CS21003 - Tutorial 10

November 3rd, 2017

1. You are given an undirected graph where each edge has cost 1. A path p between u and v is a shortest path if the sum of cost of the edges on p is minimum among all the paths between u and v. Given two vertices u and v of the graph you need to find the cost of shortest path between u and v. How can this be done? Can this idea be generalised if all the edges have same positive cost c?

2. Prove or disprove: BFS and DFS algorithms on a undirected, connected graph $G = (V, E)$, produce the same tree if and only if G is a tree.

3. There are N variables $x_1, x_2, \ldots, x_N$ and M relations of the form $x_i < x_j$ where $i \neq j$. A subset S of relations is called inconsistent if there does not exist any assignment of variables that satisfies all the relations in S. e.g, $\{x_1 < x_2, x_2 < x_1\}$ is inconsistent. You need to find if there is an inconsistent subset of M.

4. Let G be a strongly connected digraph, and you run DFS over G. Which of the 4 types of edges can exist in G?

5. There are N cities in Magicland. Some of the cities are connected by some roads. A road connects two cities and is bi-directional, i.e., we can go either way through a road. There is a path from a city i to every other city j and there is no cycle among the cities. You need to find a pair of cities (i, j) such that the length of the path between i and j is maximum among all such pairs. The length of a path is the number of edges on the path. [**Hint:** There is a trivial $O(n.(n + e))$ algorithm to solve this. However, can you give an $O(n + e)$ algorithm that uses some given property?]

6. Let $G = (V, E)$ be an undirected graph. A vertex $v \in V$ is called a cut vertex or an articulation point if the removal of v (and all edges incident upon v) increases the number of connected components in G. Your task is to find all cut vertices in G. What is the running time of this algorithm?

7. When an adjacency-matrix representation is used, most graph algorithms require time $\Omega(V^2)$, but there are some exceptions. Show that determining whether a directed graph G contains a universal sink (a vertex with in-degree $|V| - 1$ and out-degree 0) can be determined in time $O(V)$, given an adjacency matrix for G.

8. Given a directed acyclic graph G, design an $O(n + m)$ time algorithm which finds the length of the longest path of the graph.
   (a) Find a topological sort of the given DAG and let $v_1, v_2, \ldots, v_n$ be a topological sort, i.e., each edge is from a vertex v_i to another vertex v_j with $j > i$. Let $A[i]$ be the longest path of the graph starting at v_i. Find a formula for computing $A[i]$.
   (b) If we compute $A[1], A[2], \ldots, A[n]$, what would be the final solution?
   (c) Write a dynamic program for filling array A. What is the running time of this algorithm?
   (d) Run DFS and compute $A[i]$ during $DFS(v_i)$. How do you compute $A[i]$ during $DFS(v_i)$? What is the running time of this algorithm?