# Space-Efficient Data Structures

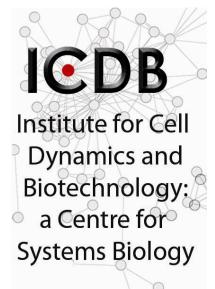
### Francisco Claude Gonzalo Navarro











### Outline

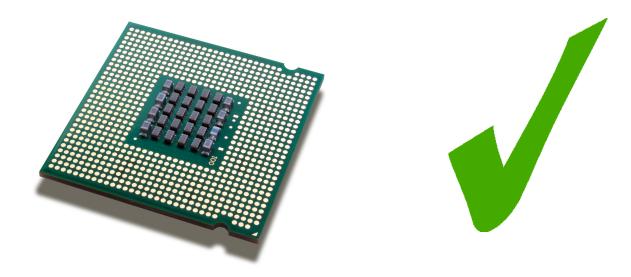
- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

### Outline

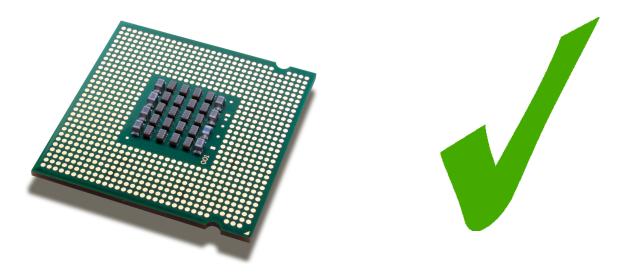
Motivation

- Basics
- Bitmaps
- Sequences
- Applications

• Processor speed increasing



• Processor speed increasing

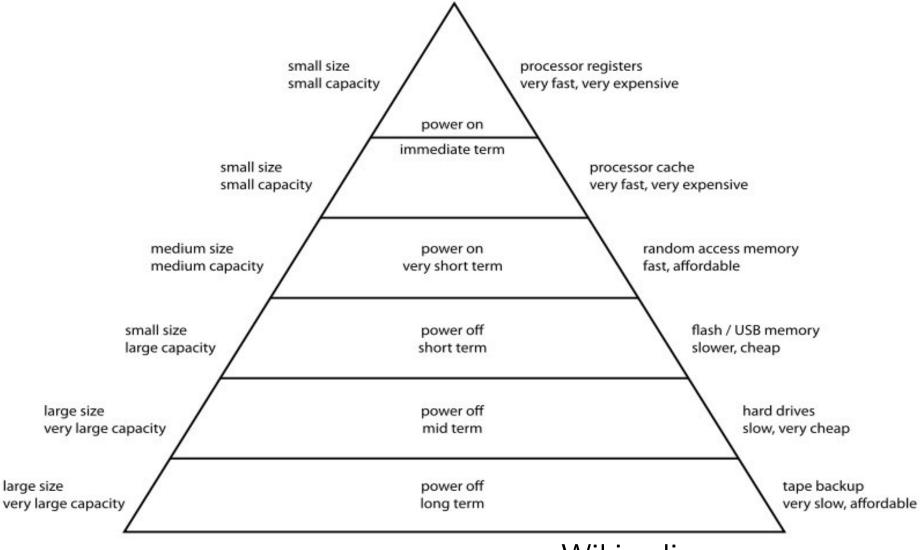


• Disks have not evolved at the same pace



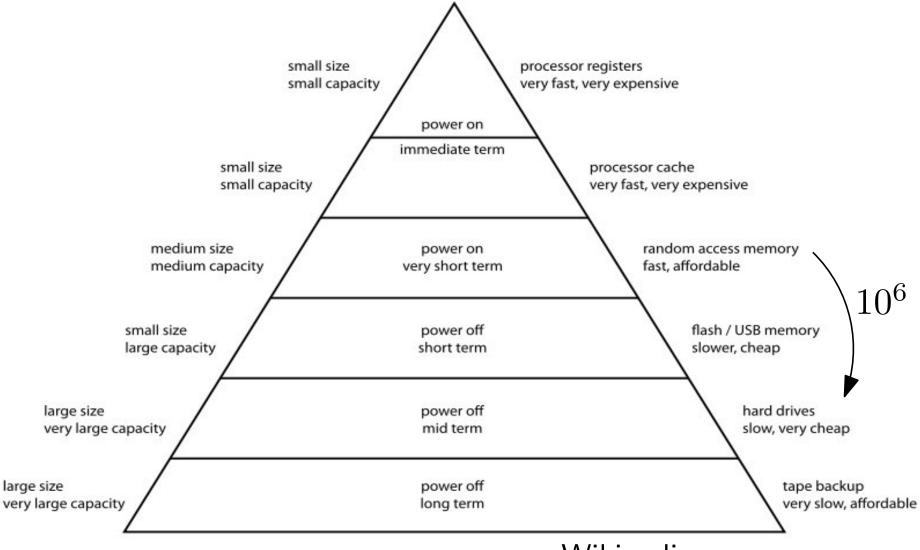


#### Computer Memory Hierarchy



source: Wikipedia

#### Computer Memory Hierarchy



source: Wikipedia

Web Graph uk-union-2006-06-2007-05

Nodes: 133, 633, 040

Edges: 5,507,679,822

Web Graph uk-union-2006-06-2007-05

Nodes: 133, 633, 040

Edges: 5,507,679,822

A plain representation requires 22GB!

Web Graph uk-union-2006-06-2007-05

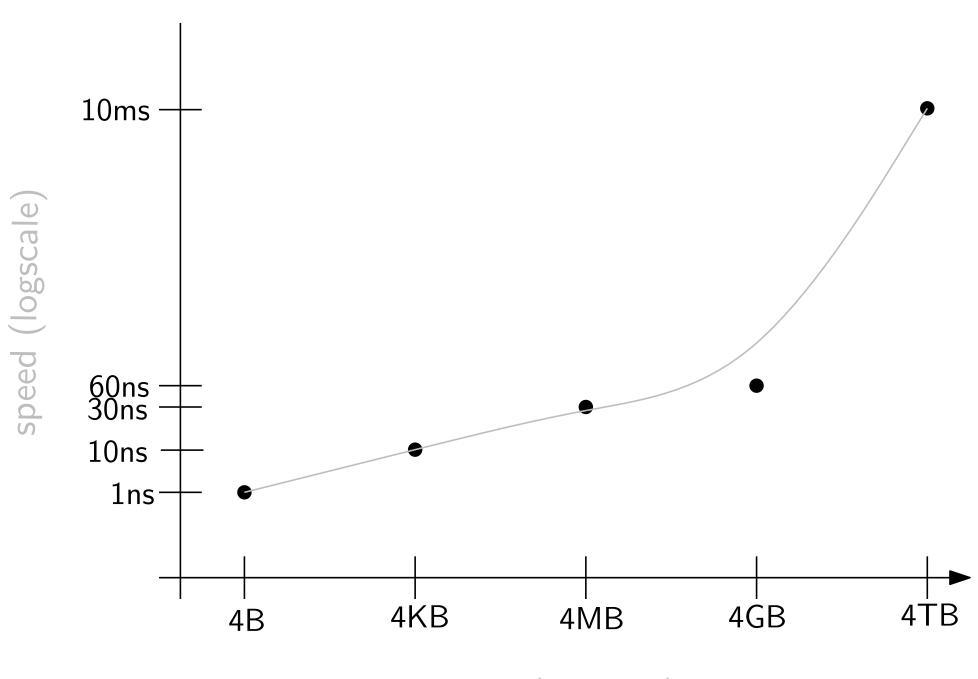
Nodes: 133, 633, 040

Edges: 5, 507, 679, 822

A plain representation requires 22GB!

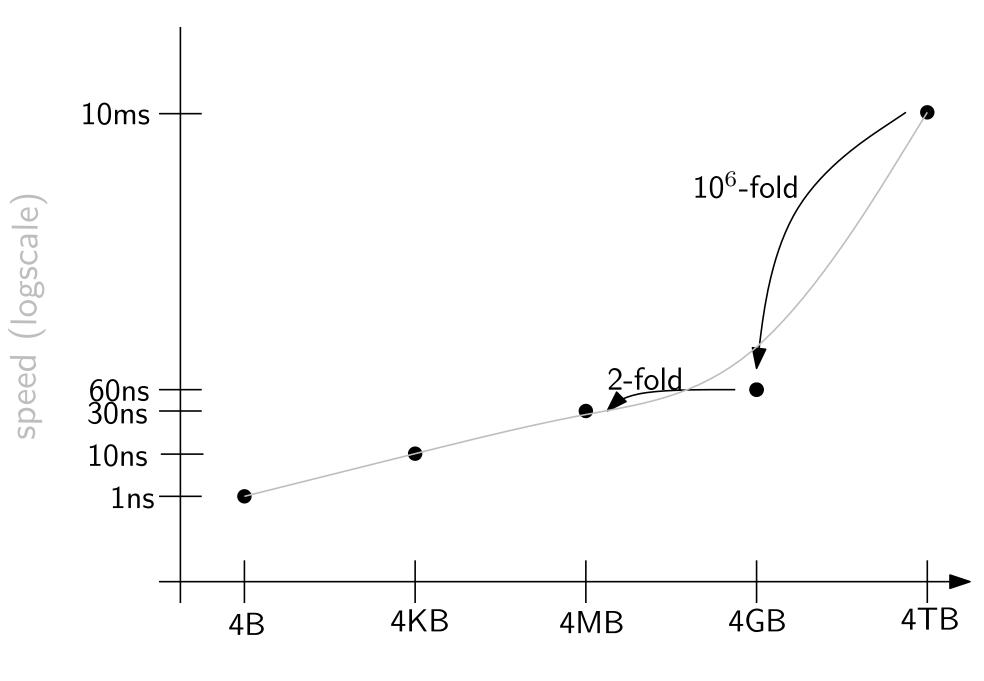
If we use a space-efficient representation: < 2GB





memory (logscale)





memory (logscale)

### Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

### Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

#### **Basics**

- Plain representation of data
- Zero-order compression
- High-order compression

### Plain Representation of Data

#### Array:

- ullet length n
- ullet maximum value m

$$n \lceil \log_2(m+1) \rceil$$
 bits

### Plain Representation of Data

#### Array:

- length n
- $\bullet$  maximum value m

$$n \lceil \log_2(m+1) \rceil$$
 bits

| 5   | 3   | 4   | 1   | 0   | 4   | 2   | 4   | 1   | 0   |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 101 | 011 | 100 | 001 | 000 | 100 | 010 | 100 | 001 | 000 |

On a 32-bits machine this requires 1 word

### Arrays in LIBCDS

```
size_t N;
cout << "Enter array length: ";</pre>
cin >> N;
uint * A = new uint[N];
for(size_t i=0;i<N;i++) {
  cout << "Enter element at position " << i << ": ";</pre>
 cin >> A[i];
Array * a = new Array(A,N);
delete [] A;
cout << "Size: " << a->getSize() << " bytes" << endl;</pre>
for(uint i=0;i<N;i++)</pre>
  cout << "A[" << i << "]=" << a->getField(i) << endl;
delete a;
```

#### Arrays in LIBCDS

```
size_t N;
uint M;
cout << "Enter array length: ";</pre>
cin >> N;
cout << "Enter the maximum value to be stored: ";
cin >> M;
Array *a = new Array(N,M);
for(size_t i=0;i<N;i++) {</pre>
  uint tmp;
  cout << "Enter element at position " << i << ": ";</pre>
 cin >> tmp;
  a->setField(i,tmp);
cout << "Size: " << a->getSize() << " bytes" << endl;</pre>
```

```
for(uint i=0;i<N;i++)
  cout << "A[" << i << "]=" << a->getSize() << bytes << end;

for(uint i=0;i<N;i++)
  cout << "A[" << i << "]=" << a->getField(i) << end];

delete a;</pre>
```

### Zero-order Compression

Can we do better? It depends

S = aaabbcaaabbcaaad

$$H_0(S) = \sum_{c \in \Sigma} \frac{n_c}{n} \log_2 \frac{n}{n_c}$$

$$H_0(S) = 1.5919$$

| symbol | $n_{symbol}$ |
|--------|--------------|
| а      | 9            |
| b      | 4            |
| С      | 2            |
| d      | 1            |

### Zero-order Compression

#### Can we do better? It depends

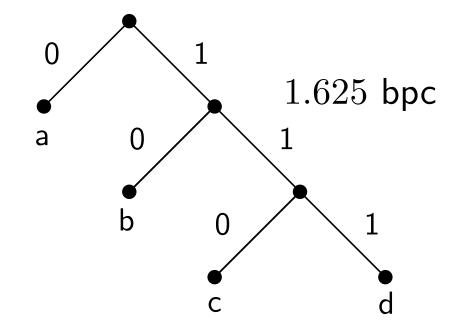
S = aaabbcaaabbcaaad

$$H_0(S) = \sum_{c \in \Sigma} \frac{n_c}{n} \log_2 \frac{n}{n_c}$$

$$H_0(S) = 1.5919$$

| symbol | $n_{symbol}$ |
|--------|--------------|
| а      | 9            |
| b      | 4            |
| С      | 2            |
| d      | 1            |

Huffman



### High-order Compression

Can we exploit other things?

S = aaabbcaaabbcaaad

### High-order Compression

Can we exploit other things?

S = aaabbcaaabbcaaad

Yes, for example P(a|b) = 0

$$H_k(S) = \frac{1}{n} \sum_{A \in \Sigma^k} |T_A| H_0(T_A)$$

$$H_1(S) = 0.9387$$

### Things to Remember

- $\lceil \log_2(m+1) \rceil$  bits to represent a number  $\leq m$
- ullet Compression:  $H_0$  and  $H_k$

$$H_k \le H_0 \le \log_2 m$$

### Things to Remember

- $\lceil \log_2(m+1) \rceil$  bits to represent a number  $\leq m$
- ullet Compression:  $H_0$  and  $H_k$

$$H_k \le H_0 \le \log_2 m$$

One step forward: ordinal trees

$$C_n = \frac{1}{n+1} \binom{2n}{n} \approx \frac{4^n}{n^{3/2} \sqrt{\pi}}$$

$$\Rightarrow 2n + o(n)$$
 bits

### Outline

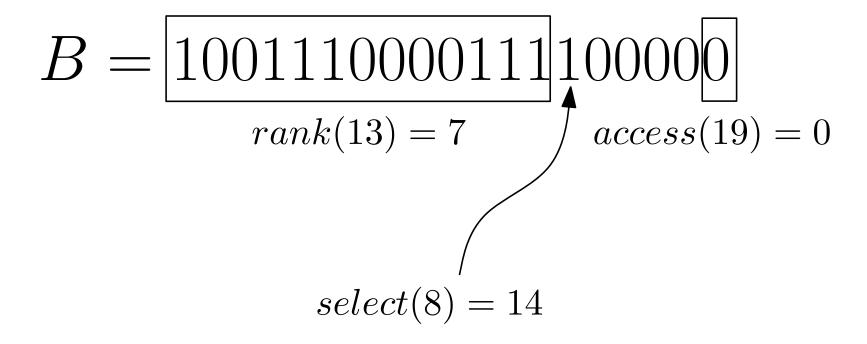
- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

### Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

$$B = 1001110000111100000$$

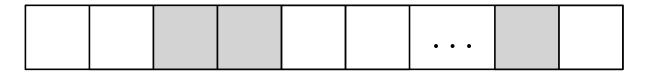
$$B = 1001110000111100000$$
 $rank(13) = 7$ 
 $access(19) = 0$ 
 $select(8) = 14$ 



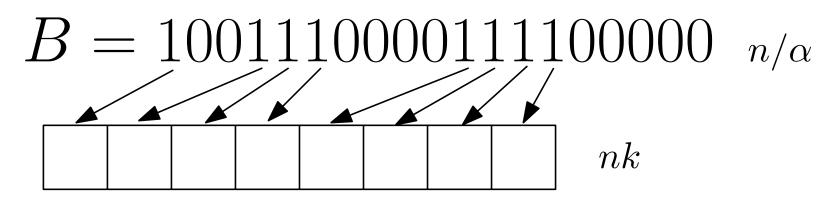
Is this useful?

