Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering

Representation of Integers in a Linear Recurrent System

Reprezentace přirozených čísel v lineárním rekurentním systému

Bachelor's Degree Project

Author: **Ramina Khusnutdinova**

Supervisor: **Ing. Petr Ambrož, Ph.D.**

Language advisor: **Mgr. Hana Čápová**

Academic year: 2019/2020

České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská

Katedra: matematiky Akademický rok: 2019/2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student:	Ramina Khusnutdinova
Studijní program:	Aplikace přírodních věd
Obor:	Aplikovaná informatika
Název práce (česky):	Reprezentace přirozených čísel v lineárním rekurentním systému
Název práce (anglicky):	Representation of integers in a linear recurrent system

Pokyny pro vypracování:

1. Nastudujte definici reprezentace přirozených čísel v rekurentně zadaných číselných soustavách.
2. Implementujte program hledající všechny rozvoje daného přirozeného čísla v soustavě zadané rekurencí $G_n = 2G_{n-1} + G_{n-2}$.
3. Nastudujte známé vlastnosti funkce $R(n)$, která udává počet rozvojů přirozeného čísla n do Fibonacciho soustavy, zkoumejte analogickou funkci pro číselnou soustavu z bodu 2.
4. Nalezněte vhodné algoritmy pro generování posloupnosti $(R(n))_{n\in\mathbb{N}}$, tyto algoritmy implementujte a odhadněte jejich složitost.
5. Na základě grafického znázorňování zkoumejte vlastnosti grafu funkce $R(n)$.

Doporučená literatura:

1. J. Berstel, An exercise on Fibonacci representations. RAIRO - Theoretical Informatics and Applications 35, 2001, 491–498.
2. M. Edson, L. Q. Zamboni, On representations of positive integers in the Fibonacci base. Theoretical Computer Science 326, 2004, 241–260.
3. F. Maňák, Reprezentace přirozených čísel ve Fibonacciho soustavě. Diplomová práce FJFI ČVUT, Praha, 2005.

Jméno a pracoviště vedoucího bakalářské práce:

Ing. Petr Ambrož, Ph.D.
Katedra matematiky , FJFI ČVUT v Praze, Trojanova 13, 120 00 Praha 2

Jméno a pracoviště konzultanta:

Datum zadání bakalářské práce: 31.10.2019

Datum odevzdání bakalářské práce: 7.7.2020

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 23. října 2019

garant oboru

vedoucí katedry

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta jaderná a fyzikálně inženýrská
-1-

děkan

Acknowledgment:

I wish to express my sincere thanks to my supervisor Ing. Petr Ambrož, Ph.D. for his expert guidance and show my gratitude to Mgr. Hana Čápová for her language assistance. I would also like to thank my friends and family for the unceasing encouragement, support, and attention.

Author's declaration:

I declare that this Bachelor's Degree Project is entirely my own work and I have listed all the used sources in the bibliography.

Prague, July 7, 2020 Ramina Khusnutdinova

Název práce: **Reprezentace přirozených čísel v lineárním rekurentním systému**

Autor: Ramina Khusnutdinova

Obor: Aplikovaná informatika

Druh práce: Bakalářská práce

Vedoucí práce: Ing. Petr Ambrož, Ph.D., Katedra matematiky, FJFI ČVUT v Praze, Trojanova 13

Abstrakt: Hlavním cílem tohoto projektu je studium vlastností pozičního numeračního sytému, jehož báze je posloupnost splňující rekurenci $G_n = 2G_{n-1} + G_{n-2}$. Zajímá nás především funkce $R(n)$ udávající počet reprezentací čísla n ve studovaném systému. Popisujeme algoritmus nalezení hodnoty $R(n)$ pro dané n. Na základě výpočetních výsledků uvádíme několik hypotéz/pozorování o chování funce $R(n)$ a o symetriích jejího grafu.

Klíčová slova: lineární rekurentní systém, počet reprezentací, redundance.

Title: **Representation of integers in a linear recurrent system**

Author: Ramina Khusnutdinova

Abstract: The main aim of this project is to study the properties of the positional numeration system whose base is a sequence satisfying the recurrence $G_n = 2G_{n-1}+G_{n-2}$. We are mainly interested in the function $R(n)$ indicating the number of representations of the number n in the studied system. There is a description of an algorithm for finding the value of $R(n)$ for a given n. Based on the computational results, several hypotheses/observations about behaviour of the function $R(n)$ and about the symmetries of its graph are presented.

Key words: linear recurrent system, number of representations, redundancy.

Contents

Introduction

There are various types of numeration systems, i.e. systems used for representing numbers. Each system has its own field where it can be used. The decimal system, for example, is commonly used in ordinary life. The binary system is a base principle of all computers. The heximal and the hexadecimal systems are used for coding as well, however, all these systems are based on the main idea of being *positional*. Numbers in such numeration systems are represented by the powers of one natural number $b \geqslant 2$, which is called the *base*. A system being positional means that the position of a coefficient in the representation of a number is important – it determines which power of the base is multiplied by this coefficient. Nevertheless, there are systems, which are not based on this principle. Roman numerals, for example, are not the positional ones, as position of a symbol in the representation bears no relation to its value – the value of each symbol is fixed.

Some systems are even more complicated: the Fibonacci numeration system is one of them. Even though this system is close to the binary system and it is positional, numbers in this system can have more than one representation.

