CMSC 451 Final Exam

This exam is closed-book and closed-notes. You may use two sheets of notes (front and back). You may use any algorithms or results given in class. The total point value is 120 points. Good luck!

Problem 1. (15 points, 5 points each) Throughout, assume that distances are measured using the L_1 (or Manhattan) metric. Consider the point set and minimum spanning tree shown in Fig. 1.

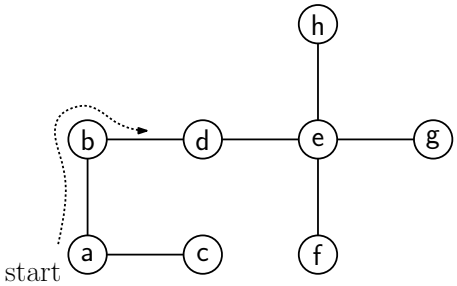

Figure 1: TSP heuristics.

(a) List the sequence of vertices (starting and ending at a) that result from a *twice-around tour* of the minimum spanning tree and *short-cutting*. For consistency, your tour should proceed in clockwise order, as indicated in the figure.

(b) Let's execute the first part of Christofides algorithm. Show (by drawing a figure) which vertices are involved in the minimum-weight matching and which edges are in the matching, when applied to the spanning tree from part (a).

(c) Let's complete Christofides algorithm. Determine an Eulerian circuit (starting at a), and then list the final TSP tour that results from short-cutting.

Problem 2. (40 points: 3–8 points each) Short answer questions. (Unless otherwise specified, explanations are not required but may be given for partial credit.)

(a) Given an undirected graph $G = (V, E)$ with n vertices and m edges, what (if anything) can you say about the *average vertex degree* in G?

(b) Given an acyclic directed graph $G = (V, E)$, define a *topological ordering* of the vertices. Assuming the graph has n vertices and m edges, how long does it take to compute a topological ordering.

(c) Let $G = (V, E)$ be a digraph, where $n = |V|$ and $m = |E|$. What is the worst-case running time of the Bellman-Ford algorithm on G? (No explanation needed.)

(d) True or False: The Floyd-Warshall algorithm produces a correct result if it is applied to a directed graph that has negative cost edges, assuming there is no negative cost cycle.

(e) Given two strings X and Y, where $|X| = m$ and $|Y| = n$ and $m \leq n$. As a function of m and n, what is the maximum possible edit (Levenshtein) distance between these strings? (Assume that the allowed editing operations are insert, delete, and replace/change.)

(f) Given an s-t network G and a valid flow f in this network, let G_f denote the *residual network*. Suppose there is no path from s to t in G_f. What can be said about the flow f?

(g) A scientist returns from the future in their time machine and informs us that: (1) 3SAT is solvable in $O(n^9)$ time, and (2) no algorithm for 3SAT exists that runs faster than $\Omega(n^9)$ time. Which of the following holds as a consequence? (List all that are true. No explanations are required.)

(i) All NP-complete problems are solvable in polynomial time.

(ii) All NP-complete problems are solvable in $O(n^9)$ time.

(iii) All problems in NP, even those that are not NP-complete, are solvable in polynomial time.

(iv) No NP-complete problem can be solved faster than $O(n^9)$ time in the worst case.

Problem 3. (10 points) Recall the drone-delivery problem from Homework 4. The *original requirements* from the homework problem still apply:

- There are m drone stations and n customers.
- Customer j has ordered $o_j \geq 1$ deliveries, which must be provided by stations that are within 10 miles of the customer. This customer is satisfied if at least $\max(1, o_j - 2)$ deliveries arrive.
- Each drone station makes at most 2 deliveries to any single customer and a total of at most 5 deliveries overall.

To this, we add the following *new requirements*:

- To be profitable, each drone station must make a total of at least 3 deliveries.
- The total number of deliveries from all stations to all customers, must be at least 20 and can be at most 50.

Present a reduction from this problem either to max-flow or the circulation problem, and provide a figure illustrating the elements of your reduction. Briefly justify your reductions correctness. A formal proof is *not* required.

Problem 4. (15 points) In the *Clique decision problem* (Clique) we are given a graph $G = (V, E)$ and an integer k, where $0 \leq k \leq |V|$, and we are asked whether there exist a subset $V' \subseteq V$ of size k such that all the vertices of V' are adjacent (see Fig. 2).

Suppose that you have a polynomial time oracle `clique`(G, k) that returns "yes" if G has a clique of size k and "no" otherwise.

