Homework #9: Approximation Algorithms
Due Date: Thursday, April 30, 1998 (5:00pm)

Collaboration Policy

You may collaborate with others on this entire homework. However, please re-read the policy guidelines from Handout #1.

Reading

You should read all of Chapter 37 of CLR, although Section 37.4 will be the most directly related part for this particular homework.

Practice

Recall, these exercises are purely for your own practice. You need only turn in the official problems.

- Do CLR 37.1-1, 37.1-2, 37.1-3.
- Do CLR 37.3-1.
- Do CLR 37-2.

Problems

The entire homework this week will involve the Knapsack problem. The problem is defined on page 335 of the text, as follows. "A thief robbing a store finds n items; the ith item is worth v_i dollars, and weighs w_i pounds, where v_i and w_i are (positive) integers. He wants to take as valuable a load as possible, but he can only carry at most W pounds in his knapsack for some integer W. What items should he take?" Without loss of generality, we assume that all objects have weight $w_i \leq W$ (or else the thief could not take that object ever). Assume that the maximum individual value of an objects is equal to $v_{max} = \max_i v_i$.

Formally, the goal is to pick a subset $S \subseteq \{1, \ldots, n\}$, satisfying the inequality $\sum_{i \in S} w_i \leq W$, while maximizing the value $\sum_{i \in S} v_i$. We let S^* be the actual set of objects in the optimal solution to the problem, and we let $V^* = \sum_{i \in S^*} v_i$ denote the value of the optimum solution.

a. (10 points)
   Prove that no polynomial algorithm can calculate V^* unless $\mathtt{P} = \mathtt{NP}$. (*Hint:* notice the relationship between the `Subset-Sum` problem and this).

b. (20 points)
Show that the value V^* can be calculated using dynamic programming, in $O(nV^*)$ time. (*Hint:* Consider all feasible ways for the thief to get total value v by using a subset of the first i items. Let $W(i, v)$ be the weight of the lightest such set, if one exists.)

c. (5 points)
Can we conclude that $\mathtt{P} = \mathtt{NP}$, based on the above two parts? Explain why or why not.

By keeping associated data, the algorithm you (hopefully) gave in part b can be used to calculate the actual set S^* in addition to V^*. We will use this presumed algorithm in order to design an arbitrarily good approximation algorithm for the Knapsack problem.

Given the original instance $\mathcal{I}$ of the Knapsack problem, we create a *scaled* instance $\mathcal{I}'$ by leaving the weights of the objects alone, but scaling the values down so that $v_i' = \lfloor \frac{v_i}{k} \rfloor$ for some constant k (we will soon choose an actual value for k). Now we use the algorithm from part b to find the optimal set of objects S' for $\mathcal{I}'$, with value $V' = \sum_{i \in S'} v_i'$. We return this same set S' of objects as our solution to $\mathcal{I}$, and so our solution to the *original* problem will have value $\hat{V} = \sum_{i \in S'} v_i$.

d. (10 points)
Prove that

$$v_{max} \leq V^* \leq n \cdot v_{max}, \tag{1}$$

$$v'_{max} \leq V' \leq n \cdot v'_{max}. \tag{2}$$

e. (5 points)
Argue that

$$\sum_{i \in S'} \lfloor \frac{v_i}{k} \rfloor \geq \sum_{i \in S^*} \lfloor \frac{v_i}{k} \rfloor \tag{3}$$

f. (20 points)
Prove that

$$\hat{V} \geq V^* - k|S^*|. \tag{4}$$

(*Hint:* Use the tautology $x - 1 < \lfloor x \rfloor \leq x$.)

g. (15 points)
Argue that the *relative error bound* (defined as Equation 37.2 of CLR), satisfies the following:

$$\frac{V^* - \hat{V}}{V^*} \leq \frac{k|S^*|}{v_{max}} \leq \frac{nk}{v_{max}}. \tag{5}$$

h. (15 points)
Given a specific value $\epsilon > 0$, choose $k = \frac{\epsilon \cdot v_{max}}{n}$. Show that the relative error bound from part g is at most ϵ, and that the running time is $O(n^3 \frac{1}{\epsilon})$ time.

Extra Credit

Do CLR 37-3.