Università degli Studi di Palermo
Dipartimento di Matematica e Informatica
Dottorato di Ricerca in Matematica e Informatica

Balancing and clustering of words: a combinatorial analysis of the Burrows & Wheeler Transform

Giovanna Rosone
giovanna@math.unipa.it

Advisor:
Prof. Antonio Restivo

Coordinator:
Prof. Camillo Trapani

Palermo, 17 Marzo 2010

Burrows-Wheeler Transform (BWT)

- The BWT is a well known transformation introduced in [M. Burrows and D. Wheeler, A block sorting data compression algorithm, Technical report, DIGITAL System Research Center, 1994]
- The BWT is a reversible transformation that produces a permutation $bwt(v)$ of an input sequence v, defined over an ordered alphabet $\mathcal{A}$, so that occurrences of a given symbol tend to occur in clusters in the output sequence.
- Traditionally the major application of the Burrows-Wheeler Transform has been for Data Compression. The BWT represents for instance the heart of the BZIP2 algorithm.
- Today, there are reports of the application of the BWT in bio-informatics, full-text compressed indexes, prediction and entropy estimation, and shape analysis in computer vision, etc. Moreover, there exist several variants and extensions of such a transform.

Burrows-Wheeler Transform (BWT)

- The BWT is a well known transformation introduced in [M. Burrows and D. Wheeler, A block sorting data compression algorithm, Technical report, DIGITAL System Research Center, 1994]
- The BWT is a reversible transformation that produces a permutation $bwt(v)$ of an input sequence v, defined over an ordered alphabet $\mathcal{A}$, so that occurrences of a given symbol tend to occur in clusters in the output sequence.
- Traditionally the major application of the Burrows-Wheeler Transform has been for Data Compression. The BWT represents for instance the heart of the BZIP2 algorithm.
- Today, there are reports of the application of the BWT in bio-informatics, full-text compressed indexes, prediction and entropy estimation, and shape analysis in computer vision, etc. Moreover, there exist several variants and extensions of such a transform.

Burrows-Wheeler Transform (BWT)

- The BWT is a well known transformation introduced in [M. Burrows and D. Wheeler, A block sorting data compression algorithm, Technical report, DIGITAL System Research Center, 1994]
- The BWT is a reversible transformation that produces a permutation $bwt(v)$ of an input sequence v, defined over an ordered alphabet $\mathcal{A}$, so that occurrences of a given symbol tend to occur in clusters in the output sequence.

- Traditionally the major application of the Burrows-Wheeler Transform has been for Data Compression. The BWT represents for instance the heart of the BZIP2 algorithm.
- Today, there are reports of the application of the BWT in bio-informatics, full-text compressed indexes, prediction and entropy estimation, and shape analysis in computer vision, etc. Moreover, there exist several variants and extensions of such a transform.

Preliminaries

- Let $\mathcal{A}$ denote a non-empty finite set of symbols. The elements of $\mathcal{A}$ are called letters (symbols or characters) and the set $\mathcal{A}$ is called an alphabet.
- A word over an alphabet $\mathcal{A}$ is a finite sequence of letters from $\mathcal{A}$.
- The empty word ε is the empty sequence.

- Two words $u, v \in \mathcal{A}^*$ are conjugate, if $u = xy$ and $v = yx$ for some $x, y \in \mathcal{A}^*$. Thus conjugate words are just cyclic shifts of one another.
- Let $[v]$ denote the conjugacy classes of v.
- A conjugacy class can also be represented as a circular word. Hence in what follows we will use "circular word" and "conjugacy class" as synonym.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I

- Example: $v = international$

$i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l$	1	$a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n$
$n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i$	2	$a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n$
$t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n$	3	$e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t$
$e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t$	4	$i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l$
$r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e$	5	$i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t$
$n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r$	6	$l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a$
$a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n$	7	$n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o$
$t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a$	8	$n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r$
$i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t$	9	$n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i$
$o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i$	10	$o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i$
$n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o$	11	$r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e$
$a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n$	12	$t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n$
$l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a$	13	$t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
    - a permutation $bwt(v)$ of the letters of v.
    - the index I
- Example: $v = international$

$international$	1	$alinternation$
$nternationali$	2	$ationalintern$
$ternationalin$	3	$ernationalint$
$ernationalint$	4	$international$
$rnationalinte$	5	$ionalinternat$
$nationalinter$	6	$linternationa$
$ationalintern$	7	$nalinternatio$
$tionalinterna$	8	$nationalinter$
$ionalinternat$	9	$nternationali$
$onalinternati$	10	$onalinternati$
$nalinternatio$	11	$rnationalinte$
$alinternation$	12	$ternationalin$
$linternationa$	13	$tionalinterna$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I

- Example: $v = international$

$$
\begin{array}{ccccccccccccc}
i & n & t & e & r & n & a & t & i & o & n & a & l \\
n & t & e & r & n & a & t & i & o & n & a & l & i \\
t & e & r & n & a & t & i & o & n & a & l & i & n \\
e & r & n & a & t & i & o & n & a & l & i & n & t \\
r & n & a & t & i & o & n & a & l & i & n & t & e \\
n & a & t & i & o & n & a & l & i & n & t & e & r \\
a & t & i & o & n & a & l & i & n & t & e & r & n \\
t & i & o & n & a & l & i & n & t & e & r & n & a \\
i & o & n & a & l & i & n & t & e & r & n & a & t \\
o & n & a & l & i & n & t & e & r & n & a & t & i \\
n & a & l & i & n & t & e & r & n & a & t & i & o \\
a & l & i & n & t & e & r & n & a & t & i & o & n \\
l & i & n & t & e & r & n & a & t & i & o & n & a
\end{array}
\qquad
\begin{array}{r|ccccccccccccc}
1 & a & l & i & n & t & e & r & n & a & t & i & o & n \\
2 & a & t & i & o & n & a & l & i & n & t & e & r & n \\
3 & e & r & n & a & t & i & o & n & a & l & i & n & t \\
4 & i & n & t & e & r & n & a & t & i & o & n & a & l \\
5 & i & o & n & a & l & i & n & t & e & r & n & a & t \\
6 & l & i & n & t & e & r & n & a & t & i & o & n & a \\
7 & n & a & l & i & n & t & e & r & n & a & t & i & o \\
8 & n & a & t & i & o & n & a & l & i & n & t & e & r \\
9 & n & t & e & r & n & a & t & i & o & n & a & l & i \\
10 & o & n & a & l & i & n & t & e & r & n & a & t & i \\
11 & r & n & a & t & i & o & n & a & l & i & n & t & e \\
12 & t & e & r & n & a & t & i & o & n & a & l & i & n \\
13 & t & i & o & n & a & l & i & n & t & e & r & n & a
\end{array}
$$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I
- Example: $v = international$

		M
$i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l$	1	$a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n$
$n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i$	2	$a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n$
$t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n$	3	$e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t$
$e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t$	4	$i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l$
$r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e$	5	$i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t$
$n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r$	6	$l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a$
$a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n$	7	$n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o$
$t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a$	8	$n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r$
$i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t$	9	$n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i$
$o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i$	10	$o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i$
$n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o$	11	$r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e$
$a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n$	12	$t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n$
$l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a$	13	$t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I

- Example: $v = international$

i	n	t	e	r	n	a	t	i	o	n	a	l
n	t	e	r	n	a	t	i	o	n	a	l	i
t	e	r	n	a	t	i	o	n	a	l	i	n
e	r	n	a	t	i	o	n	a	l	i	n	t
r	n	a	t	i	o	n	a	l	i	n	t	e
n	a	t	i	o	n	a	l	i	n	t	e	r
a	t	i	o	n	a	l	i	n	t	e	r	n
t	i	o	n	a	l	i	n	t	e	r	n	a
i	o	n	a	l	i	n	t	e	r	n	a	t
o	n	a	l	i	n	t	e	r	n	a	t	i
n	a	l	i	n	t	e	r	n	a	t	i	o
a	l	i	n	t	e	r	n	a	t	i	o	n
l	i	n	t	e	r	n	a	t	i	o	n	a

M

	F ↓												L ↓
1	a	l	i	n	t	e	r	n	a	t	i	o	n
2	a	t	i	o	n	a	l	i	n	t	e	r	n
3	e	r	n	a	t	i	o	n	a	l	i	n	t
4	i	n	t	e	r	n	a	t	i	o	n	a	l
5	i	o	n	a	l	i	n	t	e	r	n	a	t
6	l	i	n	t	e	r	n	a	t	i	o	n	a
7	n	a	l	i	n	t	e	r	n	a	t	i	o
8	n	a	t	i	o	n	a	l	i	n	t	e	r
9	n	t	e	r	n	a	t	i	o	n	a	l	i
10	o	n	a	l	i	n	t	e	r	n	a	t	i
11	r	n	a	t	i	o	n	a	l	i	n	t	e
12	t	e	r	n	a	t	i	o	n	a	l	i	n
13	t	i	o	n	a	l	i	n	t	e	r	n	a

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that denotes the position of the original word v after the lexicographical sorting of its conjugates.
- Example: $v = international$

$$
\begin{array}{ccccccccccccc}
i & n & t & e & r & n & a & t & i & o & n & a & l \\
n & t & e & r & n & a & t & i & o & n & a & l & i \\
t & e & r & n & a & t & i & o & n & a & l & i & n \\
e & r & n & a & t & i & o & n & a & l & i & n & t \\
r & n & a & t & i & o & n & a & l & i & n & t & e \\
n & a & t & i & o & n & a & l & i & n & t & e & r \\
a & t & i & o & n & a & l & i & n & t & e & r & n \\
t & i & o & n & a & l & i & n & t & e & r & n & a \\
i & o & n & a & l & i & n & t & e & r & n & a & t \\
o & n & a & l & i & n & t & e & r & n & a & t & i \\
n & a & l & i & n & t & e & r & n & a & t & i & o \\
a & l & i & n & t & e & r & n & a & t & i & o & n \\
l & i & n & t & e & r & n & a & t & i & o & n & a
\end{array}
$$

$$
\begin{array}{r|ccccccccccccc}
 & \multicolumn{13}{c}{M} \\
 & F & & & & & & & & & & & & L \\
 & \downarrow & & & & & & & & & & & & \downarrow \\
1 & a & l & i & n & t & e & r & n & a & t & i & o & n \\
2 & a & t & i & o & n & a & l & i & n & t & e & r & n \\
3 & e & r & n & a & t & i & o & n & a & l & i & n & t \\
I \rightarrow 4 & i & n & t & e & r & n & a & t & i & o & n & a & l \\
5 & i & o & n & a & l & i & n & t & e & r & n & a & t \\
6 & l & i & n & t & e & r & n & a & t & i & o & n & a \\
7 & n & a & l & i & n & t & e & r & n & a & t & i & o \\
8 & n & a & t & i & o & n & a & l & i & n & t & e & r \\
9 & n & t & e & r & n & a & t & i & o & n & a & l & i \\
10 & o & n & a & l & i & n & t & e & r & n & a & t & i \\
11 & r & n & a & t & i & o & n & a & l & i & n & t & e \\
12 & t & e & r & n & a & t & i & o & n & a & l & i & n \\
13 & t & i & o & n & a & l & i & n & t & e & r & n & a
\end{array}
$$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that denotes the position of the original word v after the lexicographical sorting of its conjugates.
- Example: $v = international$

