(u, a, r))$, i.e. $a' \sim 2r - a_0$ (this holds, by definition of $\sim$) and $|2u + a_0 - \delta + 2r - a_0| \leq 1$. The latter is equivalent to $|2(u + r) - \delta| \leq 1$, which holds by Lemma 11(a) and Lemma 15(a).
2. Since $h(u, a, r) = (\delta : \mathrm{add}(2u + a_0 - \delta, a'), u + r)$, we need to show that $h(g(u, a, r)) = (\mathrm{add}(2u + a_0 - \delta, a'), 2(u + r) - \delta)$, which, however, holds by definition of h and g.

Part (c) follows from (a) and (b): Since $\mathrm{tail}^n(a) \sim (2^n r - \mathrm{appr}_n(a))$ we have

$$\mathrm{add}(\mathrm{appr}_n(a), \mathrm{tail}^n(a)) \sim \mathrm{appr}_n(a) + (2^n r - \mathrm{appr}_n(a)) = 2^n r,$$

since the latter is assumed to have an absolute value less and equal to 1.

Part (d) is proved by a trivial application of Lemma 10. □

5.1 Average

The average of signed digit streams is defined via an auxiliary function $\mathrm{av}_{\mathrm{aux}}$ representing the mapping $(r, p, i) \mapsto \frac{r+p+i}{4}$ $(r, p \in [-1, 1], -2 \leq i \leq 2)$:

Definition 16 (Corecursive definition of $\mathrm{av}_{\mathrm{aux}}$)

$$\begin{aligned}
&\mathrm{av}_{\mathrm{aux}} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathbb{Z} \to \mathrm{SD}^\omega \\
&\mathrm{av}_{\mathrm{aux}}(a, b, i) = \delta : \mathrm{av}_{\mathrm{aux}}(a', b', j - 4\delta) \\
&\quad \text{where } j = 2i + a_0 + b_0, \quad \delta = \mathrm{sg}_2(j)
\end{aligned}$$

Lemma 17 (Correctness of $\mathrm{av}_{\mathrm{aux}}$) *If $a \sim r$, $b \sim p$, and $|i| \leq 2$, then* $\mathrm{av}_{\mathrm{aux}}(a, b, i) \sim \frac{r+p+i}{4}$.

Proof We use Lemma 10 with

$$\begin{aligned}
S((a, r), (b, p), i) &:= a \sim r \wedge b \sim p \wedge |i| \leq 2, \\
h((a, r), (b, p), i) &:= \left(\mathrm{av}_{\mathrm{aux}}(a, b, i), \frac{r + p + i}{4}\right),
\end{aligned}$$

$$g((a,r),(b,p),i) := ((a',2r-a_0),(b',2p-b_0),j-4\delta)$$
$$\text{where } j = 2i + a_0 + b_0, \delta = \mathrm{sg}_2(j).$$

Assume $S((a,r),(b,p),i)$.

1. $S(g((a,r),(b,p),i))$: $|j-4\delta| \le 2$ holds by Lemma 11(b). The rest is clear.
2. $\frac{r+p+i}{4} \le 1$ holds, since $|i| \le 2$. Furthermore,

$$\begin{aligned} h(g((a,r),(b,p),i)) &= h((a',2r-a_0),(b',2p-b_0),j-4\delta) \\ &= \left(\mathrm{av}_{\mathrm{aux}}(a',b',j-4\delta), \frac{r+p+i}{2} - \delta\right), \end{aligned}$$

which is as required, since $\mathrm{av}_{\mathrm{aux}}(a,b,i) = \delta : \mathrm{av}_{\mathrm{aux}}(a',b',j-4\delta)$. □

Using $\mathrm{av}_{\mathrm{aux}}$, the average function can be easily defined:

Definition 18 (Average function av)

$$\mathrm{av} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$
$$\mathrm{av}(a,b) = \mathrm{av}_{\mathrm{aux}}(a',b',a_0+b_0)$$

Theorem 2 (Correctness of the average function) *For every* $a,b \in \mathrm{SD}^\omega$, $r,p \in [-1,1]$, *if* $a \sim r$ *and* $b \sim p$, *then* $\mathrm{av}(a,b) \sim (r+p)/2$.

