Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 387 (2007) 298–312
Theoretical Computer Science
www.elsevier.com/locate/tcs

An extension of the Burrows–Wheeler Transform☆

S. Mantaci, A. Restivo, G. Rosone, M. Sciortino*

University of Palermo, Dipartimento di Matematica ed Applicazioni, Via Archirafi 34, 90123 Palermo, Italy

Abstract

We describe and highlight a generalization of the Burrows–Wheeler Transform (`bwt`) to a multiset of words. The extended transformation, denoted by `ebwt`, is reversible. Moreover, it allows to define a bijection between the words over a finite alphabet A and the finite multisets of conjugacy classes of primitive words in A^*. Besides its mathematical interest, the extended transform can be useful for applications in the context of string processing. In the last part of this paper we illustrate one such application, providing a similarity measure between sequences based on `ebwt`.

Keywords: Burrows–Wheeler transform; Sequence comparison; Alignment-free distance measure

0. Introduction

In 1994 Michael Burrows and David Wheeler introduced (cf. [1]) a reversible transformation on words (called the *Burrows–Wheeler Transform* after their names, and denoted by `bwt`) that aroused considerable interest and curiosity in the field of Data Compression.

Almost at the same time, in 1993, Ira M. Gessel and Christophe Reutenauer produced a paper in the field of combinatorics on permutations (cf. [7]), where a technique was introduced in order to find a bijection between words over a given alphabet and multisets of conjugacy classes of primitive words, over the same alphabet.

In [4] Crochemore, Désarménien and Perrin show the strict connection between the Burrows–Wheeler transformation and the Gessel–Reutenauer bijection. In fact they remarked that the Burrows–Wheeler Transform is connected to a particular case of the bijection given by Gessel and Reutenauer. This important remark has inspired the extension of the Burrows–Wheeler Transform (denoted by `ebwt`) to a multiset of primitive words that we introduce in the present paper. Notice that this is not the only extended version of the `bwt`. In fact, for instance, in [6] Ferragina et al. formalize an extension of the Burrows–Wheeler Transform to trees.

The heart of this paper is mainly devoted to describing the formal definition of the extended Burrows–Wheeler transformation and to give its algorithmic construction. As for `bwt`, we show, by providing a suitable algorithm,

☆ Partially supported by the Italian MIUR PRIN project "Automi e Linguaggi Formali: aspetti matematici ed applicativi" and by MIUR FIRB Italy–Israel project "Pattern Matching and Discovery in Discrete Structures, with applications to Bioinformatics".

* Corresponding author.
E-mail addresses: sabrina@math.unipa.it (S. Mantaci), restivo@math.unipa.it (A. Restivo), giovyros@virgilio.it (G. Rosone), mari@math.unipa.it (M. Sciortino).

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2007.07.014

		F					L
		↓					↓
	1	a	a	b	r	a	c
I →	2	a	b	r	a	c	a
	3	a	c	a	a	b	r
	4	b	r	a	c	a	a
	5	c	a	a	b	r	a
	6	r	a	c	a	a	b

Fig. 1. The matrix of all cyclic rotations of the word $w = abraca$.

that such an extended transformation is reversible, that is, it is possible to recover the original multiset S, once that its image by ebwt is known. We also show some of the properties of ebwt, in particular that, differently from the classical bwt, any word can be obtained as output of ebwt. The study of ebwt allows also to highlight and realize some distinctive features of bwt.

We remark that such an extended transformation, besides being an interesting combinatorial tool for the study of properties of sets of words, is also motivated by a couple of application. In [13] the authors show that such an extended transformation can be used as a preprocessing step for a new compression method, that in some cases is more effective than the technique used by most bwt-based compressors.

In the last part of this paper we illustrate a second application of ebwt, consisting in a new method for comparing sequences. This method is based on the observation that a special feature of ebwt is that the greater is the number of segments shared by two words u and v, the greater is the mixing of symbols coming from u and v in the output of the transformation, applied on the set $\{u, v\}$. Therefore, ebwt can be applied in order to define a new combinatorial method for comparing sequences, that, intuitively, takes into account how much the characters of the words to be compared are shuffled by the transformation ebwt. The implementation of this intuitive idea can have several different formalizations. In order to give the flavor of the applicative aspect of this transformation, in Section 5 we give one possible formalization of the comparison method based on the extended transformation, expressed by the distance measure δ. A deeper study of the comparison method based on ebwt has been developed in [15], where different formalizations of distance measures are given. Note that the distance measures based on ebwt can be placed in the context of alignment-free distances (cf. [5,19,18,9]), and it is particularly suitable, for instance, to capture evolutionary relations between biological sequences of different species.

In the last section the comparison method described here is validated by applying the distance δ to the whole mitochondrial genome phylogeny problem.

1. Preliminaries

The Burrows–Wheeler transform (bwt from now on) was introduced in 1994 by Burrows and Wheeler [1] and it represents an extremely useful tool for textual lossless data compression. The idea is to apply a reversible transformation in order to produce a permutation of the characters of an input string w, defined over an alphabet A, so that the string becomes easier to compress. Actually the transformation leads to group characters together so that the probability of finding a character close to another instance of the same character is substantially increased. The goal of this section is to give the description of the working of bwt in order to introduce the notation and some properties that will be used in next sections.

Let A be a finite ordered alphabet. We denote by A^* the set of words over A. We say that two words $x, y \in A^*$ are *conjugate* if $x = uv$ and $y = vu$ for some $u, v \in A^*$. A word x is called a *cyclic shift* or a *cyclic rotation* of y if x and y are conjugate. A word $v \in A^*$ is *primitive* if $v = u^n$ implies $v = u$ and $n = 1$.

The transformation bwt processes a word $w = w_1 \cdots w_n$ by constructing all n cyclic rotations of w and sorting them lexicographically. The output of $\mathtt{bwt}(w)$ consists of the pair (L, I), where L is the sequence of the last character of each rotation in the sorted list and I is the position of the original word in the list.

For instance, suppose we want to compute $\mathtt{bwt}(w)$ where $w = abraca$. Consider the matrix M, shown in Fig. 1, which consists of all cyclic shifts of w, lexicographically sorted.

The last column $L = caraab$ of the matrix and $I = 2$ are the output of bwt. The first column F, instead, contains the sequence of the characters of w lexicographically sorted.

In [1] the following properties concerning bwt have been proved:

(1) For all $i = 1, \ldots, n$, $i \neq I$, the character $L[i]$ is followed by $F[i]$ in the original string;
(2) for each character z, the ith occurrence of z in F corresponds to the ith occurrence of z in L.

From the above properties it follows that the Burrows–Wheeler transform is reversible in the sense that, given L and the index I, it is possible to recover the original string w. Actually, according to Property 2, we can define a permutation $\tau: \{1, \ldots, n\} \to \{1, \ldots, n\}$ giving the correspondence between the positions of characters of the first and the last column of the matrix M. The function τ represents also the order in which we have to rearrange the elements of F to reconstruct the original word w. Hence, starting from the position I, we can recover the word w as follows:

$$w_i = F[\tau^{i-1}(I)], \quad \text{where } \tau^0(x) = x, \quad \text{and} \quad \tau^{i+1}(x) = \tau(\tau^i(x)).$$

We show, for instance, how the reconstruction works for the example in Fig. 1:

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 5 & 6 & 1 & 3 \end{pmatrix},$$

$$\begin{aligned} w_1 &= F[2] = a \\ w_2 &= F[4] = b \\ w_3 &= F[6] = r \\ w_4 &= F[3] = a \\ w_5 &= F[5] = c \\ w_6 &= F[1] = a. \end{aligned}$$

If we do not care about the index I, the Burrows–Wheeler Transform defines a map $\mathcal{BWT}$ from A^* to itself such that if w is a words of A^*, $\mathcal{BWT}(w)$ is the word L obtained by concatenating the characters in the last column of the matrix of sorted cyclic rotations.

