15-451/651 Algorithm Design & Analysis, Spring 2024

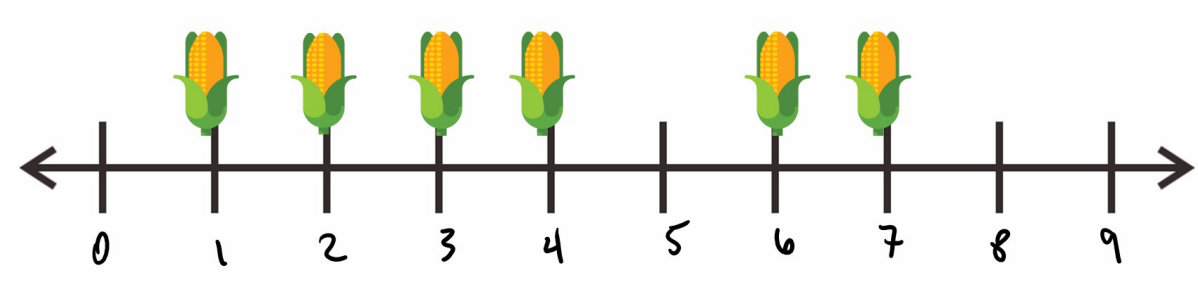
Recitation #4

Objectives

- Practice applying the *union-find* data structure as an algorithm ingredient
- Familiarize with the SegTree data structure and using it to speed up algorithms
- Understand how to build SegTrees with custom associative operators

Recitation Problems

1. **(Corn Intervals)** You are a serf and you have a singular strip of land shaped like a number line. Your capitalist overlords have given you a list of n integer locations to plant cornstalks at in the order they are given. However, you are also lazy and only want to harvest good intervals of corn. A good interval $[i, j]$ is defined as a interval containing at least k consecutive cornstalks such that there are no cornstalks at $i-1$ and $j+1$.



In this picture, if $k = 3$, the interval $[1, 4]$ is the only good interval.

Write an algorithm keeping track of where you have planted corn such that at any time, you can determine the number of good intervals in $O(1)$. You're allowed $O(\log n)$ time to process each time you plant a cornstalk.

2. **(Abby's Favorite Problem)** Suppose we start with some array of integers A. Given a sequence of query intervals in the form $[l_1, r_1), [l_2, r_2), ..., [l_m, r_m)$, return the maximum element and how many times it appears in $A[l_i], ..., A[r_i - 1]$ in $O(\log n)$ time for each query $[l_i, r_i)$

3. **(Crossing intervals)** Suppose we have a list of n intervals $I_i = [a_i, b_i]$ where $0 \leq a_i, b_i < 2n$, such that the endpoints of all of the intervals are distinct (no two intervals ever share an endpoint at either end). A pair of intervals I_i and I_j are *crossing* if they overlap but one does not strictly contain the other. We want to devise an algorithm to count the number of pairs of crossing intervals.

(a) Give a simple $O(n^2)$ algorithm for the problem

(b) Come up with a more efficient $O(n \log n)$ algorithm by making use of a SegTree

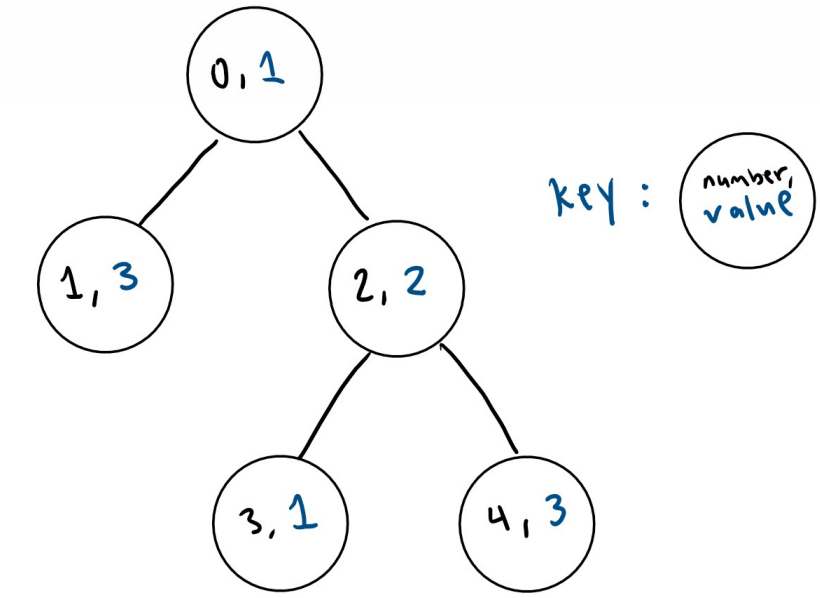
4. **(Optional: Tree Path)** There is a tree consisting of n nodes numbered from 0 to $n-1$ and exactly $n-1$ edges. Each node also has an integer value.

   You are also given a set E of undirected edges where $(i, j) \in E$ means that there is an undirected edge connecting nodes i and j.

   A good path is a simple path that satisfies the following conditions:

   The starting node and the ending node are distinct nodes with the same value. All nodes between the starting node and the ending node have values less than or equal to the starting node (and the ending node).

   Note that a path and its reverse are counted as the same path.



   In the picture above, there is 1 good path: 1–>0–>2–>4.

   (a) Let's assume you know the maximum value in the tree, v_{max}. Find the number of good paths where the starting/ending node have value v_{max}.

   (b) Now let's broaden the problem- find the total number of good paths in the tree in $O(n \log n)$.