(a) (5 points) Explain how to use the oracle to determine the size k^* of the largest clique in G.

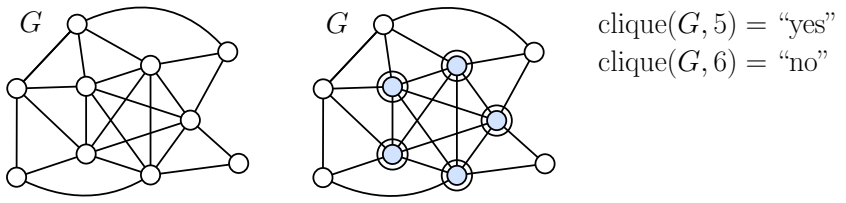

Figure 2: Clique decision problem.

(b) (10 points) Given the optimal clique size k^* (from part (a)), explain how to use the oracle to compute the vertices of any clique of size k^* in G in polynomial time. Provide a short justification (not a detailed proof) of your algorithm's correctness and its running time.

Problem 5. (20 points) Given a directed graph $G = (V, E)$, a *Hamiltonian path pair* consists of two paths π_1 and π_2 in G such that:

- both π_1 and π_2 start at the same vertex and both end at the same vertex,
- other than the start and end, every vertex appears in *exactly one* of these paths, and
- both paths have the same number of vertices (see Fig. 3).

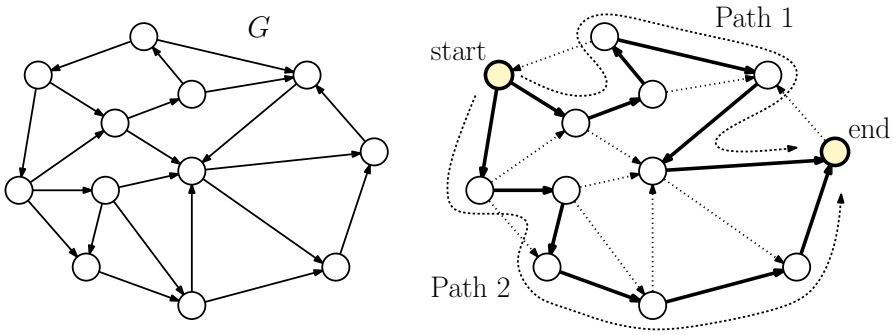

Figure 3: The Hamiltonian Path Pair problem (HPP).

The Hamiltonian Path Pair problem (HPP) is as follows. Given a directed graph G, does there exist a Hamiltonian path pair. (Note that G must have an even number of vertices, or else the answer is trivially "no".)

Prove that HPP is NP-complete by completing the following two parts.

(a) (5 points) Prove that HPP $\in$ NP.

(b) (15 points) Prove that HPP is NP-hard. (Hint: Reduction from directed Hamiltonian path, DHP.)

Note: If you don't see how to do this, for half credit you can give the proof from Homework 5 that undirected Hamiltonian Path (HP) is NP-complete. The reduction is from Directed Hamiltonian Path (DHP).

Problem 6. (20 points) In this problem, we will consider a variant of the subset-sum (SS) approximation algorithm, which we call *balanced subset sum* (BSS). We are given two disjoint sets of positive integers, $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$, and a target value t. We are asked to find a subset of numbers $S' \subseteq A \cup B$ whose sum is as close as possible to t, but without exceeding t. In addition, the number of elements of S' that come from A must be the same as the number of elements of S' that come from B. (Note: It is the numbers of elements that must match, not the sum of these elements.)

(a) (10 points) Present an algorithm that solves this problem (*exactly*) in time that is polynomial in n and t. Briefly explain your algorithm and derive its running time.

Justify the correctness of your algorithm and derive its running time. (You may reuse operations from lecture, such as `merge`(L_1, L_2) and `trim`(L, t).)

(b) (10 points) Present a polynomial-time approximation scheme (PTAS) for BSS. That is, given ε, such that $0 < \varepsilon < 1$, your algorithm should produce a valid answer z such that $(1 - \varepsilon)z^* \leq z \leq z^*$, where z^* is the optimum BSS solution. The running time should be polynomial in $1/\varepsilon$, n, and $\log t$.

Justify the correctness of your algorithm and derive its running time. (You may reuse operations from lecture, such as `merge`(L_1, L_2) and `compress`(L, δ, t).)