Proof Assume $a \sim r$ and $b \sim p$. Hence $a' \sim 2r - a_0$ and $b' \sim 2p - b_0$, and by Lemma 17

$$\mathrm{av}(a,b) = \mathrm{av}_{\mathrm{aux}}(a',b',a_0+b_0) \sim \frac{2r-a_0+2p-b_0+a_0+b_0}{4} = \frac{r+p}{2}. \quad \square$$

5.2 Multiplication

The multiplication of signed digit streams is defined via an auxiliary function representing the mapping $(r,p,q,i) \mapsto \frac{rp+q+i}{4}$ ($r,p,q \in [-1,1]$, $-2 \le i \le 2$):

Definition 19 (Corecursive definition of $\mathrm{mp}_{\mathrm{aux}}$)

$$\mathrm{mp}_{\mathrm{aux}} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathbb{Z} \to \mathrm{SD}^\omega$$
$$\mathrm{mp}_{\mathrm{aux}}(a,b,c,i) = \delta : \mathrm{mp}_{\mathrm{aux}}(a',b,e'',j-4\delta)$$
$$\text{where } e := \mathrm{av}_{\mathrm{aux}}(\mathrm{mdig}(a_0,b),c',i),\ j := 2e_0 + e_1 + c_0 + i,\ \delta := \mathrm{sg}_2(j)$$

Lemma 20 (Correctness of $\mathrm{mp}_{\mathrm{aux}}$) *For every* $a,b,c \in \mathrm{SD}^\omega$, $r,p,q \in [-1,1]$, *if* $a \sim r$, $b \sim p$, $c \sim q$ *and* $|i| \le 2$, *then* $\mathrm{mp}_{\mathrm{aux}}(a,b,c,i) \sim \frac{rp+q+i}{4}$.

Proof We use Lemma 10 with

$$S((a,r),(b,p),(c,q),i) := a \sim r \wedge b \sim p \wedge c \sim q \wedge |i| \leq 2,$$

$$h((a,r),(b,p),(c,q),i) := \left(\mathrm{mp}_{\mathrm{aux}}(a,b,c,i), \frac{rp+q+i}{4}\right),$$

$$g((a,r),(b,p),(c,q),i)$$

$$:= ((a', 2r-a_0),(b,p),(e'', a_0p+2q-c_0+i-2e_0-e_1), j-4\delta) \quad \text{where}$$

$$e := \mathrm{av}_{\mathrm{aux}}(\mathrm{mdig}(a_0,b),c',i),\ \ j := 2e_0+e_1+c_0+i,\ \text{and } \delta := \mathrm{sg}_2(j).$$

Assume $S((a,r),(b,p),(c,q),i)$.

1. $S(g((a,r),(b,p),(c,q),i))$: $|j-4\delta| \leq 2$ holds by Lemma 11(b). It remains to be checked that $e'' \sim a_0p+2q-c_0+i-2e_0-e_1$. By the assumptions $b \sim p$ and $c \sim q$ together with Lemmas 15(d) and 17 we obtain $e' \sim \frac{a_0p+2q-c_0+i}{2} - e_0$ and therefore $e'' \sim a_0p+2q-c_0+i-2e_0-e_1$.
2. We have $h((a,r),(b,p),(c,q),i) = (\delta : \mathrm{mp}_{\mathrm{aux}}(a', b, e'', j-4\delta), \frac{rp+q+i}{4})$ (where $\frac{rp+q+i}{4} \leq 1$) and

