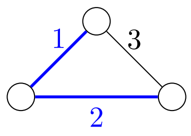
CSE421: Design and Analysis of Algorithms April 23, 2020

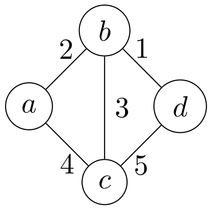
Homework 4

Shayan Oveis Gharan Due: April 30th, 2020 at 23:59 PM

P1) Prove or disprove the following claim: Suppose we are given a graph G where the cost of edge e is c_e where all c_e's are positive. For simplicity, assume that all c_e's are distinct. Suppose T is the minimum spanning tree which is the output of the Kruskal's algorithm. Now suppose we update the cost of every edge e to c_e^2. Then, T remains the minimum spanning tree of G. For example, in the following graph the tree shown in blue is the output of the Kruskal's algorithm. In this case if we update the costs to $1, 4, 9$ respectively the blue tree remains a minimum spanning tree.



P2) Given a connected undirected weighted graph $G = (V, E)$ with positive weights on the edges, i.e., $c_e > 0$ for all e. Design a polynomial time algorithm to find the *minimum cost* set of edges $F \subseteq E$ such that if we delete all edges of F the remaining graph has no cycles. For simplicity assume that for any two edges $e, f \in E$, $c_e \neq c_f$. For example in the following example the optimum set F is $F = \{(a, b), (b, d)\}$.



P3) Suppose you are choosing between the following three algorithms:

a) Algorithm A solves the problem by dividing it into seven subproblems of half the size, recursively solves each subproblem, and then combines the solution in linear time.

b) Algorithm C solves the problem by dividing it into twenty seventh subproblems of one ninth the size, recursively solves each subproblem, and then combines the solutions in cubic time.

c) Algorithm B solves problems of size n by recursively solving two subproblems of size $n - 4$, and then combines the solution in constant time.

What are the running times of each of these algorithms? To receive full credit, it is enough to write down the running time.

P4) Given an array $a_1, \ldots, a_n$ of n distinct integers, design an $O(\log n)$ time algorithm that finds a_i such that $a_i > a_{i-1}$ and $a_i > a_{i+1}$. Note that you may output a_1 if $a_1 > a_2$ and you may output a_n if $a_n > a_{n-1}$. For example, given the sequence $3, 4, 6, 5, 1, 2$ you can output 6 or 2. If no such a_i exists, output "Impossible".

P5) **Extra Credit** The spanning tree game is a 2-player game. Each player in turn selects an edge. Player 1 starts by deleting an edge, and then player 2 fixes an edge (which has not been deleted yet); an edge fixed cannot be deleted later on by the other player. Player 2 wins if he succeeds in constructing a spanning tree of the graph; otherwise, player 1 wins.

The question is which graphs admit a winning strategy for player 1 (no matter what the other player does), and which admit a winning strategy for player 2.

Show that player 1 has a winning strategy if and only if G does not have two edge-disjoint spanning trees. Otherwise, player 2 has a winning strategy.