The Exact Real Arithmetical Algorithm in Binary Continued Fractions

Petr Kůrka
Center for Theoretical Study
Academy of Sciences and Charles University in Prague
Jilská 1, CZ-11000 Praha 1, Czechia

***Abstract*—The exact real binary arithmetical algorithm is an on-line algorithm which computes the sum, product or ratio of two real numbers to arbitrary precision. The algorithm works in general Möbius number systems which represent real numbers by infinite products of Möbius transformations. We consider a number system of binary continued fractions in which this algorithm is computed faster than in the binary signed system. Moreover, the number system of binary continued fractions circumvents the problem of nonredundancy and slow convergence of continued fractions.**

***Index Terms*—exact real arithmetic; Möbius number systems**

I. Introduction

Exact real arithmetical algorithms sketched in an unpublished manuscript of Gosper [1] have been developed in Vuillemin [12], Kornerup and Matula [4], [3] or Potts [11]. These algorithms perform a sequence of **input absorptions** and **output emissions** and update their inner state which may be a (2×2)-matrix in the case of a Möbius transformation or a $(2\times 2\times 2)$-tensor in the case of binary operations like addition, multiplication or division. The algorithms work in **Möbius number systems** which generalize both positional number systems and continued fraction systems and represent real numbers by infinite products of Möbius transformations (Kůrka [5]).

Konečný [2] and Kůrka and Vávra [9] show that rational functions of degree at least 2 cannot be computed by a finite state transducer. This means that the state tensor of the binary arithmetical algorithm cannot be bounded during the computation: otherwise we would have a finite state transducer for the quadratic (rational) function x^2. Computer simulations suggest that the bit length of the state tensor (the number of bits needed for its representation) grows linearly with time. This implies quadratic time complexity for the algorithm. The rate of growth depends on the number system in question and it is smaller in modular systems, whose transformations have unit determinant, than in positional systems. However, modular systems like continued fractions are neither contractive nor redundant (see Kůrka and Delacourt [8]). This means that the convergence in them is slow, they need long words to attain a given precision. Nonredundancy means that the binary algorithm does not work always properly: for some inputs it does not give any output.

In the present paper we consider a number system of **binary continued fractions** which circumvents both these disadvantages while it keeps the advantage of faster computation. The system uses a **compression code**, which codes the integer entries of a continued fraction by their binary code and provides a faster convergence. The problem of redundancy is solved by a **parallel binary arithmetical algorithm** which pursues two parallel branches in the cases in which the the standard binary algorithm would be stalled.

II. Sofic Subshifts

For a finite alphabet A denote by $A^* = \bigcup_{m\geq 0} A^m$ the set of finite words and by $A^+ = \bigcup_{m>0} A^m$ the set of nonempty finite words. Here A^0 consists of the empty word λ. The length of a word $u = u_0 \dots u_{m-1} \in A^m$ is denoted by $|u| = m$. Denote by $A^{\mathbb{N}}$ the Cantor space of infinite words with the metric $d(u,v) = 2^{-\min\{k\geq 0 : u_k \neq v_k\}}$. If $v = u_{[i,j)} = u_i \dots u_{j-1}$ for some $0 \leq i \leq j \leq |u|$, then v is a **subword** of u ($v \sqsubseteq u$). The **shift map** $\sigma : A^{\mathbb{N}} \to A^{\mathbb{N}}$ is defined by $\sigma(u)_i = u_{i+1}$. A **subshift** is a nonempty set $\Sigma \subseteq A^{\mathbb{N}}$ which is closed and σ-invariant, i.e., $\sigma(\Sigma) \subseteq \Sigma$. If $D \subseteq A^*$ then

$$\Sigma_D = \{x \in A^{\mathbb{N}} : \ \forall u \sqsubseteq x, u \notin D\}$$

is a subshift (provided it is nonempty) with **forbidden words** D. Any subshift can be obtained in this way. A subshift is uniquely determined by its **language**

$$\mathcal{L}(\Sigma) = \{u \in A^* : \ \exists x \in \Sigma, u \sqsubseteq x\}.$$

A subshift Σ is of **finite type** (SFT), if $\Sigma = \Sigma_D$ for some finite set $D \subset A^*$. A subshift is **sofic** if it can be generated by a deterministic graph.

A **deterministic graph** over an alphabet A is a triple $G = (B, E, \mathbf{i})$, where B is a finite set of vertices, $\mathbf{i} \in B$ is an initial vertex and $E \subseteq B \times A \times B$ is a set of labelled edges such that for every $p \in B$, $a \in A$ there exists at most one $q \in B$ with $(p,a,q) \in E$. In this case we write $p \xrightarrow{a} q$. The source and target maps $s, t : E \to B$ are the projections $s(p,a,q) = p$, $t(p,a,q) = q$ and the labelling map $\ell : E \to A$ is the projection $\ell(p,a,q) = a$. A path is a finite or infinite word $w \in E^* \cup E^{\mathbb{N}}$ such that $t(w_i) = s(w_{i+1})$. The subshift $\Sigma_G \subseteq A^{\mathbb{N}}$ consists of all labels of all paths of G. A subshift Σ is sofic, if $\Sigma = \Sigma_G$ for some deterministic graph G.

III. Matrices, Transformations and Intervals

The **extended real line** $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$ is the one-dimensional projective space, i.e., the space $\mathbb{P}(\mathbb{R}^2)$ of one-dimensional subspaces of $\mathbb{R}^2$. A point $x \in \overline{\mathbb{R}}$ is represented by **homogeneous coordinates** $x = (x_0, x_1) \in \mathbb{R}^2 \backslash \{(0,0)\}$, and $x = y$ iff $\det(x,y) = x_0 y_1 - x_1 y_0 = 0$. We regard $x \in \overline{\mathbb{R}}$ as a column vector, and write it usually as $x = \frac{x_0}{x_1}$, for example $\infty = \frac{1}{0}$. As a topological space, $\overline{\mathbb{R}}$ is homeomorphic to the circle via the **stereographic projection** $d : \overline{\mathbb{R}} \to \mathbb{T} = \{z \in \mathbb{C} : \ |z| = 1\}$ given by $d(x) = (ix+1)/(x+i)$.

A **regular Möbius transformation** $M : \overline{\mathbb{R}} \to \overline{\mathbb{R}}$ is a self-map of $\overline{\mathbb{R}}$ of the form

$$M(x) = \frac{M_{00}x + M_{01}}{M_{10}x + M_{11}} = \frac{M_{00}x_0 + M_{01}x_1}{M_{10}x_0 + M_{11}x_1},$$

where $\det(M) = M_{00}M_{11} - M_{01}M_{10} \neq 0$. A transformation is determined by a (2×2)-matrix which we write as a pair of its left and right columns $M = [\frac{M_{00}}{M_{10}}, \frac{M_{01}}{M_{11}}]$. A nonzero multiple of M determines the same transformation so we denote by $\mathbb{M}(\mathbb{R}) = \{M \in \mathbb{P}(\mathbb{R}^{2\times 2}) : \ \det(M) \neq 0\}$ the set of regular projective (2×2)-matrices. Each element of $\mathbb{M}(\mathbb{R})$ is a one-dimensional subspace of the space of matrices $\mathbb{R}^{2\times 2}$, so two matrices are equivalent if one is a nonzero multiple of the other. Besides these regular transformations we consider also nonzero **singular transformations** $\mathbb{M}^0(\mathbb{R}) = \{M \in \mathbb{P}(\mathbb{R}^{2\times 2}) : \ \det(M) = 0\}$. A singular transformation M has a unique unstable point $u(M) \in \overline{\mathbb{R}}$ with $M(u(M)) = \frac{0}{0}$ and a unique stable point $s(M) \in \overline{\mathbb{R}}$ such that $M(x) = s(M)$ for each $x \neq u(M)$. For example, for $M = [\frac{0}{0}, \frac{0}{1}]$ we have $s(M) = \frac{0}{1}$, $u(M) = \frac{1}{0}$.

An **interval** is a connected subset of $\overline{\mathbb{R}}$. **Improper intervals** are the empty set, singletons, their complements and the full interval $\overline{\mathbb{R}}$. A **proper** interval $I \subset \overline{\mathbb{R}}$ has two different endpoints $a, b \in \overline{\mathbb{R}}$. The interval I is **closed** if $a, b \in I$ and **open** if $a, b \in \overline{\mathbb{R}} \backslash I$. There are two disjoint open intervals I, J with the endpoints a, b and the union of their closures is $\overline{\mathbb{R}}$. We might distinguish these intervals by the order of a, b and write $I = (a, b)$, $J = (b, a)$ but this is not convenient, since for a transformation M with $\det(M) < 0$ we would get $M(I) = (M(b), M(a))$. A better possibility is to define the open interval with endpoints a, b as the set $\{ax_0 + bx_1 : \ x_0, x_1 > 0\}$ of convex combinations of a, b. The two intervals I, J are then defined by the matrices $P = [\frac{a_0}{a_1}, \frac{b_0}{b_1}]$, $Q = [\frac{a_0}{a_1}, \frac{-b_0}{-b_1}]$. Define the sign $\mathrm{sgn}(x) \in \{-1, 0, 1\}$ of $x \in \overline{\mathbb{R}}$ as the sign of $x_0 x_1$. The open and closed intervals of $P \in \mathbb{P}(\mathbb{R}^{2\times 2})$ are

$$\begin{aligned} P^o &= \{x \in \overline{\mathbb{R}}, \mathrm{sgn}(P^{-1}x) > 0\}, \\ P^c &= \{x \in \overline{\mathbb{R}}, \mathrm{sgn}(P^{-1}x) \geq 0\}. \end{aligned}$$

