SIMPLE ALGORITHM FOR SORTING THE FIBONACCI STRING ROTATIONS

Manolis Christodoulakis[1], Costas S. Iliopoulos[1],
Yoan José Pinzón Ardila[2]

[1]King's College London, Department of Computer Science
London WC2R 2LS, UK
email: {manolis,csi}@dcs.kcl.ac.uk

[2]Pontificia Bolivariana University, Faculty of Informatics
Km 7 Via Piedecuesta, Colombia
email: yoan@pinzon.co.uk

Abstract: In this paper we focus on the combinatorial properties of the Fibonacci strings rotations. We first present a simple formula that, in constant time, determines the rank of any rotation (of a given Fibonacci string) in the lexicographically-sorted list of all rotations. We then use this information to deduce, also in constant time, the character that is stored at any one location of any given Fibonacci string. Finally, we study the output of the Burrows-Wheeler Transform (BWT) on Fibonacci strings to prove that when BWT is applied to Fibonacci strings it always produces a sequence of 'b's followed by a sequence of 'a's.

Key Words: block-sorting; Fibonacci strings; data compression; text compression; BWT Transformation.

1 Introduction

Fibonacci strings[1] have been widely studied and are considered to be a matter of common knowledge, see, for example, [1] for a good reference. Fibonacci strings are important in many contexts, but they are frequently often cited in journal articles and elsewhere as *worst-case scenarios* for string pattern matching algorithms such as KMP[2], Boyer-Moore and Aho-Corasick automaton, and in *string statistics* like for computing all the Abelian squares in a given string [3]. Another domain in which the combinatorial properties of the Fibonacci strings are of great interest is in some aspects of mathematics and physics, such as number theory, fractal geometry, formal language, computational complexity, quasicrystals, etc.

Informally, a Fibonacci string F_n is a string of characters with the property that each successive string of the sequence is obtained as the concatenation of the previous two. For example, the first five Fibonacci strings are: `b`, `a`, `ab`, `aba` and `abaab` (*c.f.* also Fig. 1(left)). Here we are concerned with the lexicographic ordering of the rotations of a Fibonacci string, we show that for a given rotation of a particular Fibonacci string, one can identify the order of that rotation in the lexicographically-sorted list of all the rotations of F_n, without the need for explicit sorting of the rotations. The inverse problem, consisting of finding the rotation that has a given order in the sorted list, can also be solved without sorting. In addition, we show how the ordering of the rotations can be used to determine the symbols of any Fibonacci string without using the traditional recursive definition of Fibonacci strings or the Golden Ratio ϕ.

Analysing rotations of strings can be useful for algorithms whose operation depends on rotations of strings and their lexicographic ordering. One such algorithm is the block-sorting transformation known as Burrows-Wheeler Transform (BWT) [4] used to bring repeated characters together as a preliminary to compression. When BWT is applied to a string x of length n, it produces the lexicographically-sorted list of all n rotations of x and outputs the last symbol of every rotation of the sorted list together with the rank of the $0th$ rotation. By making best use of the already mentioned rationale, we show how to compute the output of BWT when applied on Fibonacci strings without engaging in any costly sorting operation. In particular, we prove that the output is always the permutation that consists of all the '`b`'s that are contained in the particular Fibonacci string, followed by all its '`a`'s. Fibonacci strings are closely related to Sturmian words[3], hence related work can be found in [5],

[1] *a.k.a.* Fibonacci words.

[2] Knuth-Morris-Pratt.

[3] Sturmian words are infinite words over a two-letter alphabet of minimal subword complexity which are not eventually periodic, or, in other words, that have exactly $n+1$ factors

n	F_n	f_n
0	`b`	1
1	`a`	1
2	`ab`	2
3	`aba`	3
4	`abaab`	5
5	`abaababa`	8
6	`abaababaabaab`	13
7	`abaababaabaababaababa`	21

(a) Fibonacci strings and numbers

rank (ρ)	*rotation index* (i)	*rotation*
0	7	$\mathcal{R}_7 =$ `aabaabab`
1	2	$\mathcal{R}_2 =$ `aababaab`
2	5	$\mathcal{R}_5 =$ `abaabaab`
3	0	$\mathcal{R}_0 =$ `abaababa`
4	3	$\mathcal{R}_3 =$ `ababaaba`
5	6	$\mathcal{R}_6 =$ `baabaaba`
6	1	$\mathcal{R}_1 =$ `baababaa`
7	4	$\mathcal{R}_4 =$ `babaabaa`

(b) Lexicographically-sorted rotations of F_5

Figure 1: Fibonacci strings and their rotations.

where Mantaci et. al. derived a very similar result using a different approach.

