CSE 4502/5717
BIG DATA ANALYTICS
EXAM 1 REVIEW

TOPICS FOR EXAM 1:

* RANDOMIZED ALGORITHMS
* PARALLEL ALGORITHMS
* SINGLE DISK ALGORITHMS

RANDOMIZED ALGORITHMS:

A RAND. ALG. MAKES COIN FLIPS within the body of the alg.

RAND. ALG.
- MONTE CARLO
- LAS VEGAS

By High prob. we mean a prob.
of $\geq 1 - n^{-\alpha}$,

$n \rightarrow$ INPUT SIZE
$\alpha \rightarrow$ PROB. PARAMETER.

PROBLEM 1:

INPUT: AN ARRAY $A[1:n]$
$\exists \frac{n}{2}$ COPIES of one element.
The other elements are DISTINCT.

OUTPUT: Repeated element.

A Las Vegas Alg.

ONE BASIC STEP.

Repeat

Flip an n-sided coin to get i;
Flip an n-sided coin to get j;
if $i \neq j$ & $A[i] = A[j]$ then
we output $A[i]$ and quit;

Forever

ANALYSIS:-

Prob [Success in one basic step] $= \dfrac{\frac{n}{2}\left(\frac{n}{2}-1\right)}{n^2}$

$\simeq \dfrac{1}{4}$.

$\Rightarrow$ Prob. [Failure in one basic step]

$\simeq 3/4$.

$\Rightarrow$ Prob. [Failure in k steps] $\leq \left(\frac{3}{4}\right)^k$.

We want this to be $\leq n^{-\alpha}$.

$$\left(\frac{3}{4}\right)^k \leq n^{-\alpha}.$$

$$k \log\left(\frac{3}{4}\right) \leq -\alpha \log n$$

$$\Rightarrow \quad \boxed{k \geq \frac{\alpha \log n}{\log\left(\frac{4}{3}\right)}}$$

$\Rightarrow$ RUN TIME $= \tilde{O}(\log n)$

Recall: $f(n) = \tilde{O}(g(n))$

If $f(n) \leq c\alpha\, g(n) \quad \forall n \geq n_0$

with a prob. of $\geq (1 - n^{-\alpha})$

For some constants c and n_0.

Example: **INPUT:** $A[1:n]$.

Output: An element of $A \geq$ the Median of A.

Sample K elements, Find and Output the max of the Sample.

Analysis: Prob. [A Random element is incorrect] is $\leq \frac{1}{2}$.

Output of our alg is incorrect iff all the k elements are $\leq$ the median of A.

Prob. of this is $\leq \left(\frac{1}{2}\right)^k$.

