



ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta jaderná a fyzikálně inženýrská



Spektra reálných a komplexních čísel a jejich aplikace

Spectra of real and complex numbers and their applications

Výzkumný úkol

Author: **Bc. Pavla Veselá**

Supervisor: **prof. Ing. Zuzana Masáková, Ph.D.**

Academic year: **2019/2020**

VÝZKUMNÝ ÚKOL

Posluchač: **Pavla Veselá** Akademický rok: 2019/2020
Vedoucí práce: prof. Ing. Zuzana Masáková, Ph.D. Katedra: KM
Konzultant: Obor: MINF
Název práce: Spektra reálných a komplexních čísel a jejich aplikace


Pokyny pro vypracování:

1. Nastudujte definici a základní vlastnosti spektra čísla $\beta \in \mathbb{C}$ s abecedou $\mathcal{A} \subset \mathbb{C}$.
2. Implementujte algoritmus generování spektra komplexního Pisotova čísla β s abecedou \mathcal{A} v algebraickém tělese $\mathbb{Q}(\beta)$.
3. Zkoumejte vztah vlastností spektra a číselných systémů, zejména s ohledem na reprezentovatelnost komplexních čísel a možnost on-line algoritmů pro aritmetické operace.
4. K dané komplexní Pisotově bázi β a abecedě cifer $\mathcal{A} \subset \mathbb{Q}(\beta)$ implementujte algoritmus, který rozhodne o existenci (β, \mathcal{A}) -reprezentací všech komplexních čísel. Studujte možnost rozšíření na nepisotovské báze.
5. Popište algoritmus pro předzpracování dělitelů v on-line algoritmu pro dělení v dané číselné soustavě.

Literatura:

1. Ch. Frougny and E. Pelantová. Two applications of the spectrum of numbers, Acta Mathematica Hungarica, 156(2) (2018), 391–407.
2. Ch. Frougny, M. Pavelka, E. Pelantová, and M. Svobodová. On-line algorithms for multiplication and division in real and complex numeration systems, Discrete Mathematics and Theoretical Computer Science, 21(3) (2019).
3. T. Vávra. Periodic representations in Salem bases. Accepted to Israel Journal of Mathematics, (2019).

Datum zadání: **31.10.2019**
Datum odevzdání: **1.6.2020**


vedoucí katedry

pracoviště školitele: Katedra matematiky FJFI, ČVUT v Praze, Trojanova 13, 120 00 Praha 2
email / telefon: zuzana.masakova@fjfi.cvut.cz / +420 608 606 825

Acknowledgement:

I would like to express my gratitude to my supervisor prof. Ing. Zuzana Masáková, Ph.D for her willingness, patience, and useful suggestions without which this work would never have been completed.

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS20/183/OHK4/3T/14

Čestné prohlášení:

Prohlašuji, že jsem tuto práci vypracovala samostatně a uvedla jsem všechnu použitou literaturu.

V Praze dne 16. března 2021

Pavla Veselá

Contents

Introduction	5
1 Preliminaries	6
1.1 Combinatorics on words	6
1.2 Algebraic numbers	6
1.3 Number representations	7
2 Spectrum of real and complex numbers	10
2.1 Algorithm for generating of the spectrum	10
2.2 Discreteness of the spectrum	12
2.3 Relative density	13
3 Parallel addition	15
3.1 Local functions and parallel addition	15
3.2 Neighbour-free parallel addition	16
4 On-line division	20
4.1 Algorithm for on-line division	20
4.2 OL Property	21
4.3 Preprocessing	22
4.4 Time complexity	26
5 Description of the program	27
5.1 Generating of spectrum	27
5.2 Finding of WRZ and SRZ polynomials	28
5.3 Finding of H in algorithm for preprocessing	28
5.4 Preprocessing	28
6 Fast on-line division for selected numeration systems	31
6.1 Quadratic Pisot unit	31
6.2 Golden ratio	33
6.3 Silver ratio	34
6.4 d -Bonacci numbers	35
6.5 Penney number	38
6.6 Eisenstein number	40
Conclusions	45
Bibliography	47

Introduction

Basic arithmetic operations are the essence of any calculations. In classical numeration systems with integer base, algorithms for computing addition, multiplication and division are well known. The situation differs for non-standard numeration systems with non-integer base or alphabet. In order to make calculations of arithmetic operations effective, algorithms for parallel addition and on-line multiplication and division were introduced by Trivedi and Ercegovic [16]. They were later modified for non-standard numeration systems in [5].

In redundant numeration systems an algorithm for parallel addition where the digit of the result depends only on fixed number of neighbouring digits can be designed. Such an algorithm allows us to perform addition in constant time. On-line arithmetic is a mode of computation where operands and results are processed in a digit serial manner, starting with the most significant digit.

The purpose of this work is to study properties of numeration systems and the corresponding spectra and the impact of these properties on feasibility of on-line arithmetic operations. Second aim is to write computer programs in order to apply these theoretical concepts to several cases of numeration systems.

The work is organized as follows. Chapter 1 is dedicated to basic definitions and concepts of combinatorics on words, algebraic numbers and number representations. In Chapter 2 the spectrum of numeration system is defined and algorithm for generating of this set is described. We discuss the relation of properties of the spectrum to the particular properties of the numeration system.

Chapters 3 and 4 contain basic definitions and concept of parallel addition and on-line division. Various properties and necessary conditions on feasibility of these operations are presented.

In Chapter 5, documentation for the computer programs implementing algorithms for generating of the spectrum, finding polynomial satisfying the strong and weak representation of zero property, finding of accurate value of parameter H in algorithm for preprocessing and algorithm for preprocessing for on-line division can be found.

Chapter 6 contains application of the presented theory to several numeration system. In particular systems with base being a quadratic Pisot number, d -Bonacci, Penney and Eisenstein number are treated. For each given base several alphabets are discussed in order to perform given arithmetic operations. The results of computer programs are presented.

Chapter 1

Preliminaries

1.1 Combinatorics on words

In this section we will introduce basic definitions and concepts of combinatorics on words.

A non-empty finite set A is called *alphabet*. Elements of the alphabet are called *letters*. A *word over the alphabet A* is defined as a finite sequence of letters from A . We denote the empty word ε .

Let $u = u_0u_1 \cdots u_m$ and $v = v_0v_1 \cdots v_n$ be words, where $u_i, v_j \in A$ for $i = 0, \dots, m$ and $j = 0, \dots, n$. Then *concatenation of words* is defined as $uv = u_0 \cdots u_mv_0 \cdots v_n$.

The set of words over the alphabet A with operation concatenation of words is a monoid of words A^* . Concatenation of words is obviously associative and the empty word ε is the unit of the monoid A^* .

We further define:

- The *length of a word* $u = u_0 \cdots u_{n-1} \in A^*$ is defined as the number of its letters, we write $|u| = n$. The number of occurrences of a letter $a \in A$ in the word $u \in A^*$ is denoted as $|u|_a$.
- Let $u, v, v^{(1)}, v^{(2)} \in A^*$ be words such that $u = v^{(1)}v^{(2)}$. Then the word $v^{(1)}$ is a *prefix*, v is a *factor* and $v^{(2)}$ is a *suffix* of the word u .
- A sequence $\mathbf{u} = (u_i)_{i=0}^\infty$ where $u_i \in A$ for $i \in \mathbb{N}$ is called an *infinite word over the alphabet A* . The set of all infinite words over A is denoted $A^\mathbb{N}$.

The definition of prefix, factor and suffix can be extended to $\mathbf{u} \in A^\mathbb{N}$. Then prefix and factor are finite words and suffix is an infinite word.

The infinite concatenation $uuu \cdots$ for a finite word u is denoted u^ω .

Let A be an alphabet equipped with a total order $<$ and $\mathbf{u} = u_1u_2u_3 \cdots, \mathbf{v} = v_1v_2v_3 \cdots$ be words over the alphabet A . The *lexicographical order over $A^\mathbb{N}$* is defined as

$$\mathbf{u} < \mathbf{v} \quad \Leftrightarrow \quad u_i < v_i \quad \text{for the smallest index } i \text{ such that } u_i \neq v_i.$$

1.2 Algebraic numbers

In this section we recall basic definitions concerning algebraic numbers.

A number $\beta \in \mathbb{C}$ is called *algebraic* if β is a root of a monic polynomial with rational coefficients

$$P(x) = x^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0, \quad \text{where } a_0, \dots, a_{n-1} \in \mathbb{Q}.$$

Let β be an algebraic number. Then among all polynomials from $\mathbb{Q}[x]$ (polynomials with rational coefficients) with root β , there exists one monic polynomial f of minimal degree. Moreover, f divides all polynomials $g \in \mathbb{Q}[x]$ such that $g(\beta) = 0$.

The polynomial f defined above is called *minimal polynomial of the number β* . The degree of the polynomial is the *degree of the number β* . Two algebraic numbers are *algebraic conjugates* if they have the same minimal polynomial. A number $\beta \in \mathbb{C}$ is called *algebraic integer* if there exists a monic polynomial $f \in \mathbb{Z}[x]$ such that $f(\beta) = 0$.

We consider two special cases of algebraic integers, Pisot and complex Pisot numbers.

Definition 1.1. A real algebraic integer $\beta > 1$ whose algebraic conjugates are less than 1 in absolute value is called a *Pisot number*.

Definition 1.2. Let $\beta \in \mathbb{C} \setminus \mathbb{R}$ be an algebraic integer such that $|\beta| > 1$. If all algebraic conjugates of β except for $\bar{\beta}$ are less than 1 in absolute value, then β is called a *complex Pisot number*.

Example 1.3. Here are some examples of Pisot and complex Pisot numbers:

- rational Pisot number: every $m \in \mathbb{N}, m \geq 2$ with minimal polynomial of the form $x - m$.
- quadratic Pisot number:
 - root of a monic polynomial of the form $x^2 - ax - b$ where $a, b \in \mathbb{Z}$ and $a > |b - 1|$.
 - e.g. golden ratio $\tau = \frac{1+\sqrt{5}}{2}$ with minimal polynomial of the form $x^2 - x - 1$.
- d -Bonacci number, $d \geq 2$, which is a root of $x^d - x^{d-1} - \dots - x - 1$.
 - for $d = 2$ the d -Bonacci number is equal to the golden ratio τ .
 - for $d = 3$ the so-called Tribonacci number.
- quadratic complex Pisot number:
 - root of a monic polynomial $x^2 - ax - b$ where $a, b \in \mathbb{Z}, |b| \geq 2$ and $a^2 + 4b < 0$.
 - Eisenstein number $-1 + \omega$, where $\omega = \exp \frac{2\pi i}{3}$ is the third root of unity, with minimal polynomial $x^2 + 3x + 3$.
 - Penney number $-1 + i$ with minimal polynomial $x^2 + 2x + 2$.
 - Knuth number $-2i$ with minimal polynomial $x^2 + 4$.

1.3 Number representations

Definition 1.4. Let β be a complex number, $|\beta| > 1$, and $A \subseteq \mathbb{C}$ be an alphabet of digits. By a *numeration system* we understand the ordered pair (β, A) .

A (β, A) -*representation* of a complex number z is a convergent series $\sum_{i=-\infty}^m z_i \beta^i$ where $z_i \in A$. We usually identify the representation with the infinite word of digits.

We denote $z = z_m z_{m-1} \dots z_1 z_0 . z_{-1} z_{-2} \dots$, or $z = 0 . z_{-1} z_{-2} \dots$, respectively. If the digits in the representation form an eventually periodic sequence with a period $u \in A^*$, we write $(u)^\omega$ for the infinite repetition.

We say that the (β, A) -representation of z is finite, if only finitely many digits are non-zero. In the notation, we usually omit the suffix 0^ω . The set of numbers with finite (β, A) -representation is denoted by

$$\text{Fin}_A(\beta) = \{x \in \mathbb{C} : x = \sum_{i=k}^l x_i \beta^i, k, l \in \mathbb{Z}, x_i \in A\}.$$

The decision whether every z in a certain domain has a (β, A) -representation for some β and some alphabet A is generally a difficult problem.

In the following we mention several classes of numeration systems where the situation is solved.

Positive base and integer alphabet

The case of positive base $\beta > 1$ and alphabet of consecutive non-negative digits $A = \{a \in \mathbb{Z} : 0 \leq a < \beta\}$ has been considered by Rényi [15] in 1957.

Theorem 1.5. *Let $\beta > 1$ be a real number. For every $x \in [0, 1)$ there is a (β, A) -representation $x = \sum_{i \geq 1} x_i \beta^{-i}$, where $x_i \in A = \{a \in \mathbb{Z} : 0 \leq a < \beta\}$ for $i \geq 1$.*

We can obtain such a representation by using the greedy algorithm. The representation then satisfies

$$\sum_{i > j} x_i \beta^{-i} < \beta^{-j} \quad (1.1)$$

for every $j \in \mathbb{N}$.

Definition 1.6. Let $\beta > 1$ be a real number and $x_i \in \mathbb{N}_0$ for $i \in \mathbb{N}$. A β -representation satisfying (1.1) is called β -*expansion*, we denote

$$d_\beta(x) = x_1 x_2 \cdots .$$

The so called *Rényi expansion of 1*, denoted $d_\beta(1) = t_1 t_2 t_3 \cdots$ where $t_i \in \mathbb{N}_0$ for $i \in \mathbb{N}$, is defined as

$$t_1 = \lfloor \beta \rfloor, \quad t_2 t_3 t_4 = d_\beta(\beta - \lfloor \beta \rfloor).$$

We say that β is a *Parry number* if $d_\beta(1)$ is eventually periodic. If $d_\beta(1)$ is finite (i.e. eventually periodic with the period 0^ω), β is a *simple Parry number*.

Proposition 1.7. *Let $x \in [0, 1)$. Then $d_\beta(x)$ is lexicographically the greatest among all sequences $(y_i)_{i \geq 1}$, $y_i \in \mathbb{N}_0$, such that $x = \sum_{i \geq 1} y_i \beta^{-i}$.*

Let us mention that a (β, A) -representation of any positive number x can be obtained by shifting the fractional point from $d_\beta(\frac{x}{\beta^k})$ where k is an integer such that $\frac{x}{\beta^k} \in [0, 1)$.

For representing negative numbers, one needs the minus sign.

Negative base and integer alphabet

If $\beta < -1$ is a real number and $A = \{0, 1, \dots, \lfloor \beta \rfloor\}$, one can obtain a (β, A) -representation using a modification of the greedy algorithm. The modification was introduced by Ito and Sadahiro in [11].