$$
\begin{array}{ccccccccccccc}
i & n & t & e & r & n & a & t & i & o & n & a & l \\
n & t & e & r & n & a & t & i & o & n & a & l & i \\
t & e & r & n & a & t & i & o & n & a & l & i & n \\
e & r & n & a & t & i & o & n & a & l & i & n & t \\
r & n & a & t & i & o & n & a & l & i & n & t & e \\
n & a & t & i & o & n & a & l & i & n & t & e & r \\
a & t & i & o & n & a & l & i & n & t & e & r & n \\
t & i & o & n & a & l & i & n & t & e & r & n & a \\
i & o & n & a & l & i & n & t & e & r & n & a & t \\
o & n & a & l & i & n & t & e & r & n & a & t & i \\
n & a & l & i & n & t & e & r & n & a & t & i & o \\
a & l & i & n & t & e & r & n & a & t & i & o & n \\
l & i & n & t & e & r & n & a & t & i & o & n & a
\end{array}
$$

$$
M
$$

$$
\begin{array}{r|ccccccccccccc}
 & F & & & & & & & & & & & & L \\
 & \downarrow & & & & & & & & & & & & \downarrow \\
1 & a & l & i & n & t & e & r & n & a & t & i & o & n \\
2 & a & t & i & o & n & a & l & i & n & t & e & r & n \\
3 & e & r & n & a & t & i & o & n & a & l & i & n & t \\
I \rightarrow 4 & i & n & t & e & r & n & a & t & i & o & n & a & l \\
5 & i & o & n & a & l & i & n & t & e & r & n & a & t \\
6 & l & i & n & t & e & r & n & a & t & i & o & n & a \\
7 & n & a & l & i & n & t & e & r & n & a & t & i & o \\
8 & n & a & t & i & o & n & a & l & i & n & t & e & r \\
9 & n & t & e & r & n & a & t & i & o & n & a & l & i \\
10 & o & n & a & l & i & n & t & e & r & n & a & t & i \\
11 & r & n & a & t & i & o & n & a & l & i & n & t & e \\
12 & t & e & r & n & a & t & i & o & n & a & l & i & n \\
13 & t & i & o & n & a & l & i & n & t & e & r & n & a
\end{array}
$$

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

How does BWT work?

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that denotes the position of the original word v after the lexicographical sorting of its conjugates.
- Example: $v = international$

$$\begin{array}{ccccccccccccc}
i & n & t & e & r & n & a & t & i & o & n & a & l \\
n & t & e & r & n & a & t & i & o & n & a & l & i \\
t & e & r & n & a & t & i & o & n & a & l & i & n \\
e & r & n & a & t & i & o & n & a & l & i & n & t \\
r & n & a & t & i & o & n & a & l & i & n & t & e \\
n & a & t & i & o & n & a & l & i & n & t & e & r \\
a & t & i & o & n & a & l & i & n & t & e & r & n \\
t & i & o & n & a & l & i & n & t & e & r & n & a \\
i & o & n & a & l & i & n & t & e & r & n & a & t \\
o & n & a & l & i & n & t & e & r & n & a & t & i \\
n & a & l & i & n & t & e & r & n & a & t & i & o \\
a & l & i & n & t & e & r & n & a & t & i & o & n \\
l & i & n & t & e & r & n & a & t & i & o & n & a
\end{array}$$

M

	$F \downarrow$												$L \downarrow$
1	a	l	i	n	t	e	r	n	a	t	i	o	n
2	a	t	i	o	n	a	l	i	n	t	e	r	n
3	e	r	n	a	t	i	o	n	a	l	i	n	t
$I \rightarrow 4$	i	n	t	e	r	n	a	t	i	o	n	a	l
5	i	o	n	a	l	i	n	t	e	r	n	a	t
6	l	i	n	t	e	r	n	a	t	i	o	n	a
7	n	a	l	i	n	t	e	r	n	a	t	i	o
8	n	a	t	i	o	n	a	l	i	n	t	e	r
9	n	t	e	r	n	a	t	i	o	n	a	l	i
10	o	n	a	l	i	n	t	e	r	n	a	t	i
11	r	n	a	t	i	o	n	a	l	i	n	t	e
12	t	e	r	n	a	t	i	o	n	a	l	i	n
13	t	i	o	n	a	l	i	n	t	e	r	n	a

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

Each row of M is a conjugate of v in lexicographic order.

Properties

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that is useful in order to recover the original word v.
- Example: $v = international$
  $bwt(v) = L = nntltaoriiena$ and $I = 4$

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$$
\begin{array}{rr|ccccccccccccc}
 & & F & & & & & & M & & & & & & L \\
 & & \downarrow & & & & & & & & & & & & \downarrow \\
 & 1 & a & l & i & n & t & e & r & n & a & t & i & o & n \\
 & 2 & a & t & i & o & n & a & l & i & n & t & e & r & n \\
 & 3 & e & r & n & a & t & i & o & n & a & l & i & n & t \\
I \rightarrow & 4 & i & n & t & e & r & n & a & t & i & o & n & a & l \\
 & 5 & i & o & n & a & l & i & n & t & e & r & n & a & t \\
 & 6 & l & i & n & t & e & r & n & a & t & i & o & n & a \\
 & 7 & n & a & l & i & n & t & e & r & n & a & t & i & o \\
 & 8 & n & a & t & i & o & n & a & l & i & n & t & e & r \\
 & 9 & n & t & e & r & n & a & t & i & o & n & a & l & i \\
 & 10 & o & n & a & l & i & n & t & e & r & n & a & t & i \\
 & 11 & r & n & a & t & i & o & n & a & l & i & n & t & e \\
 & 12 & t & e & r & n & a & t & i & o & n & a & l & i & n \\
 & 13 & t & i & o & n & a & l & i & n & t & e & r & n & a
\end{array}
$$

Properties

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that is useful in order to recover the original word v.
- Example: $v = international$
  $bwt(v) = L = nntltaoriiena$ and $I = 4$

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

	F ↓						M						L ↓
1	a	l	i	n	t	e	r	n	a	t	i	o	n
2	a	t	i	o	n	a	l	i	n	t	e	r	n
3	e	r	n	a	t	i	o	n	a	l	i	n	t
$I \rightarrow$ 4	i	n	t	e	r	n	a	t	i	o	n	a	l
5	i	o	n	a	l	i	n	t	e	r	n	a	t
6	l	i	n	t	e	r	n	a	t	i	o	n	a
7	n	a	l	i	n	t	e	r	n	a	t	i	o
8	n	a	t	i	o	n	a	l	i	n	t	e	r
9	n	t	e	r	n	a	t	i	o	n	a	l	i
10	o	n	a	l	i	n	t	e	r	n	a	t	i
11	r	n	a	t	i	o	n	a	l	i	n	t	e
12	t	e	r	n	a	t	i	o	n	a	l	i	n
13	t	i	o	n	a	l	i	n	t	e	r	n	a

Properties

- BWT takes as input a text v and produces:
  - a permutation $bwt(v)$ of the letters of v.
  - the index I, that is useful in order to recover the original word v.
- Example: $v = international$
  $bwt(v) = L = nntltaoriiena$ and $I = 4$

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$$
\begin{array}{rr|ccccccccccccc}
 & & F & & & & & & M & & & & & & L \\
 & & \downarrow & & & & & & & & & & & & \downarrow \\
 & 1 & a & l & i & n & t & e & r & n & a & t & i & o & n \\
 & 2 & a & t & i & o & n & a & l & i & n & t & e & r & n \\
 & 3 & e & r & n & a & t & i & o & n & a & l & i & n & t \\
I \rightarrow & 4 & i & n & t & e & r & n & a & t & i & o & n & a & l \\
 & 5 & i & o & n & a & l & i & n & t & e & r & n & a & t \\
 & 6 & l & i & n & t & e & r & n & a & t & i & o & n & a \\
 & 7 & n & a & l & i & n & t & e & r & n & a & t & i & o \\
 & 8 & n & a & t & i & o & n & a & l & i & n & t & e & r \\
 & 9 & n & t & e & r & n & a & t & i & o & n & a & l & i \\
 & 10 & o & n & a & l & i & n & t & e & r & n & a & t & i \\
 & 11 & r & n & a & t & i & o & n & a & l & i & n & t & e \\
 & 12 & t & e & r & n & a & t & i & o & n & a & l & i & n \\
 & 13 & t & i & o & n & a & l & i & n & t & e & r & n & a
\end{array}
$$

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = i$

	F	L	
1	a	n	1
2	a	n	2
3	e	t	3
4	i	l	4
5	i	t	5
6	l	a	6
7	n	o	7
8	n	r	8
9	n	i	9
10	o	i	10
11	r	e	11
12	t	n	12
13	t	a	13

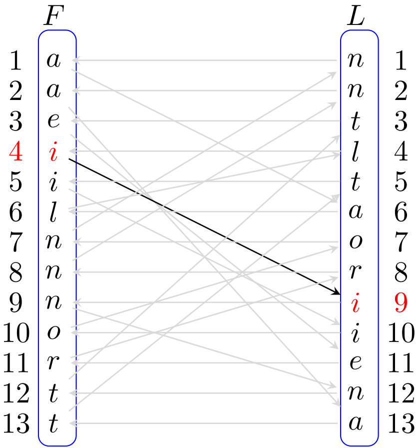
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = i$



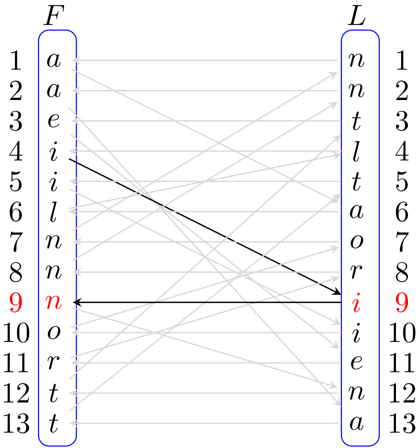
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = in$



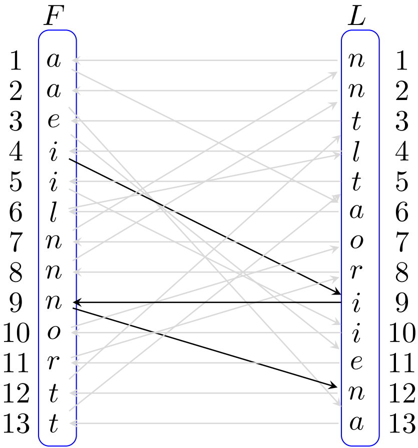
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = in$