### Hashing Example

- ullet Static hashtable with n elements implemented with linear probing
- Expected successful seach cost = 3 ( $\alpha = 0.8$ )
- ullet Each key requires k bits

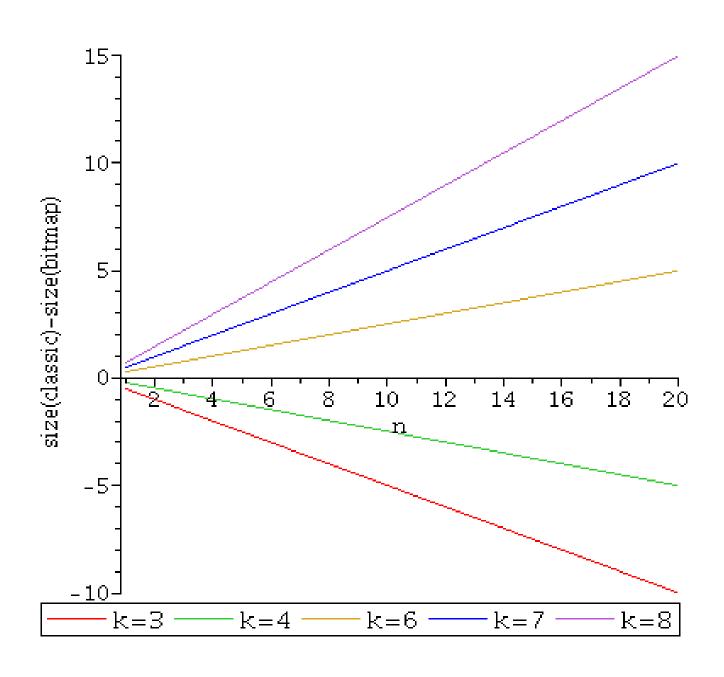


Standard solution requires  $k\frac{n}{\alpha}$  bits



Requires  $\frac{n}{\alpha} + nk$  bits

# Hashing Example



#### Bitmaps: Example Applications

- Linear-probing hashing
- Perfect hashing
- Partial Sums
- Bitmaps are basic building blocks

- Plain [Jacobson, Clark and Munro]
- Compressed [Raman et al.]
- Very Sparse [Okanohara and Sadakane]

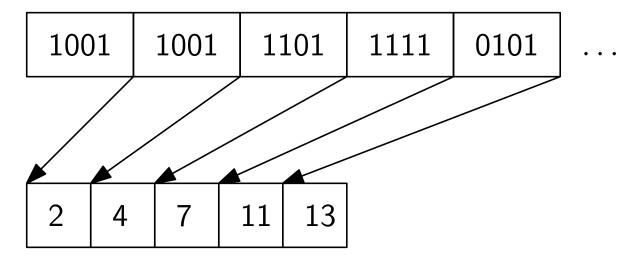
Rank

B = 10010101011110101011...

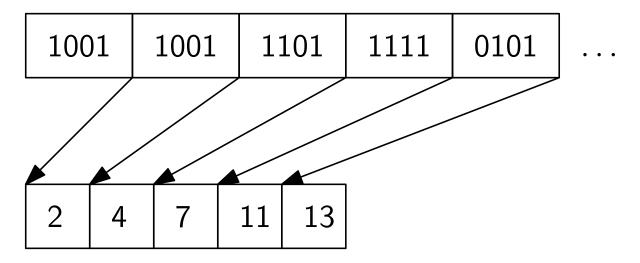
1 1 1 2 2 3 3 4 4 5 6 7 7 ...

O(1) rank  $n \lg n$  bits of space

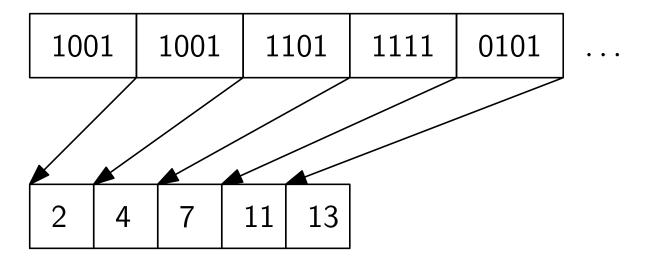
#### Rank



If we sample every s bits, we require  $\frac{n \lg n}{s}$  bits. Rank takes O(s) time.



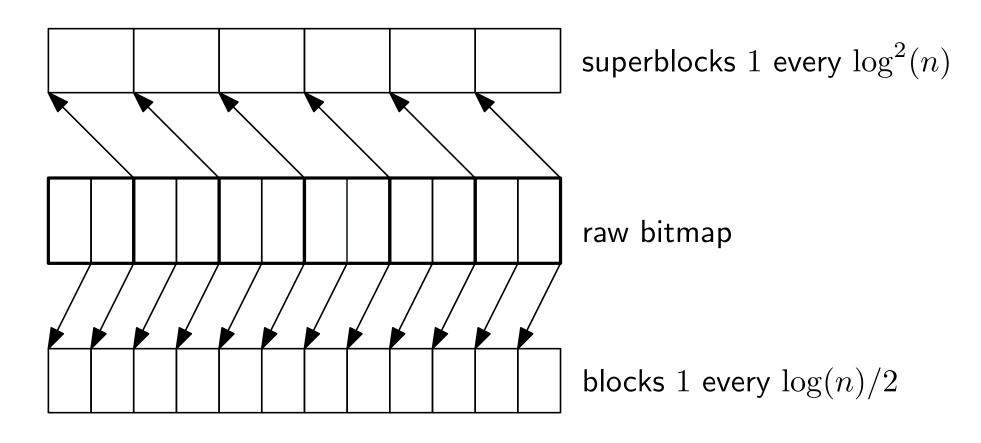
If we sample every s bits, we require  $\frac{n \lg n}{s}$  bits. Rank takes O(s) time.

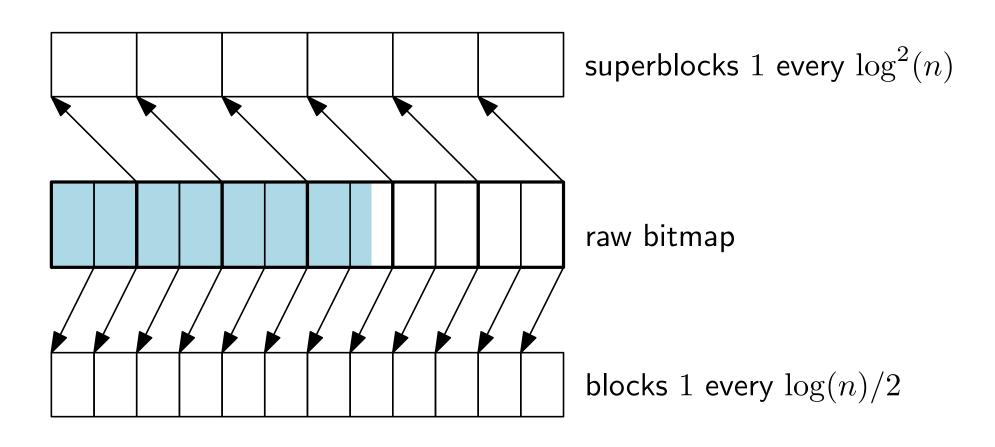


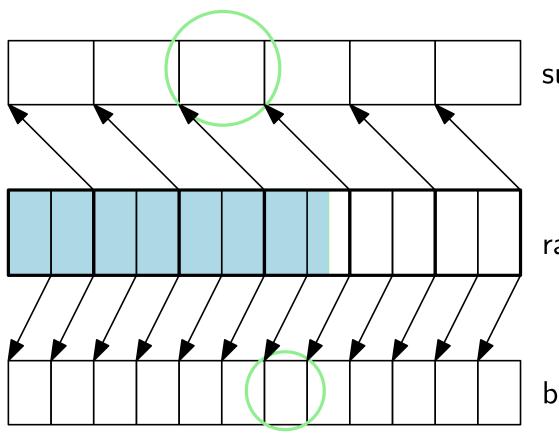
If we sample every s bits, we require  $\frac{n \lg n}{s}$  bits. Rank takes O(s) time.

| 1001 | 1001 |
|------|------|
|------|------|

Recurse inside blocks sampling every b. This requires  $\frac{n \lg s}{b}$  bits and now we can answer in O(b) time.







superblocks 1 every  $\log^2(n)$ 

raw bitmap

blocks 1 every  $\log(n)/2$ 

overall space:  $n + n/\log n + \frac{2n}{\log n}\log\log n$  bits

So far we have n+o(n) bits and we can answer in  $O(\log(n))$  time. The nice thing is that we can handle blocks of size  $\log(n)/2$  in constant time.

#### Idea:

List all blocks of size  $\log(n)/2$  and store the rank up to any position in a table for lookup.

So far we have n+o(n) bits and we can answer in  $O(\log(n))$  time. The nice thing is that we can handle blocks of size  $\log(n)/2$  in constant time.

#### Idea:

List all blocks of size  $\log(n)/2$  and store the rank up to any position in a table for lookup.

#### Space required:

 $2^{\log(n)/2}\log(n)\log\log(n)/2 \approx \sqrt{n}\log(n)\log\log(n)$  bits. This is o(n) bits.

#### **Universal Tables**

Example with size 8.

| 0000000      | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|--------------|---|---|---|---|---|---|---|---|
| 0000001      | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0000010      | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 01100101     | 0 | 1 | 2 | 2 | 2 | 3 | 3 | 4 |
| <br>11111111 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Size in practice:

| $\log(n)/2$ | size |
|-------------|------|
| 8           | 2KB  |
| 16          | 1MB  |

#### **Universal Tables**

Example with size 8.

| 00000000     | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|--------------|---|---|---|---|---|---|---|---|
| 0000001      | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0000010      | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| <br>01100101 | 0 | 1 | 2 | 2 | 2 | 3 | 3 | 4 |
| <br>11111111 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Size in practice:

| $\log(n)/2$ | size |
|-------------|------|
| 8           | 2KB  |
| 16          | 1MB  |

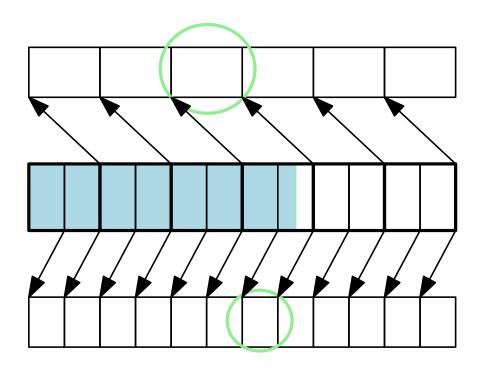
In practice we use popcnt

Superblocks:  $O(n/\log n)$  bits

Blocks:  $O(n \log \log n / \log n)$  bits

Table:  $O(\sqrt{n} \log n \log \log n)$  bits

3 steps answer rank



Partition according to the number of 1s

$$\mathsf{B} = \underbrace{1010010100100110011001001}_{\log^2 n \ 1\mathsf{s}}$$

Partition according to the number of 1s

$$\mathsf{B} = \underbrace{1010010100100110011001001001}_{\log^2 n \text{ 1s}}$$

Sparse superblock: length  $\geq \log^3 n \cdot \log \log n$ 

We store the answer for all sparse superblocks in plain form

Space required:  $O(n/\log\log n) + n/\log^2 n + o(n)$  bits

Partition according to the number of 1s

$$\mathsf{B} = \underbrace{1010010100100110011001001001}_{\log^2 n \ 1\mathsf{s}}$$

Store the positions where each dense superblock begins

Divide every superblock into blocks of  $(\log \log n)^2 1s$ 

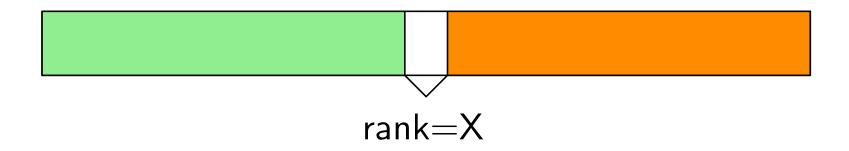
We consider a block sparse if the size is  $\geq 4(\log \log n)^4$  $\Rightarrow O(n/\log n)$  bits for storing the answers

We recurse again! The next level is small enough.

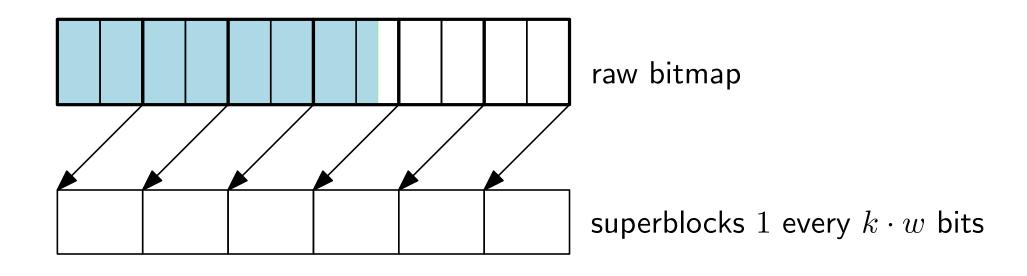
The solution is quite complicated and does not work well in practice

The solution is quite complicated and does not work well in practice

Practical solution: binary search using rank



#### Implementation in LIBCDS



k popcounts for rank

select requires a binary search over the superblocks + sequential search

#### Usage in LIBCDS

```
size_t N;
cout << "Length of the bitmap: ";</pre>
cin >> N;
uint * bs = new uint[uint_len(N,1)];
for(uint i=0;i<N;i++) {
  uint b;
  cout << "bit at position " << i << ": ";</pre>
  cin >> b;
  if(b==0) bitclean(bs,i);
  else bitset(bs,i);
BitSequenceRG * bsrg = new BitSequenceRG(bs,N,20);
cout << "rank(" << N/2 << ")=" << bsrg->rank1(N/2) << endl;</pre>
cout << "select(1) = " << bsrg->select1(1) << endl;</pre>
cout << "size = " << bsrg->getSize() << endl;</pre>
delete bsrg;
```

$$\mathsf{B} = 101001010010010110010101000$$

| Class | Bitmap | Offset |  |  |
|-------|--------|--------|--|--|
| 0     | 000    | 0      |  |  |
| 1     | 001    | 0      |  |  |
|       | 010    | 1      |  |  |
|       | 100    | 2      |  |  |
| 2     | 011    | 0      |  |  |
|       | 101    | 1      |  |  |
|       | 110    | 2      |  |  |
| 3     | 111    | 0      |  |  |

$$\mathsf{B} = \boxed{101001010010010110010101000}$$

| Class | Bitmap | Offset |   |   |      |   |   |   |   |   |   |   |
|-------|--------|--------|---|---|------|---|---|---|---|---|---|---|
| 0     | 000    | 0      | • |   |      |   |   |   |   |   |   |   |
| 1     | 001    | 0      |   |   |      |   |   |   |   |   |   |   |
|       | 010    | 1      |   |   |      |   |   |   |   |   |   |   |
|       | 100    | 2      |   |   |      |   |   |   |   |   |   |   |
| 2     | 011    | 0      |   |   | l _a |   |   |   |   |   |   |   |
|       | 101    | 1      |   | 2 | 1    | 1 | 1 | 1 | 2 | 1 | 2 | 0 |
|       | 110    | 2      | 0 | 1 | 0    | 1 | 1 | 1 | 2 | 1 | 1 | 0 |
| 3     | 111    | 0      |   |   |      |   |   |   |   |   |   |   |

Blocks of size  $b = \log(n)/2$ 

C requires  $2n \log \log n / \log n = o(n)$  bits

0?

Blocks of size  $b = \log(n)/2$ 

C requires  $2n \log \log n / \log n = o(n)$  bits

0?

We can represent an element of class  $c_i$  with  $\lceil \log {b \choose c_i} \rceil$  bits

Total space:  $\sum_{i=0}^{n/b} \lceil \log {b \choose c_i} \rceil \le nH_0(B) + O(n/\log n)$  bits



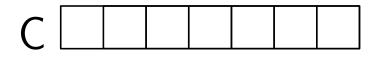
$$\sum_{i=1}^{n/b} \left\lceil \log \binom{b}{c_i} \right\rceil \leq \sum_{i=1}^{n/b} \log \binom{b}{c_i} + n/b$$

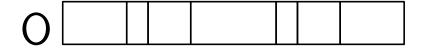
$$= \log \prod_{i=1}^{n/b} \binom{b}{c_i} + O(n/\log n)$$

$$\leq \log \binom{(n/b)b}{\sum_{i=1}^{n/b} c_i} + O(n/\log n)$$

$$= \log \binom{n}{m} + O(n/\log n)$$

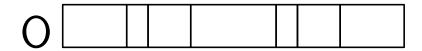
$$\leq nH_0(B) + O(n/\log n)$$





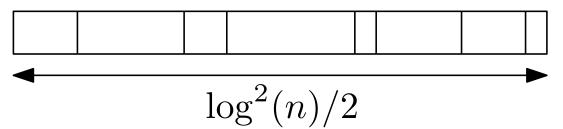
How can we support constant time access?