Theorem 1.8. *Let $\beta < -1$ be a real number. For every $x \in I_\beta = \left[\frac{\beta}{1-\beta}, \frac{1}{1-\beta} \right)$ there is a (β, A) -representation $x = \sum_{i \geq 1} x_i \beta^{-i}$ where $x_i \in A = \{a \in \mathbb{Z} : 0 \leq a \leq \beta\}$ for $i \geq 1$.*

Let us mention that for all $x \notin I_\beta$ there is an integer k such that $\frac{x}{\beta^k} \in I_\beta$. Thus we can obtain a (β, A) -representation of any real number x without using the minus sign.

We are interested in redundant systems where any number has generally multiple (β, A) -representations. Only in such numeration systems we can perform arithmetic operations using effective algorithms, in particular, parallel addition and on-line multiplication and division.

Chapter 2

Spectrum of real and complex numbers

As will be seen later, certain features of (β, A) -numeration systems depend on the properties of the corresponding spectrum of the number β .

Definition 2.1. Let β be a complex number, $|\beta| > 1$, and let $A \subset \mathbb{C}$ be a finite alphabet of digits. The A -spectrum of β is the set

$$X^A(\beta) = \left\{ \sum_{k=0}^n a_k \beta^k : n \in \mathbb{N}, a_k \in A \right\}.$$

The spectrum was first considered by Erdős [2] in the case that β is a real number, $\beta > 1$, and $A = \{0, 1, 2, \dots, m\}$. Then $X^A(\beta)$ is discrete (has no accumulation point) and its elements can be arranged into an increasing sequence $0 = x_0 < x_1 < \dots$. Many authors have been studying the value $l_m(\beta) = \liminf_{k \rightarrow \infty} (x_{k+1} - x_k)$.

We will discuss several geometric properties of the spectrum in Section 2.2 and Section 2.3.

2.1 Algorithm for generating of the spectrum

Let β be a complex number, $|\beta| > 1$, and let $A \subset \mathbb{C}$ be a finite alphabet of digits. Suppose that $X^A(\beta)$ is discrete. We can obtain the set of the points from the finite set $X^A(\beta) \cap B_R(0)$, where $B_R(0)$ denotes an open ball of radius $R > 0$ centered at the origin, see [10].

To ensure that our algorithm works, we use the following observation.

Remark. If $z \in X^A(\beta)$ satisfies $|z| > \frac{\max\{|a|:a \in A\}}{|\beta|-1}$, then $|\beta z + a| > |z|$.

We gradually create two sequences of sets $(X_i)_{i \geq 0}$ and $(\tilde{X}_i)_{i \geq 1}$. The algorithm is composed from several steps:

- Set $X_0 = A$ and $M = \max \left\{ R, \frac{\max\{|a|:a \in A\}}{|\beta|-1} \right\}$.
- Compute $\tilde{X}_n = X_{n-1} \cup \left(\bigcup_{a \in A} \beta X_{n-1} + a \right)$.
- Set $X_n = \tilde{X}_n \cap B_M(0)$.

The algorithm stops when for some $n \geq 1, n \in \mathbb{N}$, the condition $X_n = X_{n-1}$ is satisfied.

In Figure 2.1 a gradual generation of the spectrum of the golden ratio $\tau = \frac{1+\sqrt{5}}{2}$ over the alphabet

$$A = \left\{ 0, \exp\left(\frac{2\pi i}{5}\right), \exp\left(\frac{4\pi i}{5}\right), \exp\left(\frac{6\pi i}{5}\right), \exp\left(\frac{8\pi i}{5}\right), \exp(2\pi i) \right\}$$

consisting of vertices of a regular pentagon and 0 can be seen. The spectrum is displayed up to the fourth power of τ , i.e. sets \tilde{X}_n for $n \leq 4$.

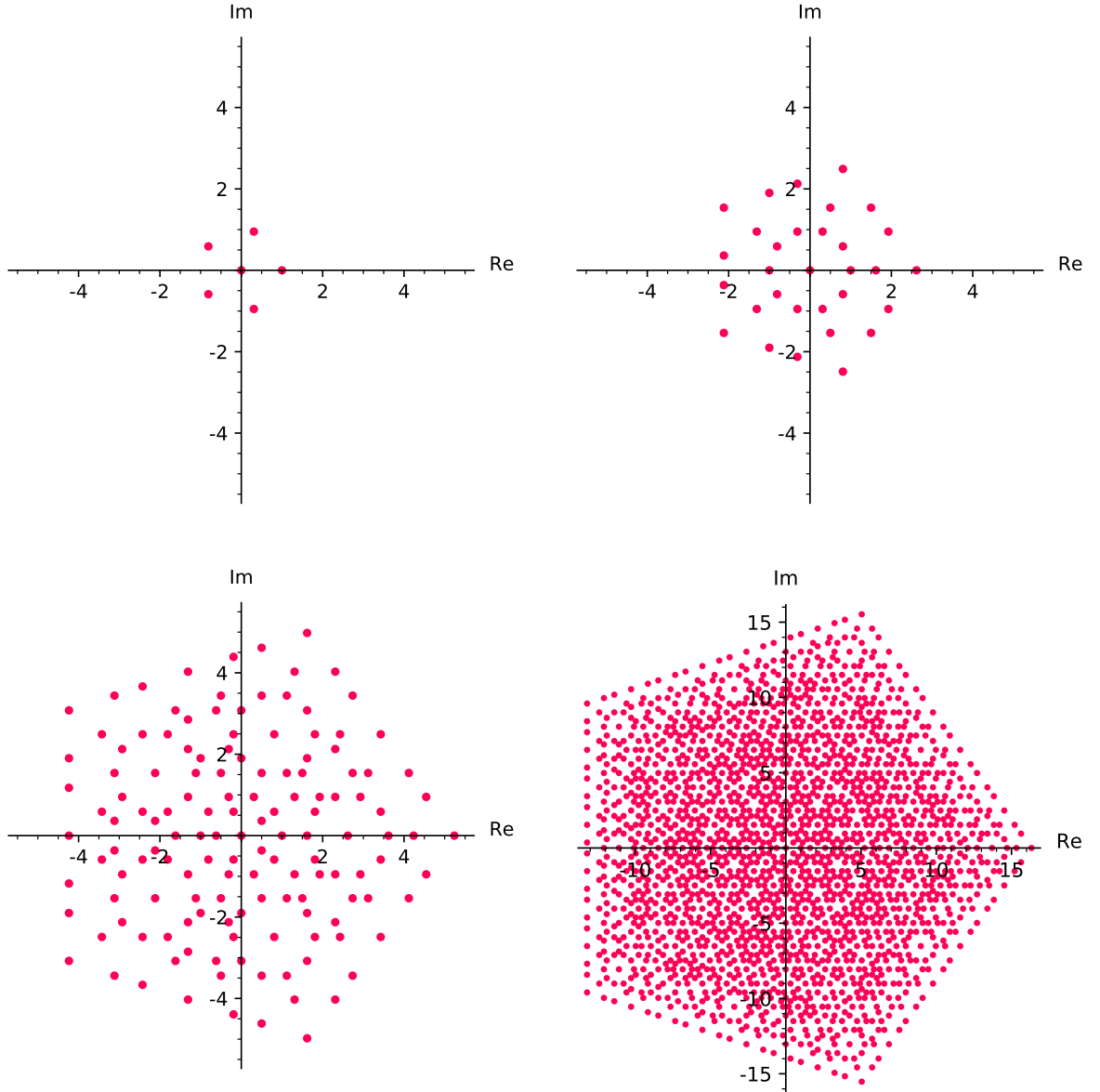


Figure 2.1: Generation of the spectrum $X^A(\tau)$ of the golden ratio $\tau = \frac{1+\sqrt{5}}{2}$ and the alphabet A consisting of a regular pentagon and the origin. Sets \tilde{X}_n for $n = 0, 1, 2, 4$ are displayed.

2.2 Discreteness of the spectrum

Usually, discreteness is a topological property, i.e. that any element of the discrete set is isolated. In the euclidean topology of \mathbb{C} , it is equivalent to the following definition.

Definition 2.2. A point set $X \subset \mathbb{C}$ is *discrete* in \mathbb{C} , if it has no accumulation point in \mathbb{C} . Analogically we define a set discrete in \mathbb{R} .

The property of the spectrum $X^A(\beta)$ to have an accumulation point is linked to (β, A) -representations of 0. Discreteness of the spectrum is also necessary and sufficient for the possibility of performing on-line division in the corresponding numeration system.

In case of a positive base $\beta > 1$ and a non-negative alphabet, the spectrum is always discrete in \mathbb{R} . The question of discreteness of the spectrum is much less obvious if the alphabet contains both positive and negative digits, or in the case of a complex base β and $A \subset \mathbb{C}$.

Let us cite several results. The case for β real and a symmetric alphabet is completely answered.

Theorem 2.3 ([1], [3]). *Let $\beta > 1$ be a real number and $A = \{-M, \dots, M\}$ be an alphabet. Then the spectrum $X^A(\beta)$ has an accumulation point if and only if $\beta < M + 1$ and β is not Pisot.*

If the alphabet is not symmetric or the base is not real, one can state at least a sufficient condition for discreteness of the spectrum.

Theorem 2.4 ([6], [9]). *Let β be a complex number, $|\beta| > 1$, and let $A \subset \mathbb{Q}(\beta)$ be an alphabet, $0 \in A$. If*

1. *β is a real number and if β or $-\beta$ is Pisot,*
2. *or $\beta \in \mathbb{C} \setminus \mathbb{R}$ is complex Pisot,*

then $X^A(\beta)$ has no accumulation point.

In [6] a necessary and sufficient condition for the discreteness of the spectrum was given using the notion of rigid representation of 0.

Definition 2.5. Let $z_1 z_2 \dots$ be a β -representation of $z = 0$ in the alphabet A , i.e. $\sum_{i \geq 1} z_i \beta^{-i} = 0$ with $z_i \in A$. Then $z_1 z_2 \dots$ is said to be *rigid* if

$$0.z_1 z_2 \dots z_j \neq 0.0 z'_2 \dots z'_j$$

for all $j \geq 2$ and for all $z'_2 \dots z'_j$ in A^* .

Notation. The signed digit (-1) is denoted $\bar{1}$.

Example 2.6. Let $\beta = 2$ and the alphabet $A = \{\bar{1}, 0, 1\}$. Then 0 has two non-trivial representations

$$0 = 0.1\bar{1}\bar{1}\bar{1} \dots = 0.\bar{1}111 \dots$$

They are not rigid, since $0.1\bar{1} = 0.01$ and $0.\bar{1}1 = 0.0\bar{1}$.

Theorem 2.7 ([6], Thm 3.5). *Let β be a complex number, $|\beta| > 1$ and let $A \subset \mathbb{C}$ be an alphabet. Then $X^A(\beta)$ has an accumulation point if and only if 0 has a rigid (β, A) -representation.*

Using the above theorem, one can derive a partial result for a real base $\beta > 1$ and a general alphabet of consecutive digits.

Proposition 2.8 ([6]). *Let $\beta > 1$ be a real number and let $\{-1, 0, 1\} \subset A = \{m, \dots, 0, \dots, M\} \subset \mathbb{Z}$ be an alphabet.*

1. *Zero has a non-trivial (β, A) -representation if and only if β satisfies*

$$\beta \leq \max\{M + 1, -m + 1\}. \quad (2.1)$$

2. *If $\beta \leq \max\{M + 1, -m + 1\}$ and β is not a Parry number, then 0 has a rigid (β, A) -representation and thus the spectrum $X^A(\beta)$ has an accumulation point.*

Proof. Let $d_\beta(1) = x_1x_2\cdots$ be the β -expansion of 1 satisfying (1.1). Then from the definition of d_β we have $\beta - 1 \leq x_1 < \beta, x_i \leq x_1$ for $i \geq 2$ and

$$0 = 0.\bar{1}x_1x_2\cdots = 0.1\bar{x}_1\bar{x}_2\cdots.$$

Thus we obtained two non-trivial representations of 0 in the base β over two different alphabets $\{-\lceil\beta\rceil + 1, \dots, \bar{1}, 0, 1\}$ and $\{\bar{1}, 0, 1, \dots, \lceil\beta\rceil - 1\}$ respectively.

Therefore if the alphabet A contains either set $\{-1, 0, 1, \dots, x_1\}$ or set $\{-x_1, \dots, -1, 0, 1\}$, zero has a non-trivial (β, A) -representation. In other words, $x_1 \in A$ implies that $M \geq x_1 \geq \beta - 1$ and similarly $-x_1 \in A$ implies $m \leq -x_1 \leq -\beta + 1$ which in total means that $\beta \leq \max\{M + 1, -m + 1\}$.

On the other hand, assume that β does not satisfy (2.1), i.e. $M < \beta - 1$ and $m > -\beta + 1$. Then for any $z = \sum_{i \geq 1} z_i \beta^{-i}$ with $z_i \in A, z_1 \geq 1$ the following stands

$$z \geq \frac{1}{\beta} + \sum_{i \geq 2} \frac{m}{\beta} = \frac{\beta - 1 + m}{\beta(\beta - 1)} > 0.$$

Similarly, if $z_1 \leq -1$, then $z < 0$. Therefore 0 does not have a non-trivial (β, A) -representation.

Now let us assume that β is not a Parry number. Let $(r_n)_{n \geq 1}$ be the sequence of the n th tails of the β -expansion of 1, $r_n = 0.x_{n+1}x_{n+2}\cdots$. Since β is not a Parry number, (r_n) is injective. In [6] was proven that if $z_1z_2\cdots$ is (β, A) -representation of 0 and the sequence $(r_n)_{n \geq 1}$ is injective, then the spectrum $X^A(\beta)$ has an accumulation point. By Theorem 2.7, zero has a rigid (β, A) -representation. \square

2.3 Relative density

Definition 2.9. Let $A \subset \mathbb{C}$. We say that A is *relatively dense* if there exists some $\varepsilon > 0$ such that for all $z \in \mathbb{C}$ the condition $B_\varepsilon(z) \cap A \neq \emptyset$ holds.

Relative density of the spectrum has the following implication for the numeration system.

Theorem 2.10 ([10], Thm 2.9). *Let β be a complex number, $|\beta| > 1$, and let $A \subset \mathbb{C}$ be finite. If $X^A(\beta)$ is discrete, then the following are equivalent:*

1. *$X^A(\beta)$ is relatively dense in \mathbb{C} .*
2. *Every $z \in \mathbb{C}$ is (β, A) -representable.*

A numeration system (β, A) in which every complex number is representable is sometimes called *complete*.

