

An example for HW2:Q2

S = "AAABABBBCCC"

T = "AAACACCCBBB"

$\delta(x,x) = 2, \delta(x,y) = -1, \delta(-,x) = \delta(x,-) = -1,$

Best single alignment

AAABA---BBB

||| | |||

AAACACCCBBB

$$2(7) + -1(1) + -1(3) = 10$$

But the correct answer would be

AAA CCC

||| and |||

AAA CCC

or

AAA BBB

||| and |||

AAA BBB

$$2(6) + -1(0) + -1(0) = 12$$

Burrows-Wheeler Transform & FM-Index

CS 4364/5364

Memory Requirements

Human genome sequence (~3,000,000 bases/characters)

- Suffix Tree -- 40 GB
- Suffix Array -- 16 GB
- FM-Index -- 1.5GB

"Clearly" beyond the capacity of a
normal personal computer..

Burrows-Wheeler Transform

Remember our old friend the suffix array?

$T = \text{mississippi\$}$

SA_T	
12	\$
11	i\$
8	ippi\$
5	issippi\$
2	ississippi\$
1	mississippi\$
10	pi\$
9	ppi\$
7	sippi\$
4	sissippi\$
6	ssippi\$
3	ssissippi\$

Burrows-Wheeler Transform

Remember our old friend the suffix array?

$T = \text{mississippi\$}$

SA_T	
12	\$ mississippi
11	i\$mississipp
8	ippi\$mississ
5	issippi\$miss
2	ississippi\$m
1	mississippi\$
10	pi\$mississip
9	ppi\$mississi
7	sippi\$missis
4	sissippi\$mis
6	ssippi\$missi
3	ssissippi\$mi

Burrows-Wheeler Transform

Remember our old friend the suffix array?

$T = \text{mississippi\$}$

SA_T	
12	\$ mississippi
11	i\$ mississipp
8	ippi\$ mississ
5	issippi\$ miss
2	ississippi\$ m
1	mississippi\$
10	pi\$ mississip
9	ppi\$ mississi
7	sippi\$ missis
4	sissippi\$ mis
6	ssippi\$ missi
3	ssissippi\$ mi

Burrows-Wheeler Transform

Remember our old friend the suffix array?

$T = \text{mississippi\$}$

SA_T	BWT_T
12	i
11	p
8	s
5	s
2	m
1	\$
10	p
9	i
7	s
4	s
6	i
3	i

$$BWT_T = \begin{cases} T[SA_T[i] - 1] & \text{if } SA_T[i] > 1 \\ \$ & \text{if } SA_T[i] = 1 \end{cases}$$

Burrows-Wheeler Transform

Claim given $BWT_T = L$, one can reconstruct T .

- let V and W be two suffixes of T such that $V < W$ (lexicographic order)
- assume both V and W are preceded by an a in T , then suffix $aV < aW$
- Separately, consider suffix $T[i..n]$ which corresponds to position p_i in SA_T
- if $L[p_i] = a$ is the k^{th} occurrence of a in L , then the suffix $T[i-1..n]$ is the k^{th} suffix in \bar{a} (the region of SA of suffixes starting with a)

- define the last-to-first mapping:

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

- similarly

$LF^{-1}(j) = i$, where $SA[i] = (SA[j] + 1) \%_1 n$

$$i \%_1 n = \begin{cases} n & \text{if } i = 0 \\ i & \text{if } i \in [1 \dots n] \\ 1 & \text{if } i = n + 1 \end{cases}$$

i	0	1	2	3	...	$n-2$	$n-1$	n	$n+1$
$i \%_1 n$	n	1	2	3	...	n-2	n-1	n	1

SA_T	BWT_T
1 12	\$mississippi <i></i>
2 11	i\$mississipp <i>p</i>
3 8	ippi\$mississ <i>s</i>
4 5	issippi\$miss <i>s</i>
5 2	ississippi\$ m
6 1	mississippi\$
7 10	pi\$mississippi <p></p>
8 9	ppi\$mississ <i>i</i>
9 7	sippi\$missis
10 4	sissippi\$ m <i>s</i>
11 6	ssippi\$missi
12 3	ssissippi\$ m <i>i</i>

