String Algorithms and Data Structures

FM Index

CS 199-225
Brad Solomon

April 3, 2023

Department of Computer Science
Reversible permutation of the characters of a string

Burrows-Wheeler Transform

$abababa$

$\text{Sort}$

$\text{Burrows-Wheeler Matrix}$

$\text{BWT}(T)$

They're sorted by right-context

Any ranking we give to characters in \( T \) will match in \( F \) and \( L \)
Burrows-Wheeler Transform: LF Mapping

Another way to visualize:

\[
\begin{array}{cccccccccccccccc}
\$ & a_3 & $ & a_3 & $ & a_3 & $ & a_3 & $ & a_3 & $ & a_3 & $ & a_3 & \$ & a_3 \\
& a_3 & b_1 & a_3 & b_1 & a_3 & b_1 & a_3 & b_1 & a_3 & b_1 & a_3 & b_1 & \ \\
a_1 & b_0 & a_1 & b_0 & a_1 & b_0 & a_1 & b_0 & a_1 & b_0 & a_1 & b_0 & \ \\
a_2 & a_1 & a_2 & a_1 & a_2 & a_1 & a_2 & a_1 & a_2 & a_1 & a_2 & a_1 & \ \\
a_0 & $ & a_0 & $ & a_0 & $ & a_0 & $ & a_0 & $ & a_0 & $ & a_0 & \ \\
b_1 & a_2 & b_1 & a_2 & b_1 & a_2 & b_1 & a_2 & b_1 & a_2 & b_1 & a_2 & b_1 & \ \\
b_0 & a_0 & b_0 & a_0 & b_0 & a_0 & b_0 & a_0 & b_0 & a_0 & b_0 & a_0 & b_0 & \ \\
T: & a_0 & b_0 & a_1 & a_2 & b_1 & a_3 & $ \\
\end{array}
\]
A review of ‘F’ and ‘L’

$L = \text{CGGGCC}\$ \quad \Sigma = \text{"ACGT"}

How can we represent $F$?
A review of ‘F' and ‘L’

$L = \text{CGGGCC}$  \quad \Sigma = “\text{ACGT}”$

How can we represent $F$?

As a full text string:  \quad F = $\text{CCCGGG}$

As a map<string, int>:  \quad F = \{‘$’: 1, ‘C’: 3, ‘G’: 3\}

As a vector<int>:  \quad F = [0, 3, 3, 0]
A review of ‘F’ and ‘L’

\( \text{BWT}(T) = e\$lppa \)

What row index in \( F \) contains ‘e’?

What row index in \( L \) contains ‘e’?

What row index in \( F \) contains the second ‘p’?
FM Index

An index combining the BWT with a few small auxiliary data structures

Core of index is **first (F)** and **last (L) rows** from BWM:

$L$ is the same size as $T$

$F$ can be represented as array of $|\Sigma|$ integers (or not stored at all!)

We’re discarding $T$ — *we can recover it from L!*
FM Index: Querying

\[ P = \text{A A A} \]

\[
\begin{align*}
\$ & \quad \text{B B B A A A} & \quad \text{A}_0 \\
\text{A}_0 & \quad \$ \quad \text{B B B A} & \quad \text{A}_1 \\
\text{A}_1 & \quad \text{A} \quad \$ \quad \text{B B B} & \quad \text{A}_2 \\
\text{A}_2 & \quad \text{A A} \quad \$ \quad \text{B B} & \quad \text{B}_0 \\
\text{B}_0 & \quad \text{A A A} \quad \$ \quad \text{B} & \quad \text{B}_1 \\
\text{B}_1 & \quad \text{B A A A} \quad \$ \quad \text{B} & \quad \text{B}_2 \\
\text{B}_2 & \quad \text{B B A A A} \quad \text{A} \quad \$ & \\
\end{align*}
\]
FM Index: Querying

$P = B \ A \ B$

$ 
\begin{array}{cccccc}
  & \$ & B & B & B & A & A & A_0 \\
  A_0 & $ & B & B & B & A & A_1 \\
  A_1 & A & $ & B & B & B & A_2 \\
  A_2 & A & A & $ & B & B & B_0 \\
  B_0 & A & A & A & $ & B & B_1 \\
  B_1 & B & A & A & A & $ & B_2 \\
\end{array}$
FM Index: Lingering Issues
FM Index: Lingering Issues

(1) Scanning for preceding character in $L$ is slow

(2) Need way to find where matches occur in $T$:

We don’t store ranks!

Current output: $[3, 4]$

Location in $T$: $[0, 3]$

This is where our auxiliary data structures come in…
FM Index: Fast rank calculations

Is there a fast way to determine which *specific* bs precede the as in our range?