For an alphabet composed of any complex numbers, a necessary condition for relative density (and thus completeness of the corresponding numeration system) is formulated using the number of letters in the alphabet.

Theorem 2.11 ([10], Thm 2.4). *Let β be a complex number, $|\beta| > 1$, and let $A \subset \mathbb{C}$ be finite. If $\#A < |\beta|^2$, then the set $X^A(\beta)$ is not relatively dense.*

In case of a real base β and a real alphabet A , one can state an analogy of Theorem 2.10, namely that relative density of the spectrum in \mathbb{R} is equivalent to completeness of the system in \mathbb{R} .

Pedicini focused on the case where the alphabet is composed of any (not necessarily integer) real numbers.

Theorem 2.12 ([14]). *Let $\beta > 1$ be a real number and let $A = \{a_1, a_2, \dots, a_m\} \subset \mathbb{R}$ be an alphabet where $a_1 < a_2 < \dots < a_m$. If*

$$\max_{1 \leq j \leq m-1} \{a_{j+1} - a_j\} < \frac{a_m - 1}{\beta - 1},$$

then every $x \in \left(\frac{a_1}{\beta-1}, \frac{a_m}{\beta-1}\right]$ has a representation $x = 0.x_1x_2 \dots$.

Chapter 3

Parallel addition

In order to execute on-line division and multiplication fast, we need to ensure that the numeration system satisfies conditions for parallel addition [7].

In usual algorithm for addition performed by hand, the most significant digit of the sum may depend on the least significant digit of the input, as can be seen on the simple example

$$\begin{array}{r} 999 \dots 9 \\ +1 \\ \hline 1000 \dots 0 \end{array}$$

This is the case in numeration systems which are not redundant. On the other hand in redundant numeration systems one may design algorithm for addition in which the digit on position j of the result depends only on digits $j+t, \dots, j-r$ of the inputs for fixed r, t . The addition is then performed in constant time.

Let us formalize the above description of parallel addition and its properties, see [4].

3.1 Local functions and parallel addition

When performing addition of numbers written in base β with digits in a given alphabet A , $x = \sum_{i \geq 0} x_i \beta^i, y = \sum_{j \geq 0} y_j \beta^j$ where $x_i, y_j \in A$, one aims to obtain the sum

$$z = x + y = \sum_{i \geq 0} (x_i + y_i) \beta^i$$

again in the form $z = \sum_{k \geq 0} z_k \beta^k, z_k \in A$. In fact, it consists of transforming a string of digits over the alphabet $A + A$ into a string of digits over A , so that the value of the corresponding number remains unchanged. Such a transformation is called *digit set conversion*.

Definition 3.1. Let $\beta \in \mathbb{C}$ be a base, $|\beta| > 1$ and let A and B be two alphabets of complex numbers containing 0. A *digit set conversion* in base β from A to B is a function $\varphi : A^{\mathbb{Z}} \rightarrow B^{\mathbb{Z}}$ such that

1. for any $u = (u_j)_{j \in \mathbb{Z}} \in A^{\mathbb{Z}}$ with a finite number of non-zero digits, the image $v = \varphi(u) = (v_j)_{j \in \mathbb{Z}} \in B^{\mathbb{Z}}$ has only a finite number of non-zero digits as well,
2. $\sum_{j \in \mathbb{Z}} v_j \beta^j = \sum_{j \in \mathbb{Z}} u_j \beta^j$.

In order that the digit set conversion is performed in constant time, it should be computable using a local function.

Definition 3.2. A function $\varphi : A^{\mathbb{Z}} \rightarrow B^{\mathbb{Z}}$ is said to be *p-local* if there exist two non-negative integers r and t satisfying $p = r + t + 1$, and a function $\psi : A^p \rightarrow B$ such that for any $u = (u_j)_{j \in \mathbb{Z}} \in A^{\mathbb{Z}}$ and its image $v = \varphi(u) = (v_j)_{j \in \mathbb{Z}} \in B^{\mathbb{Z}}$, we have $v_j = \psi(u_j + t \cdots u_{j-r} - r)$ for every $j \in \mathbb{Z}$.

Definition 3.3. A digit set conversion φ in base $\beta \in \mathbb{C}, |\beta| > 1, \varphi : A^{\mathbb{Z}} \rightarrow B^{\mathbb{Z}}$ is said to be *computable in parallel* if it is a *p-local* function for some $p \in \mathbb{N}$.

Also, a digit set conversion in base β from A to B is said to be *block parallel computable* if there exists some $k \in \mathbb{N}$ such that the digit set conversion in base β^k from $A_{(k)}$ to $B_{(k)}$ is computable in parallel. When the specification of k is needed, we say *k-block parallel computable*.

In this terminology, the original parallel addition is 1-block parallel addition.

Definition 3.4. The algorithm for parallel addition is *neighbour-sensitive* if the decision for position i depends on the digit at position $i - 1$. Otherwise we say that the algorithm for parallel addition is *neighbour-free*.

In [7] a sufficient condition for existence of a parallel addition algorithm in base β was given.

Theorem 3.5 ([4]). *Let β be an algebraic number such that $|\beta| > 1$ and all its conjugates in modulus differ from 1. Then there exists an alphabet A of consecutive integers containing 0 such that addition on $\text{Fin}A(\beta)$ can be performed in parallel.*

Parallel addition is possible only in numeration systems with sufficiently large alphabet with respect to the base. An estimate on the necessary cardinality of the alphabet was formulated in [8].

Theorem 3.6. *Let β , with $|\beta| > 1$, be an algebraic integer of degree d with minimal polynomial*

$$f(x) = x^d - a_{d-1}x^{d-1} - a_{d-2}x^{d-2} - \cdots - a_1x - a_0.$$

Let A be an alphabet of consecutive integers containing 0 and 1. If addition in $\text{Fin}_A(\beta)$ is computable in parallel, then $\#A \geq |f(1)|$. If, moreover, β is a positive real number, $\beta > 1$, then $\#A \geq |f(1)| + 2$.

A method of constructing parallel algorithms in given numeration systems was presented in [13].

3.2 Neighbour-free parallel addition

Given a base β , it may be possible to design several different algorithms for parallel addition with different alphabets of digits. One is of course interested in algorithms in numeration systems with small alphabets. However, the digit set conversion is then performed using *p-local* function with large p , or, one is even forced to use *k-block* algorithms with $k \geq 2$. On the other hand, most simple parallel algorithms are 1-block neighbour free, on the expense of taking a larger alphabet.

We will consider base β an algebraic number, $|\beta| > 1$, and search for an alphabet A so that the numeration system (β, A) allows a neighbour-free algorithm to be performed.

We will define a property which will ensure that the algorithms used for parallel addition are neighbour-free.

Definition 3.7. Let β be a complex number, $|\beta| > 1$. We say that β satisfies *t-representation of zero property* where $t \in \mathbb{N}$ if there exist coefficients $b_k, b_{k-1}, \dots, b_1, b_0, b_{-1}, \dots, b_{-h}$ with $b_i \in \mathbb{Z}$ for some $k, h \in \mathbb{N}_0$ such that β is a root of the polynomial

$$T(x) = b_k x^k + b_{k-1} x^{k-1} + \dots + b_1 x + b_0 + b_{-1} x^{-1} + \dots + b_{-h} x^{-h} \quad (3.1)$$

and

$$b_0 > t \sum_{i \in \{-h, \dots, k\} \setminus \{0\}} |b_i|.$$

We say that β satisfies the *strong representation of zero property* (SRZ) if $t \geq 2$ and β satisfies the *weak representation of zero property* (WRZ) if $t \geq 1$.

Notation. We set $B = b_0$ and $M = \sum_{i \in \{-h, \dots, k\} \setminus \{0\}} |b_i|$.

The proof of the following theorem can be found in [7].

Theorem 3.8. Let β be a complex number, $|\beta| > 1$, and let A be a symmetric alphabet of the form $A = \{-a, \dots, 0, \dots, a\}$. If β satisfies SRZ, then there exists a neighbour-free algorithm which realizes addition in constant time in parallel in $\text{Fin}_A(\beta)$ where $a = \lceil \frac{B-1}{2} \rceil + \lceil \frac{B-1}{2(B-2M)} \rceil M$.

If β satisfies WRZ, then there exists a neighbour-free algorithm which realizes addition in constant time in parallel in $\text{Fin}_A(\beta)$ where $a = \lceil \frac{B-1}{2} \rceil + M$.

We can find strong and weak polynomials of β using a constructive proof of the following proposition, again taken from [7].

Proposition 3.9. Let β be an algebraic number of degree d with algebraic conjugates $\beta_1, \beta_2, \dots, \beta_d$ (including β itself). Let $|\beta_i| \neq 1$ for all $i \in \{1, 2, \dots, d\}$ and $|\beta| > 1$. Then for any $t \geq 1$ there exists a polynomial

$$Q(x) = a_m x^m + a_{m-1} x^{m-1} + \dots + a_1 x + a_0 \in \mathbb{Z}[x]$$

and an index $i_0 \in \{1, \dots, m\}$ such that

$$Q(\beta) = 0 \quad \text{and} \quad |a_{i_0}| > t \sum_{\substack{i=0 \\ i \neq i_0}}^m |a_i|.$$

Proof. Let us denote the minimal polynomial of β as

$$F(x) = \prod_{i=1}^d (x - \beta_i) = x^d + f_{d-1} x^{d-1} + \dots + f_1 x + f_0 \in \mathbb{Q}[x].$$

Let M be the companion matrix of the polynomial $F(x)$, i.e.

$$M = \begin{pmatrix} -f_{d-1} & 1 & 0 & 0 & \dots & 0 \\ -f_{d-2} & 0 & 1 & 0 & \dots & 0 \\ -f_{d-3} & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -f_1 & 0 & 0 & 0 & \dots & 1 \\ -f_0 & 0 & 0 & 0 & \dots & 0 \end{pmatrix} \in \mathbb{Q}^{d \times d}.$$

We can easily prove that

$$\det(M - xI) = (-1)^d F(x),$$

where I denotes the unit matrix of the size $d \times d$. We can also notice that the numbers β_i are eigenvalues of M . We define for any $n \in \mathbb{N}$ the following polynomial:

$$F_n(x) = \prod_{i=1}^d (x - \beta_i^n) = f_d(n)x^d + f_{d-1}(n)x^{d-1} + \cdots + f_1(n)x + f_0(n).$$

As the matrix M^n has eigenvalues $\beta_1^n, \dots, \beta_d^n$, it can be also easily proven that

$$\det(M^n - xI) = (-1)^d \prod_{i=1}^d (x - \beta_i^n) = (-1)^d F_n(x).$$

Since $M \in \mathbb{Q}^{d \times d}$, the matrix M^n is also from $\mathbb{Q}^{d \times d}$ and therefore the determinant $\det(M^n - xI)$ is a polynomial with rational coefficients and consequently, for all $n \in \mathbb{N}$, the polynomial $F_n(x)$ has also rational coefficients. We can see that for every $n \in \mathbb{N}$ and for every $i \leq d$ the following equality stands:

$$f_i(n) = (-1)^{d-i} \sum_{\{j_1, j_2, \dots, j_i\} \in S_i} \beta_{j_1}^n \beta_{j_2}^n \cdots \beta_{j_i}^n, \quad (3.2)$$

where S_i denotes the set of all subsets of $\{1, 2, \dots, d\}$ with i elements.

Without loss of generality, we assume that $|\beta_1| \geq |\beta_2| \geq \cdots \geq |\beta_d|$ and denote j_0 the greatest index for which the algebraic conjugate $|\beta_{j_0}| > 1$, i.e. $j_0 = \max\{i : |\beta_i| > 1\}$. Our choice of the index j_0 ensures that

$$\left| \frac{\beta_{j_1} \beta_{j_2} \cdots \beta_{j_r}}{\beta_1 \beta_2 \cdots \beta_{j_0}} \right| < 1$$

for every subset $\{j_1, j_2, \dots, j_r\} \subset \{1, 2, \dots, d\}$ such that $\{j_1, j_2, \dots, j_r\} \neq \{1, 2, \dots, j_0\}$. Therefore the following stands

$$\lim_{n \rightarrow \infty} \frac{\beta_{j_1}^n \beta_{j_2}^n \cdots \beta_{j_r}^n}{\beta_1^n \beta_2^n \cdots \beta_{j_0}^n} = 0.$$

Since the coefficients $f_i(n)$ of the polynomial $F_n(x)$ satisfy (3.2) for every $n \in \mathbb{N}$, we can deduce that the limit

$$\lim_{n \rightarrow \infty} \frac{f_i(n)}{\beta_1^n \beta_2^n \cdots \beta_{j_0}^n} = \begin{cases} 0 & \text{for every } i \in \{1, \dots, d\}, i \neq j_0, \\ (-1)^{d-i} & \text{for } i = j_0. \end{cases}$$

This implies that for every $t \geq 1$ there exists some $n_0 = n_0(t) \in \mathbb{N}$ such that

$$|f_{j_0}(n_0)| > t \sum_{\substack{i=1 \\ i \neq j_0}}^m |f_i(n_0)|.$$

We can equivalently write

$$\frac{|f_{j_0}(n_0)|}{|\beta_1^{n_0} \beta_2^{n_0} \cdots \beta_{j_0}^{n_0}|} > t \sum_{\substack{i=1 \\ i \neq j_0}}^m \frac{|f_i(n_0)|}{|\beta_1^{n_0} \beta_2^{n_0} \cdots \beta_{j_0}^{n_0}|}.$$

Such $n_0 = n_0(t)$ exists, because the right side of the inequality has the limit 0 and the left side has the limit 1.

Therefore, to construct the polynomial $Q(x)$, we fix n_0 and denote K the least common multiple of the denominators of the rational numbers $f_1(n_0), \dots, f_{d-1}(n_0), f_d(n_0)$. Then we get the polynomial $Q(x) = KF_{n_0}(x^{n_0})$ and the index $i_0 = n_0 j_0$. Thus the obtained polynomial has the required properties. \square

Examples of the use of the construction presented in the above proof are given in Chapter 6.

Chapter 4

On-line division

On-line arithmetic is a mode of computation where operands and results are processed in a digit serial manner, starting with the most significant digit. We will focus on the algorithm for on-line division in particular.

The algorithm for on-line division was introduced by Trivedi and Ercegovic for computation in integer bases with symmetric alphabet [16]. In [5] a modification was presented for non-standard numeration systems for arbitrary base $\beta, |\beta| > 1$ (in general a complex number) and alphabet A (in general a finite set of complex numbers).

