This lecture is about efficient data structures for searching in static strings. Think of the strings we're searching in as (large) files, or entire disks. First we'll see how to store a bunch of strings in linear space subject to searching for a string of length P in $O(P)$ time, and answering predecessor queries for a string of length P in $O(P + \lg \Sigma)$ time, where Σ is the size of the alphabet. Then we'll see how to store one string (or more) in linear space subject to determining whether it has a given substring of length P in $O(P)$ time, counting the number of such substring occurrences in $O(P)$ time, and listing k occurrences in $O(P + k)$ time. Finally, we'll see how to build this data structure in linear time. We'll make use of several tools developed in previous lectures: weight-balanced binary search trees (similar to L3), leaf trimming (L15), Cartesian trees (similar to L15), range minimum queries (L15), least common ancestor (L15), and level ancestor (L15).

6.851 Advanced Data Structures

Spring 2012