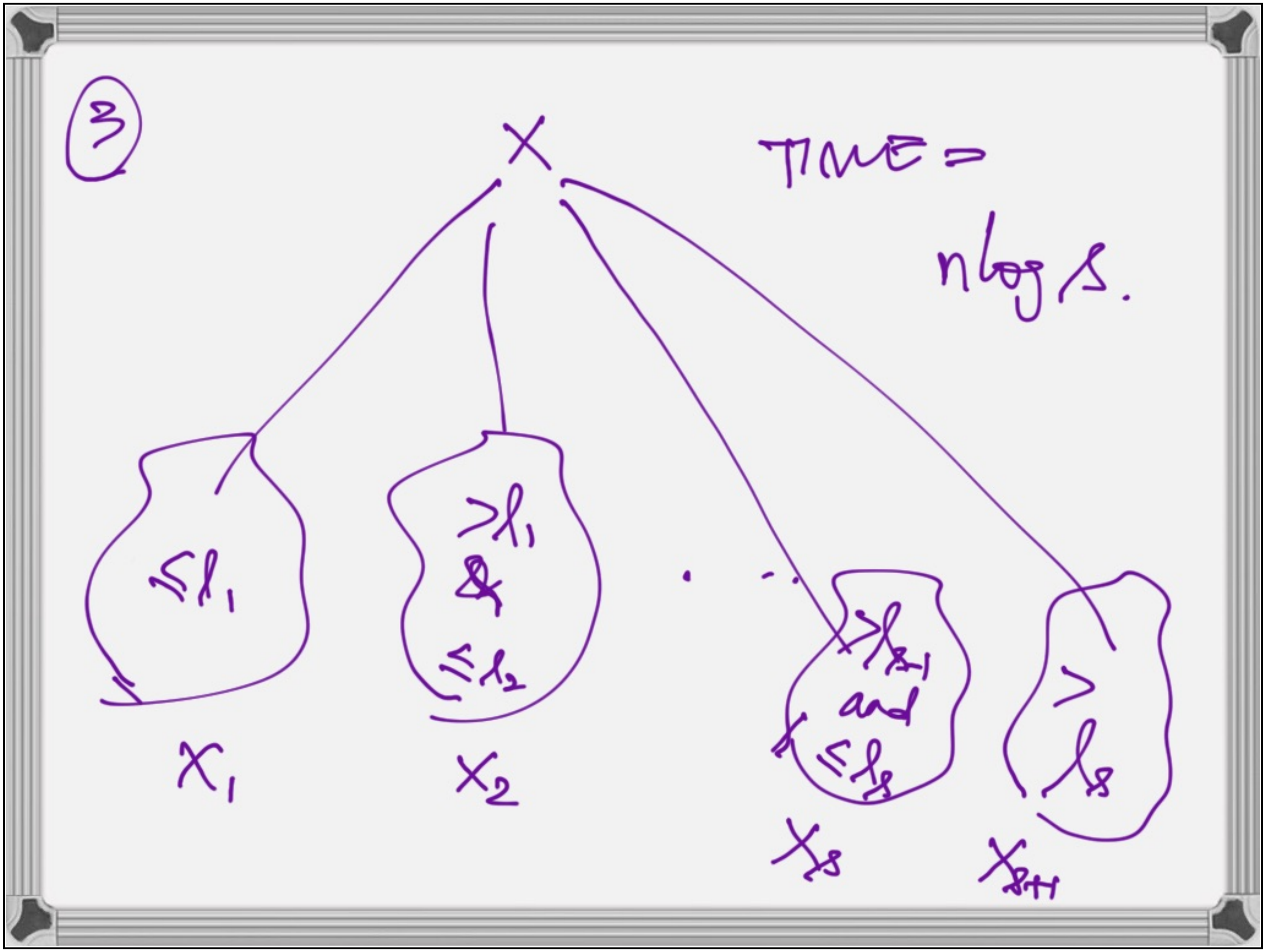
We want this to be $\leq n^{-\alpha}$.

$$\left(\frac{1}{2}\right)^k \leq n^{-\alpha} \Rightarrow -k \leq -\alpha \log n \Rightarrow \boxed{k \geq \alpha \log n}.$$

FRAZER & McKELLAR'S IDEA:

INPUT: $X = k_1, k_2, \ldots, k_n$; Output: Sorted X;