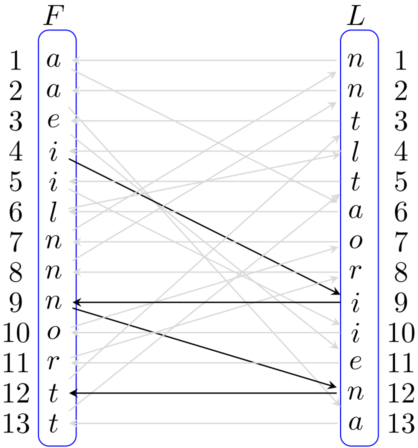
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = int$



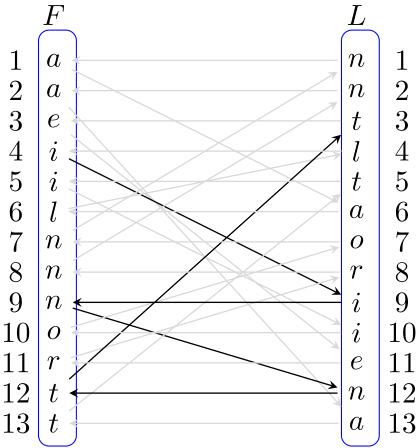
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = int$



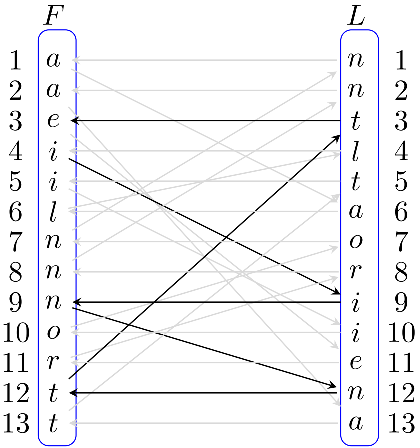
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = inte$



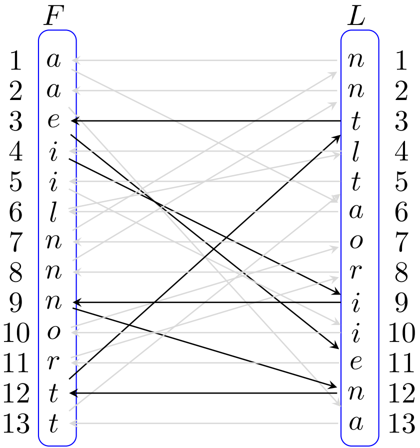
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = inte$



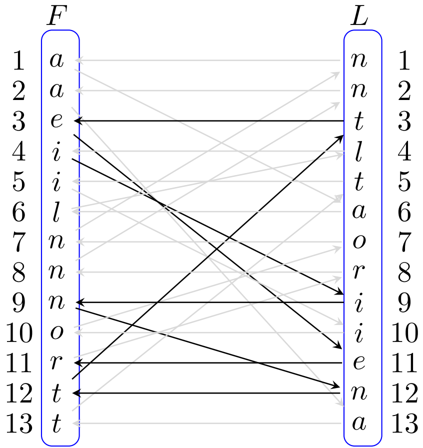
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = inter$



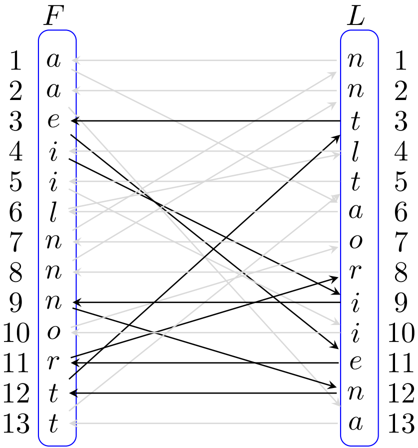
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = inter$



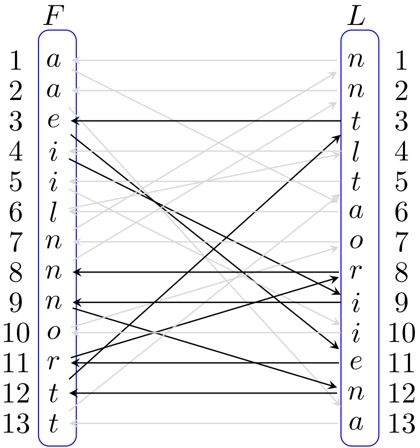
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = intern$



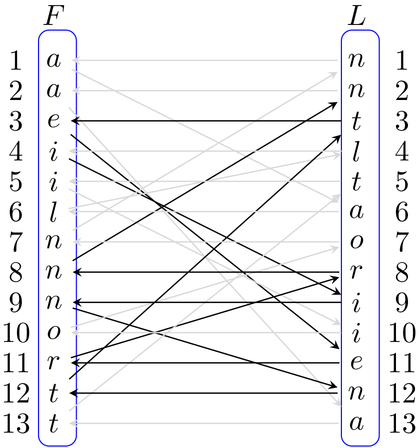
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = intern$



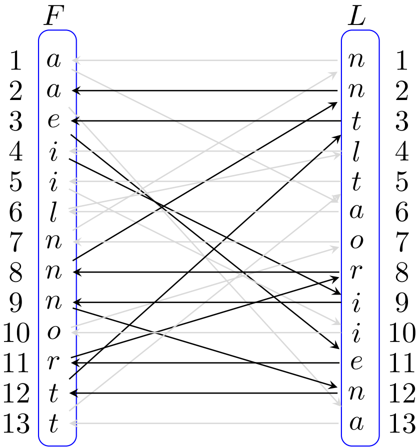
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = interna$



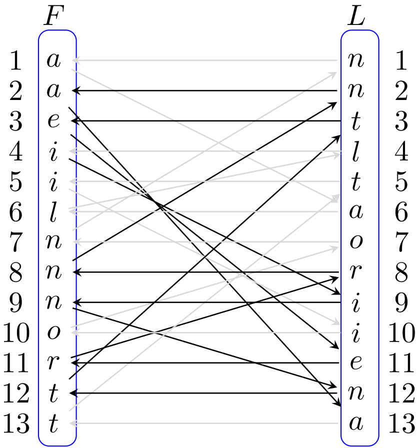
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = interna$



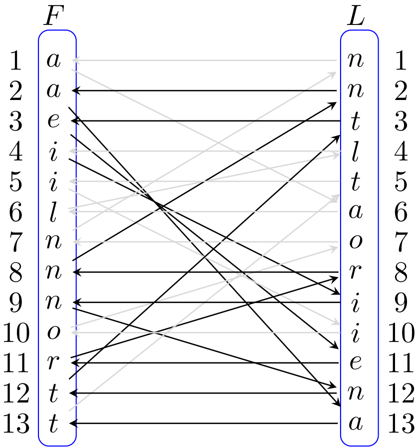
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internat$



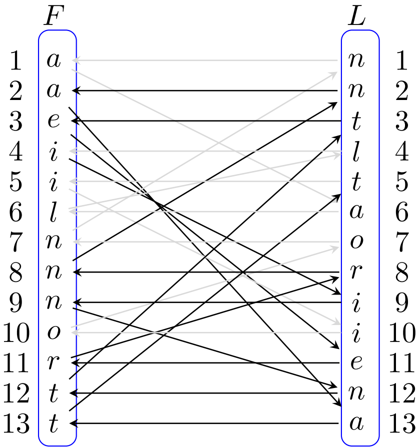
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internat$



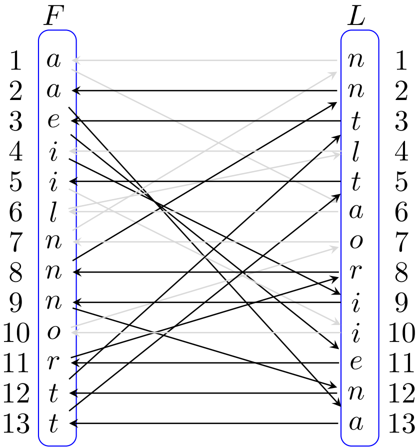
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internati$



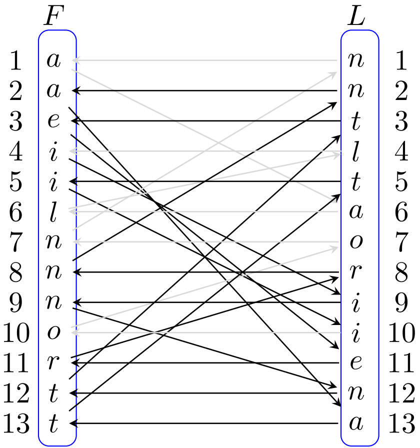
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internati$



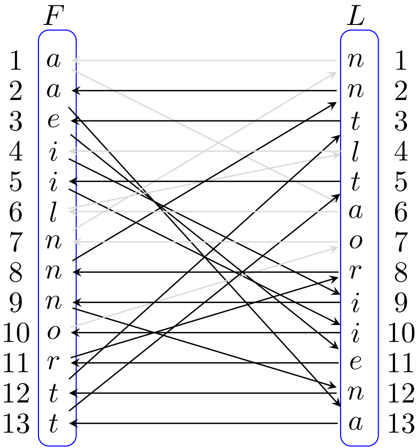
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internatio$



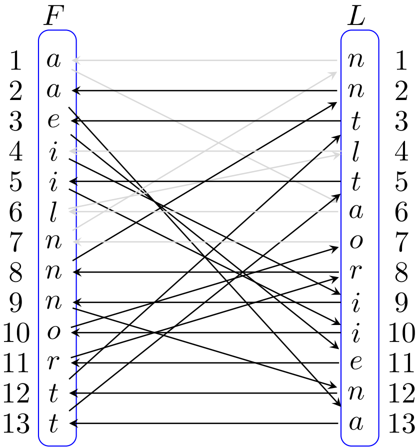
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internatio$



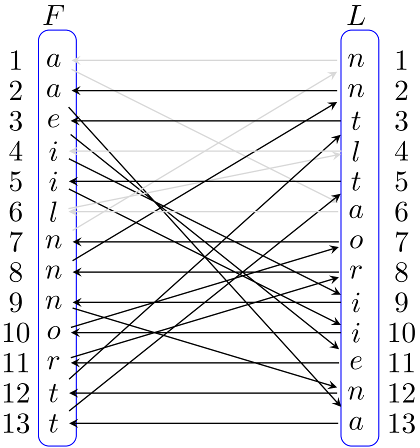
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internation$



