COT5405: ANALYSIS OF ALGORITHMS

Final Exam A

Date: May 5, 2006, Friday
Time: 7:30am – 9:30am
Professor: Alper Üngör (Office CSE 430)

This is a closed book exam. No collaborations are allowed. Your solutions should be concise, but complete, and handwritten clearly. Use only the space provided in this booklet, including the even numbered pages. Write your initials on each sheet. You should answer all the questions to get full credit.

GOOD LUCK!

1. [20= points] TRUE/FALSE

For each of the following statements decide whether it is true or false.

(a) T/F. Recall the 2-approximation algorithm described in class for the vertex cover problem. Using this algorithm, we can output all the vertices that are not in the obtained vertex cover. This output is a 2-approximation solution for the maximum independent set problem.

(b) T/F. Given a graph G we want to find whether G contains a complete subgraph (clique) of size 5. This problem is NP-Complete because it is a special case of the CLIQUE problem.

(c) T/F. The running time of the Quicksort algorithm is $O(n^3)$.

(d) T/F. Minimum spanning tree algorithms due to Prim and Kruskal and the shortest path algorithm of Dijkstra are examples of greedy algorithms, while the all-pairs shortest path algorithm of Floyd and Warshall is a dynamic-programming algorithm.

(e) T/F. Consider the decision version of the Shortest Path Problem: Given a graph $G = (V, E)$, two vertices $u, v \in V$, and an integer k, find whether there is a path from u to v of length at most k. This problem is in NP.

(f) T/F. Euclidean TSP problem is NP-hard implies that Metric TSP is also NP-hard.

2. [20=10+10 points] STRINGS

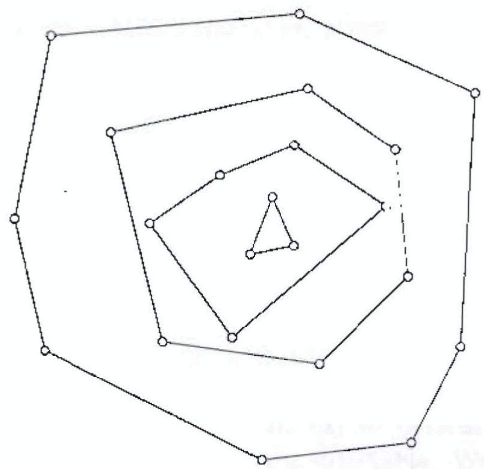
A *palindrome* is a text string that is exactly the same as its reversal, such as RACECAR, and ANASTASMUMSATSANA.

(a) Describe and analyze an algorithm to find the longest prefix of a given string that is also a palindrome. For instance, OTTO is the longest palindromic prefix of OTTOMAN, while S is the the longest palindromic prefix of SNOWMAN. For full credit your algorithm should run in $O(n)$ time.

(b) Describe and analyze an algorithm to find the length of the longest subsequence of a given string that is also a palindrome. For instance, KONOK is the longest palindromic subsequence of OKONOMIYAKI. You do not need to compute the actual subsequence, just its length. For full credit your algorithm should run in $O(n^2)$ time.

3. [20 points] GEOMETRIC ALGORITHMS

Given a set P of points in the plane, define the convex layers of P as follows: The first convex layer of P is just the convex hull of P. For all $i > 1$, the ith convex layer is the convex hull of P after the vertices of the first $i - 1$ layers have been removed. Describe an $O(n^2)$-time algorithm to find all convex layers of a given set of n points.



4. [20=10+10 points] SATISFIABILITY AND P VS. NP

A boolean formula is in disjunctive normal form (DNF) if it consists of clauses of conjunctions (ANDs) joined togehter by disjunctions (ORs). for example the formula

$$(a \wedge \neg b \wedge d) \vee (\neg a \wedge c) \vee (b \wedge \neg c \wedge \neg d)$$

is in DNF. DNF-SAT is the problem that asks, given a boolean formula in DNF, whether the formula is satisfiable.

(a) Show that DNF-SAT is in P.

(b) What is wrong with the following argument that P=NP?

> Suppose that we are given a boolean formula in CNF with at most three literals per clause, and we want to know if it is satisfiable. We can use the distributive law to construct an equivalent formula in DNF. For example,
>
> $$(a \vee b \vee \neg c) \wedge (\neg a \vee \neg b) = (a \wedge \neg b) \vee (b \wedge \neg a) \vee (\neg c \wedge \neg a) \vee (\neg c \wedge \neg b)$$
>
> Now we can use the answer to part (a) to determine, in polynomial time, whether the resulting DNF formula is satisfiable. We have just solved 3CNF-SAT in polynomial time! Since 3CNF-SAT is NP-hard, we must conclude that P=NP.

5. [20 points] DECISION VS. OPTIMIZATION PROBLEM

Suppose you have a black box that magically solves CLIQUE (the decision problem) in constant time. That is, given a graph and an integer k, the black box tells you, in constant time, whether or not the graph has clique of size k or larger.

(a) Using the black box, design and analyze a polynomial-time algorithm that computes the size of the largest clique in a graph.

(b) Using the black box, design and analyze a polynomial-time algorithm that finds a subset of the vertices that is a clique of largest size in the graph.