1. Pick a Random Sample S from X, with $|S| = s$.
2. Sort the sample to get $l_1, l_2, \ldots, l_s$.


3
X
TIME =
n log s.
≤ ℓ1
> ℓ1
&
≤ ℓ2
> ℓs-1
and
≤ ℓs
>
ℓs
X1
X2
Xs
Xs+1

4 **For** $1 \le i \le \ell(r_t)$ **do**

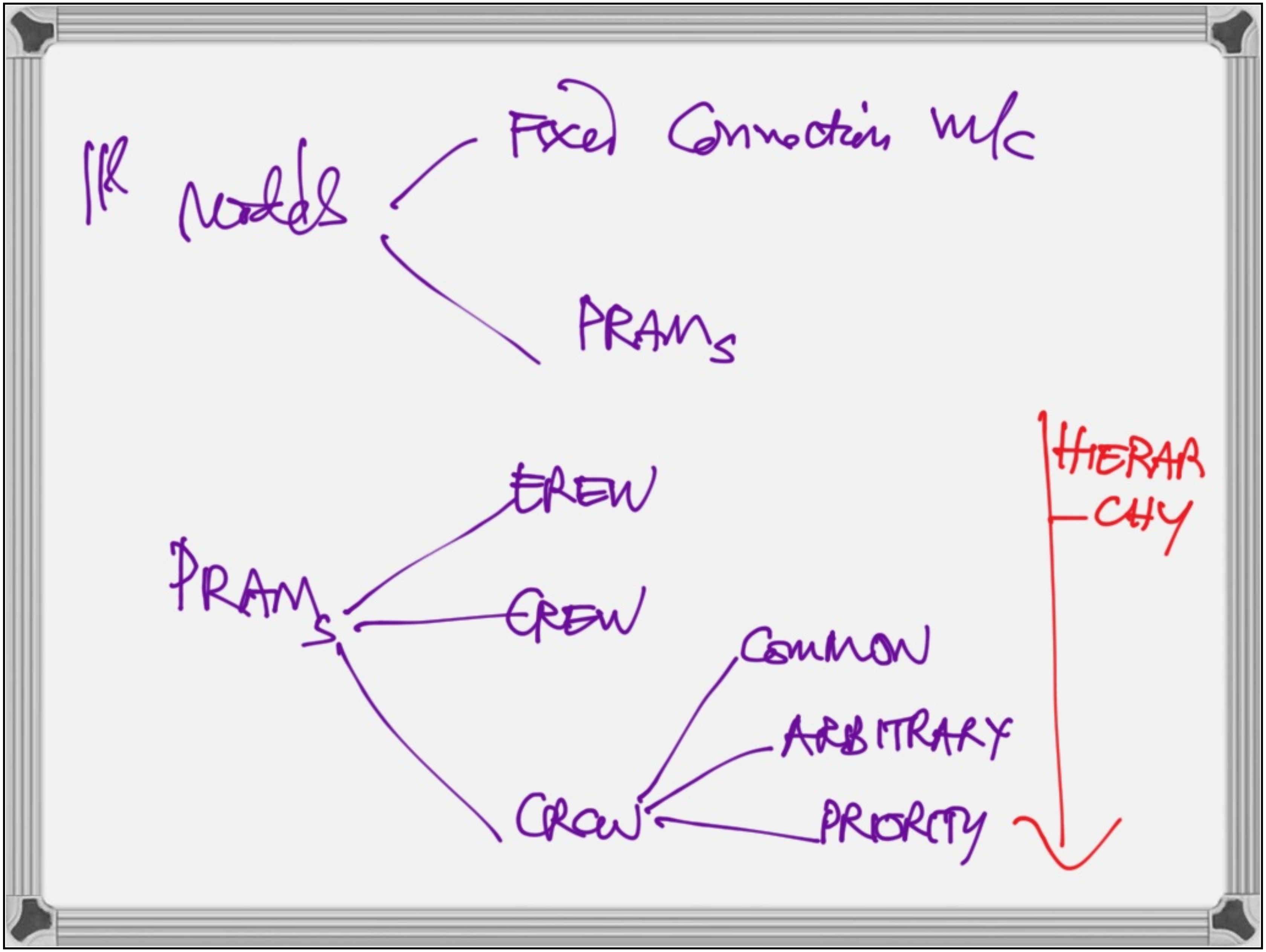
Sort X_i and output

\# of Comparisons

$$= n \log n + \tilde{O}(n \log \log n)$$

PARALLEL ALGORITHMS:

FACT: IF T is the RUN TIME of a $\parallel$ alg that uses P processors, then $T \geq \frac{S}{P}$, where S is the run time of the best alg. that solves the same problem.



||l Models
Fixed Connection M/c
PRAMs
PRAMs
EREW
CREW
CRCW
COMMON
ARBITRARY
PRIORITY
HIERARCHY

Example: **INPUT:** $X = b_1, b_2, \ldots, b_n$

Output: $b_1 \wedge b_2 \wedge \ldots \wedge b_n$.

FACT: We can solve this in O(1) time using n Common CRCW PRAM Proc.

Algorithm:

1. Processor 1 writes a 1 in Result;
2. **For** $1 \le i \le n$ in ‖ **do**
   if $b_i = 0$ then Processor i tries to write a 0 in Result;

PROBLEM: INPUT: $X = k_1, k_2, \ldots, k_n$.

OUTPUT: Max of X.

FACT: Can be solved in $O(1)$ time using n^2 COMMON CRCW PRAM PROCESSORS.

ALGORITHM:

EVERY GROUP Checks if its key is the CORRECT answer.

For $1 \le i,j \le n$ in ||el **do**

Processor (i,j) Computes

b_{ij} = "Is $k_i \ge k_j$?"

