| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |
| | | | | |
| 6 | 7 | 8 | 9 | Total |

**The University of Texas at Austin**
**Department of Electrical and Computer Engineering**

**EE w360C — Algorithms — Summer 2013**

**Final Exam — Saturday 17 August 2013**

EID: ________________________

Name: ________________________

## Instructions.

- You have 3 hours to complete the exam. The maximum possible score is 120 including 40 marks bonus questions.
- The exam consists of 9 questions and 12 printed pages.
- It is a closed book / closed notes exam. No calculators, laptops, or other devices are allowed.
- Write your answers legibly on the test pages. Use back of test pages for scratch work. Show intermediate answers and process of solving questions.
- If there is any confusion, write down your assumptions and proceed to answer the question.
- In questions where you have to write an algorithm, you can describe it with reference to the algorithms we discussed in class. For example, you can say BFS with some additional operation at one step. Or use the output of topological sort directly etc.

| |
|---|
| 20 pts |

**1.** Answer briefly. However, a yes/no or a number answer is not enough. You need to give an argument and reason to get any marks.

(a) Given a polynomial algorithm and an exponential algorithm and a problem of size 1000, which will take less time?

*Solution*:

(b) If algorithm A is slower than algorithm B on all instances of a given problem with size ¿ 1000, can you say $A = O(B)$ or $B = O(A)$?

*Solution*:

(c) If an algorithm divides the problem into 3 parts of equal size and spends logarithmic time on dividing and combining parts, write its recurrence relation.

*Solution*:

(d) Solve the above recurrence relation using Master's theorem.

*Solution*:

(e) Prove that its not important to write the base of logarithm in big O notation.

*Solution*:

(f) Prove that a bipartite graph cannot contain an odd cycle.
*Solution*:

(g) Prove that in a directed graph, if every node has incoming edges, then the graph contains a cycle.
*Solution*:

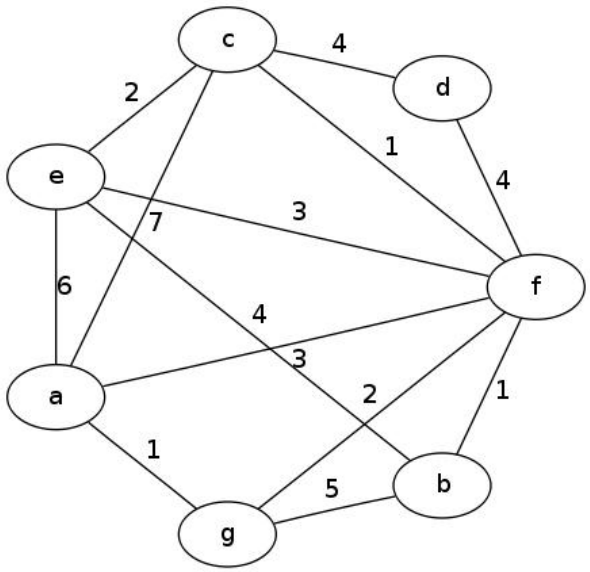
(h) What is the conservation condition in a network flow?
*Solution*:

(i) Prove that in the cut found in previous part, all edges crossing the cut from sink side to source side have no flow.
*Solution*:

(j) In a dynamic programming algorithm, is the iteration technique better than the memorization technique in terms of time complexity?
*Solution*:

**2.** Find the minimum spanning tree using Prim's algorithm. List the steps taken by the algorithm.

12 pts




*Solution*:

| |
|---|
| |
| 12 pts |

**3.** Apply the longest common subsequence algorithm over the words "entertain" and "tertiary". Show the dynamic programming matrix M. Trace back the actual longest common subsequence on your matrix.

*Solution*:

**4.** Suppose now that you're given an $n \times n$ grid graph $G$. (An $n \times n$ grid graph is just the adjacency graph of an $n \times n$ chessboard. To be completely precise, it is a graph whose node set is the set of all ordered pairs of natural numbers $(i, j)$, where $1 \leq i \leq n$ and $1 \leq j \leq n$; the nodes $(i, j)$ and $(k, l)$ are joined by an edge if and only if $|ik| + |jl| = 1$.)

12 pts

Each node $v$ is labeled by a real number $x_v$; you may assume that all these labels are distinct. Show how to find a local minimum of $G$ using only $O(n)$ probes to the nodes of $G$. (Note that $G$ has $n_2$ nodes.) A node $v$ of $T$ is a local minimum if the label $x_v$ is less than the label $x_w$ for all nodes $w$ that are joined to $v$ by an edge.

*Solution*:

**5.** Assume that all edge costs are distinct in a graph $G = (V, E)$. Let $S$ be any subset of nodes that is neither empty nor equal to all of $V$, and let edge $e = (v, w)$ be the minimum-cost edge with one end in $S$ and the other in $VS$. Prove by contradiction that every minimum spanning tree contains the edge $e$. 12 pts

*Solution*:

**6.** Suppose you are given a directed graph $G = (V, E)$, with a positive integer capacity $c_e$ on each edge $e$, a designated source $s \in V$, and a designated sink $t \in V$. You are also given an integer maximum $s$–$t$ flow in G, defined by a flow value $f_e$ on each edge $e$.

12 pts

Now suppose we pick a specific edge $e \in E$ and increase its capacity by one unit. Show how to find a maximum flow in the resulting capacitated graph in time $O(m+n)$, where $m$ is the number of edges in $G$ and $n$ is the number of nodes.

*Solution*:

**7. Bonus:** Answer briefly. However, a yes/no or a number answer is not enough. You need to give an argument and reason to get any marks.

16 pts

(a) Show how to solve the decision version of a problem using the optimization version as a blackbox.
*Solution*:

(b) If problem A can be solved using problem B as a blackbox and B has a polynomial algorithm, what can we say about their time complexity?
*Solution*:

(c) If problem A can be solved using problem B as a blackbox, which is the harder problem and why?
*Solution*:

(d) To show two problems A and B are equally hard, we need to reduce them to each other. But if A is NP-Complete, we are done after showing just one reduction. Which reduction is that and why is it enough?
*Solution*:

(e) If someone solves the knapsack problem in polynomial time, what can you say about the vertex cover problem.
*Solution*:

(f) Given a polynomial algorithm for the knapsack problem, is P=NP ?
*Solution*:

(g) Prove that the decision form of independent set problem (is there an independent set of size $k$ in a graph) is in NP?
*Solution*:

(h) Consider the decision problem: is there a path between nodes $u$ and $v$ of cost $\leq k$. Is this problem in P, is it NP, is it NP-Complete?
*Solution*:

**8. Bonus:** Given ten items and a knapsack of capacity 115, find a way using the dynamic programming algorithm to fill it such that it achieves a value at least as good as the optimal solution for capacity 100. Show the filled matrix M. Your matrix should be as small as possible.

| | |
|---|---|
| | 12 pts |

| Item | Weight | Value |
|---|---|---|
| 1 | 73 | 96 |
| 2 | 46 | 81 |
| 3 | 53 | 65 |
| 4 | 61 | 73 |
| 5 | 34 | 54 |

*Solution*:

**9. Bonus:** Consider the problem of trying to collect baseball cards. Baseball cards come in sealed packets; you don't know what you're getting until you open up the package. Consider packets $P_1, P_2, \ldots P_m$, each of which contains some subset of this year's available baseball cards.

| |
|---|
| |
| 12 pts |

Consider the decision problem that asks whether it is possible to collect all of this year's available cards by buying fewer than $k$ packets of cards. Prove that this problem is NP-Complete using the Vertex Cover Problem (which is known to be NP-Complete).

*Solution*: