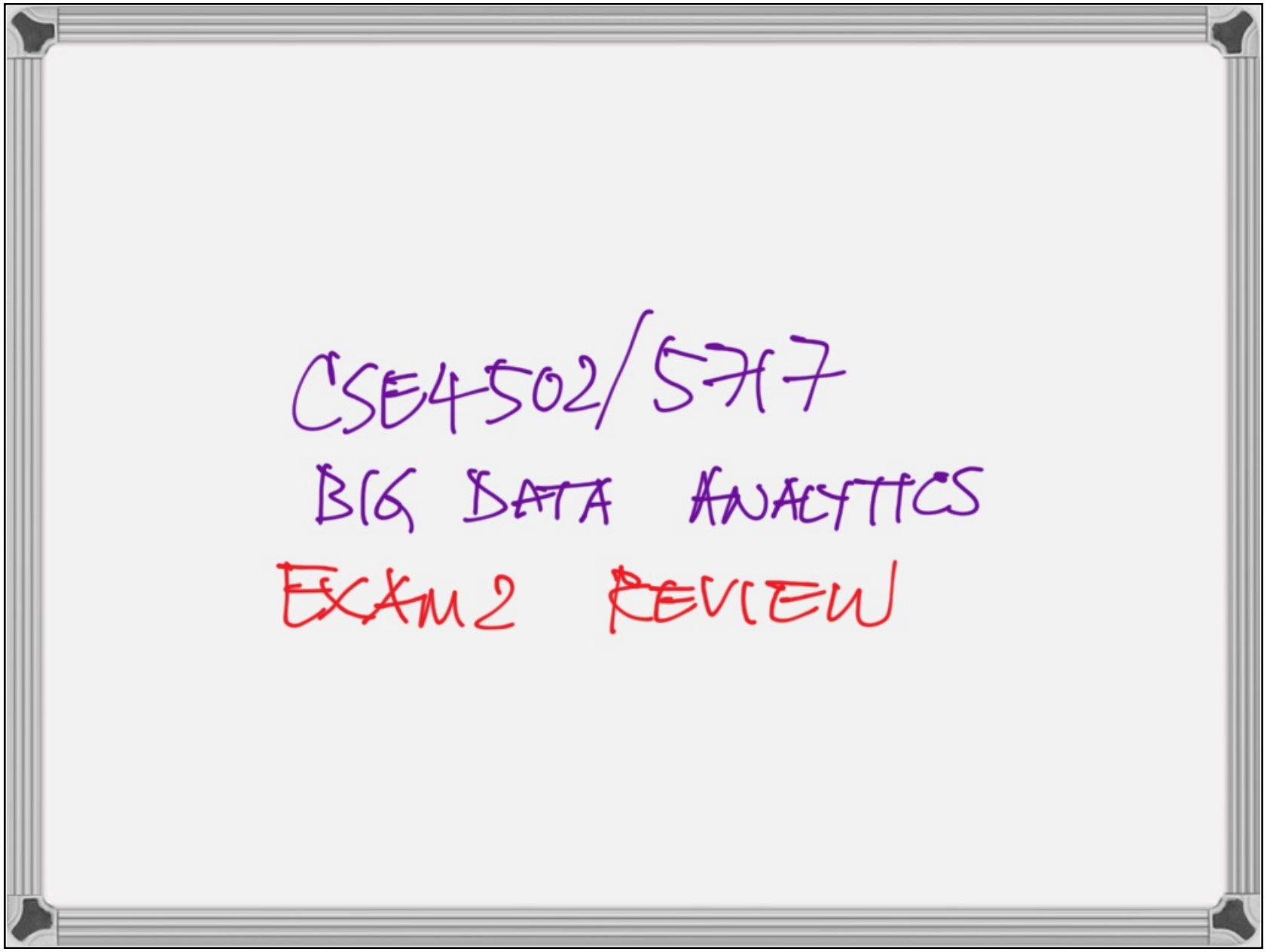
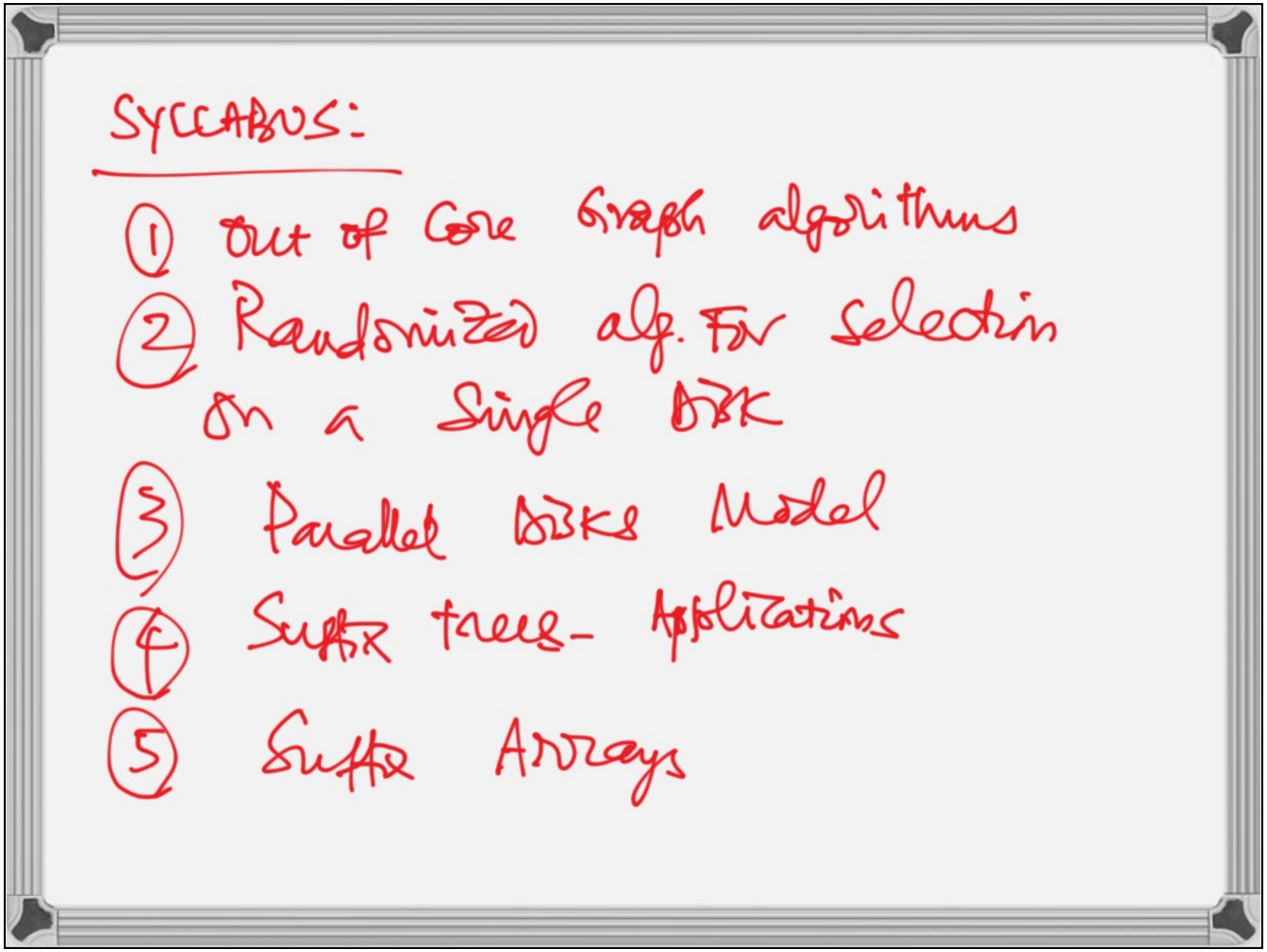

CSE4502/5717
BIG DATA ANALYTICS
EXAM 2 REVIEW


SYLLABUS:
1) Out of Core Graph algorithms
2) Randomized alg. For selection on a Single DISK
3) Parallel DISKS Model
4) Suffix trees - Applications
5) Suffix Arrays

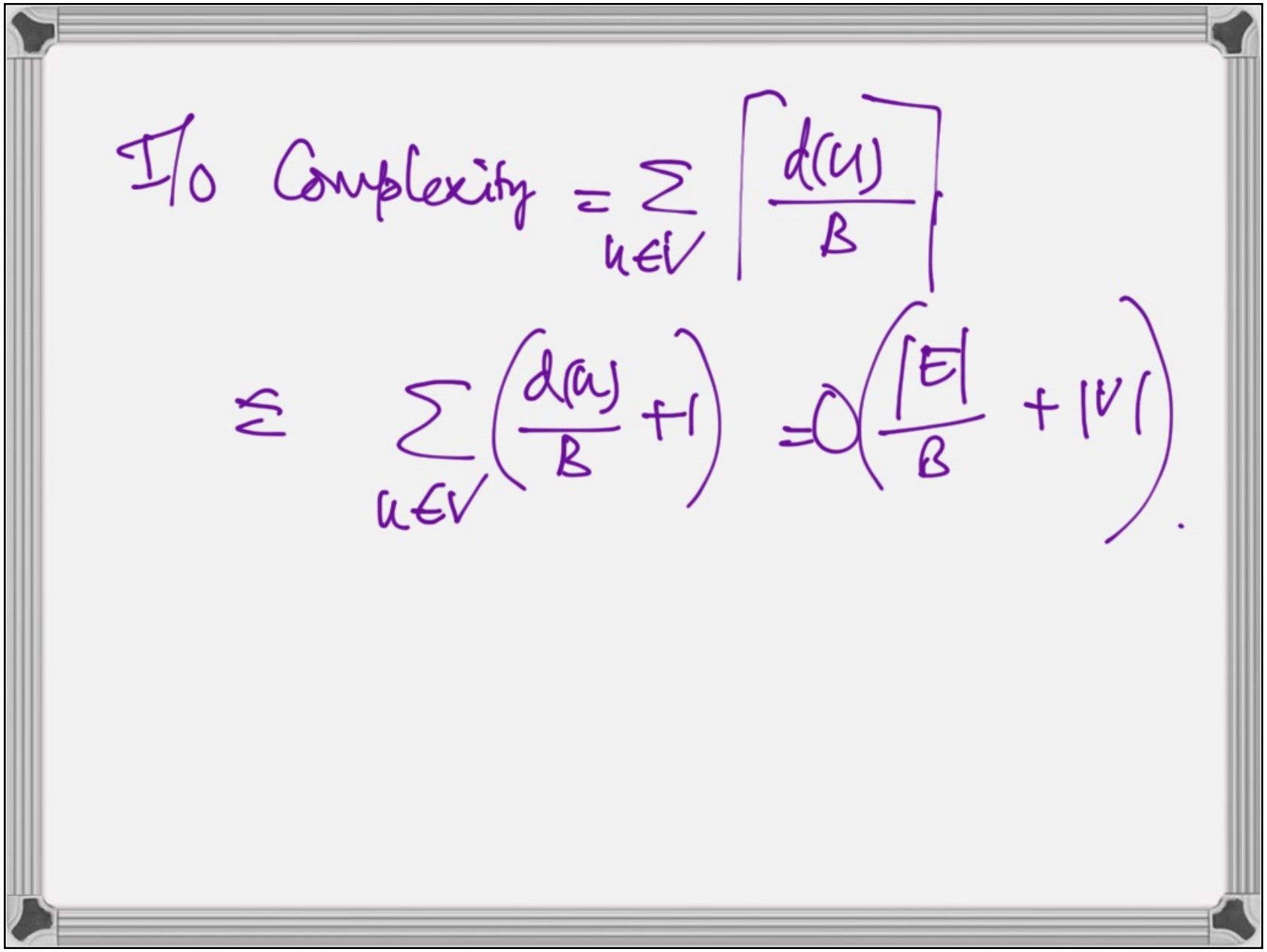
① Graph Algorithms:

$G(V, E)$.

DFS in a SINGLE DISK.

Graph is supplied as adjacency lists.

Let $d(u)$ be the degree of node u.


I/O Complexity = $\sum_{u \in V} \left\lceil \frac{d(u)}{B} \right\rceil$
$\leq \sum_{u \in V} \left(\frac{d(u)}{B} + 1 \right) = O\left(\frac{|E|}{B} + |V| \right)$.

(2) Rand. Alg. for Selection.

Input: $X = k_1, k_2, \ldots, k_n$, $1 \le i \le n$.

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

IDEA: Pick a Random Sample S From X. Let $|S| = s$.

Identify 2 elements in S s.t.

