BWT for repetitive texts: Run-length FM index

Ben Langmead

BWT runs

```
row_row_row_your_boat
row_row_row_your_boat
row_row_row_your_boat$
```

↓ BWT

```
trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
```

↓ RLE

(**t**, 1), (**r**, 3), (**w**, 9), (**o**, 3), (**_**, 3), (**b**, 3), (**y**, 3), (**r**, 9), (**u**, 3), (**t**, 2), (**$**, 1), (**_**, 6), (**a**, 3), (**o**, 12), (**_**, 3)

Run-length FM Index

	Count		Locate	
	Space	Time	Space	Time
FM Index (2000)	$O(n)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
RLFM Index (2005)	$O(r)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
r-index (2018)	$O(r)$	$O(m)$	$O(r)$	$O(m + \text{occ})$

n = reference length, m = query length,
r = # BWT runs

(bounds simplified)

RLFM: Mäkinen V, and Navarro G. Succinct suffix arrays based on run-length encoding. Annual Symposium on CPM. Springer, Berlin, Heidelberg. 2005. pp45–56.

r-index: Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

Run-length FM Index

	Count		Locate	
	Space	Time	Space	Time
FM Index (2000)	$O(n)$	$O(m)$	$O(n)$	$O(m + \mathrm{occ})$
RLFM Index (2005)	$O(r)$	$O(m)$	$O(n)$	$O(m + \mathrm{occ})$
r-index (2018)	$O(r)$	$O(m)$	$O(r)$	$O(m + \mathrm{occ})$

n = reference length, m = query length,
r = # BWT runs

(bounds simplified)

RLFM: Mäkinen V, and Navarro G. Succinct suffix arrays based on run-length encoding. Annual Symposium on CPM. Springer, Berlin, Heidelberg. 2005. pp45–56.

r-index: Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

Run-length FM Index

How?

	Count		Locate	
	Space	Time	Space	Time
FM Index (2000)	$O(n)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
RLFM Index (2005)	$O(r)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
r-index (2018)	$O(r)$	$O(m)$	$O(r)$	$O(m + \text{occ})$

n = reference length, m = query length,
r = # BWT runs

(bounds simplified)

RLFM: Mäkinen V, and Navarro G. Succinct suffix arrays based on run-length encoding. Annual Symposium on CPM. Springer, Berlin, Heidelberg. 2005. pp45–56.

r-index: Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

Run-length FM Index

How?

	Count		Locate	
	Space	Time	Space	Time
FM Index (2000)	$O(n)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
RLFM Index (2005)	$O(r)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
r-index (2018)	$O(r)$	$O(m)$	$O(r)$	$O(m + \text{occ})$

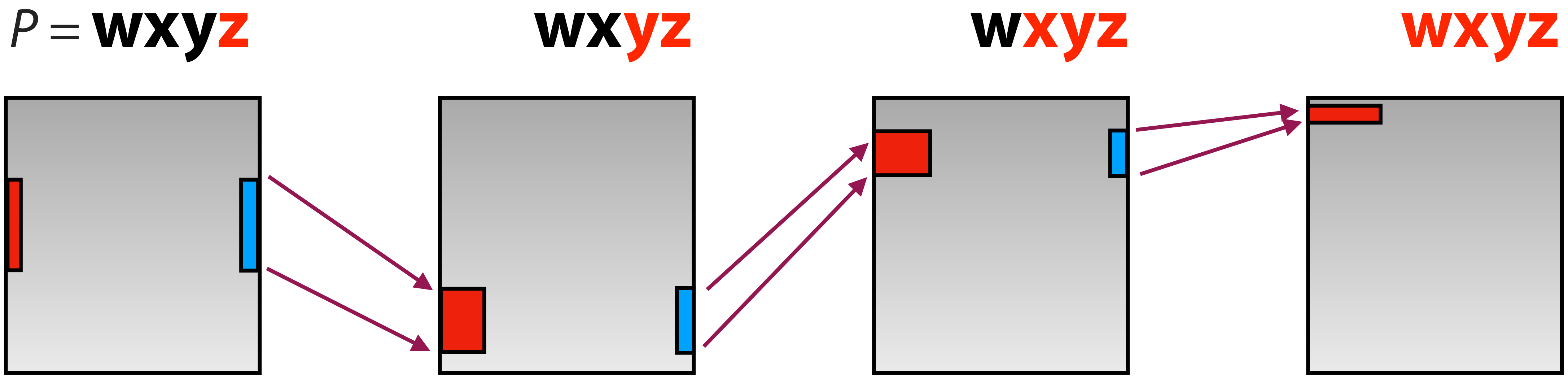
n = reference length, m = query length,
r = # BWT runs

(bounds simplified)

RLFM: Mäkinen V, and Navarro G. Succinct suffix arrays based on run-length encoding. Annual Symposium on CPM. Springer, Berlin, Heidelberg. 2005. pp45–56.

r-index: Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

FM Index: query



P = **wxyz**

wxyz

wxyz

wxyz

"Backward search"

FM Index: query

$P =$ **aba**

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
b_1	a	a	b	a	\$	a_3

	Rank	Skip	Range
a		1 x \$ = 1 1 x \$ + 5 x **a** = 5	1 5
b	$L.\text{rank}_b(1) = 0$ $L.\text{rank}_b(5) = 2$	1 x \$ + 5 x **a** = 5	0 + 5 = 5 2 + 5 = 7
a	$L.\text{rank}_a(5) = 2$ $L.\text{rank}_a(7) = 4$	1 x \$ = 1	0 + 1 = 3 2 + 1 = 5

FM Index: query

L `trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___`

F `$____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy`

FM Index: query

$i = 35, \ c =$ o

$$\downarrow$$

L `trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaooooooooooooo___`

F `$____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy`

$$LF(\mathrm{i}, c) \leftarrow \underbrace{C[c]}_{\text{"skip to the } c \text{ section"}} + \underbrace{L\,.\,\mathrm{rank}_c(\mathrm{i})}_{\text{"skip to } c \text{ of correct rank"}}$$

FM Index: query

$i = 35, \; c = \texttt{o}$

$\downarrow$

L `trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___`

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

F `$____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy`

$$LF(\mathrm{i}, c) \leftarrow \underbrace{C[c]}_{\text{"skip to the } c \text{ section"}} + \underbrace{L\,.\,\mathrm{rank}_c(\mathrm{i})}_{\text{"skip to } c \text{ of correct rank"}}$$