4.1 Algorithm for on-line division

The algorithm for on-line division in (β, A) numeration system has two parameters:

- the delay $\delta \in \mathbb{N}$,
- the minimal value (in modulus) of the denominator $D_{min} > 0$.

We will work with (β, A) -representations of:

- the numerator $N = \sum_{j=1}^{+\infty} n_j \beta^{-j}$,
- the denominator $D = \sum_{j=1}^{+\infty} d_j \beta^{-j}$,
- their quotient $Q = \sum_{j=1}^{+\infty} q_j \beta^{-j}$.

Partial sums are denoted by $N_k = \sum_{j=1}^k n_j \beta^{-j}$, $D_k = \sum_{j=1}^k d_j \beta^{-j}$, and $Q_k = \sum_{j=1}^k q_j \beta^{-j}$.

The inputs of the algorithm are two (possibly infinite) strings representing the nominator N and the denominator D , namely

$$\begin{aligned} &0.n_1 n_2 \cdots n_\delta n_{\delta+1} n_{\delta+2} \cdots \quad \text{where } n_j \in A \text{ and } n_1 = n_2 = \cdots = n_\delta = 0, \text{ and} \\ &0.d_1 d_2 d_3 \cdots \quad \text{where } d_j \in A \text{ satisfying } |D_k| \geq D_{min} \text{ for all } k \in \mathbb{N}, k \geq 1. \end{aligned} \quad (4.1)$$

The output is an arbitrarily long string $0.q_1 q_2 q_3 \cdots$ corresponding to a (β, A) -representation of the quotient

$$Q = \frac{N}{D} = \sum_{j=1}^{+\infty} q_j \beta^{-j}.$$

Note that the representation of Q starts behind the fractional point. This is ensured by the setting of the algorithm.

We realize the on-line division in iterative steps. In the beginning, we set $W_0 = q_0 = Q_0 = 0$. For $k \geq 1$, the step of the iteration proceeds by calculation of

$$W_k = \beta(W_{k-1} - q_{k-1}D_{k-1+\delta}) + (n_{k+\delta} - Q_{k-1}d_{k+\delta})\beta^{-\delta}. \quad (4.2)$$

The k -th digit q_k of the representation of the quotient $Q = \frac{N}{D}$ is evaluated by **Select**, function of the values of the auxiliary variable W_k and the interim representation $D_{k+\delta}$, so that

$$q_k = \text{Select}(W_k, D_{k+\delta}) \in A.$$

The selection function is chosen so to ensure correctness of the algorithm for on-line division. You can see more details about the setting of the function **Select** in [5]. We omit the description as it is technical and not important for our study.

The algorithm works for the numeration systems which satisfy the so called OL property.

Definition 4.1. A numeration system (β, A) has the *OL Property* if there exists a bounded set I such that $0 \in I$ and

$$\beta \text{cl}(I) \subset \bigcup_{a \in A} (I^\circ + a).$$

Whether the on-line division can be performed depends not only on the OL Property of the numeration system, but also on the condition on the string representing the denominator (4.1). This fact is stated in the following theorem.

Theorem 4.2 ([5], Thm 3.12). *Assume that the OL Property is satisfied in the (β, A) -numeration system. Let strings $0.n_1n_2\cdots$ and $0.d_1d_2\cdots$ satisfying condition (4.1) represent numbers N and D respectively. Then computing $\frac{N}{D}$ can be performed by the Trivedi-Ercegovac algorithm.*

4.2 OL Property

In this section, we will discuss cases in which the OL Property is satisfied. For real bases and integer alphabets, a sufficient condition for OL Property is given by the following lemma.

Lemma 4.3 ([5], Lemma 4.1). *Let β be a real number, $|\beta| > 1$, and let $A = \{m, \dots, 0, \dots, M\} \subset \mathbb{Z}$ be an alphabet where $m \leq 0 \leq M$. If*

$$|\beta| < \#A = M - m + 1,$$

then the numeration system (β, A) has the OL Property.

The particular cases for $\beta > 1$ or $\beta < -1$ are in detail discussed in [5]. For complex bases and integer alphabets, we cite only a general result for systems with a symmetric alphabet [5].

Theorem 4.4. *Let $\beta \in \mathbb{C} \setminus \mathbb{R}$ with $|\beta| > 1$ and let $A = \{-M, \dots, M\} \subset \mathbb{Z}$ be an alphabet where $M \geq 1$. If*

$$\beta\bar{\beta} + |\beta + \bar{\beta}| < \#A = 2M + 1,$$

then the numeration system (β, A) has the OL Property.

Besides that, redundant Eisenstein system and redundant Penney system were treated in [5].

Proposition 4.5. Let $\beta = -1 + \omega$ where $\omega = \exp \frac{2\pi i}{3}$ is the third root of unity and $A = \{0, \pm 1, \pm \omega, \pm \omega^2\}$ (redundant Eisenstein system). The (β, A) -numeration system satisfies the OL Property.

Proposition 4.6. Let $\beta = -1 + i$ and $A = \{0, \pm 1, \pm i\}$ (redundant Penney system). The (β, A) -numeration system satisfies the OL Property.

4.3 Preprocessing

When making division, we need that the divisor stays away from 0. By definition of the on-line algorithm, this means that the value of all the prefixes of the divisor $d_1 d_2 \dots$ must be greater in absolute value than $D > 0$. So the divisor must be preprocessed before making the division.

Example. Let us take $\beta = 2$ and the alphabet $A = \{\bar{1}, 0, 1\}$. From Example 2.6 we know that zero has two non-trivial representations in (β, A) numeration system. This means that even if a divisor has a representation with infinitely many non-zero digits, its numerical value can be still 0, which we have to avoid.

So we consider only divisors not starting on $1\bar{1}\dots$ because they can be rewritten to $01\dots$. For the same reason we do not consider divisors of the type $\bar{1}1\dots$. It can be shown that any divisor with a different prefix has a non-zero value.

Definition 4.7. We say that a complex numeration system (β, A) allows preprocessing if there exists $D_{min} > 0$ and a finite list \mathcal{L} of identities of the type $0.w_k \dots w_0 = 0.0u_{k-1} \dots u_0$ with digits in A such that any string $d_1 d_2 \dots$ on A without prefix $w_k \dots w_0$ from \mathcal{L} satisfies $|0.d_1 d_2 \dots d_j| > D_{min}$ for all $j \in \mathbb{N}$.

We must have at least $d_1 \neq 0$ after preprocessing, so the preprocessing starts by shifting the fractional point to the most significant non-zero digit of the (β, A) -representation of the divisor. After preprocessing the value of the original divisor v has been changed into a new divisor d which is just a shift of the original one, so $d = v\beta^k$ for some $k \in \mathbb{Z}$.

If zero has only the trivial (β, A) -representation, we can equivalently rewrite this fact as

$$\inf \mathcal{R} > 0, \quad \text{where } \mathcal{R} = \left\{ \left| \sum_{i \geq 1} z_i \beta^{-i} \right| : z_1 \neq 0, z_i \in A \right\}.$$

In this case the numeration system (β, A) allows preprocessing, since D_{min} is trivially equal to $\inf \mathcal{R}$ and the list of rewriting rules is empty.

Example. Let $\beta = \frac{3+\sqrt{5}}{2}$ and the alphabet $A = \{-1, 0, 1\}$. According to Proposition 2.8 the numeration system (β, A) does not have non-trivial 0 representation since $\beta > 2$. Therefore the parameter $D_{min} = \inf \mathcal{R} = \frac{\beta-1+\min A}{\beta(\beta-1)} = 0.145898\dots$ and the list of rewriting rules is empty.

The following theorem links the property of the spectrum $X^A(\beta)$ of having an accumulation point to the preprocessing in on-line division in the (β, A) -numeration system. Since this theorem is directly linked to our research problem, we include it with its proof, which is based on several lemmas, see [6].

Theorem 4.8 ([6], Thm 5.4). *Let β be a complex number and let $A \subset \mathbb{C}$ be an alphabet. The numeration system (β, A) allows preprocessing if and only if the spectrum $X^A(\beta)$ has no accumulation point.*

Notation. In the following three lemmas we will use notation

$$H = \max \left\{ \left| \sum_{i \geq 1} d_i \beta^{-i} \right| : d_i \in A \text{ for all } i \in \mathbb{N} \right\}. \quad (4.3)$$

Lemma 4.9. *If 0 has a rigid (β, A) -representation then the numeration system (β, A) does not allow preprocessing.*

Proof. Let us presume that there exists a rigid representation of 0, denote

$$0 = 0.z_1 z_2 z_3 \cdots, \quad (4.4)$$

and assume that preprocessing is possible with the value of parameter $D_{\min} > 0$. First we find $j \in \mathbb{N}$ such that $\frac{H}{|\beta|^j} < D_{\min}$. Then we consider the string $0.z_1 z_2 z_3 \cdots z_j 000 \cdots$. Since the representation of 0 is rigid, no prefix of the string $z_1 z_2 z_3 \cdots z_j$ is contained in the list of rewriting rules. But from (4.4) we know that $|0.z_1 z_2 z_3 \cdots z_j| = |0.\underbrace{000 \cdots 0}_{j\text{-times}} z_{j+1} z_{j+2} \cdots| < \frac{H}{|\beta|^j} < D_{\min}$,

which is a contradiction. \square

Lemma 4.10. *Let us assume that $X^A(\beta)$ has no accumulation point and fix $K > 0$. Then there exists $m \in \mathbb{N}$ such that any string $x_{m-1} x_{m-2} \cdots x_1 x_0$ of length m over A satisfies either*

$$|x_{m-1} \beta^{m-1} + x_{m-2} \beta^{m-2} + \cdots + x_1 \beta + x_0| \geq K$$

or there exists a string $y_{k-1} y_{k-2} \cdots y_1 y_0$ of length $k < m$ over A such that

$$\begin{aligned} & x_{m-1} \beta^{m-1} + x_{m-2} \beta^{m-2} + \cdots + x_1 \beta + x_0 \\ &= y_{k-1} \beta^{k-1} + y_{k-2} \beta^{k-2} + \cdots + y_1 \beta + y_0. \end{aligned}$$

Proof. Since the spectrum $X^A(\beta)$ has no accumulation point, the set of points in $X^A(\beta)$ in absolute value smaller than K is finite, i.e. the set $S = \{z \in X^A(\beta) : |z| < K\}$ is finite.

We will denote for every $x \in X^A(\beta)$ the number

$$\rho(z) = \min \left\{ n \in \mathbb{N} : z = \sum_{i=0}^n z_i \beta^i \text{ where } z_i \in A \right\}$$

and $m = 2 + \max\{\rho(z) : z \in S\}$. Let us take some x with (β, A) -representation

$$x = x_{k-1} \beta^{k-1} + x_{k-2} \beta^{k-2} + \cdots + x_1 \beta + x_0, \quad x_i \in A.$$

Obviously, $x \in X^A(\beta)$. Then either $|x| \geq K$, which is the first case, or $x \in S$. In that case $k \leq \max\{\rho(z) : z \in S\} \leq m - 1$. \square

Lemma 4.11. *If the spectrum $X^A(\beta)$ has no accumulation point, then there exists $D_{\min} > 0$ and $m \in \mathbb{N}$ such that for all infinite strings $d_1 d_2 \cdots$ over A one has*

(i) *either $|0.d_1 d_2 \cdots d_j| \geq D_{\min}$ for all $j \in \mathbb{N}$,*

(ii) or $0.d_1d_2\cdots d_m \neq 0.0d'_2d'_3\cdots d'_m$ for some string $d'_2d'_3\cdots d'_m \in A^*$.

Proof. Let $\varepsilon > 0$ and let us apply Lemma 4.10 where $K = H + \varepsilon$ to get the parameter $m \in \mathbb{N}$. We will denote the set

$$\mathcal{D} = \{|0.d_1d_2\cdots d_j| : j < m \text{ and } 0.d_1d_2\cdots d_j \neq 0.0d'_2d'_3\cdots d'_j\}$$

which does not contain 0. Since $X^A(\beta)$ has no accumulation point, the set \mathcal{D} is finite. Thus $D' = \min \mathcal{D} > 0$.

We consider an infinite string $d_1d_2\cdots$ and assume that $0.d_1d_2\cdots d_m \neq 0.0d'_2d'_3\cdots d'_m$ for all $d'_2d'_3\cdots d'_m \in A^*$. There may be two cases:

- $j < m$ where $j \in \mathbb{N}$. Then $0.d_1d_2\cdots d_j \neq 0.0d'_2\cdots d'_j$, otherwise it would mean that $0.d_1d_2\cdots d_m = 0.0d'_2d'_3\cdots d'_jd_{j+1}d_m$ which is a contradiction. Hence $|0.d_1d_2\cdots d_j| \geq D'$.
- $j \geq m$ where $j \in \mathbb{N}$. In this case

$$\begin{aligned} |0.d_1d_2\cdots d_j| &\geq |0.d_1d_2\cdots d_m| - \frac{1}{|\beta|^m} |0.d_{m+1}d_{m+2}\cdots d_j| \\ &\geq \frac{1}{|\beta|^m} K - \frac{1}{|\beta|^m} H = \frac{\varepsilon}{|\beta|^m}. \end{aligned}$$

Therefore, we can set the value

$$D_{min} = \min \left\{ D', \frac{\varepsilon}{|\beta|^m} \right\} \quad (4.5)$$

and the theorem is hereby proved. \square

Proof of Theorem 4.8. If the spectrum $X^A(\beta)$ has an accumulation point, by Theorem 2.7 zero has a rigid (β, A) -representation. Lemma 4.9 implies that the numeration system (β, A) does not allow preprocessing.

Assume that the spectrum does not have an accumulation point. Then Lemma 4.11 enables us to find a list \mathcal{L} of identities for preprocessing. \square

Using the constructive proof of Lemma 4.11 we can perform the algorithm for preprocessing in the following section.

4.3.1 Algorithm for preprocessing

In this section we will construct the algorithm for preprocessing. Every preprocessing consist of three steps:

1. Given a complex base β where $|\beta| > 1$, find an alphabet A ensuring on-line division in the numeration system (β, A) is possible.
2. Check if the numeration system (β, A) has a non-trivial 0 representation.
3. If there exists a non-trivial 0 representation, find the minimal list of rewriting rules and the value of parameter D_{min} .

4.3.1.1 Step 1

Step 1 of the algorithm of the preprocessing is settled by the following theorems, see [5].

