Practice Midterm #2

Instructions. This is a 150-minute practice test from last year’s final exam, with a total of 105 points available. There are five pages and seven (eight) sets of problems. Note that you may just give the name of an algorithm to answer a question if the algorithm has been discussed in class, without rewriting its code. Pace Yourself and Good Luck!

Problem 1. (40 pts total) Please give concise answers for the following questions.

Q1. A sequence of $n$ operations are performed on a data structure initially in state $D_1$. The $i$th operation transforms the data structure from State $D_i$ to state $D_{i+1}$, $1 \leq i \leq n$. Let

$$c(D_i) = \begin{cases} 
2\sqrt{i}, & \text{if } i \text{ is a perfect square (i.e., } 1^2, 2^2, 3^2, \ldots) \\
2, & \text{otherwise}
\end{cases}$$

denote the cost of performing the $i$th operation, $1 \leq i \leq n$. Use the potential function

$$\Phi(D_i) = i - \left(\sqrt{i} - 1\right)^2$$

to show that the amortized cost for performing one operation of this sequence is constant when $i$ is a perfect square.

Q2. Leonhard Euler in 1736 first considered the following mathematical puzzle: The city of Königsberg has seven bridges $b_i, i = 1, \ldots, 7$, with two river banks $A$ and $B$ and two islands $C$ and $D$, as shown in the figure below. Euler wondered if it is possible to start at some place in the city, cross every bridge exactly once, and return to the starting place. Model this problem as a graph problem. (You do not need to answer the following question now, but you might want to work on it during the winter break: Does such a tour exist? Why?)
Q3. Think of the network shown below as a highway map, and the number recorded next to each arc as the maximum elevation encountered in traversing the arc. A traveler plans to drive from node 1 to node 12 on this highway. This traveler dislikes high altitudes and so would like to find a path connecting node 1 to node 12 that minimizes the maximum altitude. Give an efficient algorithm to find the best path for this traveler. **Justify your algorithm.** (Hint: This is related to a homework problem on graph algorithms.)

Q4. Professor Vai presents the following edge reweighting method for the shortest-path problem with negative weights. Letting $w^* = \min_{(u,v) \in E} \{w(u, v)\}$, he defines $w'(u, v) = |w(u, v) - w^*|$ for all edges $(u, v) \in E$. With this reweighting method, can he always correctly compute a shortest path? Why?

Q5. Let $G = (V, E)$ be a directed graph with edge costs modelled by the corresponding weights. The **bottleneck** of a path is defined as the **minimum** edge cost among all the edges on the path. Suppose that we want to find a **maximum** bottleneck path between each pair of vertices. Show how to modify Floyd-Warshall’s all-pair shortest-path algorithm to solve this problem in $O(V^3)$ time.

```
Floyd-Warshall(W)
1. n = W.rows
2. D(0) = W
3. for k = 1 to n
4. let $D^{(k)} = (d^{(k)}_{ij})$ be a new $n \times n$ matrix
5. for i = 1 to n
6. for j = 1 to n
7. $d^{(k)}_{ij} = \min(d^{(k-1)}_{ij}, d^{(k-1)}_{ik} + d^{(k-1)}_{kj})$
8. return $D^{(n)}$
```

Q6. Professor Chang claims the apparent paradox between statements S1 and S2. Nevertheless, he may not be wrong (i.e., both S1 and S2 could be true simultaneously). Give two possible reasons for this phenomenon.

S1: $P \neq NP$. In other words, there does not exist a polynomial time algorithm for any NP-complete problem.

S2: There exists a transformation (reduction) from some particular instance of the NP-complete HP problem to a shortest path problem solvable by Dijkstra’s algorithm.
Q7. Consider the Degree-Constrained Spanning-Tree problem (DCST) described below.

- **Instance:** $G = (V, E)$, a positive integer $K \leq |V|$.
- **Question:** Is there a spanning tree for $G$ in which no vertex has degree (number of edges connected to a vertex) larger than $K$?

We know that the Hamiltonian Path (HP) problem of finding a simple path that visits each vertex in a given graph exactly once is NP-complete. Prove that DCST is NP-hard.

Q8. Consider the following “closest point” heuristic for the traveling-salesman problem (TSP) whose cost function satisfies the triangle inequality. Begin with a trivial cycle consisting of a single arbitrary vertex. At each step, identify a vertex $u$ not on the cycle whose distance to any vertex on the cycle is minimum. Let the nearest vertex to $u$ on the cycle be $v$. Extend the cycle by inserting $u$ just after $v$. Analyze the performance bound of this approximation algorithm.

Problem 2. (10 pts total) Alan decides to follow Mayor Ko’s footsteps to bike for 380 kilometers from Taipei to his hometown Kaohsiung to cast his first vote for the presidential election this Saturday. He needs to take a rest at a 7-11 convenience store to eat a banana and drink water to travel 20 kilometers, and his map gives the distance between each pair of convenience stores on his route. He wishes to make as few stops at convenience stores as possible along the way. Give an efficient algorithm to determine at which convenience stores he should stop and prove the optimality of your algorithm.

