# On Burrows Wheeler Transform and Bioinformatics

Ananth Kalyanaraman

October 30, 2013

# Burrows Wheeler Transform: Introduction

- Burrows Wheeler Transform (BWT) is a transformation originally invented for data compression [BW94].
- It was later adopted in the bioinformatics domain.
- One of the most popular application of BWTs in bioinformatics is in the problem of read mapping [LD09, LD10, LTPS09, TS09]. This has a direct application in genome re-sequencing and targeted re-sequencing projects.
- In this lecture, we will define the Burrows Wheeler Transform and review its application in pattern matching.

# Definition of Burrows Wheeler of Transform

**Notation and definitions:**

| | |
|---|---|
| $s$ | input string of length $n$ |
| $s[i]$ | character at index $i$ of $s$ (indexing starts at 1) |
| $s[i \ldots j]$ | substring of $s$ starting at index $i$ and ending at index $j$ e.g., $s[1 \ldots n] = s$ |
| $\Sigma$ | string alphabet |
| $\$$ | end of string symbol (i.e., $s[n] = \$$) s.t. $\$ \notin \Sigma$ |
| $\prec$ | operator to denote lexicographically smaller |
| $\cdot$ | string concatenation operator |
| rot$(i)$ | the cyclic permutation of $s$ which starts at index $i$ (i.e., $s[i \ldots n] \cdot s[1 \ldots i-1]$) |
| SA[1...n] | suffix array of $s$ (i.e., lexicographically sorted array of suffixes of $s$) |
| $R[1 \ldots n]$ | build an array of strings $R$ s.t. $R[i] = rot(SA[i])$ |

## Definition

$BWT(s)$: **Given an input string** $s$ **of length** $n$**,** $BWT(s)$ **is an array of size** $n$ **where** $BWT[i] = R[i][n]$**.**

# BWT: Example

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $s =$ | b | a | n | a | n | a | \$ |

All (left) rotations:

| Suff id. | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | b | a | n | a | n | a | \$ |
| 2 | a | n | a | n | a | \$ | b |
| 3 | n | a | n | a | \$ | b | a |
| 4 | a | n | a | \$ | b | a | n |
| 5 | n | a | \$ | b | a | n | a |
| 6 | a | \$ | b | a | n | a | n |
| 7 | \$ | b | a | n | a | n | a |

$R[1 \dots n]$: Suffix array of $s$ with rotations

| SA | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | \$ | b | a | n | a | n | a |
| 6 | a | \$ | b | a | n | a | n |
| 4 | a | n | a | \$ | b | a | n |
| 2 | a | n | a | n | a | \$ | b |
| 1 | b | a | n | a | n | a | \$ |
| 5 | n | a | \$ | b | a | n | a |
| 3 | n | a | n | a | \$ | b | a |

---

$BWT(s)$ = annb\$aa (which is same as the last column in the $R$ table).

# BWT properties

Given an input string $s$ and its BWT transform $BWT(s)$, let:

| | |
|---|---|
| $\ell(x_i)$ | denote the $i^{th}$ occurrence of $x$ in the <u>l</u>ast column of $R$<br>(Note: this is same as the $i^{th}$ occ. of $x$ in $BWT(s)$)<br>E.g., $\ell(a_1) = \underline{\text{a}}\text{nnb\$aa}$; $\ell(a_2) = \text{annb\$}\underline{\text{a}}\text{a}$ |
| $ind(\ell(x_i))$ | denote the index in $s$ corresponding to $\ell(x_i)$<br>E.g., $ind(\ell(a_1)) = 6$; $ind(\ell(a_2)) = 4$ |
| $f(x_i)$ | denote the $i^{th}$ occurrence of $x$ in the <u>f</u>irst column of $R$<br>E.g., $f(a_1) = \text{\$}\underline{\text{a}}\text{aabnn}$; $f(a_2) = \text{\$a}\underline{\text{a}}\text{abnn}$ |
| $ind(f(x_i))$ | denote the index in $s$ corresponding to $f(x_i)$ |

# Last to first property of BWTs

## Lemma

**Last column to first column property:** *The $i^{th}$ occurrence of character $x$ in the last column of $R$ is same as the $i^{th}$occurrence of $x$ in the first column of $R$ — i.e., $ind(\ell(x_i)) = ind(f(x_i))$.*

## Proof.

For any $i < n$, since $\ell(x_i)$ occurs before $\ell(x_{i+1})$ in the last column of $R$ (same as the $BWT(s)$),

$$\Rightarrow \quad s[ind(\ell(x_i)) + 1 \ldots n] \prec s[ind(\ell(x_{i+1}) + 1 \ldots n]$$