Theorem 4.12. *Let β be a real number with $|\beta| > 1$ and $A = \{m, \dots, 0, \dots, M\} \subset \mathbb{Z}$. Let us assume that $m \leq 0 < M$ for $\beta > 1$, and $m \leq 0 \leq M$ for $\beta < -1$. If $|\beta| < \#A = M - m + 1$, then division in the numeration system (β, A) is on-line performable by the Trivedi-Ercegovac algorithm.*

For complex bases and integer alphabet we cite the following result.

Theorem 4.13. *Let $\beta \in \mathbb{C} \setminus \mathbb{R}$ with $|\beta| > 1$ and let $A = \{-M, \dots, M\} \subset \mathbb{Z}$ be an alphabet where $M \geq 1$. If*

$$\beta\bar{\beta} + |\beta + \bar{\beta}| < \#A = 2M + 1,$$

then division in the numeration system (β, A) is on-line performable by the Trivedi-Ercegovac algorithm.

Besides that, redundant Eisenstein system and redundant Penney system were treated in [5].

Theorem 4.14. *Let $\beta = -1 + i$ and $A = \{0, \pm 1, \pm i\}$ (redundant Penney system) or let $\beta = -1 + \omega$ where $\omega = \exp \frac{2\pi i}{3}$ is the third root of unity and $A = \{0, \pm 1, \pm \omega, \pm \omega^2\}$ (redundant Eisenstein system). Then division in (β, A) -numeration system is on-line performable by the Trivedi-Ercegovac algorithm.*

4.3.1.2 Step 2

We will divide step 2 of the algorithm of the preprocessing into two cases. The first case is for β real. In this case the answer is given by the following theorem which is also part of Proposition 2.8.

Theorem 4.15 ([5]). *Let $\beta > 1$ be a real number and $\{-1, 0, 1\} \subset A = \{m, \dots, 0, \dots, M\} \subset \mathbb{Z}$. Then 0 has a non-trivial (β, A) -representation if and only if*

$$\beta \leq \max\{M + 1, -m + 1\}.$$

The second case, for β complex, is more complicated. Partial answer is given in the following statement.

Theorem 4.16 ([17]). *Let $H > 0$ and let $\beta \in \mathbb{C}$ be an algebraic number without conjugates on the unit circle. Then, there is a finite automaton $Z(H)$ which recognizes words $d_m \cdots d_0 \in \{-H, \dots, H\}^*$ which satisfy $\sum_{i=0}^m d_i \beta^i = 0$.*

4.3.1.3 Step 3

We use our program to find the minimal list of rewriting rules and the value of the parameter D . Description of the program can be found in Chapter 5.

4.4 Time complexity

We also consider the time complexity of the algorithm for on-line division. In the algorithm, the inputs can be infinite strings. Therefore, by time complexity we understand the number of elementary operations needed to get n digits of the output.

The time complexity depends on the number of steps needed to compute the auxiliary value W_k and on the Select function. If both these operations can be performed in constant time, the time complexity of computing the first n digits of the result is $\mathcal{O}(n)$.

As we can see from (4.2), the time complexity of computation of W_k heavily depends on whether we can perform addition and subtraction fast. If we can compute addition in parallel, the time complexity of this operation is $\mathcal{O}(1)$.

Conditions on the numeration system (β, A) so that it allows parallel addition were discussed in Chapter 3.

Chapter 5

Description of the program

In this chapter several computer programs implementing algorithms from the previous chapters are described. In particular, the programs implementing algorithm for

- generating of spectrum,
- finding of WRZ and SRZ polynomials,
- finding of H in algorithm for preprocessing,
- preprocessing for on-line division

will be discussed. The source code can be found on the following GitHub page:

<https://github.com/pajav7/VU.git> .

Due to its extensive functionality for number theory, an open-source software SageMath was used. SageMath is Python-based software built on top of various open-source packages like NumPy, matplotlib etc.

The main tool of SageMath used is function `NumberField` which takes at least 1 parameter (the minimal polynomial \mathbf{f} of the generator of number field) and creates a number field:

```
N.<beta> = NumberField(f).
```

If \mathbf{f} is not irreducible, the function will result in error. Function `places` was also used on each number field to get a set of real and complex places as homomorphisms:

```
N.places().
```

In the following sections, we will discuss details of implementation of particular algorithms and tools of SageMath used in functions of the programs.

5.1 Generating of spectrum

The purpose of this program is to generate sets \tilde{X}_n from the algorithm presented in Section 2.1. It is realized in function `getSpectrumForM` which has 3 main arguments:

- list in which the spectrum will be stored,
- alphabet A ,
- integer n , maximal degree of β ,

and other arguments used while plotting the set \tilde{X}_n .

5.2 Finding of WRZ and SRZ polynomials

This program is used for finding polynomials which satisfy weak and strong representation of zero property. The algorithm is described in constructive proof of Proposition 3.9.

While implementing this algorithm, the function `companion_matrix` was used on minimal polynomial f of algebraic number β in order to create a companion matrix of f . Then the program searches in a loop for a power of companion matrix of f which ensures that characteristic polynomial of the resulting matrix satisfy WRZ or SRZ. After finding such polynomial, we need to substitute for variable x by x^i where i is the particular power of companion matrix of f .

5.3 Finding of H in algorithm for preprocessing

Finding of the accurate value of parameter H is crucial for the algorithm for preprocessing since the value of D_{min} depends on it. This search is realized in function `computeH` which takes only 1 parameter - the alphabet A .

If the base β of numeration system (β, A) is real, we can find the value of H simply by appointing into the following equation:

$$H = \frac{\max(A)}{|\beta| - 1}. \quad (5.1)$$

If β is complex Pisot number, we will search in a loop for the minimal degree i of base β such that β^i is a real number. Then we generate the set \tilde{X}_i and find an element a which is maximal in absolute value. Then we obtain the value of H as

$$H = \frac{|a|}{|\beta^i| - 1}.$$

If there is no such i , the program will search up to the preselected upper bound, for example 10, and then the parameter H will be set as (5.1).

The specific implementation of this algorithm is using the function `getSpectrumForm`.

5.4 Preprocessing

The reason why we need to perform preprocessing for on-line division is in detail explained in Chapter 4. Implementation of the algorithm from Section 4.3.1 is organised in one script and is composed from several functions. The input of the program are two parameters

1. the minimal polynomial of the base β ,
2. the alphabet of digits A .

The output of the program is the list of rewriting rules and the value of the parameter D_{min} .

Before we run the program, we have to verify step 2 of the algorithm for preprocessing which is ensured by the function `kontrolaNetrivReprezentace`. Otherwise the program would give us false results. Then the function `computeH` is run in order to obtain the most accurate value of parameter H .

Then we can run the program to find particular lists of rewriting rules and corresponding parameter D_{min} . This search is realized in function `getSpectrumForUptoM` with 4 parameters:

- list where the spectrum will be stored,
- alphabet A ,
- value of H ,
- maximal degree of the base β .

We use auxiliary sets to ensure faster computation. The set *spectrum* contains every element of the spectrum. The set *mnozinaNezmensitelnych* contains only those elements which does not have a prefix which can be rewritten in a shorter way. The set L contains those elements of the spectrum which can be rewritten in a shorter way. To every one of these sets there exists one additional set which contains the particular (β, A) -representations of corresponding elements.

This program has several deviations from the program for generating of the spectrum. When trying to add a new element $x = x_1x_2 \cdots x_n$ into the spectrum, first we need to check if the value of this element is not the same as the value of any other element of the spectrum. If there is an element $z = z_1z_2 \cdots z_m$ with the same value, we check if $n > m$ or if $z_1 = 0$. Depending on the result, we choose the appropriate path according to the following:

1. If there is an element with the same value:
 - (a) If $n > m$ or $z_1 = 0$: We only add $x_1x_2 \cdots x_n \rightarrow z_1z_2 \cdots z_m$ to the set of rewriting rules.
 - (b) Else: We do not add this element anywhere.
2. Else: We add x into the sets *spectrum* and *mnozinaNezmensitelnych*.

In the end of every iteration, we calculate parameter m as the maximal degree of β and parameter D_{min} from (4.5). Then we use another auxiliary set to ensure that the list of rewriting rules is minimal. This is achieved by checking if the absolute value of every rewriting rule while adding absolute value of the smallest string possible in (β, A) numeration system ($\frac{\min(A)}{\beta-1}$) is smaller than particular value of D_{min} . The description of the functions used in the program follows.

vytvoritRetezec is a function of 2 parameters k, i where k is the index of element from which we create a new element by

$$x = S[k] \cdot \beta + i.$$

vytvoritPravidlo takes two representations of the same value *first* and *second* and creates a rewriting rule of the form

$$first \rightarrow second.$$

zapsatPravidlo uses the previous function to add a new rule and its value into the corresponding sets.

pridatNovyPrvek adds a new element into the sets.

kontrolaNovehoPrvku is a function which checks if the new element can not be rewritten by already existing rules or by extending the rules by zeros.

`kontrolaStarychPrvku` uses the previous function to check if every element of the set *mnozinaNezmensitelnych* can not be shortened. It is used in the end when we want to calculate the value of the parameter D_{min} .

`kontrolaNetrivReprezentace` checks if the numeration system (β, A) does have non-trivial representation of 0.

`getDforNetrivRepres` computes the parameter D_{min} in the case if the previous function results in false.

These programs are applied in Chapter 6 to several cases of numeration systems. We created pictures of the spectrum, found the minimal alphabet for neighbour-free parallel addition and performed preprocessing for on-line division. While performing preprocessing we can just take the minimal list of rewriting rules which is the first non-empty list and corresponding parameter D_{min} or we can take greater list which will result in greater D_{min} . Greater D_{min} is more convenient for the algorithm for the actual on-line division, but the corresponding list of rewriting rules can be much greater which means that preprocessing of the divisor can take much longer. This is illustrated on examples.

Chapter 6

Fast on-line division for selected numeration systems

Let us discuss several cases of numeration systems where the base β is either a quadratic Pisot unit, d -Bonacci, Penny or Eisenstein number. For each chosen base we discuss the choice of the alphabet in order that the necessary prerequisites for fast algorithms are satisfied. Then we perform the preprocessing.

6.1 Quadratic Pisot unit

First let us consider the case when the base β is a quadratic Pisot unit, i.e. has minimal polynomial $x^2 - ax - b$ where $b = \pm 1$. From Example 1.3 we can see that β is a Pisot unit only for $a \geq |b - 1|$. We will consider only symmetric alphabets A of consecutive integers.

6.1.1 Overview of alphabets

In order that all real numbers have (β, A) representation, the alphabet must contain at least $\lceil \beta \rceil$ digits, i.e.

$$\begin{aligned} \#A &\geq a + 1 && \text{for } b = 1 \\ \#A &\geq a && \text{for } b = -1. \end{aligned}$$

By Theorem 4.12, the same cardinality of the alphabet A is sufficient for the possibility of on-line division.

However, if we want the on-line algorithm to have linear complexity, we also need parallel addition. By [8], it requires alphabet of the size at least a for $b = -1$, but at least $a + 2$ for $b = 1$.

Consequently, that a symmetric alphabet allowing linear time on-line division is given as follows.

Proposition 6.1. *Let $\beta > 1$ be a quadratic Pisot unit with minimal polynomial $x^2 - ax - b$, $b = \pm 1$ and let $A = \{-m, \dots, 0, \dots, m\} \subset \mathbb{Z}$. If linear time on-line division is possible in the (β, A) numeration system, then*

$$\begin{aligned} A_{\text{odd}} &= \left\{ -\frac{a+b}{2}, \dots, 0, \dots, \frac{a+b}{2} \right\} && \text{for } a \text{ odd.} \\ A_{\text{even}} &= \left\{ -\frac{a+b+1}{2}, \dots, 0, \dots, \frac{a+b+1}{2} \right\} && \text{for } a \text{ even.} \end{aligned}$$

6.1.2 Preprocessing

First let us assume that a is an odd number. To verify if the numeration system (β, A_{odd}) allows preprocessing, we need to verify the existence of a non-trivial 0 representation. From Proposition 2.8 zero has a non-trivial (β, A_{odd}) -representation if and only if $\beta \leq \frac{a+b}{2} + 1$. Using the fact that β is a specific root of a quadratic polynomial:

$$\beta = \frac{a + \sqrt{a^2 + 4b}}{2} \stackrel{?}{\leq} \frac{a+b}{2} + 1$$

$$a^2 \leq 5.$$

Thus for a an odd number, 0 has a non-trivial representation for $a = 1$ and $b = 1$. This special case will be discussed in Section 6.2. For $a \geq 3$, zero has only trivial representation in the numeration system (β, A_{odd}) . Therefore the list of rewriting rules from the preprocessing is empty and we can calculate the value of D_{\min} directly [6]:

$$D_{\min} = \frac{\beta - 1 + \min\{A_{\text{odd}}\}}{\beta(\beta - 1)} = \frac{\frac{a + \sqrt{a^2 + 4b}}{2} - 1 - \frac{a+b}{2}}{\frac{a + \sqrt{a^2 + 4b}}{2} \left(\frac{a + \sqrt{a^2 + 4b}}{2} - 1 \right)}$$

$$D_{\min} = \frac{\sqrt{a^2 + 4b} - 2 - b}{2 \left(a^2 + \sqrt{a^2 + 4b}(a - 1) + 2b - a \right)}. \quad (6.1)$$

Now let us assume that a is an even number. For the alphabet A_{even} , we also have to verify the existence of a non-trivial (β, A_{even}) -representation of 0:

$$\frac{a + \sqrt{a^2 + 4b}}{2} \stackrel{?}{\leq} \frac{a+b+1}{2} + 1$$

$$a^2 \leq 10 + 2b.$$

This condition is satisfied for a an even number only if $a = 2$ and $b = 1$. We will focus on this case in Section 6.3. For $a \geq 4$ even there is also only trivial 0 representation and the list of rewriting rules is empty. The value of the parameter D_{\min} is slightly different for a an odd number:

$$D_{\min} = \frac{\beta - 1 - \min\{A_{\text{even}}\}}{\beta(\beta - 1)} = \frac{\frac{a + \sqrt{a^2 + 4b}}{2} - 1 - \frac{a+b+1}{2}}{\frac{a + \sqrt{a^2 + 4b}}{2} \left(\frac{a + \sqrt{a^2 + 4b}}{2} - 1 \right)}$$

$$D_{\min} = \frac{\sqrt{a^2 + 4b} - 3 - b}{2 \left(a^2 + \sqrt{a^2 + 4b}(a - 1) + 2b - a \right)}. \quad (6.2)$$

To summarize, for a numeration system with the base β a Pisot unit with minimal polynomial $x^2 - ax - b$ where $b = \pm 1$ and the alphabets A_{odd} and A_{even} , for any a except for the two cases which will be discussed in detail in Sections 6.2 and 6.3, the fast on-line division can be performed with an empty list of rewriting rules and the value of the parameter D_{\min} given by (6.1) and (6.2), respectively.

