UNIVERSITY OF PALERMO
PHD JOINT PROGRAM:
UNIVERSITY OF CATANIA - UNIVERSITY OF MESSINA
XXXV CYCLE

DOCTORAL THESIS

Repetitiveness Measures based on String Attractors and Burrows-Wheeler Transform: Properties and Applications

Author:
Giuseppe ROMANA

Supervisor:
Prof. Marinella SCIORTINO

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in

Mathematics and Computational Sciences

June 12, 2023

Signed:

Date:

UNIVERSITY OF PALERMO

Abstract

Department of Mathematics and Computer Sciences

Doctor of Philosophy

Repetitiveness Measures based on String Attractors and Burrows-Wheeler Transform: Properties and Applications

by Giuseppe ROMANA

(EN) The problem of storing and indexing huge amount of data in a compact space is a fundamental task in Computer Science. Over the years different solutions have been proposed which exploit redundancies of information within texts to compress them, and their efficiency pushed the scientific community to consider the sizes of the compact representations of texts as measures of repetitiveness of the text. Two measures in particular have raised a discrete interest: the measure r, which refers to the number of equal-letter runs of the Burrows-Wheeler Transform, widely used in different tools in bioinformatics and data compression; the size γ^* of a smallest string attractor, which is asymptotically smaller than other classical compression schemes.

The goal of this thesis is to produce a qualitative analysis of r and γ as measures of repetitiveness, by identifying what properties of words or operations affect this sizes. We further show how the structure of the BWT allows to solve classical problem in bioinformatics in a space which grows at least as r. On string attractors on the other hand, we present the first combinatorial study on the measure γ^*, and we introduce new measures based on string attractors, which denote different properties of words. Finally, we extend the study to a circular variant of string attractor, and other one for infinite words, which can be used to characterize families of strings related to Sturmian words.

UNIVERSITY OF PALERMO

Abstract

Department of Mathematics and Computer Sciences

Doctor of Philosophy

Repetitiveness Measures based on String Attractors and Burrows-Wheeler Transform: Properties and Applications

by Giuseppe ROMANA

(IT) Memorizzare e indicizzare grandi quantità di dati in uno spazio compresso è un problema fondamentale in Informatica. Nel corso degli anni sono state proposte diverse soluzioni che traggono vantaggio dalla ridondanza di informazioni contenute in testi per comprimere, e la loro efficacia ha spinto la comunità scientifica a considerare le dimensioni delle rappresentazioni compresse come misure di ripetitività del testo. Due misure in particolare hanno recentemente suscitato un discreto interesse: la misura r, che si riferice al numero di run della trasformata di Burrows-Wheeler (BWT), largamente usata in diversi applicativi nell'ambito bioinformatico e in compressione dati; la taglia γ^* di uno string attractor minimo, che è asintoticamente minore di altri schemi di compressione classici.

L'obiettivo di questa tesi è quello di produrre un'analisi qualitativa per r e γ come misure di ripetitività, individuando gli effetti che proprietà di parole o operazioni hanno sulle stringhe. Mostriamo inoltre come la struttura della BWT permetta di risolvere classici problemi di bioinformatica in uno spazio che cresce almeno come r. Sugli string attractor invece presentiamo il primo studio combinatorio sulla misura γ^*, introducendo anche nuove misure legate agli string attractor che catturano diverse proprietà delle parole. Infine, estendiamo lo studio ad una variante circolare, e ad una per parole infinite, che possono essere usate per caratterizzare famiglie di stringhe legate alle parole Sturmiane.

Acknowledgements

Although I wanted to keep this section short, there are a lot of people who I feel compelled to thank for the realization of this thesis, especially considering the problematic period due to the pandemic outbreak that took place during my Ph.D. studies.

First of all, I want to thank my parents, relatives, and friends from Palermo, who have always supported me during the last three and more years.

A special thanks to my supervisor Marinella Sciortino, for her guidance and friendship during all the years that I have spent as a student, from the bachelor to the Ph.D., and hopefully for the years to come.

Many thanks to all the researchers with whom I have had the opportunity to discuss and collaborate: Gabriele Fici, Zsuzsanna Lipták, Simon Puglisi, Dominik Köppl, Travis Gagie, Massimiliano Rossi, and many more. I also want to thank Marie-Pierre Béal and Solon Pissis, for their fruitful comments and observations on the work of this thesis. A special mention goes to Antonio Restivo, whose passion for theoretical computer science inevitably influenced my decision to stay in the field, and for that I am sincerely grateful.

Thanks to Elena Biagi, Davide Cenzato, Lorenzo Di Rocco, Esteban Gabory, France Gheeraert, Sara Giuliani, Francesco Masillo, Luca Prigioniero, Nicola Rizzo, Manon Stipulanti, Stefan Hoffman, Cristian Urbina, and to all the new friends and valuable researchers that I have met along the way, who have been a spark of joy for me after two difficult first years.

Last but not least, I want to deeply thank my girlfriend, Eleonora, who has always believed in me, even when I did not, and whose love and patience were essential, especially in the darkest times.

Contents

Abstract **iii**

Acknowledgements **vii**

1 Introduction **1**

1.1 Contributions of the Thesis 3

1.2 Outline 5

2 Preliminaries **7**

2.1 Finite and Infinite Words 7

2.2 Morphisms 13

2.3 Repetitiveness of Words 15

2.3.1 Combinatorial Notions of Repetitiveness 16

2.3.2 Compressor based Repetitiveness Measures 17

3 Burrows-Wheeler Transform, BWT-runs, and Measure r **23**

3.1 Burrows-Wheeler Transform, Measures r and $r_{\$}$ 24

3.2 Factors of a Word and BWT-runs 27

3.3 Effects of Edit Operations on r 30

4 Measure r on Purely Morphic Words **47**

4.1 Comparison with other Repetitiveness Measures 49

4.2 Combinatorial Properties of Binary Morphisms 52

4.3 Logarithmic BWT-Runs of Binary Purely Morphic Words 58

4.4 Run-Length Compression of Purely Morphic Words 65

5 Effects of Morphisms on the BWT-Runs **69**

5.1 Abelian Order-Preserving and Order-Reversing Morphisms 69

5.2 Characterization of Sturmian Morphisms via r 74
5.3 On the Sensitivity of r for the Application of Morphisms 77

6 Applications in BWT-Runs Bounded Space 81
6.1 Computing Maximal Unique Matches with the r-Index 82
6.1.1 Maximal Unique Matches and Extended Matching Statistics . . 82
6.1.2 Checking Maximality and Uniqueness of Matches 84
6.1.3 Computing the Second Longest Match 88
6.1.4 Experimental Results 91
6.2 Burrows-Wheeler Transform for Collections of Sampled Strings 93
6.2.1 Description of the Problem 94
6.2.2 Preliminary Observations and Future Developments 96

7 String Attractors of Finite Words 97
7.1 Combinatorial Study of the Measure γ^* 99
7.2 New String Attractor based Measures 103

8 Circular Variant of String Attractors 109
8.1 Circular String Attractor 109
8.2 Characterization of Standard Sturmian Words via Circular String Attractors 114
8.2.1 Circular Span 114
8.2.2 Standard Sturmian Words and Bounded Circular Span 116
8.3 Checking the Attractor Property in Linear Time 117
8.3.1 Checking the Circular Attractor Property in Linear Time 118
8.3.2 Novel Algorithm for Checking the Attractor Property 123

9 String Attractors and Infinite Words 129
9.1 String Attractor Profile Function, Factor Complexity, and Recurrence . 130
9.1.1 Examples of String Attractor Profile Functions 131
9.1.2 The Bounded Case 133
9.1.3 The Case of Purely Morphic Words 134
9.2 Span and Leftmost Complexities 136
9.3 The Case of Sturmian Words 139

9.3.1 String Attractor based Complexities for Characteristic Sturmian Words 139
9.3.2 Characterization of Sturmian and Quasi-Sturmian Words 142

10 String Attractors and k-Bonacci Like Morphisms **145**
10.1 String Attractor based Complexities for k-Bonacci Words 145
10.1.1 k-Bonacci Words 146
10.1.2 String attractor profile function 148
10.1.3 Leftmost complexity 152
10.1.4 Span complexity 153
10.2 General Case of Fixed-Points of k-Bonacci-like Morphisms 155
10.2.1 k-Bonacci like Words and Morphisms 155
10.2.2 Fun with Numeration Systems 157
10.2.3 Link to String Attractors 161

11 Conclusions **167**
11.1 Future Directions of Research 168

Bibliography **171**

List of Figures

2.1 The parse-tree for a SLP generating the word $w = abracadabra$ 19

3.1 Example of BWT for the word *abracadabra*. 25

3.2 Example of BWT for the word *abracadabra$*. 26

3.3 Matrix of sorted rotations and corresponding Burrows-Wheeler Transforms of the words `abaababaabaa` and `abaababaabaab`. 31

5.1 BWT-matrices for the words $w = abbaba$, $\varphi(w)$, and $\tilde{\varphi}(w)$ respectively. 75

6.1 Application of Lemma 109 to compute LCP[LF(q)] by extending the result of the last LCE query. 90

6.2 Human chromosome 19 and SARS-CoV2 genomes dataset construction CPU time and peak memory usage. 93

8.1 Circular representation of the standard Sturmian words *aba.ba.ababaababaababa.ab* and *ababaababaababa.ab.aba.ba*. 117

8.2 Matrix of the sorted rotations for the finite Fibonacci word $f_6 =$ `abaababaabaab`. The underlined positions correspond to the positions of a circular string attractor $\Gamma_c = \{12, 13\}$. We underline in the other rotations the corresponding position 119

8.3 Running example of Algorithm 1 on the word $f_5 = \texttt{abaababa}$. The underlined positions in f_5 correspond to the positions of the circular string attractor Γ_c, and the corresponding succ_c and c-LCP arrays are shown right below. Each subfigure shows one of the 8 iterations of the algorithm, and the stack S at the end of the iteration. The dashed boxes surround the prefixes of the current rotation for which we have not found an occurrence crossing a position in Γ_c. The lengths of these prefixes correspond to the ranges in S. 124

8.4 Running example of Algorithm 2 on the word $f' = \texttt{ababaaba\$}$. The underlined positions in f' correspond to the positions of a set Γ that we want to check whether it is a string attractor for f'. The corresponding succ and LCP arrays are shown right below. Each subfigure shows one of the 4 iterations of the algorithm, and the status of the stack S at the end of the iteration. The dashed boxes surround the prefixes of the current rotation for which we have not found an occurrence crossing a position in Γ. The lengths of these prefixes correspond to the ranges in S. 127

9.1 Factor complexity function $p_{\mathbf{x}}$, recurrence, and string attractor profile function $s_{\mathbf{x}}$ for some infinite words. 132

10.1 Case 1 in the proof of Proposition 200. 151

10.2 Case 2 in the proof of Proposition 200 with $v = \prod_{i=3}^{k} b_{n-k+1-i}$. 151

10.3 The automaton $\mathcal{A}_{\mu_c}$ for $c = 102$ 159

10.4 Representation of the proof of the second claim of Theorem 235. As we warned the reader before, elements in a string attractor are indexed starting at 1 (in red), while indices of letters in $\mathbf{u}$ start at 0. 165

List of Tables

3.1 Scheme of the BWT-matrix of a word w_k with $k > 5$. 38

3.2 BWTs of the words w_k and its variants after an insertion, a deletion, and a substitution respectively. 44

6.1 Example of extended matching statistics for the pattern $P = AACCTAA$ with respect to the text $T = ACACTCTTACACCATATCATCAA\$$. . . 87

6.2 Dataset used in the experiments with MUM-PHINDER. 91

9.1 For $n \in [1, 8]$, the length-n prefix of the Fibonacci word $\mathbf{x} = abaababaabaab \cdots$ and its leftmost string attractor Γ_n. 141

10.1 The first few finite Tribonacci words $(b_n^{(3)})_{0 \leq n \leq 5}$ (some particular decomposition is highlighted for a latter purpose, see Proposition 193). . 146

10.2 Construction of the sequences $(u_n)_{n \geq 0}$ and $(U_n)_{n \geq 0}$ for $c = 102$ 156

10.3 Illustration of the numeration system $\mathcal{S}_c$ for $c = 102$ 158

Chapter 1

Introduction

With the advent of the new millennium, new generation technologies forced us to confront with unpredictable challenges. To name a few contexts: collections for DNA sequencing, pangenomic datasets, astronomical images, web storage, etc. The global amount of these data produced increase year by year, at a speed rate that outperform the Moore's Law. Luckily for us, some of these domains of application are full of redundancies, which can be exploited to get compact representations of data. In the last two decades, the problem of accessing information directly from the compressed representation of words has been crucial. The field that deals with this problem took the name of compressed full-text indexing. Ideally, we would like to index words in a space close to their Kolmogorov complexity [75], that is the size of the smallest program returning them. However, the Kolmogorov complexity is not computable, and finding a universal index that properly works regardless of the representation adopted seems an unreachable task. On the other hand, the *Shannon entropy* [122] measures the average number of bits required to represent a letter from a source with a certain probability distribution. On finite words, if we suppose that the frequencies of the letters or factors in the word follow the probability distribution of a source, we can compute on the word the so called *empirical entropy*. With respect to the Kolmogorov complexity, the empirical entropy is achievable [63, 126], however different studies show that it is insensitive to long repetitions, ending up in a representation space of the same order as the length of the word [47, 100, 99].

To overcome this problem, over the years different approaches have been attempted, but without any doubts there are very few compressors as efficient as the

family of *dictionary based compressors* [124]. The underlying common idea is the following: the compressed text is no longer a concatenation of letters that takes (on average) less space, but instead the text is represented through pointers to substrings, likely to contain *redundant information*. Indeed, the more repetitive the text is, the more likely is to represent words in a compact space. For very repetitive strings, a variety of compressors have shown up remarkable results in terms of compression ratio [81, 124, 70, 69, 102]. Moreover, for most of the cases it is possible to index directly the compressed text without decompressing it. The efficiency of these schemes on repetitive datasets has motivated considering them as *measures of repetitiveness* [99]. A theoretical study of these measures is then become necessary, to understand what properties of words affect them, and how these measures are related.

Out of all the schemes proposed, the *Burrows-Wheeler Transform* [21], in short BWT, has plenty of applications in data compression and bioinformatics, and it is at the basis of some of the most used tools in the fields [79, 121, 82, 43, 49]. The BWT is a reversible permutation of the word which *boosts* the performance of 0th-order entropy compressors. For instance, the run length encoding tends to be more effective on the BWT with respect to the word when it is repetitive. The number of equal-letter runs of the BWT induced from the run length encoding, that we call BWT-runs, is usually denoted by r. What makes the BWT so appreciated is the fact that it can be computed in linear time, and recently it has been proved that it is possible to index the text in a space proportional to r [110].

On the other hand, the very recent notion of *string attractor* has been introduced in the field of data compression as a unifying framework for all dictionary based compressors. A string attractor consists in a set Γ of γ positions of a word such that each distinct factors has at least an occurrence crossing a position in Γ, and by γ^* we denote the size of a smallest string attractor of a word. The measure γ^* has been proved to be asymptotically smaller than the size of any dictionary-compressor based measure [66, 5], since its definition is intrinsically the basic concept underlying all these schemes. This, along with the fact that it is possible to index words in $\mathcal{O}(\gamma \log \frac{n}{\gamma})$ words of space [31], makes the string attractor and the measure γ^* very appealing in the field of compressed full text indexing. In general however, finding

γ^* is an NP-complete problem [66, 68].

For the work of this thesis, we evaluate the measure r of the number of Burrows-Wheeler transform and the size γ^* of a smallest string attractor of a word γ^* [66], as measures of repetitiveness.

On the Burrows-Wheeler transform we obtain novel bounds related to particular substrings that occur in the original word. Then, we show that in the worst-case scenario, the measure r is sensitive when small edits are applied to words, and we test its efficiency on families of words containing very long repetitions, namely *purely morphic words*. Moreover, we show how we can use properties of the BWT to extend the functionality of compressed text indexes in a space proportional to r, mainly with applications in settings where the data are highly redundant (e.g., in pangenomics).

On string attractors, we present the first combinatorial study on the measure γ^*. We introduce new measures related to string attractors, which takes into account the *density* of information within words. We further introduce the circular variant of string attractors, and present a novel characterization for powers of standard Sturmian words. Its introduction is followed by the first algorithm that checks whether a set is a circular string attractor of a word. Finally, we extend the measures based on string attractors to infinite words, and show new characterizations of words *captured* by these new string attractor based complexities. Extending the comprehension of such measures and complexities lead to interesting applications in the field of Combinatorics on Words.

1.1 Contributions of the Thesis

Most of the material of this thesis are contained from seven papers, listed here in chronological order of publication.

[90] Sabrina Mantaci, Antonio Restivo, Giuseppe Romana, Giovanna Rosone, and Marinella Sciortino. *A Combinatorial View on String Attractors*. Theoretical Computer Science 850, pp. 236-248, 2021.

[45] Andrea Frosini, Ilaria Mancini, Simone Rinaldi, Giuseppe Romana, and Marinella Sciortino. *Burrows-Wheeler Transform on Purely Morphic Words*. In Data Compression Conferenc (DCC 2022), Poster. IEEE, 2022.

[46] Andrea Frosini, Ilaria Mancini, Simone Rinaldi, Giuseppe Romana, and Marinella Sciortino. *Logarithmic Equal-Letter Runs for BWT of Purely Morphic Words*. In International Conference on Developments in Language Theory (DLT 2022), pp. 139-151. Springer, 2022.

[57] Sara Giuliani, Giuseppe Romana, and Massimiliano Rossi *Computing Maximal Unique Matches with the r-Index*. In International Symposium on Experimental Algorithms (SEA 2022), pp. 22:1-22:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[112] Antonio Restivo, Giuseppe Romana, and Marinella Sciortino. *String Attractors and Infinite Words*. In Latin American Theoretical Informatics Symposium (LATIN 2022), pp. 426-442, Springer, 2022.

[44] Gabriele Fici, Giuseppe Romana, Marinella Sciortino, and Cristian Urbina. *On the Impact of Morphisms on BWT-Runs*. In Annual Symposium on Combinatorial Pattern Matching (CPM 2023). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023 (accepted, to be published).

[55] Sara Giuliani, Shunsuke Inenaga, Zsuzsanna Lipták, Giuseppe Romana, Marinella Sciortino, and Cristian Urbina: *Bit catastrophes for the Burrows-Wheeler Transform*. In International Conference on Developments in Language Theory (DLT 2023), pp. 86–99. Springer, 2023.

[53] France Gheeraert, Giuseppe Romana, Manon Stipulanti. *String attractors of fixed points of k-bonacci-like morphisms*. In Combinatorics on Words (WORDS 2023), pp. 192–205. Springer, 2023.

Additional unpublished content can be found in Sections 6.2, 8.2, and 8.3.

1.2 Outline

We assume the reader to be familiar with fundamentals in Computability, Logic, and Set Theory. In Chapter 2, we introduce the basic concepts on words and repetitiveness, that will be used throughout the thesis. For this aim, here we provide basic definitions and notations from the Combinatorics on Words field, in addition with some data structures from the full-text indexing world. Moreover, we introduce to the reader different interpretations of repetitiveness known in literature.

In Chapter 3, we present a qualitative study of the measure r. We will show the role that some *special* factors of words play in the size of the BWT-runs, and test its *sensitivity* to the effect of simple edit operations.

A *morphism*, from the Formal Language Theory, can be seen as a mathematical tool to extend repetitions of words. In Chapter 4, we explore the size r on the family of *purely morphic finite words*. For the case of binary purely morphic words, we show both upper and lower bounds on r, which are tight for most of the cases. Then, in Chapter 5 we deal with the analogous problem from a different perspective, that is the response of the BWT to the application of morphisms. This has resulted in a novel characterization of the well-known family of *Sturmian morphisms*.

We conclude the study of the measure r in Chapter 6, by showing properties of the Burrows-Wheeler Transform that can be used for practical applications in *pangenomics*, and more in general in bioinformatics. First, we deal with the problem of building a compact index in a space proportional to r able to locate *maximal unique matches*. Then, we are interested in the efficiency of the BWT for storing collections of samples. The common goal of both contributions is indeed to make accessible data and algorithms to researchers with limited space resources.

The notion of *string attractor* has been given very recently in the Data Compression field to define a unifying framework for dictionary-based compressors. The size γ^* of a smallest string attractor is asymptotically smaller than most of dictionary-based compressor schemes, but its definition can be easily seen as object of study in Combinatorics on Words. In Chapter 7, we study the notion of string attractor from different perspectives. First we observe how the measure γ behaves when classical combinatorial operations are applied. We prove that the measure is not *monotone*

with respect to the concatenation, and that, unlike r, the size γ^* may differ for rotations of the same word. Then, we introduce two novel measures related to string attractors: the *span measure* and the *leftmost measure*. These measures spot combinatorial properties of words that the measure γ^* by itself would miss otherwise.

Given the vulnerability of γ^* with respect to rotations, In Chapter 8 we introduce the *circular string attractor*, which aims to consider the factors crossing two consecutive occurrences of the same word as exploitable information to represent the text. Here we analogously define the *circular span*, and show that a bound on this measure can be used to characterize *standard Sturmian words*. Moreover, we provide the first algorithm for checking if a set is a circular string attractor of a word. From an equivalence shown later, this induce a novel algorithm for checking if a set is a string attractor, based on suffix arrays and longest common prefix array.

In Chapter 9 we extend the link between string attractors and classical notions of repetitiveness from the Combinatorics on Words field. More in detail, we extend the literature on the *string attractor profile function* of infinite words. On infinite words, we introduce new complexities based on the leftmost and the span measure, and show that the boundedness of these functions characterize *eventually periodic words* and *quasi-Sturmian words*, respectively. Finally, we continue on Chapter 10 by considering the case study of fixed point of k-bonacci like morphisms. The way the results are obtained wants to introduce a new perspective on the link between the computation of string attractors for a wide family of repetitive words by using notions from the enumeration system world.

Chapter 2

Preliminaries

Throughout the thesis, we use a standard mathematical notation. Here, we list the notions that are assumed to be known in the next chapters. First, we present the basic terminology that will be used throughout the paper, along with well known properties and definitions in combinatorics on words. Then, we introduce concepts and measures related to repetitiveness of words.

2.1 Finite and Infinite Words

Finite Words An *alphabet* Σ is a set of elements called *letters* (or *characters*). If the size of Σ is finite, then we denote by $\sigma = |\Sigma|$ its cardinality. A *word* (or *string*) w is a sequence of letters from an alphabet Σ, and its *length* is denoted by $|w|$. We denote with ε the *empty word*, that is the only word such that $|\varepsilon| = 0$. We denote by Σ^* the set of all words over Σ, by $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$ the set of all non-empty words over Σ, and by $\Sigma^n = \{w \in \Sigma^* \mid |w| = n\}$ the set of all words of length n, for some $n \in \mathbb{N}_0$. We denote by $\mathsf{alph}(w) \subseteq \Sigma$ the set of letters from Σ that occurs in w.

Factors of Words For any pair $i, j \in [1, n]$, we denote by $w[i, j]$, the *factor* (or *substring*) of w starting at positions i and ending at positions j. Further, if $i = 1$ or $j = n$, then $w[i, j]$ is called *prefix* or *suffix* respectively. Note that $w[1, n] = w$, and if $i > j$, we assume that $w[i, j] = \varepsilon$. The set of factors of a word w is denoted by $F(w)$. We say that a factor $w[i, j]$ is *proper* if either $i \neq 1$ or $j \neq n$, i.e. when $w[i, j] \neq w$.

Concatenation of Words Given two words $u, v \in \Sigma^*$, if $u = u[1]u[2] \cdots u[|u|]$ and $v = v[1]v[2] \cdots v[|v|]$, the *concatenation* of u and v, denoted by $u \cdot v$ or simply

uv, is the word $u[1]u[2]\cdots u[|u|]\cdot v[1]v[2]\cdots v[|v|]$. Given an indexed set words $C = \{w_1, w_2, \ldots, w_m\}$, we denote by $\prod_{k=1}^{m} w_k = w_1 w_2 \cdots w_m$ the concatenation of the words from C taken in order.

Reverse Given a word $w \in \Sigma^n$, we denote by w^R the *reverse* of w, that is the word w spelled backwards from right to left. In other words, if $w = w[1]w[2]\cdots w[n]$, then $w^R = w[n]w[n-1]\cdots w[1]$.

Special Factors A factor u of w is *left special* (*right special*) if there exist $a, b \in \Sigma$ with $a \neq b$ such that both au and bu (ua and ub) are factors of w. A factor u is *bispecial* if it is both left and right special. Inspired by the notation in [22], we denote by $e_r(u) = |\{a \in \Sigma \mid ua \in F(w)\}| - 1$ the *number of right extensions of u in w* minus 1, and by $e_\ell(u) = |\{a \in \Sigma \mid au \in F(w)\}| - 1$ the *number of left extensions of u in w* minus 1. The bispecial factors of w can be classified according to the number of their extensions. In particular, a factor u is *strictly bispecial* if $|F(w) \cap \Sigma u \Sigma| = (e_r(u)+1)(e_\ell(u)+1)$, u is *weakly bispecial* if $|F(w) \cap \Sigma u \Sigma| = \max\{e_r(u), e_\ell(u)\} + 1$. We denote by $SBS(w)$ and $WBS(w)$ the set of strictly and weakly bispecial factors of w, respectively.

Order Relations of Words If on the alphabet is defined a total order among its elements, then we can induce different orders on words. Given two words $u, v \in \Sigma^*$, we say that $u \leq v$ if u is prefix of v, or there exists a common prefix w, the suffixes $u', v' \in \Sigma^*$ of u and v respectively, and two letters $a < b \in \Sigma$, such that $u = wau'$ and $v = wbv'$. We refer to this order of the words as *lexicographical order*.

The *colexicographical order* is defined by $u \leq_{\mathsf{colex}} v$ if $u^R \leq v^R$.

The *genealogical order*, also known as *radix order*, on Σ^* induced by $\leq$ is defined as follows: for two words $u, v \in \Sigma^*$, we say that u is *genealogically smaller than v*, and we write $u <_{\text{gen}} v$, if either $|u| < |v|$, or $|u| = |v|$ and $u = wau'$ and $v = wbv'$ for some letters a, b with $a < b$. We write again $u \leq_{\text{gen}} v$ if u is genealogically smaller than or equal to v.

Power of Words and Periods For some $n \geq 0$, we denote by w^n the concatenation of w with itself n times. A finite word w is called *primitive* if $w = u^p$ if and only if $w = u$ and $p = 1$. On the other hand, we say that w is a *power* of a word u if there

exists an integer $p > 1$ such that $w = u^p$, and we call u *root* of w. For all $p > 0$ such that $w[i] = w[i+p]$ for all $i \in [1, |w| - p]$, we say that p is a *period* of w. In this case, there exists a rational $q > 1$ such that $w = u^q$, with $q = \frac{k}{|u|}$ for some $k > |u|$, and w is called *fractional power*. Note that a fractional power w is a power when k is a multiple of $|u|$, otherwise it is primitive.

Rotations of Words Given two words $w_1, w_2 \in \Sigma^*$, we say that w_1 is a *conjugate* (or *cyclic rotation*, or simply *rotation*) of w_2 if there exist $u, v \in \Sigma^*$ such that $w_1 = uv$ and $w_2 = vu$. We denote by $\mathcal{R}(w)$ the set of rotations of the word w. Note that $\mathcal{R}(w_1) = \mathcal{R}(w_2)$ if and only if w_1 and w_2 are respectively conjugate. A finite word w is then *primitive* if and only if $|\mathcal{R}(w)| = |w|$, i.e., when all rotations of w are distinct. A primitive word w is called *Lyndon* if it is the smallest in lexicographical order among its rotations.

Circular Factors of Words Given a word w, we refer by $\mathcal{C}(w) = \bigcup_{w' \in \mathcal{R}(w)} F(w')$ to the set of *circular factors* of w, that is the set of factors of all rotations of w. A circular factor $u \in \mathcal{C}(w)$ can be denoted by a pair of positions (i, j) in w, where $(i, j) = w[i, j]$, if $i \leq j$, or $(i, j) = w[i, n] \cdots w[1, j]$ otherwise. All the definitions related to bispecial factors can be extended to circular factors by considering $\mathcal{C}(w)$ instead of $F(w)$. We denote by $cSBS(w)$ and $cWBS(w)$ the set of strictly and weakly bispecial factors of w, respectively.

Suffix Array, Inverse Suffix Array, and Longest Common prefix array The *Suffix array* (SA) of a word $w \in \Sigma^n$ is an array of length n such that $w[\mathsf{SA}[i], n] < w[\mathsf{SA}[j], n]$ for any $1 \leq i < j \leq n$. The *Inverse Suffix array* (ISA) is the inverse of SA, i.e. $\mathsf{ISA}[i] = j$ if and only if $\mathsf{SA}[j] = i$.

Given two words $u, v \in \Sigma^*$, let $\ell cp(u, v)$ be the the longest common prefix between u and v, that is $\ell cp(u, v) = u[1, |\ell cp(u, v)|] = v[1, |\ell cp(u, v)|]$, but $u[|\ell cp(u, v)| + 1] \neq v[|\ell cp(u, v)| + 1]$ (assuming $\ell cp(u, v) < \min\{|u|, |v|\}$). The *Longest Common Prefix array* (LCP) of $w \in \Sigma^n$ is an array of length n such that $\mathsf{LCP}[1] = 0$ and $\mathsf{LCP}[i] = |\ell cp(w[\mathsf{SA}[i-1], n], w[\mathsf{SA}[i], n))|$, for any $0 < i < n$.

Distances Given two words u and v having the same length, the *Hamming distance* between u and v, denoted as $d_H(u, v)$, is the number of positions at which the corresponding letters in u and v are different.

Given a word $w \in \Sigma^n$, an index $p \geq 0$, and a letter $a \in \Sigma$, we refer with *edit operation* to any of the following operations on words:

- *insertion*: add the letter a in w at position p, that is

$$ins(w, p, a) = w[1, p] \cdot a \cdot w[p + 1, n];$$

- *deletion*: delete the letter in w at position p, that is

$$del(w, p) = w[1, p - 1] \cdot w[p + 1, n];$$

- *substitution*: replace the letter in position p in w with a, that is

$$sub(w, p, a) = w[1, p - 1] \cdot a \cdot w[p + 1, n].$$

Given two words $u, v \in \Sigma^*$, the *edit distance* between u and v is the minimum number of edit operations to apply on u to obtain v and vice versa, and it is denoted by $\mathsf{ed}(u, v)$.

Example 1. *Let us consider the words* $u = \texttt{abbabb}$ *and* $v = \texttt{babbab}$. *One can see that* $d_H(u, v) = 4$, *since* u *and* v *differ at positions* 1, 2, 4, *and* 6. *On the other hand, one can see that* $u' = del(u, 6) = \texttt{abbab}$ *and* $ins(u', 1, \texttt{b}) = \texttt{babbab} = v$. *By exhaustive research, one can check that this is the minimum number of edit operations needed to go from* u *to* v, *and therefore* $\mathsf{ed}(u, v) = 2$.

Infinite Words An *infinite word* $\mathbf{x}$ on the alphabet Σ is an infinite sequence of letters from Σ, that we can denote by $\mathbf{x} = a_1 a_2 \cdots a_n \cdots$, where $a_i \in \Sigma$ for all integer $i > 0$. We denote by Σ^ω the set of all infinite words on Σ. An infinite word $\mathbf{x}$ is *periodic* if there exists $p > 0$ such that $\mathbf{x}[i] = \mathbf{x}[i + p]$, for all $i > 0$. It follows that there exists $v \in \Sigma^p$ such that $\mathbf{x} = v^\omega$, where v^ω denotes the concatenation of infinite copies of v. An infinite word $\mathbf{x}$ is called *eventually periodic* if there exist finite words $u, v \in \Sigma^*$

such that $\mathbf{x} = uv^\omega$. If $\mathbf{x}$ is neither eventually periodic nor periodic, then $\mathbf{x}$ is called *aperiodic*.

Recurrence and appearance functions An infinite word $\mathbf{x}$ is said to be *recurrent* if every factor that occurs in $\mathbf{x}$ occurs infinitely often in $\mathbf{x}$, or equivalently that each factor of $\mathbf{x}$ appear at least two times. The *recurrence function* $R_x(n)$ gives, for each n, the least integer m (or ∞ if no such m exists) such that every block of m consecutive letters in x contains at least an occurrence of each factor of x of length n. An infinite word $\mathbf{x}$ is *uniformly recurrent* if $R_\mathbf{x}(n) < \infty$ for each $n > 0$. $R_\mathbf{x}(n) - n + 1$ is the maximum gap between successive occurrences of any factor, when all factors of length n are considered. If $R_\mathbf{x}(n)$ is linear, $\mathbf{x}$ is called *linearly recurrent*. It is easy to see that an eventually periodic word $\mathbf{x} = uv^\omega$ that is not periodic it is not recurrent. On the other hand, if $\mathbf{x}$ is periodic (the case $u = \varepsilon$) then $\mathbf{x}$ is linearly recurrent. Therefore, a recurrent word is either aperiodic or periodic. Given an infinite word $\mathbf{x}$, $A_\mathbf{x}(n)$ denotes the length of the shortest prefix containing all the factors of $\mathbf{x}$ of length n. The function $A_\mathbf{x}(n)$ is called *appearance function* of $\mathbf{x}$.

Remark 2. *It is known that $A_\mathbf{x}(n) \leq R_x(n)$ (see [2]). Moreover, for any infinite word $\mathbf{x}$ and for each $n > 0$, since $|\Sigma|$ is finite, $A_\mathbf{x}(n)$ is always defined and $A_\mathbf{x}(n) \geq p_\mathbf{x}(n) + n - 1 = \Omega(n)$.*

k-power free and ω-power free words Given an integer $k > 1$, an infinite word $\mathbf{x} \in \Sigma^\omega$ is called *k-power free* if, for all $u \in \Sigma^*$, it holds that $u^k \notin F(\mathbf{x})$, that is $\mathbf{x}$ does not contain k consecutive occurrences of any factor. An infinite word $\mathbf{x} \in \Sigma^\omega$ is called *ω-power free* if, for each factor $u \in \Sigma^*$, there exists $k \geq 1$ such that $u^k \notin F\mathbf{x}$.

Sturmian Words Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. A word $w \in \Sigma^*$ is called *balanced* if the difference of the number of a's (or, equivalently, b's) in every two factors of the same length of w is at most 1. An infinite word $\mathbf{x}$ is balanced if every finite factor of $\mathbf{x}$ is balanced. A finite word w is *circularly balanced* if each word in $\mathcal{R}(w)$ is balanced.

An infinite word over $\Sigma = \{a, b\}$ is a *Sturmian word* if it has exactly $n + 1$ distinct factors of length n for every $n \geq 0$. The theory of Sturmian words is very well

studied (see [84] for a reference). For example, the following characterization is well known.

Theorem 3. *An infinite word over $\Sigma = \{\mathtt{a}, \mathtt{b}\}$ is Sturmian if and only if it is balanced and aperiodic.*

A class of Sturmian words, called *characteristic Sturmian words*, can be constructed by using finite words, called *standard Sturmian words*, defined recursively as follows. Given an infinite sequence of integers $(d_0, d_1, d_2, \ldots)$, with $d_0 \geq 0, d_i > 0$ for all $i > 0$, called *directive sequence*, the associated standard Sturmian words are defined by $s_0 = b$, $s_1 = a$, and $s_{i+1} = s_i^{d_{i-1}} s_{i-1}$, for $i \geq 1$. A characteristic Sturmian word is the limit of an infinite sequence of standard Sturmian words, i.e., $s = \lim_{i \to \infty} s_i$. A very well-known example of characteristic Sturmian word is the *infinite Fibonacci word*, obtained from the directive sequence $(1, 1, 1, \ldots)$, that is the word

$$f = \mathtt{abaababaabaababaababa} \cdots .$$

We refer to standard Sturmian prefixes s_i of f as the *finite Fibonacci words*. Note that in this case, the size $|s_i|$ grows as the sequence of Fibonacci numbers.

We denote by *Stand* the set of all standard Sturmian words. Standard Sturmian words appear as extremal case for several algorithms and data structures ([71, 23, 94, 120]).

For instance, a characterization of (rotations of) powers of standard Sturmian words can be derived from their circular balance [113]

Lemma 4. *Let w be a finite word. Then, $w \in \mathcal{R}(s^\ell)$ for some $s \in$ Stand and some $\ell > 0$ if and only if w is circularly balanced.*

Let PER be the set of all words $v \in \{\mathtt{a}, \mathtt{b}\}^*$ having two periods p and q such that $gcd(p, q) = 1$, and $|v| = p + q - 2$. It is known that $Stand = \{a, b\} \cup PER\{ab, ba\}$ [86]. Given a word $w \in Stand$, we denote by $\pi(w)$ its prefix of length $|w| - 2$, belonging to the set PER, uniquely defined by using previous equality. By using a property of words in PER (cf. [86]), $\pi(w) = QxyP = PyxQ$, where $x \neq y$ are letters and Q and P are uniquely determined palindromes. So, a standard Sturmian word $w = \pi(w)ba$

can be decomposed as $w = QxyPba = PyxQba$. Any factorization of this type is called *PER-decomposition* of w.

2.2 Morphisms

In Formal Language theory, a *homomorphism*, frequently simply referred as *morphism*, is an algebraic tool that is frequently used as an automatic system to generate words.

Given two alphabets Σ and Σ', a morphism $\mu : \Sigma^* \mapsto \Sigma'^*$ maps the words over the alphabet Σ to words over the alphabet Σ', and that preserves the concatenation, that is $\mu(uv) = \mu(u)\mu(v)$ for all $u, v \in \Sigma^*$. For any word $w \in \Sigma$, we refer to $\mu(w)$ as the *image of w under μ* (simply *image of w* if μ is clear from the context). If it is defined an order within the letters of $\Sigma = \{a_1 < a_2 < \ldots < a_\sigma\}$, then we can define the morphism with the notation $\mu \equiv (u_1, u_2, \ldots, u_\sigma)$, where $u_i = \mu(a_i) \in \Sigma'^*$ for all $i \in [1, \sigma]$.

A morphism $\mu : \Sigma^* \mapsto \Sigma^*$ is called *primitive* if there exists k such that, for each pair of letters $a, b \in \Sigma$, it holds that $b \in F(\mu^k(a))$, that is b appear in $\mu^k(a)$.

A morphism $\mu : \Sigma^* \mapsto \Sigma'^*$ is called *non-erasing* if $\mu(a) \neq \varepsilon$ for all $a \in \Sigma$. If $|\mu(a)| = k$ for all $a \in \Sigma$ and some integer $k > 0$, then the morphism μ is called *k-uniform*. If a morphism μ is 1-uniform, then μ is also called a *coding*.

A morphism $\mu : \Sigma^* \mapsto \Sigma^*$ is called *elementary* if it does not exists a composition of two morphisms $\eta_1 : \Sigma^* \mapsto \Sigma'^*$ and $\eta_2 : \Sigma'^* \mapsto \Sigma^*$ with $|\Sigma| > |\Sigma'|$ such that $\mu = \eta_2 \eta \eta_1$. A morphism that is not elementary is called *simplifiable*.

Growth Function of Morphisms Given a morphism $\mu : \Sigma^* \mapsto \Sigma^*$ and a letter $a \in \Sigma$, the *growth function* μ_a returns for each $n \geq 0$ the length of the word obtained after applying n times μ on a, i.e., $\mu_a(n) = |\mu^n(a)|$. A letter $a \in \Sigma$ is *growing* for μ if $\lim_{n \to +\infty} \mu_a(n) = +\infty$. On the contrary, a is *bounded* for μ if there exists $k > 0$ such that $\mu_a(n) < k$ for all $n > 0$. A morphism μ is called *growing* if all letters Σ are growing for μ, and it is called *non-growing* if there exists $a \in \Sigma$ that is bounded for μ. Clearly, if μ is a coding then it is non-growing, while if μ is k-uniform for some $k \geq 2$ then $\mu_a(n) = k^n$ for all $n \geq 0$, and therefore it is growing.

It is known that for growing morphisms, for all $a \in \Sigma$ the growth function is $\mu_a(n) = \Theta(n^{e_a}\rho_a^n)$, for some $e_a \geq 0$ and some $\rho_a > 1$ [118, 117]. Growing morphisms can be classified based on their growth functions. A growing morphism $\mu : \Sigma \mapsto \Sigma$ is called:

1. *quasi-uniform* if, for all $a \in \Sigma$, there exists $\rho > 1$ such that $\mu_a(n) = \Theta(\rho^n)$;
2. *polynomially divergent* if there exist $a, b \in \Sigma$ such that $\mu_a(n) = \Theta(n^{e_a}\rho^n)$ and $\mu_b(n) = \Theta(n^{e_b}\rho^n)$, for some $\rho > 1$ and two distinct $e_a \neq e_b \geq 0$;
3. *exponentially divergent* if there exist $a, b \in \Sigma$ such that $\mu_a(n) = \Theta(n^{e_a}\rho_a^n)$ and $\mu_b(n) = \Theta(n^{e_b}\rho_b^n)$, for some $e_a, e_b \geq 0$ and two distinct $\rho_a \neq \rho_b > 1$.

It is known that any primitive morphism is quasi-uniform.

Purely morphic words Given a morphism $\mu : \Sigma \mapsto \Sigma$, a *fixed-point* of μ is any word w that verifies $\mu(w) = w$. A morphism $\mu : \Sigma \mapsto \Sigma$ is called *prolongable* on $a \in \Sigma$ if $\mu(a) = au$, for some $u \in \Sigma^+$. From the definition, one can see that if μ is prolongable on a, then $\mu^n(a) = au\mu(u)\mu^2(u)\cdots\mu^{n-1}(u)$ for all n, and therefore each $\mu^n(a)$ is prefix of $\mu^{n+1}(a)$. Then, there exists an infinite word $\mathbf{x}$ fixed-point of μ obtained after applying the morphism infinitely many times, i.e., $\lim_{n\to\infty} \mu^n(a) = \mu(\mathbf{x}) = \mathbf{x}$, and it is called *purely morphic word*. Further, a word w is called *purely morphic finite word* if $w = \mu^n(a)$ for some $n > 0$.

Example 5. *Let $\tau \equiv (\mathtt{ab}, \mathtt{ba})$ denote the Thue-Morse morphism. The first iterations of the morphism τ, and the purely morphic word $\mathbf{t} = \tau^\infty(\mathtt{a})$ are shown below.*

$$\begin{aligned}
\tau(\mathtt{a}) &= \mathtt{ab} \\
\tau^2(\mathtt{a}) &= \mathtt{abba} \\
\tau^3(\mathtt{a}) &= \mathtt{abbabaab} \\
\tau^4(\mathtt{a}) &= \mathtt{abbabaabbaababba} \\
&\vdots \\
\mathbf{t} = \tau^\infty(\mathtt{a}) &= \mathtt{abbabaabbaababbabaababbaabbabaab}\cdots
\end{aligned}$$

Acyclic and Cyclic Morphisms A morphism $\mu : \Sigma^* \mapsto \Sigma'^*$ is *cyclic* if there exists $u \in \Sigma'^*$ such that $\mu(a) \in \{u\}^*$, for each $a \in \Sigma$. Otherwise, it is called *acyclic*. Note that

the fixed point of a cyclic morphism is periodic. In the case of a binary morphism, it is known that μ is cyclic if and only if $\mu(\texttt{ab}) = \mu(\texttt{ba})$.

Sturmian morphisms A *Sturmian morphism* is a morphism that maps infinite Sturmian words to infinite Sturmian words. Some combinatorial characterizations of Sturmian morphisms have been proved in [11]. In particular, a binary morphism μ is Sturmian if and only if it is acyclic and *balanced* (i.e., it maps balanced words to balanced words). Berstel and Séébold [11] proved the following characterization:

Theorem 6. *An acyclic morphism μ is Sturmian if and only if it is locally Sturmian, that is, there exists a Sturmian word s such that $\mu(s)$ is Sturmian.*

Let us denote the following morphisms:

$$E : \begin{cases} a & \mapsto & b \\ b & \mapsto & a \end{cases} \qquad \varphi : \begin{cases} a & \mapsto & ab \\ b & \mapsto & a \end{cases} \qquad \tilde{\varphi} : \begin{cases} a & \mapsto & ba \\ b & \mapsto & a \end{cases}$$

The morphism φ is called the *Fibonacci morphism*, since its fixed point is the Fibonacci word $abaababaabaababaab\cdots$. The monoid $\{E, \varphi, \tilde{\varphi}\}^*$ generated by E, φ, and $\tilde{\varphi}$, by using the composition operator $\circ$, is known as the *Sturm monoid*. The following theorem, proved in [97], shows the combinatorial structure of Sturmian morphisms.

Theorem 7. *A morphism is Sturmian if and only if it belongs to $\{E, \varphi, \tilde{\varphi}\}^*$.*

Sturmian morphisms can be used to characterize the rotations of standard Sturmian words, as described in the following lemma (see [106, 32]).

Lemma 8. *Let w be a finite word. Then, $w \in \mathcal{R}(s^\ell)$ for some $s \in Stand$ and some $\ell > 0$ if and only if there exists a Sturmian morphism $\mu \in \{E, \varphi, \tilde{\varphi}\}$ such that $w = (\mu(a))^\ell$.*

2.3 Repetitiveness of Words

Everyone has a vague idea of what repetitions are. Between the words `abababababab` and `bdacabbcdc`, instinctively one perceive that first is *more repetitive* than the second one. Even though different definitions related to redundancies in words have been given, still the scientific community has not agreed on a universal measure to quantify their degree of repetitiveness.

In this Section, we introduce different notions of repetitiveness given in the literature. First we present some notions from the Combinatorics on Words field. Then, we show some measures based on sizes of the compressed representations of the words, using different compression schemes.

2.3.1 Combinatorial Notions of Repetitiveness

Factor Complexity Given an infinite word $\mathbf{x} \in \Sigma^\omega$, the *factor complexity (function)* $p_\mathbf{x} : \mathbb{N} \mapsto \mathbb{N}$ of $\mathbf{x}$ counts for each integer $n > 0$, the number of distinct factors that occur in $\mathbf{x}$, that is:

$$p_\mathbf{x}(n) = |\Sigma^n \cap F(\mathbf{x})|.$$

If we assume that Σ is finite, such a function always exists. For all finite words $w \in \Sigma^*$, we denote by p_w the analogous function on finite words. The famous Morse-Hedlund theorem shows a characterization of eventually periodic words based on the factor complexity (see [84] for a reference).

Theorem 9 (Morse-Hedlund theorem)**.** *Let $\mathbf{x}$ be an infinite word. The following are equivalent.*

1. *The word $\mathbf{x}$ is eventually periodic.*
2. *We have $p_\mathbf{x}(n+1) = p_\mathbf{x}(n)$ for some integer n.*
3. *The complexity function $p_\mathbf{x}$ is bounded by a constant.*

From the point 2. of Theorem 9, one can observe that the factor complexity of an aperiodic word is at least linear, as described in the following Corollary.

Corollary 10. *An infinite word $\mathbf{x} \in \Sigma^\omega$ is aperiodic if and only if $p_\mathbf{x}(n) > n$.*

For a word $w \in \Sigma^n$, the *circular factor complexity* of w, denoted by c_w, counts for each $k \in [1, n]$ the number of distinct circular factors of w, that is $c_w(k) = |\mathcal{C}(w) \cap \Sigma^k|$.

Recurrence and Appearance Functions An infinite word $\mathbf{x}$ is said to be *recurrent* if every factor of $\mathbf{x}$ occurs infinitely often (in $\mathbf{x}$). The *recurrence function* $R_\mathbf{x}(n)$ returns, for each n, the least integer m (or ∞ if no such m exists) such that every block of m consecutive letters in $\mathbf{x}$ contains at least an occurrence of each length-n factor of $\mathbf{x}$.

An infinite word $\mathbf{x}$ is *uniformly recurrent* if $R_{\mathbf{x}}(n) < \infty$ for each $n \geq 1$. $R_{\mathbf{x}}(n) - n + 1$ is the maximum distance between consecutive occurrences of any factor, when all factors of length n are considered. If $R_{\mathbf{x}}(n)$ is linear, then $\mathbf{x}$ is *linearly recurrent*.

It is easy to see that an eventually periodic word $\mathbf{x} = uv^{\omega}$ with $u \neq \varepsilon$ is not recurrent, while on the other hand, if $\mathbf{x}$ is periodic (the case $u = \varepsilon$) then $\mathbf{x}$ is linearly recurrent. Therefore, a recurrent word is either aperiodic or periodic.

Given an infinite word $\mathbf{x}$, $A_{\mathbf{x}}(n)$ denotes the length of the shortest prefix containing all the length-n factors of x. The function $A_{\mathbf{x}}(n)$ is called the *appearance function* of $\mathbf{x}$.

Parikh Vectors Given a word $w \in \Sigma^*$, we denote by $|w|_a$ the number of occurrences in w of the letter $a \in \Sigma$. The *Parikh vector* of $w \in \{a_1, a_2, \ldots, a_\sigma\}^*$, denoted as $P(w)$, is the σ-tuple $(|w|_{a_1}, \ldots, |w|_{a_\sigma})$.

2.3.2 Compressor based Repetitiveness Measures

Grammars and Straight Line Programs A *(context-free) grammar* is defined through the tuple $G = (V, \Sigma, R, S)$, where:

1. V is the set of *nonterminal characters* (or *variables*);
2. Σ is the set of *terminal characters*;
3. $R : V \mapsto (V \cup \Sigma)^*$ is the set of rules, that is an application from the letters in V to words over terminal and/or nonterminal characters;
4. $S \in V$ is the *axiom*, or *initial variable*, to which we start applying rules from R.

We denote by $\mathcal{L}(G)$ the set of words in Σ^* that can be generated by G.

A *Straight Line Program* (SLP) [70] consists in a particular context-free grammar that generates only one string, that is $|\mathcal{L}(\mathsf{SLP})| = 1$. A SLP allows only two type of rules:

- $X \to AB$, with A, B *nonterminals*;
- $X \to a$, with a *terminal*.

If the grammar contains also some rules of the following type

- $X \to Y^k$, with Y *nonterminal* and $k > 2$

then we have a particular type of SLP, the *run-length SLP* (RLSLP) [102]. The problems of finding the smallest SLP or RLSLP of a word w, respectively of size g^* and g^*_{rl}, are NP-hard [26, 62, 69].

Based on the rules applied, we can build the associated *parse-tree*, as shown in the following example.

Example 11. *Consider the word*

$$w = abracadabra.$$

The following rules represent a SLP that refers to w:

- $S \to XY$
- $X \to X_1X_2$
- $X_1 \to AB$
- $X_2 \to RA$
- $Y \to Y_1X;$
- $Y_1 \to CY_2;$
- $Y_2 \to AD$
- $A \to a$
- $B \to b$
- $C \to c$
- $D \to d$
- $R \to r$

In Figure 2.1 it is reported the parse-tree for w.

Lempel-Ziv Factorization The *Lempel-Ziv Factorization* [81] consists in a factorization of a word $w \in \Sigma^n$ with a sequence of triples (s, l, c) where:

- $s \in [1, n]$ is the position where the longest common substring to be copied starts;
- $l \geq 0$ is the length of the longest common substring to be copied;
- $c \in \Sigma$ is the letter to concatenate with the longest common substring copied.

In other word, the i-th triple (s_i, l_i, c_i) tells us that the portion of string built so far until the i-th triple is obtained as follows:

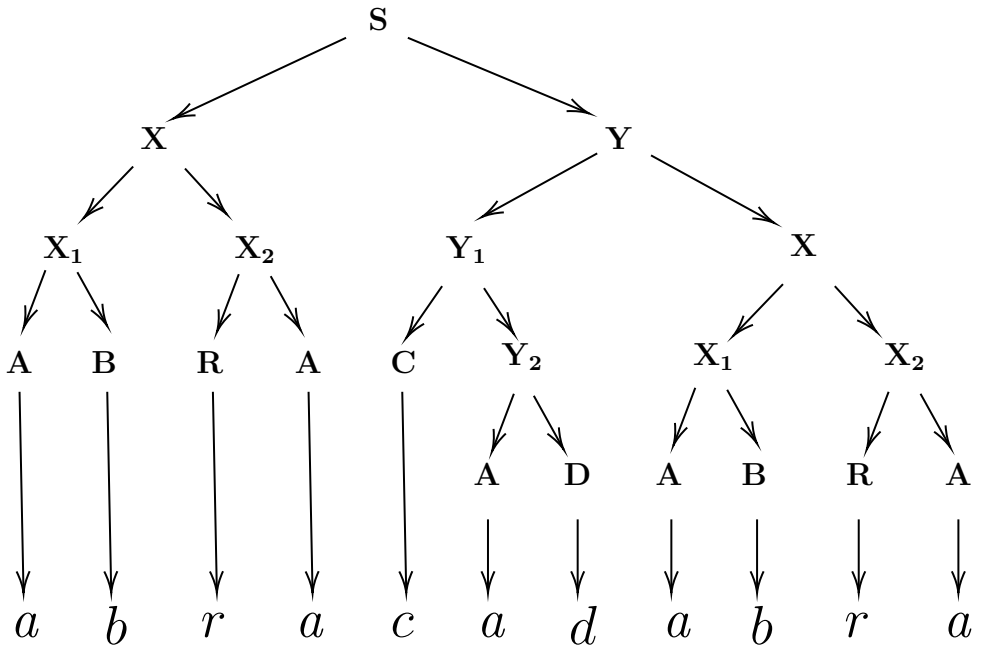


FIGURE 2.1: The parse-tree for a SLP generating the word $w = abracadabra$

$$w_i = w_{i-1} \cdot w_{i-1}[s_i, (s_i + l_i - 1)] \cdot c_i.$$

We denote by z the measure which computes the size of the Lempel-Ziv factorization of a word.

Example 12. *Consider $w = abracadabra$. By using the LZ-77 algorithm, we can replace the word by the sequence of triples*

$$LZ77(w) = (0,0,a)(0,0,b)(0,0,r)(1,1,c)(1,1,d)(1,3,a)$$

where:

- $(0,0,a) \to a$
- $(0,0,b) \to b$
- $(0,0,r) \to r$
- $(1,1,c) \to w[1,1] \cdot c = ac$
- $(1,1,d) \to w[1,1] \cdot d = ad$
- $(1,3,a) \to w[1,3] \cdot a = abra.$

It follows that $z(w) = |LZ(w)| = 6$.

δ-Measure Given a word $w \in \Sigma^n$, the measure δ strongly depends on the factor complexity p_w. Such a measure is in fact defined as $\delta(w) = \max_{k\in[1,n]}\{\frac{p_w(k)}{k}\}$ [111].

L-Systems, Macro Systems, and NU-Systems *Lindenmayer systems* (or *CD0L-systems*, or simply *L-systems*), originally developed by Lyndenmayer to model simple organisms [83], had a huge impact in the formal language field [117]. An L-system can be defined through a tuple $L = (\Sigma, \mu, w, \eta)$, where:

- Σ is an alphabet;
- $\mu : \Sigma^* \mapsto \Sigma^*$ is a morphism;
- w is the *axiom word* of the L-system;
- $\eta : \Sigma^* \mapsto \Sigma'^*$ is a coding.

The language of the words generated by L is $\mathcal{L}(L) = \{\eta(\mu^i(w)) \mid i \in \mathbb{N}\}$, and it is of size $\ell = \sum_{a\in\Sigma} |\mu(a)|$.

Example 13. *Let us consider the alphabet* $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, *the morphism* $\mu \equiv (\mathtt{ab}, \mathtt{bc}, \mathtt{cc})$, *and the coding* $\eta \equiv (\mathtt{a}, \mathtt{a}, \mathtt{b})$. *The first words of the L-system* $L = (\Sigma, \mu, \mathtt{a}, \eta)$ *are:*

$$\begin{aligned}
\eta(\mu(\mathtt{a})) &= \eta(\mathtt{ab}) &&= \mathtt{aa}\\
\eta(\mu^2(\mathtt{a})) &= \eta(\mathtt{abcb}) &&= \mathtt{aaba}\\
\eta(\mu^3(\mathtt{a})) &= \eta(\mathtt{abcbcccb}) &&= \mathtt{aababbba}\\
\eta(\mu^4(\mathtt{a})) &= \eta(\mathtt{abcbcccbcccccccb}) &&= \mathtt{aababbbabbbbbbba}\\
&\dots
\end{aligned}$$

The infinite word $\mathbf{c} = \eta(\mu^\infty(\mathtt{a}))$ *is also known as the* characteristic sequence of the power of 2, *since it verifies that* $\mathbf{c}[i] = \mathtt{a}$ *if and only if* $i = 2^j$ *for some* $j \in \mathbb{N}$.

A *macro system* [101] is a tuple $M = (V, \Sigma, R, S)$ where:

- V is the set of nonterminals or variables;
- Σ is the set of terminals;
- $R : V \mapsto (V \cup \Sigma \cup \{A[i,j] \mid A \in V, i, j \in \mathbb{N}\})^*$ is the set of rules that uniquely maps nonterminals into sequences of nonterminals and terminals;
- $S \in V$ is the axiom, or initial variable.

If a word w is uniquely generated by M by applying all the rules in R, then M has size $m = |\Sigma| + \sum_{A \in V} |R(A)|$.

From the combination of L-systems and macro systems, Navarro and Urbina introduced the notion of *NU-system* [101]. A NU-system is a tuple $N = (V, R, S, \lambda, d, n)$, where:

- V is the set of variables;
- $R : V \mapsto (V \cup E)^+$ is the set of rules that uniquely maps variables into sequences of variables and *extraction* rules, that are of the type

$$E = \{A(l)[i, j] \mid A \in V, l, i, j \in \mathbb{N}\};$$

- $S \in V$ is the axiom;
- $\lambda : V \mapsto V$ is a coding;
- d is the depth;
- n is the length of the word produced.

Note that $A(l)$ denotes the word obtained after applying l times the rules from R in parallel on each letter. If N is the smallest NU-system that generates a word $w = S(d)[1, n]$, we denote its size by $\nu = \sum_{A \in V} |R(A)|$. From the definition, which combines both L and macro systems, it can be derived that $\nu = \mathcal{O}(\min(\ell, m))$. Moreover, the measures ν and δ are incomparable [101].

Chapter 3

Burrows-Wheeler Transform, BWT-runs, and Measure r

The Burrows-Wheeler Transform, named after its designers Michael Burrows and David Wheeler, is a reversible permutation of strings, defined in the field of Data Compression [21]. Since its introduction, the Burrows-Wheeler Transform (BWT) has been a hot topic for the scientific community. Such a permutation tends to be more *compressible* with standard techniques: on repetitive texts, 0th-order entropy compressors on the BWT are efficient as higher-order entropy compressors on the original word [95]. A standard compression technique to apply on the BWT is the *run-length encoding*, which consists in replacing maximal equal letter runs of the BWT, called *BWT-runs*, in pairs of the form (a, ℓ), where a is the letter repeated and $\ell > 0$ is the length of the BWT-run. The size of the run-length encoding of the BWT is usually referred by r, and this measure tends to be smaller than the run-length encoding of the word itself when abundant and long repetitions occur in the word. Due its efficiency, it is currently at the basis of many compressors, like bzip [121], and different notions of BWT have been given in time [40, 92, 51].

Based on the properties of the Burrows-Wheeler transform, Ferragina and Manzini will later introduce the FM-index [43], a full text index at the base of widely used bioinformatics tools [79, 82]. Although we can store and query the BWT in compressed space [88], the number of samples of the suffix array required by the index grows with the length of the uncompressed text. The renovated interest from the combinatorial community is indeed given by the recent introduction of the r-index by Gagie et al. [49], which permits optimal searching time of *one* occurrence of a

pattern in a space proportional to r. The efficiency of the r-index truly depends on the repetitiveness of the dataset, which is the case for practical applications in *pangenomics*. Due to its properties in compressed full-text indexing, the size r has started to be considered as a measure of repetitiveness of finite words [99]. The Burrows-Wheeler Transform has also found applications in Combinatorics on Words [94], and recent studies have expanded the knowledge on the measure r [93, 54, 20, 55, 56, 44].

In this chapter, we focus on the study of the measure r from different theoretical aspects, in order to evaluate the measure r.

First, in Section 3.1 we describe the notion of Burrows-Wheeler Transform, and the associated measure r.

In Section 3.2 we show what properties of words influence the number of BWT-runs. As it will be clear later, circular bispecial factors play a crucial role on the measure r, and they represent natural bounds, even if not always tight. This relationship with bispecial factors appeared in [46].

In Section 3.3 we present some novel results on the sensitivity of the measure r, that is the capacity of the structure of the BWT of being affected when a simple edit operation is applied. In particular, we will focus on the additive sensitivity, and show a lower bound for all edit operations. For the insertion, our bound is of a significantly greater order than the previous $\Omega(\log n)$ proposed by Akagi et al. [1]. The results from this section appeared in [55].

3.1 Burrows-Wheeler Transform, Measures r and $r_\$$

Given an ordered alphabet Σ, let $w \in \Sigma^n$ be a finite word. We denote by $\mathcal{M} = \{\{w_1, w_2, \ldots, w_n\}\}$ the ordered multiset of rotations of w, i.e., $w_i \leq w_{i+1}$ for all $i \in [1, n-1]$. This multiset can be seen as a matrix $n \times n$, where $\mathcal{M}[i][j] = w_i[j]$. Note that if w is primitive, $\mathcal{M}$ is a set. The *Burrows-Wheeler Transform*, in short BWT, is the concatenation of the last letters of $\mathcal{M}$ taken in order. In Figure 3.1, the BWT for the word *abracadabra*\$ can be read in the last column L in bold from top to bottom.