The remainder of this paper is organised as follows. In the next section, we provide some basic definitions and prove some properties of the Fibonacci numbers which will play a key role in proving the main results in the succeeding sections. In Sect. 3 we prove that the rank of any rotation of a Fibonacci string in the sorted list of all rotations of the particular Fibonacci string, can be computed in constant time. Section 4 explains how to use the above results to instantly deduce the symbol stored in any position of a Fibonacci string. Finally, in Sect. 5 we prove why the output of BWT when applied to a Fibonacci string, produces a sequence of '`b`'s followed by a sequence of '`a`'s. Concluding remarks follow in the last section.

2 Preliminaries

We define Fibonacci number by $f_0 = 1, f_1 = 1, f_n = f_{n-1} + f_{n-2}$ and Fibonacci strings are defined by $F_0 = \mathtt{b}, F_1 = \mathtt{a}, F_n = F_{n-1}F_{n-2}$, for $n \geq 2$. Obviously, $|F_i| = f_i$. See Fig. 1(a) for some examples.

Definition 1. The *ith* rotation of a string $x = x_0 \ldots x_{n-1}$ is defined by the string $\mathcal{R}_i(x) = x_i x_{i+1} \ldots x_{n-1} x_0 x_1 \ldots x_{i-1}$.

Note that $\mathcal{R}_{i+j}(x) = \mathcal{R}_i(\mathcal{R}_j(x)) = \mathcal{R}_j(\mathcal{R}_i(x))$. Thus the ith rotation[4] can be defined for $0 < i \geq n$ as $\mathcal{R}_i(x) = \mathcal{R}_{i \bmod n}(x)$. For F_5, for example, Fig. 1(b) gives the sorted list of all rotations.

of length n for each $n \geq 0$.

[4]In the sequel, when refering to the ith rotation, we imply the $(i \bmod n)th$ rotation.

We denote by $rank(i,x)$ the rank of $\mathcal{R}_i(x)$ in the lexicographically-sorted list of all rotations of x. We write $rot(\rho,x)$ to denote the index of the rotation with rank ρ, that is, $rot(\rho,x)=i$ iff $rank(i,x)=\rho$. For instance, in Fig. 1(b) $rank(3,F_5)=4$, and $rot(5,F_5)=6$.

Next, we state, without proof, two easily established lemmas that will be required later. The first is an elementary result from number theory, while the second corresponds to Fibonacci number analysis.

Lemma 1 ([7, page 243]). *The congruence* $ax \equiv b \pmod{n}$ *has a unique solution* $x \in [0,n)$ *if* a *is relatively prime to* n.

Lemma 2 ([8, page 151]). f_n *is relatively prime to* f_{n-1}*, for every* $n \geq 2$.

2.1 Some New Properties of Fibonacci Numbers

Here we prove some properties of Fibonacci numbers which will be used in the proofs of subsequent lemmas regarding Fibonacci strings.

Lemma 3. f_n *is relatively prime to* f_{n-2}*, for every* $n \geq 2$.

Proof. Assume f_n is not relatively prime to f_{n-2}; that is, $f_n = mk$ and $f_{n-2} = m\ell$ for some integers m,k,ℓ, where $m \neq 1$ and $k > \ell$. Then

$$f_n = f_{n-1} + f_{n-2} \iff mk = f_{n-1} + m\ell \iff m(k-\ell) = f_{n-1}$$

and thus f_{n-1} is not relatively prime to f_n, since they have a common factor, $m \neq 1$. This contradicts Lemma 2. □

Lemma 4.

$$f_{n-1}^2 \bmod f_n = \begin{cases} -1, & \text{if } n \text{ odd} \\ 1, & \text{if } n \text{ even} \end{cases} \qquad \text{for } n \geq 2$$

Proof. By Cassini's identity [2, page 80] $f_{n-1}f_{n+1} - f_n^2 = (-1)^n$.

$$\begin{aligned}
f_{n-1}f_{n+1} - f_n^2 &= (-1)^n \implies \\
f_{n-1}(f_n + f_{n-1}) - f_n^2 &= (-1)^n \implies \\
f_{n-1}f_n + f_{n-1}f_{n-1} - f_n^2 &= (-1)^n \implies \\
(f_{n-1}f_n + f_{n-1}^2 - f_n^2) \bmod f_n &= (-1)^n \bmod f_n \implies \\
f_{n-1}^2 \bmod f_n &= (-1)^n \bmod f_n \implies \\
f_{n-1}^2 \bmod f_n &= \begin{cases} -1 & \text{if } n \text{ odd} \\ 1 & \text{if } n \text{ even} \end{cases}
\end{aligned}$$