6.2 Golden ratio

Let us focus on the case with base $\beta = \tau = \frac{1+\sqrt{5}}{2}$ the golden ratio, a root of the polynomial $x^2 - x - 1$.

6.2.1 Overview of the alphabets

First we will consider the alphabet $A = \{-1, 0, 1\}$. The number of elements of A ensures that every number in \mathbb{C} has (β, A) -representation according to Theorem 2.10. A is also a minimal symmetric integer alphabet for parallel addition and on-line division by Theorems 3.6 and 4.12.

In this numeration system, 0 has a non-trivial representation. Therefore we can use our program to compute the value of D_{min} and the list of rewriting rules. Preprocessing for this case was already completed in [6]. For the alphabet A the numeration system (β, A) does not satisfy WRZ, so the neighbour-free parallel addition algorithms can not be performed.

In order to compute addition in parallel by a neighbour-free algorithm, we will use the construction of the polynomial W from the proof of Proposition 3.9.

Set $n_0 = 2$. We denote the companion matrix $M = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$ of the polynomial $x^2 - x - 1$ and compute the determinant

$$\det(M^2 - xI) = \begin{vmatrix} 2-x & 1 \\ 1 & 1-x \end{vmatrix} = x^2 - 3x + 1 = G_2(x).$$

The polynomial W is of the form

$$W(x) = G_2(x^2) = x^4 - 3x^2 + 1.$$

Polynomial W obviously is WRZ and the minimal alphabet for neighbour-free parallel addition A_{nf} is given by Theorem 3.8:

$$a = \left\lceil \frac{B-1}{2} \right\rceil + M = 3.$$

So we have the alphabet $A_{nf} = \{-3, -2, -1, 0, 1, 2, 3\}$.

6.2.2 Preprocessing with alphabet A

In this section results from [6] will be cited. Let us consider the base τ and the alphabet $A = \{-1, 0, 1\}$ and the minimal list of rewriting rules

$$\mathcal{L}_0 : \quad 10\bar{1} \rightarrow 010, \quad 1\bar{1}0 \rightarrow 001, \quad 1\bar{1}\bar{1} \rightarrow 000.$$

In [6] it is shown that if no rule from \mathcal{L}_0 can be applied, then $D_{min} = \frac{1}{\tau^5} = 0.090169 \dots$.

In order to obtain the best possible value of parameter D_{min} the list of rewriting rules have to be extended to the infinite list of rules \mathcal{L} :

$$\begin{aligned}
(1\bar{1})^k 0 &\rightarrow 00(10)^{k-1} 1 \text{ for } k \geq 1, \\
(1\bar{1})^k \bar{1} &\rightarrow 00(10)^{k-1} 0 \text{ for } k \geq 1, \\
(1\bar{1})^k 10\bar{1} &\rightarrow 01(00)^k 0 \text{ for } k \geq 0, \\
(1\bar{1})^k 100 &\rightarrow 01(00)^k 1 \text{ for } k \geq 0, \\
(1\bar{1})^k 11 &\rightarrow 01(00)^{k-1} 10 \text{ for } k \geq 1, \\
10\bar{1}^k 0 &\rightarrow 0^{k+1} 11 \text{ for } k \geq 0, \\
10\bar{1}^k 1 &\rightarrow 0^k 101 \text{ for } k \geq 1, \\
100 &\rightarrow 011.
\end{aligned}$$

By considering the list \mathcal{L} , we obtain the value $D_{min} = \frac{1}{\tau^2} = 0.381966 \dots$.

6.2.3 Preprocessing with alphabet A_{nf}

Consider the base τ and the alphabet A_{nf} which satisfy WRZ and therefore can be used for neighbour-free parallel addition. We use our program for preprocessing for on-line division. The minimal list of rewriting rules contains 142 rules that start with positive number (in total there are 284 rules) and $D_{min} = \frac{7\tau-11}{\tau^8(\tau-1)} = 0.011236 \dots$ which is obtained for the strings of length $m = 8$.

6.3 Silver ratio

Let us consider the base $\beta = 1 + \sqrt{2}$ a root of the polynomial $x^2 - 2x - 1$.

6.3.1 Overview of the alphabets

First we discuss the numeration system (β, A) where A is a symmetric integer alphabet $A = \{-2, -1, 0, 1, 2\}$. The alphabet A enables us to represent every number in \mathbb{C} and to perform on-line division according to Theorem 4.12. A is minimal concerning the computability of the parallel addition, but it does not enable to use neighbour-free algorithms for parallel addition.

In order to use these algorithms, we need to realize the construction of polynomial W . In this case we also set $n_0 = 2$, denote the companion matrix of the polynomial $x^2 - 2x - 1$ by $M = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$ and compute the determinant

$$\det(M^2 - xI) = \begin{vmatrix} 5-x & 2 \\ 2 & 1-x \end{vmatrix} = x^2 - 6x + 1 = G_2(x).$$

We get the polynomial W in the form

$$W(x) = G_2(x^2) = x^4 - 6x^2 + 1.$$

From the coefficients of W we obtain the minimal alphabet for neighbour-free parallel addition $A_{nf} = \{-5, \dots, 0, \dots, 5\}$.

6.3.2 Preprocessing with alphabet A

Now we use our program for preprocessing for on-line division on numeration system (β, A) . For this setting, we obtain the parameter $D_{min} = \frac{4-\beta}{\beta^3} = 0.112698\dots$ and the list of rewriting rules containing 12 rules in total for the strings of length $m = 3$. Let us state only the rules starting on positive number since the rules are symmetric:

$$\begin{aligned} 1\bar{1}x &\rightarrow 01(x+1) && \text{for } x \leq -1, \\ 1\bar{2}x &\rightarrow 00(x+1) && \text{for } x \leq 1, \end{aligned}$$

and the value of D_{min} is obtained from (4.5), specifically

$$D_{min} = \frac{|\varepsilon|}{\beta^m} = \frac{|1\bar{2}2| - H}{\beta^3} = 0.112698\dots$$

where $H = \beta - 1 = 1.414213\dots$.

In order to compute on-line division with greater parameter D_{min} a list of rewriting rules containing at least 28 identities in total has to be considered. For example a list containing the following rules (starting on a positive digit):

$$\begin{aligned} 1\bar{1}x &\rightarrow 01(x+1) && \text{for } x \leq 0, \\ 1\bar{2}x &\rightarrow 00(x+1) && \text{for } x \leq 1, \\ 1\bar{2}2x &\rightarrow 011(x-1) && \text{for } -1 \leq x \leq 1, \\ 1\bar{2}2\bar{2}x &\rightarrow 0012(x+2) && \text{for } x \leq 0, \\ 1\bar{2}2\bar{2}1 &\rightarrow 00202 \end{aligned}$$

corresponds to the parameter $D_{min} = 0.161471\dots$ which is obtained for strings of length $m = 5$. We can see that the value of parameter D_{min} is not much greater than for the minimal list of rewriting rules but list of identities contains more than twice as many rules as the minimal list, which is not very convenient for fast on-line division.

6.3.3 Preprocessing with alphabet A_{nf}

By using our program to compute the minimal list of rewriting rules and the parameter D_{min} on the numerical system (β, A_{nf}) , we obtain the list of rewriting rules containing 212 identities in total and $D_{min} = \frac{7\beta-14}{\beta^6(\beta-1)} = 0.010355\dots$ for strings of length $m = 6$.

To summarize, we can see that for smaller alphabets we cannot compute the parallel addition neighbour-free, but there is much smaller list of rewriting rules and bigger parameter D_{min} than in the second case which has a huge impact on time spent realizing the algorithm for on-line division.

6.4 d -Bonacci numbers

Let us consider the base β a d -Bonacci number, a root of the polynomial of the form $x^d - x^{d-1} - \dots - x - 1$ for $d \geq 2, d \in \mathbb{N}$.

6.4.1 Overview of the alphabets

First we will focus on numeration systems (β, A) where β is d -Bonacci number for some d and $A = \{\bar{1}, 0, 1\}$ is the alphabet. Every number in \mathbb{C} has a (β, A) -representation. Considering the alphabet A , only algorithms for k -block parallel addition can be performed by Theorem 3.6. Also the polynomial $x^d - x^{d-1} - \dots - x - 1$ does not satisfy WRZ defined in Definition 3.7, therefore neighbour-free parallel addition can not be realized. However, we will see that on-line division is possible to be performed.

Since for every d -Bonacci number, $d \geq 2$, their minimal polynomial does not satisfy WRZ, in order to perform a neighbour-free parallel addition, a weak polynomial must be found using the algorithm from the proof of Proposition 3.9.

The greatest digit a of the minimal alphabet $A = \{-a, \dots, 0, \dots, a\}$ for neighbour-free parallel addition where $d \geq 2$ can be seen in Table 6.1. We can see that the preprocessing for on-line division would take much longer for numeration systems with alphabets which allow neighbour-free parallel addition. The benefit of neighbour-free parallel addition would not be useful since the number of rewriting rules is 284 for $d = 2$ and more than 84000 for $d = 3$. For $d > 3$, the program for preprocessing does not even stop since the number of rules is too high.

parameter d	digit a
2	3
3	12
4	7
5	12
6	29
7	72
8	88
9	170
10	289
11	639
12	1345

Table 6.1: The greatest digit a of the alphabet $A = \{-a, \dots, 0, \dots, a\}$ for neighbour-free parallel addition for d -Bonacci numbers as bases of numeration systems.

6.4.2 Preprocessing with alphabet A

Let the alphabet be $A = \{-1, 0, 1\}$. First we have to verify if 0 has non-trivial (β, A) -representation which according to Theorem 4.15 is equivalent to the condition

$$\beta \leq 2.$$

We can use Cauchy's estimate for root localization for the polynomial $x^d - x^{d-1} - \dots - x - 1$, i.e. all roots of the polynomial $P(x) = \sum_{i=0}^n a_i x^i$, $a_n \neq 0 \neq a_0$, satisfy the following condition

$$\frac{|a_0|}{|a_0| + \max\{|a_n|, \dots, |a_1|\}} < |x| < 1 + \frac{\max\{|a_{n-1}|, \dots, |a_0|\}}{|a_n|}.$$

After substitution for coefficients of minimal polynomial we obtain the following estimate for all the roots

$$\frac{1}{2} < |x| < 2.$$

Therefore 0 has non-trivial (β, A) -representation.

We can use our program for preprocessing for on-line division to compute the minimal list of rewriting rules and corresponding parameter D_{min} . For $d \geq 2$ we obtain the following value of the parameter D_{min} and the list of identities \mathcal{L} , $|\mathcal{L}| = 3$ (considering only identities having prefix 1 since the identities with prefix $\bar{1}$ can be obtained by change of signs):

$$D_{min} = \left\lfloor \frac{2\beta - 3}{\beta^{d+1}(\beta - 1)} \right\rfloor$$

$$\mathcal{L} = \left\{ 1(\bar{1})^{d-2}xy \rightarrow (0)^{d-1}(x+1)(y+1) \right\}.$$

where $x, y \in \{-1, 0\}$ and $x + y < 0$.

In order to obtain the greatest value of the parameter D_{min} possible, we need to consider rules of arbitrary length. The following proposition states a partial result.

Proposition 6.2. *Let β be a d -Bonacci number where $d \geq 2, d \in \mathbb{N}$ and let $A = \{-1, 0, 1\}$ be an alphabet. Let the list of rewriting rules contain the following rule of arbitrary length $kd+1, k \in \mathbb{N}$*

$$1a_{1,1} a_{1,2} \cdots a_{1,d-1} 1a_{2,1} a_{2,2} \cdots a_{2,d-1} 1 \cdots 1a_{k,1} a_{k,2} \cdots a_{k,d-1} b \rightarrow \quad (6.3)$$

$$0(a_{1,1} + 1) \cdots (a_{1,d-1} + 1)1(a_{2,1} + 1) \cdots (a_{2,d-1} + 1)1 \cdots (a_{k,d-1} + 1)(b + 1)$$

where $a_{i,j}, b \in \{-1, 0\}$ for $i = 1, \dots, k$ and $j = 1, \dots, d-1$. Then the value of the parameter D_{min} is

$$D_{min} = \left\lfloor 0.(1(\bar{1})^{d-1})^\omega \right\rfloor = \frac{1}{\beta(\beta^d - 1)}.$$

Proof. If the rule (6.3) can not be used for string x , then x must be of the form

$$x = 0.1 a_{1,1} a_{1,2} \cdots a_{1,d-1} 1 a_{2,1} a_{2,2} \cdots a_{2,d-1} 1 \cdots 1 a_{j,1} a_{j,2} \cdots a_{j,l-1} 1 \cdots$$

for some $1 \leq l < d$. Let us denote two strings

$$y = 0.(1(\bar{1})^{d-1})^{j-1} 1(\bar{1})^{l-1} 1\bar{1}^\omega,$$

$$z = 0.(1(\bar{1})^{d-1})^\omega = D_{min}.$$

Clearly $x \geq y$ and $y > z$. It remains to be proven that the value of D_{min} can not be improved considering the list of identities \mathcal{L} .

$$y = 0.(1\bar{1}^{d-1})^{j-1} 1\bar{1}^{l-1} \left[\begin{array}{c} 1 \\ \bar{1} \end{array} \right] \bar{1}^\omega$$

$$z = 0.(1\bar{1}^{d-1})^{j-1} 1\bar{1}^{l-1} \left[\begin{array}{c} 1 \\ \bar{1} \end{array} \right] \bar{1}^{d-l-1} (1\bar{1}^{d-1})^\omega$$

We denote the first index where y and z differ $H = d(j-1) + l + 1$. We study the difference of y and z :

$$y - z = \frac{2}{\beta^H} - \frac{2}{\beta^{H+d-l}} \left(1 + \frac{1}{\beta^d} + \frac{1}{\beta^{2d}} + \cdots \right)$$

$$= \frac{2}{\beta^H} - \frac{2}{\beta^{H+d-l}} \frac{\beta^d}{\beta^d - 1} = 2 \frac{\beta^d - 1 - \beta^l}{\beta^H(\beta^d - 1)}.$$

Since $d > l \geq 1$, $y > z$. For $j \rightarrow +\infty$ the difference $y - z \rightarrow 0$. Therefore with the list of rewriting rules \mathcal{L} the value of parameter D_{min} for preprocessing can not be improved. \square

The previous proposition does not specify if the value D_{min} can be improved for some bigger list of rewriting rules. If we compare for $d = 2$ this value of D_{min} with the value obtained in [6], we can verify that

$$D_{min} = |z| = \frac{1}{\beta(\beta^d - 1)} \stackrel{d:=2}{=} \frac{1}{\beta^2}$$

is the best possible.

