Change log:

- Problem 4 (version 2): added that the numbers are distinct.

This problem set has a total of 10 written problems on 2 pages. While some of these were adapted from coding problems, I’m taking written solutions only due to the proximity to end-of-term.

Written solutions should be submitted to Canvas by 2pm Monday Apr 26, 2021.

The first six problems are all variants of divide-and-conquer. They should be approachable from first principles.


2. Show that given a (black-box) function that takes length $n$ binary vectors to a single binary output:

   $$ f : \{0,1\}^n \to \{0,1\}^n $$

   such that $f(0,0,0,\ldots,0) = 0$ and $f(1,1,1,\ldots,1) = 1$, we can find some input $x$ and some index $i$ with $x_i = 0$ such that the value of $f$ changes when $x_i$ is toggled from 0 to 1, by querying the value of the function $O(\log n)$ times.

3. An $n$-by-$n$ 0-1 grid is called doubly monotone if all the 0s come before all the 1s in each row, and also each column. Show that we can determine the number of 1s in some hidden doubly monotone grid by querying the values of $O(n)$ entries.

4. In a 2D array with distinct numbers, a local maxima is a cell whose value is larger than all its neighbors. Show that we can find a local maxima of a hidden $n$-by-$n$ array by querying the values of $O(n)$ entries.

5. Consider the following query process for finding a hidden vertex $\hat{p}$ on a tree: each time one can query a vertex $p$, and if $p \neq \hat{p}$, the edge from $p$ that leads to $\hat{p}$ is returned. Show that $\hat{p}$ can always be found after $O(\log n)$ queries.

6. Give a polynomial time algorithm for querying a given tree optimally. That is, it guarantees that the worst case (over possibilities of $\hat{p}$), the number of queries needed to find $\hat{p}$ is minimized.
NOTE: this type of search/pursuit game on tree has many extensions, e.g. [https://dmoj.ca/problem/noi07p6](https://dmoj.ca/problem/noi07p6).

The next four problems require using fast matrix multiplication, but only as a black box. Throughout them, let the matrix multiplication exponent be $\omega$.

7. Show that in a directed, unit weighted, graph on $n$ vertices, the shortest cycle can be found in $O(n^\omega \log n)$ time.

8. Define the gather time of a directed graph with at least one edge leaving each vertex as:

   (a) Initially there is someone at each vertex.
   
   (b) At each time step, everyone needs to go to a neighbor via an outgoing edge.
   
   (c) The gather time is the earliest time at which everyone can be at some vertex.

   Show that the gather time of a directed graph can be computed in $O(n^\omega \log n)$ time.

9. Consider an array $A$ of length $n$ with integers in the range $1 \ldots k$. Let $\text{NumIncreasing}[i][x]$ be the number of increasing subsequences of $A[1 \ldots i]$ whose last element has value $x$ (the empty sequence gets 0), and let $\overrightarrow{\text{NumIncreasing}}[i]$ be the vector form of this over $0 \leq x \leq k$, with $\overrightarrow{\text{NumIncreasing}}[0]$ defined as the indicator vector with 1 on entry 0, 0 everywhere else. Show that for each index $i > 0$, there is some matrix $M(i)$ (dependent on $A[i]$) such that

   $\overrightarrow{\text{NumIncreasing}}[i] = M(i) \overrightarrow{\text{NumIncreasing}}[i-1],$

   and furthermore, for any $p > k$, $M(i)$ is invertible modulo $p$.

10. Show we can pre-process $A[1 \ldots n]$ in $O(nk^2)$ time such that for any query $[l \ldots r]$, we can compute the number of increasing subsequences of $A[l \ldots r]$ in $O(k^\omega)$ time.

    **HINT:** the query should be a single matrix multiplication.