For $1 \le i \le n$ in ||el **do**

Processors in G_i Compute

the Boolean AND of $b_{i1}, b_{i2}, b_{i3}, \dots, b_{in}$

Let C_i be the Boolean AND.

For $1 \le i \le n$ in ||el do if $C_i = 1$ then Proc. $(i,1)$ writes k_i in Result

PROBLEM: PREFIX COMP.

INPUT: $X = k_1, k_2, \ldots, k_n \in \Sigma$.

OUTPUT: $k_1, k_1 \oplus k_2, k_1 \oplus k_2 \oplus k_3, \ldots,$
$k_1 \oplus k_2 \oplus \cdots \oplus k_n$.

$\oplus \rightarrow$ ARBITRARY UNIT TIME, BINARY & ASSOCIATIVE OPERATOR.

THEOREM: we can solve this in $O(\log n)$ TIME USING n CREW PRAM PROCESSORS.

$k_1, k_2, \ldots, k_{\frac{n}{2}},$ | $k_{\frac{n}{2}+1}, \ldots, k_n$

$\frac{n}{2}$ PROC $\Downarrow$ REC. PREFIX | $\frac{n}{2}$ PROC $\Downarrow$ REC. PREFIX

$k_1', k_2', \ldots, k_{\frac{n}{2}}', \; k_{\frac{n}{2}+1}', \ldots, k_n'$

PRE-ADD

$\Downarrow$ output

Let $T(n)$ be the RUN TIME of this alg on any input of size n.

$$T(n) = T\left(\frac{n}{2}\right) + \Theta(1)$$

$$= \Theta(\log n).$$

FACT. We can also do prefix comp. in $O(\log n)$ TIME using $\frac{n}{\log n}$ CREW PRAM PROC.

AN ALG. IS OPTIMAL IF $T = \frac{S}{P}$.

It is asymptotically optimal if $T = O\left(\frac{S}{P}\right)$.

WORK DONE BY A

|| ALG. = P.T.

Example: INPUT: $X = k_1, k_2, \ldots, k_n$; K

Output: output elements of X that are $\leq K$ FIRST; Then output the rest

We can solve this using a prefix comp.

5, 8, 12, 11

$A[i] = 0$ if $k_i > K$

$= 1$ if $k_i \le K$.

$X = 5, 36, 8, 17, 12, 25, 18, 19, 24, 11$

$\boxed{K=12}$

1	0	1	0	1	0	0	0	0	1
1	1	2	2	3	3	3	3	3	4

PREFIX SUMS ↑

Example: Find the Rank of an element.

Input: $X = k_1, k_2, \ldots, k_n$; $k \in X$.

Output: $\text{Rank}(k, X) \triangleq |\{q \in X : q < k\}| + 1$

We can solve this in $O(\log n)$ time using $\frac{n}{\log n}$ CREW PRAM PROC

$\Rightarrow$ We can sort n elements in $O(\log n)$ time using $\frac{n^2}{\log n}$ CREW PRAM PROC.

SLOW-DOWN LEMMA:

IF A ‖ₗ ALG. takes time T using P proc., the same alg can be run on a P'-processor m/c in $O\left(\frac{PT}{P'}\right)$ time, FOR any $P' \le P$.

Out-of-Core Computing:

B → BLOCK SIZE.

PROBLEM: SORTING n #'s.

Theorem: Any sorting alg. needs

$$\Omega\left(\frac{n}{B}\frac{\log(n/M)}{\log(M/B)}\right)$$ I/O operations.

Example:- TWO_WAY MERGE.