Particular values of parameter D_{min} and number of elements of corresponding minimal list of rewriting rules for several settings of parameter d is shown in Table 6.2. Notice that for increasing value of d (i.e. for bases with minimal polynomials of greater degree) parameter D_{min} is decreasing but number of rewriting rules stays the same. Even though the algorithm for on-line division can be realized with less suitable parameter D_{min} for greater d , the corresponding minimal list of rewriting rules remains small which indicates that preprocessing for on-line division will be fast even for greater d .

parameter d	$\#\mathcal{L}$	D_{min}	D_{min}^P
2	6	0.090169...	0.381966...
3	6	0.070646...	0.104109...
4	6	0.034645...	0.040514...
5	6	0.016710...	0.017931...
6	6	0.008138...	0.008414...
7	6	0.004001...	0.004066...
8	6	0.001980...	0.001996...

Table 6.2: Number of elements of the minimal list of rewriting rules \mathcal{L} and corresponding parameter $D_{min} = \frac{\lambda(\beta)}{\beta^{d+1}}$ where $\lambda(\beta) = 2 - \frac{1}{\beta-1}$ for preprocessing for numeration system where the base is a d -Bonacci number and the alphabet is $\{-1, 0, 1\}$. Value of parameter D_{min}^P from Proposition 6.2 is displayed.

6.5 Penney number

Let us consider Penney number as the base $\beta = -1 + i$, a root of the polynomial $x^2 + 2x + 2$.

6.5.1 Overview of the alphabets

We will discuss two alphabets $A_i = \{0, \pm 1, \pm i\}$ and $A_r = \{-2, -1, 0, 1, 2\}$. Each of these alphabets enables us to perform different operations. The fact that in numeration systems (β, A_i) and (β, A_r) every complex number is representable is direct corollary of the following proposition.

Proposition 6.3. *Let $\beta = -1 + i$ and let $A_i = \{\pm 1, \pm i, 0\}$ and $A_r = \{-2, -1, 0, 1, 2\}$ be alphabets. Then*

$$X^{A_i}(\beta) = X^{A_r}(\beta) = \mathbb{Z}[i].$$

Proof. From [12] we know that $X^{\{0,1\}} = \mathbb{Z}[i]$. Let us take $\sum_{i \geq 0} x_i \beta^i$ with $x_i \in \{\pm 1, \pm i, 0\}$. Then

$$\sum_{i \geq 0} x_i \beta^i = \sum_{i \geq 1} a_i \beta^i \pm i \sum_{i \geq 1} b_i \beta^i \in \mathbb{Z}[i]$$

where $a_i, b_i \in \mathbb{Z}[i]$. Therefore $X^A(\beta) \subset X^{\{0,1\}}(\beta) = \mathbb{Z}[i]$. The opposite inclusion trivially applies.

Proof of equivalence with $X^{\hat{A}}(\beta)$ would be similar. \square

In Figure 6.1 we can see figures of subsets of the spectrum $X^{A_i}(\beta) = X^{A_r}(\beta)$ which display that the previous proposition stands and which also show the difference between generating these two spectra. The final set is equal to $\mathbb{Z}[i]$ which can be seen in the third figure. Both alphabets A_i and A_r allows us to perform algorithms for on-line division and parallel addition according to Theorems 4.12 and 3.6. The polynomial $x^2 + 2x + 2$ does not satisfy WRZ, therefore the neighbour-free parallel addition can not be realized.

6.5.2 Preprocessing with alphabet A_i

Let us consider the alphabet $A = \{0, \pm 1, \pm i\}$. Since the numerical system (β, A) is redundant, clearly 0 has non-trivial (β, A) -representation. Thus we can use our program to compute the minimal list of rewriting rules L . The base β is complex, therefore we have to use our program to finding the accurate value of parameter H :

$$H = \frac{|1\bar{1}\bar{i}i|}{|\beta|^4 - 1} = |2\beta + 3| = 2.236067 \dots$$

Using the previous value of H , we obtain the minimal list of rewriting rules which contains 60 identities and the parameter $D_{min} = \frac{|2\beta+4|-|2\beta+3|}{|\beta|^4} = 0.148089 \dots$. In Table 6.3 we can see identities from the minimal list starting on 1. All other identities can be obtained by multiplying by -1 and i .

$11 \rightarrow 0i$	$1\bar{i} \rightarrow 0\bar{1}$	$101 \rightarrow 0\bar{1}\bar{i}$	$10\bar{1} \rightarrow 0i\bar{i}$	$10i \rightarrow 00\bar{i}$	$1\bar{1}i \rightarrow 0\bar{1}\bar{i}$
$1ii \rightarrow 0i\bar{i}$	$1001 \rightarrow 0i\bar{i}i$	$100\bar{1} \rightarrow 00\bar{i}i$	$100i \rightarrow 0\bar{1}\bar{i}1$	$100\bar{i} \rightarrow 00\bar{i}1$	$10\bar{i}1 \rightarrow 0\bar{1}\bar{i}1$
$10\bar{i}i \rightarrow 0i\bar{i}i$	$1\bar{1}0\bar{i} \rightarrow 0\bar{1}\bar{i}1$	$1i0\bar{1} \rightarrow 0i\bar{i}i$			

Table 6.3: Minimal list of rewriting rules for preprocessing for numeration system with Penney number as the base and the alphabet $A_i = \{0, \pm 1, \pm i\}$. Only rules starting on 1 are displayed since the rest of the identities can be obtained by multiplying by -1 and i .

6.5.3 Preprocessing with alphabet A_r

Let us perform preprocessing for the numeration system (β, A_r) where $A_r = \{-2, -1, 0, 1, 2\}$. We also start by finding the accurate value of parameter H :

$$H = \frac{|2\bar{1}22|}{|\beta|^4 - 1} = \frac{|8\beta + 12|}{3} = 2.981423 \dots$$

The program then gives us the minimal list of rewriting rules containing 66 identities in total and the value of parameter $D_{min} = \frac{|10\bar{1}1| - \frac{8\beta+12}{3}}{|\beta|^4} = \frac{3|\beta+5| - |8\beta+12|}{12} = 0.285420 \dots$.

In Table 6.4 we can see the minimal list of rewriting rules for preprocessing for numeration system (β, A_r) .

$100 \rightarrow 0\bar{2}\bar{2}$	$110 \rightarrow 0\bar{1}\bar{2}$	$111 \rightarrow 0\bar{1}\bar{1}$	$112 \rightarrow 0\bar{1}0$	$120 \rightarrow 00\bar{2}$	$121 \rightarrow 00\bar{1}$
$122 \rightarrow 000$	$222 \rightarrow 0\bar{2}\bar{2}$	$101 \rightarrow 0\bar{2}\bar{1}$	$102 \rightarrow 0\bar{2}0$	$10\bar{1}0 \rightarrow 0\bar{1}\bar{1}\bar{2}$	$10\bar{1}\bar{1} \rightarrow 0\bar{1}\bar{1}\bar{1}$
$10\bar{1}\bar{2} \rightarrow 0012$	$10\bar{2}0 \rightarrow 0\bar{1}\bar{2}\bar{2}$	$10\bar{2}\bar{1} \rightarrow 0\bar{1}\bar{2}\bar{1}$	$10\bar{2}\bar{2} \rightarrow 0002$	$11\bar{1}0 \rightarrow 00\bar{1}\bar{2}$	$11\bar{1}\bar{1} \rightarrow 00\bar{1}\bar{1}$
$11\bar{1}\bar{2} \rightarrow 00\bar{1}0$	$11\bar{2}0 \rightarrow 00\bar{2}\bar{2}$	$11\bar{2}\bar{1} \rightarrow 00\bar{2}\bar{1}$	$11\bar{2}\bar{2} \rightarrow 00\bar{2}0$	$12\bar{1}\bar{1} \rightarrow 01\bar{1}\bar{1}$	$12\bar{1}\bar{2} \rightarrow 01\bar{1}0$
$1\bar{1}0\bar{1} \rightarrow 0\bar{2}01$	$1\bar{1}0\bar{2} \rightarrow 0\bar{1}\bar{2}\bar{2}$	$1\bar{1}\bar{1}\bar{1} \rightarrow 0\bar{2}\bar{1}\bar{1}$	$1\bar{1}\bar{1}\bar{2} \rightarrow 0\bar{1}\bar{1}\bar{2}$	$1\bar{1}\bar{2}\bar{1} \rightarrow 0\bar{2}\bar{2}\bar{1}$	$1\bar{1}\bar{2}\bar{2} \rightarrow 0\bar{1}\bar{0}\bar{2}$
$21\bar{1}\bar{2} \rightarrow 0\bar{1}\bar{1}\bar{2}$	$21\bar{2}\bar{2} \rightarrow 0\bar{1}\bar{2}\bar{2}$	$22\bar{1}\bar{2} \rightarrow 00\bar{1}\bar{2}$			

Table 6.4: Minimal list of rewriting rules for preprocessing for numeration system with Penney number as the base and the alphabet $A_r = \{-2, -1, 0, 1, 2\}$. Identities starting on a positive digit are displayed.

We can see that both alphabets A_i and A_r are convenient for on-line division in different aspects. The minimal list of rewriting rules using alphabet A_i contains less identities than in the second case (which means that preprocessing of the divisor is going to be faster) but the parameter D_{min} is also lower which on the other hand is not that convenient.

6.6 Eisenstein number

Set the base $\beta = -1 + \omega$ an Eisenstein number, where $\omega = \exp \frac{2\pi i}{3}$ is the third root of unity, a root of the polynomial $x^2 + 3x + 3$.

6.6.1 Overview of the alphabets

In order to represent every number in \mathbb{C} in numeration system with the base $\beta = -1 + \omega$, the alphabet $\{0, 1, 2\}$ will suffice.

The alphabet $\{-2, -1, 0, 1, 2\}$ will enable us to perform 3-block parallel addition. In order to use 1-block algorithms we would have to compute parallel addition using the alphabet $\{-3, \dots, 0, \dots, 3\}$

According to Theorem 4.12 cardinality of the alphabet A has to satisfy condition $\#A > |\beta| = 6$ to have OL property (i.e. to enable on-line division). Therefore alphabet $\{-3, \dots, 0, \dots, 3\}$ is the minimal integer alphabet for on-line division.

If we consider complex alphabet $\{0, \pm 1, \pm \omega, \pm \omega^2\}$, we can also represent every number in \mathbb{C} and perform parallel addition. In [5] it was proven that the numeration system with alphabet A_i satisfy OL property.

We will consider two alphabets $A_i = \{0, \pm 1, \pm \omega, \pm \omega^2\}$ and $A_r = \{-3, -2, -1, 0, 1, 2, 3\}$ for preprocessing for on-line division. First let us prove the following proposition.

Proposition 6.4. *Let $\beta = -1 + \omega$ be a Eisenstein number and let $A_i = \{\pm 1, \pm \omega, \pm \omega^2, 0\}$ and $A_r = \{-3, \dots, 3\}$ be alphabets. Then*

$$X^{A_i}(\beta) = X^{A_r}(\beta) = \mathbb{Z}[\omega].$$

Proof. From [12] we know that

$$X^{\{0,1,2\}}(\beta) = \mathcal{O}_{\mathbb{Q}(\sqrt{-3})} = \{a + b\frac{1+i\sqrt{3}}{2} : a, b \in \mathbb{Z}\} = \mathbb{Z}[\omega].$$

Let us focus on $X^{A_i}(\beta) = X^{\{0,1,2\}}$. Clearly $X^{A_i}(\beta) \subset \mathbb{Z}[\omega]$. Let $x = \sum_{j \geq 0} a_j \beta^j$ where $a_j \in \{0, 1, 2\}$. In order to prove $X^{\{0,1,2\}}(\beta) \subset X^{A_i}(\beta)$ we need to find $b_j \in A_i = \{\pm 1, \pm \omega, \pm \omega^2, 0\}$

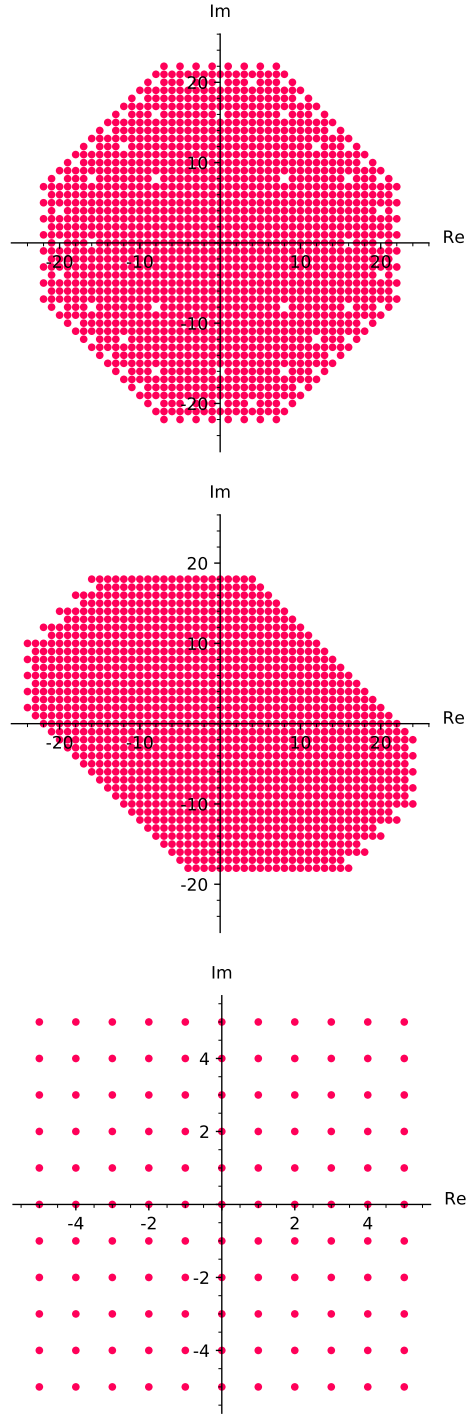


Figure 6.1: The spectrum $X^{A_1}(\beta)$ and the spectrum $X^{A_2}(\beta)$ for the base $\beta = -1 + i$ a Penney number and $A_1 = \{0, \pm 1, \pm i\}$ (i.e. the redundant Penney system) and $A_2 = \{-2, -1, 0, 1, 2\}$. In particular, sets \tilde{X}_7 and \tilde{X}_6 are displayed. The third figure is part of resulting spectrum which corresponds to $\mathbb{Z}[i]$.

for $j \geq 0$ such that $x = \sum_{j \geq 0} b_j \beta^j$. It will suffice to rewrite all the 2 in the string $a_k a_{k-1} a_{k-2} \dots$. Two identities will be used in order to rewrite every 2 in the string into the alphabet A_i :

$$\begin{aligned} 2 &= \beta^2(-\omega^2), \\ -\omega^2 + 1 &= \omega + 2. \end{aligned}$$

We will presume that the first occurrence of number 2 is on position j , i.e. $a_j = 2$, and we find minimal index $l > 0$ such that $a_{j+2l} = 0$. Then the string will be rewritten according to the following rule using the identities stated above:

$$\begin{aligned} b_j &= 0, \\ b_{j+2m} &= \begin{cases} -\omega^2 & \text{if } a_{j+2m} = 2 \text{ and } m < l, \\ \omega & \text{if } a_{j+2m} = 1 \text{ and } m < l. \end{cases} \\ b_{j+2l} &= -\omega^2. \end{aligned}$$

When we rewrite all indexes according to the previous rule, we repeat the whole algorithm again. The algorithm is final since in every step we rewrite at least one digit 2 and no new digits which need to be rewritten appear. Odd positions have to be handled separately. The inclusion $X^{\{0,1,2\}}(\beta) \subset X^{A_i}(\beta)$ is hereby proven.

