[The?] Lower Bound on Sorting

We're all aware of a number of comparison-based sorting algorithm that are generally referred to as quadratic algorithms. There've also Merge Sort and Quicksort which have a time complexity of $\Theta(n\lg n)$ in the average case.

Below is a recap of these comparison-based sorting algorithms:

Algorithm	Worst-case (# of steps)	Best-Case (# of steps)	Avg.case (# of steps)	Worst-case (# of swaps)	Inplace?
Selection Sort	$\Theta(n^2)$ Why Big-Theta	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n)$	Yes
Insertion Sort	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	Yes
MergeSort	$\Theta(n\lg n)$	$\Theta(n\lg n)$	$\Theta(n\lg n)$	$\Theta(n\lg n)$	No*
Quicksort	$\Theta(n^2)$**	$\Theta(n\lg n)$	$\Theta(n\lg n)$	$\Theta(n^2)$	Yes

* In the classical implementation of the merge procedure

** As we shall see, the worst-case of Quicksort can be easily avoided.

So, let's ask ourselves: Between Selection Sort and Insertion Sort, which algorithm would you choose?

Now, for 'significantly' large values of n (the size of the input), we may want to avoid the hidden costs. For small values of n, the constants hidden under the asymptotic ~~notations~~ notations can play a significant role in determining the performance of an algorithm.

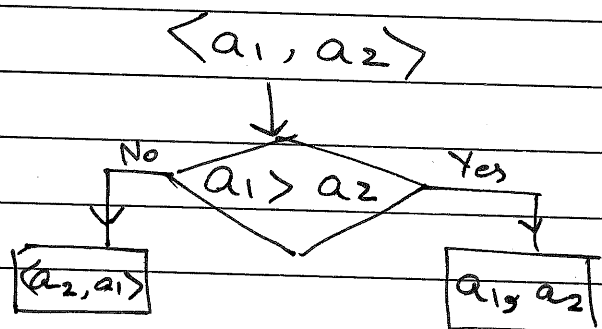
A Lower Bound on Sorting

Comparison-based Sort : All comparison-based sorting algorithms determine the sorting order using some sort key [for integers, it's well-defined, eg.]

Sorting is a Decision Problem

* Given a single element ⟨a_1⟩, it is already sorted, hence, no comparison needed.

* Given two elements ⟨a_1, a_2⟩ determine the sorted order by comparison. We build a decision-tree



Claim: In the worst-case, any (pairwise) comparison-based sorting algorithm (on an input of 'n' elements) [assume, wlog that all elements are pairwise distinct] requires at least $n \lg n$ comparisons

or has a lower bound of $\Omega(n \lg n)$

$\Omega(n \lg n)$ is an existential lowerbound for [pairwise] comparison-based sorting. Can't do better than that!

