Lower bound for comparison-based sorting — $\Theta(n \lg n)$ (worst-case)

Selection Problem

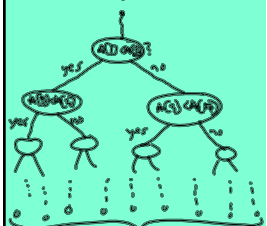
"Comparison-based"

means no assumptions are made about the keys except that they are **comparable**
(given two keys k_1, k_2 we can compare them to see which comes first: "$k_1 < k_2$?" is a comparison.

Minimum possible assumption; most general type of sorting algo.
Lower bound is "information based."

Any comparison-based sorting algorithm can be modeled by a **decision tree**.

Input: $A[1..n]$ of n keys unsorted. Assume that all keys are distinct.
(either $A[i] < A[j]$ or $A[j] < A[i]$ for all $i \neq j$).



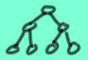
Assuming a correct sorting algorithm, $A[1..n]$ is sorted at each leaf.
Consider two distinct initial arrangements of $A[1..n]$

e.g. $n=3$ $\quad A[1..3] = \langle 1,3,2 \rangle$
vs $\quad A[1..3] = \langle 3,2,1 \rangle$
$\quad A[1..3] = \langle 3,1,2 \rangle$

For all initial arrangements to wind up sorted, the computation must treat each distinctly at some point, leading to distinct computational paths down the tree.

So, the number of leaves in the tree is $\geq$ number of possible initial arrangements
$= n!$

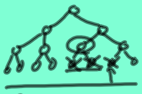
Binary tree of depth k



$k=2$
$4 = 2^2$ leaves

In general, the complete binary tree (all levels full) of depth k has 2^k leaves (exactly)

$\leq 2^k$ leaves in any tree of depth k



Summary
1. Any decision tree for a comparison-based sort has $\geq n!$ leaves
2. Any binary tree of depth k has $\leq 2^k$ leaves

Consider a decision-tree of depth k for a sorting algo on n items.

$\#\text{leaves} \le 2^k$

$\#\text{leaves} \ge n!$

So $n! \le 2^k$

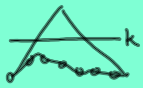
$k \ge \underline{\lg(n!)} \ge ?$

$$\lg(n!) = \lg\left(\prod_{i=1}^{n} i\right)$$
$$= \sum_{i=1}^{n} \lg i$$
$$\ge \sum_{i=\lceil n/2 \rceil}^{n} \lg i$$
$$\ge \sum_{i=\lceil n/2 \rceil}^{n} \lg\left(\frac{n}{2}\right)$$
$$\ge \left(\frac{n}{2}\right) \lg\left(\frac{n}{2}\right)$$
$$= \frac{n}{2}(\lg n - 1)$$
$$= \frac{n \lg n}{2} - \frac{n}{2} = \Omega(n \lg n)$$

So $\underbrace{\text{depth } k}_{} = \underline{\Omega(n \lg n)}$

= worst-case # of comparisons //

∴ time is $\Omega(n \lg n)$



Selection Problem

Given an array $A[1..n]$ & value k $(1 \le k \le n)$,

find the <u>k'th order statistic</u> in $A[1..n]$, i.e., the k'th smallest element in $A[1..n]$.
$= A(k)$ if A were sorted,
(A may not be sorted).

One approach:
```
Build MinHeap(A)         — Θ(n)
Do k times:
    x := ExtractMin(A)  ] lg n
return x
```

Time $\Theta(n + k \lg n)$

worst case: $k = n$. $\Theta(\underline{n \lg n})$

Another approach:
```
sort A[1..n]  —— Θ(n lg n)
return A(k)
```

Randomized Selection runs in $\Theta(n)$ expected time

Deterministic Selection in $\Theta(n)$ time. (optimal)

```
RandomizedSelect(A[p..q], k)
// find k'th smallest element
// in A(p..q)
// Top-level call: RandomizedSelect(A[1..n], k)
  m := RandomizedPartition(A[p..q])
//p ≤ m ≤ q and A(m) contains the
   randomly selected pivot
// A[i] ≤ A[m] if i < m
// A[i] ≥ A[m] if i > m
 if m-p = k-1 then        (m-p = size of left list)
     return A[m]
 else if m-p > k-1 then
     return RandomizedSelect(A[p..m-1], k)     (A[p..m-1] = left list)
 else // m-p < k-1
   return RandomizedSelect
          (A[m+1..q], k-(m-p)-1)
```

Sep 29-1:00 PM

Deterministic Selection in linear time
(worst-case)

Partitions just as with the randomized version,

Median-of-5 method:

Chop $A[1..n]$ into blocks $B_1, B_2, \dots, B_{\frac{n}{5}}$ of 5 contiguous elements each

$B_1 = A[1..5]$
$B_2 = A[6..10]$
$\vdots$

Compute the median x_i for each block B_i

$B[1..\frac{n}{5}] = \langle x_1, x_2, \dots, x_{\frac{n}{5}} \rangle$

Recursively find the median x of $B[1..\frac{n}{5}]$ ($\frac{n}{10}$'th order stat)

Partition using x as the pivot.

Proceed as before: recurse into one sublist or the other.

Let $T(n)$ be the run time

$$T(n) = cn + T\left(\tfrac{n}{5}\right) + T(\quad ?\quad)$$

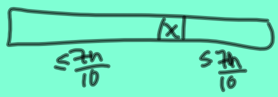
Count # elements $< x$ in $A[1..n]$: $\frac{n}{10}$ elements in B list $< x$

x_i is one of these elements

B_i

So $\frac{n}{10} + \frac{n}{10} + \frac{n}{10} = \frac{3n}{10}$ elements $< x$ at least.

Similarly, at least $\frac{3n}{10}$ elements $> x$.

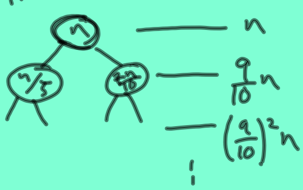


So

$$\boxed{T(n) \le cn + T\left(\tfrac{n}{5}\right) + T\left(\tfrac{7n}{10}\right)}$$

$\therefore T(n) = \Theta(n)$

Tree method:



$$T(n) \le \sum_{i=0}^{\infty} n\left(\frac{9}{10}\right)^i = n \underbrace{\sum_{i=0}^{\infty}\left(\frac{9}{10}\right)^i}_{<\infty}$$

$\therefore T(n) = \Theta(n)$.

Sep 29-1:29 PM