(1) The i^{th} smallest $\&$ $x \in [l_1, l_2]$

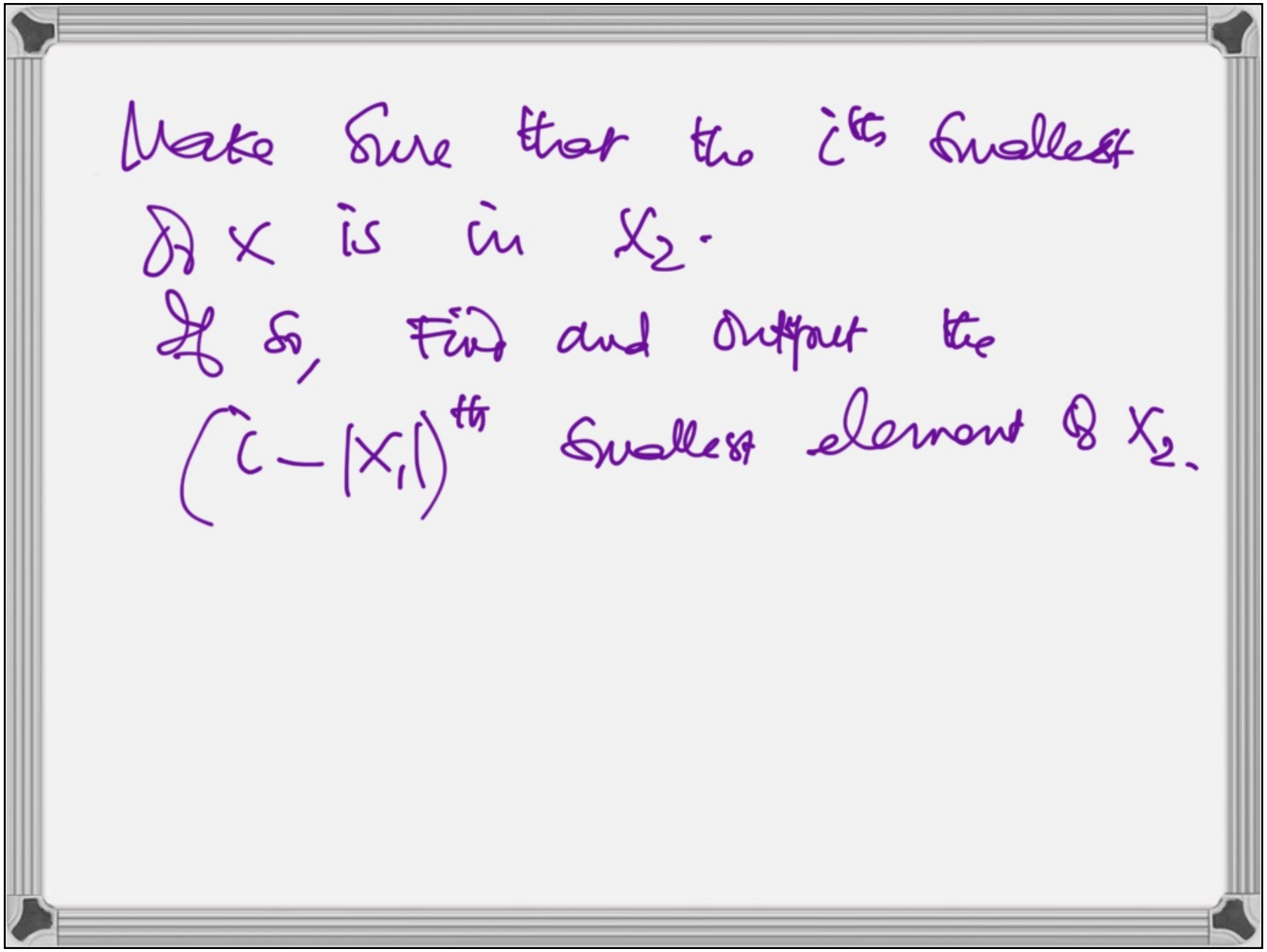
where l_1 and l_2 are the 2 elements

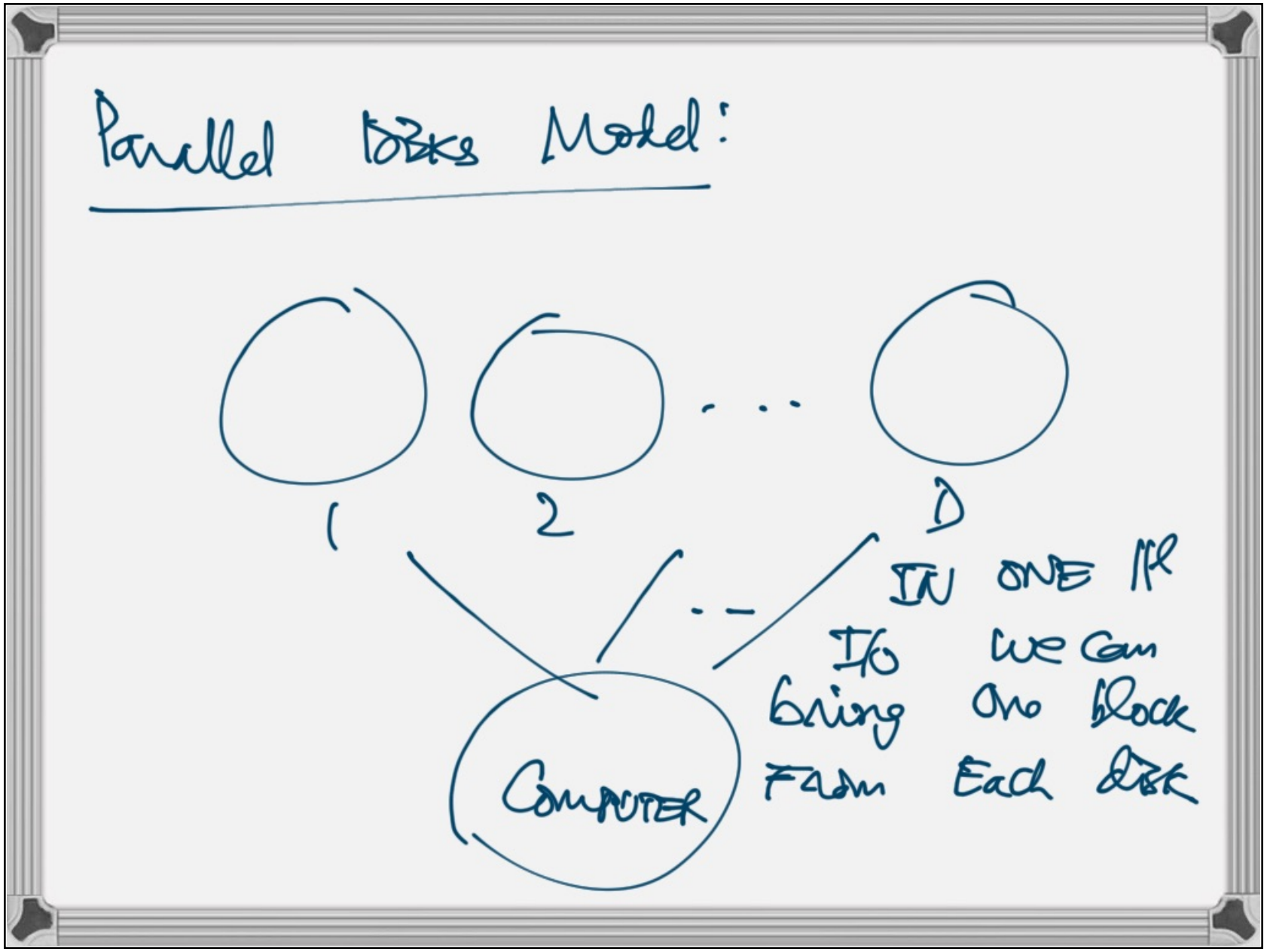
and

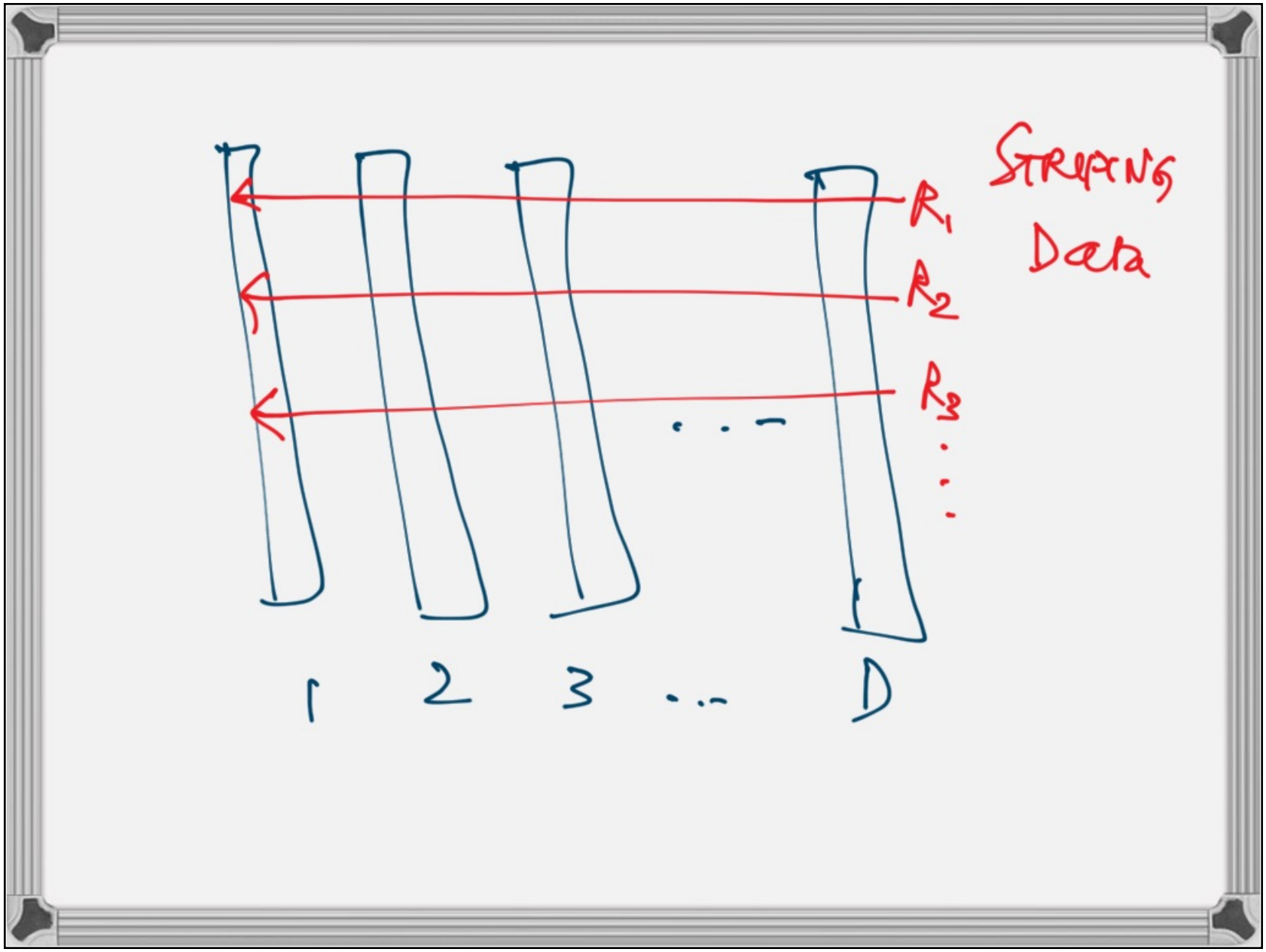
(2) $\left| \{ q \in X : l_1 \le q \le l_2 \} \right|$ is "small".

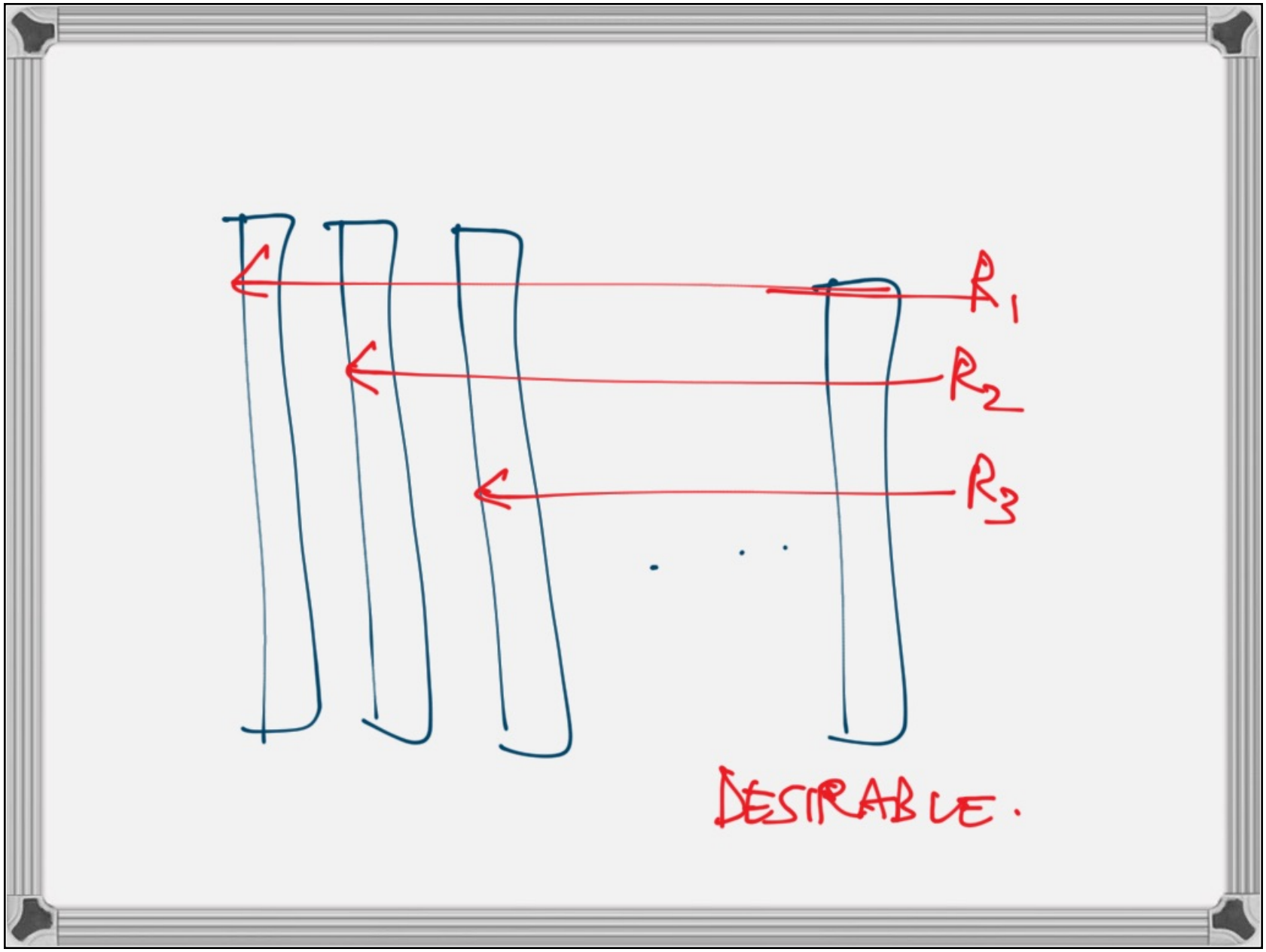
Let $X_1 = \{ q \in X : q < l_1 \}$

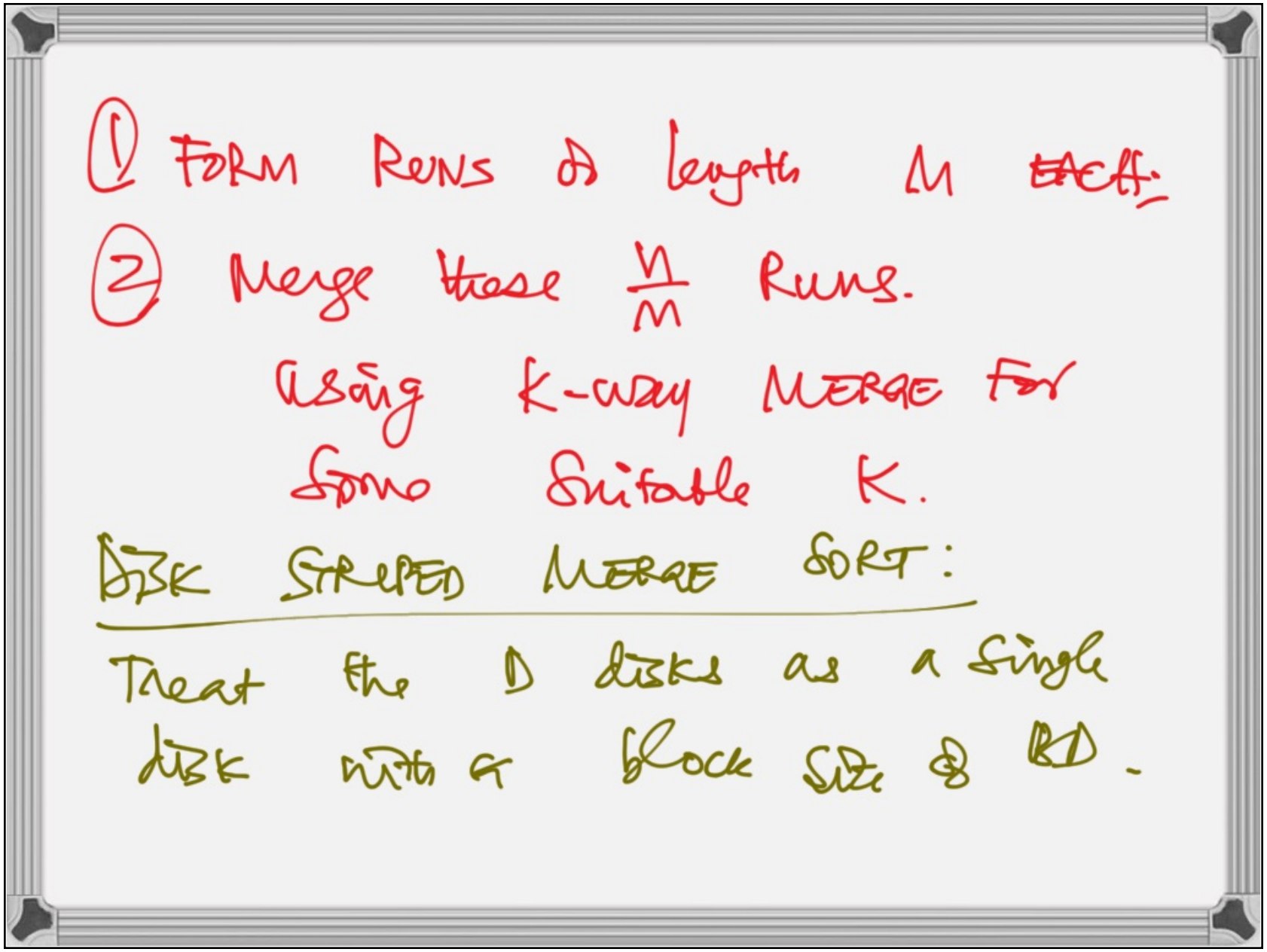
$X_2 = \{ q \in X : l_1 \le q \le l_2 \}$.

