MADALGO

CENTER FOR MASSIVE DATA ALGORITHMICS

Ph.d. Defense

Allan Grønlund Jørgensen

April 29th - 2010

DANMARKS GRUNDFORSKNINGSFOND
DANISH NATIONAL RESEARCH FOUNDATION

ARHUS UNIVERSITY

Outline

- My Work
- Range Queries
- Range Mode
- Range Selection

My Work

A Linear Time Algorithm for the k Maximal Sums Problem (MFCS'07)
Selecting Sums in Arrays (ISAAC'08)

Data Structures for Range Median Queries (ISAAC'09)

Cell Probe Lower Bounds and Approximations
for Range Mode (ICALP'10)

Priority Queues Resilient to Memory Faults (WADS'07)
Optimal Resillient Dictionaries (ESA'07)

Fault Tolerant External Memory Algorithms (WADS'09)

Counting in the Presence of Memory Faults (ISAAC'09)

My Work

A Linear Time Algorithm for the k Maximal Sums Problem (MFCS'07)
Selecting Sums in Arrays (ISAAC'08)

Data Structures for Range Median Queries (ISAAC'09)

Cell Probe Lower Bounds and Approximations
for Range Mode (ICALP'10)

Range Median and Selection: Cell Probe Lower Bounds
and Adaptive Data Structures (In submission)

Priority Queues Resilient to Memory Faults (WADS'07)
Optimal Resillient Dictionaries (ESA'07)

Fault Tolerant External Memory Algorithms (WADS'09)

Counting in the Presence of Memory Faults (ISAAC'09)

Geometric Computations on Indecisive Points (submitted soon)

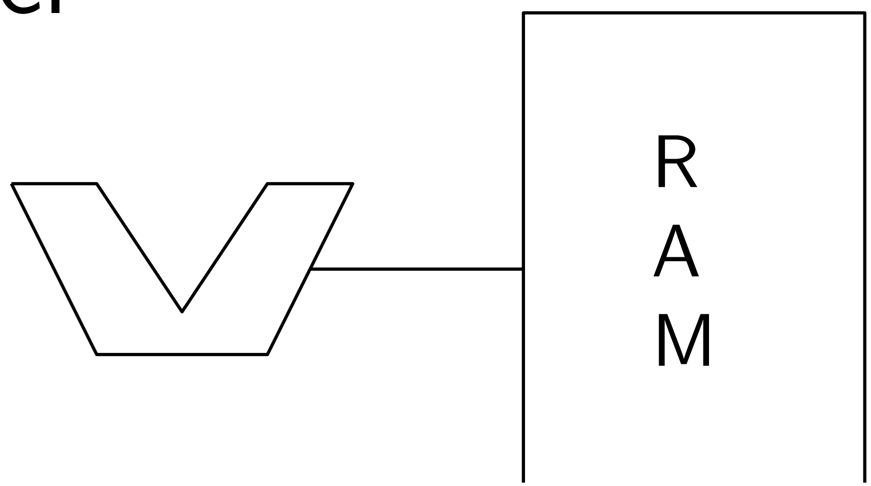
This Talk

Data Structures for Range Median Queries
with Gerth S. Brodal (ISAAC'09)

Cell Probe Lower Bounds and Approximations for Range Mode
With Mark Greve, Kasper D. Larsen and Jakob Truelsen (ICALP'10)

Range Median and Selection: Cell Probe Lower Bounds
and Adaptive Data Structures
with Kasper D. Larsen (In submission)

RAM model



- memory of w bit cells
- each memory accces takes constant time
- word operations in constant time (arithmetic, shifts etc.)
- $\log n = \Theta(w)$

Range Queries

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad \cdots \quad \mid a_j \mid \quad \cdots \quad \mid a_n}$$

Range Queries

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Data Structure
Given indices i, j compute $f(A[i..j])$

Range Queries

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline\end{array}$$

Data Structure

Given indices i, j compute $f(A[i..j])$

Range Queries

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Data Structure
Given indices i, j compute $f(A[i..j])$

- Query Time
- Space
- Preprocessing Time

Range Queries

$$A = \begin{array}{|c|c|c|c|c|c|c|c|} \hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline \end{array}$$

Data Structure

Given indices i, j compute $f(A[i..j])$

- Query Time
- Space
- Preprocessing Time

Dynamic - Not discussed today

Functions

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \cdots \mid a_j \mid \cdots \mid a_n}$$

Functions

$$A = \boxed{a_1}\boxed{a_2}\boxed{\cdots}\boxed{a_i}\boxed{\quad\cdots\quad}\boxed{a_j}\boxed{\quad\cdots\quad}\boxed{a_n}$$

- Average

Functions

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \cdots \mid a_j \mid \cdots \mid a_n}$$

- Average
- Minimum/Maximum

Functions

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline\end{array}$$

- Average
- Minimum/Maximum
- Median (this talk)

Functions

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline \end{array}$$

- Average
- Minimum/Maximum
- Median (this talk)
- k'th smallest (this talk)

Functions

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

- Average
- Minimum/Maximum
- Median (this talk)
- k'th smallest (this talk)
- Rank of e

Functions

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline \end{array}$$

- Average
- Minimum/Maximum
- Median (this talk)
- k'th smallest (this talk)
- Rank of e
- Mode (this talk)

Example

$$[-1, -2, 0, 0, 0, 1, 0, -5, -1, -2, -1, 0, 4, 0, -2, -1, -1, -3]$$

Example

$$[-1, -2, 0, 0, 0, 1, 0, -5, -1, -2, -1, 0, 4, 0, -2, -1, -1, -3]$$

$$[0, 0, 1, 1, 1, 3, 1, 0, 0, 0, 0, 1, 3, 1, 0, 0, 0, 0]$$

Example

Goal Difference in AGF games

Nov. Dec. March April

$$[-1,-2,0,0,0,1,0,-5,-1,-2,-1,0,4,0,-2,-1,-1,-3]$$

$$[0,0,1,1,1,3,1,0,0,0,0,1,3,1,0,0,0,0]$$

Example

Goal Difference in AGF games

Nov. Dec. March April

$$[-1, -2, 0, 0, 0, 1, 0, -5, -1, -2, -1, 0, 4, 0, -2, -1, -1, -3]$$

$$[0, 0, 1, 1, 1, 3, 1, 0, 0, 0, 0, 1, 3, 1, 0, 0, 0, 0]$$

March:

Average: -5/6

Min: -5

Max: 4

Median: -1

Rank of -1: 4

Mode: -1

Words on Space

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

There are only $O(n^2)$ ranges

Words on Space

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

There are only $O(n^2)$ ranges

There are only queries $O(n^2)$ ($O(n^3)$ for selection and rank)

Words on Space

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

There are only $O(n^2)$ ranges

There are only queries $O(n^2)$ ($O(n^3)$ for selection and rank)

Table with $O(n^2)$ entries $O(1)$ query time

Words on Space

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

There are only $O(n^2)$ ranges

There are only queries $O(n^2)$ ($O(n^3)$ for selection and rank)

Table with $O(n^2)$ entries $O(1)$ query time

$O(n^2)$ space is BAD!!!!!

Words on Space

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

There are only $O(n^2)$ ranges

There are only queries $O(n^2)$ ($O(n^3)$ for selection and rank)

Table with $O(n^2)$ entries $O(1)$ query time

$O(n^2)$ space is BAD!!!!!

Near-Linear Space: $n \log^{O(1)} n$
Polynomial Space: $n^{1+O(1)}$

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline \end{array}$$

Given i, j return a most frequently occuring element in $A[i, j]$

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Given i, j return a most frequently occuring element in $A[i, j]$

Example:

A = [a b b a a b a b b a a b]
1 2 3 4 5 6 7 8 9 . . .

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Given i, j return a most frequently occuring element in $A[i, j]$

Example:

A = [a b b a a b a b b a a b]
1 2 3 4 5 6 7 8 9 . . .

mode$(A[1..5]) = a$

Range Mode

$A =$

a_1	a_2	$\cdots$	a_i	$\cdots$	a_j	$\cdots$	a_n

Given i, j return a most frequently occuring element in $A[i, j]$

Example:

A = [a b b a a b a b b a a b]
1 2 3 4 5 6 7 8 9 . . .

mode$(A[1..5]) = a$

mode$(A[2..6]) = b$

Range Mode - Upper Bounds

	Time	Space
Only Store Input	$O(n)$	$O(n)$
Complete tabulation	$O(1)$	$O(n^2)$
Bit tricks	$O(1)$	$O(\frac{n^2 \log\log n}{\log^2 n})$
Tradeoff	$O(n^\varepsilon)$	$O(n^{2-2\varepsilon})$

[Petersen 08, Petersen, Grabowski 08]

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Ideally $n \log^{O(1)} n$ space and $\log^{O(1)} n$ time.

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline\end{array}$$

Ideally $n \log^{O(1)} n$ space and $\log^{O(1)} n$ time.

We have tried and failed :o(

Range Mode

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline \end{array}$$

Ideally $n \log^{O(1)} n$ space and $\log^{O(1)} n$ time.

We have tried and failed :o(

We were able to approximate efficiently

	Time	Space
3-approximation	$O(1)$	$O(n)$
$(1+\varepsilon)$-approximation	$O(\log \frac{1}{\varepsilon})$	$O(\frac{n}{\varepsilon})$

What About Lower Bounds?

