Overview

Can we guarantee fast search, insert, delete, min, max, floor, ceiling, rank, and select in a symbol table with Comparable keys? This week, you will learn that the answer to this question is a resounding Yes! and that a relatively compact implementation discovered just five years ago can do the job. Then, we consider applications and generalizations of binary search trees to problems in computational geometry.

Lecture: Balanced Search Trees. In this lecture, our goal is to develop a symbol table with guaranteed logarithmic performance for search and insert (and many other operations). We begin with 2-3 trees, which are easy to analyze but hard to implement. Next, we consider red-black binary search trees, which we view as a novel way to implement 2-3 trees as binary search trees. Finally, we introduce B-trees, a generalization of 2-3 trees that are widely used to implement file systems.

Lecture: Geometric Applications of BSTs. We start with 1d and 2d range searching, where the goal is to find all points in a given 1d or 2d interval. To accomplish this, we consider kd-trees, a natural generalization of BSTs when the keys are points in the plane (or higher dimensions). We also consider intersection problems, where the goal is to find all intersections among a set of line segments or rectangles.

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

Programming Assignment: Kd-Trees. Your programming assignment is to implement kd-trees, which can form the basis for fast search/insert in geometric applications and in multidimensional databases.

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

Suggested Readings. Section 3.3 in Algorithms, 4th edition.

✓ Complete