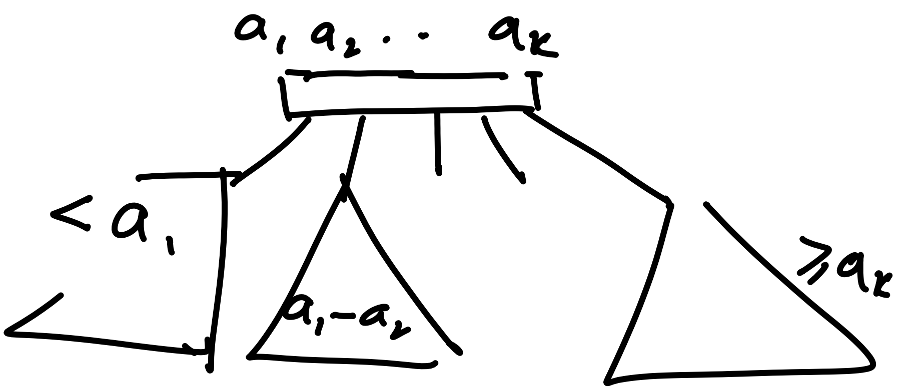
Lecture 22, Oct 27, CSL630

B trees vs. Skip Lists

William Pugh



$\log_k n$

k-ary search tree
(k = B)

$k \to \log_2 k$

$\log_k n \times \boxed{\text{time to determine which subtree}}$

$$\frac{\log n}{\log k} \times k$$

$$\frac{\log n}{\log k} \cdot \log k \quad - \quad \log n$$

Most keys will be in secondary memory when # keys is very large

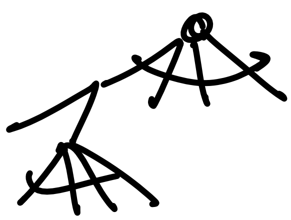
Performance is determined by # accesses to secondary memory (slower by $10^3 - 10^4$)

External memory model

2 levels — primary 0 cost
— secondary 1 cost

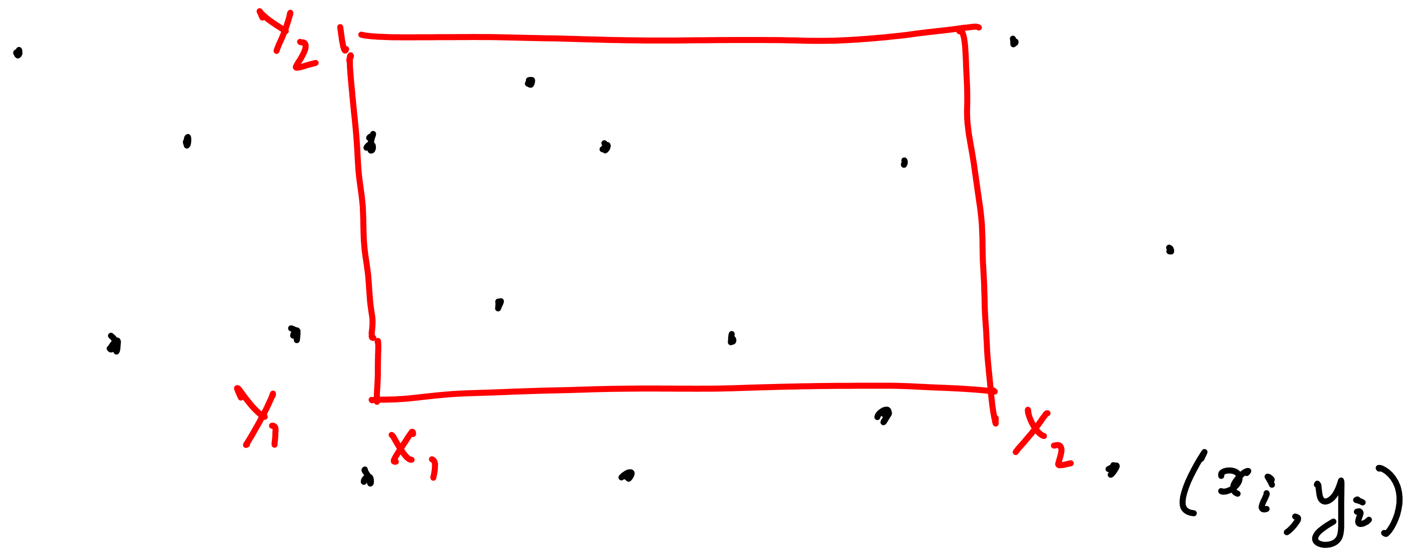
Goal is to minimize memory cost

Skip Lists can thought of as
a K-ary search tree tree
where the expected value of $K = 2$