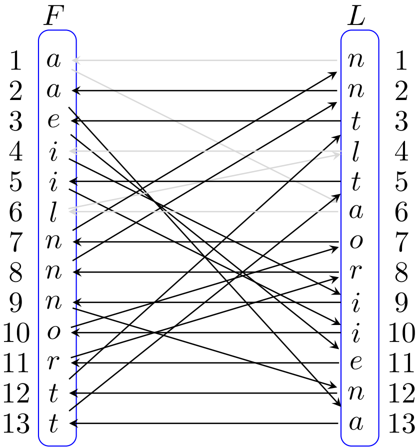
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internation$



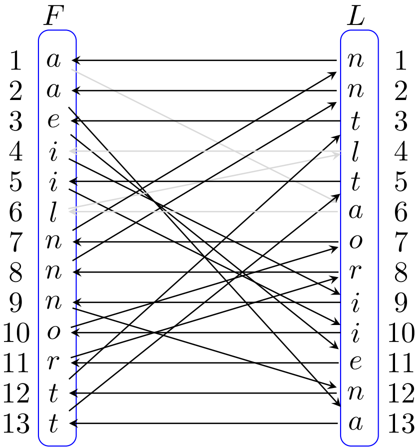
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internationa$



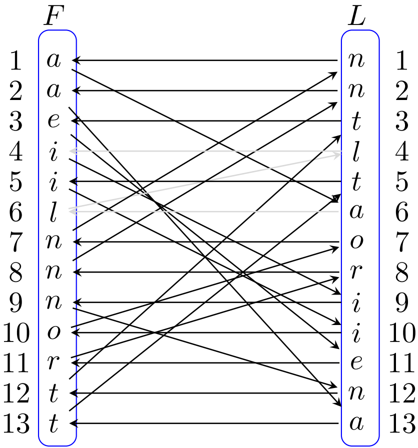
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = internationa$



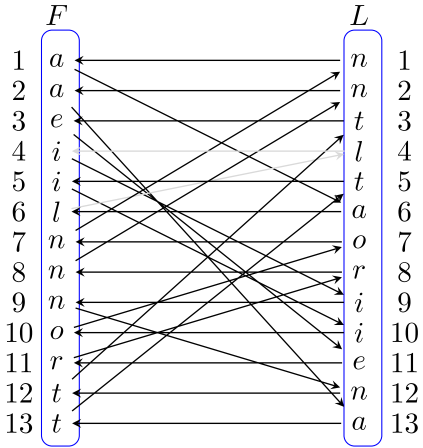
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = international$



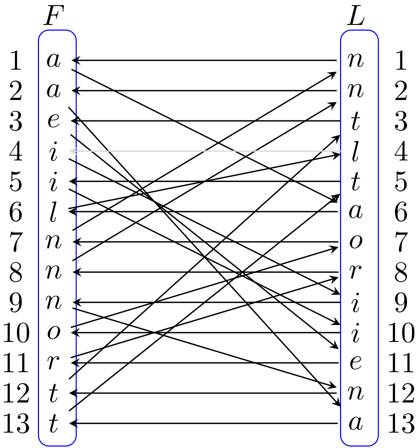
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = international$



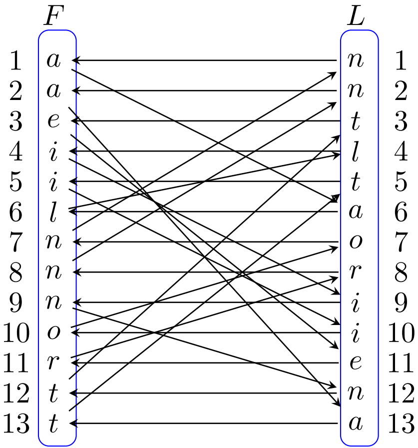
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = international$



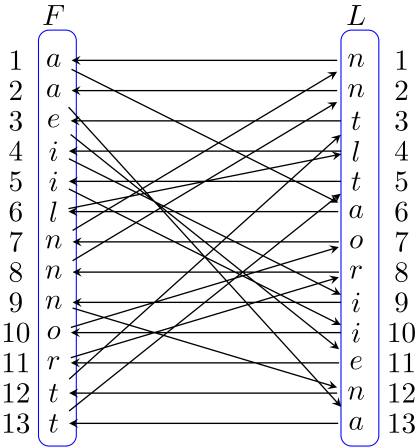
- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Reverse

$bwt(v) = L = nntltaoriiena$ and $I = 4$.

- The first character of v is $F[I]$.
- For any character α, the ith occurrence of α in F corresponds to the ith occurrence of α in L.
- For all $i \neq I$, the character $L[i]$ is followed in v by $F[i]$;

$v = international$



- Notice that if we except the index, all the mutual conjugate words have the same Burrows-Wheeler Transform.
- Hence, the BWT can be thought as a transformation acting on circular words.

Why Useful?

INTUITION

Let us consider the effect of BWT on a segment of a BWT-sorted file for Shakespeares Hamlet.

F	$\ldots L$
ot look upon his like again.	... n
ot look upon me; Lest with th	... n
ot love on the wing,-- As I p	... h
ot love your father; But that	... n
ot made them well, they imita	... n
ot madness That I have utter'	... n
ot me'? Ros. To think, my lor	... n
ot me; no, nor woman neither,	... n
ot me? Ham. No, by the rood,	... g
ot mend his pace with beating	... n
ot mine own. Besides, to be d	... n
ot mine. Ham. No, nor mine no	... n
ot mock me, fellow-student. I	... n
ot monstrous that this player	... n
ot more like. Ham. But where	... n
ot more native to the heart,	... n
ot more ugly to the thing tha	... n
ot more, my lord. Ham. Is not	... j
ot move thus. Oph. You must s	... n
ot much approve me.--Well, si	... n

The factor *ot* is normally preceded by *n*, but occasionally by *h*, *g* or *j*.

The characters preceding *ot* are grouped together.

Extensive experimental work confirms this “clustering effect” (M. Burrows and D. Wheeler,1994, P. Fenwick, 1996).

BWT-based compression - Intuition

$$v \longrightarrow \boxed{BWT} \longrightarrow bwt(v) \longrightarrow \boxed{\text{Compressor}} \longrightarrow \text{Output}$$

- Traditionally, compression ratio of BWT-based compression algorithms are usually measured by using $H_k(v)$.
  - G. Manzini, 2001,
  - F. Ferragina, R. Giancarlo, G. Manzini, M. Sciortino, 2005
- $H_0(v)$: Maximum compression we can get without context information where a fixed codeword is assigned to each alphabet character (e.g.: Huffman code).
- $H_k(v)$: Lower bound for compression with order-k contexts: the codeword representing each symbol depends on the k symbols preceding it.

BWT-based compression - Intuition

$v \longrightarrow \boxed{BWT} \longrightarrow bwt(v) \longrightarrow \boxed{\text{Compressor}} \longrightarrow \text{Output}$

- Traditionally, compression ratio of BWT-based compression algorithms are usually measured by using $H_k(v)$.
  - G. Manzini, 2001,
  - F. Ferragina, R. Giancarlo, G. Manzini, M. Sciortino, 2005

- $H_0(v)$: Maximum compression we can get without context information where a fixed codeword is assigned to each alphabet character (e.g.: Huffman code).

- $H_k(v)$: Lower bound for compression with order-k contexts: the codeword representing each symbol depends on the k symbols preceding it.

Questions

- H. Kaplan, S. Landau and E. Verbin, 2007, report in their paper some empirical results which seem to indicate that achieving good bounds with respect to H_k does not necessarily guarantee good compression results in practice. So they ask the question:

  whether there is another statistic (more appropriate than H_k) that actually capture the compressibility of the input text.

- H. Kaplan and E. Verbin, 2007 observe that such compressors work well in practice (in particular on English text). They ask the following question:

  what kind of regularity is there in English text that compressors exploit?

Answers

What kind of regularity is there?

The solution:
Balance of the input text.
Our idea is that one obtains a more compressible string as output of BWT if its input is very close to be balanced.

Is there a more appropriate statistic?

The solution:
Local Entropy of the input text.
We introduce the notion of local entropy as a measure of the degree of balance of a text.

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

Answers

What kind of regularity is there?

The solution:
Balance of the input text.
Our idea is that one obtains a more compressible string as output of BWT if its input is very close to be balanced.

Is there a more appropriate statistic?

The solution:
Local Entropy of the input text.
We introduce the notion of local entropy as a measure of the degree of balance of a text.

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

Answers

What kind of regularity is there?

The solution:
Balance of the input text.
Our idea is that one obtains a more compressible string as output of BWT if its input is very close to be balanced.

Is there a more appropriate statistic?

The solution:
Local Entropy of the input text.
We introduce the notion of local entropy as a measure of the degree of balance of a text.

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

Answers

What kind of regularity is there?

The solution:
Balance of the input text.
Our idea is that one obtains a more compressible string as output of BWT if its input is very close to be balanced.

Is there a more appropriate statistic?

The solution:
Local Entropy of the input text.
We introduce the notion of local entropy as a measure of the degree of balance of a text.

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

Answers

What kind of regularity is there?

The solution:
Balance of the input text.
Our idea is that one obtains a more compressible string as output of BWT if its input is very close to be balanced.

Is there a more appropriate statistic?

The solution:
Local Entropy of the input text.
We introduce the notion of local entropy as a measure of the degree of balance of a text.

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

Balanced words: definition

- A (finite or infinite) word v is *balanced* if for each letter a of the alphabet $\mathcal{A}$ and for all factors u and u' of v s.t. $|u| = |u'|$ we have that

$$||u|_a - |u'|_a| \leq 1$$

- A finite word v is *circularly balanced* if all its conjugates are balanced.

Example

- $w = cacbcac$ is a circularly balanced word.
- $v = \underline{ac}ac\underline{bb}c$ is an unbalanced word.
- $u = \underline{b}ab\underline{aa}baa\underline{b}$ is a balanced but not circularly balanced word.

Denote by $\mathcal{B}$ the set of circularly balanced words.

Laurent Vuillon. Balanced words. *Bull. Belg. Math.Soc.*, 10(5):787–805, 2003.

Balanced words: definition

- A (finite or infinite) word v is *balanced* if for each letter a of the alphabet $\mathcal{A}$ and for all factors u and u' of v s.t. $|u| = |u'|$ we have that
$$||u|_a - |u'|_a| \leq 1$$
- A finite word v is *circularly balanced* if all its conjugates are balanced.

Example

- $w = cacbcac$ is a circularly balanced word.
- $v = \underline{aca}c\underline{bb}c$ is an unbalanced word.
- $u = \underline{b}ab\underline{aa}baa\underline{b}$ is a balanced but not circularly balanced word.

Denote by $\mathcal{B}$ the set of circularly balanced words.

Laurent Vuillon. Balanced words. *Bull. Belg. Math.Soc.*, 10(5):787–805, 2003.