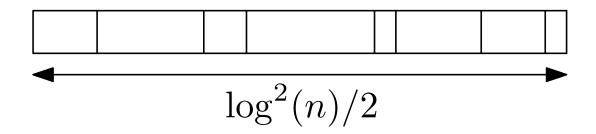


How can we support constant time access?

Store a pointer to O every log(n) blocks



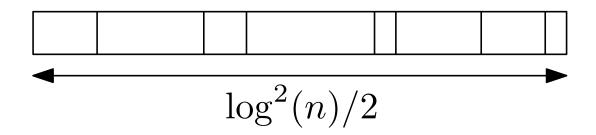
 $O(n/\log n)$  extra bits



Store offset for each element

One element is at most  $O(\log \log n)$  bits

Sum offsets:  $O(n \log \log n / \log n)$  bits



Store offset for each element

One element is at most  $O(\log \log n)$  bits

Sum offsets:  $O(n \log \log n / \log n)$  bits

Same idea behind rank

|   | Class | Bitmap | Offset |                                |
|---|-------|--------|--------|--------------------------------|
| _ | 0     | 000    | 0      |                                |
|   | 1     | 001    | 0      |                                |
|   |       | 010    | 1      | Space required for this table? |
|   |       | 100    | 2      | •                              |
|   | 2     | 011    | 0      |                                |
|   |       | 101    | 1      |                                |
|   |       | 110    | 2      |                                |
|   | 3     | 111    | 0      |                                |

| Class | Bitmap | Offset |                                |
|-------|--------|--------|--------------------------------|
| 0     | 000    | 0      |                                |
| 1     | 001    | 0      |                                |
|       | 010    | 1      | Space required for this table? |
|       | 100    | 2      | •                              |
| 2     | 011    | 0      |                                |
|       | 101    | 1      |                                |
|       | 110    | 2      |                                |
| 3     | 111    | 0      |                                |

$$b2^b + b^2 = O(\sqrt{n}\log n)$$
 bits

We can access the bitmap in constant time

What about rank and select?

We can access the bitmap in constant time

What about rank and select?

We can access  $b = \log(n)/2$  bits at the time

This replaces the plain representation for the solutions already shown

Constant time rank, select and access within  $nH_0(B) + o(n)$  bits

Practical Implementation

The table and C are easy

O and its sampling?

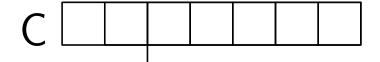
Practical Implementation

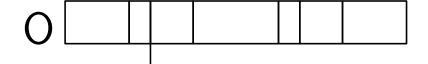
The table and C are easy

O and its sampling?

We only keep superblocks (parameter)

Superblocks are traversed linearly





```
size_t N;
cout << "Length of the bitmap: ";</pre>
cin >> N;
uint * bs = new uint[uint_len(N,1)];
for(uint i=0;i<N;i++) {
  uint b;
  cout << "bit at position " << i << ": ";</pre>
  cin >> b;
  if(b==0) bitclean(bs,i);
  else bitset(bs,i);
BitSequenceRRR * bsrrr = new BitSequenceRRR(bs,N,16);
cout << "rank(" << N/2 << ")=" << bsrrr->rank1(N/2) << endl;
cout << "select(1) = " << bsrrr->select1(1) << endl;</pre>
cout << "size = " << bsrrr->getSize() << endl;</pre>
delete bsrrr;
```

## Very Sparse Bitmaps

The previous solution does not work well for very sparse bitmaps

store S[i] = select(B, i) and solve rank with binary search

## Very Sparse Bitmaps

The previous solution does not work well for very sparse bitmaps

store S[i] = select(B, i) and solve rank with binary search

$$S[i]$$

$$h_i \qquad l_i$$

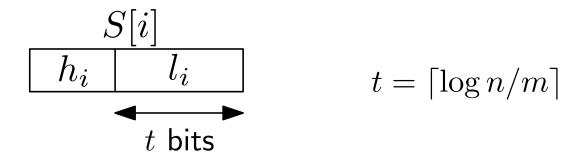
$$t \text{ bits}$$

$$t = \lceil \log n/m \rceil$$

Two arrays: H and L

H is stored in a bitmap of length 2m L is stored in  $m \log n/m + O(m)$  bits

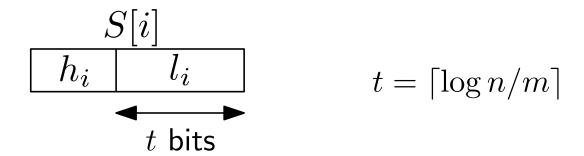
# Very Sparse Bitmaps



Two arrays: H and L

| $h_1$ | $h_2$ |  | $h_m$ |
|-------|-------|--|-------|
|-------|-------|--|-------|

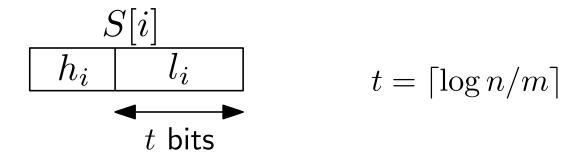
Positions  $h_i + i$  are ones



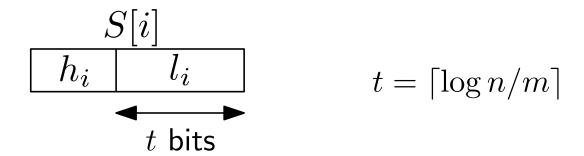
Two arrays: H and L

Positions  $h_i + i$  are ones

$$h_i = select(H, i) - i$$
  
$$h_m + m \le n/2^t + m \le 2m$$

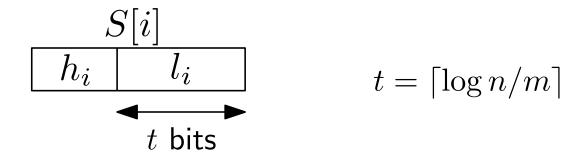


Two arrays: H and L



Two arrays: H and L

$$select(B, i) = (select(H, i) - i) \cdot 2^t + L[i]$$

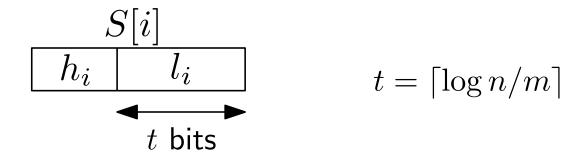


Two arrays: H and L

$$select(B, i) = (select(H, i) - i) \cdot 2^t + L[i]$$

rank:

$$\begin{split} i &= h \cdot 2^t + l \\ x &= 1 + rank(H, select_0(H, h)) \\ y &= rank(H, select_0(H, h+1)) \\ \text{binary search } L[x, y] \end{split}$$



Two arrays: H and L

Space: O(m) for H and  $m \log n/m$  for L

select: O(1)

rank:  $O(\log n/m)$ 

access:  $O(\log n/m)$ 

```
size_t N;
cout << "Length of the bitmap: ";</pre>
cin >> N;
uint * bs = new uint[uint_len(N,1)];
for(uint i=0;i<N;i++) {
  uint b;
  cout << "bit at position " << i << ": ";</pre>
  cin >> b;
  if(b==0) bitclean(bs,i);
  else bitset(bs,i);
BitSequenceSDArray * bss = new BitSequenceSDArray(bs,N);
cout << "rank(" << N/2 << ")=" << bss->rank1(N/2) << endl;
cout << "select(1) = " << bss->select1(1) << endl;</pre>
cout << "size = " << bss->getSize() << endl;</pre>
delete bss:
```

# Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

# Outline

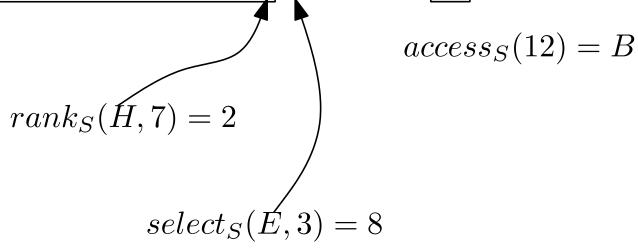
- Motivation
- Basics
- Bitmaps
- Sequences -
- Applications

# Sequences

# S = EHDHACEEGBCBGCF

# Sequences





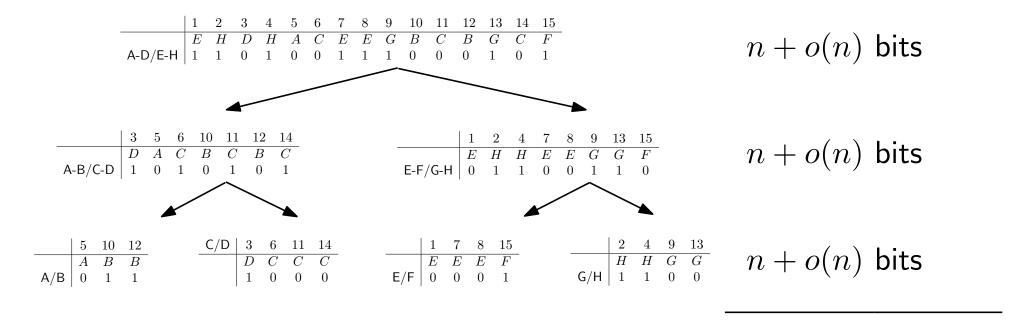
$$n = |S|$$

$$\sigma = |\Sigma|$$

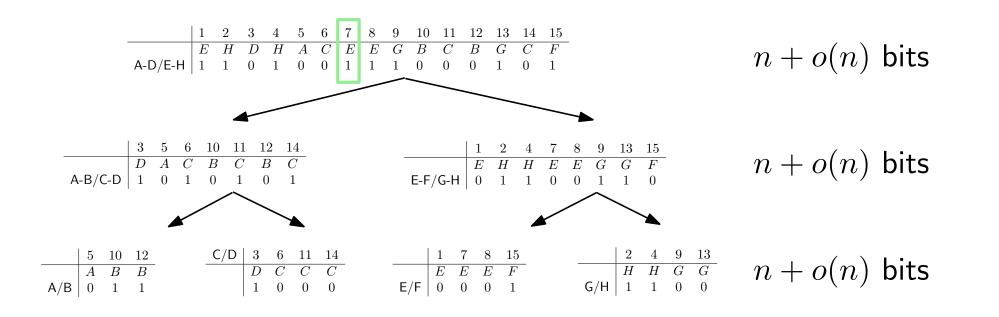
Space:  $n\lceil \log \sigma \rceil$  bits

# Sequences

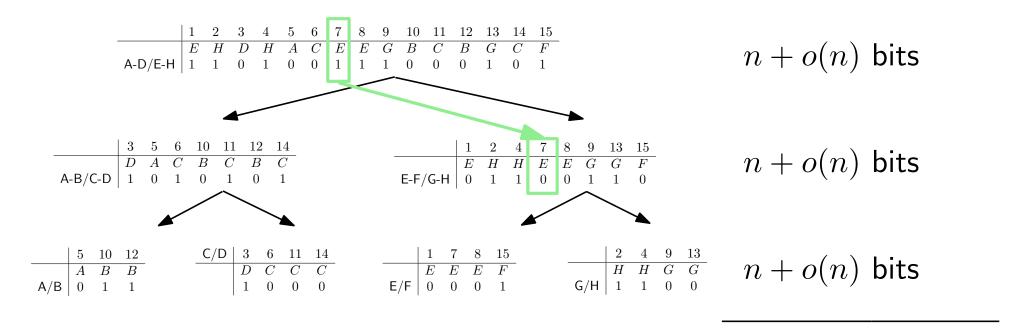
- Wavelet Trees [Grossi et al.]
- GMR [Golynski et al.]
- Alphabet Partitioning [Barbay et al.]



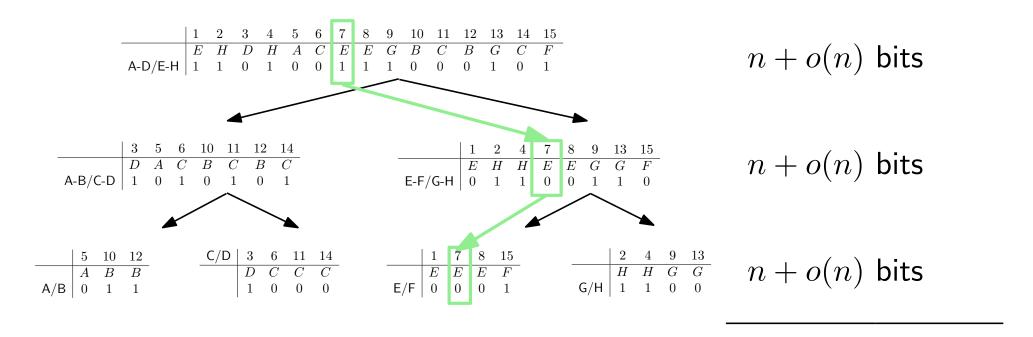
$$n\lceil\log\sigma\rceil(1+o(1))$$



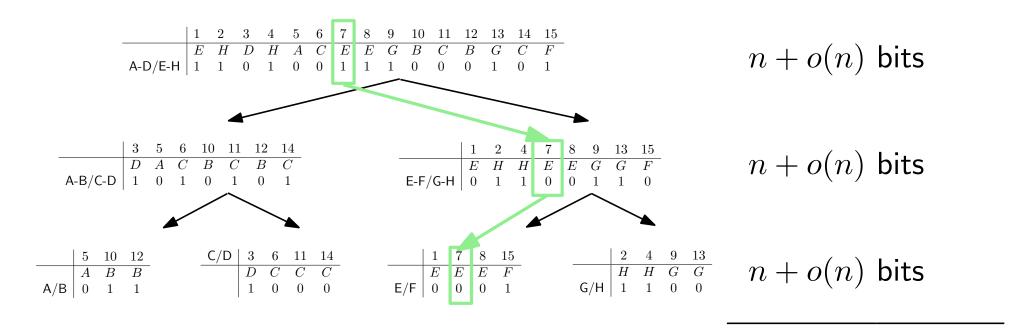
$$n\lceil \log \sigma \rceil (1 + o(1))$$



$$n\lceil \log \sigma \rceil (1 + o(1))$$

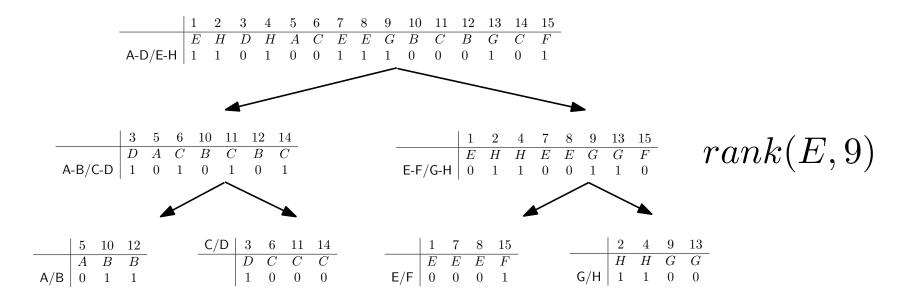


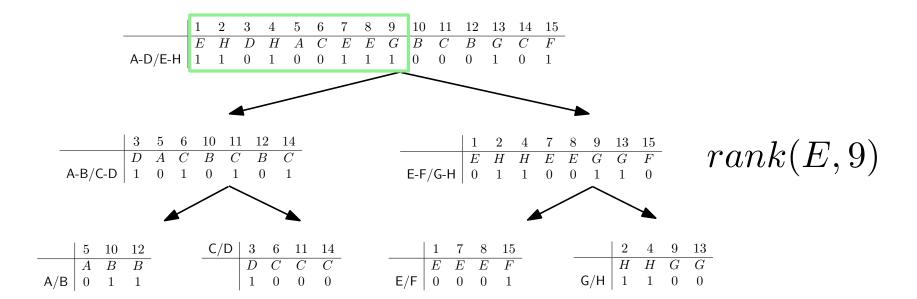
$$n\lceil \log \sigma \rceil (1 + o(1))$$