For a singular $P = [\frac{a}{c}, \frac{b}{d}]$ we have $P^{-1} = [\frac{d}{-c}, \frac{-b}{a}]$ and

$$\begin{aligned} \mathrm{sgn}(u(P)) < 0 \ &\Rightarrow \ P^o = \emptyset, P^c = \{s(P)\}, \\ \mathrm{sgn}(u(P)) = 0 \ &\Rightarrow \ P^o = \emptyset, P^c = \overline{\mathbb{R}}, \\ \mathrm{sgn}(u(P)) > 0 \ &\Rightarrow \ P^o = \overline{\mathbb{R}} \setminus \{s(P)\}, P^c = \overline{\mathbb{R}}. \end{aligned}$$

See Kůrka [7] for a proof. For $P, Q \in \mathbb{M}(\mathbb{R})$ we have

$$P^o \subseteq Q^o \text{ iff } P^c \subseteq Q^c \text{ iff } \mathrm{sgn}(Q^{-1}P) \geq 0.$$

Here $\mathrm{sgn}(Q^{-1}P) \geq 0$ iff either all entries of $Q^{-1}P$ are nonnegative or all are nonpositive. The image of a set $I \subseteq \overline{\mathbb{R}}$ by a transformation M is $M(I) = \{Mx : \ x \in I\} \cap \overline{\mathbb{R}}$. For $P, Q \in \mathbb{M}(\mathbb{R})$ and $M \in \mathbb{M}(\mathbb{R}) \cup \mathbb{M}^0(\mathbb{R})$ we get

$$\mathrm{sgn}(Q^{-1}MP) \geq 0 \ \Rightarrow \ M(P^c) \subseteq Q^c.$$

The **length** of $P = [\frac{a_0}{a_1}, \frac{b_0}{b_1}]$ is defined by

$$|P| = \frac{1}{2} - \frac{1}{\pi} \arctan \frac{a_0 b_0 + a_1 b_1}{|a_0 b_1 - a_1 b_0|}.$$

IV. Möbius Number Systems

An **iterative system** over an alphabet A is a system of Möbius transformations $F = \{F_a \in \mathbb{M}(\mathbb{R}) : \ a \in A\}$. For a finite word $u \in A^n$ we denote by $F_u = F_{u_0} \circ \cdots \circ F_{u_{n-1}}$ the composition. We define the **convergence space** $\mathbb{X}_F \subseteq A^{\mathbb{N}}$ and the **value map** $\Phi : \mathbb{X}_F \to \overline{\mathbb{R}}$ by

$$\begin{aligned} \mathbb{X}_F &= \{u \in A^{\mathbb{N}} : \ \lim_{n\to\infty} F_{u_{[0,n)}}(i) \in \overline{\mathbb{R}}\}, \\ \Phi(u) &= \lim_{n\to\infty} F_{u_{[0,n)}}(i). \end{aligned}$$

Here i is the imaginary unit. Thus $u \in A^{\mathbb{N}}$ belongs to $\mathbb{X}_F$ iff the limit $\lim_{n\to\infty} F_{u_{[0,n)}}(i)$ exists and belongs to $\overline{\mathbb{R}}$. In this case $x = \lim_{n\to\infty} F_{u_{[0,n)}}(z)$ for every complex z with nonzero imaginary part and (usually) for most real numbers z as well. We say that (F, Σ) is a **Möbius number system**, if F is an iterative system and $\Sigma \subseteq \mathbb{X}_F$ is a subshift such that $\Phi : \Sigma \to \overline{\mathbb{R}}$ is continuous and surjective.

For a large class of number systems (F, Σ) with sofic subshift Σ there exists a deterministic graph $G = (B, E, \mathbf{i})$ and a system of closed intervals $V = \{V_p : \ p \in B\}$ such that $\Sigma = \Sigma_G$, $V_{\mathbf{i}} = \overline{\mathbb{R}}$, and $V_p = \cup\{F_a(V_q) : \ p \xrightarrow{a} q\}$ for each $p \in B$ (see Kůrka [5]). For $p \neq \mathbf{i}$, V_p is a proper interval represented by a matrix which we denote also by $V_p \in \mathbb{M}(\mathbb{R})$. The system is **redundant**, if for each $p \in B$, $\{F_a(V_q^\circ) : \ p \xrightarrow{a} q\}$ is an open cover of V_p°, so that the intervals $F_a V_q$ overlap. For each word $u \in \mathcal{L}_G$ we have $\Phi[u] = \{\Phi(x) : \ x_{[0,|u|)} = u\} = F_u V_{t(u)}$, where $t(u)$ is the target of the unique path with source $\mathbf{i}$ and label u. These intervals can be computed recursively by the multiplication from the right. For an edge $p \xrightarrow{a} q$ of G we have matrices

$$H_{p,a,q} = \begin{cases} F_a V_q & \text{if} \quad p = \mathbf{i} \\ V_p^{-1} F_a V_q & \text{if} \quad p \neq \mathbf{i} \end{cases}$$

For a path $\mathbf{i} \xrightarrow{u_0} p_1 \xrightarrow{u_1} \cdots\cdots \xrightarrow{u_{n-1}} p_n$ we get

$$\Phi[u] = F_u V_{p_n} = H_{\mathbf{i},u_0,p_1} H_{p_1,u_1,p_2} \cdots H_{p_{n-1},u_{n-1},p_n}.$$

If $p \neq \mathbf{i}$, then $F_a V_q \subseteq V_p$, so $\mathrm{sgn}(H_{p,a,q}) \geq 0$. An important characteristic of a number system is the speed of its convergence measured by the lengths $|\Phi[u]|$ of their intervals. Set

$$\underline{L} = \liminf_{n\to\infty} \sqrt[n]{\underline{L}_n}, \ \overline{L} = \limsup_{n\to\infty} \sqrt[n]{\overline{L}_n},$$

where

$$\begin{aligned} \underline{L}_n &= \min\{|\Phi[u]| : \ u \in \mathcal{L}(\Sigma), |u| = n\}, \\ \overline{L}_n &= \max\{|\Phi[u]| : \ u \in \mathcal{L}(\Sigma), |u| = n\}. \end{aligned}$$

p	V_p	a	q	F_aV_q	$H_{p,a,q}$
λ	$\overline{\mathbb{R}}$	$\overline{0}$	$\overline{0}$	$[\frac{1}{2},\frac{1}{-2}]$	$[\frac{1}{2},\frac{1}{-2}]$
		$\overline{1}$	0	$[\frac{-1}{1},\frac{0}{1}]$	$[\frac{-1}{1},\frac{0}{1}]$
		0	0	$[\frac{-1}{2},\frac{1}{2}]$	$[\frac{-1}{2},\frac{1}{2}]$
		1	0	$[\frac{0}{1},\frac{1}{1}]$	$[\frac{0}{1},\frac{1}{1}]$
$\overline{0}$	$[\frac{1}{4},\frac{1}{-4}]$	1	$\overline{01}$	$[\frac{1}{4},\frac{1}{1}]$	$[\frac{4}{0},\frac{5}{3}]$
		$\overline{0}$	$\overline{0}$	$[\frac{1}{2},\frac{1}{-2}]$	$[\frac{3}{1},\frac{1}{3}]$
		$\overline{1}$	$\overline{01}$	$[\frac{-1}{1},\frac{-1}{4}]$	$[\frac{3}{5},\frac{0}{4}]$
$\overline{01}$	$[\frac{-1}{2},\frac{1}{1}]$	0	0	$[\frac{-1}{2},\frac{1}{2}]$	$[\frac{3}{0},\frac{1}{4}]$
		1	0	$[\frac{0}{1},\frac{1}{2}]$	$[\frac{1}{1},\frac{0}{3}]$
$\overline{01}$	$[\frac{-1}{1},\frac{1}{2}]$	$\overline{1}$	0	$[\frac{-1}{1},\frac{0}{1}]$	$[\frac{3}{0},\frac{1}{1}]$
		0	0	$[\frac{-1}{2},\frac{1}{2}]$	$[\frac{4}{1},\frac{0}{3}]$
0	$[\frac{-1}{1},\frac{1}{1}]$	$\overline{1}$	0	$[\frac{-1}{1},\frac{0}{1}]$	$[\frac{2}{0},\frac{1}{1}]$
		0	0	$[\frac{-1}{2},\frac{1}{2}]$	$[\frac{3}{1},\frac{1}{3}]$
		1	0	$[\frac{0}{1},\frac{1}{1}]$	$[\frac{1}{1},\frac{0}{2}]$