Problem 3. (10 pts total) Consider the chessboard shown below. Note that some squares are shaded, denoting blockages. Any tour must not visit these shaded squares.

(a) (5 pts) We wish to determine a shortest path, if one exists, that starts at the square designated by $s$ and after visiting the minimum number of squares, ends at the square designated by $t$. Formulate this problem on a graph. Give an efficient algorithm to solve this problem. What is the time complexity of your algorithm?

(b) (5 pts) Suppose that each boundary of two squares models the penalty or profit for passing through it, i.e., the cost/weight could be positive or negative. Formulate this problem of finding the minimum cost tour on a graph. Give an efficient algorithm to solve this problem. What is the time complexity of your algorithm?
Problem 4. (10 pts total) A farmer wishes to transport a truckload of eggs from one city to another city through a given road network. The truck will incur a certain amount of breakage on each road segment; let \( w(u,v) \) denote the fraction of broken eggs if the truck traverses the road segment \((u,v)\), where \(0 < w(u,v) \leq 1\). To solve this problem, you are provided with a binary-code solver \(\text{Dijkstra}(G, w, s)\) for solving the single-source shortest-paths problem, where \(G\) is an input graph, \(w\) is the weight function, and \(s\) is the source. Show how you can apply this binary-code solver to find the route (path) with the minimum total breakage between \(s\) and other vertices. Justify the correctness of your algorithm? (Hint: How can you modify the graph and/or its edge weights? Notice that you cannot change the code of the Dijkstra solver because the given is a binary code.)

\[
\text{Dijkstra}(G, w, s) \\
1. \text{Initialize-Single-Source}(G, s) \\
2. \quad S = \emptyset \\
3. \quad Q = G.V \\
4. \quad \text{while} \ Q \neq \emptyset \\
5. \quad \quad u = \text{Extract-Min}(Q) \\
6. \quad \quad S = S \cup \{u\} \\
7. \quad \quad \text{for each vertex} \ v \in G.\text{Adj}[u] \\
8. \quad \quad \quad \text{Relax}(u, v, w) \\
9. \text{Initialize-Single-Source}(G, s) \\
1. \quad \text{for each vertex} \ v \in G.V \\
2. \quad \quad v.d = \infty \\
3. \quad \quad v.\pi = \text{NIL} \\
4. \quad \quad s.d = 0 \\
5. \quad \text{Relax}(u, v, w) \\
1. \quad \text{if} \ v.d > u.d + w(u, v) \\
2. \quad \quad v.d = u.d + w(u, v) \\
3. \quad \quad v.\pi = u
\]

Problem 5. (10 pts total) In the flow network shown below, the number beside an edge denotes its corresponding capacity. Apply the Edmonds-Karp algorithm to find a maximum flow from \(s\) to \(t\) in the network. Show every augmentation path (but not the whole network to save time) and explain why the final flow you obtained is maximum.

![Flow Network Diagram]
Problem 6. (15 pts total) You are arranging a dating party for \( p \) female members \( f_1, f_2, \ldots, f_p \) and \( p \) male members \( m_1, m_2, \ldots, m_p \). Each female member ranks two male members she would like to date, ranking them according to her preference.

(a) (6 pts) We say that a dating assignment is a feasible assignment if every female member dates with a male member within her preference list. How would you find a feasible assignment?

(b) (9 pts) A feasible assignment is said to be \( k \)-feasible if it assigns at most \( k \) female members to their second most preferred male members. For a given \( k \), develop an algorithm for determining a \( k \)-feasible assignment.

Problem 7. (10 pts total) A vertex cover of an undirected graph \( G = (V, E) \) is a subset \( V' \subseteq V \) such that if \( (u, v) \in E \), then \( u \in V' \) or \( v \in V' \) (or both). The Vertex-Cover Problem (VC) is to find a vertex cover of minimum size in \( G \); let the decision problem of VC be \( VC_D \).

The 3-CNF-SAT problem (3SAT) is defined as follows:

- **Definition:** 3SAT = \{< \phi >: boolean formula \( \phi \) in 3-conjunctive normal form is satisfiable\}.

Use the reduction from 3SAT shown below to prove that \( VC_D \) is NP-complete. Notice that no partial credit will be given for any reduction not from 3SAT. (Hint: What is the size of the vertex cover for the reduced graph?)

\[
\begin{align*}
\phi &= (x_1 \lor \neg x_3 \lor \neg x_4) \land (\neg x_1 \lor x_2 \lor \neg x_4)
\end{align*}
\]

Problem 8. (bonus) (This problem can be answered by email by 1pm January 18, 2016 after the final exam.) Please list the corrections to the class notes and lectures you made in this semester, if any. Please give specific information on the corrections, e.g., page numbers of the class notes, if possible.