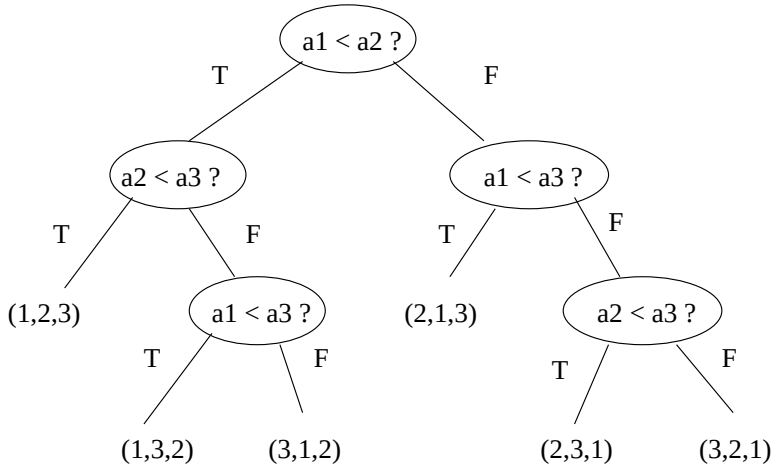
Can we sort in better than $n \lg n$?

Any comparison-based sorting program can be thought of as defining a decision tree of possible executions.

Running the same program twice on the same permutation causes it to do exactly the same thing, but running it on different permutations of the same data causes a different sequence of comparisons to be made on each.



Claim: the height of this decision tree is the worst-case complexity of sorting.

Once you believe this, a lower bound on the time complexity of sorting follows easily.

Since any two different permutations of n elements requires a different sequence of steps to sort, there must be at least $n!$ different paths from the root to leaves in the decision tree, ie. at least $n!$ different leaves in the tree.

Since only binary comparisons (less than or greater than) are used, the decision tree is a binary tree.

Since a binary tree of height h has at most 2^h leaves, we know $n! \leq 2^h$, or $h \geq \lg(n!)$.

By inspection $n! > (n/2)^{n/2}$, since the last $n/2$ terms of the product are each greater than $n/2$. By Sterling's approximation, a better bound is $n! > (n/e)^n$ where $e = 2.718$.

$$h \geq \lg(n/e)^n = n \lg n - n \lg e = \Omega(n \lg n)$$