Number redundancy (ability to have more than one representation) may be very quaint. Avizienis [1], for instance, used a redundant variant of the decimal system to construct an algorithm for parallel addition.

As the best known redundant system, the Fibonacci linear recurrent system is the one which is well-studied, but there are some variations of it, which are still unresearched. One of these systems, which we study here, is so-called *Raminacci linear recurrent system.* In this project we investigate and present properties and features of this system.

The study is divided into four main chapters: a brief description of the most common numeration systems (1), an acquaintance with the Raminacci numeration system and the Greedy algorithm (2), methods of finding the total number of representations $R(n)$ including the C++ program and results (3), and observations about the symmetry of the graph of $R(n)$ (4).

Chapter 1

Numeration systems

1.1 Common positional numeration systems

Every number can be represented using different systems which are called numeration systems. The numeration system is a technique to represent numbers in various ways for different purposes. The decimal system, for example, is the best known and the most used one. Every number $X \in \mathbb{N}$ in the decimal system is represented by the sum:

$$X = \sum_{i=1}^{k} \alpha_i 10^i, \tag{1.1}$$

for some $k \in \mathbb{N}, \alpha_i \in \{0, \ldots, 9\}$ are called coefficients, and 10 is the base of this system. Binary and hexadecimal numeration systems are the next most common. There are two differences from the decimal system, namely the base and the set of coefficients. In the binary system, $X \in \mathbb{N}$ is represented as

$$X = \sum_{i=1}^{k} \alpha_i 2^i, \tag{1.2}$$

where $\alpha_i \in \{0, 1\}$ and the base is 2. Similarly, X in the hexadecimal system is represented

$$X = \sum_{i=1}^{k} \alpha_i 16^i, \tag{1.3}$$

where $\alpha_i \in \{0, \ldots, 15\}$ and the base is 16.

Remark. Representations in such numeration systems are written with coefficients of the sum only, not using the whole sum. For example, number 1024, as it is normally used, is a representation of number 1024 in the decimal system:

$$4 \times 10^0 + 2 \times 10^1 + 0 \times 10^2 + 1 \times 10^3 = 4 + 20 + 0 + 1000 = 1024.$$

Similarly, 1101011 in the binary system represents number

$$1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 0 \times 2^4 + 1 \times 2^5 + 1 \times 2^6 = \\ 1 + 2 + 0 + 8 + 0 + 32 + 64 = 107.$$

We will use notation $(X)_k$ for the representation of X in the numeration system with the base k. For example, $(107)_2 = 1101011$.

Remark. There is only one representation for each number in such numeration systems due to the natural number base.

1.2 More exotic numeration systems

All three numeration systems mentioned in the previous section have the same type of base $\{B^i : i \geq 0\}$, where $B \in \mathbb{N},\ B \geq 2$. Not only powers of a natural number can be used as a base of the numeration system. In general, any strictly increasing sequence $(B_i)_{i \geq 0}$ can be taken. Then a natural number

X has the representation

$$X = \sum_{i=0}^{k} \alpha_i B_i. \tag{1.4}$$

The best known numeration system using such a base is the Fibonacci numeration system based on the sequence of the Fibonacci numbers

$$F_{n+2} = F_{n+1} + F_n, \tag{1.5}$$

where

$$F_{-1} = 1, \quad F_0 = 1. \tag{1.6}$$

The Fibonacci numbers are used as a base (B_i) for this system. In this particular case $\alpha_i \in \{0, 1\}$. More information about the Fibonacci numeration system may be found in [2, 3, 4].

Such a numeration system is an example of what is called a linear recurrent system, i.e. of a system which base (B_i) is given by the linear recurrence.

Chapter 2

Raminacci numeration system

2.1 Acquaitance with Raminacci system

This bachelor project deals with the linear recurrent system with the base $(G_i)_{i\geq 0}$, where

$$G_i = 2G_{i-1} + G_{i-2}, \quad \forall i \geq 2, \tag{2.1}$$

and

$$G_0 = 1, \ \ G_1 = 1. \tag{2.2}$$

First elements of G_i are given in Table 3.1:

G_0	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	G_9
1	1	3	7	17	41	99	239	577	1393

Table 2.1: First ten elements of the sequence $(G_i)_{i\geq 0}$

Using the previous information from Chapter 1 about bases and coefficients, every $X \in \mathbb{N}$ in the Raminacci system is expressed as

$$X = \sum_{i=1}^{k} \alpha_i G_i, \tag{2.3}$$

for some $k \in \mathbb{N}$.

As in the decimal numeration system, X is represented by the sequence of coefficients of its representation. We write

$$(X)_G = \alpha_k \alpha_{k-1} \cdots \alpha_1. \tag{2.4}$$

Concerning the set of digits, there is a known result [3] stating that the coefficients are $\alpha_i \in \{0, \ldots, M\}$ where

$$M = \sup \left\{ \frac{G_{i+1}}{G_i} \;\middle|\; i \geq 1 \right\}. \tag{2.5}$$

For our numeration system, using (2.1) we get

$$\frac{G_{i+1}}{G_i} = \frac{2G_i + G_{i-1}}{G_i} = \frac{2 + \frac{G_{i-1}}{G_i}}{1}, \tag{2.6}$$

and thus

$$M = \sup \left\{ 2 + \frac{G_{i-1}}{G_i} \;\middle|\; i \geq 1 \right\}. \tag{2.7}$$

Since $\frac{G_{i-1}}{G_i} < 1$, it is obvious that $\sup M = 2$. Thus, the alphabet of the Raminacci linear recurrent system consists of three numbers: $A = \{0, 1, 2\}$.