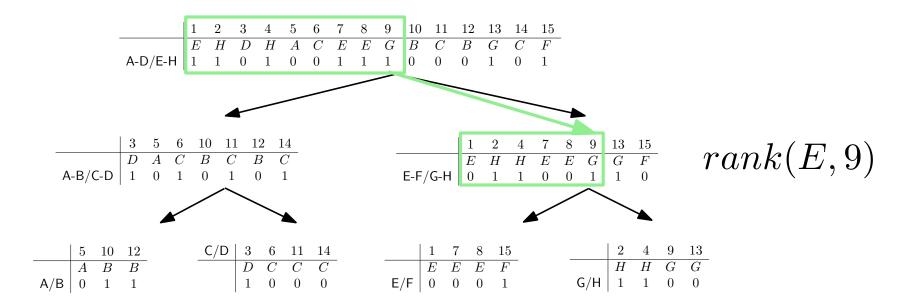


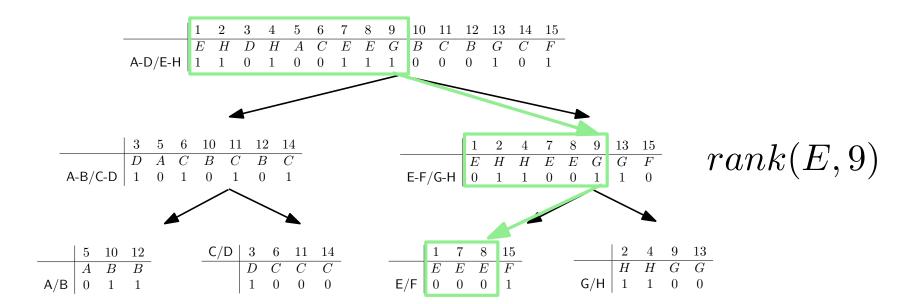
$$n\lceil \log \sigma \rceil (1 + o(1))$$

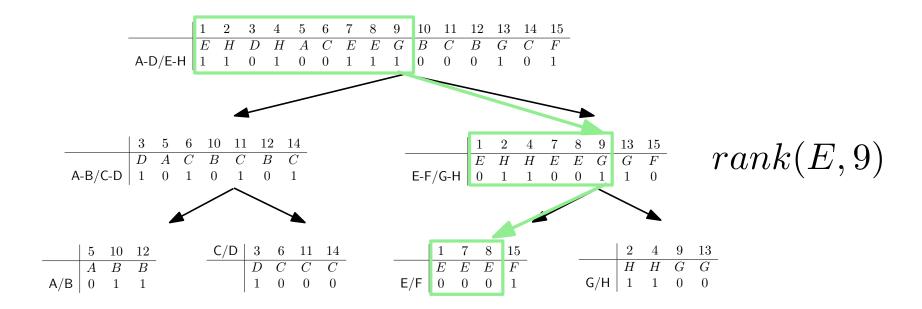
access takes  $O(\log \sigma)$  time



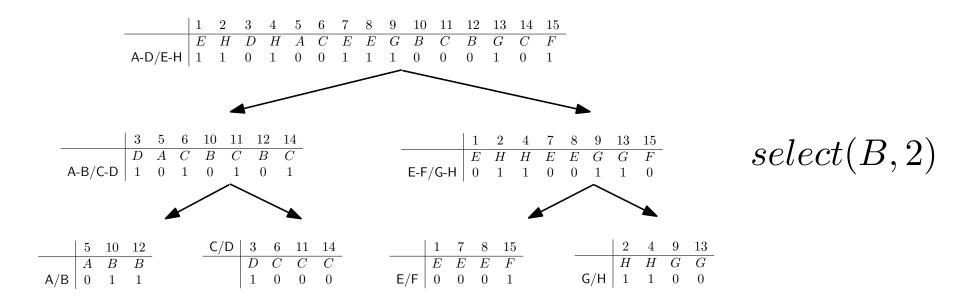


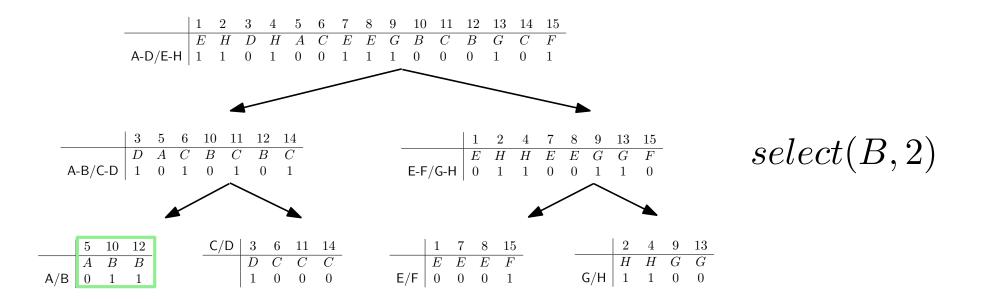


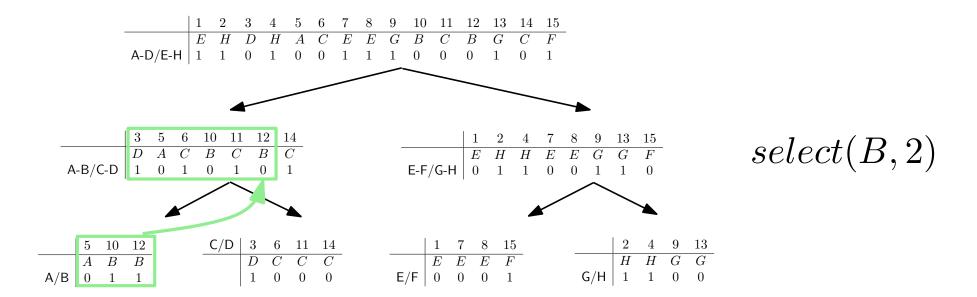


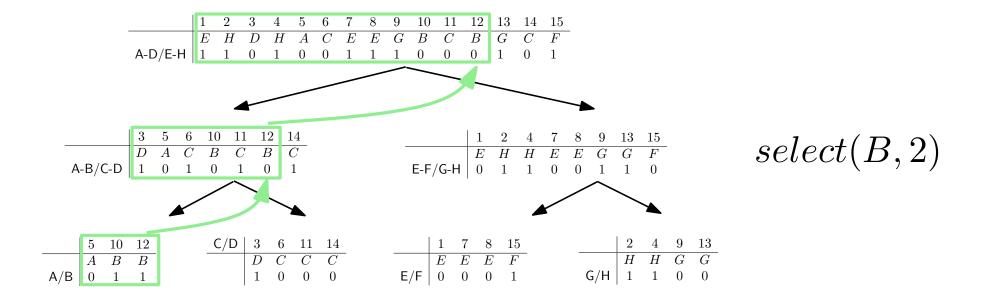


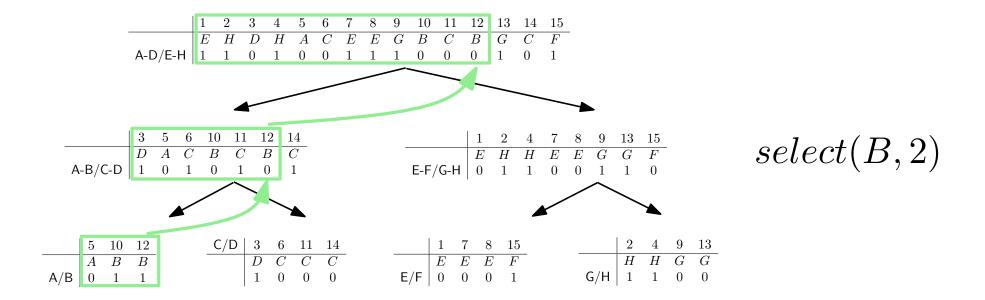
rank also takes  $O(\log \sigma)$  time











select takes  $O(\log \sigma)$  time

We can save pointers

Other codifications work too

Huffman Shape

Space:  $nH_0(S) + o(n \log \sigma)$  bits

Query time:  $O(H_0(S))$  expected

We can save pointers



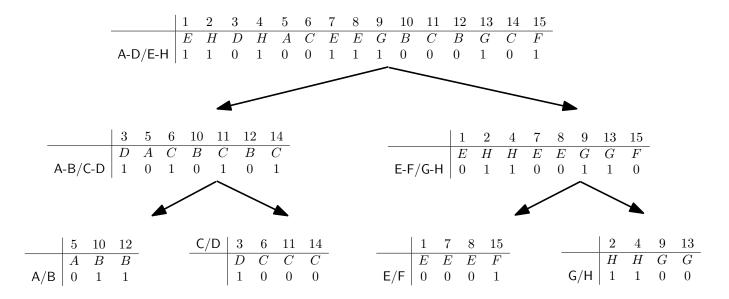
Other codifications work too

Huffman Shape

Space:  $nH_0(S) + o(n \log \sigma)$  bits

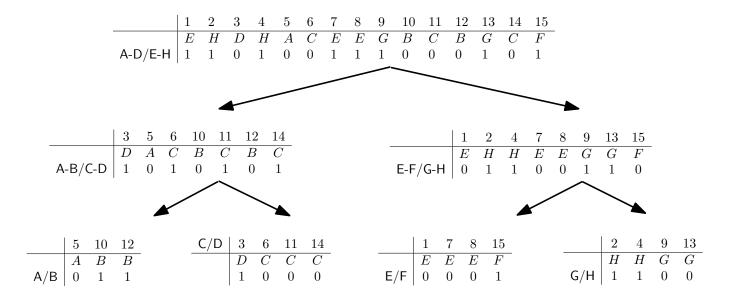
Query time:  $O(H_0(S))$  expected

| 1              | 2 | 3              | 4              | 5              | 6 | 7              | 8              | 9              | 10 | 11 | 12 | 13             | 14 | 15             |
|----------------|---|----------------|----------------|----------------|---|----------------|----------------|----------------|----|----|----|----------------|----|----------------|
| $\overline{E}$ | H | $\overline{D}$ | $\overline{H}$ | $\overline{A}$ | C | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | C  | B  | $\overline{G}$ | C  | $\overline{F}$ |
| 1              | 1 | 0              | 1              | 0              | 0 | 1              | 1              | 1              | 0  | 0  | 0  | 1              | 0  | 1              |
| 1              | 0 | 1              | 0              | 1              | 0 | 1              | 0              | 1              | 1  | 0  | 0  | 1              | 1  | 0              |
| 0              | 1 | 1              | 1              | 0              | 0 | 0              | 0              | 0              | 0  | 1  | 1  | 1              | 0  | 0              |

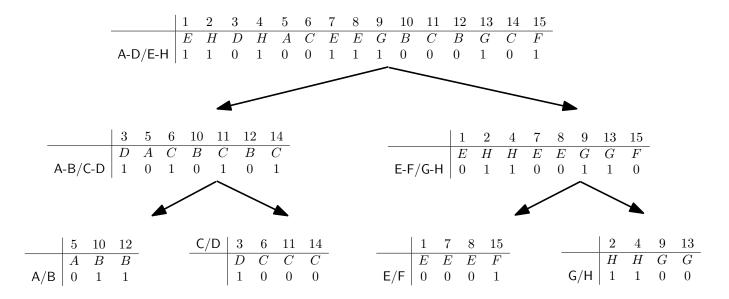


| 1              | 2              | 3              | 4              | 5              | 6              | 7              | 8              | 9              | 10 | 11 | 12 | 13             | 14             | 15             |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----|----|----|----------------|----------------|----------------|
| $\overline{E}$ | $\overline{H}$ | $\overline{D}$ | $\overline{H}$ | $\overline{A}$ | $\overline{C}$ | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | C  | B  | $\overline{G}$ | $\overline{C}$ | $\overline{F}$ |
| 1              | 1              | 0              | 1              | 0              | 0              | 1              | 1              | 1              | 0  | 0  | 0  | 1              | 0              | 1              |
| 1              | 0              | 1              | 0              | 1              | 0              | 1              | 0              | 1              | 1  | 0  | 0  | 1              | 1              | 0              |
| 0              | 1              | 1              | 1              | 0              | 0              | 0              | 0              | 0              | 0  | 1  | 1  | 1              | 0              | 0              |

| 1              | 2 | 3              | 4 | 5              | 6              | 7              | 8              | 9              | 10 | 11 | 12 | 13             | 14 | 15             |
|----------------|---|----------------|---|----------------|----------------|----------------|----------------|----------------|----|----|----|----------------|----|----------------|
| $\overline{E}$ | H | $\overline{D}$ | H | $\overline{A}$ | $\overline{C}$ | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | C  | B  | $\overline{G}$ | C  | $\overline{F}$ |
|                | 1 | 0              | 1 | 0              | 0              | 1              | 1              | 1              | 0  | 0  | 0  | 1              | 0  | 1              |
| 1              | 0 | 1              | 0 | 1              | 0              | 1              | 0              | 1              | 1  | 0  | 0  | 1              | 1  | 0              |
| 0              | 1 | 1              | 1 | 0              | 0              | 0              | 0              | 0              | 0  | 1  | 1  | 1              | 0  | 0              |



| 1              | 2 | 3              | 4              | 5              | 6              | 7              | 8              | 9              | 10 | 11             | 12 | 13             | 14             | 15             |
|----------------|---|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----|----------------|----|----------------|----------------|----------------|
| $\overline{E}$ | H | $\overline{D}$ | $\overline{H}$ | $\overline{A}$ | $\overline{C}$ | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | $\overline{C}$ | B  | $\overline{G}$ | $\overline{C}$ | $\overline{F}$ |
| 1              | 1 | 0              | 1              | 0              | 0              | 1              | 1              | 1              | 0  | 0              | 0  | 1              | 0              | 1              |
| 1              | 0 | 1              | 0              | 1              | 0              | 1              | 0              | 1              | 1  | 0              | 0  | 1              | 1              | 0              |
| 0              | 1 | 1              | 1              | 0              | 0              | 0              | 0              | 0              | 0  | 1              | 1  | 1              | 0              | 0              |



| 1              | 2              | 3              | 4              | 5              | 6              | 7              | 8              | 9              | 10 | 11             | 12 | 13             | 14             | 15             |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----|----------------|----|----------------|----------------|----------------|
| $\overline{E}$ | $\overline{H}$ | $\overline{D}$ | $\overline{H}$ | $\overline{A}$ | $\overline{C}$ | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | $\overline{C}$ | B  | $\overline{G}$ | $\overline{C}$ | $\overline{F}$ |
| 1              | 1              | 0              | 1              | 0              | 0              | 1              | 1              | 1              | 0  | 0              | 0  | 1              | 0              | 1              |
| 1              | 0              | 1              | 0              | 1              | 0              | 1              | 0              | 1              | 1  | 0              | 0  | 1              | 1              | 0              |
| 0              | 1              | 1              | 1              | 0              | 0              | 0              | 0              | 0              | 0  | 1              | 1  | 1              | 0              | 0              |

Saving pointers is useful for large alphabets:  $O(\sigma \log n)$  bits

We showed a solution with  $O(\log \sigma)$  pointers

| 1              | 2 | 3              | 4 | 5              | 6 | 7              | 8              | 9              | 10 | 11             | 12 | 13             | 14 | 15             |
|----------------|---|----------------|---|----------------|---|----------------|----------------|----------------|----|----------------|----|----------------|----|----------------|
| $\overline{E}$ | H | $\overline{D}$ | H | $\overline{A}$ | C | $\overline{E}$ | $\overline{E}$ | $\overline{G}$ | B  | $\overline{C}$ | B  | $\overline{G}$ | C  | $\overline{F}$ |
| 1              | 1 | 0              | 1 | 0              | 0 | 1              | 1              | 1              | 0  | 0              | 0  | 1              | 0  | 1              |
| 1              | 0 | 1              | 0 | 1              | 0 | 1              | 0              | 1              | 1  | 0              | 0  | 1              | 1  | 0              |
| 0              | 1 | 1              | 1 | 0              | 0 | 0              | 0              | 0              | 0  | 1              | 1  | 1              | 0  | 0              |

We can reduce it further to 1 pointer

With a higher fan-out we can get  $O(\log \sigma / \log \log n)$  time per query.