□

Corollary 1.

$$f_{n-2}^{-1} \bmod f_n = \begin{cases} f_{n-1} & \text{if } n \text{ odd} \\ f_{n-2} & \text{if } n \text{ even} \end{cases}$$

Proof. By Lemma 4, for n odd:

$$\begin{aligned} f_{n-1}^2 \bmod f_n &= -1 \iff \\ f_{n-1}(f_n - f_{n-2}) \bmod f_n &= -1 \iff \\ f_{n-1}f_n - f_{n-1}f_{n-2} \bmod f_n &= -1 \iff \\ f_{n-1}f_{n-2} \bmod f_n &= 1 \iff \\ f_{n-2}^{-1} \bmod f_n &= f_{n-1} \end{aligned}$$

By Lemma 4, for n even:

$$\begin{aligned} f_{n-1}^2 \bmod f_n &= 1 \iff \\ (f_n - f_{n-2})^2 \bmod f_n &= 1 \iff \\ (f_n^2 - 2f_nf_{n-2} + f_{n-2}^2) \bmod f_n &= 1 \iff \\ f_{n-2}^2 \bmod f_n &= 1 \iff \\ f_{n-2}^{-1} \bmod f_n &= f_{n-2} \end{aligned}$$

□

3 Ranking the Rotations of Fibonacci Strings

Lemma 5. *For every integer $n \geq 2$, $F_n = F_{n-2}F_{n-3}\dots F_1u$, where*

$$u = \begin{cases} \texttt{ba} & \text{if } n \text{ odd} \\ \texttt{ab} & \text{if } n \text{ even} \end{cases}$$

Proof. This follows from Lemma 2.8 in [6]. □

Lemma 6. *The ith rotation of F_n $\mathcal{R}_i(F_n)$, for $i \in [0, f_n)$, $n \geq 2$, matches the $(i + f_{n-2})$th rotation, $\mathcal{R}_{i+f_{n-2}}(F_n)$ in all but two positions. Moreover, if $i \neq f_{n-1} - 1$ the two mismatches occur in consecutive positions.*

Proof. Consider $i = 0$, then $\mathcal{R}_0(F_n) = F_n = F_{n-2}F_{n-3}\dots F_1u$, by Lemma 5, where $u = \texttt{ba}$ if n is odd, and $u = \texttt{ab}$ if even. Then the $(i + f_{n-2})th$ rotation is

$$\mathcal{R}_{f_{n-2}}(F_n) = F_{n-3}\dots F_1uF_{n-2} \tag{1}$$

but also

$$\mathcal{R}_0(F_n) = F_n = F_{n-1}F_{n-2} = F_{n-3}\dots F_1u'F_{n-2} \tag{2}$$

where F_{n-1} has been written in the form given by Lemma 5, and $u' = \texttt{ab}$ if $n-1$ is even ($i.e$ n is odd), $u' = \texttt{ba}$ for $n-1$ odd ($i.e$ n is even). So for $i = 0$

the rotations do not match at positions $f_{n-1}-2$ and $f_{n-1}-1$ (the positions where the two symbols of u occur; *see* (1) and (2)).

For any $i \in [0, f_n)$ the rotations $\mathcal{R}_i(F_n) = \mathcal{R}_i(\mathcal{R}_0(F_n))$ and $\mathcal{R}_{i+f_{n-2}}(F_n) = \mathcal{R}_i(\mathcal{R}_{f_{n-2}}(F_n))$ do not match in the same two symbols located now at positions $f_{n-1}-2-i$ and $f_{n-1}-1-i$ (modulo f_n). These two positions are unconsecutive only for rotation $i = f_{n-1}-1$, because for this rotation, the first symbol of u will be located at position (f_n-1), and the second symbol of u will be located at position 0. $\square$

Lemma 7. *The* ith *rotation of* F_n *(*$n \geq 2$*),* $\mathcal{R}_i(F_n)$*, is lexicographically smaller (*resp. *larger) than the* $(i + f_{n-2})$th *rotation,* $\mathcal{R}_{i+f_{n-2}}(F_n)$*, for* n *odd (*resp. *even), for all* $i \in [0, f_n)$*,* $i \neq f_{n-1}-1$*. For* $i = f_{n-1}-1$*, the* ith *rotation is lexicographically larger (*resp. *smaller) for* n *odd (*resp. *even).*

