CS170 Final Examination 14 Dec 1999

NAME: ______________________

TA: ______________________

Be clear and concise. You may use the number of points assigned to each problem as a rough estimate for the number of minutes you want to allocate to the problem. The total number of points is 140. (This means that we hope you will have enough time to finish the exam!)

1	
2	
3	
4	
5	
6	
7	
Total	

1. **20 points.** Suppose we are given a sequence of n matrices A_1 through A_n. Your job is to write a subroutine to compute their product $P = A_1 \times A_2 \times \cdots \times A_n$. Here A_i has r_i rows and c_i columns, where $c_i = r_{i+1}$ so all the products are defined. Since you work for Microslop, which has a monopoly on all software, and a strategic relationship with Wintel, you want to construct an algorithm to *maximize* the running time, so that customers will want to buy newer, faster computers from Wintel to multiply their matrices.

   To avoid tipping off government antitrust inspectors, your algorithm must appear to be reasonable. This means that it must choose an order in which to multiply the matrices, and multiply any pair or matrices using a reasonable algorithm. In particular, if matrix B has r rows and s columns, and matrix C has s rows and t columns, then you should multiply $B \times C$ using the standard algorithm that does $2rst$ arithmetic operations.

   For example, if $n = 3$ then your algorithm can choose either $(A_1 \times A_2) \times A_3$, which does $2r_1c_1c_2 + 2r_1c_2c_3$ flops, or $A_1 \times (A_2 \times A_3)$, which does $2r_2c_2c_3 + 2r_1c_1c_3$ flops, whichever one does more flops.

   Let $MAXP(i, j)$ be the number of arithmetic operations your algorithm uses to multiply $A_i \times \cdots \times A_j$.

   (1) Give a formula for $MAXP(i, i + 1) =$

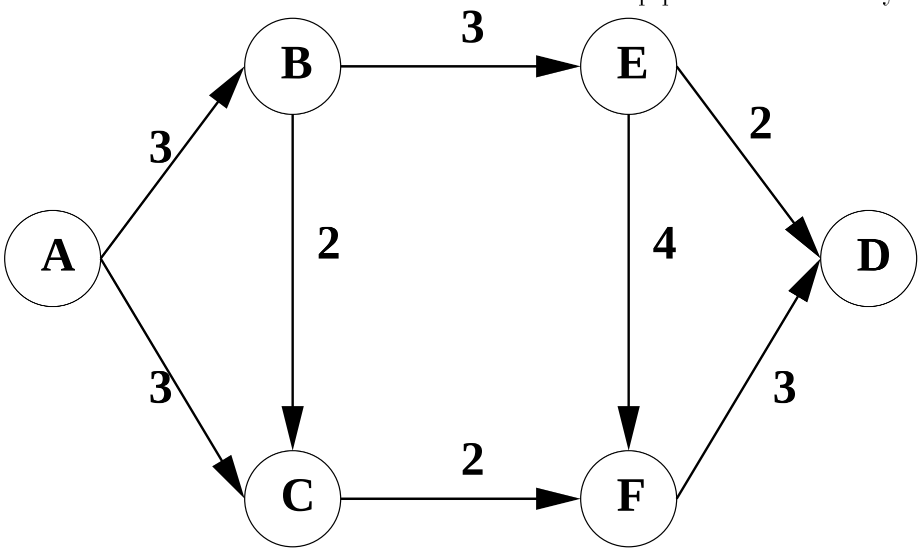
   (2) Give a recurrence equation: for $j > i + 1 > 0$, $MAXP(i, j) =$

   (3) Give an algorithm for computing $MAXP(i, j)$, and the order in which to multiply matrices to maximize the number of arithmetic operations. Use explicit loops in your solution. Because you have pride in your work despite your employer, you want your algorithm to compute $MAXP(i, j)$ to run as quickly as possible.

   Brief justification of your algorithm. Analyze the running time, in a $O(\cdot)$ sense.

2. **20 points.**

(a) The figure below represents a network of oil pipelines, which can transport oil from the source node A to the destination node D. Each pipeline is labeled by its capacity.



Compute the maximum flow from the source to the destination. Also fill in the table below of augmenting paths connecting source to destination in the order that your algorithm discovers them, along with the amount of net flow each path contributes. To make your answer unique when you have a choice of vertices to visit when traversing the graph, first visit the vertex with the lexicographically lower label (A before B, etc.). We have filled in the first path as an example:

Augmenting Path	Net flow contribution
A – B – C – F – D	2

(b) What edges form a minimum cut of the above network?

3. **20 points.**

Consider the network shown in the previous question.

(a) As an employee of the oil company, you are responsible for investing in new pipeline capacity. You can add capacity to any existing pipeline (edge); let c be the cost of adding one unit of capacity to any pipeline. Let C_e be the amount of new capacity added to edge e. C_e may or may not be an integer. The total profit TP is equal to

the price of oil d times the maxflow of the network,
minus the cost of the new capacity, which is equal to c times the new capacity added
minus fixed maintenance costs, equal to 100.

For example, if you added one unit of capacity $C_{AC} = 1$ to the pipeline from A to C, and if the maxflow of the network also increased by 1 from M to $M + 1$, then the total profit is $TP = d \cdot (M + 1) - c \cdot C_{AC} - 100 = d \cdot (M + 1) - c - 100$. If the maxflow does not increase, then $TP = M \cdot d - c - 100$.