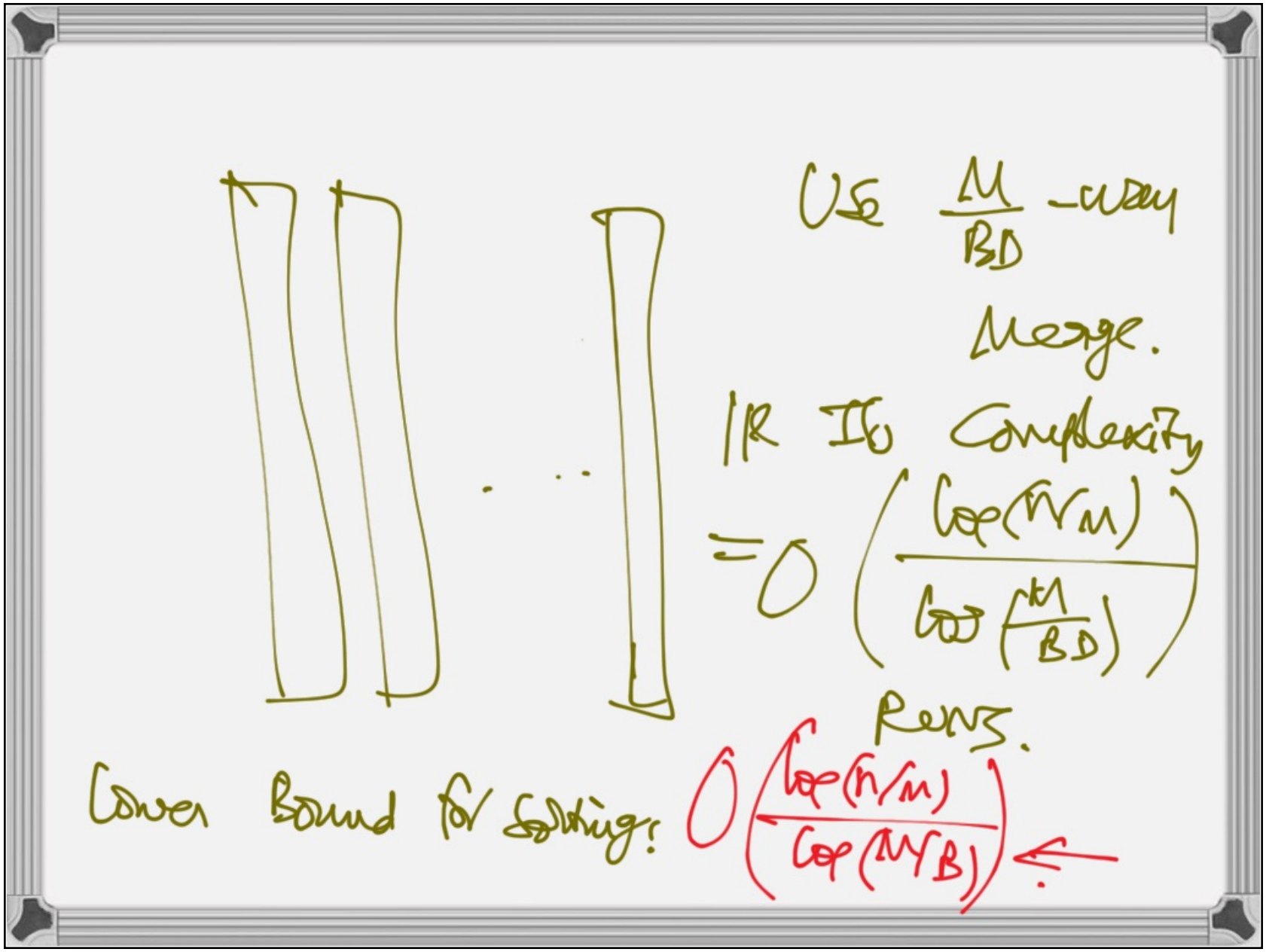

Make sure that the i^{th} smallest
of x is in X_2.
If so, Find and output the
$(i-|X_1|)^{th}$ smallest element of X_2.

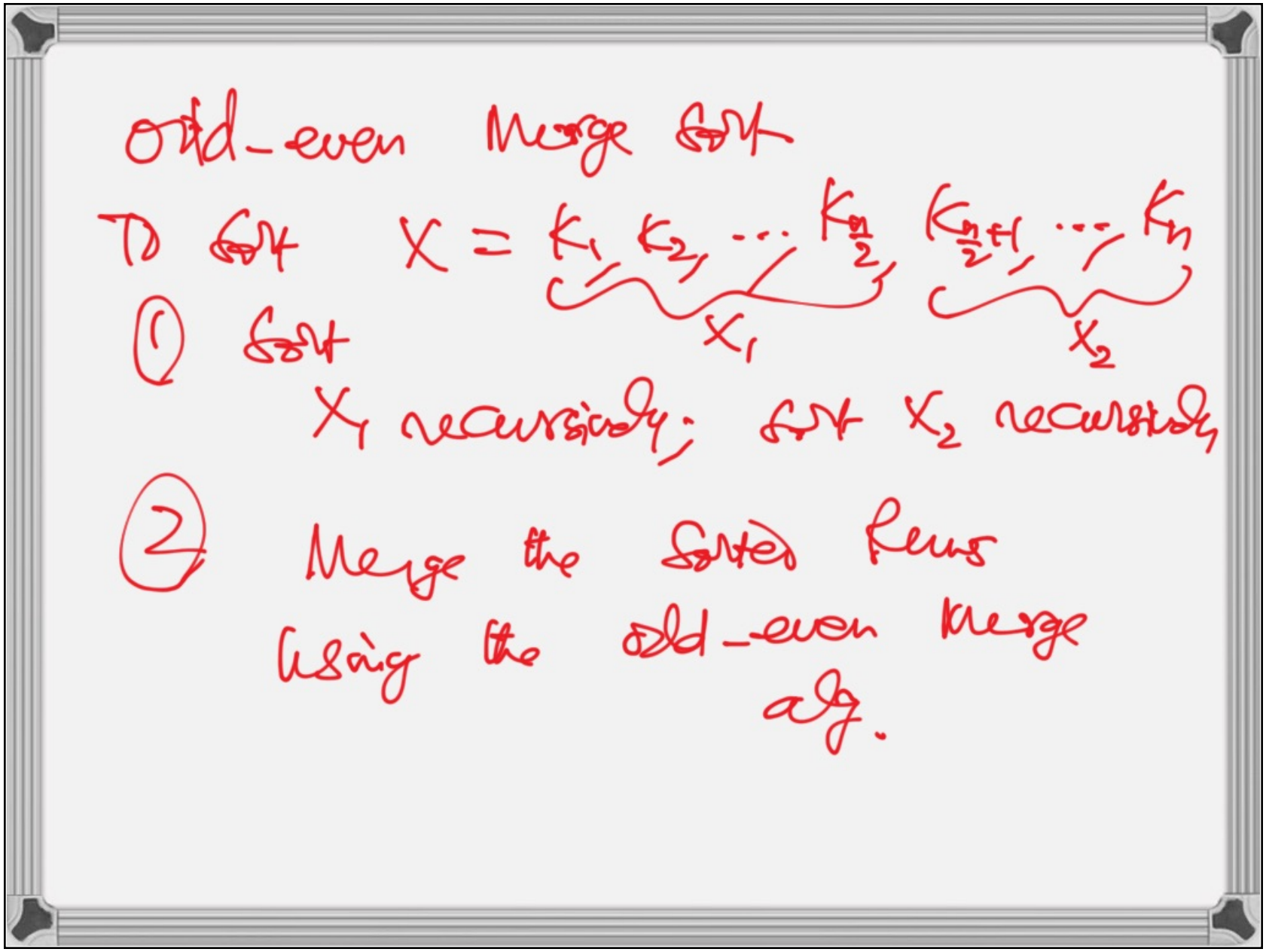

Parallel DBMS Model:
1
2
D
COMPUTER
IN ONE I/O we can bring one block FROM Each DISK

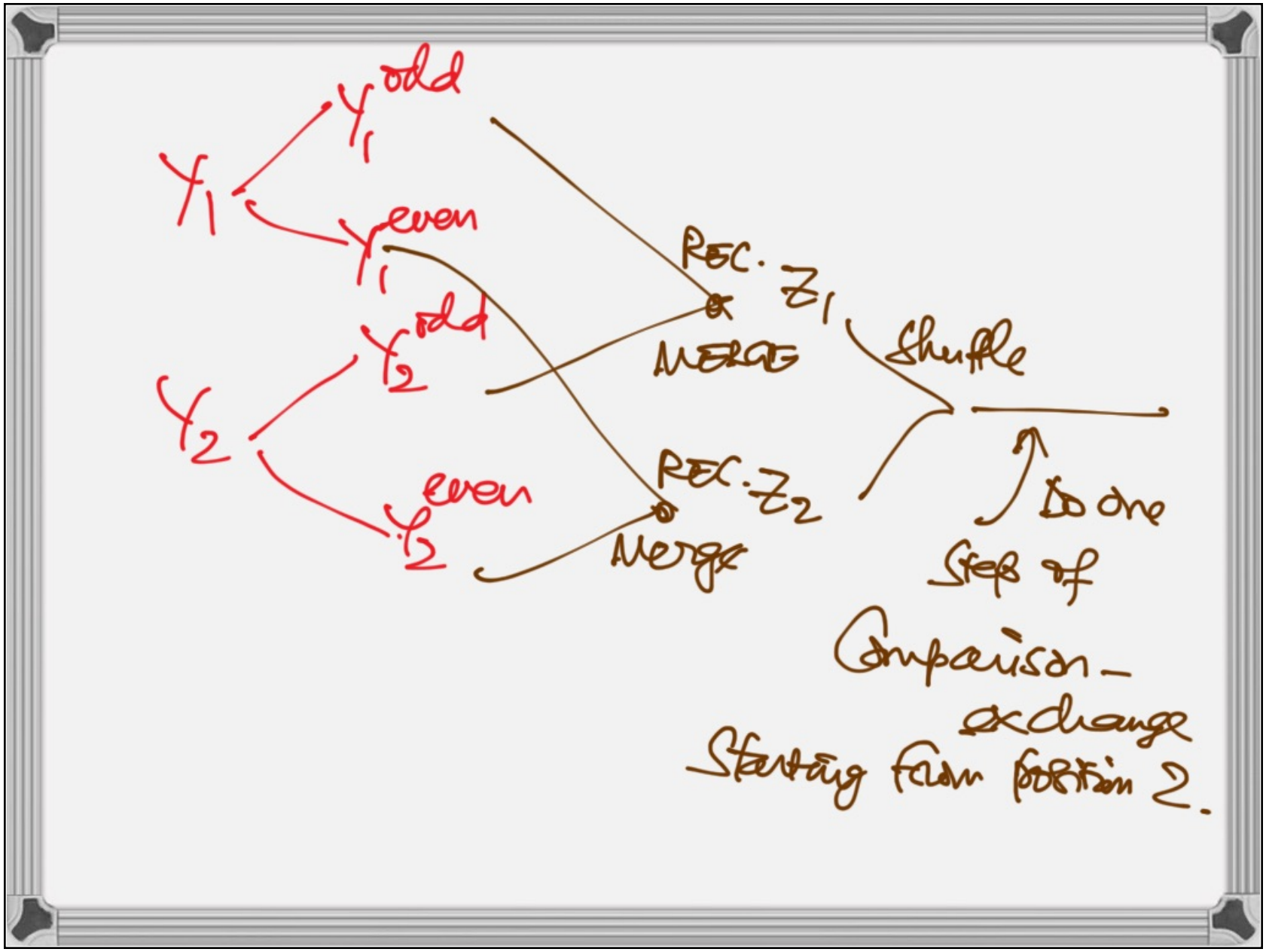

STRIPING
Data
R1
R2
R3
1
2
3
D

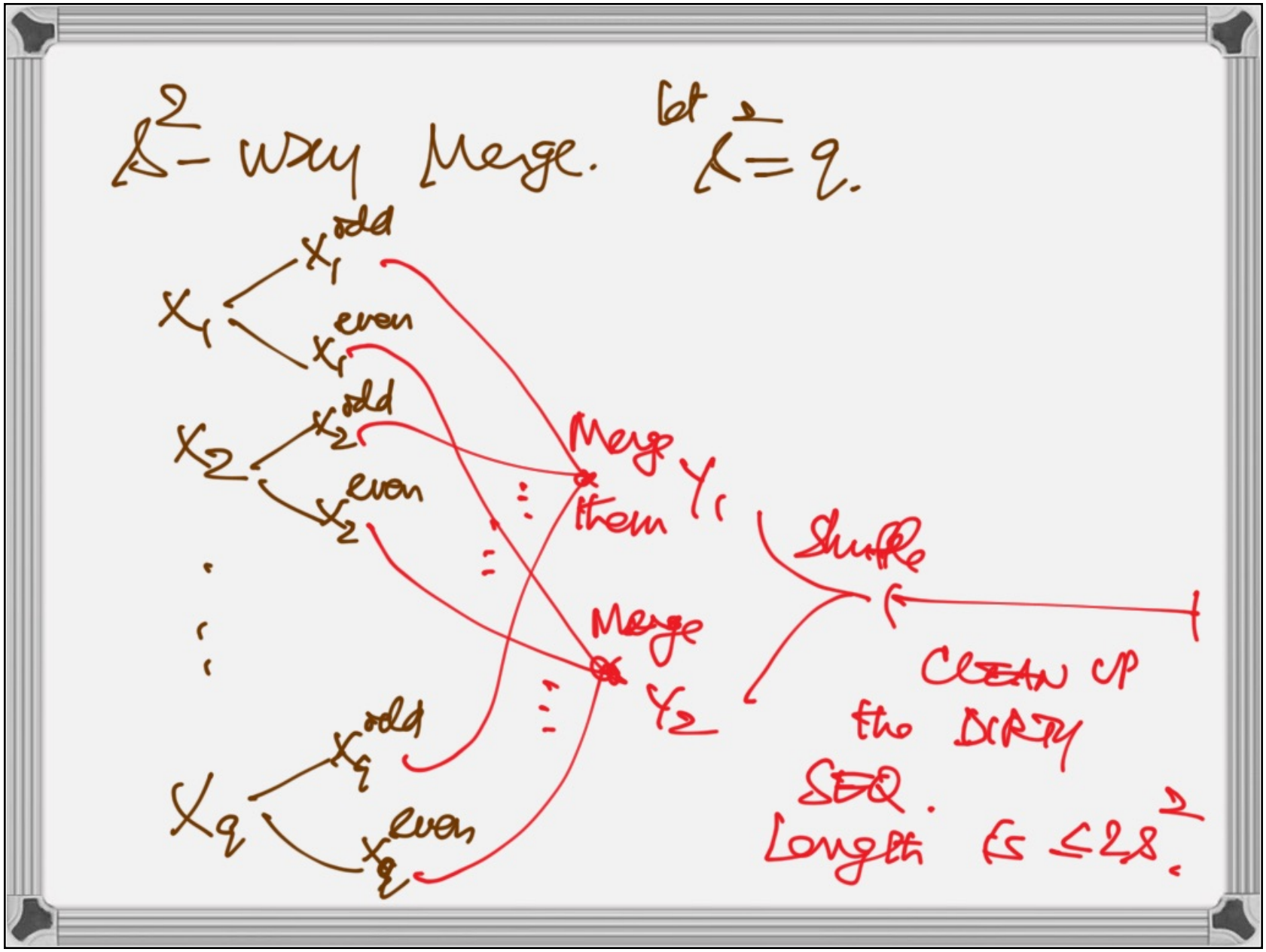

R1
R2
R3
DESIRABLE.

