make history.

THE UNIVERSITY of ADELAIDE

Lower bound of time computational complexity for comparison-based sorting algorithms

Dr. Anna Kalenkova

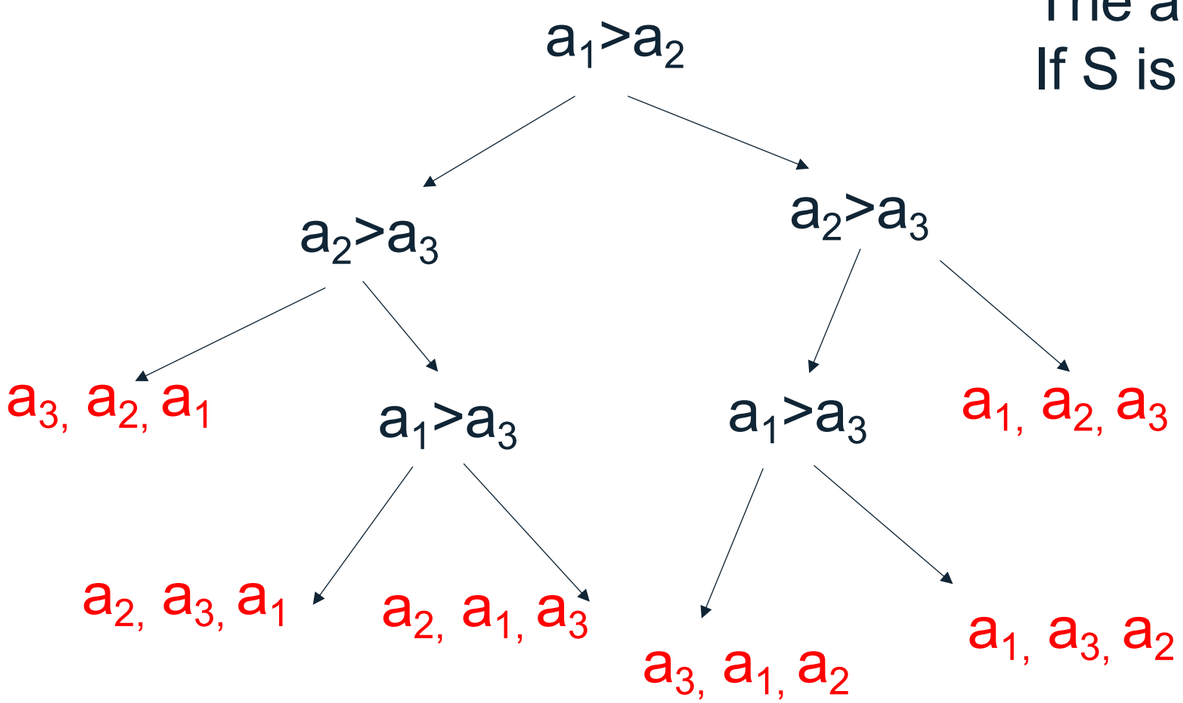
Comparison-based sorting algorithms

- Suppose we need to sort an array: $a_1, a_2, \ldots, a_n$.

- If all the elements are distinct, there n! possible results of sorting (all possible permutations), but only one is correct!

- Each comparison-based sorting algorithm builds a decision tree. Let's consider such a tree.

