

Final Exam

December 11, 2001

READ THIS PAGE. DO NOT TURN TO THE NEXT PAGE UNTIL YOU ARE TOLD YOU CAN START.

- NAME: _____
- This is the CS 441T exam. If you are a CS 539T student please raise your hand so I can give you the CS 539T exam.
- If any question is not clear to you, come up to the front and I will answer any questions you have. There is no point cost (unlike with the hints) if you want to ask for clarification about the problem itself.
- Please write up all solutions clearly and legibly in the space provided. If needed use the back of the last page of the exam for extra space to write-up your solution for any problem. If you need scratch paper please come up front.
- When you are asked to give a lower bound, spend at most 10 minutes determining the best bound you can prove and then write-it up. Your write-up should have the following steps:
 - State the lower bound which you will prove.
 - Clearly describe your adversary strategy.
 - Prove the stated lower bound.
 - (OPTIONAL) Describe an alternate adversary strategy that you think is better but for which you can't prove any better bound. Also feel free to give a few sentences of thoughts about how you might prove a better bound with it.
- For any problem in which you want to prove a problem is NP-hard, you must use one of the following for your reduction: CIRCUIT-SAT, SAT, 3-CNF-SAT, CLIQUE, VERTEX-COVER, PARTITION, SUBSET-SUM, HAM-CYCLE, TSP. If you would like me to write the definition for any of these problems, I will (for no point deduction). Just come to the front.
- For any problem if you feel that you are stuck and need a hint, please come to the front and for an appropriate reduction in the maximum score you can get on that problem a hint will be provided.
- Relax and begin working when instructed to start.

HAPPY HOLIDAYS

1. (10 pts) Answer each of the following questions and briefly explain your answer. You are given as facts that 3-CNF-SAT and TSP (traveling salesman problem) are NP-complete and that there is a polynomial time algorithm for 2-CNF-SAT.

(a) True or false: 3-CNF-SAT \leq_p TSP.

(b) True or false: If $P \neq NP$, then no problem in NP can be solved in polynomial time.

(c) True or false: 3-CNF-SAT \leq_p 2-CNF-SAT (assume that $P \neq NP$).

(d) True or false: For a maximization problem P and an approximation A , the approximation ratio for A is defined as

$$\max_{\text{inputs } I} \left(\frac{\text{cost of } A\text{'s solution for } I}{\text{cost of the optimal solution for } I} \right)$$

(e) True or false: For linear programming if the extra constraint that all variables must taken an integral values is added then there is still a polynomial time algorithm to find the optimal solution.

2. (10 pts) Give the best lower bound you can for the problem of finding the median among the $2n + 1$ elements in the two sorted arrays $A[1] < A[2] < \dots < A[n] < A[n + 1]$ and $B[1] < B[2] < \dots < B[n]$ using a comparison-based algorithm that can only learn about the array elements by posing questions of the form is $A[i] < B[j]$ for $1 \leq i \leq n + 1$ and $1 \leq j \leq n$.

3. (10 pts) You are to prove that the graph problem (GP) defined below is NP-hard. For GP you are given as input a directed graph $G = (V, E)$, a source vertex $s \in V$, a destination vertex $t \in V$, and a set $C = \{(a_1, b_1), (a_2, b_2), \dots, (a_k, b_k)\}$ of pairs of vertices from V . The question is whether or not there is a directed path from s to t in G that contains at most one vertex in each pair in C .

4. (10 pts) Consider the problem of determining whether an undirected graph G with 8 vertices is connected (i.e. whether there is an undirected path between each pair of vertices). Assume that the algorithm is restricted to ask questions of the form “Is there an edge between i and j ?”. Give the best adversary argument you can on the number of such questions required to determine if G is connected. You will only receive only 3 pts for giving a lower bound of 16.

Hint: Think about how to extend the adversary technique given in the practice problem solution. For a deduction of 3 points I will show you the solutions posted for that problem. Think recursively.

5. (15 pts) Consider the following scheduling problem. There is a single resource on which only one job at a time can be run. For each job i ($1 \leq i \leq n$) there are two inputs given: the processing time p_i for job i and the payment function $f_i(t)$ which specifies the amount which job i will pay if it is completed at time t . The payment functions can be any non-increasing function. That is for $t_1 < t_2$ it must be $f_i(t_1) \geq f_i(t_2)$. Finally, you are given as part of the input a precedence relation (which can be viewed as a directed acyclic graph) where if there is an edge from job i to job j this means that job i must be scheduled before job j is scheduled. The goal is to find a legal schedule in which all n jobs are run that maximizes the payoff. Give a greedy solution for this algorithm and prove it is correct. You need NOT analyze the time complexity.

Additional space for problem 5.

6. (10 pts) Consider the following on-line problem. You have a video-on-demand service but are limited by your equipment to play only one movie (for one person) at a time. In an on-line fashion you receive requests where each request is for a 2 hour movie and a slack period $s \geq 0$ which indicates how long the customer is willing to wait for the movie to start. Furthermore, once you start a movie for a customer you must finish showing it before servicing another customer. Your goal is to decide which requests to service (and when to service them) as to maximize the number of jobs serviced.

Here is an on-line algorithm. If at any point in time you are not currently showing a movie and there is a customer available, then pick the one with the shortest time before their slack period ends and start their movie. Requests whose slack period ends before being selected are not serviced.

- (a) Consider the following input. There are two requests. The first comes at 6pm with a slack of 3 hours. The second comes at 7pm with a slack of 0. What profit would the above on-line algorithm obtain and what profit is obtained by an optimal solution.

(b) Prove this algorithm is 2-competitive.

Additional Work for Problem _____

Problem	Points Possible	Points Received
1	10	
2	10	
3	10	
4	10	
5	10	
6	15	
total	65	