# Last to first property of BWTs

## Lemma

**Last column to first column property:** *The $i^{th}$ occurrence of character $x$ in the last column of $R$ is same as the $i^{th}$occurrence of $x$ in the first column of $R$ — i.e., $ind(\ell(x_i)) = ind(f(x_i))$.*

## Proof.

For any $i < n$, since $\ell(x_i)$ occurs before $\ell(x_{i+1})$ in the last column of $R$ (same as the $BWT(s)$),

$$\Rightarrow \quad s[ind(\ell(x_i)) + 1 \dots n] \prec s[ind(\ell(x_{i+1}) + 1 \dots n]$$

$$\Rightarrow \quad x \cdot s[ind(\ell(x_i)) + 1 \dots n] \prec x \cdot s[ind(\ell(x_{i+1})) + 1 \dots n]$$

# Last to first property of BWTs

### Lemma

**Last column to first column property:** *The $i^{th}$ occurrence of character $x$ in the last column of $R$ is same as the $i^{th}$occurrence of $x$ in the first column of $R$ — i.e., $ind(\ell(x_i)) = ind(f(x_i))$.*

### Proof.

For any $i < n$, since $\ell(x_i)$ occurs before $\ell(x_{i+1})$ in the last column of $R$ (same as the $BWT(s)$),

$$\begin{aligned}
&\Rightarrow \quad s[ind(\ell(x_i)) + 1 \ldots n] \prec s[ind(\ell(x_{i+1}) + 1 \ldots n] \\
&\Rightarrow \quad x \cdot s[ind(\ell(x_i)) + 1 \ldots n] \prec x \cdot s[ind(\ell(x_{i+1})) + 1 \ldots n] \\
&\Rightarrow \quad s[ind(\ell(x_i)) \ldots n] \prec s[ind(\ell(x_{i+1})) \ldots n] \\
&\qquad\qquad (\because s[ind(\ell(x_i))] = s[ind(\ell(x_{i+1}))] = x)
\end{aligned}$$

# Last to first property of BWTs

## Lemma

**Last column to first column property:** *The $i^{th}$ occurrence of character $x$ in the last column of $R$ is same as the $i^{th}$occurrence of $x$ in the first column of $R$ — i.e., $ind(\ell(x_i)) = ind(f(x_i))$.*

## Proof.

For any $i < n$, since $\ell(x_i)$ occurs before $\ell(x_{i+1})$ in the last column of $R$ (same as the $BWT(s)$),

$$
\begin{aligned}
\Rightarrow\quad & s[ind(\ell(x_i)) + 1 \ldots n] \prec s[ind(\ell(x_{i+1}) + 1 \ldots n] \\
\Rightarrow\quad & x \cdot s[ind(\ell(x_i)) + 1 \ldots n] \prec x \cdot s[ind(\ell(x_{i+1})) + 1 \ldots n] \\
\Rightarrow\quad & s[ind(\ell(x_i)) \ldots n] \prec s[ind(\ell(x_{i+1})) \ldots n] \\
& \qquad (\because s[ind(\ell(x_i))] = s[ind(\ell(x_{i+1}))] = x) \\
\Rightarrow\quad & ind(f(x_i)) = ind(\ell(x_i)) \ (\because \text{ the above inequality holds } \forall i < n \\
& \qquad \text{and the first column represents the suffix array})
\end{aligned}
$$

# Implications of the Last to first property of BWTs

- Can be useful in both reconstruction and pattern matching procedures.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $s =$ | b | a | n | a | n | a | \$ |

$R[1 \ldots n]$: Suffix array of $s$ with rotations

| SA | $f$ | | | | | | $\ell$ |
|---|---|---|---|---|---|---|---|
| 7 | \$ | b | a | n | a | n | a |
| 6 | a | \$ | b | a | n | a | n |
| 4 | a | n | a | \$ | b | a | n |
| 2 | a | n | a | n | a | \$ | b |
| 1 | b | a | n | a | n | a | \$ |
| 5 | n | a | \$ | b | a | n | a |
| 3 | n | a | n | a | \$ | b | a |

# BWT functions

| | |
|---|---|
| $BWT(s)$ | compute the $BWT(s)$ for a given string $s$ |
| $BWT\text{-}Inverse(BWT(s))$ | compute the string $s$ given its BWT transform $BWT(s)$ |
| $PatternMatch(BWT(s), p)$ | search for a given pattern $p$ (of length $m$) in string $s$ using its BWT transform |

# Algorithm: *BWT-Inverse*($BWT(s)$)

### Definition

Let $next(i)$ denote the row index in $R$ corresponding to the occurrence of $BWT[i]$ in the first column.