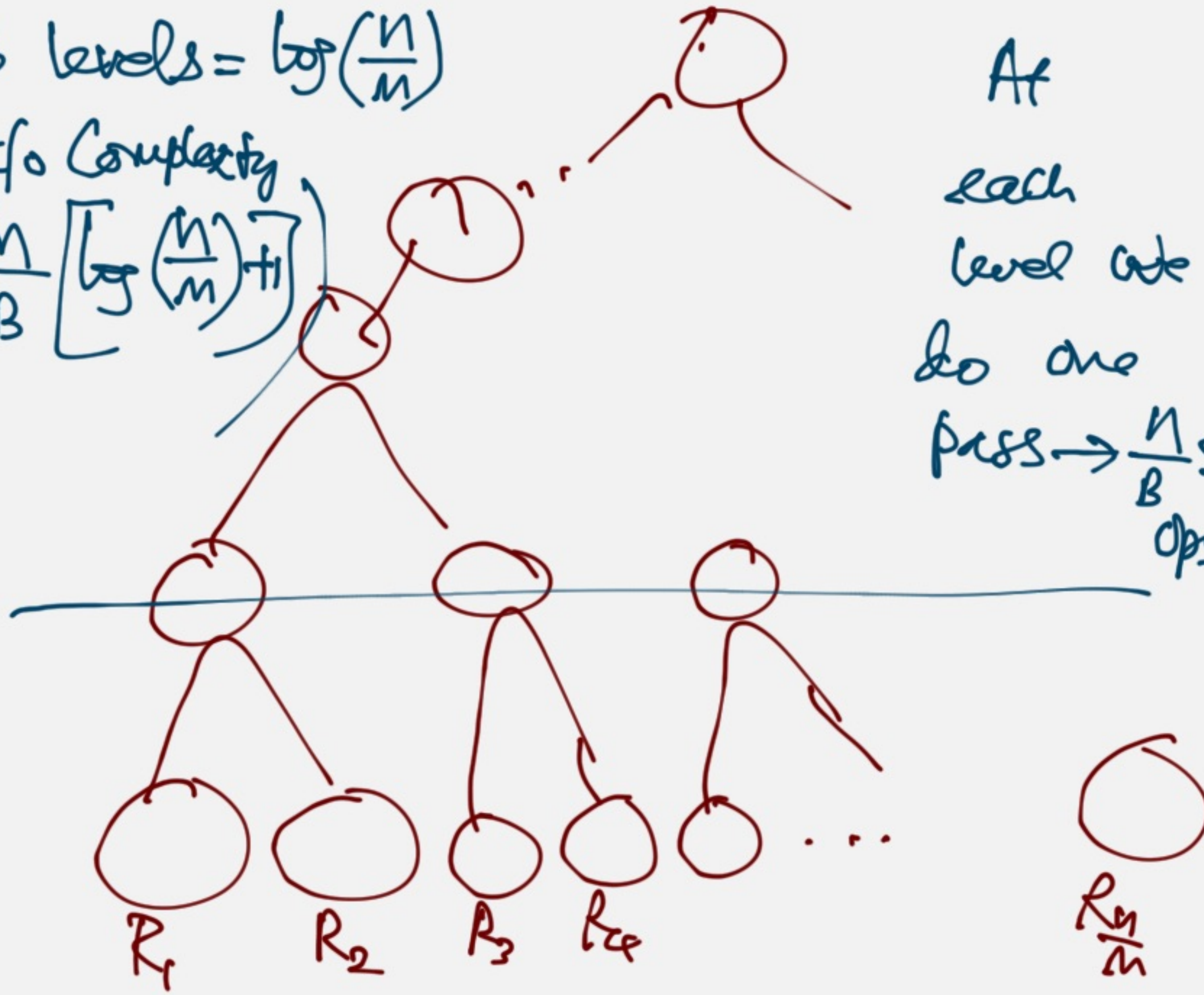
(1) IN ONE PASS, FORM RUNS of length M EACH.

(2) Merge the $\frac{n}{M}$ RUNS using the 2-way Merge.