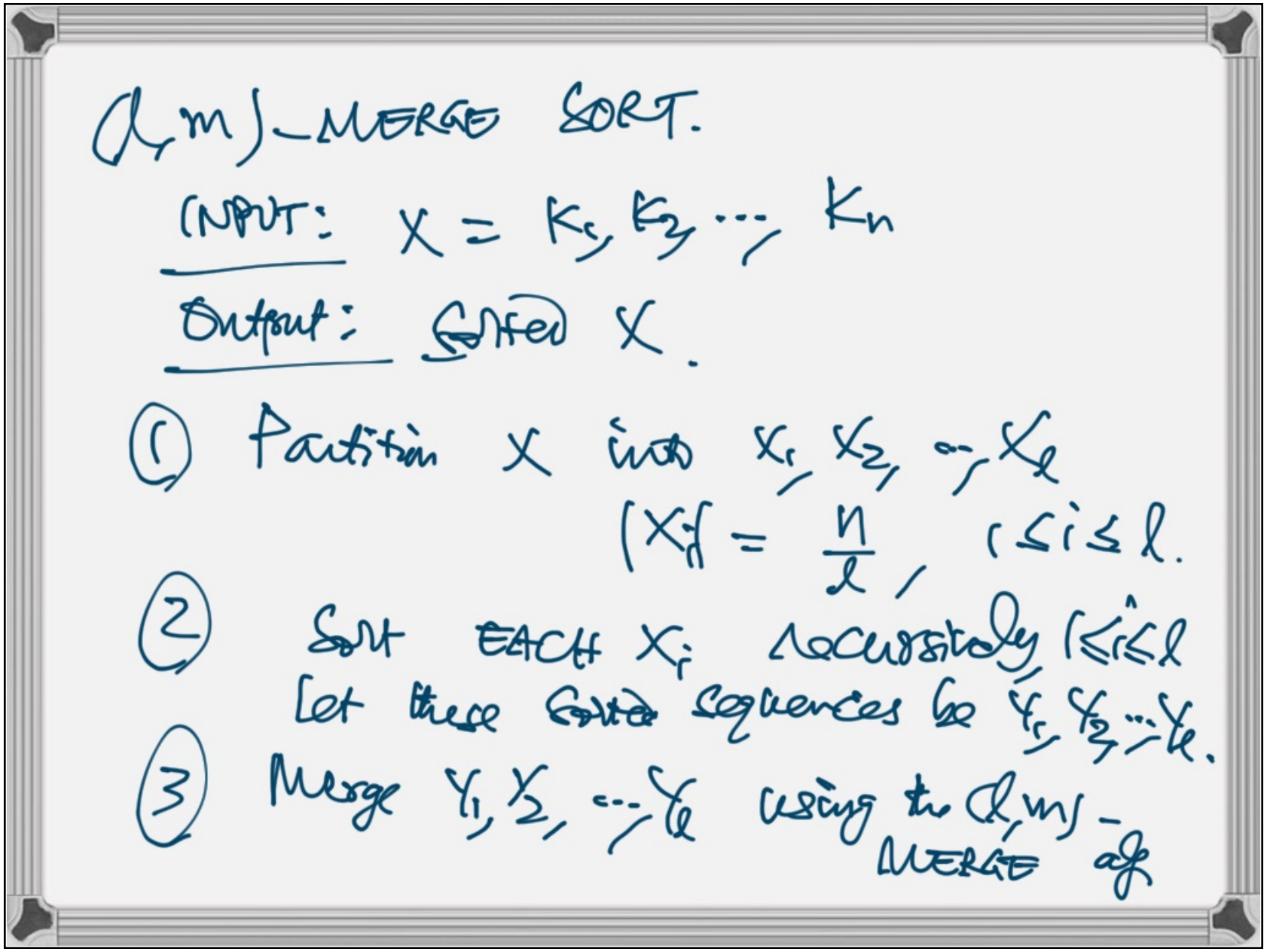

1) Form Runs of length M each.
2) Merge these N/M Runs.
Using k-way MERGE for some Suitable K.
DISK STRIPED MERGE SORT:
Treat the D disks as a single disk with a block size of BD.

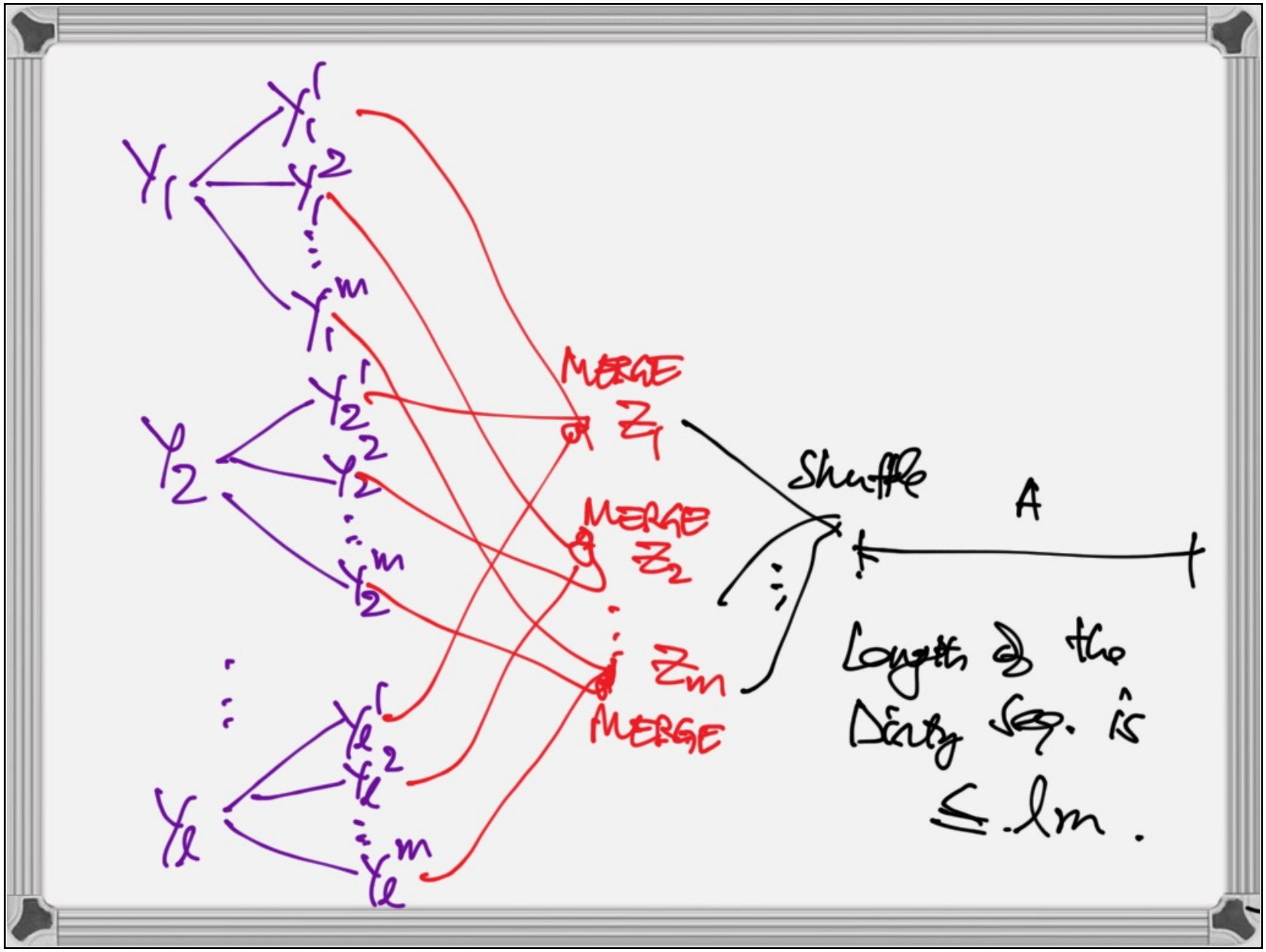

Use $\frac{M}{B}$ - way
Merge.
IR IO Complexity
$= O\left(\frac{\log(N/M)}{\log\left(\frac{M}{B}\right)}\right)$
Runs.
Lower Bound for Sorting: $O\left(\frac{\log(n/m)}{\log(M/B)}\right)$

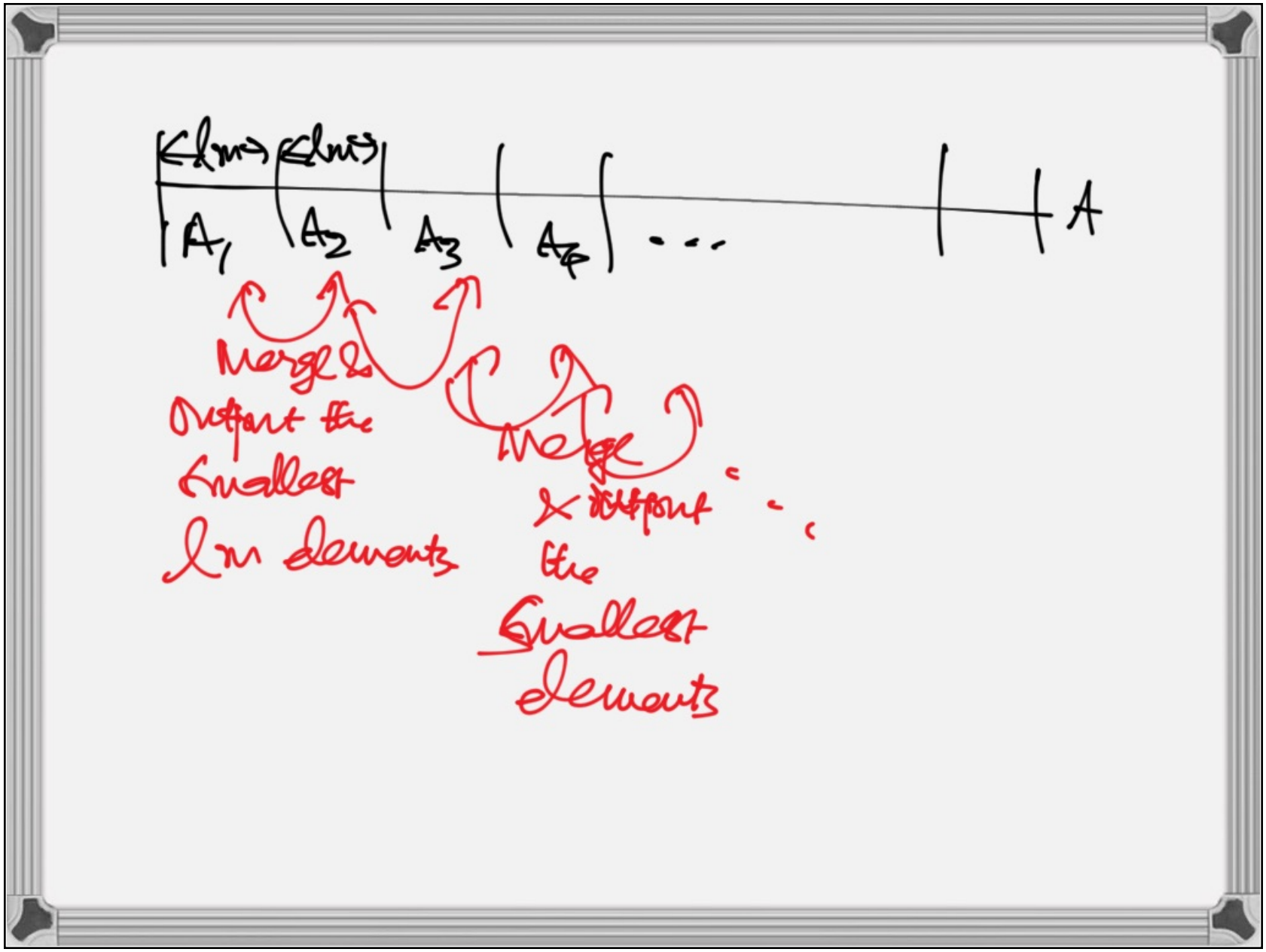

odd-even Merge Sort
To Sort $X = k_1, k_2, \ldots, k_{\frac{n}{2}}, k_{\frac{n}{2}+1}, \ldots, k_n$
X_1
X_2
(1) Sort X_1 recursively; Sort X_2 recursively
(2) Merge the Sorted Runs using the odd-even Merge alg.


odd
even
odd
even
REC. Z1
MERGE
REC. Z2
MERGE
Shuffle
Do one
Step of
Comparison-
exchange
Starting from position 2.


s^2-way Merge.
Let $s = q$.
odd
even
Merge them
Merge
Shuffle
CLEAN UP the DIRTY SEQ.
Length is $\leq 2s^2$.


(k, m)-MERGE SORT.
INPUT: X = K1, K2 ..., Kn
Output: Sorted X.
(1) Partition X into X1, X2, ..., Xl
|Xi| = n/l, 1 ≤ i ≤ l.
(2) Sort EACH Xi recursively, 1 ≤ i ≤ l
Let these sorted sequences be Y1, Y2 ... Yl.
(3) Merge Y1, Y2, ..., Yl using the (k, m)-
MERGE alg


MERGE
MERGE
MERGE
Shuffle
A
Length of the
Derby seq. is
≤ lm.


A1
A2
A3
A4
...
A
Merge &
output the
smallest
2 in elements
Merge
& output
the
smallest
elements

Parallel I/O Complexity

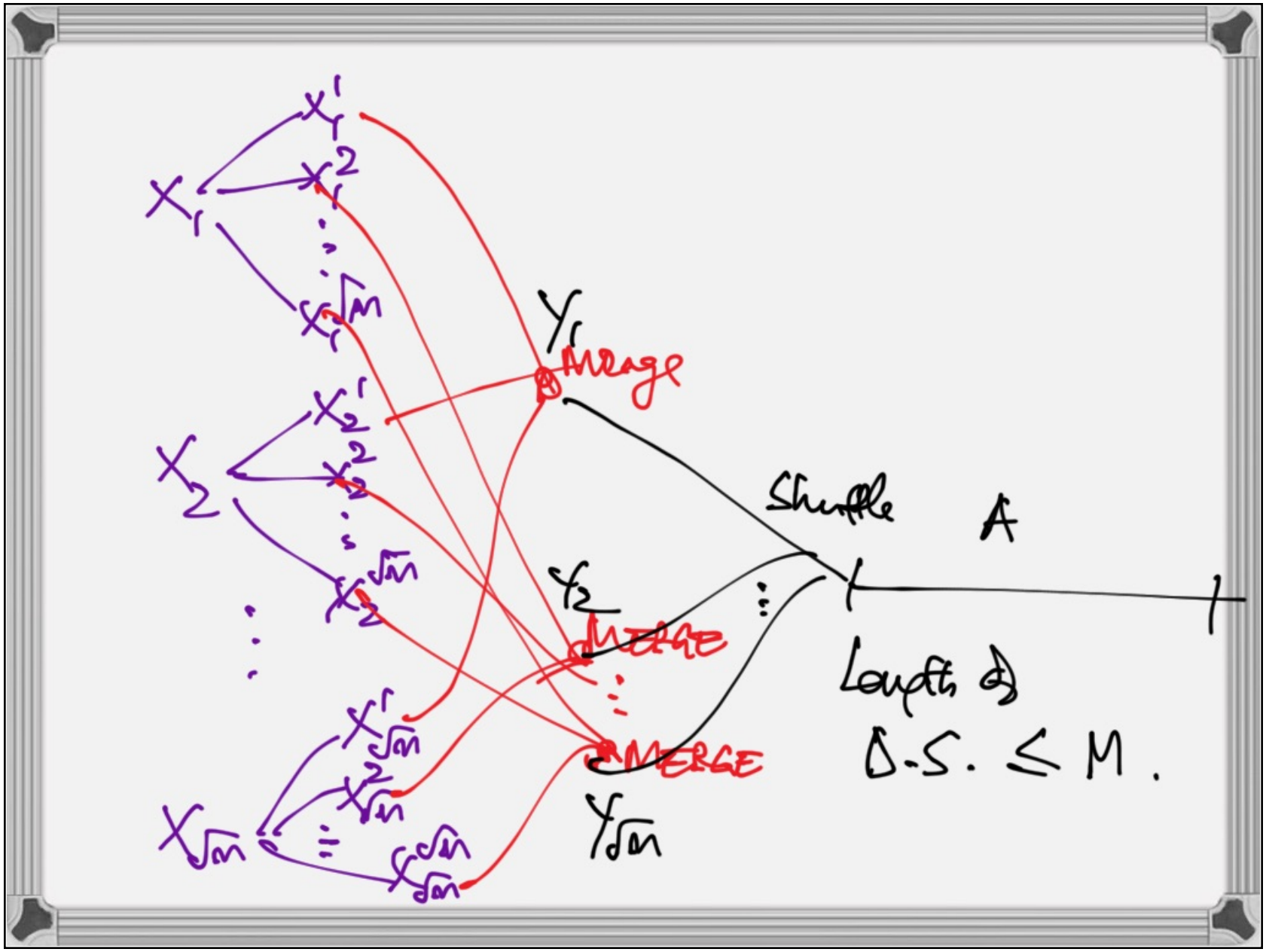
$$= \left\lceil \frac{\log(n/M)}{\log\left(\min\left\{\sqrt{n}, \frac{M}{B}\right\}\right)} + 1 \right\rceil^2.$$