2.2 Greedy algorithm for Raminacci system

To find a representation of a number in the Raminacci linear recurrent system an algorithm was created. The following steps show the so-called Greedy algorithm for finding a representation of a natural number X in the Raminacci linear recurrent system, called *greedy representation*.

Algoritmus 1: Greedy representation in Raminacci system

input : An integer $X \geq 1$

output: Representation $(X)_G = \alpha_k \cdots \alpha_1$

Find $k \in \mathbb{N}$ such that $G_k \leq X < G_{k+1}$

$\alpha_k \leftarrow \left\lfloor \frac{X}{G_k} \right\rfloor$

$r_k \leftarrow X - \alpha_k G_k$

for $i \leftarrow k-1$ **to** 1 **do**

    $\alpha_i \leftarrow \left\lfloor \frac{r_{i+1}}{G_i} \right\rfloor$

    $r_i \leftarrow r_{i+1} - \alpha_i G_i$

end

To explain the principle of this algorithm in more detail number 90 will be used as an example. In the first step two numbers closest to the X from Raminacci numbers are found. For this particular number we get $G_k = 41, G_{k+1} = 99$, where coefficient $k = 5$.

In the next step coefficient α_5 is obtained as the result of integer division of X by G_k. For number 90 $\alpha_5 = 2$. Next, the residue r_5 is found, $r_5 = 8$.

The rest of the coefficients are gradually calculated in the `for` cycle:

n	α_n	r_n
4	0	8
3	1	1
2	0	1
1	1	0

After all the steps are done, number 90 can be rewritten as the sum

$$90 = 2 \times 41 + 1 \times 7 + 1 \times 1, \tag{2.8}$$

and the greedy representation for this number is

$$(90)_G = 20101. \tag{2.9}$$

The next section explains how and why numbers in the Raminacci system can have more than one representation.

2.3 Other representations

The basic difference of the Raminacci and other linear recurrent systems on the one hand, from the numeration systems like decimal on the other hand is the possibility of existence of more than one representation for the number given. As an example the number from Section 2.2 can be taken. With a close look at (2.8) we can notice that such a sum is not the only way to represent this number. For example, another way is

$$90 = 41 + 2 \times 17 + 2 \times 7 + 1 \tag{2.10}$$

and the representation for this is

$$(90)_G = 12201. \tag{2.11}$$

Both representations in (2.9) and (2.11) has the same numerical value and represent the same number. If we compare them, we can notice that a block of numbers '201' in (2.9) was replaced by '122' in (2.11). However, the value of the representation did not change. This is due to the fact that the base of our system fulfils (2.1). The following notation is introduced for the representation of such cases.

Let $\alpha_k\alpha_{k-1}\cdots\alpha_1$ and $\beta_l\beta_{l-1}\cdots\beta_1$ be two representations in the Raminacci system. We write

$$\alpha_k\cdots\alpha_1 \equiv_G \beta_l\cdots\beta_1 \tag{2.12}$$

if

$$\sum_{i=1}^{k}\alpha_i G_i = \sum_{i=1}^{l}\beta_i G_{i.} \tag{2.13}$$

For example, since $2G_3 + 1G_1 = 15 = 1G_3 + 2G_2 + 2G_1$ we write

$$201 \equiv_G 122. \tag{2.14}$$

From that simple example we can say that due to the linear recurrence and alphabet $A = \{0, 1, 2\}$ the existence of other representations of a number depends on special combinations of three consecutive digits, which will be called *blocks*. For the Raminacci system there are four such blocks:

$$\begin{aligned} 100 &\equiv_G 021 \\ 101 &\equiv_G 022 \\ 200 &\equiv_G 121 \\ 201 &\equiv_G 122. \end{aligned} \tag{2.15}$$

The representation (2.9) can now be tested for all these four blocks and some more representations can be found. Our representation has two blocks from the list above: '201' and '101'. If every combination of replacements of blocks is used, we will get three more representations:

$$\begin{aligned} 20101 &\equiv_G 12201 \\ 20101 &\equiv_G 20022 \\ 20101 &\equiv_G 12122. \end{aligned}$$

The representation found by the greedy algorithm – greedy representation – has a prominent position among all representations of a given $X \in \mathbb{N}$. It is the greatest one in the so-called *radix order*.

Definition 2.1. Let $x = x_n \cdots x_1$ and $y = y_n \cdots y_1$ be two strings (sequences) of symbols in a totally ordered alphabet. We say that x is bigger than y in a radix order $x >_{rad} y$ if one of the following possibilities occurs:

- $n > m$, i.e. x is longer than y;
- $n = m$, and there is an index $r \in \mathbb{N}$ such that $x_r > y_r$ and $x_i = y_i$ for every $r < i \leq n$.

The radix order is very similar to the lexicographical order, i.e. alphabetical order, which is used in dictionaries. These two orders coincide for strings of the same lengths, but for the strings of different length they differ – in the radix order the length of a string is more important.

Example. *Let us have an alphabet* $A = \{0, 1, 2\}$ *with a classical order* $0 < 1 < 2$. *Then*

$$21 <_{rad} 102 <_{rad} 110.$$

In the lexicographical order, the second inequality is the same, but 21 is bigger than both of the strings of length three.