$$LF^{-1}(9) = 3$$

$$LF(9) = 11$$

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

SA_T	BWT_T
1	i
2	p
3	s
4	s
5	m
6	\$
7	p
8	i
9	s
10	s
11	i
12	i

$$x_1 = 1$$

$$\begin{array}{c} BWT_T[x_1] \\ \downarrow \\ i\$ \end{array}$$

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

SA_T	BWT_T
1	i
2	p
3	s
4	s
5	m
6	\$
7	p
8	i
9	s
10	s
11	i
12	i

$$x_2 = LF(x_1) = LF(1) = 2$$

$BWT_T[x_2]$
↓
p i \$

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

SA_T	BWT_T
1	i
2	p
3	s
4	s
5	m
6	\$
7	p
8	i
9	s
10	s
11	i
12	i

$$x_3 = LF(x_2) = LF(2) = 7$$

$BWT_T[x_3]$
↓
ppi\$

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

SA_T	BWT_T
1	i
2	p
3	s
4	s
5	m
6	\$
7	p
8	i
9	s
10	s
11	i
12	i

$$x_4 = LF(x_3) = LF(7) = 8$$

$BWT_T[x_4]$
↓
ippi\$

$LF(i) = j$, where $SA[j] = (SA[i] - 1) \%_1 n$

SA_T	BWT_T
1	i
2	p
3	s
4	s
5	m
6	\$
7	p
8	i
9	s
10	s
11	i
12	i

$$x_5 = LF(x_4) = LF(8) = 3$$

$BWT_T[x_5]$
↓
sippi\$

FM Index

The Ferragine-Manzini Index (FM-Index) of a string T consists of the following:

- $\text{BWT}_{T\$}$ -- the Burroughs-Wheeler Transform of the terminated string
- $C[x]$ -- for $x \in \Sigma$ counts occurrences of characters lexicographically **smaller** than x in T
- $\text{occ}(x, a)$ -- number of occurrences of $x \in \Sigma$ in $\text{BWT}_{T\$}[1 \dots a]$
 - we will omit the details of this data structure
 - can be stored in $O\left(\frac{n \log \log n}{\log n}\right)$ bits
 - lookup in $O(1)$ time

Counting Occurrences

Input

- pattern, $P = p_1, p_2, p_3, \dots, p_m$
- count array, C
- $BWT_{T\$}, L$
- occ data structure

Output

- number of occurrences of P in T

```
 $i = m$ 
 $(sp, ep) = (1, n)$ 
while  $sp \leq ep$  and  $i \geq 1$  do
     $c = p_j$ 
     $sp = C[c] + occ(c, sp-1) + 1$ 
     $ep = C[c] + occ(c, ep)$ 
     $i = i - 1$ 
if  $ep < sp$  then
    return 0
else
    return  $ep - sp + 1$ 
```