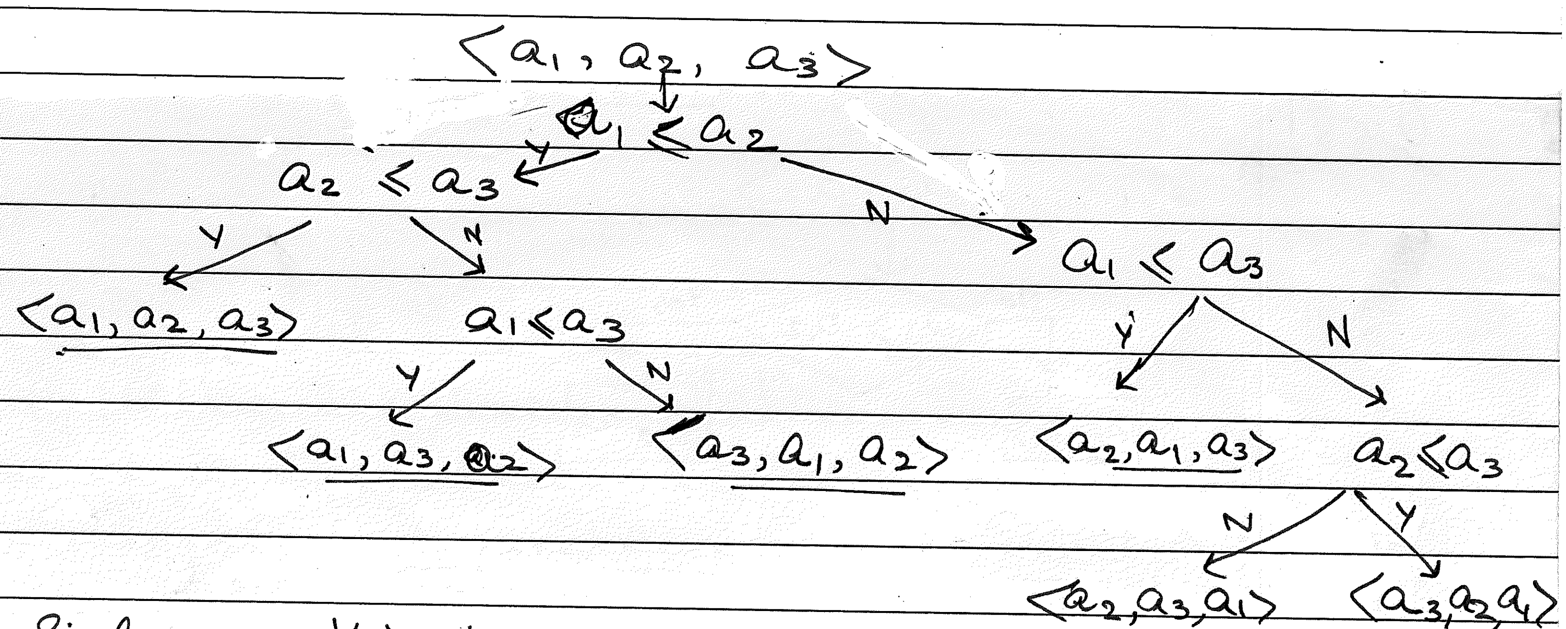
[It's not an universal lowerbound]

Sorting as a Decision Problem

Given a sequence of n pairwise distinct elements
$$\langle a_1, a_2, \ldots, a_n \rangle$$
find a permutation
$$\langle a_1', a_2', \ldots, a_n' \rangle \text{ s.t.}$$

$$\langle a_1' \leqslant a_2' \leqslant a_3' \leqslant \cdots \leqslant a_n' \rangle$$ | $n!$ permutations

Let's consider the case for a small value of $n (= 3)$. There are $3! = 6$ possible arrangements. Let's draw a decision tree



Six leaves. Verify!

The [Sorting] Decision Tree is :

* A binary tree.
* The longest chain of pairwise comparisons is bounded by the height of the decision tree
* A binary tree of height 'h' has at most 2^h leaves.

Date

Now, given 'n' # of elements, there are $n!$ arrangements and hence, $n!$ leaves in the decision tree.

So, we arrive at a decision (a leaf) with a [correct] sorted ordering in at most h (heigh of the tree) time.

$$\therefore \underbrace{n!}_{\text{actual \# of leaves}} \leq \underbrace{2^h}_{\text{the maximum \# of leaves possible in a binary tree}}$$ [by definition]

[What kills an exponent?]

Taking log on both sides,

$$\lg(n!) \leq \lg(2^h)$$

or

$$\lg(n!) \leq h \lg_2 2$$

or

$$\lg(n!) \leq h$$

Now,

here's the claim: $\boxed{\lg n! = \Theta(n \lg n)}$

Hence,

$$n \lg n \leq h \quad \text{or,} \quad \boxed{h = \Omega(n \lg n)}$$

Recap:

* A decision tree for sorting is a binary tree, where a leaf is an ordering of the n elements, and an internal node is just a pairwise comparison $(a_i \leq a_j)$

* Execution time corresponds to a path from the root to a terminal node (leaf). The longest path from the root to a leaf is the height 'h' of the tree.

Date

* A leaf node corresponds to a result of a computation

* A lower bound on sorting is basically a ~~lower~~ lower bound on the height, which is $\Omega(n \lg n)$

ie,

We cannot do better than ~~in~~ a comparison-based sorting algorithm.

