

1. (20 points)

Consider the following maximization problem:

Input: An exact 3CNF formula  $F = C_1 \wedge C_2 \dots \wedge C_m$ . The goal is to maximize the number of clauses satisfied by at least 2 literals.

(a) (15 points)

Sketch a randomized algorithm such that the expected number of clauses returned (i.e. satisfied by at least 2 variables) is  $m/2$ . Briefly explain why the expected number of clauses is  $m/2$ .

SOLUTION: Randomly set each propositional variable  $x_i$  so that  $Prob[\tau(x_i) = true] = Prob[\tau(x_i) = false] = 1/2$ . It follows that for every clause  $C_j$ , the probability that  $C_j$  is satisfied by all three literals is  $\frac{1}{8}$ , and the probability that  $C_j$  is satisfied by exactly two of the three literals is  $\binom{3}{2}\frac{1}{8} = \frac{3}{8}$  so that the probability that  $C_j$  is satisfied by at least 2 literals is  $\frac{1}{2}$ . Thus by linearity of expectations, the expected number of clauses satisfied by at least 2 literals is  $m/2$ .

(b) (5 points)

By what method can you convert your randomized algorithm into a deterministic algorithm that always returns at least  $m/2$  clauses.

SOLUTION: This questions was phrased so that you only need say “by the method of conditional expectations”. A more complete answer would be to say how to choose the truth assignment to a variable based on the current set of unsatisfied clauses by choosing the assignment so as to maximize the remaining expectation when all unset variables are set uniformly at random.

2. (20 points)

Consider the following *2 from 4 frequency set cover problem*: We are given a collection of sets  $\mathcal{S} = \{S_1, \dots, S_m\}$  for  $S_i \subseteq U$  with the property that every  $u \in U$  occurs in exactly four different sets  $S_i$  in  $\mathcal{S}$ . There is also a (positive valued) cost function  $c : \mathcal{S} \rightarrow \mathbb{R}^+$  and we let  $c_i$  denote the cost of set  $S_i$ . A feasible solution is a sub-collection  $\mathcal{S}' \subseteq \mathcal{S}$  such that every  $u \in U$  occurs in at least two different sets  $S_i$  in  $\mathcal{S}'$ . The goal is to find a feasible solution  $\mathcal{S}'$  so as to minimize the cost  $c(\mathcal{S}') = \sum_{i:S_i \in \mathcal{S}'} c_i$ .

(a) (10 points) Formulate the *2 from 4 frequency set cover problem* as a  $\{0, 1\}$  IP

SOLUTION: Minimize  $\sum_i c_i \cdot x_i$  subject to  
 $\sum_{i:u \in S_i} x_i \geq 2$  for each  $u \in U$ .  
 $x_i \in \{0, 1\}$  for each  $i$

(b) (10 points) Show how to use LP relaxation + rounding to obtain a  $c$ -approximation algorithm for some constant  $c$ . What constant  $c$  can you obtain? Explain why your rounded solution is a feasible solution to the IP and why it provides a  $c$ -approximation.

SOLUTION: The LP is to replace  $x_i \in \{0, 1\}$  by  $0 \leq x_i \leq 1$ . (Note: in the LP, we do want to insist that  $x_i \leq 1$  since it is not clear that an optimal LP would never choose  $x_i > 1$  as is the case for the standard set cover problem.) Let  $\{x_i^*\}$  be an optimal LP solution. Set  $x'_i = 1$  if  $x_i^* \geq 1/3$ , and 0 otherwise. Claim 1:  $\{x'_i\}$  is an IP solution. Consider any  $u \in U$  and by renaming lets say that  $u$  occurs in  $S_1, S_2, S_3, S_4$ . Then we have the LP constraint:  
 $x_1^* + x_2^* + x_3^* + x_4^* \geq 2$ . We need to show that at least two of the  $x'_1, x'_2, x'_3, x'_4$  are set to 1 in the rounded solution. Suppose that at most one such  $x'_i$  is set to 1. Then at least three of the  $x_i^*$  are such that  $x_i^* < 1/3$  and hence  $x_1^* + x_2^* + x_3^* + x_4^* < 2$ .

Claim 2:  $\sum_i c_i \cdot x'_i \leq \sum_i c_i \cdot 3 \cdot x_i^* = 3 \cdot LP - OPT \leq 3OPT$ . This is immediate.

3. (20 points) Consider following knapsack decision problem:

An input item  $I_j = (s_j, v_j)$  where  $s_j$  is the size of the item and  $v_j$  is its value. All parameters are positive integers say represented in binary. Let  $L = \{(I_1, \dots, I_n, W, V): \exists \mathcal{I} \text{ such that } \sum_{I_j \in \mathcal{I}} s_j \leq W \text{ and } \sum_{I_j \in \mathcal{I}} v_j \geq V\}$ . Note that  $L$  is obviously in NP.

(a) (10 points)

Show how to transform the SUBSET-SUM problem to the knapsack decision problem thereby showing that the knapsack decision problem is NP-hard (and therefore also NP complete).

SOLUTION: Let  $(a_1, a_2, \dots, a_n; t)$  be an input instance  $x$  of the SUBSET-SUM problem. Transform  $x$  to an instance  $\phi(x)$  of the knapsack decision problem by setting  $s_j = v_j = a_j$  for  $1 \leq j \leq n$  and setting  $W = V = t$ . It is not difficult to see that  $x \in \text{SUBSET-SUM}$  iff  $\phi(x)$  is a yes instance for the knapsack decision problem.

(b) (10 points)

The knapsack optimization problem is to find an optimal subset  $\mathcal{I}$ . That is, given input  $\{I_1, \dots, I_n, W\}$ , the goal is to find a subset  $\mathcal{I} \subseteq \{I_1, \dots, I_n\}$  such that  $\sum_{I_j \in \mathcal{I}} s_j \leq W$  so as to maximize  $\sum_{I_j \in \mathcal{I}} v_j$ . Show how to polynomial time reduce the knapsack optimization problem to the knapsack decision problem. Hint: first find the optimum value of a feasible subset.

SOLUTION: As in the hint, use binary search and the knapsack decision problem to determine an optimal value  $V^*$ . We now need to construct an optimal solution  $\mathcal{I}$  by considering each input item to see if it is needed in an optimal solution. Lets just consider whether or not input item  $I_1$  is needed. We construct a new knapsack decision problem by setting  $W' = W - s_1$  and  $V' = V^* - v_1$  and remove  $I_1$  for the set of items. If there is no solution to this new knapsack decision problem then item  $I_1$  cannot be part of an optimal solution and it is removed and we continue by finding an optimal solution for the input  $\{I_2, \dots, I_n, W\}$  searching for an optimal solution having value  $V^*$ . Otherwise, item  $I_1$  is part of at least one optimal solution and we place in  $\mathcal{I}$ . We continue by finding an optimal solution for the input  $\{I_2, \dots, I_n, W - s_1\}$  searching for an optimal solution having value  $V^* - v_1$