CS5050 Advanced Algorithms

Fall Semester, 2015

Assignment 5: Dynamic Programming

Due Date: Friday, Nov. 13, 2015 (**at the beginning of CS5050 class**)

1. In class, we have studied the following knapsack problem. We are given n items of sizes $a_1, a_2, \ldots, a_n$, which are positive integers. We are also given a knapsack of size M, which is also a positive integer. We want to determine whether there is a subset S of the items such that the sum of the sizes of the items in S is exactly M. If there exists such a subset S, we call S a *feasible subset*.

   In fact, it is possible that there are multiple feasible subsets. In this exercise, you are asked to design an $O(nM)$ time dynamic programming algorithm to compute the *number* of feasible subsets.

   For example, suppose the item sizes are $3, 5, 6, 2, 7$, and $M = 13$. Then, there are two feasible subsets $\{6, 7\}$ and $\{2, 5, 6\}$. Thus your algorithm should return 2.

   (20 points)

2. This is a problem from a job interview and the problem is closely related to the knapsack problem we studied in class. I got the problem from Vahe, a student from our class. Vahe got the problem from his friend, who got the problem during his interview with Goldman Sachs in Salt Lake City.

   Given a set A of n positive integers $\{a_1, a_2, \ldots, a_n\}$ and another positive integer M, find a subset of numbers of A whose sum is closest to M. In other words, find a subset A' of A such that the absolute value $|M - \sum_{a \in A'} a|$ is minimized, where $\sum_{a \in A'} a$ is the total sum of the numbers of A'. For the sake of simplicity, you only need to return the sum of the elements of the solution subset A' without reporting the actual subset A'.

   For example, suppose $A = \{1, 4, 7, 12\}$ and $M = 15$. Then, the solution subset is $A' = \{4, 12\}$, and thus your algorithm only needs to return $4 + 12 = 16$ as the answer.

   Design a dynamic programming algorithm for the problem and your algorithm should run in $O(nK)$ time in the worst case, where K is the sum of all numbers of A.

   (20 points)

3. Here is a more common variation of the knapsack problem. We are given n items of sizes $a_1, a_2, \ldots, a_n$, which are positive integers. Further, for each $1 \le i \le n$, the i-th item a_i has a positive value $value(a_i)$ (you may consider $value(a_i)$ as the amount of dollars the item is worth). The knapsack size is a positive integer M.

   Now the goal is to find a subset S of items such that the sum of the sizes of all items in S is **at most** M (i.e., $\sum_{a_i \in S} a_i \le M$) and the sum of the values of all items in S is **maximized** (i.e., $\sum_{a_i \in S} value(a_i)$ is maximized).

Design an $O(nM)$ time dynamic programming algorithm for the problem. For simplicity, you only need to report the sum of the values of all items in the optimal solution subset S and you do not need to report the actual subset S.

(20 points)

4. Given an array $A[1 \dots n]$ of n distinct numbers, design an $O(n^2)$ time dynamic programming algorithm to find a longest monotonically increasing subsequence of A. Your algorithm needs to report not only the length but also the actual longest subsequence (i.e., report all elements in the subsequence).

   Here is a formal definition of a *longest monotonically increasing subsequence of* A (refer to the following example). First of all, a *subsequence* of A is a subset of numbers of A such that if a number a appears in front of another number b in the subsequence, then a is also in front of b in A. Next, a subsequence of A is *monotonically increasing* if for any two numbers a and b such that a appears in front of b in the subsequence, a is smaller than b. Finally, a *longest monotonically increasing subsequence of* A refers to a monotonically increasing subsequence of A that is longest (i.e., has the maximum number of elements).

   For example, if $A = \{20, 5, 14, 8, 10, 3, 12, 7, 16\}$, then a longest monotonically increasing subsequence is $5, 8, 10, 12, 16$. Note that the answer may not be unique, in which case you only need to report one such longest subsequence.

(20 points)

Total Points: 80