CIS 102: Fall 2001

Homework Assignment # 5

Dynamic Programming

Part A

1. **Matrix Chain Multiplication Problem:**
   This problem and the solution has been discussed in the textbook and in the class. A succint description and the associated pseudo-code has been distributed in the class as well under the heading 8.1 (Iterated Matrix Product).

   Please be aware that there are two problems. The first problem is to find the order in which to multiply the matrices to minimize the number of multiplications. The second problem is to find the least number of multiplications.

   In this problem you are to design a dynamic programming algorithm for both the problems. Do the following for the first problem and then for the second problem. Describe the table and what does each entry in the table mean? How will the table be initialized? In which order the table will be filled? What is the recurrence? How will you use the table to find the order of multiplications (for the first problem) and the actual number of multiplications (for the second problem)? Compute the asymptotic complexity of the algorithms. It is very important that you practice writing your own solutions to this problem even though you may have perfect understanding of the solution.

   Practice the solution to the above problem when there are 4 matrices with order 2×5, $5x4$, $4x1$, and $1x10$. (In actual quiz, these numbers may vary).

2. **0-1 Knapsack Problem:**
   Consider the 0-1 knapsack problem: 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 integers. For every item, the thief has to make a binary choice: whether to take the item or leave it. He wants to take as valuable a load as possible, but he can carry at most W pounds in his knapsack for some integer W.

   There are two problems. First, what is the value of the most valuable load he can steal? Second, what items he should take?

   You are to design a dynamic programming algorithm for both the problems. Describe the table and what does each entry in the table mean? How will the table be initialized? In which order the table will be filled? What is the recurrence? How will you use the table to find what is the total value of the goods stolen (for the first problem) and which goods should be stolen (for the second problem)? Compute the asymptotic complexity of the algorithms. It is very important that you practice writing your own solutions to this problem even though you may have perfect understanding of the solution.

Pseudo-code solution to this problem has been provided in a hard copy handout under 8.2 (The knapsack Problem).

Practice the above problem when you are stealing 4 items worth 2, 4, 5, and 3 dollars and weighing 1, 2, 3 and 1 lbs. and the weight of the thief's bag is 4 lbs. (In actual quiz, these numbers may vary).

3. **Coin Changing Problem:**
Given the k coin values $c_0 < c_1 < c_2 < \cdots < c_{k-1}$ (where $c_0 = 1$), and a value v, find a way to give the value v in change using as few coins as possible (sufficient coins of each denomination are available).

   There are two problems. First, what is the minimum number of coins required for the change? Second, how many coins of each type will be given for this change?

   You are to design a dynamic programming algorithm for both the problems. Describe the table and what does each entry in the table mean? How will the table be initialized? In which order the table will be filled? What is the recurrence? How will you use the table to find the minimum number of coins (for the first problem) and how many coins of each type (for the second problem)? Compute the asymptotic complexity of the algorithms. It is very important that you practice writing your own solutions to this problem even though you may have perfect understanding of the solution.

   Practice the above problem with coin denominations 1, 3, and 4 seeking change for 6. (In actual quiz, these numbers may vary).

4. **Canoe Rental Problem:**
There are n trading posts numbered 1 to n as you travel downstream. At any trading post i you can rent a canoe to be returned at any of the downstream trading posts $j > i$. You are given a cost array $R(i, j)$ giving the cost of these rentals for all $1 \leq i < j \leq n$. We can assume that $R(i, i) = 0$, and that you can't go upriver (so perhaps $R(i, j) = \infty$ if $i > j$). For example, one cost array with $n = 4$ might be the following.

C		to j			
		1	2	3	4
	1	0	2	3	7
from	2	–	0	2	4
i	3	–	–	0	2
	4	–	–	–	0

   The problem is to find a dynamic programming algorithm that computes the cheapest sequence of rentals taking you from post 1 all the way down to post n. In this example, the cheapest way is to rent canoes from post 1 to post 3, and then from post 3 to post 4 for a total cost of 5. The second problem is to find the least cost associated with this sequence.

You are to design a dynamic programming algorithm for both the problems. Describe the table and what does each entry in the table mean? How will the table be initialized? In which order the table will be filled? What is the recurrence? How will you use the table to find what is the cheapest sequence of canoe rentals (for the first problem) and the least cost of the canoe rentals (for the second problem)? Compute the asymptotic complexity of the algorithms. It is very important that you practice writing your own solutions to this problem even though you may have perfect understanding of the solution.

Solution to this problem has been provided on the web.

Practice the above problem with numbers given in the table. (In actual quiz, these numbers may vary).

Part B

1. **Optimal Binary Search Trees:**
   The problem and the solution is described in a hard copy handout titled 8.3 (Optimal Binary Search Trees).

2. **Partition problem**:

   > Given a sequence n positive integers, $k_1, k_2, \ldots, k_n$, that sum to s (you can assume that s is even), find a subset I of $\{1, \ldots, n\}$ such that
   > $$\sum_{i \in I} k_i = \sum_{i \notin I} k_i = s/2$$
   > or determine that there is no such subset.

   You are to find a dynamic programming solution to this problem. If a subset exists, you need to find the integers in the subset I.

Remark: I expect you to understand this material by November 8. There will be a quiz from Part A with numbers. Students desirous of getting a grade of B- or better must take an additional quiz which describes the solution from problems in Part A (Yes, I do mean Part A).

Solutions to Part B will be posted. More ambitious students can attempt to write their own solutions to Part B problems. If this does not suffice your appetite for excellence, please come and talk to me and I will assign additional dynamic programming problems. If you write any of these solutions, please ensure that you attach them to your portfolio at the end of the quarter.