Due: Wednesday, December 1, beginning of lecture

NOTE: Each problem set only counts 5% of your mark, but it is important to do your own work (but see below). These assignments will be followed by term tests, each worth 15% of your final grade. You may consult with others concerning the general approach for solving problems on assignments, but you must write up all solutions entirely on your own. As an experiment you may chose to work in pairs and then submit one assignment. Anything else is *plagiarism*, and is subject to the University's Code of Behavior. You will receive 1/5 points for any question/subquestion for which you say "I do not know how to answer this question". You will receive .5/5 points if you just leave the question blank.

1. Problem 9 on page 284 of text.

2. Problem 10 on page 284 of text.

3. Consider the simple knapsack problem where $v_i = w_i$ for all i. We know that there is an algorithm (using dynamic programming and rounding) that uses time $O(n^3/\epsilon)$ time and computes a solution within a factor $(1+\epsilon)$ of the optimal solution for the (general) knapsack problem. Now consider the simple knapsack problem where $v_i = w_i$ for all i. We want faster approximation algorithm for this problem.

   (a) Problem 10 on page 474 of the text provides a greedy algorithm for this problem. Prove that it does indeed always produce a solution within a factor of 2 of the optimal solution; that is, $Greedy(\mathcal{I}) \geq (1/2)OPT(\mathcal{I})$ for every input $\mathcal{I} = (w_1, w_2, \ldots, w_n; W)$. Here we assume $w_i \leq W$ for all i.

   (b) Find an $O(n \log n)$ time algorithm ALG such that $ALG(\mathcal{I}) \geq (2/3)OPT(\mathcal{I})$ for every input $\mathcal{I}$ as claimed in part (c) of problem 10.
   Hint: Partition the inputs into three sets, those with weight $w_i \leq W/3$, those with weight $W/3 < w_i \leq (2/3)W$ and those with weight $w_i > (2/3)W$. Your algorithm should be "greedy-like" as will be explained in class.

4. Consider the set cover problem as defined in Chapter 11. Now suppose that we restrict attention to those inputs (which we will call *frequency f instances*) where each universe element occurs in at most f sets.

   (a) Show that the vertex cover problem can be viewed as a a set cover problem where every input is a frequency 2 instance.

   (b) Show how to represent the set cover problem as an integer programming problem. Then show how to use linear programming and rounding to derive a factor f approximation algorithm for every frequency f instance of the set cover problem.

5. Problem 1 on page 549 of the text.

6. Consider the edge weighted version of the global min cut problem where every edge e has an integer capacity $c(e)$.

(a) Suggest how to utilize or generalize the randomized algorithm for the unweighted gloabl min cut (as in section 13.2 of the text) so as to solve the global min capacity cut problem.

(b) What is the running time needed for your algorithm to produce a global min capacity cut with probability at least .5?

(c) Provide a justification for your probability estimate.

7. Problem 2 on pages 502,503 of text.