Balanced words: definition

- A (finite or infinite) word v is *balanced* if for each letter a of the alphabet $\mathcal{A}$ and for all factors u and u' of v s.t. $|u| = |u'|$ we have that

$$||u|_a - |u'|_a| \leq 1$$

- A finite word v is *circularly balanced* if all its conjugates are balanced.

Example

- $w = cacbcac$ is a circularly balanced word.
- $v = \underline{ac}ac\underline{bb}c$ is an unbalanced word.
- $u = \underline{b}ab\underline{aa}baa\underline{b}$ is a balanced but not circularly balanced word.

Denote by $\mathcal{B}$ the set of circularly balanced words.

Laurent Vuillon. Balanced words. *Bull. Belg. Math.Soc.*, 10(5):787–805, 2003.

Extremal cases: Constant gap words and Clustered words

- A finite word v is *constant gap* if, for each letter a, the distance (the number of letters) between two consecutive occurrences of a is constant (in circular way).

Example

The word $v = abcabdabcabe$ is a constant gap word.

- Constant gap words are a special case of circularly balanced words.

- The word v is a *clustered word* if the number of runs is equal to the size of alphabet.

Example

The word $w = ddddddccccaaaaabbb$ is a clustered word.

Extremal cases: Constant gap words and Clustered words

- A finite word v is *constant gap* if, for each letter a, the distance (the number of letters) between two consecutive occurrences of a is constant (in circular way).

Example

The word $v = abcabdabcabe$ is a constant gap word.

- Constant gap words are a special case of circularly balanced words.

- The word v is a *clustered word* if the number of runs is equal to the size of alphabet.

Example

The word $w = ddddddccccaaaaabbb$ is a clustered word.

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & & & & & & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$. Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & & & & & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$.
Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & & & & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$.
Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & & & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$. Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & 1 & & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$. Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & 1 & 3 & & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$. Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & 1 & 3 & 0 & \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$.
Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & 1 & 3 & 0 & 3 \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$.
Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Statistic: Local Entropy based on Distance Coding

Distance coding: for each symbol of the input word, the DC algorithm outputs the distance to the previous occurrence of the same symbol (in circular way).

Example

$$\begin{array}{rccccccc} v = & a & c & b & c & a & a & b \\ dc(v) = & 1 & 4 & 2 & 1 & 3 & 0 & 3 \end{array}$$

Let $v = b_1 b_2 \cdots b_n$, $b_i \in \mathcal{A}$ and $dc(v) = d_1 d_2 \cdots d_n$, where $0 \leq d_i < n$. Define the Local Entropy of v:

$$LE(v) = \frac{1}{n} \sum_{i=1}^{n} \log(d_i + 1)$$

Local entropy (LE) was considered by

- J. L. Bentley, D. D. Sleator, R. E. Tarjan, and V. K. Wei, 1986
- G. Manzini, 2001
- H. Kaplan, S. Landau and E. Verbin, 2007

Bounds

Theorem

For any word v one has:

- $G(v) \leq LE(v) \leq H_0(v)$
- $LE(v) = H_0(v)$ *if and only if v is a constant gap word.*
- $LE(v) = G(v)$ *if and only if v is a clustered word.*

where

$$H_0(v) = \sum_{a \in \mathcal{A}} \frac{|v|_a}{|v|} \log \frac{|v|}{|v|_a},$$

$$G(v) = \sum_{a \in \mathcal{A}} \frac{1}{|v|} [\log(|v| - |v|_a + 1)]$$

The notion of local entropy is a measure of the degree of balance of a text.

$|v|_a$ denotes the number of occurrences of the letter a in the word v.

Measure

- For any word v:

$$\delta(v) = \frac{H_0(v) - LE(v)}{H_0(v) - G(v)}, \qquad \tau(v) = \frac{LE(v) - G(v)}{H_0(v) - G(v)}$$

- Now, by using δ and τ, we can test, in a quantitative way, our intuition, i.e. the more balanced the input word is, the more local similarity is found in the BWT of the string, the better the compression is.

- The experiments reported in the next slide confirm our intuition: actually they show that when $\delta(v)$ is less than 0.23, then $\tau(bwt(v))$ is less than 0.3 and the BWT-based compressor has good performances.

Experiments

File name	Size	H_0	Bst	Gzip	Diff %	$\delta(v)$	$\tau(bwt(v))$
bible	4,047,392	4.343	796,231	1,191,071	9.755	0.117	0.233
english	52,428,800	4.529	11,533,171	19,672,355	15.524	0.136	0.238
etext99	105,277,340	4.596	24,949,871	39,493,346	13.814	0.141	0.264
english	104,857,600	4.556	23,993,810	39,437,704	14.728	0.143	0.250
dblp.xml	52,428,800	5.230	4,871,450	9,034,902	7.941	0.152	0.093
dblp.xml	104,857,600	5.228	9,427,936	17,765,502	7.951	0.153	0.090
dblp.xml	209,715,200	5.257	18,522,167	35,897,168	8.285	0.162	0.088
dblp.xml	296,135,874	5.262	25,597,003	50,481,103	8.403	0.164	0.086
world192	2,473,400	4.998	430,225	724,606	11.902	0.174	0.183
rctail96	114,711,151	5.154	11,429,406	24,007,508	10.965	0.178	0.097
sprot34.dat	109,617,186	4.762	18,850,472	26,712,981	7.173	0.215	0.206
jdk13c	69,728,899	5.531	3,187,900	7,525,172	6.220	0.224	0.041
howto	39,886,973	4.857	8,713,851	12,638,334	9.839	0.231	0.229
rfc	116,421,901	4.623	17,565,908	26,712,981	7.857	0.239	0.163
w3c2	104,201,579	5.954	7,021,478	15,159,804	7.810	0.246	0.058
chr22.dna	34,553,758	2.137	8,015,707	8,870,068	2.473	0.341	0.575
pitches	52,428,800	5.633	18,651,999	16,884,651	-3.371	0.530	0.344
pitches	55,832,855	5.628	19,475,065	16,040,370	-6.152	0.533	0.337

Practical application: the computation of $\delta(v)$ is a fast test for the choice between bst and gzip.

Experiments

File name	Size	H_0	Bst	Gzip	Diff %	$\delta(v)$	$\tau(bwt(v))$
bible	4,047,392	4.343	796,231	1,191,071	9.755	0.117	0.233
english	52,428,800	4.529	11,533,171	19,672,355	15.524	0.136	0.238
etext99	105,277,340	4.596	24,949,871	39,493,346	13.814	0.141	0.264
english	104,857,600	4.556	23,993,810	39,437,704	14.728	0.143	0.250
dblp.xml	52,428,800	5.230	4,871,450	9,034,902	7.941	0.152	0.093
dblp.xml	104,857,600	5.228	9,427,936	17,765,502	7.951	0.153	0.090
dblp.xml	209,715,200	5.257	18,522,167	35,897,168	8.285	0.162	0.088
dblp.xml	296,135,874	5.262	25,597,003	50,481,103	8.403	0.164	0.086
world192	2,473,400	4.998	430,225	724,606	11.902	0.174	0.183
rctail96	114,711,151	5.154	11,429,406	24,007,508	10.965	0.178	0.097
sprot34.dat	109,617,186	4.762	18,850,472	26,712,981	7.173	0.215	0.206
jdk13c	69,728,899	5.531	3,187,900	7,525,172	6.220	0.224	0.041
howto	39,886,973	4.857	8,713,851	12,638,334	9.839	0.231	0.229
rfc	116,421,901	4.623	17,565,908	26,712,981	7.857	0.239	0.163
w3c2	104,201,579	5.954	7,021,478	15,159,804	7.810	0.246	0.058
chr22.dna	34,553,758	2.137	8,015,707	8,870,068	2.473	0.341	0.575
pitches	52,428,800	5.633	18,651,999	16,884,651	-3.371	0.530	0.344
pitches	55,832,855	5.628	19,475,065	16,040,370	-6.152	0.533	0.337

Practical application: the computation of $\delta(v)$ is a fast test for the choice between bst and gzip.

First conclusions

Our intuition

The more balanced the input word is, the more local similarity one has after BWT, and the better the compression is.

The notion of local entropy is a measure of the degree of balance of a text.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$

$$f_{n+1} = f_n f_{n-1} \; (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$
$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$

$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$
$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$
$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$

$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$

$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$f_0 = b$ $\quad f_4 = abaab$

$f_1 = a$ $\quad f_5 = abaababa$

$f_2 = ab$ $\quad f_6 = abaababaabaab$

$f_3 = aba$ $\quad f_7 = abaababaabaababaababa$

$$f_0 = b \qquad f_1 = a$$

$$f_{n+1} = f_n f_{n-1} \; (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Extremal case: Balanced words - Binary alphabet

- An infinite aperiodic sequence v is balanced if and only if v is a sturmian sequence.
- An infinite periodic sequence v^ω is balanced if and only if v is a conjugate of a standard word.

Fibonacci words

$$f_0 = b \qquad f_4 = abaab$$
$$f_1 = a \qquad f_5 = abaababa$$
$$f_2 = ab \qquad f_6 = abaababaabaab$$
$$f_3 = aba \qquad f_7 = abaababaabaababaababa$$

$$f_0 = b \qquad f_1 = a$$
$$f_{n+1} = f_n f_{n-1} \ (n \geq 1)$$

Standard words

Directive sequence $d_1, d_2, \ldots, d_n, \ldots$, with $d_1 \geq 0$ and $d_i > 0$ for $i = 2, \ldots, n, \ldots$.

$$s_0 = b \qquad s_1 = a \qquad s_{n+1} = s_n^{d_n} s_{n-1} \text{ for } n \geq 1$$

Standard words are special prefixes of Sturmian sequences.

Binary alphabets

Theorem (S. Mantaci, A. Restivo and M. Sciortino, 2003)

Given a word $v \in \{a, b\}$, the following conditions are equivalent:

1. $bwt(v) = b^p a^q$, *with* $p, q \geq 1$*;*
2. v *is a circularly balanced word;*
3. v *is a conjugate of a power of a Standard words.*

Example

$v = abaababa$ is a standard word and $bwt(v) = b^3 a^5$.

Circularly Balanced words on larger alphabets

- If $|\mathcal{A}| > 2$, the general structure of circularly balanced words is not known.
  E. Altman, B. Gaujal, and A. Hordijk, 2000
  R. Mantaci, S. Mantaci, and A. Restivo, 2008
- We note that the notion of circularly balanced words over an alphabet of size larger than two also appears in the statement of the Fraenkel's conjecture.
- As a direct consequence of a result of Graham, one has that balanced sequences on a set of letters having different frequencies must be periodic, i.e. of the form v^ω, where v is a circularly balanced word.