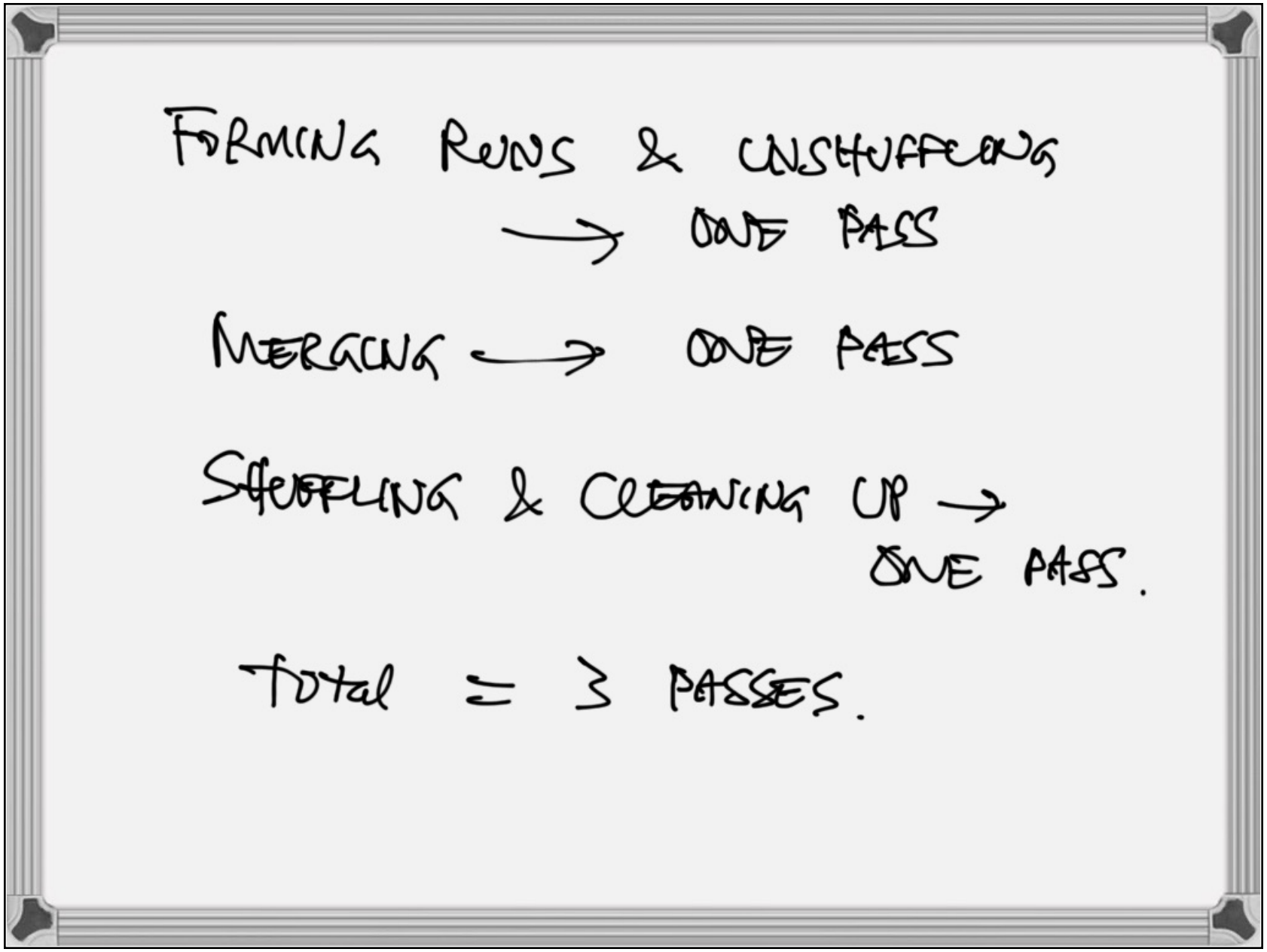
Example: Sort $M\sqrt{M}$ elements where $B = D = \sqrt{M}$.

(1) Form $\sqrt{M}$ runs of length M EACH. Let the runs be

$X_1, X_2, \ldots, X_{\sqrt{M}}$

(2) Merge these runs using (l, m)-MERGE adj. with $l = m = \sqrt{M}$.


Merge
Shuffle
A
Length of
O.S. ≤ M.
MERGE


FORMING RUNS & UNSHUFFLING
→ ONE PASS
MERGING → ONE PASS
SHUFFLING & CLEANING UP →
ONE PASS.
TOTAL = 3 PASSES.

SUFFIX TREES.

$T = a_1 a_2 \ldots a_m \in \Sigma^m$

Suffix tree on T is a rooted tree S.t.

(1) $\exists$ a leaf for every suffix of T

(2) Every internal node has a degree of ≥ 2

(3) Every edge is labelled with a substring of T

(4) For any node the labels on no two outgoing edges can start with the same character

(5) We can label the leaves with integers in the range [1, m].

(6) The concatenation of the edge labels starting from the root

and ending in a leaf labelled i, will be the i^{th} suffix of T i.e $a_i a_{i+1} \cdots a_n$.

We can construct a suffix tree on T in $O(n)$ time.

We always put a \$ @ the end of every string, where \$ $\notin \Sigma$.

PROBLEMS:

(1) STRING MATCHING.

INPUT: $T = t_1 t_2 \dots t_m$ $\quad m \gg n.$

$P = p_1 p_2 \dots p_n$

Output: All the occurrences of P in T.

Construct a Suffix tree Q on T.
Start from the root & keep to find a match for P along a path.

RUN TIME = $O(m + k + n)$ — k: # of matches.

Given Multiple Strings $S_1, S_2, \ldots, S_q$

we can Construct a G.S.T. on $S_1, S_2, \ldots, S_q$ in time

$$O\left(\sum_{i=1}^{q} |S_i|\right).$$

EVERY is labelled with a pair (i,j).

$\Rightarrow$ this leaf corr. to the j^{th} suffix of S_i.

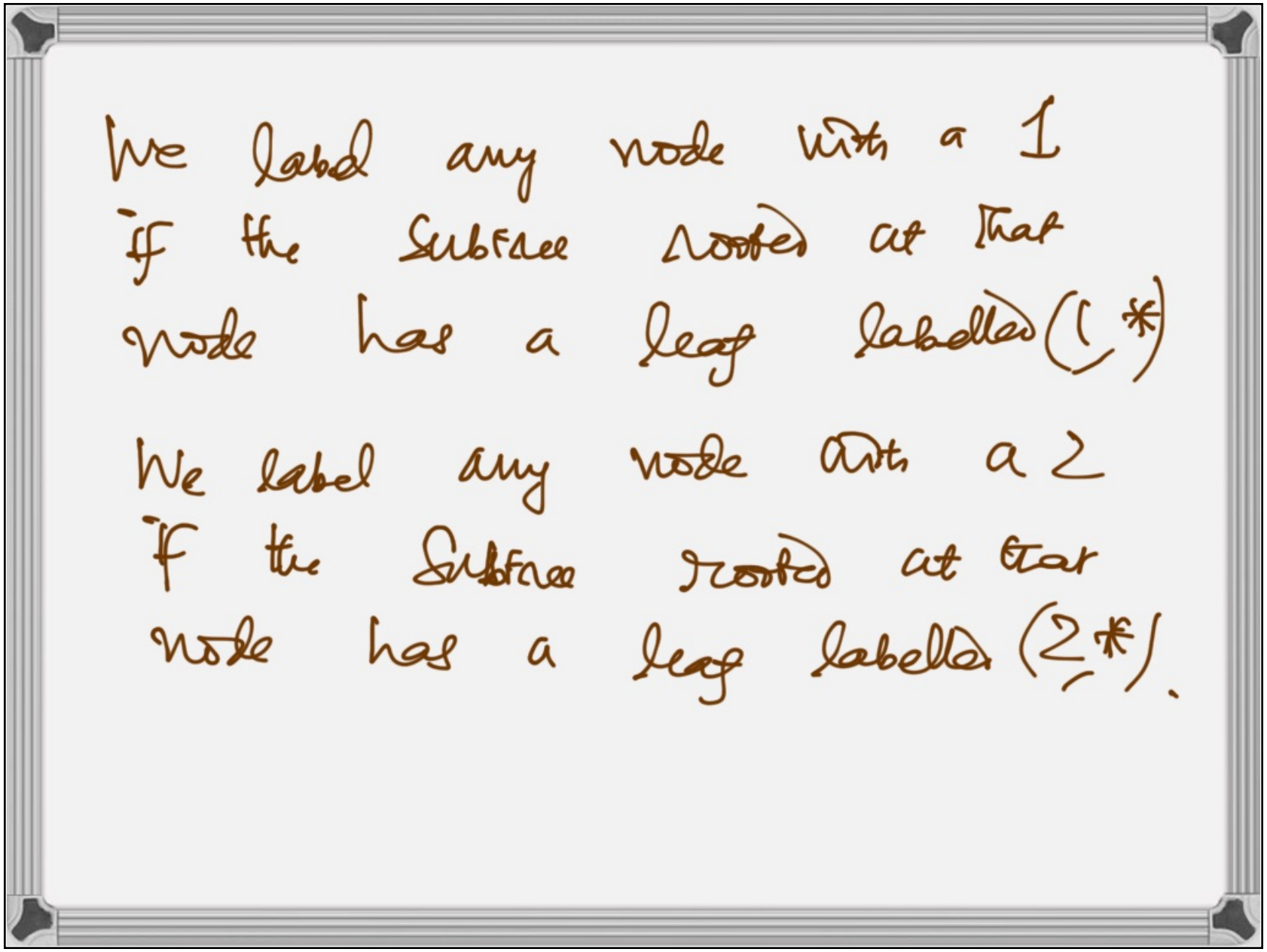
LONGEST COMMON SUBSTRING:

INPUT: S_1, S_2.

Output: LCS of S_1 & S_2

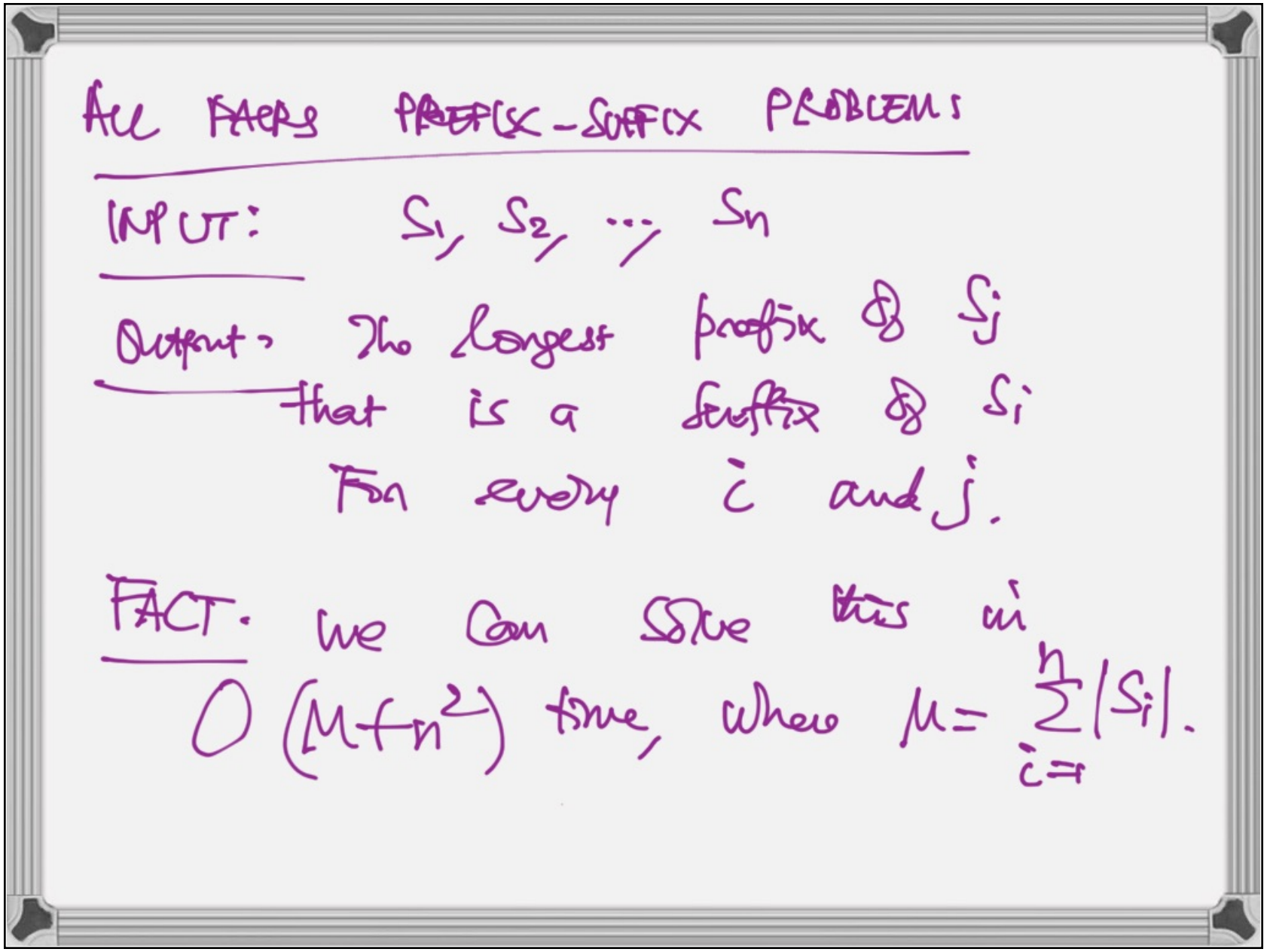
IDEA. Construct a G.S.T. Q on S_1 and S_2. $\Rightarrow$ TIME $= O(n)$

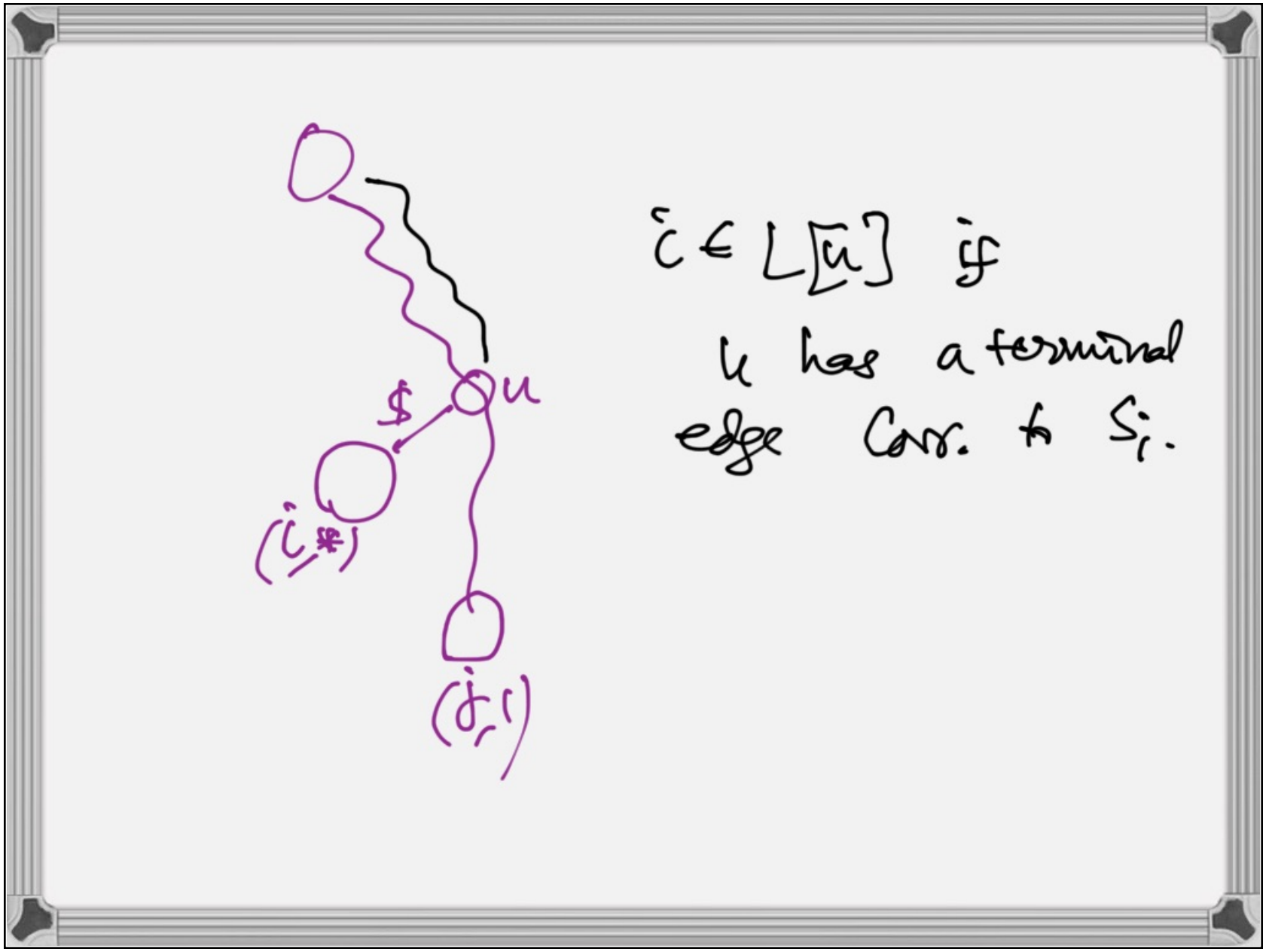
If $|S_1| = |S_2| = n$.