FM Index: query

$i = 35,\ c = \mathrm{o}$

L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

$L\,.\,\mathrm{rank}_{\mathrm{o}}(35) = 3$

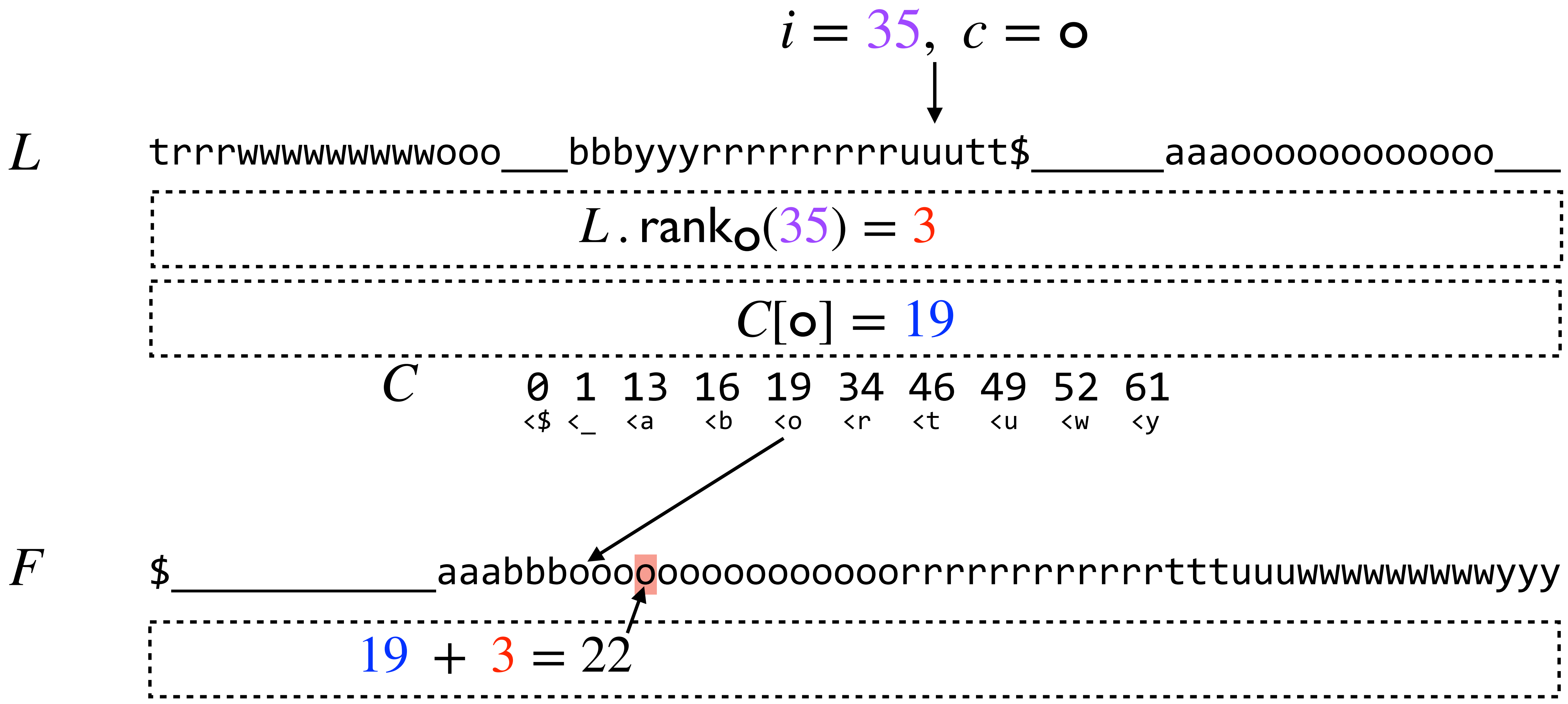
$C[\mathrm{o}] = 19$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

F $_____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy

$$LF(\mathrm{i}, c) \leftarrow \underbrace{C[c]}_{\text{"skip to the } c \text{ section"}} + \underbrace{L\,.\,\mathrm{rank}_c(\mathrm{i})}_{\text{"skip to } c \text{ of correct rank"}}$$

FM Index: query



$i = 35,\ c = \text{o}$

L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

$L\,.\,\text{rank}_{\text{o}}(35) = 3$

$C[\text{o}] = 19$

C 0 1 13 16 19 34 46 49 52 61

<\$ <_ <a <b <o <r <t <u <w <y

F $____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy

$19 + 3 = 22$

$$LF(\text{i}, c) \leftarrow \underbrace{C[c]}_{\text{"skip to the } c \text{ section"}} + \underbrace{L\,.\,\text{rank}_c(\text{i})}_{\text{"skip to } c \text{ of correct rank"}}$$

FM Index: query

$i = 28, \; c = \mathtt{r}$

L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

F $____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy

FM Index: query

$i = 28, \; c =$ r

L

```
trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
```

$L\,.\,\mathrm{rank}_{\mathsf{r}}(28) = 6$

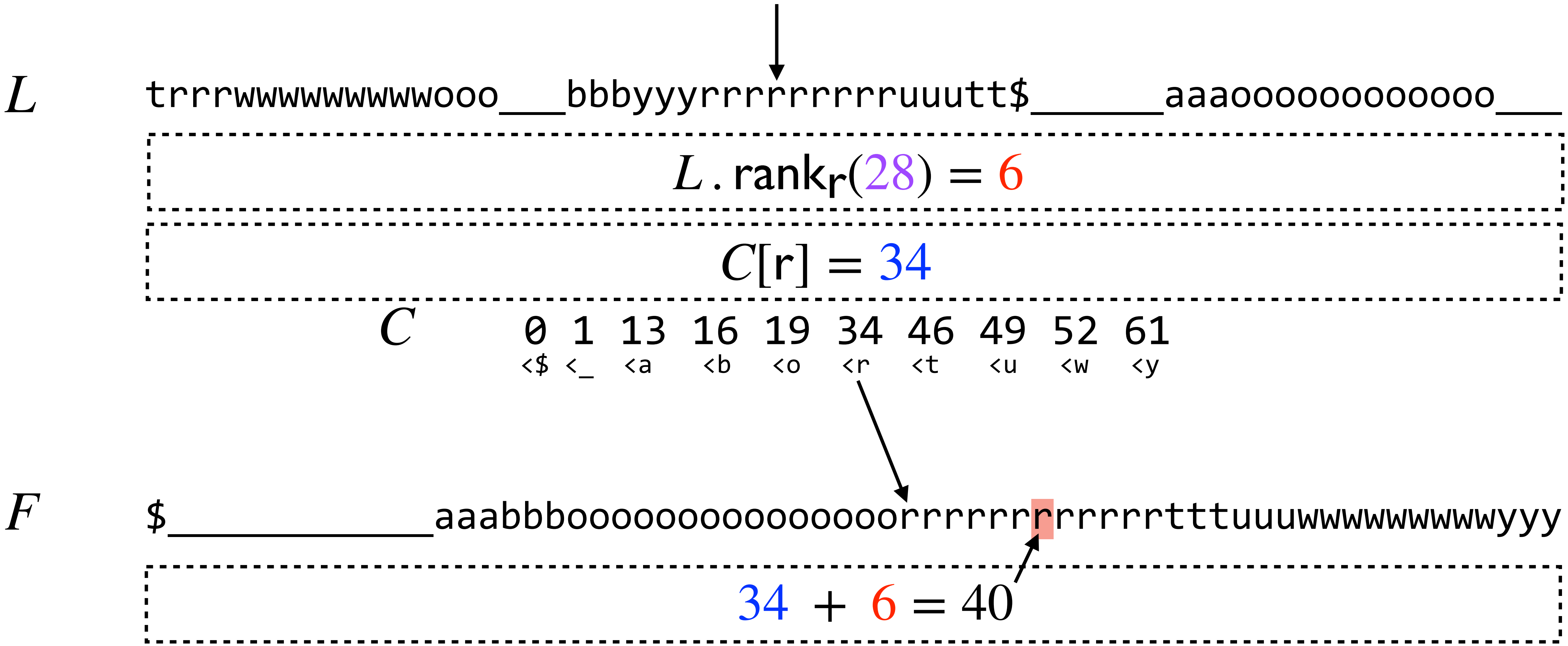
$C[\mathsf{r}] = 34$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