Cell Probe Model

- Memory of w bit cells
- Complexity is the number of memory probes
- All other computation is free
- Strong enough to include RAM model

Cell Probe Model

- Memory of w bit cells
- Complexity is the number of memory probes
- All other computation is free
- Strong enough to include RAM model

Use strategy initiated by [Miltersen et. al. 93]
refined by Pătrașcu and Thorup

Lower Bound

Range Mode:

Data structure of size S needs $\Omega(\frac{\log n}{\log(Sw/n)})$ time for a query

Lower Bound

Range Mode:

Data structure of size S needs $\Omega(\frac{\log n}{\log(Sw/n)})$ time for a query

Near-Linear space data structures need $\Omega(\log n / \log \log n)$ time

Lower Bound

Range Mode:

Data structure of size S needs $\Omega(\frac{\log n}{\log(Sw/n)})$ time for a query

Near-Linear space data structures need $\Omega(\log n / \log\log n)$ time

Constant query time data structures need $n^{1+\Omega(1)}$ space

Communication Complexity

Alice

Bob

Communication Complexity

Predetermined function f

Input X

Alice

Input Y

Bob

Communication Complexity

Predetermined function f

Input X

Alice

Compute $f(X, Y)$

Minimize communication

Input Y

Bob

Simulating Data Structures

Reduce f to N/k queries on data structure.

Simulating Data Structures

Reduce f to N/k queries on data structure.

Map Alice's input to queries and Bobs to data structure input

Simulating Data Structures

Reduce f to N/k queries on data structure.

Map Alice's input to queries and Bobs to data structure input

Data Structure with query time t and space S gives protocol:

Simulating Data Structures

Reduce f to N/k queries on data structure.

Map Alice's input to queries and Bobs to data structure input

Data Structure with query time t and space S gives protocol:

Bob builds data structure

Alice simulates query algorithm for all queries in parallel

Adress(es) $\Longrightarrow$

$\Longleftarrow$ Contents

Simulating Data Structures

Reduce f to N/k queries on data structure.

Map Alice's input to queries and Bobs to data structure input

Data Structure with query time t and space S gives protocol:

Bob builds data structure

Alice simulates query algorithm for all queries in parallel

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N)$ bits

Bob sends twN/k bits

Adress(es) $\Longrightarrow$

$\Longleftarrow$ Contents

Blocked Lopsided Set Disjointness

Universe $U = [1, 2, \ldots, n] \times [1, 2, \ldots, B]$

Blocked Lopsided Set Disjointness

Universe $U = [1, 2, \ldots, n] \times [1, 2, \ldots, B]$

$X = \{(1, B_1), \ldots, (N, B_N)\}$

$Y \subseteq U$

Blocked Lopsided Set Disjointness

Universe $U = [1, 2, \ldots, n] \times [1, 2, \ldots, B]$

$X = \{(1, B_1), \ldots, (N, B_N)\}$ $Y \subseteq U$

Determine if $X \cap Y \neq \emptyset$

Blocked Lopsided Set Disjointness

Universe $U = [1, 2, \ldots, n] \times [1, 2, \ldots, B]$

$X = \{(1, B_1), \ldots, (N, B_N)\}$

$Y \subseteq U$

Determine if $X \cap Y \neq \emptyset$

Either Alice sends $\Omega(N \log B)$
or Bob sends $\Omega(NB^{1-\delta})$ bits
[Miltersen et al. 93, Pătrașcu 08]

Lower Bound for Range Mode

Lower bound on determining the frequency of the mode

Lower Bound for Range Mode

Lower bound on determining the frequency of the mode

Reduce Blocked LSD to N/k queries

Lower Bound for Range Mode

Lower bound on determining the frequency of the mode

Reduce Blocked LSD to N/k queries

Map X to N/k queries

Map Y to input, $n = O(NB)$

Lower Bound for Range Mode

Lower bound on determining the frequency of the mode

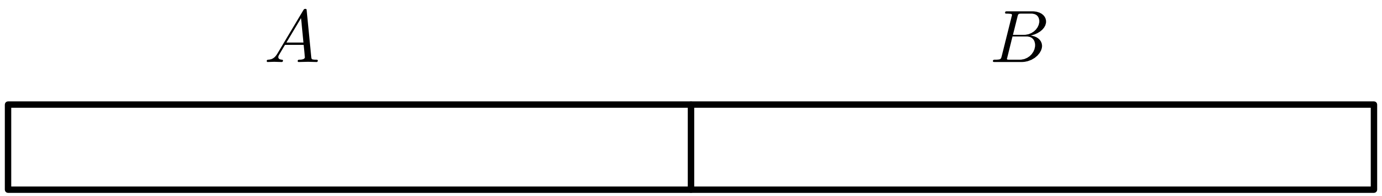
Reduce Blocked LSD to N/k queries

Map X to N/k queries

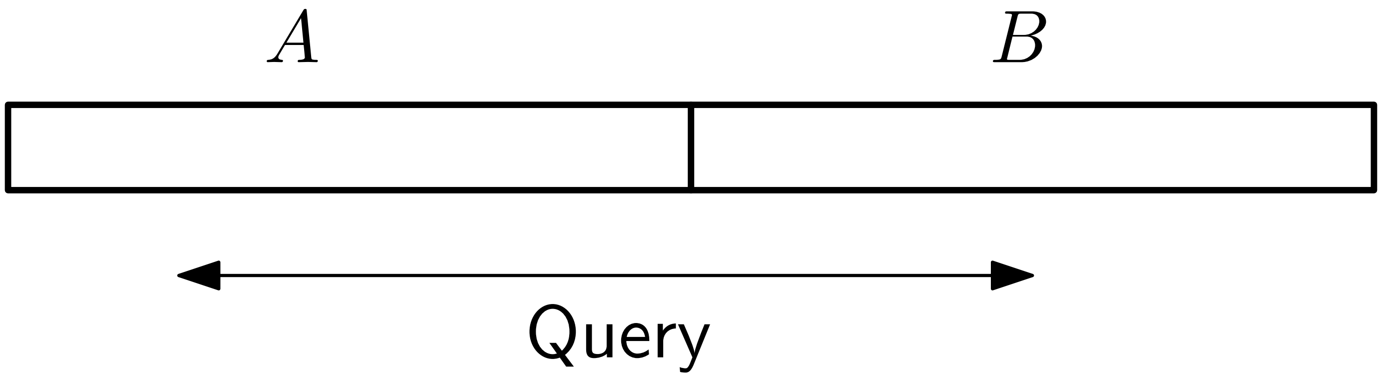
Map Y to input, $n = O(NB)$

The answer to the N/k queries determine whether $X \cap Y \neq \emptyset$

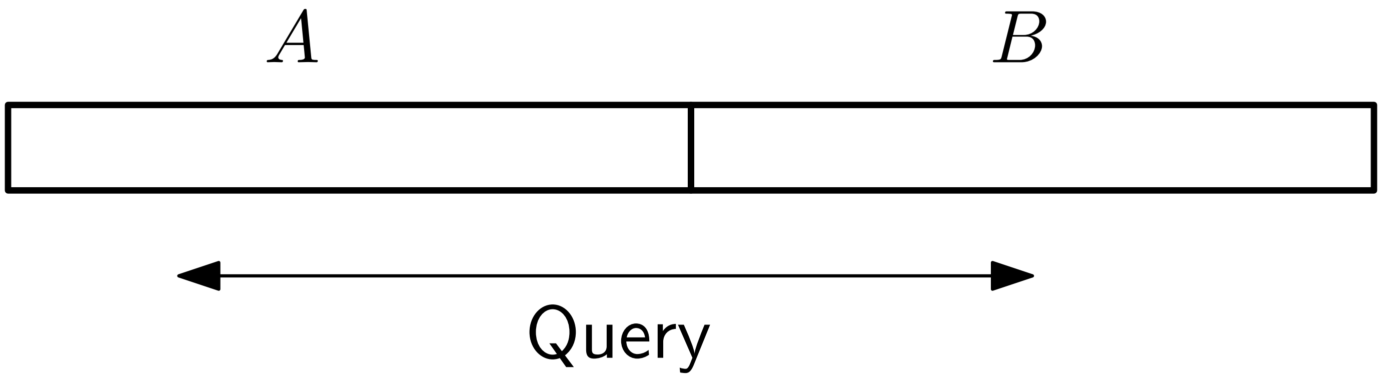
Lower Bound - Intuition



Lower Bound - Intuition



Lower Bound - Intuition



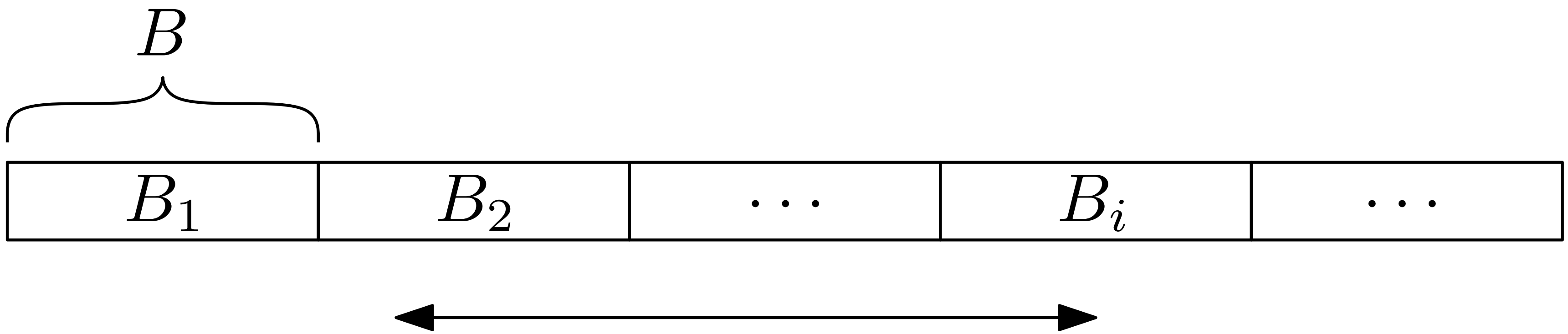
Solve set disjointness on A and B!!!