Proposition 1. *Two words $x, y \in A^*$ are conjugate if and only if $\mathcal{BWT}(x) = \mathcal{BWT}(y)$.*

Proof. Note that if x and y are conjugate then $\texttt{bwt}(x) = (L, I_x)$ and $\texttt{bwt}(y) = (L, I_y)$. So, $\mathcal{BWT}(x) = \mathcal{BWT}(y)$. Conversely, if $\mathcal{BWT}(x) = \mathcal{BWT}(y)$, since the permutation τ for the inverse transformation is univocally defined, the only difference between $\texttt{bwt}(x)$ and $\texttt{bwt}(y)$ can be in the choice of the index, corresponding to the choice of different conjugates. Then either $x = y$ (if $I_x = I_y$) or x is a conjugate of y (if $I_x \neq I_y$). □

The proposition stated below shows that there exists a relation between the image by $\mathcal{BWT}$ of a word and the power of the word itself. The proof of this proposition can be found in [16]:

Proposition 2. *For $u, v \in A^*$, u is a conjugate of v^d if and only if $\mathcal{BWT}(v) = a_1 \cdots a_n$ and $\mathcal{BWT}(u) = a_1^d \cdots a_n^d$.*

By Proposition 1, it follows that $\mathcal{BWT}$ is not injective, but it defines an injective mapping from the set of all conjugacy classes of words in A^* to A^*. However, the following example shows that $\mathcal{BWT}$ is not surjective.

Example 3. Let us consider the sequence $u = bccaaab$. By applying the reverse of bwt, it is easy to verify that there exists no sequence w such that $\mathcal{BWT}(w) = u$.

Note that the function $\mathcal{BWT}$ has been recently studied from a combinatorial point of view (see [4,16]). For instance in [16] it is proved that $\mathcal{BWT}$ provides a further characterization of Standard Words (cf. [12, Chapter 3]).

2. The Extended Burrows–Wheeler Transformation

This section is devoted to describe an extension of the Burrows–Wheeler Transform to a multiset of words and to illustrate some of its properties. Such an extension is strictly related to a bijection, introduced by Gessel and Reutenauer in 1993, between the multisets of words over a finite alphabet A and the words of A^*. In a recent paper [4] Crochemore, Désarménien and Perrin have shown that the Burrows–Wheeler Transform is connected to a particular case of the bijection given by Gessel and Reutenauer. This important remark has inspired the definition of the extension of the Burrows–Wheeler Transform (denoted by ebwt) to a multiset of primitive words, described in the present paper.

2.1. A new order relation between words

In order to define the extended transformation we need to introduce an order relation between words, that is different from the usual lexicographical one.

Recall that (cf. [11]) every word $v \in A^*$ can be written in a unique way as a power of a primitive word, i.e. there exists a unique primitive word w and a unique integer k such that $v = w^k$. We denote w by $root(v)$ and k by $exp(v)$.

If u is a word in A^*, we denote by u^ω the infinite word obtained by infinitely iterating u, i.e. $u^\omega = uuuuu \ldots$.

The lexicographic ordering $<_{lex}$ is naturally defined on infinite words. Given two infinite words $x = x_1 x_2 \ldots$ and $y = y_1 y_2 \ldots$, with $x_i, y_i \in A$, we say that $x <_{lex} y$ if there exists an index $j \in \mathbb{N}$ such that $x_i = y_i$ for $i = 1, 2, \ldots, j-1$ and $x_j < y_j$. Note that if $x = y$, the relation $<_{lex}$ is not defined. Remark that $u^\omega = v^\omega$ if and only if $root(u) = root(v)$.

Definition 4. Let u, v be two words over a finite alphabet A. We say that

$$u \preceq_\omega v \iff \begin{cases} exp(u) \leq exp(v) & \text{if } root(u) = root(v) \\ u^\omega <_{lex} v^\omega & \text{otherwise.} \end{cases}$$

It is easy to verify that $\preceq_\omega$ is a total order. We also remark that this order relation is different from the lexicographic one. For instance $ab <_{lex} aba$ but $aba \preceq_\omega ab$. Although when $root(u) \neq root(v)$ the $\preceq_\omega$-order of u and v is defined by using infinite words, the following proposition shows that it is possible to decide their mutual $\preceq_\omega$-ordering by extending them up to the length $|u| + |v| - \gcd(|u|, |v|)$. Such a bound is a consequence of a well-known result of Periodicity on Words, the Fine and Wilf Theorem (cf. [17,11]). For a given finite or infinite word w, we denote by $pref_k(w)$ the prefix of w of length k.

Proposition 5. *Given two words u and v, with $root(u) \neq root(v)$,*

$$u \preceq_\omega v \iff pref_k(u^\omega) <_{lex} pref_k(v^\omega),$$

where $k = |u| + |v| - \gcd(|u|, |v|)$.

Proof. If neither u is a prefix of v nor v is a prefix of u, then $u \preceq_\omega v$ if and only if $u <_{lex} v$. So it suffices to consider the length of the shortest sequence, in order to decide the mutual $\preceq_\omega$-ordering of u and v. Let us consider the case where one sequence is a prefix of the other one, for instance u is a prefix of v. In this case we need to iterate u and v, respectively, until we find out the first index i such that the ith character of u^ω is different from the ith character of v^ω. The Fine and Wilf theorem guarantees that $i \leq |u| + |v| - \gcd(|u|, |v|)$. In fact, if we had that u^ω equals v^ω up to the length $|u| + |v| - \gcd(|u|, |v|)$, then we would have that the prefix of u^ω of length $|u| + |v| - \gcd(|u|, |v|)$ would be a sequence having two periods, $|u|$ and $|v|$, but not their greatest common divisor, since u and v are not powers of the same sequence. □

The bound given in Proposition 5 is tight: this is a consequence of the tightness of such a bound in the Fine and Wilf Theorem.

Example 6. We can consider the words $u = abaab$ and $v = abaababa$. One can see that $v \preceq_\omega u$ and u^ω and v^ω differ for the character in position $12 = 5 + 8 - 1$. Remark also that $u <_{lex} v$.

$$\overbrace{abaab}^{u}\overbrace{abaab}^{u}\overbrace{a\boldsymbol{b}\cdots}^{u}$$
$$\underbrace{abaababa}_{v}\underbrace{aba\boldsymbol{a}\cdots}_{v}$$

2.2. The Extended Burrows–Wheeler Transform

We introduce now the transformation `ebwt` under the hypothesis that the words considered are primitive. Actually this hypothesis is not very restrictive, since in practice almost all the processed texts are primitive (or become primitive by adding an end-of-string symbol).

	$M_{\mathcal{C}}$	$pref_6$	M_L	M_χ	M_F
1	*abac*	*abacab*	**c**	1	*a*
2	*abc*	*abcabc*	**c**	0	*a*
3	*abcb*	*abcbab*	**b**	0	*a*
4	*acab*	*acabac*	**b**	0	*a*
5	*acb*	*acbacb*	**b**	0	*a*
6	*babc*	*babcba*	**c**	0	*b*
7	*baca*	*bacaba*	**a**	0	*b*
8	*bac*	*bacbac*	**c**	0	*b*
9	*bca*	*bcabca*	**a**	1	*b*
10	*bcba*	*bcbabc*	**a**	0	*b*
11	*caba*	*cabaca*	**a**	0	*c*
12	*cab*	*cabcab*	**b**	0	*c*
13	*cbab*	*cbabcb*	**b**	1	*c*
14	*cba*	*cbacba*	**a**	1	*c*

Fig. 2. The output of $\texttt{ebwt}(S)$ is the pair (M_L, M_χ) where $M_L = ccbbbcacaaabba$ and $M_\chi = 10000000100011$.