A powerful property of the Burrows-Wheeler Transform is that it is reversible. In fact, consider the last column L ($= \mathsf{bwt}(w)$). It is easy to check that ordering lexicographically its elements returns the first column F of the matrix $\mathcal{M}$. Burrows

F										L
$\downarrow$										$\downarrow$
a	a	b	r	a	c	a	d	a	b	**r**
a	b	r	a	a	b	r	a	c	a	**d**
a	b	r	a	c	a	d	a	b	r	**a**
a	c	a	d	a	b	r	a	a	b	**r**
a	d	a	b	r	a	a	b	r	a	**c**
b	r	a	a	b	r	a	c	a	d	**a**
b	r	a	c	a	d	a	b	r	a	**a**
c	a	d	a	b	r	a	a	b	r	**a**
d	a	b	r	a	a	b	r	a	c	**a**
r	a	a	b	r	a	c	a	d	a	**b**
r	a	c	a	d	a	b	r	a	a	**b**

FIGURE 3.1: Example of BWT for the word *abracadabra*.

and Wheeler proved that the i-th occurrence of a letter $a \in \Sigma$ in L corresponds to the i-th occurrence of a in F [21]. We denote by $\mathsf{LF} : [1, n] \mapsto [1, n]$ the *LF-mapping* that uniquely associates the letters in L with the corresponding in F, that is $\mathsf{LF}[i] = j$ if $\mathsf{SA}[i] = \mathsf{SA}[j] + 1 \mod n$. From the definition, one has that $\mathsf{bwt}(w)[i]$ is preceded in w by $\mathsf{bwt}(w)[\mathsf{LF}[i]]$, for all $i \in [1, n]$. Thus, starting from the occurrence of the last letter of w in $\mathsf{bwt}(w)$, we can reconstruct the whole word by applying the LF mapping exactly n times.

Note that the $\mathsf{bwt}(w)$ consists in a permutation of the word w. However, the BWT tends to have longer equal-letter runs then the original word, particularly in highly repetitive texts.

Let $w \in \Sigma^*$ be a finite word. We can then factorize $w = a_1^{p_1} a_2^{p_2} \cdots a_k^{p_k}$, for some $k > 0$ such that $a_i \in \Sigma$ and $p_i > 0$ for all $i \in [1, k]$, and $a_j \neq a_{j+1}$ for all $j \in [1, k-1]$. The *run-length encoding* rle of the word w is

$$\mathsf{rle}(w) = (a_1, p_1)(a_2, p_2) \cdots (a_k, p_k).$$

The measure r for the word is then the number of equal-letter runs of the BWT of w, that is $r(w) = |\mathsf{rle}(\mathsf{bwt}(w))|$. Moreover, for a given word w, we use the term *BWT-run* to refer to any equal-letter run from $\mathsf{rle}(\mathsf{bwt}(w))$.

Different variants of the BWT have been presented over the years [40, 50, 78], and its definition has been extended for collections of sequences too [92, 10, 25]. In the context of combinatorial pattern matching, for a word $w \in \Sigma^*$ it is a standard

procedure computing the BWT after appending to w a special letter $\$ \notin \Sigma$ such that $\$ < a$ for all $a \in \Sigma$. By doing this, the order of the rotations in $\mathcal{R}(w)$ is equivalent to the order of the suffixes of w. To distinguish the size of the run-length encoding for these cases, we denote by $r_\$(w) = r(w\$)$. Note that most of the literature on the size of the BWT in compressed data structures concerns the measure $r_\$$ (see [99] for a reference). In Figures 3.1 and 3.2 we can compare the matrices of the sorted rotations and the corresponding BWT for the words *abracadabra* and *abracadabra*\$. One can notice that $r(abracadabra) = 7 \neq r_\$(abracadabra) = 8$. However, this difference can grow logarithmically with the length n of the word [55].

F											L
↓											↓
\$	*a*	*b*	*r*	*a*	*c*	*a*	*d*	*a*	*b*	*r*	**a**
a	\$	*a*	*b*	*r*	*a*	*c*	*a*	*d*	*a*	*b*	**r**
a	*b*	*r*	*a*	\$	*a*	*b*	*r*	*a*	*c*	*a*	**d**
a	*b*	*r*	*a*	*c*	*a*	*d*	*a*	*b*	*r*	*a*	**\$**
a	*c*	*a*	*d*	*a*	*b*	*r*	*a*	\$	*a*	*b*	**r**
a	*d*	*a*	*b*	*r*	*a*	\$	*a*	*b*	*r*	*a*	**c**
b	*r*	*a*	\$	*a*	*b*	*r*	*a*	*c*	*a*	*d*	**a**
b	*r*	*a*	*c*	*a*	*d*	*a*	*b*	*r*	*a*	\$	**a**
c	*a*	*d*	*a*	*b*	*r*	*a*	\$	*a*	*b*	*r*	**a**
d	*a*	*b*	*r*	*a*	\$	*a*	*b*	*r*	*a*	*c*	**a**
r	*a*	\$	*a*	*b*	*r*	*a*	*c*	*a*	*d*	*a*	**b**
r	*a*	*c*	*a*	*d*	*a*	*b*	*r*	*a*	\$	*a*	**b**

FIGURE 3.2: Example of BWT for the word *abracadabra\$*.

In the field of combinatorics on words, the measure r has been used to obtain a novel characterization of powers of standard Sturmian word [94]. We summarize this characterization along with the others from Lemmas 4 and 8 in the next theorem.

Theorem 14. *Let w be a word such that $\mathsf{alph}(w) = \{\mathtt{a}, \mathtt{b}\}$. Then the following are equivalent:*

1. *w is circularly balanced;*
2. *$w \in \mathcal{R}(s^\ell)$, for some standard Sturmian word s and for some $\ell > 0$;*
3. *$w = (\mu(a))^\ell$ for a Sturmian morphism μ and for some $\ell > 0$;*
4. *$r(w) = 2$.*

3.2 Factors of a Word and BWT-runs

In this section, we show some results that relate the measures r and $r_\$$ of a word w with the number of the factors of the word itself. Moreover, we prove that some combinatorial properties of the factors of a word affect the structure of its Burrows-Wheeler Transform and the measure r. Recall that for a given word w, the measure $\delta(w)$ returns, among all lengths $1 \leq k \leq |w|$, the maximum ratio between the number of distinct k-length factors of w over the length k. The following relationship between δ and the measure $r_\$$ has been proved by [67].

Theorem 15 ([67], Theorem 3.7)**.** *Every word w of length n satisfies*

$$r_\$(w) = \mathcal{O}(\delta(w) \log \delta(w) \cdot \max\{1, \log \frac{n}{\delta(w) \log \delta(w)}\}).$$

Even though this refers to the measure $r_\$$, the following lemma shows how to extend this relationship to the measure r.

Lemma 16. *Let $w \in \Sigma^*$ be a Lyndon word. Then, for all $u, v \in \Sigma^*$ such that $w = uv$, $r(vu) + 1 \leq r_\$(w) \leq r(vu) + 2$.*

Proof. Let $\mathcal{M}$ and $\mathcal{M}_\$$ be the matrix of the sorted rotations of w and $w\$$. The first rotation in $\mathcal{M}_\$$ is the one starting with the \$, that is preceded by the last letter of w. Since w is Lyndon, and therefore it is the first rotation in $\mathcal{M}$, it follows that $\mathsf{bwt}(w)[1] = \mathsf{bwt}(w\$)[1]$. Let $\mathsf{rot}(v)$ and $\mathsf{rot}^\$(v)$ be the sorted rotations with prefix v respectively from $\mathcal{M}$ and $\mathcal{M}_\$$, for any v such that $w = uv$. From the hypothesis on the primitivity of w, there can not be two equal rotations in any $\mathsf{rot}(v)$. Since every prefix u of any Lyndon word is the smallest $|u|$-length circular factor that occur in w, vu is the first rotation in lexicographical order in $\mathsf{rot}(v)$. Analogously, $v\$u$ is the first rotation in lexicographical order in $\mathsf{rot}^\$(v)$, while the other stays in the same relative order as $\mathsf{rot}(v)$. In correspondence of such rotations, both $\mathsf{bwt}(w\$)$ and $\mathsf{bwt}(w)$ are the same, with the solely exception when $v = w$ since $\mathsf{rot}(w) = \{w\}$ and $\mathsf{rot}^\$(w) = \{w\$\}$. It follows that $\mathsf{bwt}(w\$) = \mathsf{bwt}(w)[1]\$\mathsf{bwt}(w)[2, n]$. If $\mathsf{bwt}(w)[1] = \mathsf{bwt}(w)[2]$, then the \$ split the first run in $\mathsf{bwt}(w)$ with a consequent increase of two BWT-runs, otherwise it is only one more for the run of the \$. □

In other words, the measure r of a word is of the same order as the measure $r_\$$ of its Lyndon rotation. One may observe that in order to extend Theorem 15 to measure r, the measure δ should refer to Lyndon words. However, the following lemma shows that the difference between the δ-measures of two words in the same conjugacy class is less than 1.

Lemma 17. *Let $u, v \in \Sigma^*$ be two finite words. Then, $|\delta(uv) - \delta(vu)| < 1$.*

Proof. Clearly, one has that $c_{uv}(k) = c_{vu}(k)$ for all $k \in [1, n]$. Moreover, for every word $w \in \Sigma^n$, it holds that $\mathcal{C}_w(k) = F_w(k) \cup \{w[n-i+1, n] \cdot w[1, k-i] \mid i \in [1, k-1]\}$. Then, for all $k \in [1, n]$, it follows that $p_w(k) \leq c_w(k) \leq p_w(k) + k - 1$, and by reformulating the relation we obtain $c_w(k) - k + 1 \leq p_w(k) \leq c_w(k)$. If we divide all members by k, we can see that $\frac{p_w(k)}{k} \in [\frac{c_w(k)}{k} - 1 + \frac{1}{k}, \frac{c_w(k)}{k}]$. Such a range has size $1 - \frac{1}{k} < 1$ for all $k \in [1, n]$, and it is the same for all the rotations of w.

We now prove our thesis by contradiction. Let us assume that there exist two words $u, v \in \Sigma^*$ such that $\delta(uv) - \delta(vu) > 1$, and let $k^* \in [1, n]$ be such that $\delta(uv) = \frac{p_{uv}(k^*)}{k^*}$. But then, since $\frac{p_{uv}(k^*)}{k^*} - \frac{p_{vu}(k^*)}{k^*} < 1$ for all u, v, we would have $\delta(uv) = \frac{p_{uv}(k^*)}{k^*} > \frac{p_{vu}(k^*)}{k^*} > \delta(vu)$, contradiction. □

Thus, we can write the following corollary.

Corollary 18. *Every word w of length n satisfies*

$$r(w) = \mathcal{O}(\delta(w) \log \delta(w) \cdot \max\{1, \log \frac{n}{\delta(w) \log \delta(w)}\}).$$

Proof. Let $w = vu$ such that uv is Lyndon. By Lemmas 16 and 17, we have that

$$\begin{aligned} r(w) = \Theta(r_\$(uv)) &= \mathcal{O}(\delta(uv) \log \delta(uv) \cdot \max\{1, \log \frac{n}{\delta(uv) \log \delta(uv)}\}) \\ &= \mathcal{O}(\delta(w) \log \delta(w) \cdot \max\{1, \log \frac{n}{\delta(w) \log \delta(w)}\}). \end{aligned}$$

□

Such a result remarks how the number of distinct factors affects the maximum number of BWT-runs, regardless of the order of the alphabet. Nonetheless, not all factors have the same impact on r.

We now show how the circular bispecial factors can be used to derive both a lower and an upper bound on the number of BWT-runs.

Lemma 19. *Let $w \in \Sigma^*$ be a finite word. Then,*

$$\sum_{u \in cWBS(w)} \min\{e_l(u), e_r(u)\} + 1 \leq r(w) \leq \sum_{u \in cBS(w)} e_r(u) + 1.$$

Proof. Let $y = \mathsf{bwt}(w)$, and let $S = \{p_1 < p_2 < \ldots < p_{r-1}\}$ be the set of positions in y of the last letter of the first $r-1$ BWT-runs. This means that the equal-letter runs of y are $y[1, p_1]$, $y[p_1+1, p_2]$, ..., $y[p_{r-2}+1, p_{r-1}]$, and $y[p_{r-1}+1, n]$. It follows that $y[p_j] \neq y[p_j+1]$ for every $1 \leq j \leq r-1$. Let moreover w_j be the jth conjugate of w in lexicographic order. We can then define the set $W_S = \{w_{p_1}, w_{p_2}, \ldots, w_{p_{r-1}}\}$, that is the multiset of rotations of w corresponding to the positions in S. We observe that every $u_i = lcp(w_{p_i}, w_{p_i+1})$ is a bispecial circular factor, since $w_{p_i} = u_i a v' a'$ and $w_{p_i+1} = u_i b v'' b'$, for some $a, b, a', b' \in \Sigma$ such that $a < b$ and $a' \neq b'$, and some $v', v'' \in \Sigma^+$ that end with distinct letters.

Let us prove the first implication. Let $e(u) = \max\{e_l(u), e_r(u)\}$. By definition, for any $u \in cWBS(w)$, there are exactly $e(u)+1$ distinct circular factors of w of the type aua', for some $a, a' \in \Sigma$. Let $S_u = \{q_1 < q_2 < \ldots < q_{e_r(u)}\}$ be the set of positions in the BWT such that $w_{q_j} = uav_1$ and $w_{q_j+1} = ubv_2$, for every $j \in [1..e_r(u)]$, some $a, b \in \Sigma$ such that $a < b$ and some $v_1, v_2 \in \Sigma^*$. If $e_l(u) \leq e_r(u)$, then $e_l(ua) = 0$ (i.e. there exists only one circular left extension of ua) for all $a \in \Sigma$. It follows that there are at least $e_l(u)$ rotations such that $w_{q_i} = uav_1$ and $w_{q_i+1} = ubv_2$ end with distinct letters, i.e. there are at least $e_l(u)$ changes of letters in y that uniquely correspond to the weak bispecial circular factor u. On the other hand, if $e_l(u) > e_r(u)$, then $e_r(au) = 0$ (i.e. there exists only one circular right extension of au) for all $a \in \Sigma$. It follows that if there exist two consecutive cyclic rotations w_q and w_{q+1} such that $w_q = uav_1'$ and $w_{q+1} = ubv_2'$, then w_q and w_{q+1} end with different letters, i.e. there are at least $e_r(u)$ changes of letters in y that correspond to u.

We prove the second direction of the implication by contradiction. Let us suppose that there exist $p_{i_1} < p_{i_2} < \ldots < p_{i_{e_r(u_i)+1}} \in S$ such that $lcp(w_{p_{i_j}}, w_{p_{i_j}+1}) = u_i$ for each $j \in [1, e_r(u)+1]$. By the pigeonhole principle, in W_S there are at least two distinct rotations $w_{p_{i_j}} = u_i a v$ and $w_{p_{i_{j'}}} = u_i a v'$, for some $j < j'$, some $a \in \Sigma$, and

some $v, v' \in \Sigma^*$. However, one can see that $u_i a = lcp(w_{p_{i_j}}, w_{p_{i_{j'}}})$. By definition, $w_{p_j+1} = u_i b v''$, for some $b \in \Sigma$ such that $b > a$, and some $v'' \in \Sigma^*$. Thus, we obtain that $w_{p_j} < w_{p_{j'}} < w_{p_j+1}$, but this is impossible because w_{p_j} and w_{p_j+1} are consecutive, contradiction. □

The same upper bound has been defined on the measure $r_\$$ by [8]. However, as it will be shown at the end of the next section, the bounds on the measure r can not be directly derived from it.

3.3 Effects of Edit Operations on r

In this section, we focus on the effects that an edit operation may have on the measure r of a word. We start right away by observing how the BWT behaves when some combinatorial operation is considered.

A very well known property of the BWT is that, for every word $w \in \Sigma^*$ and every integer $n > 0$, it holds that $r(w) = r(w^n)$ [94]. Further, the structure of the BWT can be used to characterize powers of words.

Theorem 20 ([94])**.** *Let $u \in \Sigma^m$ be a word such that $\mathsf{bwt}(u) = a_1 a_2 \cdots a_m$, where $a_i \in \Sigma$ for all $i \in [1, m]$. Then, for a word w it holds that $\mathsf{bwt}(w) = a_1^p a_2^p \cdots a_m^p$ for some $p \geq 1$ if and only if $w = u^p$.*

This property clearly shows the efficacy of the BWT on periodic words. On the other hand, the BWT of a word does not have such a predictable behaviour when the concatenation is involved. In fact, the measure r is not monotone, that is there exist $w \in \Sigma^*$ and $a \in \Sigma$ such that $r(w) > r(wa)$.

Theorem 21. *The measure r is not monotone.*

In fact, characteristic Sturmian words have infinite prefixes that are power of standard Sturmian words, but not all prefixes are standard Sturmian words, and from the characterization of Theorem 4 the thesis follows. In Figure 3.3 is shown the BWT for the prefixes of the Fibonacci word f of length 12 and 13 respectively. One can see that $r(f[1, 12]) = 6$, while $r(f[1, 13]) = r(f[1, 12] \cdot \mathtt{b}) = 2$. More recently, we have formalized this property in [55].

	BWT
aaabaababaa	b
aabaaabaaba	b
aabaababaab	a
aababaabaaa	b
abaaabaabab	a
abaabaaabaa	b
abaababaaba	a
ababaabaaab	a
baaabaababa	a
baabaaabaab	a
baababaabaa	a
babaabaaaba	a

	BWT
aabaababaaba	b
aababaabaaba	b
aababaababaa	b
aababaabaaba	b
abaababaabaa	b
abaababaabab	a
ababaabaabab	a
aabaababaaba	a
babaababaaba	a
baabaababaab	a
baababaabaab	a
babaabaababa	a
babaababaaba	a

FIGURE 3.3: Matrix of sorted rotations and corresponding Burrows-Wheeler Transforms of the words abaababaabaa and abaababaabaab.

Akagi et al. [1], studied the *sensitivity* of different repetitiveness measures, that is the maximal increase that a measure can have on a word after a simple edit operation. More in detail, given a repetitive measure λ, they have defined the *worst-case multiplicative sensitivity* for insertion, deletion, and substitution as follows:

$$\mathsf{MS}_{ins}(\lambda, n) = \max_{w \in \Sigma^n}\{\frac{\lambda(w')}{\lambda(w)} \mid w' \in \Sigma^{n+1}, \mathsf{ed}(w, w') = 1\},$$

$$\mathsf{MS}_{sub}(\lambda, n) = \max_{w \in \Sigma^n}\{\frac{\lambda(w')}{\lambda(w)} \mid w' \in \Sigma^{n}, \mathsf{ed}(w, w') = 1\},$$

$$\mathsf{MS}_{del}(\lambda, n) = \max_{w \in \Sigma^n}\{\frac{\lambda(w')}{\lambda(w)} \mid w' \in \Sigma^{n-1}, \mathsf{ed}(w, w') = 1\}.$$

Analogously, they have defined the *worst-case additive sensitivity*:

$$\mathsf{AS}_{ins}(\lambda, n) = \max_{w \in \Sigma^n}\{\lambda(w') - \lambda(w) \mid w' \in \Sigma^{n+1}, \mathsf{ed}(w, w') = 1\},$$

$$\mathsf{AS}_{sub}(\lambda, n) = \max_{w \in \Sigma^n}\{\lambda(w') - \lambda(w) \mid w' \in \Sigma^{n}, \mathsf{ed}(w, w') = 1\},$$

$$\mathsf{AS}_{del}(\lambda, n) = \max_{w \in \Sigma^n}\{\lambda(w') - \lambda(w) \mid w' \in \Sigma^{n-1}, \mathsf{ed}(w, w') = 1\}.$$

In this section, we show novel results on the worst-case additive and multiplicative sensitivity of the measure r. In [1], the authors proved a tight upper bound $\mathcal{O}(\log n \log r)$ on all three multiplicative sensitivities, by using the relationship showed

in the previous section in Theorem 15 [67]. On the other hand, they showed a logarithmic lower bound only for the sensitivities of the insertion. These bounds were obtained from the results by Giuliani et al. [54] from the BWT of the *Fibonacci-plus words*, defined as follows:

Definition 22. *For all $k \geq 4$, let f_k be the kth Fibonacci word. Then, the kth order Fibonacci-plus word f_k^+ is either of the form $f_k\mathtt{b}$, if k is even, or $f_k\mathtt{a}$, if k is odd.*

They showed that the measure r of the reverse of these words grows as the order of the Fibonacci-plus word considered.

Proposition 23 ([54], Proposition 3)**.** *Let $(f_k^+)^R$ be the Fibonacci-plus word of kth order, for some $k \geq 4$. If k is even, then $r((f_k^+)^R) = k$. If k is odd, then $r((f_k^+)^R) = k - 1$.*

In particular, note that $r((f_k^+)^R) = r(c(f_k)^R) = \Theta(k) = \Theta(\log n)$, where c is either $\mathtt{a}$ or $\mathtt{b}$ if k is odd or even respectively. On the other hand, since $f_k^R \in \mathcal{R}(f_k)$ for every k, by Theorem 4 it follows that $r(f_k^R) = r(f_k) = 2$, the following lower-bounds can be deduced.

Corollary 24 ([1], Corollary 2)**.** *The following lower bounds on the sensitivity of the measure r over binary words $w \in \{\mathtt{a}, \mathtt{b}\}^*$ hold:*

$MS_{ins}(r, n) = \Theta(\log n)$, $AS_{ins}(r, n) = \Theta(\log n)$.

More recently, we have proved that the same logarithmic bound on the multiplicative sensitivity can be reached to the remaining edit operations, namely deletion and substitution, when these are applied rotations of Fibonacci words.

Proposition 25. *Let f_{2k} be the Fibonacci word of length n and order $2k > 4$, and $\widehat{f_{2k}} = f_{2k}[1, n-1]$, and $\widehat{f_{2k}^R} = (f_{2k}[2, n])^R$. Then, the following structures for the BWT hold:*

1. $bwt(\widehat{f_{2k}}) = \mathtt{b}^{k-1}\mathtt{ab}^{F_{2k-3}-k+1}\mathtt{ab}^{F_{2k-5}} \cdots \mathtt{b}^{F_5}\mathtt{ab}^{F_3}\mathtt{aba}^{F_{2k-1}-k+1}$;
2. $bwt(\widehat{f_{2k}^R}\mathtt{b}) = \mathtt{b}^{F_{2k-2}-k+1}\mathtt{a}^{F_0}\mathtt{ba}^{F_2}\mathtt{b} \cdots \mathtt{a}^{F_{2k-4}}\mathtt{ba}^{F_{2k-2}-2}\mathtt{ba}$.

In particular, both have $2k$ runs.

In this section, we will focus on the worst-case additive sensitivity of the measure r. In fact, for insertion, deletion, and substitution, we can obtain a larger lower bound than the $\Omega(\log n)$ by Akagi et al. [1].

Theorem 26. *The following lower bounds on the sensitivity of the measure r over binary words $w \in \{\mathtt{a}, \mathtt{b}\}$ hold:*

1. $AS_{ins}(r, n) = \Omega(\sqrt{n})$;
2. $AS_{del}(r, n) = \Omega(\sqrt{n})$;
3. $AS_{sub}(r, n) = \Omega(\sqrt{n})$.

To prove them, we have considered the following family of words: given an integer k, let $s_i = \mathtt{ab}^i\mathtt{aa}$, $e_i = \mathtt{ab}^i\mathtt{aba}^{i-2}$ for all $2 \leq i \leq k-1$, and $q_k = \mathtt{ab}^k\mathtt{a}$. The words for which the gap announced holds is for $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$, for all $k > 5$. In the next lemma, we explicit the length n of w_k in terms of the integer k.

Lemma 27. *Let $n = |w_k|$ for some $k > 2$. It holds that $n = (3k^2 + 7k - 18)/2$. Moreover, it holds that $k = \Theta(\sqrt{n})$.*

Proof. From their definitions, we have that, for all $i \in [2, k-1]$, $|s_i| = |\mathtt{ab}^i\mathtt{aa}| = i + 3$, and $|e_i| = |\mathtt{ab}^i\mathtt{aba}^{i-2}| = i + (i-2) + 3 = 2i + 1$, while $|q_k| = |\mathtt{ab}^k\mathtt{a}| = k + 2$. Hence,

$$\begin{aligned}
|w_k| &= \sum_{i=2}^{k-1}(|s_i|) + \sum_{i=2}^{k-1}(|e_i|) + |q_k| \\
&= \sum_{i=2}^{k-1}(i+3) + \sum_{i=2}^{k-1}(2i+1) + (k+2) \\
&= \sum_{i=2}^{k-1}(3i+4) + k + 2 \\
&= 3\sum_{i=2}^{k-1}(i) + 4(k-2) + k + 2 \\
&= 3\frac{k^2-k}{2} - 3 + 5k - 6 \\
&= \frac{3k^2 + 7k - 18}{2}.
\end{aligned}$$

One can observe that $n = \Theta(k^2)$, and symmetrically $k = \Theta(\sqrt{n})$. □

We say that a set S of words is *prefix-free* if for all pairs $u, v \in S$, u is not prefix of v (and vice versa). Given the regular construction of w_k, we can use its factors to deduce the order of its rotations.

Lemma 28. *Let $k > 2$ be an integer. Then, $s_2 < e_2 < s_3 < e_3 < \ldots < s_{k-1} < e_{k-1} < q_k$. Moreover the set $\bigcup_{i=2}^{k-1}\{s_i, e_i\} \cup \{q_k\}$ is prefix-free.*

Proof. The proof easily follows from the longest run of b's after the first a. In fact, for every pair of indices i, j such that $2 \leq i < j \leq k-1$, we have that $s_i = \mathtt{ab}^i\mathtt{aa} < \mathtt{ab}^i\mathtt{aba}^{i-2} = e_i < \mathtt{ab}^i\mathtt{bb}^{j-i-1}\mathtt{aa} = s_j < \mathtt{ab}^i\mathtt{bb}^{j-i-1}\mathtt{aba}^{j-2} = e_j < \mathtt{ab}^i\mathtt{bb}^{j-i-1}\mathtt{bb}^{k-j-1}\mathtt{a} = q_k$. From the comparison above, one can see that for every $i \in [2, k-1]$, s_i differs from e_i at position $i+3 = |s_i|$, while for every $j > i$, s_i and e_i both differ from s_j, e_j and q_k at position $i+2 < |s_i| \leq |e_i|$. Thus, since there is no word in $\bigcup_{i=2}^{k-1}\{s_i, e_i\} \cup \{q_k\}$ that is prefix of another within the same set, such a set is prefix-free. □

The BWT of the word w_k is described in the following lemmas. Given the multitude of cases to analyse, we consider for each lemma only a range of consecutive rotations that share a common prefix. To do so, given a word w and a factor u, we use the notation $\mathsf{rot}^w(u)$ to denote the factor in $\mathsf{bwt}(w)$ in correspondence of the rotation starting with u. When w is clear from the context, we simply write $\mathsf{rot}(u)$.

Lemma 29 ($\mathsf{rot}(\mathtt{a}^{k-2}\mathtt{b})$)**.** *Given the word $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ for some $k > 5$, the first rotation in the BWT matrix is $\mathtt{a}^{k-3} q_k \cdots \mathtt{b}$.*

Proof. The first rotation in lexicographic order must start with the longest run of a's. By definition of w_k, the longest run of a's has length $k-2$, and it is obtained by concatenating the suffix $\mathtt{a}^{k-3}$ of e_{k-1} with q_k, which is preceded by a b (otherwise we could extend the run of a's). □

Lemma 30 ($\mathsf{rot}(\mathtt{a}^i\mathtt{b})$ for $4 \leq i \leq k-3$)**.** *Given the word $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ for some $k > 5$, and an integer $4 \leq i \leq k-3$, the rotations in the BWT matrix starting with $\mathtt{a}^i\mathtt{b}$ are $\mathtt{a}^{i-1} s_{i+2} \cdots \mathtt{b} < \mathtt{a}^{i-1} s_{i+3} \cdots \mathtt{a} < \ldots < \mathtt{a}^{i-1} s_{k-1} \cdots \mathtt{a} < \mathtt{a}^{i-1} q_k \cdots \mathtt{a}$.*

Proof. One can notice that, for all $4 \leq i \leq k-3$, the (circular) factor $\mathtt{a}^i\mathtt{b}$ can only be obtained, for all $i+2 \leq j \leq k$, from the concatenation of the suffix $\mathtt{a}^{i-1}$ of e_{j-1}, with either the prefix ab of s_j, if $i+2 \leq j \leq k-1$, or the prefix ab of q_k, if $j = k$. By Lemma 28, we can sort these rotations according to the lexicographic order of $\bigcup_{j=i}^{k-1}\{s_j\} \cup \{q_k\}$. Note that all these rotations end with an a, with the exception of the rotation starting with $\mathtt{a}^{i-1} s_{i+2}$, since it is where the only occurrence of $\mathtt{ba}^i\mathtt{b}$ can be found. □

Lemma 31 ($\mathsf{rot}(\mathtt{aaab})$)**.** *Given the word $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ for some $k > 5$, the first five rotations in the BWT matrix starting with* aaab *are $\mathtt{aa} e_2 \cdots \mathtt{b} < \mathtt{aa} e_3 \cdots \mathtt{b} < \mathtt{aa} e_4 \cdots \mathtt{b} <$*

$\mathtt{aa}s_5 \cdots \mathtt{b} < \mathtt{aa}e_5 \cdots \mathtt{b}$, *while the remaining are* $\mathtt{aa}s_6 \cdots \mathtt{a} < \mathtt{aa}e_6 \cdots \mathtt{b} < \ldots < \mathtt{aa}s_{k-1} \cdots \mathtt{a} < \mathtt{aa}e_{k-1} \cdots \mathtt{b} < \mathtt{aa}q_k \cdots \mathtt{a}$.

Proof. Analogously to the proof of Lemma 30, some of the rotations starting with aaab can be obtained, for all $5 \leq j \leq k$, from the concatenation of the suffix aa of e_{j-1}, with either the prefix ab of s_j, if $5 \leq j \leq k-1$, or the prefix ab of q_k, if $j = k$. However, in this case we have more rotations starting with aaab, that are those rotations starting with the suffix aa of $s_{j'}$ concatenated with the prefix ab of $e_{j'}$, for all $2 \leq j' \leq k-1$. Thus, all the rotations starting with aaab are sorted according to the lexicographic order of the words in $\bigcup_{j=5}^{k-1}\{s_j\} \cup \bigcup_{j'=2}^{k-1}\{e_{j'}\} \cup \{q_k\}$. Note that all the rotations starting either with $\mathtt{aa}s_j$, for all $6 \leq j \leq k-1$, or with $\mathtt{aa}q_k$, end with a. On the other hand, the rotations starting either with $\mathtt{aa}s_5$ or with $\mathtt{aa}e_j$, for all $2 \leq j \leq k-1$, end with a b. □

Lemma 32 (rot(aab)). *Given the word* $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ *for some* $k > 5$, *the first five rotations in the BWT matrix starting with* aab *are* $\mathtt{a}s_2 \cdots \mathtt{b} < \mathtt{a}e_2 \cdots \mathtt{a} < \mathtt{a}e_3 \cdots \mathtt{a} < \mathtt{a}s_4 \cdots \mathtt{b} < \mathtt{a}e_4 \cdots \mathtt{a}$, *while the remaining are* $\mathtt{a}s_5 \cdots \mathtt{a} < \mathtt{a}e_5 \cdots \mathtt{a} < \ldots < \mathtt{a}s_{k-1} \cdots \mathtt{a} < \mathtt{a}e_{k-1} \cdots \mathtt{a} < \mathtt{a}q_k \cdots \mathtt{a}$.

Proof. Each of the rotations starting with aaab from Lemma 31 induces a rotation starting with aab, obtained by shifting on the left one letter a. It follows that all of these rotations end with an a. The other rotations starting with aab are the one obtained by concatenating the suffix a of e_3 and the prefix ab of s_4, and the one obtained by concatenating the suffix a of q_k and the prefix ab of s_2. Moreover, both the rotations end with a b. The thesis follows by sorting the rotations according to the lexicographic order of the words in $\{s_2\} \cup \bigcup_{j=4}^{k-1}\{s_j\} \cup \bigcup_{j'=2}^{k-1}\{e_{j'}\} \cup \{q_k\}$. □

Lemma 33 (rot(ab)). *Given the word* $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ *for some* $k > 5$, *the first* $k-2$ *rotations in the BWT matrix starting with* ab *are* $\mathtt{aba}^{k-3} q_k \cdots \mathtt{b} < \mathtt{aba}^{k-4} s_{k-1} \cdots \mathtt{b} < \ldots < \mathtt{ab}s_3 \cdots \mathtt{b}$, *the following four rotations are* $s_2 \cdots \mathtt{a} < e_2 \cdots \mathtt{a} < s_3 \cdots \mathtt{b} < e_3 \cdots \mathtt{a}$, *and the remaining are* $s_4 \cdots \mathtt{a} < e_4 \cdots \mathtt{a} < \ldots < s_{k-1} \cdots \mathtt{a} < e_{k-1} \cdots \mathtt{a} < q_k \cdots \mathtt{a}$.

Proof. For any two distinct integers $i, i' \geq 0$, we have that $\mathtt{aba}^i\mathtt{b} < \mathtt{aba}^{i'}\mathtt{b}$ if and only if $i > i'$. Thus, the first rotation in lexicographic order starting with ab is the one which is followed by the longest run of a's. The smallest of these rotations can be

found by concatenating the suffix $\mathtt{aba}^{k-3}$ of e_{k-1} with the prefix ab of q_k, followed by the suffix $\mathtt{aba}^{i-2}$ of e_{i-1} concatenated with the prefix ab of s_i, for all $3 \leq i \leq k-1$ taken in decreasing order. By construction of e_i, for all $3 \leq i \leq k-1$, these rotations must end with a b.

The remaining rotations starting with ab are exactly those rotations having as prefix either s_i or e_i, for all $2 \leq i \leq k-1$, or q_k. Note that all of these rotations are obtained by shifting on the left one letter a from the rotations starting with aab from Lemma 32, with the exception of the one starting with s_3. It follows that the latter ends with a b, while all the other rotations with an a. □

Lemma 34 (rot(ba)). *Given the word* $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$ *for some* $k > 5$, *the first* $k-5$ *rotations in the BWT matrix starting with* ba *are* $\mathtt{ba}^{k-3} q_k \cdots \mathtt{a} < \mathtt{ba}^{k-4} s_{k-1} \cdots \mathtt{a} < \ldots < \mathtt{ba}^3 s_6 \cdots \mathtt{a}$, *followed by* $\mathtt{baa} e_2 \cdots \mathtt{b} < \mathtt{baa} e_3 \cdots \mathtt{b} < \mathtt{baa} e_4 \cdots \mathtt{b} < \mathtt{baa} s_5 \cdots \mathtt{a} < \mathtt{baa} e_5 \cdots \mathtt{b}$, *then by* $\mathtt{baa} e_6 \cdots \mathtt{b} < \mathtt{baa} e_7 \cdots \mathtt{b} < \ldots < \mathtt{baa} e_{k-1} \cdots \mathtt{b} < \mathtt{ba} s_2 \cdots \mathtt{b} < \mathtt{ba} s_4 \cdots \mathtt{a}$, *and finally by* $\mathtt{baba}^{k-3} q_k \cdots \mathtt{b} < \mathtt{baba}^{k-4} s_{k-1} \cdots \mathtt{b} < \ldots < \mathtt{bab} s_3 \cdots \mathtt{b} < \mathtt{b} s_3 \cdots \mathtt{a}$.

Proof. One can notice that we have as many circular occurrences of ba as the number of maximal (circular) runs of b's in w_k. Then, for all $2 \leq i \leq k-1$, we have (i) one run of b's in s_i, and (ii) two runs in e_i, and (iii) one run in q_k.

For the case (i), we have one rotation starting with $\mathtt{baa} e_i$, for each $2 \leq i \leq k-1$. Since each run of b's within each word from $\bigcup_{i=2}^{k-1} \{s_i\}$ is of length at least 2, all rotations in (i) end with a b.

For the case (ii), for all $2 \leq i \leq k-1$, we can distinguish between two subcases, based on where ba starts: if either (ii.a) from the first run of b's in e_i, or (ii.b) from the second one. For the case (ii.a), we can see that these rotations are of the type $\mathtt{baba}^{i-2} s_{i+1}$, if $2 \leq i < k-2$, and $\mathtt{baba}^{k-3} q_k$. Analogously to the case (i), each rotations for case (ii.a) end with a b. Each rotation in (ii.b) is obtained by shifting two letters on the right each rotation in (ii.a). Therefore, all of these rotations end with an a and have prefixes of the type $\mathtt{ba}^{i-2} s_{i+1}$, if $2 \leq i < k-2$, or $\mathtt{ba}^{k-3} q_k$.

For the case (iii), the rotation starting with ba in q_k has $\mathtt{ba} s_2$ as prefix, and it is preceded by a b.

Observe that only for (ii.b) we have rotations starting with baaaa. Hence, the first rotation in lexicographic order is the one starting with $\mathtt{ba}^{k-3}q_k$, followed by those starting with $\mathtt{ba}^{k-4}s_{k-1} < \mathtt{ba}^{k-5}s_{k-2} < \ldots < \mathtt{baaa}s_6$.

Among the remaining rotations, those having prefix baaa either start with $\mathtt{baa}s_5$ from (ii.b), or $\mathtt{baa}e_i$ from (i), for all $2 \leq i \leq k-1$. Thus, by Lemma 28, we can sort them according to the order of the words in $\{s_5\} \cup \bigcup_{i=2}^{k-1}\{e_i\}$. Then, the remaining rotations with prefix baa are those starting with $\mathtt{ba}s_2$ from (iii), and $\mathtt{ba}s_4$ from (ii.b). Finally, let us focus on the rotations from case (ii.a). These rotations are sorted according to the length of the run of a's following the common prefix bab, similarly to the sorting of the rotations from the case (ii.b). The last rotation left is the one starting with $\mathtt{b}s_3$ from case (ii.b). Since this rotation is greater than each word from case (ii.a), this is the greatest rotation of w_k starting with ba and the thesis follows. □

Lemma 35 (rot($\mathtt{b}^j\mathtt{a}$) for all $2 \leq j \leq k-1$)**.** *Given the word* $w_k = (\prod_{i=2}^{k-1} s_i e_i)q_k$ *for some* $k > 5$*, and an integer* $2 \leq i \leq k-2$*, the first* $k-i$ *rotations in the BWT matrix starting with* $\mathtt{b}^i\mathtt{a}$ *are* $\mathtt{b}^i\mathtt{aa}e_i \cdots \mathtt{a} < \mathtt{b}^i\mathtt{aa}e_{i+1} \cdots \mathtt{b} < \ldots < \mathtt{b}^i\mathtt{aa}e_{k-1} \cdots \mathtt{b} < \mathtt{b}^i\mathtt{a}s_2 \cdots \mathtt{b}$*, followed by* $\mathtt{b}^i\mathtt{aba}^{k-3}q_k \cdots \mathtt{b} < \mathtt{b}^i\mathtt{aba}^{k-4}s_{k-1} \cdots \mathtt{b} < \ldots < \mathtt{b}^i\mathtt{aba}^{i-1}s_{i+2} \cdots \mathtt{b} < \mathtt{b}^i\mathtt{aba}^{i-2}s_{i+1} \cdots \mathtt{a}$.

Proof. All runs of b's of length at least $2 \leq i \leq k-2$, either appear in (i) s_j or (ii) e_j, for all $i \leq j \leq k-1$, or in (iii) q_k. Let us consider the three cases separately. For all $i \leq j \leq k-1$, the rotation starting within s_j (i) has as prefix $\mathtt{b}^i\mathtt{aa}e_j$. For all $i \leq j \leq k-2$, the rotation starting within e_j (ii) has as prefix $\mathtt{b}^i\mathtt{aba}^{j-2}s_{j+1}$, and for $j = k-1$ we have the rotation with prefix $\mathtt{b}^i\mathtt{aba}^{k-3}q_k$. Finally, the rotation starting within q_k (iii) has as prefix $\mathtt{b}^i\mathtt{a}s_2$.

By construction, we can see that first we have all the rotations from case (i) sorted according to the lexicographic order of the words in $\bigcup_{j=i}^{k-1}\{e_i\}$ (Lemma 28), then we have the rotation from case (iii), and finally the rotation from case (ii), sorted according to the decreasing length of the run of a's following the common prefix $\mathtt{b}^i\mathtt{ab}$.

Moreover, note that only when the run of b's is of length exactly i the rotation end with an a. Thus, the only for the rotations ending with an a are those starting within s_i and e_i, i.e. those with prefix $\mathtt{b}^i\mathtt{a}e_i$ and $\mathtt{b}^i\mathtt{aba}^{i-2}s_{i+1}$. □

Block prefix	Ordering factor	BWT
$\mathtt{a}^{k-2}\mathtt{b}$	$\mathtt{b}^{k-1}\mathtt{a}$	b
$\mathtt{a}^{k-3}\mathtt{b}$	$\mathtt{b}^{k-2}\mathtt{aa}$	b
	$\mathtt{b}^{k-1}\mathtt{a}$	a
$\vdots$	$\vdots$	$\vdots$
$\mathtt{a}^4\mathtt{b}$	$\mathtt{b}^5\mathtt{aa}$	b
	$\mathtt{b}^6\mathtt{aa}$	a
	$\vdots$	$\vdots$
	$\mathtt{b}^{k-1}\mathtt{a}$	a
aaab	bab	b
	bbaba	b
	bbbabaa	b
	bbbbaa	b
	bbbbabaaa	b
	bbbbbaa	a
	bbbbbabaaaa	b
	$\vdots$	$\vdots$
	$\mathtt{b}^{k-2}\mathtt{aa}$	a
	$\mathtt{b}^{k-2}\mathtt{aba}^{k-3}$	b
	$\mathtt{b}^{k-1}\mathtt{a}$	a

Block prefix	Ordering factor	BWT
aab	baa	b
	bab	a
	bbaba	a
	bbbaa	b
	bbbabaa	b
	bbbbaa	a
	bbbbabaaa	a
	$\vdots$	$\vdots$
	$\mathtt{b}^{k-2}\mathtt{aa}$	a
	$\mathtt{b}^{k-2}\mathtt{aba}^{k-3}$	a
	$\mathtt{b}^{k-1}\mathtt{a}$	a
ab	$\mathtt{a}^{k-3}q_k$	b
	$\mathtt{a}^{k-4}s_{k-1}$	b
	$\vdots$	$\vdots$
	s_3	b
	baa	a
	bab	b
	bbaa	b
	bbaba	a
	bbbaa	a
	bbbabaa	a
	$\vdots$	$\vdots$
	$\mathtt{b}^{k-1}\mathtt{a}$	a

Block prefix	Ordering factor	BWT
ba	$\mathtt{a}^{k-4}q_k$	a
	$\mathtt{a}^{k-5}s_{k-1}$	a
	$\vdots$	$\vdots$
	$\mathtt{a}^2s_6$	a
	$\mathtt{a}e_2$	b
	$\mathtt{a}e_3$	b
	$\mathtt{a}e_4$	a
	$\mathtt{a}s_5$	a
	$\mathtt{a}e_5$	b
	$\mathtt{a}e_6$	b
	$\vdots$	$\vdots$
	$\mathtt{a}e_{k-1}$	b
	s_2	b
	s_4	a
	$\mathtt{ba}^{k-3}q_k$	b
	$\mathtt{ba}^{k-4}s_{k-1}$	b
	$\vdots$	$\vdots$
	$\mathtt{b}s_3$	b
	bbbaa	a

Block prefix	Ordering factor	BWT
bba	$\mathtt{a}e_2$	a
	$\mathtt{a}e_3$	b
	$\vdots$	$\vdots$
	$\mathtt{a}e_{k-1}$	b
	s_2	b
	$\mathtt{ba}^{k-3}q_k$	b
	$\mathtt{ba}^{k-4}s_{k-1}$	b
	$\vdots$	$\vdots$
	$\mathtt{ba}s_4$	b
	$\mathtt{ba}s_3$	a
$\vdots$	$\vdots$	$\vdots$
$\mathtt{b}^{k-1}\mathtt{a}$	$\mathtt{a}e_{k-1}$	a
	s_2	b
	$\mathtt{ba}^{k-3}q_k$	a
$\mathtt{b}^k\mathtt{a}$	s_2	a

TABLE 3.1: Scheme of the BWT-matrix of a word w_k with $k > 5$.

Lemma 36 ($\mathtt{rot}(\mathtt{b}^k\mathtt{a})$)**.** *Given the word* $w_k = (\prod_{i=2}^{k-1} s_i e_i)q_k$ *for some* $k > 5$*, the last four rotations of the BWT matrix are* $\mathtt{b}^{k-1}\mathtt{aa}e_{k-1} \cdots \mathtt{a} < \mathtt{b}^{k-1}\mathtt{a}s_2 \cdots \mathtt{b} < \mathtt{b}^{k-1}\mathtt{aba}^{k-3}q_k \cdots \mathtt{a} < \mathtt{b}^k\mathtt{a}s_2 \cdots \mathtt{a}$.

Proof. Observe that the only rotations with prefix $\mathtt{b}^{k-1}\mathtt{a}$ either start within s_{k-1}, or q_k, or e_{k-1}. These rotations have prefix respectively $\mathtt{b}^{k-1}\mathtt{aa}e_{k-1}$, $\mathtt{b}^{k-1}\mathtt{a}s_2$, and $\mathtt{b}^{k-1}\mathtt{aba}^{k-3}q_k$. One can see that these rotations taken in this order are already sorted, and only the rotation starting within q_k ends with a b, while the other two with an a. Finally, the only occurrence of $\mathtt{b}^k$ is within q_k. Hence, the last rotation in lexicographic order starts with $\mathtt{b}^k\mathtt{a}s_2$, and since the run of b's is maximal it ends with an a, and the thesis follows. □

The following proposition summarises the results obtained in the previous lemmas, and properly describe the size of the run-length encoding of $\mathtt{bwt}(w_k)$.

Proposition 37. *Given an integer $k > 5$, let $w_k = (\prod_{i=2}^{k-1} s_i e_i) q_k$. Then, it holds that*

$$\begin{aligned}
rot(\mathtt{a}^i\mathtt{b}) &= \mathtt{ba}^{k-i-2} \text{ for all } 4 \leq i \leq k-2,\\
rot(\mathtt{a}^3\mathtt{b}) &= \mathtt{b}^5(\mathtt{ab})^{k-6}\mathtt{a},\\
rot(\mathtt{a}^2\mathtt{b}) &= \mathtt{baaba}^{2k-8},\\
rot(\mathtt{ab}) &= \mathtt{b}^{k-2}\mathtt{aaba}^{2k-6},\\
rot(\mathtt{ba}) &= \mathtt{a}^{k-5}\mathtt{bbbab}^{k-4}\mathtt{ab}^{k-2}\mathtt{a},\\
rot(\mathtt{b}^j\mathtt{a}) &= \mathtt{ab}^{2k-2j-1}\mathtt{a} \text{ for all } 2 \leq j \leq k-1, \text{ and}\\
rot(\mathtt{b}^k\mathtt{a}) &= \mathtt{a}.
\end{aligned}$$

Hence, $bwt(w_k) = \prod_{i=2}^{k-1} rot(\mathtt{a}^{k-i}\mathtt{b}) \cdot \prod_{i=1}^{k} rot(\mathtt{b}^i\mathtt{a})$. Moreover, it holds that $r(w_k) = 6k - 12$.

Proof. The words $\mathtt{rot}(\mathtt{a}^{k-2}\mathtt{b})$, $\mathtt{rot}(\mathtt{a}^i\mathtt{b})$ for all $4 \leq i \leq k-2$, $\mathtt{rot}(\mathtt{a}^3\mathtt{b})$, $\mathtt{rot}(\mathtt{a}^2\mathtt{b})$, $\mathtt{rot}(\mathtt{ab})$, $\mathtt{rot}(\mathtt{ba})$, $\mathtt{rot}(\mathtt{b}^j\mathtt{a})$ for all $2 \leq j \leq k-1$, and $\mathtt{rot}(\mathtt{b}^k\mathtt{a})$, are the concatenations of the last letters of the rotations from Lemma 29, Lemma 30, Lemma 31, Lemma 32, Lemma 33, Lemma 34, Lemma 35, and Lemma 36 respectively. Moreover, every rotation used to build $\mathtt{rot}(\mathtt{a}^i\mathtt{b})$ is smaller than each rotation used to build $\mathtt{rot}(\mathtt{a}^{i'}\mathtt{b})$, for every $1 \leq i' < i \leq k-2$. Symmetrically, every rotation used to build $\mathtt{rot}(\mathtt{b}^j\mathtt{a})$ is greater than each rotation used to build $\mathtt{rot}(\mathtt{b}^{j'}\mathtt{a})$, for every $1 \leq j' < j \leq k$. Since we have considered all the disjoint ranges of rotations of w_k based on their common prefix, the word $\prod_{i=2}^{k-1} \mathtt{rot}(\mathtt{a}^{k-i}\mathtt{b}) \cdot \prod_{i=1}^{k} \mathtt{rot}(\mathtt{b}^i\mathtt{a})$ is the BWT of w_k.

With the structure of $\mathtt{bwt}(w_k)$, we can easily derive its number of runs. The word $\prod_{i=2}^{k-4}(\mathtt{rot}(\mathtt{a}^{k-i}\mathtt{b}))$ has exactly $2(k-6)$ runs: we start with 2 runs from $\mathtt{rot}(\mathtt{a}^{k-2}\mathtt{b})\mathtt{rot}(\mathtt{a}^{k-3}\mathtt{b}) = \mathtt{bba}$, and then, concatenating each other $\mathtt{rot}(\mathtt{a}^i\mathtt{b})$ up to $\mathtt{rot}(\mathtt{a}^4\mathtt{b})$ adds 2 new runs each.

It is easy to see that $\mathtt{rot}(\mathtt{aaab})$, $\mathtt{rot}(\mathtt{aab})$, and $\mathtt{rot}(\mathtt{ab})$, have $2(k-5)$, 4, and 4 runs, respectively. Moreover, the boundaries between these words do not merge, nor with the one in the previous paragraph in the case of $\mathtt{rot}(\mathtt{aaab})$.

The word $\mathtt{rot}(\mathtt{ba})$ has exactly 7 runs but it merges with $\mathtt{rot}(\mathtt{ab})$ and $\mathtt{rot}(\mathtt{bba})$, hence we only charge 5 runs to this word.

The remaining part of the BWT, i.e., $\prod_{i=2}^{k}(\mathsf{rot}(\mathtt{b}^i\mathtt{a}))$, has $2(k-2)+1$ runs: we start with 3 runs from $\mathsf{rot}(\mathtt{bba})$, and then, concatenating each other $\mathsf{rot}(\mathtt{b}^i\mathtt{a})$ up to $\mathsf{rot}(\mathtt{b}^{k-1}\mathtt{a})$ adds 2 new runs each. The word $\mathsf{rot}(\mathtt{b}^k\mathtt{a})$ does not add new runs, as it consists only of an $\mathtt{a}$ that merges with the previous one.

Overall, we have $2(k-6)+2(k-5)+4+4+5+2(k-2)+1 = 6k-12$, and the claim holds. □

A graphical interpretation is detailed in Table 3.1. The *block prefix* column shows the common prefix shared by all the rotations of the block. The *ordering factor* column shows the factor following the block prefix of a rotation, which affects the relative order of the rotations within the same block. The BWT column shows the last letter of each rotation. The dashed lines divide sub-ranges of rotations for which the BWT follows distinct patterns.

To prove Theorem 26, in the next lemmas we show the behaviour of BWT when an edit operation is applied on w_k. For a given word $w \neq \epsilon$, the words w^{ins}, w^{del}, and w^{sub} are obtained by applying on w an insertion, a deletion, and a substitution of a letter respectively. Since a single edit operation does not completely change the relative order of some rotations, instead of proving the whole structure from the beginning, we often compare how the edit operation changes either the relative order or the last letter of the rotations of w_k. In other words, for specific factors v of w_k, we compare $\mathsf{rot}(v, w_k)$ with $\mathsf{rot}(v, w'_k)$, where w'_k is obtained after applying to w_k an edit operation. For simplicity of exposition, in this part we use the notation $\mathsf{rot}(v)$ and $\mathsf{rot}^\star(v)$ to denote the BWT in correspondence of the rotations with prefix $v \in \Sigma^*$ of w_k and w'_k respectively.

For the insertion, we have considered the word w_k^{ins} obtained by appending the letter $\mathtt{a}$ to w_k.

Lemma 38 (BWT of $w_k\mathtt{a}$)**.** *Given an integer $k > 5$, for $w_k\mathtt{a}$ it holds that*

$$
\begin{aligned}
rot^\star(\mathtt{a}^i\mathtt{b}) &= \mathtt{ba}^{k-i-2} \textit{ for all } 4 \leq i \leq k-2,\\
rot^\star(\mathtt{a}^3\mathtt{b}) &= \mathtt{bb}^5(\mathtt{ab})^{k-6}\mathtt{a},\\
rot^\star(\mathtt{a}^2\mathtt{b}) &= \mathtt{aaaba}^{2k-8},\\
rot^\star(\mathtt{ab}) &= \mathtt{b}^{k-2}\mathtt{aaba}^{2k-6},\\
rot^\star(\mathtt{ba}) &= \mathtt{a}^{k-5}\mathtt{bbbbab}^{k-5}\mathtt{ab}^{k-2}\mathtt{a},\\
rot^\star(\mathtt{b}^j\mathtt{a}) &= \mathtt{bab}^{2k-2j-2}\mathtt{a} \textit{ for all } 2 \leq j \leq k-1 \textit{ and}\\
rot^\star(\mathtt{b}^k\mathtt{a}) &= \mathtt{a}.
\end{aligned}
$$

Hence, $bwt(w_k\mathtt{a}) = \prod_{i=2}^{k-1} rot^\star(\mathtt{a}^{k-i}\mathtt{b}) \cdot \prod_{i=1}^{k} rot^\star(\mathtt{b}^i\mathtt{a})$. Moreover, it holds that $r(w_k\mathtt{a}) = 8k - 20$.

Proof. By Lemmas 29 and 30, we can see that appending an a after q_k does not affect the BWT in the range of rotations having $\mathtt{a}^i\mathtt{b}$ as prefix, for all $4 \leq i \leq k-2$. Thus, $\mathrm{rot}^\star(\mathtt{a}^i\mathtt{b}) = \mathrm{rot}(\mathtt{a}^i\mathtt{b})$ for all $4 \leq i \leq k-2$.

The rotation starting with $\mathtt{aa}s_2$, which is not a circular factor of w_k, ends with a b. By Lemma 31, we can see that such a rotation is the smallest one with prefix aaab in lexicographic order, while the other rotations maintain their relative order. Therefore, $\mathrm{rot}^\star(\mathtt{aaab}) = \mathtt{b} \cdot \mathrm{rot}(\mathtt{aaab})$.

By Lemma 32, the rotation with prefix $\mathtt{a}s_2$ is still the smallest rotation starting with aab, with the difference that in this case, it ends with the last a of q_k. It follows that $\mathrm{rot}^\star(\mathtt{aab})$ is obtained by replacing the first b of $\mathrm{rot}(\mathtt{aab})$ with an a.

Both the order and the last letter of all the rotations having as prefix ab described in Lemma 33 is not affected from the insertion of the a, and therefore $\mathrm{rot}^\star(\mathtt{ab}) = \mathrm{rot}(\mathtt{ab})$.

Let us now consider all the rotations of w_k with prefix $\mathtt{b}^j\mathtt{a}s_2$, for all $1 \leq j \leq k$. One can notice that $w_k\mathtt{a}$ does not have any rotation starting with $\mathtt{b}^j\mathtt{a}s_2$, for all $1 \leq j \leq k$, but instead it has rotations starting with $\mathtt{b}^j\mathtt{aa}s_2$. Thus, for all $1 \leq j \leq k-1$, to obtain $\mathrm{rot}^\star(\mathtt{b}^j\mathtt{a})$ from $\mathrm{rot}(\mathtt{b}^j\mathtt{a})$ we have to remove the b in correspondence of the rotations starting with $\mathtt{b}^j\mathtt{a}s_2$, and add a b in correspondence of the rotations $\mathtt{b}^j\mathtt{aa}s_2$.

By Lemmas 34, 35, and 36, such rotations are placed right before the rotation starting with $\mathtt{b}^j\mathtt{aa}e_2$.

Finally, the last rotation has still the same prefix $\mathtt{b}^k\mathtt{a}$ and ends with an $\mathtt{a}$, and the thesis follows. □

The word w_k^{del} is obtained by removing the last letter from w_k, that is $w_k^{del} = w_k[1, n-1]$. For simplicity of exposition, we use the notation $\widehat{w_k} = w_k[1, n-1]$.

Lemma 39 (BWT of $\widehat{w_k}$)**.** *Given an integer $k > 5$, for $\widehat{w_k}$ it holds that*

$$
\begin{aligned}
rot^\star(\mathtt{a}^i\mathtt{b}) &= \mathtt{ba}^{k-i-2} \textit{ for all } 4 \le i \le k-2,\\
rot^\star(\mathtt{a}^3\mathtt{b}) &= \mathtt{b}^5(\mathtt{ab})^{k-6}\mathtt{a},\\
rot^\star(\mathtt{a}^2\mathtt{b}) &= \mathtt{aaba}^{2k-8},\\
rot^\star(\mathtt{ab}) &= \mathtt{b}^{k-2}\mathtt{baba}^{2k-6},\\
rot^\star(\mathtt{ba}) &= \mathtt{a}^{k-5}\mathtt{bbbab}^{k-5}\mathtt{ab}^{k-2}\mathtt{ba},\\
rot^\star(\mathtt{b}^j\mathtt{a}) &= \mathtt{ab}^{2k-2j-2}\mathtt{ab} \textit{ for all } 2 \le j \le k-1 \textit{ and}\\
rot^\star(\mathtt{b}^k\mathtt{a}) &= \mathtt{a}.
\end{aligned}
$$

Hence, $bwt(\widehat{w_k}) = \prod_{i=2}^{k-1} rot^\star(\mathtt{a}^{k-i}\mathtt{b}) \cdot \prod_{i=1}^{k} rot^\star(\mathtt{b}^i\mathtt{a})$. Moreover, it holds that $r(\widehat{w_k}) = 8k - 20$.

Proof. Analogously to the previous lemma, if we look in Lemmas 29, 30, and 31, at the structure of the BWT in correspondence of the rotations starting with $\mathtt{a}^i\mathtt{b}$, for all $3 \le i \le k-2$, we can notice that the order or the letters in the BWT is not affected. Thus, for all $3 \le i \le k-2$, we have $\mathtt{rot}^\star(\mathtt{a}^i\mathtt{b}) = \mathtt{rot}(\mathtt{a}^i\mathtt{b})$.

Since the last $\mathtt{a}$ of q_k is omitted, the circular factor $\mathtt{a}s_2$ does not appear anymore in $\widehat{w}$. Thus, $\mathtt{rot}^\star(\mathtt{aab})$ is obtained by removing the first $\mathtt{b}$ from $\mathtt{rot}(\mathtt{aab})$, since by Lemma 32 it is in correspondence of the rotation with prefix $\mathtt{a}s_2$.

On the other hand we can observe from Lemma 33 that the rotation with prefix s_2 maintains its relative order also in $\widehat{w_k}$, but its last letter is now a $\mathtt{b}$ instead of an $\mathtt{a}$.

For each $1 \le j \le k$, the rotation starting with $\mathtt{b}^j\mathtt{a}s_2$ of w_k does not appear in $\widehat{w_k}$, but in fact it is replaced by one having $\mathtt{b}^js_2$ as prefix and ending in the same way. When $j = 1$, by Lemma 34 such a rotation is located between the last two rotations

with the prefix $\mathtt{ba}$, which start by $\mathtt{bab}s_3$ and $\mathtt{b}s_3$ respectively. When $2 \leq j \leq k-1$, by Lemmas 35 and 36, the rotation starting with $\mathtt{b}^j s_2$ is greater than all the other rotations with prefix $\mathtt{b}^j\mathtt{a}$. Thus, for all $1 \leq j \leq k-1$, we obtain $\mathrm{rot}^\star(\mathtt{b}^j\mathtt{a})$ by moving the $\mathtt{b}$ in correspondence of the rotation starting with $\mathtt{ba}s_2$ from $\mathrm{rot}(\mathtt{b}^j\mathtt{a})$ and placing it in correspondence of $\mathtt{b}^j s_2$. Finally, the last rotation has still the same prefix $\mathtt{b}^k\mathtt{a}$ and ends with an $\mathtt{a}$, and the thesis follows. □

For the last case, we consider the word w_k^{sub} obtained by replacing the last $\mathtt{a}$ of w_k with a $\mathtt{b}$, that is $w_k^{sub} = \widehat{w_k}\mathtt{b}$.

Lemma 40 (BWT of $\widehat{w_k}\mathtt{b}$)**.** *Given an integer $k > 5$, for $\widehat{w_k}\mathtt{b}$ it holds that*

$$
\begin{aligned}
rot^\star(\mathtt{a}^i\mathtt{b}) &= \mathtt{ba}^{k-i-2} \textit{ for all } 4 \leq i \leq k-2,\\
rot^\star(\mathtt{a}^3\mathtt{b}) &= \mathtt{b}^5(\mathtt{ab})^{k-6}\mathtt{a},\\
rot^\star(\mathtt{a}^2\mathtt{b}) &= \mathtt{aaba}^{2k-8},\\
rot^\star(\mathtt{ab}) &= \mathtt{b}^{k-2}\mathtt{baba}^{2k-6},\\
rot^\star(\mathtt{ba}) &= \mathtt{a}^{k-5}\mathtt{bbbab}^{k-5}\mathtt{ab}^{k-2}\mathtt{ba},\\
rot^\star(\mathtt{b}^j\mathtt{a}) &= \mathtt{ab}^{2k-2j-2}\mathtt{ab} \textit{ for all } 2 \leq j \leq k-1,\\
rot^\star(\mathtt{b}^k\mathtt{a}) &= \mathtt{b} \textit{ and}\\
rot^\star(\mathtt{b}^{k+1}\mathtt{a}) &= \mathtt{a}.
\end{aligned}
$$

Hence, $bwt(\widehat{w_k}\mathtt{b}) = \prod_{i=2}^{k-1} rot^\star(\mathtt{a}^{k-i}\mathtt{b}) \cdot \prod_{i=1}^{k+1} rot^\star(\mathtt{b}^i\mathtt{a})$. Moreover, it holds that $r(\widehat{w_k}\mathtt{b}) = 8k-20$.