More generally, given a range in $L$ and a character to search, how can we quickly find all matches (and their ranks)?
**FM Index: Occurrence Table**

Idea: pre-calculate cumulative # a, b in $L$ up to every row:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$L$</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**FM Index: Occurrence Table**

Idea: pre-calculate cumulative # as, bs in $L$ up to every row:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>a</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$$</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>a</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>
**FM Index: Occurrence Table**

Query: ‘aba’

Idea: pre-calculate cumulative # `a`s, `b`s in `L` up to every row:

<table>
<thead>
<tr>
<th>$</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FM Index: Occurrence Table
**FM Index: Occurrence Table**

**Query:** 'aba'

Idea: pre-calculate cumulative # as, bs in L up to every row:

<table>
<thead>
<tr>
<th>$</th>
<th>a</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>a</td>
<td>$</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

0 bs up to & including this row

2 bs up to & including this row
FM Index: Occurrence Table

Query: ‘aba’

Idea: pre-calculate cumulative # a$s, b$s in $L$ up to every row:

\[
\begin{array}{c|c|c|c}
F & L & a & b \\
\$ & a & 1 & 0 \\
a & b & 1 & 1 \\
a & b & 1 & 2 \\
a & a & 2 & 2 \\
a & $ & 2 & 2 \\
b & a & 3 & 2 \\
b & a & 4 & 2 \\
\end{array}
\]
What two indices should I look up? What ranks did we find?

<table>
<thead>
<tr>
<th>F</th>
<th>L</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>a</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
FM Index: Occurrence Table

An index combining the BWT with a few small auxiliary data structures

Occurrence table speeds up \( L \) lookup by implicitly storing ranks

Table is \( m \times |\Sigma| \) integers — that's worse than a suffix array!
Next idea: pre-calculate # as, bs in L up to some rows, e.g. every 5\textsuperscript{th} row. Call pre-calculated rows checkpoints.

| $  | a | 1 | 0 |
| a  | b |    |   |
| a  | b |    |   |
| a  | b |    |   |
| a  | a |    |   |
| a  | $ |    |   |
| b  | a | 3  | 2 |
| b  | a |    |   |
FM Index: Occurrence Table

To resolve a lookup for a non-checkpoint row, walk to nearest checkpoint. Use value at that checkpoint, *adjusted for characters we saw along the way.*

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>a</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
If checkpoints are $O(1)$ distance apart, lookups are $O(1)$
FM Index: Occurrence Table

An index combining the BWT with *a few small auxiliary data structures*

Occurrence table speeds up $L$ lookup by implicitly storing **ranks**

Checkpoints reduce the storage costs (Still $O(m)$ but better than SA)
FM Index: Querying

Problem 2: We don’t know where the matches are in T...

\[ P = \text{aba} \]

Got the same range, [3, 4], we would have got from suffix array

<table>
<thead>
<tr>
<th>F</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>$</td>
</tr>
<tr>
<td>a</td>
<td>$</td>
</tr>
</tbody>
</table>

\[ [3, 4] \]

Where are these?

Index: 0, 3
**FM Index: Suffix Array Sampling**

Idea: store some suffix array elements, but not all

<table>
<thead>
<tr>
<th>$F$</th>
<th>$L$</th>
<th>$SA'$ (evens only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$$ a b a a b a</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>a $ a b a a b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a a b a $ a b</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>a b a $ a b a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a b a a b a $</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>b a $ a b a a</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>b a a b a $ a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Lookup for row 4 succeeds

Lookup for row 3 fails - SA entry was discarded
### FM Index: Suffix Array Sampling

LF Mapping tells us that “a” at the end of row 3 corresponds to...

...“a” at the beginning of row 2

| $ | a | b | a | a | b | a | 2 |
| a | $ | a | b | a | a | b | 6 |
| a | a | b | a | $ | a | b | 0 |
| a | b | a | $ | a | b | a | 4 |
| b | a | $ | a | b | a | a | 4 |
| b | a | a | b | a | $ | a | 6 |

If saved SA values are $O(1)$ positions apart in $T$, resolving index is $O(1)$ time.
FM Index: Suffix Array Sampling

Many LF-mapping steps may be required to get to a sampled row:

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td></td>
</tr>
</tbody>
</table>

Starting here

\[ F \quad L \quad SA' \text{ (every 4th)} \]

Starting here
FM Index: Suffix Array Sampling

Many LF-mapping steps may be required to get to a sampled row:

\[ \begin{array}{cccc}
F & L & \text{SA'} (\text{every 4th}) \\
\$ & a & b & a \ a \ b \ a \\
a & a & b & a \ a \ b \\
a & a & b & a \ b \\
a & b & a \ a \ b & a \\
a & b & a \ a \ b & a \\
b & a \ b & a \ a \ b & a \\
b & a \ b & a \ a \ b & a \\
\end{array} \]
FM Index: Suffix Array Sampling