Fraenkel's conjecture

Let $\mathcal{A}_k = \{a_1, a_2, \ldots, a_k\}$. For each $k > 2$, there is only one circularly balanced word $F_k \in \mathcal{A}_k^*$, having different frequencies. It is defined recursively as follow $F_1 = a_1$ and $F_k = F_{k-1} a_k F_{k-1}$ for all $k \geq 2$.

Generalization to alphabets with more than two letters

Theorem (S. Mantaci, A. Restivo and M. Sciortino, 2003)

Given a word $v \in \{a, b\}$, the following conditions are equivalent:

1. v *is a Simple BWT word;*
2. v *is a circularly balanced word;*
3. v *is a conjugate of a power of a Standard words.*

In alphabets with more than two letters, the following sets do not coincide:

1. simple BWT words;
2. circularly balanced words;
3. finite epistandard words (a generalization of the Standard words).

Simple BWT words

In 2008, Simpson and Puglisi introduced the notion of *Simple BWT words*.

Let v be a word over a finite ordered alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$, with $a_1 < a_2 < \ldots < a_k$. The word v is a *simple BWT word* if

$$bwt(v) = a_k^{n_k} a_{k-1}^{n_{k-1}} \cdots a_2^{n_2} a_1^{n_1}$$

for some non-negative integers $n_1, n_2, \ldots, n_k$.

We denote by $\mathcal{S}$ the set of the simple BWT words.

Example

$v = acbcbcadad \in \mathcal{S}$, in fact $bwt(v) = ddcccbbaaa$.

Simpson and Puglisi get a constructive characterization of the set $\mathcal{S}$ in the case of three letters alphabet.

Simple BWT words

In 2008, Simpson and Puglisi introduced the notion of *Simple BWT words*.

Let v be a word over a finite ordered alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$, with $a_1 < a_2 < \ldots < a_k$. The word v is a *simple BWT word* if

$$bwt(v) = a_k^{n_k} a_{k-1}^{n_{k-1}} \cdots a_2^{n_2} a_1^{n_1}$$

for some non-negative integers $n_1, n_2, \ldots, n_k$.

We denote by $\mathcal{S}$ the set of the simple BWT words.

Example

$v = acbcbcadad \in \mathcal{S}$, in fact $bwt(v) = ddcccbbaaa$.

Simpson and Puglisi get a constructive characterization of the set $\mathcal{S}$ in the case of three letters alphabet.

Simple BWT words

In 2008, Simpson and Puglisi introduced the notion of *Simple BWT words*.

Let v be a word over a finite ordered alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$, with $a_1 < a_2 < \ldots < a_k$. The word v is a *simple BWT word* if

$$bwt(v) = a_k^{n_k} a_{k-1}^{n_{k-1}} \cdots a_2^{n_2} a_1^{n_1}$$

for some non-negative integers $n_1, n_2, \ldots, n_k$.

We denote by $\mathcal{S}$ the set of the simple BWT words.

Example

$v = acbcbcadad \in \mathcal{S}$, in fact $bwt(v) = ddcccbbaaa$.

Simpson and Puglisi get a constructive characterization of the set $\mathcal{S}$ in the case of three letters alphabet.

Simple BWT words

In 2008, Simpson and Puglisi introduced the notion of *Simple BWT words*.

Let v be a word over a finite ordered alphabet $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$, with $a_1 < a_2 < \ldots < a_k$. The word v is a *simple BWT word* if

$$bwt(v) = a_k^{n_k} a_{k-1}^{n_{k-1}} \cdots a_2^{n_2} a_1^{n_1}$$

for some non-negative integers $n_1, n_2, \ldots, n_k$.

We denote by $\mathcal{S}$ the set of the simple BWT words.

Example

$v = acbcbcadad \in \mathcal{S}$, in fact $bwt(v) = ddcccbbaaa$.

Simpson and Puglisi get a constructive characterization of the set $\mathcal{S}$ in the case of three letters alphabet.

Matrix M and R

	F ↓	M	L ↓
1	a	$l\ i\ n\ t\ e\ r\ n\ a\ t\ i\ o$	n
2	a	$t\ i\ o\ n\ a\ l\ i\ n\ t\ e\ r$	n
3	e	$r\ n\ a\ t\ i\ o\ n\ a\ l\ i\ n$	t
4	i	$n\ t\ e\ r\ n\ a\ t\ i\ o\ n\ a$	l
5	i	$o\ n\ a\ l\ i\ n\ t\ e\ r\ n\ a$	t
6	l	$i\ n\ t\ e\ r\ n\ a\ t\ i\ o\ n$	a
7	n	$a\ l\ i\ n\ t\ e\ r\ n\ a\ t\ i$	o
8	n	$a\ t\ i\ o\ n\ a\ l\ i\ n\ t\ e$	r
9	n	$t\ e\ r\ n\ a\ t\ i\ o\ n\ a\ l$	i
10	o	$n\ a\ l\ i\ n\ t\ e\ r\ n\ a\ t$	i
11	r	$n\ a\ t\ i\ o\ n\ a\ l\ i\ n\ t$	e
12	t	$e\ r\ n\ a\ t\ i\ o\ n\ a\ l\ i$	n
13	t	$i\ o\ n\ a\ l\ i\ n\ t\ e\ r\ n$	a

180° ↷

F_R ↓	R	L_R ↓
a	$n\ r\ e\ t\ n\ i\ l\ a\ n\ o\ i$	t
n	$i\ l\ a\ n\ o\ i\ t\ a\ n\ r\ e$	t
e	$t\ n\ i\ l\ a\ n\ o\ i\ t\ a\ n$	r
i	$t\ a\ n\ r\ e\ t\ n\ i\ l\ a\ n$	o
i	$l\ a\ n\ o\ i\ t\ a\ n\ r\ e\ t$	n
r	$e\ t\ n\ i\ l\ a\ n\ o\ i\ t\ a$	n
o	$i\ t\ a\ n\ r\ e\ t\ n\ i\ l\ a$	n
a	$n\ o\ i\ t\ a\ n\ r\ e\ t\ n\ i$	l
t	$a\ n\ r\ e\ t\ n\ i\ l\ a\ n\ o$	i
l	$a\ n\ o\ i\ t\ a\ n\ r\ e\ t\ n$	i
t	$n\ i\ l\ a\ n\ o\ i\ t\ a\ n\ r$	e
n	$r\ e\ t\ n\ i\ l\ a\ n\ o\ i\ t$	a
n	$o\ i\ t\ a\ n\ r\ e\ t\ n\ i\ l$	a

The matrix R is obtained from M by a rotation of 180°: it follows that the ith conjugate of M is the reverse of the $(n-i+1)$th conjugate of R.

Theorem

A word $v \in \mathcal{S}$ if and only if $M = R$.

Matrix M and R

	F						M						L
	↓												↓
1	a	l	i	n	t	e	r	n	a	t	i	o	n
2	a	t	i	o	n	a	l	i	n	t	e	r	n
3	e	r	n	a	t	i	o	n	a	l	i	n	t
4	i	n	t	e	r	n	a	t	i	o	n	a	l
5	i	o	n	a	l	i	n	t	e	r	n	a	t
6	l	i	n	t	e	r	n	a	t	i	o	n	a
7	n	a	l	i	n	t	e	r	n	a	t	i	o
8	n	a	t	i	o	n	a	l	i	n	t	e	r
9	n	t	e	r	n	a	t	i	o	n	a	l	i
10	o	n	a	l	i	n	t	e	r	n	a	t	i
11	r	n	a	t	i	o	n	a	l	i	n	t	e
12	t	e	r	n	a	t	i	o	n	a	l	i	n
13	t	i	o	n	a	l	i	n	t	e	r	n	a

180° ↷

F_R						R						L_R
↓												↓
a	n	r	e	t	n	i	l	a	n	o	i	t
n	i	l	a	n	o	i	t	a	n	r	e	t
e	t	n	i	l	a	n	o	i	t	a	n	r
i	t	a	n	r	e	t	n	i	l	a	n	o
i	l	a	n	o	i	t	a	n	r	e	t	n
r	e	t	n	i	l	a	n	o	i	t	a	n
o	i	t	a	n	r	e	t	n	i	l	a	n
a	n	o	i	t	a	n	r	e	t	n	i	l
t	a	n	r	e	t	n	i	l	a	n	o	i
l	a	n	o	i	t	a	n	r	e	t	n	i
t	n	i	l	a	n	o	i	t	a	n	r	e
n	r	e	t	n	i	l	a	n	o	i	t	a
n	o	i	t	a	n	r	e	t	n	i	l	a

The matrix R is obtained from M by a rotation of 180°: it follows that the ith conjugate of M is the reverse of the $(n-i+1)$th conjugate of R.

Theorem

A word $v \in \mathcal{S}$ if and only if $M = R$.

Matrix M and R

A word $v \in \mathcal{S}$ iff
$M = R$

$$
\begin{array}{ccccccccccc}
a & c & a & d & a & c & b & b & c & a & d \\
a & c & b & b & c & a & d & a & c & a & d \\
a & d & a & c & a & d & a & c & b & b & c \\
a & d & a & c & b & b & c & a & d & a & c \\
b & b & c & a & d & a & c & a & d & a & c \\
b & c & a & d & a & c & a & d & a & c & b \\
c & a & d & a & c & a & d & a & c & b & b \\
c & a & d & a & c & b & b & c & a & d & a \\
c & b & b & c & a & d & a & c & a & d & a \\
d & a & c & a & d & a & c & b & b & c & a \\
d & a & c & b & b & c & a & d & a & c & a
\end{array}
$$

$$v_i = \widetilde{v_{n-i+1}}$$

So $[v]$ and its factors are closed under reverse. Under these conditions each conjugate of v has the two palindrome property (cf. Simpson and Puglisi, 2008).

A word v has the *two palindrome property* if v is product of two palindromes, i.e. it can be written as xy where x and y are palindromes or empty.

Matrix M and R

A word $v \in \mathcal{S}$ iff

$$M = R$$

$$
\begin{array}{r l}
 & a\ c\ a\ d\ a\ c\ b\ b\ c\ a\ d \\
v_i \rightarrow & \color{red}{a\ c\ b\ b\ c\ a\ d\ a\ c\ a\ d} \\
 & a\ d\ a\ c\ a\ d\ a\ c\ b\ b\ c \\
 & a\ d\ a\ c\ b\ b\ c\ a\ d\ a\ c \\
 & b\ b\ c\ a\ d\ a\ c\ a\ d\ a\ c \\
 & b\ c\ a\ d\ a\ c\ a\ d\ a\ c\ b \\
 & c\ a\ d\ a\ c\ a\ d\ a\ c\ b\ b \\
 & c\ a\ d\ a\ c\ b\ b\ c\ a\ d\ a \\
 & c\ b\ b\ c\ a\ d\ a\ c\ a\ d\ a \\
v_{n-i+1} \rightarrow & \color{red}{d\ a\ c\ a\ d\ a\ c\ b\ b\ c\ a} \\
 & d\ a\ c\ b\ b\ c\ a\ d\ a\ c\ a
\end{array}
$$

$$v_i = \widetilde{v_{n-i+1}}$$

So $[v]$ and its factors are closed under reverse. Under these conditions each conjugate of v has the two palindrome property (cf. Simpson and Puglisi, 2008).