Let $S = \{u_1, \ldots u_k\}$ be a multiset of (not necessarily distinct) primitive words of A^*. We denote by

$$\|S\| = \sum_{i=1}^{k} |u_i|$$

and by

$$H = \max\{|u_i| + |u_j| - \gcd(|u_i|, |u_j|) \mid i, j = 1, \ldots, k\}.$$

Consider the set $\mathcal{C}(S)$ of all the conjugates of the words in S. We can associate to each $w \in \mathcal{C}(S)$ the triplet $(pref_H(w^\omega), L(w), \chi_S(w))$, where $pref_H(w^\omega)$ is the prefix of w^ω of length H, $L(w)$ denotes the last character of w and χ_S is the characteristic function of S, that is

$$\chi_S(w) = \begin{cases} 1 & \text{if } w \in S \\ 0 & \text{otherwise.} \end{cases}$$

Consider the set of all these triplets and sort it by taking as sorting key the first field and using the $<_{lex}$ relation. By Proposition 5, this sorting induces on the words of $\mathcal{C}(S)$ the same order as obtained by applying the $\preceq_\omega$-order relation on the words of $\mathcal{C}(S)$. We can arrange this sorted list in a table $M(S)$. We denote by $M_L(S)$ and $M_\chi(S)$ the sequences obtained by concatenating the second and the third components, respectively, of the triplets of the table $M(S)$. Note that where there is no danger of ambiguity, we will use M, M_L and M_χ instead of $M(S)$, $M_L(S)$ and $M_\chi(S)$, respectively. If we denote by $M_{\mathcal{C}}$ the sorted list, with respect to the $\preceq_\omega$-order, of the words in $\mathcal{C}(S)$, M_L is the word obtained from the concatenation of the last characters of elements in $M_{\mathcal{C}}$ and M_χ is the characteristic vector saying which elements in the list are coming from S.

Definition 7. The Extended Burrows–Wheeler Transform of a multiset S, denoted by $\texttt{ebwt}(S)$, is the pair (M_L, M_χ).

Remark 8. We note that, since in the word M_χ the number of occurrences of 0's is larger than 1's, it could be more reasonable, from a computational point of view, to define $\texttt{ebwt}(S)$ as the pair $(M_L, \mathcal{I})$, where $\mathcal{I}$ is the set of indices i such that $M_\chi[i]$ is equal to 1.

We can extend the table M by adding a new column M_F, such that $M_F[i]$ contains the first character of $M_{\mathcal{C}}[i]$. Notice that since the elements in $M_{\mathcal{C}}$ are $\preceq_\omega$-sorted, then the characters in M_F are alphabetically sorted.

Example 9. Let $S = \{abac, cbab, bca, cba\}$. We represent in Fig. 2 the table M after the lexicographic sorting on the $pref_6$'s component (the second column) of its rows. For sake of clearness we add also the column (the first one) $M_{\mathcal{C}}$ containing the $\preceq_\omega$-ordered list of all $w \in \mathcal{C}(S)$. The second column contains the $<_{lex}$-sorted list of prefixes of length $H = 6$, the third column the word M_L and the fourth column the characteristic array of S, M_χ.

The complexity of the algorithm for computing $\texttt{ebwt}$ is upper bounded by the time needed to get the sorted list of the conjugates of the words in S, that can be handled by a suitable generalization of the suffix array. A generalization

of the "skew algorithm" (cf. [8]) for the construction of the generalized suffix array, allows to obtain such a sorted list in linear time on the total size of the set S. Then the algorithm has complexity $O(||S||)$.

The following proposition shows two important properties connecting the characters of M_L and M_F.

Proposition 10. *Let S be a multiset of primitive words and let $\texttt{ebwt}(S) = (M_L, M_\chi)$. Let M_F be the sequence of the first characters of the sorted list. The following properties hold:*

(1) *For every i such that $M_\chi[i] = 0$, $M_F[i]$ follows $M_L[i]$ in one word in S.*
(2) *For a fixed character $a \in A$, its occurrences in M_F appear in the same order as in M_L, i.e. its kth instance of a in M_F corresponds to its kth instance of a in M_L.*

Proof. The proof of Item 1 is quite straightforward. Since $M_\chi[i] = 0$ then $M_\mathcal{C}[i] \notin S$. Then it is a conjugate of one of the original words $w \in S$. This means that in w the character $M_L[i]$ is followed by the character $M_F[i]$. In order to prove Item 2 we can consider the sequence M' of words such that $M'[i]$ is obtained by rotating $M_\mathcal{C}[i]$ one character to the right, so that the sequence M_L appears as the concatenation of the first characters of the words in M'. Notice that the words of M' are $\preceq_\omega$-sorted starting from the second character. So for those words beginning with the same character a in M' it is true that they must appear in $\preceq_\omega$-order to one another, since they are $\preceq_\omega$-sorted starting from the second character, and they all begin by a. So for any character a, the words in $M_\mathcal{C}$ that begin with a are in the same relative order as the words in M' that begin with a. This proves that the relative order of the different occurrences of a in M_F and in M_L are the same. □

Remark 11. We note that, when the set S contains only one element, the extended transformation works exactly as the Burrows–Wheeler Transform. In fact, the $\preceq_\omega$-ordering of the cyclic shifts of a word is equivalent to the lexicographic one, because all the words have the same length. Moreover, according to Remark 8, the set of indices contains a single value. So, if $S = \{w\}$, then $\texttt{ebwt}(S) = \texttt{bwt}(w)$.

3. The inverse transformation

In this section we show that the extended transformation is reversible, that is, we can recover the original set S from $\texttt{ebwt}(S)$.

Given the table M obtained in the computation of the ebwt, we can define a permutation θ on $\{1, \ldots, \|S\|\}$ as follows: $\theta(i) = j$ if $M_F[i]$ and $M_L[j]$ correspond to the same character in a word of S, according with the Item 2 of Proposition 10. In other words, if $M_F[i]z$ is a conjugate of a word $w_0 \in S$, for some $z \in A^*$, then the permutation θ associates i to the integer j corresponding to the position in the sorted list of the next left rotation of w_0, that is $zM_F[i] = zM_L[j]$.

Remark 12. Notice that the construction of the permutation θ does not depend on the knowledge of $M_\mathcal{C}$ but only on the knowledge of M_L. In fact for each position $i \in \{1, \ldots, \|S\|\}$, we can consider the symbol $a = M_F[i]$: if $M_F[i]$ is the hth occurrence of a in M_F, then $\theta(i) = j$, where j is the position where the hth occurrence of a appears in M_L. Notice that Item 2 of Proposition 10 has as consequence that this permutation relates all and only the positions where the conjugates of the same word appear in the sorted list. This translates to the fact that the permutation θ can be decomposed into as many cycles as the number of words in the multiset S. This fact is of fundamental relevance for the invertibility of ebwt.