Lower Bound - Intution cont.

$$\overbrace{\boxed{\quad B_1 \quad}}^{B}\boxed{\quad B_2 \quad}\boxed{\quad \cdots \quad}\boxed{\quad B_i \quad}\boxed{\quad \cdots \quad}$$

Each block is a permutation of $[1, 2, \ldots, B]$

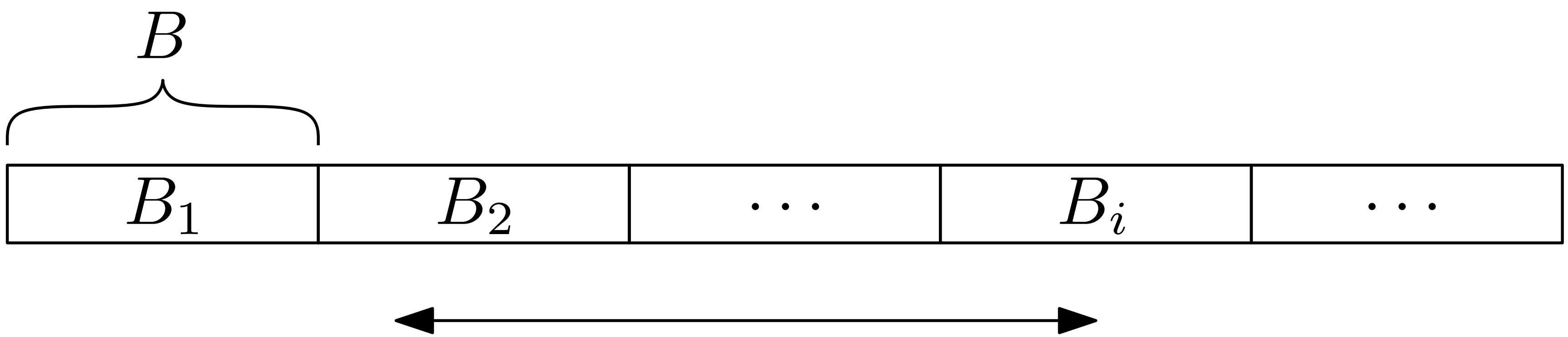
Lower Bound - Intution cont.



Each block is a permutation of $[1, 2, \ldots, B]$

Query answers set disjointness on subsets of two blocks

Lower Bound - Intution cont.



Each block is a permutation of $[1, 2, \ldots, B]$

Query answers set disjointness on subsets of two blocks

First block is subset of Alice's input

Second block is subset of Bob's input

Lower Bound Construction

$$U = [N] \times [B]$$

Split $[N]$ into subsets of k elm.

$$X = \{(i, B_i)\}_{i=1,\ldots,N} \qquad Y = \{(i, b)\}$$

$X_1 = \{(i, B_i\}_{i=1,\ldots,k}$ $\qquad$ $Y_1 = Y \cap \{1, \ldots, k\} \times [B]$

$X_2 = \{(i, B_i\}_{i=k+1,\ldots,2k}$ $\qquad$ $Y_2 = Y \cap [k+1, \ldots, 2k] \times [B]$

$\vdots$ $\qquad$ $\vdots$

$X_{N/k} = \{(i, B_i)\}_{i=N/k+1,\ldots,N}$ $\qquad$ $Y_{N/k} = Y \cap [N/k+1, \ldots, N] \times [B]$

Lower Bound cont.

Make an array (range mode input) of two parts

Part 1: B^k options Alice blocks are made of $P_1, \ldots, P_{B^k}$

Part 2: Bobs input

Lower Bound cont.

Make an array (range mode input) of two parts

Part 1: B^k options Alice blocks are made of $P_1, \ldots, P_{B^k}$

Part 2: Bobs input

$m : [k] \times [B] \rightarrow [kb]$ (for $[ik+1, (i+1)k] \times B$ modulo with k)

Lower Bound cont.

Make an array (range mode input) of two parts
Part 1: B^k options Alice blocks are made of $P_1, \ldots, P_{B^k}$
Part 2: Bobs input

$m : [k] \times [B] \rightarrow [kb]$ (for $[ik+1, (i+1)k] \times B$ modulo with k)

Part 2: $[m(Y_i), [kb] \setminus m(Y_i)]_{i=1,\ldots,N/k}$

$[kb] = [1, 8]$ and $m(Y_1) = \{1, 4, 6, 7\}$ then $\{1, 4, 6, 7, 2, 3, 5, 8\}$

Lower Bound cont.

Make an array (range mode input) of two parts
Part 1: B^k options Alice blocks are made of $P_1, \dots, P_{B^k}$
Part 2: Bobs input

$m : [k] \times [B] \to [kb]$ (for $[ik+1, (i+1)k] \times B$ modulo with k)

Part 2: $[m(Y_i), [kb] \setminus m(Y_i)]_{i=1,\dots,N/k}$

$[kb] = [1, 8]$ and $m(Y_1) = \{1, 4, 6, 7\}$ then $\{1, 4, 6, 7, 2, 3, 5, 8\}$

Part 1: $[[kb] \setminus m(P_i), m(P_i)]_{i=1,\dots,B^k}$

Lower Bound cont.

Make an array (range mode input) of two parts
Part 1: B^k options Alice blocks are made of $P_1, \ldots, P_{B^k}$
Part 2: Bobs input

$m : [k] \times [B] \to [kb]$ (for $[ik+1,(i+1)k] \times B$ modulo with k)

Part 2: $[m(Y_i), [kb] \setminus m(Y_i)]_{i=1,\ldots,N/k}$

$[kb] = [1,8]$ and $m(Y_1) = \{1,4,6,7\}$ then $\{1,4,6,7,2,3,5,8\}$

Part 1: $[[kb] \setminus m(P_i), m(P_i)]_{i=1,\ldots,B^k}$

Range Mode Input: Part 1 followed by Part 2

Notice that Alice knows Part 1

Lower Bound cont.

Bob sends $|Y_i|$ for $i = 1, \ldots, N/k$

Lower Bound cont.

Bob sends $|Y_i|$ for $i = 1, \ldots, N/k$

Alice maps each X_i to query

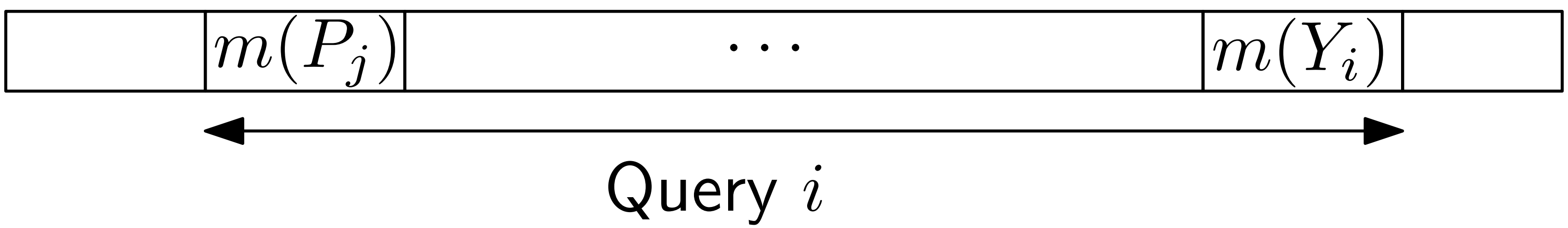
Compute j such that $P_j = X_i$

Lower Bound cont.

Bob sends $|Y_i|$ for $i = 1, \ldots, N/k$

Alice maps each X_i to query

Compute j such that $P_j = X_i$

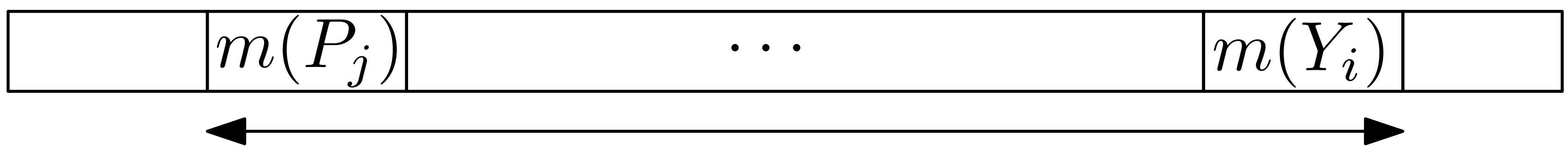


Lower Bound cont.

Bob sends $|Y_i|$ for $i = 1, \ldots, N/k$

Alice maps each X_i to query

Compute j such that $P_j = X_i$



Query i

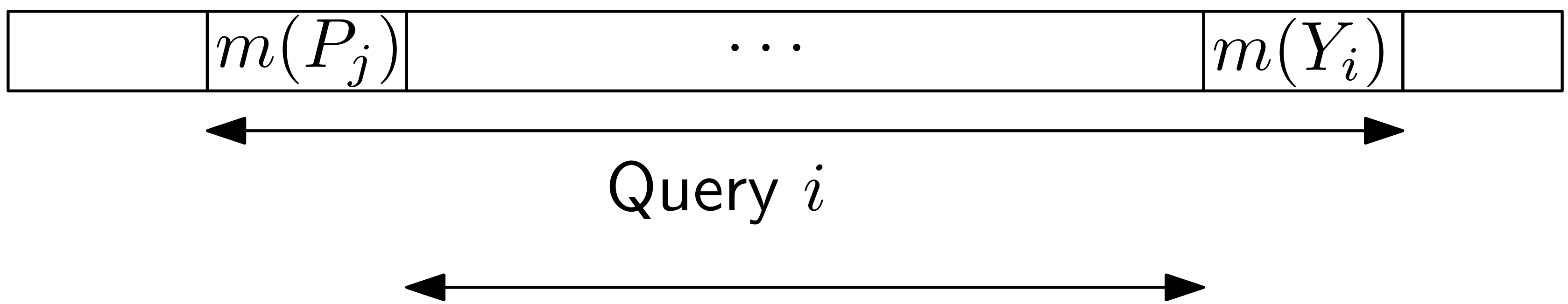
Alice knows that number of blocks here

Lower Bound cont.