F

```
$____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy
```

FM Index: query



$i = 28, \;\; c = \mathrm{r}$

L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

$L\,.\,\mathrm{rank}_{\mathrm{r}}(28) = 6$

$C[\mathrm{r}] = 34$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

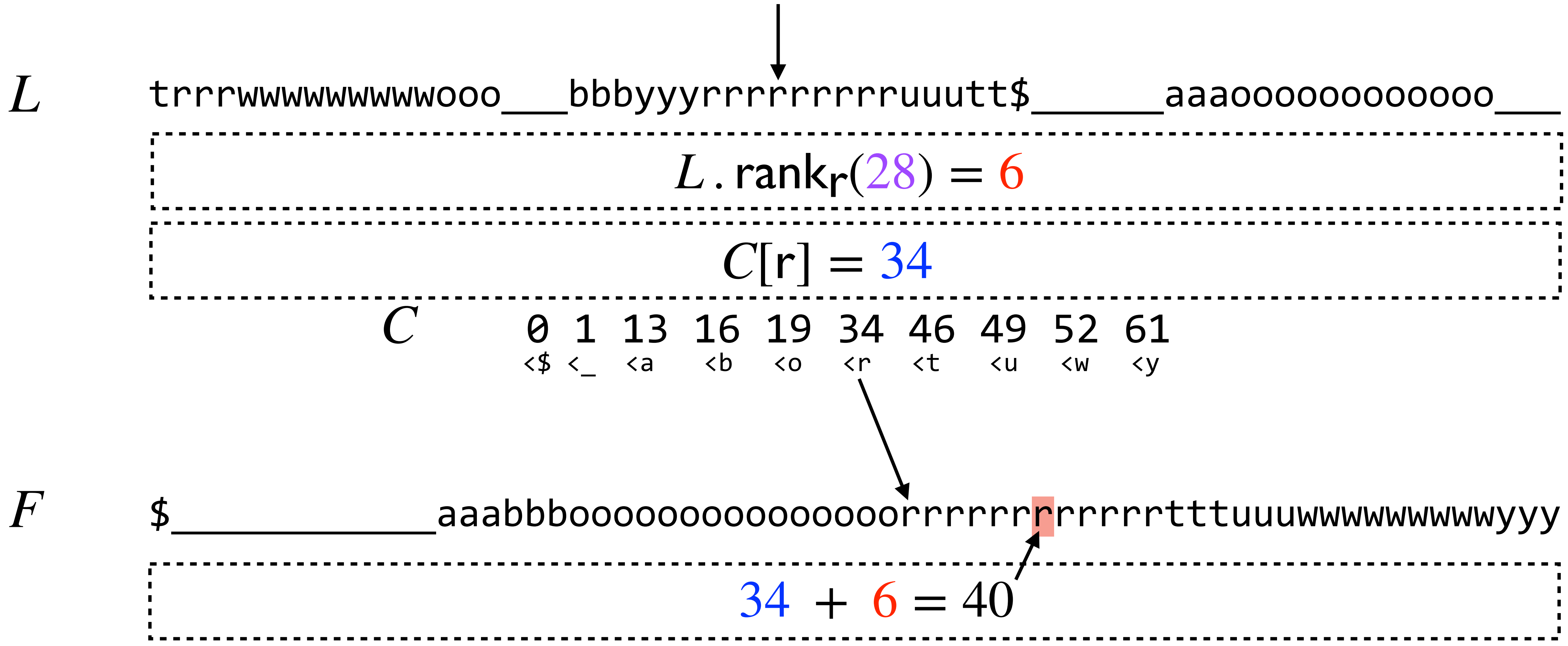
F $____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy

$34 + 6 = 40$

Balanced wavelet tree for $L\,.\,\mathrm{rank}_{x}(\,\ldots\,)$ takes

FM Index: query

$i = 28,\ c = \mathrm{r}$



L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

$L\,.\,\mathrm{rank}_{\mathrm{r}}(28) = 6$

$C[\mathrm{r}] = 34$

C 0 1 13 16 19 34 46 49 52 61

<$ <_ <a <b <o <r <t <u <w <y

F $____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy

$34 + 6 = 40$

Balanced wavelet tree for $L\,.\,\mathrm{rank}_{\mathrm{x}}(\,.\,.\,.\,)$ takes $O(n \log \sigma)$ bits

Can we attack $O(n)$, replacing with $O(r)$ & minor functions of n?