TABLE I
THE DETERMINISTIC GRAPH FOR THE BINARY SIGNED SYSTEM

V. THE BINARY SIGNED SYSTEM

The classical binary signed system for the interval $[-1,1]$ is based on the iterative system $F_a(x) = \frac{x+a}{2}$ with $a \in \{\overline{1},0,1\} = \{-1,0,1\}$. To get a number system for whole $\overline{\mathbb{R}}$, we add the transformation $F_{\overline{0}}(x) = 2x$ and we forbid some words. We get a system with alphabet $A = \{\overline{1},0,1,\overline{0}\}$, transformations with matrices

$$F_{\overline{1}} = [\tfrac{1}{0},\tfrac{-1}{2}],\ F_0 = [\tfrac{1}{0},\tfrac{0}{2}],\ F_1 = [\tfrac{1}{0},\tfrac{1}{2}],\ F_{\overline{0}} = [\tfrac{2}{0},\tfrac{0}{1}],$$

and forbiden words $D = \{\overline{1}0,0\overline{0},1\overline{0},\overline{0}0,\overline{0}1\overline{1},\overline{01}1\}$. The letter $\overline{0}$ can appear only at the beginning of a word, so each element of Σ_D has the form $\overline{0}^m u$, where $m \geq 0$ and $u \in \{\overline{1},0,1\}^{\mathbb{N}}$. The value map is given by

$$\Phi(\overline{0}^m u) = \sum_{i\geq 0} u_i \cdot 2^{m-i-1}$$

and $\Phi(\overline{0}^\omega) = \infty$. Note that the words $\overline{0}1\overline{1},\overline{01}1$ should be forbidden. Otherwise we would get $\Phi(\overline{0}^n 1\overline{1}^\omega) = 0$, and Φ would not be continuous at $\overline{0}^\omega$. We have a deterministic graph with $B = \{\lambda,\overline{0},\overline{0}1,\overline{01},0\}$ whose elements represent proper prefixes of the forbidden words (see Table I). The binary system is redundant, since for each p, the open intervals $F_a(V_q^o)$ with $p \xrightarrow{a} q$, cover V_p^o. For its length quotients we get $\underline{L} = \overline{L} = \frac{1}{2}$

VI. CONTINUED FRACTIONS

A **regular continued fraction** is an expression

$$a_0 + \frac{1}{a_1+}\frac{1}{a_2+}\cdots \quad = \quad a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2+ \ddots}}$$

where a_0 is an integer and a_n are positive integers for $n > 0$. The value of such a continued fraction is

$$\lim_{n\to\infty} a_0 + \frac{1}{a_1+}\cdots\frac{1}{a_n} = \lim_{n\to\infty}\frac{p_n}{q_n},$$

where the **convergents** p_n, q_n are defined by $p_{-1} = 1$, $q_{-1} = 0$, $p_0 = a_0$, $q_0 = 1$, $p_1 = 1 + a_1a_0$, $q_1 = a_1$, $p_n = p_{n-2} + a_np_{n-1}$, $q_n = q_{n-2} + a_nq_{n-1}$. The **CF system** (see Niqui [10] or Kůrka [6]) has the alphabet $A = \{0,1,2,3\}$, transformations with matrices

$$F_0 = [\tfrac{1}{1},\tfrac{0}{1}],\ F_1 = [\tfrac{1}{0},\tfrac{1}{1}],\ F_2 = [\tfrac{1}{-1},\tfrac{0}{1}],\ F_3 = [\tfrac{1}{0},\tfrac{-1}{1}].$$

and forbidden words $D = \{02,03,12,13,20,21,30,31\}$, so $\Sigma_D = \{0,1\}^{\mathbb{N}} \cup \{2,3\}^{\mathbb{N}}$. A word $u \in \{0,1\}^{\mathbb{N}}$ can be written as $u = 1^{a_0}0^{a_1}1^{a_2}\cdots$, where $a_0 \geq 0$ and $a_i > 0$ for $i > 0$. The sequence of a_i may be finite if its last element a_n is infinite. The value mapping $\Phi : \Sigma_D \to \overline{\mathbb{R}}$ of the CF system is given by

$$\begin{aligned}\Phi(1^{a_0}0^{a_1}1^{a_2}\cdots) &= a_0 + \frac{1}{a_1+}\frac{1}{a_2+}\cdots\\ \Phi(3^{a_0}2^{a_1}3^{a_2}\cdots) &= -\left(a_0 + \frac{1}{a_1+}\frac{1}{a_2+}\cdots\right)\end{aligned}$$

If $a_n = \infty$ for some $n > 0$, then we use $1/\infty = 0$ to get a finite continued fraction. We have a deterministic graph with vertices $B = \{\lambda,0,1\}$ and edges $\lambda \xrightarrow{0,1} 0 \xrightarrow{0,1} 0$, $\lambda \xrightarrow{2,3} 1 \xrightarrow{2,3} 1$. The intervals are $V_0 = [\frac{0}{1},\frac{1}{0}]$, $V_1 = [\frac{-1}{0},\frac{0}{1}]$ (see Table II). The CF system is not redundant, since its intervals F_aV_q with $p \xrightarrow{a} q$ do not overlap. Its upper length quotient is $\overline{L} = 1$ which implies slow convergence.

p	V_p	a	q	F_aV_q	$H_{p,a,q}$
λ	$\overline{\mathbb{R}}$	3	1	$[\frac{-1}{0},\frac{-1}{1}]$	$[\frac{-1}{0},\frac{-1}{1}]$
		2	1	$[\frac{-1}{1},\frac{0}{1}]$	$[\frac{-1}{1},\frac{0}{1}]$
		0	0	$[\frac{0}{1},\frac{1}{1}]$	$[\frac{0}{1},\frac{1}{1}]$
		1	0	$[\frac{1}{1},\frac{1}{0}]$	$[\frac{1}{1},\frac{1}{0}]$
1	$[\frac{-1}{0},\frac{0}{1}]$	3	1	$[\frac{-1}{0},\frac{-1}{1}]$	$[\frac{1}{0},\frac{1}{1}]$
		2	1	$[\frac{-1}{1},\frac{0}{1}]$	$[\frac{1}{1},\frac{0}{1}]$
0	$[\frac{0}{1},\frac{1}{0}]$	0	0	$[\frac{0}{1},\frac{1}{1}]$	$[\frac{1}{0},\frac{1}{1}]$
		1	0	$[\frac{1}{1},\frac{1}{0}]$	$[\frac{1}{1},\frac{0}{1}]$

TABLE II
THE DETERMINISTIC GRAPH FOR THE CF SYSTEM

VII. TENSORS

Binary arithmetical operations like addition or multiplication are obtained from bilinear tensors $T : \mathbb{R}^2\times\mathbb{R}^2 \to \mathbb{R}^2$ given by $T(x,y)_k = \sum_{i=0}^1\sum_{j=0}^1 T_{kij}x_iy_j$. The tensor T determines a function $T : \overline{\mathbb{R}}\times\overline{\mathbb{R}} \to \overline{\mathbb{R}}\cup\{\frac{0}{0}\}$ defined by

$$T(x,y) = \frac{(T_{000}x_0 + T_{010}x_1)y_0 + (T_{001}x_0 + T_{011}x_1)y_1}{(T_{100}x_0 + T_{110}x_1)y_0 + (T_{101}x_0 + T_{111}x_1)y_1}$$

A nonzero multiple of a tensor defines the same function on $\overline{\mathbb{R}}\times\overline{\mathbb{R}}$, so tensors are points of the projective space $\mathbb{P}(\mathbb{R}^{2\times2\times2})$. We write them as (2×4)-matrices $T = [\frac{T_{000}}{T_{100}},\frac{T_{010}}{T_{110}},\frac{T_{001}}{T_{101}},\frac{T_{011}}{T_{111}}]$. For a tensor T, vectors $x,y,z \in \overline{\mathbb{R}}$ and matrices P,Q,R we have matrices zT, T^*x, T_*y and tensors T^*P, T_*Q and RT defined by

$$\begin{aligned}(T^*x)_{kj} &= \textstyle\sum_i T_{kij}x_i, & (T^*P)_{kij} &= \textstyle\sum_p T_{kpj}P_{pi},\\ (T_*y)_{ki} &= \textstyle\sum_j T_{kij}y_j, & (T_*Q)_{kij} &= \textstyle\sum_q T_{kiq}Q_{qj},\\ (zT)_{ij} &= \textstyle\sum_k z_kT_{kij}, & (RT)_{kij} &= \textstyle\sum_r R_{kr}T_{rij}.\end{aligned}$$

Then $(T^*P)^*x = T^*(Px)$, $(T_*Q)_*y = T_*(Qy)$. The operations with the first and second argument commute, so we adopt notations

$$\begin{aligned} T(x,y) &= (T^*x)y = (T_*y)x, \\ T(x,Q) &= (T^*x)Q = (T_*Q)^*x, \\ T(P,y) &= (T_*y)P = (T^*P)_*y, \\ T(P,Q) &= (T_*P)^*Q = (T^*Q)_*P. \end{aligned}$$

The multiplication from the left commutes with the multiplication from the right: $R(T^*P) = (RT)^*P = RT^*P$. For a matrix $M = [\frac{M_{00}}{M_{10}}, \frac{M_{01}}{M_{11}}]$ we denote its left and right columns by M_{-0}, M_{-1}, and the upper and lower row by M_{0-}, M_{1-}. Similarly for a tensor T we denote by T_{k--}, T_{-i-}, T_{--j} the **marginal matrices** obtained from T by fixing a coordinate, and T_{-ij}, T_{k-j}, T_{ki-} **marginal vectors** obtained by fixing two coordinates. A simple algebra shows that the tensor $T(P,Q)$ consists of T-images of the columns of P and Q: $T(P,Q)_{-i-} = T(P_{-i},Q)$, $T(P,Q)_{--j} = T(P,Q_{-j})$, $T(P,Q)_{-ij} = T(P_{-i},Q_{-j})$. The image of intervals $I, J \subseteq \overline{\mathbb{R}}$ by a tensor T is

$$T(I,J) = \{z \in \overline{\mathbb{R}} : \exists x \in I, \exists y \in J, z = T(x,y)\}$$

In arithmetical algorithms we have to verify whether $T(I,J)$ is included in a given interval. We have an algebraic criterion which is formally similar to the inclusion criterion for intervals. The sign of a tensor is defined similarly as the sign of a matrix: it is nonnegative if there exists nonzero s such that all sT_{kij} are nonnegative.