### Classes Implemented

WaveletTree

WaveletTreeNoptrs

WaveletTree

Sequence

Length

Coder

Bitmap Builder

Mapper

WaveletTreeNoptrs

Sequence

Length

Bitmap Builder

Mapper

Classes Implemented

WaveletTree

WaveletTreeNoptrs

WaveletTree

Sequence

Length

**→** Coder

Bitmap Builder

Mapper

WaveletTreeNoptrs

Sequence

Length

Bitmap Builder

Mapper

#### Wavelet Trees

Classes Implemented

**WaveletTree** 

WaveletTreeNoptrs

WaveletTree

Sequence

Length

**→** Coder

Bitmap Builder Mapper

WaveletTreeNoptrs

Sequence

Length

→ Bitmap Builder

Mapper

#### Wavelet Trees

Classes Implemented

**WaveletTree** 

WaveletTreeNoptrs

WaveletTree

Sequence

Length

**→** Coder

→ Bitmap Builder

→ Mapper

WaveletTreeNoptrs

Sequence

Length

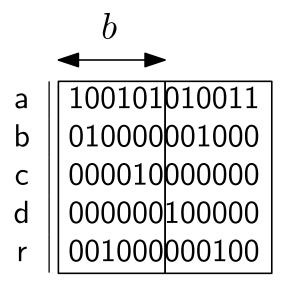
→ Bitmap Builder

**→** Mapper

#### Wavelet Trees

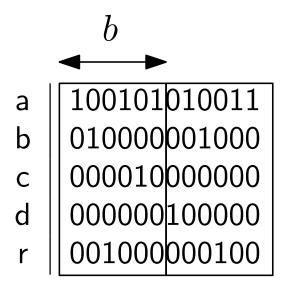
```
size_t N;
uint s;
cout << "Length: ";</pre>
cin >> N;
uint * seq = new uint[N];
for(size_t i=0;i<N;i++) {
  uint v;
  cout << "Element at position " << i << ": ";</pre>
  cin >> seq[i];
WaveletTree * wt1 = new WaveletTree(seq, N,
    new wt_coder_huff(seq, N,
                       new MapperNone()),
    new BitSequenceBuilderRG(20),
    new MapperNone());
cout << "size = " << wt1->getSize() << " bytes" << endl;
```

## S = abracadabraa



 $B = 10001000 \ 1010 \ 101 \ 110 \ 1010$ 

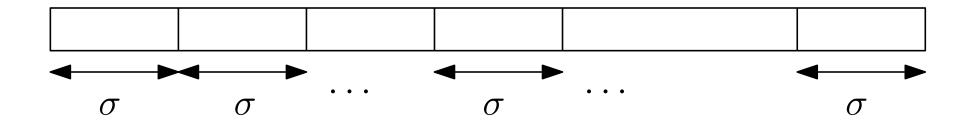
# S = abracadabraa



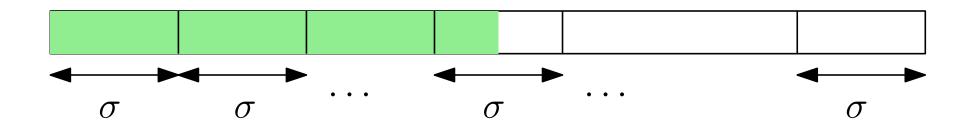
 $B = 10001000 \ 1010 \ 101 \ 110 \ 1010$ 

Space:  $n\sigma/b + n$  bits

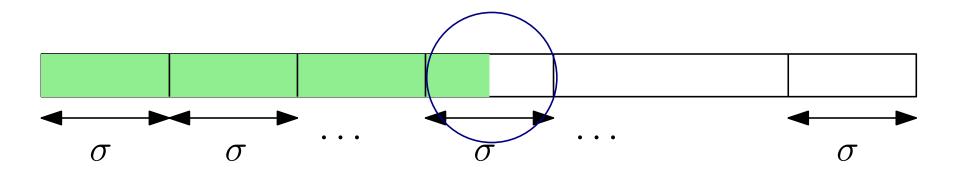
We can solve rank and select for multiples of  $\sigma$ 



We can solve rank and select for multiples of  $\sigma$ 



We can solve rank and select for multiples of  $\sigma$ 



We need to solve rank, select and access for sequences of length  $\sigma$  These are called *chunks* 

```
X = 100000100010000
\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]
```

# S=abcaaccbbcaa

access(3)

X = 100000100010000

 $\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$ 

$$X = 100000100010000$$

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

$$\pi^{-1}(3) = 9$$

$$X = 100000100010000$$

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

$$\pi^{-1}(3) = 9$$

## S=abcaaccbbcaa

X = 100000100010000  $\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$   $\pi^{-1}(3) = 9$ 

```
select(b, 2)
```

$$X = 100000100010000$$

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

```
select(b,2)
```

$$X = 100000100010000$$

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

```
select(b,2)
```

$$X = 100000100010000$$

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

# S=abcaaccbbcaa

X = 100000100010000

```
rank(c, 8)
```

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

# S=abcaaccbbcaa

```
rank(c, 8)
```

X = 100000100010000

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

```
X = 100000100010000
\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]
```

## S=abcaaccbbcaa

rank(c, 8)

X = 100000100010000

$$\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$$

With a Y-Fast trie, the search takes  $O(\log \log \sigma)$ 

## S=abcaaccbbcaa

rank(c, 8)

X = 100000100010000

 $\pi = [1, 4, 5, 11, 12, 2, 8, 9, 3, 6, 7, 10]$ 

With a Y-Fast trie, the search takes  $O(\log \log \sigma)$ 

In practice we use binary search

We can represent a permutation over  $\sigma$  values in  $\sigma \log \sigma (1+o(1))$  bits

 $\pi(i)$  takes O(1) time and  $\pi^{-1}(i)$  takes  $O(\log \log \sigma)$  [Munro et al.]

GMR supports rank and access in  $O(\log\log\sigma)$  time and select in constant time, within space  $n\log\sigma + n\cdot o(\log\sigma)$  bits of space.

In LIBCDS

SequenceGMR

chunk\_length

PermutationBuilderMRRR

Sequence Builder GMR Chunk

```
size_t N;
uint s;
cout << "Length: ";</pre>
cin >> N;
uint * seq = new uint[N];
for(size_t i=0;i<N;i++) {</pre>
  uint v;
  cout << "Element at position " << i << ": ";</pre>
  cin >> seq[i];
SequenceGMR * gmr = new SequenceGMR(seq, N, 5u,
  new BitSequenceBuilderRG(20),
  new SequenceBuilderGMRChunk(
     new BitSequenceBuilderRG(20),
     new PermutationBuilderMRRR(
                  20,
                  new BitSequenceBuilderRG(20)));
cout << "size = " << gmr->getSize() << " bytes" << endl;</pre>
```

| Symbol | Fred |
|--------|------|
| Α      | 1    |
| В      | 2    |
| C      | 3    |
| D      | 1    |
| Е      | 3    |
| F      | 1    |
| G      | 2    |
| Н      | 2    |

| Symbol | Freq | Symbol | Freq |
|--------|------|--------|------|
| Α      | 1    | C      | 3    |
| В      | 2    | E      | 3    |
| C      | 3    | В      | 2    |
| D      | 1    | G      | 2    |
| Ε      | 3    | Н      | 2    |
| F      | 1    | A      | 1    |
| G      | 2    | D      | 1    |
| Н      | 2    | F      | 1    |
|        | I    |        |      |

| Symbol | Freq |             | Symbol | Freq |         |
|--------|------|-------------|--------|------|---------|
| Α      | 1    |             | C      | 3    | Class 1 |
| В      | 2    |             | Е      | 3    |         |
| C      | 3    |             | B      | 2    | Class 2 |
| D      | 1    | <del></del> | G      | 2    |         |
| Е      | 3    |             | Н      | 2    | Class 3 |
| F      | 1    |             | Α      | 1    | 0.000   |
| G      | 2    |             | D      | 1    |         |
| Н      | 2    |             | F      | 1    | Class 4 |

| Symbol | Freq |         |    |
|--------|------|---------|----|
| C      | 3    | Class 1 |    |
| E      | 3    |         | C: |
| B      | 2    | Class 2 |    |
| G      | 2    |         | O: |
| Н      | 2    | Class 3 |    |
| Α      | 1    |         |    |
| D      | 1    |         |    |
| F      | 1    | Class 4 |    |

| Symbol | Freq |         |    |
|--------|------|---------|----|
| C      | 3    | Class 1 |    |
| E      | 3    |         | C: |
| B      | 2    | Class 2 |    |
| G      | 2    |         | O: |
| Н      | 2    | Class 3 |    |
| Α      | 1    |         |    |
| D      | 1    |         |    |
| F      | 1    | Class 4 |    |

| Symbol | Freq |           |    |   |
|--------|------|-----------|----|---|
| C      | 3    | Class 1   |    |   |
| E      | 3    | CI O      | C: | 2 |
| B      | 2    | Class 2   |    |   |
| G      | 2    |           | O: | 0 |
| Н      | 2    | Class 3   |    |   |
| Α      | 1    |           |    |   |
| D      | 1    |           |    |   |
| F      | 1    | — Class 4 |    |   |

| Symbol | Freq |         |      |
|--------|------|---------|------|
| C      | 3    | Class 1 |      |
| Е      | 3    |         | C: 2 |
| B      | 2    | Class 2 |      |
| G      | 2    |         | O: 0 |
| Н      | 2    | Class 3 |      |
| A      | 1    |         |      |
| D      | 1    |         |      |
| F      | 1    | Class 4 |      |

| Symbol | Freq |              |    |   |   |
|--------|------|--------------|----|---|---|
| C      | 3    | Class 1      |    |   |   |
| Е      | 3    |              | C: | 2 | 3 |
| B      | 2    | Class 2<br>— |    |   |   |
| G      | 2    |              | O: | 0 | 1 |
| Н      | 2    | Class 3      |    |   |   |
| А      | 1    |              |    |   |   |
| D      | 1    |              |    |   |   |
| F      | 1    | —<br>Class 4 |    |   |   |

$$S = EHDDHACEEGBCBGCF$$

| Symbol | Freq |         |        |       |
|--------|------|---------|--------|-------|
| C      | 3    | Class 1 |        |       |
| Е      | 3    |         | C: 2 3 |       |
| B      | 2    | Class 2 |        |       |
| G      | 2    |         | O: 0 1 | • • • |
| Н      | 2    | Class 3 |        |       |
| Α      | 1    |         |        |       |
| D      | 1    |         |        |       |
| F      | 1    | Class 4 |        |       |

$$S = EHDHACEEGBCBGCF$$

C: 2 3 ...

O: 0 1 ...

Represent C using a Wavelet Tree

$$S = EHDHACEEGBCBGCF$$

C: 2 3 ...

O: 0 1 ...

Represent C using a Wavelet Tree

How many classes do we have?

$$S = EHDHACEEGBCBGCF$$

C: 2 3 ...

O: 0 1 ...

Represent C using a Wavelet Tree

How many classes do we have?

Space for C is  $O(n \log \log \sigma)$ , queries take constant time.

S = EHDHACEEGBCBGCF

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

S = EHDHACEEGBCBGCF

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

S = EHDHACEEGBCBGCF

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| $O_1$                             |                           |
|-----------------------------------|---------------------------|
| $O_2$ : 0, 0, 0, 1, 1             |                           |
| O <sub>3</sub> : 1, 3, 1, 2, 0, 0 | $\rangle \approx nH_0(B)$ |
| $O_4$ : 0                         |                           |

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

| S =      | EHDHACEEGBCBGCF | 7 |
|----------|-----------------|---|
| <u> </u> |                 |   |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| O | 1 |
|---|---|
|   |   |

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

access(7)

| Symbol | Freq |   |
|--------|------|---|
| C      | 3    |   |
| E      | 3    | _ |
| В      | 2    |   |
| G      | 2    |   |
| Н      | 2    |   |
| А      | 1    |   |
| D      | 1    |   |
| F      | 1    |   |

| S =      | EHDHACEEGBCBGCF |
|----------|-----------------|
| <u> </u> |                 |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| $O_1$ |
|-------|
|-------|

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

access(7)

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| А      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

S = EHDHACEEGBCBGCF

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

 $O_1$ 

 $O_2$ : 0, 0 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

access(7)

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| O | 1 |
|---|---|
|   |   |

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

| S = | EHDHACE | EGBCBGCF |
|-----|---------|----------|
|     |         |          |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| $\smile$ 1 |
|------------|
|------------|

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

| S _        | EHDHACE | EGBCBGCF |
|------------|---------|----------|
| <b>5</b> — | DHDHAOD | EGDCDGCF |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| Е      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

$$S = EHDHACEEGBCBGCF$$

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

$$S = EHDHACEEGBCBGCF$$

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| Е      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

| 5 | _ | EHDHA | CEECE | RCRCCF |
|---|---|-------|-------|--------|
|   |   | DHDHA | ODDOL | ODGOI  |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| Α      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

S = EHDHACEEGBCBGCF

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

| Symbol | Freq |
|--------|------|
| C      | 3    |
| E      | 3    |
| В      | 2    |
| G      | 2    |
| Н      | 2    |
| Α      | 1    |
| D      | 1    |
| F      | 1    |

 $O_1$ 

 $O_2$ : 0, 0, 0, 1, 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| S =      | EHDHACEEGBCBGCF |
|----------|-----------------|
| <u> </u> |                 |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2, 1, 2, 3, 1, 4

|   | Symbol | Freq |  |
|---|--------|------|--|
|   | C      | 3    |  |
|   | E      | 3    |  |
|   | В      | 2    |  |
| • | G      | 2    |  |
|   | Н      | 2    |  |
|   | Α      | 1    |  |
|   | D      | 1    |  |
|   | F      | 1    |  |

 $O_1$ 

 $O_2$ : 0, 0, 0, 1) 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| S =      | EHDHACEEGBCBGCF |
|----------|-----------------|
| <u> </u> |                 |

C: 2, 3, 3, 3, 3, 1, 2, 2, 3, 2 1, 2, 3, 1, 4

| $\smile$ $\Box$ |
|-----------------|
|-----------------|

 $O_2$ : 0, 0, 0, 1) 1

 $O_3$ : 1, 3, 1, 2, 0, 0

 $O_4$ : 0

| Symbol | Freq |  |
|--------|------|--|
| C      | 3    |  |
| E      | 3    |  |
| В      | 2    |  |
| G      | 2    |  |
| Н      | 2    |  |
| А      | 1    |  |
| D      | 1    |  |
| F      | 1    |  |

. . .

```
SequenceBuilder * sb1 = new SequenceBuilderWaveletTree(
                           new BitSequenceBuilderRG(20),
                            new MapperNone());
SequenceBuilder * sb2 = new SequenceBuilderGMRChunk(
                           new BitSequenceBuilderRG(20),
                            new PermutationBuilderMRRR(
                                20,
                                new BitSequenceBuilderRG(20)));
SequenceBuilder * sb3 = new SequenceBuilderGMR(
                           new BitSequenceBuilderRG(20), sb2);
SequenceAlphPart * ap = new SequenceAlphPart(seq, N, Ou,
                            sb1, sb3);
cout << "size = " << ap->getSize() << " bytes" << endl;</pre>
```

. . .

## Summary

#### BitSequence:

- BitSequenceRG
- BitSequenceRRR
- BitSequenceSDArray

#### Sequence:

- WaveletTree
- WaveletTreeNoptrs
- SequenceGMR
- SequenceGMRChunk
- AlphPart

### Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

#### Outline

- Motivation
- Basics
- Bitmaps
- Sequences
- Applications

## **Applications**

- Permutation
- Graph
- Grid
- Text FM-index [Ferragina and Manzini]

$$\pi = [1, 3, 7, 4, 2, 5, 6]$$

$$\pi = [1, 3, 7, 4, 2, 5, 6]$$

$$\pi(i) = access(i)$$

$$\pi = [1, 3, 7, 4, 2, 5, 6]$$

$$\pi(i) = access(i)$$

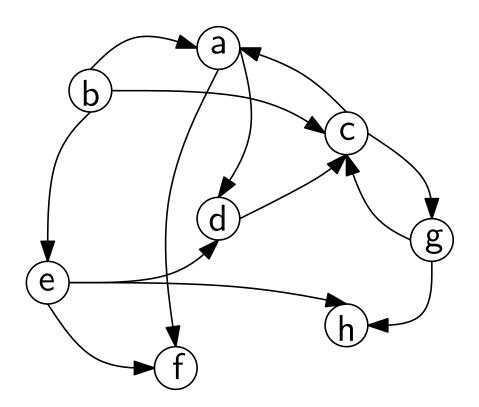
$$\pi^{-1}(i) = select(i)$$

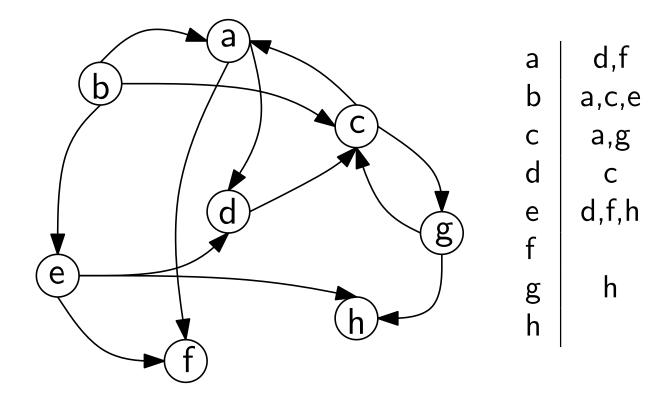
$$\pi = [1, 3, 7, 4, 2, 5, 6]$$

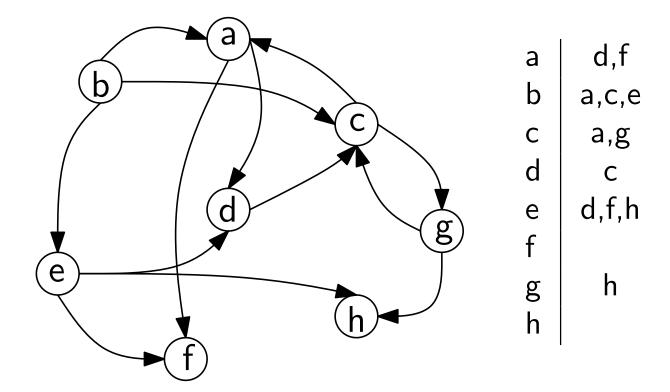
$$\pi(i) = access(i)$$

$$\pi^{-1}(i) = select(i)$$

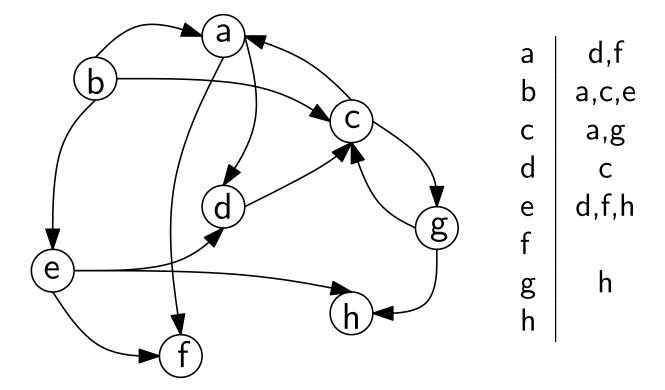
MRRR achieves  $(1, \log \log n)$ Wavelet Trees achieve  $(\log n, \log n)$ GMR achieves  $(\log \log n, 1)$ 

