Overview

This week, we consider two different search problems involving strings. In the first lecture, we build upon the ideas we considered for string sorts to develop string search methods that are even faster than hashing and even more flexible than binary search trees. In the second lecture we consider classic algorithms for the substring search problem, where the goal is to find a given substring in a large text.

Lecture 7: Tries. In this lecture we consider specialized algorithms for symbol tables with string keys. Our goal is a data structure that is as fast as hashing and even more flexible than binary search trees. We begin with multiway tries; next we consider ternary search tries. Finally, we consider character-based operations, including prefix match and longest prefix, and related applications.

Lecture 8: Substring Search. In this lecture we consider algorithms for searching for a substring in a piece of text. We begin with a brute-force algorithm, whose running time is quadratic in the worst case. Next, we consider the ingenious Knuth–Morris–Pratt algorithm whose running time is guaranteed to be linear in the worst case. Then, we introduce the Boyer–Moore algorithm, whose running time is sublinear on typical inputs. Finally, we consider the Rabin–Karp fingerprint algorithm, which uses hashing in a clever way to solve the substring search and related problems.

To Do:

- **Programming Assignment: Boggle.** Write a program to play the word-game Boggle.
- **Job Interview Questions.** Algorithmic interview questions based on the lecture material.
- **Suggested Readings.** Section 5.2 and 5.3 in *Algorithms, 4th edition*.
-

✓ Complete Go to next item