---

**BWT-Inverse(BWT(s))**

{
  Init $s[1 \ldots n]$
  $j \leftarrow 1$
  **for** $i \leftarrow n$ downto 1 **do:**
    $s[i] \leftarrow BWT[next[j]]$
    $j \leftarrow next[j]$
  **endfor**
  output $s$
}

| R | *f* | | | | | | *BWT* | next |
|---|---|---|---|---|---|---|---|---|
| 1 | \$ | b | a | n | a | n | a | 2 |
| 2 | a | \$ | b | a | n | a | n | 6 |
| 3 | a | n | a | \$ | b | a | n | 7 |
| 4 | a | n | a | n | a | \$ | b | 5 |
| 5 | b | a | n | a | n | a | \$ | 1 |
| 6 | n | a | \$ | b | a | n | a | 3 |
| 7 | n | a | n | a | \$ | b | a | 4 |

# PatternMatch algorithm with an example

Input: $BWT(s)$ for string $s$ of length $n$; pattern $p$ of length $m$.
Example: $BWT(s) = annb\$aa$, $p = ana$

Step 1)

| | | | $\downarrow$ |
|---|---|---|---|
| $p$ | a | n | a |

| | R | $f$ | | | | | | $BWT$ | next |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | \$ | b | a | n | a | n | a | 2 |
| $\rightarrow$ | 2 | a | \$ | b | a | n | a | n | 6 |
| $\rightarrow$ | 3 | a | n | a | \$ | b | a | n | 7 |
| $\rightarrow$ | 4 | a | n | a | n | a | \$ | b | 5 |
| | 5 | b | a | n | a | n | a | \$ | 1 |
| | 6 | n | a | \$ | b | a | n | a | 3 |
| | 7 | n | a | n | a | \$ | b | a | 4 |

# PatternMatch algorithm with an example

Input: $BWT(s)$ for string $s$ of length $n$; pattern $p$ of length $m$.
Example: $BWT(s) = annb\$aa$, $p = ana$

Step 2)

| | | ↓ | |
|---|---|---|---|
| $p$ | a | n | a |

| | R | $f$ | | | | | | $BWT$ | next |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | \$ | b | a | n | a | n | a | 2 |
| | 2 | a | \$ | b | a | n | a | n | 6 |
| | 3 | a | n | a | \$ | b | a | n | 7 |
| | 4 | a | n | a | n | a | \$ | b | 5 |
| | 5 | b | a | n | a | n | a | \$ | 1 |
| → | 6 | n | a | \$ | b | a | n | a | 3 |
| → | 7 | n | a | n | a | \$ | b | a | 4 |

# PatternMatch algorithm with an example

Input: $BWT(s)$ for string $s$ of length $n$; pattern $p$ of length $m$.
Example: $BWT(s) = annb\$aa$, $p = ana$

Step 3)

| | $\downarrow$ | | |
|---|---|---|---|
| $p$ | a | n | a |

| | R | $f$ | | | | | | $BWT$ | next |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | \$ | b | a | n | a | n | a | 2 |
| | 2 | a | \$ | b | a | n | a | n | 6 |
| $\rightarrow$ | 3 | a | n | a | \$ | b | a | n | 7 |
| $\rightarrow$ | 4 | a | n | a | n | a | \$ | b | 5 |
| | 5 | b | a | n | a | n | a | \$ | 1 |
| | 6 | n | a | \$ | b | a | n | a | 3 |
| | 7 | n | a | n | a | \$ | b | a | 4 |

Pattern $p$ found.

# References


Michael Burrows and David J Wheeler.
A block-sorting lossless data compression algorithm.
1994.
01618.

Heng Li and Richard Durbin.
Fast and accurate short read alignment with BurrowsâĂŞWheeler transform.
*Bioinformatics*, 25(14):1754–1760, 2009.
02571.

Heng Li and Richard Durbin.
Fast and accurate long-read alignment with BurrowsâĂŞWheeler transform.
*Bioinformatics*, 26(5):589–595, 2010.
00597.

Ben Langmead, Cole Trapnell, Mihai Pop, and Steven L Salzberg.
Ultrafast and memory-efficient alignment of short DNA sequences to the human genome.
*Genome Biol*, 10(3):R25, 2009.
03052.

Cole Trapnell and Steven L Salzberg.
How to map billions of short reads onto genomes.
*Nature biotechnology*, 27(5):455, 2009.
00144.