The radix order corresponds to the order of numbers' representations by value.

Theorem 2.2. *Let* $(x)_G = x_n \cdots x_1$ *be a greedy representation of a number* $x \in \mathbb{N}$ *in the Raminacci numeration system. Then*

1. $x_n \neq 0$;
2. $(x)_G$ *is greater in radix order than any representation of* x.

Proof. 1. It follows from Algorithm 1;

2. The greedy representation of x is the longest of all representations of x, it comes from Algorithm 1: the most significant coefficient x_n appears from the division of x with the biggest possible element of the base. At the same time, this digit is the biggest possible, since it corresponds to the biggest multiple of G_n, which is smaller than x:

$$x_n = \max\{k \in \mathbb{N} \mid kG_n < x\}.$$

Let us have another representation of the number x and denote it as $(x)_G = \widetilde{x}_m \cdots \widetilde{x}_1$.

Due to the previous considerations about coefficient x_n, one of the following possibilities occurs:

a) $n > m$;

b) $n = m$ and $x_n > \widetilde{x}_n$;

c) $n = m$ and $x_n = \widetilde{x}_n$;

In cases a) and b) it is obvious from the Definition 2.1 that

$$x_n \cdots x_1 >_{rad} \widetilde{x}_m \cdots \widetilde{x}_1.$$

In the case c) words $x_n \cdots x_1$ and $\widetilde{x}_n \cdots \widetilde{x}_1$ have a common prefix (of at least length 1). From both of the words, we remove the maximal common prefix and receive words $x_l \cdots x_1$ and $\widetilde{x}_l \cdots \widetilde{x}_1$, where $l < n$ and $x_l \neq \widetilde{x}_l$.

Since the greedy algorithm in every step uses the biggest possible digit (step 5 in Algorithm 1), $x_l > \widetilde{x}_l$ necessarily holds and

$$x_n \cdots x_1 >_{rad} \widetilde{x}_m \cdots \widetilde{x}_l.$$

□

It may seem quite easy to find the total number of representations. Since all the changeable blocks are known from the greedy representation, the total number of them may be easily found by the formula

$$N = 2^n, \tag{2.16}$$

where n is the number of changeable blocks in the greedy representation.

Unfortunately, such a method cannot be used. Let us take the number 800 to see the problem in more detail. Using the greedy algorithm we find the representation for this number:

$$(800)_G = 10201101. \tag{2.17}$$

As can be seen above, the greedy representation has two changeable blocks: '201' and '101'. Using the formula from (2.16) we can predict the total number of representations, which will be $N = 2^2 = 4$. And here is the list of these representations we got using this method:

$$\begin{gathered} 10122101 \\ 10201022 \\ 10122022. \end{gathered}$$

With a close look at the first and the last representations it can be noticed that a new block '101' appears, which can be changed too, but which was not predicted from the greedy representation. After a change we get two more representations for the number 800:

$$\begin{gathered} 02222101 \\ 02222022. \end{gathered}$$

The total number of representations for the number 800 is 6, even though the result which was given by the formula (2.16) was 4. It is clear now that representations can be created not only from the greedy one, but from each other. For this reason, formula (2.16) can be used only as a lower estimate on the number of representations of the given number.

Chapter 3

Finding the number of representations in Raminacci system

3.1 Introduction to issue

As mentioned in the previous Chapter, every natural number may have more than one representation in the Raminacci linear recurrent system. In this Chapter we investigate the function $R : \mathbb{N} \to \mathbb{N}$ given by the following definition.

Definition 3.1. Let $X \in \mathbb{N}$. Then $R(X)$ is the number of representations of X in the Raminacci numeration system.

Using the greedy algorithm it is possible to find the first, greedy representation, but finding others is more complicated. Nevertheless, there is a way for finding the number of representations using the information and representation properties we already have. This principle is used in the method called *Direct method*.

3.2 Direct method

The direct method of finding more representations in the Raminacci system is based on the greedy representation and four changeable blocks (2.15). The goal is to find all the representations of the number given by replacing the blocks (2.15) with their numerical equivalences. We have already seen that new representations are generated not only from the greedy representation, but from other representations, too.

To create a system which will help us to find other representations, a suitable data structure should be found. The use of this structure would be the preservation of intermediate results, i.e. the creation of the list of all generated representations together with the information about all representations which have been processed (used in the process). The `set` container from the standard template library [6] turned out to be a suitable data structure for this purpose.

`Set` is an associative container, which contains ordered unique objects. There are situations when during the process of finding representations we can generate a representation, which has already been found. The uniqueness of elements in the `set` will manage this problem and will not allow including an object, which already exists on the list, again.

Remark. Theorem 2.2 says that the greedy representation is maximal in radix, but not in the lexicographical order. First of all, in the algorithm, we generate the greedy representation, then we get every other representation by replacing changeable blocks in it. Even if we were to replace the first block at the very beginning and get a shorter representation, we replace the leading digit with 0. In the direct method, every representation we work with is of the same length, and for strings of the same length, the lexicographical and radix orders coincide.

The orderliness of the objects in the container is helpful as well: all the representations are saved as `string` type, `set<string>` is then arranged lexicographically. This is advantageous, since the greedy representation is lexicographically the biggest of all the representations of the given number. With every adjustment on the representation according to (2.15), the new appearing representation is lexicographically smaller, i.e. it is put into the container above the currently processed representation.