Proof. For the rotations in correspondence of the rotations starting with an $\mathtt{a}$, notice that replacing the last $\mathtt{a}$ of w_k for a $\mathtt{b}$ or removing the last $\mathtt{a}$ affects the BWT in the same way. Therefore, $\mathrm{rot}^\star(\mathtt{a}^i\mathtt{b})$ is the same as Lemma 39 for all $1 \leq i \leq k-2$.

The same behaviour can be noticed on the rotations with prefix $\mathtt{b}^j\mathtt{a}$, for all $1 \leq j \leq k-1$, while the rotation starting with $\mathtt{b}^k\mathtt{a}$ is now preceded by a $\mathtt{b}$.

With respect to the other edit operations, we have the range of rotations starting with $\mathtt{b}^{k+1}\mathtt{a}$, which consists solely in $\mathtt{b}^{k+1}s_2 \cdots \mathtt{a}$. □

In Table 3.2 is shown the BWTs of the words w_k, $w_k^{ins} = w_k\mathtt{a}$, $w_k^{del} = \widehat{w_k}$, and $w_k^{sub} = \widehat{w_k}\mathtt{b}$, grouped by the shared common prefix of the rotations. The word in

the intersection of the column $\mathsf{rot}(u)$ with the row w is the range of $\mathsf{bwt}(w)$ in correspondence to all the rotations having u as a prefix. The columns $\mathsf{rot}(\mathsf{a}^i\mathsf{b})$ and $\mathsf{rot}(\mathsf{b}^j\mathsf{a})$ represent ranges of columns for $i \in [k-2, 4]$ (in that order) and $j \in [2, k-1]$, respectively. Note that the all u in the headings of the columns are prefix-free among them, and cover all the possible ranges for the set of words considered. The BWT of each word is the concatenation of all the words in its row from left to right. In the last column appears the number of BWT runs of each of these words.

Word	$\mathsf{rot}(\mathsf{a}^i\mathsf{b})$	$\mathsf{rot}(\mathsf{a}^3\mathsf{b})$	$\mathsf{rot}(\mathsf{a}^2\mathsf{b})$	$\mathsf{rot}(\mathsf{ab})$
w_k	ba^{k-i-2}	$\mathsf{b}^5(\mathsf{ab})^{k-6}\mathsf{a}$	baaba^{2k-8}	$\mathsf{b}^{k-2}\mathsf{aaba}^{2k-6}$
$w_k\mathsf{a}$	ba^{k-i-2}	$\mathsf{bb}^5(\mathsf{ab})^{k-6}\mathsf{a}$	aaaba^{2k-8}	$\mathsf{b}^{k-2}\mathsf{aaba}^{2k-6}$
$\widehat{w_k}$	ba^{k-i-2}	$\mathsf{b}^5(\mathsf{ab})^{k-6}\mathsf{a}$	aaba^{2k-8}	$\mathsf{b}^{k-2}\mathsf{baba}^{2k-6}$
$\widehat{w_k}\mathsf{b}$	ba^{k-i-2}	$\mathsf{b}^5(\mathsf{ab})^{k-6}\mathsf{a}$	aaba^{2k-8}	$\mathsf{b}^{k-2}\mathsf{baba}^{2k-6}$

Word	$\mathsf{rot}(\mathsf{ba})$	$\mathsf{rot}(\mathsf{b}^j\mathsf{a})$	$\mathsf{rot}(\mathsf{b}^k\mathsf{a})$	$\mathsf{rot}(\mathsf{b}^{k+1})$	$r(\cdot)$
w_k	$\mathsf{a}^{k-5}\mathsf{bbbab}^{k-4}\mathsf{ab}^{k-2}\mathsf{a}$	$\mathsf{ab}^{2k-2j-1}\mathsf{a}$	a	ϵ	$6k-12$
$w_k\mathsf{a}$	$\mathsf{a}^{k-5}\mathsf{bbbbab}^{k-5}\mathsf{ab}^{k-2}\mathsf{a}$	$\mathsf{bab}^{2k-2j-2}\mathsf{a}$	a	ϵ	$8k-20$
$\widehat{w_k}$	$\mathsf{a}^{k-5}\mathsf{bbbab}^{k-5}\mathsf{ab}^{k-2}\mathsf{ba}$	$\mathsf{ab}^{2k-2j-2}\mathsf{ab}$	a	ϵ	$8k-20$
$\widehat{w_k}\mathsf{b}$	$\mathsf{a}^{k-5}\mathsf{bbbab}^{k-5}\mathsf{ab}^{k-2}\mathsf{ba}$	$\mathsf{ab}^{2k-2j-2}\mathsf{ab}$	b	a	$8k-20$

TABLE 3.2: BWTs of the words w_k and its variants after an insertion, a deletion, and a substitution respectively.

Finally, we compare the measures r over w_k and the words obtained by applying an edit operation.

Proposition 41. *There exists an infinite family of words w such that:*

1. $r(w^{ins}) - r(w) = \Theta(\sqrt{n})$;
2. $r(w^{del}) - r(w) = \Theta(\sqrt{n})$;
3. $r(w^{sub}) - r(w) = \Theta(\sqrt{n})$;

where w^{ins}, w^{del}, and w^{sub} are words obtained after applying an insertion, a deletion, and a substitution to w respectively.

Proof. Such a family is composed of the words w_k with $k > 5$. Let $n = |w_k|$. If $w_k^{ins} = w_k\mathsf{a}$, $w_k^{del} = \widehat{w_k}$, and $w_k^{sub} = \widehat{w_k}\mathsf{b}$, from Table 3.2 we have that $r(w_k\mathsf{a}) = r(\widehat{w_k}) = r(\widehat{w_k}\mathsf{b}) = r(w_k) + (2k-8)$. From Lemma 27 it can be derived that $2k - 8 = \Theta(\sqrt{n})$. □

This lower bound is the best on the additive sensitivity of the measure r known so far, and it represents a significative improvement over the previous lower bound $\Omega(\log n)$ by [1].

On the other hand, for the family of words w_k for all $k \geq 5$, it holds that $r(w) = 6k - 12 = \Theta(\sqrt{n})$, and for each $w'_k \in \{w_k\mathtt{a}, \widehat{w_k}, \widehat{w_k}\mathtt{b}\}$, we have $r(w) = 8k - 20 = \Theta(\sqrt{n})$. This implies that for our example we obtain for r a $\Theta(\sqrt{n})$ additive sensitivity, while the multiplicative sensitivity is not affected at all, that is $r(w'_k)$ grows as $r(w_k)$. In particular, we can express such a lower bound in terms of r over the original word.

Theorem 42. *The following lower bounds on the sensitivity of the measure r over binary words $w \in \{\mathtt{a}, \mathtt{b}\}$ with respect to measure r hold:*

1. $AS_{ins}(r, n) = \Omega(r)$;
2. $AS_{del}(r, n) = \Omega(r)$;
3. $AS_{sub}(r, n) = \Omega(r)$.

Proof. The proof follows by comparing the size r over w_k from Proposition 37, with r over $w_k\mathtt{a}$, $\widehat{w_k}$, and $\widehat{w_k}\mathtt{b}$, from Lemmas 38, 39, and 40 respectively. For the insertion, for all $k \geq 5$ we have that $r(w_k\mathtt{a}) - r(w_k) = 2k - 8$, and therefore $\mathsf{AS}_{ins}(r, n) \geq 2k - 8 \geq \frac{1}{9}(6k - 12) \geq \frac{1}{9}r(w_k) = \Omega(r)$. The analogous bounds are obtained for $\mathsf{AS}_{del}(r, n)$ and $\mathsf{AS}_{sub}(r, n)$ in the same way. □

Best to our knowledge, it has not been found an example of word for which an edit operation affects both the multiplicative and additive sensitivity more than a logarithmic factor.

Furthermore, Akagi et al. showed by using a bound in [67] that an upper bound on any worst-case additive sensitivity of the measure r is $\mathcal{O}(r \log n \log r)$. We wonder then if the bound from Theorem 42 is the best we could get, or if there is actually something in between $\Omega(r)$ and $O(r \log n \log r)$.

We conclude this chapter with a final remark on the measures $r_\$$. Quite often r and $r_\$$ have been improperly considered as interchangeable measures. By using the same words of this section, it was possible to derive the following separation between the two measures.

Proposition 43. *There exists an infinite family of words w such that $r_{\$}(w)/r(w) = \Theta(\log n)$, where $n = |w|$. Moreover, there exists another infinite family of words w' such that $r_{\$}(w') - r(w') = \Theta(\sqrt{n'})$, where $n' = |w'|$.*

Chapter 4

Measure r on Purely Morphic Words

Despite the theoretical limits of the measure r described in the previous chapter, the Burrows-Wheeler Transform is still at the basis of the most used tools in bioinformatics for handling huge collections of genomes. Indeed, the BWT gives its best when it is applied on highly repetitive texts. It is therefore natural to wonder how the BWT behaves when families of repetitive words are considered.

Some families of prolongable morphisms can be seen as mechanism to create strings with long repetitions. For instance, let us consider a k-uniform morphism $\mu : \Sigma \mapsto \Sigma$ for some $k \geq 2$, and let t be the smallest integer such that $\mu^t(a)$ contains $m \geq 2$ occurrences of any letter $b \in \Sigma$. Then, for all $n > 0$, all the following iterations $\mu^{n+t}(a) = \mu^n(\mu^t(a))$ will contain m occurrences of the factors $\mu^n(b)$, and the lengths of these factors grows exponentially with n.

Even though not all morphisms verify the property just mentioned, Pansiot [104] proved that the factor complexity of purely morphic words cannot grow more than a quadratic function. More in detail, given a morphism $\mu : \Sigma \mapsto \Sigma$, Pansiot [104] showed how to relate the factor complexity with the growth functions over the letters in Σ.

Theorem 44 ([104]). *Let $\mathbf{x} = \mu^\infty(a)$ be an infinite aperiodic word for some morphism $\mu : \Sigma \mapsto \Sigma$, and let $p_{\mathbf{x}}$ be its factor complexity.*

1. *If μ is growing, then $p_{\mathbf{x}}(n)$ is $\Theta(n)$, $\Theta(n \log\log n)$ or $\Theta(n \log n)$ if μ is quasi-uniform, polynomially divergent or exponentially divergent, respectively.*

2. *If μ is not-growing, let B be the set of its bounded letters:*

   (a) *if $\mathbf{x}$ has arbitrarily large factors of B^* then $p_{\mathbf{x}}(n) = \Theta(n^2)$*

   (b) *if the factors of B^* in $\mathbf{x}$ have bounded length then $p_{\mathbf{x}}(n)$ can be any of $\Theta(n)$, $\Theta(n \log\log n)$, or $\Theta(n \log n)$.*

Representing strings through applications of morphisms is indeed a well-known subject. Both L-systems [117] and the more recent ν-systems [101], are based on morphisms, and their space of representation is in fact optimal, since these can be represented in an extreme compact space. However, this representation implies an extra cost for the extraction of the string.

The problem of considering the BWT of purely morphic words has been already object of study in the literature. For instance, Mantaci et al. [94] implicitly proved that at each iteration of a Sturmian morphism $\mu : \{\mathtt{a}, \mathtt{b}\} \mapsto \{\mathtt{a}, \mathtt{b}\}$ the BWT is perfectly clustered, that is, $r(\mu^n(\mathtt{a})) = 2$ for all $n > 0$ (see Theorem 14), while more recently Blrek et al. [20] have studied the BWT of different families of purely morphic words, like the finite Thue-Morse words generated from the morphism $\tau \equiv (\mathtt{ab}, \mathtt{ba})$. For this family of words, they showed that the $r(\tau^n(\mathtt{a})) = r(\tau^n(\mathtt{b})) = 2n$ for all $n > 0$. Given the regularity on the construction of words via morphisms, it is legit to wonder if the same $\mathcal{O}(n)$ upper bound holds for other classes of morphic sequences.

In this chapter, we focus on the problem of studying the measure r for purely morphic words.

First, in Section 4.1 we try to obtain theoretical bounds based on the comparison between r and other notions of repetitiveness already explored for purely morphic words.

Then, in Section 4.2 we show some combinatorial properties of binary morphisms, and how these can be used in Section 4.3 to prove that, for a given morphism $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \{\mathtt{a}, \mathtt{b}\}^*$ prolongable on $\mathtt{a}$, the measure r grows linearly with the iteration of the morphism, that is $r(\mu^n(\mathtt{a})) = \mathcal{O}(n)$.

Finally, in Section 4.4 we make a comparison between the measure r of a purely morphic word and the size of the run-length encoding applied on the word itself. This study extend the results on the compression ratio of the BWT by Mantaci et al. [93] to the family of purely morphic words.

4.1 Comparison with other Repetitiveness Measures

The Burrows-Wheeler Transform is indeed efficient on periodic words. In fact, for every word w, it holds that $r(w^n) = r(w) \leq |w|$ for all $n > 0$. More in general, this result can be extended for prefixes of eventually periodic words.

Lemma 45. *Let $\mathbf{x} = uv^\omega$ be a eventually periodic word, for some $u \in \Sigma^*$ and some primitive word $v \in \Sigma^+$. Then, $r(\mathbf{x}[1, n]) = \mathcal{O}(1)$ for every $n > 0$.*

Eventually periodic words are a clear example of words for which the measure r is independent from the length of the prefix considered. We remark that also purely morphic words can be eventually periodic, as showed in the following example.

Example 46. *Let $\mu \equiv (\mathtt{ab}, \mathtt{bb})$. One can see that the first iterations of the morphism μ over* a *are*

$$\mu(\mathtt{a}) = \mathtt{ab}$$
$$\mu^2(\mathtt{a}) = \mathtt{abbb}$$
$$\mu^3(\mathtt{a}) = \mathtt{abbbbbbb}$$
$$\mu^4(\mathtt{a}) = \mathtt{abbbbbbbbbbbbbbb}$$

and the fixed point is $\mathbf{x} = \mu^\infty(\mathtt{a}) = \mathtt{ab}^\omega$.

The study of compression schemes applied to morphic sequences is not new in the field. For instance, Constantinescu and Ilie [33] have studied the size of the LZ77-factorization z of purely morphic words, and proved the following results.

Proposition 47 ([33])**.** *For a non-erasing morphism μ that admits the fixed point $\mathbf{x} = \mu^\infty(a)$, for any iteration i of the morphism it holds that $z(\mu^i(a))$ is either $\Theta(1)$, if $\mathbf{x}$ is eventually periodic, or $\Theta(i)$, otherwise, where $i \geq 0$ is the iteration of the morphism.*

Note that the same $\mathcal{O}(1)$ upper bound is shared with the size z when the purely morphic words are eventually periodic. If we focus on aperiodic words, we can use the bound of Proposition 47 to get a basic bound on r.

Corollary 48. *Let $\mu : \Sigma \mapsto \Sigma$ be a non-erasing morphism prolongable on $a \in \Sigma$, and let $\mathbf{x} = \mu^\infty(a)$ be its fixed point. If $\mathbf{x}$ is aperiodic, then $r(\mu^i(a)) = \mathcal{O}(i^3)$, where $i \geq 0$ is the iteration of the morphism.*

Proof. From Proposition 47 it holds that $z(\mu^i(a)) = \Theta(i)$. Moreover, [67] showed that for every word w holds that $r(w) = \mathcal{O}(z(w)\log^2 n)$, where $n = |w|$, and by [117] we know that for each purely morphic word there exists a constant $\rho_a > 1$ such that $\mu_a(i) = \mathcal{O}(\rho_a^i)$, where μ_a is the growth function of μ over $\mathtt{a}$. Hence, we get $r(\mu^i(a)) = \mathcal{O}(z(\mu^i(a))\log^2 \mu_a(i)) = \mathcal{O}(i \log^2 \rho_a^i) = \mathcal{O}(i^3)$. □

However, with bounded-size alphabets and bounded-size length of the images, we were not able to find a sequence of finite purely morphic words for which the measure r seems to grow more than linearly. Nonetheless, we can obtain a tighter upper bound with the comparison with the measure δ.

At the beginning of this chapter we have mentioned that for any purely morphic word $\mathbf{x} = \mu^\infty(a)$, its the factor complexity $p_\mathbf{x}$ grows at most as a quadratic function. Since $p_{\mu^n(a)}(k) \leq p_\mathbf{x}(k)$ for all $n, k > 0$, we can use the factor complexity to get an upper bound on the δ-measure. In particular, when $p_\mathbf{x}$ is linear, we obtain a tight upper bound.

Proposition 49. *Let $\mu : \Sigma \mapsto \Sigma$ be a morphism prolongable on $a \in \Sigma$, and let $\mathbf{x} = \mu^\infty(a)$ be its fixed point. If $p_\mathbf{x}(n) = \Theta(n)$, then $r(\mu^i(a)) = \mathcal{O}(i)$, where $i \geq 0$ is the iteration of the morphism.*

Proof. Let $w_i = \mu^i(a)$ be the ith iterate of the morphism $\mu : \Sigma \mapsto \Sigma$ on the letter $a \in \Sigma$. If $p_x(n) = \Theta(n)$, then there exist $t > 0$ such that $p_{w_i}(k) \leq p_\mathbf{x}(k) \leq t \cdot k$, for all $k, i > 0$. This implies that $\delta(w_i) \leq t$, for all $i \geq 0$. Thus, $\delta(w_i) \in \mathcal{O}(1)$ implies that there exists a constant t' such that $\delta(w_i)\log\delta(w_i) \leq t'$. Moreover, [117], showed that for all morphic sequences it holds $\mu_a(n) = \mathcal{O}(\rho_a^n)$, for some $\rho_a > 1$. By Theorem 15, recall that for all words w it holds that $r_\$(w) = \mathcal{O}(\delta(w)\log\delta(w)\max\{1, \log\frac{|w|}{\delta(w))}\}$. Since t' and ρ_a are constant values, and assuming without loss of generality that

$\log \frac{|w_i|}{\delta(w_i)\log\delta(w_i)} > 1$, there exist constant values $c_1, c_2 > 0$ such that:

$$\begin{aligned}\delta(w_i)\log\delta(w_i)\log\frac{|w_i|}{\delta(w_i)\log\delta(w_i)} &\leq t'(\log|w_i| - \log t')\\ &\leq t'(c_1 \cdot i\log\rho_a - \log t')\\ &\leq c_2 \cdot i.\end{aligned}$$

Finally, by Lemmas 16 and 17, it follows that $r(w_i) = \Theta(r_\$(w_i)) = \mathcal{O}(i)$. □

To show that this bound is tight, it is sufficient to consider the Thue-Morse morphism τ, for which [20] showed that $r(\tau^n(\mathtt{a})) = 2n = \Theta(n)$. Analogous bounds can be obtained for the purely morphic words $\mathbf{x}$ with either factor complexity $\Theta(n\log\log n)$, $\Theta(n\log n)$, or $\Theta(n^2)$, but for these cases the measure δ is not bounded by a constant.

Proposition 50. *Let $\mu : \Sigma \mapsto \Sigma$ be a morphism prolongable on $a \in \Sigma$, and let $\mathbf{x} = \mu^\infty(a)$ be its fixed point. If we denote by $i \geq 0$ is the iteration of the morphism, the following bounds hold:*

1. *$p_\mathbf{x}(n) = \Theta(n\log\log n)$, then $r(\mu^i(a)) = \mathcal{O}(i\log i\log\log i)$;*
2. *$p_\mathbf{x}(n) = \Theta(n\log n)$, then $r(\mu^i(a)) = \mathcal{O}(i^2\log i)$.*
3. *$p_\mathbf{x}(n) = \Theta(n^2)$, then $r(\mu^i(a)) = \mathcal{O}(i^2\log i)$.*

Proof. Analogously to the proof of Proposition 49, let $w_i = \mu^i(a)$ be the ith iterate of the morphism $\mu : \Sigma \mapsto \Sigma$ on the letter $a \in \Sigma$. Moreover, recall that for all morphic sequences it holds $\mu_a(i) = \mathcal{O}(\rho_a^i)$, for some $\rho_a > 1$ ([117]). In all three cases, we aim to apply Theorem 15.

1. $p_\mathbf{x}(n) = \Theta(n\log\log n)$: Since $p_{w_i}(k) \leq p_\mathbf{x}(k) \leq t \cdot k\log\log k$ for every $i \geq 0$, $k > 0$ and for some $t > 0$, we have $\delta(w_i) \leq t \cdot \log\log|w_i|$. Hence, $\delta(w_i)\log\delta(w_i) \cdot \max\{1, \log\frac{|w_i|}{\delta(w_i)\log\delta(w_i)}\} \leq t'\log\log|w_i|\log\log\log|w_i| \cdot \log|w_i|$ for some $t' > 0$, and by Theorem 15, we have $r_\$(w_i) = \mathcal{O}(i\log i\log\log i)$.

2. $p_\mathbf{x}(n) = \Theta(n\log n)$: Analogously to the previous case, $p_{w_i}(k) \leq p_\mathbf{x}(k) \leq t \cdot k\log k$, for any $i \geq 0$, $k > 0$ and for some $t > 0$, and therefore $\delta(w_i) \leq t \cdot \log|w_i|$. Hence, $\delta(w_i)\log\delta(w_i) \cdot \max\{1, \log\frac{|w_i|}{\delta(w_i)\log\delta(w_i)}\} \leq t'\log|w_i|\log\log|w_i| \cdot \log|w_i|$ for some $t' > 0$, and by Theorem 15, we have $r_\$(w_i) = \mathcal{O}(i^2\log i)$.

3. $p_{\mathbf{x}}(n) = \Theta(n^2)$: For this last case we take a different route. Recall that $\delta(w) \leq z(w)$ for every word w [74]. Then, by Proposition 47 follows that $\delta(w_i) = O(z(w_i)) = O(i)$. Thus, as for case 2., if we replace i to δ to the bound from Theorem 15 get $r_{\$}(w_i) = \mathcal{O}(i^2 \log i)$.

Finally, by Lemmas 16 and 17, the same bounds holds for r. □

In general, we want to point out that a different order of the letters of the alphabet can affect the size of the BWT [10], but this is not the case for binary words. In fact, given a binary word $w \in \{\mathtt{a}, \mathtt{b}\}^*$ with $\mathtt{a} < \mathtt{b}$, computing the BWT of w after changing the order of the alphabet is equivalent to compute the BWT for $\overline{w}$, and for binary words hold that $r(w) = r(\overline{w})$. We will return to this result later in the next chapter.

4.2 Combinatorial Properties of Binary Morphisms

In this section we focus on combinatorial properties of binary morphisms and binary purely morphic words. Recall that the order of a binary alphabet is not relevant for the size of the BWT. Thus, from now on we assume that a binary morphism $\mu : \Sigma_2 \mapsto \Sigma_2$ is defined over the alphabet $\Sigma_2 = \{\mathtt{a}, \mathtt{b}\}$ with $\mathtt{a} < \mathtt{b}$, and that it is prolongable on $\mathtt{a}$.

To study the BWT of binary purely morphic words, we describe the structure of different morphisms based on common properties of their fixed points. Note that, unlike words built on greater alphabets, the size r is independent from the order of the letters, since the other order possible is the symmetric. In fact, given word w, if $\leq^c$ is the reverse order over $\{\mathtt{a}, \mathtt{b}\}$, then for each pair of rotations $w_1, w_2 \in \mathcal{R}(w)$ holds that $w_1 \leq w_2$ if and only if $w_1 \geq^c w_2$, and therefore the BWT is symmetric too.

The first group that we handle is a characterization of binary morphisms with eventually periodic fixed points. Here we report the procedure that Pansiot presented to decide whether or not a word is eventually periodic [105]. The correctness is detailed in the same work.

Proposition 51 (Corollary 3 of [105]). *If $\mathbf{x} = \mu^\infty(a)$, with $\mu : \Sigma^* \mapsto \Sigma^*$ elementary, we can decide if $\mathbf{x}$ is eventually periodic.*

Proof. Let $G, B \subseteq \Sigma$ be respectively the set of growing letters and the set of bounded letters.

- If $\mu(a)$ contains only one occurrence of one letter from G, then $\mathbf{x}$ is eventually periodic.
- If $\mu(a)$ contains several occurrences of letters from G, then:
  - compute the length n of the shortest prefix that contains two occurrences of the same letter $\mathbf{x}[n] \in G$;
  - we can then factorize $\mathbf{x}[1, n] = u_0 c_1 u_1 c_2 u_2 \cdots u_{k-1} c_k u_k \cdots c_m u_m c_k$, where $c_i \neq c_{i'} \in C$ for all $1 \leq i < i' \leq m$, $c_k = x[n]$ for some $k \in [1, m]$, and $u_j \in B^*$ for all $j \in [0, m]$;
  - if for all $c_i u'$ prefix of $c_i u_i$ for all $i \in [1, m]$ the factor $c_i u'$ is not right special, then $\mathbf{x}$ is eventually periodic.

The correctness of the procedure is given by [105, Lemma 2]. □

We can use the procedure described to derive the following proposition.

Proposition 52. *Let $\mathbf{x} = \mu^\infty(\mathtt{a})$ be an infinite binary eventually periodic word, where $\mu \equiv (\alpha, \beta)$. Then, one of the following cases occurs:*

1. *$\alpha = u^p$ and $\beta = u^q$, for some $u \in \{\mathtt{a}, \mathtt{b}\}^*$ and some $p, q \geq 1$;*
2. *$\alpha = \mathtt{ab}^k$ and $\beta = \mathtt{b}^\ell$, for some $k \geq 1$, $\ell \geq 1$;*
3. *$\alpha = (\mathtt{ab})^p\mathtt{a}$ and $\beta = (\mathtt{ba})^q\mathtt{b}$ for some $p, q \geq 1$;*
4. *$\alpha = (\mathtt{ab}^p)^q\mathtt{a}$ and $\beta = \mathtt{b}$, for some $p \geq 1$, $q \geq 1$.*

Proof. Let μ be a simplifiable morphism. This means that exist $\eta_1 : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \{\mathtt{a}\}^*$ and $\eta_2' : \{\mathtt{a}\}^* \to \{\mathtt{a}, \mathtt{b}\}^*$ such that $\mu(w) = \eta_2(\eta_1(w))$ for every $w \in \{\mathtt{a}, \mathtt{b}\}^*$. More into detail, we have that $\eta_1(\mathtt{a}) = \mathtt{a}^p$ and $\eta_1(\mathtt{b}) = \mathtt{a}^q$ for some $p, q > 0$, and $\eta_2(\mathtt{a}) = u$ for some $u \in \{\mathtt{a}, \mathtt{b}\}^*$. It is easy to see that the fixed point of these morphisms is $\mathbf{x} = u^\infty$, that is eventually periodic, and therefore $\mu(\mathtt{a}) = \eta_2(\eta_1(\mathtt{a})) = \eta_2(\mathtt{a}^p) = u^p$, and $\mu(\mathtt{b}) = \eta_2(\eta_1(\mathtt{b})) = \eta_2(\mathtt{a}^q) = u^q$ (that is the case *1.*).

Let us suppose that μ is elementary, and let G and B be the set of growing and bounded letters respectively. Since we assume the morphism is prolongable on $\mathtt{a}$,

we have that $\mathtt{a} \in G$. Note that this can occur only when $|\mu(\mathtt{a})|_{\mathtt{a}} = 1$ and $\mathtt{b} \in B$, that is $\mu \equiv (\mathtt{ab}^k, \mathtt{b})$ for every $k \geq 1$ (case *2.* for $\ell = 1$). From the procedure described in the proof of Proposition 51, if the fixed point of the morphism contains only one growing letter, then it is eventually periodic. Note that this can occur only when $|\mu(\mathtt{a})|_{\mathtt{a}} = 1$ and $\mathtt{b} \in B$, that is $\mu \equiv (\mathtt{ab}^k, \mathtt{b})$ for every $k \geq 1$ (case *2.* for $\ell = 1$).

Otherwise, let n be the length of the shortest prefix containing two occurrences of a same letter $c \in G$. We can then factorize

$$\mathbf{x}[1, n] = u_0 c_1 u_1 c_2 u_2 \cdots u_{k-1} u_k u_k \cdots u_{m-1} c_m u_m c_k,$$

with $c_i \in G$ for every $i \in [1, m]$ and $u_j \in B^*$ for every $j \in [0, m]$. It follows that $c_i \neq c_j$ for every $1 \leq i \neq j \leq m$. By Proposition 51, if for every prefix $c_i u'$ of every $c_i u_i$ factor of $\mathbf{x}[1, n]$ it holds that $e_r(c_i u') = 0$, then $\mathbf{x}$ is eventually periodic.

If $\mathtt{b} \in B$, then $\mathbf{x} = \mathtt{ab}^p\mathtt{a}$ for some $p \geq 0$. It follows that in order to have an eventually periodic word, $e_r(\mathtt{ab}^q) = 0$ for every $0 \leq q \leq p$, and the fixed point is $\mathbf{x} = (\mathtt{ab}^p)^\infty$. Since $\mu(\mathtt{a})$ is a prefix of $\mathbf{x}$, we have that $\mu(\mathtt{a}) = (\mathtt{ab}^p)^q\mathtt{ab}^{p'}$, for some $p, q \geq 1$ and $0 \leq p' \leq p$ (notice that if $p = 0$ or $q = 0$, then $\mathbf{x} = \mathtt{a}^\infty$ or $\mathbf{x} = \mathtt{ab}^\infty$ respectively). However, we can see that $\mu^2(\mathtt{a}) = \mu((\mathtt{ab}^p)^q\mathtt{ab}^{p'}) = (((\mathtt{ab}^p)^q\mathtt{ab}^{p+p'})^q(\mathtt{ab}^p)^q\mathtt{ab}^{p+p'})$. Note that the words $\mathtt{ab}^p\mathtt{a}$ and $\mathtt{ab}^{p+p'}\mathtt{a}$ appear as factors of $\mathbf{x}$. Since if $p' > 0$ we would have $e_r(\mathtt{ab}^p) = 1$, it follows that $p' = 0$ and $\mu \equiv ((\mathtt{ab}^p)^q\mathtt{a}, \mathtt{b})$ is eventually periodic for every pair $p, q \geq 1$ (case *4.*).

If $\mathtt{b} \in G$ (and therefore $B = \emptyset$), then either $\mathbf{x}[1, n] = \mathtt{abb}$, or $\mathbf{x}[1, n] = \mathtt{aba}$. Since it must hold that $e_r(\mathtt{a}) = e_r(\mathtt{b}) = 0$, then either $\mathbf{x} = \mathtt{ab}^\infty$ or $\mathbf{x} = (\mathtt{ab})^\infty$, whether the prefix of $\mathbf{x}$ is $\mathtt{abb}$ or $\mathtt{aba}$ respectively. If $\mathtt{abb}$ is prefix of $\mathbf{x}$, since $\mu(\mathtt{a})$ is prefix of the fixed point, it follows that $\mu \equiv (\mathtt{ab}^k, \mathtt{b}^\ell)$ for some $k \geq 1$ and $\ell > 1$ (case *2.* for $\ell > 1$). If $\mathtt{aba}$ is prefix of $\mathbf{x}$, note that by definition $\mathbf{x} = \mu(\mathbf{x}) = (\alpha\beta)^\infty$. Therefore, either $\mu \equiv ((\mathtt{ab})^p, (\mathtt{ab})^q)$ for some $p, q \geq 1$ (which falls in case *1.* again), or $\mu \equiv ((\mathtt{ab})^p\mathtt{a}, (\mathtt{ba})^q\mathtt{b})$ for some $p, q \geq 1$ (case *3.*), and the thesis follows. □

Symmetrically, we can use the characterization of Proposition 52 to recognize when a fixed point of a morphism is aperiodic. Recall that any primitive morphism μ is quasi-uniform. Hence, if $\mathbf{x} = \mu^\infty(\mathtt{a})$ is aperiodic, then by Theorem 44 its factor

complexity is always linear. Let us cover then the remaining cases, that is when μ is not primitive.

With the next lemma we show a general structure of non-primitive morphisms with aperiodic fixed-point.

Lemma 53. *Let μ be a binary non-primitive morphism prolongable on* $\mathtt{a}$ *with fixed point* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *aperiodic. Then* $\mu \equiv (\mathtt{a}u, \mathtt{b}^\ell)$ *for some* $u \in \Sigma^+$ *such that* $|u|_\mathtt{a}, |u|_\mathtt{b} \geq 1$ *and for some* $\ell \geq 1$.

Proof. In order to be prolongable on $\mathtt{a}$, we must have $\mu(\mathtt{a}) = \mathtt{a}u$ for some $u \in \Sigma^+$. Moreover, $|u|_\mathtt{b} \geq 1$, otherwise $\mu^\infty(\mathtt{a}) = \mathtt{aaaaaa}\ldots$. Since $\mu(\mathtt{a})$ contains both $\mathtt{a}$ and $\mathtt{b}$, and the morphism is not primitive, then $|\mu(\mathtt{b})|_\mathtt{a} = 0$, i.e. $\mu(\mathtt{b}) = \mathtt{b}^\ell$ for some $\ell \geq 1$. Finally, we can observe that if $|u|_\mathtt{a} = 0$, then we have the morphism $\mu \equiv (\mathtt{ab}^k, \mathtt{b}^\ell)$ that is eventually periodic (Proposition 52). Therefore, $|u|_\mathtt{a} \geq 1$ and the thesis follows. □

With the next propositions, we describe how to recognize the factor complexity of a non-primitive purely morphic word based on the structure of the morphism that generates it.

Proposition 54. *Let $\mu = (\alpha, \beta)$ be a non-growing morphism prolongable on* $\mathtt{a}$ *and let* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *be its aperiodic fixed point. Then, one of the following cases must occur:*

1. $\alpha = \mathtt{a}u\mathtt{ba}^k$, $\beta = \mathtt{b}$, *for some* $u \in \Sigma^*$ *and* $k \geq 1$, *and* $p(n) = \Theta(n)$;
2. $\alpha = \mathtt{a}u\mathtt{ab}^k$, $\beta = \mathtt{b}$, *for every* $u \in \Sigma^*$ *and* $k \geq 1$, *and* $p_\mathbf{x}(n) = \Theta(n^2)$.

Proof. Let B be the set of bounded letters in the morphism μ. By Theorem 44, we know that every morphism that generates an infinite word $\mathbf{x} = \mu^\infty(\mathtt{a})$ with factor complexity $p_\mathbf{x}(n) = \Theta(n^2)$ is non-growing and $\mathbf{x}$ contains infinitely many runs of letters over B^* with unbounded lengths. By Lemma 53, we know that $\mu \equiv (\mathtt{a}u', \mathtt{b}^\ell)$, for some u' such that $|u'|_\mathtt{a}, |u'|_\mathtt{b} \geq 1$ and $\ell \geq 1$. Since μ is growing on $\mathtt{a}$, we have that $\ell = 1$ (that is $\mu(\mathtt{b}) = \mathtt{b}$). One can see that, if $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b})$ for some $k \geq 1$, then $\mathbf{x}$ contains the factor $\mathtt{ab}^{k+i}\mathtt{a}$ for infinitely many $i > 0$, and therefore $\mathbf{x} = \Theta(n^2)$ (more details can be found later in Lemma 58).

On the other hand, if $\mu \equiv (\mathtt{a}u\mathtt{ba}^k, \mathtt{b})$, then there is a finite number of values $i \geq 0$ for which the words $\mathtt{ab}^i\mathtt{a}$ are factors of $\mathbf{x}$. By [104, Theorem 4.1], we know

that if the number of factors of $\mu^\infty(\mathtt{a})$ in B^* is bounded, then there exist a growing morphism $\mu' : \Sigma'^* \mapsto \Sigma'^*$ with fixed point $\mathbf{x}' = \mu'^\infty(a')$ for some $a' \in \Sigma'$, a morphism $\eta : \Sigma'^* \mapsto \Sigma^*$ such that $\eta(\mu'^\infty(a')) = \mu^\infty(\mathtt{a})$, and $p_{\mathbf{x}}(n) = \Theta(p_{\mathbf{x}'}(n))$.

In the proof of [104, Theorem 4.1] it is described how to build the alphabet Σ', and the morphisms μ' and η_2. The correctness of the procedure can be verified in the same work.

Let us consider as a unique letter every factor of the type $u_1 c u_2$ that occurs in $\mathbf{x}$, where c belongs to the set G of growing letters, and $u_1, u_2 \in B^*$. Since $\mathtt{a} \in G$ and $\mathtt{b} \in B$, it follows that these letters can be only of the type $\{[\mathtt{b}^i\mathtt{a}\mathtt{b}^j], \mathtt{b}^i\mathtt{a}\mathtt{b}^j \in F(\mathbf{x})\}$, and therefore the alphabet is finite by hypothesis.

Let $\mu(u_1\mathtt{a}u_2) = v_1\mathtt{a}v_2\mathtt{a}v_2\mathtt{a}\cdots v_{m-1}\mathtt{a}v_m$, for some $u_1, u_2 \in B^*$, some $k \geq 1$, and some $v_i \in B^*$ for all $i \in [1, m]$. Then, $\mu'([u_1\mathtt{a}u_2]) = [v_1\mathtt{a}][v_2\mathtt{a}]\cdots[v_{m-1}\mathtt{a}v_m]$. Since we can factorize α as $\mu(\mathtt{a}) = \mathtt{a}\mathtt{b}^{p_1}\mathtt{a}\mathtt{b}^{p_2}\cdots\mathtt{b}^{p_m}\mathtt{a}$ for some $p_i \geq 0$ for all $i \in [1, m]$, we can derive Σ', μ' and η as follows:

- the alphabet $\Sigma' = \{[\mathtt{a}]\} \cup \bigcup_{i=1}^{m}\{[\mathtt{b}^{p_i}\mathtt{a}]\}$;
- the morphism $\mu' : \Sigma'^* \mapsto \Sigma'^*$ as $\mu'([\mathtt{a}]) = [\mathtt{a}][\mathtt{b}^{p_1}\mathtt{a}][\mathtt{b}^{p}2\mathtt{a}]\cdots[\mathtt{b}^{p_m}\mathtt{a}]$, and for each j such that $[\mathtt{b}^j\mathtt{a}] \in \Sigma'$, we have $\mu'([\mathtt{b}^{p_i}\mathtt{a}]) = [\mathtt{b}^{p_i}\mathtt{a}][\mathtt{b}^{p_1}\mathtt{a}][\mathtt{b}^{p_2}\mathtt{a}]\cdots[\mathtt{b}^{p_m}\mathtt{a}]$;
- the morphism $\eta : \Sigma'^* \mapsto \Sigma^*$ as $\eta([u]) = u$, for all $u \in \Sigma^*$.

One can verify that $\eta(\mu'^i([\mathtt{a}])) = \mu^i(\mathtt{a})$ for all $i \geq 0$. Moreover, the morphism μ' is $(m+1)$-uniform, for some $m \geq 1$, and by Theorem 44 it follows that $p'_{\mathbf{x}}(n) = \Theta(n)$ when $\mathbf{x}'$ is aperiodic. Thus, by Proposition 51, in order for $\mathbf{x}'$ to be aperiodic (and subsequently also $\mathbf{x}$), we need that $\mu \neq ((\mathtt{a}\mathtt{b}^p)^q\mathtt{a}, \mathtt{b})$ for every $p, q \geq 1$, and as a consequence we have that $|\Sigma'| > 2$. For all the other cases, the thesis follows since $p_{\mathbf{x}}(n) = \Theta(p_{\mathbf{x}'}(n)) = \Theta(n)$. □

Proposition 55. *Let $\mu = (\alpha, \beta)$ be a growing non-primitive morphism prolongable on $\mathtt{a}$ and let $\mathbf{x} = \mu^\infty(\mathtt{a})$ be its aperiodic fixed point. Then, $\mu \equiv (\mathtt{a}v, \mathtt{b}^\ell)$, for some $\ell \geq 2$ and $v \in \Sigma^+$ such that $|v|_\mathtt{a}, |v|_\mathtt{b} \geq 1$. Moreover, let $n_\mathtt{a} = |\mathtt{a}v|_\mathtt{a}$. Then, it holds that:*

1. *$n_\mathtt{a} < \ell$ if and only if $p_{\mathbf{x}}(n) = \Theta(n)$;*
2. *$n_\mathtt{a} = \ell$ if and only if $p_{\mathbf{x}}(n) = \Theta(n \log\log n)$;*

3. $n_{\mathtt{a}} > \ell$ *if and only if* $p_{\mathbf{x}}(n) = \Theta(n \log n)$.

Proof. The first claim can be deduced from Lemma 53 and the hypothesis of μ being growing (that is, $B = \emptyset$).

For the rest of the proof, we use the characterization of morphisms defined in Theorem 44, and show that under the conditions above mentioned, we obtain a quasi-uniform, a polynomially divergent, and an exponentially divergent morphism respectively. Clearly, $\mu_{\mathtt{b}}(i) = \ell^i = \Theta(\ell^i)$. What is left is to compute $\mu_{\mathtt{a}}$ and compare it with $\mu_{\mathtt{b}}$.

Let us define analogously $n_{\mathtt{b}} = |\mathtt{a}v|_{\mathtt{b}}$. For any word $w \in \{\mathtt{a}, \mathtt{b}\}^*$, by hyphotesis on μ we have that $|\mu(w)|_{\mathtt{a}} = n_{\mathtt{a}}|w|_{\mathtt{a}}$ and $|\mu(w)|_{\mathtt{b}} = n_{\mathtt{b}}|w|_{\mathtt{a}} + \ell|w_{\mathtt{b}}|$. By induction, we can prove that $|\mu^i(\mathtt{a})|_{\mathtt{a}} = n_a^i$, $|\mu^i(\mathtt{a})|_{\mathtt{b}} = \frac{n_{\mathtt{b}}(n_{\mathtt{a}}^i - \ell^i)}{n_{\mathtt{a}} - \ell}$, if $n_{\mathtt{a}} \neq \ell$, and $|\mu^i(\mathtt{a})|_{\mathtt{b}} = n_{\mathtt{b}} i \ell^{i-1}$, if $n_{\mathtt{a}} = \ell$. We can then distinguish the three cases.

1. If $n_{\mathtt{a}} < \ell$, then we can rewrite the growth function on $\mathtt{a}$ as

$$\begin{aligned}
\mu_{\mathtt{a}}(i) &= |\mu^i(\mathtt{a})|_{\mathtt{a}} + |\mu^i(\mathtt{a})|_{\mathtt{b}} \\
&= n_{\mathtt{a}}^i + \frac{n_{\mathtt{b}}(n_{\mathtt{a}}^i - \ell^i)}{n_{\mathtt{a}} - \ell} \\
&= \frac{n_{\mathtt{a}}^i(\ell - n_{\mathtt{a}}) + n_{\mathtt{b}}(\ell^i - n_{\mathtt{a}}^i)}{\ell - n_{\mathtt{a}}} \\
&= \frac{n_{\mathtt{b}}\ell^i + n_{\mathtt{a}}^i(\ell + n_{\mathtt{b}} - n_{\mathtt{a}})}{\ell - n_{\mathtt{a}}}.
\end{aligned}$$

   From this description, we can see that

$$\frac{n_{\mathtt{b}}\ell^i}{\ell - n_{\mathtt{a}}} \leq \frac{n_{\mathtt{b}}\ell^i + n_{\mathtt{a}}^i(\ell + n_{\mathtt{b}} - n_{\mathtt{a}})}{\ell - n_{\mathtt{a}}} \leq \frac{n_{\mathtt{b}}\ell^i + \ell^i(\ell + n_{\mathtt{b}} - n_{\mathtt{a}})}{\ell - n_{\mathtt{a}}} = \frac{\ell^i(\ell + 2n_{\mathtt{b}} - n_{\mathtt{a}})}{\ell - n_{\mathtt{a}}}.$$

   Thus, $\mu_a(i) = \Theta(\ell^i)$, and by Theorem 44 the factor complexity is $p_{\mathbf{x}}(n) = \Theta(n)$.

2. If $n_{\mathtt{a}} = \ell$, then

$$\begin{aligned}
\mu_{\mathtt{a}} &= |\mu^i(\mathtt{a})|_{\mathtt{a}} + |\mu^i(\mathtt{a})|_{\mathtt{b}} \\
&= \ell^i + n_{\mathtt{b}} i \ell^{i-1} \\
&= \ell^{i-1}(n_{\mathtt{b}} i + \ell) \\
&= \frac{\ell^i(n_{\mathtt{b}} i + \ell)}{\ell}.
\end{aligned}$$

Since ℓ and $n_{\mathtt{b}}$ are constant values, easily follows that $\mu_{\mathtt{a}}(i) = \Theta(i\ell^i)$, and therefore μ is polynomially divergent and $p_{\mathtt{x}}(n) = \Theta(n \log\log n)$.

3. If $n_{\mathtt{a}} > \ell$, we can rewrite the description from case *1.* to obtain a different bound.

$$\begin{aligned}
\mu_{\mathtt{a}}(i) &= |\mu^i(\mathtt{a})|_{\mathtt{a}} + |\mu^i(\mathtt{a})|_{\mathtt{b}} \\
&= n_{\mathtt{a}}^i + \frac{n_{\mathtt{b}}(n_{\mathtt{a}}^i - \ell^i)}{n_{\mathtt{a}} - \ell} \\
&= \frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} - \ell) + n_{\mathtt{b}}(n_{\mathtt{a}}^i - \ell^i)}{n_{\mathtt{a}} - \ell} \\
&= \frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} + n_{\mathtt{b}} - \ell) - n_{\mathtt{b}}\ell^i}{n_{\mathtt{a}} - \ell}.
\end{aligned}$$

From this description we can deduce the following bounds:

$$\begin{aligned}
n_{\mathtt{a}}^i &= \\
\frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} - \ell)}{n_{\mathtt{a}} - \ell} &= \frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} + n_{\mathtt{b}} - \ell) - n_{\mathtt{b}}n_{\mathtt{a}}^i}{n_{\mathtt{a}} - \ell} \qquad \leq \frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} + n_{\mathtt{b}} - \ell) - n_{\mathtt{b}}\ell^i}{n_{\mathtt{a}} - \ell} \\
&\leq \frac{n_{\mathtt{a}}^i(n_{\mathtt{a}} + n_{\mathtt{b}} - \ell)}{n_{\mathtt{a}} - \ell}
\end{aligned}$$

Thus, $\mu_{\mathtt{a}}(i) = \Theta(n_{\mathtt{a}}^i)$ with $n_{\mathtt{a}} \neq \ell$. Then, μ is exponentially divergent and $p_{\mathtt{x}}(n) = \Theta(n \log n)$.

Since we have exhausted all the possible cases, the implication goes in the other direction as well, and the thesis follows. □

In the next section, we show how to take advantage of the structure of the morphisms to get a theoretical bound on the measure r in the case of binary morphisms.

4.3 Logarithmic BWT-Runs of Binary Purely Morphic Words

Here we use the results from the previous section to compute the number of BWT-runs for binary purely morphic words.

We have already shown in Section 4.1 that the measure $r = \mathcal{O}(1)$ when the prefix of an eventually periodic word is considered. On aperiodic words, a first bound that can be deduced from the previous sections is the following.

Proposition 56. *Let* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *be the fixed point of the morphism* $\mu : \Sigma^* \mapsto \Sigma^*$. *If* μ *is quasi-uniform, then* $r(\mu^i(\mathtt{a})) = \mathcal{O}(i)$, *where* $i \geq 0$ *is the iteration of the morphism. Moreover, if* $n = \mu_{\mathtt{a}}(i)$, *then* $r(\mu^i(\mathtt{a})) = \mathcal{O}(\log n)$.

Proof. The proof follows from Proposition 49 and from the definition of quasi-uniform morphisms. □

From the proposition above, we can extend this result to all primitive morphisms.

Corollary 57. *Let* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *be the fixed point of the morphism* $\mu : \Sigma^* \mapsto \Sigma^*$. *If* μ *is primitive, then* $r(\mu^i(\mathtt{a})) = \mathcal{O}(i) = \mathcal{O}(\log n)$, *where* $i \geq 0$ *is the iteration of the morphism.*

Proof. The proof follows since any primitive morphism is quasi-uniform and by Proposition 56. □

Note that these results are independent from the size of the alphabet Σ considered. For the rest of the section, we will try to cover the remaining binary cases, namely non-primitive morphisms with aperiodic fixed point for which the factor complexity is either $\Theta(n \log\log n)$, $\Theta(n \log n)$, or $\Theta(n^2)$.

Given a morphism $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \{\mathtt{a}, \mathtt{b}\}^*$, let $R_i = \{q \mid \mathtt{ab}^q\mathtt{a} \in \mathcal{C}(\mu^i(\mathtt{a}))\}$. If $n_{\mathtt{a}} = |\mu(\mathtt{a})|_{\mathtt{a}}$, one can verify that $|R_1| \leq n_{\mathtt{a}}$, since we can factorize $\mu(\mathtt{a}) = \mathtt{ab}^{p_1}\mathtt{ab}^{p_2}\mathtt{a}\cdots\mathtt{ab}^{p_{n_{\mathtt{a}}}}$, with $p_i \geq 0$ for all $i \in [1, n_{\mathtt{a}}]$. As it will be clear later in this section, we can use the growth of $|R_i|$ to obtain tighter bounds with respect to those obtained in Proposition 50 through the comparison with δ and z measures.

In the next lemmas, we describe the elements that R_i contains in the case of non-primitive morphisms, according to whether μ is growing or not.

Lemma 58. *Let* $\mu = (\alpha, \beta)$ *be a non-growing morphism prolongable on* $\mathtt{a}$ *and let* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *be its fixed point.*

1. *If* $\alpha = \mathtt{a}u\mathtt{ba}^k$, $\beta = \mathtt{b}$, *for some* $u \in \{\mathtt{a}, \mathtt{b}\}^*$ *and* $k \geq 1$, *then* $R_i = R_1$ *for every* $i \geq 1$;
2. *If* $\alpha = \mathtt{a}u\mathtt{ab}^k$, $\beta = \mathtt{b}$, *for some* $u \in \{\mathtt{a}, \mathtt{b}\}^*$ *and* $k \geq 1$, *then* $R_i = \bigcup_{h=1}^{i} \bigcup_{j=1}^{n_{\mathtt{a}}-1} \{t_j + (h-1)k\} \cup \{ik\}$ *for every* $i \geq 1$.

Proof. Suppose $\mu(\mathtt{a}) = \mathtt{a}u\mathtt{ab}^k$ with $k \geq 1$. We can see that for $R_1 = \bigcup_{j=1}^{n_{\mathtt{a}}} \{t_j\}$ the thesis holds. By induction, suppose it holds for R_i. We can see that all $\mathtt{ab}^q\mathtt{a} \in \mathcal{C}(\mu^{i+1}(\mathtt{a}))$, for

some $q \in R_{i+1}$, either are factors of $\mu(a) = \mathtt{a}u\mathtt{ab}^k$ (that is $q = t_j$ for some $1 \leq j < n_\mathtt{a}$) or there exists $q' = q - k$ such that $q' \in R_i$ and $\mu(\mathtt{ab}^{q'}\mathtt{a}) = \mathtt{a}u \cdot \mathtt{ab}^q\mathtt{a} \cdot u\mathtt{ab}^k$. Hence, $R_{i+1} = \bigcup_{j=1}^{n_\mathtt{a}-1}\{t_j\} \cup \bigcup_{q' \in R_i}\{q' + k\} = \bigcup_{h=1}^{i+1}\bigcup_{j=1}^{n_\mathtt{a}-1}\{t_j + (h-1)k\} \cup \{(i+1)k\}$. For case *1*, note that the proof holds also in this case but with $t_{n_a} = k = 0$, hence we obtain $R_{i+1} = \bigcup_{j=1}^{n_\mathtt{a}}\{t_j\} = R_1$ and the thesis follows. □

With an analogous proof, the following result can be proven for all growing non-primitive morphisms.

Lemma 59. *Let μ be a growing non-primitive binary morphism. If $\mathbf{x} = \mu^\infty(\mathtt{a})$ is eventually periodic, then $\mu \equiv (\mathtt{ab}^k, \mathtt{b}^\ell)$, for all $k \geq 1$ and $\ell > 1$, and $R_i = k\sum_{j=0}^{i-1}\ell^j$, for all $i \geq 1$. Otherwise, it holds that $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$, $k \geq 0$, $\ell > 1$, $u \in \{\mathtt{a},\mathtt{b}\}^*$ and $R_i = \bigcup_{h=1}^{i}\bigcup_{j=1}^{n_\mathtt{a}-1}\{\frac{\ell^{h-1}((\ell-1)t_j+k)-k}{\ell-1}\} \cup \{k\frac{\ell^i-1}{\ell-1}\}$, for every $i \geq 1$.*

For some of the cases above mentioned, we can show the growth of the set R_i is linear with i.

Proposition 60. *Let $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ a binary morphism with $k \geq 0$, $\ell \geq 1$ and $k + \ell > 1$ and aperiodic fixed point. Then $|R_i| = \Theta(i)$, where $i \geq 0$ is the iteration of the morphism.*

Proof. By definition, we have that $n_\mathtt{a} \geq 2$. Let us consider the case $\ell = 1$. By Lemma 58 (case 2), it follows that there exists $t_1 \in R_1$ such that $\mathtt{ab}^{t_1+hk}\mathtt{a} \in \mathcal{C}(\mu^i(\mathtt{a}))$ for every $0 \leq h \leq i-1$. This means that $|R_i| \geq i$. Moreover, for every positive integer d such that $\mathtt{ab}^d\mathtt{a} \in \mathcal{C}(\mu^i(\mathtt{a}))$, it follows that $d = ik$ or $d = t_j + hk$ for each $t_j \in R_1$ (with $1 \leq j \leq n_\mathtt{a} - 1$) and for every $0 \leq h \leq i-1$. This provides an upper bound for $|R_i|$, i.e., $|R_i| \leq i(n_\mathtt{a} - 1) + 1$. The case $\ell > 1$ can be proved by using a similar argument and by applying Lemma 59. In fact, for every positive integer d such that $\mathtt{ab}^d\mathtt{a} \in \mathcal{C}(\mu^i(\mathtt{a}))$, it follows that $d = k\frac{\ell^i-1}{\ell-1}$ or $d = \frac{\ell^h((\ell-1)t_j+k)-k}{\ell-1}$ for each $t_j \in R_1$ (with $1 \leq j \leq n_\mathtt{a} - 1$) and for every $0 \leq h \leq i-1$. Therefore, $|R_i| = \Theta(i)$. □

Here we start to see how to relate the notion of the set R_i with the measure r, and more in particular with bispecial factors.

Lemma 61. *Let $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ be a binary morphism, for some $k \geq 0, \ell \geq 1$. If v is a circular bispecial factor of $\mu^i(\mathtt{a})$, for some iteration $i \geq 1$, then $w = \mathtt{b}^k\mu(v)$ is a bispecial circular factor of $\mu^{i+1}(\mathtt{a})$.*

Proof. If v is a circular bispecial factor in $\mu^{n-1}(\mathtt{a})$, then we can suppose w.l.g. $\mathtt{a}v\mathtt{a}$ and $\mathtt{b}v\mathtt{b}$ are circular factors of $\mu^{n-1}(\mathtt{a})$. This means that $\mu(\mathtt{a}v\mathtt{a}) = \mathtt{a}u\mathtt{ab}^k\mu(v)\mathtt{a}u\mathtt{ab}^k$ and $\mu(\mathtt{b}v\mathtt{b}) = \mathtt{b}^\ell\mu(v)\mathtt{b}^\ell$ are circular factors of $\mu^n(\mathtt{a})$. Note that every β is circularly preceded by $\mathtt{b}^k$, i.e. $\mathtt{ab}^k\mu(v)\mathtt{a}$ and $\mathtt{bb}^k\mu(v)\mathtt{b}$ are circular factors of $\mu^n(\mathtt{a})$. □

The following lemmas will be used to find the circular bispecial factors of finite purely morphic words.

Lemma 62. *Let $\mu \equiv (\alpha, \beta) = (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ for some $k \geq 0$, $\ell \geq 1$ such that $k + \ell > 1$, and let m be the length of the longest runs of $\mathtt{b}$'s that occurs in $\mathtt{a}u\mathtt{ab}^k$. Then, for every $i \geq 2$, every circular factor v of $\mu^i(\mathtt{a})$ of length $|v| > |\mu(\alpha)| + (2-\ell)m - k + \ell^2 M$ contains $\mathtt{b}^{m+1}$ as factor, where $M = \max\{\lfloor \frac{m-(\ell+1)k}{\ell^2} \rfloor, 0\}$.*

Proof. Let $n_\mathtt{a} = |\mu(\mathtt{a})|_\mathtt{a}$. Then we can uniquely factorize $\mu(\mathtt{a}) = \mathtt{ab}^{p_1}\mathtt{ab}^{p_2}\mathtt{a}\cdots\mathtt{ab}^{p_{n_\mathtt{a}}}$, for some $p_1, p_2, \ldots, p_{n_\mathtt{a}-1} \geq 0$ and $p_{n_\mathtt{a}} = k \geq 0$. Let j be an index such that $p_j = m$. Since $\mu^i(\mathtt{a}) \in \{\mu(\alpha), \mu(\beta)\}^*$ for every $i \geq 2$, where $\mu(\alpha) = \prod_{i=1}^{n_\mathtt{a}} \mathtt{a}u\mathtt{ab}^{p_{n_\mathtt{a}}+\ell p_i}$ and $\mu(\beta) = \mathtt{b}^{\ell^2}$, we have that $\mathtt{b}^{\ell p_j + p_{n_\mathtt{a}}}$ occurs in $\mu(\alpha)$, for every $1 \leq j \leq n_\mathtt{a}$. Note that, with the exception of the case $k = 0$ and $\ell = 1$ (that we are not considering since otherwise $k + \ell = 1$), $\ell p_j + p_{n_\mathtt{a}} = \ell m + k > m$, that is $\mu(\alpha)$ has at least an occurrence of the factor $\mathtt{b}^{m+1}$. Then the longest circular factor of a string $w \in \{\mu(\alpha), \mu(\beta)\}^*$ that does not contain $\mathtt{b}^{m+1}$ must be a factor of $\mu(\alpha)\mu(\beta)^M\mu(\alpha)$, where $M = \max\{\lfloor \frac{m-(\ell+1)p_{n_\mathtt{a}}}{\ell^2} \rfloor, 0\}$ and it is $v^* = \mathtt{b}^{p_j}\prod_{i=j+1}^{n_\mathtt{a}} \mathtt{a}u\mathtt{ab}^{p_{n_\mathtt{a}}+\ell p_i} \cdot (\mathtt{b}^{\ell^2})^M \cdot \prod_{i=1}^{j-1} \mathtt{a}u\mathtt{ab}^{p_{n_\mathtt{a}}+\ell p_i} \cdot \mathtt{a}u\mathtt{ab}^{p_j}$, since if we extend either on the left or on the right we find another $\mathtt{b}$ and we have $\mathtt{b}^{p_j+1} = \mathtt{b}^{m+1}$ as factor. We can see that $|v^*| = |\mu(\alpha)| + (2-\ell)p_j - k + \ell^2 M$. □

By using the techniques developed in the proof of Lemma 62, we can derive the following.

Lemma 63. *Let $\mu \equiv (\alpha, \beta) = (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ for some $k \geq 0$, $\ell \geq 1$ such that $k + \ell > 1$, and let m be the length of the longest runs of $\mathtt{b}$'s that occurs in $\mathtt{a}u\mathtt{ab}^k$. Then, the circular factors of a string in $\{\mu(\alpha), \mu(\beta)\}^*$ having maximal length and not containing $\mathtt{b}^{m+1}$, must be factor of $\mu(\alpha)\mu(\beta)^h\mu(\alpha)$, for some $0 \leq h \leq \max\{m - 2k, 0\}$.*

Now, we describe the structure of circular bispecial factors of $\mu^i(\mathtt{a})$, for all non-primitive binary morphisms.

Lemma 64. *Let $\mu \equiv (\alpha, \beta) = (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ for some $k \geq 0$, $\ell \geq 1$, let m be the length of the longest equal-letter run of $\mathtt{b}$'s that occurs in $\mathtt{a}u\mathtt{ab}^k$, and let $M = \max\{\lfloor \frac{m-(\ell+1)k}{\ell^2} \rfloor, 0\}$. Then, every circular bispecial factor w of $\mu^{i+1}(\mathtt{a})$, $i \geq 1$, either appears as a circular factor of a word in $\bigcup_{j=0}^{M}\{\mu(\alpha)\mu(\beta)^j\mu(\alpha)\}$ or $w = \mathtt{b}^k\mu(v)$, for some circular bispecial factor v in $\mu^i(\mathtt{a})$ (or $w = \mathtt{b}^h$, for some $h \geq 1$, when $k \geq 0$ and $\ell > 1$).*

Proof. Consider the case $\ell = 1$. By Lemma 63, if w is not a circular factor of a word in $\bigcup_{j=0}^{M}\{\mu(\alpha)\mu(\beta)^j\mu(\alpha)\}$, then it holds that $w = y\mathtt{b}^{m+1}z$, for some $y, z \in \Sigma^*$. Note that the last $m-k+1$ letters of $\mathtt{b}^{m+1}$ can be obtained only as $\mu(\mathtt{b})$. Since $\{\mu(\mathtt{a}), \mu(\mathtt{b})\}$ is a prefix code, z can be uniquely factorized as a sequence of $\mu(\mathtt{a})$'s and $\mu(\mathtt{b})$'s, i.e. $w = y\mathtt{b}^k\mu(\mathtt{b})^{m-k+1}\mu(v)z'$, for some $v \in \mathcal{C}(\mu^{n-1}(\mathtt{a}))$ and some proper prefix z' of $\mu(\mathtt{a})$ or $\mu(\mathtt{b})$. From the fact that w is right special, we can prove that $z' = \varepsilon$. In fact, if $z' \neq \varepsilon$, then z' is a proper prefix of α and w must be followed by $\alpha_{|z'|+1}$, i.e. w would not be right special. Moreover, $y\mathtt{b}^k\mu(\mathtt{b}^{m-k+1})$ can be uniquely factorized as a sequence of $\mu(\mathtt{a})$'s and $\mu(\mathtt{b})$'s, unless for a prefix $\mathtt{b}^j$, with $j \leq k$, since that prefix could be a proper suffix of α, a run of β's or a combination of both. However, if $j < k$, then w must be preceded by $\mathtt{b}^{k-j}$, that is w is not left special and therefore $j = k$.

