coursera

Overview

Our lectures this week are based on two classic algorithms that were invented over 50 years ago, but are still important and relevant today, as implementations of one or both of them are found in virtually every software system and research on new variants of these classic methods is ongoing. Our treatment ranges from the mathematical models that explain why these methods are efficient to the details of adapting them to real-world applications on modern systems.

Lecture: Mergesort. We study the *mergesort* algorithm and show that it guarantees to sort any array of n items with at most $n \lg n$ compares. We also consider a nonrecursive, bottom-up version. We prove that any compare-based sorting algorithm must make at least $\sim n \lg n$ compares in the worst case. We discuss using different orderings for the objects that we are sorting and the related concept of stability.

Lecture: Quicksort. We introduce and implement the *randomized quicksort* algorithm and analyze its performance. We also consider randomized quickselect, a quicksort variant which finds the kth smallest item in linear time. Finally, consider 3-way quicksort, a variant of quicksort that works especially well in the presence of duplicate keys.

Exercises. (*Sorry, we are still waiting for Coursera to migrate the exercises from the old platform.*) Drill exercises on the lecture material.

Programming Assignment: Collinear Points. Your programming assignment is a typical example of a problem that could not be solved without a fast sorting algorithm, properly applied. It is a classic problem in computational geometry: Given a set of points in the plane, design an algorithm to find all line segments that contain 4 or more points.

Job Interview Questions. Algorithmic interview questions based on the lecture material.

Suggested Readings. Section 2.2 and 2.3 in *Algorithms, 4th edition.*

✓ Complete