Many LF-mapping steps may be required to get to a sampled row:

Starting here ➔

<table>
<thead>
<tr>
<th>$</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>$</td>
<td>a</td>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
</tr>
</tbody>
</table>

$SA'$ (every 4th)

Starting here ➔

<table>
<thead>
<tr>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
FM Index: Suffix Array Sampling

Many LF-mapping steps may be required to get to a sampled row:

\[
\begin{array}{c|c|c|c|c|c}
F & L & SA' (every 4th) \\
$ & a & b & a & a & b \\
a & $ & a & b & a & a \\
a & a & b & a & $ & b \\
a & $ & a & b & a & a \\
\end{array}
\]

Missing value = 0 (SA val at destination) + 3 (# steps to destination) = 3
FM Index: Suffix Array Sampling

An index combining the BWT with *a few small auxiliary data structures*

Stores all index positions in T with $O(1)$ extra work to calculate

```
$ a b a a b a
a $ a b a a b
a a b a $ a b
```

```
<table>
<thead>
<tr>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
</tbody>
</table>
```

Index: 0

Index: $0 + 3 = 3$

```
| a b a $ a b a |
| a b a a b a $|
| b a $ a b a a |
| b a a b a $ a |
```

Three steps

**Let's put all these pieces together...**
FM Index: Querying

$P = \text{aba}$

\[
\begin{array}{cccc}
F & L \\
\$ & a & b & a & a & b & a_0 \\
a_0 & $ & a & b & a & a & b \\
a_1 & a & b & a & $ & a & b \\
a_2 & b & a & $ & a & b & a_1 \\
a_3 & b & a & a & b & a & $ \\
b & a & $ & a & b & a & a_2 \\
b & a & a & b & a & $ & a_3 \\
\end{array}
\]

get_frange()
pair<int, int> get_frange(string c, int s, int e)

Input:

string c: The char we are looking for in F
int s: The starting rank value
int e: The ending rank value

Output:
A pair of values (index start, index end)

What are c, s, and e?

What are the output values?
FM Index: Querying

$P = \text{aba}$

\[
F = \begin{array}{c}
\$ a b a a b a_0 \\
a_{0} \$ a b a a b \\
a_1 a b a \$ a b \\
a_2 b a \$ a b a_1 \\
a_3 b a a b a \$
\end{array}
\]

\[
L = \begin{array}{c}
\$ a b a a b a_0 \\
a_0 \$ a b a a b \\
a_1 a b a \$ a b \\
a_2 b a \$ a b a_1 \\
a_3 b a a b a \$
\end{array}
\]

get\_frange() get\_lrange()
pair<int, int> get_lrange(string c, int s, int e)

Input:
- **string c**: The char we are looking for in F
- **int s**: The starting **index** of our range
- **int e**: The ending **index** of our range

Output:
- A pair of values (# occurrences start, end)

What are c, s, and e?

What are the output values?
FM Index: Querying

$P = \text{aba}$

$F$

\begin{align*}
\text{a} & \text{b} \text{a} \text{a} \text{b} \text{a}_0 \\
\text{a}_0 & \text{a} \text{b} \text{a} \text{a} \text{b} \\
\text{a}_1 & \text{a} \text{b} \text{a} \$ \text{a} \text{b} \\
\text{a}_2 & \text{b} \text{a} \$ \text{a} \text{b} \text{a}_1 \\
\text{a}_3 & \text{b} \text{a} \text{a} \text{b} \$ \\
\text{b} & \text{b} \text{a} \$ \text{a} \text{b} \text{a}_2 \\
\text{b} & \text{b} \text{a} \text{a} \text{b} \$ \text{a}_3 \\
\end{align*}$

$L$

get\_frange()

get\_lrange()
pair<int, int> get_frange(string c, int s, int e)

Input:

- **string c**: The char we are looking for in *F*
- **int s**: The starting rank value
- **int e**: The ending rank value

Output:

- A pair of values (index start, index end)

What are *c*, *s*, and *e*?