If $\ell > 1$, the proof holds but with the exception of bispecial factors $w = \mathtt{b}^h$, with $h \geq m+1$, that do not occur in $\bigcup_{h=0}^{M}\{\mu(\alpha)\mu(\beta)^h\mu(\alpha)\}$. In fact, only in this case we can not uniquely factorize any of the factor of w (observe that $b\mu(\beta) = \mu(\beta)b$). Let m' be the greatest value such that $\mathtt{ab}^{m'}\mathtt{a}$ is a factor of $\mu^n(\mathtt{a})$. Since $\mathtt{ab}^{m'}\mathtt{a} = \mathtt{ab}^h\mathtt{b}^{m'-h}\mathtt{a} = \mathtt{ab}^{m'-h}\mathtt{b}^h\mathtt{a}$ for every $1 \leq h < m'$, it holds that $\mathtt{ab}^h\mathtt{b}$ and $\mathtt{bb}^h a$ occur in $\mu^n(\mathtt{a})$. It follows that $\mathtt{b}^h$ is bispecial for every $0 \leq h < m'$ and the thesis follows. □

Here we obtain the first tight lower bound for the classes of binary purely morphic words generated by a non-primitive morphism.

Proposition 65. *Let $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ be a non-primitive morphism with $k \geq 0$, $\ell \geq 1$ with aperiodic fixed point. Then $r(\mu^i(\mathtt{a})) = \mathcal{O}(i)$, where $i \geq 0$ is the iteration of the morphism.*

Proof. Let m be the longest run of $\mathtt{b}$'s that occurs in $\mathtt{a}u\mathtt{ab}^k$ and let $M = \max\{\lfloor \frac{m-(\ell+1)k}{\ell^2} \rfloor, 0\}$. Let us consider the following three sets:

- $BS_0(\mu^i(\mathtt{a})) = \{v \in \{\mathtt{a}, \mathtt{b}\}^* \mid v$ is a bispecial circular factor of a word in $\bigcup_{i=0}^{M}\{\mu(\alpha)\mu(\beta)^i\mu(\alpha)\}\}$,

- $BS_{\mathtt{b}}(\mu^i(\mathtt{a})) = \{\mathtt{b}^h \mid \mathtt{b}^h \in \mathcal{C}(\mu^i(\mathtt{a}))\}$, and

- $BS_{\mu}(\mu^i(\mathtt{a})) = \{\mathtt{b}^k\mu(v) \mid v \in \mathcal{C}(\mu^{i-1}(\mathtt{a}))$ and $|v|_{\mathtt{a}} \geq 1\}$.

From Lemma 64, we can see that $BS(\mu^i(\mathtt{a})) = BS_0(\mu^i(\mathtt{a})) \cup BS_{\mathtt{b}}(\mu^i(\mathtt{a})) \cup BS_{\mu}(\mu^i(\mathtt{a}))$ (note that the intersection can be non-empty). It is easy to see that $|BS_0| = \mathcal{O}(1)$. Moreover, since $\mu^i(\mathtt{a}) = \mathtt{a}u\mu(u)\mu^2(u)\cdots\mu^{i-1}(u)$, we can see that every element $w \in BS_{\mu}(\mu^i(\mathtt{a}))$ that do not belong to $BS_{\mu}(\mu^{i-1}(\mathtt{a}))$ has to cross $\mu^{i-1}(u)$ and $w = \mathtt{b}^k\mu(v)$ for some v that crosses $\mu^{i-2}(u)$. Iterating the procedure, we can see that $|BS_{\mu}(\mu^i(\mathtt{a}))| = \mathcal{O}(i)$. If $\ell = 1$, from Lemma 64, we can see that also $BS_{\mathtt{b}}(\mu^i(\mathtt{a})) = \mathcal{O}(i)$. It follows that $r(\mu^i(\mathtt{a})) \leq |BS(\mu^i(\mathtt{a})| \leq |BS_0(\mu^i(\mathtt{a})| + |BS_{\mu}(\mu^i(\mathtt{a})| + |BS_{\mathtt{b}}(\mu^i(\mathtt{a})| = \mathcal{O}(i)$.

On the other hand, if $\ell > 1$, clearly $|BS_b(\mu^i(\mathtt{a}))| = \Omega(\ell^i)$, and therefore $|BS(\mu^i(\mathtt{a}))| = \Omega(\ell^i)$. However, recall that each change of letters in $\mathtt{bwt}(w)$ uniquely corresponds to a unique bispecial factor, that is the longest common prefix between the two conjugates in correspondence of the change of letter (see Lemma 19 and recall that $e_l(u), e_r(u) \leq 1$ for binary alphabets). Let m' be the length of the longest run of $\mathtt{b}$'s in $\mu^i(\mathtt{a})$. Note that $m' \in R_i$. We define $\overline{R}_i = \{0, 1, \ldots, m'-1\} \setminus R_i$, i.e. $\overline{R}_i$ is the set of indices $0 \leq \overline{q} < m'$ such that $\mathtt{ab}^{\overline{q}}\mathtt{a}$ is not a factor of $\mu^i(\mathtt{a})$. By Lemma 59 we know that $|R_i|$ grows as $\mathcal{O}(i)$. This implies that, even though $|\overline{R}_i| \in \Omega(\ell^i)$, we can split $\overline{R}_i$ in $K \leq |R_i|$ subsets of maximal sizes that contain consecutive lengths of runs of $\mathtt{b}$'s, i.e. $P_j = \{h_j, h_j+1, \ldots, h_j+d_j \in \overline{R}_i \mid h_j-1, h_j+d_j+1 \notin \overline{R}_i\}$, for every $1 \leq j \leq K$. We can see that $\mathtt{b}^h$ is the longest common prefix of two consecutive conjugate words C_t and C_{t+1} of w, for some $0 \leq t < n$ if and only if $C_t = \mathtt{b}^h\mathtt{a}z$ and $C_{t+1} = b^{h+1}az'$, for some $z, z' \in \Sigma^*$ (if $b < a$ then just switch the order of C_t with C_{t+1}). Note that we can assign to each set P_j the maximal interval $[s_j \ldots s_j + \ell_j]$ (for some $0 \leq s_j, \ell_j \leq |\mu^i(\mathtt{a})|$) of sorted conjugates $\{C_{s_j}, C_{s_j+1}, \ldots, C_{s_j+\ell_j}\}$ with prefix with $\mathtt{b}^h\mathtt{a}$, for every $h \in P_j$. Let $y = \mathtt{bwt}(\mu^i(\mathtt{a}))$. Since by definition $\mathtt{ab}^h\mathtt{a}$ is not a factor of $\mu^i(\mathtt{a})$ for every $h \in \overline{R}_i$, then $y_{s_j}y_{s_j+1}\cdots y_{s_j+\ell_j} = b^{\ell_j+1}$, and therefore a change of letter in y can not correspond to a bispecial factor $\mathtt{b}^h$ with $h \in P_j$, with the only possible exception for $\mathtt{b}^{h_j+d_j}$ if $y_{s_j+\ell_j+1} = \mathtt{a}$, since $\mathtt{ab}^{h_j+d_j+1}\mathtt{a}$ is a factor of $\mu^i(\mathtt{a})$ and $lcp(C_{s_j+\ell_j}, C_{s_j+\ell_j+1}) = \mathtt{b}^{h_j+d_j}$. Finally, since there are $K \leq |R_i|$ of these intervals, we have that $r(\mu^i(\mathtt{a})) \leq |BS_0(\mu^i(\mathtt{a}))| + |BS_{\mu}(\mu^i(\mathtt{a}))| + 2|R_i| = \mathcal{O}(i)$. □

As shown in the next proposition, the set R_i can be used to obtain a lower bound as well.

Proposition 66. *Let $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b}^\ell)$ be a non-primitive morphism with $k \geq 0$, $\ell \geq 1$ and $\mathbf{x} = \mu^\infty(\mathtt{a})$ is aperiodic. Then $r(\mu^i(\mathtt{a})) = \Omega(|R_i|)$, where $i \geq 0$ is the iteration of the morphism. Moreover, when μ is not growing with $p_{\mathbf{x}}(n) = \Theta(n^2)$ or μ is growing, then $r(\mu^i(\mathtt{a})) = \Omega(i)$.*

Proof. Let $C_1 < C_2 < \ldots < C_n$ be the lexicographically sorted rotations of $\mu^i(\mathtt{a})$ and let m' be the length of the longest run of $\mathtt{b}$'s in $\mu^i(\mathtt{a})$. We denote by (s_j, ℓ_j) the first index of rotations with prefix $\mathtt{b}^j\mathtt{a}$ and the number of these rotations respectively. It follows that $\mathsf{rot}(\mathtt{b}^j\mathtt{a}) = \{C_{s_j}, C_{s_j+1}, \ldots, C_{s_j+\ell_j-1}\}$, for every $j \in [0, m']$.

Let $y = \mathsf{bwt}(\mu^i(a))$. We can see that for every $j \in [0, m']$, if $\mathsf{rot}(\mathtt{b}^j\mathtt{a})$ exists, then $y[s_{j-1}, s_{j-1} + \ell_{j-1} - 1]$ contains at least ℓ_j occurrences of $\mathtt{b}$'s. In fact, for every $h \geq 0$, if there exist ℓ_j rotations such that $C_j = \mathtt{b}^{h+1}\mathtt{a}u$ for some $u \in \Sigma^*$, then there exist as much rotations $C_{j'} = \mathtt{b}^h\mathtt{a}u\mathtt{b}$. It follows that, for every $j \in [0, m'-1]$, the factor $y[s_j, s_j + \ell_j - 1]$ contains at least one $\mathtt{b}$.

Moreover, by definition, we know that if $j \in R_i$, then there is at least an occurrence $\mathtt{a}$ in $y[s_j, s_j + \ell_j - 1]$. Hence, for every $0 \leq j' \leq \lceil \frac{|R_i|-1}{2} \rceil$ such that $q_{2j'+1} \in R_i$, we have that in $y[s_{2j'+1}, s_{2(j'+1)+1} + \ell_{2(j'+1)+1} - 1]$ it must occur $\mathtt{aba}$ as subsequence, since there is at least one run of $\mathtt{a}$'s in $y[s_{2j'+1}, s_{2j'+1} + \ell_{2j'+1}]$, at least one run of $\mathtt{b}$'s in $y[s_{2(j'+1)}, s_{2(j'+1)} + \ell_{2(j'+1)} - 1]$ and at least one run of $\mathtt{a}$'s in $y[s_{2(j'+1)+1}, s_{2(j'+1)+1} + \ell_{2(j'+1)+1} - 1]$. Hence, $r(\mu^i(\mathtt{a})) = |\mathsf{rle}(y)| \geq |\mathsf{rle}(y[s_1, s_{m'} + \ell_{m'} - 1])| \geq |\mathsf{rle}((\mathtt{ab})^{\lceil \frac{|R_i|-1}{2} \rceil}\mathtt{a})| = \Omega(|R_i|)$.

By Proposition 54 we know that $p_{\mathbf{x}}(n) = \Theta(n^2)$, if and only if $\mu \equiv (\mathtt{a}u\mathtt{ab}^k, \mathtt{b})$ for some $u \in \Sigma^*$ and $k \geq 1$. If μ is growing and non-primitive then, by Proposition 55, there exist $\ell \geq 2$ and $u, v \in \{\mathtt{a}, \mathtt{b}\}^*$, with $v = u\mathtt{ab}^k$ for some $k \geq 0$ and $|v|_{\mathtt{b}} \geq 1$, such that $\mu \equiv (\mathtt{a}v, \mathtt{b}^\ell)$. Finally, by using Proposition 60, the thesis follows. □

In the next theorem, we summarize the results on the Burrows-Wheeler Transform of finite purely morphic words, grouped by classes of factor complexity. Once again, recall that this theorem holds independently from the order defined on the letters of the alphabet.

Theorem 67. *Let* $\mathbf{x} = \mu^\infty(\mathtt{a})$ *be an aperiodic fixed point for some morphism* $\mu : \{\mathtt{a},\mathtt{b}\}^* \mapsto \{\mathtt{a},\mathtt{b}\}^*$ *prolongable on* $\mathtt{a}$*, and let* $n = \mu_{\mathtt{a}}(i)$*, where* $i \geq 0$ *is the iteration of the morphism. The following bounds hold:*

1. *if* $p_{\mathbf{x}}(n) = \Theta(n)$*, then* $r(\mu^i(\mathtt{a})) = \mathcal{O}(i) = \mathcal{O}(\log n)$*;*
2. *if* $p_{\mathbf{x}}(n) = \Theta(n \log\log n)$*, then* $r(\mu^i(\mathtt{a})) = \Theta(i) = \Theta(\log n)$*;*
3. *if* $p_{\mathbf{x}}(n) = \Theta(n \log n)$*, then* $r(\mu^i(\mathtt{a})) = \Theta(i) = \Theta(\log n)$*;*
4. *if* $p_{\mathbf{x}}(n) = \Theta(n^2)$*, then* $r(\mu^i(\mathtt{a})) = \Theta(i) = \Theta(\log n)$*.*

Proof. All primitive morphisms are quasi-uniform, i.e. $\mu_{\mathtt{a}}(i) = \Theta(\rho^i)$ for some $\rho > 1$. By Lemma 53, all morphisms non-primitive with aperiodic fixed point verifies that $|\mu(\mathtt{a})|_{\mathtt{a}} \geq 2$, and therefore $\mu_{\mathbf{x}} = \Omega(2^i)$, and for each morphic sequences there exists a constant $\rho > 1$ such that $\mu_{\mathbf{x}}(i) = \mathcal{O}(\rho^i)$. For both cases, one can see that $i = \Theta(\log n)$.

The case *1.* follows from Proposition 56 and Theorem 44.

The cases *2.-4.* can be derived from the structures of the morphisms of Propositions 55 and 54, and by Propositions 65 and 66 the thesis follows. □

Note that only for the case *1.* of Theorem 67 we do not have $\Omega(i)$ as lower bound. In fact, for any Sturmian morphism φ' holds that, for all $i \geq 1$, $\varphi'^i(\mathtt{a})$ is a standard Sturmian word, and for any standard Sturmian word s holds that $r(s) = 2 = \mathcal{O}(1)$ (Theorem 14). We conclude this section with the following question concerning the gap between $\mathcal{O}(1)$ and $\mathcal{O}(i)$.

Question 68. *Let* $\mu : \{\mathtt{a},\mathtt{b}\}^* \mapsto \{\mathtt{a},\mathtt{b}\}^*$ *be a binary morphism prolongable on* $\mathtt{a}$*. If* $r(\mu^i(\mathtt{a})) \neq \mathcal{O}(1)$*, does it follow that* $r(\mu^i(\mathtt{a})) = \Omega(i)$ *(and vice versa)?*

4.4 Run-Length Compression of Purely Morphic Words

Despite the theoretical flaws on the measure r observed in the previous chapter, the BWT is still widely used for its efficiency on compressing highly repetitive text, while giving the possibility index the text in a space proportional to r. Nonetheless, typical problems of string pattern matching have been dealt for run-length compressed words [89, 125]. Mantaci et al. [93] have considered the problem of finding

words that can not be *clustered* from the BWT, and introduced the notion of *BWT-clustering ratio*, defined for any word w as $\rho(w) = \frac{r(w)}{|\mathsf{rle}(w)|}$. They proved that $\rho(w) \leq 2$ for any word w, and found an infinite family of words for which their bound is tight. Later, Brlek et al. proved that for Thue-Morse morphism the measure $\rho(\tau^i(\mathtt{a}))$ tends to 0 as i grows [20].

In this section, we extend this result to other families of purely morphic words, with a particular focus on the binary case for which we have proved tight bounds on the measure r. In particular, we are interested in classifying morphisms based on the efficiency of the run-length encoding on BWT with respect to the word.

Definition 69. *A morphism μ prolongable on $a \in \Sigma$ is* BWT-highly compressible *if* $\limsup_{i\to\infty} \rho(\mu^i(a)) = 0$, *where ρ is the BWT-clustering ratio.*

To facilitate the reading, given a morphism μ prolongable on a letter $a \in \Sigma$, we denote by $\mathsf{runs}_{\mu,a}(i) = |\mathsf{rle}(\mu^i(a))|$. Clearly, the function $\mathsf{runs}_{\mu,a}$ is independent from the order of the letters in Σ.

Example 70. *Let us consider the infinite word $\mathbf{x} = \mu^\infty(\mathtt{a})$, where the binary morphism $\mu : \{\mathtt{a},\mathtt{b}\}^* \mapsto \{\mathtt{a},\mathtt{b}\}^*$ is defined as follows:*

$$\mu : \begin{array}{ccc} \mathtt{a} & \mapsto & \mathtt{abab} \\ \mathtt{b} & \mapsto & \mathtt{bb} \end{array} .$$

In this case $\mu_{\mathtt{a}}(i) = (i+1)2^i$ and $\mu_{\mathtt{b}}(i) = 2^i$, i.e., μ is polynomially divergent, and $p_{\mathbf{x}}(n) = \Theta(n \log\log n)$. We know by Theorem 67 that $r(\mu^i(\mathtt{a})) = \Theta(i)$. Moreover, one can see that $\mathsf{runs}_{\mu,\mathtt{a}} = 2^{i+1} = \Theta(2^i)$. Hence, μ is BWT-highly compressible.

Experimentally, we have observed that this behavior is not shared for all morphisms. For instance on the morphism $\psi : \{\mathtt{a},\mathtt{b},\mathtt{c}\}^* \mapsto \{\mathtt{a},\mathtt{b},\mathtt{c}\}^*$ defined as $\psi(\mathtt{a}) = \mathtt{abc}$, $\psi(\mathtt{b}) = \mathtt{bb}$ and $\psi(\mathtt{c}) = \mathtt{ccc}$, one can verify that

$$\mathsf{runs}_{\psi,\mathtt{a}}(i) = |\mathsf{rle}(\psi^i(\mathtt{a}))| = |(\mathtt{a},1)(\mathtt{b},1)(\mathtt{c},1)(\mathtt{b},2)(\mathtt{c},3)\cdots(\mathtt{b},2^i)(\mathtt{c},3^i)| = 2i+1.$$

On the other hand, tests suggest that $r(\psi^i(\mathtt{a})) = 4i$ for all $i > 2$, which is close to upper bound for the BWT-clustering ratio.

A first approach to the problem is to understand what properties of morphisms influence the growth of the run-length encoding on the iteration of the morphism.

Definition 71. *Let $\mu : \Sigma \mapsto \Sigma$ be a morphism such that $\mathsf{runs}_{\mu,a}(i) \leq K$, for every $i > 0$, for some $a \in \Sigma$, and some $K > 0$. Then we say that μ is* run-bounded on *a.*

Clearly, the function $\mathsf{runs}_{\mu,a}$ is monotone whenever μ is prolongable on a. In the next proposition, we give un upper bound on the measure $\mathsf{runs}_{\mu,a}$ under certain conditions.

Proposition 72. *Let μ be a morphism prolongable on $a \in \Sigma$ and let $\mathcal{R} = \{b \in \Sigma \mid \mu$ is run-bounded on $b\}$. If $\mu(a) = au$ with $u \in \mathcal{R}^+$ then $\mathsf{runs}_{\mu,a}(i) = \mathcal{O}(i)$, where $i \geq 0$ is the iteration of the morphism.*

Proof. Recall that $\mu^i(a) = au\mu(u)\mu^2(u)\cdots\mu^{i-1}(u)$ for every $i \geq 1$. Since μ is run-bounded on every letter in u,then $|\mathsf{rle}(\mu^i(u))| \leq K \cdot |u|$, for some $K > 0$ and every $i > 0$. Hence, we have that $\mathsf{runs}_{\mu,a}(i) \leq 1 + \sum_{j=0}^{i-1} |\mathsf{rle}(\mu^j(u))| \leq (K \cdot |u|) \cdot i = \mathcal{O}(i)$. □

Note that the morphism from Example 70 falls in the case described in Proposition 72. It is open the question whether the converse of the statement is true. Symmetrically, we obtain the following lower bound.

Proposition 73. *Let μ be a morphism prolongable on $a \in \Sigma$ such that $|\mathsf{rle}(\mu(a))| \geq 2$. If there exists $t > 0$ such that a occurs at least twice in $\mu^t(a)$, then the growth of $\mathsf{runs}_{\mu,a}$ is exponential.*

Proof. Let $t > 0$ be the smallest integer such that a occurs twice in $\mu^t(a)$. We prove by induction that $|\mu^{j\cdot t}(a)|_a$ grows as $\Omega(2^j)$. By definition of t, the case $j = 1$ is trivial. For the inductive step, assume it holds that $|\mu^{j\cdot t}(a)|_a = \Omega(2^j)$. Let a_k be the kth occurrence of a in $\mu^{j\cdot t}(a)$. This implies that $\mu^{(j+1)t}(a) = \mu^t(\mu^{jt}(a)) = \mu^t(a_1 \cdots a_2 \cdots a_{2^j} \cdots) = \mu^t(a_1)\cdots\mu^t(a_2)\cdots\mu^t(a_{2^j})\cdots$. Since $\mu^t(a_i)$ produces at least 2 a's for each i and the morphism is prolongable, then $|\mu^{(j+1)\cdot t}(a)|_a \geq 2|\mu^{j\cdot t}(a)|_a \in \Omega(2^{j+1})$. Note that since $|\mathsf{rle}(\mu(a))| \geq 2$, then $\mathsf{runs}_{\mu,a}(j \cdot t + 1) \geq 2|\mu^{j\cdot t}(a)|_a = \Omega(2^{j+1})$ (this holds even if we assume that the last letter of $\mu(a)$ is an a as well, since this would imply that $|\mathsf{rle}(\mu(a))| \geq 3$). Since every t steps the growth of the function is exponential, the overall growth is exponential too. □

The following corollary can be proved by using Propositions 49 and 73.

Corollary 74. *Let $\mu : \Sigma^* \mapsto \Sigma^*$ be a primitive morphism and prolongable on $a \in \Sigma$ and let $n = |\mu^i(a)|$, where $i \geq 1$ is the iteration of the morphism. It holds that $r(\mu^i(a)) = \mathcal{O}(\log n)$ and $\lim_{i\to\infty} \rho(\mu^i(a)) = 0$, therefore μ is BWT-highly compressible.*

We conclude this section with this final theorem on the efficiency of the BWT when we consider binary morphisms.

Theorem 75. *Let $\mu \equiv (\alpha, \beta)$ be a binary morphism prolongable on $\mathtt{a}$ such that $\mu \not\equiv (\mathtt{ab}^m, \mathtt{b}^n)$ for every $m \geq 1$, $n \geq 1$. Then $\lim_{i\to\infty} \rho(\mu^i(\mathtt{a})) = 0$, consequently μ is BWT-highly compressible.*

Proof. The proof is analogous if μ is prolongable on $\mathtt{b}$. If $\mu^\infty(\mathtt{a})$ is eventually periodic, then $r(\mu^i(\mathtt{a}))$ is $\Theta(1)$. By Propositions 52 and 73 we have that $\mathsf{runs}_{\mu,\mathtt{a}}(i)$ grows exponentially with i, except the case $\mu \equiv (\mathtt{ab}^m, \mathtt{b}^n)$. Then $\lim_{i\to\infty} \rho(\mu^i(\mathtt{a})) = 0$. If $\mu \equiv (\mathtt{ab}^m, \mathtt{b}^n)$, then $\rho(\mu^i(\mathtt{a})) = 1$ for every $m, n \geq 1$. Let us suppose $\mu^\infty(\mathtt{a})$ is aperiodic. If μ is primitive, the thesis follows from Corollary 74. If μ is not primitive, then $r(\mu^i(\mathtt{a})) = \mathcal{O}(i)$ (Theorem 65). Moreover, by using Propositions 73, 54 and 55, we have that $\mathsf{runs}_{\mu,\mathtt{a}}(i) = \Omega(2^i)$. Thus, $\lim_{i\to\infty} \rho(\mu^i(\mathtt{a})) = 0$ for any binary morphism with aperiodic purely morphic word, and the thesis follows. $\square$

Chapter 5

Effects of Morphisms on the BWT-Runs

In the previous chapter, we have studied the behavior of the Burrows Wheeler Transform when the word considered is a finite purely morphic word. Here we deal with a similar problem from another perspective: given any word $w \in \Sigma^*$ and a morphism $\mu : \Sigma^* \mapsto \Sigma^*$, how do the BWT-runs change when the μ is applied on w? In other words, how is the size $r(\mu(w))$ related to $r(w)$? This problem has brought us to new interesting characterizations of morphisms based on the effect that a morphism has on the BWT.

First, in Section 5.1 we show some properties of words and morphisms, which will become handy later in this chapter. Then, in Section 5.2 we show the main result, that is a new characterization of Sturmian morphisms based on its effect on the Burrows-Wheeler transform.

The results of this chapter appeared at the 34th Annual Symposium on Combinatorial Pattern Matching (CPM 2023) [44].

5.1 Abelian Order-Preserving and Order-Reversing Morphisms

In this section, we introduce the notions of abelian order-preserving an abelian order-reversing morphisms, and show how the measure r varies when one of these morphisms is applied to a word. Let us start by introducing some definitions regarding the rotations of morphic images of words.

Definition 76. *Let $\mu : \Sigma^* \mapsto \Gamma^*$ be a morphism. Then, we define the multisets*

$$\mathcal{I}_\mu(w) = \{\mu(w') \mid w' \in \mathcal{R}(w)\}$$
$$\mathcal{S}_\mu(w) = \{v\mu(w')u \mid u, v \in \Gamma^+, uv = \mu(a) \text{ for some } a \in \Sigma, \text{ and } aw' \in \mathcal{R}(w)\}.$$

The multiset $\mathcal{I}_\mu(w)$ corresponds to the rotations of $\mu(w)$ obtained by applying μ to the rotations of w. The multiset $\mathcal{S}_\mu(w)$ corresponds to all the remaining rotations of $\mu(w)$. We refer to the multiset $\mathcal{I}_\mu(w)$ as the *I-rotations* of $\mu(w)$, and to the multiset $\mathcal{S}_\mu(w)$ as the *S-rotations* of $\mu(w)$.

We continue by showing a lemma on the Hamming distance of two words with the same Parikh vector.

Lemma 77. *Let $w_1, w_2 \in \Sigma^*$ be such that $w_1 \neq w_2$ and $P(w_1) = P(w_2)$. Then, $d_H(w_1, w_2) \geq 2$.*

Proof. By definition of d_H, we have that $d_H(w_1, w_2) = 0$ if and only if $w_1 = w_2$. So, let us suppose by contradiction that $d_H(w_1, w_2) = 1$. Then, there exist two finite words $u, v \in \Sigma^*$ and two distinct indices $i < j \in [1, \sigma]$ such that $w_1 = ua_iv$ and $w_2 = ua_jv$. It follows that the Parikh vectors of w_1 and w_2 are respectively

$$P(w_1) = (|u|_{a_1} + |v|_{a_1}, \ldots, |u|_{a_i} + |v|_{a_i} + 1, \ldots, |u|_{a_j} + |v|_{a_j}, \ldots, |u|_{a_\sigma} + |u|_{a_\sigma})$$

and

$$P(w_2) = (|u|_{a_1} + |v|_{a_1}, \ldots, |u|_{a_i} + |v|_{a_i}, \ldots, |u|_{a_j} + |v|_{a_j} + 1, \ldots, |u|_{a_\sigma} + |u|_{a_\sigma}).$$

Thus, we obtain that the $P(w_1) \neq P(w_2)$, a contradiction. □

Since all the words in the same conjugacy class share the same Parikh vector, we can derive the following.

Corollary 78. *Let $w \in \Sigma^*$ be a word. Then, for every word $w' \in \mathcal{R}(w)$ such that $w' \neq w$, one has $d_H(w, w') \geq 2$.*

Here we consider two notions that we can use to classify morphisms.

Definition 79. *A morphism μ is* abelian order-preserving *if for every pair of words $u \neq v$ such that $P(u) = P(v)$, it holds that $u < v \iff \mu(u) < \mu(v)$.*

A morphism μ is abelian order-reversing *if for every pair of $u \neq v$ such that $P(u) = P(v)$, it holds that $u < v \iff \mu(u) > \mu(v)$.*

We want to point out that, in general, a morphism can be neither of both, as shown in the next simple example.

Example 80. *Let us consider the word $\mathtt{abc} \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^*$, where $\mathtt{a} < \mathtt{b} < \mathtt{c}$, and the morphisms $\mu, \eta : \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^* \mapsto \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^*$, defined as $\mu \equiv (\mathtt{b}, \mathtt{a}, \mathtt{c})$ and $\eta \equiv (\mathtt{ba}, \mathtt{ba}, \mathtt{ba})$. The order of the rotations is $\mathtt{abc} > \mathtt{bca} > \mathtt{cab}$. One can see, $\mu(\mathtt{abc}) = \mathtt{bac} < \mathtt{acb} = \mu(\mathtt{bca})$, so μ is not abelian order-preserving. On the other hand, $\mu(\mathtt{bca}) = \mathtt{acb} < \mathtt{cba} = \mu(\mathtt{cab})$, and therefore it is not abelian order-reversing either. Since $\eta(\mathtt{abc}) = \eta(\mathtt{bca}) = \eta(\mathtt{cab})$, also η is neither abelian order-preserving or -reversing.*

However, when binary acyclic morphisms are considered, we have a different scenario, as showed in the next lemma.

Lemma 81 ([44])**.** *Let $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \Sigma^*$ be an acyclic morphism. Then, μ is either abelian order-preserving or abelian order-reversing.*

A direct consequence of Lemma 81 is the order induced on the rotations of a binary morphism.

Corollary 82. *Let $w \in \{\mathtt{a}, \mathtt{b}\}^*$ be a binary word and let $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \{\mathtt{a}, \mathtt{b}\}^*$ be an acyclic morphism. Then, for all pairs of rotations $u, v \in \mathcal{R}(w)$, either $u < v \iff \mu(u) < \mu(v)$ (when μ is abelian order-preserving), or $u < v \iff \mu(u) > \mu(v)$ (when μ is abelian order-reversing).*

We have introduced new measures of sensitivities to study how the action of a morphism affects the BWT-runs.

Definition 83. *Let $\mu : \Sigma^* \mapsto \Sigma^*$ be a morphism and $w \in \Sigma^*$ a word. We define*

$$\Delta_\mu^+(w) = r(\mu(w)) - r(w)$$

and

$$\Delta_\mu^\times(w) = \frac{r(\mu(w))}{r(w)}.$$

In other words, $\Delta^+_\mu(w)$ and $\Delta^\times_\mu(w)$ are respectively the additive and multiplicative increase of the size r when the morphism μ is applied on the word w.

In the next theorem we show a tight lower bound on the measure Δ^+_μ.

Theorem 84. *Let $\mu : \{\mathtt{a},\mathtt{b}\}^* \mapsto \Sigma^*$ be an acyclic morphism. Then $\Delta^+_\mu(w) \geq 0$ for every $w \in \{\mathtt{a},\mathtt{b}\}^*$.*

Proof. Let $\mu \equiv (\alpha, \beta)$. Since $r(w) = r(w^m)$ for every $w \in \Sigma^*$ and $m > 1$, let us assume that w is primitive. For the proof, we assume that $|\alpha| \geq |\beta|$, and the other case is treated symmetrically. First, let us consider the case where β is not a suffix of α. Let moreover $x \in \Sigma^*$ be the longest common suffix between α and β. It follows that there exist $\alpha', \beta' \in \Sigma^+$ such that $\alpha = \alpha' x$ and $\beta = \beta' x$, and that the last letter of α' is different from the last of β' (otherwise x would be longer). Let $\mathcal{R}_x(\mu(w))$ denote the multiset of rotations of $\mu(w)$ with x as a prefix. Note that if $x = \varepsilon$, then $\mathcal{R}_x(\mu(w)) = \mathcal{I}_\mu(w)$. Since x appears in both α and β, it follows that $|\mathcal{R}_x(\mu(w))| \geq |w|$. Specifically, for each $i \in [1, |w|]$, there exists $t_i \in \mathcal{R}_x(\mu(w))$ such that $t_i = x\mu(w[i+1, |w|] \cdot w[1, i-1])v$, where v is either α' or β', depending on whether $w[i]$ is a or b respectively. The lexicographical order of these $|w|$ rotations of $\mu(w)$ with the same prefix correspond to the lexicographical order of the rotations in $\mathcal{I}_\mu(w)$, since by Corollary 78 the words $\bigcup_{i=1}^{|w|}\{\mu(w[i+1, |w|] \cdot w[1, i-1])\}$ must differ in at least one position. By Corollary 82 this is either in the same or in the reverse order with respect to the sorting of the rotations of w. Thus, there exists an injective coding $\lambda : \{\mathtt{a},\mathtt{b}\}^* \mapsto \Sigma'^* \subseteq \Sigma$ such that either $\lambda(\mathsf{bwt}(w))$ or $\lambda(\mathsf{bwt}(w)^R)$ is a subsequence of $\mathsf{bwt}(\mu(w))$, and therefore $r(\mu(w)) \geq r(w)$.

Let us now consider the case where β is suffix of α. Then, there exists a primitive word $u \in \Sigma^+$ and two integers $p \geq q \geq 1$ such that $\beta = u^q$, and $\alpha = \alpha' u^p$, with $\alpha' \in \Sigma^+$ that does not have u as suffix. Note that $\alpha' \neq \varepsilon$, otherwise we would have $\alpha\beta = u^p u^q = u^q u^p = \beta\alpha$, i.e. μ would not be acyclic. Let x be the longest common suffix between α' and u. If $x \neq \alpha'$, from analogous arguments to the case where β is not a suffix of α, we have at least $r(w)$ equal-letter runs in $\mathcal{R}_{xu^p}(\mu(w))$. Otherwise, if $x = \alpha'$, let us consider the word $y \in \Sigma^+$ such that $u = yx$. We can then consider the longest common suffix x' between xy and yx, which must be a proper suffix

(otherwise u would not be primitive), and apply the same reasoning over the set $\mathcal{R}_{x'xu^p}(\mu(w))$ and the thesis follows. □

In general, Theorem 84 does not hold in the case of larger alphabets, as the following example shows.

Example 85. *Consider the acyclic morphism* $\mu \equiv (\mathtt{b}, \mathtt{a}, \mathtt{c})$. *Then,* $bwt(\mathtt{bcba}) = \mathtt{bcab}$ *and* $bwt(\mu(\mathtt{bcba})) = bwt(\mathtt{acab}) = \mathtt{cbaa}$, *and therefore* $\Delta^+_\mu(\mathtt{bcba}) = 3 - 4 = -1 < 0$.

As consequence of Theorem 84, we obtain the following lower bound on $\Delta^\times_\mu$.

Corollary 86. *Let* $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \Sigma^*$ *be an acyclic morphism. Then,* $\Delta^\times_\mu(w) \geq 1$, *for every* $w \in \{a, b\}^*$.

We conclude this section with a characterization of cyclic and acyclic binary morphisms in terms of the number of BWT-runs in the images of the words.

Theorem 87. *A morphism* $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \Sigma^*$ *is cyclic if and only if there exists* $k > 0$ *such that* $r(\mu(w)) = k$ *for all* $w \in \{\mathtt{a}, \mathtt{b}\}^*$.

Proof. If $\mu \equiv (\alpha, \beta)$ is cyclic then there exists a primitive word $u \in \Sigma^*$ such that $\alpha = u^p$ and $\beta = u^q$, for some $p, q \geq 0$. Therefore, for each word $w \in \{\mathtt{a}, \mathtt{b}\}^*$, we have $r(\mu(w)) = r(u^{p \cdot |w|_\mathtt{a} + q \cdot |w|_\mathtt{b}}) = r(u)$. The other implication is a consequence of Theorem 84. In fact, by contraposition for each $k > 0$ we can find a word w such that $r(w) > k$. For instance, for the i-th Thue–Morse finite word holds that $r(\tau^i(\mathtt{a})) = 2i$ [20]. Hence, for any acyclic morphism μ and for all $i > \frac{k}{2}$, we have that $r(\mu(\tau^i(\mathtt{a}))) \geq r(\tau^i(\mathtt{a})) = 2i > k$ as well. □

As a consequence, we obtain a stronger result than Theorem 84.

Corollary 88. *A morphism* $\mu : \{\mathtt{a}, \mathtt{b}\}^* \mapsto \Sigma^*$ *is acyclic if and only if* $\Delta^+_\mu(w) \geq 0$ *for every* $w \in \{\mathtt{a}, \mathtt{b}\}^*$.

Proof. Theorem 84 proves the first implication, while the other direction can be deduced from the characterization of cyclic morphism in Theorem 87. □

5.2 Characterization of Sturmian Morphisms via r

In this section, we show that Sturmian morphisms are characterized by their effect on the BWT of any binary word.

Recall that a Sturmian morphism is obtained from the any composition of the following morphisms:

$$E : \begin{cases} \mathtt{a} \mapsto \mathtt{b} \\ \mathtt{a} \mapsto \mathtt{a} \end{cases} \qquad \varphi : \begin{cases} \mathtt{a} \mapsto \mathtt{ab} \\ \mathtt{b} \mapsto \mathtt{a} \end{cases} \qquad \tilde{\varphi} : \begin{cases} \mathtt{a} \mapsto \mathtt{ba} \\ \mathtt{b} \mapsto \mathtt{a} \end{cases}$$

Next, we show that every Sturmian morphism fixes the number of BWT-runs. From the definition of E, φ, and $\tilde{\varphi}$, and by Lemma 81, we derive the following.

Lemma 89. *Let $w \in \{\mathtt{a}, \mathtt{b}\}^*$ be a binary word. Then, for all pairs of rotations u and v of w, and for each $\chi \in \{E, \varphi, \tilde{\varphi}\}$, it holds that $u < v$ if and only if $\chi(u) > \chi(v)$.*

We prove now that the number of BWT-runs is preserved when any Sturmian morphism is applied on a binary word.

Lemma 90. *Let $w \in \{\mathtt{a}, \mathtt{b}\}^*$ be a binary word with $|alph(w)| = 2$. Then, for all $\chi \in \{E, \varphi, \tilde{\varphi}\}$, one has $r(w) = r(\chi(w))$. More in details, one has $bwt(E(w)) = \overline{bwt(w)^R}$ and $bwt(\varphi(w)) = bwt(\tilde{\varphi}(w)) = \overline{bwt(w)^R} \cdot \mathtt{a}^{|w|_\mathtt{a}}$.*

Proof. Since for each word w and each integer $k > 0$ we have $r(w) = r(w^k)$, let us assume that w is a primitive word. From Lemma 89, the case $\chi = E$ is trivial: in fact, from it follows that $\mathsf{bwt}(E(w)) = \overline{\mathsf{bwt}(w)^R}$, and therefore $r(w) = r(E(w))$.

For the case $\chi = \varphi$ one can observe that every $\mathtt{b}$ that occurs in $\varphi(w)$ is obtained from $\varphi(\mathtt{a})$, and therefore it is always preceded by an $\mathtt{a}$. Thus, the rotations of $\varphi(w)$ left to cover are all those starting with an $\mathtt{a}$, which therefore must also start with either $\varphi(\mathtt{a})$ or $\varphi(\mathtt{b})$. By Lemma 89, and by observing that $\varphi(\mathtt{a})$ ends with a $\mathtt{b}$ and $\varphi(\mathtt{b})$ ends with an $\mathtt{a}$, we have that $\mathsf{bwt}(\varphi(w)) = \overline{\mathsf{bwt}(w)^R} \cdot \mathtt{a}^{|w|_\mathtt{a}}$. Thus, we need to check if the run of $\mathtt{a}$'s at the end merges with the last letter of $\overline{\mathsf{bwt}(w)^R}$. This is equivalent to check that the first letter of $\mathsf{bwt}(w)$ is a $\mathtt{b}$, and by contradiction if the first rotation in lexicographical order is $u\mathtt{a}$ for some $u \in \{\mathtt{a}, \mathtt{b}\}^{n-1}$, then $\mathtt{a}u$ is a conjugate of w and $\mathtt{a}u < u\mathtt{a}$ for each binary word w, a contradiction.

For the case $\chi = \tilde{\varphi}$, each b's that appear in $\tilde{\varphi}(w)$ is obtained as first letter of the image $\tilde{\varphi}(\mathtt{a})$. This means that every b is preceded by the last letter of either $\tilde{\varphi}(\mathtt{a})$ or $\tilde{\varphi}(\mathtt{b})$, which in both cases it is an a. Let us consider then the rotations starting with an a. Whether this first a is obtained as $\tilde{\varphi}(\mathtt{b})$ or as proper suffix of $\tilde{\varphi}(\mathtt{a})$, we can write prefixes of these remaining n rotations of $\tilde{\varphi}(w)$ as $\mathtt{a} \cdot \tilde{\varphi}(u)$, for some $u \in \{\mathtt{a}, \mathtt{b}\}^{n-1}$. By Lemma 77 and Lemma 89, we can see that we can sort these rotations according to the reverse order with respect to the order of the words in $\mathcal{R}(w)$. Thus, if a rotation of w is of the type $u\mathtt{a}$, then the rotation starting with $\mathtt{a}\tilde{\varphi}(u)$ is followed by (and ends with) a b. On the other hand, if we had $u\mathtt{b}$, then the rotation with prefix $\mathtt{a}\tilde{\varphi}(u)$ is exactly $\tilde{\varphi}(\mathtt{b}u)$, which it must ends with an a. Then, $\mathsf{bwt}(\tilde{\varphi}(w)) = \overline{\mathsf{bwt}(w)^R} \cdot \mathtt{a}^{|w|_\mathtt{a}}$, and analogously to the case $\chi = \varphi$ it holds that the run $\mathtt{a}^{|w|_\mathtt{a}}$ merges with the last run of $\overline{\mathsf{bwt}(w)^R}$ and the thesis holds. □

Note that from Lemma 90 we can derive a method to construct $\mathsf{bwt}(\mu(w))$ starting from $\mathsf{bwt}(w)$, for every Sturmian morphism μ and every binary word w.

An example of the effect of the morphisms φ and $\tilde{\varphi}$ on the BWT is shown in Figure 5.1.

$M(w)$		$M(\varphi(w))$		$M(\tilde{\varphi}(w))$		
aabba	*b*	**a.a.ab.a.ab.a**	**b.**	*a.*	*a.a.ba.a.ba.*	*b*
abaab	*b*	**a.ab.a.ab.ab.**	**a.**	**a.**	**a.ba.a.ba.b**	**a.**
abbab	*a*	**a.ab.ab.a.a.a**	**b.**	*a.*	*a.ba.ba.a.a.*	*b*
baabb	*a*	**ab.a.a.ab.a.a**	**b.**	*a.*	*ba.a.a.ba.a.*	*b*
babaa	*b*	**ab.a.ab.ab.a.**	**a.**	**a.**	**ba.a.ba.ba.**	**a.**
bbaba	*a*	**ab.ab.a.a.ab.**	**a.**	**a.**	**ba.ba.a.a.b**	**a.**
		b.a.a.ab.a.ab.	*a*	**b**	**a.a.a.ba.a.b**	**a.**
		b.a.ab.ab.a.a.	*a*	**b**	**a.a.ba.ba.a.**	**a.**
		b.ab.a.a.ab.a.	*a*	**b**	**a.ba.a.a.ba.**	**a.**

FIGURE 5.1: BWT-matrices for the words $w = abbaba$, $\varphi(w)$, and $\tilde{\varphi}(w)$ respectively.

For $M(\varphi(w))$ and $M(\tilde{\varphi}(w))$, the dots separate the images of letters from w. The rotations in bold of $M(\varphi(w))$ and $M(\tilde{\varphi}(w))$ correspond to the words in $\mathcal{I}_\varphi(w)$ and $\mathcal{I}_{\tilde{\varphi}}(w)$ respectively. The block of rotations in gray at the end of both $M(\varphi(w))$ and $M(\tilde{\varphi}(w))$ are in correspondence of the equal-letter run of a's of length $|w|_a$, which occurs for every $w \in \{\mathtt{a}, \mathtt{b}\}^*$. One can see that $\mathsf{bwt}(\varphi(w)) = \mathsf{bwt}(\tilde{\varphi}(w)) = \overline{\mathsf{bwt}(w)^R} \cdot \mathtt{a}^{|w|_\mathtt{a}}$.

A clear consequence of Lemma 90 is that, for each binary word w, the words $\varphi(w)$ and $\tilde{\varphi}(w)$ are conjugate. More in detail, the images are just one shift away from each other. In fact, for any binary word $w = w_1 w_2 \cdots w_n$ we have that $\varphi(w) = \varphi(w_1 w_2 \cdots w_n) = \mathtt{a}v_1\mathtt{a}v_2 \cdots \mathtt{a}v_n$, where for each $i \in [1, n]$ we have $v_i = \mathtt{b}$ if $w_i = \mathtt{a}$, or $v_i = \varepsilon$ if $w_i = \mathtt{b}$. On the other hand, for the same word w we have $\tilde{\varphi}(w) = \tilde{\varphi}(w_1 w_2 \cdots w_n) = v_1\mathtt{a}v_2 \cdots \mathtt{a}v_n\mathtt{a}$, where analogously to the previous case $v_i = \mathtt{b}$ if $w_i = \mathtt{a}$, or $v_i = \varepsilon$ if $w_i = \mathtt{b}$. Hence, $\tilde{\varphi}(w)$ can be obtained by applying a shift to the left on $\varphi(w)$.

Here we show the main theorem of this chapter, giving a new characterization of Sturmian morphisms.

Theorem 91. *Let μ be a binary morphism. Then, the following are equivalent:*

1. $\Delta^{+}_{\mu}(w) = 0$ *for every word w with* $|\mathit{alph}(w)| = 2$*;*

2. *μ is a Sturmian morphism.*

Proof. By Theorem 7 and Lemma 90, all Sturmian morphisms preserve the number of BWT-runs. Conversely, suppose that μ preserves the number of BWT-runs. By Theorem 84, such a morphism must be acyclic. Let $s = \lim_{i\to\infty} s_i$ be a characteristic Sturmian word. For every i, the word $\mu(s_i)$ has 2 runs in its BWT, hence it is circularly balanced (Theorem 14). Let us consider the word $\mu(s) = \lim \mu(s_i)$. Since it is obtained by applying an acyclic morphism to a Sturmian word, it is balanced and aperiodic [22]. Then, $\mu(s)$ is Sturmian by using Theorem 3, whence μ is a Sturmian morphism by applying Theorem 6. □

This characterization represents another link between the fields of formal languages and combinatorics on words with notions from data compression and indexed structures. Note that such a characterization can be used to obtain a new proof for Theorem 14.

It would be interesting to explore more practical applications. For instance, one may think of the problem of a hidden data structure working in $\mathcal{O}(polylogr)$ space, which is indexed for string pattern matching and accessible by multiple parties. If actor A wants to hide the content of a binary text, this can be mapped through a Sturmian morphism $\varphi^{\star} = \chi_1 \circ \chi_2 \circ \cdots \circ \chi_k$, with $\chi_i \in \{E, \varphi, \tilde{\varphi}\}$ for all $i \in [1, k]$,

and represent the mapped text in the same space r, while the k-tuple of morphisms $K = (\chi_1, \chi_2, \dots \chi_k)$ would be the key to access the real text.

The first problem that one may observe is that this regularity may be easily cracked, since any acyclic binary morphism is a code. However, the Sturmian morphism can be chosen to make this operation unfeasible. For instance, if $\varphi^\star = \varphi^{100}$, any a in the text would be mapped into the 100th Fibonacci word, whose length is $|\varphi^\star(\mathtt{a})| = 354224848179261915075$, while the measure r stays untouched, but we can choose k way larger than this. Further, since $|\chi_1 \circ \chi_2 \circ \cdots \circ \chi_k(a)| < 2^k$, we can represent the extended first and last runs of the BWT in $\mathcal{O}(k)$ bits of space.

On the other hand, the pattern should be increased of the same size, leading to the analogous time and space resources needed to crack the key. Nonetheless, recall that the original BWT stays untouched, meaning that either the $\mathsf{bwt}(w)$, $\overline{\mathsf{bwt}(w)}$, or their reverse, appear as factor of $\mathsf{bwt}(\varphi^\star(w))$, and its location can be found through the key K. This may suggest the design of an indexed data structure which indexes only n consecutive letters of the BWT of $\varphi^\star(w)$, and that can be properly queried when the key K is known.

5.3 On the Sensitivity of r for the Application of Morphisms

We conclude this chapter with some general remarks on the impact that morphisms have on the Burrows-Wheeler Transform of binary words.

After characterizing Sturmian morphisms as those morphisms fixing the BWT-runs on binary words, it is natural wondering if the application of any other morphism, instead of preserving the number of runs in the BWT, increases the size r by a given constant $k > 0$. If such a constant exists, it has to be an even integer. In fact, unless considering powers of a single letter, the BWT of any binary word starts with a b and ends with an a.

In this section, we show some final remarks on the effect of morphisms on the measure r. First, we present a family of morphisms with a fixed $2k$ increase for all $k > 0$. Then, we show that this effect is not shared by all morphisms, i.e., we can find an example of morphism such that the difference is in the BWT-runs is arbitrarily large.

In the same work, we have showed that for every $k > 0$, we can find an infinite number of morphisms that increase the the BWT-runs of any binary word by exactly $2k$. Among these, for $k = 1$ we have found a family of morphisms by the well-known Thue-Morse morphism $\tau \equiv (\mathtt{ab}, \mathtt{ba})$, defined as follows.

Definition 92. *A binary morphism is* Thue–Morse-like *if it has the form* $\tau_{p,q} \equiv (ab^p, ba^q)$ *for some* $p, q > 0$.

The structure of the BWT of Thue–Morse words has been studied before and it is well understood [20, 35]. We have generalized such results by showing how to derive $\mathsf{bwt}(\mu(w))$ from $\mathsf{bwt}(w)$ for every Thue–Morse-like morphism μ and every binary word w. The following lemma summarizes the effect of Thue–Morse-like morphisms on the BWT.

Lemma 93. *[44] For every binary word w such that $\mathsf{alph}(w) = \{a, b\}$, it holds that*

$$\mathsf{bwt}(\tau_{p,q}(w)) = b^{|w|_b} a^{(q-1)|w|_b} \cdot \overline{\mathsf{bwt}(w)} \cdot b^{(p-1)|w|_a} a^{|w|_a},$$

and that $r(\tau_{p,q}(w)) = r(w) + 2$.

As a consequence of Lemma 91 and Lemma 93, we obtain the following corollary.

Corollary 94. *Given an integer $t \geq 0$, there exist infinite family of binary morphisms μ such that $\Delta^+_\mu(w) = 2t$ and $\Delta^\times_\mu(w) \leq t + 1$, for every word w with $|\mathsf{alph}(w)| = 2$.*

Proof. Any morphism of this type is obtained from the compositions of t Thue–Morse-like morphisms, which increase the size of the BWT by $2t$ runs, and as many Sturmian morphisms desired, since each of these morphisms preserve the measure r, and therefore $\Delta^+_\mu(w) = 2t$. Moreover, since $r(w) \geq 2$, we have that $\Delta^\times_\mu(w) = \frac{r(\mu(w))}{r(w)} = \frac{r(w)+2t}{r(w)} = 1 + \frac{2t}{r(w)} \leq t + 1$. □

Building binary words with any desired BWT-runs is another brick into a deeper understanding of the measure r. As a consequence, we can extend the results by Brlek et al. from [20] on the number of BWT-runs of words generated by iterating the composition of the Fibonacci morphism with the Thue–Morse morphism to any composition of Sturmian morphisms and Thue–Morse-like morphisms.

Analogously to definitions of [1], we introduce the following.

Definition 95. *Given a measure of repetitiveness λ, the* additive sensitivity *and* multiplicative sensitivity *for a morphism μ are respectively, the functions*

$$AS_\mu(\lambda, n) = \max_{w \in \Sigma^n} \lambda(\mu(w)) - \lambda(w) \text{ and } MS_\mu(\lambda, n) = \max_{w \in \Sigma^n} \frac{\lambda(\mu(w)}{\lambda(w)}.$$

One can see that, when the measure r is considered, this is equivalent to:

- $\mathsf{AS}_\mu(r, n) = \max_{w \in \Sigma^n}(\Delta^+_\mu(w))$;
- $\mathsf{MS}_\mu(r, n) = \max_{w \in \Sigma^n}(\Delta^\times_\mu(w))$.

If it truly holds that any morphism increase the measure r by a constant value, then for each morphism μ there would exist a k such that $\mathsf{AS}_\mu(r, n) \leq k$, and this would have consequences to Question 68. However, as we have proved in [44], this is not the case.

Lemma 96 ([44])**.** *Let θ be the period-doubling morphism. For any positive integer k there exist a word w such that $\Delta^+_\theta(w) > k$.*

Let us consider the words $s_i = \mathtt{ab}^i\mathtt{a} \cdot u_i$ and $e_i = \mathtt{ab}^i\mathtt{a} \cdot u_i^R$, where $u_i = \mathtt{a}^{2k-i}\mathtt{ba}^{i-2}$, for some $i > 0$. The words used in the proof of Lemma 96 are obtained as $w_k = (\prod_{i=2}^k s_i e_i)a^k$. Note that Question 68 is still open, since $\theta^i(\mathtt{a}) \neq w_k$ for all pairs $i, k > 1$. Furthermore, one can notice that $|w_k| = \Theta(k^2)$. Thus, we obtain the following lower bound on the sensitivity for the period doubling morphism.

Proposition 97. *Let θ be the period-doubling morphism. It holds that $AS_\theta(r, n) = \Omega(\sqrt{n})$.*

We wonder then what properties of morphisms affect in different ways the Burrows-Wheeler Transform.

On one hand, it would be interesting to investigate if for a binary morphism μ with $\mu(\mathtt{a})$ and $\mu(\mathtt{b})$, there exists a $k >$ such that $\mathsf{AS}_\mu(r, n) \leq k$ for every n. Note in fact that the period doubling morphism does not fall in this category ($\theta(\mathtt{b}) = \mathtt{b}^2$). Experimentally, the analogous behavior of the period doubling morphism on the BWT has been noticed for analogous morphisms having a power as image of a letter.

At the other end of the spectrum, one can investigate what properties of words trigger the increasing effect of morphisms on the BWT. Such a study could allow to obtain new lower bounds on the worst-case scenario sensitivities.

Chapter 6

Applications in BWT-Runs Bounded Space

Despite the theoretical limitations of r as a measure of repetitiveness, the Burrows-Wheeler Transform is still at the basis of the most used tools in Bioinformatics [82, 79]. Its efficiency in the field is mostly due to the high number of repetitions that occur within the biological sequences, and after the introduction of the r-index [49], new methods have been implemented for solving pattern matching problems in pangenomical contexts [116, 16].

In this chapter, we present some applications of the Burrows-Wheeler Transform, with practical usage in bioinformatics.

In Section 6.1, we describe what properties can be used to find Maximal Unique Matches in a space proportional to r. Compared to some of the most used tools to solve the same problem, we drastically use less space, proposing an accessible solution to researchers with limited resources. Our implementation of the algorithm is available at `https://github.com/saragiuliani/mum-phinder`. The work of the thesis is part of a paper appeared during the 20th International Symposium on Experimental Algorithms (SEA 2022) [57].

In Section 6.2, we present a preliminary work on the size of the BWT for collections of sampled strings, with applications for storing and indexing biological reads. This work has been developed during a visiting period at the University of Helsinki, under the supervision of prof. Simon J. Puglisi.

6.1 Computing Maximal Unique Matches with the r-Index

The alignment of sequences is one of the major tasks in bioinformatics, with applications for genome sequencing, genealogical tree building, species classification, etc. This problem is mostly dealt by keeping track of *seeds*, that are substrings which can be used as anchors for the alignment. In literature, it is typical to use *maximal exact matches* (in short MEMs) as seeds of the alignment [30, 19]. Very recently, Rossi et al. [116] designed and implemented an algorithm which computes MEMs in a space proportional to $\mathcal{O}(r + g)$, where the two measures refer to size of the BWT-runs and of a grammar representing the reference string.

Intuitively, the less occurrences of a same maximal exact match we have in a text and in the pattern, the more likely is to get a proper alignment: this is the idea which motivates the notion of *maximal unique match* (MUM), that is a maximal match that occurs only once in both the text and the pattern. The problem of finding MUMs is not new in literature, and different approaches have been proposed, but most of them operate in a space linearly proportional to the size of the text [76, 96, 37]. When huge amount of data are involved, like in a pangenomic context where hundreds of GigaByte may be required just to represent the collection of the sequences, these solutions are not feasible.

First, we explain the problem and introduce the notions that will be used throughout the section. Then, we explain how to build a reusable index of a text taking $\mathcal{O}(r + g)$ space, and supporting detection of maximal unique matches in time linearly proportional to the size of pattern. Finally, we show some experimental results obtained on real data collections.

6.1.1 Maximal Unique Matches and Extended Matching Statistics

Given an alphabet Σ and a letter $\$ \notin \Sigma$, we denote the set $\Sigma_\$ = \Sigma \cup \{\$\}$. Throughout the section, we will refer to the word $T \in \Sigma_\$^*$ as *text*, and the word $P \in \Sigma^*$ as the *pattern*. More in detail, we expect $T = u\$$, for some $u \in \Sigma^*$. Given a text $T \in \Sigma^n$ and a pattern $P \in \Sigma^m$, we refer to any factor in P that also occurs in T as a *match*. A match M in P can be defined as a pair (i, ℓ) such that $M = P[i, i + \ell - 1]$. We say that M is *maximal* if the match can not be extended neither on the left nor on the right,

i.e., either $i = 1$ or $P[i-1, i+\ell-1]$ does not occur in T, and either $i = m-\ell+1$ or $P[i, i+\ell]$ does not occur in T.

Definition 98. *Given a text T and a pattern P, a* Maximal Unique Match *(MUM) is a maximal match that occurs exactly once in T and P.*

Example 99. *Let T = ACACTCTTAC**ACC**ATATCATCAA\$ be the text and P = AACCTAA the pattern. The factor* **AA** *is maximal in P and occurs only once in T, while it is repeated in P at positions* 1 *and* 6*. The factor* **CT** *of P starting in position 4 is a maximal match that occurs only once in P, but it is not unique in T. The factor* **CC** *of P starting in position 3 is unique in both T and P, but both can be extended on the left with an* **A***. On the other hand, the factor $P[2,4] = T[11,13]$ =***ACC** *is a MUM.*

From now on, we refer to the set of all maximal unique matches between T and P as MUMs. Boucher et al. [17] have recently developed an algorithm to compute *maximal exact matches* (MEMs), which are the maximal matches that are not necessarily unique neither in T nor P. Their approach consist in building an index for T, taking $\mathcal{O}(r+g)$ words of space, where r is the number of runs of the BWT of T, and g is the size of a SLP representing the text T (to recall the definition, see Subsection 2.3.2).

The main data structure taking $\mathcal{O}(r)$ words is the *r-index*, which consists in a combination of the run-length encoding of the BWT and $\mathcal{O}(r)$ samples of the suffix array in correspondence of the boundaries of the BWT-runs [49]. The construction of the r-index and other data structures taking $\mathcal{O}(r)$ words of space are computed in a space proportional to size of the parse and the dictionary obtained through the *prefix-free parsing*. Since the notion of prefix-free parsing is out of the scope of this thesis, we delegate its definition and effects to related works [18, 103]. Even though there is not a strong theoretical literature on the size of the prefix-free parsing, the reader should be aware that experimentally it has been shown to be extremely succinct for highly repetitive texts.

Then, in the approach of Boucher et al. they compute in streaming the *matching statistics*, for which we report the definition as given in [17].

Definition 100. *The matching statistics MS of a pattern $P \in \Sigma^m$ with respect to a text $T \in \Sigma_\$^n$ is an array of (position, length)-pairs $MS[1,m]$ such that*

- $P[i, i + MS[i].len - 1] = T[MS[i].pos, MS[i].pos + MS[i].len - 1]$;
- *either* $i = m - MS[i].len + 1$, *or* $P[i, i + MS[i].len]$ *does not occur in* T.

That is, $MS[i].pos$ *is the starting position in* T *of an occurrence of the longest prefix of* $P[i, m]$ *that occurs in* T*, and* $MS[i].len$ *is its length.*

A known property of the matching statistics is that for all $i \in [1, m-1]$, $\mathsf{MS}[i+1].\mathsf{len} \geq \mathsf{MS}[i].\mathsf{len} - 1$.

Even though the Toehold Lemma by Policriti and Prezza [110] allows the r-index to support pattern matching queries in $\mathcal{O}(r)$ space, to the best of our knowledge it does not permit all by itself to compute maximal exact matches, or the matching statistics of the pattern P. Thus, the solution proposed by Boucher et al. [17] uses extra $\mathcal{O}(g)$ words to create an index supporting *LCE queries*, where for a given text T, we have that $\mathsf{LCE}(i, j) = |\ell cp(T[i, n], T[j, n])|$ for all $i, j \in [1, n]$. As shown later, the computation of the matching statistics can be detected by performing LCE queries. We do not know whether it is possible to create a $\mathcal{O}(r)$ space index supporting LCE queries, thus the need for the extra $\mathcal{O}(g)$ space. To show how to further compute MUMs within the same space bound, we extend the definition of MS array with an additional information field to each entry.