$$\begin{aligned} &h(g((a,r),(b,p),(c,q),i)) \\ &\quad = h((a',2r-a_0),(b,p),(e'',a_0p+2q-c_0+i-2e_0-e_1),j-4\delta) \\ &\quad = \left(\mathrm{mp}_{\mathrm{aux}}(a',b,e'',j-4\delta), \frac{rp+q+i}{2} - \delta\right), \end{aligned}$$

as required. □

Definition 21 (Multiplication function mp)

$$\mathrm{mp} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$

$$\mathrm{mp}(a,b) = \mathrm{scale}_2(\mathrm{mp}_{\mathrm{aux}}(a,b,[0],0))$$

Here, $[0]$ denotes the stream of zeros (corecursively defined by $[0] = 0 : [0]$). Clearly $[0] \sim 0$.

Theorem 3 (Correctness of the multiplication function) *For every* $a, b \in \mathrm{SD}^\omega$, $r, p \in [-1,1]$, *if* $a \sim r$ *and* $b \sim p$, *then* $\mathrm{mp}(a,b) \sim rp$.

Proof Using Lemma 20, we have $\mathrm{mp}_{\mathrm{aux}}(a,b,[0],0) \sim \frac{rp}{4}$ and $|rp| \leq 1$. It follows, by Lemma 15(c), that $\mathrm{scale}_2(\mathrm{mp}_{\mathrm{aux}}(a,b,[0],0)) \sim rp$. □

5.3 Division

In order to compute a quotient p/q on the level of signed digit streams it is quite clear that it does not suffice to require $q \neq 0$: We must assume that q is bounded away from 0 by a given constant. For simplicity, we assume $q \in [1/4, 1]$. One can

then easily derive the general case $q \in [-1, -1/2^n] \cup [1/2^n, 1]$ (where $n \in \mathbb{N}$ will be an extra parameter of the general division algorithm). Since the result of the division must be in $[-1, 1]$ we have to require additionally $|p| \leq q$.

Definition 22 (Corecursive definition of division div) In order to understand the definition, assume $a \sim p$, $b \sim q$, $q \in [1/4, 1]$ and $|p| \leq q$.

$$\mathrm{div} : \mathrm{SD}^\omega \to \mathrm{SD}^\omega \to \mathrm{SD}^\omega$$
$$\mathrm{div}(a, b) = \delta : \mathrm{div}(\tilde{a}, b)$$

where δ and $\tilde{a}$ are defined as follows:

Case (1) $\mathrm{appr}_5(a) \geq \mathrm{appr}_3(b)$ (hence $4p \geq q - 1/4$):
Set $\delta := 1$, $\tilde{a} := \mathrm{scale}_2(\mathrm{av}(a, \mathrm{mdig}(-1, 0 : b)))$ $(\sim 2p - q =: \tilde{p})$.
Case otherwise:
  Subcase (2) $\mathrm{appr}_3(a) \geq 0$ (hence $p \geq -1/8$):
  Set $\delta := 0$, $\tilde{a} := \mathrm{scale}_1(a)$ $(\sim 2p =: \tilde{p})$.
  Subcase (3) otherwise:
  Set $\delta := -1$, $\tilde{a} := \mathrm{scale}_2(\mathrm{av}(a, 0 : b))$ $(\sim 2p + q =: \tilde{p})$.

Theorem 4 (Correctness of the division function) *Let* $p, q \in [-1, 1]$ *such that* $q \in [1/4, 1]$ *and* $|p| \leq q$. *For all* $a, b \in \mathrm{SD}^\omega$, *if* $a \sim p$ *and* $b \sim q$, *then* $\mathrm{div}(a, b) \sim p/q$.

Proof Regarding q and b as fixed parameters and assuming $q \in [1/4, 1]$ and $b \sim q$, we apply Lemma 10 with

$$S(a, p) := a \sim p \wedge |p| \leq q,$$
$$h(a, p) := (\mathrm{div}(a, b), p/q),$$
$$g(a, p) := (\tilde{a}, \tilde{p}) \quad \text{where } \tilde{a} \text{ and } \tilde{p} \text{ are as in the definition of div.}$$

Assume $S(a, p)$.