Definition 1: We say that T is a **regular tensor**, if for each $x, y, z \in \overline{\mathbb{R}}$, the matrices zT, T^*x, T_*y are nonzero. Denote by $\mathbb{T}(\mathbb{R}) \subseteq \mathbb{P}(\mathbb{R}^{2\times2\times2})$ the space of regular tensors. A tensor is regular iff its pairs of marginal matrices are linearly independent, i.e., if $T_{0--} \neq T_{1--}$, $T_{-0-} \neq T_{-1-}$ and $T_{--0} \neq T_{--1}$ are different points of the projective space $\mathbb{P}(\mathbb{R}^{2\times2})$. Examples of regular tensors are $[\frac{1}{0}, \frac{0}{0}, \frac{0}{0}, \frac{0}{1}]$ (multiplication), $[\frac{0}{0}, \frac{1}{0}, \frac{1}{0}, \frac{0}{1}]$ (addition), or $[\frac{0}{0}, \frac{1}{0}, \frac{0}{1}, \frac{0}{0}]$ (division). If T is a regular tensor and M is a regular matrix, then MT, T^*M and T_*M are regular tensors.

Theorem 2 (Kůrka [7]): Let T be a regular tensor and let P, Q, R be regular matrices. Then $\operatorname{sgn}(R^{-1}T(P,Q)) \geq 0$ iff $T(P^c, Q^c) \subseteq R^c$.

Accordingly we say that a regular tensor $X \in \mathbb{T}(\mathbb{R})$ is included in a matrix $R \in \mathbb{M}(\mathbb{R})$ and write $X \subseteq R$, if $\operatorname{sgn}(R^{-1}X) \geq 0$.

VIII. The Binary Arithmetical Algorithm

Consider a sofic number system (F, Σ_G) with a deterministic graph $G = (B, E, \mathbf{i})$ and intervals

$$\{V_p \in \mathbb{M}(\mathbb{R}) : p \in B \setminus \{\mathbf{i}\}\}.$$

The **binary algorithm** for the arithmetical operations like addition, multiplication or division is based on computing a path in the **binary graph** whose vertices are $(X,p,q,r) \in \mathbb{T}(\mathbb{R}) \times B^3$ and the edges are

$$\begin{array}{lll} (X,p,q,r) & \xrightarrow{a,\lambda,\lambda} & (X^*H_{(p,a,p')}, p', q, r) \text{ if } p \xrightarrow{a} p' \\ (X,p,q,r) & \xrightarrow{\lambda,b,\lambda} & (X_*H_{(q,b,q')}, p, q', r) \text{ if } q \xrightarrow{b} q' \\ (X,p,q,r) & \xrightarrow{\lambda,\lambda,c} & (F_c^{-1}X, p, q, r') \text{ if } r \xrightarrow{c} r', p, q \neq \mathbf{i}, \\ & & \qquad \operatorname{sgn}(V_{r'}^{-1}F_c^{-1}X) \geq 0 \end{array}$$

The first rule is an **x-absorption** of a letter of the first argument, the second rule is an **y-absorption** of a letter of the second argument, and the third rule is an **emission** of a letter of the output. The label of an edge is a triple consisting of x-input, y-input and output. The label of a path is the concatenation of the labels of its edges. If $(X, \mathbf{i}, \mathbf{i}, \mathbf{i}) \xrightarrow{u,v,w} (Y, p, q, r)$ is a finite path, then $\mathbf{i} \xrightarrow{u} p$, $\mathbf{i} \xrightarrow{v} q$, $\mathbf{i} \xrightarrow{w} r$ and $Y = F_w^{-1}X(F_uV_p, F_vV_q) \subseteq V_r$. If $(T, \mathbf{i}, \mathbf{i}, \mathbf{i}) \xrightarrow{u,v,w}$ is an infinite path with infinite $u, v, w \in A^{\mathbb{N}}$, then $u, v, w \in \Sigma_G$ and $T(\Phi(u), \Phi(v)) = \Phi(w)$.

The binary graph represents a nondeterministic algorithm for arithmetic operations. To get a deterministic algorithm, we use a **selector** $s : \mathbb{T}(\mathbb{R}) \times B \to A \cup \{\text{'x'}, \text{'y'}, \text{'xy'}\}$ which chooses an admissible emission or an absorption. If $s(X,r) = c \in A$, then the algorithm performs an emission with edge $r \xrightarrow{c} r'$ (r' is determined by r and c). Otherwise the algorithm performs either an x-absorption or an y-absorption or both. In Table III we give in a Python-like syntax a **greedy selector**, which chooses an emission whenever it is possible. If not it chooses either an x-absorption or an y-absorption or both. To choose a convenient kind of absorption we consider all edges $r \xrightarrow{c} r'$ and evaluate the tensor $Y = V_{r'}^{-1}F_c^{-1}X$. If for some c, i, j, $\operatorname{sgn}(Y_{-i-}) \geq 0$, and $\operatorname{sgn}(Y_{--j}) < 0$, then $X_{-i-} \subseteq F_cV_{r'}$ but $X_{--j} \not\subseteq F_cV_{r'}$ and we select an x-absorption to get a smaller interval X_{--j}. If $\operatorname{sgn}(Y_{--j}) \geq 0$, and $\operatorname{sgn}(Y_{-i-}) < 0$, then $X_{--j} \subseteq F_cV_{r'}$ but $X_{-i-} \not\subseteq F_cV_{r'}$ and we select an y-absorption to get a smaller interval X_{-i-}. We select both absorptions if both or none of these two conditions is satisfied.

```
def s(X,r):
    for r --c--> r':
        if sgn(V_{r'}^{-1} F_c^{-1} X) ≥ 0: return c
    x, y=False,False
    for r --c--> r':
        I = F_c V_{r'}
        s_0, s_1 = sgn(I^{-1} X_{-0-}), sgn(I^{-1} X_{-1-})
        s_2, s_3 = sgn(I^{-1} X_{--0}), sgn(I^{-1} X_{--1})
        if s_0 ≥ 0 & s_1 ≥ 0 & s_2 ≥ 0 & s_3 ≥ 0: return c
        if (s_0 ≥ 0 ∨ s_1 ≥ 0) & (s_2 < 0 ∨ s_3 < 0) : x = True
        if (s_2 ≥ 0 ∨ s_3 ≥ 0) & (s_0 < 0 ∨ s_1 < 0) : y = True
    if (x & y) ∨ (¬x & ¬y): return 'xy'
    if x: return 'x'
    if y: return 'y'
```

TABLE III
A GREEDY SELECTOR FOR THE BINARY ALGORITHM

A sample run of the algorithm can be seen in Table IV which shows the state tensor together with the matrices which act upon it, input and output letters and finally the greatest common divisor of the resulting product tensor (before the next step, the entries of the tensor are divided by their common gcd). In the first step we start with the multiplication tensor $T = [\frac{1}{0}, \frac{0}{0}, \frac{0}{0}, \frac{0}{1}]$, whose marginal

X	u	v	w	gcd
$[\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]*[\frac{-1}{2},\frac{1}{2}]*[\frac{-1}{-2},\frac{-1}{2}]$	0	$\overline{0}$		1
$[\frac{1}{-4},\frac{-1}{-4},\frac{1}{4},\frac{-1}{4}]*[\frac{3}{1},\frac{1}{3}]$		$\overline{0}$		4
$[\frac{1}{-2},\frac{-1}{-2},\frac{1}{2},\frac{-1}{2}]*[\frac{2}{0},\frac{1}{1}]*[\frac{3}{1},\frac{1}{3}]$	$\overline{1}$	$\overline{0}$		8
$[\frac{1}{-1},\frac{0}{-1},\frac{1}{1},\frac{0}{1}]*[\frac{3}{1},\frac{1}{3}]*[\frac{3}{1},\frac{1}{3}]$	0	$\overline{0}$		4
$[\frac{1}{0},\frac{0}{2}].[\frac{3}{-2},\frac{1}{-2},\frac{3}{2},\frac{1}{2}]$			$\overline{0}$	1
$[\frac{3}{-4},\frac{1}{-4},\frac{3}{4},\frac{1}{4}]*[\frac{1}{1},\frac{0}{2}]*[\frac{3}{1},\frac{1}{3}]$	1	$\overline{0}$		8
$[\frac{1}{0},\frac{0}{2}].[\frac{2}{-2},\frac{1}{-2},\frac{2}{2},\frac{1}{2}]$			$\overline{0}$	1
$[\frac{2}{-4},\frac{1}{-4},\frac{2}{4},\frac{1}{4}]*[\frac{1}{1},\frac{0}{2}]*[\frac{3}{1},\frac{1}{3}]$	1	$\overline{0}$		4
$[\frac{1}{0},\frac{0}{2}].[\frac{3}{-4},\frac{2}{-4},\frac{3}{4},\frac{2}{4}]$			$\overline{0}$	1
$[\frac{3}{-8},\frac{2}{-8},\frac{3}{8},\frac{2}{8}]*[\frac{3}{1},\frac{1}{3}]$		$\overline{0}$		4
$[\frac{1}{0},\frac{0}{2}].[\frac{3}{-4},\frac{2}{-4},\frac{3}{4},\frac{2}{4}]$			$\overline{0}$	1
$[\frac{3}{-8},\frac{2}{-8},\frac{3}{8},\frac{2}{8}]*[\frac{4}{0},\frac{5}{3}]$		1		4
$[\frac{3}{-8},\frac{2}{-8},\frac{6}{-4},\frac{4}{-4}]*[\frac{3}{1},\frac{1}{3}]*[\frac{1}{1},\frac{0}{3}]$	0	1		3
$[\frac{1}{0},\frac{0}{2}].[\frac{11}{-16},\frac{9}{-16},\frac{22}{-16},\frac{18}{-16}]$			$\overline{0}$	1
$[\frac{2}{0},\frac{1}{1}].[\frac{11}{-32},\frac{9}{-32},\frac{22}{-32},\frac{18}{-32}]$			$\overline{1}$	2
$[\frac{2}{0},\frac{0}{1}].[\frac{-5}{-16},\frac{-7}{-16},\frac{6}{-16},\frac{2}{-16}]$			0	2
$[\frac{-5}{-8},\frac{-7}{-8},\frac{6}{-8},\frac{2}{-8}]*[\frac{3}{1},\frac{1}{3}]$		0		1
$[\frac{-9}{-32},\frac{-19}{-32},\frac{13}{-32},\frac{-1}{-32}]*[\frac{3}{1},\frac{1}{3}]*[\frac{2}{0},\frac{1}{1}]$	0	$\overline{1}$		4
$[\frac{2}{0},\frac{-1}{1}].[\frac{-23}{-64},\frac{-33}{-64},\frac{-2}{-64},\frac{-14}{-64}]$			1	2
$[\frac{9}{-32},\frac{-1}{-32},\frac{30}{-32},\frac{18}{-32}]*[\frac{1}{1},\frac{0}{2}]*[\frac{3}{1},\frac{1}{3}]$	1	0		2