Theorem 13. *The transformation* ebwt *that associates to a multiset S the pair $(M_L(S), M_\chi(S))$ is injective.*

Proof. Let S and T two multisets such that it holds $(M_L(S), M_\chi(S)) = (M_L(T), M_\chi(T))$. Since $M_L(S) = M_L(T)$ then also $M_F(S) = M_F(T)$. This implies that the permutations θ_S and θ_T obtained by associating each character of $M_L(S)$ and $M_L(T)$ with the same occurrence of that character in $M_F(S)$ and $M_F(T)$, respectively, are equal. We recall also the well-known combinatorial property, that for any permutation there exists a unique decomposition into disjoint cycles: $\theta_S = \theta_T = \sigma_1\sigma_2 \cdots \sigma_k$. From Item 1 of Proposition 10, as noticed in Remark 12, one can derive that k is equal to the cardinality of the multiset, so S and T have the same number of words. Recall also that each σ_i corresponds to a conjugacy class of a word in S and in T. We can reconstruct the words of S as described in the following. Because of primitivity of words in the multiset S, for every $j = 1, \ldots, k$, there exists a unique index i such that $M_\chi(S)[i] = 1$, that is moved by σ_j. Let $i_1, i_2, \ldots, i_k$ be the indices where $M_\chi(S)[i_1] = M_\chi(S)[i_2] = \cdots = M_\chi(S)[i_k] = 1$. So, one can reconstruct each word in S by using Item 2 of Proposition 10:

$$u_1 = M_F[i_1]M_F[\theta_S(i_1)]M_F[\theta_S^2(i_1)]\cdots M_F[\theta_S^{l_1}(i_1)],$$
$$u_2 = M_F[i_2]M_F[\theta_S(i_2)]M_F[\theta_S^2(i_2)]\cdots M_F[\theta_S^{l_2}(i_2)],$$
$$\cdots$$
$$u_k = M_F[i_k]M_F[\theta_S(i_k)]M_F[\theta_S^2(i_k)]\cdots M_F[\theta_S^{l_k}(i_k)],$$

where $l_1, l_2, \ldots, l_k$ are the lengths of the cycles $\sigma_1, \sigma_2, \ldots, \sigma_k$, respectively.

Since T is also a multiset of primitive words, $\theta_T = \theta_S$ and $M_\chi(T) = M_\chi(S)$, we can easily deduce that $T = S$. □

The previous proposition allows us to define the algorithm REVERSE that, given the extended transform of a multiset S of primitive words $\texttt{ebwt}(S) = (M_L, M_\chi)$, is able to recover S.

```
Algorithm REVERSE (ebwt(S) = (M_L, M_χ));
1. Create M_F by alphabetically sorting the characters of M_L;
2. Build the permutation θ;
3. S := ∅; N := length(M_L);
4. for i = 1, ..., N do
5.   if M_χ[i] = 1 then
6.       u := ϵ; {ϵ is the empty word}
7.       j := i;
8.       repeat
9.           u := u + M_F[j]; {+ is concatenation between strings}
10.          j := θ(j);
11.      until j = i;
12.      S := S ∪ {u};
13. Return S;
```

The following example describes the working of the algorithm REVERSE.

Example 14. Given $M_L = ccbbbcacaaabba$ and $M_\chi = 10000000100011$ as obtained in Example 9, one can construct M_F

$$M_F = aaaaabbbbbcccc$$

and build the permutation

$$\theta = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 7 & 9 & 10 & 11 & 14 & 3 & 4 & 5 & 12 & 13 & 1 & 2 & 6 & 8 \end{pmatrix}.$$

whose decomposition into disjoint cycles is the following:

$$\theta = (1\ 7\ 4\ 11)(2\ 9\ 12)(3\ 10\ 13\ 6)(5\ 14\ 8).$$

By starting from the indices $i = 1, 9, 13, 14$ for which $M_\chi[i] = 1$, we get respectively the words

$$u_1 = M_F[1]M_F[7]M_F[4]M_F[11] = abac, \quad u_2 = M_F[9]M_F[12]M_F[2] = bca,$$
$$u_3 = M_F[13]M_F[6]M_F[3]M_F[10] = cbab, \quad u_4 = M_F[14]M_F[8]M_F[5] = cba.$$

Proposition 15. *The algorithm* REVERSE *recovers the original multiset S in time $O(||S||)$.*

Proof. Both M_F in line 1 and θ in line 2 can be obtained by applying a counting sort algorithm, keeping the information of the corresponding positions of the elements in the unsorted and in the sorted arrays, during the construction of the sorted array. This is obtained in time $O(||S||)$.

Regarding to the cycle in line 4, the cost on the i's such that $M_\chi[i] = 0$ is constant (giving approximately a time $O(||S||)$ on all of such elements), whereas for each j's such that $M_\chi[j] = 1$, the cost is proportional to the length of the word recovered starting from j. By summing up on all these j's we get $O(||S||)$. Then the total complexity of the algorithm REVERSE is $O(||S||)$. □

	M'_C			M'_L	M'_χ	M'_F
1	a	a	b	**b**	0	a
2	a	b		**b**	1	a
3	a	b	a	**a**	1	a
4	b	a		**a**	0	b
5	b	a	a	**a**	0	b

Fig. 3. The table M' associated to the set $S = \{ab, aba\}$, when the lexicographic order is applied for sorting the conjugates of the words in S.

Remark 16. Algorithm REVERSE does not work if we use, in the computation of `ebwt`, the lexicographic order instead of the $\preceq_\omega$-order because the Item 2 of Proposition 10 might not be true. For instance, if the lexicographic order is applied for the computation of the transformation on the set $S = \{ab, aba\}$ (see the table M' in Fig. 3), one can notice that the first occurrence of b in M'_F would correspond to the second occurrence of b in M'_L. In this way the permutation θ would not satisfy Item 2 of Proposition 10, and the algorithm REVERSE would not work correctly. In fact the application of the algorithm REVERSE on $(M'_L, M'_\chi) = (bbaaa, 01100)$ with this permutation θ would not recover the original set S. Actually REVERSE$((bbaaa, 01100))$=$\{abaab, ababa\}$.

One can note that in order to assure the surjectivity of `ebwt`, we need the transformation to be defined on a *multiset* S and not simply on a set. Example 17 describes the working of algorithm REVERSE when some words of the recovered multiset are equal.

Example 17. Let $M_L = bbbbbaaaaabaaa$ and $M_\chi = 00011001000000$ be the input of algorithm REVERSE. We can obtain $M_F = aaaaaaaabbbbbb$ and the permutation θ defined as follows:

$$\theta = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 6 & 7 & 8 & 9 & 10 & 12 & 13 & 14 & 1 & 2 & 3 & 4 & 5 & 11 \end{pmatrix}.$$

whose decomposition into disjoint cycles is the following:

$$\theta = (1\ 6\ 12\ 4\ 9)(2\ 7\ 13\ 5\ 10)(3\ 8\ 14\ 11).$$

The algorithm reconstructs the word $abaab$ twice, by using the index 4 and the index 5, and the word $abba$ by using the index 8. So, we get the multiset $S = \{abaab, abaab, abba\}$.

Remark 18. According to the notation used in Remark 8, if $\texttt{ebwt}(S) = (M_L, \mathcal{I})$ and θ is the permutation defined between M_F and M_L, the set $\mathcal{I}$ should contain exactly one integer for each cycle in which θ is decomposed. Since $\texttt{ebwt}(S) = \texttt{bwt}(w)$ when $S = \{w\}$ (see Remark 11), it is easy to see that the words that are obtained as output of the Burrows–Wheeler Transform are exactly the ones where the permutation θ is a unique cycle. In this case the set $\mathcal{I}$ contains only one integer. For a characterization of the permutations defined by `bwt` see [4], [12, Chapter 11].

4. Combinatorial properties of the extended transformation

In this section we consider some properties of the extended transformation when we do not consider the characteristic vector M_χ. In particular we prove the bijectivity that such a transformation induces between the family of the multisets of conjugacy classes and the set A^*.

If we do not care about M_χ in the output of `ebwt`, then we can define a map $\mathcal{EBWT}$ such that if S is a multiset of primitive words of A^*, then $\mathcal{EBWT}(S) = M_L(S)$, where $M_L(S)$ is the word obtained by concatenating the characters of the column M_L of the table M obtained by $\preceq_\omega$-sorting the set $\mathcal{C}(S)$, as in the computation of $\texttt{ebwt}(S)$.