We label any node with a 1
if the subtree rooted at that
node has a leaf labelled (1,*)
We label any node with a 2
if the subtree rooted at that
node has a leaf labelled (2,*).

Traverse Q one more time to look for the node u that is labelled with 1 and 2 and whose starting depth is the largest. Output the path label of u.

Run time = $O(n)$.


ALL PAIRS PREFIX-SUFFIX PROBLEM:
INPUT: $S_1, S_2, \ldots, S_n$
Output: The longest prefix of S_j that is a suffix of S_i For every i and j.
FACT. We can solve this in $O(M+n^2)$ time, where $M = \sum_{i=1}^{n} |S_i|$.

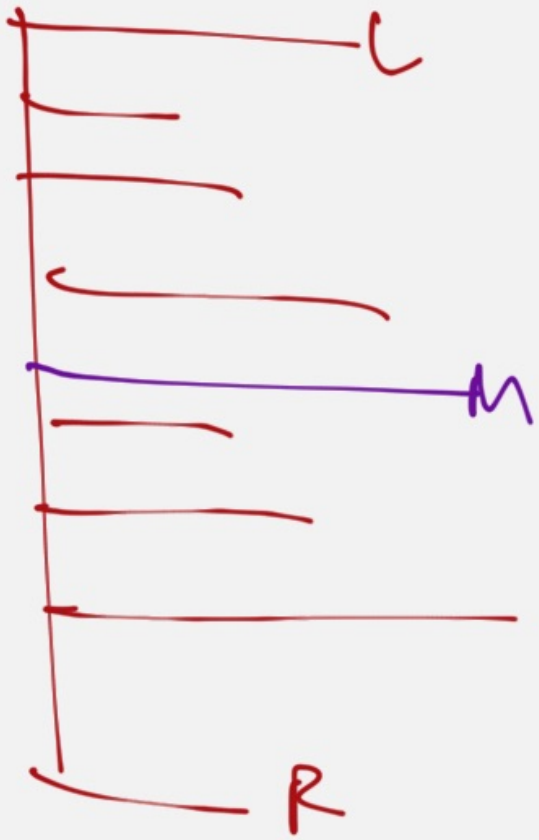

$
u
(i, *)
(j, 1)
i ∈ L[u] if
u has a terminal
edge Corr. to S_i.

Suffix Arrays:

$T = t_1 t_2 \ldots t_m$

Suffix Array for T is an Array SA [1:m] of integers, s.t. SA[i] is the starting position in T of the i^{th} smallest suffix of T.

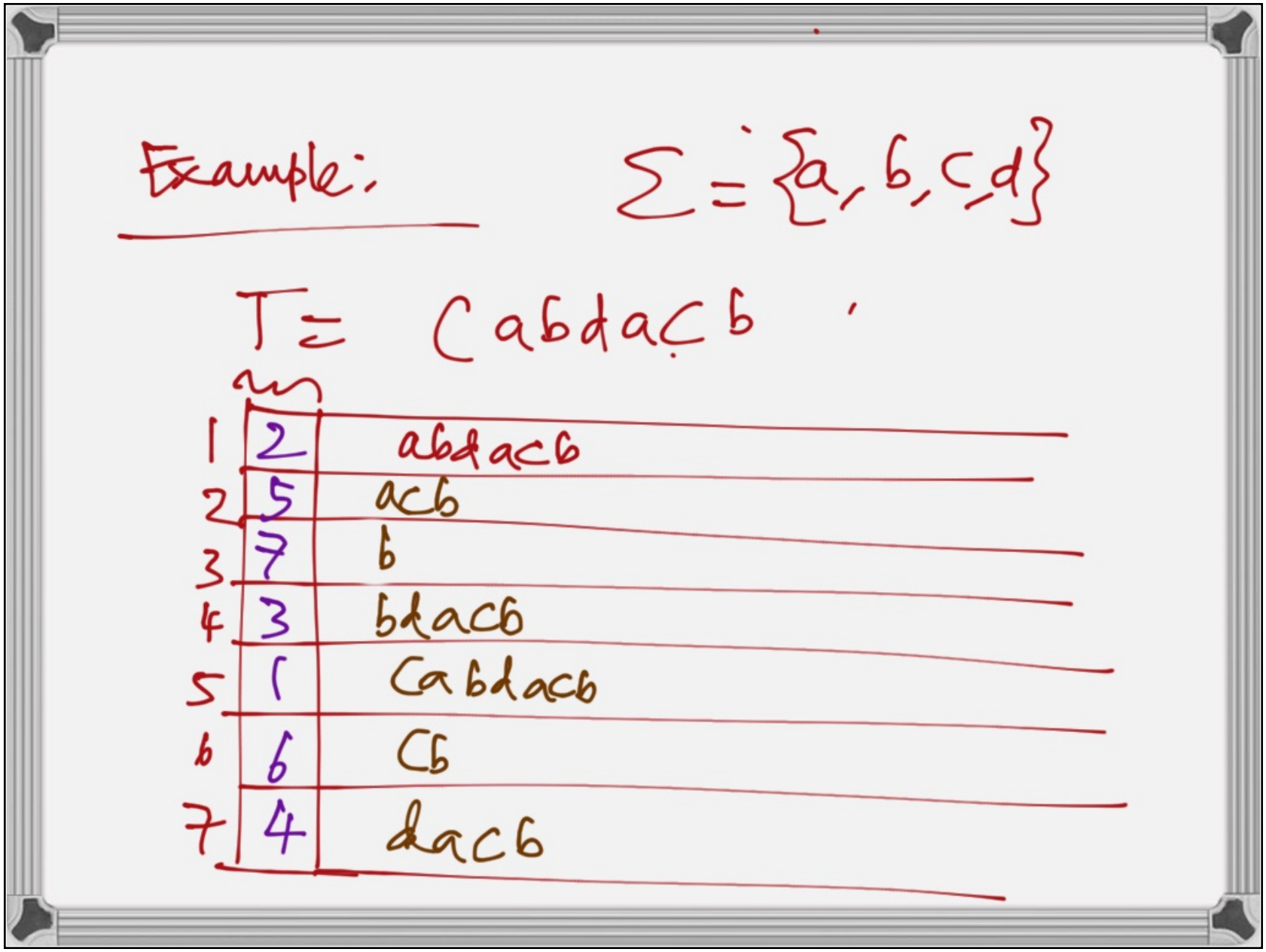
STRING MATCHING USING A SUFFIX ARRAY:

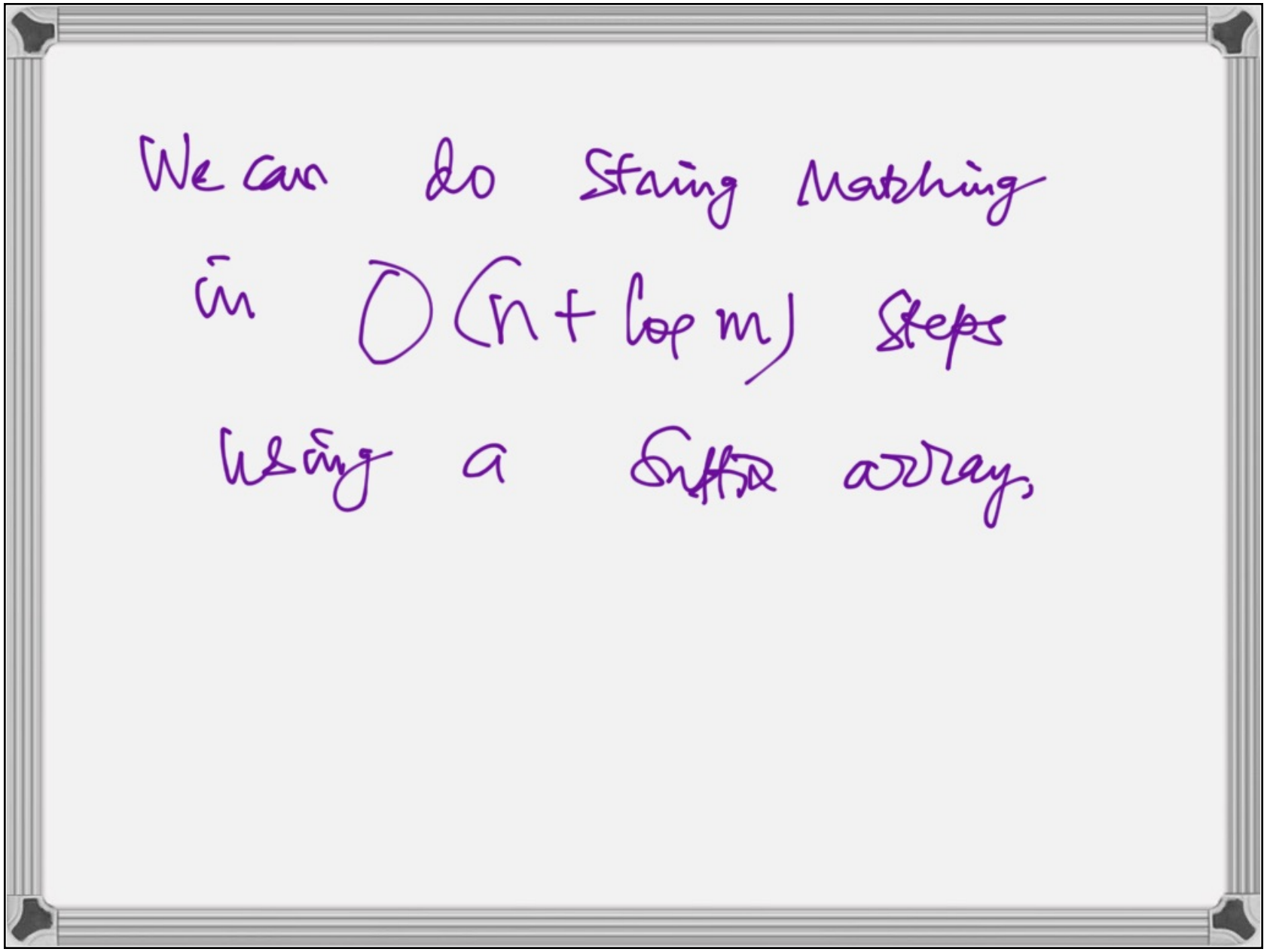


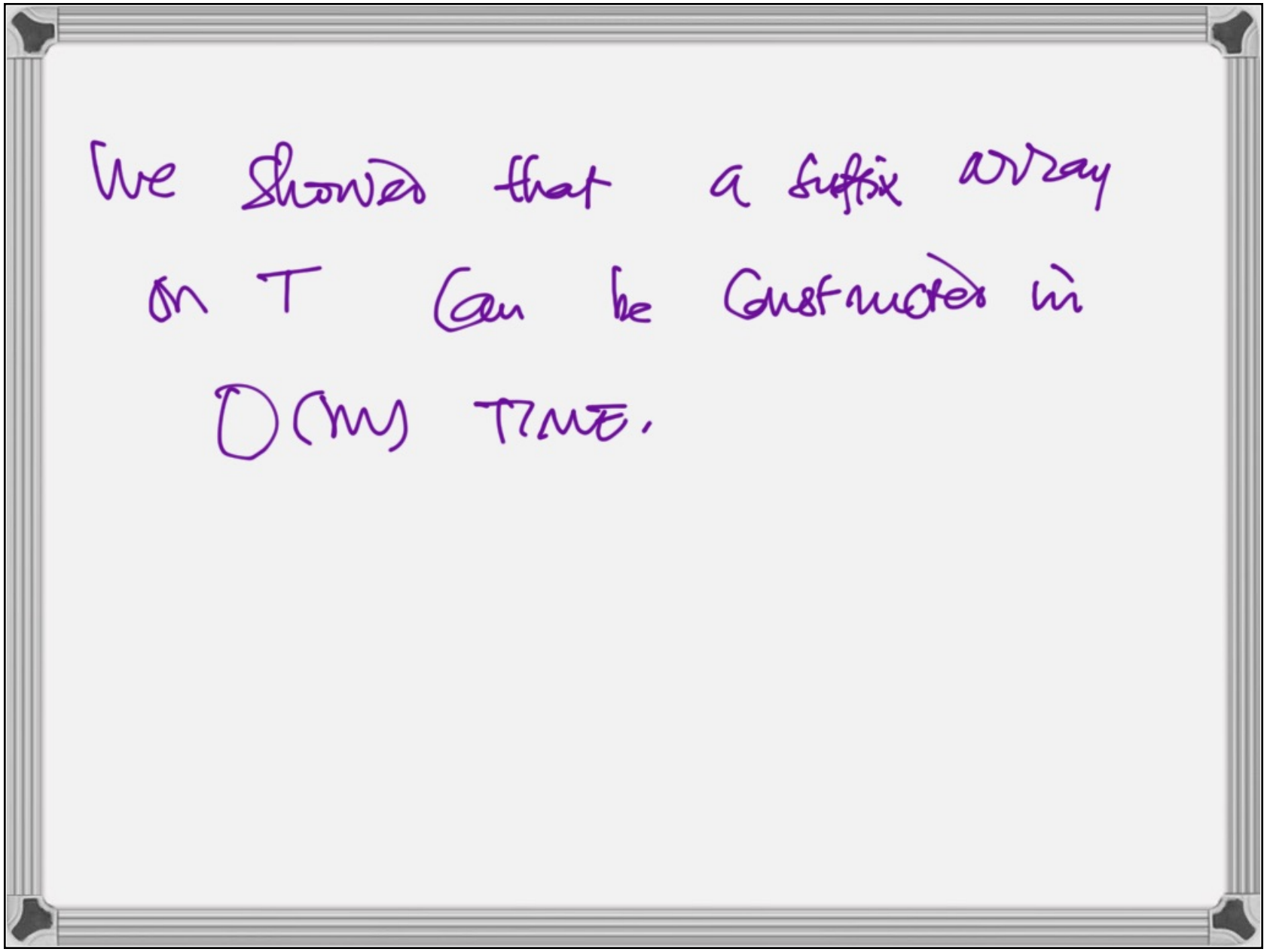
P

TIME FOR BINARY SEARCH

$= O(n \log m)$


Example:
Σ = {a, b, c, d}
T = cabdacb
sn
1 2 abdacb
2 5 acb
3 7 b
4 3 bdacb
5 1 cabdacb
6 6 cb
7 4 dacb


We can do String Matching
in O(n + log m) steps
using a Suffix array.