Definition 101. *Given a text* $T \in \Sigma_{\$}^{n}$ *and a pattern* $P \in \Sigma^{m}$*, we define the extended matching statistics* eMS *as an array of* $(pos, len, slen)$*-tuples* $eMS[1, m]$ *such that*

- $eMS[i].pos = MS[i].pos$*, and* $eMS[i].len = MS[i].len$;
- $eMS[i].slen$ *is the largest value* ℓ *for which there exists* $p \neq eMS[i].pos$ *such that* $P[i, i + \ell - 1] = T[p, p + \ell - 1]$.

In other words, $eMS[i].slen$ *is the length of the second longest match of a prefix* $P[i, m]$ *in* T.

Note that $\mathsf{eMS}[i].\mathsf{slen} \leq \mathsf{eMS}[i].\mathsf{len}$, for any $i \in [1, m]$.

6.1.2 Checking Maximality and Uniqueness of Matches

Here we explain how to check if a match is maximal and unique in T from the eMS array. Lemma 102 shows how to verify if a match occurs only once in T.

Lemma 102. *Given a text $T \in \Sigma_{\$}^{n}$, a pattern $P \in \Sigma^{m}$, and the eMS array computed for P with respect to T, let $M = P[i, i + eMS[i].len - 1] = T[eMS[i].pos, eMS[i].pos + eMS[i].len - 1]$ be a maximal match between P and T. Then, M occurs exactly once in T if and only if $eMS[i].slen < eMS[i].len$.*

Proof. For the first direction, we assume by contradiction that M is unique in T and that $\mathsf{eMS}[i].\mathsf{slen} \geq \mathsf{eMS}[i].\mathsf{len}$. By definition, $\mathsf{eMS}[i].\mathsf{slen} \leq \mathsf{eMS}[i].\mathsf{len}$, hence we assume $\mathsf{eMS}[i].\mathsf{slen} = \mathsf{eMS}[i].\mathsf{len}$. By definition of $\mathsf{eMS}[i].\mathsf{slen}$ there exists $p \neq \mathsf{eMS}[i].\mathsf{pos}$ such that $M = P[i, i + \mathsf{eMS}[i].\mathsf{slen} - 1] = T[p, p + \mathsf{eMS}[i].\mathsf{slen} - 1] = T[\mathsf{eMS}[i].\mathsf{pos}, \mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} - 1]$, that contradicts the assumption that M occurs only once in the text T.

Analogously, assume that $\mathsf{eMS}[i].\mathsf{slen} < \mathsf{eMS}[i].\mathsf{len}$ and that there exists a position $j \neq \mathsf{eMS}[i].\mathsf{pos}$ such that $T[j, j + \mathsf{eMS}[i].\mathsf{len} - 1] = T[\mathsf{eMS}[i].\mathsf{pos}, \mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} - 1]$. However, this is in contradiction with the definition of $\mathsf{eMS}[i].\mathsf{slen}$ and the assumption of $\mathsf{eMS}[i].\mathsf{slen} < \mathsf{eMS}[i].\mathsf{len}$, concluding the proof. □

We check the maximality of a match in the pattern using an analogous approach as in [116], that we summarize with the following Lemma.

Lemma 103. *Given a text $T \in \Sigma_{\$}^{n}$, a pattern $P \in \Sigma^{m}$, and the eMS array computed for P with respect to T, let $M = P[i, i + eMS[i].len - 1]$ be a match with a text T. Then, M is a maximal match if and only if either $i = 1$, or $eMS[i - 1].len \leq eMS[i].len$ for all $i \in [2, m]$.*

Proof. First we show that if $M = P[i\mathcal{L}i + \mathsf{eMS}[i].\mathsf{len})$ is a maximal match then either $i = 0$ or $\mathsf{eMS}[i - 1].\mathsf{len} \leq \mathsf{eMS}[i].\mathsf{len}$. Let us assume that M is not maximal and either $i = 0$ or $\mathsf{eMS}[i - 1].\mathsf{len} \leq \mathsf{eMS}[i].\mathsf{len}$, hence either $P[i, i + \mathsf{eMS}[i].\mathsf{len}]$ occurs in T, or $P[i - 1, i + \mathsf{eMS}[i].\mathsf{len} - 1]$ occurs in T. The former case is in contradiction with the definition of eMS, hence $P[i - 1, i + \mathsf{eMS}[i].\mathsf{len} - 1]$ occurs in T. This implies that $i > 0$, and that $\mathsf{eMS}[i - 1].\mathsf{len} = \mathsf{eMS}[i].\mathsf{len} + 1$, in contradiction with the hypothesis that $\mathsf{eMS}[i - 1].\mathsf{len} \leq \mathsf{eMS}[i].\mathsf{len}$.

Now we show that if either $i = 0$ or $\mathsf{eMS}[i - 1].\mathsf{len} \leq \mathsf{eMS}[i].\mathsf{len}$ then M is a maximal match. By definition of $\mathsf{eMS}[i].\mathsf{len}$, we know that either $i + \mathsf{eMS}[i].\mathsf{len} = m$, or $P[i, i + \mathsf{eMS}[i].\mathsf{len}]$ does not occur in T, that is M cannot be extended on the right in P. If $i = 0$ we can not further extend the match M on the left, hence $P[1, \mathsf{eMS}[1].\mathsf{len}] -$

1. If $i > 0$, then by definition of matching statistics it holds that $\mathsf{eMS}[i-1].\mathsf{len} \leq \mathsf{eMS}[i].\mathsf{len} + 1$. Note that if there exists a letter $a \in \Sigma$ such that $P[i-1, i-1+\mathsf{eMS}[i-1].\mathsf{len}-1] = aM$ and aM occurs in T, then $\mathsf{eMS}[i-1] = \mathsf{eMS}[i]+1$. Hence if $\mathsf{eMS}[i-1] = \mathsf{eMS}[i]+1$ then it is easy to see that M is not maximal because it can be extended on the left. Furthermore, it follows that if $\mathsf{eMS}[i-1] \leq \mathsf{eMS}[i]$, then M cannot be extended on the left, hence it is maximal and the thesis follows. □

Let $\mathcal{L} \subseteq [1, m]$ be the subset of positions in P such that both Lemmas 102 and 103 hold, i.e., $\mathcal{L}$ contains all the positions in P where a maximal match unique in T starts. One can notice that if a match $M_i = P[i, i+\mathsf{eMS}[i].\mathsf{len}-1]$ is a MUM, then $i \in \mathcal{L}$.

We first show that given $i \in \mathcal{L}$, if a match M_i is not unique in P, then the second occurrence of M_i in P is contained in another maximal match unique in T.

Lemma 104. *Given a text $T \in \Sigma_{\$}^{n}$, a pattern $P \in \Sigma^{m}$, and the eMS array computed for P with respect to T, let $\mathcal{L}$ be the subset of positions in P such that $M_i = P[i, i+\mathsf{eMS}[i].\mathsf{len}-1]$ is maximal and occurs only once in T, for all $i \in \mathcal{L}$. Then, M_i is not unique in P if and only if there exist $i' \in \mathcal{L} \setminus \{i\}$ and two strings $u, v \in \Sigma^*$ such that $M_{i'} = uM_iv$ is a factor of P.*

Proof. Let us assume by contradiction that such i' does not exist, then let $j \notin \mathcal{L}$ be a position in P such that $P[j, j+|M_i|-1] = M_i$. Since $j \notin \mathcal{L}$ then either $P[j, j+|M_i|-1]$ is not unique in T, or it is not maximal. The former case it contradicts $i \in \mathcal{L}$ because $P[j, j+|M_i|-1] = M_i$ occurs twice in T. Hence, $P[j, j+|M_i|-1]$ occurs only once in T and it is not maximal, therefore there exists $k \in \mathcal{L}$ such that $k \leq j$ and $|M_k| > |M_i|$, which contradict the hypothesis.

The other direction of the proof is straightforward since by definition of $M_{i'}$, either M_i occurs twice in P or it is not maximal. □

The following lemma shows, for any $i \in \mathcal{L}$, if a match M_i is unique in P by using the eMS array.

Lemma 105. *Given a text $T \in \Sigma_{\$}^{n}$, a pattern $P \in \Sigma^{m}$, and the eMS array computed for P with respect to T, let $\mathcal{L}$ be the subset of positions in P such that $M_i = P[i, i+\mathsf{eMS}[i].\mathsf{len}-1]$ is maximal and occurs only once in T, for all $i \in \mathcal{L}$. Then, M_i occurs only once in P if and only if, for all $i' \in \mathcal{L} \setminus \{i\}$, either $\mathsf{eMS}[i].\mathsf{pos} < \mathsf{eMS}[i'].\mathsf{pos}$ or $\mathsf{eMS}[i].\mathsf{len} + \mathsf{eMS}[i].\mathsf{pos} > \mathsf{eMS}[i'].\mathsf{len} + \mathsf{eMS}[i'].\mathsf{pos}$.*

Proof. We first show that if M_i occurs only once in P then for all $i' \in \mathcal{L} \setminus \{i\}$, either $\mathsf{eMS}[i].\mathsf{pos} < \mathsf{eMS}[i'].\mathsf{pos}$, or $\mathsf{eMS}[i].\mathsf{len} + \mathsf{eMS}[i].\mathsf{pos} > \mathsf{eMS}[i'].\mathsf{len} + \mathsf{eMS}[i'].\mathsf{pos}$. Since $\mathcal{L}$ contains only positions of maximal matches unique in T, then for all for $i \in \mathcal{L}$ we can map M_i to its occurrence in the text $T[\mathsf{eMS}[i].\mathsf{pos}, \mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} - 1]$. Since M_i occurs only once in T, by Lemma 104 we have that $\mathsf{eMS}[i'].\mathsf{pos} = \mathsf{eMS}[i].\mathsf{pos} - |u|$ and $\mathsf{eMS}[i'].\mathsf{len} = \mathsf{eMS}[i].\mathsf{len} + |u| + |v|$. Hence, $\mathsf{eMS}[i'].\mathsf{pos} \leq \mathsf{eMS}[i].\mathsf{pos}$ and $\mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} \leq \mathsf{eMS}[i'].\mathsf{pos} + \mathsf{eMS}[i'].\mathsf{len}$.

We now show the other direction of the implication. If given a position $i \in \mathcal{L}$ for all $i' \in \mathcal{L} \setminus \{i\}$, either $\mathsf{eMS}[i].\mathsf{pos} < \mathsf{eMS}[i'].\mathsf{pos}$ or $\mathsf{eMS}[i].\mathsf{len} + \mathsf{eMS}[i].\mathsf{pos} > \mathsf{eMS}[i'].\mathsf{len} + \mathsf{eMS}[i'].\mathsf{pos}$ then M_i occurs only once in P. Assuming by contradiction that there exists a position $i \in \mathcal{L}$ such that for all $i' \in \mathcal{L} \setminus \{i\}$, either $\mathsf{eMS}[i].\mathsf{pos} < \mathsf{eMS}[i'].\mathsf{pos}$ or $\mathsf{eMS}[i].\mathsf{len} + \mathsf{eMS}[i].\mathsf{pos} > \mathsf{eMS}[i'].\mathsf{len} + \mathsf{eMS}[i'].\mathsf{pos}$ and M_i does not occur only once in P, then by Lemma 104 there exist $j \in \mathcal{L}$ and two possibly empty strings u, v such that $M_j = uM_iv$ is a factor of P. It is easy to see that $\mathsf{eMS}[j].\mathsf{pos} = \mathsf{eMS}[i].\mathsf{pos} - |u|$ and $\mathsf{eMS}[j].\mathsf{len} = \mathsf{eMS}[i].\mathsf{len} + |u| + |v|$. Hence, $\mathsf{eMS}[j].\mathsf{pos} \leq \mathsf{eMS}[i].\mathsf{pos}$ and $\mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} \leq \mathsf{eMS}[j].\mathsf{pos} + \mathsf{eMS}[j].\mathsf{len}$, in contradiction with the hypothesis, concluding the proof. □

We can summarize the previous lemmas in the following theorem.

Theorem 106. *Given a text $T \in \Sigma_{\$}^n$, a pattern $P \in \Sigma^m$, and the eMS array computed for P with respect to T, for all $0 \leq i < m$, $M_i = P[i, i + \mathsf{eMS}[i].\mathsf{len})$ is a MUM if and only if $i \in \mathcal{L}$ and Lemma 105 holds.*

Example 107. *Let T = ACACTCTTAC**ACC**ATATCATCAA\$ be the text and P = AACCTAA the pattern. In the Table 6.1, we report the values of the eMS of P with respect to T.*

i	1	2	3	4	5	6	7
P[i]	A	A	C	C	T	A	A
eMS[i].pos	22	11	12	6	7	22	9
eMS[i].len	2	3	2	2	2	2	1
eMS[i].slen	1	2	1	2	2	1	1

TABLE 6.1: Example of extended matching statistics for the pattern $P = AACCTAA$ with respect to the text $T = ACACTCTTACACCATATCATCAA\$$.

It is easy to check that $\mathcal{L} = \{1, 2, 6\}$*, where* $\mathcal{L}$ *contains those indices* i *which verify both Lemma 102* ($\mathit{eMS}[i].\mathit{slen} < \mathit{eMS}[i].\mathit{len}$) *and Lemma 103 (either* $i = 1$ *or* $\mathit{eMS}[i-1].\mathit{len} \leq \mathit{eMS}[i].\mathit{len}$*). Note that* $\mathit{eMS}[0].\mathit{pos} = \mathit{eMS}[6].\mathit{pos}$ *and* $\mathit{eMS}[1].\mathit{len} = \mathit{eMS}[6].\mathit{len}$*, and by Lemma 105 we know that* $P[1, 2](= P[6, 7])$ *is repeated in* P*. Since* $\mathit{eMS}[2].\mathit{pos} < \mathit{eMS}[1].\mathit{pos} = \mathit{eMS}[6].\mathit{pos}$*, by Theorem 106 the match* $P[2, 4] = T[11, 13] = ACC$ *is a MUM.*

By Lemmas 102 and 103, we can check in constant time if an index i belongs to $\mathcal{L}$. We can build $\mathcal{L}$ in streaming while computing the eMS array. Observe that a match $M_i = P[i, i + \mathsf{eMS}[i].\mathsf{len} - 1]$ such that $i \in \mathcal{L}$ is a MUM if and only if it is not fully contained into another candidate, i.e. it does not exist $j \in \mathcal{L} \setminus \{i\}$ such that (i) $\mathsf{eMS}[j].\mathsf{pos} \leq \mathsf{eMS}[i].\mathsf{pos}$, and (ii) $\mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} \leq \mathsf{eMS}[j].\mathsf{pos} + \mathsf{eMS}[j].\mathsf{len}$ (Theorem 106). Hence, if we sort the elements in $\mathcal{L}$ according to $\mathsf{eMS}[i].\mathsf{pos}$, starting from $\mathcal{L}[1]$ can compare it with the following element, and if both factors are not contained into the other, we store in the set MUMs the one with the smallest starting position, and keep track of the other one. Otherwise, we simply discard the one that is repeated and continue with the following iteration. To handle the special case when two candidates $i \neq j \in \mathcal{L}$ are such that $T[\mathsf{eMS}[i].\mathsf{pos}, \mathsf{eMS}[i].\mathsf{pos} + \mathsf{eMS}[i].\mathsf{len} - 1] = T[\mathsf{eMS}[j].\mathsf{pos}, \mathsf{eMS}[j].\mathsf{pos} + \mathsf{eMS}[j].\mathsf{len} - 1]$, we further keep track whether the current maximal match has more occurrences. This final procedure, excluding the building time for $\mathcal{L}$ that is done in streaming, takes $\mathcal{O}(|\mathcal{L}| \log |\mathcal{L}|)$ time, since the sorting of the indexes in $\mathcal{L}$ dominates the overall cost.

6.1.3 Computing the Second Longest Match

Now we show how we can compute eMS, extending the algorithm presented by Boucher et al. [17] while preserving the same space-bound.

We can apply verbatim the algorithm of [17] to compute the $\mathsf{eMS}[i].\mathsf{pos}$ and $\mathsf{eMS}[i].\mathsf{len}$ while we extend the algorithm to include the computation of $\mathsf{eMS}[i].\mathsf{slen}$. The following lemma shows how to find the second longest match using the LCP array.

Lemma 108. *Given a text* $T \in \Sigma_{\$}^{n}$*, a pattern* $P \in \Sigma^{m}$*, and the eMS array computed for* P *with respect to* T*, let* $P[i, i + \mathit{eMS}[i].\mathit{len}) = T[\mathit{eMS}[i].\mathit{pos}, \mathit{eMS}[i].\mathit{pos} + \mathit{eMS}[i].\mathit{len})$ *and*

$q = ISA[eMS[i].pos]$. *Then, for all* $0 \leq q < n$, $eMS[i].slen = \max\{LCP[q], LCP[q+1]\}$, *where* $LCP[n] = 0$.

Proof. Let us consider the set $\mathcal{T} = \{T_1 < T_2 < \ldots < T_n\}$ of the lexicographically sorted rotations of T. Then, for all $i \in [1, m]$, at least one rotation of T starting with the second longest match $P[i, i + \mathsf{eMS}[i].\mathsf{slen} - 1]$ must be adjacent to $T_q = T[\mathsf{eMS}[i].\mathsf{pos}, n]$ in $\mathcal{T}$. Hence, assuming $q \neq 1$ and $q \neq n$, $\mathsf{eMS}[i].\mathsf{slen}$ is either the LCP value between T_{q-1} and T_q or between T_q and T_{q+1}, that are respectively $\mathsf{LCP}[q]$ and $\mathsf{LCP}[q+1]$. Note that if $q = n$ only $\mathsf{LCP}[n]$ is available, that is $\mathsf{eMS}[i].\mathsf{slen}$ must be $\mathsf{LCP}[n]$. □

However, notice that storing the LCP would require $\mathcal{O}(n)$ words of space. What we do with our implementation is, in fact, that we store $\mathcal{O}(r)$ samples of the LCP array, in correspondence to the boundaries of the BWT-runs. With the next lemma, we show how the remaining LCP values needed to compute the eMS array can be retrieved either by extending the previous longest match, or via an LCE query.

Lemma 109. *Given a text* $T \in \Sigma_{\$}^{n}$, *let* LCP, SA *and* ISA *be respectively the longest common prefix array, suffix array, and inverse suffix array of* T. *Let us moreover denote with* i, j *the pair of integers such that* $q - 1 = LF[i]$ *and* $q = LF[j]$, *for all* $1 < q \leq n$. *If* $bwt[i] \neq bwt[j]$, *then* $LCP[q] = 0$, *otherwise* $LCP[q] = LCE(SA[i], SA[j]) + 1$.

Proof. Let T_q be the q-th rotation in lexicographic order. Note that if $T_q = \$$, then $\mathsf{LCP}[q] = \mathsf{LCP}[q+1] = 0$. For all $1 \leq q < n$, if $T_{q-1} = au$ and $T_q = bv$ for some $a < b \in \Sigma$ and some strings $u, v \in \Sigma^*$, then $\mathsf{LCP}[q] = 0$. On the other hand, if $T_{q-1} = au$ and $T_q = av\$$, then $\mathsf{LCP}[q] = 1 + lcp(u\$, v\$)$. The thesis follows by observing that the suffixes $u\$$ and $v\$$ respectively correspond to T_i and T_j. □

To support LCE queries, we use the Straight Line Programs defined by Gagie et al. [48] of size g. In Figure 6.1, we show a graphical interpretation of Lemma 109. In this example, the longest match found so far (the green bar) is the prefix of the rotation with index q, that we know it can be extended by one position on the left (the orange square). Instead of storing $\mathsf{LCP}[\mathsf{LF}(w)]$, we can compute through an LCE between $\mathsf{SA}[q]$ and $\mathsf{SA}[q_p]$. Note that the latter is stored since it is in correspondence of the boundary of its BWT-run. Analogously, at the following step, since the

BWT in correspondence to the second longest match pointed by $\mathsf{LF}(q_p)$ agrees with $\mathsf{bwt}[\mathsf{LF}(q)]$ (blue squares), it follows that $\mathsf{LCP}[\mathsf{LF}^2(q)] = \mathsf{LCP}[\mathsf{LF}(q)] + 1$.

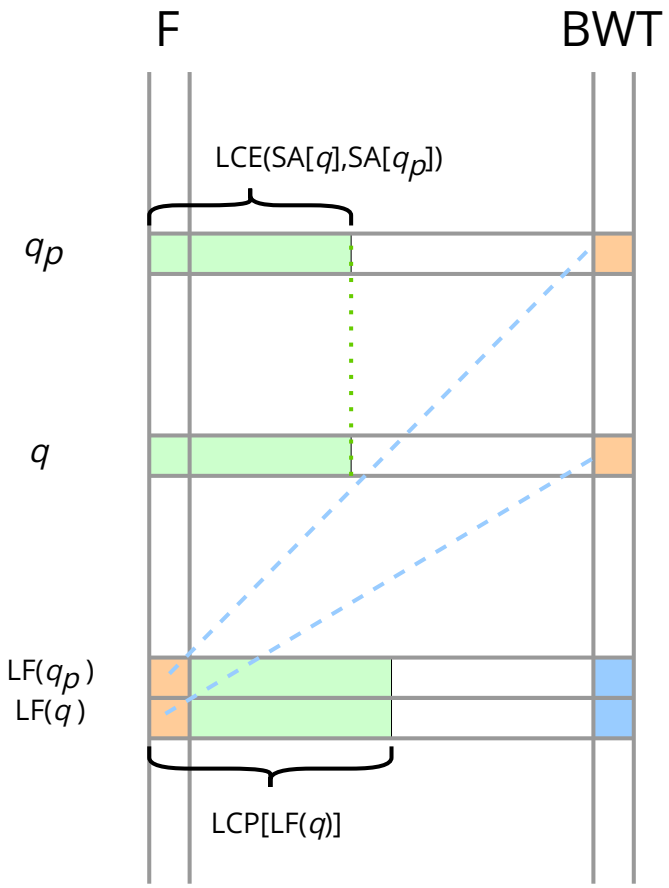


FIGURE 6.1: Application of Lemma 109 to compute $\mathsf{LCP}[\mathsf{LF}(q)]$ by extending the result of the last LCE query.

We have designed an algorithm that uses the properties above mentioned, which performances are described in the next theorem. Correctness and details of the proof are delegated to the original work [57].

Theorem 110. *Given a text $T \in \Sigma_{\$}^*$, we can build a data structure taking $\mathcal{O}(r+g)$ space that allows to compute the set MUMs between any pattern $P \in \Sigma^m$ and T in $\mathcal{O}(m \cdot (t_{LF} + t_{LCE} + t_{pred}))$ time, where t_{LF}, t_{LCE}, and t_{pred} are respectively the time complexities to support one LF, one LCE, and one predecessor query.*

We want to point out that while the BWT is central to our procedure, the $\mathcal{O}(g)$ words that we store are only needed for the LCE queries. Thus, our algorithm can be *plugged* with any data structure supporting LCE queries, with possible trade-off in terms of time or space.

6.1.4 Experimental Results

We implemented our algorithm for computing MUMs and measured its performances on real biological datasets. We performed the experiments on a desktop computer equipped with 3.4 GHz Intel Core i7-6700 CPU, 8 MiB L3 cache. and 16 GiB of DDR4 main memory. The machine had no other significant CPU tasks running, and only a single thread of execution was used. The OS was Linux (Ubuntu 16.04, 64bit) running kernel 4.4.0. All programs were compiled using `gcc` version 8.1.0 with `-O3 -DNDEBUG -funroll-loops -msse4.2` options. We recorded the runtime and memory usage using the wall clock time, CPU time, and maximum resident set size from `/usr/bin/time`.

Setup We compare our method (MUM-PHINDER) with MUMmer [96] (`mummer`). We tested two versions of `mummer`, v3.27 [76] (`mummer3`) and v4.0 [96] (`mummer4`). We executed `mummer` with the `-mum` flag to compute MUMs that are unique in both the text and the pattern, `-l 1` to report all MUMs of length at least 1, and `-n` to match only A,C,G,and T letters. We setup MUM-PHINDER to produce the same output as `mummer`. We did not consider algorithms that do not produce an index for the text that can be queried with different patterns without reconstructing the index, e.g. the algorithm described in Mäkinen et al. [87, Section 11.1.2]. The experiments that exceeded 16 GB of memory were omitted from further consideration.

No. seqs	n (MB)	n/r
1	59	1.92
2	118	3.79
4	236	7.47
8	473	14.78
16	946	29.19
32	1892	57.63
64	3784	113.49
128	7568	222.23
256	15 136	424.93
512	30 272	771.53

(A) Collections of chromosome 19.

No. seqs	n (MB)	n/r
1562	46	459.57
3125	93	515.42
6250	186	576.47
12 500	372	622.92
25 000	744	704.73
50 000	1490	848.29
100 000	2983	1060.07
200 000	5965	1146.24
300 000	8947	1218.82

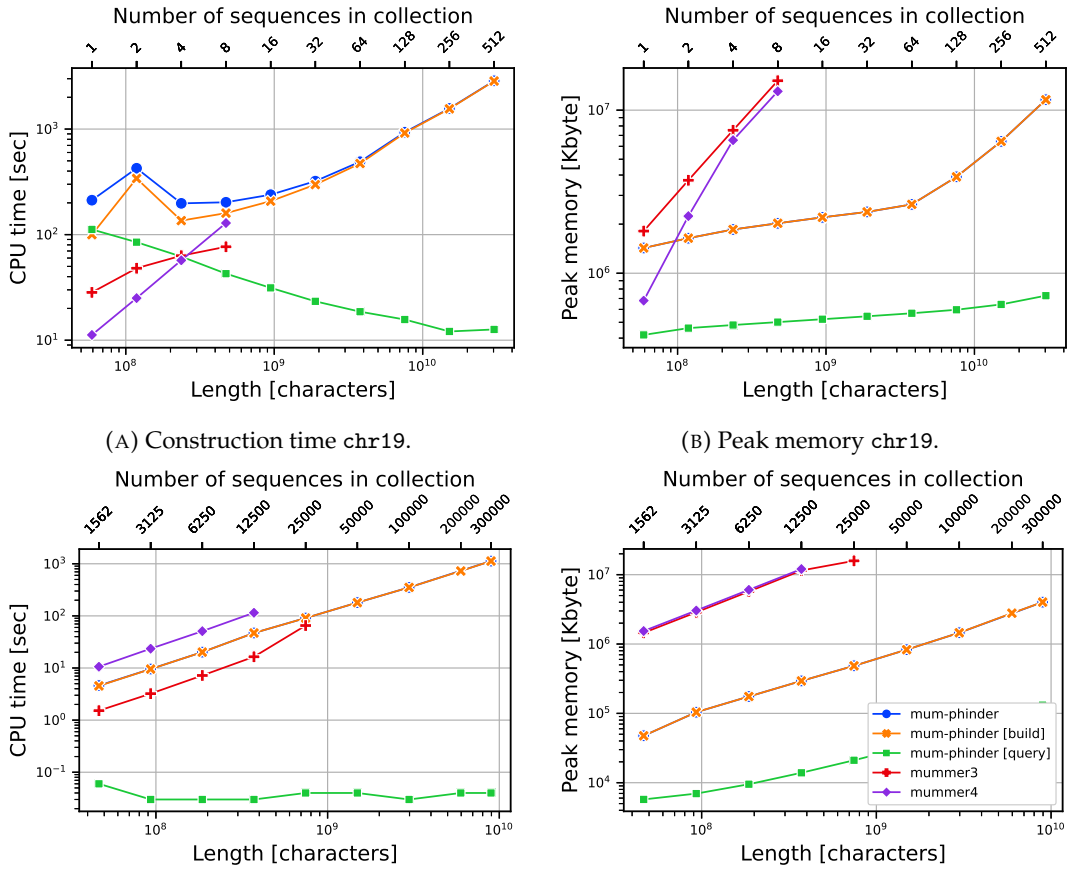
(B) Collections of SARS-CoV2 genomes.

TABLE 6.2: Dataset used in the experiments with MUM-PHINDER.

Datasets For querying the datasets, we used the first haplotype of chromosome 19 of the sample NA21144 from the 1000 Genomes Project, and the genome with accession number MZ477765 from EBI's COVID data portal [58]. For each collection of datasets of the human chromosome 19 (`chr19`) dataset in Table 6.2a and for the SARSCoV2 (`sars-cov2`) dataset in Table 6.2b, we report the number of sequences (No. seqs), the length n in Megabytes (MB), and the ratio n/r, where r is the number of runs of the BWT for each number of sequences in a collection.

Results In Figure 6.2 we show the construction and query time and space for MUM-PHINDER and `mummer`. Since `mummer` is not able to decouple the construction of the suffix tree from the query, for our method we report the sum of the running times for construction and query, and the maximum resident set size of the two steps. We observe that on `chr19` `mummer3` is up to 9 times faster than MUM-PHINDER, while using up to 8 times more memory, while `mummer4` is up to 19 times faster than MUM-PHINDER, while using up to 7 times more memory. However both `mummer3` and `mummer4` cannot process more than 8 haplotypes of `chr19` due to memory limitations. MUM-PHINDER was able to build the index and query in 48 minutes for 512 haplotypes of `chr19` while using less than 11.5 GB of RAM. On `sars-cov2`, `mummer3` is up to 6.5 times faster than MUM-PHINDER, while using up to 24 times more memory, while `mummer4` is up to 1.2 times slower than MUM-PHINDER, while using up to 25 times more memory. `mummer3` was not able to process more than 25,000 genomes while `mummer4` were not able to query mote than 12,500 genomes of `sars-cov2` due to memory limitations.

In Figure 6.2 we also show the construction time and space for MUM-PHINDER. In particular, we compare MUM-PHINDER with `mummer3` and `mummer4`. For MUM-PHINDER we report a breakdown of the construction (build) and query time and space. Note that for MUM-PHINDER, we consider as time the sum of construction and query time, while for memory we consider the maximum between construction and query memory. We observe that the construction time grows with the number of sequences in the dataset, however the query time decreases while increasing the number of sequences in the index with a 9x speedup when moving from 1 to 512 haplotypes of `chr19`. From our experiments, the more expensive routine in terms of time is related



(A) Construction time `chr19`.

(B) Peak memory `chr19`.

(C) Construction time `sars-cov2`.

(D) Peak memory `sars-cov2`.

FIGURE 6.2: Human chromosome 19 and SARS-CoV2 genomes dataset construction CPU time and peak memory usage.

to the number LCE queries performed, which are not executed in case of extendable matches. Even with limited resources, from the ratio n/r in Table 6.2 we can observe how this grows with n, remarking the efficiency of the Burrows-Wheeler Transform in a pangenomic context.

6.2 Burrows-Wheeler Transform for Collections of Sampled Strings

In bioinformatics, most of the problems require a substantial amount of space and time resources, mostly due to the abundance of data. The advent of new generation sequencing technology has indeed increased the quality of assembled genomes, as well the size of the collected reads for assembling them. These reads usually cover the genome multiple times, requiring greater order of space than just the reference

sequence. Up to this moment, there is not a standard procedure to store and collect biological reads. A first approach to the problem has been investigated by Badkobeh et al. [4] with the LZ-77 factorization.

In this section, we focus on the problem of studying the Burrows-Wheeler Transform for sampled strings from a common reference. Even though this study is at an early stage on a simplified instance of the problem, some preliminary results suggest the efficiency of the BWT in this context.

6.2.1 Description of the Problem

Let $T \in \Sigma^n$. Given two parameter $d, m > 0$ such that $m \geq 2d$, let us consider the multiset $\mathcal{S}$ of samples from T defined as

$$\mathcal{S} = \bigcup_{i=1}^{\lfloor \frac{n-m}{d} \rfloor + 1} \{T[i, \min\{i+m-1, n\}]\}.$$

Note that this is the same setting as the one proposed by Badkobeh et al. [4]. We rise the following question.

Question 111. *Can we represent the samples in $\mathcal{S}$ in terms of n, $r_{\$}(T)$, m, or d?*

A trivial bound is the size of the multiset $\mathcal{S}$, that is the sum of the lengths of all samples. Since we expect n much greater than m, we have $|\mathcal{S}| = \Theta(\frac{n \cdot m}{d})$.

Cenzato and Liptak [25] have analyzed different variants of the Burrows-Wheeler Transform for collections of strings. Most of these variants consist in concatenating in a unique string S all sequences from the collection, with separator letters between two consecutive samples. Based on their results, we have opted to build S by using the colexicographical order of the collections [34], that is

$$S = \prod_{i=1}^{\lfloor \frac{n-m}{d} \rfloor + 1} (T_i\$),$$

where $T_i \in \mathcal{S}$ is the ith sample in $\mathcal{S}$ taken in colexicographical order, that is $T_i \leq_{\mathsf{colex}} T_{i+1}$, or equivalently $T_i^R \leq T_{i+1}^R$, for all $i \in [1, \lfloor \frac{n-m}{d} \rfloor]$.

From now on, we refer with $\mathcal{M}$ and $\mathcal{M}_{\mathcal{S}}$ to the matrix of the lexicographically sorted suffixes of $T\$$ and S respectively. Moreover, we denote by UP the set of the

shortest prefixes of the rotations in $\mathcal{M}$ that occur only once in T, taken in the same order. If we assume that T is primitive, the array UP contains exactly n distinct prefixes.

Under certain conditions, the BWT of S is strictly greater than the one for $T\$$.

Lemma 112. *Given a text $T \in \Sigma^n$, let LCP be the LCP-array for $T\$$, and $\ell = \max LCP + 1$. If $\ell > d$ and $m \geq 2\ell$, then $r(S) > r(T\$)$.*

Proof. If $\ell = 1$ then the proof is trivial, since we would have $r(S) > n = r(T\$)$, so let us suppose that $\ell > 0$.

Let $p_i \in \mathsf{UP}$ be the ith prefix from UP. One can notice that $|p_1| = \mathsf{LCP}[1] + 1$, $|p_{n+1}| = \mathsf{LCP}[n+1] + 1$, and $|p_i| = \max\{\mathsf{LCP}[i], \mathsf{LCP}[i+1]\} + 1$ for each $i \in [2, n]$. Let $a_i \in \Sigma \cup \{\$\}$ such that $a_i = \mathsf{bwt}(T\$)[i]$, for all $i \in [1, n+1]$. Indeed, $a_i p_i$ occurs in T, for all $i \in [1, n+1]$, and it is unique by hypothesis on p_i. We can prove that, for each $i \in [1, n]$, $a_i p_i$ has at least an occurrence in S. In fact, the factors of T that do not occur in S are either of the form (i) $T[j, j+M-1]$ with $M > m$, for all $j \in [1, n]$, or (ii) $w[j', j'+m']$ with $j' \mod d \not\equiv 0$ and $m - (j' \mod d) < m' \leq m$. Since (i) $|a_i p_j| \leq \ell + 1 \leq m < M$, and (ii) $|a_j p_j| \leq \ell + 1 \leq \ell + (\ell - d) = 2\ell - d \leq m - d < m'$. In other words, $a_i p_i$ has to occur as factor of some $T_i \in \mathcal{S}$. It follows that to the ith rotation in $\mathcal{M}$ we can associate at least one rotation in $\mathcal{M}_{\mathcal{S}}$ starting with the same prefix p_i and ending with the same letter a_i. Then, $\mathsf{bwt}(T\$)$ is a subsequence of $\mathsf{bwt}(S)$ and the thesis follows. □

The result is not surprising by itself, but it helps us to understand the structure of the BWT of S. Let (x_i, y_i) be the pairs of indexes such that the range $[x_i, y_i]$ of rotations in $\mathcal{M}_{\mathcal{S}}$ starting with p_i, where $p_i \in \mathsf{UP}$. From the proof of Lemma 112, we can deduce the following result.

Lemma 113. *Let T be a text, and let m, ℓ, and d verify the conditions of Lemma 112. Then, either $bwt(S)[x_i, y_i] = a_i^{y_i - x_i + 1}$ or $bwt(S)[x_i, y_i] = a_i^{y_i - x_i}\$$.*

Proof. Suppose that $p_i \in \mathsf{UP}$ is not prefix of any sample in $\mathcal{S}$. Since $a_i p_i$ is unique in T for all $i \in [1, n]$, it follows that the BWT in correspondence the rotations starting with p_i in $\mathcal{M}_{\mathcal{S}}$ consists of a runs of a_i's.

On the other hand, suppose that p_i is prefix of the sample $T_i \in \mathcal{S}$. Since the factor $\$p_i$ occurs only once in S, there must be only one occurrence of a \$ sign in

$\mathsf{bwt}(S)[x_i, y_i]$. Note that the rotations in the range $[x_i, y_i]$ are sorted according to the distance to the next \$, which is maximal when p_i is prefix of a factor (at most as the size m of the samples), and therefore $\mathsf{bwt}(S)[y_i] = \$$. □

6.2.2 Preliminary Observations and Future Developments

In general, for a random generated sequence we expect that most of the values of the values of the LCP array to be approximately $\log(n)$. Indeed, even if m does not surpass all values from the LCP, we still have that $\mathsf{rle}(\mathsf{bwt}(S)[x_i, y_i]) = \mathcal{O}(1)$, for all i that verify the conditions of Lemma 113, which we expect to be the case for the vast majority of rotations. Experimentally, this seems the case also for biological sequences, since genomes appear random but usually present long repetitions within themselves. Moreover, for each prefix $p \in \mathsf{UP}$ longer than m, we expect this can increase the number of BWT-runs at most by a $\mathcal{O}(p - m)$ additive factor.

We tested this method on a few DNA sequences of hundreds of thousands of base pairs from the Pizza&Chili corpus [109], with different settings for the parameters m (50, 100, 200, 400, and 500) and d (2, 4, 8, 16, 32, 64, 100), always keeping in mind the constraint $m \geq 2d$. In our experience, under this setting the size m is not relevant, probably due to the property just mentioned, for which we expect an average length of factors from UP of length $\log_\sigma n \approx 10 < m$. On the other hand, the parameter d, which rules over the number of samples considered, has a greater impact, but in all the cases we never went over 3.5 times the size of the original BWT. Even though we do not have run enough experiments to prove our thesis, it is indeed a path to investigate: in fact, with respect to the approach of Badkobeh et al. [4], the order in which the samples are collected does not affects the BWT.

We believe that with a further theoretical background and more experimental results, we can show that any collection of samples (of any length, from any position) can be bounded through the BWT in a space proportional to the size of the reference. Moreover, we expect to obtain better result by using the *optimal BWT* by Bentley et al. [10], that is the BWT for collection of strings ordered in such a way to minimize the number of BWT-runs, and for which an implementation for computing it has been recently made available [24].

Chapter 7

String Attractors of Finite Words

The notion of *string attractor* has been recently introduced by Kempa and Prezza [66], and it represented an outbreak in the Data Compression field.

Definition 114. *A* string attractor *of a word $w \in \Sigma^n$ is a set of γ positions $\Gamma = \{p_1, \ldots, p_\gamma\}$ such that every factor $w[i, j]$ has an occurrence $w[i', j'] = w[i, j]$ with $p_k \in [i', j']$, for some $k \in [1, \gamma]$.*

In other words, a set Γ is a string attractor of $w \in \Sigma^n$ if we can find an occurrence of every distinct factor in w passing through a position in Γ. We say that an occurrence $w[i, j]$ *crosses* a position $p \in \Gamma$ if $p \in [i, j]$, and symmetrically that p is *crossed by* $w[i, j]$.

Example 115. *Consider the word w built over the alphabet $\Sigma = \{$a,b,c,d,r$\}$:*

$$w = \underline{abra}\underline{c}\underline{a}\underline{d}abra$$

The set $\Gamma^* = \{1, 2, 3, 5, 7\}$ (the underlined positions) is a string attractor for w. In fact, let us consider the factors that do not cross any position in Γ. One can check that these factors are listed in the set $\{a, b, r, ab, br, ra, abr, bra, abra\}$, and all of them have another occurrence in the w crossing a position $p_k \in \Gamma^*$. Note that Γ^* is a minimum string attractor: in fact, $|\Gamma^*| = |\Sigma|$, and to cover all letters we need at least $|\Gamma| \geq |alph(w)| = |\Sigma|$. Note that there could be more string attractors of minimum size: in the case of w, also $\Gamma'^* = \{1, 2, 5, 7, 10\}$ is a string attractor of w, and it is also minimum.

For a given word w, we denote by γ^* the size of a smallest string attractor.

In their work, Kempa and Prezza proved that a big family of compressor schemes can be seen as heuristic techniques for solving the same problem: finding a string

attractor of a word. To mention a few of them: the size z of the *Lempel-Ziv factorization* [81], the measure b^* of the smallest *bidirectional macro scheme* [124], the measures g^* of the smallest *grammar* [70] and g^*_{rl} of the smallest *run-length grammar* [102], the *BWT-runs* $r_\$$ [21], and so on. The results on the above mentioned measures are summarized in the following theorem [66].

Theorem 116. *For any word $w \in \Sigma^n$, it holds that $\gamma^* \leq \min\{z, b^*, g^*, g^*_{rl}, r_\$\}$.*

Moreover, for each measure $X \in \{z, b^*, g^*, g^*_{rl}, r_\$\}$, there exists a word such that $\gamma^* = o(X)$.

Christiansen et al. [31] showed how to build for a word $w \in \Sigma^n$ a self index of size $\mathcal{O}(\gamma \log \frac{n}{\gamma})$. Even though the problem of finding γ^* is, in general, NP-complete [66], combinatorial properties of words can be used to induce information on the string attractor or the size γ^* [90, 112, 119], while new heuristics based on the reduction to the MAX-SAT problem seems to return interesting approximation [6].

Indeed, as observed in the Example 115, the size of the alphabet used is a natural lower bound. More precisely, since each position of a string attractor for a word w is crossed by at most k distinct factors of length k, the following relationship between γ^* and the measure $\delta = \max_{k \in [1,n]}\{\frac{p_w(k)}{k}\}$ can be derived [66, 31].

Theorem 117. *For any word w, it holds that $\delta \leq \gamma^*$.*

Kociumaka et al. [72] showed that analogous bounds to those by Christiansen et al. can be obtained in function of δ, which can be asymptotically smaller than γ^* [73]. Nonetheless, as it will be clear from the results of this thesis, there are properties of string attractors that can be associated with unique properties of words.

In Section 7.1, we present a quality study on the measure γ^*. In particular, we show how the size of a string attractor can be bounded when combinatorial operations are applied. The results from this section have appeared in [91, 90].

Next, in Section 7.2, we introduce two new notions related to the distribution of the positions of string attractors: the *leftmost measure* and the *span measure*. All these three measures are related among them, and similarly influenced by the factor complexity. These notions will be used in the next chapters to get novel characterizations of words. The study of these measures appeared in [112].

7.1 Combinatorial Study of the Measure γ^*

In this section, we explore the notion of String Attractor from a combinatorial perspective, and observe how the measure γ^* is perturbed by classical combinatorial operations. Part of the results in this chapter appeared in a conference paper at the 20th edition of the Italian Conference on Theoretical Computer Science [91], and later extended for a journal version [90].

This first result shows that, unlike the measure r, the string attractor is not affected when one reads a word backwards.

Proposition 118. *Let w be a word and let w^R denote its reverse. Then, $\gamma^*(w) = \gamma^*(w^R)$.*

Proof. Let $\Gamma = \{p_1, p_2, \ldots, p_\gamma \mid 1 \leq p_i \leq |w|, 1 \leq i \leq \gamma\}$ be a string attractor for w and consider the corresponding positions $\Gamma^R = \{n - p_\gamma + 1, n - p_{\gamma-1} + 1, \ldots, n - p_1 + 1\}$ in w^R. Let v be any factor of w^R. Then its reverse v^R is a factor of w. Since Γ is a string attractor for w, then there exists a position $p_i \in \Gamma$ that intercepts an occurrence of v^R. Therefore an occurrence of v is intercepted by $n - p_i + 1 \in \Gamma^R$, i.e. Γ^R is a string attractor for w^R. In particular if Γ is a smallest string attractor for w, then Γ^R is a smallest string attractor for w^R, since otherwise we could find a smaller string attractor for w, in contradiction with the hypothesis of minimality. □

If we consider two words $u, v \in \Sigma^*$, with the following proposition we give an upper bound on the size γ^* of the concatenation, expressed in terms of the size of the string attractors of u and v.

Proposition 119. *Let u and v two words, then $\gamma^*(uv) \leq \gamma^*(u) + \gamma^*(v) + 1$.*

Proof. Let $\Gamma^*(u)$ and $\Gamma^*(v)$ be smallest string attractors for u and v, respectively. Then $\Gamma^*(u) \cup \{p + |u| \mid p \in \Gamma^*(v)\}$ covers all the factors of u and v but might not cover some of new factors that appear across the concatenation point of u and v. In this case it is sufficient to add the last position of the prefix u (or the first position of the suffix v) to have a string attractor for uv. □

The bound from Proposition 119 is tight, as showed in the next example.

Example 120. *Let $u = \underline{\mathtt{b}}\mathtt{aa}\underline{\mathtt{a}}\mathtt{ba}$ and $v = \mathtt{c}\underline{\mathtt{d}}\mathtt{cc}\underline{\mathtt{c}}\mathtt{d}$ be two words in which the positions of the respective smallest string attractors are underlined. If we consider the concatenation $uv =$*

baaabacdcccd, the set underlined positions represent one of the smallest string attractors for uv, as one can verify, having cardinality 5.

However, when the concatenation involves multiple copies of the word itself, a stricter upper bound can be deduced.

Proposition 121. *Let w a word over the alphabet Σ. Then $\gamma^*(w^n) \leq \gamma^*(w) + 1$. Moreover, $\gamma^*(w^n) = \gamma^*(w^2)$ for any $n \geq 2$.*

Proof. Let $\Gamma^*(w)$ be a smallest string attractor for w. The positions of $\Gamma^*(w)$ cover in w^n all the factors of w, and therefore the only factors of w^n that might not be covered are those crossing two consecutive occurrences of w. Therefore it is sufficient to add the last position of the first (or any) occurrence of w in order to cover all these factors.

To prove that $\gamma^*(w^n) = \gamma^*(w^2)$ for any $n \geq 2$, we show that for any smallest string attractor for $\Gamma^*(w^n)$ we can deduce a string attractor of at most the same size for w^2 and the other way around. Given a smallest string attractor $\Gamma^*(w^n)$ with $n > 2$, we can create the set $\Delta = \{1 + (p-1) \mod \ell \mid p \in \Gamma^*(w^n)\}$, where $\ell = |w|$ (note that $|\Delta| \leq \gamma^*(w^n)$). Consider the set $\Gamma = \Delta \setminus \{p_1\} \cup \{p_1 + \ell\}$, where $p_1 = \min \Delta$, i.e., Γ contains the positions of Δ with its first position moved to the second occurrence of w. We now show that Γ is a string attractor for w^2. All the factors u that do not have occurrences overlapping two consecutive w's are covered in w^n by some position $p' \in \Gamma^*(w^n)$, i.e. $p' \in [i, j]$ where $w^n[i, j] = u$. Since ℓ is a period of w^n, it is easy to see that $w[1 + (i-1) \mod \ell, 1 + (j-1) \mod \ell)] = u$, hence either u crosses $1 + (p' - 1) \mod \ell$ in the first occurrence of w or, if $1 + (p' - 1) \mod \ell = p_1$, u crosses $(p_1 + \ell)$. For all the factors $u = w[i, \ell] \cdot w[1, j]$ that overlap two consecutive w's, if $i \leq p_\gamma$ or $j \geq p_1$ (where $p_\gamma = \max \Delta$), then u crosses p_γ or $(p_1 + \ell)$ respectively, so let us assume $i > p_\gamma$ and $j < p_1$. By construction of Γ we can deduce that u occurs also in $w^n[i', j']$ with $(i' \mod \ell) \neq (i \mod \ell)$ and $(j' \mod \ell) \neq (j \mod \ell)$ which crosses a position in $\Gamma^*(w^n)$. Given the periodicity of w^n, it is easy to see that either u occurs in w or u occurs again overlapping two consecutive w's. In both cases u crosses one of the positions in Γ.

Let now $\Gamma^*(w^2)$ be a smallest string attractor and consider the set $\Gamma' = \Gamma^*(w^2)$ if all of its positions lie within the second occurrence of w, otherwise $\Gamma' = \{p + \ell \mid p \in \Gamma^*(w^2)\}$. Since w^n has period ℓ, it is easy to see that the positions in Γ' point to the

same letters of $\Gamma^*(w^2)$, possibly moved in the following occurrence of w. Therefore, all the factors of w^n that are also factors of w^2 are already covered. The remaining factors are of the type $u = w[i,\ell] \cdot w^k \cdot w[1,j]$, with $k \in [1, n-2]$, $1 \leq i, j \leq \ell$. In this case, there is also an occurrence of u starting at the first or at the second w. Since all the positions of Γ' lie within the second or possibly within the third occurrence of w, u has to cross a position in Γ', and the thesis follows. □

Analogously to Proposition 119, the bound from Proposition 121 is tight.

Example 122. *Let us consider the word* $u = \texttt{abbaab}$. *It is easy to check that the only smallest string attractors for* u *are* $\Gamma_1 = \{2,4\}$ *and* $\Gamma_2 = \{3,5\}$. *In order to find a smallest string attractor for* $u^2 = \texttt{abbaababbaab}$, *we remark that neither* Γ_1 *nor* Γ_2 *(neither any string attractor obtained from them by moving some position from the first to the second occurrence of* u*) cover all the new factors that appear after the concatenation. A way to get a smallest string attractor for* u^2 *is to add to* Γ_1 *or* Γ_2, *the position corresponding either to the end of the first occurrence of* u *or the beginning of the second occurrence. For instance,* $\Gamma = \{2,4,6\}$ *is a smallest string attractor for* u^2.

Nonetheless, the size $\gamma^*(u^n)$ can be equal to $\gamma^*(u)$, although different positions for the string attractor may have to be taken.

Example 123. *Let* $u = \texttt{a}\underline{\texttt{ba}}\texttt{b}\underline{\texttt{c}}\texttt{bc}$ *be a word whose smallest string attractor is* $\{2,3,5\}$ *(the underlined letters). Then* $u^2 = \texttt{ab}\underline{\texttt{a}}\texttt{bc}\underline{\texttt{bc}}\texttt{ababcbc}$ *has a string attractor* $\{3,6,7\}$ *of cardinality* 3. *Remark that* $\{2,3,5\}$ *is not a string attractor for* u^2.

Although γ^* is sensitive to the concatenation operation, the following theorem shows that it is not a monotone measure, in the sense that there exist words $w = uv$ such that $\gamma^*(u) > \gamma^*(w)$. This answers to a problem posed by [74] in a preliminary version of a later work ([73]).

Theorem 124. *The measure* γ^* *is not monotone.*

Proof. We show the statement by showing an example where monotonicity does not hold. For each $n > 0$, let us consider $w = \texttt{a}\underline{\texttt{b}}\texttt{b}\underline{\texttt{b}}\texttt{a}^n\underline{\texttt{a}}\texttt{b}$. In this case $\gamma^*(w) = 3$ since by exhaustive search $\{2,4,n+5\}$ is a smallest string attractor for w. On the other hand it is easy to verify that $\{4,n+5\}$ is a string attractor for $w\texttt{b} = \texttt{abb}\underline{\texttt{b}}\texttt{a}^n\underline{\texttt{a}}\texttt{bb}$, then $\gamma^*(w\texttt{b}) = 2$. □

As described in the next proposition, not only the measure γ^* is not monotone, but the difference $\gamma^*(u) - \gamma^*(u^n)$ could become arbitrarily large.

Proposition 125. *For each $t > 0$, there exists an alphabet Σ_t and a word $w_t \in \Sigma_t^*$, such that $\gamma^*(w_t) - \gamma^*(w_t^n) > t$, for each $n \geq 2$.*

Proof. Let us consider $m = t + 3$. We can define the string $w_t = v_1 v_2 v_3 v_4 v_5$ over the alphabet $\Sigma_t = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\} \cup \{\$_1, \$_2, \ldots, \$_{2m-1}\}$ such that

$$
\begin{aligned}
v_1 &= \mathtt{c}^{m-2}\mathtt{d}^{m-1}\underline{\$_1},\\
v_2 &= \mathtt{a}^{m-1}\underline{\mathtt{a}}\mathtt{b}^{m-1}\underline{\mathtt{b}}\mathtt{c}^{m-1}\underline{\mathtt{c}}\mathtt{d}^{m-1}\underline{\mathtt{d}},\\
v_3 &= \prod_{k=2}^{m-1} \underline{\$_k}\mathtt{a}^{m-k}\mathtt{b}^{m-1}\underline{\mathtt{c}}\mathtt{c}^{k-1},\\
v_4 &= \prod_{k=1}^{m-1} \underline{\$_{m-1+k}}\mathtt{b}^{m-k}\underline{\mathtt{c}}\mathtt{c}^{m-2}\mathtt{d}^{k},\\
v_5 &= \underline{\$_{2m-1}}\mathtt{a}^{m-1}\mathtt{b}^{m-1}\underline{\mathtt{c}}.
\end{aligned}
$$

The positions in a smallest string attractor for w_t are underlined. To prove it, note that the factors $\mathtt{a}^{m-1}\mathtt{b}^{m-1}\mathtt{c}$, $\mathtt{a}^{m-j}\mathtt{b}^{m-1}\mathtt{c}^j$, with $2 \leq j \leq m-1$, and $\mathtt{b}^{m-j}\mathtt{c}^{m-1}\mathtt{d}^j$, with $1 \leq j \leq m-1$, appear once in v_5, v_3 and v_4, respectively. So, $\gamma^*(w_t) = 4m + 1$. If we consider w_t^2, all the above mentioned factors occur in $v_5 v_1$, and they are crossed by the rightmost occurrence of $\mathtt{c}$ in v_5. Moreover, every other factor that appears in v_3, v_4 or v_5 which does not contain any $\$_i$ letter, has another occurrence that it is crossed either by the rightmost position in v_5 or by one of the underlined positions in v_2. Therefore, in order to obtain a smallest string attractor for w_t^2 we can remove the rightmost positions from each block of v_3 and v_4. Then, $\gamma^*(w_t^2) = 4m + 1 - (2m - 3) = 2m + 4$, and therefore $\gamma^*(w_t) - \gamma^*(w_t^2) = 4m + 1 - (2m + 4) = 2m - 3 = 2t + 3 > t$. Finally, by Proposition 121 we have $\gamma^*(w_t^2) = \gamma^*(w_t^n)$ and the thesis hold. □

A direct consequence of the Proposition 125 is that γ^* of the concatenation can be lower than the same measure on the words separately.

Corollary 126. *There exist an alphabet Σ and words $u, v \in \Sigma^*$ such that $\gamma^*(uv) < \min\{\gamma^*(u), \gamma^*(v)\}$.*

Proof. For some $t > 0$, let Σ_t and $w_t \in \Sigma_t$ be the alphabet and the word from the proof of Proposition 125 respectively. The thesis follows by considering $\Sigma = \Sigma_t$, $u = w_t$, and $v = w_t[1, m]$, where $m \in [|v_1v_2v_3| + 1, |w_t|]$. □

If we look carefully at the definition of w_t in the proof of Proposition 125, we can deduce a similar results on the rotations of in $\mathcal{R}(w_t)$.

Proposition 127. *For each $t > 0$, there exists an alphabet Σ_t and a word $w_t = uv \in \Sigma_t^*$, such that $\gamma^*(uv) - \gamma^*(vu) > t$.*

Proof. The thesis follows by considering the word $w_t = v_1v_2v_3v_4v_5$ defined in the proof of Proposition 125 and its conjugate $w'_t = v_2v_3v_4v_5v_1$. The thesis follows by analogous arguments. □

7.2 New String Attractor based Measures

Previously in this chapter, we have shown the quality of γ^* as a measure of repetitiveness. However, the size on its own is often not sufficient to understand the structure or properties of a word. On the other hand, from the distribution of the positions it is possible to infer other properties of words.

In this section, we introduce two new notions related to the string attractors of a word. As it will be shown later in the thesis, these measure can be used to characterize different notions of words. The first measure is the notion of span of a word, which gives the minimum distance between the last and the first positions of any string attractor.

Definition 128. *Let w be a finite word and let $\mathcal{G}$ be the set of all string attractors for w. The* (string attractor) span *of w is the value $\mathsf{span}(w) = \min_{\Gamma \in \mathcal{G}}(\max \Gamma - \min \Gamma)$. We will also abusively say that the quantity $(\max \Gamma - \min \Gamma)$ is the span of the string attractor Γ.*

Example 129. *Let us consider the word $w = \overline{\mathtt{ab}}\mathtt{c}\overline{\mathtt{c}}\underline{\mathtt{ab}}\mathtt{c}$ on the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. One can see that the sets $\Gamma_1 = \{4, 5, 6\}$ (underlined positions) and $\Gamma_2 = \{1, 2, 4\}$ (overlined positions) are two suitable string attractors for w. These string attractors are of minimal size as $|\Gamma_1| = |\Gamma_2| = |\Sigma|$ but they have different spans. Moreover, since all of the positions of Γ_1 are consecutive, it is of minimal span and therefore $\mathsf{span}(w) = 6 - 4 = 2$.*

The span can be used to derive an upper-bound on the number of distinct factors, as shown below.

Proposition 130. *Let w be a finite word over Σ. Then $|F(w) \cap \Sigma^n| \leq n + \mathsf{span}(w)$ for all $1 \leq n \leq |w|$.*

Proof. Let Γ be a string attractor of minimal span and write $\delta = \min \Gamma$ and $\delta' = \max \Gamma$. Then, the interval $\Gamma' = [\delta, \delta']$ contains Γ and is a string attractor for w. Since every factor has an occurrence crossing a position in Γ', it is possible to find all length-n factors of w by considering a window of length n, sliding from position $\max\{\delta - n + 1, 1\}$ to position $\min\{\delta', |w| - n + 1\}$. One can see that this interval is of size at most $\delta' - (\delta - n + 1) + 1 = n + \mathsf{span}(w)$. This ends the proof. □

Note that this bound is tight. In fact, for the family of standard Sturmian words, there exists a string attractor containing two consecutive positions, i.e., with span 1. The structure of these string attractors is described in the following theorem. The proof is delegated to the original work [90].

Theorem 131. *For each $w \in Stand$ with $|w| \geq 2$, let η be the length of the longest palindromic proper prefix of $\pi(w)$, the set $\Gamma_1 = \{\eta + 1, \eta + 2\}$ or the set $\Gamma_2 = \{|w| - \eta - 3, |w| - \eta - 2\}$ is a smallest string attractor for w.*

In addition, we may compare strings attractors of a given word according to their rightmost positions. More specifically we will want the string attractor having the smallest such position. This gives the notion of *leftmost string attractors* as defined below.

Definition 132. *Let w be a finite word and let $\mathcal{G}$ be the set of all string attractors for w. We define a* leftmost string attractor *for w as a string attractor $\Gamma \in \mathcal{G}$ such that, for all $\Gamma' \in \mathcal{G}$, we have $\max \Gamma \leq \max \Gamma'$. The* (string attractor) leftmost measure *of w is then $\mathsf{lm}(w) = \max \Gamma$ where Γ is a leftmost string attractor.*

Example 133. *We resume Example 129. First, we have $4 = \max \Gamma_2 < \max \Gamma_1 = 6$. Second, the set $\Delta = \{1, 2, 3\}$ is not a string attractor for w. Therefore $\mathsf{lm}(w) = 4$.*

Examples 129 and 133 show that for the finite word $w = abccabc$, these two measures can be obtained by distinct string attractors. In fact, in this case, it is not

possible to find a leftmost string attractor having minimal span since $\{2, 3, 4\}$ is not a string attractor.

Similarly to what we did for the span, we can use the leftmost measure to obtain an upper-bound on the number of distinct factors.

Proposition 134. *Let w be a finite word over Σ. Then $|F(w) \cap \Sigma^n| \leq lm(w)$ for all $1 \leq n \leq |w|$.*

Proof. The proof follows the same lines as that of Proposition 130 by considering a leftmost string attractor Γ, and $\Gamma' = [1, \max \Gamma]$ instead. □

In Examples 129 and 133, we can see that $\gamma*(w) - 1 \leq \mathsf{span}(w) \leq \mathsf{lm}(w) - 1$. This is a general result as shown below.

Proposition 135. *Let w be an finite word. Then, $\gamma^*(w) - 1 \leq span(w) \leq lm(w) - 1$.*

Proof. Let Γ be a string attractor of w with minimal span. It contains at most $\max \Gamma - \min \Gamma + 1 = \mathsf{span}(w) + 1$ elements, therefore $\gamma^*(w) \leq \mathsf{span}(w) + 1$.

Let Γ' be a leftmost string attractor of w. Its span is at most $\max \Gamma' - 1 = \mathsf{lm}(w) - 1$, therefore $\mathsf{span}(w) \leq \mathsf{lm}(w) - 1$. □

The following proposition shows how the size of the smallest string attractor, the span and the leftmost measure of a word yield bounds on the corresponding measures for its image under a morphism.

Proposition 136. *Let $\varphi : \Sigma^* \mapsto \Sigma'^*$ be a morphism. Then, there exists $K \geq 1$ which depends only on φ such that, for every $w \in \Sigma^+$, the following hold:*

1. $\gamma^*(\varphi(w)) \leq 2\gamma^*(w) + K$;
2. $span(\varphi(w)) \leq K \cdot span(w)$;
3. $lm(\varphi(w)) \leq K \cdot lm(w)$.

Proof. Consider any string attractor Γ for w. We show how one can build a valid string attractor for $\varphi(w)$ starting from Γ in two steps.