Note: $\Omega(n \lg n)$ is an existential lower bound. The Universal lower bound on sorting is $\Omega(n)$, which is the fastest way for insertion sort to terminate, when the input is sorted or nearly sorted.

ie, the best-case of insertion sort is the Universal lower bound for sorting.

Now, let's show $\lg(n!) = \Theta(n \lg n)$

By definition: $f(n) = \Theta(g(n))$ iff $f(n) = O(g(n)) \wedge f(n) = \Omega(g(n))$

I) Let us show $f(n) = O(g(n))$ ie $\lg(n!) = O(n \lg n)$

Now,

$$\text{L.H.S} = \lg(n!) = \lg(\underbrace{(n)(n-1)(n-2)\ldots(2)(1)}_{n\text{-terms}})$$

Recall:

$f(n) = O(g(n))$ iff

$\exists\, c, n_0 > 0$ s.t.

$0 \leq f(n) \leq c.g(n)$,

$\forall n \geq n_0$

Date

$$lg(n!) = lg(\underbrace{n(n-1)(n-2)\cdots 2.1}_{n\text{-terms}})$$

Applying lg we get:

$$lg(n) + lg(n-1) + \cdots + lg(2) + lg(1)$$

or

$$lg(n!) = lg(1) + lg(2) + \cdots + lg(n)$$

$$\leq lg(n) + lg(n) + \cdots + lg(n)$$

[Replacing all terms by n; no change in effect $\leq$]

or

$$lg(n!) \leq n.\, lg n \quad \text{or,} \quad \boxed{lg(n!) = O(n\, lg n)}$$

II) Now, let's show $lg\, n! = \Omega(n\, lg n)$

$$\because lg\, n! = lg((n)(n-1)\cdots(2)(1))$$

$$= \underbrace{lg(1) + lg(2) + \cdots + lg\left(\tfrac{n}{2}\right)}_{n/2 \text{ terms}} + \underbrace{lg\left(\tfrac{n}{2}+1\right) + \cdots + lg(n)}_{n/2 \text{ terms}}$$

[wlog, we're assuming $n = 2^k$, $k > 0$]

Now, discard the first $n/2$ terms from R.H.S., we get

$$lg\, n! \geq \underbrace{lg\left(\tfrac{n}{2}+1\right) + lg\left(\tfrac{n}{2}+2\right) + \cdots + lg(n)}_{n/2 \text{ terms}}$$

or $lg\, n! \geq \frac{n}{2}\, lg\left(\frac{n}{2}+1\right)$

Date

Replacing all remaining $\frac{n}{2}$ terms by $\lg\left(\frac{n}{2}+1\right)$, no effect on $\geq$, the inequality

ie,

$$\lg n! \geq \frac{n}{2}\lg\left(\frac{n}{2}+1\right)$$

or

$$\lg n! \geq \frac{n}{2}\lg\frac{n}{2} + \frac{n}{2}$$

$$\geq \frac{1}{2}\left[n \lg\frac{n}{2}\right] + \frac{n}{2}$$

$$\geq \frac{1}{2}\left[n\lg n - \lg 2\right] + \frac{n}{2}$$

$$\geq \frac{1}{2}\left[n\lg n - 1\right] + \frac{n}{2}$$

or

$$\lg n! \geq \frac{1}{2} n\lg n + \frac{n}{2} - O(1)$$

or

$$\boxed{\lg n! = \Omega(n \lg n)}$$

$f(n) = \Omega(g(n))$ iff
$\exists c, n_0 > 0$:
$f(n) \geq c.\,g(n),\ \forall n \geq n_0$

Hence,

$$\boxed{\lg n! = \Theta(n \lg n)}$$

This is the result we used to find the lower bound on comparison-based sorting.

You can also show the same result using Stirling's approximation

[Try it as HW]

Stirling's Approximation: $\left[\ln n! = n\ln n - n + O(\ln n)\right]$

or $\lg n! = n\lg n - n\lg_2 e + O(\lg n)$