Bob sends $|Y_i|$ for $i = 1, \ldots, N/k$

Alice maps each X_i to query

Compute j such that $P_j = X_i$



Alice knows that number of blocks here

$+2$ iff $X_i \cap Y_i \neq \emptyset$ and $+1$ otherwise

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

$N/k \log(Bk) = o(NB^{1/2})$ so $twN/k = \Omega(NB^{1/2})$

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

$N/k \log(Bk) = o(NB^{1/2})$ so $twN/k = \Omega(NB^{1/2})$

Constrain $B > w^2$ and $\log B \geq 1/2 \log(Sk/N)$ then $t = \Omega(k)$

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

$N/k \log(Bk) = o(NB^{1/2})$ so $twN/k = \Omega(NB^{1/2})$

Constrain $B > w^2$ and $\log B \geq 1/2 \log(Sk/N)$ then $t = \Omega(k)$

Bobs array is of size $NB + B^k kB$

Make it $O(n)$ by setting $k = O(\log_B n)$

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

$N/k \log(Bk) = o(NB^{1/2})$ so $twN/k = \Omega(NB^{1/2})$

Constrain $B > w^2$ and $\log B \geq 1/2 \log(Sk/N)$ then $t = \Omega(k)$

Bobs array is of size $NB + B^k kB$

Make it $O(n)$ by setting $k = O(\log_B n)$

Maximize k and set $B = \max\{w^2, Sk/n\}$

Lower Bound

Alice sends $t \log \binom{S}{N/k} = O(tk \log(Sk/N) = \Omega(N \log B)$ bits

Or

Bob sends $twN/k + O(N/k \log(Bk)) = \Omega(NB^{1/2})$ bits

$N/k \log(Bk) = o(NB^{1/2})$ so $twN/k = \Omega(NB^{1/2})$

Constrain $B > w^2$ and $\log B \geq 1/2 \log(Sk/N)$ then $t = \Omega(k)$

Bobs array is of size $NB + B^k kB$

Make it $O(n)$ by setting $k = O(\log_B n)$

Maximize k and set $B = \max\{w^2, Sk/n\}$

Obtain $t = \Omega(k) = \Omega(\log n / \log(Sw/n)$

Decision Version of Range Mode

Range k-frequency

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad\cdots\quad \mid a_j \mid \quad\cdots\quad \mid a_n}$$

Decision Version of Range Mode

Range k-frequency

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline\end{array}$$

Given i, j return whether there is an element occuring exactly k times in $A[i, j]$

Decision Version of Range Mode

Range k-frequency

$A =$

a_1	a_2	$\cdots$	a_i	$\cdots$	a_j	$\cdots$	a_n

Given i, j return whether there is an element occuring exactly k times in $A[i, j]$

Space	Query	k
S	$\Omega(\log n / \log(Sw/n))$	$2 \leq k = O(1)$
$O(n)$	$O(\log n / \log \log n)$	any
$O(n \log n)$	$O(\log^2 \log n)$	$k = 1$

Reduction to Rectangle Stabbing

$$k = 3$$

$\dots$	x	$\dots$	x	$\dots$	x	$\dots$	x	$\dots$	x	$\dots$

Reduction to Rectangle Stabbing

$$k = 3$$

$$[x_1, x_2]\ [x_3, x_4]\ [x_5, x_6]\ [x_7, x_8]\ [x_9, x_{10}][x_{11}, x_{12}]$$

$\dots$	x	$\dots$	x	$\dots$	x	$\dots$	x	$\dots$	x	$\dots$

Reduction to Rectangle Stabbing

$$k = 3$$

$$[x_1, x_2]\ [x_3, x_4]\ [x_5, x_6]\ [x_7, x_8]\ [x_9, x_{10}][x_{11}, x_{12}]$$

$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$

$$[x_1, x_2] \times [x_7, x_8]$$

Reduction to Rectangle Stabbing

$$k = 3$$

$$[x_1, x_2]\ [x_3, x_4]\ [x_5, x_6]\ [x_7, x_8]\ [x_9, x_{10}][x_{11}, x_{12}]$$

$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$	x	$\ldots$

$[x_1, x_2] \times [x_7, x_8]$ $[x_3, x_4] \times [x_9, x_{10}]$

Reduction to Rectangle Stabbing

$$k = 3$$

$$[x_1, x_2]\ [x_3, x_4]\ [x_5, x_6]\ [x_7, x_8]\ [x_9, x_{10}][x_{11}, x_{12}]$$

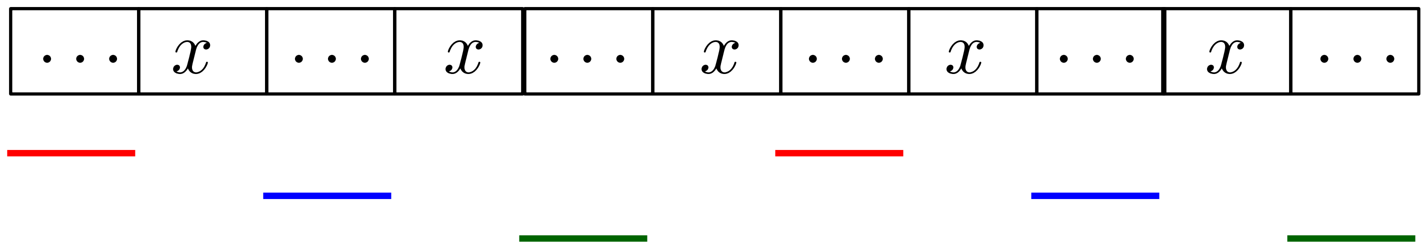
$\cdots$	x	$\cdots$	x	$\cdots$	x	$\cdots$	x	$\cdots$	x	$\cdots$

$$[x_1, x_2] \times [x_7, x_8] \quad [x_3, x_4] \times [x_9, x_{10}] \quad [x_5, x_6] \times [x_{11}, x_{12}]$$

Reduction to Rectangle Stabbing

$$k = 3$$

$$[x_1, x_2]\ [x_3, x_4]\ [x_5, x_6]\ [x_7, x_8]\ [x_9, x_{10}][x_{11}, x_{12}]$$



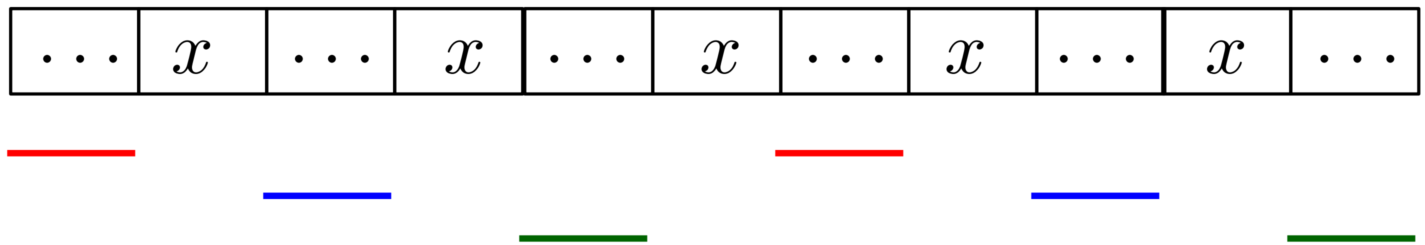
$$[x_1, x_2] \times [x_7, x_8] \quad [x_3, x_4] \times [x_9, x_{10}] \quad [x_5, x_6] \times [x_{11}, x_{12}]$$

Input gives $O(n)$ rectangles

Reduction to Rectangle Stabbing

$$k = 3$$

$[x_1, x_2]$ $[x_3, x_4]$ $[x_5, x_6]$ $[x_7, x_8]$ $[x_9, x_{10}]$ $[x_{11}, x_{12}]$



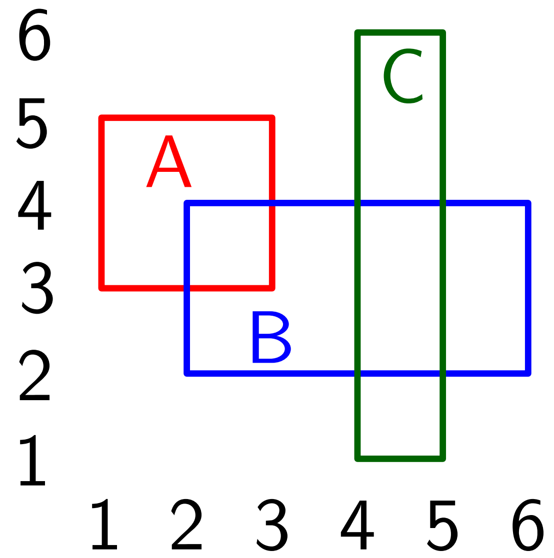
$[x_1, x_2] \times [x_7, x_8]$ $[x_3, x_4] \times [x_9, x_{10}]$ $[x_5, x_6] \times [x_{11}, x_{12}]$

Input gives $O(n)$ rectangles

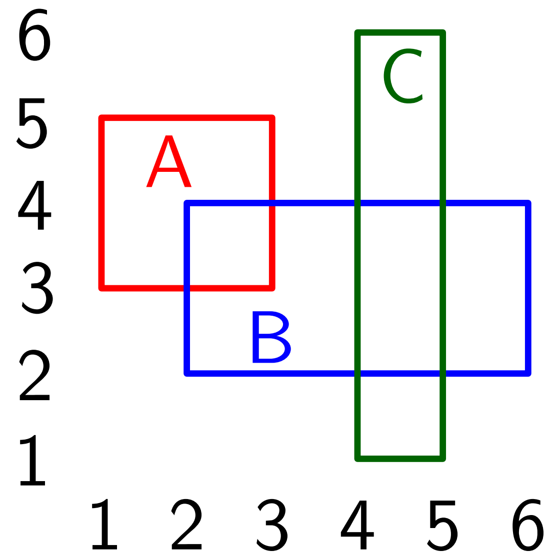
$O(n)$ space $O(\log n / \log \log n)$ query time