Step 1. First, we consider the factors of the images of letters, i.e., the elements of $F_\varphi = \bigcup_{a \in \Sigma} F(\varphi(a))$. Recall that for every letter $a \in \Sigma$ there is at least one position $j \in \Gamma$ such that $w[j] = a$ and let us denote j_a such a position for the letter a. Then, for

every $a \in \Sigma$ we can choose any minimum string attractor Γ_a of $\varphi(a)$ and overlay it on the occurrence of $\varphi(w[j_a])$ to cover the factors of $\varphi(a)$. In other words, every element of F_φ has an occurrence in w crossing at least a position in

$$\mathcal{T}^\varphi = \bigcup_{a\in\Sigma} \{|\varphi(w[1, j_a - 1])| + \delta \colon \delta \in \Gamma_a\}.$$

Step 2. Let us now consider the other factors of w, i.e. the elements of $\mathcal{F} = F(\varphi(w)) \setminus F_\varphi$. To cover these factors, we define two sets of positions. Let $\mathcal{T}^f = \{|\varphi(w[1, j-1])| + 1 \mid j \in \Gamma\}$ be the set of positions corresponding to the first letter of $\varphi(w_j)$, where j is a position in Γ. Analogously, we define the set $\mathcal{T}^l = \{|\varphi(w[1, j])| \mid j \in \Gamma\}$ as the set of positions corresponding to the last letter of some $\varphi(w_j)$ with $j \in \Gamma$.

Let $u \in \mathcal{F}$ and let v be a factor of w of minimal length such that u is a factor of $\varphi(v)$. Observe that, by definition of $\mathcal{F}$, v is of length at least 2. As v is a factor of w, it has an occurrence crossing some position $j \in \Gamma$. By minimality of v, we know that u has an occurrence crossing either the first position of $\varphi(w_j)$ or the last position of $\varphi(w_j)$ (or both). Therefore, u crosses a position in $\mathcal{T}^f$ or $\mathcal{T}^l$.

As a consequence of the previous two steps, $\Gamma' = \mathcal{T}^\varphi \cup \mathcal{T}^f \cup \mathcal{T}^l$ is a string attractor for $\varphi(w)$. Recall that this construction can be done starting from any string attractor Γ of w, giving different corresponding string attractors Γ'. Now let us denote $\ell = \max_{a\in\Sigma} |\varphi(a)|$, that is ℓ is the longest image of a letter in Σ. To obtain the three claimed inequalities, we now consider different string attractors Γ of w.

1. If Γ is such that $|\Gamma| = \gamma^*(w)$, then

$$\gamma^*(\varphi(w)) \le |\Gamma'| \le |\mathcal{T}^f| + |\mathcal{T}^l| + |\mathcal{T}^\varphi| \le 2\gamma^*(w) + \sum_{a\in\Sigma} \gamma^*(\varphi(a)).$$

2. If Γ is such that $\max\Gamma - \min\Gamma = \mathsf{span}(w)$, then by construction we have $\min\Gamma' = |\varphi(w[1, \min\Gamma - 1])| + 1 \in \mathcal{T}^f$ and $\max\Gamma' = |\varphi(w[1, \max\Gamma])| \in \mathcal{T}^l$, and therefore

$$\begin{aligned}\mathsf{span}(\varphi(w)) &\le |\varphi(w[1, \max\Gamma])| - (|\varphi(w[1, \min\Gamma - 1])| + 1)\\ &= |\varphi(w[\min\Gamma, \max\Gamma])| - 1\\ &\le \ell \cdot (\mathsf{span}(w) + 1).\end{aligned}$$

3. If Γ is such that $\max \Gamma = \mathsf{lm}(w)$, then

$$\mathsf{lm}(\varphi(w)) \leq \max \Gamma' = |\varphi(w[1, \max \Gamma])| \leq \ell \cdot \mathsf{lm}(w).$$

To end the proof, we can choose $K = \ell \cdot |\Sigma|$ and the conclusion will follow for all three cases. Note that K is independent from w. $\square$

Chapter 8

Circular Variant of String Attractors

Circular notions of words have been considered in different fields. Indeed, *circular words* (or *necklaces*) have been object of several studies in Combinatorics on Words. A very well-known family of circular words, the de Bruijn sequences, is at the basis of very recent indexed data structures [108, 7, 9]. Also in nature we can find examples of circular words, like the genomes of some families of viruses.

In Section 8.1 we present the notion of *circular string attractor*, that is a set of positions covering all circular factors of a word. We further show some of its combinatorial properties, and compare the size γ_c^* of a smallest circular string attractor with the measures γ^* and r. Most of the results from this section appeared in [90].

In Section 8.2, we show an application of circular string attractors, that is a new characterization for Standard Sturmian words.

Finally, in Section 8.3, we present the first algorithm to check whether a set is a circular string attractor of a string.

8.1 Circular String Attractor

Let us introduce a variant for the notion of string attractor, which further takes into account factors that occur at the boundaries of words.

Definition 137. *Let $w \in \Sigma^*$ and $n = |w|$. A set of γ_c positions $\Gamma_c = \{j_1, j_2, \ldots j_{\gamma_c}\} \subseteq [1, n]$ is a* circular string attractor *of a word w if each circular factor of w has at least a*

circular occurrence that crosses a position of Γ_c. Moreover, we denote with γ_c^* the size of the smallest circular string attractor.

Example 138. *Let $w = a\underline{b}bb\underline{c}aaa\underline{a}caaa$ be a word over the alphabet $\Sigma = \{a, b, c\}$. The set $\Gamma = \{2, 5, 8\}$ is a string attractor for w, since it covers any of its factors, but it is not a circular string attractor since the circular factor (in blue) caaaa escapes from it. On the other hand, the set $\Gamma_c = \{1, 4, 9\}$ is a circular string attractor for $w = \underline{a}bb\underline{b}caaa\underline{c}aaa$ but it is not a string attractor. In fact, the factor aaa (in blue), fully contained in w, is covered only if we consider its circular occurrence.*

The definition above can be easily extended to circular words. For simplicity of exposition though, we consider the notion on linear words. Despite the similar definitions, Example 138 shows that the sets Γ and Γ_c can be independent, even though in this case the sizes γ^* and γ_c^* are the same. On the other hand, we know by Proposition 127 that γ^* could arbitrarily increase when a conjugate of a word is considered, while the behaviour of γ_c^* is different, as it can be easily inferred from the definition.

Proposition 139. *Let $w = uv$ be a finite word, for some $u, v \in \Sigma^*$. A set Γ_c is a circular string attractor of w if and only if the set $\bigcup_{i \in \Gamma_c} \{i - |u| \mid i > |u|\} \cup \bigcup_{j \in \Gamma_c} \{i + |v| \mid i > |u|\}$ is a circular string attractor of $w' = vu$.*

Proof. By hypothesis, there exist $u, v \in \Sigma^*$ such that $w = uv$ and $w' = vu$. Let Γ_c^* be a smallest circular string attractor for w. Then, we can partition Γ_c^* into two sets $\Gamma_u = \{i \in \Gamma_c^* \mid i \leq |u|\}$ and $\Gamma_v = \{i \in \Gamma_c^* \mid i > |u|\}$, i.e., the sets containing the positions of Γ_c^* falling respectively in u and in v. One can notice that the set $\Gamma'^* = \{i - |u| \mid i \in \Gamma_v\} \cup \{i + |v| \mid i \in \Gamma_u\}$ is a circular string attractor for $w' = vu$, since it covers all the same circular factors of w. Since this procedure is symmetric for w', the thesis holds. □

A direct consequence is the following equivalence between the measures γ_c of two rotations.

Corollary 140. *Let $w \in \Sigma^*$ be a finite word, and let $w' \in \mathcal{R}(w)$ be a rotation of w. Then, $\gamma_c^*(w) = \gamma_c^*(w')$.*

In the following proposition we derive an upper bound on γ_c^* by considering the minimum size of the smallest string attractor of the words in the conjugacy class.

Proposition 141. *Let w a word in Σ^*. Then, $\gamma_c^*(w) \leq \gamma^*(v) + 1$ for each v conjugate of w.*

Proof. Let $\Gamma^*(v)$ be a string attractor having minimum size for v. If $\Gamma^*(v)$ is also a circular string attractor then we are done. Otherwise, there must be some *strictly* circular factor that overlaps two consecutive occurrences of w which is not fully contained in w and that is not covered by any position $j \in \Gamma^*(v)$. Note that these factors must cross the end and the beginning of two occurrences of w. Hence, the set $\Gamma^*(v) \cup \{1\}$ and $\Gamma^*(v) \cup \{n\}$ are both circular string attractors of w. □

In the previous chapter, Propositions 125 and 127 suggested that we can take advantage of circular factors to capture more repetitions within the text. The following proposition shows that the size of a smallest circular string attractor of a word can be arbitrarily lower than the size of a smallest string attractor of the word itself.

Proposition 142. *For each $t > 0$, there exists an alphabet Σ_t and a word $w_t \in \Sigma_t^*$, such that $\gamma^*(w_t) - \gamma_c^*(w_t) > t$.*

Proof. Consider the word $w_t = v_1 v_2 v_3 v_4 v_5$, as defined in the proof of Proposition 125. From the same proof, we know that $\gamma^*(w_t) = 4m + 1$, where $m = t + 3$. Let us consider the rotation $w_t' = v_2 v_3 v_4 v_5 v_1$. From the proof of Proposition 127, we know that we can build a string attractor Γ', of size $\gamma^*(w_t') = 2m + 4$ for w_t' containing the position $|w_t'|$. It follows that Γ' is also a circular string attractor of w_t', and by Proposition 141 $\gamma_c^*(w_t) = \gamma_c^*(w_t') \leq |\Gamma'| = 2m + 4$. Finally, $\gamma^*(w_t) - \gamma_c^*(w_t) \geq 4m + 1 - (2m + 4) = 2t + 3 > t$. □

On the other hand, one can verify that the notion of circular string attractor of a word is strictly related to the notion of string attractor for a power of the word itself. In the next lemma, we show how a string attractor of a power induces a circular string attractor, and vice versa.

Lemma 143. *Let $w \in \Sigma^n$ and $\Gamma_c = \{j_1, j_2, \ldots j_{\gamma_c}\} \subseteq [1, n]$ a set of positions in w. Then, Γ_c is a circular string attractor of w if and only if $\Gamma' = \{j_k + n \mid j_k \in \Gamma_c\}$ is a string attractor of w^3.*

Proof. For the first implication, note that the positions in Γ' correspond to the positions of Γ_c in the central occurrence of w. Moreover, it is easy to check that any k-length factor u of w^3 is a circular factor of w, with $k = 1, 2 \ldots, n$. In fact, if u lies entirely in an occurrence of w, then it is a circular factor of w as well. Otherwise, u overlaps two consecutive occurrences of w, which is still a circular factor of w. Moreover, as we have picked the positions in Γ', every factor u has an occurrence crossing a position $j_i \in \Gamma'$, either this occurrence is fully contained in w or lies across two consecutive w's. For any factor v such that $|v| > n$, if $v = w[i, n] \cdot w \cdot w[1, j]$, with $1 \leq i, j \leq n$, then v has an occurrence which crosses all the positions in Γ', since it contains the central occurrence of w in w^3. Otherwise, $v = w[i, n] \cdot w[1, j]$, with $|w[i, n]| + |w[1, j]| > n$. Since v appears twice in w^3 and w^3 has period n, one of the two occurrences of v has to cross a position $j_i \in \Gamma'$ in the central occurrence of w.

For the other direction, note that since, Γ' is a string attractor for w^3 and, as mentioned above, any k-length factor u in w^3 is a circular factor of w with $k = 1, 2, \ldots, n$, using the previous reasoning we can easily verify that Γ_c is a circular string attractor for w. □

Example 144. *Let* $w = \texttt{aabaacaabc}$ *be a word on the alphabet* $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{c}\}$. *A circular string attractor of the word* w *is* $\Gamma_c = \{3, 5, 10\}$. *In fact, every factor fully contained in* w *has an occurrence crossing a position* $j \in \Gamma' = \{13, 15, 20\}$, *except for* $u = \texttt{caab}$ *and its prefixes* $\texttt{ca}$ *and* $\texttt{caa}$. *However,* u *also occurs between two consecutive occurrences of* w:

$$w^3 = \texttt{aabaacaabc} \cdot \texttt{aabaacaabc} \cdot \texttt{aabaacaabc}.$$

As we can see, u *and its prefixes cross the position* $20 \in \Gamma'$. *For what concerns any other factor that lies in* $w \cdot w$ *but does not appear in* w, *note that the position* 20 *covers them all too.*

We can then derive the following relationship between the measures γ^* and γ_c^* for powers of words.

Corollary 145. *Let* $w \in \Sigma^*$ *be a finite word. Then,* $\gamma_c^*(w) = \gamma^*(w^n)$, *for all integers* $n > 0$.

Proof. Direct consequence from Proposition 127 and Lemma 143. □

In other words, a circular string attractor can not capture more information than a normal string attractor when a power of a word is considered. On the other hand, the size γ_c^* is always preserved when a power of a word is considered, as well the structure of the circular string attractor.

Lemma 146. *Let $w \in \Sigma^*$ be a finite word and $\Gamma_c = \{p_1, p_2, \ldots, p_{\gamma_c}\} \subseteq [1, |w|]$. Then, Γ_c is a circular string attractor of w if and only if, for all $n > 1$ and for all $d \in [0, n-1]^{\gamma_c}$, the set $\Gamma' = \bigcup_{i \in [1,\gamma_c]} \{p_i + d_i|w|\}$ is a circular string attractor of w^n.*

Proof. For the first direction, note that since the word w^n has period n, all circular factors of length at most $|w|$ which cross the position p_i for some $i \in [1, \gamma_c]$ have another occurrence crossing the position $p_i + d_i|w|$. For all the circular factors of w^n of length greater than $|w|$, for the same argument we can find (at least) n occurrences through w^n at distance $|w|$, covering the whole word, and therefore these factors can be moved to the any occurrence of w where a position from Γ' falls.

For the second implication, by hypothesis the set Γ_c is a circular string attractor for w^n. Symmetrically to the previous direction, and by Lemma 143, the factors of w must be fully covered by Γ_c, and the thesis follows. □

As mentioned before, the size $r_\$$ of a word is a natural upper bound for the measure γ^*. Analogously, we show that the *Toehold Lemma* by [110] can be easily extended to circular factors of words, and therefore that we can retrieve a circular string attractor of size r from the BWT of a word.

Theorem 147. *For every word $w \in \Sigma^*$, it holds that $\gamma_c^*(w) \leq r(w)$.*

Proof. We show the proof analogously to the proof of Theorem 3.9 by [66], that is we find a circular string attractor for w of size exactly r. Let Γ_r be the set of positions in correspondence to the letters at the beginning of each BWT-run. To prove that each circular factor u of w has an occurrence (i, j) crossing at least a position in Γ_r, consider the indexes $I, J \in [1, n]$ such that $L[I]$ and $L[J]$ correspond to the occurrences of the letters $w[i]$ and $w[j]$ in w. It follows that $J = \mathsf{LF}^{n-j}[I]$. Moreover, let ℓ be the length of the circular factor u, and let $[s_1, e_1], [s_2, e_2], \ldots, [s_\ell, e_\ell]$ be the sequence of BWT-runs visited in the column L while applying the *LF-mapping* from $w[j]$ to $w[i]$, i.e. $L[l_t, r_t]$ contains $L[\mathsf{LF}^{t-1}[J]]$ for any $t \in [1, \ell]$. Consider the value $\Delta = \min\{\mathsf{LF}^t[J] - l_t | t \in$

$[1, \ell]\}$, that is the minimum distance between a visited letter of the BWT with the first element of the BWT-run where it lies.

Recall that $\Gamma_r = \{p_k | w[p_k]$ corresponds to $L[l_k]$ for any $k \in [1, r]\}$. Hence, if $\Delta = \mathsf{LF}^m[J] - l_m = 0$ for some $m \in [1, \ell]$, then $\mathsf{LF}^m[J] = l_m$ and the occurrence (i, j) of u has a letter which crosses a position in Γ_r. Otherwise, assume $\Delta = \mathsf{LF}^m[J] - l_m > 0$. It is easy to see that if two positions p_1, p_2 belong to the same BWT-run in L then $L[p_2] = L[p_1] + (p_2 - p_1)$. Thus, if we pick $J' = J - \Delta$, at any step we have $L[\mathsf{LF}^{t-1}[J']] = L[\mathsf{LF}^{t-1}[J]]$ for any $t \in [1, \ell]$, which means that there exists another circular occurrence (i', j') of u, where $J' = \mathsf{LF}^{n-j'}[I]$. Since $w[\mathsf{LF}^m[J']]$ corresponds to $L[l_m]$, the circular occurrence (i', j') contains a position from Γ_r. □

8.2 Characterization of Standard Sturmian Words via Circular String Attractors

In this section, we show an application of circular string attractor in Combinatorics on Words, namely a characterization of words through novel notions related to circular string attractors.

8.2.1 Circular Span

Similarly to the measure of span on linear words, let us introduce the definition of *circular span* of a string attractor.

Definition 148. *Let w be a finite word and let $\mathcal{G}_c(w)$ be the set of all circular string attractors of w. The* circular (string attractor) span *of w is the value*

$$\textit{c-span}(w) = \min_{w' \in \mathcal{R}(x)} \left\{ \min_{\Gamma_c \in \mathcal{G}(w')} (\max \Gamma_c - \min \Gamma_c) \right\},$$

that is the minimum span among the rotations of w.

Example 149. *Let us consider the word $w =$* `abcabababababdab`. *One can see that the set $\Gamma_c = \{3, 12, 13, 14\}$ (underlined positions in w) is a circular string attractor for w. Moreover, since the first and the last position in Γ_c correspond to the unique occurrence of* `c` *and* `d` *respectively, we have that $\mathsf{span}(w) = 14 - 3 = 11$.*

Let us now consider the rotation $w' = \underline{\mathtt{d}}\mathtt{abab}\underline{\mathtt{cab}}\mathtt{ababababab}$, *and the set* $\Gamma'_c = \{1, 6, 7, 8\}$ *(underlined positions in* w'*). Also in this case* Γ'_c *is a circular string attractor. Furthermore,* Γ'_c *is a circular string attractor of minimum span for* w'*, since the first position correspond to the only occurrence of* d*, and the last is at the beginning of the first occurrence of the factor* baba*. By exhaustive research, one can check that* $\textsf{c-span}(w) = \max \Gamma'_c - \min \Gamma'_c = 8 - 1 = 7$.

Analogously to the measure γ^*_c of conjugate words, the definition of the measure c-span directly yields the following result.

Lemma 150. *For every pair of words* $u, v \in \Sigma^*$*, it holds that* $\textsf{c-span}(uv) = \textsf{c-span}(vu)$.

Similarly, we have the same bound for the power of a word.

Lemma 151. *For every word* $w \in \Sigma^*$*, it holds that* $\textsf{c-span}(w) = \textsf{c-span}(w^n)$*, for every* $n > 1$.

Proof. By Lemma 146, any circular string attractor of w is a circular string attractor for w^n, thus $\textsf{c-span}(w) \geq \textsf{c-span}(w^n)$. On the other hand, since $\textsf{c-span}(w) < |w|$, there exists a circular string attractor Γ_c of minimum span for w^n such that all positions falls within an occurrence of $w' = vu$, where $w = uv$ for some $u, v \in \Sigma^*$. By Proposition 139, we can see that the word w'^n, obtained by rotating on the left by $|u|$ letters the word w^n, must have the same minimum span. Always by Lemma 146, the same span holds for w'. Therefore, $\textsf{c-span}(w) = \textsf{c-span}(w') \leq \textsf{c-span}(w'^n) = \textsf{c-span}(w^n)$, and the thesis follows. □

In the following proposition, we give a bound on the circular factor complexity of a word w. Since $p_w(n) \leq c_w(n)$ for all finite $w \in \Sigma^*$, the analogous bound for p_w also holds.

Lemma 152. *Let* w *be a finite word over* Σ*. Then* $|\mathcal{C}(w) \cap \Sigma^n| \leq n + \textsf{c-span}(w)$ *for all* $1 \leq n \leq |w|$.

Proof. Let Γ_c be a circular string attractor of some rotation $w' \in \mathcal{R}(w)$ that minimizes $\textsf{c-span}(w)$. Clearly, the number of circular factors is the same, and therefore we can focus on the word w'.

Let $\ell_c = \min \Gamma_c$, and $r_c = \max \Gamma_c$. Then, the interval $\Gamma'_c = [\ell_c, r_c]$ contains Γ_c, and it is a circular string attractor for w'. Since every circular factor has an occurrence crossing a position in Γ'_c, it is possible to find all length-n circular factors of w' by considering a circular window $(i, (i+n-1) \mod |w'|)$ of length n, sliding either from position $\ell_c - n + 1$, if $\ell_c - n + 1 \geq 1$, or $|w| - (n - \ell_c) + 1$ otherwise, to position r_c. One can see that this interval is of size at $r_c - (\ell_c - n + 1) + 1 = n + \text{c-span}(w)$. This ends the proof. □

8.2.2 Standard Sturmian Words and Bounded Circular Span

By refining a result stated in a previous work ([91]), we show that the notion of circular string attractors can be used to state the main result of this section, providing a new characterization of the conjugacy classes of powers of the standard Sturmian words. To do so, we use the following result, proved by [15], that states that the conjugates of standard Sturmian words are uniquely characterized by the circular factor complexity.

Theorem 153 ([15]). *Let w be a word of length $n \geq 2$. The following statements are equivalent:*

1. *w is a conjugate of a standard Sturmian word;*
2. *for $k = 0, 1, \ldots, n-1$, $c_w(k) = k + 1$;*
3. *$c_w(n-2) = n - 1$ and w is primitive.*

Theorem 154. *Let $w \in \Sigma^*$ be a finite word. The word w is a conjugate of a power of a binary standard Sturmian word if and only if $\text{c-span}(w) = 1$.*

Proof. Let $s' = uv$ be a primitive standard Sturmian word such that $w = (vu)^p$, for some words $u, v \in \{\mathtt{a}, \mathtt{b}\}^*$, and an integer $p \geq 1$. Let us start with the first implication. By combining the Theorem 14 and Theorem 147, we can find a circular string attractor Γ_c of s' by taking the positions of the first or the last occurrence from each run of BWT-runs, according to whether s' ends with $\mathtt{ab}$ or $\mathtt{ba}$ respectively. One can notice that Γ_c consists of two consecutive positions, since it corresponds to the positions from Theorem 131 (details in the proof from [90]). Thus, by Lemmas 146 and 150, it follows that $\text{c-span}(s') = \text{c-span}(uv) = \text{c-span}(vu) = \text{c-span}((vu)^p) = \text{c-span}(w) = 1$.

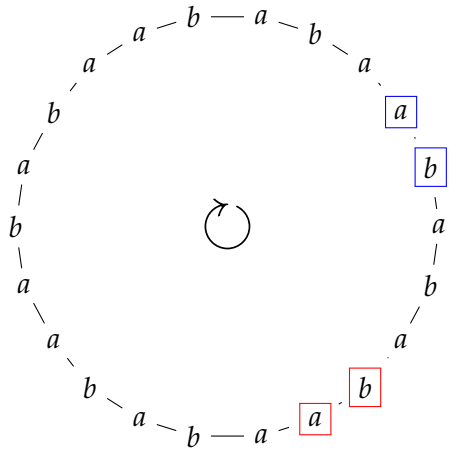


FIGURE 8.1: Circular representation of the standard Sturmian words $aba.ba.ababaababaababa.ab$ and $ababaababaababa.ab.aba.ba$.

For the other direction of the implication, let w' be the primitive word such that $w = (w')^q$, for some integer $q \geq 1$. By Lemma 146, we know that $\text{c-span}(w) = \text{c-span}(w')$. From Lemma 4.1 by [15], the word w' is primitive if and only if $c_w(k) \geq k+1$, for $k \in [1, n-1]$. On the other hand, by Lemma 152, the circular factor complexity of w' can not be more than $k+1$, for all $k \in [1, |w'|]$. Thus, by Theorem 153 w' is a conjugate of a standard Sturmian word and the thesis follows. ☐

Remark 155. *It is known that each conjugacy class of standard Sturmian words exactly contains two standard Sturmian words Xab and Xba, where $X \in PER$, for which we have given the definition in Section 2.1. This means that each word in the conjugacy class has two circular string attractors consisting of 2 consecutive positions, as depicted in the example in Figure 8.1. Both couples of boxed consecutive positions represent a circular string attractor for each word in the correspondent conjugacy class.*

8.3 Checking the Attractor Property in Linear Time

The problem of finding a minimum-size string attractor for a given n-length word w is an NP-Complete problem [66]. From the equivalence of Corollary 145, and with the reduction that will be shown later in Lemma 161, it is possible to deduce that the same problem is as hard as on regular string attractor. On the other hand, in [90], [77], and [119], the size γ^* has been studied for infinite families of words by using combinatorial arguments. Before proving how to construct (circular) string

attractors for infinite families of words, it is good habit to run experiments on some words of the family, to check if the thesis fails for some counterexample.

In this section, we present two novel algorithms for checking if a set Γ is a (circular) string attractor for a word $w \in \Sigma^n$. Throughout the section, we assume that Σ is an integer alphabet of polynomial size with respect to n. Under these conditions, the algorithm here presented operated in $\mathcal{O}(n)$ time using $\mathcal{O}(n \log n)$ bits, where n is the length of w. Even if the bounds are the same as one of the algorithms presented by [68], we use data structures easier to implement. First we show the properties that these data structure verify, then we use these results to prove the correctness of the algorithm proposed.

8.3.1 Checking the Circular Attractor Property in Linear Time

Given a word w of length $n > 0$, let us consider the matrix $\mathcal{M}(w) = \{w_1, \ldots, w_n\}$ of the rotations of w sorted in lexicographical order. For any factor u of w, we denote with $\mathcal{I}_u$ the set multiset of rotations from $\mathcal{M}(w)$ starting with u, taken in lexicographical order. To give to the reader a first look into the idea of the algorithm, let us consider the 6th finite Fibonacci word $f_6 = \texttt{abaababaabaab}$. In Figure 8.2, it is shown the matrix of sorted rotations of f_6, and for each rotation the positions of a circular string attractor, derived by Theorem 147, are underlined. One can verify that, for each circular factor $u \in \mathcal{C}(f_6)$, the rotations having u as prefix are consecutive, and at least one of these prefixes crosses a position of the circular string attractor. Note that this is not a matter of chance, since each occurrence of any circular factor is a prefix of a rotation.

If we denote with w_k the kth rotation in lexicographical order, then for every $u \in Fact(w)$ we can find unique indices $\ell_u \leq r_u$ such that $\mathcal{I}_u = \{w_{\ell_u}, w_{\ell_u+1}, \ldots, w_{r_u}\}$.

Let CA be the *conjugate array* of a word w, defined as:

$$\mathsf{CA}[i] = j \text{ if } w_i = w[j, n]w[1, j-1].$$

a	a	b	a	a	b	a	b	a	a	b	a	b
a	a	b	a	b	a	a	b	a	a	b	a	b
a	a	b	a	b	a	a	b	a	b	a	a	b
a	b	a	a	b	a	a	b	a	b	a	a	b
a	b	a	a	b	a	b	a	a	b	a	a	b
a	b	a	a	b	a	b	a	a	b	a	b	a
a	b	a	b	a	a	b	a	a	b	a	b	a
a	b	a	b	a	a	b	a	b	a	a	b	a
b	a	a	b	a	a	b	a	b	a	a	b	a
b	a	a	b	a	b	a	a	b	a	a	b	a
b	a	a	b	a	b	a	a	b	a	b	a	a
b	a	b	a	a	b	a	a	b	a	b	a	a
b	a	b	a	a	b	a	b	a	a	b	a	a

FIGURE 8.2: Matrix of the sorted rotations for the finite Fibonacci word $f_6 =$ abaababaabaab. The underlined positions correspond to the positions of a circular string attractor $\Gamma_c = \{12, 13\}$. We underline in the other rotations the corresponding position

Analogously, we denote by c-LCP the *circular longest common prefix array*, defined as follows:

$$\text{c-LCP}[i] = \begin{cases} 0 & \text{if } i = 1 \\ |\ell cp(w_{i-1}, w_i)| & \text{otherwise} \end{cases} .$$

The following lemma summarises how to detect the indices ℓ_u and r_u from the c-LCP array.

Lemma 156. *Let $w \in \Sigma^n$, for some $n > 0$. For each $u \in \mathcal{C}(w) \setminus \{\varepsilon\}$, let $1 \leq \ell_u \leq r_u \leq n$ be the indices such that $\mathcal{I}_u = \{w_{\ell_u}, w_{\ell_u+1}, \ldots, w_{r_u}\}$. The following holds:*

1. *$\ell_u = 1$, or c-LCP$[\ell_u] < |u|$;*
2. *$r_u = n$, or c-LCP$[r_u + 1] < |u|$;*
3. *c-LCP$[k] \geq |u|$, for all $k \in [\ell_u + 1, r_u]$.*

Proof. For case *1.*, suppose $\ell_u > 1$. By contradiction, if c-LCP$[\ell_u] \geq u$, then also the rotation w_{ℓ_u+1} has u as prefix, that is a contradiction by hypothesis on ℓ_u.

Case *2.* is treated symmetrically.

Case *3.* follows by observing that all the rotations in $\mathcal{I}_u$ are consecutive and share the same prefix of length $|u|$. □

Given an ordered set $\Gamma_c = \{p_1, p_2, \ldots, p_{\gamma_c}\}$ and a word $w \in \Sigma^n$, let $\mathsf{succ}_c \in \{0, 1, \ldots, n-1\}^n$ be the array of circular distances of each position $i \in [1, n]$ of w to the next position p in Γ_c, that is:

$$\mathsf{succ}_c[i] = \begin{cases} p_1 - i & \text{if } 1 \leq i \leq p_1 \\ p_{j+1} - i & \text{if } p_j < i \leq p_{j+1}, \text{ for all } j \in [1, \gamma_c - 1] \\ (n - i) + p_1 & \text{if } p_{\gamma_c} < i \end{cases} .$$

Example 157. *Let us consider the word* $w = \mathtt{abbabaa}$ *and the set* $\Gamma_c = \{3, 5, 6\}$ *(the underlined positions in* w*). Then,* $\mathsf{succ}_c = \{2, 1, 0, 1, 0, 0, 3\}$.

By Lemma 146, we know that a set Γ_c is a circular string attractor for the word w^n, for any $n > 1$, if and only if $\Gamma'_c = \bigcup_{p \in \Gamma_c} \{(i - 1 \mod |w|) + 1\}$ is a circular string attractor of w. Thus, we can assume that w is primitive, otherwise we can find its root u in linear time and check whether the set Γ'_c obtained as just described is a circular string attractor of u.

Lemma 158. *Let* $w \in \Sigma^n$ *be a primitive word, and let* $\Gamma \subseteq [1, n]$ *be a set of positions in* w*, for some* $n > 0$*. Then,* Γ_c *is a circular string attractor of* w *if and only if, for all* $u \in \mathcal{C}(w)$*, there exists* $i \in [\ell_u, r_u]$ *such that* $\mathsf{succ}_c[\mathsf{CA}[i]] < |u|$.

Proof. For the first implication, if Γ_c is a circular string attractor, then for each $u \in \mathcal{C}(w)$ there exists at least one occurrence that is crossed by a position $p \in \Gamma$. Let $i \in [\ell_u, r_u]$ the index of the rotation starting with such an occurrence of u. Thus, if we project the position from Γ_c in the ith rotation, such a position falls at most at distance $|u|$, and therefore $\mathsf{succ}_c[\mathsf{CA}[i]]$ is at most $|u| - 1$.

The other direction is treated symmetrically. In fact, by contradiction, if for all $i \in [\ell_u, r_u]$ it holds that $\mathsf{succ}_c[\mathsf{CA}[i]] \geq |u|$, then none of the occurrences of u cross a position in Γ_c, and therefore it can not be a circular string attractor, contradiction. □

Thus, in order to check whether a set Γ_c is a circular string attractor, we need to check if, for all $u \in \mathcal{C}(w)$, there exists $i \in [\ell_u, r_u]$ such that $\mathsf{succ}_c[\mathsf{CA}[i]] < |u|$. In Algorithm 1 we describe the procedure designed. We store in a stack S the ranges of lengths of the prefixes encountered and for which we have not found a position in Γ_c crossing an occurrence yet. We then proceed by comparing in order the c-LCP with

the succ_c array. We use Lemma 156 to understand if we are still in the range $[\ell_u, r_u]$ without even knowing u, but just by using the c-LCP array (line 8). Note that this is legit since for all $u, v \in \Sigma^*$ it holds that $[\ell_{uv}, r_{uv}] \subseteq [\ell_u, r_u]$, i.e., we can not leave the range $[\ell_u, r_u]$ before checking if the factor uv is covered within $[\ell_{uv}, r_{uv}]$. This implies that the stack S will contain increasing lengths from bottom to top. The consecutive lengths of the prefixes for which we do not have found yet an occurrence crossing a position in Γ_c are inserted as a pair (s, e) in a stack S.

Algorithm 1: Algorithm for checking if a set Γ_c is a circular string attractor of a word w

```
1  S ← empty stack
2  succ_c ← computeCircSucc(w, Γ)
3  CA ← computeConjugateArray(w)
4  c-LCP ← computeCircularLongestCommonPrefixArray(w)
5  for i ∈ [1, n] do
6      if S is not empty then
7          (s, e) ← S.pop()
8          if c-LCP[i] < e then
9              return false
10         else
11             if c-LCP[i] < succ_c[CA[i]] then
12                 if e = c-LCP[i] then
13                     S.push((s, succ_c[CA[i]]))
14                 else
15                     S.push((s, e))
16                     if c-LCP[i] < succ_c[CA[i]] then
17                         S.push((c-LCP[i] + 1, succ_c[CA[i]]))
18             else
19                 while S is not empty ∧ succ_c[SA[i]] ≤ s do
20                     (s, e) ← S.pop()
21                 if s < succ_c[CA[i]] then
22                     S.push((s, min{e, succ_c[CA[i]]}))
23     else
24         if c-LCP[i] < succ_c[CA[i]] then S.push((c-LCP[i] + 1, succ_c[CA[i]]))
25 if S is empty then
26     return true
27 else
28     return false
```

We can then obtain the following.

Theorem 159. *Given a word $w \in \Sigma^n$ and a set $\Gamma_c \subseteq [1, n]$, Algorithm 1 checks whether or not the set Γ_c is a circular string attractor for w in $O(n)$ time using $O(n \log n)$ bits of space.*

Proof. The conjugate array CA can be computed in linear time (see the work by [16]). The c-LCP array can be computed in linear time from CA with analogous techniques developed by [65], or [64]. Each pair (s, e) in the stack S can be represented in $\mathcal{O}(\log n)$ bits. The main loop (line 5) is executed at most n times. Since we add at most one pair in S at each iteration of the main loop, it holds that the number of elements in the stack is $|S| = \mathcal{O}(n)$, occupying at most $\mathcal{O}(n \log n)$ bits of space. Further, the inner loop in line 19 where we empty the stack S can be executed at most $|S|$ times in total. Thus, Algorithm 1 works in $\mathcal{O}(n)$ time using $\mathcal{O}(n \log n)$ bits of space and the thesis follows. □

Example 160. *In Figure 8.3, we show the steps of Algorithm 1 applied on the 5th finite Fibonacci word $f_5 =$ abaababa with the set $\Gamma_c = \{4, 5\}$. The matrix of the sorted rotations is shown to give to the reader a graphical interpretation of the procedure, however recall that we do not use it.*

The stack S is initially empty, so at the iteration $i = 1$ we start to fill it with the lengths of the prefixes of the first rotation in lexicographical order that need to cross a position from Γ_c. Since $succ_c[CA[1]] = 4$, this means that all prefixes of the first rotation with length greater than 4 cross a position from Γ_c. On the other hand, we can not say anything yet on the prefixes from length 1 to 4, which are a, aa, aab, *and* aaba. *In fact, since $c\text{-}LCP[1] = 0 < 4 = succ_c[CA[1]]$, we push in the stack S the pair $(1, 4)$ (line 24), as displayed in Subfigure 8.3a.*

On the iteration $i = 2$, we extract from the top of the stack the ranges of lengths $(1, 4)$, and we compare the maximum (4) with $c\text{-}LCP[2]$, since we want to check if there is another occurrence of all the factors represented in the stack. Since $c\text{-}LCP[2] = 4$, we have another rotation with prefix aaba, *and therefore such a rotation starts by* a, aa, *and* aab *as well. In Subfigure 8.3b, since $succ_c[CA[2]] = 1$, analogously to the previous case all prefixes longer than 1 cross a position from Γ_c, and therefore it is left only* a *to cover, and we insert in the stack the range $(1, 1)$ (line 22).*

Then, at the iteration $i = 3$ (Subfigure 8.3c), the prefix a *still occurs ($c\text{-}LCP[3] = 1$), but it does not cross a position from Γ_c ($succ_c[CA[3]] = 6$), so we extend the range from $(1, 1)$ to*

$(1,6)$ *and insert it in* S *(line 13), that is we keep track of the factors* `ab`, `aba`, `abaa`, `abaab`, *and* `abaaba`.

The algorithm proceeds in Subfigure 8.3d by shrinking the range $(1,6)$ *to* $(1,3)$ *(line 22), and then at the following iteration in Subfigure 8.3e the stack is emptied for the first time, since* $succ_c[CA[5]] = 0$ *and the condition of line 21 is not met.*

We keep following the procedure for the iterations 6, 7, *and* 8, *respectively shown in Subfigures 8.3f, 8.3g, and 8.3h. Since at the end of the main loop the stack is empty, each circular factor crosses at least a position from* Γ_c, *and therefore we return* ***true***.

8.3.2 Novel Algorithm for Checking the Attractor Property

If we consider the same problem on regular string attractors, Kempa et al. [68] presented two algorithms for checking whether a set Γ is a string attractor for a word w: the first working in $O(n)$ time using $O(n \log n)$ bits of space, and the second in $O(n \log n)$ time using $O(n)$ bits of space. However, both algorithms are based on suffix trees and other data structures supporting range-minimum queries, for which the implementation is not trivial in all programming languages.

Here we present a novel algorithm, induced from the strategies developed in Algorithm 1 and the following lemma, taking the same time and space complexities as Algorithm 1.

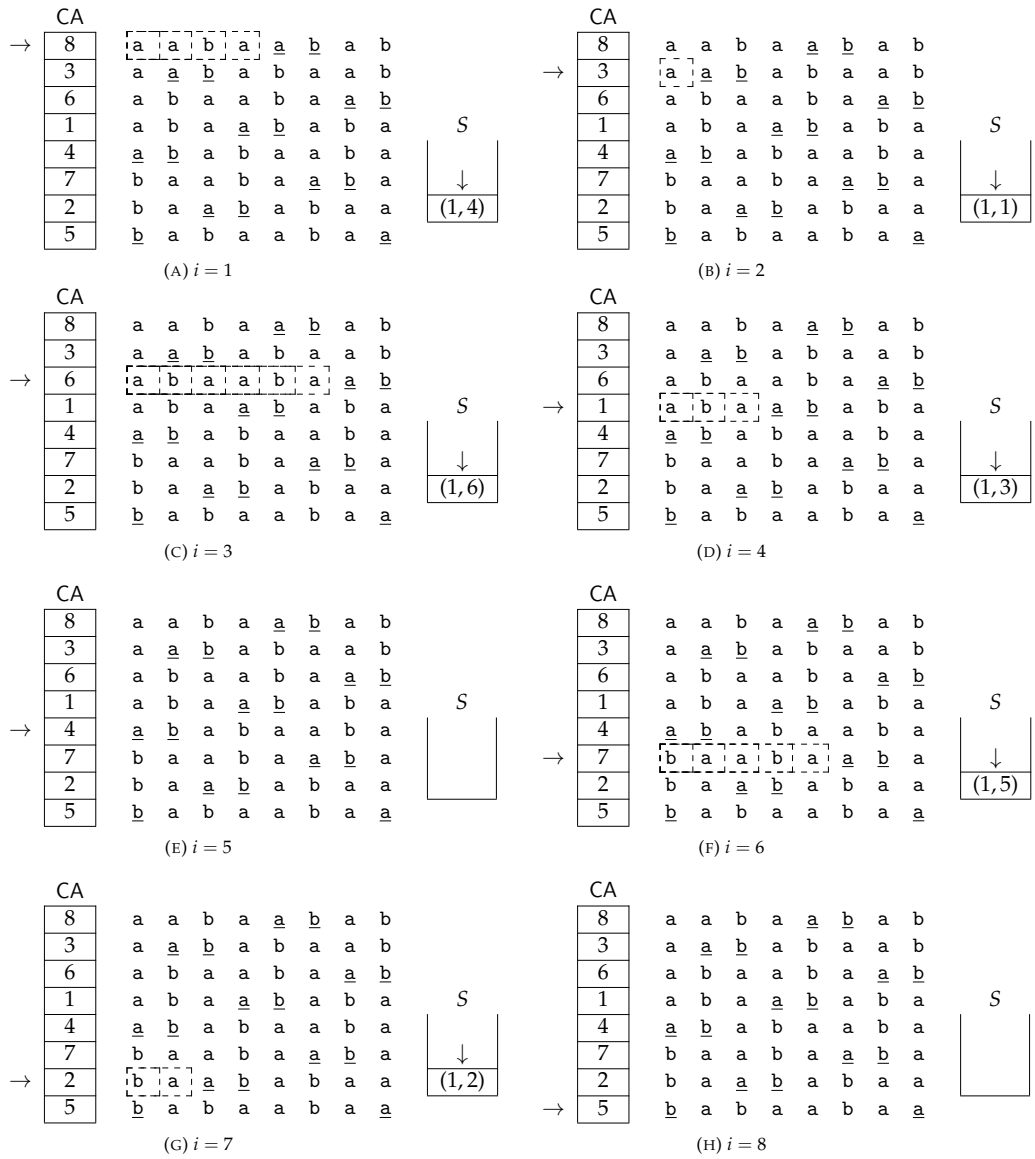
Lemma 161. *Let* $w \in \Sigma^n$ *be a finite word and let* $\$ \notin \Sigma$. *A set* Γ *is a string attractor for* w *if and only if* $\Gamma \cup \{n+1\}$ *is a circular string attractor for* $w\$$.

Proof. Let $\mathcal{C}_\$(w\$)$ denote the set of all circular factors of $w\$$ containing the letter $\$$. One can observe that $\mathcal{C}(w\$) = F(w) \cup \mathcal{C}_\$(w\$)$. Indeed, $F(w) \cap \mathcal{C}_\$(w\$) = \emptyset$, and the position $\{n+1\}$ is crossed by all and only circular factors from $\mathcal{C}_\$(w\$)$. Thus, the factors to cover in w are the same as the circular factors in $\mathcal{C}(w\$) \setminus \mathcal{C}_\$(w\$)$, and since they are located in the same positions in both words the thesis follows. □

Recall that appending a $\$$ smaller than any other letter in Σ implies that $\mathsf{CA} = \mathsf{SA}$ and $\mathsf{c\text{-}LCP} = \mathsf{LCP}$. We can then derive the following theorem.

Theorem 162. *Given a word* $w \in \Sigma^n$ *and a set* $\Gamma \subseteq [1,n]$, *Algorithm 2 checks whether or not the set* Γ *is a string attractor for* w *in* $O(n)$ *time using* $O(n \log n)$ *bits of space.*

$i =$	1	2	3	4	5	6	7	8
$f_5 =$	a	b	a	$\underline{a}$	$\underline{b}$	a	b	a
$\mathsf{succ}_c =$	3	2	1	0	0	6	5	4
c-LCP =	0	4	1	6	3	0	5	2



(A) $i = 1$ (B) $i = 2$ (C) $i = 3$ (D) $i = 4$ (E) $i = 5$ (F) $i = 6$ (G) $i = 7$ (H) $i = 8$

FIGURE 8.3: Running example of Algorithm 1 on the word $f_5 =$ abaababa. The underlined positions in f_5 correspond to the positions of the circular string attractor Γ_c, and the corresponding succ_c and c-LCP arrays are shown right below. Each subfigure shows one of the 8 iterations of the algorithm, and the stack S at the end of the iteration. The dashed boxes surround the prefixes of the current rotation for which we have not found an occurrence crossing a position in Γ_c. The lengths of these prefixes correspond to the ranges in S.

Proof. The correctness of Algorithm 2 is derived from its equivalence in Lemma 161 with Algorithm 1. The suffix array SA, the LCP array, and the succ array can be computed in $\mathcal{O}(n)$ time using $\mathcal{O}(n \log n)$ bits of space. Since Algorithm 2 takes the same time and space as Algorithm 1, the thesis follows. □

Algorithm 2: Algorithm for checking if a set Γ is a string attractor for a word w

```
1  S ← empty stack
2  Γ ← Γ ∪ {n + 1}
3  SA ← computeSuffixArray(w$)
4  succ ← computesucc(w$, Γ)
5  LCP ← computeLongestCommonPrefixArray(w$)
6  for i ∈ [1, n] do
7      if S is not empty then
8          (s, e) ← S.pop()
9          if LCP[i] < e then
10             return false
11         else
12             if LCP[i] < succ[SA[i]] then
13                 if e = LCP[i] then
14                     S.push((s, succ[SA[i]]))
15                 else
16                     S.push((s, e))
17                     if LCP[i] < succ[SA[i]] then
                           S.push((LCP[i] + 1, succ[SA[i]]))
18             else
19                 while S is not empty ∧ succ[SA[i]] ≤ ℓ do
20                     (s, e) ← S.pop()
21                 if s < succ[SA[i]] then
22                     S.push((s, min{e, succ[SA[i]]}))
23     else
24         if LCP[i] < succ[SA[i]] then S.push((LCP[i] + 1, succ[SA[i]]))
25 if S is empty then
26     return true
27 else
28     return false
```

Example 163. *Let us consider the word $f' =$ `ababaaba`, that is a rotation of the word f_5 from Example 160, and let us consider the set of positions $\Gamma = \{7, 8\}$. In Figure 8.4, the iterations of Algorithm 2 with f' and Γ in input are shown. Recall that we compute the*

LCP, $succ$, and SA arrays for the word $f'\$$, and we extend Γ with the position of the $\$$, i.e. $\Gamma = \{7, 8, 9\}$.

As shown in Subfigure 8.4a, at first the stack is empty and the condition of line 24 is not met, since $LCP[1] = succ[SA[1]] = 0$, and therefore nothing is added into S. The same procedure occurs at the following iteration, in Subfigure 8.4b.

Since at the third iteration the stack is still empty, each factor that is prefix of the first two rotations has an occurrence crossing at least a position in Γ. Since this time $1 = LCP[3] < succ[SA[3]] = 2$, we add to the stack in line 24 only the factors that have not occurred yet (i.e. of length at least $LCP[3] + 1$) and that are not crossing a position in Γ (i.e. of length at most $succ[SA[3]]$), and therefore we add the range $(2, 2)$. As shown in Subfigure 8.4c, such a range corresponds to the factor `aa`.

Finally, in Subfigure 8.4d, one can see that the factor `aa` *does not occur as prefix in the following rotations. In fact, at the iteration $i = 4$ Algorithm 2 checks in line 9 whether the factor that we are looking for occurs again as prefix by comparing $LCP[4] = 1$ with the maximum value from the top of the stack. Since the condition is not met, the algorithm returns* ***false****, i.e. the factor* `aa` *is not crossed by any position in Γ, and therefore Γ is not a string attractor.*

$i =$	1	2	3	4	5	6	7	8	9
$f'\$ =$	a	b	a	b	a	a	b	a	$
succ =	6	5	4	3	2	1	0	0	0
LCP =	0	0	1	1	3	3	0	2	2

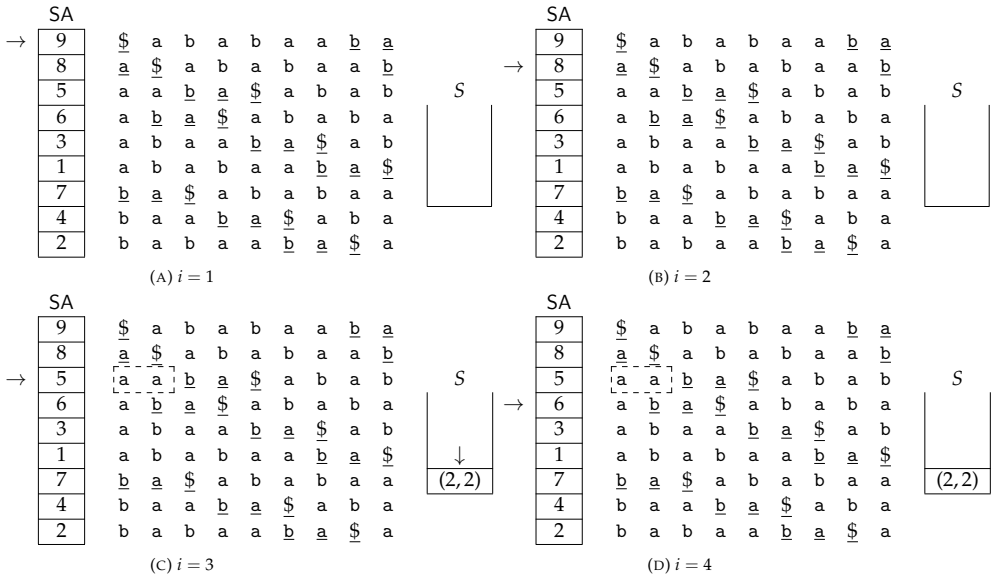


FIGURE 8.4: Running example of Algorithm 2 on the word $f' =$ ababaaba$. The underlined positions in f' correspond to the positions of a set Γ that we want to check whether it is a string attractor for f'. The corresponding succ and LCP arrays are shown right below. Each subfigure shows one of the 4 iterations of the algorithm, and the status of the stack S at the end of the iteration. The dashed boxes surround the prefixes of the current rotation for which we have not found an occurrence crossing a position in Γ. The lengths of these prefixes correspond to the ranges in S.

Chapter 9

String Attractors and Infinite Words

In Chapter 7, we have showed some relationship of the measure γ^* with classical notions of complexity for finite words. Schaeffer and Shallit [119] extended the notion of string attractor on infinite words by defining the *string attractor profile function* $s_{\mathbf{x}}$, which counts for each n the size of the smallest string attractor for the n-length prefix of an infinite word $\mathbf{x}$. In particular, they have considered the very well-known family of *automatic words* (see [2] for a reference), and showed that if the string attractor profile is bounded by a constant, they can list for each prefix a smallest string attractor with the tool `Walnut` [98]. Moreover, they have related the linearly recurrence of words with the function $s_{\mathbf{x}}$ of infinite words. By Theorem 124, in general, the function $s_{\mathbf{x}}$ is not monotone.

In this chapter, we extend the link between two worlds: Data Compression and Combinatorics on Words. We relate new bounds on the string attractor based complexities with respect to classical combinatorial notions of repetitiveness.

In Section 9.1, we further investigate the relationship between string attractor profile function with the factor complexity and the recurrence of words.

In Section 9.2, we introduce two new complexities based on the measures earlier defined in Chapter 7: the *span complexity*, and the *leftmost complexity*. Analogously to the function $s_{\mathbf{x}}$, the two functions return the measure span and lm for each prefix of an infinite word. We show how these function are related to each other, and what properties of words can be deduced from their boundedness.

Finally, in Section 9.3.1 we show how to use the span complexity to obtain a new

characterization for Sturmian words. This result can be further extended, and cover the more general family of *quasi-Sturmian words*, which are the *simplest* aperiodic words after the Sturmian words.

The work of this chapter appeared in [112].

9.1 String Attractor Profile Function, Factor Complexity, and Recurrence

In this section, we explore the growth of the size of the smallest string attractor when considering larger and larger prefixes of an infinite word. Such an idea was first considered in [119].

Definition 164. *Let* $\mathbf{x}$ *be an infinite word. For any* $n \geq 1$*, we denote by* $s_{\mathbf{x}}(n)$ *the size of a smallest string attractor for the prefix of* $\mathbf{x}$ *of length* n*. The function* $s_{\mathbf{x}}$ *is called the* string attractor profile function *of* $\mathbf{x}$*.*

We will show the link between the string attractor profile function and different notions measuring the repetitiveness of factors within infinite sequences of letters. Recall that, for a given infinite word $\mathbf{x}$, the appearance function $A_{\mathbf{x}}$ returns for each $n > 0$ the length of the shortest prefix of $\mathbf{x}$ that contains all n-length distinct factors of $\mathbf{x}$, while the factor complexity function $p_{\mathbf{x}}$ counts for each $n > 0$ the number of distinct factors of $\mathbf{x}$ of length n. We start by establishing a bond between the appearance, factor complexity, and string attractor profile functions. In particular, it extends the result by [66] to infinite words, and shows that upper bounds on $s_{\mathbf{x}}$ induce upper bounds on $p_{\mathbf{x}}$.

Theorem 165. *Let* $\mathbf{x}$ *be an infinite word. For all* $n \geq 1$*, we have* $p_{\mathbf{x}}(n) \leq n \cdot s_{\mathbf{x}}(A_{\mathbf{x}}(n))$*.*

Proof. Let us consider the value $A_{\mathbf{x}}(n)$ representing the length of the smallest prefix of $\mathbf{x}$ containing all the factors of $\mathbf{x}$ of length n. Since the alphabet is finite, the value $A_{\mathbf{x}}(n)$ is finite. By definition $s_{\mathbf{x}}(A_{\mathbf{x}}(n))$ is the size of the smallest string attractor of the prefix of length $A_{\mathbf{x}}(n)$. Therefore, each factor of $\mathbf{x}$ of length n crosses at least one element of the string attractor. Since each element of the string attractor is crossed by at most n distinct factors of $\mathbf{x}$ of length n, one has $p_{\mathbf{x}}(n) \leq n \cdot s_{\mathbf{x}}(A_{\mathbf{x}}(n))$. □

Let $z_{\mathbf{x}}$ denote the *Lempel-Ziv complexity* of $\mathbf{x}$, that is $z_{\mathbf{x}} = z(\mathbf{x}[1, n])$. Using the link between string attractors and LZ77 parsings, we easily obtain an upper bound on $s_{\mathbf{x}}$ as follows.

Proposition 166. *Let* $\mathbf{x}$ *be an infinite word. Then* $s_{\mathbf{x}}(n) = O\left(\frac{n}{\log n}\right)$.

Proof. Using Theorem 116, we have $s_{\mathbf{x}}(n) \leq z_{\mathbf{x}}(n)$. Therefore, to conclude, it suffices to use the following upper bound on $z_{\mathbf{x}}(n)$ from [81]: the number of LZ-phrases for a length-n word on an alphabet of size σ is bounded by $\frac{n}{(1-\epsilon_n)\log_\sigma n}$, where $\epsilon_n = 2\,\frac{1+\log_\sigma(\log_\sigma(\sigma n))}{\log_\sigma n}$. □

It is possible to construct an infinite word $\mathbf{x}$ for which there exists a subsequence of positive integers n_i, for $i \geq 1$, such that $s_{\mathbf{x}}(n_i) = \Theta(\frac{n_i}{\log n_i})$. For instance, such a word $\mathbf{x}$ can be constructed by using a suitable sequence of de Brujin words. However, having information about the values of the string attractor profile function on a subsequence n_i does not allow us to precisely determine its behavior for the remaining values of n. Therefore, we do not know whether the bound of Proposition 166 is tight.

However, if we assume that the appearance function is linear, a better bound on the function $s_{\mathbf{x}}$ can be obtained as stated below.

Theorem 167 ([119])**.** *Let* $\mathbf{x}$ *be an infinite word. If* $A_{\mathbf{x}}(n) = \Theta(n)$*, then* $s_{\mathbf{x}}(n) = O(\log n)$.

9.1.1 Examples of String Attractor Profile Functions

Here we show the behavior of the string attractor profile function of some infinite words.

First, we provide a non-recurrent infinite word having linear complexity function and unbounded string attractor profile function.

Example 168. *Let us consider the characteristic sequence* $\mathbf{c} = 1101000100000001\cdots$ *of the powers of* 2*, i.e.,* $c_i = 1$ *if* $i = 2^j$ *for some* $j \geq 0$*, and* 0 *otherwise. It is easy to see that* $\mathbf{c}$ *is aperiodic and not recurrent because the factor* 11 *has just one occurrence. It is known that* $p_{\mathbf{c}}(n)$ *and* $A_{\mathbf{c}}(n)$ *are* $\Theta(n)$ *([2]). One can prove that* $s_{\mathbf{c}}(n) = \Theta(\log n)$ *([74, 90, 119]).*

Then Example 169 shows that there exist recurrent (not uniformly) infinite words with unbounded string attractor profile function.

Example 169. *Let $\mu : \{0,1\}^* \to \{0,1\}^*$ be a 3-uniform morphism defined by $\mu(0) = 010$ and $\mu(1) = 111$. One can notice that the fixed point $\mathbf{u} = \mu^\infty(0) = 01011101011111111010\cdots$ is aperiodic, and it holds that $p_{\mathbf{u}}(n) = \Theta(n)$. Indeed it is recurrent, but not uniformly, since we have infinitely many runs of 1's with unbounded length. Moreover, note that the set of factors $\{01^{3^i}0 \mid i > 0\}$ do not overlap among them, thus we need at least one position from Γ for each of this factors, i.e., $s_{\mathbf{u}}$ is unbounded.*

However, many classical infinite words in literature have a known string attractor profile function and it is bounded (or even constant). It is the case of the Thue–Morse word [77, 119], and the period-doubling word [119], or the family of words defined by Holub in [61], showed in the following example.

Example 170. *Let us consider the following infinite word $\mathbf{u}$ introduced by Holub in [61] and defined as follows. Let $\{n_i\}_{i\geq 1}$, be an increasing sequence of positive integers with $n_1 \geq 2$. Then we define inductively the sequence $(u_i)_{i\geq 0}$ as $u_0 = \varepsilon$, $u_i = u_{i-1}0(u_{i-1}1)^{n_i}u_{i-1}$. Let us consider $\mathbf{u} = \lim_{i\to\infty} u_i$. It has been proven in [61] that $\mathbf{u}$ is uniformly recurrent but not linearly recurrent. Moreover, for each $i \geq 1$, $\mathbf{u}$ can be factorized as a product of words u_i0 and u_i1, i.e., $\mathbf{u} = u_ic_1u_ic_2u_ic_3\cdots$, where $c_j \in \{0,1\}$. More precisely, each occurrence of u_i starts at position that is a multiple of $|u_i| + 1$. By using the above properties, we can prove that $p_{\mathbf{u}}(n) = 2n$. Furthermore, it is possible to prove that, for $i \geq 0$, the set*

$$\left\{|u_i| + 1, \sum_{k=0}^{i}(|u_k| + 1), 2|u_i| + 2\right\}.$$

is a string attractor for u_{i+1} and a string attractor of constant size can be deduced for each prefix of u_{i+1}. Hence, $s_{\mathbf{u}}(n)$ is $\Theta(1)$.

Infinite word x	$p_x(n)$	Recurrence	$s_x(n)$
Period-doubling word p (Ex. 179)	$\Theta(n)$	Linearly recurrent	2 [119]
Thue-Morse word t (Ex. 5)	$\Theta(n)$	Linearly recurrent	4 [77]
Holub word u (Ex. 170)	$\Theta(n)$	Uniformly recurrent	3
Charact. Sturmian word s (Thm. 185)	$\Theta(n)$	Uniformly recurrent	2
Power of 2 charact. sequ. c (Ex. 168)	$\Theta(n)$	Not recurrent	$\Theta(\log n)$ [73]

FIGURE 9.1: Factor complexity function $p_{\mathbf{x}}$, recurrence, and string attractor profile function $s_{\mathbf{x}}$ for some infinite words.

9.1.2 The Bounded Case

Supported by the previous subsection, it is relevant to study the family of infinite words having bounded string attractor profile functions. Observe that we already know the following result.

Theorem 171 ([119])**.** *Let* $\mathbf{x}$ *be an infinite word. If* $\mathbf{x}$ *is linearly recurrent (i.e.,* $R_{\mathbf{x}}(n) = \Theta(n)$*), then* $s_{\mathbf{x}}(n) = \Theta(1)$*.*

However, in Example 170, we have shown that there exist uniformly (and not linearly) recurrent words for which $s_{\mathbf{x}}$ is bounded. Therefore, the previous theorem is not a characterization. In this section we gather results in this direction.

First, we analyze how the boundedness of $s_{\mathbf{x}}$ structures the infinite word $\mathbf{x}$ and we show that if an infinite word has its string attractor profile function bounded by some constant value, then it has at most linear factor complexity. More precisely, we have the following result.

Theorem 172. *Let* $\mathbf{x}$ *be an infinite word. If* $s_{\mathbf{x}} = \Theta(1)$*, then either* $\mathbf{x}$ *is eventually periodic, or* $\mathbf{x}$ *is* ω*-power free and* $p_{\mathbf{x}} = \Theta(n)$*.*

Proof. First observe that, by Theorem 165, if k is such that $s_{\mathbf{x}}(n) < k$ for each $n \geq 1$, then $p_{\mathbf{x}}(n) \leq n \cdot k$ for each $n \geq 1$. Therefore, the factor complexity is (at most) linear. Towards a contradiction, let us assume now that $\mathbf{x}$ is aperiodic and not ω-power free. Then there exists a factor w of $\mathbf{x}$ such that, for every $q \geq 1$, w^q is factor of $\mathbf{x}$. Moreover, $\mathbf{x} \neq uw^\omega$ for any $u \in \Sigma^*$, otherwise $\mathbf{x}$ would be eventually periodic. It follows that there exists an increasing sequence $(q_j)_{j\geq 1}$ such that, for each j, there exist a proper suffix s_j of w, a proper prefix p_j of w, and two letters a_j and b_j such that $a_j s_j$ is not a suffix of w, $p_j b_j$ is not a prefix of w, and $a_j s_j w^{q_j} p_j b_j$ is a factor of $\mathbf{x}$. As any position can cover at most two such factors, $s_{\mathbf{x}}$ is unbounded. □

Nonetheless we do not know whether Theorem 172 is a characterization or not. This raises the following open question.

Question 173. *Let* $\mathbf{x}$ *be* ω*-power free word such that* $p_{\mathbf{x}}$ *is linear. Is* $s_{\mathbf{x}}(n)$ *bounded by a constant value?*

Observe that Examples 168 and 169 do not allow to give a negative answer to the previous question as the words are not ω-power free. As a matter of fact, tackling a weaker version of the question, where instead $\mathbf{x}$ is uniformly recurrent word with linear $p_{\mathbf{x}}$, might be easier.

