Lower Bound for Sorting (with comparisons)

Cormen Sect 8.1

Sorting algorithms we have seen so far are based on comparisons $T(n) \simeq$ # of comparisons

$a_1 \; a_2 \; a_3 \; a_4 \; \ldots \; a_n$ to sort

Question is $a_i \overset{?}{\leq} a_j$

Comparison-based sorting algorithm

Pros:
- change the resulting ordering just by implmenting a different comparator
- work for any kind of object where a cmp function can be defined

Cons
- design a faster algorithms without limiting ourself to comparisons

Prove that $\Omega(n \log n)$ time is the best we can hope for with comparison-based sorting algos

$A = a_1 \ a_2 \ a_3 \ \ldots \ a_n$

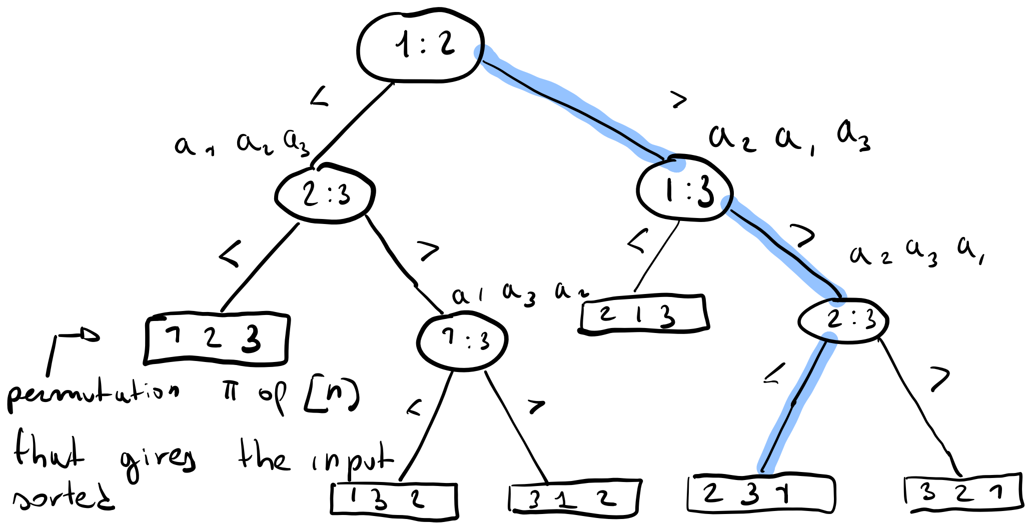
Assume wlog that $\forall i, j \quad a_i \neq a_j$

Decision-tree model

A decision tree is a binary tree that repprensents comparisons performed by a fixed comparison-based algorithm on any input of a fixed size n

Decision-tree for insertion sort on $n = 3$

$a_1 \ a_2 \ a_3$



- The execution of the algorithm on a input of size n corresponds to a root-to-a-leaf path eg

A	7	4	5
	a_1	a_2	a_3
π	2	3	1
A	4	5	7

- The running time of IS in the worst case is Ω (height of tree)
- Any correct sorting algorithm must be able to produce any permutation

  why? if not, it is wrong on some input

  if $\pi = 1\ 3\ 2$ is not a leaf
  there exists a input say 5 9 7
  for which the algorithm is not correct!!

  $\Rightarrow$

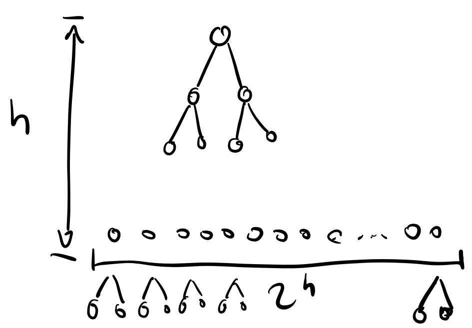
  there exist at least $n!$ leaves ($\Omega(n!)$)

Any sorting algorithm based on comparisons need $\Omega(n \log n)$ time (or comparisons)

- worst case time $= \Omega(\text{height of the tree})$
  $= \Omega(h)$
- $\Omega(n!)$ leaves
- Binary tree of height h has no more than 2^h leaves

  proof

  - base case $h = 0$ ○ root # leaves $= 2^0 = 1$
  - inductive step: true for any h,
    we want to prove this for $h+1$



leaves is $2^h \cdot 2 = 2^{h+1}$ $\boxed{\square}$

$2^h \geq n! \Rightarrow h \geq \log_2(n!)$ Stirling approx of $n!$

$\Rightarrow h \underset{=}{\cancel{\geq}} \Omega(n \log n)$