The following lemma (cf. Proposition 1) shows that $\mathcal{EBWT}$ is not injective, but in the next theorem it is proved that this map induces a bijection on conjugacy classes. Such a result, that plays an important role both in Combinatorics on Words and in the study of Free Lie Algebras, was firstly synthesized in a theorem, due to Gessel and Reutenauer (cf. [7]), and re-proved below by using the concepts and the notation introduced in the present paper.

We say that two multisets S and T are *conjugate* if there exists a bijection $\varphi\colon S \to T$ such that for each $u \in S$, $\varphi(u)$ is a conjugate of u.

Lemma 19. *Let $S = \{u_1, u_2, \ldots, u_k\}$ and $T = \{v_1, v_2, \ldots, v_h\}$ be two multisets of primitive words of A^*. Then $\mathcal{EBWT}(S) = \mathcal{EBWT}(T)$ if and only if T and S are conjugate.*

	M'_C	M'_L	M'_F
1	$a\ a\ b$	**b**	a
2	$a\ b$	**b**	a
3	$a\ b\ a$	**a**	a
4	$b\ a$	**a**	b
5	$b\ a\ a$	**a**	b

(a)

	M'_C	M'_L	M'_F
1	$a\ a\ b\ a\ b$	**b**	a
2	$a\ b\ a\ a\ b$	**b**	a
3	$a\ b\ a\ b\ a$	**a**	a
4	$b\ a\ a\ b\ a$	**a**	b
5	$b\ a\ b\ a\ a$	**a**	b

(b)

Fig. 4. (a) The table $M'(S)$ associated to the lexicographic ordering of the conjugates of words in S. (b) The table $M'(T)$ associated to the lexicographic ordering of the conjugates of T. For each table M', the column M'_C represent the sorted list of the conjugates, the column M'_L the sequence of their last symbols and the column M'_F the sorted characters of M'_L.

Proof. If T and S are conjugate, then $\mathcal{C}(S) = \mathcal{C}(T)$. It follows, by the definition of the transformation `ebwt`, that $\mathcal{EBWT}(S) = \mathcal{EBWT}(T)$. Conversely, if $\mathcal{EBWT}(S) = \mathcal{EBWT}(T)$, then $M_L(S) = M_L(T)$. So, by using the same reasoning as in the proof of Theorem 13, we can deduce that θ_S and θ_T are equal, then they have the same decomposition in cycles, i.e. $\theta_S = \theta_T = \phi_1\phi_2\cdots\phi_k$. From Item 1 of Proposition 10, as noticed in Remark 12, one can derive that k is equal to the cardinality of the multiset, so S and T must have the same number of words. Recall also that each ϕ_i corresponds to a conjugacy class of a word in S and a word in T. Notice that, in the algorithm REVERSE, the role of M_χ consists in characterize a representative in each conjugacy class. If we choose for each cycle an integer moved by the cycle itself, then by applying the algorithm REVERSE we can reconstruct a multiset of primitive words $\{w_1, \ldots, w_k\}$, that is conjugate both to T and S. Then by transitivity, T and S are conjugate. □

Theorem 20. *The Extended Burrows–Wheeler Transform defines a bijection between A^* and the family $\mathcal{F}$ of finite multisets of conjugacy classes of primitive words in A^*.*

Proof. Let $S = \{u_1, \ldots, u_k\}$ be a multiset of primitive words in A^*. Let us denote by $\tilde{S}$ the multiset of the conjugacy classes represented by the words in S, i.e. $\tilde{S} = \{[u_1], \ldots, [u_k]\}$ where $[u]$ denotes the conjugacy class represented by u. Consider the function $\Phi\colon \mathcal{F} \to A^*$ that is defined as follows: $\Phi(\tilde{S}) = \mathcal{EBWT}(S)$. By previous lemma, it follows that Φ is well defined. Moreover, it also follows that Φ is injective. We prove that Φ is also surjective. In fact given any word $L \in A^*$, it is always possible to obtain the sorted sequence F of the characters in L and determine the permutation θ associated to L and F. The permutation allows to apply the lines 3–13 of the Algorithm REVERSE in order to reconstruct a representative element of each conjugacy class belonging to $\tilde{S}$, by constructing M_χ as follows. If $\theta = \sigma_1\sigma_2\cdots\sigma_k$ and σ_1 moves 1, we have to choose k integers $i_1, \ldots, i_k$ where $i_1 = 1$ and, for $j = 2, \ldots, k$, i_j is the smallest integer moved by σ_j and initialize $M_\chi[i_j] = 1$ for $j = 1, \ldots, k$. So we can always find a multiset $S = \{u_1, \ldots, u_k\}$ whose image by $\mathcal{EBWT}$ is L. It follows $\Phi(\tilde{S}) = L$. □

Where no confusion arises, we can identify $\mathcal{EBWT}$ and the bijection Φ defined in the proof of the previous theorem. So, where there is no danger of ambiguity, we will apply indifferently $\mathcal{EBWT}$ both to words and to conjugacy classes.

We can note that, as described in Remark 11 for `ebwt`, if the set S consists of only one word w then $\mathcal{EBWT}(S) = \mathcal{BWT}(w)$. Nevertheless, differently from $\mathcal{BWT}$, the function $\mathcal{EBWT}$ is also surjective. Actually, while there exist words that are not image by $\mathcal{BWT}$ of any word, it is always possible, for any word, to find a multiset S of conjugacy classes of primitive words such that the image of S by $\mathcal{EBWT}$ is equal to that word. An instance is given in the following example.

Example 21. Let $w = bccaaab$ as in Example 3. It is easy to verify that w is the word obtained by applying $\mathcal{EBWT}$ to the conjugacy classes of the words ab and $abcac$.

Remark 22. The injectivity of the function $\mathcal{EBWT}$ on conjugacy classes defined in Theorem 20 does not hold if we use the lexicographic order instead of the $\preceq_\omega$-order. In fact, if we consider the conjugacy classes of the set $S = \{ab, aba\}$ and the set $T = \{abaab\}$, and the lexicographic ordering between the conjugates is applied, we get in both case to the same sequence of "last symbols" $bbaaa$, as shown in Fig. 4. The problem is that if we lexicographically sort the cyclic shifts of the words in S, Item 2 of Proposition 10 is not verified. Actually, as shown in Fig. 4(a), the first occurrence of the character b in $M'_F(S)$ corresponds to the second occurrence of b in $M'_L(S)$.

Recall that a word u is a *subword* of a word $v = a_1a_2\cdots a_n$, if there exist some indices $1 \le i_1 < i_2 < \cdots < i_k \le n$ such that $u = a_{i_1}a_{i_2}\cdots a_{i_k}$.

The following propositions describe some interesting combinatorial properties of the transformation $\mathcal{EBWT}$.

Proposition 23. *Let S_1 and S_2 be two multisets of primitive words of A^*. If $S_1 \subseteq S_2$ then $\mathcal{EBWT}(S_1)$ is a subword of $\mathcal{EBWT}(S_2)$.*

Proof. It easily comes from the fact that the $\preceq_\omega$-ordering of elements in S_2 preserves the $\preceq_\omega$-ordering of elements in S_1. □

Recall that, given two words $u, v \in A^*$, the *shuffle* of u and v, denoted by $u \circ v$, is the subset of A^* defined as:

$$\{u_1v_1u_2v_2 \cdots u_nv_n \mid n \geq 0, u_i, v_i \in A^*, u = u_1u_2 \cdots u_n, v = v_1v_2 \cdots v_n\}.$$

Proposition 24. *Let X and Y be two multisets of primitive words of A^*. If $S = X \cup Y$, then $\mathcal{EBWT}(S)$ belongs to the set $\mathcal{EBWT}(X) \circ \mathcal{EBWT}(Y)$.*

Proof. Since $X, Y \subseteq S$, from Proposition 23 it follows that $\mathcal{EBWT}(X)$ and $\mathcal{EBWT}(Y)$ are subwords of $\mathcal{EBWT}(S)$. So, by definition of the shuffle operator, the thesis follows. □

The shuffle is an associative operation, so we can trivially define the shuffle of a sequence of words.