1. $S(g(a, p))$: It is clear that $\tilde{a} \sim \tilde{p}$. In order to see that $|\tilde{p}| \leq q$ we consider the three cases.
   Case (1) $4p \geq q - 1/4$: $\tilde{p} = 2p - q \in [-1/4, q] \subseteq [-q, q]$.
   Case (2) $4p < q - 1/4$ and $p \geq -1/8$: $\tilde{p} = 2p \in [-1/4, q - 1/8] \subseteq [-q, q]$.
   Case (3) $4p < q - 1/4$ and $p < -1/8$: We have $-q \leq p \leq -1/8$, hence $-2q \leq 2p \leq -1/4$. Therefore, $-q \leq 2p + q = \tilde{p} \leq q$.
2. $h(a, p) = (\delta : \mathrm{div}(\tilde{a}, b), p/q)$ and $h(g(a, p)) = (\mathrm{div}(\tilde{a}, b), \tilde{p}/q)$. Hence it remains to be shown that $\tilde{p}/q = 2p/q - \delta$, i.e. $\tilde{p} = 2p - \delta q$. This clearly holds. □

5.4 Limit

Finally, we define a function $\lim : (\mathbb{N} \to \mathrm{SD}^\omega) \to \mathrm{SD}^\omega$ that represents the limit operation on fast converging sequences.

First, we define an auxiliary function that "normalises" a signed digit stream into an equivalent one beginning with a prescribed digit, whenever possible.

Definition 23

$$\text{norm} : \text{SD} \to \text{SD}^\omega \to \text{SD}^\omega$$

$$\text{norm}(\delta, a) = \text{scale}_1(\text{add}(-\delta/2, a))$$

Lemma 24 *If $a \sim r$ and $|2r - \delta| \le 1$, then $\delta : \text{norm}(\delta, a) \sim r$.*

Proof Since $a \sim r$ and $|r - \delta/2| \le 1/2$ we have $\text{add}(-\delta/2, a) \sim (r - \delta/2)$, by Lemma 15(b), and therefore $\text{norm}(\delta, a) = \text{scale}_1(\text{add}(-\delta/2, a)) \sim (2r - \delta)$, by Lemma 15(c). Since $|r| \le 1$, it follows $\delta : \text{norm}(\delta, a) \sim r$. □

Definition 25 (Corecursive definition of lim)

$$\lim : (\mathbb{N} \to \text{SD}^\omega) \to \text{SD}^\omega$$

$$\lim(\alpha) = \delta : \lim(\tilde{\alpha})$$

$$\text{where } \delta := \text{sg}_{1/4}\left(\frac{1}{8}\text{appr}_3(\alpha(0))\right), \quad \tilde{\alpha}(n) := \text{norm}(\delta, \alpha(n+1))$$

Theorem 5 (Correctness of the limit function) *Let $\alpha : \mathbb{N} \to \text{SD}^\omega$ be a sequence of signed digit streams representing a sequence of real numbers $\rho : \mathbb{N} \to [-1, 1]$ converging fast to $r \in \mathbb{R}$, that is, $|\rho(n)| \le 1$, $\alpha(n) \sim \rho(n)$ and $|\rho(n) - r| \le 2^{-(n+4)}$ for all $n \in \mathbb{N}$. Then $\lim(\alpha) \sim r$.*

Proof We apply Lemma 10 with

$$S(\alpha, \rho, r) := \forall n.\alpha(n) \sim \rho(n) \wedge |\rho(n) - r| \le 2^{-(n+4)},$$

$$h(\alpha, \rho, r) := (\lim(\alpha), r),$$

$$g(\alpha, \rho, r) := (\tilde{\alpha}, \tilde{\rho}, 2r - \delta) \quad \text{where } \delta \text{ and } \tilde{\alpha} \text{ are as in the definition of lim, and}$$

$$\tilde{\rho}(n) := 2\rho(n+1) - \delta.$$

Assume $S(\alpha, \rho, r)$.