We showed that a suffix array
on T can be constructed in
O(n) time.

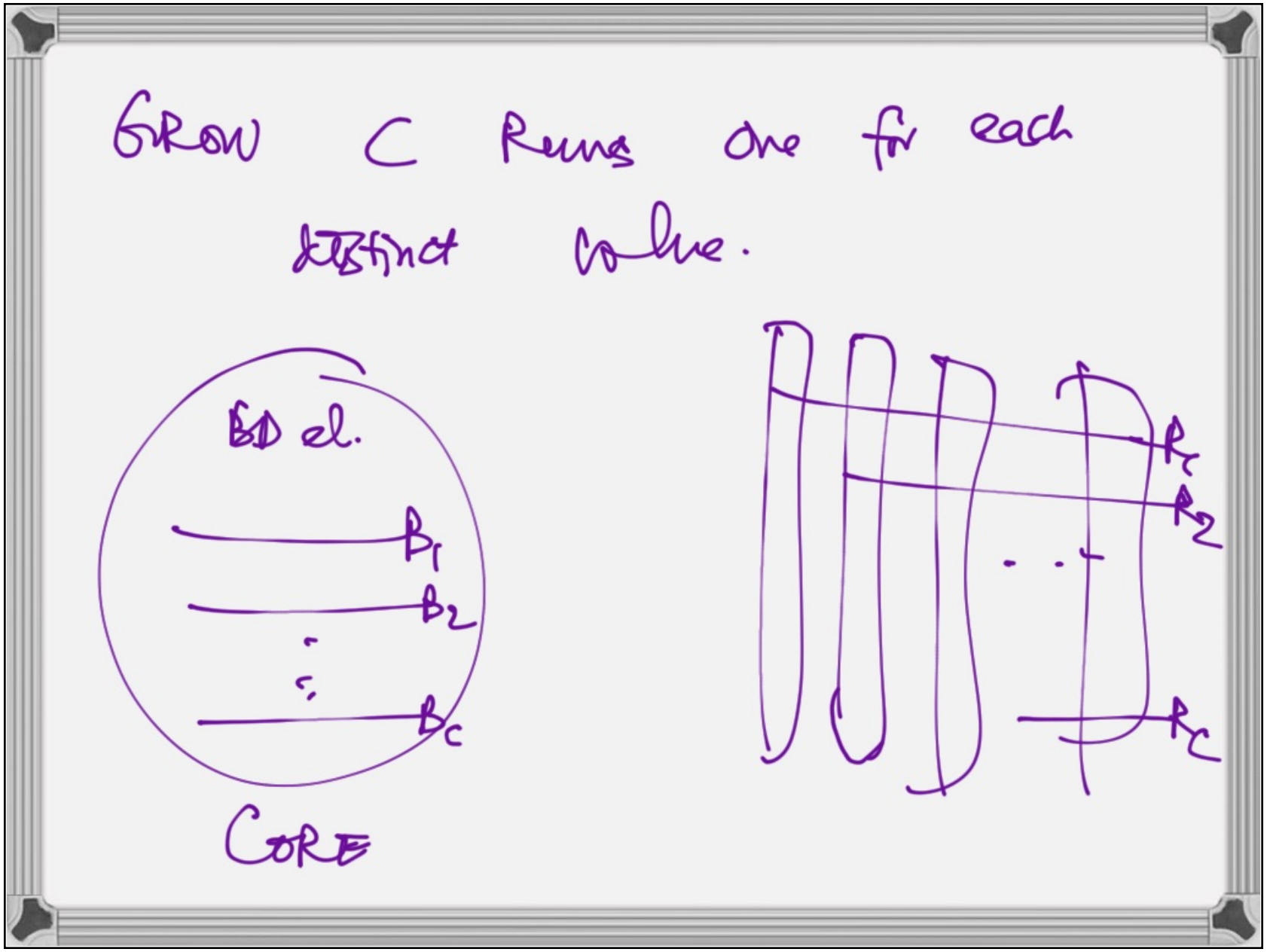
Model Exam 2:

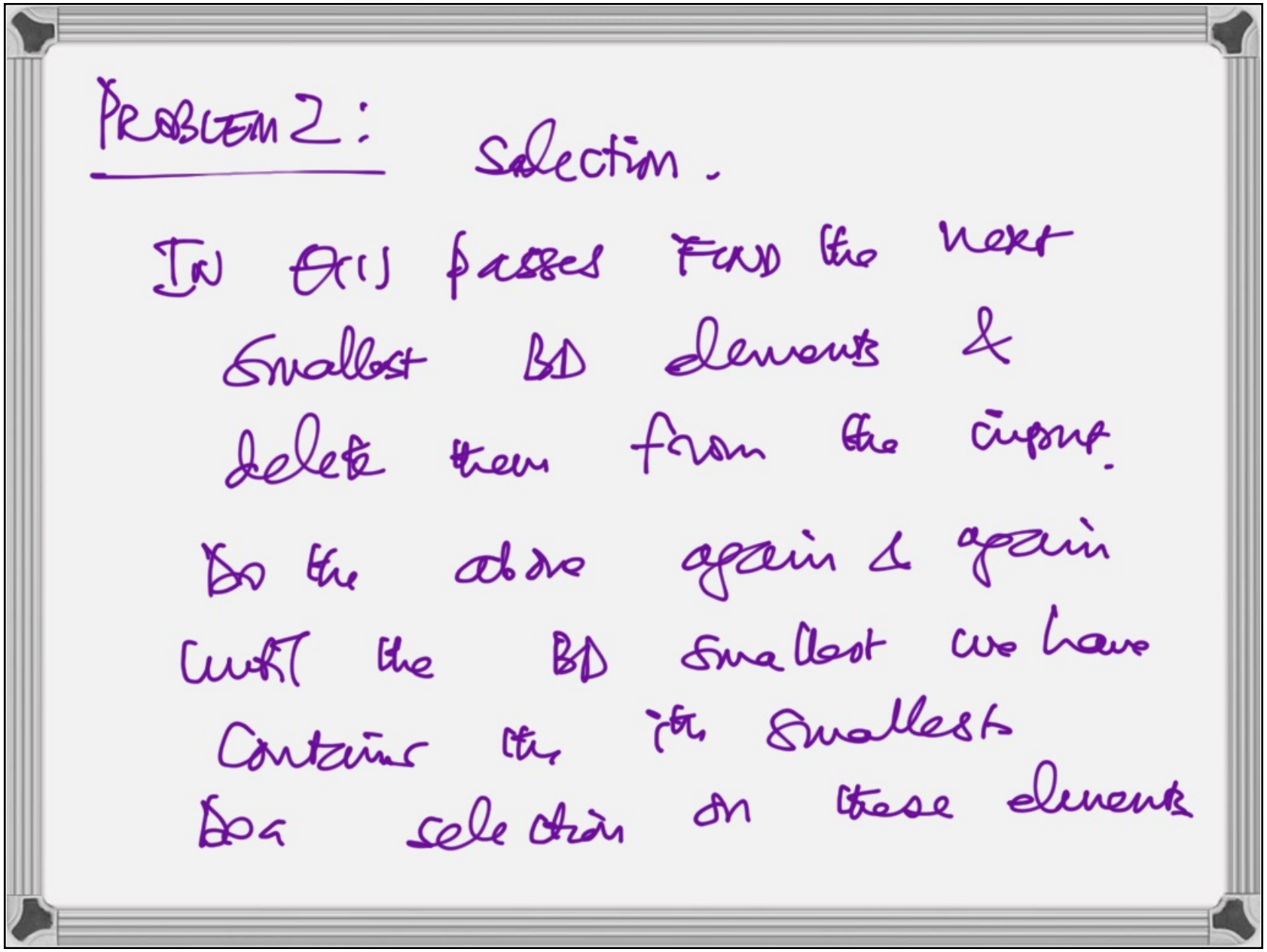
INPUT: $X = K_1, K_2, \ldots, K_n$.

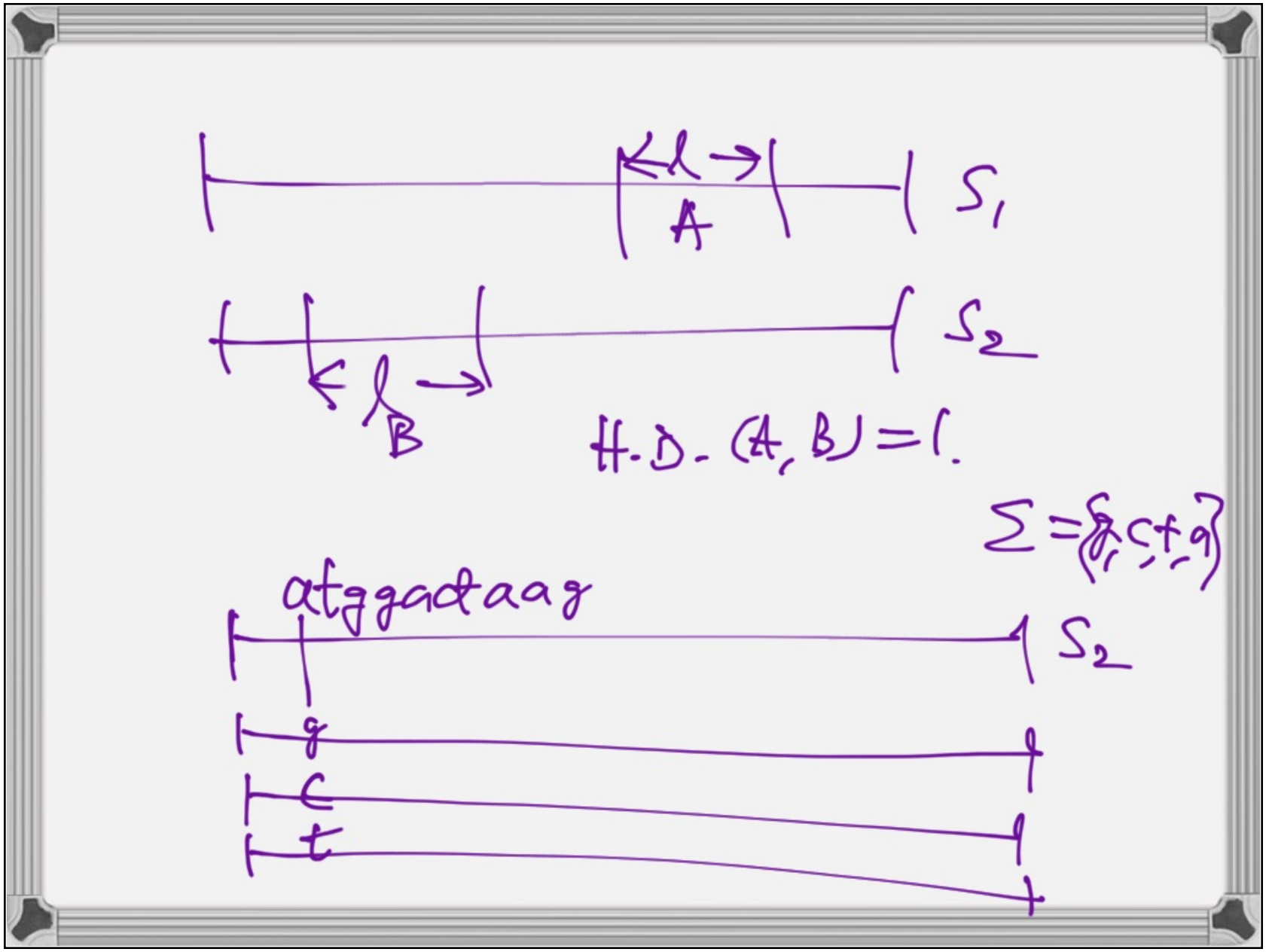
# of distinct elements = C.

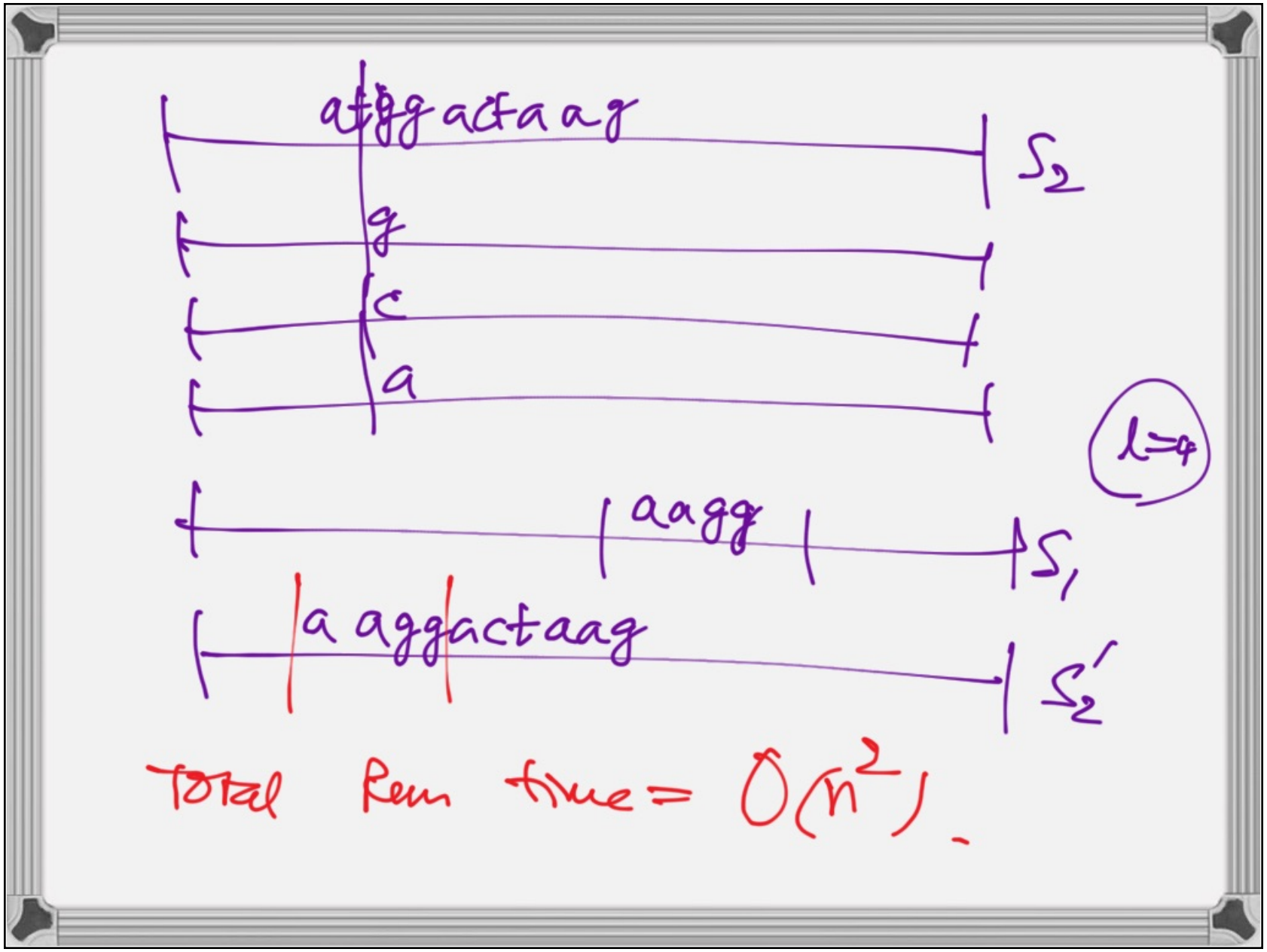
Sort in two passes.

IDEA: Bring BD elements @ a time.


GROW C Runs one for each
distinct value.
BD el.
B1
B2
Bc
CORE
R1
R2
Rc


PROBLEM 2: Selection.
In O(1) passes FIND the next
smallest BD elements &
delete them from the input.
Do the above again & again
until the BD smallest we have
Contains the ith smallest
Do a selection on these elements


l
A
S_1
S_2
l
B
H.D. (A, B) = 1.
$\Sigma = \{a, c, t, g\}$
atggactaag
S_2
g
c
t


aggactaag
S_2
g
c
a
l=4
aagg
S_1
aaggactaag
S_2'
Total Run time = $O(n^2)$.