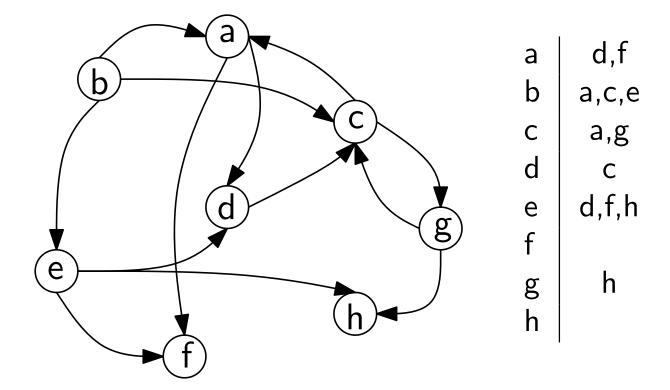


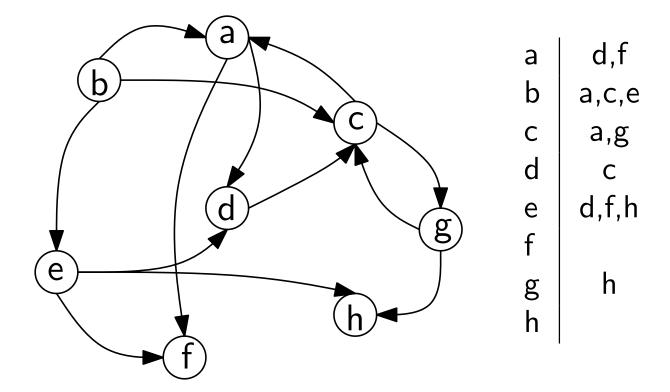




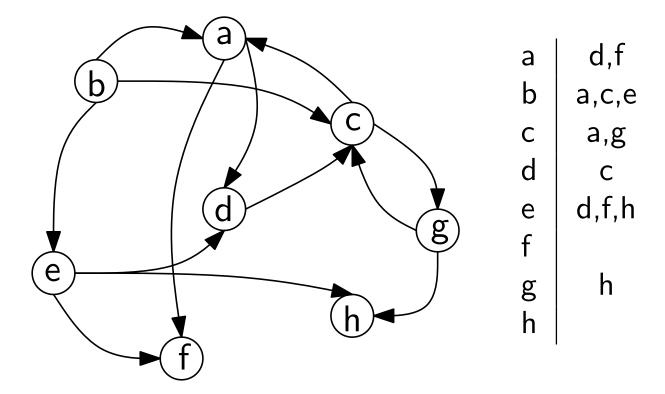
```
1
                                                   1
              1
                                  1
B
S
         0
            0
                 0
                    0
                       0
                             0
                               0
                                      0 1
                                           0 0
                                                 0
                                                       1
                                                          0 1
                                                 h
         d
                  a
                                           d
                       е
                                g
                                      С
                    С
                             a
```

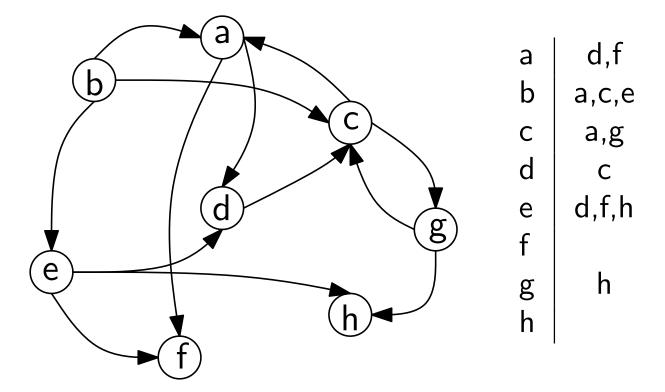


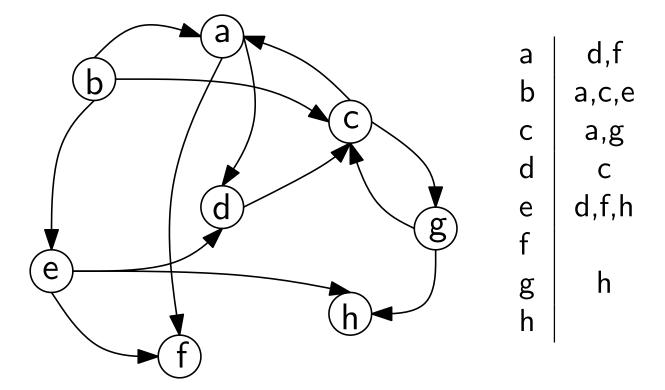


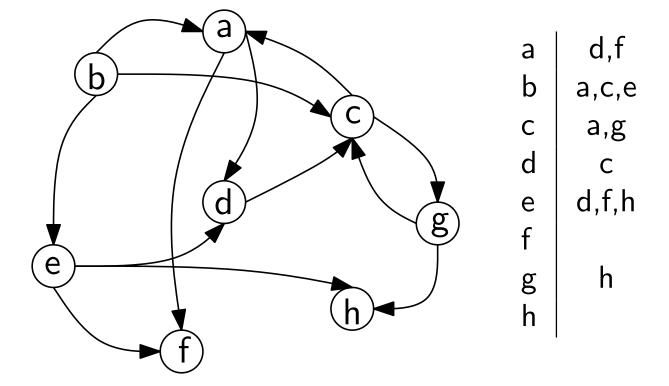


```
1
                                                      1
B
S
                               0
                                 0
                                     1
                                                    0
h
                                                          1
                                                             0 1
            0
                   0
                        0
                                        0
                                          1
                                              0 0
         0
                     0
                                              d
                                  g
                                        С
                         e
                      С
                               a
```









- Space:  $m \log n(1 + o(1))$
- Retrieve Neighbors:  $O(\log \log n)$
- Retrieve Reverse Neighbors: O(1)
- Check Connection:  $O(\log \log n)$

- Space:  $m \log n(1 + o(1))$
- Retrieve Neighbors:  $O(\log \log n)$
- Retrieve Reverse Neighbors: O(1)
- Check Connection:  $O(\log \log n)$

Adjacency list requires  $n \log m + m \log n$ 

- Neighbors O(1)
- Reverse Neighbors?
- Check Connection?

### Graphs

- Space:  $m \log n(1 + o(1))$
- Retrieve Neighbors:  $O(\log \log n)$
- Retrieve Reverse Neighbors: O(1)
- Check Connection:  $O(\log \log n)$

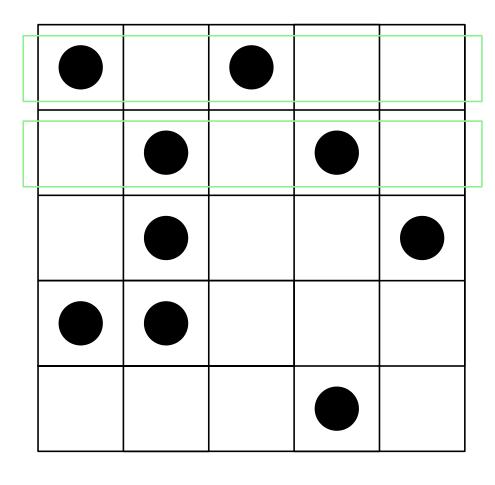
#### Adjacency list requires $n \log m + m \log n$

- Neighbors O(1)
- Reverse Neighbors?
- Check Connection?

#### Adjacency matrix requires $n^2$

- Neighbors: O(n)
- Reverse Neighbors: O(n)
- Check Connection: O(1)

|  |  |  | 1, 3 |
|--|--|--|------|
|  |  |  |      |
|  |  |  |      |
|  |  |  |      |
|  |  |  |      |