Run-lengthifying

L `trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___`

+

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

Run-lengthifying

L trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

B 1100100000000100100100100100000000100101100000100100000000000100

trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___

S trwo_byrut$_ao_

Run-lengthifying

L
```
trrrwwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
```

```
trrrwwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
```

B
```
11001000000000100100100100100000000100101100000100100000000000100
```

```
trrrwwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
```

S
```
trwo_byrut$_ao_
```

Run-lengthifying

$i = 35, \; c = \texttt{o}$

L `trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___`

Run-lengthifying

$i = 35,\ c =$ `o`

B `1100100000000100100100100100000000100101100000100100000000000100`

S `trwo_byrut$_ao_`

Run-lengthifying

$i = 35, \; c = \texttt{o}$

↓

B `1100100000000100100100100100000000100101100000100100000000000100`

$B.\text{rank}_1(35 + 1) = 9$

S `trwo_byrut$_ao_`

Run-lengthifying

$i = 35, \ c =$ o

↓

B 1100100000000100100100100100000000100101100000100100000000000100

$B.\mathrm{rank}_1(35 + 1) = 9$

S **trwo_byrut$_ao_**

C

	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

Run-lengthifying

$i = 35, \; c = \texttt{o}$

↓

B `1100100000000100100100100100000000100101100000100100000000000100`

$B\,.\,\mathrm{rank}_1(35 + 1) = 9$

S **`trwo_byrut$_ao_`**

$S\,.\,\mathrm{rank}_{\texttt{o}}(9 - 1) = 1$

$C[\texttt{o}] = 19$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

F `$____________aaabbbooooooooooooooorrrrrrrrrrrrtttuuuwwwwwwwwwyyy`

We have **some** but **not all** of the answer. We can jump to the "o section" but we **don't know how far in to jump.**

Run-lengthifying

Make B' by concatenating runs in alphabetical groups:

```
   trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
B  1100100000000100100100100100000000100101100000100100000000000100
```

```
$: 0000000000000000000000000000000000000001000000000000000000000000
_: 0000000000000000100000000000000000000000100000000000000000000100
a: 0000000000000000000000000000000000000000000000100000000000000000
b: 0000000000000000000100000000000000000000000000000000000000000000
o: 0000000000000100000000000000000000000000000000000100000000000000
r: 0100000000000000000000000100000000000000000000000000000000000000
t: 1000000000000000000000000000000000000101000000000000000000000000
u: 0000000000000000000000000000000000100000000000000000000000000000
w: 0000100000000000000000000000000000000000000000000000000000000000
y: 0000000000000000000000100000000000000000000000000000000000000000
```

Run-lengthifying

Make B' by concatenating runs in alphabetical groups:

```
   trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
B  1100100000000100100100100100000000100101100000100100000000000100

$:                                         1
_:                 100                     100000               100
a:                                                 100
b:                    100
o:                100                                 100000000000
r:  100                     100000000
t: 1                                    10
u:                                     100
w:     100000000
y:                       100
```

Run-lengthifying

Make B' by concatenating runs in alphabetical groups:

```
  trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
B 1100100000000100100100100100000000100101100000100100000000000100
```

$:	1
_:	100100000100
a:	100
b:	100
o:	100100000000000
r:	100100000000
t:	110
u:	100
w:	100000000
y:	100

B'

Run-lengthifying

Make B' by concatenating runs in alphabetical groups:

```
  trrrwwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$______aaaoooooooooooo___
B 1100100000000100100100100100000000100101100000100100000000000100
```

```
    $: 1
    _: 100100000100
    a: 100
    b: 100
B'  o: 100100000000000
    r: 100100000000
    t: 110
    u: 100
    w: 100000000
    y: 100
```

Run-lengthifying

$i = 35, \; c = \texttt{o}$

B `1100100000000100100100100100000000100101100000100100000000000100`

$B\,.\,\mathrm{rank}_1(35 + 1) = 9$

S `trwo_byrut$_ao_`

$S\,.\,\mathrm{rank}_{\texttt{o}}(9 - 1) = 1$

$C[\texttt{o}] = 19$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

$B'[\texttt{o}]$ `100100000000000`

Run-lengthifying

$i = 35,\ c = \texttt{o}$

↓

B `1100100000000100100100100100000000100101100000100100000000000100`

$B.\mathrm{rank}_1(35 + 1) = 9$

S `trwo_byrut$_ao_`

$S.\mathrm{rank}_{\texttt{o}}(9 - 1) = 1$

$C[\texttt{o}] = 19$

```
C   0  1  13  16  19  34  46  49  52  61
   <$ <_  <a  <b  <o  <r  <t  <u  <w  <y
```

$B'[\texttt{o}]$ `1001000000000000`

$B'[\texttt{o}].\mathrm{select}_1(1) = 3$

$19 + 3 = 22$ ✅

Run-lengthifying

$i = 28, \; c = \mathtt{r}$

B 1100100000000100100100100100000000100101100000100100000000000100

S trwo_byrut$_ao_

C

0	1	13	16	19	34	46	49	52	61
<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

$B'[\mathtt{r}]$ 100100000000

Run-lengthifying

$i = 28, \; c = \texttt{r}$

B `1100100000000100100100100100000000100101100000100100000000000100`

$B.\mathrm{rank}_1(28 + 1) = 8$

S `trwo_byrut$_ao_`

$S.\mathrm{rank}_{\texttt{r}}(8 - 1) = 1$

$C[\texttt{r}] = 34$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

$B'[\texttt{r}]$ `100100000000`

Run-lengthifying

$i = 28, \; c = \mathsf{r}$

↓

B 1100100000000100100100100100000000100101100000100100000000000100

$B\,.\,\mathsf{rank}_1(28 + 1) = 8$

S trwo_byrut$_ao_

$S\,.\,\mathsf{rank}_\mathsf{r}(8 - 1) = 1$

$C[\mathsf{r}] = 34$

C

0	1	13	16	19	34	46	49	52	61
<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

$B'[\mathtt{r}]$ 100100000000

$B'[\mathsf{r}]\,.\,\mathsf{select}_1(1) = 3$

Run-lengthifying

$i = 28, \;\; c = \texttt{r}$

B `1100100000000100100100100100000000100101100000100100000000000100`

$B\,.\,\mathrm{rank}_1(28\; + \;1) = 8$

$B\,.\,\mathrm{select}_1(8) = 25$

S `trwo_byrut$_ao_`

$S\,.\,\mathrm{rank}_{\texttt{r}}(8\; - \;1) = 1$

$C[\texttt{r}] = 34$

C	0	1	13	16	19	34	46	49	52	61
	<$	<_	<a	<b	<o	<r	<t	<u	<w	<y

$B'[\texttt{r}]$ `100100000000`

$B'[\texttt{r}]\,.\,\mathrm{select}_1(1) = 3$

$34 + 3\; + \;(28\; - \;25) = 40$ ✅

Run-length FM Index

Given offset off and character c

FM index:

```
$r \leftarrow L.\mathrm{rank}_c(off + 1)$
$s \leftarrow C[c]$
next_off $\leftarrow s + r$
```

RLFM index:

```
$r \leftarrow B.\mathrm{rank}_1(off + 1)$
$o \leftarrow B.\mathrm{select}_1(r)$
$pv \leftarrow S.\mathrm{rank}_c(r - 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow B'[c].\mathrm{select}_1(pv)$
next_off $\leftarrow s_1 + s_2 + (off - o)$
```

Run-length FM Index

What takes **space**?