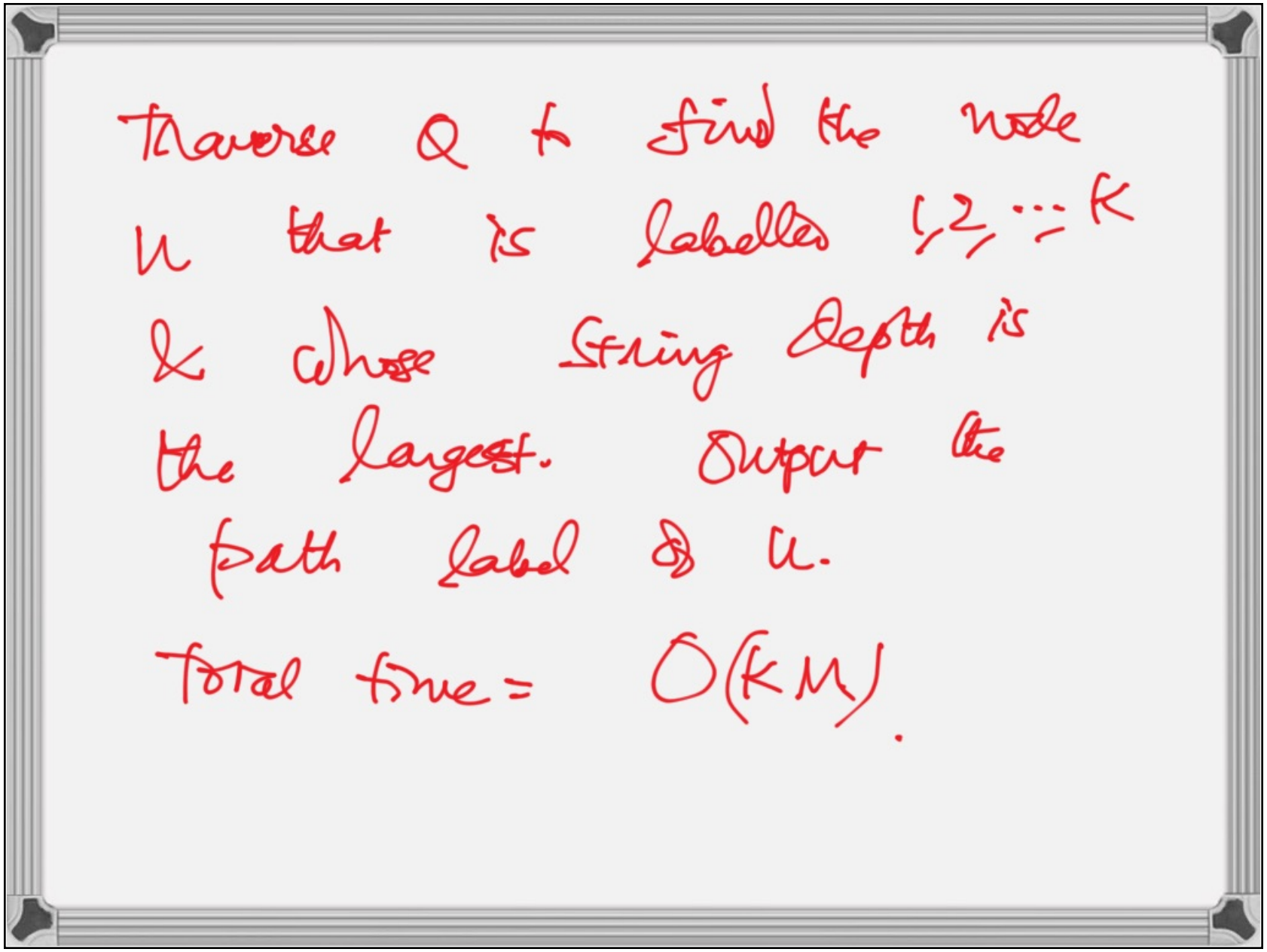
PROBLEM 4.

CONSTRUCT A GST ON $S_1, S_2, \ldots, S_k$.

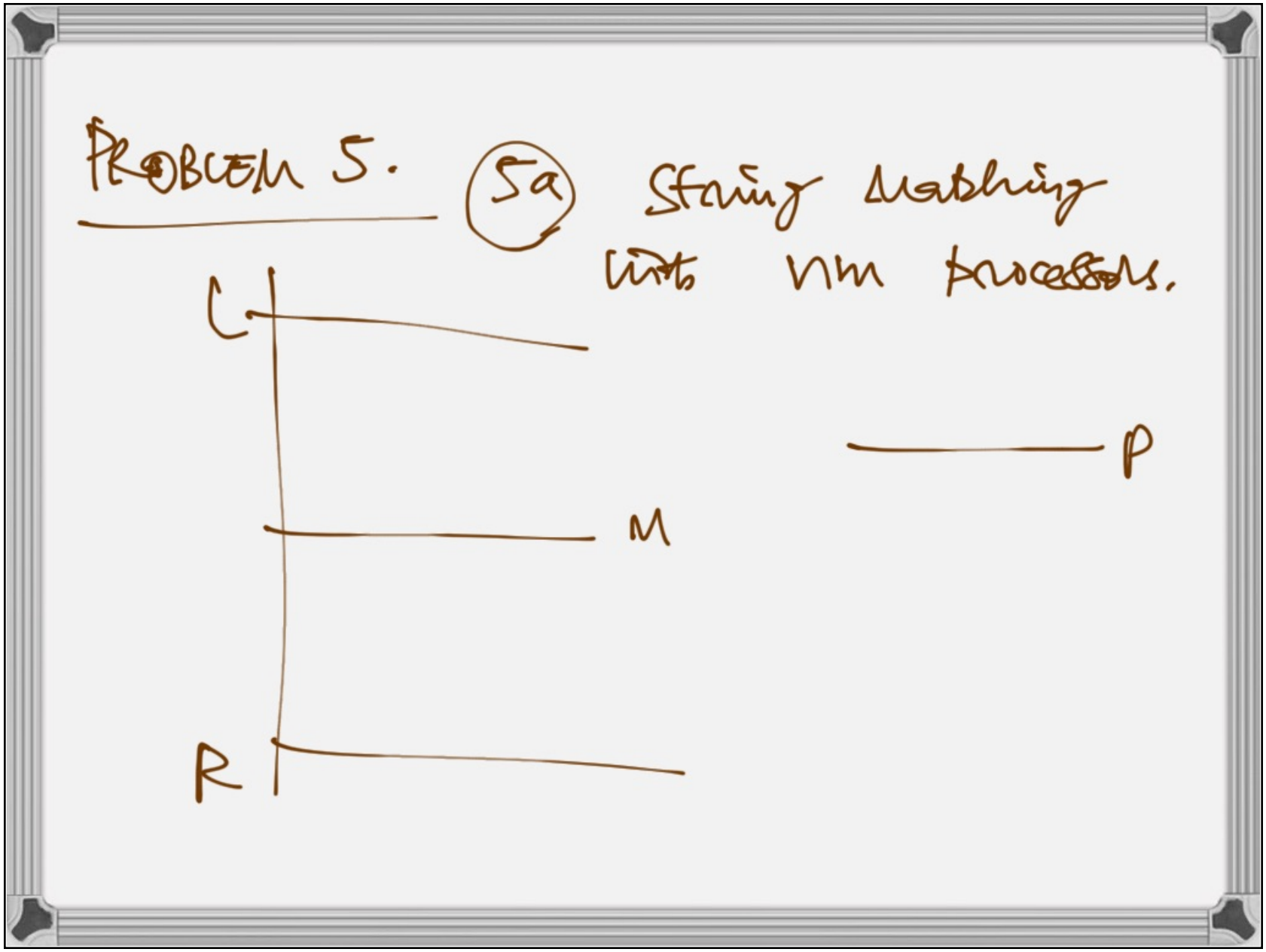
TIME = $O(M)$.

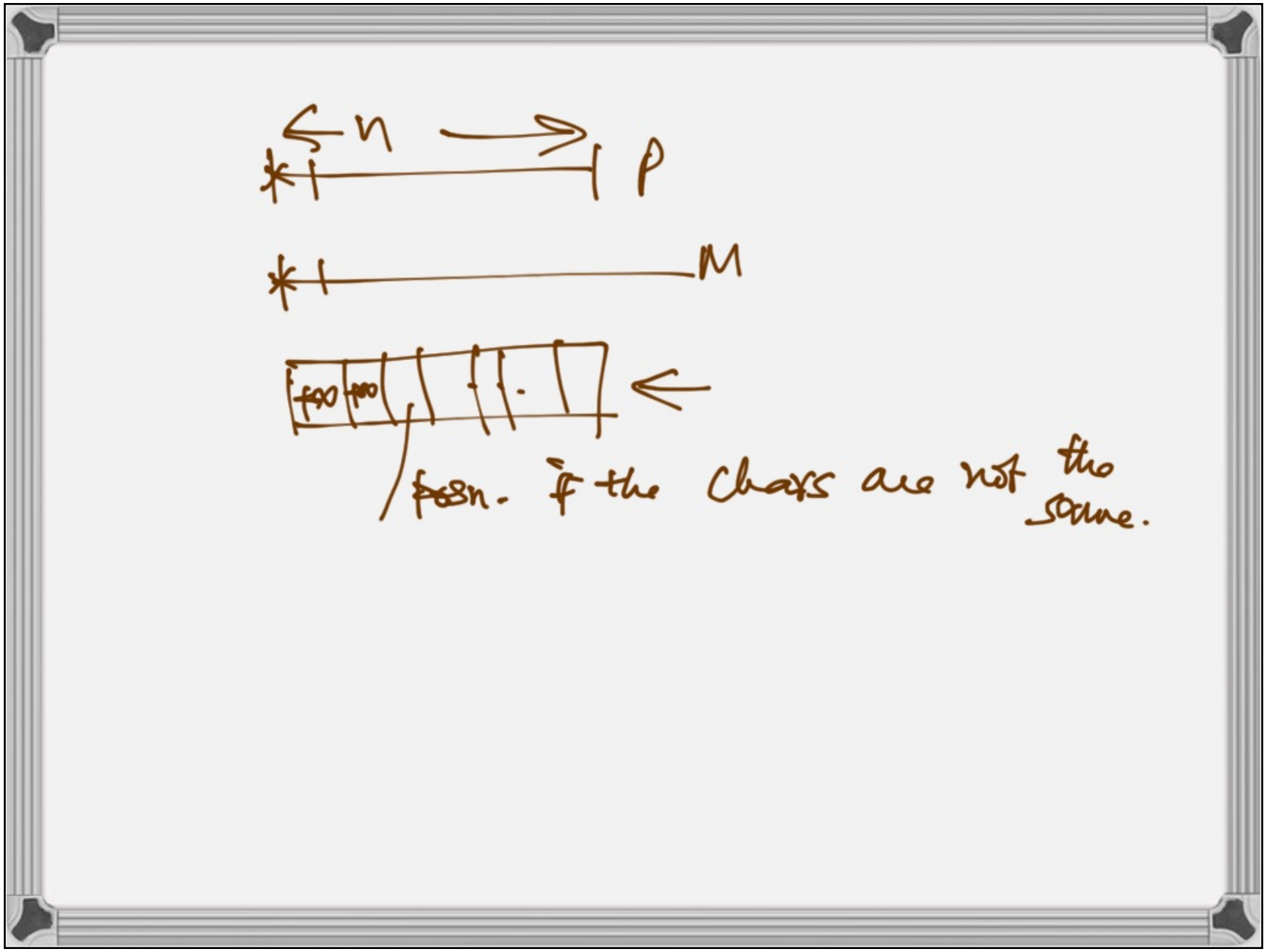
FOR $1 \le i \le k$ do

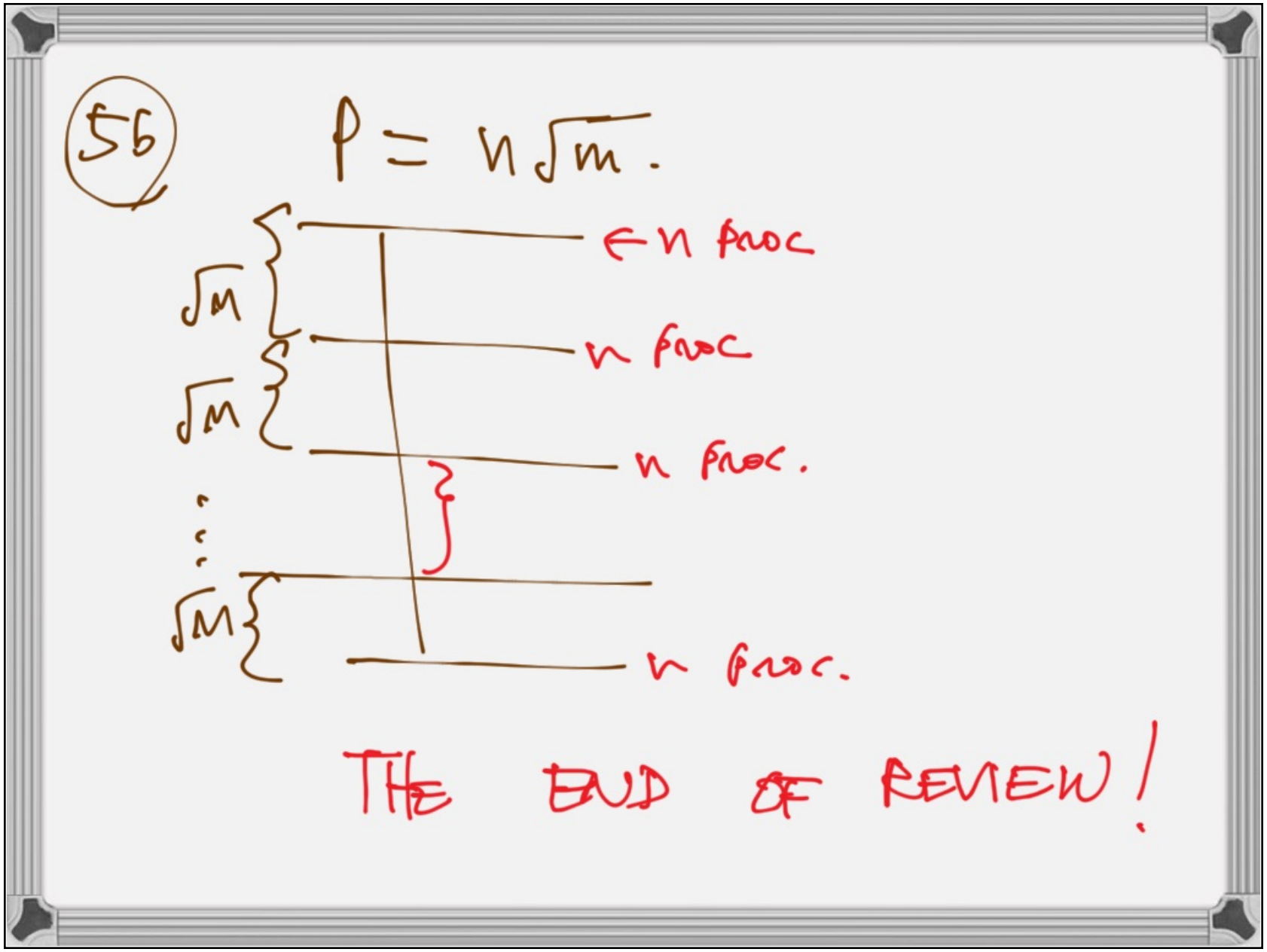
Traverse Q. Label a node N with i if its subtree has a leaf corr. to S_i


Traverse Q to find the node
u that is labelled 1, 2, ... K
& whose string depth is
the largest. Output the
path label of u.
Total time = O(KM).

PROBLEM 5. (5a) Starting Matching Units Nm Processes.




n
p
M
foo foo
foo. if the chars are not the same.


56
P = n√m.
∈ n proc
√m
n proc
√m
n proc.
√m
n proc.
THE END OF REVIEW!