ADA Homework Assignment 3

Deadline : April 26 (Monday) 11.59 pm.

The theory assignment has to be done in a team of at most two members, as already selected by you. The solutions are to be typed either as a word document or latex-ed and uploaded as pdf on GC. We shall strictly not accept solutions written in any other form. Remember that both team members need to upload the HW solution on GC.

Collaboration across teams or seeking help from any sources other than the lectures, notes and texts mentioned on the homepage will be considered an act of plagiarism.

Some questions below are from the text by Jeff Erickson. You can find an online copy here.

Problem 1 (10 points) Jatin plans to expand his DJ operation and go on a tour. He has to travel to a music festival and wants to minimize the total distance of travel. However, on every road, his fans gather and ask him for autographs. But he is too shy to sign too many autographs.

More formally, the road network is modeled by diected graph $G = (V, E)$ and Jatin's journey starts from vertex s and ends at vertex t. For each road e, he knows the road length ℓ_e and how many fans will ask for autographs on that road a_e. Jatin doesn't want to sign more than N autographs. Give a poly$(N, |V|, |E|)$ time algorithm to find the shortest $s - t$ path P satisfying $\sum_{e \in P} a_e \leq N$, or report that no such path exists. (If you plan to use a DP , then just write subproblem, recurrence, final solution, pseudocode and runtime. No proof of correctness required. For any other (non-DP) algorithm, you must give formal proof of correctness.)

Problem 2 (10 points) Solve problem 15 from the above text. (You just need to give the proper construction and explain why computing flow gives you the correct answer. No formal proof of correctness required. But the construction should be absolutely clear. Use figures if necessary)

Bonus Problem (9 points). Suppose you are given a directed graph $G(V, E)$ with positive weights on the edges, as source vertex s and a destination t. Also, the graph has a special property - the shortest $s - t$ path includes every other vertex in the graph.

Now the edges are vulnerable and can fail. So, you want to maintain alternate shortest path distances between s and t in case some edge $e \in E$ fails. Design an $O(E \log V)$ algorithm that computes, for all $e \in E$, the shortest $s - t$ path distance in the graph $G \setminus e$. Your final output should be an array of size E where the entry for edge e should contain the shortest $s - t$ path distance in $G \setminus e$.

(You need to describe the algorithm properly with all data structures you are using and explain its working. No formal proof of correctness required)