The whole process of generating representations is possible to put into one `for` cycle:

```
set<string> m_r;
m_r.insert(greedy(N));
for(set<string>::iterator i = m_r.rbegin();
                    i != m_r.rend(); i++)
        analyze(m_r, *i);
```

where

- the `greedy(N)` function finds (and returns as `string`) the greedy representation of the number `N` (see Algorithm 1);
- the `analyze(set<string> m_r, string rep)` goes through the representation `rep`, and when it hits one of the blocks from (2.15), this block is replaced with its numerical equivalent and this generated representation is inserted into `m_r` (see Algorithm 2);

Algoritmus 2: Process of blocks replacement

```
function analyze(set⟨string⟩ m_r, string rep)
    blocks ← {‘100’:‘021’, ‘101’:‘022’, ‘200’:‘121’, ‘201’:‘122’}
    for i ← 0 to rep.length() − 3 do
        if rep[i,i + 2] is in blocks.keys() then
            string new_r = rep
            new_r[i,i + 2] ← blocks[rep[i,i + 2]]
            m_r.insert(new_r)
        end
    end
```

Since any representation generated using changeable blocks from (2.15) is lexicographically smaller than the currently processed one, it is inserted into `m_r` above the element pointed to by the iterator `i`.

The iterator `i` goes till the 1st element of `m_r`, therefore this new inserted representation will be analyzed with time as well.

To explain the algorithm of finding more representations in more detail we will continue using the number 90 as an example.

At the very first step the greedy representation of the number 90 from (2.9) is taken into account: $(90)_G = 20101$. From now $t_5t_4t_3t_2t_1 = 20101$.

This greedy representation is inserted into `m_r`:

`m_r`:		
	`rend();`	
	20101	$\leftarrow i$

Now, as shown above, we start working with $t_5t_4t_3t_2t_1 = 20101$. We see a block ’201’, which starts at the coefficient t_5, and a block ’101’ starting at t_3. By replacing these two blocks using (2.15), we generate two representations: 12201 and 20022. After they are inserted into the `m_r`, we get:

m_r:

`rend();`	
12201	
20022	
20101	$\leftarrow i$

There are no more changeable blocks in the greedy representation, so we can move the iterator `i` to the next element of `m_r`: 20022. From now $t_5t_4t_3t_2t_1 = 20022$.

m_r:

`rend();`	
12201	
20022	$\leftarrow i$
20101	

There is a block '200' starting at t_5. After this block is replaced by its equivalent, we generate the representation 12122. This representation is new (did not appear on the list before), so it can be inserted into `m_r`:

m_r:

`rend();`	
12122	
12201	
20022	$\leftarrow i$
20101	

There are no more blocks in 20022 to change, so we move on to the next representation: $t_5t_4t_3t_2t_1 = 12201$.

The changeable block starts at t_3. After the replacement, we generate the representation 12122. However, the same representation has already been generated and exists in the `m_r`. This representation will not be inserted into the list again. Strictly speaking, we call the insert method of the `set` container, but the method will not insert 12122 into `m_r` since it is already there.

Moving on, there is only one representation left: 12122. It does not contain any changeable blocks from (2.15), so it will stay as it is, which means we reached the end of the `m_r`.

In the end we found four representations of the number 90 in the Raminacci system:

$$
\begin{aligned}
(90)_G &= 20101, \\
(90)_G &= 20022, \\
(90)_G &= 12201, \\
(90)_G &= 12122.
\end{aligned} \tag{3.1}
$$

therefore $R(90) = 4$.

3.3 Results

During the use of the program for finding representations in the Raminacci linear recurrent system, the first ten thousand numbers were tested. Table 3.1 shows the number of representations for each of the first 100 integers.

n	$R(n)$	n	$R(n)$	n	$R(n)$	n	$R(n)$
1	1	26	1	51	2	76	2
2	1	27	1	52	2	77	2
3	1	28	1	53	2	78	2
4	1	29	1	54	2	79	2
5	1	30	1	55	3	80	2
6	1	31	2	56	3	81	1
7	2	32	2	57	1	82	3
8	2	33	1	58	2	83	3
9	1	34	2	59	2	84	2
10	1	35	2	60	2	85	2
11	1	36	2	61	2	86	2
12	1	37	2	62	2	87	2
13	1	38	2	63	2	88	2
14	2	39	2	64	1	89	4
15	2	40	1	65	2	90	4
16	1	41	3	66	2	91	2
17	2	42	3	67	1	92	2
18	2	43	2	68	1	93	2
19	2	44	2	69	1	94	2
20	2	45	2	70	1	95	2
21	2	46	2	71	1	96	3
22	2	47	2	72	2	97	3
23	1	48	4	73	2	98	1
24	2	49	4	74	1	99	3
25	2	50	2	75	2	100	3

Table 3.1: First 100 integers and their number of representations

The graph in Figure 3.1 shows the results, where all the first 10,000 integers and their numbers of representations were put together.

