30.4 Summary

Insertion Sort Sweet Spots. We concluded our discussion of insertion sort by observing that insertion sort is very fast for arrays that are almost sorted, i.e. that have Θ(N) inversions. It is also fastest for small N (roughly N≤15).

Partitioning. Partioning an array on a pivot means to rearrange the array such that all items to the left of the pivot are ≤ the pivot, and all items to the right are ≥ the pivot. Naturally, the pivot can move during this process.

Partitioning Strategies. There are many particular strategies for partitioning. You are not expected to know any particular startegy.

Quicksort. Partition on some pivot. Quicksort to the left of the pivot. Quicksort to the right.

Quicksort Runtime. Understand how to show that in the best case, Quicksort has runtime Θ(NlogN), and in the worse case has runtime $\Theta(N^2)$.

Pivot Selection. Choice of partitioning strategy and pivot have profound impacts on runtime. Two pivot selection strategies that we discussed: Use leftmost item and pick a random pivot. Understand how using leftmost item can lead to bad performance on real data.

Randomization. Accept (without proof) that Quicksort has on average Θ(NlogN) runtime. Picking a random pivot or shuffling an array before sorting (using an appropriate partitioning strategy) ensures that we're in the average case.

Quicksort properties. For most real world situations, quicksort is the fastest sort.

Previous
30.3 Quicksort Performance Caveats

Next
30.5 Exercises

Last updated 1 year ago