Words and Automata - Homework

Exercise 9. Burrows-Wheeler Transform

Mikhail Dubov

January 11, 2016

Burrows-Wheeler Transform (BWT)

- Transformation on words: $w \mapsto BWT(w)$
- Constructed as follows:
  1. List all the **cyclic shifts** of w:
     - $w_1 w_2 \dots w_n$
     - $w_2 w_3 \dots w_1$
     - ...
     - $w_n w_1 \dots w_{n-1}$
  2. Sort them in the alphabetical order as the rows of a $n \times n$ table
  3. The last column contains $BWT(w)$

BWT: Example

Let $w = abracadabra$.

Step 1: Compute cyclic shifts.

	1	2	3	4	5	6	7	8	9	10	11
1	a	b	r	a	c	a	d	a	b	r	a
2	b	r	a	c	a	d	a	b	r	a	a
3	r	a	c	a	d	a	b	r	a	a	b
4	a	c	a	d	a	b	r	a	a	b	r
5	c	a	d	a	b	r	a	a	b	r	a
6	a	d	a	b	r	a	a	b	r	a	c
7	d	a	b	r	a	a	b	r	a	c	a
8	a	b	r	a	a	b	r	a	c	a	d
9	b	r	a	a	b	r	a	c	a	d	a
10	r	a	a	b	r	a	c	a	d	a	b
11	a	a	b	r	a	c	a	d	a	b	r

BWT: Example

Let $w = abracadabra$.

Step 2: Sort.

	1	2	3	4	5	6	7	8	9	10	11
11	a	a	b	r	a	c	a	d	a	b	r
8	a	b	r	a	a	b	r	a	c	a	d
1	a	b	r	a	c	a	d	a	b	r	a
4	a	c	a	d	a	b	r	a	a	b	r
6	a	d	a	b	r	a	a	b	r	a	c
9	b	r	a	a	b	r	a	c	a	d	a
2	b	r	a	c	a	d	a	b	r	a	a
5	c	a	d	a	b	r	a	a	b	r	a
7	d	a	b	r	a	a	b	r	a	c	a
10	r	a	a	b	r	a	c	a	d	a	b
3	r	a	c	a	d	a	b	r	a	a	b

BWT: Example

Let $w = abracadabra$.

Step 3: Take the last column: $BWT(abracadabra) = rdarcaaaabb$.

	1	2	3	4	5	6	7	8	9	10	11
1	a	a	b	r	a	c	a	d	a	b	r
2	a	b	r	a	a	b	r	a	c	a	d
3	a	b	r	a	c	a	d	a	b	r	a
4	a	c	a	d	a	b	r	a	a	b	r
5	a	d	a	b	r	a	a	b	r	a	c
6	b	r	a	a	b	r	a	c	a	d	a
7	b	r	a	c	a	d	a	b	r	a	a
8	c	a	d	a	b	r	a	a	b	r	a
9	d	a	b	r	a	a	b	r	a	c	a
10	r	a	a	b	r	a	c	a	d	a	b
11	r	a	c	a	d	a	b	r	a	a	b

Exercise

Show that $w \mapsto BWT(w)$ is a **bijection**.

Exercise

Show that $w \mapsto BWT(w)$ is a **bijection**.

To show that a function $f : X \to Y : x \mapsto f(x)$ is a bijection, we must show that it is:

Exercise

Show that $w \mapsto BWT(w)$ is a **bijection**.

To show that a function $f : X \to Y : x \mapsto f(x)$ is a bijection, we must show that it is:

1. **well-defined**:
   - $f \subseteq X \times Y$
   - the domain of f is X
   - $\langle x, y_1 \rangle, \langle x, y_2 \rangle \in f \implies y_1 = y_2$

Exercise

Show that $w \mapsto BWT(w)$ is a **bijection**.