Corollary 25. *Given $S = \{u_1, \ldots, u_k\}$, then $\mathcal{EBWT}(S)$ is an element of the shuffle $\mathcal{BWT}(u_1) \circ \mathcal{BWT}(u_2) \circ \cdots \circ \mathcal{BWT}(u_k)$.*

Proof. The proof comes from the previous proposition and from the fact that, by using Remark 11, if the multiset S consists of only one word w, then $\mathcal{EBWT}(S) = \mathcal{BWT}(w)$. □

5. Comparing sequences by using `ebwt`

In this section we apply our extended transformation `ebwt` in order to introduce a new method for comparing sequences. The new distance between words defined here is simple and efficient to compute, and it is particularly advantageous in the case of comparison of a set of sequences. We remark that our distance is not based on sequence alignment. Several alignment-free distance measures have recently been introduced (see for instance [5,19,18,9] and references therein) since they better fit with the problem of comparing genomic sequences than the methods based on sequence alignment. In fact, alignment-based methods compare sequences by considering only local edit operations on their fragments. Instead, the recent developments in genome sequences technologies have allowed to handle the complete genome of many different species, and have highlighted that, in order to capture evolutionary and functional mechanisms of different species, we need to consider a new set of sequence modifications, that involve recombination or shuffling of segments of genome. The distance we define takes into account such kind of modifications and therefore it can be successfully applied to compare genomic sequences, as shown in Section 6.

We define a new notion of distance between two sequences. Such a notion is based on the following intuitive idea: when `ebwt` is applied to the set $S = \{u, v\}$, if the same segment s occurs both in u and v, then the conjugates of u and v starting by s are likely to be close in the $\preceq_\omega$-sorted list of conjugates. This implies that the greater is the number of segments shared by u and v, the greater is the mixing of the conjugates of u and v in the sorted list. The comparison method based on transformation `ebwt` will measure how similar u and v are, by taking into account how much their conjugates are mixed. In this section, in order to show an application of the extended Transformation, we define a distance measure that computes the number of alternations in the above list between the conjugates of u and those of v. A more detailed study of the class of distance measures based on `ebwt` can be found in [15], where other different formalizations of distance measures based on `ebwt` are given.

Formally, let $S = \{u_1, u_2, \ldots, u_k\}$ be a multiset of primitive words in A^*, let $m = \sum_{i=1}^{k} |u_i|$ and let $M_{\mathcal{C}} = w_1, w_2, \ldots, w_m$ be the sorted list of the conjugates of the elements of S obtained during the computation of $\texttt{ebwt}(S)$. Consider the new alphabet $\Sigma = \{U_1, U_2, \ldots, U_k\}$. The *coloring* of $\texttt{ebwt}(S)$ is the map $\gamma\colon \{1, 2, \ldots, m\} \to \Sigma$ defined, for as:

$$\gamma(i) = U_j \qquad \text{if } w_i \text{ is a conjugate of } u_j.$$

We denote by M_γ the word over Σ^* such that $M_\gamma[i] = \gamma(i)$.

The following example describes the coloring of `ebwt`, by adding the sequence M_γ as a column to the table M associated to the computation of `ebwt`, as shown in Fig. 5.

	M_C	M_L	M_γ	M_χ
1	*aaabbbb*	*b*	*U*	1
2	*aabbbab*	*b*	*V*	0
3	*aabbbba*	*a*	*U*	0
4	*abaabbb*	*b*	*V*	1
5	*ababbab*	*b*	*Z*	1
6	*ababbab*	*b*	*Z*	0
7	*abbabab*	*b*	*Z*	0
8	*abbbaba*	*a*	*V*	0
9	*abbbbaa*	*a*	*U*	0
10	*baaabbb*	*b*	*U*	0
11	*baabbba*	*a*	*V*	0
12	*babaabb*	*b*	*V*	0
13	*bababab*	*b*	*Z*	0
14	*bababba*	*a*	*Z*	0
15	*babbaba*	*a*	*Z*	0
16	*bbaaabb*	*b*	*U*	0
17	*bbabaab*	*b*	*V*	0
18	*bbababa*	*a*	*Z*	0
19	*bbbaaab*	*b*	*U*	0
20	*bbbabaa*	*a*	*V*	0
21	*bbbbaaa*	*a*	*U*	0

Fig. 5. The table M associated to $\texttt{ebwt}(S)$ in which the column M_γ of the coloring is added.

Example 26. Let $S = \{u, v, z\}$, where $u = aaabbbb$, $v = abaabbb$ and $z = abababb$. Let U, V, Z be the colors associated, by the map γ, to u, v, z, respectively. As one can see in Fig. 5, $M_\gamma = UVUVZZZVUUVVZZZUVZUVU$.

The definition of coloring allows us to introduce a new notion of distance measure between two sequences that takes into account the alternation of the symbols coming from different sequences in the output of the transformation `ebwt`.

Definition 27. Let $u, v \in A^*$ be two sequences and let us consider $M_\gamma(u, v) = U^{n_1}V^{n_2}U^{n_3}\cdots V^{n_k}$. We define the measure $\delta(u, v)$ as follows:

$$\delta(u, v) = \sum_{\substack{i=1,\\ n_i \neq 0}}^{k} (n_i - 1)$$

A description of the computation of distance δ is given in the Example 29.

The following proposition provides some properties of the measure δ.

Proposition 28. *The following statements hold:*

(1) $\delta(u, v) = \delta(v, u)$*, i.e. the measure* δ *is symmetric.*
(2) *If* u *and* v *are conjugate, then* $\delta(u, v) = 0$.
(3) *If* u' *is a conjugate of* u *and* v' *is a conjugate of* v*, then* $\delta(u, v) = \delta(u', v')$*. Therefore,* δ *is a distance measure for conjugacy classes.*

M_C					M_γ
a	a	b	c		U
a	b	c	a		U
a	b	c	c	b	V
b	a	b	c	c	V
b	c	a	a		U
b	c	c	b	a	V
c	a	a	b		U
c	b	a	b	c	V
c	c	b	a	b	V

Fig. 6. The $\preceq_\omega$-sorted list of conjugates of *bcaa* and *ccbab* together with the coloring.

Proof. The Item 1 follows from the symmetry of the transformation `ebwt`. The proof of Item 2 comes from the fact that if u and v are conjugate then $M_\gamma(u, v) = (UV)^{|u|}$, hence for every $i = 1, \ldots, k$ one has that $n_i = 1$. Note that we use a stable ordering of the cyclic rotations of u and v in the computation of $\texttt{ebwt}(u, v)$. Item 3 holds because the definition of δ uses the transformation `ebwt` but does not depend on the sequence M_χ. □

Example 29. Let us consider the sequences $u = bcaa$ and $v = ccbab$. In Fig. 6 the sorted list of conjugates of u and v and the coloring of $\texttt{ebwt}(u, v)$ are shown. In this case $M_\gamma(u, v) = U^2V^2UVUV^2$, so it is easy to compute $\delta(u, v) = 3$.

Actually, even if we refer to δ as a "distance" between conjugacy classes, it is not a metric because neither it does obey the triangle inequality (see Example 30) nor the condition $\delta(u, v) = 0$ implies that u and v are conjugate (see Example 31).