[JáJá, Mortensen, Shi 04]

Reduction from Rectangle Stabbing

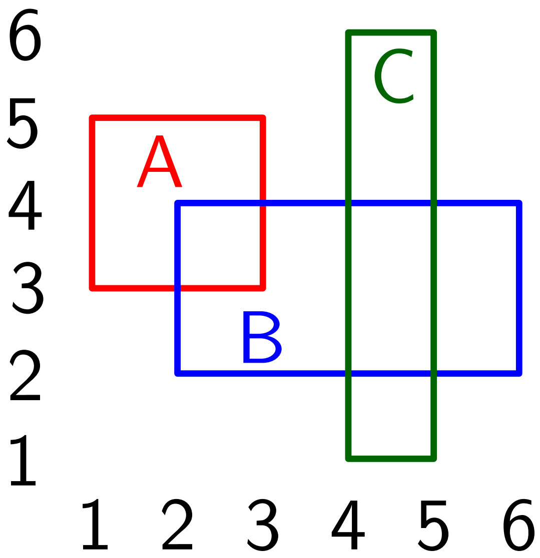


Reduction from Rectangle Stabbing



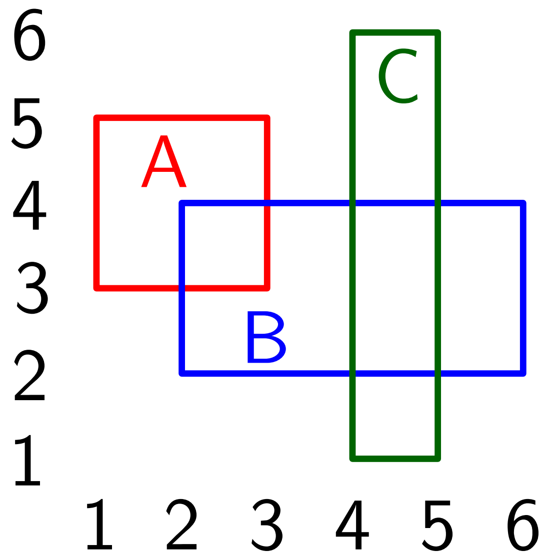
A A

Reduction from Rectangle Stabbing



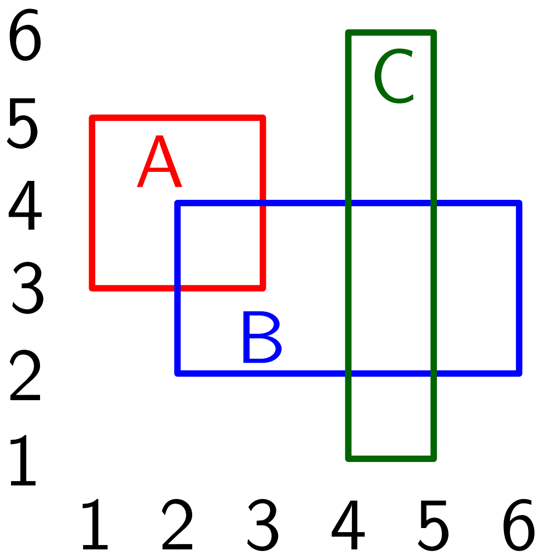
A A B B

Reduction from Rectangle Stabbing



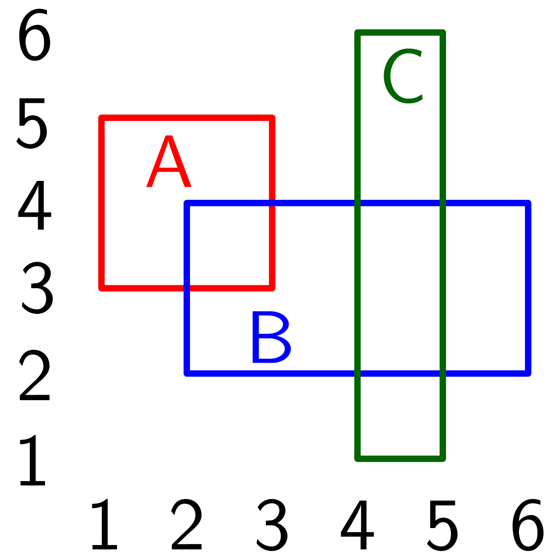
A A B B A

Reduction from Rectangle Stabbing



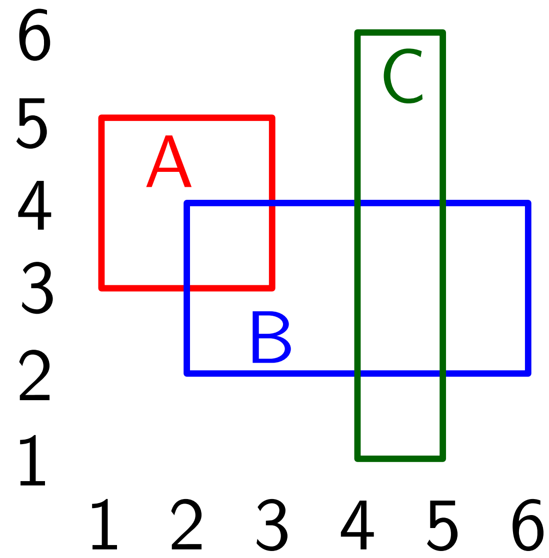
A A B B A C

Reduction from Rectangle Stabbing



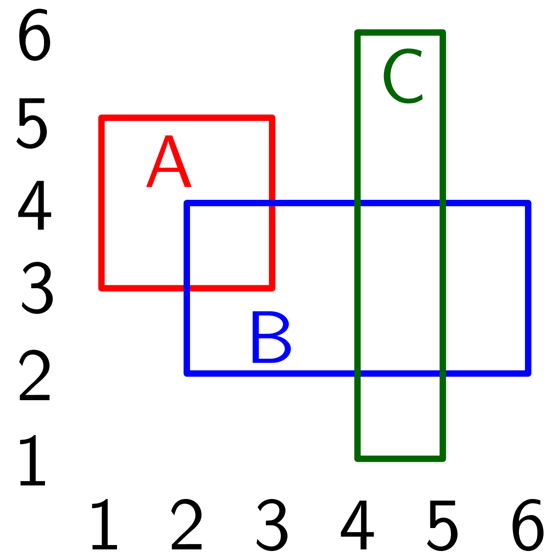
A A B B A C C C

Reduction from Rectangle Stabbing



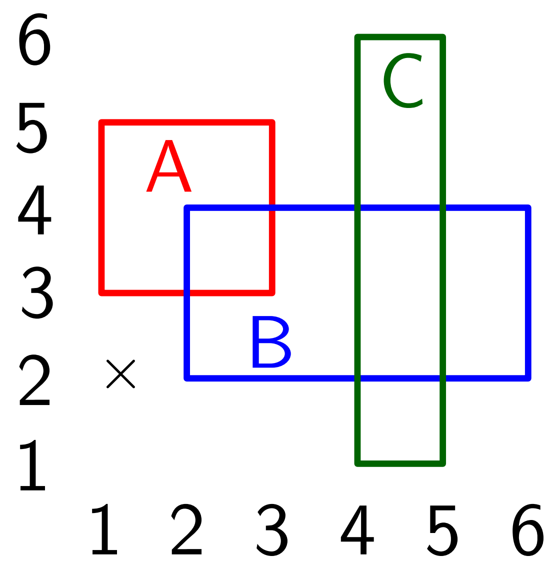
A A B B A C C C B

Reduction from Rectangle Stabbing



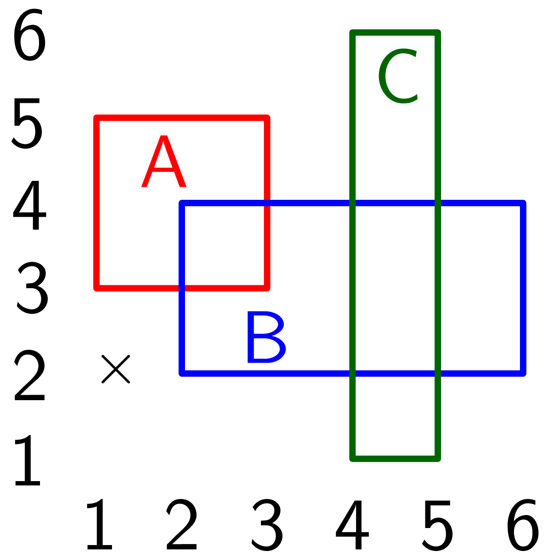
A A B B A C C C B C A B A A B B C C

Reduction from Rectangle Stabbing



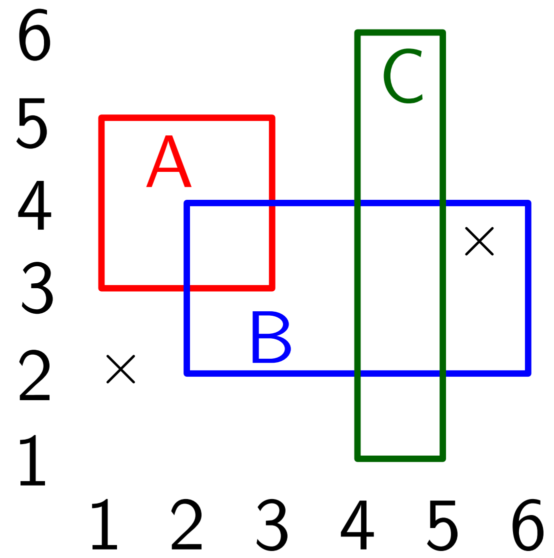
A A B B A C C C B C A B A A B B C C

Reduction from Rectangle Stabbing



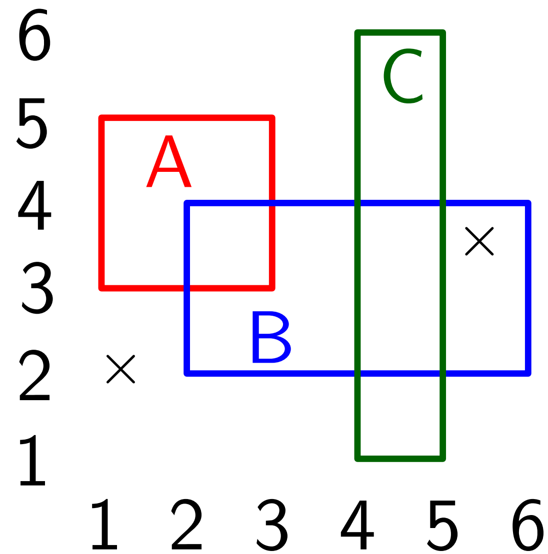
A A B B A C C C B C A B A A B B C C

Reduction from Rectangle Stabbing

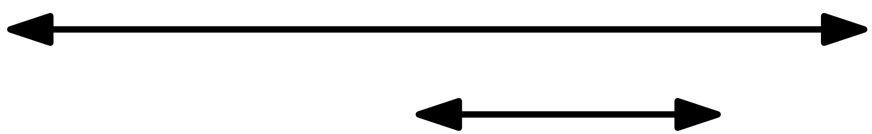


A A B B A C C C B C A B A A B B C C

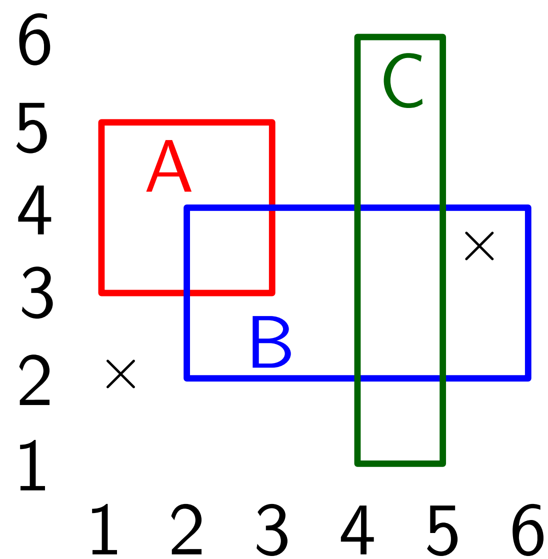
Reduction from Rectangle Stabbing



A A B B A C C C B C A B A A B B C C



Reduction from Rectangle Stabbing



A A B B A C C C B C A B A A B B C C

Space S needs $\Omega(\log n / \log(Sw/n))$ time for a query.

Pătrașcu 08]

Range Mode Open Problem

	Space	Query
Range Mode	$O(n^{2-2\varepsilon})$	$O(n^{\varepsilon})$
Range Mode	S	$\Omega(\log n / \log(Sw/n))$
Range k freq.	$O(n)$	$O(\log n / \log\log n)$