Proof. From the proof of Lemma 6 we know that

$$\mathcal{R}_0(F_n) = F_{n-3}\ldots F_1 u' F_{n-2} \quad \text{and} \quad \mathcal{R}_{f_{n-2}}(F_n) = F_{n-3}\ldots F_1 u F_{n-2}$$

where $u' = \mathtt{ab}$ and $u = \mathtt{ba}$ when n is odd, $u' = \mathtt{ba}$ and $u = \mathtt{ab}$ when n is even. Thus, the $0th$ rotation is lexicographically smaller (*resp.* larger) from the $f_{n-2}th$ for n odd (*resp.* even). The same is true for every other rotation $i \neq f_{n-1}-1$, since the two symbols of u (and u') occupy consecutive positions.

For $i = f_{n-1}-1$ and n odd

$$\begin{aligned} \mathcal{R}_{f_{n-1}-1}(F_n) &= \mathtt{b}F_{n-2}\ldots F_1\mathtt{a} \quad (u' = \mathtt{ab}) \\ \mathcal{R}_{f_{n-1}-1+f_{n-2}}(F_n) = \mathcal{R}_{f_n-1}(F_n) &= \mathtt{a}F_{n-2}\ldots F_1\mathtt{b} \quad (u = \mathtt{ba}). \end{aligned}$$

Consequently $\mathcal{R}_i$ is lexicographically larger than $\mathcal{R}_{i+f_{n-2}}$. Similarly, for n even $\mathcal{R}_i$ is lexicographically smaller than $\mathcal{R}_{i+f_{n-2}}$. $\square$

Theorem 1. *The rotation of* F_n $rot(\rho, F_n)$ *with rank* ρ *in the lexicographically-sorted list of all the rotations of* F_n*, for* $n \geq 2$*,* $\rho \in [0..f_n)$*, is the rotation*

$$rot(\rho, F_n) = \begin{cases} (\rho \cdot f_{n-2} - 1) \bmod f_n & \text{if } n \text{ odd} \\ (-(\rho+1) \cdot f_{n-2} - 1) \bmod f_n & \text{if } n \text{ even} \end{cases}$$

Proof. We will prove the theorem by constructing the list of lexicographically sorted rotations. Consider n odd, intuitively, rotation $\mathcal{R}_i = \mathcal{R}_{f_n-1}$ is the smallest and therefore the first in the sorted list (it is the only rotation not preceded by $\mathcal{R}_{i-f_{n-2}}$, using Lemma 7). We will prove later that no other rotation can be smaller. Now, consider that $\mathcal{R}_i = \mathcal{R}_{f_n-1}$ occupies position 0 in the sorted list. By Lemma 7, underneath (but maybe not immediately below, but at some later point) there will be $\mathcal{R}_{i+f_{n-2}}$. This rotation at the same time will be followed by $\mathcal{R}_{i+2f_{n-2}}$, followed by ..., followed by $\mathcal{R}_{i+kf_{n-2}}$ $(k \geq 2)$, for as long as

$$i + kf_{n-2} \neq f_{n-1} - 1 \pmod{f_n}$$

(by Lemma 7). We solve the following equation to find the smallest k for which the above inequality is not true:

$$\begin{aligned} i + kf_{n-2} &= f_{n-1} - 1 \pmod{f_n} \\ f_n - 1 + kf_{n-2} &= f_{n-1} - 1 \pmod{f_n} \\ f_n + kf_{n-2} &= f_{n-1} \pmod{f_n} \\ f_n - f_{n-1} + kf_{n-2} &= 0 \pmod{f_n} \\ f_{n-2} + kf_{n-2} &= 0 \pmod{f_n} \\ (k+1)f_{n-2} &= 0 \pmod{f_n} \end{aligned}$$

which means that $(k+1)f_{n-2}$ and f_n share a common factor $m \neq 1$. By Lemma 3, f_{n-2} is relatively prime to f_n, thus it must be $k + 1 = 0 \pmod{f_n}$, or identically $k = f_n - 1$.[5] Therefore, there are no more rotations left out which could possibly be placed anywhere between the rotations that we have already inserted in the sorted list. Hence the ρth position in the sorted list is occupied by rotation