To show that a function $f : X \to Y : x \mapsto f(x)$ is a bijection, we must show that it is:

1. **well-defined**:
   - $f \subseteq X \times Y$
   - the domain of f is X
   - $\langle x, y_1 \rangle, \langle x, y_2 \rangle \in f \implies y_1 = y_2$
2. **injective**:
   $f(x_1) = f(x_2) \implies x_1 = x_2$

Exercise

Show that $w \mapsto BWT(w)$ is a **bijection**.

To show that a function $f : X \to Y : x \mapsto f(x)$ is a bijection, we must show that it is:

1. **well-defined**:
   - $f \subseteq X \times Y$
   - the domain of f is X
   - $\langle x, y_1 \rangle, \langle x, y_2 \rangle \in f \implies y_1 = y_2$
2. **injective**:
   $f(x_1) = f(x_2) \implies x_1 = x_2$
3. **surjective**:
   for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Is BWT a bijection?

$BWT : X \to Y : x \mapsto BWT(x)$

Well-defined?

Is BWT a bijection?

$BWT : X \to Y : x \mapsto BWT(x)$

Well-defined? **YES.**

Is BWT a bijection?

$BWT : X \to Y : x \mapsto BWT(x)$

Well-defined? **YES.**

- X, Y are both sets of all words
- $BWT(x)$ is, by construction, defined for any word x and is a permutation of the letters of x
- So $BWT(x)$ maps a word x to another word of the same length containing the same characters
- The construction of $BWT(x)$ is deterministic, so $BWT(x) = y_1, BWT(x) = y_2 \implies y_1 = y_2$

Is BWT a bijection?

$$BWT(x_1) = BWT(x_2) \implies x_1 = x_2$$

Injective?

Is BWT a bijection?

$$BWT(x_1) = BWT(x_2) \implies x_1 = x_2$$

Injective? **Strictly speaking, NO.**

Is BWT a bijection?

$$BWT(x_1) = BWT(x_2) \implies x_1 = x_2$$

Injective? **Strictly speaking, NO.**

- $BWT(x)$ is injective up to a **conjugacy class**
- But, say, $BWT(abracadabra) = BWT(cadabraabra) = rdarcaaaabb$
- Ways to make $BWT(x)$ truly injective:
  - Use a special **termination symbol** ($abracadabra\$$) to be able to reconstruct x from $BWT(x)$ in a unique way
  - Alternatively, output the **index** I of the row at which x appears in the table for $BWT(x)$ (e.g. for $BWT(abracadabra)$, $I = 3$)

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective?

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective?

Can we actually reconstruct $w = BWT^{-1}(BWT(w))$?

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective?

Can we actually reconstruct $w = BWT^{-1}(BWT(w))$? **YES!**

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective?

Can we actually reconstruct $w = BWT^{-1}(BWT(w))$? **YES!**

Make use of the following observations:

- We have only the last column in the table, the $BWT(w)$
- The first column in the table can be reconstructed by sorting $BWT(w)$
- In each row (except for the row I where we have w), $Last(i)$ precedes $First(i)$ in the original word
- Due to the way the table was constructed, words starting from the same letter in the *last* column appear in lexicographical order relative to one another

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

ca

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

ca

aca

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

ca

aca

$raca$

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

ca

aca

$raca$

$braca$

BWT is reversible

$BWT(abraca) = caraab$

	1	2	3	4	5	6
1	a	a	b	r	a	c
2	a	b	r	a	c	a
3	a	c	a	a	b	r
4	b	r	a	c	a	a
5	c	a	a	b	r	a
6	r	a	c	a	a	b

a

ca

aca

$raca$

$braca$

$abraca$

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective?