Range $2 \leq k$ freq.	S	$\Omega(\log n / \log(Sw/n))$
Range 1 freq.	$O(n \log n)$	$O(\log^2 \log n)$

Close the gap betwen upper bound and lower bound

Range Selection

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline \end{array}$$

Given i, j, k return the k'th smallest element in $A[i, j]$

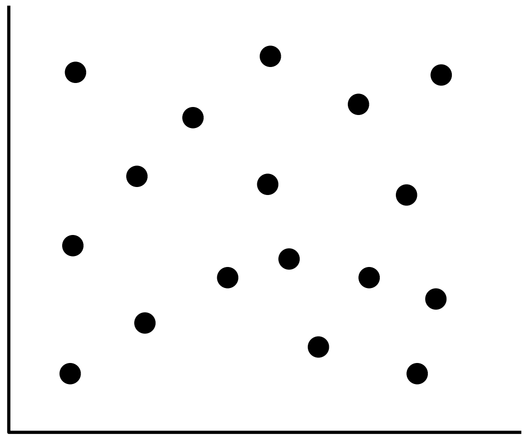
Range Selection

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_j & \cdots & a_n \\ \hline \end{array}$$

Given i, j, k return the k'th smallest element in $A[i, j]$

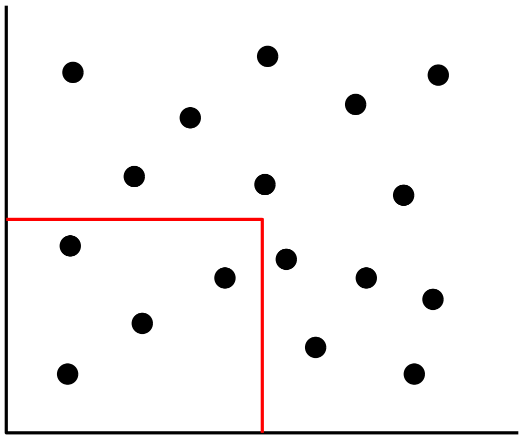
- Prefix Selection $(i = 1)$
- Range Median $(k = \lceil (j - i)/2 \rceil)$
- Bounded Rank Prefix Selection $(k \leq \kappa)$
- Fixed Rank Range Selection (k fixed for all queries)

Relation to Range Counting



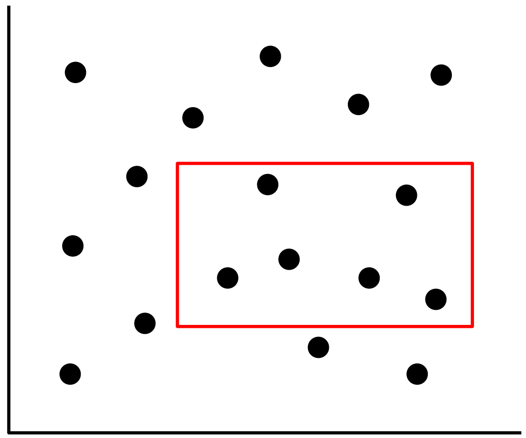
Relation to Range Counting

Dominance Counting (x, y)



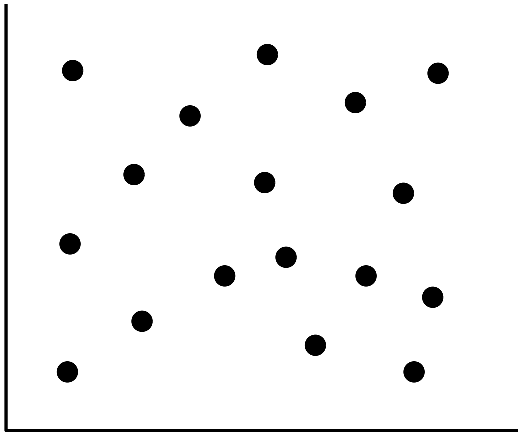
Relation to Range Counting

4-Sided Range Counting (x_1, x_2, y_1, y_2)



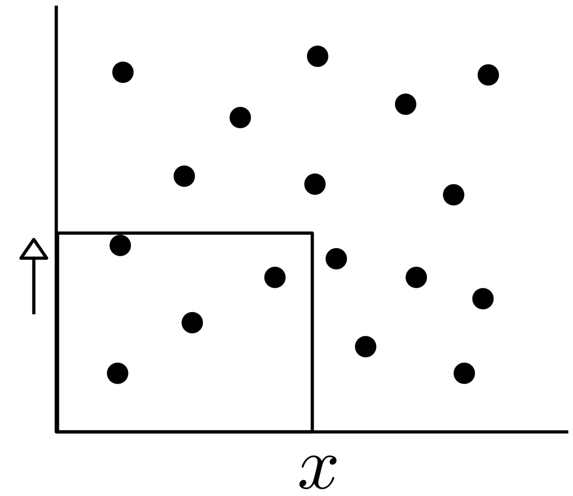
Relation to Range Counting

Prefix Selection (x, k)



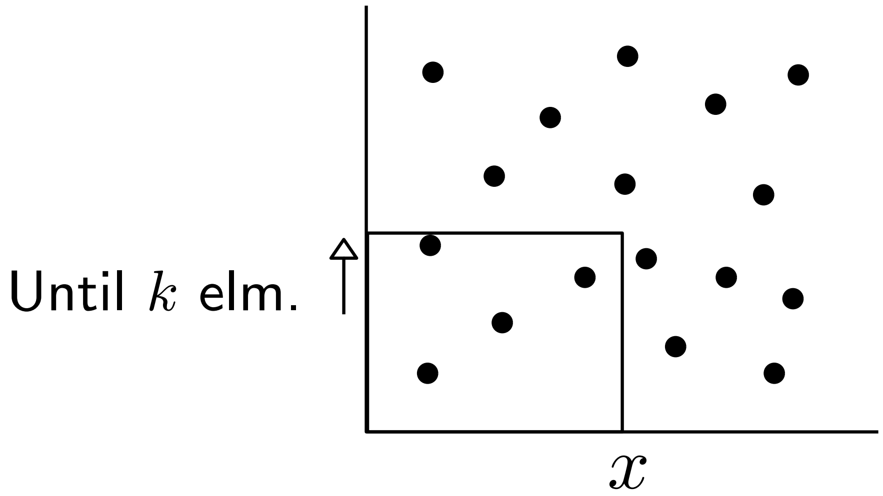
Relation to Range Counting

Prefix Selection (x, k)

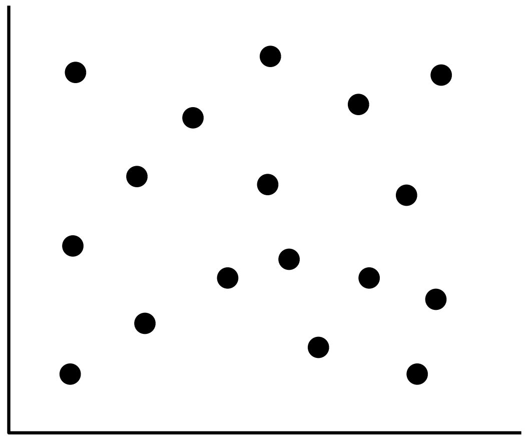


Relation to Range Counting

Prefix Selection (x, k)



Relation to Range Counting



Tight Bounds for Dominance Counting

$O(n)$ space $O(\log n / \log\log n)$ query time

[JáJá, Mortensen, Shi 04]

Space S needs $\Omega(\log n / \log(Sw/n))$ time for a query.
[Pătrașcu 07,08]

Range Selection Results

Static Data Structure:

$O(n)$ space and $O(\log n/\log\log n)$ query time.