$T = [\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]$ (multiplication)
$u = \overline{0}1011001$, $\Phi[u] = [\frac{-20}{128},\frac{-19}{128}]$,
$v = \overline{0}\overline{0}\overline{0}\overline{0}\overline{0}\overline{0}11\overline{0}\overline{1}0$, $\Phi[v] = [\frac{84}{1},\frac{92}{1}]$,
$T(\Phi[u],\Phi[v]) = [\frac{-420}{32},\frac{-399}{32},\frac{-460}{32},\frac{-437}{32}]$,
$w = \overline{0}\overline{0}\overline{0}\overline{0}\overline{0}101$, $\Phi[w] = [\frac{-16}{1},\frac{-8}{1}]$

TABLE IV
THE BINARY ALGORITHM IN THE BINARY SIGNED SYSTEM

matrices are included in no F_aV_q, so both x-absorption and y-absorption are used. In the second step we get $X = [\frac{1}{-4},\frac{-1}{-4},\frac{1}{4},\frac{-1}{4}] = [\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]*[\frac{-1}{2},\frac{1}{2}]*[\frac{-1}{-2},\frac{-1}{2}]$. Thus $X_{--0} = [\frac{1}{-4},\frac{-1}{-4}] \subseteq F_0V_0 = [\frac{-1}{2},\frac{1}{2}]$, and $X_{--1} = [\frac{1}{4},\frac{-1}{4}] \subseteq F_0V_0$ but $X_{-0-} = [\frac{1}{-4},\frac{1}{4}]$ and $X_{-1-} = [\frac{-1}{-4},\frac{-1}{4}]$ are included in no F_aV_q, so the y-absorption is chosen. The first emission occurs at step 5 with the state tensor $X = [\frac{3}{-2},\frac{1}{-2},\frac{3}{2},\frac{1}{2}]$ which is included in $F_{\overline{0}}V_{\overline{0}} = [\frac{1}{2},\frac{1}{-2}]$.

Define the **bit length** of an integer $n \in \mathbb{Z}$ by

$$\mathrm{bit}(n) = \lfloor \log_2(|n|+1) \rfloor$$

and the bit length of a matrix or tensor as the sum of the bit lengths of its components. For $M \in \mathbb{M}(\mathbb{R})$ and $T \in \mathbb{T}(\mathbb{R})$, the time complexity of the operations MT, T^*M, T_*M is of the order of $\mathrm{bit}(M)\cdot\mathrm{bit}(T)$. The time complexity of the binary algorithm is characterized by the growth of the bit length of its state tensor. Computer simulations suggest that the bit length grows linearly with the sum $|u|+|v|+|w|$ of the lengths of the input and output words: $\mathrm{bit}(X) \approx C(|u|+|v|+|w|)$. In one step of the algorithm the state tensor is multiplied by several matrices which are constant for a given number system. Thus the time complexity of the binary algorithm is proportional to $C(|u|+|v|+|w|)^2$.

For the binary signed system we get estimates $C \approx 2.66$, for the CF system we get $C \approx 0.82$. Thus the binary algorithm runs faster with the CF system. However, the CF system has two disadvantages. Since it is not redundant, the computation may not give any outputs. This happens for example if we try to add $u = 0^{a_0}1^{a_1}0^{a_2}\cdots$ and $v = 2^{a_0}3^{a_1}2^{a_2}\cdots$ with the same sequences of a_n, which represent opposite numbers. If the inputs to the algorithm are random, this happens very rarely, nevertheless there may occur long sequences of absorptions without any emission. If the inputs $u, v \in \Sigma_D$ are distributed randomly, then their length quotients $L_u = \sqrt[n]{|\Phi[u_{[0,n)}]|}$ and L_v approach the value 0.45 but for the output w, L_w usually does not approach any limit and may fluctuate close to 1. Thus the length of the output interval may converge to zero quite slowly.

n	$\mathbf{b}(n)$	$\mathbf{p}(n)$
0	0	$-$
1	1	0
2	10	100
3	11	101
4	100	11000
5	101	11001
6	110	11010
7	111	11011
8	1000	1110000
9	1001	1110001
10	1010	1110010
11	1011	1110011
12	1100	1110100
∞	$-$	1^ω

u	$\mathbf{c}(u)$	v	$\mathbf{d}(v)$
0	00	0	λ
1	01	1	λ
2	10	00	0
3	11	01	1
00	001	10	2
01	000	11	3
10	010	000	01
11	011	001	00
22	101	010	10
23	100	011	11
32	110	100	23
33	111	101	22
000	001	110	32
001	00100	111	33

TABLE V
THE BINARY AND PREFIX CODES (LEFT), AND THE COMPRESSION AND DECOMPRESSION CODES (RIGHT)

IX. THE COMPRESSION AND DECOMPRESSION CODES

We are going to modify the CF system by coding words $0^{a_0}1^{a_1}\cdots$ as sequences of binary representations of the integers a_n. For $a \in \{0,1\}$ we denote by $\overline{a} = 1-a \in \{0,1\}$ its complement. For integers $n \in \mathbb{Z}$ and $k > 0$ we denote by $|n|_k = n \bmod k$. Denote by $\mathbf{b} : \mathbb{N} \to \{0,1\}^+$ the binary code defined by $\mathbf{b}(0) = 0$ and $\mathbf{b}(n) = u \in \{0,1\}^{k+1}$, where $2^k \le n < 2^{k+1}$ and $n = 2^k u_0 + \cdots + u_k$. Define the prefix code $\mathbf{p} : \{1,2,\ldots,\infty\} \to \{0,1\}^+ \cup \{1^\omega\}$ by $\mathbf{p}(\infty) = 1^\omega$ and $\mathbf{p}(n) = 1^k 0u$, where $2^k \le n < 2^{k+1}$, $|u| = k$, and $n = 2^k + 2^{k-1}u_0 + \cdots u_{k-1}$, so $\mathbf{b}(n) = 1u$ (see Table V left).

Define the compression code $\mathbf{c} : \Sigma_D \to \{0,1\}^{\mathbb{N}}$ by

$$\begin{aligned}
\mathbf{c}(0^{a_0}1^{a_1}0^{a_2}\cdots) &= 00\mathbf{p}(a_0)\mathbf{p}(a_1)\mathbf{p}(a_2)\cdots\\
\mathbf{c}(1^{a_0}0^{a_1}1^{a_2}\cdots) &= 01\mathbf{p}(a_0)\mathbf{p}(a_1)\mathbf{p}(a_2)\cdots\\
\mathbf{c}(2^{a_0}3^{a_1}2^{a_2}\cdots) &= 10\mathbf{p}(a_0)\mathbf{p}(a_1)\mathbf{p}(a_2)\cdots\\
\mathbf{c}(3^{a_0}2^{a_1}3^{a_2}\cdots) &= 11\mathbf{p}(a_0)\mathbf{p}(a_1)\mathbf{p}(a_2)\cdots
\end{aligned}$$

Here all a_i are positive. The sequence $(a_n)_n$ may be finite if its last element is ∞. The compression code is bijective. Its inverse is the **decompression code** $\mathbf{d} : \{0,1\}^{\mathbb{N}} \to \Sigma_D$. Both codes are continuous in the Cantor topology, so they act also on finite words: For $u \in \Sigma_D$, $\mathbf{c}(u) \in \{0,1\}^*$ is the longest common prefix of all $\mathbf{c}(v)$ with $v \in [u]$ and the length of $\mathbf{c}(u)$ goes to infinity with $|u| \to \infty$. Similarly, for $u \in \{0,1\}^*$, $\mathbf{d}(u)$ is the longest common prefix of all

$\mathbf{d}(v)$ with $[u]$. Thus $\mathbf{dc}(u)$ is a prefix of u and $\mathbf{cd}(u)$ is a prefix of u (see Table V right).