Counting Occurrences

$i = m$

$(sp, ep) = (1, n)$

while $sp \leq ep$ **and** $i \geq 1$ **do**

$c = p_j$

$sp = C[c] + occ(c, sp-1) + 1$

$ep = C[c] + occ(c, ep)$

$i = i - 1$

if $ep < sp$ **then**

return 0

else

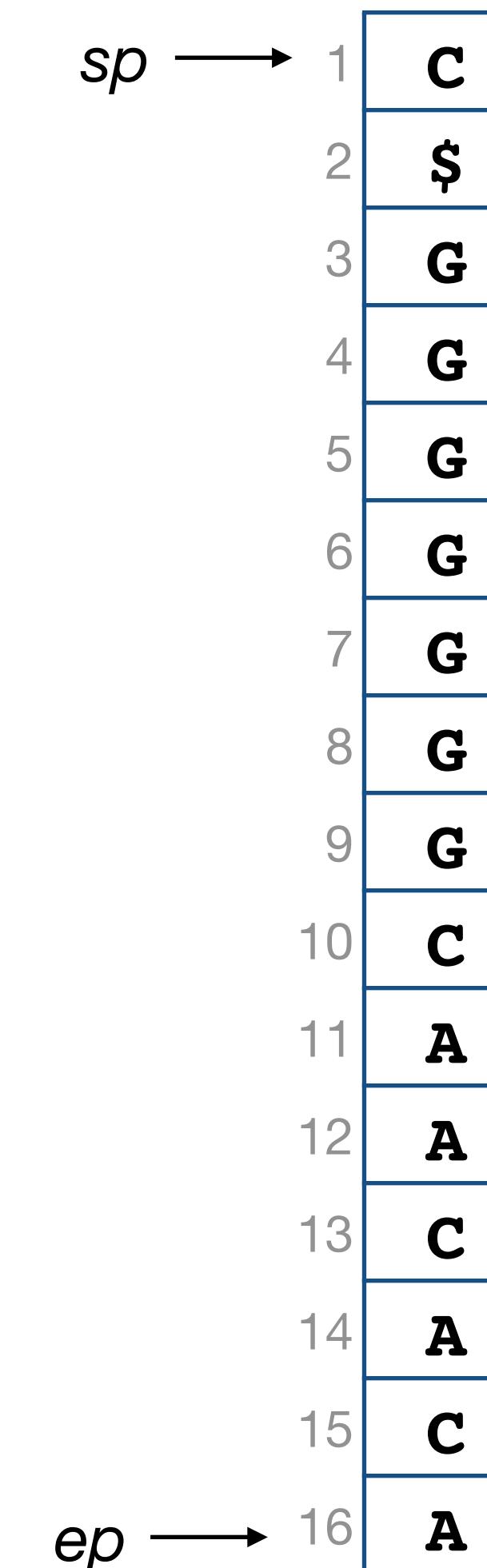
return $ep - sp + 1$

\$	0
A	1
C	5
G	9

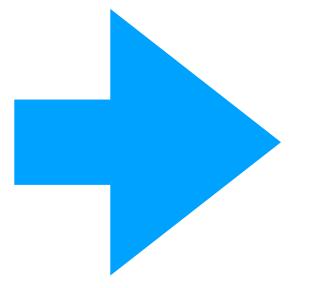
$$P = AGC$$
$$\uparrow$$
$$p_i$$

$$C[c] + occ(c, 0) + 1$$
$$5 + 0 + 1$$

$$C[c] + occ(c, 16)$$
$$5 + 4$$



\$	0
A	1
C	5
G	9



1	C
2	\$
3	G
4	G
5	G
6	G
7	G
8	G
9	G
10	C
11	A
12	A
13	C
14	A
15	C
16	A

AGAGCGAGAGCGCGC

Counting Occurrences

$i = m$

$(sp, ep) = (1, n)$

while $sp \leq ep$ **and** $i \geq 1$ **do**

$c = p_j$

$sp = C[c] + occ(c, sp-1) + 1$

$ep = C[c] + occ(c, ep)$

$i = i - 1$

if $ep < sp$ **then**

return 0

else

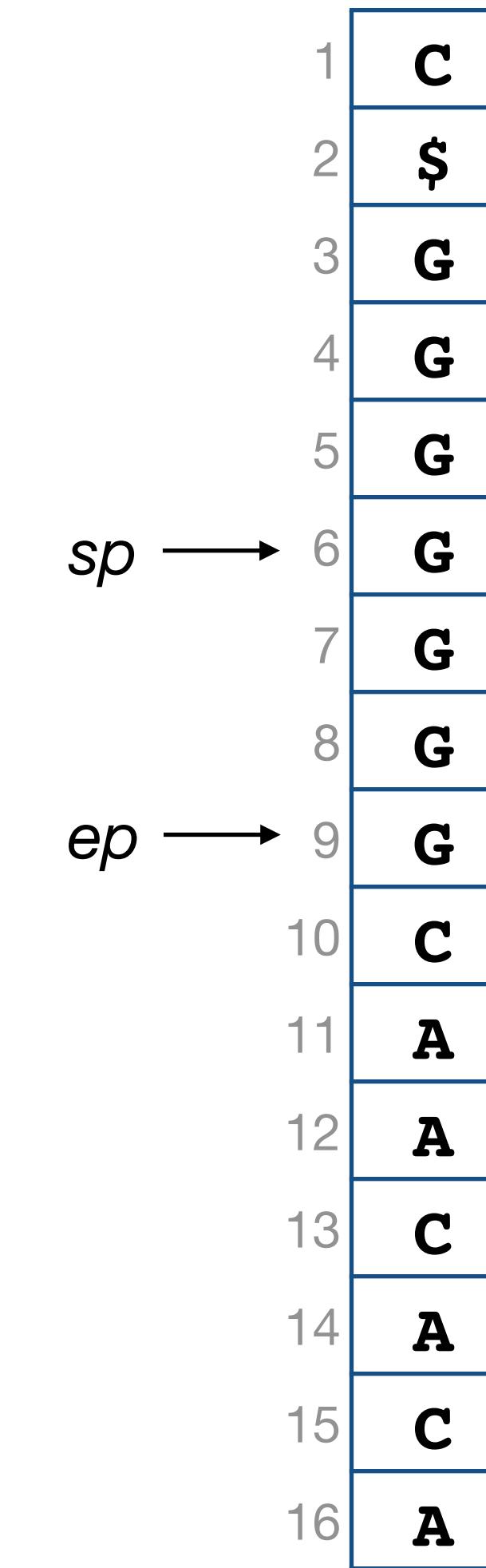
return $ep - sp + 1$

\$	0
A	1
C	5
G	9

$$P = AGC$$
$$\uparrow$$
$$p_i$$

$$C[G] + occ(G, 5) + 1$$
$$9 + 3 + 1$$

$$C[G] + occ(G, 9)$$
$$9 + 7$$



Counting Occurrences

$i = m$

$(sp, ep) = (1, n)$

while $sp \leq ep$ **and** $i \geq 1$ **do**

$c = p_j$

$sp = C[c] + occ(c, sp-1) + 1$