Range Selection Results

Static Data Structure:

$O(n)$ space and $O(\log n / \log\log n)$ query time.

So it is not harder than range counting

Range Selection Results

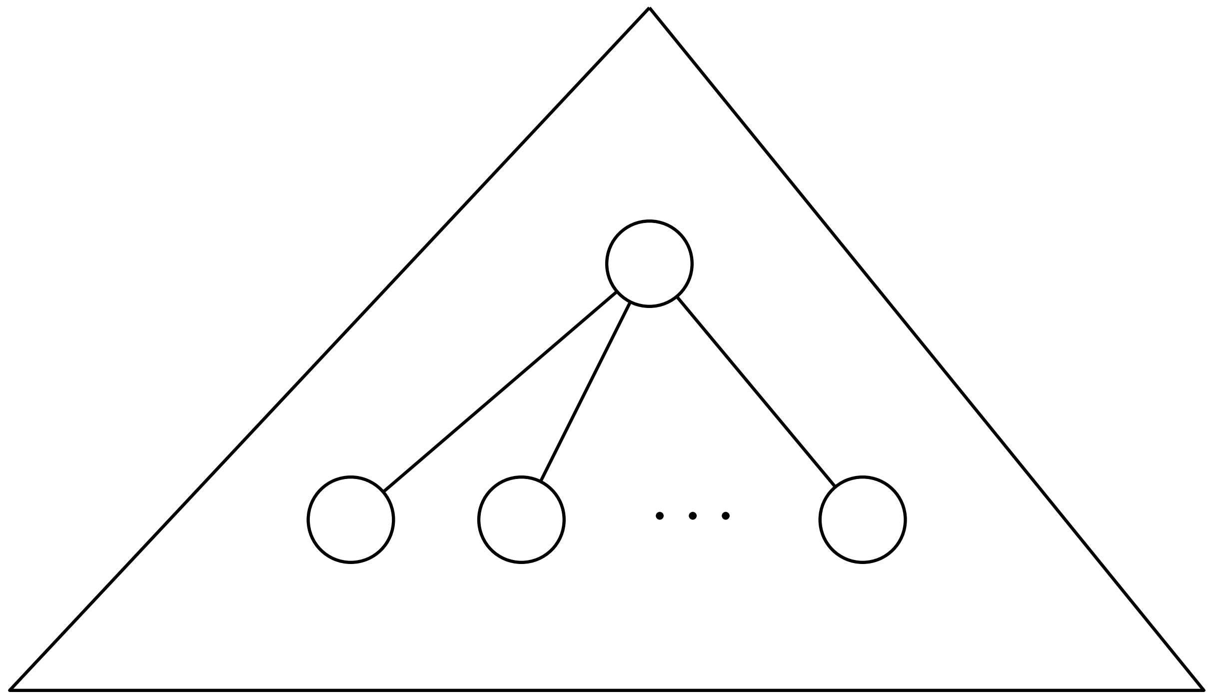
Static Data Structure:
$O(n)$ space and $O(\log n / \log \log n)$ query time.

So it is not harder than range counting

Dynamic Data Structure:
$O(n \frac{\log n}{\log \log n})$ space and $O((\log n / \log \log n)^2)$ query time

Range Selection Data Structure

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_j & \quad\cdots\quad & a_n \\ \hline\end{array}$$

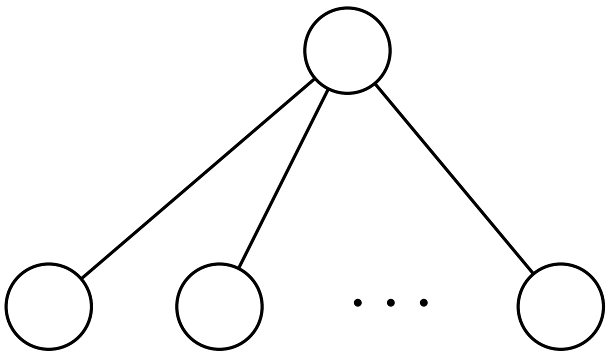


Leaves contain input in sorted order

Fanout $= \log^{\varepsilon} n, 0 < \varepsilon < 1$

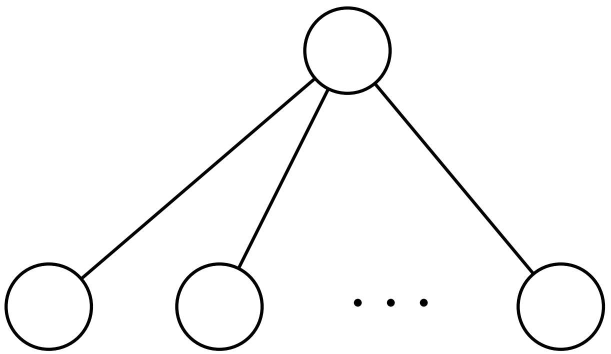
Guiding A Query

Query (i, j, k)



Guiding A Query

Query (i, j, k)



For each subtree T_a compute $|A[i,j] \cap T_a|$

Example: 5, 14, 7,17,10

Store as prefix sums: 5,19,26,43,53

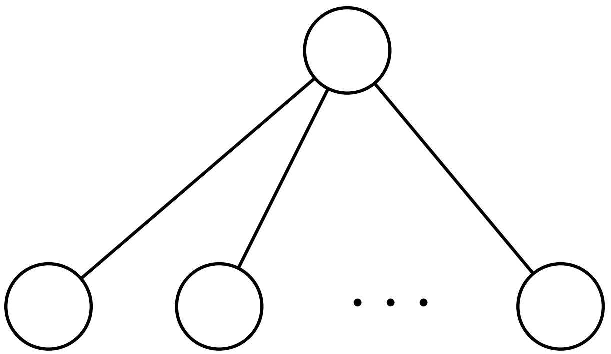
Guiding A Query

Query (i, j, k)

For each subtree T_a compute $|A[i,j] \cap T_a|$

Example: 5, 14, 7,17,10

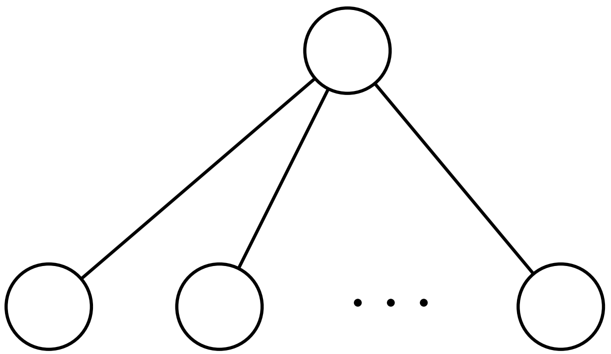
Store as prefix sums: 5,19,26,43,53



Next node is determined by succesor search for k

Guiding A Query

Query (i, j, k)



For each subtree T_a compute $|A[i, j] \cap T_a|$

Example: 5, 14, 7,17,10

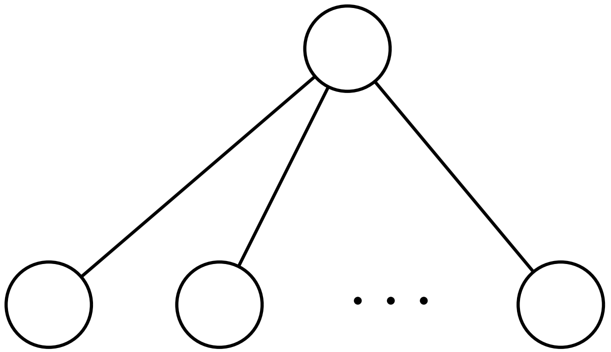
Store as prefix sums: 5,19,26,43,53

Next node is determined by succesor search for k

For constant time only use $\log^{1-\varepsilon}$ bits of each - fit in one word

Guiding A Query

Query (i, j, k)



For each subtree T_a compute $|A[i, j] \cap T_a|$

Example: 5, 14, 7,17,10

Store as prefix sums: 5,19,26,43,53

Next node is determined by succesor search for k

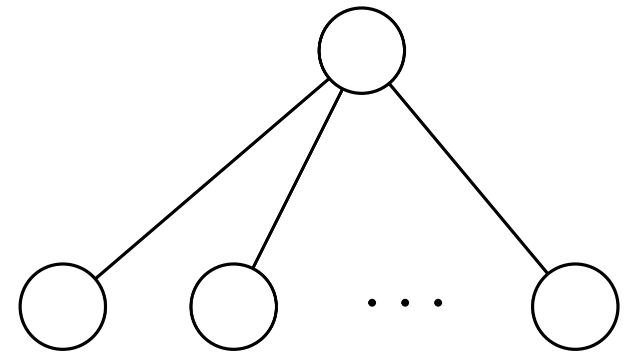
For constant time only use $\log^{1-\varepsilon}$ bits of each - fit in one word

Good	"Bad"
05..	14..
12..	42..
17..	42..
32..	42..
45..	42..
76..	47..

$$k = 4215$$

Guiding A Query

Query (i, j, k)



For each subtree T_a compute $|A[i,j] \cap T_a|$

Example: 5, 14, 7,17,10

Store as prefix sums: 5,19,26,43,53

Next node is determined by succesor search for k

For constant time only use $\log^{1-\varepsilon}$ bits of each - fit in one word

Good	"Bad"
05..	14..
12..	42..
17..	42..
32..	42..
45..	42..
76..	47..

$$k = 4215$$

Following
00..
00..
00..
00..
00..
00..