# of levels = $\log\left(\frac{n}{M}\right)$

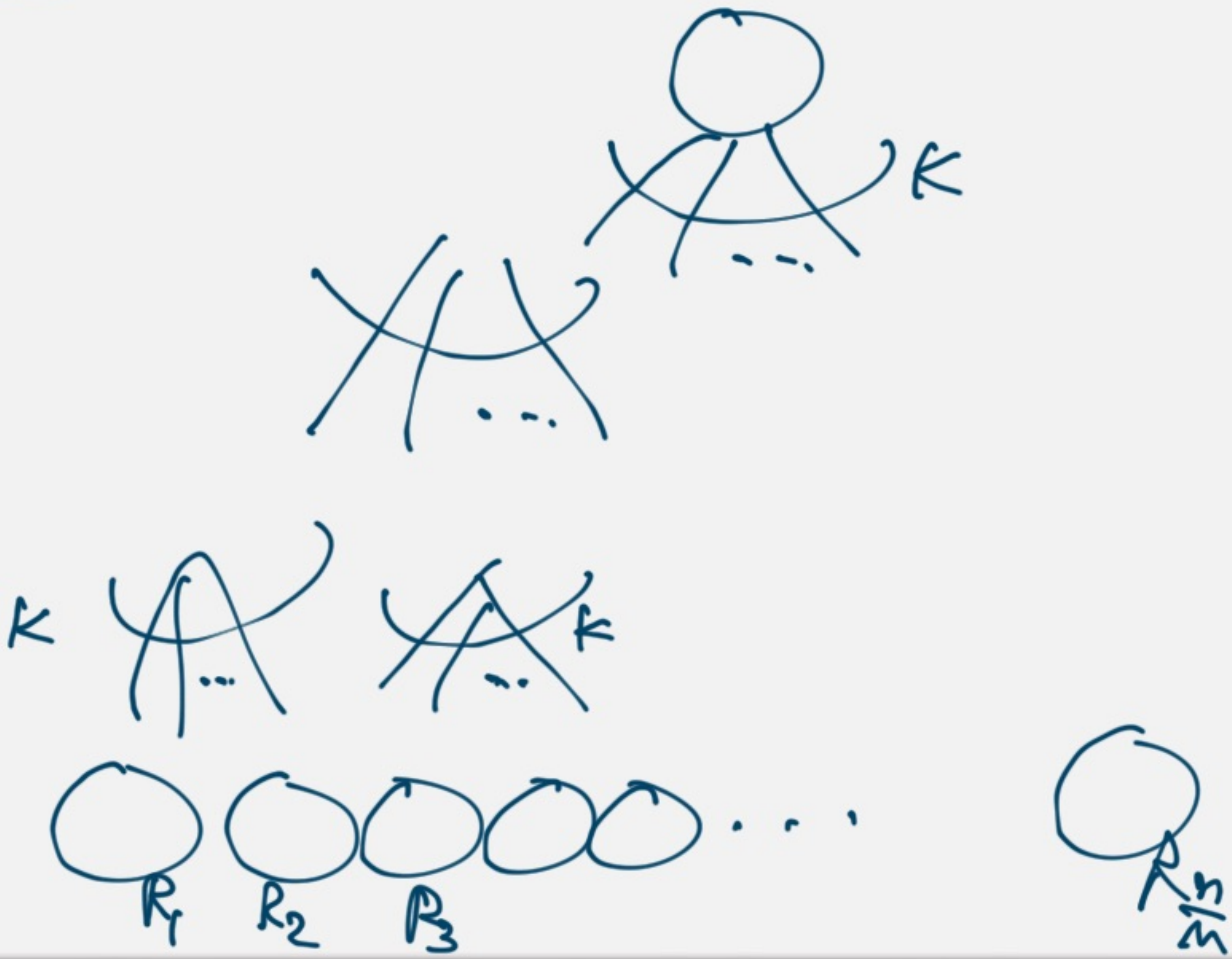
$\Rightarrow$ I/O Complexity

$= O\left(\frac{n}{B}\left[\log\left(\frac{n}{M}\right)+1\right]\right)$



At each level we do one pass $\rightarrow \frac{n}{B}$ I/O ops.

Next IDEA: Use K-way MERGE.



# of levels = $\dfrac{\log\left(\frac{n}{M}\right)}{\log K}$

$\Rightarrow$ I/O Complexity $= O\left(\frac{n}{B}\left[\frac{\log(n/M)}{\log K}+1\right]\right)$