Both codes can be computed by transducers, which are infinite graphs whose labels are pairs u/v of input and output words. The states of the compression transducer are $(s,a,n) \in \{0,1\}^2 \times \mathbb{N}$ where $s \in \{0,1\}$ is the sign, $a \in \{0,1\}$ is the digit and $n \in \mathbb{N}$ counts the number of digits. The initial state is $\mathbf{i} = (0,0,0)$. Its transitions are

$$\begin{array}{rcl}
(0,0,0) & \overset{a^k/\lfloor \frac{a}{2} \rfloor |a|_2 1^{\lfloor \log_2(k) \rfloor}}{\longrightarrow} & (\lfloor \tfrac{a}{2} \rfloor, |a|_2, k), \\
(\lfloor \tfrac{a}{2} \rfloor, |a|_2, n) & \overset{a^k/1^{\lfloor \log_2(n+k) \rfloor - \lfloor \log_2(n) \rfloor}}{\longrightarrow} & (\lfloor \tfrac{a}{2} \rfloor, |a|_2, n+k) \\
 & & \text{if } n > 0 \\
(\lfloor \tfrac{a}{2} \rfloor, \overline{|a|_2}, n) & \overset{a^k/0\sigma(\mathbf{b}(n))1^{\lfloor \log_2(k) \rfloor}}{\longrightarrow} & (\lfloor \tfrac{a}{2} \rfloor, |a|_2, k) \\
 & & \text{if } n > 0
\end{array}$$

The transducer accepts on input either single letters, or more generally words of the form a^k, where $a \in A$. For example we have a path

$$\begin{array}{lllll}
(0,0,0) & \overset{2^2/101}{\longrightarrow} & (1,0,2) & \overset{2/\lambda}{\longrightarrow} & (1,0,3) \quad \overset{3/01}{\longrightarrow} \\
(1,1,1) & \overset{3^5/11}{\longrightarrow} & (1,1,6) & \overset{3/\lambda}{\longrightarrow} & (1,1,7)
\end{array}$$

which yields $\mathbf{c}(2^3 3^7) = 1010111$. The inverse decompression code $\mathbf{d} = \mathbf{c}^{-1} : \{0,1\}^{\mathbb{N}} \to \Sigma_D$ is computed by a transducer with states $(s,a,b,n) \in \{0.1\}^3 \times (\{-1\} \cup \mathbb{N})$, where s is the sign, a is the digit, n is the count of the letters, $b = 0$ if the count increases and $b = 1$ if the count decreases. The initial state is $\mathbf{j} = (0,0,0,-1)$ and the transitions are

$$\begin{array}{rcl}
(0,0,0,-1) & \overset{s/\lambda}{\longrightarrow} & (s,0,0,0) \\
(s,0,0,0) & \overset{a/2s+a}{\longrightarrow} & (s,a,0,1) \\
(s,a,0,n) & \overset{1/(2s+a)^n}{\longrightarrow} & (s,a,0,2n) \text{ if } n > 0 \\
(s,a,0,n) & \overset{0/\lambda}{\longrightarrow} & (s,a,1,\tfrac{n}{2}) \text{ if } n > 1 \\
(s,a,0,1) & \overset{0/2s+\overline{a}}{\longrightarrow} & (s,\overline{a},0,1) \\
(s,a,1,n) & \overset{1/(2s+a)^n}{\longrightarrow} & (s,a,1,\tfrac{n}{2}) \text{ if } n > 1 \\
(s,a,1,n) & \overset{0/\lambda}{\longrightarrow} & (s,a,1,\tfrac{n}{2}) \text{ if } n > 1 \\
(s,a,1,1) & \overset{1/2s+a, 2s+\overline{a}}{\longrightarrow} & (s,\overline{a},0,1) \\
(s,a,1,1) & \overset{0/2s+\overline{a}}{\longrightarrow} & (s,\overline{a},0,1)
\end{array}$$

If we feed the decompression transducer with the word $\mathbf{c}(2^3 3^7) = 1010111$, we get a path

$$\begin{array}{lllllll}
(0,0,0,-1) & \overset{1/\lambda}{\longrightarrow} & (1,0,0,0) & \overset{0/2}{\longrightarrow} & (1,0,0,1) & \overset{1/2}{\longrightarrow} \\
(1,0,0,2) & \overset{0/\lambda}{\longrightarrow} & (1,0,1,1) & \overset{1/23}{\longrightarrow} & (1,1,0,1) & \overset{1/3}{\longrightarrow} \\
(1,1,0,2) & \overset{1/3^2}{\longrightarrow} & (1,1,0,4) & & &
\end{array}$$

giving $\mathbf{d}(1010111) = 2^3 3^4$ which is a prefix of $2^3 3^7$.

The **binary continued fraction** system (BCF) is defined by the value mapping $\Psi = \Phi \circ \mathbf{c} : \{0,1\}^{\mathbb{N}} \to \overline{\mathbb{R}}$, where $\Phi : \Sigma_D \to \overline{\mathbb{R}}$ is the value mapping of the CF system. We define the length quotients similarly as in Möbius number systems with Φ replaced by Ψ.

```
def s(X):
    a = −1
    for j in [0, 1, 2, 3]:
        I = F_j V_{j/2}
        if sgn(I^{-1} X) ≥ 0: a = j
    if a > −1:
        n, m, u = 1, 1, True
        I = F_a V_{a/2}
        while True:
            J = F_a^m I
            if sgn(J^{-1} X) ≥ 0:
                I = J
                n = n + m
            else: u=False
            if u: m = 2m
            else:
                if m > 1: m = m/2
                else: return a^n
    x, y=False,False
    for j in [0, 1, 2, 3]:
        I = F_j V_{j/2}
        s_0, s_1 = sgn(I^{-1} X_{−0−}), sgn(I^{-1} X_{−1−})
        s_2, s_3 = sgn(I^{-1} X_{−−0}), sgn(I^{-1} X_{−−1})
        if (s_0 ≥ 0 ∨ s_1 ≥ 0) & (s_2 < 0 ∨ s_3 < 0) : x = True
        if (s_2 ≥ 0 ∨ s_3 ≥ 0) & (s_0 < 0 ∨ s_1 < 0) : y = True
    if (x & y) ∨ (¬x & ¬y): return 'xy'
    if x: return 'x'
    if y: return 'y'
```

TABLE VI
THE SELECTOR FOR THE BCF BINARY ALGORITHM

X. THE BINARY BCF ALGORITHM

We now modify the general binary algorithm for the BCF system. Recall that the graph has vertices $B = \{\lambda, 0, 1\}$ and the intervals have matrices $V_0 = [\frac{0}{1}, \frac{1}{0}]$, $V_1 = [\frac{-1}{0}, \frac{0}{1}]$. For the matrices $H_a = V_{\lfloor \frac{a}{2} \rfloor}^{-1} F_a V_{\lfloor \frac{a}{2} \rfloor}$ we get

$$H_0 = H_3 = [\tfrac{1}{0}, \tfrac{1}{1}], \; H_1 = H_2 = [\tfrac{1}{1}, \tfrac{0}{1}].$$

For $u \in \{0,1\}^n$ we set $H_u = H_{u_0} \circ \cdots \circ H_{u_{n-1}}$. The states of the modified binary graph are (X,p,q,r), where $X \in \mathbb{T}(\mathbb{R})$, $p,q \in \{0,1\}^3 \times \{-1,0,1,2,\ldots\}$ are states of the decompression transducer, and $r \in \{0,1\}^2 \times \mathbb{N}$ is a state of the compression transducer. The initial state is $(T, \mathbf{j}, \mathbf{j}, \mathbf{i})$ where $\mathbf{j} = (0,0,0,-1)$ is the initial state of the decompression transducer and $\mathbf{i} = (0,0,0)$ is the initial state of the compression transducer. The transitions are

$$\begin{array}{rcl}
(X,p,q,r) & \overset{a,\lambda,\lambda}{\longrightarrow} & (X^* V_a, p', q, r) \text{ if } p_3 < 0, p \overset{a/b}{\longrightarrow} p' \\
(X,p,q,r) & \overset{\lambda,a,\lambda}{\longrightarrow} & (X_* V_a, p, q', r) \text{ if } q_3 < 0, q \overset{a/b}{\longrightarrow} q' \\
(X,p,q,r) & \overset{a,\lambda,\lambda}{\longrightarrow} & (X^* H_b, p', q, r) \text{ if } p_3 \geq 0, p \overset{a/b}{\longrightarrow} p' \\
(X,p,q,r) & \overset{\lambda,a,\lambda}{\longrightarrow} & (X_* H_b, p, q', r) \text{ if } q_3 \geq 0, q \overset{a/b}{\longrightarrow} q' \\
(X,p,q,r) & \overset{\lambda,\lambda,b}{\longrightarrow} & (F_a^{-k} X, p, q, r') \text{ if } p_3, q_3 \geq 0, \\
 & & X \subseteq F_a^k V_{\lfloor \frac{a}{2} \rfloor}, \; r \overset{a^k/b}{\longrightarrow} r'
\end{array}$$

If $(T, \mathbf{j}, \mathbf{j}, \mathbf{i}) \overset{u,v,w}{\longrightarrow}$ is an infinite path with infinite $u,v,w \in \{0,1\}^{\mathbb{N}}$, then $\Psi(w) = T(\Psi(u), \Psi(v))$. We consider a selector $s : \mathbb{T}(\mathbb{R}) \to A^+ \cup \{\text{'x'}, \text{'y'}, \text{'xy'}\}$ which depends only on the state $X \in \mathbb{T}(\mathbb{R})$. It searches a letter $a \in A$ such that $X \subseteq F_a V_{\lfloor \frac{a}{2} \rfloor}$. If it finds such a letter, it finds the maximum k such that $X \subseteq F_a^k V_{\lfloor \frac{a}{2} \rfloor}$ and outputs a^k. This can be done in $\lfloor \log_2 k \rfloor$ steps (see Table VI). If no such letter exists then it finds one of the absorptions