Example 30. Consider the sequences $u_1 = abaab$, $u_2 = babab$, $u_3 = abbba$. It is possible to compute that

$$M_\gamma(u_1, u_2) = U_1^3U_2^2U_1^2U_2^3,$$
$$M_\gamma(u_2, u_3) = U_3U_2^2U_3^2U_2^2U_3U_2U_3,$$
$$M_\gamma(u_1, u_3) = U_1U_3U_1^2U_3U_1U_3U_1U_3^2.$$

Such sequences show that δ does not satisfy the triangle inequality. In fact $\delta(u_1, u_2) = 6$, $\delta(u_2, u_3) = 3$ and $\delta(u_1, u_3) = 2$.

Example 31. Let $u = aabc$ and $v = abbc$. Although the two sequences are not conjugate, $\delta(u, v) = 0$ since $M_\gamma(u, v) = (UV)^4$.

The following two propositions show that if we endow A^* with an equivalence relation called Parikh-equivalence, that we define in the following, then the measure δ is a semi-metric on conjugacy classes.

Let $A = \{a_1, a_2, \ldots, a_t\}$ be a finite ordered alphabet. We can associate to a word $w \in A^*$ its *Parikh-vector* $P(w)$, i.e. a t-tuple of non-negative integers $(i_1, i_2, \ldots, i_t)$ such that each i_j counts how many a_j's occur in w.

Two words $u, v \in A^*$ are called *Parikh-equivalent* if $P(u) = P(v)$. We note that if two words are Parikh-equivalent then they have the same length. In this section we show that if we consider the Parikh-equivalent words in A^*, then measure δ becomes a semi-metric, i.e. it is a positive measure that satisfies the symmetry, the identity of indiscernibles but not the triangle inequality (see Example 34).

Proposition 32. *Let u, v be two Parikh-equivalent words in A^*. If $\delta(u, v) = 0$ then u and v are conjugate.*

Proof. We denote by n the length of w and v. Let us consider $M_\gamma(u, v) = U^{n_1}V^{n_2}U^{n_3}\cdots V^{n_k}$. Since $\delta(u, v) = 0$, from definition of δ it follows that $n_1 = 0$ or 1 and $n_i = 1$ for each $i \geq 2$. Moreover, since $|u| = |v|$ then $n_1 = 0$ if and only if $n_k = 0$, i.e. $M_\gamma(u, v) = (UV)^n$ or $(VU)^n$ of length $2n$. Let θ be the permutation on $\{1, \ldots, 2n\}$ defined as in Section 2 from $M_F(u, v)$ to $M_L(u, v)$. It is easy to see that θ is decomposable into two disjoint cycles $\sigma = (i_1 i_2 \cdots i_n)$ and $\tau = (j_1 j_2 \cdots j_n)$ where i_h and j_h are even and odd integers, respectively, for each $1 \leq h \leq n$. Let $M_L(u, v) = s_1 s_2 \cdots s_n$ be the extended transformation, where $s_i \in A^2$. In order to prove that u and v are conjugate it suffices to show that s_i is a square, for each $i \geq 1$. This is equivalent to proving that $\theta(2i - 1) = h \Rightarrow \theta(2i) = h + 1$. If we prove such statement the thesis follows because, if we use the Proposition 25

then we can prove that $\mathcal{BWT}(u) = \mathcal{BWT}(v)$. In fact, since $M_L(u, v) = \mathcal{EBWT}(u, v) \in \mathcal{BWT}(u) \circ \mathcal{BWT}(v)$, we can deduce that the elements of $M_L(u, v)$ are taken alternatively one from $\mathcal{BWT}(u)$ and one from $\mathcal{BWT}(v)$. Let us suppose that $1 \leq j \leq n$ is the first index such that $s_j = xy$, where $x, y \in A$ and $x \neq y$. Then there should be an integer i such that $M_F(u, v)[2i] = y$. Since w and v are Parikh-equivalent, each character a_i has the same number of occurrences both in w and v. So $M_F(u, v)$ contains an even number of occurrences of y, we have that $M_F(u, v)[2i - 1] = y$. So, by Item 2 of Proposition 14, there exists $2k - 1 < 2j$ such that $\theta(2i - 1) = 2k - 1$. Since $x \neq y$, then $2k - 1 < 2j - 2$. We note that in $M_L(u, v)$ the character y cannot appear between positions $2k$ and $2j - 1$. So, $s_k = yz$ with $y \neq z$ appears in $M_L(u, v)$ and $k < j$ and this fact contradicts the minimality of j. The thesis follows by using Proposition 1. □

We note that given two words u and v having the same length n, the fact that $\delta(u, v) = 0$ does not imply in general that u and v are conjugate (see Example 31). In case of words over a binary alphabet $\{a, b\}$, the equality of the length implies the Parikh-equivalence.

Proposition 33. *Let u and v be two words over the alphabet $\{a, b\}$ having the same length n. If $\delta(u, v) = 0$ then u and v are conjugate.*

Proof. As in the proof of previous proposition, $\delta(u, v) = 0$ and $|u| = |v|$ imply that $M_\gamma(u, v) = (UV)^n$ or $(VU)^n$ and its length is $2n$. Let θ be the permutation on $\{1, \ldots, 2n\}$ defined as in Section 2 from $M_F(u, v)$ to $M_L(u, v)$. It is easy to see that θ is decomposable into two disjoint cycles $\sigma = (i_1 i_2 \cdots i_n)$ and $\tau = (j_1 j_2 \cdots j_n)$ where i_h and j_h are even and odd integers, respectively, for each $1 \leq h \leq n$. Denoted by $|u|_a$ the number of occurrences of the character a in u, we note that since u and v are words over a binary alphabet we can deduce that $|u|_a = |v|_a$. Note that if $|u|_a \geq |v|_a + 2$, then there would be two consecutive U's in $M^\gamma(u, v)$, a contradiction. Then we can state that $||u|_a - |v|_a| \leq 1$. Actually, if $|u|_a = |v|_a + 1$ and $|u|_b + 1 = |v|_b$ then the first occurrence of b in $M_F(u, v)$ should appear at an even position $2i$. So, by definition of θ and since b must appear as first symbol in $M_L(u, v)$, we should have $\theta(2i) = 1$ and this contradicts the fact that σ and τ are disjoint. Since $|u|_a = |v|_a$ and $|u|_b = |v|_b$, u and v are Parikh-equivalent. By Proposition 32, it follows that u and v are conjugate. □

Example 34. We note that, even if we consider Parikh-equivalent words, the measure δ is a semi-metric, i.e. it does not satisfy the triangle inequality. In fact, let $u = aaaaaabbbb$, $v = aaabaababb$ and $w = aaabaabbba$ be three Parikh-equivalent words. One can verify that $\delta(u, v) = 9$, $\delta(u, w) = 5$ and $\delta(w, v) = 3$.

We remark that the computation time of the distance $\delta(u, v)$ linearly depends on the computation time of the extended transform of u and v. Therefore its computation is performed in $O(|u|+|v|)$. This distance is then very simple to compute, differently from other alignment-free distance measures, such as, for instance, the Block-Edit distance, whose computation is an NP-Complete problem (cf. [10]). We recall that the Block-Edit distance between two words u and v computes the number of character insertion, deletion and substitution and block insertion, deletion and relocation needed in order to obtain v from u. One can wonder whether the distance δ can be considered an approximation of the Block-Edit distance. Unfortunately this is not the case. In fact, consider, for instance, the alphabet $A \cup \{\$_1, \$_2, \$_3, \$_4\}$ such that, for all $a \in A$, $a < \$_1 < \$_2 < \$_3 < \$_4$ and consider the words $u = a_1 a_2 \cdots a_n \$_1 a_1 a_2 \cdots a_n \$_2$ and $v = a_1 a_2 \cdots a_n \$_3 a_1 a_2 \cdots a_n \$_4$, $a_i \in A$ for all $i = 1, \ldots, n$. It is easy to see that just two character substitutions allow to obtain v from u, so their Block Edit distance is 2. On the other side one can notice that for a fixed character a_i, its two occurrences in u are close in the transformation, followed by its two occurrences in v. So each block of characters coming from the same word have length 2, which give a distance $\delta(u, v) = |u|$.