$ep = C[c] + occ(c, ep)$

$i = i - 1$

if $ep < sp$ **then**

return 0

else

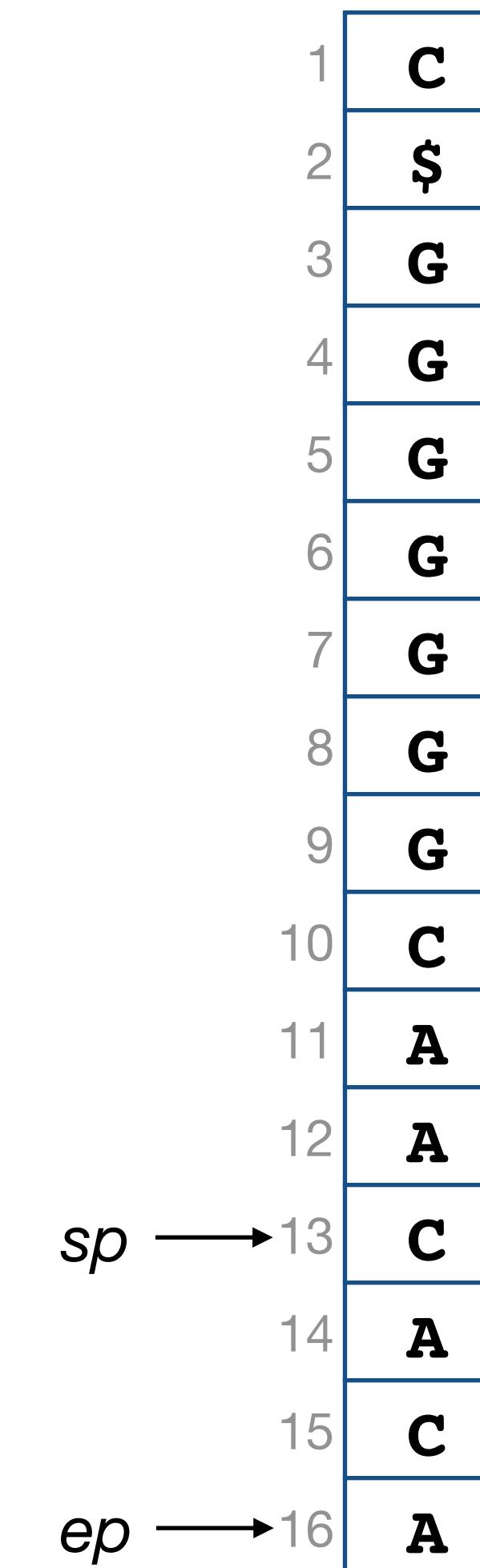
return $ep - sp + 1$

\$	0
A	1
C	5
G	9

$$P = AGC$$
$$\uparrow$$
$$p_i$$

$$C[A] + occ(A, 13) + 1$$
$$1 + 2 + 1$$

$$C[A] + occ(A, 16)$$
$$1 + 4$$



Counting Occurrences

```
i = m  
(sp, ep) = (1, n)  
while sp ≤ ep and i ≥ 1 do  
    c = pj  
    sp = C[c] + occ(c, sp-1)+1  
    ep = C[c] + occ(c, ep)  
    i = i - 1  
if ep < sp then  
    return 0  
else  
    return ep - sp + 1
```

\$	0
A	1
C	5
G	9

$$P = \text{AGC}$$
$$\uparrow$$
$$p_i$$

The BWT Index doesn't contain the SA, how do we recover the positions in T?

1	C	\$
2	\$	AGAGCGAGAGC GCGC
3	G	AGAGCGCGC\$
4	G	AGCGAGAGCGCGC\$
5	G	AGCGCGC\$
6	G	C\$
7	G	CGAGAGCGCGC\$
8	G	CGC\$
9	G	CGCGC\$
10	C	GAGAGCGCGC\$
11	A	GAGCGAGAGCGCGC\$
12	A	GAGCGCGC\$
13	C	GC\$
14	A	GCGAGAGCGCGC\$
15	C	GCGC\$
16	A	GCGCGC\$

Group Exercise

Given a string $S = S[1\dots n]$ and a number k . Find the smallest substring of S that occurs at least k times, if it exists. Show how to solve this problem in $O(n)$ time.

What about the longest?