X	u	$\mathbf{d}_u$	v	$\mathbf{d}_v$	w	$\mathbf{d}_w$
$[\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]$	1		0			
$[\frac{0}{0},\frac{0}{1},\frac{-1}{0},\frac{0}{0}]$	0	2	0	0		
$[\frac{0}{1},\frac{0}{1},\frac{-1}{1},\frac{0}{1}]$					10	2
$[\frac{0}{1},\frac{0}{1},\frac{-1}{0},\frac{0}{1}]$	1	2	0	1		
$[\frac{-1}{3},\frac{0}{2},\frac{-1}{1},\frac{0}{1}]$					1	2
$[\frac{-1}{2},\frac{0}{2},\frac{-1}{0},\frac{0}{1}]$	0		0	0		
$[\frac{-1}{2},\frac{0}{2},\frac{-2}{2},\frac{0}{3}]$						2
$[\frac{-1}{1},\frac{0}{2},\frac{-2}{0},\frac{0}{3}]$	1	23	1	0		
$[\frac{-1}{3},\frac{-1}{5},\frac{-3}{6},\frac{-3}{11}]$					1	2^2
$[\frac{-1}{1},\frac{-1}{3},\frac{-3}{0},\frac{-3}{5}]$	1	3	1	0^2		
$[\frac{-1}{1},\frac{-2}{4},\frac{-5}{2},\frac{-10}{13}]$	1	3^2	1	0^4		
$[\frac{-1}{1},\frac{-4}{6},\frac{-9}{6},\frac{-36}{41}]$	0		1	0^8		
$[\frac{-1}{1},\frac{-4}{6},\frac{-17}{14},\frac{-68}{89}]$	1	3^2	0			
$[\frac{-1}{1},\frac{-6}{8},\frac{-17}{14},\frac{-102}{117}]$	0	2	1	0^8		
$[\frac{-7}{9},\frac{-6}{8},\frac{-175}{203},\frac{-150}{181}]$						2
$[\frac{-7}{2},\frac{-6}{2},\frac{-175}{28},\frac{-150}{31}]$					0101	3^3
$[\frac{-1}{2},\frac{0}{2},\frac{-91}{28},\frac{-57}{31}]$			1	0^4		
$[\frac{-1}{2},\frac{0}{2},\frac{-95}{36},\frac{-57}{39}]$			1	0^2		
$[\frac{-1}{2},\frac{0}{2},\frac{-97}{40},\frac{-57}{43}]$			1	01		
$[\frac{-99}{44},\frac{-57}{47},\frac{-98}{42},\frac{-57}{45}]$					1	3
$[\frac{-55}{44},\frac{-10}{47},\frac{-56}{42},\frac{-12}{45}]$	0	3				
$[\frac{-55}{44},\frac{-65}{91},\frac{-56}{42},\frac{-68}{87}]$	1	3				
$[\frac{-55}{44},\frac{-120}{135},\frac{-56}{42},\frac{-124}{129}]$	1	3^2				
$[\frac{-55}{44},\frac{-230}{223},\frac{-56}{42},\frac{-236}{213}]$						3
$[\frac{-11}{44},\frac{-7}{223},\frac{-14}{42},\frac{-23}{213}]$					0011	2^3

$T = [\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]$ (multiplication)
$u = 1010111010011$, $\mathbf{d}(u) = 2^3 3^6 23^4$, $\Psi[u] = [\frac{-7}{22},\frac{-34}{107}]$
$v = 0000111101111$, $\mathbf{d}(v) = 010^{31}1$, $\Psi[v] = [\frac{33}{65},\frac{32}{63}]$,
$T(\Psi[u],\Psi[v]) = [\frac{-231}{1430},\frac{-1122}{6955},\frac{-224}{1386},\frac{-1088}{6741}]$
$w = 1011010110011$, $\mathbf{d}(w) = 2^6 3^5 2^2$, $\Psi[w] = [\frac{-11}{68},\frac{-5}{31}]$

TABLE VII
THE BINARY ALGORITHM IN THE BCF SYSTEM

similarly as the selector of the general binary algorithm. A sample run of the algorithm is in Table VII.

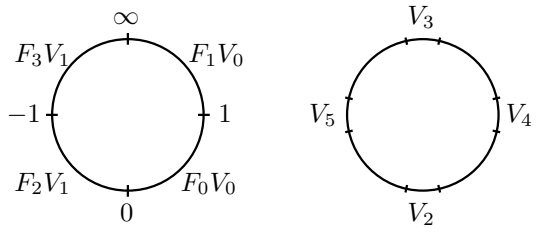

Fig. 1. The intervals of the BCF system

XI. THE PARALLEL BCF BINARY ALGORITHM

The BCF system solves the problem of slow convergence, but there still remains the problem of redundancy. Since the BCF system is not redundant, the binary algorithm need not give any output. We therefore consider a modified algorithm which in such a case pursues simultaneously two alternative output branches. For this reason we consider intervals which contain the endpoints $\frac{0}{1}$, $\frac{1}{0}$, $\frac{1}{1}$, $\frac{-1}{1}$ of the intervals $F_a V_{\lfloor \frac{a}{2} \rfloor}$ (see Fig. 1). These are

X	u	$\mathbf{d}_u$	v	$\mathbf{d}_v$	w	$\mathbf{d}_w$
$[\frac{1}{0},\frac{0}{0},\frac{0}{0},\frac{0}{1}]$	1		0			
$[\frac{0}{0},\frac{0}{1},\frac{-1}{0},\frac{0}{0}]$	0	2	0	0		
$[\frac{0}{1},\frac{0}{1},\frac{-1}{1},\frac{0}{1}]$					10	2
$[\frac{0}{1},\frac{0}{1},\frac{-1}{0},\frac{0}{1}]$	1	2	0	1		
$[\frac{-1}{3},\frac{0}{2},\frac{-1}{1},\frac{0}{1}]$					1	2
$[\frac{-1}{2},\frac{0}{2},\frac{-1}{0},\frac{0}{1}]$	0		0	0		
$[\frac{-1}{2},\frac{0}{2},\frac{-2}{2},\frac{0}{3}]$						2
$[\frac{-1}{1},\frac{0}{2},\frac{-2}{0},\frac{0}{3}]$	1	23	1	0		
$[\frac{-1}{3},\frac{-1}{5},\frac{-3}{6},\frac{-3}{11}]$					1	2^2
$[\frac{-1}{1},\frac{-1}{3},\frac{-3}{0},\frac{-3}{5}]$	1	3	1	0^2		
$[\frac{-1}{1},\frac{-2}{4},\frac{-5}{2},\frac{-10}{13}]$	1	3^2	1	0^4		
$[\frac{-1}{1},\frac{-4}{6},\frac{-9}{6},\frac{-36}{41}]$						2
$[\frac{-1}{1},\frac{-4}{6},\frac{-9}{6},\frac{-36}{41}]$					001	3
$[\frac{-1}{0},\frac{-4}{2},\frac{-9}{-3},\frac{-36}{5}]$	0		1	0^8		
$[\frac{0}{1},\frac{2}{6},\frac{-3}{6},\frac{5}{41}]$	0		1	0^8		
$[\frac{-1}{0},\frac{-4}{2},\frac{-17}{-3},\frac{-68}{21}]$	1	3^2	0			
$[\frac{0}{1},\frac{2}{6},\frac{-3}{14},\frac{21}{89}]$					0	2
$[\frac{-1}{0},\frac{-6}{2},\frac{-17}{-3},\frac{-102}{15}]$					010	3
$[\frac{0}{1},\frac{2}{8},\frac{-3}{11},\frac{21}{110}]$					1	2
$[\frac{-1}{0},\frac{-4}{2},\frac{-20}{-3},\frac{-87}{15}]$	0	2	1	0^8		
$[\frac{0}{1},\frac{2}{10},\frac{-3}{8},\frac{21}{131}]$	1	3^2	0			
$[\frac{-5}{2},\frac{-4}{2},\frac{-147}{28},\frac{-119}{31}]$					1	3^2
$[\frac{0}{1},\frac{2}{12},\frac{-3}{8},\frac{15}{147}]$ $\perp$						
$[\frac{-1}{2},\frac{0}{2},\frac{-91}{28},\frac{-57}{31}]$			1	0^4		
$[\frac{-1}{2},\frac{0}{2},\frac{-95}{36},\frac{-57}{39}]$			1	0^2		
$[\frac{-1}{2},\frac{0}{2},\frac{-97}{40},\frac{-57}{43}]$			1	01		
$[\frac{-99}{44},\frac{-57}{47},\frac{-98}{42},\frac{-57}{45}]$					1	3
$[\frac{-55}{44},\frac{-10}{47},\frac{-56}{42},\frac{-12}{45}]$	0	3				
$[\frac{-55}{44},\frac{-65}{91},\frac{-56}{42},\frac{-68}{87}]$					000	2
$[\frac{-55}{44},\frac{-65}{91},\frac{-56}{42},\frac{-68}{87}]$						3
$[\frac{-55}{-11},\frac{-65}{26},\frac{-56}{-14},\frac{-68}{19}]$	1	3				
$[\frac{-11}{44},\frac{26}{91},\frac{-14}{42},\frac{19}{87}]$					001	2
$[\frac{-55}{-11},\frac{-120}{15},\frac{-56}{-14},\frac{-124}{5}]$					01	3^2
$[\frac{-11}{33},\frac{26}{117},\frac{-14}{28},\frac{19}{106}]$	1	3				
$[\frac{-77}{-11},\frac{-90}{15},\frac{-84}{-14},\frac{-114}{5}]$					1	3^4
$[\frac{-11}{33},\frac{15}{150},\frac{-14}{28},\frac{5}{134}]$	1	3^2				