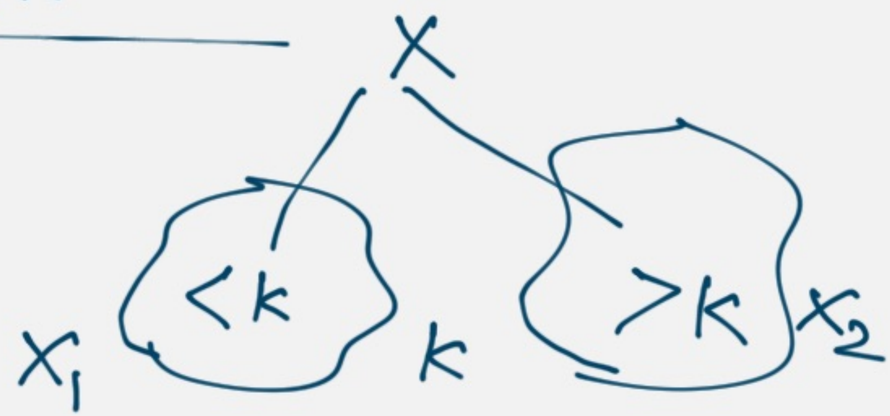
We pick $K = \frac{M}{B} \Rightarrow$ Asymptotic Optimality.

PROBLEM: **Selection:**

INPUT: $X = k_1, k_2, \ldots, k_n$; $1 \leq i \leq n$.

Output: The i^{th} smallest element of X.

QuickSelect: Pick a pivot $k \in X$.



CASE 1: if $|X_1| = i-1$ then output k & quit;

CASE 2: if $|X_1| \geq i$ then output QuickSelect(X_1, i);

CASE 3: if $|X_1|+1 < i$ then output QuickSelect$(X_2, i-|X_1|-1)$;

FACT: We can do selection on a single DBK in $O(\frac{n}{B})$ I/Os.

Model Exam:

P1: $|A| = |B| = n$; $A \cap B = n^{3/4}$.

Repeat

Best step.

Pick a Random $A[i]$;
Do a Bin. Srch in B;
If $A[i] \in B$ Output $A[i]$
& Quit;

Forever

Prob. Success in one Basic Step $= \dfrac{n^{3/4}}{n} = n^{-1/4}$

Prob. Failure in one Step $= \left(1 - \dfrac{1}{n^{1/4}}\right)$

Prob Failure in k Steps

$$= \left(1-\frac{1}{n^{1/4}}\right)^k = \left(1-\frac{1}{n^{1/4}}\right)^{n^{1/4}\cdot\frac{k}{n^{1/4}}}$$

$$\leq \exp\left(-\frac{k}{n^{1/4}}\right) \leq n^{-\alpha} \text{ if}$$

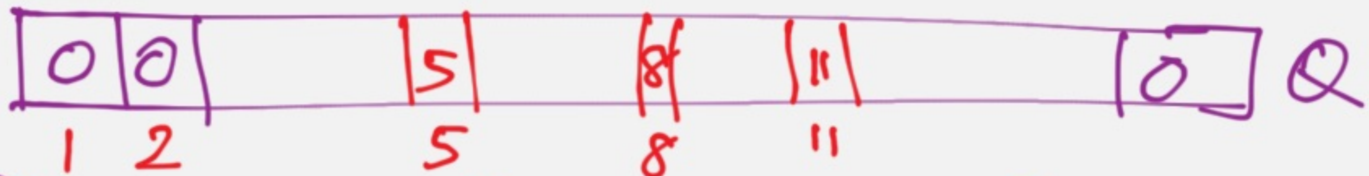
$$-\frac{k}{n^{1/4}} \leq -\alpha \log n \Rightarrow k \geq \alpha n^{1/4} \log n.$$

$$\Rightarrow \text{RUNTIME} = \tilde{O}(n^{1/4} \log^2 n)$$

#2 : Pick q elements @ Random
From EACH X_i; Find &
Output the MAX of all
of these kq elements.

P3 Assume $P = 10n$.

Use an array of size $10n$:



1. **For** $1 \le i \le n$ in PR **do**

   Processor i writes $A[i]$ in $Q[A[i]]$;

A: 5, 11, 8, 7, 12, 14, 3

(2) For $1 \le i \le n$ in $\mathbb{R}$ do

if $Q[B[i]] \neq 0$ then

Proc. i outputs "$\exists$ a common element" & exit;