Just for the network above, write down a linear program that computes the amount C_e to augment each pipeline e in order to maximize the total profit TP.

What are the variables in your program?

What are the constraints?

What function is to be maximized?

(b) When the price of oil d is very low, then it may be that adding capacity is not profitable. But when the price of oil is very high, it is very profitable to add capacity. For planning purposes, your boss asks you for the minimum price of oil d, such that for any greater price $TP > 0$ and at least one $C_e > 0$, i.e. the most profitable thing to do is add capacity. Say how you would use your solution to the previous problem to answer this question. (You should use your solution to the previous part as a "black-box", that returns the total profit TP and all the C_e as a function of the current network, and the costs d and c. It is ok to compute d just to the nearest penny, i.e. within $\pm.01$.) Your answer to this question should be an algorithm.

4. **20 points.** A 2-*Dominating set* of a graph $G(V, E)$ is a subset D of the vertices V, such that every vertex $v \in V$ is either in D, or can be connected to a vertex $d \in D$ by a path containing 1 or 2 edges. Show that the 2-DOMINATING SET problem, defined below, is NP–complete. You may assume that INDEPENDENT SET is NP–complete.

   - 2-DOMINATING SET: Given a graph G and an integer K, is it true that G has a 2-*dominating set* of size at most K?

     It is in NP *because ...*

     Reduction from to

     Justification :

5. **20 points.** APPROXIMATING DOMINATING SET. Let $G = (V, E)$ be a graph. A *dominating set* D is a subset of V such that every vertex $v \in V$ is either in D or adjacent to a vertex in D. The question of whether a set G has a dominating set of at most K vertices is known to be NP-complete.

Consider the following *approximation algorithm* for computing a dominating set:

```
AD = empty set
mark all vertices in V as "undominated"
while there are undominated vertices in V
      pick any undominated vertex v from V
      let N_v be the set containing v and all its nearest neighbors
      AD = AD ∪ N_v
      mark all the vertices in N_v, and all the neighbors of vertices in N_v, as "dominated"
end while
```

(a) Show that the set AD computed by the above algorithm is a dominating set.

(b) Show that any dominating set D must include at least one member of each N_v chosen by the algorithm

(c) Let δ be the largest number of neighbors of any vertex in G (ignore self edges). Prove that AD has no more than $\delta + 1$ times as many vertices as the smallest dominating set.

6. **20 points.**

You are running on a computer with 8-bit 2s complement arithmetic. One integer fits in one 8-bit byte.

(a) What is the representable range of integers?

(b) What is the value returned after the following computations? Express your answers as decimal integers. Which operations signal integer overflow?

126+1
126+2
-126 - 2
-126 - 3

(c) How many 8-bit 2s complement integers x have the property that after computing the 8-bit 2s complement integer $y = 37 \cdot x$, then $y = 10$? Justify your answer.

(d) How many 8-bit 2s complement integers x have the property that after computing the 8-bit 2s complement integer $y = 16 \cdot x$, then $y = 16$? Justify your answer.

7. **20 points.**

True or False? No explanation required, except for partial credit. Each correct answer is worth 1 point, but 1 point will be *subtracted* for each wrong answer, so answer only if you are reasonably certain.

(a) If A and B are NP-complete problems and we have a reduction directly from A to B, and A has a k-approximation algorithm, then B has a k^2-approximation algorithm.

(b) Let X be any NP-complete problem. MAX-FLOW can be reduced to X in polynomial time.

(c) If $d|a$, $d|b$, and $d \leq gcd(a, b)$, then $d|gcd(a, b)$.

(d) If a graph has a Hamilton tour, then it also has an Euler tour.

(e) An undirected graph $G = (V, E)$ is connected if and only if $|E| > |V|$.

(f) Removing an edge from a directed graph G always increases the number of strongly connected components.

(g) If a graph G has a unique topological sort, then it cannot have a strongly connected component of size greater than 1.

(h) A graph G without self-loops has a cycle iff DFS finds a backedge.

(i) An integer $n \geq 2$ is prime iff for every a not congruent to 0 mod n, $a^{n-1} \equiv 1 \bmod n$.

(j) An integer $n \geq 2$ cannot be composite if $a^n \equiv a \bmod n$ for all a.

(k) Let G be a directed weighted graph, and let u, v be two vertices. Then the shortest path from u to v remain the shortest path when 1 is added to every edge weight.

(l) If a graph is stored in memory in adjacency list form, with the adjacency list for each node stored in an array, then DFS makes better use of a cache (fast computer memory) than BFS.

(m) If all edge weights are unique, the solution (flows along each edge) of max flow is unique.

(n) Given x, x^{55} can be computed with 8 or fewer multiplications.

(o) CLIQUE can be reduced to HAMILTONIAN-PATH in polynomial time.

(p) A problem is in NP if and only if it reduces to CLIQUE in polynomial time.

(q) If $P \neq NP$, then a problem is NP-Complete if it reduces to 3-SAT.

(r) If $P = NP$, then the SHORTEST-PATH problem is NP-Complete.

(s) If p is prime, then $x^2 \equiv 1 \bmod p$ has only two solutions.

(t) You filled in your name and your TA's name on the first page of this test.