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective? **NO!**

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective? **NO!**

- $BWT^{-1}(t)$ is not defined on all the words
- For example, try to compute $BWT^{-1}(bccaaa)$:

	1	2	3	4	5	6
1	a	?	?	?	?	b
2	a	?	?	?	?	c
3	a	?	?	?	?	c
4	b	?	?	?	?	a
5	c	?	?	?	?	a
6	c	?	?	?	?	a

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective? **NO!**

- $BWT^{-1}(t)$ is not defined on all the words
- For example, try to compute $BWT^{-1}(bccaaa)$:

	1	2	3	4	5	6
1	a	?	?	?	?	b
2	a	?	?	?	?	c
3	a	?	?	?	?	c
4	b	?	?	?	?	a
5	c	?	?	?	?	a
6	c	?	?	?	?	a

c

Is BWT a bijection?

for all $y \in Y$ there is some $x \in X$ such that $f(x) = y$

Surjective? **NO!**

- $BWT^{-1}(t)$ is not defined on all the words
- For example, try to compute $BWT^{-1}(bccaaa)$:

	1	2	3	4	5	6
1	a	?	?	?	?	b
2	a	?	?	?	?	c
3	a	?	?	?	?	c
4	b	?	?	?	?	a
5	c	?	?	?	?	a
6	c	?	?	?	?	a

c

ac

Bijective version of BWT

! Burrows-Wheeler Transform is **reversible**, but **not invertible**.

Bijective version of BWT

- ! Burrows-Wheeler Transform is **reversible**, but **not invertible**.
- ! Burrows-Wheeler Transform is **injective**, but **not bijective**.

Bijective version of BWT

! Burrows-Wheeler Transform is **reversible**, but **not invertible**.
! Burrows-Wheeler Transform is **injective**, but **not bijective**.

However, there is a bijective version of the BWT (**Burrows-Wheeler-Scott Transform**) [3]:

1. Obtain the *Lyndon factorization* of w: $w = l_1^{n_1} \dots l_r^{n_r}$, where $r \geq 0$, $n_1, \dots, n_r \geq 1$, and $l_1 > \cdots > l_r$ are Lyndon words
2. Sort the rotations of all *Lyndon words* of the input (they are no longer of the same length! Strings of different lengths are compared *as if both are repeated infinitely*)
3. $BWST(w)$ is the last character of each rotation in the sorted output

Burrows-Wheeler-Scott Transform: Example

$w = bcbccbcbcabb$

Lyndon factorization: $w = (bcbcc)(bc)^2(abb)$

	Rotations of $l_1 \dots l_r$
1	b c b c c
2	c b c b c
3	c c b c b
4	b c c b c
5	c b c c b
6	b c
7	c b
8	b c
9	c b
10	a b b
11	b a b
12	b b a

Burrows-Wheeler-Scott Transform: Example

$w = bcbccbcbcabb$
$BWST(w) = bbaccccbbcbb$

	Sorted rotations
1	a b b a b b . . .
2	b a b b a b . . .
3	b b a b b a . . .
4	b c b c b c . . .
5	b c b c b c . . .
6	b c b c c b . . .
7	b c c b c b . . .
8	c b c b c b . . .
9	c b c b c b . . .
10	c b c b c c . . .
11	c b c c b c . . .
12	c c b c b c . . .

Context

Burrows-Wheeler Transform in practice:

- BWT is a reversible transformation that tends to group characters together: $BWT(abracadabra) = rdarcaaaabb$
- **Text compression** algorithms based on the BWT and move-to-front coding [1] achieve:
  - compression within a percent or so of that achieved by the best statistical modelling techniques
  - speeds comparable to those of algorithms based on the techniques of Lempel and Ziv
- With some additional information about the correspondance between the BWT and the Suffix Array for the text, enables **pattern matching** in $O(|P| + occ \cdot \log |T|)$

$\Longrightarrow$ Many practical **full-text indexes** are based on the **BWT** [2]

References

[1] M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Technical report, 1994.

[2] P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, pages 390–, Washington, DC, USA, 2000. IEEE Computer Society.

[3] Manfred Kufleitner. On bijective variants of the burrows-wheeler transform. In Jan Holub and Jan Ždárek, editors, *Proceedings of the Prague Stringology Conference 2009*, pages 65–79, Czech Technical University in Prague, Czech Republic, 2009.