A word v has the *two palindrome property* if v is product of two palindromes, i.e. it can be written as xy where x and y are palindromes or empty.

Matrix M and R

A word $v \in \mathcal{S}$ iff

$$M = R$$

$$\begin{array}{r l}
 & a\ c\ a\ d\ a\ c\ b\ b\ c\ a\ d \\
v_i \rightarrow & a\ c\ b\ b\ c\ a\ d\ a\ c\ a\ d \\
 & a\ d\ a\ c\ a\ d\ a\ c\ b\ b\ c \\
 & a\ d\ a\ c\ b\ b\ c\ a\ d\ a\ c \\
 & b\ b\ c\ a\ d\ a\ c\ a\ d\ a\ c \\
 & b\ c\ a\ d\ a\ c\ a\ d\ a\ c\ b \\
 & c\ a\ d\ a\ c\ a\ d\ a\ c\ b\ b \\
 & c\ a\ d\ a\ c\ b\ b\ c\ a\ d\ a \\
 & c\ b\ b\ c\ a\ d\ a\ c\ a\ d\ a \\
v_{n-i+1} \rightarrow & d\ a\ c\ a\ d\ a\ c\ b\ b\ c\ a \\
 & d\ a\ c\ b\ b\ c\ a\ d\ a\ c\ a
\end{array}$$

$$v_i = \widetilde{v_{n-i+1}}$$

So $[v]$ and its factors are closed under reverse. Under these conditions each conjugate of v has the two palindrome property (cf. Simpson and Puglisi, 2008).

A word v has the *two palindrome property* if v is product of two palindromes, i.e. it can be written as xy where x and y are palindromes or empty.

Balanced and Simple BWT words

$$\mathcal{B} \neq \mathcal{S}$$

The set of circularly balanced words over more than two letters alphabets does not coincide with the set of Simple BWT words.

Example

- $v = cacbcac$ is circularly balanced and $bwt(v) = ccccbaa$
- $w = ababc$ is circularly balanced and $bwt(w) = cbaab$
- $u = \underline{ac}ac\underline{bb}c$ is not balanced and $bwt(u) = cccbbaa$

A generalization of Sturmian: Episturmian

- An infinite word t on $\mathcal{A}$ is *episturmian* (Droubay, J. Justin, G. Pirillo, 2001) if:
  - $F(t)$ (its set of factors) is closed under reversal;
  - t has at most one left special factor (or equivalently, right special factor) of each length.
- t is standard episturmian if all of its left special factors are prefixes of it.
- An infinite word on the finite alphabet $\mathcal{A}$ is standard episturmian if and only if it can be obtained by the Rauzy rules for $\mathcal{A}$.

Let s be an infinite word, then a factor u of s is *right* (resp. *left*) *special* if there exist $x, y \in \mathcal{A}$, $x \neq y$, such that $ux, uy \in F(s)$ (resp. $xu, yu \in F(s)$).

X. Droubay, J. Justin, G. Pirillo, Episturmian words and some constructions of de Luca and. Rauzy, Theoret. Comput. Sci. 255, 2001.
A. Glen and J. Justin. Episturmian words: a survey. *RAIRO Theoretical Informatics and Applications*, 2009.

A generalization of Sturmian: Episturmian

- An infinite word t on $\mathcal{A}$ is *episturmian* (Droubay, J. Justin, G. Pirillo, 2001) if:
  - $F(t)$ (its set of factors) is closed under reversal;
  - t has at most one left special factor (or equivalently, right special factor) of each length.
- t is standard episturmian if all of its left special factors are prefixes of it.
- An infinite word on the finite alphabet $\mathcal{A}$ is standard episturmian if and only if it can be obtained by the Rauzy rules for $\mathcal{A}$.

Let s be an infinite word, then a factor u of s is *right* (resp. *left*) *special* if there exist $x, y \in \mathcal{A}$, $x \neq y$, such that $ux, uy \in F(s)$ (resp. $xu, yu \in F(s)$).

X. Droubay, J. Justin, G. Pirillo, Episturmian words and some constructions of de Luca and. Rauzy, Theoret. Comput. Sci. 255, 2001.
A. Glen and J. Justin. Episturmian words: a survey. *RAIRO Theoretical Informatics and Applications*, 2009.

A generalization of Sturmian: Episturmian

- An infinite word t on $\mathcal{A}$ is *episturmian* (Droubay, J. Justin, G. Pirillo, 2001) if:
  - $F(t)$ (its set of factors) is closed under reversal;
  - t has at most one left special factor (or equivalently, right special factor) of each length.
- t is standard episturmian if all of its left special factors are prefixes of it.
- An infinite word on the finite alphabet $\mathcal{A}$ is standard episturmian if and only if it can be obtained by the Rauzy rules for $\mathcal{A}$.

Let s be an infinite word, then a factor u of s is *right* (resp. *left*) *special* if there exist $x, y \in \mathcal{A}$, $x \neq y$, such that $ux, uy \in F(s)$ (resp. $xu, yu \in F(s)$).

X. Droubay, J. Justin, G. Pirillo, Episturmian words and some constructions of de Luca and. Rauzy, Theoret. Comput. Sci. 255, 2001.
A. Glen and J. Justin. Episturmian words: a survey. *RAIRO Theoretical Informatics and Applications*, 2009.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

A generalization of Standard: Finite epistandard

Rauzy rules.

	$Rules$	1	2	3	4
R_0		a	b	c	d
R_1	1	a	ab	ac	ad
R_2	1	a	aab	aac	aad
R_3	4	$aada$	$aadaab$	$aadaac$	aad
R_4	3	$aadaacaada$	$aadaacaadaab$	$aadaac$	$aadaacaad$

- Let $|\mathcal{A}| = k$. A word $v \in \mathcal{A}^*$ is called finite epistandard if v is an element of a k-tuples R_n, for some $n = 1$.
- We denote by $\mathcal{EP}$ the set of words that are powers of a conjugate of a finite epistandard word.

Balancing and Epistandard

$$\mathcal{B} \neq \mathcal{EP}$$

The set of circularly balanced words over more than two letters alphabets does not coincide with the set of conjugate of powers of epistandard words.

Example

- $v = aadaacaad$ is epistandard, but it is not circularly balanced.
- $u = abcabdabcabe$ is circularly balanced, but it is not epistandard.

Palindromic Richness

- The number of distinct palindromic factors (including ε) of a word v is at most $|v| + 1$.
- A finite word v is (palindromic) rich if it has exactly $|v| + 1$ distinct palindromic factors, including the empty word ε.
- A factor of finite rich word is rich.

Example

$v = ccaacb$ is rich, $|v| = 6$, in fact: $P(v) = \{\varepsilon, c, cc, caac, a, aa, b\}$, $|P(v)| = 7$.

X. Droubay, J. Justin, G. Pirillo, Episturmian words and some constructions of de Luca and. Rauzy, Theoret. Comput. Sci. 255, 2001.
A. Glen, J. Justin, S. Widmer, and L. Q. Zamboni. Palindromic richness. *European Journal of Combinatorics*, 30(2):510–531, 2009.

Palindromic Richness

- The number of distinct palindromic factors (including ε) of a word v is at most $|v|+1$.
- A finite word v is (palindromic) rich if it has exactly $|v|+1$ distinct palindromic factors, including the empty word ε.
- A factor of finite rich word is rich.

Example

$v = ccaacb$ is rich, $|v| = 6$, in fact: $P(v) = \{\varepsilon, c, cc, caac, a, aa, b\}$, $|P(v)| = 7$.

X. Droubay, J. Justin, G. Pirillo, Episturmian words and some constructions of de Luca and. Rauzy, Theoret. Comput. Sci. 255, 2001.
A. Glen, J. Justin, S. Widmer, and L. Q. Zamboni. Palindromic richness. *European Journal of Combinatorics*, 30(2):510–531, 2009.

Circularly rich words

Lemma (Glen, Justin, Widmer and Zamboni, 2009)

For a finite word v, the following properties are equivalent:

1. v^ω *is rich;*
2. v^2 *is rich;*
3. v *is a product of two palindromes and all of the conjugates of* v *(including itself) are rich.*

- We say that a finite word v is circularly rich if the infinite word v^ω is rich.
- We denote by $\mathcal{R}$ the set of the circularly rich words.

Example

$v = bbaca$, $|v| = 5$ is circularly rich, in fact:
$P(v^2) = \{\varepsilon, a, b, c, bb, aca, bacab, bbacabb, acabbaca, cabbac, abba\}$,
$|P(v^2)| = 11.$

Circularly rich words

Lemma (Glen, Justin, Widmer and Zamboni, 2009)

For a finite word v, the following properties are equivalent:

1. v^ω *is rich;*
2. v^2 *is rich;*
3. v *is a product of two palindromes and all of the conjugates of* v *(including itself) are rich.*

- We say that a finite word v is circularly rich if the infinite word v^ω is rich.
- We denote by $\mathcal{R}$ the set of the circularly rich words.

Example

$v = bbaca$, $|v| = 5$ is circularly rich, in fact:
$P(v^2) = \{\varepsilon, a, b, c, bb, aca, bacab, bbacabb, acabbaca, cabbac, abba\}$,
$|P(v^2)| = 11$.

Circularly rich words

Lemma (Glen, Justin, Widmer and Zamboni, 2009)

For a finite word v, the following properties are equivalent:

1. v^ω *is rich;*
2. v^2 *is rich;*
3. v *is a product of two palindromes and all of the conjugates of* v *(including itself) are rich.*

- We say that a finite word v is circularly rich if the infinite word v^ω is rich.
- We denote by $\mathcal{R}$ the set of the circularly rich words.

Example

$v = bbaca$, $|v| = 5$ is circularly rich, in fact:
$P(v^2) = \{\varepsilon, a, b, c, bb, aca, bacab, bbacabb, acabbaca, cabbac, abba\}$,
$|P(v^2)| = 11$.

Circularly rich words

Lemma (Glen, Justin, Widmer and Zamboni, 2009)

For a finite word v, the following properties are equivalent:

1. v^ω *is rich;*
2. v^2 *is rich;*
3. v *is a product of two palindromes and all of the conjugates of* v *(including itself) are rich.*

- We say that a finite word v is circularly rich if the infinite word v^ω is rich.
- We denote by $\mathcal{R}$ the set of the circularly rich words.

Example

$v = bbaca$, $|v| = 5$ is circularly rich, in fact:
$P(v^2) = \{\varepsilon, a, b, c, bb, aca, bacab, bbacabb, acabbaca, cabbac, abba\}$,
$|P(v^2)| = 11$.