FM index:

```
$r \leftarrow L.\mathrm{rank}_c(off + 1)$
$s \leftarrow C[c]$
next_off $\leftarrow s + r$
```

RLFM index:

```
$r \leftarrow B.\mathrm{rank}_1(off + 1)$
$o \leftarrow B.\mathrm{select}_1(r)$
$pv \leftarrow S.\mathrm{rank}_c(r - 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow B'[c].\mathrm{select}_1(pv)$
next_off $\leftarrow s_1 + s_2 + (off - o)$
```

Run-length FM Index

What takes **space**?

FM index:

```
$r \leftarrow \boxed{L\,.\,\mathrm{rank}_c(\mathit{off} + 1)}$
$s \leftarrow C[c]$
$\mathrm{next\_off} \leftarrow s + r$
```

Rank on wavelet tree of L

RLFM index:

```
$r \leftarrow \boxed{B\,.\,\mathrm{rank}_1(\mathit{off} + 1)}$
$o \leftarrow \boxed{B\,.\,\mathrm{select}_1(r)}$
$pv \leftarrow \boxed{S\,.\,\mathrm{rank}_c(r - 1)}$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow \boxed{B'[c]\,.\,\mathrm{select}_1(pv)}$
$\mathrm{next\_off} \leftarrow s_1 + s_2 + (\mathit{off} - o)$
```

Rank on wavelet tree of S

Rank+select ("predecessor") on bitvector B

Select on bitvector B'

Run-length FM Index

Our rank/select strategy has been Jacobson/Clark

$n + \check{o}(n)$ bits, $O(1)$ time

Need a function of r, not n

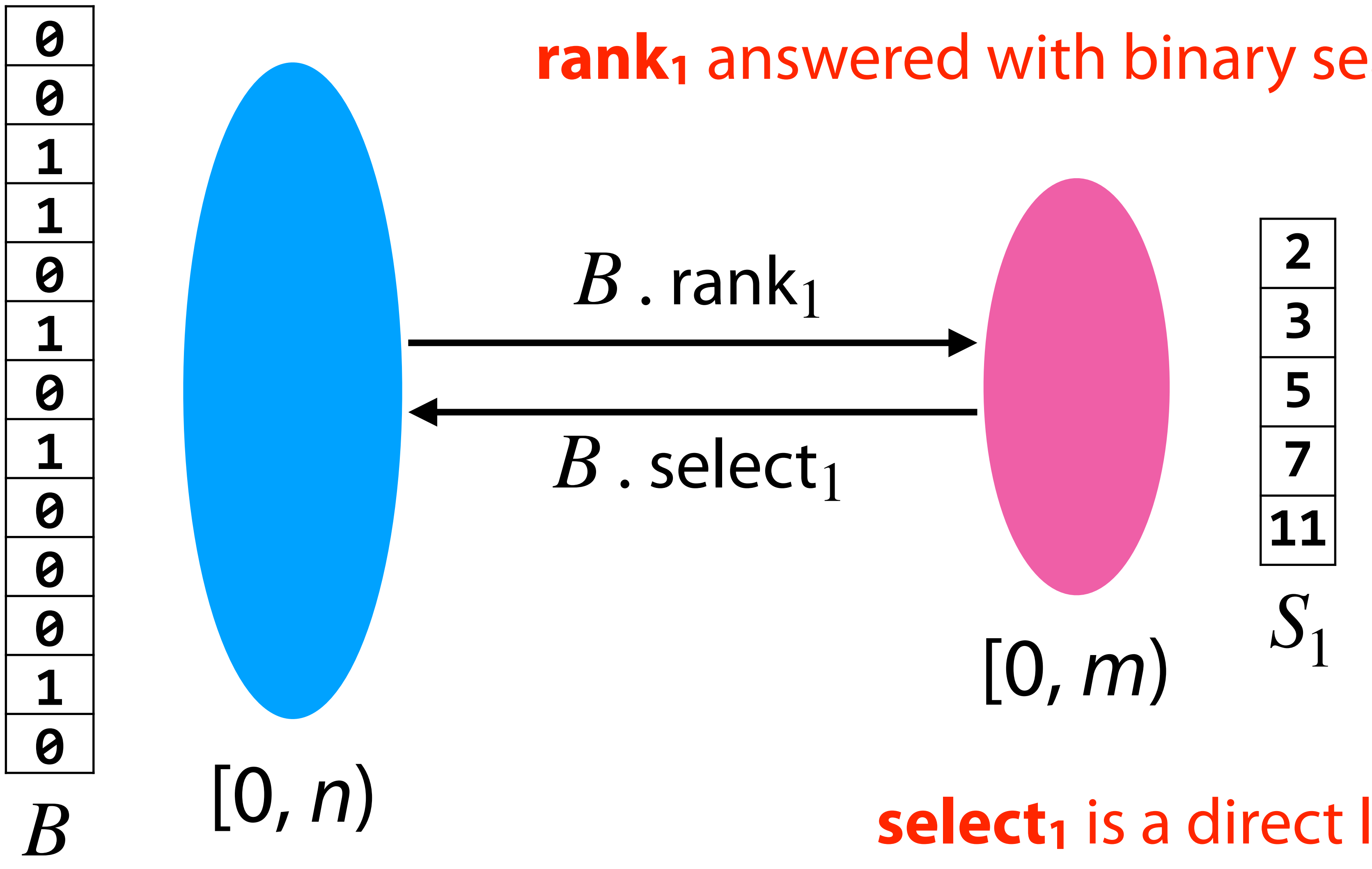
RLFM index:

```
$r \leftarrow \boxed{B\,.\,\text{rank}_1(\mathit{off} + 1)}$
$o \leftarrow \boxed{B\,.\,\text{select}_1(r)}$
$pv \leftarrow S\,.\,\text{rank}_c(r - 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow \boxed{B'[c]\,.\,\text{select}_1(pv)}$
$\text{next\_off} \leftarrow s_1 + s_2 + (\mathit{off} - o)$
```