$$(f_n - 1 + \rho f_{n-2}) \bmod f_n = (\rho f_{n-2} - 1) \bmod f_n.$$

For n even, we construct the sorted list in a similar fashion, only now we start by placing $\mathcal{R}_{f_n-1}$ at the bottom of the list (position $f_n - 1$), and place any $\mathcal{R}_{i+f_{n-2}}$ atop rotation i. Thus now, $\mathcal{R}_{(f_n-1+kf_{n-2}) \bmod f_n} = \mathcal{R}_{(kf_{n-2}-1) \bmod f_n}$ take up position $f_n - k - 1$; that is, the ρth position is occupied by rotation

$$((f_n - \rho - 1)f_{n-2} - 1) \bmod f_n = (-(\rho + 1)f_{n-2} - 1) \bmod f_n.$$

□

Corollary 2. *The rank of the ith rotation of F_n, $rank(i, F_n)$, in the lexicographically sorted list of all the rotations of F_n, for $i \in [0..f_n)$, $n \geq 2$, is:*

$$rank(i, F_n) = \begin{cases} ((i+1) \cdot f_{n-2}) \bmod f_n & \text{if } n \text{ odd} \\ ((i+1) \cdot f_{n-2} - 1) \bmod f_n & \text{if } n \text{ even.} \end{cases}$$

Proof. For n odd, by Theorem 1, the ith position is occupied by rotation $(i \cdot f_{n-2} - 1) \bmod f_n$, thus the ith rotation is located at position

$$(i+1) \cdot f_{n-2}^{-1} \bmod f_n = (i+1) \cdot f_{n-2} \bmod f_n$$

since, by Lemma 1, $f_{n-2}^{-1} = f_{n-2}$ for n odd.

Similarly, for n even, by Theorem 1, the ith position is occupied by rotation

$$(-(i+1) \cdot f_{n-2} - 1) \bmod f_n = ((i+1) \cdot f_{n-1} - 1) \bmod f_n$$

thus the ith rotation is located at position

$$((i+1) \cdot f_{n-1}^{-1} - 1) \bmod f_n = ((i+1) \cdot f_{n-2} - 1) \bmod f_n$$

since, by Lemma 1, $f_{n-1}^{-1} = f_{n-2}$ for n even. □

[5] Note that, by Lemma 1, this solution is unique in $[0, f_n)$.

4 Predicting the Symbols of Fibonacci Strings

Lemma 8. *The number of '*`a`*'s in* F_n *(*$n \geq 2$*) is* f_{n-1}*.*

Proof. By induction.

- [basis] The number of '`a`'s in $F_2 =$ `ab` is $f_{2-1} = f_1 = 1$.
- [hypothesis] Assume that the number of '`a`'s in F_k is f_{k-1}, for all $k \in [2, n)$.
- [induction proof] The number of '`a`'s in $F_n = F_{n-1}F_{n-2}$ is the sum of '`a`'s in F_{n-1} and F_{n-2}, *i.e.* by induction hypothesis $f_{n-2} + f_{n-3} = f_{n-1}$.

□

Lemma 9. *The number of '*`b`*'s in* F_n *(*$n \geq 2$*) is* f_{n-2}*.*

Theorem 2. *For all* $i \in [0, f_n)$*, the* ith *symbol of* F_n *(*$n \geq 2$*) is*

$$F_n[i] = \begin{cases} \texttt{a}, & \textit{if } n \text{ odd } \textit{and } ((i+1) \cdot f_{n-2}) \bmod f_n < f_{n-1}, \\ & \textit{or } n \text{ even } \textit{and } ((i+1) \cdot f_{n-2} - 1) \bmod f_n < f_{n-1} \\ \texttt{b}, & \text{otherwise} \end{cases}$$

Proof. Observe that, the ith symbol of F_n is the first symbol of the ith rotation. In the lexicographically-sorted list of rotations, all rotations that start with '`a`' appear before all rotations that start with '`b`'. Therefore, $F_n[i]$ will be '`a`' iff the ith rotation has rank less than f_{n-1}; otherwise it is '`b`'. □

5 Burrows-Wheeler Transform on Fibonacci Strings

Lemma 10. *The first* f_{n-2} *rotations in the lexicographically-sorted list of rotations of* F_n *(*$n \geq 2$*) end in '*`b`*'.*

Proof. The last symbol of the ith rotation is the $((i + f_n - 1) \bmod f_n)th$ symbol of F_n; that is, the $((i-1) \bmod f_n)th$ symbol of F_n.