$u = 1010111010011$, $\mathbf{d}(u) = 2^3 3^6 23^4$, $\Psi[u] = [\frac{-7}{22},\frac{-34}{107}]$,
$v = 0000111101111$, $\mathbf{d}(v) = 010^{31}1$, $\Psi[v] = [\frac{33}{65},\frac{32}{63}]$,
$T(\Psi[u],\Psi[v]) = [\frac{-231}{1430},\frac{-1122}{6955},\frac{-224}{1386},\frac{-1088}{6741}]$,
$w = 101101011000011$, $\mathbf{d}(w) = 2^6 3^4 23^4$, $\Psi[w] = [\frac{-5}{31},\frac{-24}{149}]$,
$w = 101101011001$, $\mathbf{d}(w) = 2^6 3^5 2$, $\Psi[w] = [\frac{-6}{37},\frac{-5}{31}]$

TABLE VIII
THE PARALLEL BINARY ALGORITHM IN THE BCF SYSTEM

intervals $V_2 = [\frac{-1}{c},\frac{1}{c}]$, $V_3 = [\frac{c}{1},\frac{c}{-1}]$, $V_4 = [\frac{c}{c+1},\frac{c+1}{c}]$, $V_5 = [\frac{-c-1}{c},\frac{-c}{c+1}]$, where $c \geq 2$ is an integer parameter, which regulates the degree of parallelism. If c is large then the intervals V_i are small and branchings occurr only exceptionally.

If $r = (0,0,0)$ is the initial output state and X is a tensor which contains 0, then X is included neither in $F_0V_0 = [\frac{0}{1},\frac{1}{1}]$ nor in $F_2V_1 = [\frac{-1}{1},\frac{0}{1}]$. However, if X is sufficiently small, it is included in V_2 and there exists maximum $k > 0$ such that $X \subseteq F_0^k V_2$ and $X \subseteq F_2^k V_2$. In such a case the selector selects both output possibilities $w = 0^k$ and $w' = 2^k$ (which are fed into the compression transducer) and the

```
a = −1
for j in [4, 5]:
    G_0, G_1 = F_{|j|_2} V_{j−2}, F_{2+|j|_2} V_{j−2}
    if sgn(G_0^{−1} X) ≥ 0 and sgn(G_1^{−1} X) ≥ 0: a = j
if a ≥ 4:
    n, m, u = 1, 1,True
    G_0, G_1 = F_{|a|_2} V_{a−2}, F_{2+|a|_2} V_{a−2}
    while True:
        G_2, G_3 = F^m_{|a|_2} G_0, F^m_{2+|a|_2} G_1
        if sgn(G_2^{−1} X) ≥ 0 and sgn(G_3^{−1} X) ≥ 0:
            G_0, G_1 = G_2, G_3
            n = n + m
        else: u=False
        if u: m = 2 ∗ m
        else:
            if m > 1: m = m/2
            else: return a^n
for j in [6, 7]:
    Y = V_{j−2}^{−1} X
    if sgn(Y) ≥ 0: return j
```

TABLE IX
THE SELECTION OF PARALLEL PATHS IN THE PARALLEL BCF BINARY ALGORITHM

algorithm continues with two branches with state tensors $F_0^{-k}X$ and $F_2^{-k}X$. If at some later step we get $X \subseteq F_0V_0$ in the first branch, then we know that this branch is the correct one and we close the other branch. On the other hand, if we get $X \subseteq F_2V_1$ in the first branch, then we know that the first branch is incorrect, so we close it and proceed with the computation of the second branch. Similarly if $X \subseteq F_0V_0$ occurs at some later step in the second branch, we close it and if $X \subseteq F_2V_1$ in the second branch, then we close the first branch. We proceed similarly if X is a small tensor which contains ∞. In this case the selector finds the largest k such that $X \subseteq F_1^kV_3$ and $X \subseteq F_3^kV_3$.

If r is not the initial state $(0, 0, 0)$ then either $X \subseteq V_0$ or $X \subseteq V_1$. In the former case it may happen that X contains 1, so neither $X \subseteq F_0V_0$ nor $X \subseteq F_1V_0$, but $X \subseteq V_4$. Then $F_0^{-1}X$ contains ∞ while $F_1^{-1}X$ contains 0, so we pursue both possibilities $c = 0$, $c = 1$. In succeeding steps we test whether $X \subseteq F_1^kV_3$ in the first branch and output 1^k. In the second branch we test whether $X \subseteq F_0^kV_2$ and output 0^k. If at some later step we get in some branch $X \subseteq V_0$, then we know that this branch is the correct one and we close the second branch. On the other hand, if we get in some branch $X \subseteq V_1$, then we know that this branch is incorrect, so we close it and continue with the other branch. Similarly, if $X \subseteq V_1$ we test whether $X \subseteq V_5$ and proceed analogously. Thus the selector for the parallel algorithm is obtained from the selector of Table VI by inserting the procedure of Table IX just before the selection of the absorption steps.

A sample run of the parallel algorithm is in Table VIII whose inputs are the same as those of Table VII. The parallel branches are written between succeeding horizontal lines. The first branch occurs at step 12 with tensor $X = [\frac{-1}{1}, \frac{-4}{6}, \frac{-9}{6}, \frac{-36}{41}]$ which is included in $V_5 = [\frac{-3}{2}, \frac{-2}{3}]$ (with the parameter $c = 2$). The two branches are pursued in parallel till the step 17 when the state tensor $X = [\frac{-5}{2}, \frac{-4}{2}, \frac{-147}{28}, \frac{-119}{31}]$ is included in $V_1 = [\frac{-1}{0}, \frac{0}{1}]$. This means that the first branch is the correct one and the other branch with tensor $X = [\frac{0}{1}, \frac{2}{12}, \frac{-3}{8}, \frac{15}{147}]$ is closed. This is indicated by the character $\perp$. Next branching occurs at step 24 with the state tensor $X = [\frac{-55}{44}, \frac{-65}{91}, \frac{-56}{42}, \frac{-68}{87}]$ which is included in V_5. At the final step 27 we get two outputs, whose intervals $\Psi[w] = [\frac{-5}{31}, \frac{-24}{149}]$, $\Psi[w] = [\frac{-6}{37}, \frac{-5}{31}]$ are contiguous.

XII. DISCUSSION

In the computation of the parallel algorithm, the branches may be repeatedly opened and closed again, but at any time there are at most two branches. These two branches can be computed by parallel processors which interact only if one of the branches has to be closed. The two branches may proceed in parallel indefinitely giving ever smaller output intervals. If the binary algorithm forms a part of a larger computational scheme, then both branches should be fed into the next procedure.

We have tested the BCF parallel binary algorithm for the operations of addition, multiplication and division on hundreds of examples. In runs of 10000 steps, all statistical characteristics stabilize during the first 300 steps and then fluctuate only minimally. The bit length of the state tensor stabilizes at $\mathrm{bit}(X)/(|u| + |v| + |w|) \doteq 1.34$ which is twice better than the bit length growth of the binary signed system. With random, independent, uniformly distributed inputs u, v we obtain limit input length quotients $\sqrt[n]{L_{u_{[0,n)}}} \doteq \sqrt[n]{L_{v_{[0,n)}}} \doteq 0.49$ while the output length is $\sqrt[n]{L_{w_{[0,n)}}} \doteq 0.51$. It seems therefore that for long input words the BCF number system is better suited to perform the arbitrary precision computation than the binary signed system.

REFERENCES

[1] R. W. Gosper. Continued fractions arithmetic. *Unpublished manuscript*, 1977. http://www.tweedledum.com/rwg/cfup.htm.

[2] M. Konečný. Real functions incrementally computable by finite automata. *Theoretical Computer Science*, 315(1):109–133, 2004.

[3] P. Kornerup and D. W. Matula. *Finite precision number systems and arithmetic*. Cambridge University Press, Cambridge, 2010.

[4] P. Kornerup and D. W. Matula. An algorithm for redundant binary bit-pipelined rational arithmetic. *IEEE Transactions on Computers*, 39(8):1106–1115, August 1990.

[5] P. Kůrka. Möbius number systems with sofic subshifts. *Nonlinearity*, 22:437–456, 2009.

[6] P. Kůrka. Stern-Brocot graph in Möbius number systems. *Nonlinearity*, 25:57–72, 2012.

[7] P. Kůrka. Exact real arithmetic for interval number systems. *Theoretical Computer Science*, 542:32–43, 2014.

[8] P. Kůrka and M. Delacourt. The unary arithmetical algorithm in bimodular number systems. In *2013 IEEE 21st Symposium on Computer Arithmetic ARITH-21*, pages 127–134. IEEE Computer Society, 2013.

[9] P. Kůrka and T. Vávra. Analytical functions computable by finite state transducers. In M. Holzer and M. Kutrib, editors, *Implementation and Application of Automata*, volume 8587 of *LNCS*, pages 252–263. Springer-Verlag, 2014.

[10] M. Niqui. Exact real arithmetic on the Stern-Brocot tree. *J. Discrete Algorithms*, 5(2):356–379, 2007.

[11] P. J. Potts. *Exact real arithmetic using Möbius transformations*. PhD thesis, University of London, Imperial College, London, 1998.

[12] J. E. Vuillemin. Exact real computer arithmetic with continued fractions. *IEEE Transactions on Computers*, 39(8):1087–1105, August 1990.