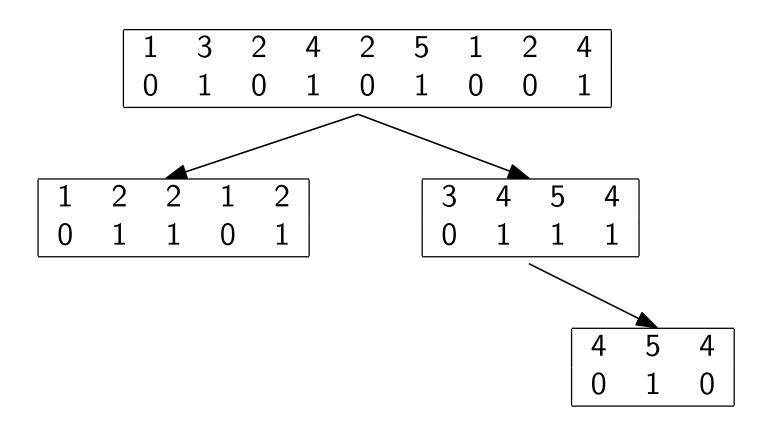


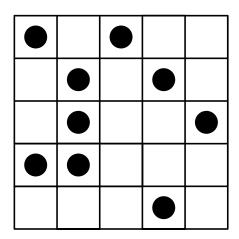
1, 3

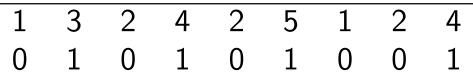
2, 4

B 1 0 0 1 0 0 1 0 0 1 0 0 1 0 T 1 3 2 4 2 5 1 2 4

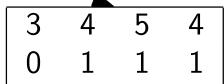




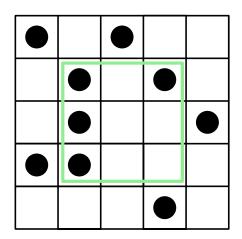


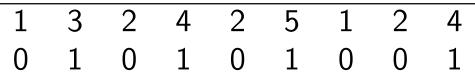


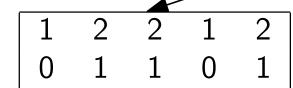




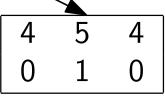
4 5 4 0 1 0



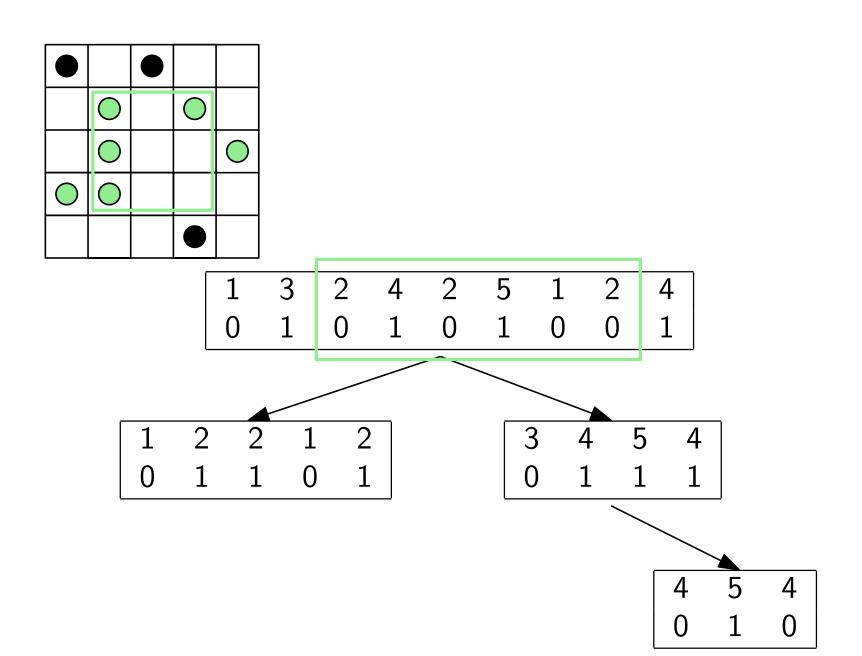




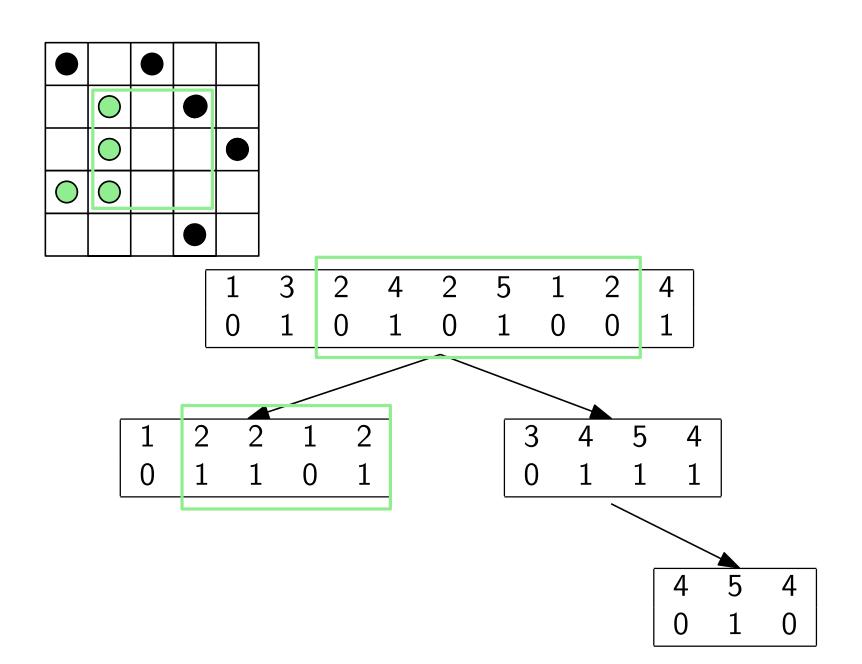




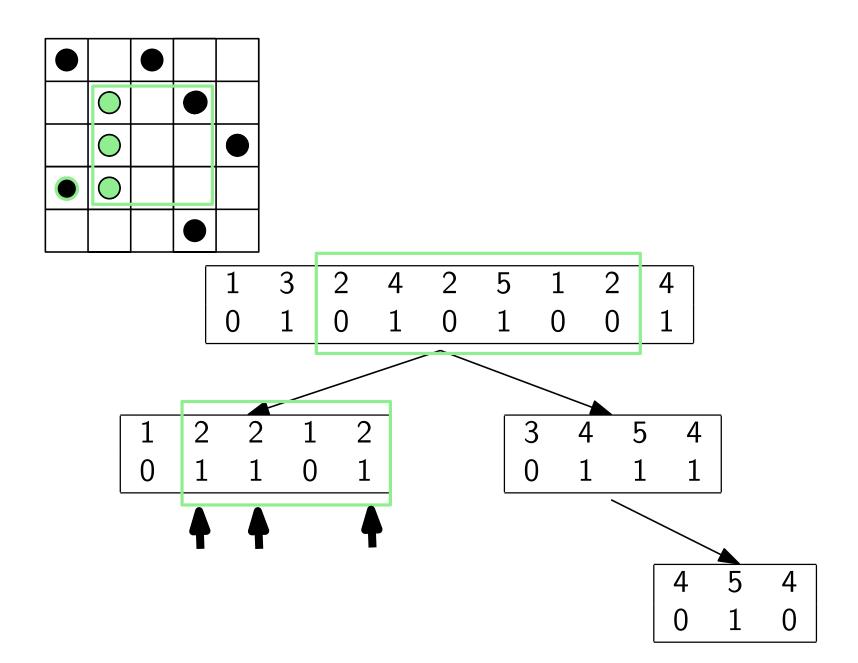
Grids



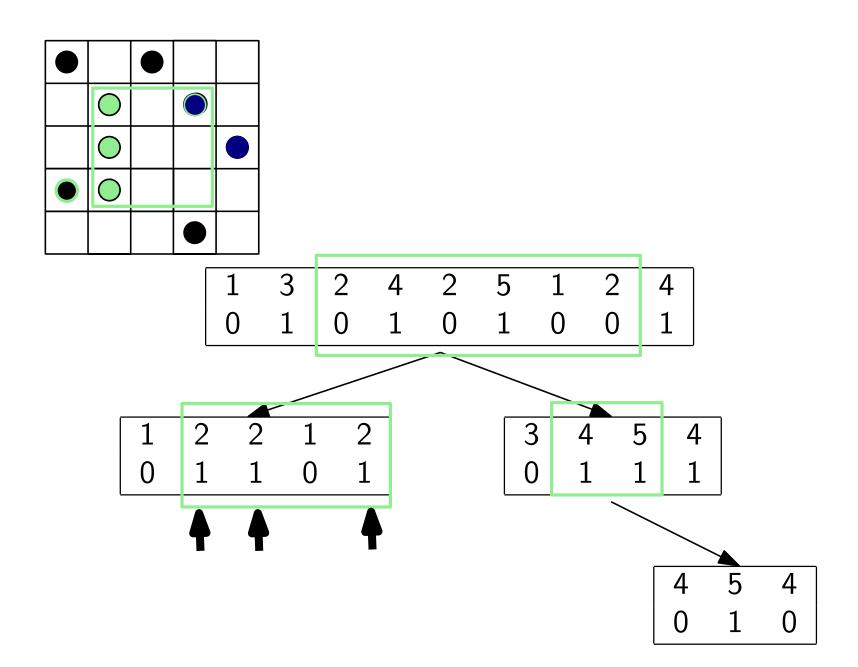
Grids

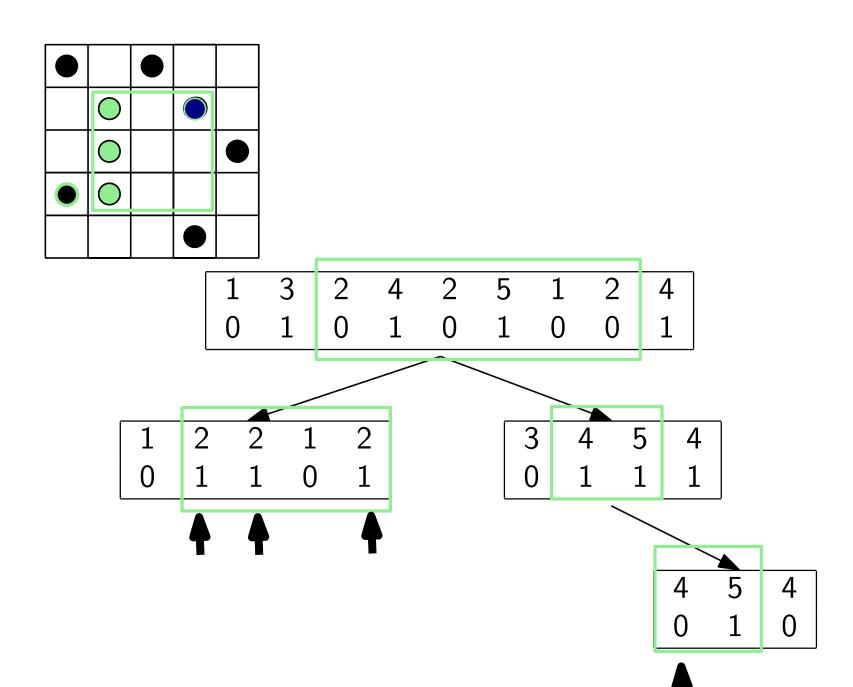


Grids



Grids

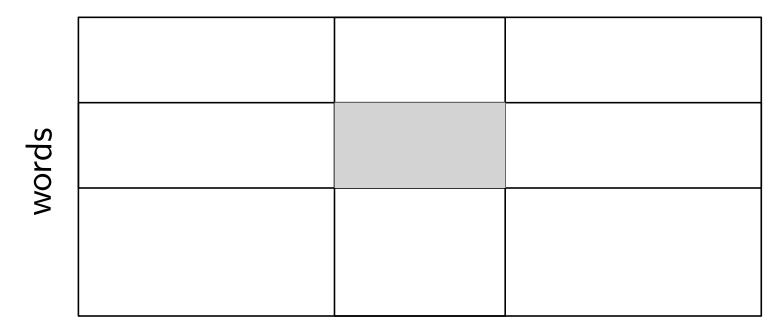




Binary Relations Graphs Inverted Indexes

Binary Relations Graphs Inverted Indexes

#### documents



#### Text

FM-Index

compressed

#### Text

FM-Index

self-index { locate count extract

compressed

Burrows-Wheeler Transform

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | L  |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| \$         | a  |    | a  | b  | a  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | a  |
| a          | \$ | a  |    | а  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  |
| a          | b  | a  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  | ı  |
| a          | b  | a  | r  | d  | а  | \$ | а  |    | a  | b  | a  | r  | а  |    | a  | ı  |
| a          |    | a  | b  | а  | r  | а  |    | a  |    | а  | b  | a  | r  | d  | а  | \$ |
| a          |    | a  | b  | а  | r  | d  | а  | \$ | а  |    | a  | b  | a  | r  | а  | ı  |
| a          |    | a  |    | а  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b  | а  | r  |
| a          | r  | a  |    | а  |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | а  | b  |
| a          | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a  |    | a  |    | а  | b  |
| <b>b</b>   | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a  |
| b          | а  | r  | d  | а  | \$ | а  |    | a  | b  | а  | r  | a  |    | a  |    | a  |
| <b>d</b>   | а  | \$ | а  |    | а  | b  | а  | r  | а  |    | a  |    | a  | b  | а  | r  |
|            | а  | b  | а  | r  | а  |    | а  |    | а  | b  | a  | r  | d  | a  | \$ | a  |
|            | a  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | a  | r  | a  |    | a  |
|            | a  |    | а  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a  |
| r          | a  |    | а  |    | а  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b  | a  |
| r          | d  | a  | \$ | a  |    | a  | b  | а  | r  | а  |    | a  |    | a  | b  | a  |

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | L   |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| \$         | a  |    | a  | b  | a  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | (a) |
| a          | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d   |
| a          | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  | ı   |
| a          | b  | a  | r  | d  | а  | \$ | а  |    | a  | b  | a  | r  | а  |    | a  | ı   |
| a          |    | a  | b  | a  | r  | а  |    | a  |    | а  | b  | a  | r  | d  | а  | \$  |
| a          |    | a  | b  | a  | r  | d  | а  | \$ | а  |    | a  | b  | a  | r  | a  | ı   |
| a          |    | a  |    | а  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b  | а  | r   |
| a          | r  | а  |    | а  |    | a  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b   |
| a          | r  | d  | а  | \$ | а  |    | а  | b  | a  | r  | a  |    | a  |    | а  | b   |
| b          | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a   |
| b          | а  | r  | d  | a  | \$ | а  |    | a  | b  | а  | r  | a  |    | a  |    | a   |
| d          | а  | \$ | а  |    | а  | b  | а  | r  | а  |    | а  |    | a  | b  | а  | r   |
|            | а  | b  | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a  | \$ | a   |
|            | а  | b  | а  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a   |
|            | a  |    | а  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a   |
| r          | a  |    | а  |    | а  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b  | a   |
| r          | d  | a  | \$ | а  |    | a  | b  | a  | r  | a  | 1  | a  |    | a  | b  | a   |

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | L   |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| \$         | a  |    | a  | b  | a  | r  | a  |    | а  |    | a  | b  | a  | r  | d  | (a) |
| (a)        | \$ | a  | I  | a  | b  | a  | r  | a  |    | a  | I  | a  | b  | a  | r  | d   |
| a          | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  | ı   |
| a          | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  | ı   |
| a          |    | a  | b  | a  | r  | a  |    | а  |    | a  | b  | а  | r  | d  | а  | \$  |
| a          |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  | ı   |
| a          |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r   |
| a          | r  | a  | ı  | a  | ı  | а  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b   |
| a          | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b   |
| b          | a  | r  | a  |    | a  |    | a  | b  | а  | r  | d  | a  | \$ | a  |    | a   |
| b          | a  | r  | d  | a  | \$ | a  |    | а  | b  | a  | r  | а  |    | а  |    | a   |
| d          | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | а  | r   |
| 1          | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a   |
| 1          | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a  | r  | a  |    | a   |
|            | а  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | а  | b  | а  | r  | a   |
| r          | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a   |
| r          | d  | a  | \$ | a  | 1  | a  | b  | а  | r  | a  | 1  | а  |    | а  | b  | a   |

Text

| $oxed{F}$ |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------------|
| \$        | a  |    | a  | b  | a  | r  | a  | Ī  | a  |    | а  | b  | a  | r  | d  | (a)            |
| (a)       | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | $ \mathbf{d} $ |
| a         | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |                |
| a         | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | a  |    | a  | 1              |
| a         |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | а  | r  | d  | а  | \$             |
| a         |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a  | r  | а  |                |
| a         |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | а  | b  | а  | r              |
| a         | r  | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | а  | <b>b</b>       |
| a         | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | a  |    | а  | <b>b</b>       |
| b         | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | a              |
| b         | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | a  |    | a              |
| d         | а  | \$ | а  | I  | а  | b  | а  | r  | a  |    | а  |    | а  | b  | а  | r              |
|           | a  | b  | a  | r  | a  |    | a  |    | a  | b  | а  | r  | d  | a  | \$ | a              |
|           | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | a  |    | a              |
|           | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | а  | b  | a  | r  | a              |
| r         | a  |    | a  | 1  | a  | b  | a  | r  | d  | a  | \$ | а  |    | а  | b  | a              |
| r         | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | а  |    | a  | b  | a              |

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | L   |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| \$         | a  |    | a  | b  | a  | r  | a  |    | а  |    | a  | b  | a  | r  | d  | (a) |
| (a)        | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | (d) |
| a          | b  | а  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  | I   |
| a          | b  | а  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  | ı   |
| a          |    | а  | b  | a  | r  | a  |    | а  |    | a  | b  | a  | r  | d  | а  | \$  |
| a          |    | а  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  | I   |
| a          |    | а  |    | a  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r   |
| a          | r  | а  |    | a  | ı  | а  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b   |
| a          | r  | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b   |
| b          | a  | r  | a  |    | a  |    | a  | b  | а  | r  | d  | a  | \$ | a  |    | a   |
| b          | a  | r  | d  | a  | \$ | a  |    | а  | b  | a  | r  | a  |    | а  |    | a   |
| d          | a  | \$ | а  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | а  | r   |
|            | a  | b  | а  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a   |
|            | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a   |
|            | а  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | a  | b  | а  | r  | a   |
| r          | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a   |
| r          | d  | a  | \$ | a  | 1  | a  | b  | а  | r  | a  | 1  | a  |    | a  | b  | a   |

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$      |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------------|
| \$         | a  |    | a  | b  | a  | r  | а  |    | а  |    | a  | b  | a  | r  | d  | (a)            |
| (a)        | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b  | a  | r  | $ \mathbf{d} $ |
| a          | b  | a  | r  | a  |    | а  |    | a  | b  | а  | r  | d  | a  | \$ | a  |                |
| a          | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |                |
| a          |    | a  | b  | a  | r  | a  |    | a  |    | а  | b  | а  | r  | d  | а  | \$             |
| a          |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b  | a  | r  | а  |                |
| a          |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | а  |    | a  | b  | а  | r              |
| a          | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | <b>b</b>       |
| a          | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | а  |    | а  |    | а  | <b>b</b>       |
| b          | a  | r  | a  |    | a  |    | a  | b  | а  | r  | d  | а  | \$ | a  |    | a              |
| b          | a  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | а  |    | a  |    | a              |
| d          | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | а  |    | a  | b  | а  |                |
| I          | a  | b  | а  | r  | a  |    | a  |    | а  | b  | а  | r  | d  | a  | \$ | a              |
|            | а  | b  | а  | r  | d  | a  | \$ | a  |    | а  | b  | а  | r  | а  |    | a              |
|            | а  |    | a  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | a  | r  | a              |
| r          | a  |    | a  | 1  | a  | b  | a  | r  | d  | а  | \$ | а  |    | a  | b  | a              |
| r          | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | а  |    | a  | b  | a              |

Text

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $\mid L \mid$ |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---------------|
| \$         | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | (a)           |
| (a)        | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | (d)           |
| a          | b  | а  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |               |
| a          | b  | а  | r  | d  | а  | \$ | a  |    | а  | b  | a  | r  | а  |    | a  |               |
| a          |    | а  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$            |
| a          |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | a  | 1             |
| a          |    | а  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r             |
| a          | r  | а  |    | а  |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b             |
| a          | r  | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | а  |    | a  | b             |
| b          | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a             |
| b          | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a             |
| d          | a  | \$ | а  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b  | a  | r             |
| I          | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a             |
| I          | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a             |
|            | а  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | а  | r  | a             |
| r          | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | а             |
| r          | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a             |

Text

| $oxed{F}$ |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                  |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|------------------|
| \$        | а  |    | а  | b  | a  | r  | а  |    | a  |    | а  | b  | а  | r  | d  | (a)              |
| (a)       | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | $ (\mathbf{d}) $ |
| a         | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |                  |
| a         | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |                  |
| a         |    | a  | b  | a  | r  | а  |    | a  |    | а  | b  | a  | r  | d  | a  | \$               |
| a         |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |                  |
| a         |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b  | а  | r                |
| a         | r  | a  |    | a  |    | а  | b  | a  | r  | d  | a  | \$ | a  |    | a  | <b>b</b>         |
| a         | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | a  |    | a  | <b>b</b>         |
| <b>b</b>  | а  | r  | a  | ı  | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a                |
| b         | а  | r  | d  | a  | \$ | а  |    | a  | b  | a  | r  | a  |    | a  |    | a                |
|           | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | а  |    | а  | b  | а  |                  |
|           | а  | b  | a  | r  | а  |    | a  | ı  | a  | b  | а  | r  | d  | a  | \$ | a                |
|           | а  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a                |
|           | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | a                |
| r         | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a                |
|           | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a                |

Text

| $oxed{F}$ |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $\mid L \mid$ |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---------------|
| \$        | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | (a)           |
| (a)       | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | (d)           |
| a         | b  | а  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |               |
| a         | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | a  |               |
| a         |    | а  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$            |
| a         |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a  | b  | а  | r  | a  |               |
| a         |    | а  |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b  | a  | r             |
| a         | r  | а  |    | а  |    | а  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b             |
| a         | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | а  |    | а  | b             |
| b         | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a             |
| b         | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a             |
| <b>d</b>  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b  | а  | r             |
| I         | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a             |
|           | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a  | r  | a  |    | a             |
|           | a  |    | а  | b  | а  | r  | d  | a  | \$ | а  |    | a  | b  | а  | r  | a             |
| r         | a  |    | a  |    | a  | b  | a  | r  | d  | а  | \$ | a  |    | a  | b  | a             |
| r         | d  | a  | \$ | a  |    | a  | b  | a  | r  | а  |    | a  |    | a  | b  | (a)           |

Text

| $oxed{F}$ |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $\mid L \mid$ |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---------------|
| \$        | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | (a)           |
| (a)       | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | (d)           |
| a         | b  | а  | r  | а  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |               |
| a         | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  | I  | a  |               |
| a         |    | а  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$            |
| a         |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |               |
| a         |    | а  |    | a  | b  | a  | r  | d  | a  | \$ | а  |    | a  | b  | a  | r             |
| a         | r  | а  |    | а  |    | а  | b  | a  | r  | d  | a  | \$ | a  | ı  | a  | b             |
| a         | r  | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | <b>(b)</b>    |
| b         | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | a             |
| b         | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a             |
| <b>d</b>  | a  | \$ | а  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | а  | r             |
| I         | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a             |
|           | a  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a             |
|           | a  |    | a  | b  | а  | r  | d  | a  | \$ | а  |    | a  | b  | a  | r  | a             |
| r         | a  |    | a  |    | a  | b  | a  | r  | d  | a  | \$ | a  |    | а  | b  | a             |
| r         | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  | 1  | a  |    | a  | b  | (a)           |