Consider n odd. By Theorem 1, the first f_{n-2} rotations are the rotations $(i \cdot f_{n-2} - 1) \bmod f_n$, $i \in [0, f_{n-2})$. The last symbol of these rotations is then $(i \cdot f_{n-2} - 2) \bmod f_n$, $i \in [0, f_{n-2})$. Whence, by using Theorem 2 we identify the last symbol of the first f_{n-2} rotations:

$$(i \cdot f_{n-2} - 2 + 1)f_{n-2} \bmod f_n = if_{n-2}^2 - f_{n-2} = i - f_{n-2} = i + f_{n-1}$$

which is $\geq f_{n-1}$, since $i \in [0, f_{n-2})$. Thus the symbol is '`b`'.

Equally for n even, by Theorem 1, the first f_{n-2} rotations are $(-(i+1) \cdot f_{n-2} - 1) \bmod f_n$, $i \in [0, f_{n-2})$. The last symbol of these rotations is then $(-(i+1) \cdot f_{n-2} - 2) \bmod f_n$, $i \in [0, f_{n-2})$. Then, by using Theorem 2 we identify the last symbol of the first f_{n-2} rotations:

$$\begin{aligned} [(-(i+1) \cdot f_{n-2} - 2 + 1) f_{n-2} - 1] \bmod f_n &= (-(i+1) \cdot f_{n-2}^2) - f_{n-2} - 1 = \\ = (i+1) - f_{n-2} - 1 &= i - f_{n-2} = i + f_{n-1} \geq f_{n-1} \end{aligned}$$

since again $i \in [0, f_{n-2})$. Thus the symbol is '`b`'. □

Corollary 3. *The last f_{n-1} rotations in the lexicographically-sorted list of rotations of F_n, $n \geq 2$, terminate in an '`a`'.*

Theorem 3. *The output of BWT when applied to F_n, $n \geq 2$ is*

$$(\underbrace{\texttt{bb}\ldots\texttt{b}}_{f_{n-2}}\underbrace{\texttt{aa}\ldots\texttt{a}}_{f_{n-1}}, k)$$

where k denote the rank of the 0th rotation in the lexicographically-sorted list, which is

$$k = \begin{cases} f_{n-2} + 1 & \text{if } n \text{ odd} \\ f_{n-2} & \text{if } n \text{ even.} \end{cases}$$

Proof. The output string of BWT is the last column of the lexicographically-sorted list of rotations, which by Lemma 10 and Corollary 3 is `bb`...`baa`...`a`.

The index produced by BWT is the rank of the initial string (the 0*th* rotation), which by Corollary 2 is

$$rank(0, F_n) = \begin{cases} f_{n-2} \bmod f_n & \text{if } n \text{ odd} \\ (f_{n-2} - 1) \bmod f_n & \text{if } n \text{ even.} \end{cases}$$

□

6 Conclusion

In this paper we focused on the combinatorial properties of the rotations of Fibonacci strings. We first presented a simple formula that determines the rank of any rotation (of a given Fibonacci string) in the lexicographically-sorted list of all rotations and then used this information to deduce, also in constant time, the symbols stored in any position of that Fibonacci string. We also proved that the output of the Burrows-Wheeler Transform (BWT) when applied to a Fibonacci string F_n, is always the permutation of F_n consisting of all the '`b`'s of F_n followed by all the '`a`'s.

References

[1] Berstel, J.: Fibonacci words—a survey. In Rozenberg, G., Salomaa, A., eds.: The Book of L. Springer-Verlag (1986) 13–27

[2] Knuth, D.E.: The Art of Computer Programming. 3rd edn. Volume 1. Addison-Wesley, Reading, Massachusetts (1997)

[3] Cummings, L.J., Smyth, W.F.: Weak repetitions in strings. The Journal of Combinatorial Mathematics and Combinatorial Computing **24** (1997) 33–48

[4] Burrows, M., Wheeler, D.: A block-sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation (1994)

[5] Mantaci, S., Restivo, A., Sciortino, M.: Burrows–Wheeler transform and Sturmian words. Information Processing Letters **86** (2003) 241–246

[6] Iliopoulos, C.S., Moore, D., Smyth, W.F.: A characterization of the squares in a Fibonacci string. Theoretical Computer Science **172** (1997) 281–291

[7] Eccles, P.: An Introduction to Mathematical Reasoning: Numbers, Sets and Functions. Cambridge University Press (1997)

[8] Koshy, T.: Elementary Number Theory With Applications. Elsevier (2001)