Comparison-based sorting algorithms

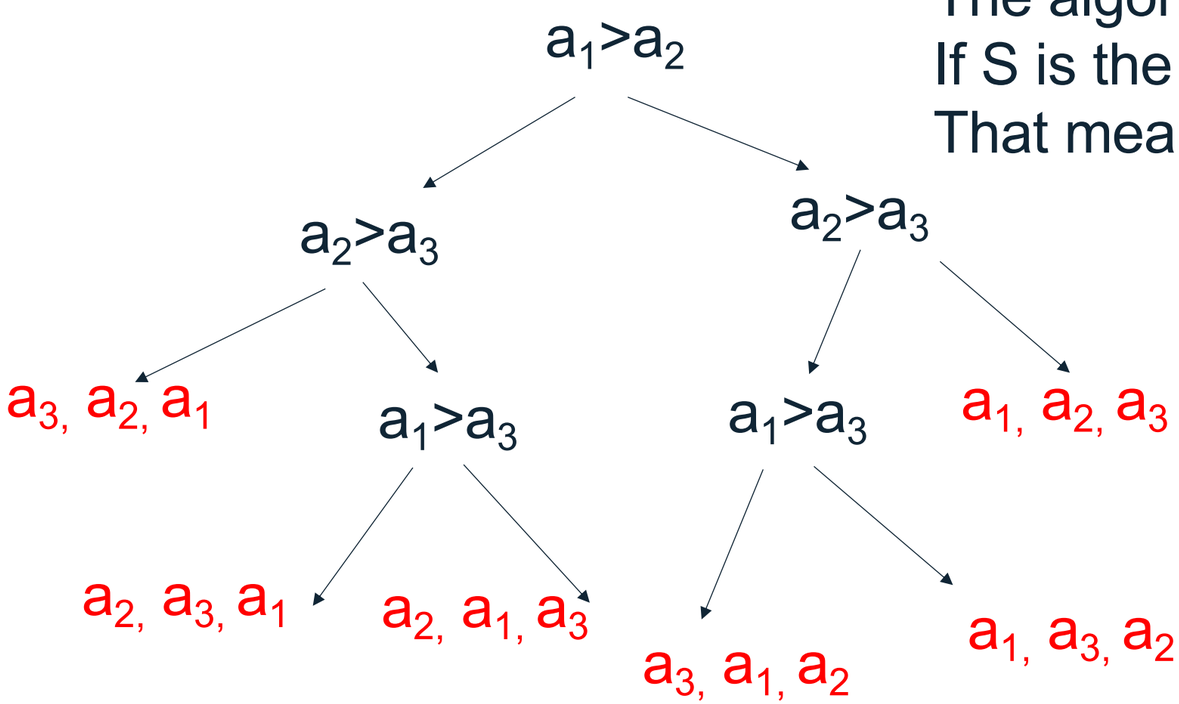


The algorithm selects a branch each time .
If S is the initial set of problems, we select S' > S/2.

The set of solutions is split each time.

THE UNIVERSITY of ADELAIDE

Comparison-based sorting algorithms

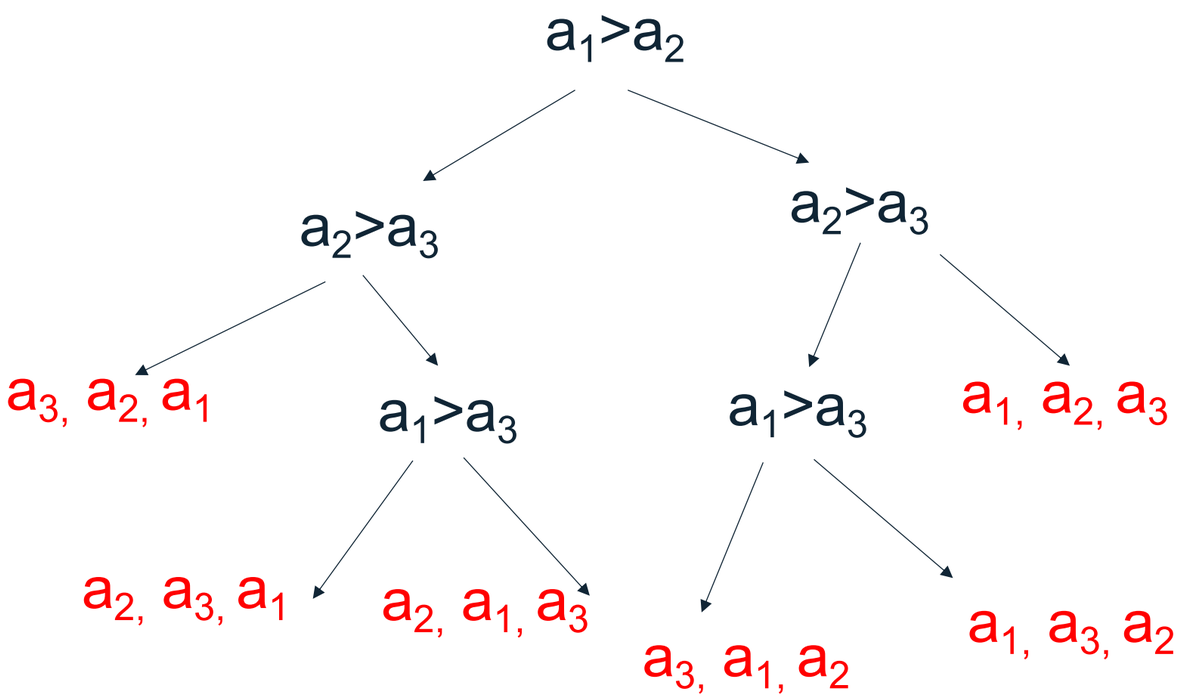


The algorithm selects a branch each time.
If S is the initial set of problems, we select S' > S/2.
That means we halving the set of solutions is the best option.

The number of leaves (possible solutions) is **n!**. Let the height of this tree (the number of comparisons) is **$\log_2(n!)$**, because we halve the set of solutions each time.

THE UNIVERSITY of ADELAIDE

Comparison-based sorting algorithms



$\log_2(n!) = log_2(n) + log_2(n-1) + \cdots + log_2(2) \geq log_2(n) + log_2(n-1) + \ \ldots + \log_2\left(\frac{n}{2}\right) \geq$

$nlog_2\left(\frac{n}{2}\right) = n(\log_2(n) - 1)$. Hence, $\log_2(n!) = \Omega(n\log(n))$.

THE UNIVERSITY of ADELAIDE