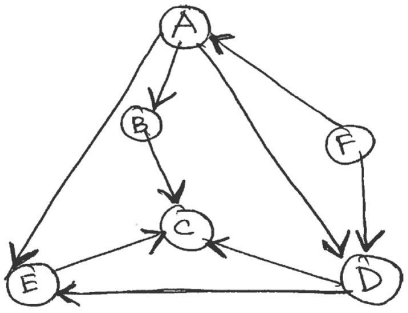
CS 721: Advanced Algorithms & Analysis

Homework 2, Fall 2019, Total 115 points

Assigned on: Thursday, 08/19/2019
Due on: Tuesday, 09/01/2019

1. (15 points) Suppose all golf players are either professionals or amateurs and between any pair of golf players there may or may not be a rivalry. Suppose we are given n golf players and a list of r pairs of golf players for which there are rivalries (note that, you can represent this information as an undirected graph). Give an $O(n + r)$-time algorithm that determines whether it is possible to designate some of the golf players as professionals and and the remainders as amateurs such that each rivalry is between a professional and an amateur. If it is possible to perform such a designation, your algorithm should produce it. You do not need to provide pseudo-code. Explain in details how your algorithm will work.
   Hint: You can perform BFS to visit all vertices and compute distances. What would odd and even distances mean?

2. Consider the following directed acyclic graph.



   (a) (10 points) Compute pre and post number for each node of the above graph starting from node A and draw graph after it is topologically sorted (linearized).

   (b) (10 points) Give a linear-time algorithm that takes as input a directed acyclic graph $G = (V, E)$ and two vertices s and t and returns the number of simple paths (i.e., does not contain cycles) from s to t in G. For example, in the above graph there are four simple paths from vertex A to C: $A \to B \to C, A \to E \to C, A \to D \to C$ and $A \to D \to E \to C$. Your algorithm needs only to count the number of paths not list them.
   Hint: First linearize the graph and locate nodes s and t. Which nodes will appear in simple paths from s to t? You need to consider only those nodes that may appear in simple paths between s and t (including s and t) in linearized order, maintain an array for whose indices are linearized ordering of the nodes and update the array elements as you process each node in the linearized order to get your final answer. What should this array contain? How will you update this array?

3. We have seen that algorithm for finding strongly connected components of a directed graph $G = (V, E)$ works as follows. In the first step, compute DFS on the reverse graph G^R and compute post numbers, Then run the undirected connected component algorithm on G, and during DFS, process the vertices in decreasing order of their post number from step 1. Now professor Smart Joe claims that the algorithm for strongly connected component would be

simpler if it runs the undirected connected component algorithm on G^R (instead of G), but during DFS, process the vertices in increasing order of their post number from step 1.

(a) (10 points) Explain when the algorithm proposed by prof. Smart Joe might produce incorrect answer.

(b) (10 points) With an example, (along with post numbers) show that professor Smart Joe is wrong.

4. We are given a directed graph $G = (V, E)$ on which each edge $(u, v) \in E$ has an associated value $r(u, v)$ which is a real number in the range $0 \leq r(u, v) \leq 1$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u, v)$ as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent.

   (a) (10 points) Give an efficient algorithm to find the most reliable path between two given vertices of this graph.
   Hint: Since probabilities are independent, this means we want find a path for which product of the reliabilities will be maximum.

   (b) (10 points) If the directed graph does not contain a cycle, can you give a better algorithm? Explain how your algorithm will work. What is the running time of your algorithm?

5. (10 points) Let $G = (V, E)$ be an undirected connected graph, where all edge weights are positive and equal. Describe an algorithm (no need to provide pseudo-code) that finds MST of G and is asymptotically more efficient than Prim's and Kruskal's algorithm. What is the running time of your algorithm?

6. (15 points) Suppose all edge weights in a graph are integers in the range 1 to $|V|$. How fast can you make Kruskal's algorithm run? What if the edge weights are integers in the range 1 to W for some constant W?
   Hint: You can use the fact that counting sort can sort n integers in the range 0 to k in $\Theta(n + k)$ time.

7. (15 points) Let T be an MST of a graph $G = (V, E)$. Suppose we decrease the weight of one of the edges of T. Show that T (with decreased edge weight) is still an MST for G. More formally, let T be a minimum spanning tree for G with edge weights given by a weight function w. Choose one edge $(x, y) \in T$ and a positive number k and define a new weight function w' by

$$w'(u, v) = \begin{cases} w(u, v), & \text{if } (u, v) \neq (x, y) \\ w(x, y) - k, & \text{if}(u, v) = (x, y) \end{cases}$$

   Show that T is also an MST for G with edge weights given by w'.
   Hint: First, find out how $w(T)$ and $w'(T)$ are related (their values differs in a single edge). In particular, which one is larger among these two? Next, consider any other spanning tree T' (different from T). You need to show that $w'(T) < w'(T')$, that is even with the new edge weights w', T is MST and T' is not. To show this, you need to consider two cases, either the edge $(x, y) \in T'$, that is the edge (x, y) whose weight differs under new edge weight w' is a member of T or $(x, y) \notin T'$. Show that in both these case, $w'(T) \leq w'(T')$.

Submision:

- All texts and diagrams must be electronically produced.
- Your name and page number should appear on each page.
- Entire assignment should be a single PDF file.

- the PDF file should be named in the format HW02_Lastname_Firstname.pdf, for example HW02_Sinha_Kaushik.pdf.
- Submit the pdf file on blackboard.
- Each homework assignment is due at midnight on due date.