We have a partial converse of Theorem 172.

Proposition 174. *Let $\mathbf{x}$ be an infinite word. If $\mathbf{x}$ is eventually periodic, then $s_{\mathbf{x}}(n) = \Theta(1)$.*

Proof. Let $u \in \Sigma^*$ and $v \in \Sigma^+$ such that $\mathbf{x} = uv^\omega$. It is easy to see that, for all $n \geq 1$, $\{1, \ldots, \min\{n, |uv|\}\}$ is a string attractor for the prefix of length n. Therefore, $s_{\mathbf{x}}(n) \leq |uv|$ for all n. □

9.1.3 The Case of Purely Morphic Words

Some repetitiveness measures have been explored when applied to fixed points of morphisms, or more specifically, to iterated images of a morphism. It is the case of the number of BWT-runs in [46] and of the LZ-complexity function in [33]. Therefore, it is natural to wonder if similar results can be obtained for the string attractor profile function.

First we present an upper bound on the string attractor profile function of purely morphic words.

Theorem 175. *Let $\mathbf{x} = \varphi^\infty(a)$ be the fixed point of a morphism φ prolongable on $a \in \Sigma$. Then, $s_{\mathbf{x}}(n) = O(i)$, where i is such that $|\varphi^i(a)| \leq n < |\varphi^{i+1}(a)|$.*

Proof of Theorem 175. For all $i \geq 0$, define $n_i = |\varphi^i(a)|$. By Proposition 47, there exist two constant $c_1, c_2 \geq 1$ such that for all $n \in [n_i, n_{i+1})$, we have $c_1 \cdot i \leq z_{\mathbf{x}}(n_i) \leq z_{\mathbf{x}}(n) \leq z_{\mathbf{x}}(n_{i+1}) \leq c_2 \cdot i + c_2$. Note that the second and third inequalities follow by monotonicity of the measure z (i.e., $z(u) \leq z(uv)$ for all $u, v \in \Sigma^*$). This implies that $z_{\mathbf{x}}(n) = \Theta(i)$ and the conclusion follows by Theorem 116. □

In the following, we provide a finer result in the case of binary purely morphic word.

Theorem 176. *Let $\mathbf{x} = \mu^\infty(a)$ be the binary fixed-point of a morphism $\mu : \{a, b\}^* \to \{a, b\}^*$ prolongable on a. Then, either $s_{\mathbf{x}}(n) = \Theta(1)$ or $s_{\mathbf{x}}(n) = \Theta(\log n)$, and it is decidable when the first or the latter occurs.*

Proof. If $\mathbf{x}$ is eventually periodic, then Proposition 174 implies that $s_{\mathbf{x}}(n) = \Theta(1)$. Suppose now $\mathbf{x}$ is aperiodic. For morphisms defined on a binary alphabet, if $\mathbf{x} = \mu^{\infty}(a)$ is aperiodic, then $|\mu^i(a)|$ grows exponentially with respect to i (see [46]). Moreover, if μ is primitive, then by [36, Theorem 1] and [2, Theorem 10.9.4] $\mathbf{x}$ is linearly recurrent, and by Theorem 171 we have that $s_{\mathbf{x}}(n) = \Theta(1)$. If μ is not primitive, as summed up in [46], then only one of the following cases occurs:

1. There exist a coding $\tau : \Sigma \to \{a, b\}^+$ and a primitive morphism $\varphi : \Sigma^* \to \Sigma^*$ such that $\mathbf{x} = \mu^{\infty}(a) = \tau(\varphi^{\infty}(a))$ [104];

2. The word $\mathbf{x}$ contains arbitrarily large factors on $\{b\}^*$.

In the first case, since τ preserves the recurrence of a word and that $\varphi^{\infty}(a)$ is linearly recurrent, then $\mathbf{x}$ is linearly recurrent as well, and by Theorem 171 $s_{\mathbf{x}}(n) = \Theta(1)$.

In the second case, one can notice that $\mathbf{x}$ is not ω-power free, and by Theorem 172 for every $k \geq 1$ exists n' such that $s_{\mathbf{x}}(n) > k$, for every $n \geq n'$. More in detail, the number of distinct maximal runs of b's grows logarithmically with respect to the length of the prefixes of $\mathbf{x}$ [46], i.e., $s_{\mathbf{x}}(n) = \Omega(\log n)$. On the other hand, by Theorem 175 we know that $s_{\mathbf{x}}(n) = O(i)$, where $i \geq 1$ is such that $|\mu^i(a)| \leq n < |\mu^{i+1}(a)|$. Since $i = \Theta(\log n)$, we can further deduce an upper bound for the string attractor profile function and it follows that $s_{\mathbf{x}}(n) = \Theta(\log n)$. Finally, from a classification in [46] we can decide, only from μ, if either $s_{\mathbf{x}}(n) = \Theta(1)$ or $s_{\mathbf{x}}(n) = \Theta(\log n)$. □

Note that the result of Theorem 176 does not contradict a possible positive answer to Question 173, because the infinite words $\mathbf{x}$ with linear factor complexity and such that $s_{\mathbf{x}}(n) = \Theta(\log n)$ are not ω-power free. Moreover, the same bounds have been obtained for a related class of words, i.e., automatic sequences, as reported in the following theorem. In short, an infinite word $\mathbf{x}$ is *k-automatic* if and only if there exist a coding $\tau : \Sigma \to \Sigma$ and a k-uniform morphism μ_k such that $\mathbf{x} = \tau(\mu_k^{\infty}(a))$, for some $a \in \Sigma$ ([2]).

Theorem 177 ([119]). *Let $\mathbf{x}$ be a k-automatic infinite word. Then, either $s_{\mathbf{x}}(n) = \Theta(1)$ or $s_{\mathbf{x}}(n) = \Theta(\log n)$, and it is decidable when the first or the latter occurs.*

In fact, Schaeffer and Shallit [119] consider a larger family of infinite words, called *automatic* words, and show that, if the string attractor profile function is bounded, it is possible to build an automaton which returns the positions of a smallest string attractor for each prefix. However, the construction of such an automaton is done case by case using `Walnut`. For some particular automatic words obtained as fixed points of morphisms, string attractors may be found by hand. It is the case of the Thue–Morse word $\mathbf{t} = 0110100110010110\cdots$, which is the fixed point of $0 \mapsto 01, 1 \mapsto 01$. In [77], Kutsukake et al. show how to induce a smallest string attractor for some specific prefixes of $\mathbf{t}$ using the very particular structure of the word. Their construction can be adapted to the other prefixes. One can wonder whether the structure within morphic words can be used in a similar fashion to detect string attractors.

9.2 Span and Leftmost Complexities

Based on these two new measures, we can define related complexity functions for infinite words, called the *span complexity* and the *leftmost complexity*, that allow us to obtain a finer classification of infinite words. Indeed, Examples 179 and 186 highlight two infinite words, the period-doubling word and the Fibonacci word, which are not distinguishable if we consider their respective string attractor profile function as they are eventually equal to 2. However, the situation is very different if we look at how the positions within a string attractor are arranged.

Definition 178. *Let* $\mathbf{x}$ *be an infinite word. The* span *and* leftmost complexities *of* $\mathbf{x}$ *are respectively defined by* $\mathsf{span}_{\mathbf{x}}(n) = \mathsf{span}(\mathbf{x}[1,n])$ *and* $\mathsf{lm}_{\mathbf{x}}(n) = \mathsf{lm}(\mathbf{x}[1,n])$ *for all* $n \geq 1$.

Example 179 shows the behavior of such measures for the period-doubling word. Proposition 180 then shows the relationship between the profile function, the span and leftmost complexities.

Example 179. *Consider the period-doubling sequence* $\mathbf{pd} = 101110101011\cdots$, *which is the fixed point of the morphism* $1 \mapsto 10, 0 \mapsto 11$. *It has been proven in [119] that* $s_{\mathbf{pd}}(n) = 2$

for all $n \geq 1$*, while*

$$\mathit{span}_{\mathbf{pd}}(n) = \begin{cases} 1, & \text{if } 2 \leq n \leq 5; \\ 2^i, & \text{if } 3 \cdot 2^i \leq n < 3 \cdot 2^{i+1} \text{ for some } i \geq 1. \end{cases}$$

The next result directly follows from Proposition 135.

Proposition 180. *Let* $\mathbf{x}$ *be an infinite word. Then,* $s_{\mathbf{x}}(n) - 1 \leq \mathit{span}_{\mathbf{x}}(n) \leq \mathit{lm}_{\mathbf{x}}(n) - 1$.

As we did for the string attractor profile function, we will now focus on the case where these new complexities are "bounded". More specifically, we will characterize the infinite words such that they are bounded on an infinite subsequence.

We first look at the leftmost complexity. We will use the following intermediate result, which can be deduced from the proofs of [90, Propositions 12 and 15].

Proposition 181. *Let* w *be a non-empty word and let* $u = w^r$, $v = w^s$ *be fractional powers of* w *with* $1 \leq r \leq s$*. If* Γ *is a string attractor of* u*, then* $\Gamma \cup \{|w|\}$ *is a string attractor of* v*.*

Proposition 182. *Let* $\mathbf{x}$ *be an infinite word. Then there exists a constant* $C \geq 1$ *such that* $\mathit{lm}_{\mathbf{x}}(n) \leq C$ *for infinitely many* n *if and only if* $\mathbf{x}$ *is eventually periodic.*

Proof. The first implication follows from Proposition 134. Indeed, for all $m \geq 1$, there exists an integer n such that $\mathsf{lm}_{\mathbf{x}}(n) \leq C$ and $\mathbf{x}[1, n]$ contains all length-m factors. Therefore, $p_{\mathbf{x}}(m) \leq C$. Using Theorem 9, this implies that x is eventually periodic.

The second implication follows from Proposition 181. Indeed, if $\mathbf{x} = uv^\omega$, then for all $n \geq 1$, $\{1, 2, \ldots, \min\{n, |uv|\}\}$ is a string attractor for the word $\mathbf{x}[1, n]$. Therefore, $\mathsf{lm}_{\mathbf{x}}(n) \leq |uv|$ for all $n \geq 1$. □

This result gives a new characterization of eventually periodic words. Observe that the proof uses the well-known characterization by Morse and Hedlund (Theorem 9). Note that, in the following, we will mostly use the contraposition of Proposition 182.

We now look at a similar description for the span.

Proposition 183. *Let* $\mathbf{x}$ *be an infinite word. If there exists a constant* $C \geq 1$ *such that* $\mathit{span}_{\mathbf{x}}(n) \leq C$ *for infinitely many* n*, then* $\mathbf{x}$ *is eventually periodic or it is recurrent and* $p_{\mathbf{x}}(n) = n + d$ *with* $d \leq C$ *for all large enough* n*.*

Proof. Let us suppose that $\mathbf{x}$ is aperiodic. We first show that $\mathbf{x}$ is recurrent. Towards a contradiction, we assume that $\mathbf{x}$ is not recurrent. Therefore, there exists a factor that only occurs once in $\mathbf{x}$. Say that this occurrence ends at position k. This implies that, for all $n \geq k$, any string attractor of $\mathbf{x}[1, n]$ contains a position smaller than or equal to k. As $\mathsf{span}_{\mathbf{x}}(n) \leq C$ for infinitely many n, then $\mathsf{lm}_{\mathbf{x}}(n) \leq k + C$ for infinitely many n, which contradicts Proposition 182.

We now show that $\mathbf{x}$ has the claimed factor complexity. For all $m \geq 1$, there exists an integer n such that $\mathsf{span}_{\mathbf{x}}(n) \leq C$ and $\mathbf{x}[1, n]$ contains all length-m factors. By Proposition 130, we have $p_{\mathbf{x}}(m) \leq m + C$. Using Theorem 9 and as $\mathbf{x}$ is aperiodic, we conclude that $p_{\mathbf{x}}(m) = m + d$ for all large enough m and for $d \leq C$. □

Note that a converse-like characterization will be given in Theorem 191.

On the other hand, for some infinite words, we know that they have maximal span complexity, as stated in the following result.

Proposition 184. *Let $\mathbf{x}$ be a linearly recurrent word such that $p_{\mathbf{x}}(n) = n + \Omega(n)$. Then $\mathsf{span}_{\mathbf{x}}(n) = \Theta(n)$.*

Proof. Since $\mathbf{x}$ is linearly recurrent, by Remark 2, there exists an integer A such that, for all m, the length-(Am) prefix of $\mathbf{x}$ contains all length-m factors of $\mathbf{x}$. For all m, n such that $n \in [Am + 1, A(m + 1)]$, Proposition 130 implies that $\mathsf{span}_{\mathbf{x}}(n) \geq p_{\mathbf{x}}(m) - m$. By assumption on the factor complexity function, we have $p_{\mathbf{x}}(m) \geq Cm$ for a constant $C > 1$. Therefore $\mathsf{span}_{\mathbf{x}}(n) \geq (C - 1)m \geq (C - 1)\left(\frac{n}{A} - 1\right)$. This shows that $\mathsf{span}_{\mathbf{x}}(n) = \Omega(n)$. But since we trivially have $\mathsf{span}_{\mathbf{x}}(n) = O(n)$, the conclusion follows. □

Note that both the linear recurrence and the constraint on the factor complexity of an infinite word $\mathbf{x}$ from Proposition 184 are required. In fact, if we consider the word $\mathbf{u}$ in Example 170, from the distribution of the factors one can notice that the string attractor in the same example is of minimum span. Thus, if the increasing sequence $\{n_i\}_i \geq 1$ needed to build $\mathbf{u}$ grows super-exponentially, the span complexity grows sublogarithmically for infinite prefixes.

On the other hand, recall that any characteristic Sturmian word $\mathbf{x}$ has factor complexity $p_{\mathbf{x}}(n) = n + 1$. In the next section, we show that for this family of words it holds that the function $\mathsf{span}_{\mathbf{x}}$ is constant.

9.3 The Case of Sturmian Words

In this section, we analyze properties of the string attractor profile function along with the two new string-attractor related complexities for Sturmian words and two related families of infinite words. On the one hand, we consider the subfamily of *characteristic Sturmian words*. On the other hand, we investigate the superfamily of *quasi-Sturmian* words, which can be considered the simplest generalizations of Sturmian words in terms of factor complexity.

9.3.1 String Attractor based Complexities for Characteristic Sturmian Words

We focus here on the family of characteristic Sturmian words, for which we can explicitly give the string attractor profile function, the span complexity and the leftmost complexity by giving string attractors realizing these. Since we denote the string attractor profile function by s, for simplicity of exposition we refer to a characteristic Sturmian word as $\mathbf{x}$, and the corresponding standard prefixes as x_i.

The following result shows that each prefix of a characteristic Sturmian word has a smallest string attractor of span 1, i.e., consisting of two consecutive positions.

Theorem 185. *Consider a directive sequence $(d_i)_{i\geq 0}$ with $q_0 \geq 1$, the corresponding sequence $(x_i)_{i\geq 0}$ of standard Sturmian words and the associated characteristic Sturmian word $\mathbf{x} = \lim_{i\to\infty} x_i$.Then we have*

$$s_{\mathbf{x}}(n) = \begin{cases} 1, & \text{if } n < |x_2|; \\ 2, & \text{if } n \geq |x_2|; \end{cases} \qquad span_{\mathbf{x}}(n) = \begin{cases} 0, & \text{if } n < |x_2|; \\ 1, & \text{if } n \geq |x_2|; \end{cases}$$

and

$$lm_{\mathbf{x}}(n) = \begin{cases} 1, & \text{if } n < |x_2|; \\ |x_k|, & \text{if } |x_k| + |x_{k-1}| - 1 \leq n \leq |x_{k+1}| + |x_k| - 2 \text{ for some } k \geq 2. \end{cases}$$

More specifically, for all $n \geq 1$, a string attractor for $\mathbf{x}[1, n]$ is given by

$$\Gamma_n = \begin{cases} \{1\}, & \text{if } n < |x_2|; \\ \{|x_k| - 1, |x_k|\}, & \text{if } |x_k| + |x_{k-1}| - 1 \leq n \leq |x_{k+1}| + |x_k| - 2 \text{ for some } k \geq 2. \end{cases}$$

Proof. We start the proof by showing the last part of the statement, i.e., we show that, for all $n \geq 1$, the given Γ_n is a string attractor for $\mathbf{x}[1, n]$. Observe first that, if $n < |x_2|$, then $\mathbf{x}[1, n] = a^n$, so $\{1\}$ is directly a string attractor. For the case $n \geq |x_2|$, we will need the following notations. For all $k \geq 2$, using [86, Theorem 3], we factorize the standard Sturmian word x_k into $x_k = C_k u_k$ where C_k is a palindrome and $u_k = ab$ if k is even and $u_k = ba$ if k is odd. We also recall the following observation from [84, Theorem 2.2.11]: for all $k \geq 3$,

$$\mathbf{x}[1, |x_k| + |x_{k-1}| - 2] = x_k C_{k-1} = C_k u_k C_{k-1},$$

therefore the previous word is periodic of period $|C_{k-1}| + 2 = |x_{k-1}|$.

Assume now that $n \geq |x_2|$ and let $k \geq 2$ be such that $|x_k| + |x_{k-1}| - 1 \leq n \leq |x_{k+1}| + |x_k| - 2$ (such a k exists since $|x_2| + |x_1| - 1 = |x_2|$). Since $\mathbf{x}[1, |x_{k+1}| + |x_k| - 2]$ is periodic of period $|x_k|$, then it is a fractional power of x_k. Therefore, using Proposition 181, it is enough to show that $\Gamma_n = \{|x_k| - 1, |x_k|\}$ is a string attractor of the length-$(|x_k| + |x_{k-1}| - 1)$ prefix of $\mathbf{x}$, that we will denote p_k.

If $k = 2$ or $k = 3$, the conclusion is direct as $p_2 = x_2 = a^{q_0} b$ and $p_3 = (a^{q_0} b)^{q_1} a a^{q_0}$. If $k \geq 4$, we use the fact that a similar result was proved for the standard Sturmian words in [90, Theorem 22]. Namely, Γ_n is a string attractor for x_{k+1}. To show that Γ_n is also a string attractor for p_k, we will show that $w := \mathbf{x}[|x_k|, |p_k|]$ does not occur elsewhere in p_k. Indeed, this will imply that for each factor of p_k its occurrence that was covered by Γ_n in x_{k+1} is an occurrence in p_k (also covered by Γ_n).

Observe that, as $k \geq 3$, $w = cC_{k-1}c$ where c is the last letter of u_k and the first letter of u_{k-1}. Note that w is not a suffix of $\mathbf{x}[1, |p_k| - 1] = x_k C_{k-1}$ as x_k ends with $u_k = dc$, $d \neq c$. Therefore, if w is a factor of $x_k C_{k-1}$, it is followed by c since $x_k C_{k-1}$ is periodic of period $|x_{k-1}| = |w|$. In particular, $C_{k-1}cc$ and $C_{k-1}cd = x_{k-1}$ are factors of $x_k C_{k-1}$. This implies that $C_{k-1}c$ is right special and, by [84], cC_{k-1} is a prefix of x_k. As $C_{k-1}c$ is also a prefix of x_k, this implies that C_{k-1} is periodic of period 1, a contradiction as $k \geq 4$. This ends the proof that w is not a factor of $x_k C_{k-1}$ and, with it, the proof that Γ_n is a string attractor of $\mathbf{x}[1, n]$.

Moreover, we directly have that Γ_n is of minimal size and of minimal span among the string attractors of $\mathbf{x}[1, n]$. It is also a leftmost string attractor as each string

n	1	2	3	4	5	6	7	8
$\mathbf{x}[1,n]$	$\underline{a}$	$\underline{ab}$	$\underline{ab}a$	$a\underline{ba}a$	$a\underline{ba}ab$	$a\underline{ba}aba$	$aba\underline{ab}ab$	$aba\underline{ab}aba$
Γ_n	$\{1\}$	$\{1,2\}$	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{2,3\}$	$\{4,5\}$	$\{4,5\}$

TABLE 9.1: For $n \in [1,8]$, the length-n prefix of the Fibonacci word $\mathbf{x} = abaababaabaab\cdots$ and its leftmost string attractor Γ_n.

attractor of $\mathbf{x}[1,n]$ will contain a position greater than or equal to $|x_k|$ to cover w. This proves the three claimed complexities. □

Example 186. *Consider the infinite Fibonacci word* $\mathbf{x} = abaababaabaababaa\cdots$*, which is a characteristic Sturmian word with directive sequence* $(1)_{i\geq 0}$*. In Table 9.1, for* $1 \leq n \leq 8$*, we exhibit the length-n prefixes of* $\mathbf{x}$ *and their respective leftmost string attractor* Γ_n*. The underlined positions in* $\mathbf{x}[1,n]$ *correspond to those in* Γ_n*, while the prefixes* $\mathbf{x}[1,n]$ *for* $n \in \{1,2,3,5,8\}$ *are standard Sturmian words.*

While infinitely many characteristic Sturmian words have the same string attractor profile function (resp., the same span complexity), the leftmost complexity uniquely determines the characteristic Sturmian word (up to exchanging the letters a and b, captured by the exchange morphism E).

Proposition 187. *If* $\mathbf{x}$ *and* $\mathbf{y}$ *are two characteristic Sturmian words such that* $\mathsf{lm}_{\mathbf{x}} = \mathsf{lm}_{\mathbf{y}}$*, then either* $\mathbf{x} = \mathbf{y}$ *or* $\mathbf{x} = E(\mathbf{y})$*.*

Proof. Let $(q_i)_{i\geq 0}$ and $(p_i)_{i\geq 0}$ be two directive sequences and let $(x_i)_{i\geq 0}$ and $(y_i)_{i\geq 0}$ be the corresponding sequences of standard Sturmian words. Now consider the associated characteristic Sturmian words $\mathbf{x}$ and $\mathbf{y}$. Without loss of generality, assume that, up to exchanging a and b, both $\mathbf{x}$ and $\mathbf{y}$ start with the letter a (i.e., $q_0, p_0 \geq 1$). The assumption that $\mathsf{lm}_{\mathbf{x}} = \mathsf{lm}_{\mathbf{y}}$ together with Theorem 185 now imply that the sequences $(|x_i|)_{i\geq 0}$ and $(|y_i|)_{i\geq 0}$ are equal. A simple inuction shows that $q_i = p_i$ for all i, therefore $\mathbf{x} = \mathbf{y}$. □

On the other hand, some prefixes of non-characteristic Sturmian words do not admit any string attractor of span 1, as shown in the following example.

Example 188. *Let* $\mathbf{x} = aaaaaabaaaaaabaaaaaaab\cdots$ *be the characteristic Sturmian word associated with the directive sequence* $(6,2,\ldots)$*. Consider the non-characteristic Sturmian word* $\mathbf{x}'$ *such that* $\mathbf{x} = aaaa \cdot \mathbf{x}'$*, hence* $\mathbf{x}' = aabaaaaaabaaaaaaab\cdots$*. Let us consider the*

prefix $\mathbf{x}'[1,14] = aabaaaaaabaaaa$. *Since* b *occurs only at positions* 3 *and* 10 *and the factor* $aaaaaa$ *only in* $\mathbf{x}'[4,9]$, *the candidates as string attractor with two consecutive positions are* $\Gamma_1 = \{3,4\}$ *and* $\Gamma_2 = \{9,10\}$. *However, one can check that the factors* $aaab$ *and* $baaaaa$ *do not cross any position in* Γ_1 *and* Γ_2 *respectively. Nonetheless, there exists a string attractor of size* 2 *that does not contain two consecutive positions, i.e.,* $\Gamma = \{4,10\}$.

9.3.2 Characterization of Sturmian and Quasi-Sturmian Words

We now turn to the families of Sturmian and quasi-Sturmian words. For each, we provide a new characterization in terms of both the span and leftmost complexities of the prefixes.

We start off with Sturmian words.

Theorem 189. *An infinite word* $\mathbf{x}$ *is Sturmian if and only if there is no integer* $C \geq 1$ *such that* $\mathsf{lm}_{\mathbf{x}}(n) < C$ *for infinitely many* $n \geq 1$, *and* $\mathsf{span}_{\mathbf{x}}(n) = 1$ *for infinitely many* $n \geq 1$.

Proof. For the first implication, let $\mathbf{x}$ be a Sturmian word. Since $\mathbf{x}$ is aperiodic, Proposition 182 shows that $\mathsf{lm}_{\mathbf{x}}$ satisfies the statement. We now establish the claimed property on $\mathsf{span}_{\mathbf{x}}$. As $\mathbf{x}$ is aperiodic and recurrent, it has infinitely many right special prefixes. Moreover, for every such prefix v, there is a characteristic Sturmian word $\mathbf{s}$ having v^R as a prefix [85, Proposition 2.1.23]. Therefore, $\mathsf{span}(v) = \mathsf{span}(v^R) = 1$ for all long enough v by Theorem 185 and the proof of [90, Proposition 11].

For the other implication, consider an infinite word $\mathbf{x}$ satisfying the assumption. First, it is aperiodic by Proposition 182. Moreover, by assumption, for all $m \geq 1$, there exists an integer n such that $\mathbf{x}[1,n]$ contains all length-m factors and $\mathsf{span}_{\mathbf{x}}(n) = 1$. Therefore, $p_{\mathbf{x}}(m) \leq m+1$ by Proposition 130. The fact that $\mathbf{x}$ is Sturmian follows from Theorem 9. □

We conclude the section with the a novel characterization of quasi-Sturmian words. Indeed, they are defined as follows [22]: a word $\mathbf{x}$ is *quasi-Sturmian* if there exist integers d and n_0 such that $p_{\mathbf{x}}(n) = n + d$, for each $n \geq n_0$. The infinite words having factor complexity $n + d$ have been also studied in [59] where they are called "words with minimal block growth". To show our result, we will make use of the following characterization of quasi-Sturmian words [22].

Theorem 190 ([22]). *An infinite word* $\mathbf{x}$ *over the alphabet* Σ *is quasi-Sturmian if and only if it can be written as* $\mathbf{x} = w\varphi(\mathbf{s})$*, where* w *is a finite word and* $\mathbf{s}$ *is a Sturmian word on the alphabet* $\{a, b\}$*, and* φ *is a morphism from* $\{a, b\}^*$ *to* Σ^* *such that* $\varphi(ab) \neq \varphi(ba)$*.*

The following theorem shows that constant values for the span complexity at infinitely many points imply quasi-Sturmian words, i.e., the most repetitive infinite aperiodic words after the Sturmian words.

Theorem 191. *An infinite word* $\mathbf{x}$ *is quasi-Sturmian if and only if there is no constant* $C \geq 1$ *such that* $\mathsf{lm}_{\mathbf{x}}(n) < C$ *for infinitely many* $n \geq 1$ *and there exist a suffix* $\mathbf{y}$ *of* $\mathbf{x}$ *and a constant* $C' \geq 1$ *such that* $\mathsf{span}_{\mathbf{y}}(n) \leq C'$ *for infinitely many* $n \geq 1$*.*

Proof. For the first implication, as quasi-Sturmian words are aperiodic by Theorem 9, $\mathsf{lm}_{\mathbf{x}}$ satisfies the statement by Proposition 182. In addition, by Theorem 190, there exists a finite word w, a Sturmian word $\mathbf{s}$, and a morphism φ such that $\mathbf{x} = w\varphi(\mathbf{s})$. Consider the suffix $\mathbf{y} = \varphi(\mathbf{s})$. By Theorem 189, there are infinitely many integers n such that $\mathsf{span}_{\mathbf{s}}(n) = 1$, and by Proposition 136, there exists a constant $C' \geq 1$ such that, for all $N = |\varphi(\mathbf{s}[1, n])|$,

$$\mathsf{span}_{\mathbf{y}}(N) = \mathsf{span}(\varphi(\mathbf{s}[1, n])) \leq C' \cdot \mathsf{span}(\mathbf{s}[1, n]) = C'.$$

For the other implication, by Propositions 182 and 183, $p_{\mathbf{y}}(n) = n + d$ with $d \leq C'$ for all large enough n. Since $\mathbf{x} = w\mathbf{y}$ for some finite word w, we have $p_{\mathbf{x}}(n) \leq p_{\mathbf{y}}(n) + |w| = n + d + |w|$ for all large enough n. We conclude by Theorem 9 that x is quasi-Sturmian. □

Chapter 10

String Attractors and k-Bonacci Like Morphisms

In this chapter, we consider the problem of computing the string attractor based complexities for the infinite family of words that are fixed points of a *k-bonacci like morphism*, a generalization of the Fibonacci morphism which can be defined for alphabets of any size.

In Section 10.1, we focus on the stricter and well-known family of k-bonacci words, and show a detailed structure of the string attractor profile function, span and leftmost complexities.

Then, in Section 10.2 we show how to generalize these results to fixed points of k-bonacci like morphisms. As one will observe, the sting attractors for this family of words yields an interesting relationship with standard notions from the field of Combinatorics on Words.

Part of the material from this section will be presented at the conference Combinatorics on Words (WORDS 2023) [53].

10.1 String Attractor based Complexities for k-Bonacci Words

In this section, we study string attractors of prefixes of some purely morphic words over an alphabet of size $k \geq 2$, namely the so-called *k-bonacci words*. The case $k = 2$ corresponds to the famous Fibonacci word[1], which is a Sturmian word and for which

[1]Note that in Example 186, the Fibonacci word is defined on the alphabet $\{\mathtt{a}, \mathtt{b}\}$ to match the general definition of Sturmian words. In this section, for the sake of simplicity, we define it on $\{0, 1\}$ instead.

string-attractor related concepts have already been studied. For $k = 3$, each prefix of the Tribonacci word admits a string attractor of size at most 3 as shown in [119].

More generally, as k-bonacci words are *episturmian*, Dvořáková showed that each prefix admits a string attractor of size at most k [42, Theorem 10]. However, this result is not constructive in the sense that the string attractors are not explicitly given. Our contribution is to provide a constructive description of a string attractor of size at most k for each prefix. Our new approach differs from the techniques used to obtain string attractors for the Thue–Morse word, the period-doubling word, and standard Sturmian words, and may well extend to other purely morphic words. Furthermore we then study the leftmost and the span complexities of the k-bonacci words.

10.1.1 k-Bonacci Words

Let us consider an integer $k \geq 2$ and the morphism $\mu_k : \{0, \ldots, k-1\}^* \to \{0, \ldots, k-1\}^*$ defined by $\mu_k(i) = 0(i+1)$ for all $i \in \{0, 1, \ldots, k-2\}$ and $\mu_k(k-1) = 0$. The *infinite k-bonacci word* $\mathbf{b}^{(k)}$ is defined as the fixed-point $\mathbf{b}^{(k)} = \mu_k^\infty(0)$. The cases $k = 2$ and $k = 3$ correspond to the Fibonacci and Tribonacci words respectively.

Further, for all $n \geq 0$, we let $b_n^{(k)} = \mu_k^n(0)$ denote the *nth finite k-bonacci word*. We also set $b_n^{(k)} = \varepsilon$ for all $-k \leq n < 0$. For any $n \geq 0$, we let $B_n^{(k)} = |b_n^{(k)}|$ denote the length of the nth finite k-bonacci word. The sequence $(B_n^{(k)})_{n \geq 0}$ will be referred to as the sequence of *k-bonacci numbers*. When the context is clear, we will drop the superscript (k) in all of these notations.

Example 192. *For $k = 3$, we write the first few non empty finite Tribonacci words in Table 10.1.*

n	0	1	2	3	4	5
$b_n^{(3)}$	0	0 1	01 0 2	0102 01 0	0102010 0102 01	0102010010201 0102010 0102

TABLE 10.1: The first few finite Tribonacci words $(b_n^{(3)})_{0 \leq n \leq 5}$ (some particular decomposition is highlighted for a latter purpose, see Proposition 193).

Another way of seeing the sequence $(b_n^{(k)})_{n \geq -k}$ is the following, which can be proven by an easy induction. See Table 10.1 for an example with $k = 3$.

Proposition 193. *We have*

$$b_n^{(k)} = \begin{cases} \left(\prod_{i=1}^{k} b_{n-i}^{(k)}\right) \cdot n = \left(\prod_{i=1}^{n} b_{n-i}^{(k)}\right) \cdot n, & \text{if } 0 \leq n \leq k-1; \\ \prod_{i=1}^{k} b_{n-i}^{(k)}, & \text{if } n \geq k. \end{cases}$$

For the small values of n, we have a more precise description of $b_n^{(k)}$.

Lemma 194. *For all $1 \leq n \leq k$, there exists a word u_n over the alphabet $\{0, 1, \ldots, n-2\}$ such that*

$$b_n^{(k)} = \begin{cases} u_n \cdot (n-1) \cdot u_n \cdot n, & \text{if } 1 \leq n \leq k-1; \\ u_n \cdot (n-1) \cdot u_n, & \text{if } n = k. \end{cases}$$

Moreover, u_n is of length $B_{n-1}^{(k)} - 1$ thus

$$B_n^{(k)} = \begin{cases} 2B_{n-1}^{(k)}, & \text{if } 1 \leq n \leq k-1; \\ 2B_{n-1}^{(k)} - 1, & \text{if } n = k. \end{cases}$$

Proof. We proceed by induction on n, for any $1 \leq n \leq k$. For the base case, one observes that $b_1 = \varepsilon \cdot 0 \cdot \varepsilon \cdot 1$, thus we can take $u_1 = \varepsilon$ and $|u_1| = 0 = B_0 - 1$. Assume now that the claim is true for $n-1$, and let us prove it for n. We have

$$\begin{aligned} b_n &= \mu_k(b_{n-1}) = \mu_k(u_{n-1})\mu_k(n-2)\mu_k(u_{n-1})\mu_k(n-1) \\ &= \begin{cases} \mu_k(u_{n-1}) \cdot 0(n-1) \cdot \mu_k(u_{n-1}) \cdot 0n & \text{if } 1 \leq n \leq k-1 \\ \mu_k(u_{k-1}) \cdot 0(k-1) \cdot \mu_k(u_{k-1}) \cdot 0 & \text{if } n = k. \end{cases} \end{aligned}$$

Thus, we choose $u_n = \mu_k(u_{n-1})0$ which is indeed over the alphabet $\{0, 1, \ldots, n-2\}$. In particular, the first occurrence of the letter $n-1$ is at index $|u_n| + 1$. By the induction hypothesis, it also coincides with the last letter of b_{n-1} thus $|u_n| = B_{n-1} - 1$. □

We now define two sequences of integers $(P_n(k))_{n\geq 0}$ and $(Q_n(k))_{n\geq 0}$ linked to k-bonacci numbers that will help us partition $\mathbb{N}$. For all $n \geq 0$, we set

$$P_n(k) = \begin{cases} B_n^{(k)}, & \text{if } n \leq k; \\ B_n^{(k)} + B_{n-k-1}^{(k)} - 1, & \text{otherwise;} \end{cases}$$

and $Q_n^{(k)} = \sum_{i=0}^{n} B_i^{(k)}$. Observe that, if $n \geq k+1$, then $P_n^{(k)} = \left(\sum_{i=n-k-1}^{n-1} B_i^{(k)}\right) - 1$ and if $n \leq k-1$, then $Q_n^{(k)} = 2B_n^{(k)} - 1$. Moreover, if $n \geq k$, $B_{n+1}^{(k)} = \sum_{i=n-k+1}^{n} B_i \in [P_n^{(k)}, Q_n^{(k)}]$.

Example 195. *When $k = 3$, we obtain $(P_n^{(3)})_{n\geq 0} = 1, 2, 4, 7, 13, 25, 47, 87, \ldots$ and $(Q_n^{(3)})_{n\geq 0} = 1, 3, 7, 14, 27, 51, 95, 176, \ldots$.*

For any $k \geq 2$, one can show that $(Q_n^{(k)})_{n\geq 0}$ gives the lengths of palindromic prefixes of the k-bonacci word (note that the case $k = 3$ gives the sequence [123, A027084]).

Lemma 196. *For all $m \geq 1$, there exists at least one index $n \geq 0$ such $m \in [P_n^{(k)}, Q_n^{(k)}]$.*

Proof. Since $P_0 = 1$ and the sequences are increasing, it suffices to show that $P_{n+1} \leq Q_n + 1$ for all $n \geq 0$. Let us assume first that $0 \leq n \leq k-1$. Lemma 194 implies that $Q_n = \sum_{i=0}^{n} B_i = \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$. By Lemma 194 again, we obtain

$$P_{n+1} = \begin{cases} 2^{n+1}, & \text{if } 0 \leq n \leq k-2; \\ 2^{n+1} - 1, & \text{if } n = k-1. \end{cases}$$

The inequality $P_{n+1} \leq Q_n + 1$ easily follows in this case. Now assume $n \geq k$. We have

$$Q_n = \sum_{i=0}^{n} B_i = \sum_{i=0}^{n-k-1} B_i + B_{n-k} + B_{n+1} > P_{n+1}. \qquad \square$$

10.1.2 String attractor profile function

We study the string attractor profile function of the k-bonacci word $\mathbf{b}^{(k)}$, by first looking at string attractors of small prefixes, then long ones. For all $1 \leq m \leq Q_{k-1}^{(k)}$, we obtain a smallest string attractor of size at most k for the length-m prefix of $\mathbf{b}^{(k)}$.

Proposition 197. *For all $0 \leq n \leq k-1$, $\Gamma_n = \{B_0^{(k)}, \ldots, B_n^{(k)}\}$ is a minimum string attractor for $\mathbf{b}^{(k)}[1, m]$ for all $m \in [P_n^{(k)}, Q_n^{(k)}]$.*

Proof. First, notice that, by Lemma 194, the letters at positions $B_0, B_1, \ldots, B_n$ are respectively $0, 1, \ldots, n$ thus, if Γ_n is a string attractor, it is minimum. Let us prove that it is a string attractor for the given prefixes.

For $n = 0$, the interval $[P_n, Q_n]$ is the singleton $\{1\}$ thus the only prefix to consider is the prefix of length 1 and the conclusion is direct. Let us now assume that the claim is true for $n-1$ and let $m \in [P_n, Q_n]$. As $P_n = B_n$ and $Q_n = 2B_n - 1$ using Lemma 194, the word $\mathbf{b}[1, m]$ can be written as $u_{n+1} \cdot n \cdot p$ with $u_{n+1} = \mathbf{b}[1, P_n)$ and p a prefix of u_{n+1}. Moreover, since $n \leq k-1$, we have $Q_{n-1} = P_n - 1$ so $u_{n+1} = \mathbf{b}[1, Q_{n-1}]$. By the induction hypothesis, Γ_{n-1} is a string attractor for $\mathbf{b}[1, Q_{n-1}]$, therefore $\Gamma_n = \Gamma_{n-1} \cup \{B_n\}$ is a string attractor for $u_{n+1} \cdot n$. We conclude by Proposition 181 that Γ_n is a string attractor for $\mathbf{b}[1, m]$. □

Just as for shorter prefixes, we will use Proposition 181 to obtain strings attractors for longer prefixes of the k-bonacci word. To that aim, we will study prefixes that are fractional powers.

Proposition 198. *For all $n \geq 0$, $\mathbf{b}^{(k)}[1, Q_n^{(k)}] = \prod_{i=0}^{n} b_{n-i}^{(k)}$. Moreover, $\mathbf{b}^{(k)}[1, Q_n^{(k)}]$ is a fractional power of $b_n^{(k)}$.*

Proof. For $n = 0$, we directly have $\mathbf{b}[1, Q_0] = \mathbf{b}[1, 1] = b_0$, so both claims hold in this case. Assume now that the result is true for n and let us prove it for $n+1$. By the induction hypothesis, we have

$$\mu_k(\mathbf{b}[1, Q_n]) = \mu_k\left(\prod_{i=0}^{n} b_{n-i}\right) = \prod_{i=0}^{n} b_{n+1-i}.$$

As $\mathbf{b}$ is a fixed point of μ_k, $\mu_k(\mathbf{b}[1, Q_n])$ is a prefix of $\mathbf{b}$ and it is followed by the image of a letter. Thus it is followed by a letter 0, and

$$\mathbf{b}[1, Q_{n+1}] = \mathbf{b}\left[1, \sum_{i=0}^{n+1} B_i\right] = \left(\prod_{i=0}^{n} b_{n+1-i}\right) \cdot 0 = \prod_{i=0}^{n+1} b_{n+1-i}.$$

Moreover, since $\mathbf{b}[1, Q_n]$ is a fractional power of b_n by the induction hypothesis, so is $\mathbf{b}[1, Q_n] \cdot a$ for some letter $a \in \{0, 1, \ldots, k-1\}$. By applying the morphism μ_k on both words, we can conclude that $\mathbf{b}[1, Q_{n+1}] = \mu_k(\mathbf{b}[1, Q_n]) \cdot 0$ is a fractional power of $b_{n+1} = \mu_k(b_n)$. □

Using Proposition 181, we then directly have the following corollary.

Corollary 199. *For all $n \geq k$, if Γ is a string attractor for $\mathbf{b}^{(k)}[1, P_n^{(k)}]$ and if $B_n^{(k)} \in \Gamma$, then Γ is a string attractor for $\mathbf{b}^{(k)}[1, m]$ for all $m \in [P_n^{(k)}, Q_n^{(k)}]$. In particular, Γ is a string attractor for $b_{n+1}^{(k)}$.*

We now exhibit a minimum string attractor of size k for all long enough prefixes of $\mathbf{b}^{(k)}$.

Proposition 200. *Let $n \geq k$. The set $\Gamma_n = \{B_{n-k+1}^{(k)}, \ldots, B_n^{(k)}\}$ is a minimum string attractor for $\mathbf{b}^{(k)}[1, P_n^{(k)}]$.*

Proof. Notice that, as $P_n \geq B_k$, the word $\mathbf{b}[1, P_n]$ contains k different letters, which implies that, if Γ_n is a string attractor, it is minimum. We prove that it is a string attractor by induction on $n \geq k$. More precisely, we prove two claims during the induction step:

1. Γ_{n-1} is a string attractor for b_n;
2. Γ_n is a string attractor for $\mathbf{b}[1, P_n]$.

As will become clear later on, we will use the first claim to prove the second.

Let us prove the first claim. If $n = k$, Proposition 197 implies that Γ_{k-1} is a string attractor for b_k since $B_k = 2B_{k-1} - 1$ by Lemma 194. If $n \geq k+1$, by the induction hypothesis, Γ_{n-1} is a string attractor for $\mathbf{b}[1, P_{n-1}]$. By Corollary 199, Γ_{n-1} is also a string attractor for b_n.

Let us prove the second claim. Observe that, since the finite k-bonacci words are prefixes of each others, $\mathbf{b}[1, P_n] = b_n u$, where $u = \varepsilon$ if $n = k$ or u is b_{n-k-1} without its last letter if $n \geq k+1$. By Proposition 181, we deduce from the previous factorization and the first claim that $\Gamma_{n-1} \cup \{B_n\} = \Gamma_n \cup \{B_{n-k}\}$ is a string attractor for $\mathbf{b}[1, P_n]$. Thus, to prove the second claim, it remains to show that the position B_{n-k} is not needed in the string attractor, i.e., the factors of $\mathbf{b}[1, P_n]$ that are covered by position B_{n-k} are still covered by Γ_n. As the first position in Γ_n is B_{n-k+1}, it suffices to consider the factor occurrences crossing position B_{n-k} in $\mathbf{b}[1, B_{n-k+1})$. As $\mathbf{b}[1, B_{n-k+1})$ is b_{n-k+1} without its last letter, Proposition 193 implies that they are occurrences in

$$\prod_{i=1}^{k} b_{n-k+1-i} = b_{n-k} b_{n-k-1} \prod_{i=3}^{k} b_{n-k+1-i}.$$

Note that $b_{n-k}u$ is a prefix of this word. We consider two cases: either the considered occurrence is entirely contained in $b_{n-k}u$ or it crosses position $B_{n-k} + B_{n-k-1}$. Observe that, if $n \geq k+1$, these two cases are mutually exclusive.

Case 1. Since b_{n-k} is a suffix of b_n by Proposition 193, the factors having an occurrence in $b_{n-k}u$ crossing position B_{n-k} have an occurrence in $b_n u$ crossing position B_n, so they are covered by Γ_n. See Figure 10.1.

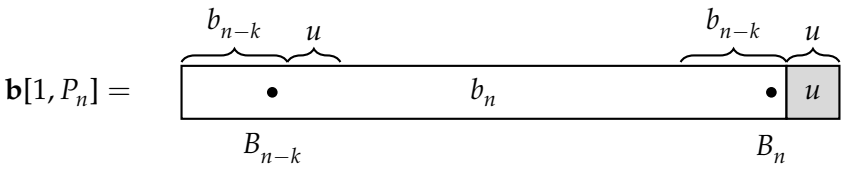


FIGURE 10.1: Case 1 in the proof of Proposition 200.

Case 2. Similarly, by Proposition 193, $b_{n-k}b_{n-k-1}$ is a suffix of b_{n-1} and $\prod_{i=3}^{k} b_{n-k+1-i} = \prod_{i=1}^{k-2} b_{n-k-1-i}$ is a prefix of b_{n-k-1}, so of b_{n-2} (again the finite k-bonacci words are prefixes of each others). As $b_{n-1}b_{n-2}$ is a prefix of b_n, we conclude that the factors having an occurrence in $\mathbf{b}[1, B_{n-k+1})$ crossing position $B_{n-k} + B_{n-k-1}$ have an occurrence in b_n crossing position B_{n-1}, so they are covered by Γ_n. See Figure 10.2.

□

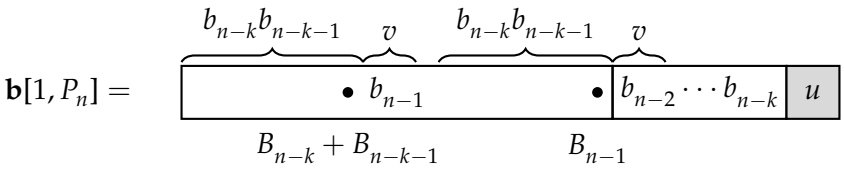


FIGURE 10.2: Case 2 in the proof of Proposition 200 with $v = \prod_{i=3}^{k} b_{n-k+1-i}$.

Putting together Propositions 197 and 200 and Corollary 199 , we explicitly obtain strings attractors for prefixes of the k-bonacci word.

Theorem 201. *For all $n \geq 0$, the set*

$$\Gamma_n = \begin{cases} \{B_0^{(k)}, \ldots, B_n^{(k)}\}, & \text{if } n \leq k-1; \\ \{B_{n-k+1}^{(k)}, \ldots, B_n^{(k)}\}, & \text{if } n \geq k; \end{cases}$$

is a minimum string attractor for $\mathbf{b}^{(k)}[1,m]$*, for all* $m \in [P_n^{(k)}, Q_n^{(k)}]$*. In particular, the string attractor profile function for* $\mathbf{b}^{(k)}$ *is given by*

$$s_{\mathbf{b}^{(k)}}(n) = \begin{cases} i+1, & \text{if } B_i^{(k)} \leq n < B_{i+1}^{(k)} \text{ for some } i \leq k-2; \\ k, & \text{if } n \geq B_{k-1}^{(k)}. \end{cases}$$

Remark 202. *Observe that, in the Tribonacci case, our sequence* $(Q_n^{(3)})_{n\geq 0}$ *is related to the sequence* $(W_n)_{n\geq 4}$ *defined in [119, Theorem 6] as follows: we have* $W_{n+3} = Q_{n+1}^{(3)}$ *for all* $n \geq 1$*. Therefore, our upper bound and that of Schaeffer and Shallit [119, Theorem 6] coincide, as well as the elements in the string attractors. However, our lower bounds are smaller than theirs. On the other hand, the string attractors obtained for the palindromic prefixes in [42] are different from ours. For instance, the case* $k = 3$ *is treated in [42, Example 8].*

10.1.3 Leftmost complexity

We can further prove that the string attractor from Theorem 201 is actually a leftmost string attractor. For the purpose of the next few results, we set $Q_{-1}^{(k)} = 0$.

Proposition 203. *The leftmost complexity of* $\mathbf{b}^{(k)}$ *satisfies* $lm_{\mathbf{b}^{(k)}}(m) = B_n^{(k)}$ *for all* $n \geq 0$ *and* $m \in [Q_{n-1}^{(k)} + 1, Q_n^{(k)}]$.

Proof. We show that the factor $\mathbf{b}[B_n, Q_{n-1}+1]$ does not occur in $\mathbf{b}$ before position B_n. This implies that, for all $m \geq Q_{n-1}+1$, any string attractor of $\mathbf{b}[1,m]$ contains a position at least equal to B_n and, combined with Theorem 201, proves the claimed leftmost complexity.

The claim is direct for $n = 0$ as $B_0 = 1 = Q_{-1}+1$. Assume now that it is true for n and let us prove it for $n+1$. By construction, the B_{n+1}th letter of $\mathbf{b}$ is the last letter of the image of the B_nth letter under μ_k, and, by Proposition 198, $\mathbf{b}[B_{n+1}+1, Q_n+1]$ is the image of $\mathbf{b}[B_n+1, Q_{n-1}+1]$, potentially followed by a letter 0 (this occurs when $\mathbf{b}[B_n, Q_{n-1}+1]$ ends with the letter $k-1$). Therefore, each occurrence of $\mathbf{b}[B_{n+1}, Q_n+1]$ in $\mathbf{b}$ is associated with the image of an occurrence of $\mathbf{b}[B_n, Q_{n-1}+1]$. Using the induction hypothesis, we conclude that $\mathbf{b}[B_{n+1}, Q_n+1]$ does not occur before position B_{n+1}. □

10.1.4 Span complexity

For the k-bonacci words $\mathbf{b}^{(k)}$, the factor complexity function is given by $p_{\mathbf{b}^{(k)}}(n) = (k-1)n+1$. Therefore, when $k \geq 3$, Proposition 184 implies that the span complexity is linear. However, the string attractors described in Section 10.1.2 do not have the smallest difference between their extreme positions. In what follows, we compute the span for infinitely many prefixes and describe string attractors (of unbounded size) having that span.

We first make the following observation which gives a lower bound on the span. Recall that we have set $Q_{-1}^{(k)} = 0$.

Proposition 204. *Let $k \geq 2$. For all $n \geq 2$, the factors $\mathbf{b}^{(k)}[i, i+Q_{n-3}^{(k)}]$ are distinct for all $i \in [B_{n-2}^{(k)}+1, B_n^{(k)}]$.*

Proof. Let us prove the result by induction on n. For $n = 2$, we need to consider the letters in $u = \mathbf{b}[2, B_2]$. If $k = 2$, then $u = 10$ and if $k \geq 3$, then $u = 102$ so all the letters are indeed distinct.

Let us now assume that the claim is true for $n \geq 2$ and let us prove it for $n+1$. We proceed by contradiction and assume that there exist $i, j \in [B_{n-1}+1, B_{n+1}]$ minimal such that $i < j$ and $\mathbf{b}[i, i+Q_{n-2}] = \mathbf{b}[j, j+Q_{n-2}]$. As $B_{n-1}+1$ marks the beginning of the image of a letter in $\mathbf{b}$ and i and j are taken minimal, we know that the factor $u = \mathbf{b}[i, i+Q_{n-2}] = \mathbf{b}[j, j+Q_{n-2}]$ begins with 0. We may assume that it also does not end with 0. Indeed, otherwise, we consider the word $u = \mathbf{b}[i, i+Q_{n-2}-1] = \mathbf{b}[j, j+Q_{n-2}-1]$ instead.

As the word u starts with 0, there exist $i' < j'$ such that $\mu_k(\mathbf{b}[1, i'-1]) = \mathbf{b}[1, i-1]$ and $\mu_k(\mathbf{b}[1, j'-1]) = \mathbf{b}[1, j-1]$. Moreover, as $Q_{n-2}+1 \geq 2$ and as u does not end with a 0, it can be uniquely desubstituted (i.e., its preimage under μ_k is unique). There thus exists ℓ such that $\mathbf{b}[i', i'+\ell] = \mathbf{b}[j', j'+\ell]$ and $\mu_k(\mathbf{b}[i', i'+\ell]) = u$.

As $|\mu_k(\mathbf{b}[1, i'-1])| = i-1 \in [B_{n-1}, B_{n+1}-1]$, we have $i' \in [B_{n-2}+1, B_n]$. The same holds for j'. Therefore, by the induction hypothesis, we have $\mathbf{b}[i', i'+Q_{n-3}] \neq \mathbf{b}[j', j'+Q_{n-3}]$. Let us take $\ell' \in [\ell, Q_{n-3}-1]$ maximal such that $\mathbf{b}[i', i'+\ell'] = \mathbf{b}[j', j'+\ell']$ and let us denote $v = \mathbf{b}[i', i'+\ell']$. By maximality of ℓ', v is right-special. Moreover, the set of factors of $\mathbf{b}$ is stable under reversal [38, Theorem 5], i.e., the reversal of any factor of $\mathbf{b}$ is also a factor. In particular, v^R is a left-special factor of $\mathbf{b}$.

Furthermore, the left-special factors of $\mathbf{b}$ are exactly its prefixes [38, Proposition 5], so v^R is a prefix of $\mathbf{b}$ and also of $\mathbf{b}[1, Q_{n-3}]$ as $\ell' \leq Q_{n-3} - 1$. However, we have

$$|\mu_k(v^R)| = |\mu_k(v)| \geq |\mu_k(\mathbf{b}[i', i' + \ell])| \geq Q_{n-2}$$

by definition of ℓ. This is a contradiction as, by Proposition 198, we have $|\mu_k(v^R)| \leq |\mu_k(\mathbf{b}[1, Q_{n-3}])| < Q_{n-2}$. □

We now describe a new string attractor for prefixes of the k-bonacci word.

Proposition 205. *Let $k \geq 2$. For all $n \geq 1$ and for all $m \in [Q_{n-1}^{(k)} + 1, Q_n^{(k)}]$, $\Gamma_n = \{Q_{n-2}^{(k)} + 1, Q_{n-2}^{(k)} + 2, \ldots, B_n^{(k)}\}$ is a string attractor of $\mathbf{b}^{(k)}[1, m]$.*

Proof. We proceed by induction on $n \geq 1$. For the base case $n = 1$, the interval $[Q_{1-1} + 1, Q_2]$ becomes $[2, 3]$, and $\Gamma_1 = \{1, 2\}$, so the conclusion follows.

Now assume that the result is true for $n \geq 1$ and we show it also holds for $n + 1$. To do so, we will use the following observation. From Proposition 198 and [3, Proposition 4.4], one may prove that $\mathbf{b}[1, Q_n]$ is a palindrome for all $n \geq -1$. By the induction hypothesis, Γ_n is a string attractor for $\mathbf{b}[1, Q_n]$. As this word is a palindrome, it also has the string attractor

$$\Gamma_n^R = \{Q_n + 1 - B_n, \ldots, Q_n + 1 - Q_{n-2} - 1\} = \{Q_{n-1} + 1, \ldots, B_n + B_{n-1}\}.$$

In particular, $\Gamma_{n+1} \supseteq \Gamma_n^R$ is a string attractor of $\mathbf{b}[1, Q_n]$ when $B_{n+1} \leq Q_n$. If $B_{n+1} > Q_n$, then $n \leq k - 1$ and $B_{n+1} = Q_n + 1$, so Γ_{n+1} is a string attractor of $\mathbf{b}[1, Q_n + 1]$. In both cases, Propositions 181 and 198 imply that Γ_{n+1} is a string attractor of $\mathbf{b}[1, m]$ for all $m \in [Q_n + 1, Q_{n+1}]$. □

Corollary 206. *Let $k \geq 2$. For all $n \geq 2$ and for all $m \in [Q_n^{(k)} - B_{n-1}^{(k)} - B_{n-2}^{(k)}, Q_n^{(k)}]$, we have $\mathsf{span}_{\mathbf{b}^{(k)}}(m) = B_n^{(k)} - Q_{n-2}^{(k)} - 1$. In particular, for infinitely many prefixes, the bound given by Proposition 130 is tight for well-chosen lengths of factors.*

Proof. Using Propositions 204 and 130, we know that for $m \geq B_n + Q_{n-3}$, we have $\mathsf{span}_{\mathbf{b}}(m) \geq B_n - B_{n-2} - Q_{n-3} - 1 = B_n - Q_{n-2} - 1$. Observe that $B_n + Q_{n-3} = Q_n - B_{n-1} - B_{n-2}$. On the other hand, using Proposition 205, we know that for $m \in [Q_{n-1} + 1, Q_n]$, we have $\mathsf{span}_{\mathbf{b}}(m) \leq B_n - Q_{n-2} - 1$.

If $k \geq 3$, then $B_n + Q_{n-3} \geq Q_{n-1} + 1$ so, for all $m \in [Q_n - B_{n-1} - B_{n-2}, Q_n]$, we have $\mathsf{span}_{\mathbf{b}}(m) = B_n - Q_{n-2} - 1$, as desired. It remains to consider $k = 2$. In that case, $B_n - Q_{n-2} - 1 = 1$ for all n therefore $\mathsf{span}_{\mathbf{b}}(m) \geq 1$ for all $m \leq 3$ and $\mathsf{span}_{\mathbf{b}}(m) \leq 1$ for all $m \leq 2$. Therefore, the conclusion follows for all $m \geq 3$. □

Observe that, for the Fibonacci word, we once again obtain that $\mathsf{span}_{\mathbf{b}^{(2)}} = 1$, as in Theorem 185.

10.2 General Case of Fixed-Points of k-Bonacci-like Morphisms

In this section, we generalize some of the results obtained for the string attractor profile function in Section 10.1 to the larger family of words which are fixed points of k-bonacci like morphisms.

To do so, we first introduce in Subsection 10.2.1 some notions related to Combinatorics on Words that will be used throughout the chapter, and we further describe the structure and properties of the fixed points of the k-bonacci like morphisms.

Then, in Subsection 10.2.2 we introduce definitions from the theory of Numeration Systems [115], which will be used in Subsection 10.2.3 to obtain a greedy procedure to compute a string attractor of bounded size for any prefix of the words considered.

10.2.1 k-Bonacci like Words and Morphisms

A celebrated result in combinatorics on words is that Lyndon words form a so-called *complete factorization of the free monoid*.

Theorem 207 (Chen-Fox-Lyndon [29])**.** *For every non-empty word $w \in A^*$, there exists a unique factorization $(\ell_1, \cdots, \ell_n)$ of w into Lyndon words over A such that $\ell_1 \geq \ell_2 \geq \cdots \geq \ell_n$.*

Several variations of Lyndon words have been considered lately: generalized Lyndon [114], anti-Lyndon [52], inverse Lyndon [14], and Nyldon [28]. In this text, we will use the second.

Definition 208. *Let $(A, \leq)$ be a totally ordered alphabet. We let $\leq_-$ denote the* inverse order *on A, i.e., $b <_- a$ if and only if $a < b$ for all $a, b \in A$. We also let $\leq_-$ denote the*

TABLE 10.2: Construction of the sequences $(u_n)_{n\geq 0}$ and $(U_n)_{n\geq 0}$ for $c = 102$

n	0	1	2	3	4	5
u_n	0	01	012	01200	012000101	012000101012012
fact. of u_n	0	$u_0^1 \cdot 1$	$u_1^1 u_0^0 \cdot 2$	$u_2^1 u_1^0 u_0^2$	$u_3^1 u_2^0 u_1^2$	$u_4^1 u_3^0 u_2^2$
U_n	1	2	3	5	9	15

inverse lexicographic order *which is the lexicographic order induced by* $\leq_-$. *A word is* anti-Lyndon *if it is Lyndon with respect to the inverse lexicographic order.*

Otherwise stated, a word is anti-Lyndon if it is primitive and lexicographically maximal among its conjugates.

Example 209. *Let* $A = \{0, 1\}$ *with* $0 < 1$, *so* $1 <_- 0$. *The first few anti-Lyndon words, ordered by length, are* 1, 0, 10, 110, 100, 1110, 1100, *and* 1000.

In this section, we consider a specific family of morphisms defined as follows. Note that they appear under the name *generic k-bonacci* morphisms in [115, Example 2.11].

Definition 210. *Let* $k \geq 2$ *be an integer and let* $c_0, \ldots, c_{k-1} \in \mathbb{N}$ *be* k *parameters often summarized in the shape of a word* $c = c_0 \cdots c_{k-1} \in \mathbb{N}^k$. *The morphism* $\mu_c : \{0, \ldots, k-1\}^* \to \{0, \ldots, k-1\}^*$ *is given by* $\mu_c(i) = 0^{c_i} \cdot (i+1)$ *for all* $i \in \{0, \ldots, k-2\}$ *and* $\mu_c(k-1) = 0^{c_{k-1}}$. *For all* $n \geq 0$, *we then define* $u_{c,n} = \mu_c^n(0)$ *and* $U_{c,n} = |u_{c,n}|$. *Moreover, we refer with* k-bonacci like morphism *and* k-bonacci like finite word *to the morphism* μ_c *and the words* u_n *respectively, for every* $k \geq 2$, $n \geq 0$, *and* $c \in \mathbb{N}^k$.

When the context is clear, we will usually omit the subscript c in Definition 210.

Example 211. *When* $c = 1^k$, *we recover the k-bonacci morphism and words. For* $k = 3$ *and* $c = 102$, *the first few iterations of the corresponding morphism* $\mu_c \colon 0 \mapsto 01, 1 \mapsto 2, 2 \mapsto 00$ *are given in Table 10.2. Some specific factorization of the words* $(u_{c,n})_{n\geq 0}$ *is highlighted in Table 10.2.*

The factorization presented in the previous example can be stated in general. It gives a recursive definition of the words $(u_{c,n})_{n\geq 0}$ and can be proven using a simple induction.