Balancing and Richness

$$\mathcal{R} \neq \mathcal{B}$$

The set of circularly balanced words over more than two letters alphabets does not coincide with the set of circularly rich words.

Example

- The word $w = bbbbbacaca$ is circularly rich, but it is not circularly balanced.
- The word $u = abcabdabcabe$ is circularly balanced, but it is not circularly rich.

$\mathcal{S} \cap \mathcal{B} = \mathcal{R} \cap \mathcal{B} = \mathcal{EP} \cap \mathcal{B}$

Theorem

Let $w \in \mathcal{A}^*$ be a circularly balanced word over $\mathcal{A}$. The following statements are equivalent:

1. *w is a simple BWT word;*
2. *w is a circularly rich word;*
3. *w is a conjugate of a power of a finite epistandard word.*

Proof: $3 \to 1$: The finite balanced epistandard words belong to $\mathcal{S}$.

From a result of Paquin and Vuillon (2006), one can prove that each finite balanced epistandard word t is of the form:

1. $t = pa_2$, with $p = Pal(a_1^m a_k a_{k-1} \cdots a_3)$, where $k \geq 3$ and $m \geq 1$;
2. $t = pa_2$, with $p = Pal(a_1 a_k a_{k-1} \cdots a_{k-\ell} a_1 a_{k-\ell-1} a_{k-\ell-2} \cdots a_3)$, where $0 \leq \ell \leq k-4$ and $k \geq 4$;
3. $t = Pal(a_1 a_k a_{k-1} \cdots a_2)$, where $k \geq 3$ (*Fraenkel's words*).

where the operator Pal is the iterated palindromic closure function.

The *palindromic right-closure* $v^{(+)}$ of a finite word v is the (unique) shortest palindrome having v as a prefix (A. de Luca, 1997).

The *iterated palindromic closure* function (J. Justin, 2005), denoted by Pal, is recursively defined as follows. Set $Pal(\varepsilon) = \varepsilon$ and, for any word v and letter x, define $Pal(vx) = (Pal(v)x)^{(+)}$.

In order to prove that t belongs to $\mathcal{S}$ it suffices to show that words of the form $(1), (2)$ and (3) have simple BWT.

Proof: $3 \to 1$: The finite balanced epistandard words belong to $\mathcal{S}$.

From a result of Paquin and Vuillon (2006), one can prove that each finite balanced epistandard word t is of the form:

1. $t = pa_2$, with $p = Pal(a_1^m a_k a_{k-1} \cdots a_3)$, where $k \geq 3$ and $m \geq 1$;
2. $t = pa_2$, with $p = Pal(a_1 a_k a_{k-1} \cdots a_{k-\ell} a_1 a_{k-\ell-1} a_{k-\ell-2} \cdots a_3)$, where $0 \leq \ell \leq k-4$ and $k \geq 4$;
3. $t = Pal(a_1 a_k a_{k-1} \cdots a_2)$, where $k \geq 3$ (*Fraenkel's words*).

where the operator Pal is the iterated palindromic closure function.

The *palindromic right-closure* $v^{(+)}$ of a finite word v is the (unique) shortest palindrome having v as a prefix (A. de Luca, 1997).

The *iterated palindromic closure* function (J. Justin, 2005), denoted by Pal, is recursively defined as follows. Set $Pal(\varepsilon) = \varepsilon$ and, for any word v and letter x, define $Pal(vx) = (Pal(v)x)^{(+)}$.

In order to prove that t belongs to $\mathcal{S}$ it suffices to show that words of the form $(1), (2)$ and (3) have simple BWT.

Proof: $2 \leftrightarrow 3$: w is circularly rich if and only if w is a conjugate of a power of a finite epistandard.

The proof is a consequence of the following results:

- The set of the episturmian sequences is a subset of the set of the rich words (Glen, Justin, Widmer and Zamboni, 2009).
- Recurrent balanced rich infinite words are precisely the balanced episturmian words (Glen, Justin, Widmer and Zamboni, 2009). Hence a balanced circularly rich word coincides with a conjugate of a power of a balanced epistandard word.

Proof: $2 \leftrightarrow 3$: w is circularly rich if and only if w is a conjugate of a power of a finite epistandard.

The proof is a consequence of the following results:

- The set of the episturmian sequences is a subset of the set of the rich words (Glen, Justin, Widmer and Zamboni, 2009).
- Recurrent balanced rich infinite words are precisely the balanced episturmian words (Glen, Justin, Widmer and Zamboni, 2009). Hence a balanced circularly rich word coincides with a conjugate of a power of a balanced epistandard word.

Proof: $2 \leftrightarrow 3$: w is circularly rich if and only if w is a conjugate of a power of a finite epistandard.

The proof is a consequence of the following results:

- The set of the episturmian sequences is a subset of the set of the rich words (Glen, Justin, Widmer and Zamboni, 2009).
- Recurrent balanced rich infinite words are precisely the balanced episturmian words (Glen, Justin, Widmer and Zamboni, 2009). Hence a balanced circularly rich word coincides with a conjugate of a power of a balanced epistandard word.

Proof: $1 \to 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \to 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \to 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \to 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \to 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \rightarrow 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Proof: $1 \rightarrow 2$: If the word w belongs to $\mathcal{S}$ then w is circularly rich.

Theorem

If the word w belongs to $\mathcal{S}$ then w is circularly rich.

We know that

- w is circularly rich if and only if w is a product of two palindromes and all the conjugates of w (including itself) are rich.
- each word $w \in \mathcal{S}$ has the two palindrome property.

We prove that

If $w \in \mathcal{S}$ then all the conjugates of w (including itself) are rich.

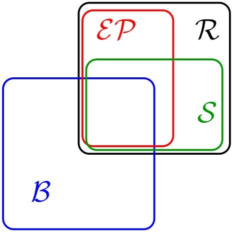
Example

The word $w = acbcbcadad \in \mathcal{S}$, in fact $bwt(acbcbcadad) = ddcccbbaaa$, and $|w|^2 = 20$, $|P(w^2)| = 21$, so w is circularly rich.

Synthesis

Under the condition of circularly balanced, the following statements are equivalent:

- $w \in \mathcal{S}$ (simple BWT word);
- w is circularly rich,
- w is a conjugate of a power of a finite epistandard.



The following example shows that there exist words **unbalanced** which belong to $\mathcal{EP} \cap \mathcal{S}$: $w = \underline{aad}\underline{aaca}a\underline{d}$ is not a circularly balanced word, $w \in \mathcal{EP}$ and $w \in \mathcal{S}$.

Conclusions

- "The regularity of the English text that BWT-based compressors exploit" is related to the balancing properties of the text itself.
- Empirical observations and theoretical results support the hypothesis: the more balanced the input word is, the more local similarity one has after BWT, and, as a consequence, the better the compression is.
- Apart from their interest for the study of the clustering effect of BWT (and of optimal performances of BWT-based compressors), our results can be considered as a contribution to combinatorics of episturmian sequences, and could provide new insight on Fraenkel's conjecture.
- The main purpose of this investigation is to state a link between methods from Combinatorics on Words and techniques from Data Compression, in order to obtain a deeper comprehension of both research field.

Conclusions

- "The regularity of the English text that BWT-based compressors exploit" is related to the balancing properties of the text itself.
- Empirical observations and theoretical results support the hypothesis: the more balanced the input word is, the more local similarity one has after BWT, and, as a consequence, the better the compression is.
- Apart from their interest for the study of the clustering effect of BWT (and of optimal performances of BWT-based compressors), our results can be considered as a contribution to combinatorics of episturmian sequences, and could provide new insight on Fraenkel's conjecture.
- The main purpose of this investigation is to state a link between methods from Combinatorics on Words and techniques from Data Compression, in order to obtain a deeper comprehension of both research field.

Conclusions

- "The regularity of the English text that BWT-based compressors exploit" is related to the balancing properties of the text itself.
- Empirical observations and theoretical results support the hypothesis: the more balanced the input word is, the more local similarity one has after BWT, and, as a consequence, the better the compression is.
- Apart from their interest for the study of the clustering effect of BWT (and of optimal performances of BWT-based compressors), our results can be considered as a contribution to combinatorics of episturmian sequences, and could provide new insight on Fraenkel's conjecture.
- The main purpose of this investigation is to state a link between methods from Combinatorics on Words and techniques from Data Compression, in order to obtain a deeper comprehension of both research field.

Conclusions

- "The regularity of the English text that BWT-based compressors exploit" is related to the balancing properties of the text itself.
- Empirical observations and theoretical results support the hypothesis: the more balanced the input word is, the more local similarity one has after BWT, and, as a consequence, the better the compression is.
- Apart from their interest for the study of the clustering effect of BWT (and of optimal performances of BWT-based compressors), our results can be considered as a contribution to combinatorics of episturmian sequences, and could provide new insight on Fraenkel's conjecture.
- The main purpose of this investigation is to state a link between methods from Combinatorics on Words and techniques from Data Compression, in order to obtain a deeper comprehension of both research field.

Further works

- To study, in a quantitative way, the compression ratio of BWT-based compressors in terms of the Local Entropy.
- To characterize the words in $\mathcal{S}$ (we have characterized the balanced words in $\mathcal{S}$).
- To characterize all words having a clusterized BWT transform (the set $\mathcal{S}$ is a proper subclass of words having a clusterized BWT transform): the order of letters in the output of BWT is very important.
  For instance, the BWT of the word $w = abacad$ is a clustered word, indeed we have that $bwt(w) = dbca^3$, but although w is a circularly balanced word, it is not a circularly rich word.

Further works

- To study, in a quantitative way, the compression ratio of BWT-based compressors in terms of the Local Entropy.
- To characterize the words in $\mathcal{S}$ (we have characterized the balanced words in $\mathcal{S}$).
- To characterize all words having a clusterized BWT transform (the set $\mathcal{S}$ is a proper subclass of words having a clusterized BWT transform): the order of letters in the output of BWT is very important.
  For instance, the BWT of the word $w = abacad$ is a clustered word, indeed we have that $bwt(w) = dbca^3$, but although w is a circularly balanced word, it is not a circularly rich word.

Further works

- To study, in a quantitative way, the compression ratio of BWT-based compressors in terms of the Local Entropy.
- To characterize the words in $\mathcal{S}$ (we have characterized the balanced words in $\mathcal{S}$).
- To characterize all words having a clusterized BWT transform (the set $\mathcal{S}$ is a proper subclass of words having a clusterized BWT transform): the order of letters in the output of BWT is very important.
  For instance, the BWT of the word $w = abacad$ is a clustered word, indeed we have that $bwt(w) = dbca^3$, but although w is a circularly balanced word, it is not a circularly rich word.

Thank you for your attention!