The probability that search in
skip lists exceeds $c \cdot 2 \log n$ is
less than $\frac{1}{n^{\alpha}}$

(Inverse polynomial bound)



$x_i \in$ from X $\quad y_i \in Y$

Report # points in the range $[X_1, X_2] \times [Y_1, Y_2]$

counting

- the actual data ← reporting version

An obvious method is brute force

$\Rightarrow$ $O(n)$ time

It may be acceptable when output (answer) is large

"Output-sensitive" query algorithm

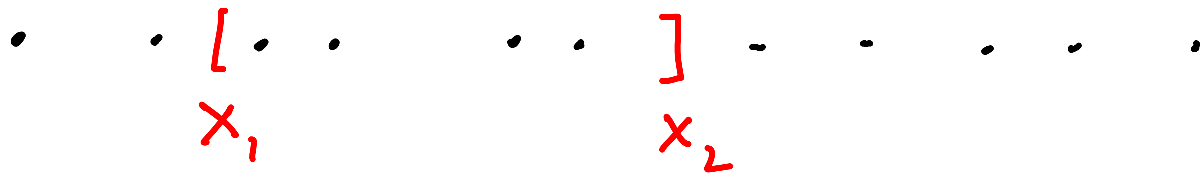
The time to search would depend on the size of the output

$$O(f(n) + k)$$

output size

fixed cost usually encountered in counting version also

One dimensional problem



Variation of binary search $\Rightarrow$

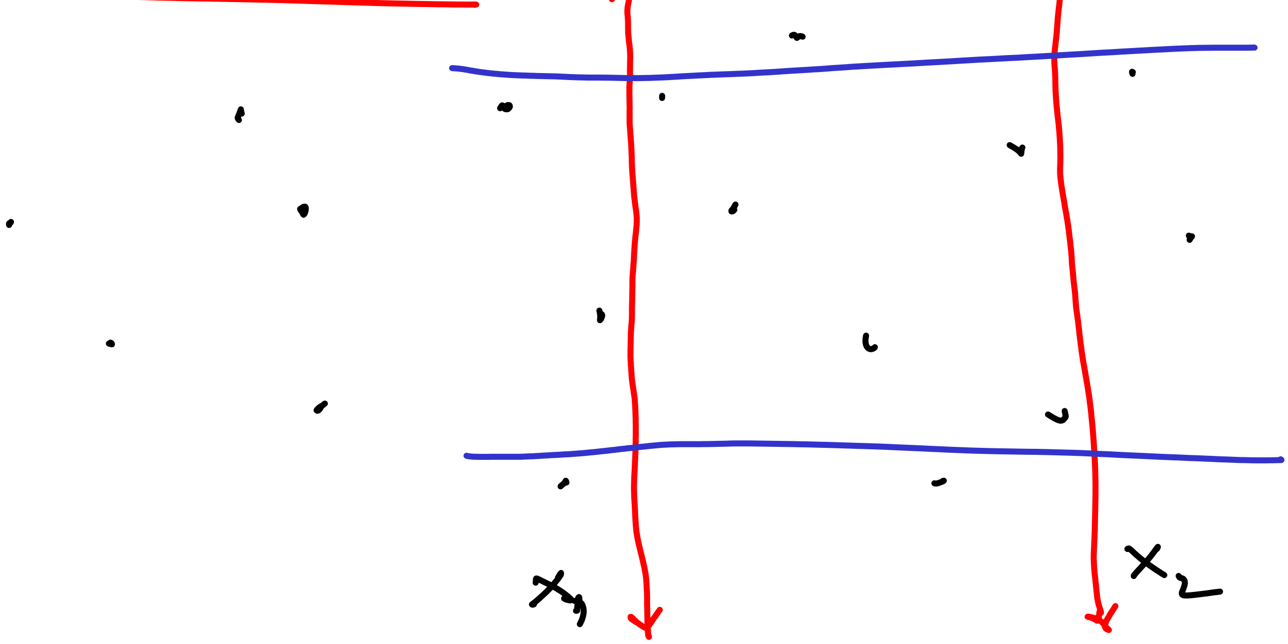
$O(\log n + K)$ query time

$O(n)$ space

$O(n \log n)$ Preprocessing time

(insertion/deletion) Update cost

Two dimensions



For $\binom{n}{2}$ distinct possibilities of $[x_1, x_2]$, we can build separate data structures in the Y direction

Query cost : $O(\log n + K)$

Space : $O(n^2) \cdot n$ $\quad O(n^3)$

unacceptable