Proposition 212. *Given an integer* $k \geq 2$, *let* $c \in \mathbb{N}^k$, *and let* $\mu_c : [0, k-1]^* \mapsto [0, k-1]^*$ *and* u_n, *for all* $n > 0$, *be the associated morphism and words from Definition 210. For all*

$c = c_0 \cdots c_{k-1} \in \mathbb{N}^k$ and all $n \geq 0$, we have

$$u_n = \begin{cases} \left(\prod_{i=0}^{n-1} u_{n-i-1}^{c_i} \right) \cdot n, & \text{if } n \leq k-1; \\ \prod_{i=0}^{k-1} u_{n-i-1}^{c_i}, & \text{if } n \geq k. \end{cases}$$

As a consequence of Proposition 212, the sequence $(U_n)_{n \in \mathbb{N}}$ respects the following recurrence relation: if $0 \leq n \leq k-1$, then $U_n = 1 + \sum_{i=0}^{n-1} c_i U_{n-i-1}$, and if $n \geq k$, then $U_n = \sum_{i=0}^{k-1} c_i U_{n-i-1}$.

In the rest of the paper, we will assume the following working hypothesis (WH) on c:

$$c = c_0 \cdots c_{k-1} \in \mathbb{N}^k \text{ with } c_0, c_{k-1} \geq 1. \tag{WH}$$

The condition $c_{k-1} \geq 1$ ensures both that the recurrence relation is of order k and that the morphism μ_c is non-erasing, which is a classical assumption in combinatorics on words. Moreover, the condition $c_0 \geq 1$ guarantees that μ_c is prolongable. Under (WH), the morphism μ_c has an infinite fixed point starting with 0 denoted $\mathbf{u} := \lim_{n \to \infty} u_n$.

We make the following combinatorial observation.

Remark 213. *Under (WH), using Proposition 212, a simple induction shows that the letter $1 \leq i \leq k-1$ can only be followed by 0 and/or $i+1$ (and only 0 in the case $i = k-1$) in $\mathbf{u}$.*

10.2.2 Fun with Numeration Systems

In this subsection, specific definitions will be recalled. For the reader unfamiliar with the theory of numeration systems, we refer to [12, Chapter 2] for an introduction and some advanced concepts.

A *numeration system* (for natural numbers) can be defined as a triple $\mathcal{S} = (A, \mathrm{rep}_{\mathcal{S}}, L)$, where A is an alphabet and $\mathrm{rep}_{\mathcal{S}} : \mathbb{N} \to A^*$ is an injective function such that $L = \mathrm{rep}_{\mathcal{S}}(\mathbb{N})$. The map $\mathrm{rep}_{\mathcal{S}}$ is called the *representation function* and L is the *numeration language*. If $\mathrm{rep}_{\mathcal{S}}(n) = w$ for some integer $n \in \mathbb{N}$ and some word $w \in A^*$, we say that w is the *representation (in $\mathcal{S}$)* of n and we define the *valuation (in $\mathcal{S}$)* of w by $\mathrm{val}_{\mathcal{S}}(w) = n$. Note that, when the context is clear, we omit the subscript $\mathcal{S}$ in rep and val.

TABLE 10.3: Illustration of the numeration system $\mathcal{S}_c$ for $c = 102$

n	0	1	2	3	4	5	6	7	8
$\mathbf{u}[0,n)$	ε	0	01	012	$012 \cdot 0$	01200	$01200 \cdot 0$	$01200 \cdot 01$	$01200 \cdot 01 \cdot 0$
$\mathrm{rep}_{\mathcal{S}_c}(n)$	ε	1	10	100	101	1000	1001	1010	1011

Any given prolongable morphism naturally gives rise to a numeration system that we will call the *associated Dumont-Thomas numeration system* [39]. These are based on particular factorizations of the prefixes of the fixed point. Here we give only the definition of the for the case of morphisms treated in this chapter.

Proposition 214 ([39]). *Let c satisfy (WH). For all $n \in \mathbb{N}$, there exist unique integers $N, \ell_0, \ldots, \ell_N \in \mathbb{N}$ such that $\ell_0 \geq 1$, $\mathbf{u}[0,n) = {u_N}^{\ell_0} \cdots u_0^{\ell_N}$, and this factorization verifies the following: u_{N+1} is not a prefix of $\mathbf{u}[0,n)$ and, for all $0 \leq i \leq N$, $u_N^{\ell_0} \cdots u_{N-i+1}^{\ell_{i-1}} u_{N-i}^{\ell_i+1}$ is not a prefix of $\mathbf{u}[0,n)$.*

Recall that a numeration system based on a suitable sequence of integers $(U_n)_{n\geq 0}$ is called *greedy* when, at each step of the decomposition of any integer, the largest possible term of the sequence $(U_n)_{n\geq 0}$ is chosen; formally, we use the Euclidean algorithm in a greedy way. As the conditions on the factorization in the previous proposition resemble that of greedy representations in numeration systems, we will refer to it as being *word-greedy*.

For a given c satisfying (WH), we then let $\mathcal{S}_c$ denote the numeration system associated with the representation function $\mathrm{rep}_{\mathcal{S}_c} : \mathbb{N} \to \mathbb{N}^*$ mapping n to $\mathrm{rep}_{\mathcal{S}_c}(n) = \ell_0 \cdots \ell_N$, where the integers $\ell_0, \ldots, \ell_N$ verify the conditions of Proposition 214 for n. By convention, we set $\mathrm{rep}_{\mathcal{S}_c}(0) = \varepsilon$.

Example 215. *Using Example 211 for $c = 102$, the representations of the first few integers are given in Table 10.3. The word-greedy factorization of each prefix is highlighted in the second row, leading to the representation of the corresponding integer in the third row.*

Remark 216. *If $\mathrm{rep}_{\mathcal{S}_c}(n) = \ell_0 \cdots \ell_N$, then $n = |u_{c,N}^{\ell_0} \cdots u_{c,0}^{\ell_N}| = \sum_{i=0}^{N} \ell_i U_{c,N-i}$. In other words, $\mathrm{val}_{\mathcal{S}_c}$ is given by the usual valuation function associated with the sequence $(U_{c,n})_{n\in\mathbb{N}}$. Such a system is sometimes called a* positional *numeration system. Note that this is not necessarily the case for the Dumont-Thomas numeration system associated with some other morphism.*

The Dumont-Thomas numeration systems are a particular case of abstract numeration systems introduced in [80]. A numeration system $\mathcal{S} = (A, \mathrm{rep}, L)$ is said

to be *abstract* if L is regular and $\mathrm{rep}(n)$ is the $(n+1)$st word of L in the genealogical order. We have the following result.

Theorem 217 (Rigo [115, Section 2.2]). *Let $\sigma : \{\alpha_0, \ldots, \alpha_d\}^* \to \{\alpha_0, \ldots, \alpha_d\}^*$ be a morphism prolongable on the letter α_0. We define the automaton $\mathcal{A}_\sigma$ for which $\{\alpha_0, \ldots, \alpha_d\}$ is the set of states, α_0 is the initial state, every state is final, and the (partial) transition function δ is such that, for each $\alpha \in \{\alpha_0, \ldots, \alpha_d\}$ and $0 \leq i \leq |\sigma(\alpha)| - 1$, $\delta(\alpha, i)$ is the $(i+1)$st letter of $\sigma(\alpha)$. If $\mathcal{S} = (A, \mathrm{rep}, L)$ is the Dumont-Thomas numeration system associated with σ, then $L = L(\mathcal{A}_\sigma) \setminus 0\mathbb{N}^*$ and $\mathrm{rep}(n)$ is the $(n+1)$st word of L in the genealogical order.*

Example 218. *For $c = 102$, the automaton $\mathcal{A}_{\mu_c}$ of Theorem 217 is depicted in Figure 10.3 (details are left to the reader). The first few accepted words (not starting with 0) are, in genealogical order, ε, 1, 10, 100, 101, 1000, 1001, 1010, and 1011, which indeed agree with the representations of the first few integers in Example 215.*

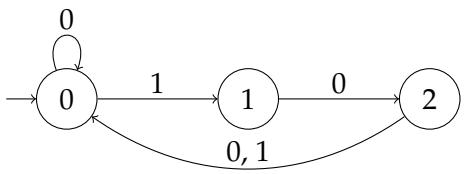


FIGURE 10.3: The automaton $\mathcal{A}_{\mu_c}$ for $c = 102$

As the automaton in Theorem 217 can be used to produce, for all $n \geq 0$, the letter $\mathbf{u}_n$ when reading $\mathrm{rep}_{\mathcal{S}_c}(n)$ by [115, Theorem 2.24], we have the following.

Corollary 219. *Let c satisfy (WH). Then the sequence $\mathbf{u}$ is $\mathcal{S}_c$-automatic.*

Similarly to what is usually done in real base numeration systems, we will let $\mathbf{d}^\star$ denote the periodization of c, that is, $\mathbf{d}^\star = (c_0 \cdots c_{k-2}(c_{k-1} - 1))^\omega$. Using Theorem 217, we deduce the next result.

Lemma 220. *Under (WH), for all $n \geq 0$, we have $\mathrm{rep}_{\mathcal{S}_c}(U_n) = 10^n$, the numbers having a representation of length $n+1$ are those in $[U_n, U_{n+1})$, and $\mathrm{rep}_{\mathcal{S}_c}(U_{n+1} - 1) = \mathbf{d}^\star[0, n]$. In particular, $U_{n+1} - 1 = \sum_{i=0}^{n} \mathbf{d}_i^\star U_{n-i}$.*

Proof. The first claim directly follows by the definition of $\mathcal{S}_c$, and the second one by the genealogical order. The number $U_{n+1} - 1$ is then represented by the maximal length-$(n+1)$ word accepted by the automaton $\mathcal{A}_{\mu_c}$, which is the length-$(n+1)$ prefix of $\mathbf{d}^\star$. □

Note that, if the numeration system $\mathcal{S}_c$ satisfies the greedy condition, this result follows from the characterization of numeration systems in terms of dynamical systems given by Bertrand-Mathis [13, 27]. However, even though the function $\mathrm{rep}_{\mathcal{S}_c}$ is obtained using the word-greedy factorization of prefixes of $\mathbf{u}$, the numeration system $\mathcal{S}_c$ is not necessarily greedy as the following example shows.

Example 221. *In Example 211 for $c = 102$, we see that $\mathbf{u}[0, 14) = 012000101 \cdot 012 \cdot 01$, so $\mathrm{rep}_{\mathcal{S}_c}(14) = 10110$, while the greedy representation of 14 associated with the sequence $(U_n)_{n\in\mathbb{N}}$ is 11000.*

In fact, we have the following two characterizations.

Lemma 222. *Let c satisfy (WH). The numeration system $\mathcal{S}_c = (A, \mathrm{rep}_{\mathcal{S}_c}, L)$ is greedy if and only if, for all $v \in L$ and for all $i \leq |v|$, the suffix of length i of v is smaller than or equal to $\mathbf{d}^\star[0, i)$. Moreover, we then have*

$$L = \{v = v_1 \cdots v_n \in \mathbb{N}^* \setminus 0\mathbb{N}^* \mid \forall\, 1 \leq i \leq n, v_{n-i+1} \cdots v_n \leq \mathbf{d}^\star[0, i)\}.$$

Proof. Let us denote $\mathcal{S} = (A', \mathrm{rep}_{\mathcal{S}}, L')$ the canonical greedy numeration system associated with the sequence $(U_n)_{n\in\mathbb{N}}$. In particular, by uniqueness, $\mathcal{S}_c$ is greedy if and only if $\mathcal{S}_c = \mathcal{S}$. As $\mathcal{S}_c$ is an abstract numeration system, $\mathrm{rep}_{\mathcal{S}_c}$ respects the genealogical order, i.e., $n \leq m$ if and only if $\mathrm{rep}_{\mathcal{S}_c}(n) \leq_{\mathrm{gen}} \mathrm{rep}_{\mathcal{S}_c}(m)$. So does $\mathrm{rep}_{\mathcal{S}}$ by [12, Proposition 2.3.45]. Hence, $\mathcal{S}_c = \mathcal{S}$ if and only if $L = L'$. Moreover, for all $n \geq 0$, $\mathrm{rep}_{\mathcal{S}}(U_n) = 10^n$, so L and L' contain the same number of length-n words by Lemma 220. Thus $L = L'$ if and only if $L \subseteq L'$. The statement holds since, by [60, Lemma 5.3] and by Lemma 220, we have

$$L' = \{v = v_1 \cdots v_n \in \mathbb{N}^* \setminus 0\mathbb{N}^* \mid \forall\, 1 \leq i \leq n, v_{n-i+1} \cdots v_n \leq \mathbf{d}^\star[0, i)\}.$$

□

Theorem 223. *Let $c = c_0 \cdots c_{k-1} \in \mathbb{N}^k$ with $c_0, c_{k-1} \geq 1$. The numeration system $\mathcal{S}_c$ is greedy if and only if $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is lexicographically maximal among its conjugates.*

Proof. Using Lemma 222 and Theorem 217, $\mathcal{S}_c$ is greedy if and only if, for all $n \in \mathbb{N}$ and for all $0 \leq i \leq k-1$, any path $\ell_0 \cdots \ell_n$ starting in State i in the automaton $\mathcal{A}_{\mu_c}$

is such that $\ell_0 \cdots \ell_n \leq \mathbf{d}^\star[0,n]$. However, by definition of $\mathcal{A}_{\mu_c}$, the lexicographically biggest path of length n starting in state i is given by the prefix of length n of $(c_i \cdots c_{k-2}(c_{k-1}-1)c_0 \cdots c_{i-1})^\omega$. We can therefore conclude that $\mathcal{S}_c$ is greedy if and only if $c_i \cdots c_{k-2}(c_{k-1}-1)c_0 \cdots c_{i-1} \leq c_0 \cdots c_{k-2}(c_{k-1}-1)$ for all $0 \leq i \leq k-1$, i.e., $c_0 \cdots c_{k-2}(c_{k-1}-1)$ is maximal among its conjugates. □

Observe that the condition of the previous result is equivalent to the fact that $c_0 \cdots c_{k-2}(c_{k-1}-1) = v^\ell$ for some anti-Lyndon v (in fact, v is the primitive root).

Example 224. *Let $k = 4$ and $c = 1011$. In this case, $c_0c_1c_2(c_3-1) = 1010 = v^2$ with $v = 10$, which is anti-Lyndon (see Example 209). The sequence U_n satisfies the recurrence relation $U_{n+4} = U_{n+3} + U_{n+1} + U_n$ with initial conditions $U_0 = 1$, $U_1 = 2$, $U_2 = 3$, and $U_3 = 5$. A simple induction shows that $(U_n)_{n\in\mathbb{N}}$ is in fact the sequence of Fibonacci numbers. Therefore the numeration system $\mathcal{S}_c$ corresponds to the classical Fibonacci numeration system, which can also be obtained with the parameter $c = 11$.*

The observation made in the previous example is more general.

Remark 225. *Let c satisfy (WH). If $c_0 \cdots c_{k-2}(c_{k-1}-1) = v^\ell$ with v anti-Lyndon, we define the word $v' := v_1 \cdots v_{|v|-1}(v_{|v|}+1)$ (simply put, we add 1 to the last letter of v). Then $c = v^{\ell-1}v'$ is a "partial" cyclization of v'. In particular, since $\mathbf{d}^\star_c = \mathbf{d}^\star_{v'}$ (where the dependence of $\mathbf{d}^\star$ on the chosen parameters is emphasized via a subscript), the numeration systems $\mathcal{S}_c$ and $\mathcal{S}_{v'}$ coincide by Lemma 220.*

For the reader familiar with the general theory of numerations, v' satisfies $v'_i \cdots v'_{|v|} < v'$ for all indices $i \in \{2,\ldots,|v|\}$. This implies that v' is the β-expansion $d_\beta(1)$ of 1 for a simple Parry number β [107]. Therefore, c is also a representation of 1 in base β.

Example 226. *We illustrate the previous remark by resuming Example 224. We have $v = 10$ and $v' = 11$. The corresponding simple Parry number is the Golden ratio φ. Observe that indeed $c = vv' = 1011$ is a representation of 1 in base φ.*

10.2.3 Link to String Attractors

Using the results and concepts of the previous sections, we now turn to the concept of string attractors in relation to the fixed points of the morphisms μ_c, $c \in \mathbb{N}^k$. We

recall that a *string attractor* of a finite word $w \in A^n$ is a set $\Gamma \subseteq [1, n]$ of positions such that every factor of w has an occurrence crossing a position in Γ.

Example 227. *The set* $\{2, 3, 4\}$ *is a string attractor of the word* $0\underline{1}\underline{2}\underline{0}01$. *Indeed, it suffices to check that the factors* 0, 1 *and* 01 *have an occurrence crossing one of the underlined positions. No smaller string attractor exists since at least one position in the set is needed per different letter in the word.*

Given an infinite word $\mathbf{x}$ and any integer $n \geq 1$, we let $s_{\mathbf{x}}(n)$ denote the size of a smallest string attractor for the length-n prefix of $\mathbf{x}$. The function $s_{\mathbf{x}} \colon n \mapsto s_{\mathbf{x}}(n)$ is called the *string attractor profile function* of $\mathbf{x}$ [119].

Warning. We would like to stress the following crucial point: in this section of the thesis, the letters of infinite words are indexed starting from 0 while the positions in a string attractor are counted starting at 1. This could be seen as confusing, but here we use standard notation for infinite words in the field of Combinatorics on Word. Where ambiguity may occur, we explicitly declare how finite words are indexed.

As we will look at prefixes of infinite words, it is natural to wonder if there is a link between the string attractors of the finite words w and wa, where a is a letter. In general, there is no trivial link although we have the result from Proposition 181 on the string attractor for fractional powers.

Since the considered infinite words are the limits of the sequence $(u_n)_{n\in\mathbb{N}}$, we are interested in the prefixes which are fractional powers of some u_n.

Definition 228. *Let c satisfy* (WH). *For all $n \geq 0$, we let q_n denote the longest prefix of $\mathbf{u}$ that is a fractional power of u_n, i.e., the longest common prefix between $\mathbf{u}$ and $(u_n)^\omega$. For all $n \geq 0$, we also let $Q_n = |q_n|$.*

The words defined above have a particular structure as stated below.

Proposition 229. *Let c satisfy* (WH). *Define $\mathbf{a}$ as the infinite concatenation of the longest anti-Lyndon prefix of the word $c_0 \cdots c_{k-2}$. Then for all $n \geq 0$, $q_n = u_n^{\mathbf{a}_0} u_{n-1}^{\mathbf{a}_1} \cdots u_0^{\mathbf{a}_n}$. In particular, $Q_n = \sum_{i=0}^{n} \mathbf{a}_i U_{n-i}$.*

Example 230. *Let us pursue Example 211 for which $c = 102$. The first few words in $(q_n)_{n\geq 0}$ are* $0, 01, 0120, 0120001, 0120001010120$. *The longest anti-Lyndon prefix of $c_0c_1 = 10$ is* 10 *itself so $\mathbf{a} = (10)^\omega$. We can easily check that the first few q_n's indeed satisfy Proposition 229.*

Lemma 231. *Let c satisfy (WH). Then $c_0 \cdots c_{k-2} \geq \mathbf{a}[0, k-2]$.*

Proof. Assume the contrary and let w be the longest anti-Lyndon prefix of $c_0 \cdots c_{k-2}$. If $|w| \leq i \leq k-2$ is the smallest index such that $c_0 \cdots c_i < \mathbf{a}[0, i]$, then $c_0 \cdots c_i = w^\ell va$ with v a proper prefix of w, a a letter, and $va < w$. So [41, Lemme 2] implies that $c_0 \cdots c_i$ is an anti-Lyndon prefix of $c_0 \cdots c_{k-2}$. As $i \geq |w|$, this contradicts the definition (maximality) of w. □

We now show how the condition obtained for the greediness of the numeration system is related to the notions we have just defined.

Proposition 232. *Let c satisfy (WH). If $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is lexicographically maximal among its conjugates, then $\mathbf{d}^\star[0, n] \leq \mathbf{a}[0, n]$ for all $n \geq 0$.*

Proof. Let w denote the longest anti-Lyndon prefix of $c_0 \cdots c_{k-2}$. We first show that $c_0 \cdots c_{k-2}(c_{k-1} - 1) \leq \mathbf{a}[0, k-1]$. If it is not the case, there exist $\ell \geq 1$, a proper prefix u of w, a letter a and a word v such that $c_0 \cdots c_{k-2}(c_{k-1} - 1) = w^\ell uav$ and $ua > w$. Then $uavw^\ell > c_0 \cdots c_{k-2}(c_{k-1} - 1)$, so $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is not maximal among its conjugates. This is a contradiction. Therefore we have $c_0 \cdots c_{k-2}(c_{k-1} - 1) \leq \mathbf{a}[0, k-1]$. By Lemma 231, we get $c_0 \cdots c_{k-2} = \mathbf{a}[0, k-2]$ and $c_{k-1} - 1 \leq \mathbf{a}_{k-1}$.

We now prove that $\mathbf{d}^\star[0, n] \leq \mathbf{a}[0, n]$ for all $n \geq 0$. If $c_{k-1} - 1 < \mathbf{a}_{k-1}$, then the conclusion is direct. If $c_{k-1} - 1 = \mathbf{a}_{k-1}$, then $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is a fractional power of w so there exist $\ell \geq 1$ and u a proper prefix of w such that $c_0 \cdots c_{k-2}(c_{k-1} - 1) = w^\ell u$. Let us write $w = uv$. If $u \neq \varepsilon$, we then have that $c_0 \cdots c_{k-2}(c_{k-1} - 1) = w^\ell u = u(vu)^\ell < uw^\ell$ as w is anti-Lyndon thus strictly greater than its conjugates. This contradicts the assumption that $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is maximal among its conjugates. Therefore, $u = \varepsilon$ and $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is a (natural) power of w. We conclude that $\mathbf{a} = \mathbf{d}^\star$, which ends the proof of the first item. □

Proposition 233. *Let c satisfy (WH). If $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is lexicographically maximal among its conjugates, then $U_{n+1} - 1 \leq Q_n$ for all $n \geq 0$.*

Proof. Let us show the claim by contraposition. So assume that there exists an integer n such that $U_{n+1} - 1 > Q_n$. Thus $q_n = u_n^{\mathbf{a}_0} \cdots u_0^{\mathbf{a}_n}$ is a proper prefix of $\mathbf{u}[0, U_{n+1} - 1)$. By Lemma 220, $\mathrm{rep}_{\mathcal{S}_c}(U_{n+1} - 1) = \mathbf{d}^\star[0, n]$, so $\mathbf{d}_0^\star$ is the largest exponent e such that u_n^e is a prefix of $\mathbf{u}[0, U_{n+1} - 1)$. This implies that $\mathbf{d}_0^\star \geq \mathbf{a}_0$. Moreover, if $\mathbf{a}_0 = \mathbf{d}_0^\star$, the

same argument implies that $\mathbf{d}_1^\star$ is the largest exponent e such that $u_n^{\mathbf{d}_0^\star} u_{n-1}^e$ is a prefix of $\mathbf{u}[0, U_{n+1}-1)$. In both cases, we have $\mathbf{d}_0^\star \mathbf{d}_1^\star \geq \mathbf{a}_0\mathbf{a}_1$. We may iterate the reasoning to obtain $\mathbf{d}^\star[0,n] \geq \mathbf{a}[0,n]$. As q_n is a proper prefix of $\mathbf{u}[0, U_{n+1}-1)$, the inequality cannot be an equality. This contradicts Proposition 232, which ends the proof. □

We will now prove that, under the conditions of the previous result, we can describe string attractors of every prefix of $\mathbf{u}$ using the elements of $(U_n)_{n\in\mathbb{N}}$. For $n \in \mathbb{N}$, we let Γ_n denote $\{U_0, \ldots, U_n\}$ if $0 \leq n \leq k-1$, $\{U_{n-k+1}, \ldots, U_n\}$ otherwise. We also define P_n by U_n if $0 \leq n \leq k-1$, $U_n + U_{n-k+1} - U_{n-k} - 1$ otherwise.

The next lemma directly follows from Proposition 233 and the definition of P_n.

Lemma 234. *Let c satisfy (WH). If $c_0 \cdots c_{k-2}(c_{k-1}-1)$ is maximal among its conjugates, then $P_n \leq U_{n+1} - 1 \leq Q_n$ for all $n \in \mathbb{N}$.*

To simplify the statement of the following theorem, we set $\Gamma_{-1} = \emptyset$.

Theorem 235. *Let $c = c_0 \cdots c_{k-1} \in \mathbb{N}^k$ with $c_0, c_{k-1} \geq 1$ and $c_0 \cdots c_{k-2}(c_{k-1}-1)$ maximal among its conjugates. Fix an integer $n \geq 0$. If $m \in [U_n, Q_n]$, then $\Gamma_{n-1} \cup \{U_n\}$ is a string attractor of $\mathbf{u}[0,m)$. Furthermore, if $m \in [P_n, Q_n]$, then Γ_n is a string attractor of $\mathbf{u}[0,m)$. In particular, $s_{\mathbf{u}}(n) \leq k+1$ for all $n \geq 1$.*

Proof. Let us simultaneously prove the two claims by induction on n. If $n = 0$, then $1 \leq m \leq c_0$, so $\mathbf{u}[0,m) = 0^m$ and the conclusion directly follows for both claims. Assume now that the claims are satisfied for $n-1$ and let us prove them for n. By Lemma 234 and the induction hypothesis, Γ_{n-1} is a string attractor of $\mathbf{u}[0, U_n - 1)$. This implies that $\Gamma_{n-1} \cup \{U_n\}$ is a string attractor of u_n so, by Proposition 181 and by definition of Q_n (Definition 228), of $\mathbf{u}[0,m)$ for all $m \in [U_n, Q_n]$. This ends the proof of the first claim.

Let us now prove the second claim. Observe that, using Proposition 181, it suffices to prove that Γ_n is a string attractor of $\mathbf{u}[0, P_n)$. If $0 \leq n \leq k-1$, then $\Gamma_n = \Gamma_{n-1} \cup \{U_n\}$ so we can directly conclude using the first claim. Thus assume that $n \geq k$. Then by the first claim, $\Gamma_n \cup \{U_{n-k}\} = \Gamma_{n-1} \cup \{U_n\}$ is a string attractor of $\mathbf{u}[0, P_n)$. Therefore, it remains to show that the position U_{n-k} is not needed in the string attractor. In other words, we prove that the factors of $\mathbf{u}[0, P_n)$ that have an

occurrence crossing position U_{n-k} (and no other position of $\Gamma_n \cup \{U_{n-k}\}$) have another occurrence crossing a position in Γ_n. More precisely, we show that they have an occurrence crossing position U_n. To help the reader with the proof, we illustrate the situation in Figure 10.4.

As the smallest position in Γ_n is U_{n-k+1}, we need to consider the factor occurrences crossing position U_{n-k} in $\mathbf{u}[0, U_{n-k+1} - 1)$. So, if we write $\mathbf{u}[0, P_n) = u_n w$, it is sufficient to show that u_{n-k} is a suffix of u_n and that $w' := \mathbf{u}[U_{n-k}, U_{n-k+1} - 1)$ is a prefix of w. Observe that

$$|w| = P_n - U_n = U_{n-k+1} - U_{n-k} - 1 \tag{10.1}$$

by definition of P_n, so $|w'| = |w|$. We will actually show that $w' = w$.

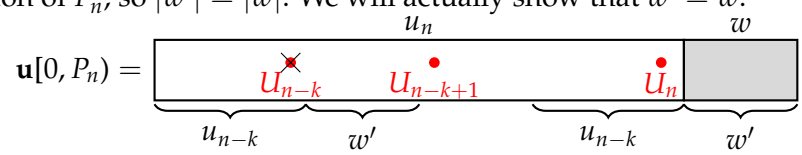


FIGURE 10.4: Representation of the proof of the second claim of Theorem 235. As we warned the reader before, elements in a string attractor are indexed starting at 1 (in red), while indices of letters in $\mathbf{u}$ start at 0.

The fact that u_{n-k} is a suffix of u_n is a direct consequence of Proposition 212 as $c_{k-1} \geq 1$ by assumption. To prove that $w' = w$, we first make the following observation: Proposition 212 again implies that u_n is followed by $u_n^{c_0-1} u_{n-1}^{c_1} \cdots u_{n-k+1}^{c_{k-1}}$ in $\mathbf{u}$. Since u_{n-k+1} is a prefix of all the words $u_{n-k+1}, \ldots, u_{n-1}$, the word u_n is in particular followed by u_{n-k+1} in $\mathbf{u}$. As $|w| \leq U_{n-k+1}$ by Equation (10.1), this implies that w is a prefix of u_{n-k+1}, so also of $\mathbf{u}$. To conclude with the claim, it is then enough to show that w' is also a prefix of $\mathbf{u}$. To prove this, we will use the numeration system $\mathcal{S}_c$ and consider two cases.

First, assume that $n - 2k + 1 \geq 0$. By definition of w' and by Proposition 212, w' is a prefix of $v := u_{n-k}^{c_0-1} \cdots u_{n-2k+1}^{c_{k-1}}$. Define the word $x = (c_0 - 1)c_1 \cdots c_{k-1} 0^{n-2k+1}$. If it begins with 0's, we consider instead the word obtained by removing the leading 0's. Note that x corresponds to a factorization of v into the words $u_{n-k}, \ldots, u_0$. As $c_0 \cdots c_{k-2}(c_{k-1} - 1)$ is maximal among its conjugates by assumption, x is in the numeration language by Lemma 222. By definition of $\mathcal{S}_c$, x is the Dumont-Thomas factorization of v, implying that v is a prefix of $\mathbf{u}$.

Second, if $n - 2k + 1 < 0$, then we conclude in a similar way by considering $v = u_{n-k}^{c_0-1} \cdots u_0^{c_{n-k}}$ and $x = (c_0 - 1)c_1 \cdots c_{n-k}$ instead. $\square$

The morphisms studied in Theorem 235 are associated with Parry numbers in the following way: there exists a simple Parry number β with $d_\beta(1) = c_0 \cdots c_{k-1}$ such that either $c = d_\beta(1)$, or $c = (c_0 \cdots c_{k-2}(c_{k-1} - 1))^\ell d_\beta(1)$ for some $\ell \geq 1$.

Theorem 235 implies that for some values of c (e.g., $c = 211$), we even have $s_{\mathbf{u}}(n) \leq k$ for all $n \geq 1$ and it is optimal for large n as every position in Γ_n covers a different letter (this can be proved using a simple induction).

Observe that the bounds given in the previous theorem are not necessarily tight. For example, if $c = 23$, then $\Gamma_2 = \{3, 9\}$ is a string attractor of the length-9 prefix $\mathbf{u}[0, 9) = 001001000$, while $P_2 = 10$. This is also the case for the k-bonacci morphisms ($c = 1^k$) where better bounds are provided in the firs section.

Chapter 11

Conclusions

In this thesis, we presented a qualitative study as repetitiveness measures on the measure r that counts the number of BWT-runs and the size γ^* of a smallest string attractor. Among the different measures considered in literature, the interest for r and γ^* is driven by two different reasons. The first one concerns the multitude of applications that the BWT has in Data Compression and Bioinformatics. The study on the notion of string attractor on the other hand, is more related to its ability to capture different properties of words in a very compact space.

For the measure r, in Chapter 3 we have extended the knowledge on its relationship with (circular) bispecial factors. Even though the upper bound is similar to the one proposed in [8], here we have presented the first lower bound in terms of the classical notions from Combinatorics on Words. We have also tested the sensitivity of the measure r to single bit operations on words, and proved that a *bit-catastrophe* can happen, that is $r(w) = \omega(r(w'))$, for some words w and w' such that $\mathsf{ed}(w, w') = 1$. Furthermore, we have extended the lower bound on the additive sensitivity of r from $\Omega(\log n)$ to $\Omega(\sqrt{n})$, reaching the same bounds of other measures considered by Akagi et al. [1].

Nonetheless, the contexts in which the Burrows-Wheeler Transform is mainly used confirms its efficiency on highly repetitive dataset. For binary purely morphic words, we have showed in Chapter 4 that we can not have more than a logarithmic number of BWT-runs, and in Chapter 5 we have measured the effect that a single application of some families of morphisms may have on any word. Indeed, it is worth investigating other relationships with properties of words, in order to find new artefacts aimed to maximize its efficiency.

In Chapter 7, we have presented the first combinatorial analysis on the measure γ^* of the smallest string attractor. The computability is in general a difficult task, and its non-monotone property complicates the case even when a solely insertion is considered. However, we have obtained multiple characterization based on string attractors through combinatorial arguments: standard Sturmian words via circular span in Chapter 8, and in Chapter 9 eventually periodic words via leftmost complexity, and infinite Sturmian and quasi-Sturmian words via span complexity. Finally, in Chapter 10 we have linked the notion of string attractor to other properties and notions typical from the Combinatorics on Words field.

To conclude, we believe that the concept of *plain text repetitiveness*, which is vague and open to different interpretations, is not something that can be uniquely represented by a number. Nonetheless, having a good approximated measure of "what we perceive repetitive", and that is easy to compute, is indeed useful. From this point of view, the measure δ probably outperform all the other definitions given so far in this thesis, especially since it possible to build indexes in a space proportional to their size. On the other hand, the notion of string attractor has proved to be able to capture some of these repetitiveness shades, more than δ (there are words that are indistinguishable by δ), and more than the other compressor-based measures (in a space asymptotically smaller). We wonder if there are other facets of string attractors that can be related to other properties of words.

11.1 Future Directions of Research

As possible future investigations, on the measure r we aim to find what properties of words trigger an increase and/or a decrease in the BWT-runs. This may help us to solve the open problems in the gaps left on the sensitivities of the measure r. Moreover, a deeper understanding on the relationship between properties of words and applications of morphisms may lead to new bounds between the BWT-runs and the size of L or NU-systems, expanding our knowledge in the field of Data Compression. Moreover, we set as goal to keep investigating on the study of r for collection of samples, with particular applications in pangenomics.

For string attractors, it is legit wonder if they *hide* other properties of words. For instance, we wonder if the size of the set of all minimum (or minimal) string attractor $\mathcal{G}' \subseteq \mathcal{G}$ of a word can denote in a symmetrical way the degree of repetitions in a text. Finally, on infinite words we aim to solve the open question on the sufficient conditions to have a bounded string attractor profile function. We further intend to understand if it is possible to build a smallest string attractors for the prefixes of purely morphic words, by using combinatorial arguments that relies on common and shared properties.

Bibliography

[1] Tooru Akagi, Mitsuru Funakoshi, and Shunsuke Inenaga. "Sensitivity of string compressors and repetitiveness measures". In: *Information and Computation* 291 (2023), p. 104999. ISSN: 0890-5401.

[2] Jean-Paul Allouche and Jeffrey Shallit. *Automatic Sequences: Theory, Applications, Generalizations*. Cambridge University Press, 2003.

[3] Petr Ambrož, Zuzana Masáková, Edita Pelantová, and Christiane Frougny. "Palindromic complexity of infinite words associated with simple Parry numbers". In: *Annales de l'Institut Fourier* 56.7 (2006), pp. 2131–2160.

[4] Golnaz Badkobeh, Sara Giuliani, Zsuzsanna Lipták, and Simon J. Puglisi. "On Compressing Collections of Substring Samples". In: *ICTCS*. Vol. 3284. CEUR Workshop Proceedings. CEUR-WS.org, 2022, pp. 136–147.

[5] Hideo Bannai, Mitsuru Funakoshi, Tomohiro I, Dominik Köppl, Takuya Mieno, and Takaaki Nishimoto. "A Separation of γ and b via Thue-Morse Words". In: *SPIRE*. Vol. 12944. Lecture Notes in Computer Science. Springer, 2021, pp. 167–178.

[6] Hideo Bannai, Keisuke Goto, Masakazu Ishihata, Shunsuke Kanda, Dominik Köppl, and Takaaki Nishimoto. "Computing NP-Hard Repetitiveness Measures via MAX-SAT". In: *ESA*. Vol. 244. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 12:1–12:16.

[7] Djamal Belazzougui and Fabio Cunial. "Fully-Functional Bidirectional Burrows-Wheeler Indexes and Infinite-Order De Bruijn Graphs". In: *CPM*. Vol. 128. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 10:1–10:15.

[8] Djamal Belazzougui, Fabio Cunial, Travis Gagie, Nicola Prezza, and Mathieu Raffinot. "Composite Repetition-Aware Data Structures". In: *CPM*. Vol. 9133. Lecture Notes in Computer Science. Springer, 2015, pp. 26–39.

[9] Djamal Belazzougui, Travis Gagie, Veli Mäkinen, Marco Previtali, and Simon J. Puglisi. "Bidirectional Variable-Order de Bruijn Graphs". In: *Int. J. Found. Comput. Sci.* 29.8 (2018), pp. 1279–1295.

[10] Jason W. Bentley, Daniel Gibney, and Sharma V. Thankachan. "On the Complexity of BWT-Runs Minimization via Alphabet Reordering". In: *ESA*. Vol. 173. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 15:1–15:13.

[11] Jean Berstel and Patrice Séébold. "A Characterization of Sturmian Morphisms". In: *MFCS*. Vol. 711. Lect. Notes Comput. Sci. Springer, 1993, pp. 281–290.

[12] Valérie Berthé and Michel Rigo, eds. *Combinatorics, Automata and Number Theory*. Vol. 135. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 2010. DOI: 10.1017/CBO9780511777653.

[13] Anne Bertrand-Mathis. "Comment écrire les nombres entiers dans une base qui n'est pas entière". In: *Acta Math. Hungar.* 54.3-4 (1989), pp. 237–241. DOI: 10.1007/BF01952053.

[14] Paola Bonizzoni, Clelia De Felice, Rocco Zaccagnino, and Rosalba Zizza. "Inverse Lyndon words and inverse Lyndon factorizations of words". In: *Adv. in Appl. Math.* 101 (2018), pp. 281–319. DOI: 10.1016/j.aam.2018.08.005.

[15] Jean-Pierre Borel and Christophe Reutenauer. "On Christoffel classes". In: *RAIRO-Theor. Inf. Appl.* 40.1 (2006), 15–27.

[16] Christina Boucher, Davide Cenzato, Zsuzsanna Lipták, Massimiliano Rossi, and Marinella Sciortino. "Computing the Original eBWT Faster, Simpler, and with Less Memory". In: *SPIRE*. Vol. 12944. Lecture Notes in Computer Science. Springer, 2021, pp. 129–142.

[17] Christina Boucher, Travis Gagie, Tomohiro I, Dominik Köppl, Ben Langmead, Giovanni Manzini, Gonzalo Navarro, Alejandro Pacheco, and Massimiliano Rossi. "PHONI: Streamed Matching Statistics with Multi-Genome References". In: *DCC*. IEEE, 2021, pp. 193–202.

[18] Christina Boucher, Travis Gagie, Alan Kuhnle, Ben Langmead, Giovanni Manzini, and Taher Mun. "Prefix-free parsing for building big BWTs". In: *Algorithms Mol. Biol.* 14.1 (2019), 13:1–13:15.

[19] Nicolas Bray and Lior Pachter. "MAVID: constrained ancestral alignment of multiple sequences". In: *Genome research* 14.4 (2004), pp. 693–699.

[20] Srecko Brlek, Andrea Frosini, Ilaria Mancini, Elisa Pergola, and Simone Rinaldi. "Burrows-Wheeler Transform of Words Defined by Morphisms". In: *IWOCA*. Vol. 11638. Lecture Notes in Computer Science. Springer, 2019, pp. 393–404.

[21] Michael Burrows and David J. Wheeler. *A block-sorting lossless data compression algorithm*. Tech. rep. DIGITAL System Research Center, 1994.

[22] Julien Cassaigne. "Sequences with grouped factors". In: *Developments in Language Theory*. Aristotle University of Thessaloniki, 1997, pp. 211–222.

[23] Giusi Castiglione, Antonio Restivo, and Marinella Sciortino. "Circular Sturmian words and Hopcroft's algorithm". In: *Theor. Comput. Sci.* 410.43 (2009), pp. 4372–4381.

[24] Davide Cenzato, Veronica Guerrini, Zsuzsanna Lipták, and Giovanna Rosone. "Computing the optimal BWT of very large string collections". In: (2023).

[25] Davide Cenzato and Zsuzsanna Lipták. "A Theoretical and Experimental Analysis of BWT Variants for String Collections". In: *CPM*. Vol. 223. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 25:1–25:18.

[26] Moses Charikar, Eric Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, Amit Sahai, and Abhi Shelat. "The smallest grammar problem". In: *IEEE Trans. Information Theory* 51.7 (2005), pp. 2554–2576.

[27] Émilie Charlier, Célia Cisternino, and Manon Stipulanti. "A Full Characterization of Bertrand Numeration Systems". In: *DLT*. Vol. 13257. Lecture Notes in Computer Science. Springer, 2022, pp. 102–114.

[28] Émilie Charlier, Manon Philibert, and Manon; Stipulanti. "Nyldon words". In: *J. Combin. Theory Ser. A* 167 (2019), pp. 60–90. DOI: 10.1016/j.jcta.2019.04.002.

[29] Kuo-Tsai Chen, Ralph H. Fox, and Roger C. Lyndon. "Free differential calculus. IV. The quotient groups of the lower central series". In: *Ann. of Math. (2)* 68 (1958), pp. 81–95. DOI: 10.2307/1970044.

[30] Jeong-Hyeon Choi, Hwan-Gue Cho, and Sun Kim. "GAME: a simple and efficient whole genome alignment method using maximal exact match filtering". In: *Computational Biology and Chemistry* 29.3 (2005), pp. 244–253.

[31] Anders Roy Christiansen, Mikko Berggren Ettienne, Tomasz Kociumaka, Gonzalo Navarro, and Nicola Prezza. "Optimal-Time Dictionary-Compressed Indexes". In: *ACM Trans. Algorithms* 17.1 (2021), 8:1–8:39.

[32] Wai-Fong Chuan. "Sturmian Morphisms and alpha-Words". In: *Theor. Comput. Sci.* 225.1-2 (1999), pp. 129–148.

[33] Sorin Constantinescu and Lucian Ilie. "The Lempel–Ziv Complexity of Fixed Points of Morphisms". In: *SIAM J. Discret. Math.* 21.2 (2007), pp. 466–481.

[34] Anthony J. Cox, Markus J. Bauer, Tobias Jakobi, and Giovanna Rosone. "Large-scale compression of genomic sequence databases with the Burrows-Wheeler transform". In: *Bioinform.* 28.11 (2012), pp. 1415–1419.

[35] M. Crochemore, T. Lecroq, and W. Rytter. *125 Problems in Text Algorithms: with Solutions*. Cambridge University Press, 2021.

[36] David Damanik and Daniel Lenz. "Substitution dynamical systems: Characterization of linear repetitivity and applications". In: *J. Math. Anal. Appl.* 321.2 (2006), pp. 766–780.

[37] Aaron C. E. Darling, Bob Mau, Frederick R. Blattner, and Nicole T. Perna. "Mauve: multiple alignment of conserved genomic sequence with rearrangements". In: *Genome Res.* 14.7 (2004), pp. 1394–1403.

[38] Xavier Droubay, Jacques Justin, and Giuseppe Pirillo. "Episturmian words and some constructions of de Luca and Rauzy". In: *Theor. Comput. Sci.* 255.1-2 (2001), pp. 539–553.

[39] Jean-Marie Dumont and Alain Thomas. "Systemes de numeration et fonctions fractales relatifs aux substitutions". In: *Theoretical Computer Science* 65.2 (1989), pp. 153–169. ISSN: 0304-3975. DOI: https://doi.org/10.1016/0304-3975(89)90041-8. URL: https://www.sciencedirect.com/science/article/pii/0304397589900418.

[40] Richard Durbin. "Efficient haplotype matching and storage using the positional Burrows-Wheeler transform (PBWT)". In: *Bioinform.* 30.9 (2014), pp. 1266–1272.

[41] Jean-Pierre Duval. "Mots de Lyndon et périodicité". In: *RAIRO Inform. Théor.* 14.2 (1980), pp. 181–191. DOI: 10.1051/ita/1980140201811.

[42] L'ubomíra Dvoráková. *String attractors of episturmian sequences*. 2022. arXiv: 2211.01660 [math.CO].

[43] Paolo Ferragina and Giovanni Manzini. "Opportunistic Data Structures with Applications". In: *In Proceedings 41st annual Symposium on Foundations of Computer ScienceFOCS*. IEEE Computer Society, 2000, pp. 390–398.

[44] Gabriele Fici, Giuseppe Romana, Marinella Sciortino, and Cristian Urbina. "On the impact of morphisms on BWT-runs". In: *CPM*. Vol. 259. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[45] Andrea Frosini, Ilaria Mancini, Simone Rinaldi, Giuseppe Romana, and Marinella Sciortino. "Burrows-Wheeler Transform on Purely Morphic Words". In: *DCC*. IEEE, 2022.

[46] Andrea Frosini, Ilaria Mancini, Simone Rinaldi, Giuseppe Romana, and Marinella Sciortino. "Logarithmic Equal-Letter Runs for BWT of Purely Morphic Words". In: *DLT*. Vol. 13257. Lect. Notes Comput. Sc. Springer, 2022, pp. 139–151.

[47] Travis Gagie. "Large alphabets and incompressibility". In: *Inf. Process. Lett.* 99.6 (2006), pp. 246–251.

[48] Travis Gagie, Tomohiro I, Giovanni Manzini, Gonzalo Navarro, Hiroshi Sakamoto, Louisa Seelbach Benkner, and Yoshimasa Takabatake. "Practical Random Access to SLP-Compressed Texts". In: *SPIRE*. Vol. 12303. Lecture Notes in Computer Science. Springer, 2020, pp. 221–231.

[49] Travis Gagie, Gonzalo Navarro, and Nicola Prezza. "Fully Functional Suffix Trees and Optimal Text Searching in BWT-Runs Bounded Space". In: *J. ACM* 67.1 (2020), 2:1–2:54.

[50] Arnab Ganguly, Rahul Shah, and Sharma V. Thankachan. "pBWT: Achieving Succinct Data Structures for Parameterized Pattern Matching and Related Problems". In: *SODA*. SIAM, 2017, pp. 397–407.

[51] Ira M. Gessel, Antonio Restivo, and Christophe Reutenauer. "A bijection between words and multisets of necklaces". In: *Eur. J. Comb.* 33.7 (2012), pp. 1537–1546.

[52] Daniele A. Gewurz and Francesca Merola. "Numeration and enumeration". In: *Eur. J. Comb.* 33.7 (2012), pp. 1547–1556.

[53] France Gheeraert, Giuseppe Romana, and Manon Stipulanti. "String Attractors of Fixed Points of k-Bonacci-Like Morphisms". In: Lecture Notes in Computer Science 13899 (2023), pp. 192–205.

[54] Sara Giuliani, Shunsuke Inenaga, Zsuzsanna Lipták, Nicola Prezza, Marinella Sciortino, and Anna Toffanello. "Novel Results on the Number of Runs of the Burrows-Wheeler-Transform". In: *47th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2021)*. Vol. 12607. LNCS. Springer, 2021, pp. 249–262.

[55] Sara Giuliani, Shunsuke Inenaga, Zsuzsanna Lipták, Giuseppe Romana, Marinella Sciortino, and Cristian Urbina. "Bit catastrophes for the Burrows-Wheeler Transform". In: *DLT*. Vol. 13911. Lecture Notes in Computer Science. Springer, 2023, pp. 86–99.

[56] Sara Giuliani, Zsuzsanna Lipták, Francesco Masillo, and Romeo Rizzi. "When a dollar makes a BWT". In: *Theor. Comput. Sci.* 857 (2021), pp. 123–146.

[57] Sara Giuliani, Giuseppe Romana, and Massimiliano Rossi. "Computing Maximal Unique Matches with the r-Index". In: *SEA*. Vol. 233. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 22:1–22:16.

[58] Peter W. Harrison, Rodrigo Lopez, Nadim Rahman, Stefan Gutnick Allen, Raheela Aslam, Nicola Buso, Carla Cummins, Yasmin Fathy, Eloy Felix, et al. "The COVID-19 Data Portal: accelerating SARS-CoV-2 and COVID-19 research through rapid open access data sharing". In: *Nucleic Acids Research* 49.W1 (2021), W619–W623.

[59] Alex Heinis. "Languages under substitutions and balanced words". In: *Journal de Théorie des Nombres de Bordeaux* 16 (2004), pp. 151–172.

[60] M. Hollander. "Greedy Numeration Systems and Regularity". In: *Theory Comput. Syst.* 31.2 (1998), pp. 111–133.

[61] Stepan Holub. "Words with unbounded periodicity complexity". In: *Int. J. Algebra Comput.* 24.6 (2014), pp. 827–836.

[62] Danny Hucke, Markus Lohrey, and Carl Philipp Reh. "The Smallest Grammar Problem Revisited". In: *String Processing and Information Retrieval - 23rd International Symposium, SPIRE 2016, Beppu, Japan, October 18-20, 2016, Proceedings*. Ed. by Shunsuke Inenaga, Kunihiko Sadakane, and Tetsuya Sakai. Vol. 9954. Lecture Notes in Computer Science. 2016, pp. 35–49.

[63] David A. Huffman. "A Method for the Construction of Minimum-Redundancy Codes". In: *Proceedings of the IRE* 40.9 (1952), pp. 1098–1101.

[64] Juha Kärkkäinen, Giovanni Manzini, and Simon J. Puglisi. "Permuted Longest-Common-Prefix Array". In: *CPM*. Vol. 5577. Lecture Notes in Computer Science. Springer, 2009, pp. 181–192.

[65] Toru Kasai, Gunho Lee, Hiroki Arimura, Setsuo Arikawa, and Kunsoo Park. "Linear-Time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications". In: *CPM*. Vol. 2089. Lecture Notes in Computer Science. Springer, 2001, pp. 181–192.

[66] D. Kempa and N. Prezza. "At the roots of dictionary compression: string attractors". In: *STOC 2018*. ACM, 2018, pp. 827–840.

[67] Dominik Kempa and Tomasz Kociumaka. "Resolution of the burrows-wheeler transform conjecture". In: *Commun. ACM* 65.6 (2022), pp. 91–98.

[68] Dominik Kempa, Alberto Policriti, Nicola Prezza, and Eva Rotenberg. "String Attractors: Verification and Optimization". In: *ESA*. Vol. 112. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 52:1–52:13. ISBN: 978-3-95977-081-1.

[69] Takuya Kida, Tetsuya Matsumoto, Yusuke Shibata, Masayuki Takeda, Ayumi Shinohara, and Setsuo Arikawa. "Collage system: a unifying framework for compressed pattern matching". In: *Theor. Comput. Sci.* 298.1 (2003), pp. 253–272.

[70] John C. Kieffer and En-Hui Yang. "Grammar-based codes: A new class of universal lossless source codes". In: *IEEE Trans. Information Theory* 46.3 (2000), pp. 737–754.

[71] Donald E. Knuth, James H. Morris, and Vaughan R. Pratt. "Fast Pattern Matching in Strings". In: *SIAM J. Comput.* 6.2 (1977), pp. 323–350.

[72] Tomasz Kociumaka, Gonzalo Navarro, and Francisco Olivares. "Near-Optimal Search Time in δ-Optimal Space". In: *LATIN*. Vol. 13568. Lecture Notes in Computer Science. Springer, 2022, pp. 88–103.

[73] Tomasz Kociumaka, Gonzalo Navarro, and Nicola Prezza. "Toward a Definitive Compressibility Measure for Repetitive Sequences". In: *IEEE Trans. Inf. Theory* 69.4 (2023), pp. 2074–2092.

[74] Tomasz Kociumaka, Gonzalo Navarro, and Nicola Prezza. "Towards a Definitive Measure of Repetitiveness". In: *LATIN*. Vol. 12118. Lect. Notes Comput. Sc. Springer, 2020, pp. 207–219.

[75] Andrej Nikolaevič Kolmogorov. "Three approaches to the quantitative definition of information". In: *Problems on Information Transmission 1*. Vol. 1. 1965, pp. 1–7.

[76] Stefan Kurtz, Adam M. Phillippy, Arthur L. Delcher, Michael Smoot, Martin Shumway, Corina Antonescu, and Steven L. Salzberg. "Versatile and open software for comparing large genomes". en. In: *Genome Biol.* 5.2 (2004), R12.

[77] Kanaru Kutsukake, Takuya Matsumoto, Yuto Nakashima, Shunsuke Inenaga, Hideo Bannai, and Masayuki Takeda. "On Repetitiveness Measures of Thue-Morse Words". In: *SPIRE*. Vol. 12303. Lect. Notes Comput. Sc. Springer, 2020, pp. 213–220.

[78] Tak Wah Lam, Ruiqiang Li, Alan Tam, Simon C. K. Wong, Edward Wu, and Siu-Ming Yiu. "High Throughput Short Read Alignment via Bi-directional BWT". In: *BIBM*. IEEE Computer Society, 2009, pp. 31–36.

[79] Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L. Salzberg. "Ultra-fast and memory-efficient alignment of short DNA sequences to the human genome". In: *Genome Biology* 10.3 (2009), R25.

[80] Pierre B. A. Lecomte and Michel Rigo. "Numeration Systems on a Regular Language". In: *Theory Comput. Syst.* 34.1 (2001), pp. 27–44. DOI: 10.1007/s002240010014.

[81] Abraham Lempel and Jacob Ziv. "On the Complexity of Finite Sequences". In: *IEEE T. Inform. Theory* 22.1 (1976), pp. 75–81.

[82] Heng Li and Richard Durbin. "Fast and accurate short read alignment with Burrows-Wheeler transform". In: *Bioinform.* 25.14 (2009), pp. 1754–1760.

[83] Aristid Lindenmayer. "Mathematical models for cellular interactions in development I. Filaments with one-sided inputs". In: *Journal of Theoretical Biology* 18.3 (1968), pp. 280–299. ISSN: 0022-5193.

[84] M. Lothaire. *Algebraic Combinatorics on Words*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2002. DOI: 10.1017/CBO9781107326019.

[85] M. Lothaire. *Combinatorics on Words*. Vol. 17. Encyclopedia of Mathematics. Reprinted in the *Cambridge Mathematical Library*, Cambridge University Press, 1997. Addison-Wesley, Reading, Mass., 1983.

[86] Aldo de Luca and Filippo Mignosi. "Some Combinatorial Properties of Sturmian Words". In: *Theor. Comput. Sci.* 136.2 (1994), pp. 361–285.

[87] Veli Mäkinen, Djamal Belazzougui, Fabio Cunial, and Alexandru I. Tomescu. *Genome-Scale Algorithm Design: Biological Sequence Analysis in the Era of High-Throughput Sequencing*. Cambridge University Press, 2015.

[88] Veli Mäkinen and Gonzalo Navarro. "Succinct Suffix Arrays based on Run-Length Encoding". In: *Nord. J. Comput.* 12.1 (2005), pp. 40–66.

[89] Veli Mäkinen, Esko Ukkonen, and Gonzalo Navarro. "Approximate Matching of Run-Length Compressed Strings". In: *Algorithmica* 35.4 (2003), pp. 347–369.

[90] Sabrina Mantaci, Antonio Restivo, Giuseppe Romana, Giovanna Rosone, and Marinella Sciortino. "A combinatorial view on string attractors". In: *Theor. Comput. Sci.* 850 (2021), pp. 236–248.

[91] Sabrina Mantaci, Antonio Restivo, Giuseppe Romana, Giovanna Rosone, and Marinella Sciortino. "String Attractors and Combinatorics on Words". In: *ICTCS*. Vol. 2504. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 57–71.

[92] Sabrina Mantaci, Antonio Restivo, Giovanna Rosone, and Marinella Sciortino. "An extension of the Burrows-Wheeler Transform". In: *Theor. Comput. Sci.* 387.3 (2007), pp. 298–312.

[93] Sabrina Mantaci, Antonio Restivo, Giovanna Rosone, Marinella Sciortino, and Luca Versari. "Measuring the clustering effect of BWT via RLE". In: *Theor. Comput. Sci.* 698 (2017), pp. 79–87.

[94] Sabrina Mantaci, Antonio Restivo, and Marinella Sciortino. "Burrows-Wheeler transform and Sturmian words". In: *Inf. Process. Lett.* 86.5 (2003), pp. 241–246.

[95] Giovanni Manzini. "An analysis of the Burrows-Wheeler transform". In: *J. ACM* 48.3 (2001), pp. 407–430.

[96] Guillaume Marçais, Arthur L. Delcher, Adam M. Phillippy, Rachel Coston, Steven L. Salzberg, and Aleksey V. Zimin. "MUMmer4: A fast and versatile genome alignment system". In: *PLoS Comput. Biol.* 14.1 (2018).

[97] Filippo Mignosi and Patrice Séébold. "Morphismes sturmiens et règles de Rauzy". In: *Journal de théorie des nombres de Bordeaux* 5.2 (1993), pp. 221–233.

[98] Hamoon Mousavi. "Automatic theorem proving in Walnut". In: (2016). Arxiv preprint arXiv:1603.06017 [cs.FL], available at http://arxiv.org/abs/1603.06017.

[99] Gonzalo Navarro. "Indexing Highly Repetitive String Collections, Part I: Repetitiveness Measures". In: *ACM Comput. Surv.* 54.2 (2022), 29:1–29:31.

[100] Gonzalo Navarro and Veli Mäkinen. "Compressed full-text indexes". In: *ACM Comput. Surv.* 39.1 (2007), p. 2.

[101] Gonzalo Navarro and Cristian Urbina. "On Stricter Reachable Repetitiveness Measures". In: *SPIRE*. Vol. 12944. Lecture Notes in Computer Science. Springer, 2021, pp. 193–206.

[102] Takaaki Nishimoto, Tomohiro I, Shunsuke Inenaga, Hideo Bannai, and Masayuki Takeda. "Fully Dynamic Data Structure for LCE Queries in Compressed Space". In: *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*. Ed. by Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier. Vol. 58. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, 72:1–72:15.

[103] Marco Oliva, Travis Gagie, and Christina Boucher. "Recursive Prefix-Free Parsing for Building Big BWTs". In: *DCC*. IEEE, 2023, pp. 62–70.

[104] Jean-Jacques Pansiot. "Complexité des Facteurs des Mots Infinis Engendrés par Morphimes Itérés". In: *ICALP*. Vol. 172. Lect. Notes Comput. Sc. Springer, 1984, pp. 380–389.

[105] Jean-Jacques Pansiot. "Decidability of Periodicity for Infinite Words". In: *RAIRO Theor. Informatics Appl.* 20.1 (1986), pp. 43–46.

[106] Geneviève Paquin. "On a generalization of Christoffel words: epichristoffel words". In: *Theor. Comput. Sci.* 410.38-40 (2009), pp. 3782–3791.

[107] William Parry. "On the β-expansions of real numbers". In: *Acta Math. Acad. Sci. Hungar.* 11 (1960), pp. 401–416. DOI: 10.1007/BF02020954.

[108] Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. "A new approach to fragment assembly in DNA sequencing". In: *RECOMB*. ACM, 2001, pp. 256–267.

[109] *Pizza&Chili Corpus*. http://pizzachili.dcc.uchile.cl.

[110] Alberto Policriti and Nicola Prezza. "LZ77 Computation Based on the Run-Length Encoded BWT". In: *Algorithmica* 80.7 (2018), pp. 1986–2011.

[111] Sofya Raskhodnikova, Dana Ron, Ronitt Rubinfeld, and Adam D. Smith. "Sublinear Algorithms for Approximating String Compressibility". In: *Algorithmica* 65.3 (2013), pp. 685–709.

[112] Antonio Restivo, Giuseppe Romana, and Marinella Sciortino. "String Attractors and Infinite Words". In: *LATIN*. Vol. 13568. Lecture Notes in Computer Science. Springer, 2022, pp. 426–442.

[113] Antonio Restivo and Giovanna Rosone. "Balancing and clustering of words in the Burrows-Wheeler transform". In: *Theor. Comput. Sci.* 412.27 (2011), pp. 3019–3032.

[114] Christophe Reutenauer. "Mots de Lyndon généralisés". In: *Sém. Lothar. Combin.* 54 (2005/07), Art. B54h, 16.

[115] Michel Rigo. *Formal Languages, Automata and Numeration Systems. 2. Applications to Recognizability and Decidability*. Networks and Telecommunications Series. ISTE, 2014.

[116] Massimiliano Rossi, Marco Oliva, Ben Langmead, Travis Gagie, and Christina Boucher. "MONI: A Pangenomic Index for Finding Maximal Exact Matches". In: *J. Comput. Biol.* 29.2 (2022), pp. 169–187.

[117] Grzegorz Rozenberg and Arto Salomaa. *The mathematical theory of L systems*. Academic press, 1980.

[118] Arto Salomaa and Matti Soittola. *Automata-Theoretic Aspects of Formal Power Series*. Texts and Monographs in Computer Science. Springer, 1978.

[119] Luke Schaeffer and Jeffrey Shallit. "String Attractors for Automatic Sequences". In: *CoRR* abs/2012.06840 (2021). arXiv: 2012.06840 [cs.FL].

[120] Marinella Sciortino and Luca Q. Zamboni. "Suffix Automata and Standard Sturmian Words". In: *DLT*. Vol. 4588. Lect. Notes Comput. Sc. Springer, 2007, pp. 382–398.

[121] Julian Seward. https://sourceware.org/bzip2/manual/manual.html. 1996. URL: "https://sourceware.org/bzip2/manual/manual.html".

[122] Claude E. Shannon. "A mathematical theory of communication". In: *Bell Syst. Tech. J.* 27.3 (1948), pp. 379–423.

[123] Neil J. A. Sloane. *The On-Line Encyclopedia of Integer Sequence*. URL: http://oeis.org.

[124] James A. Storer and Thomas G. Szymanski. "Data compression via textual substitution". In: *J. ACM* 29.4 (1982), pp. 928–951.

[125] Yuya Tamakoshi, Keisuke Goto, Shunsuke Inenaga, Hideo Bannai, and Masayuki Takeda. "An Opportunistic Text Indexing Structure Based on Run Length Encoding". In: *CIAC*. Vol. 9079. Lecture Notes in Computer Science. Springer, 2015, pp. 390–402.

[126] Ian H. Witten, Radford M. Neal, and John G. Cleary. "Arithmetic Coding for Data Compression". In: *Commun. ACM* 30.6 (1987), pp. 520–540.