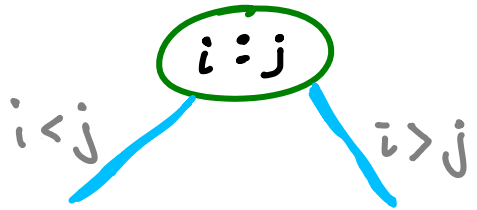
Algorithms represented as Decision Trees



internal nodes represent comparison of two elements i,j

branches represent outcome of comparison

↳ Left : i<j
Right i>j

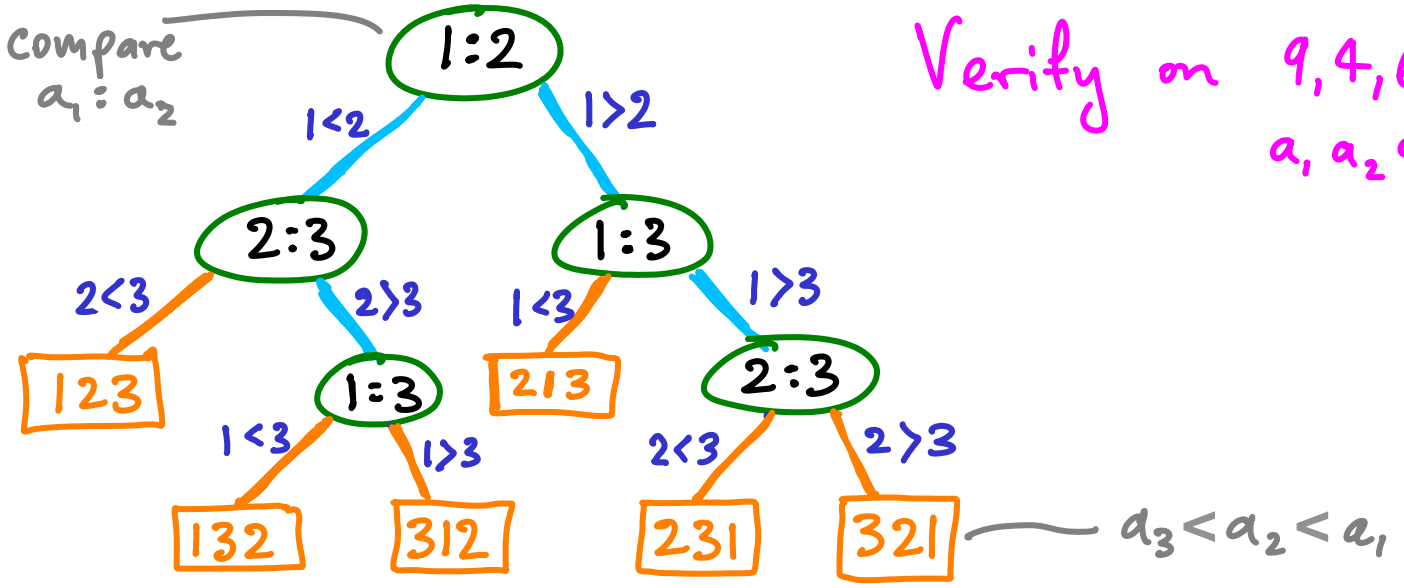
Algorithms represented as Decision Trees

internal nodes represent comparison of two elements i,j

branches represent outcome of comparison

↳ Left : $i<j$
Right $i>j$

Example: sort $a_1 a_2 a_3$



Verify on 9,4,6
$a_1 a_2 a_3$

Each leaf is a possible output

Each root→leaf path represents an execution of algo.

[Any decision-based algorithm can be encoded as a decicion tree.]

If you are designing a decision tree

it's up to you to avoid comparing the same elements many times.

The worst-case run-time is precisely the longest root-leaf path.

you shouldn't compare $a_i : a_j$ twice on one path.

↳ so max path length $= \binom{n}{2}$

for sorting

~~for~~ a good algo

If you are designing a decision tree
it's up to you to avoid comparing the same elements many times.

The worst-case run-time is precisely the longest root-leaf path.

you shouldn't compare $a_i : a_j$ twice on one path.
↳ so max path length = $\binom{n}{2}$

Why not write all algorithms this way? (so much prettier than pseudocode)
↳ It's huge and repetitive.
It really lists every possible execution of algo.
You actually might need a different tree for each n.

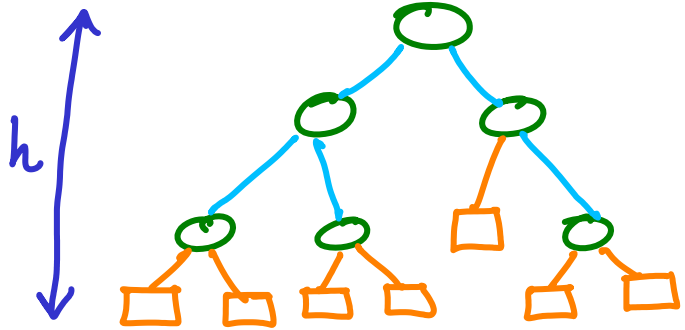
- What is the shortest possible tree for comparison-sort?

A correct decision tree for sorting must have every possible output represented at a leaf node.

↳ #leaves $\geq$?

A correct decision tree for sorting must have every permutation of the input represented at a leaf node.

↳ #leaves $\geq n!$



height of tree = worst case time = h ⇝ #leaves $\leq 2^h$

[binary tree; every node has 2 children]

so, $n! \leq$ #leaves $\leq 2^h \Rightarrow \log n! \leq \log 2^h \Rightarrow \underline{h \geq \log n!}$

Stirling's formula: $n! \geq \left(\frac{n}{e}\right)^n$

$h \geq \log\left(\frac{n}{e}\right)^n = n \cdot \log \frac{n}{e}$

$= n\log n - n\log e$

$= n\log n - \Theta(n)$

$h = \Omega(n \log n)$

extra analysis of $\log n!$ follows

$\log(n!) = O(n \log n)$ $\quad c \quad n_0$

$\log(n!) \leq \log(n^n) = n \log n \Rightarrow \log(n!) \leq 1 \cdot n \log n$ for $n \geq 1$

$\log(n!) = \Omega(n \log n)$

$\log(n!) = \log(n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdots \cdots 3 \cdot 2 \cdot 1)$

$= \log(\underbrace{n \cdot 1} \cdot \underbrace{(n-1) \cdot 2} \cdot \underbrace{(n-2) \cdot 3} \cdot \underbrace{(n-3) \cdot 4} \cdots \cdots \underbrace{(n-\frac{n}{2}) \cdot (n-\frac{n}{2})})$

$\geq \log(n \cdot n \cdot n \cdot n \cdot \cdots \cdot n)$

$= \log(n^{n/2})$ (assume n: even, otherwise $\frac{n}{\lfloor 2 \rfloor}$) $\Rightarrow \log(n!) \geq \frac{n}{2} \log n$

---

so $\frac{1}{2} n \log n \leq \log(n!) \leq n \log n$

---

in fact, Stirling's approximation: $\ln(n!) = n \cdot \ln(n) - n + O(\ln(n))$