1. $S(g(\alpha, \rho, r))$, i.e. $\text{norm}(\delta, \alpha(n+1)) \sim (2\rho(n+1) - \delta)$ and $|2\rho(n+1) - \delta - (2r - \delta)| \le 2^{-(n+3)}$ for all $n \in \mathbb{N}$: The second statement reduces to $|\rho(n+1) - r| \le 2^{-(n+5)}$, which holds by the assumption $S(\alpha, \rho, r)$.

   It remains to prove the first statement: Since, by assumption, $\alpha(0) \sim \rho(0)$ we have

$$\left|\frac{1}{8}\text{appr}_3(\alpha(0)) - \rho(0)\right| \le \frac{1}{8},$$

   by Lemma 15(a). Furthermore,

$$|\rho(0) - \rho(n+1)| \le \frac{1}{8},$$

by the fast convergence of ρ. Hence

$$\left|\frac{1}{8}\mathrm{appr}_3(\alpha(0)) - \rho(n+1)\right| \leq \frac{1}{4}.$$

Since we also have $|\rho(n+1)| \leq 1$, it follows, by Lemma 11(a), that $|2\rho(n+1) - \delta| \leq 1$. Because $\alpha(n+1) \sim \rho(n+1)$ we obtain $\delta : \mathrm{norm}(\delta, \alpha(n+1)) \sim \rho(n+1)$, by Lemma 24. It follows $\mathrm{norm}(\delta, \alpha(n+1)) \sim (2\rho(n+1) - \delta)$, as required.

2. $h(\alpha, \rho, r) = (\delta : \lim(\tilde{\alpha}), r)$ with $|r| \leq 1$, by assumption, while $h(g(\alpha, \rho, r)) = (\lim(\tilde{\alpha}), 2r - \delta)$, as required. □

Remark A simple alternative method of implementing operations on real numbers with respect to the signed digit representation is, of course, to use corresponding implementations with respect to the Cauchy representation and transfer them to the signed digit representation, using Theorem 1. However, as to be expected, the implementations obtained in this way are far less efficient.

6 Conclusion and Future Work

Using coinduction, we have proven the correctness of the conversion functions between signed digit streams and Cauchy sequences (Theorem 1) as well as the correctness of the implementations of the basic arithmetic operations with regard to the signed digit representation (Theorems 2, 3, 4, 5). We feel that our approach, based on set-theory, is very direct and simple, and more accessible (by ordinary mathematicians) than the type theoretic approach by Ciaffaglione and Gianantonio [4]. Although we appear to have used classical set-theory, all our reasoning is in fact constructive and could be easily formalised and implemented in any system that is able to express coinduction and corecursive definitions of infinite streams.

As further work we intend to use coinduction not only to *verify* algorithms, but also to *synthesise* them from proofs using a realisability interpretation of coinductive definitions as studied by Tatsuta [17]. Work in a similar direction is currently being done by Bertot [2] who uses Coq's proof search engine to construct correct real number algorithms. The expected advantages of this approach are at least twofold: Firstly, formalisations become simpler, because the approximating data (signed digit streams, or Cauchy sequences) are left implicit in the formal proofs and are made explicit only in the program extraction process. Secondly, programming and program verification are performed in one go, hence the pay-off from the formal approach is higher.

Acknowledgements We thank the anonymous referees for suggesting many improvements and drawing our attention to recent related work on coinduction.

