CS345A: Algorithms -II
Users Online : 6

Mid Semester Exam

Submitted on 14/9/2021 09:52

Instructions

- Exam opens at: 14/9/2021 08:00
- You are given an extra 10 minutes after due time to submit your exam.
- However, please note that any submissions made after the due time are marked as late submissions.

Mid Semester Exam

Question:

1. Please upload your answer on time. You must not email your answer to the instructor or any TA under any circumstance.

2. You may refer to any algorithm discussed in the lectures directly if you wish instead of reproducing it. However, any such reference must be formal, complete, and unambiguous.

3. There are 2 problems in this exam.

Problem 1: This problem consists of the following 2 questions. Attempt and submit the solution of **exactly one** of them.

Question 1: Searching for Shahrukh Khan (*marks=6*)

For any pair of persons (p,q), person p may either know person q or does not know person q. Note that if person p knows person q, it is not necessary that person q also knows p. For example, I know Shahrukh Khan but Shahrukh Khan does not know me. With this basic knowledge, now we define an algorithmic problem.

There is a set S of n persons, uniquely numbered from 1 to n. There is a $n \times n$ Binary matrix M such that for each $1 \leq$ $\leq n$ with $i \neq j$, M[i,j]=1 if person i knows person j, and 0 otherwise. A person in set S is said to be Shahrukh Khan if every other person knows him but he knows none. The following is the pseudocode of an O(n) time algorithm to determine Shahrukh Khan in the set S, if exists. As your answer, you just need to fill in the blanks. Note that each blank is a number or a variable or a Boolean expression.

```
Finding_Shahrukh_Khan(M)
{    t ← 1;
     For(i= ................ to n)
     {   If (..........................)   t ←  ...................;
     }
     flag ← true;
     For(i=1 to n)
     {   If (i ≠ t)
```

CS345A: Algorithms -II
Users Online : 6

Question 2: *Restoring Distances* (marks=10)

Let G=(V,E) be a directed graph on n=|V| vertices and m=|E| edges. Each edge has a positive edge weight (or length G is represented in the form of Adjacency lists.

Let s in V be a source vertex and every other vertex is reachable from s. Moreover, no two paths from s in G have th same length. We had computed a distance array D such that D[i] stores the distance from s to vertex i for each i. W had also computed the shortest path tree T rooted at s and stored it in a file. Some malicious hacker deleted this file Moreover, he changed **exactly** one entry in the distance array D. Unfortunately, we do not know which among the n entries has been changed. Our aim is to restore D to its original correct form.

You are told that the hacker, with the intention of causing maximum damage, changed the entry of a node which is certainly not a leaf node in the corresponding shortest paths tree rooted at s. Fortunately, you have some vague memory of the shortest paths tree and one thing you remember correctly is that no internal node in the shortest path tree T has a single child. You are given the array D, vertex s, and an adjacency lists representation of G. Your a is design an O(m+n) time algorithm to restore D to its original correct form. You may use at most O(n) extra space i you wish.

Note: As answer, you just need to describe your algorithm. There is absolutely no need to provide any intuition, analysis, or proof of correctness.

Problem 2: This problem consists of the following 2 questions. Attempt and submit the solution of **exactly one** of them.

Question 1: *Dynamic Programming* (marks=5)

There is an undirected graph G=(V,E) consisting of n=|V| vertices $v_1,...,v_n$. For each $1\leq i < n$, there is an edge betwe v_i and v_{i+1}. Hence there are only n-1 edges. Vertex v_i has a weight w_i which is a positive real number. A subset $S \subseteq V$ said to be an independent set if for each $x,y \in S$, $(x,y) \notin E$.

Weight of an independent set is the sum of weights of its vertices. Our aim is to compute the weight of the maximu weight independent set of G. Pursuing Dynamic Programming approach, let F(i) denote the weight of the maximum weight independent set for the set $\{v_1, v_2, \dots, v_i\}$. It is easy to observe that F(0)=0 and F(1)= w_1.

Note: As answer to this question, you need to just provide (1) a recursive formulation of F(i) for any i>1 along with a brief justification for this formulation, and (2) an iterative implementation of this formulation whose time complexity is a polynomial in n.

Question 2: *Neister Tree* (marks=10)

Let G=(V,E) be an undirected graph on n=|V| vertices in which each pair of vertices is joined by an edge of positive weight. We use $l_{i,j}$ to denote the weight of edge (i,j); and we will assume these weights satisfy triangle inequality, th is, $l_{i,t} \leq l_{i,j} + l_{j,t}$ for each i,j,t in V. For a subset $S\subseteq V$, we will use G[S] to denote the subgraph of G induced on the vertices of S. You are given a set $X \subseteq V$ of k vertices, where $k \leq n$ is a parameter. A Neister tree on X is a set Z so th $X\subseteq Z \subseteq V$, together with a spanning tree T of G[Z]. The weight of the Neister tree is the weight of the tree T. Desig an $O(n^{O(k)})$ time algorithm to compute a minimum weight Neister tree on X.

Note: As answer, you need to state the key insights/observations (without proofs) on the minimum weight Neister

CS345A: Algorithms -II
Users Online : 6

cs345 midsem.pdf

Grades:

Marks: Not graded