6. Experimental results

The distance introduced in previous section measures the dissimilarity between two cyclic sequences (cf. Item 3 of Proposition 28).[1] So, in order to test our method, in this section we describe the results of the application of the normalized version of our distance to the whole mitochondrial genome phylogeny, since the mitochondrial DNA can be considered as a cyclic sequence. Notice the length of the mitochondrial DNA of the different species considered are almost the same (around 16 kbytes).

[1] Recall that in order to consider not cyclic sequences, it suffices to add an end-of-string symbol # to the sequences.

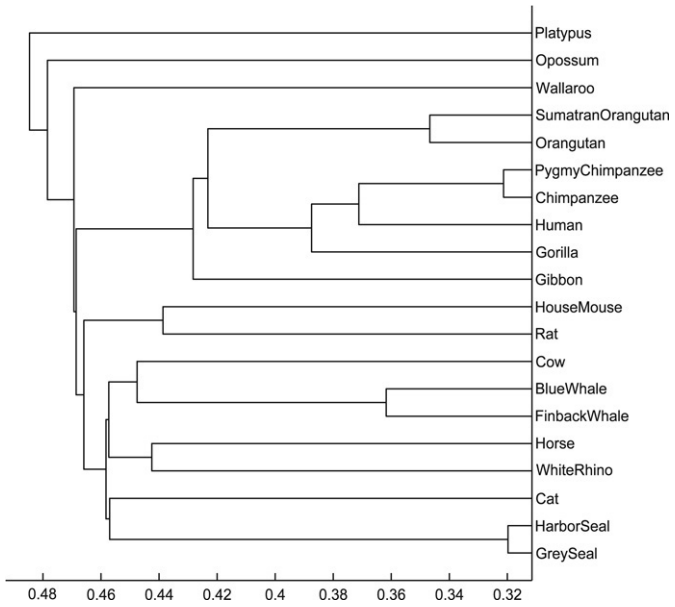

Fig. 7. The evolutionary tree built from complete mammalian mtDNA sequences of the 20 species analyzed in [2].

In the experiment described in this section we construct a phylogeny of the Eutherian orders (i.e. placental mammals) using complete unaligned mitochondrial genomes. We choose our group of sequences by using the mtDNA genomes of 20 mammals from *GenBank*, as listed in Fig. 7. Such a set of species contains placental mammals, and, as outgroups, marsupials and monotremes. As shown in the dendrogram obtained by using a single linkage clustering (see Fig. 7), our method allows to classify the analyzed species into Primates, Ferungulates, Rodents, Marsupial and Monotremes. Moreover, we obtain a phylogeny that is very close to the ones described in most of the papers in which the species considered are almost the same (cf. [18,2,9,3]). Our resulting phylogeny proposes the following grouping of the placental mammals: (Primates, (Ferungulates, Rodents)). A more detailed analysis and further experiments using different `ebwt`-based distance measures can be found in [14] and [15].

Nevertheless, the goal of this experiment is not to confirm or refute previous phylogenetic studies but rather to show that the method introduced here can be a helpful tool for the comparative genomics research community.

References

[1] M. Burrows, D.J. Wheeler, A block sorting data compression algorithm. Technical report, DIGITAL System Research Center, 1994.
[2] Y. Cao, A. Janke, P.J. Waddell, M. Westerman, O. Takenaka, S. Murata, N. Okada, S. Pääbo, M. Hasegawa, Conflict among individual mitochondrial proteins in resolving the phylogeny of eutherian orders, J. Mol. Evol. 47 (1998) 307–322.
[3] R. Cilibrasi, P. Vitányi, Clustering by compression, IEEE Trans. Inform. Theory 51 (4) (2005) 1523–1545.
[4] M. Crochemore, J. Désarménien, D. Perrin, A note on the Burrows–Wheeler transformation, Theoret. Comput. Sci. 332 (2005) 567–572.
[5] F. Ergun, S. Muthukrishnan, C. Sahinalp, Comparing sequences with segment rearrangements, in: Proc. of the FSTTCS'03, Bombay, India, in: Lecture Notes in Comput. Sci, 2003, pp. 183–194.
[6] P. Ferragina, F. Luccio, G. Manzini, S. Muthukrishnan, Structuring labeled trees for optimal succinctness, and beyond, in: Proc. of the 45th Annual IEEE Symposium on Foundations of Computer Science, 2005, pp. 198–207.
[7] I.M. Gessel, C. Reutenauer, Counting permutations with given cycle structure and descent set, J. Combin. Theory Ser. A 64 (2) (1993) 189–215.
[8] J. Kärkkäinen, P. Sanders, Simple linear work suffix array construction, in: Proc. ICALP, 2003, pp. 943–955.
[9] M. Li, X. Chen, X. Li, B. Ma, P. Vitányi, The similarity metric, IEEE Trans. Inform. Theory 12 (5) (2004) 3250–3264.
[10] D. Lopresti, A. Tomkins, Block edit models for approximate string matching, Theoret. Comput. Sci. 181 (1) (1997) 159–179.
[11] M. Lothaire, Combinatorics on Words, in: Encyclopedia of Mathematics, vol. 17, Addison-Wesley, Reading, MA, 1983 (Reprinted in the Cambridge Mathematical Library, Cambridge University Press, 1997).
[12] M. Lothaire, Algebraic Combinatorics on Words, Cambridge University Press, 2002.
[13] S. Mantaci, A. Restivo, G. Rosone, M. Sciortino, An extension of the Burrows Wheeler transform and applications to sequence comparison and data compression, in: Alberto Apostolico, Maxime Crochemore, Kunsoo Park (Eds.), Combinatorial Pattern Matching, 16th Annual Symposium, CPM 2005, 19–22 June, Jeju Island, Korea, 2005, Proceedings, in: Lecture Notes in Computer Science, vol. 3537, Springer, 2005, pp. 178–189.

[14] S. Mantaci, A. Restivo, G. Rosone, M. Sciortino, A new combinatorial approach to sequence comparison, in: Mario Coppo, Elena Lodi, G. Michele Pinna (Eds.), Proceedings of ICTCS 2005, Theoretical Computer Science, 9th Italian Conference, ICTCS 2005, 12–14 October, Siena, Italy, 2005, in: Lecture Notes in Computer Science, vol. 3701, Springer, 2005, pp. 348–359.
[15] S. Mantaci, A. Restivo, G. Rosone, M. Sciortino, A new combinatorial approach to sequence comparison (journal version), Theory Comput. Syst. (in press).
[16] S. Mantaci, A. Restivo, M. Sciortino, Burrows-Wheeler transform and Sturmian words, Inform. Process. Lett. 86 (2003) 241–246.
[17] H.S. Wilf, N.J. Fine, Uniqueness theorem for periodic functions, Proc. Amer. Math. Soc. 16 (1965) 109–114.
[18] H.H. Otu, K. Sayood, A new sequence distance measure for phylogenetic tree construction, Bioinformatics 19 (16) (2003) 2122–2130.
[19] S. Vinga, J. Almeida, Alignment-free sequence comparison – a review, Bioinformatics 19 (4) (2003) 513–523.