References

1. Bertot, Y.: Coinduction in Coq. In: Lecture Notes of TYPES Summer School 2005, Sweden, vol. II (2005)

2. Bertot, Y.: Affine functions and series with co-inductive real numbers. Math. Struc. Comput. Sci. **17**(1), 37–63 (2007)
3. Buchholz, W.: A term calculus for (co-)recursive definitions on streamlike data-structures. Ann. Pure Appl. Log. **136**, 75–90 (2005)
4. Ciaffaglione, A., Gianantonio, Di P.: A certified, corecursive implementation of exact real numbers. Theor. Comput. Sci. **351**, 39–51 (2006)
5. Chirimar, J., Howe, D.J.: Implementing constructive real analysis: preliminary report. In: Lecture Notes in Comput. Sci., vol. 613, pp. 165–178 (1992)
6. Edalat, A., Heckmann, R.: Computing with real numbers—I. The LFT approach to real number computation—II. A domain framework for computational geometry. In: Barthe, G., Dybjer, P., Pinto, L., Saraiva, J. (eds.) International Summer School on Applied Semantics, Caminha, Portugal, pp. 193–267. Springer, Berlin (2002)
7. Edalat, A., Potts, P.J.: A new representation for exact real numbers. In: Brooks, S., Mislove, M. (eds.) Mathematical Foundations of Programming Semantics (MFPS XIII), Carnegie Mellon, Pittsburgh, PA, USA. Electron. Notes Theor. Comput. Sci. (1997)
8. Gibbons, J.: Streaming representation-changers. In: Lecture Notes in Comput. Sci., vol. 3125, pp. 142–168 (2004)
9. Jacobs, B., Rutten, J.: A tutorial on (co)algebras and (co)induction. EATCS Bull. **62**, 222–259 (1997)
10. Ko, K.-I.: Complexity Theory of Real Functions. Birkhäuser, Boston (1991)
11. Lenisa, M.: From set-theoretic coinduction to coalgebraic coinduction: some results, some problems. In: Jacobs, B., Rutten, J. (eds.) Coalgebraic Methods in Computer Science CMCS'99 Conference Proceedings. Electron. Notes Theor. Comput. Sci., vol. 19 (1999)
12. Niqui, M.: Formalising exact arithmetic in type theory. In: Cooper, S.B., Löwe, B., Torenvliet, L. (eds.) New Computational Paradigms: First Conference on Computability in Europe, CiE 2005, Amsterdam, The Netherlands, June 8–12, 2005, Proceedings. Lecture Notes in Comput. Sci., vol. 3526, pp. 368–377 (2005)
13. Niqui, M.: Coinductive correctness of homographic and quadratic algorithms for exact real numbers. In: Altenkirch, T., McBride, C. (eds.) Proceedings of TYPES 2006 Workshop. Lecture Notes in Comput. Sci. (2007, to appear)
14. Plume, D.: A calculator for exact real number computation, 4th year project. Departments of Computer Science and Artificial Intelligence, University of Edinburgh (1998)
15. Rutten, J.: Universal coalgebra: a theory of systems. Theor. Comput. Sci. **249**, 3–80 (2000)
16. Schwichtenberg, H.: Inverting monotone continuous functions in constructive analysis. In: Beckmann, A., Berger, U., Löwe, B. (eds.) Logical Approaches to Computational Barriers: Second Conference on Computability in Europe, CiE 2006, Swansea, UK, June 30–July 5, 2006. Lecture Notes in Comput. Sci., vol. 3988, pp. 490–504 (2006)
17. Tatsuta, M.: Realizability of monotone coinductive definitions and its application to program synthesis. In: Proceedings of the Fourth International Conference on Mathematics of Program Construction. Lecture Notes in Comput. Sci., vol. 1422, pp. 338–364 (1998)
18. Telford, A., Turner, D.: Ensuring the productivity of infinite structures. Technical Report 551, University of Kent at Canterbury (1997)
19. Turing, A.M.: Systems of logic based on ordinals. Proc. Lond. Math. Soc. **45**, 161–228 (1939)
20. Weihrauch, K.: Computable Analysis, an Introduction. Springer, New York (2000)