F\$ aaaaaabbd III ri Lad II\$ Irbbaaraaaa

Talabaralalabarda \$

F\$ aaaaaabbd III rr Lad II\$ Irbba(a) raaaa

Talabaralal(a) barda \$

F\$ aaaaaabbd III rr Lad I () \$ Irbba(a) raaaa

Talabarala(ja) barda \$

F \$ a a a a a a a b b d l l r r L a d l l \$ l r b b a a r a a a a

$$LF(i) = rank(L[i], i) + C[L[i]]$$

Talabarala<mark>(ja</mark>barda\$

$$LF(i) = rank(L[i], i) + C[L[i]]$$

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|
| \$         | а  |    | a  | b  | а  | r  | a  |    | а  |    | a  | b  | а  | r  | d  | а         |
| a          | \$ | а  | I  | а  | b  | а  | r  | а  | I  | a  | I  | a  | b  | a  | r  | d         |
| a          | b  | а  | r  | а  | I  | а  | I  | а  | b  | a  | r  | d  | а  | \$ | а  | ı         |
| a          | b  | a  | r  | d  | а  | \$ | а  | I  | а  | b  | а  | r  | а  |    | а  | 1         |
| a          | ı  | а  | b  | а  | r  | а  | I  | а  | I  | а  | b  | а  | r  | d  | а  | \$        |
| a          | ı  | a  | b  | а  | r  | d  | а  | \$ | а  | I  | a  | b  | а  | r  | а  | 1         |
| a          | I  | а  | I  | a  | b  | а  | r  | d  | а  | \$ | a  |    | а  | b  | a  | r         |
| a          | r  | а  | I  | а  | I  | а  | b  | а  | r  | d  | a  | \$ | а  |    | а  | b         |
| a          | r  | d  | a  | \$ | a  | I  | a  | b  | а  | r  | a  |    | a  |    | а  | b         |
| b          | а  | r  | a  | ı  | a  | 1  | a  | b  | a  | r  | d  | a  | \$ | a  | ı  | a         |
| b          | a  | r  | d  | a  | \$ | а  | I  | a  | b  | a  | r  | a  | I  | a  | I  | a         |
| d          | a  | \$ | a  | ı  | a  | b  | a  | r  | a  | ı  | a  |    | a  | b  | a  | r         |
|            | a  | b  | a  | r  | а  | 1  | а  | 1  | а  | b  | a  | r  | d  | а  | \$ | a         |
|            | а  | b  | a  | r  | d  | а  | \$ | а  | I  | а  | b  | а  | r  | а  |    | a         |
|            | a  | I  | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a         |
| r          | а  | I  | a  |    | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | a         |
| r          | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a         |

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|
| \$         | а  |    | a  | b  | а  | r  | a  |    | а  |    | a  | b  | а  | r  | d  | а         |
| a          | \$ | а  | I  | а  | b  | а  | r  | а  | I  | a  | I  | a  | b  | a  | r  | d         |
| a          | b  | а  | r  | а  | I  | а  | I  | а  | b  | a  | r  | d  | а  | \$ | а  | ı         |
| a          | b  | a  | r  | d  | а  | \$ | а  | I  | а  | b  | a  | r  | а  |    | а  | 1         |
| a          | ı  | а  | b  | а  | r  | а  | I  | а  | I  | a  | b  | а  | r  | d  | а  | \$        |
| a          | ı  | a  | b  | а  | r  | d  | а  | \$ | а  | I  | a  | b  | а  | r  | а  | 1         |
| a          | I  | а  | I  | а  | b  | а  | r  | d  | а  | \$ | a  |    | а  | b  | a  | r         |
| a          | r  | а  | I  | а  | I  | а  | b  | а  | r  | d  | a  | \$ | а  |    | а  | b         |
| a          | r  | d  | a  | \$ | a  | I  | a  | b  | а  | r  | a  |    | а  |    | а  | b         |
| b          | а  | r  | a  | ı  | a  | 1  | a  | b  | a  | r  | d  | a  | \$ | a  | ı  | a         |
| b          | a  | r  | d  | a  | \$ | а  | I  | a  | b  | a  | r  | a  | I  | a  | I  | a         |
| d          | a  | \$ | a  | ı  | a  | b  | a  | r  | a  | ı  | a  |    | a  | b  | a  | r         |
|            | a  | b  | a  | r  | а  | 1  | а  | 1  | а  | b  | a  | r  | d  | а  | \$ | a         |
|            | а  | b  | a  | r  | d  | а  | \$ | а  | I  | а  | b  | а  | r  | а  |    | a         |
|            | a  | I  | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a         |
| r          | а  | I  | a  |    | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | a         |
| r          | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a         |

| lacksquare |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|
| \$         | а  |    | а  | b  | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a         |
| a          | \$ | a  | I  | a  | b  | a  | r  | a  | ı  | а  |    | a  | b  | a  | r  | d         |
| a          | b  | a  | r  | а  | I  | a  | I  | а  | b  | а  | r  | d  | а  | \$ | а  | ı         |
| a          | b  | а  | r  | d  | а  | \$ | а  | I  | а  | b  | a  | r  | а  |    | a  | ı         |
| a          | 1  | а  | b  | а  | r  | а  | ı  | а  | I  | а  | b  | a  | r  | d  | а  | \$        |
| a          | 1  | а  | b  | а  | r  | d  | a  | \$ | а  |    | а  | b  | а  | r  | а  | ı         |
| a          | 1  | а  | ı  | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r         |
| a          | r  | а  | I  | а  | I  | а  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b         |
| a          | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | а  | I  | а  | I  | а  | b         |
| b          | а  | r  | а  | I  | а  |    | а  | b  | а  | r  | d  | a  | \$ | a  |    | a         |
| b          | а  | r  | d  | а  | \$ | a  | I  | а  | b  | а  | r  | а  | I  | а  | l  | a         |
| d          | а  | \$ | а  | I  | а  | b  | а  | r  | а  |    | a  |    | а  | b  | а  | r         |
|            | а  | b  | а  | r  | а  |    | а  | I  | а  | b  | a  | r  | d  | а  | \$ | a         |
|            | а  | b  | а  | r  | d  | a  | \$ | а  | I  | а  | b  | a  | r  | а  |    | a         |
|            | а  | I  | a  | b  | а  | r  | d  | а  | \$ | а  |    | а  | b  | а  | r  | a         |
| r          | а  | I  | a  | I  | a  | b  | а  | r  | d  | а  | \$ | a  | ı  | а  | b  | a         |
| r          | d  | а  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a         |

| $oxed{F}$ |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|
| \$        | а  | I  | а  | b  | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a         |
| a         | \$ | а  | ı  | а  | b  | а  | r  | а  | ı  | a  |    | a  | b  | a  | r  | d         |
| a         | b  | а  | r  | а  | ı  | a  | l  | а  | b  | a  | r  | d  | a  | \$ | a  | ı         |
| a         | b  | a  | r  | d  | a  | \$ | а  |    | a  | b  | а  | r  | a  | I  | а  | ı         |
| a         |    | а  | b  | а  | r  | a  | 1  | а  | l  | a  | b  | a  | r  | d  | а  | \$        |
| a         |    | а  | b  | а  | r  | d  | а  | \$ | а  | I  | а  | b  | а  | r  | а  | ı         |
| а         | I  | а  | I  | а  | b  | а  | r  | d  | а  | \$ | a  | I  | а  | b  | а  | r         |
| а         | r  | а  | I  | а  | l  | а  | b  | а  | r  | d  | a  | \$ | а  | I  | а  | b         |
| a         | r  | d  | a  | \$ | a  | l  | а  | b  | а  | r  | a  | I  | a  | I  | а  | b         |
| b         | а  | r  | а  |    | а  | l  | а  | b  | а  | r  | d  | a  | \$ | а  | I  | a         |
| b         | a  | r  | d  | а  | \$ | a  | I  | а  | b  | a  | r  | a  | I  | a  | I  | a         |
| d         | а  | \$ | а  |    | a  | b  | а  | r  | а  | I  | a  | I  | а  | b  | а  | r         |
| ı         | a  | b  | а  | r  | a  |    | а  |    | а  | b  | a  | r  | d  | a  | \$ | a         |
| ı         | a  | b  | а  | r  | d  | а  | \$ | а  |    | a  | b  | a  | r  | а  | I  | a         |
| I         | а  |    | а  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b  | а  | r  | a         |
| r         | а  |    | а  |    | а  | b  | a  | r  | d  | а  | \$ | а  |    | а  | b  | a         |
| r         | d  | а  | \$ | а  |    | а  | b  | а  | r  | a  |    | а  |    | а  | b  | a         |

|   | F  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|
|   | \$ | а  |    | а  | b  | а  | r  | а  | I  | а  |    | а  | b  | а  | r  | d  | a         |
| ſ | а  | \$ | а  | I  | a  | b  | a  | r  | а  | I  | а  | I  | a  | b  | a  | r  | d         |
| ı | a  | b  | а  | r  | a  | I  | a  | ı  | а  | b  | a  | r  | d  | a  | \$ | a  |           |
| I | a  | b  | а  | r  | d  | а  | \$ | a  |    | а  | b  | a  | r  | a  |    | а  | ı         |
| П | а  | Ι  | а  | b  | a  | r  | a  | ı  | а  |    | a  | b  | a  | r  | d  | a  | \$        |
| Ш | a  | ı  | а  | b  | a  | r  | d  | a  | \$ | а  | I  | а  | b  | a  | r  | а  |           |
| Ш | a  | ı  | а  | ı  | a  | b  | а  | r  | d  | а  | \$ | а  | I  | a  | b  | а  | r         |
| T | a  | r  | а  |    | a  | I  | a  | b  | а  | r  | d  | а  | \$ | a  |    | a  | b         |
| Ц | a  | r  | d  | а  | \$ | а  | ı  | a  | b  | а  | r  | a  | I  | a  |    | a  | b         |
|   | b  | а  | r  | a  | 1  | a  | ı  | a  | b  | а  | r  | d  | a  | \$ | a  | ı  | a         |
|   | b  | а  | r  | d  | a  | \$ | a  | I  | а  | b  | a  | r  | a  | I  | a  | I  | a         |
|   | d  | а  | \$ | а  |    | а  | b  | а  | r  | а  | I  | а  | I  | a  | b  | a  | r         |
|   | I  | а  | b  | а  | r  | а  | ı  | а  |    | а  | b  | а  | r  | d  | а  | \$ | a         |
| Ш | 1  | а  | b  | а  | r  | d  | а  | \$ | а  | I  | а  | b  | a  | r  | а  | ı  | a         |
| Ц | ı  | a  |    | а  | b  | а  | r  | d  | а  | \$ | а  | I  | а  | b  | а  | r  | a         |
|   | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a  | \$ | a  | I  | а  | b  | a         |
|   | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a         |

|              | F  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | $oxed{L}$ |   |
|--------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------|---|
|              | \$ | а  |    | а  | b  | а  | r  | а  |    | а  |    | а  | b  | а  | r  | d  | a         |   |
|              | a  | \$ | а  | ı  | a  | b  | a  | r  | а  | ı  | а  | I  | a  | b  | а  | r  | d         |   |
|              | a  | b  | а  | r  | а  | I  | a  | 1  | а  | b  | a  | r  | d  | а  | \$ | a  | ı         |   |
|              | a  | b  | а  | r  | d  | a  | \$ | a  | I  | а  | b  | a  | r  | a  | I  | a  | ı         |   |
| ΠŤ           | а  | I  | а  | b  | a  | r  | a  | 1  | a  | l  | a  | b  | a  | r  | d  | a  | \$        |   |
| Ш            | a  | ı  | а  | b  | a  | r  | d  | а  | \$ | а  | I  | a  | b  | a  | r  | a  | ı         | k |
| Ш            | a  | 1  | а  | ı  | а  | b  | а  | r  | d  | а  | \$ | а  | I  | a  | b  | а  | r         |   |
| $\mathbf{I}$ | a  | r  | а  | I  | а  | I  | a  | b  | а  | r  | d  | а  | \$ | а  | I  | а  | b         |   |
| Ц            | a  | r  | d  | a  | \$ | a  | ı  | a  | b  | a  | r  | a  | I  | a  | I  | a  | b         |   |
|              | b  | а  | r  | а  | I  | a  | ı  | а  | b  | а  | r  | d  | a  | \$ | а  | ı  | a         |   |
|              | b  | а  | r  | d  | a  | \$ | a  | I  | а  | b  | a  | r  | a  | I  | а  | ı  | a         |   |
|              | d  | а  | \$ | а  | I  | а  | b  | а  | r  | а  | I  | а  | I  | а  | b  | а  | r         | 1 |
| Π            | I  | а  | b  | а  | r  | а  | ı  | a  | I  | а  | b  | а  | r  | d  | а  | \$ | a         |   |
|              |    | а  | b  | a  | r  | d  | a  | \$ | а  |    | а  | b  | a  | r  | а  |    | a         |   |
|              | ı  | а  |    | а  | b  | а  | r  | d  | а  | \$ | а  | l  | a  | b  | а  | r  | a         |   |
|              | r  | а  |    | a  | I  | а  | b  | а  | r  | d  | а  | \$ | a  | ı  | а  | b  | а         |   |
|              | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | a  | b  | a         |   |

|   | F  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |     | L  |   |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|----|---|
| ĺ | \$ | а  |    | а  | b  | а  | r  | a  |    | а  |    | а  | b  | a  | r  | d   | a  |   |
|   | a  | \$ | a  |    | a  | b  | a  | r  | a  | I  | a  |    | a  | b  | a  | r   | d  |   |
|   | a  | b  | a  | r  | а  | I  | а  | I  | а  | b  | а  | r  | d  | а  | \$ | а   | ı  |   |
|   | a  | b  | а  | r  | d  | а  | \$ | а  | I  | a  | b  | a  | r  | a  | I  | а   | ı  |   |
| П | а  | T  | a  | b  | a  | r  | а  | I  | а  | I  | a  | b  | a  | r  | d  | a   | \$ |   |
|   | a  | I  | а  | b  | а  | r  | d  | а  | \$ | a  | I  | а  | b  | а  | r  | a   | ı  |   |
|   | a  | ı  | a  | I  | a  | b  | а  | r  | d  | a  | \$ | a  |    | а  | b  | a   | r  |   |
| I | a  | r  | a  |    | а  | I  | а  | b  | а  | r  | d  | а  | \$ | а  | I  | a   | b  |   |
|   | a  | r  | d  | a  | \$ | a  | I  | а  | b  | a  | r  | a  | I  | a  | I  | a   | b  |   |
|   | b  | а  | r  | а  |    | а  | I  | а  | b  | а  | r  | d  | а  | \$ | а  | I   | a  |   |
|   | b  | a  | r  | d  | a  | \$ | а  | I  | а  | b  | a  | r  | а  | ı  | а  | ı   | a  |   |
|   | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  | ı  | a  |    | a  | b  | a   | r  |   |
|   | I  | a  | b  | а  | r  | а  | 1  | а  | 1  | a  | b  | а  | r  | d  | а  | \$  | a  |   |
|   | ı  | a  | b  | a  | r  | d  | а  | \$ | а  | I  | a  | b  | a  | r  | а  | I   | a  |   |
|   | I  | a  |    | а  | b  | a  | r  | d  | a  | \$ | a  |    | a  | b  | a  | r   | а  | П |
|   | r  | а  |    | а  |    | а  | b  | а  | r  | d  | а  | \$ | а  | ı  | а  | b ' | а  |   |
|   | r  | d  | a  | \$ | a  |    | a  | b  | a  | r  | a  |    | a  |    | а  | b   | a  |   |

Search for *lala* 1 occ

Representing L with a wavelet tree

count:  $O(m \log \sigma)$ 

extract:  $O((s_t + \ell) \log \sigma))$ 

locate:  $O\left((m + s_a occ) \log \sigma\right)$ 

Representing L with a wavelet tree

count:  $O(m \log \sigma)$ 

extract:  $O((s_t + \ell) \log \sigma))$ 

locate:  $O\left((m + s_a occ) \log \sigma\right)$ 

space:  $nH_k(T) + o(n\log\sigma)$ 

Improving what we already have

Improving what we already have

Construction of these structures

Improving what we already have

Construction of these structures

Other technical problems

Improving what we already have

Construction of these structures

Other technical problems

LIBCDS2!

#### Links

- http://libcds.recoded.cl/
- https://github.com/fclaude/libcds
- https://github.com/fclaude/libcds2

Thanks for your attention

The End