Run-length FM Index

Recall this idea, which works well for sparse bv's

rank₁ answered with binary search on S_1



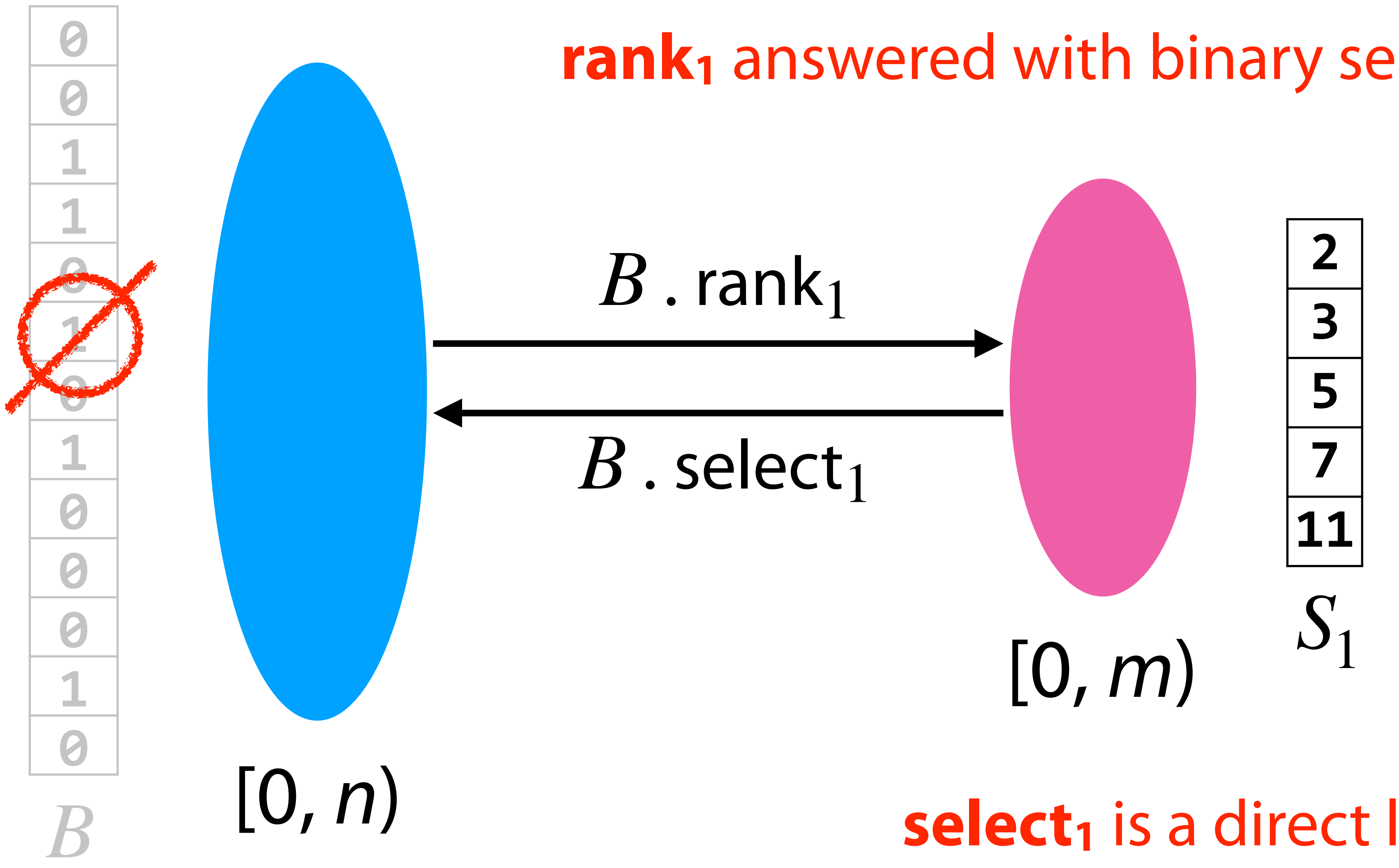
B
0
0
1
1
0
1
0
1
0
0
0
1
0

S_1
2
3
5
7
11

select₁ is a direct lookup in S_1

Run-length FM Index

Recall this idea, which works well for sparse bv's

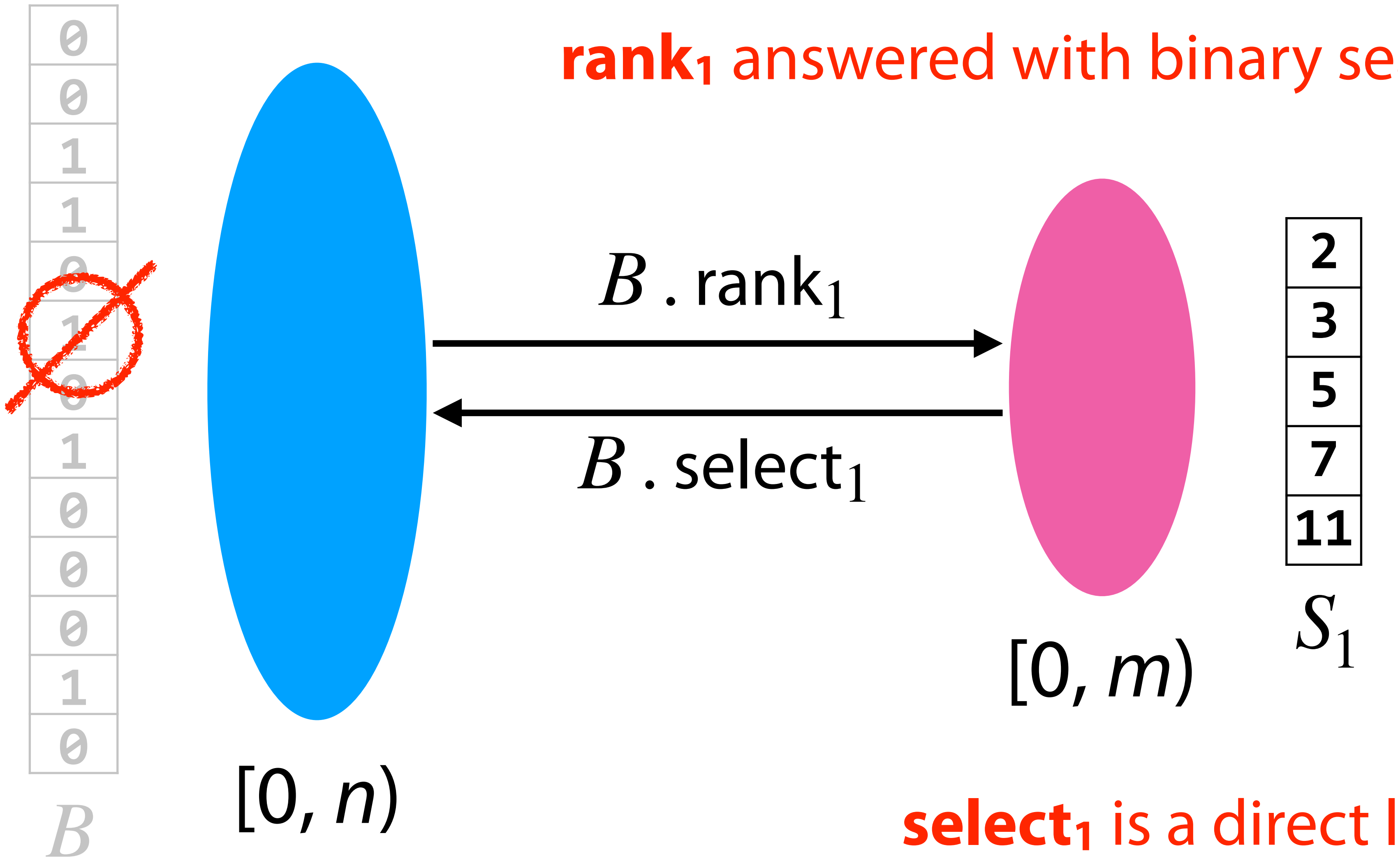


rank₁ answered with binary search on S_1

select₁ is a direct lookup in S_1

Run-length FM Index

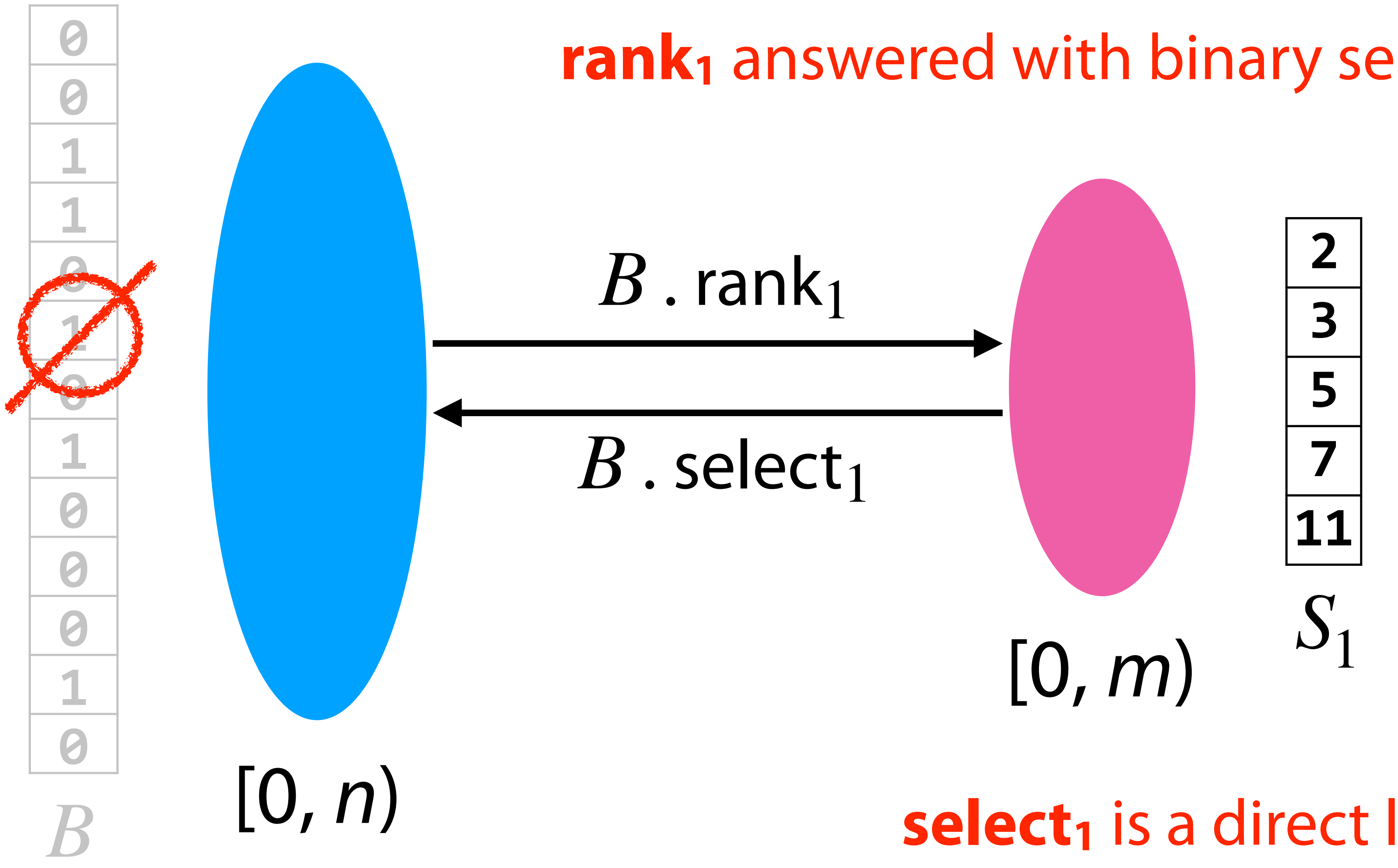
rank₁ answered with binary search on S_1



select₁ is a direct lookup in S_1

Space ——— Time

Run-length FM Index



rank₁ answered with binary search on S_1

select₁ is a direct lookup in S_1

Space

$O(m \log n)$ bits

$O(m)$ words

Time

$B\,.\,\mathrm{rank}_1$: $O(\log m)$ time

$B\,.\,\mathrm{select}_1$: $O(1)$ time

Run-length FM Index: simple structures

RLFM index:

```
$r \leftarrow B\,.\,\mathsf{rank}_1(off\ +\ 1)$
$o \leftarrow B\,.\,\mathsf{select}_1(r)$
$pv \leftarrow S\,.\,\mathsf{rank}_c(r\ -\ 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow B'[c]\,.\,\mathsf{select}_1(pv)$
$\mathsf{next\_off} \leftarrow s_1 + s_2\ +\ (off\ -\ o)$
```

Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

Belazzougui, Djamal, and Gonzalo Navarro. "Optimal lower and upper bounds for representing sequences." ACM Transactions on Algorithms (TALG) 11.4 (2015): 1-21.

Run-length FM Index: simple structures

Predecessor queries on B

Sparse bitvector: $O(\log r)$ time, $O(r)$ words of space

Rank queries on S

Balanced wavelet tree: $O(\log \sigma)$ time, $O(r \log \sigma)$ bits of space

Can do better in theory

Select queries on C & B'

$O(1)$ time, $O(r)$ space
Array of cumulative counts

RLFM index:

```
$r \leftarrow B\,.\,\text{rank}_1(off + 1)$
$o \leftarrow B\,.\,\text{select}_1(r)$
$pv \leftarrow S\,.\,\text{rank}_c(r - 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow B'[c]\,.\,\text{select}_1(pv)$
next_off $\leftarrow s_1 + s_2 + (off - o)$
```