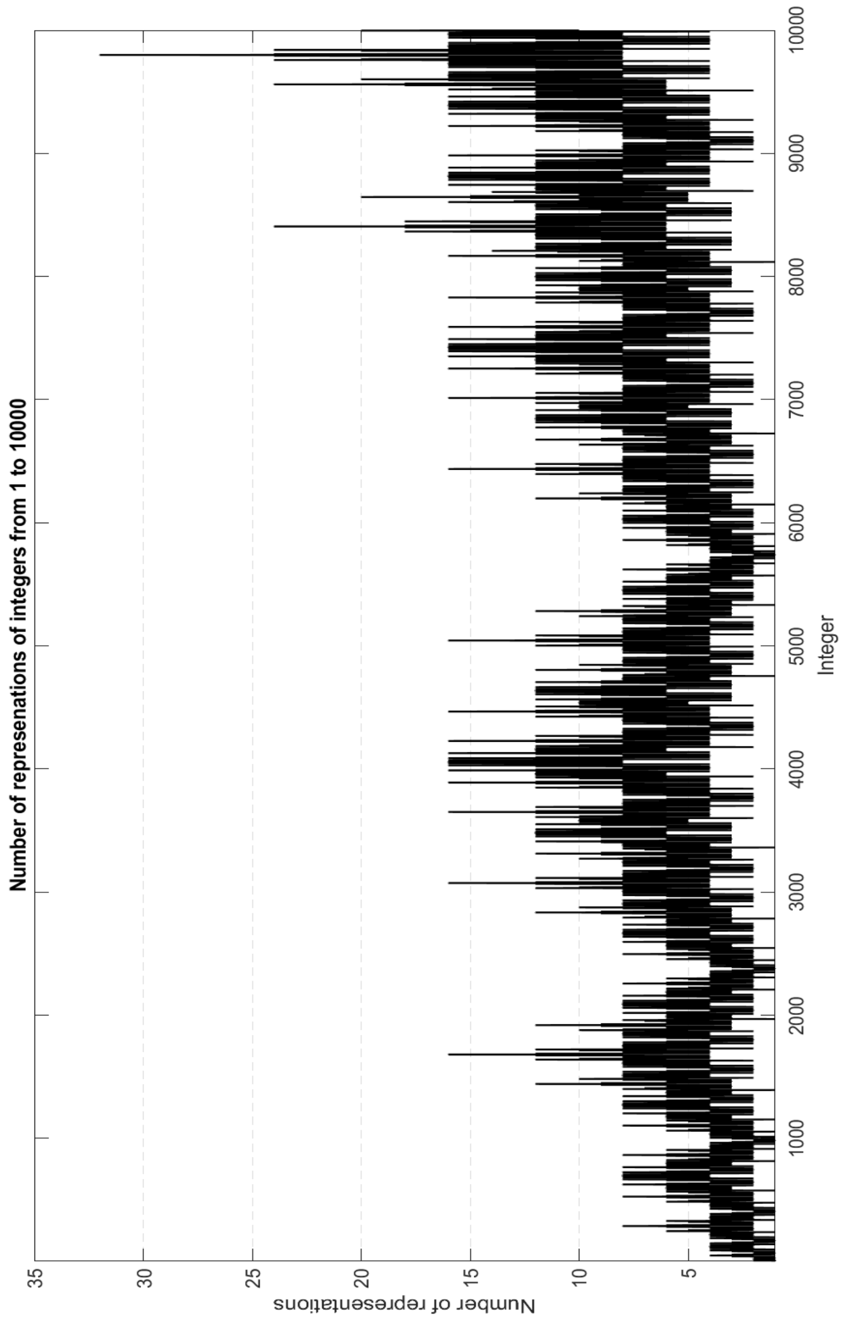

Figure 3.1: Number of representations

3.4 Numbers with one representation

It may seem to be impossible to predict how many representations a number has in the Raminacci system. However, with a close look at the graph it can noticed that some parts of it are symmetrical to one another. In fact, this symmetry represents the opportunity for finding a system or cycle, which can, for example, predict which numbers may have the same number of representations. The numbers with one representation only are one of such examples.

It was found that 110 numbers from the first 10,000 tested have only one representation. The following table lists all of these numbers.

1	16	67	166	395	576	989	2372	2785	5740
2	23	68	167	402	815	996	2375	3362	5741
3	26	69	168	405	914	1013	2376	4755	5742
4	27	70	169	406	955	1054	2377	5332	5745
5	28	71	170	407	972	1153	2378	5571	5752
6	29	74	173	408	979	1392	2379	5670	5769
9	30	81	180	409	982	1969	2382	5711	5810
10	33	98	197	412	983	2208	2389	5728	5909
11	40	139	238	419	984	2307	2406	5735	6148
12	57	156	337	436	985	2348	2447	5739	6725
13	64	163	378	477	986	2365	2546	5739	8118

Table 3.2: Integers with one representation

Even though it seems that these numbers do not have anything in common, they do share a strong connection with the Raminacci sequence. Moreover, there is a certain pattern in the manifestation of the numbers with one representation. The Raminacci sequence, however, is not the only one numbers with one representation share connection with. The sequence

$$1, 1, 2, 1, 1, 2, 3, 2, 1, 1, 2, 3, 4, 3, 2, 1, \ldots \tag{3.2}$$

obtained by concatenating the strings of raising and falling integers, i.e. of strings $(1, 2, \ldots, n-1, n, n-1, \ldots, 2, 1)$ for $n \geqslant 1$, is called the Smarandache crescendo pyramidal sequence [5]. Let the sequence (3.2) be denoted as $(S_i)_{i\geqslant 1.}$ The following observation shows how this sequence is connected with the Raminacci one in more detail.