B: 8, 15, 23, 17, 4, 6

P4 $A \rightarrow$ SORTED. $B \Rightarrow$ Not Sorted

$|A| = n$; $|B| = m$

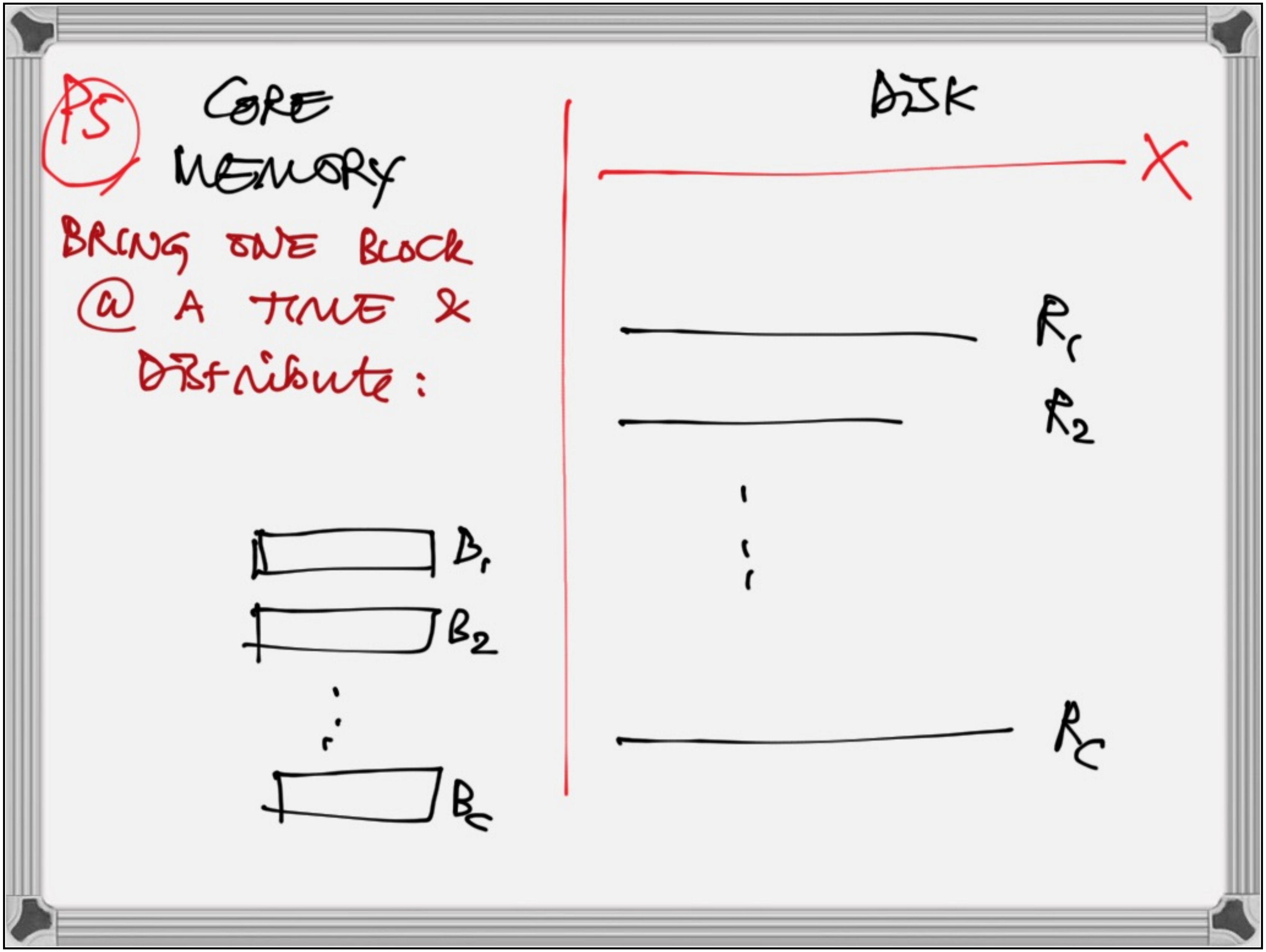
Compute $\boxed{A \cap B}$.

1 For $1 \leq i \leq m$ in PR do

Proc. i does a Bin SRCH in A to check if $B[i] \in A$.

If $B[i] \in A$ then $Q[i] = 1$;

2


PS
CORE MEMORY
BRING ONE BLOCK @ A TIME & DISTRIBUTE :
DISK
X
R_1
R_2
R_c
B_1
B_2
B_c

P6

DISK

X

Keep
A DATA STRUCTURE
such as a
Red_Black tree.
of size $M/10$.