What are the output values?
FM Index: Querying

$P = \text{aba}$

$F$  $L$

\[
\begin{array}{cccccc}
\$ & a & b & a & a & b & a_0 \\
a_0 & \$ & a & b & a & a & b \\
a_1 & a & b & a & \$ & a & b \\
a_2 & b & a & \$ & a & b & a_1 \\
a_3 & b & a & a & b & a & \$
\end{array}
\]

`get_frange()`  `get_lrange()`
get_lrange(‘a’, 5, 6)->[2,4]  

\[ P = \texttt{aba} \quad \text{get_lrange(‘a’, 5, 6)} \rightarrow [2, 4] \]

\[ P = \texttt{aba} \quad \text{get_lrange(‘a’, 5, 6)} \rightarrow [2, 4] \]

\[ \begin{array}{cccccc}
F & L & F & L \\
\$ & a & b & a & a & b & a_0 \\
a_0 & \$ & a & b & a & a & b_0 \\
a_1 & a & b & a & \$ & a & b_1 \\
a_2 & b & a & \$ & a & b & a_1 \\
a_3 & b & a & a & b & a & \$ \\
b_0 & a & \$ & a & b & a & a_2 \quad \text{[boxed]} \\
b_1 & a & a & b & a & \$ & a_3 \quad \text{[boxed]} \\
\end{array} \]

\[ \begin{array}{cccccc}
F & L & F & L \\
\$ & a & b & a & a & b & a_0 \\
a_0 & \$ & a & b & a & a & b_0 \\
a_1 & a & b & a & \$ & a & b_1 \\
a_2 & b & a & \$ & a & b & a_1 \quad \text{[boxed]} \\
a_3 & b & a & a & b & a & \$ \quad \text{[boxed]} \\
b_0 & a & \$ & a & b & a & a_2 \\
b_1 & a & a & b & a & \$ & a_3 \quad \text{[boxed]} \\
\end{array} \]

\[ \text{get_frange(‘a’, 2, 3)} \rightarrow [3, 4] \]

\[ \text{SA[3]} = 3, \ SA[4] = 0 \rightarrow \text{Return } \{0, 3\} \]
Finding all matches of $P$ occurs in $T$ in FM Index is ____________ time
Assignment 9: a_fmi

Learning Objective:

Construct a full FM Index

Implement exact pattern matching on a FM Index

Consider: How would you modify the provided code to handle sub-sampling in the Occurrence Table (OT) or Suffix Array (SA)?
Let $a =$ fraction of rows we keep

Let $b =$ fraction of SA elements we keep

FM Index consists of these, plus $L$ and $F$ columns

Note: suffix tree/array didn't have parameters like $a$ and $b$
FM Index

Components of FM Index: (blue indicates what we can adjust by changing $a$ & $b$)

First column ($F$): $\sim | \Sigma |$ integers
Last column ($L$): $m$ characters
SA sample: $m \cdot a$ integers, $a$ is fraction of SA elements kept
OT Checkpoints: $m \cdot | \Sigma | \cdot b$ integers, $b$ is fraction of tallies kept

For DNA alphabet (2 bits / nt), $T =$ human genome, $a = 1/32$, $b = 1/128$:

First column ($F$): 16 bytes
Last column ($L$): 2 bits $\times$ 3 billion chars $= 750$ MB
SA sample: 3 billion chars $\times$ 4 bytes / 32 $= \sim 400$ MB
OT Checkpoints: 3 billion $\times$ 4 alphabet chars $\times$ 4 bytes / 128 $= \sim 400$ MB

Total $\approx 1.5$ GB  ~0.5 bytes per input char
FM Index: Small Memory Footprint

- **Suffix tree**: $\geq 45$ GB
- **Suffix array**: $\geq 12$ GB
- **FM Index**: $\sim 1.5$ GB
## Suffix-Based Index Bounds

<table>
<thead>
<tr>
<th>Time: Does P occur?</th>
<th>Suffix tree</th>
<th>Suffix array</th>
<th>FM Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time: Count $k$ occurrences of P</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time: Report $k$ locations of P</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Space**

**Needs T?**

**Bytes per input character**

$m = |T|$, $n = |P|$, $k = \# \text{ occurrences of } P \text{ in } T$
### Suffix-Based Index Bounds

<table>
<thead>
<tr>
<th></th>
<th>Suffix tree</th>
<th>Suffix array</th>
<th>FM Index</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time: Does P occur?</strong></td>
<td>$O(n)$</td>
<td>$O(n \log m)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td><strong>Time: Count k occurrences of P</strong></td>
<td>$O(n + k)$</td>
<td>$O(n \log m)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td><strong>Time: Report k locations of P</strong></td>
<td>$O(n + k)$</td>
<td>$O(n \log m + k)$</td>
<td>$O(n + k)$</td>
</tr>
<tr>
<td><strong>Space</strong></td>
<td>$O(m)$</td>
<td>$O(m)$</td>
<td>$O(m)$</td>
</tr>
<tr>
<td><strong>Needs T?</strong></td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td><strong>Bytes per input character</strong></td>
<td>$&gt;15$</td>
<td>$\sim 4$</td>
<td>$\sim 0.5$</td>
</tr>
</tbody>
</table>

\[ m = |T|, \quad n = |P|, \quad k = \# \text{ occurrences of } P \text{ in } T \]