Observation 3.2. *Let $(UR_i)_{i\geqslant 1}$ be the sequence of the natural numbers with only one representation in the Raminacci numeration system. Then*

$$\begin{aligned} &UR_1 = 1, \\ &UR_i = UR_{i-1} + G_{S_i - 1,} \quad \forall i \geq 2. \end{aligned} \tag{3.3}$$

Numbers in (UR_i) may be divided into pyramid cycles of odd length, corresponding to pyramid cycles in S_i, always starting and ending with the addition of G_0 (to the previous element).

In (3.4) can be seen the first three cycles, namely

$$\begin{gathered} (UR_1) \\ (UR_{2,}\ UR_{3,}\ UR_4) \\ (UR_{5,}\ UR_{6,}\ UR_{7,}\ UR_{8,}\ UR_9). \end{gathered}$$

The example below shows these cycles creating pyramids in more detail with the last added G_i.

$$
\begin{array}{rll}
UR_1 = & 1 = & +G_0 \\
\hdashline
UR_2 = & 2 = 1 & +G_0 \\
UR_3 = & 3 = 2 & \quad +G_1 \\
UR_4 = & 4 = 3 & +G_0 \\
\hdashline
UR_5 = & 5 = 4 & +G_0 \\
UR_6 = & 6 = 5 & \quad +G_1 \\
UR_7 = & 9 = 6 & \qquad +G_2 \\
UR_8 = & 10 = 9 & \quad +G_1 \\
UR_9 = & 11 = 10 & +G_0
\end{array}
\tag{3.4}
$$

These pyramid cycles from (3.4) represent a strong connection with the next observation, creating a base for new following patterns.

Observation 3.3. *Let* $UR_{k+1}, \ldots, UR_{k+2l-1}$ *be a pyramid cycle in* (UR_i)*. Then*

a) $(UR_{k+l})_G = 1w,$ *where* $w \in \{0, 1, 2\}^*$,

b) $\forall i \in \{1, \ldots, l-1\}$ $|(UR_{k+i})_G| = |(UR_{k+l})_G| - 1$,

c) $(UR_{k+2l-1})_G = 1^m$ *for some* $m \in \mathbb{N}$.

To demonstrate these individual statements from Observation 3.3 the third cycle in (3.4) will be used as example:

$$UR_5 = 5,\ UR_6 = 6,\ UR_7 = 9,\ UR_8 = 10,\ UR_9 = 11.$$

It is said in a) that the element in the centre of a pyramid (corresponding to the peak of the sequences' S_i part) has a representation in the Raminacci numeration system beginning with the coefficient 1

$$(UR_7)_G = 102,$$

which is, in accordance with b), longer than representations in the first half of the pyramid

$$(UR_5)_G = 12,$$
$$(UR_6)_G = 20.$$

In c) it has been claimed that the last element of any pyramid has a representation which consists of ones only

$$(UR_9)_G = 111.$$

3.5 Representations of Raminacci numbers

Not only the numbers with one representation have something in common. Members of the Raminacci sequence themselves have their own system for finding the numbers of representation. Using the following theorem, we can easily find all the representations for these special numbers.

Theorem 3.4. *Let G_i be an element of the base of the Raminacci linear recurrent system. Then*

$$R(G_i) = \left\lceil \frac{i}{2} \right\rceil .$$

Proof. Let i be an odd number, i.e. $i = 2k + 1$ for some $k \in \mathbb{N}$. Then

$$(G_i)_G = 1\underbrace{0 \cdots 0}_{2k \text{ zeros}}$$

We can generate other representations of G_i using the first rewriting rule given in (2.13).

$$\begin{aligned} 1\underbrace{0\cdots0}_{2k \text{ zeros}} \equiv_G 021\underbrace{0\cdots0}_{2(k-1) \text{ zeros}} \equiv_G 02021\underbrace{0\cdots0}_{2(k-2) \text{ zeros}} \equiv_G \cdots \\ \equiv_G 0\underbrace{2020\cdots20}_{k-1 \text{ blocks } 20}21. \end{aligned} \tag{3.5}$$

In each step there is a unique rewritable block, thus there is no other representation of G_i than those in (3.5). We did k steps in (3.5), therefore

$$R(G_i) = k + 1 = \left\lfloor \frac{i}{2} \right\rfloor + 1 = \left\lceil \frac{i}{2} \right\rceil .$$

Let i be an even number, i.e. $i = 2k$ for some $k \in \mathbb{N}$. Then

$$(G_i)_G = 1\underbrace{0\cdots0}_{2(k-1) \text{ zeros}}0.$$

The proof follows analogically to the case of odd i.

$$\begin{aligned} 1\underbrace{0\cdots0}_{2(k-1) \text{ zeros}}0 \equiv_G 021\underbrace{0\cdots0}_{2(k-2) \text{ zeros}}0 \equiv_G 02021\underbrace{0\cdots0}_{2(k-3) \text{ zeros}}0 \equiv_G \cdots \\ \equiv_G 0\underbrace{20\cdots20}_{k-2 \text{ blocks } 20}210. \end{aligned} \tag{3.6}$$

In this case we did $k-1$ steps in (3.6), therefore

$$R(G_i) = k = \frac{i}{2} = \left\lceil \frac{i}{2} \right\rceil . \qquad \square$$