The rest of the proof for $X^{A_r}(\beta) = Z[\omega]$ is similar to the proof of Proposition 6.3. \square

6.6.2 Preprocessing with alphabet A_i

Let us realize preprocessing for the numeration system (β, A_i) . Since the numeration system (β, A_i) is redundant, clearly 0 has non-trivial (β, A_i) -representation. Therefore our program can be used for preprocessing. First we use our program to find the accurate value of H which is obtained in the form

$$H = \frac{|1\bar{1}\bar{\omega}\omega\omega^2\bar{\omega}^2|}{|\beta|^6 - 1} = 1.322875 \dots$$

The minimal list of rewriting rules \mathcal{L} contains 12 identities. We state only 2 rules, since the rest of the identities can be obtained by multiplication by $-1, \omega$ or ω^2 using the fact that $\omega^3 = 1$. List of rewriting rules \mathcal{L} is

$$\begin{aligned} 11 &\rightarrow 0\omega \\ 1\bar{\omega} &\rightarrow 0\bar{1} \end{aligned}$$

and corresponding parameter $D_{min} = \frac{|10|-H}{\beta^2} = 0.136391 \dots$

6.6.3 Preprocessing with alphabet A_r

First we let our program to find an accurate value of H in the form:

$$[H = \frac{|2\bar{3}\bar{3}\bar{3}\bar{3}|}{|\beta|^6 - 1} = 2.842773 \dots]$$

The minimal list of rewriting rules containing 84 identities is displayed in Table 6.5 and the corresponding value of parameter D_{min} is $D_{min} = \frac{|1\bar{1}\bar{1}\bar{1}|-H}{|\beta|^4} = 0.168458 \dots$

$110 \rightarrow 0\bar{2}\bar{3}$	$120 \rightarrow 0\bar{1}\bar{3}$	$121 \rightarrow 0\bar{1}\bar{2}$	$122 \rightarrow 0\bar{1}\bar{1}$	$123 \rightarrow 0\bar{1}0$	$132 \rightarrow 00\bar{1}$
$133 \rightarrow 000$	$100 \rightarrow 0\bar{3}\bar{3}$	$101 \rightarrow 0\bar{3}\bar{2}$	$111 \rightarrow 0\bar{2}\bar{2}$	$112 \rightarrow 0\bar{2}\bar{1}$	$113 \rightarrow 0\bar{2}0$
$130 \rightarrow 00\bar{3}$	$131 \rightarrow 00\bar{2}$	$233 \rightarrow 0\bar{3}\bar{3}$	$10\bar{1}0 \rightarrow 0\bar{2}\bar{1}3$	$10\bar{1}\bar{1} \rightarrow 0\bar{2}\bar{1}2$	$10\bar{1}\bar{2} \rightarrow 0\bar{2}\bar{1}1$
$10\bar{1}\bar{3} \rightarrow 0\bar{1}\bar{2}3$	$10\bar{2}0 \rightarrow 0\bar{2}\bar{2}3$	$10\bar{2}\bar{1} \rightarrow 0\bar{2}\bar{2}2$	$10\bar{2}\bar{2} \rightarrow 0\bar{2}\bar{2}1$	$10\bar{2}\bar{3} \rightarrow 0\bar{1}13$	$10\bar{3}0 \rightarrow 0\bar{2}\bar{3}3$
$10\bar{3}\bar{1} \rightarrow 0\bar{2}\bar{3}2$	$10\bar{3}\bar{2} \rightarrow 0\bar{2}\bar{3}1$	$10\bar{3}\bar{3} \rightarrow 0\bar{1}03$	$11\bar{1}0 \rightarrow 0\bar{1}\bar{1}3$	$11\bar{1}\bar{1} \rightarrow 0\bar{1}\bar{1}2$	$11\bar{1}\bar{2} \rightarrow 0\bar{1}\bar{1}1$
$11\bar{1}\bar{3} \rightarrow 00\bar{2}3$	$11\bar{2}0 \rightarrow 0\bar{1}\bar{2}3$	$11\bar{2}\bar{1} \rightarrow 0\bar{1}\bar{2}2$	$11\bar{2}\bar{2} \rightarrow 0\bar{1}\bar{2}1$	$11\bar{2}\bar{3} \rightarrow 0013$	$11\bar{3}\bar{2} \rightarrow 0\bar{1}\bar{3}1$
$11\bar{3}\bar{3} \rightarrow 0003$	$12\bar{1}\bar{1} \rightarrow 00\bar{1}2$	$12\bar{1}\bar{2} \rightarrow 00\bar{1}1$	$12\bar{1}\bar{3} \rightarrow 00\bar{1}0$	$12\bar{2}\bar{3} \rightarrow 00\bar{2}0$	$23\bar{1}\bar{3} \rightarrow 0\bar{1}\bar{1}3$

Table 6.5: Minimal list of rewriting rules for preprocessing for on-line division for numeration system with base Eisenstein number $-1 + \omega$ where $\omega = \exp \frac{2\pi i}{3}$ and alphabet $A_r = \{-3, -2, -1, 0, 1, 2, 3\}$. Identities starting on a positive digit are displayed.

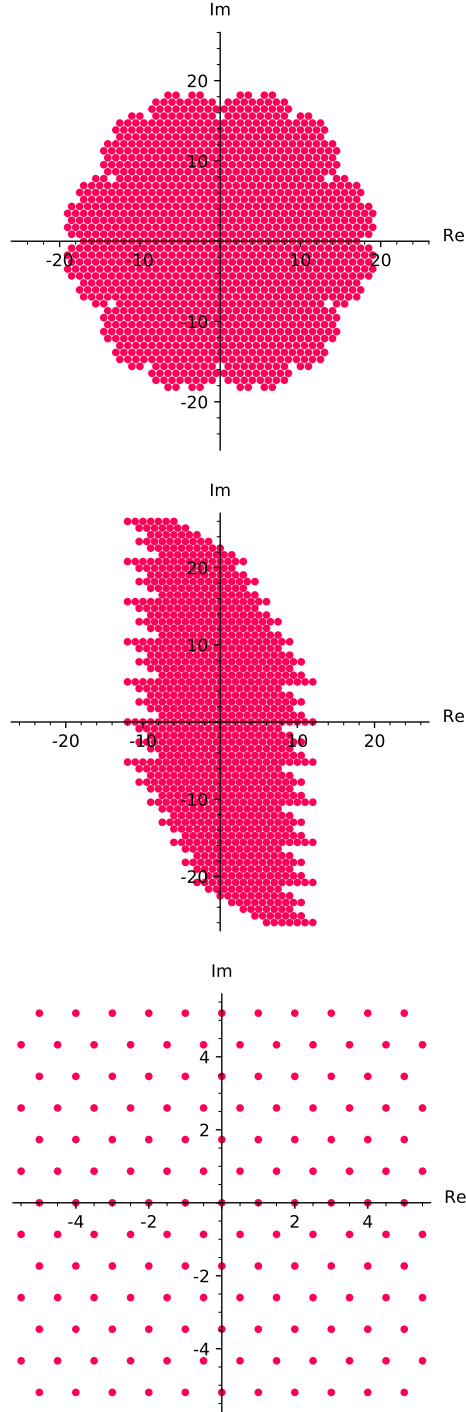


Figure 6.2: The spectrum $X^{A_1}(\beta)$ and the spectrum $X^{A_2}(\beta)$ for the base $\beta = -1 + \omega$ where $\omega = \exp \frac{2\pi i}{3}$ is the third root of unity and the alphabet $A_1 = \{0, \pm 1, \pm \omega, \pm \omega^2\}$ (i.e. the redundant Eisenstein system) and $A_2 = \{-3, -2, -1, 0, 1, 2, 3\}$. Sets \tilde{X}_5 and \tilde{X}_4 are displayed, respectively. The third figure is part of resulting spectrum which corresponds to $\mathbb{Z}[\omega]$.

Conclusions

In this work basic properties of numeration systems with non-integer base and their spectra were studied. Conditions on feasibility of several arithmetic operations for non-standard numeration systems including parallel addition and on-line division were described. Then we focused on preprocessing of divisors for on-line division.

Several computer programs were written in order to generate spectrum of numeration systems, to find size of the minimal alphabet for neighbour-free parallel addition, to perform preprocessing or to find parameters vital for correctness of preprocessing for on-line division.

These programs were then applied to several numeration systems, namely with bases quadratic Pisot number, d -Bonacci number, Penney or Eisenstein number. For each of these numeration systems we discussed minimal alphabets for representing every complex number, for computing neighbour-sensitive and neighbour-free parallel addition and for performing on-line division. Then our program was used to perform preprocessing for on-line division for numeration systems with relevant alphabets.

Results of our observations indicate that using alphabet for neighbour-free parallel addition does not make computing on-line division fast enough since the list of rewriting rules is significantly large, and consequently, preprocessing for on-line division takes much longer. In order to perform on-line division as fast as possible, it seems that finding minimal alphabet enabling parallel addition is more appropriate. For example for numeration systems with base a d -Bonacci number, the parameter D_{min} for minimal alphabet does not differ much from the maximal possible value of D_{min} and the list of rewriting rules still contains only 6 rules. Thus algorithm for on-line division is much faster using small alphabet.

Among the question remaining open is the problem of describing the OL property in a larger family of numeration system with complex base. Except the Penney and Eisenstein number, only in the case of symmetric integer alphabet we can decide whether it satisfies the OL property or not.

Bibliography

- [1] AKIYAMA, S., AND KOMORNIK, V. Discrete spectra and Pisot numbers. *J. Number Theory* 133 (2013), 375–390.
- [2] ERDÖS, P., JOÓ, I., AND KOMORNIK, V. On the sequence of numbers of the form $\varepsilon_0 + \varepsilon_1 q + \dots + \varepsilon_n q^n, \varepsilon_i \in \{0, 1\}$. *Acta Arith.* 83 (1998), 201–210.
- [3] FENG, D.-J. On the topology of polynomials with bounded integer coefficients. *J. Eur. Math. Soc. (JEMS)* 18 (2016), 181–193.
- [4] FROUGNY, C., HELLER, P., PELANTOVÁ, E., AND SVOBODOVÁ, M. k -block parallel addition versus 1-block parallel addition in non-standard numeration systems. *Theoret. Comput. Sci.* 543 (2013), 52–67.
- [5] FROUGNY, C., PAVELKA, M., PELANTOVÁ, E., AND SVOBODOVÁ, M. On-line algorithms for multiplication and division in real and complex numeration systems. *Discrete Math. Theor. Comput. Sci.* 21, 3 (2019), Paper No. 14.
- [6] FROUGNY, C., AND PELANTOVÁ, E. Two applications of the spectrum of numbers. *Acta Math. Hungar.* 156 (2018), 391–407.
- [7] FROUGNY, C., PELANTOVÁ, E., AND SVOBODOVÁ, M. Parallel addition in non-standard numeration systems. *Theoret. Comput. Sci.* 412 (2011), 5714–5727.
- [8] FROUGNY, C., PELANTOVÁ, E., AND SVOBODOVÁ, M. Minimal digit sets for parallel addition in non-standard numeration systems. *J. Integer Seq.* 16 (2013).
- [9] GARSIA, A. M. Arithmetic properties of Bernoulli convolutions. *Trans. Amer. Math. Soc.* 102 (1962), 409–432.
- [10] HARE, K., MASÁKOVÁ, Z., AND VÁVRA, T. On the spectra of Pisot-cyclotomic numbers. *Lett. Math. Phys.* (2016), 1729–1756.
- [11] ITO, S., AND SADAHIRO, T. Beta-expansions with negative bases. *Integers* 9 (2009), 239–259.
- [12] KÁTAI, I., AND KOVÁCS, B. Canonical number systems in imaginary quadratic fields. *Acta Mathematica Academiae Scientiarum Hungaricae* 37 (1981), 159–164.
- [13] LEGERSKÝ, J., AND SVOBODOVÁ, M. Construction of algorithms for parallel addition in expanding bases via extending window method. *Theoret. Comput. Sci.* 795 (2019), 547–569.
- [14] PEDICINI, M. Greedy expansions and sets with deleted digits. *Theor. Comput. Sci.* 332 (2005), 313–336.

- [15] RÉNYI, A. Representations for real numbers and their ergodic properties. *Acta Math. Hungar.* 8 (1957), 477–493.
- [16] TRIVEDI, K., AND ERCEGOVAC, M. On-line algorithms for division and multiplication. *IEEE Trans. Comput. C-26* (1977), 681 – 687.
- [17] ZAĪMI, T., AKIYAMA, S., AND THUSWALDNER, J. Comments on the height reducing property II. *Indag. Math. (N.S.)* 26 (2015), 28–39.