Range Selection NEW results

Prefix Selection:

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad \cdots \quad \mid a_x \mid \quad \cdots \quad \mid a_n}$$

Prefix Selection:

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Lower bound:
Space S needs $\Omega(\log n / \log(Sw/n))$ time for a query.

Range Selection NEW results

Prefix Selection:

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Lower bound:
Space S needs $\Omega(\log n / \log(Sw/n))$ time for a query.

Our data structure is optimal

Bounds "Equivalent" to range counting

Range Median

Prefix Selection reduces to Range Median

Range Median

Prefix Selection reduces to Range Median

$$PS = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad\cdots\quad \mid a_x \mid \quad\cdots\quad \mid a_n}$$

Range Median

Prefix Selection reduces to Range Median

$$PS = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad \cdots \quad \mid a_x \mid \quad \cdots \quad \mid a_n}$$

$$RM = \boxed{\overbrace{\infty, \ldots, \infty}^{2n \text{ times}} \mid \overbrace{-\infty, \ldots, -\infty}^{n \text{ times}} \mid a_1 \mid a_2 \mid \cdots \mid a_x \mid \quad \cdots \quad \mid a_n}$$

Range Median

Prefix Selection reduces to Range Median

$$PS = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_x & \cdots & a_n \\ \hline \end{array}$$

$$\xleftrightarrow{\hspace{6cm}}$$

query (x, k)

$$RM = \begin{array}{|c|c|c|c|c|c|c|c|}\multicolumn{1}{c}{2n \text{ times}} & \multicolumn{1}{c}{n \text{ times}} \\ \hline \infty, \ldots, \infty & -\infty, \ldots, -\infty & a_1 & a_2 & \cdots & a_x & \cdots & a_n \\ \hline \end{array}$$

Range Median

Prefix Selection reduces to Range Median

$$PS = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline \end{array}$$

query (x, k)

$$RM = \begin{array}{|c|c|c|c|c|c|c|c|}\multicolumn{1}{c}{2n \text{ times}} & \multicolumn{1}{c}{n \text{ times}} \\ \hline \infty, \ldots, \infty & -\infty, \ldots, -\infty & a_1 & a_2 & \cdots & a_x & \quad\cdots\quad & a_n \\ \hline \end{array}$$

query $(3n - 2(\lfloor x/2 \rfloor - k), x + 3n)$

Range Median

Prefix Selection reduces to Range Median

$$PS = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \quad\cdots\quad \mid a_x \mid \quad\cdots\quad \mid a_n}$$

query (x, k)

$$RM = \boxed{\overbrace{\infty, \ldots, \infty}^{2n \text{ times}} \mid \overbrace{-\infty, \ldots, -\infty}^{n \text{ times}} \mid a_1 \mid a_2 \mid \cdots \mid a_x \mid \quad\cdots\quad \mid a_n}$$

query $(3n - 2(\lfloor x/2 \rfloor - k), x + 3n)$

$$[b_1, \ldots, b_k, \ldots, b_{x/2}, \ldots, b_x]$$

Range Median

Prefix Selection reduces to Range Median

$$PS = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \cdots \mid a_x \mid \cdots \mid a_n}$$

query (x, k)

$$RM = \boxed{\overbrace{\infty, \ldots, \infty}^{2n \text{ times}} \mid \overbrace{-\infty, \ldots, -\infty}^{n \text{ times}} \mid a_1 \mid a_2 \mid \cdots \mid a_x \mid \cdots \mid a_n}$$

query $(3n - 2(\lfloor x/2 \rfloor - k), x + 3n)$

$$[-\infty, -\infty, b_1, \ldots, b_k, \ldots, b_{x/2}, \ldots, b_x]$$

Range Selection Summary

Prefix Selection, Range Selection, Range Median:

Space	Query Time
$O(n)$	$O(\log n / \log \log n)$
S	$\Omega(\log n / \log(Sw/n))$

Range Selection Summary

Prefix Selection, Range Selection, Range Median:

Space	Query Time
$O(n)$	$O(\log n / \log\log n)$
S	$\Omega(\log n / \log(Sw/n))$

Is this the whole story?

Select depends on k

Range Min ($k = 1$):

Select depends on k

Range Min ($k = 1$):

$O(n)$ space $O(1)$ query time

[Harel, Tarjan 84]

Select depends on k

Range Min ($k = 1$):

$O(n)$ space $O(1)$ query time

[Harel, Tarjan 84]

Extendable to $O(n)$ space and $O(k)$ query time

Select depends on k

Range Min ($k = 1$):

$O(n)$ space $O(1)$ query time

[Harel, Tarjan 84]

Extendable to $O(n)$ space and $O(k)$ query time

Hardness depends on k?

Select depends on k

Range Min ($k = 1$):

$O(n)$ space $O(1)$ query time

[Harel, Tarjan 84]

Extendable to $O(n)$ space and $O(k)$ query time

Hardness depends on k?

Is it easy if k is fixed for all queries?

Adaptive Data Structures

$$A = \boxed{a_1 \mid a_2 \mid \cdots \mid a_i \mid \cdots \mid a_x \mid \cdots \mid a_n}$$

Adaptive Data Structures

$A =$	a_1	a_2	$\cdots$	a_i	$\cdots$	a_x	$\cdots$	a_n

Prefix Selection:

Adaptive Data Structures

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \cdots & a_x & \cdots & a_n \\ \hline \end{array}$$

Prefix Selection:

Adaptive Data Structures

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Prefix Selection:

Data Structure:

$O(n)$ space and $O(\log k / \log\log n + \log\log n)$ query time.

Adaptive Data Structures

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Prefix Selection:

Data Structure:

$O(n)$ space and $O(\log k / \log\log n + \log\log n)$ query time.

Range Selection:

Adaptive Data Structures

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Prefix Selection:

Data Structure:

$O(n)$ space and $O(\log k/\log\log n + \log\log n)$ query time.

Range Selection:

Adaptive Data Structures

$$A = \boxed{a_1}\boxed{a_2}\boxed{\cdots}\boxed{a_i}\boxed{\quad\cdots\quad}\boxed{a_x}\boxed{\quad\cdots\quad}\boxed{a_n}$$

Prefix Selection:

Data Structure:

$O(n)$ space and $O(\log k/\log\log n + \log\log n)$ query time.

Range Selection:

Data Structure:

$O(n)$ space and $O(\log k/\log\log n + \log^2\log n)$ query time.

Adaptive Data Structures

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline a_1 & a_2 & \cdots & a_i & \quad\cdots\quad & a_x & \quad\cdots\quad & a_n \\ \hline\end{array}$$

Prefix Selection:

Data Structure:

$O(n)$ space and $O(\log k / \log\log n + \log\log n)$ query time.

Range Selection:

Data Structure:

$O(n)$ space and $O(\log k / \log\log n + \log^2 \log n)$ query time.

Are these optimal?

Range Selection NEW results

Bounded Rank Prefix Selection:

Input array and integer κ. All queries has $k \leq \kappa$.

Range Selection NEW results

Bounded Rank Prefix Selection:
Input array and integer κ. All queries has $k \leq \kappa$.

Space S needs $\Omega(\log \kappa / \log(Sw/n))$ time for a query.

Range Selection NEW results

Bounded Rank Prefix Selection:
Input array and integer κ. All queries has $k \leq \kappa$.

Space S needs $\Omega(\log \kappa / \log(Sw/n))$ time for a query.

Fixed Rank Range Selection.
Array and integer k. All queries selects the k'th smallest elm.

Range Selection NEW results

Bounded Rank Prefix Selection:
Input array and integer κ. All queries has $k \leq \kappa$.

Space S needs $\Omega(\log \kappa / \log(Sw/n))$ time for a query.

Fixed Rank Range Selection.
Array and integer k. All queries selects the k'th smallest elm.

Similar to the reduction for range median we get:
S space needs $\Omega(\log k / \log(Sw/n))$ query time

Range Selection NEW results

Bounded Rank Prefix Selection:
Input array and integer κ. All queries has $k \leq \kappa$.

Space S needs $\Omega(\log \kappa / \log(Sw/n))$ time for a query.

Fixed Rank Range Selection.
Array and integer k. All queries selects the k'th smallest elm.

Similar to the reduction for range median we get:
S space needs $\Omega(\log k / \log(Sw/n))$ query time

Our adaptive data structures are almost optimal

Range Selection NEW results

Bounded Rank Prefix Selection:
Input array and integer κ. All queries has $k \leq \kappa$.

Space S needs $\Omega(\log \kappa / \log(Sw/n))$ time for a query.

Fixed Rank Range Selection.
Array and integer k. All queries selects the k'th smallest elm.

Similar to the reduction for range median we get:
S space needs $\Omega(\log k / \log(Sw/n))$ query time

Our adaptive data structures are almost optimal

Range Min Data Structures does not generalize!!!

Range Selection Summary

	Space	Query Time
Range Selection	$O(n)$	$O(\log n / \log\log n)$
Prefix Selection	$O(n)$	$O(\log k / \log\log n + \log\log n)$
Range Selection	$O(n)$	$O(\log k / \log\log n + \log^2 \log n)$

	Space	Query Time
Range Selection	S	$\Omega(\log n / \log(Sw/n))$
Bounded Rank Prefix Sel.	S	$\Omega(\log \kappa / \log(Sw/n))$
Fixed Rank Range Selection	S	$\Omega(\log k / \log(Sw/n))$

Goto Next Slide

Thank You