As an example of the use of this theorem let us have a look at the number $G_4 = 17$. Index $i = 4$ is even, so $R(17) = 4/2 = 2$. To prove this in another way the greedy algorithm and the direct method may be used. After applying them for this particular number we see that there are, as expected, only two representations:

$$
\begin{aligned}
(17)_G &= 1000,\\
(17)_G &= 0210.
\end{aligned}
$$

The same as with the number with the even index, the number $G_9 = 1393$ will be taken. For this particular number $R(1393) = \left\lceil \frac{9}{2} \right\rceil = 5$. Once more we see the correspondence with Theorem 3.4 as there are 5 representations in the Raminacci system:

$$
\begin{aligned}
(1393)_G &= 100000000,\\
(1393)_G &= 021000000,\\
(1393)_G &= 020210000,\\
(1393)_G &= 020202100,\\
(1393)_G &= 020202021.
\end{aligned}
$$

Chapter 4

Symmetry of graph of $R(n)$

4.1 Types of symmetry

In this part of this project, we will have a closer look at the symmetry of the graph in Figure 3.1. Some parts of it do repeat, rise to the peaks and fall, creating symmetrical 'triangles'. However, this graph does not correspond to one-type symmetry only.

The whole graph is composed of longer and longer centrally symmetrical sections. There are two such types, which alternate:

- 'Acute' peak section
  - the maximum value appears *two* times only in the middle of the section;
- 'Obtuse' peak section
  - the maximum value appears *six* times in the middle of the section, moreover, it appears in other places of the section as well.

We can say from the graph that new maximum values at first appear at the acute sections, which are followed by obtuse ones with the same value of the peak.

Let us have a look at examples of both types of symmetries.

As an example of the acute peak section let us take a section from $n = 16$ to $n = 81$. For this particular part, the maximal number of representations of a number is 4:

$$\max_{16 \leq i \leq 82} R(i) = 4. \tag{4.1}$$

As previously mentioned, there are two numbers with the maximal number of representations in the section. Such two numbers for this section are 48 and 49, both of them have four representations in the Raminacci system:

$$\operatorname*{argmax}_{16 \leq i \leq 82} R(i) = \{48, 49\}. \tag{4.2}$$

Analogically, one of the examples of the obtuse peak section is a section from $n = 82$ to $n = 155$. It has the same maximal value as in (4.1). However, since the peak in this type of symmetry does not only appear in the middle of the section, the argmax part is slightly different:

$$\begin{aligned} \operatorname*{argmax}_{16 \leq i \leq 82} R(i) = \{89, 90, 106, 107, 113, 114, \overbrace{116, \ldots, 121}^{\text{center of the section}}, \\ 123, 124, 130, 131, 147, 148\}. \end{aligned} \tag{4.3}$$

During the study of symmetry, a few observations were made.

Observation 4.1. *Let* $(n-k+1, \ldots, n, n+1, \ldots, n+k)$ *be a symmetrical section of the acute type of length* $2k$*. Then*

- $R(n) = R(n+1) = 2^l$ *for some* $l \in \mathbb{N}$*;*

- $\max\limits_{i<n} R(i) < 2^l$.

Observation 4.2. *Let* $(n-k+1,\dots,n,n+1,\dots,n+k)$ *be the symmetrical section of the acute type and* $(m-l+1,\dots,m-2,m-1,m,m+1,m+$ $+2,m+3,\dots,m+l)$ *be the obtuse section, which immediately follows, then*

$$R(n) = R(n+1) = R(m-1) = \cdots = R(m+3) = 2^p \textit{ for some } p \in \mathbb{N}.$$

To sum up, for any acute type section it is not possible to have more than one peak, in which the maximum value appears twice. Since two types of symmetry alternate, after any acute section there is the obtuse section, which immediately follows it and has the same value of the peak as the section before.

Conclusion

In this project, we became acquainted with the Raminacci sequence. We also became familiar with the way of calculating Raminacci numbers and with using them to expand natural numbers into the Raminacci linear recurrent system. For studying the features and properties of this system, a greedy algorithm for finding the very first representation was created.

The properties of the Raminacci system gave us the possibility of having more than one representation for some natural numbers. A suitable algorithm for finding these representations was created as well. For faster calculations, C++ program was created.

Together with multi-representation natural numbers, numbers with one representation only were studied here. In this work, we represented the patterns these numbers have between each other. The Raminacci numbers themselves (and their number of representations) were examined here in detail.

The symmetry of the graph of $R(n)$ was mentioned as well, and a few observations and features were presented.

Raminacci program

The Raminacci program for finding representations of natural numbers may be found under this link:

`https://github.com/JaNeProgrammist/Raminacci`

The program is easy to run and use. After the program starts running, it asks the user to enter a natural number to convert it into the Raminacci system. After an integer has been entered, the Raminacci sequence will appear, and all representations for an entered number will be shown to the user. At the end of the list, the total number of representations may be found.

Bibliography

[1] A. Avizienis. *Signed-digit number representations for fast parallel arithmetic*. IRE Trans. **EC-10** (1961) 389–400.

[2] J. Berstel. *An excercise on Fibonacci representations*. RAIRO Theor. Inform. Appl. **35** (2002) 491–498.

[3] M. Lothaire. *Algebraic combinatorics on words*. Encyclopedia of Mathematics and its Applications, 90. Cambridge University Press, Cambridge (2002).

[4] F. Maňák. *Reprezentace přirozených čísel ve Fibonacciho soustavě*. Diplomová práce FJFI ČVUT (2005).

[5] F. Smarandache. *Collected Papers, Vol. 2*. Kishinev University Press, Kishinev (1997).

[6] A. Stepanov and M. Lee. *The Standard Template Library*. HP Laboratories Technical Report 95-11 (1995).