Run-length FM Index: advanced structures

Predecessor queries on B

$O(\log\log n/r)$ time, $O(r)$ space

Belazzougui & Navarro, Thm A.1

Rank queries on S

$O(\log\log \sigma)$ time, $O(r)$ space

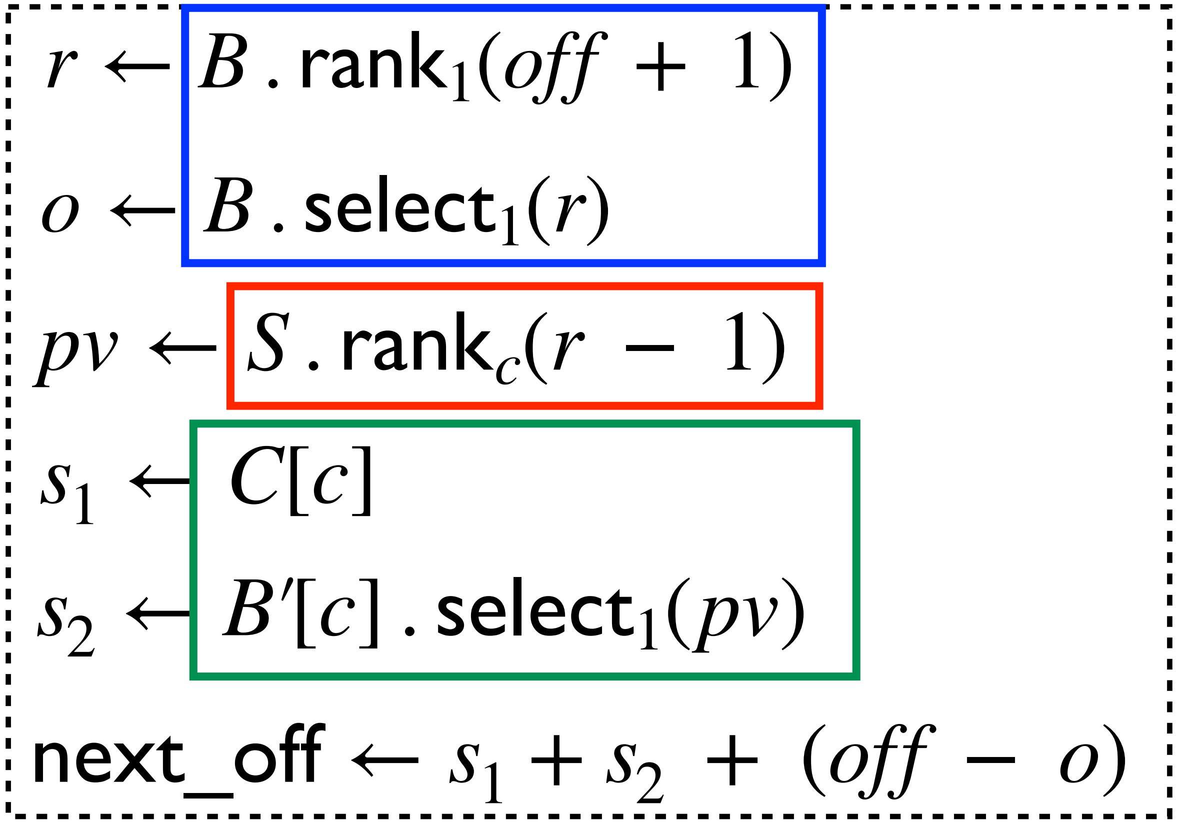
Belazzougui & Navarro, Thm 5.4

Select queries on C & B'

$O(1)$ time, $O(r)$ space

Array of cumulative counts

RLFM index:



```
$r \leftarrow B.\text{rank}_1(off + 1)$
$o \leftarrow B.\text{select}_1(r)$
$pv \leftarrow S.\text{rank}_c(r - 1)$
$s_1 \leftarrow C[c]$
$s_2 \leftarrow B'[c].\text{select}_1(pv)$
next_off $\leftarrow s_1 + s_2 + (off - o)$
```

Gagie T, Navarro G, and Prezza P. Optimal-time text indexing in BWT-runs bounded space. Proceedings of 29th SODA, ACM-SIAM. 2018. pp1459—1477.

Belazzougui, Djamal, and Gonzalo Navarro. "Optimal lower and upper bounds for representing sequences." ACM Transactions on Algorithms (TALG) 11.4 (2015): 1-21.

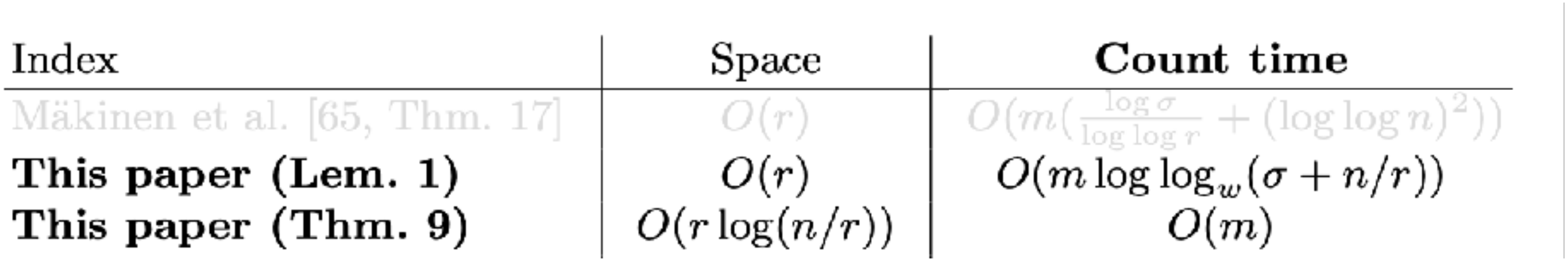
Run-length FM Index

	Count		Locate	
	Space	Time	Space	Time
FM Index (2000)	$O(n)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
RLFM Index (2005)	$O(r)$	$O(m)$	$O(n)$	$O(m + \text{occ})$
r-index (2018)	$O(r)$	$O(m)$	$O(r)$	$O(m + \text{occ})$

n = reference length, m = query length, r = # BWT runs

RLFM: Mäkinen V, and Navarro G. Succinct suffix arrays based on run-length encoding. Annual Symposium on CPM. Springer, Berlin, Heidelberg. 2005. pp45–56.

Run-length FM Index



Index	Space	**Count time**
Mäkinen et al. [65, Thm. 17]	$O(r)$	$O(m(\frac{\log\sigma}{\log\log r}+(\log\log n)^2))$
This paper (Lem. 1)	$O(r)$	$O(m\log\log_w(\sigma+n/r))$
This paper (Thm. 9)	$O(r\log(n/r))$	$O(m)$

Gagie, Travis, Gonzalo Navarro, and Nicola Prezza. "Optimal-Time Text Indexing in BWT-runs Bounded Space." arXiv preprint arXiv:1705.10382 (2017).

Question left open: Are $O(r)$ and $O(m)$ achievable at the same time?

Summary

When L is "runny," we can represent it with a few structures -- B, S, B' -- that altogether take $O(r)$ space

In practice, we might compromise either on the strict $O(r)$ space or the strict $O(m)$ count query time

Next step, Locate queries...