CSE 2012- Design and Analysis of Algorithms Practice Problem Sheet 3 (Dynamic Programming) Matrix-chain Multiplication

Practice makes you Perfect

Matrix-chain Multiplication Problem (MCM):
Given a chain of matrices $< A_1, A_2, ..., A_n >$ of n matrices, A_i is a matrix of size$p_{i-1} \times p_i$, $i = 1, 2..., n$, task is to fully parenthesize the product $A_1 * A_2 *, ... * A_n$ (to compute the product of all the given matrices in the same order as given) in such a way that the number of scalar multiplications (the number of multiplications between the entries of the given matrices) performed in computing the product $A_1 * A_2 *, ... * A_n$ is minimum.

1. A product of matrices $A_1 * A_2 *, ... * A_n$ is fully parenthesized if it is either a single matrix or the product of two fully parenthesized matrix products, surrounded by parentheses. For example, if the chain of matrices is $< A_1, A_2, A_3, A_4 >$, then we can fully parenthesize in five distinct ways:

   $$(A_1(A_2(A_3A_4))), \quad (A_1((A_2A_3)A_4)), \quad ((A_1A_2)(A_3A_4)),$$

   $$((A_1(A_2A_3)A_4)), \quad (((A_1A_2)A3)A_4)$$

   Given a chain of matrices $< A_1, A_2, ..., A_n >$ of n matrices, A_i is a matrix of size$p_{i-1} \times p_i$, $i = 1, 2..., n$, design a pseudocode (in brute-force approach) to compute the total number of ways of fully parenthesizing $A_1 * A_2 *, ... * A_n$. For each full parenthsization, your code should compute the number of scalar multiplications involved in that full parenthesization and print the corresponding full parenthesization. Analyse the pseudoocde with all the components required for the analysis.

2. Design a recursive pseudocode to output the full parenthesization X of $< A_1, A_2, ..., A_n >$ such that the total number of scalar multiplications is minimum in X among all the possible full parenthesizations of $< A_1, A_2, ..., A_n >$. Your pseudocode should compute the full parenthesiazation which has minimum scalar multiplications along with the minimum value of the scalar multiplications. Analyse the pseudoocde with all the components required for the analysis.

3. Design a pseudocode that uses the ‘dynamic programming strategy’ to solve the matrix chain multiplication problem as described above. Your pseudocode shold print the full parenthesization of an optimal solution that involves minimum number of scalar multiplications. Your pseudocode should use ‘bottom-up’ approach to compute the value of an optimal solution. Your code should have all the components of the dynamic programming : structuring the subproblems, recursive calculation of the value of an optimal solution, computing the value of an optimal solution and computation of the optimal solution, should be clearly segregarted in your code with appropriate comment statements. Analyse the pseudoocde with all the components required for the analysis.

4. Q. no 3, computes the value of an optimal solution in the bottom-up approach. Design a pseudocode for solving the matrix-chain problem such that the pseudocode computes the optimal solution in a top-down fashion. You are expected to design the pseuddocode by modifying the recursive code (for matrix-chain problem) with the memoization process. Compare the dynamic programming code of MCM (top-down approach) and the dynamic programming code of MCM (bottom-up approach) and conclude which approach is better, for the given n matrices.

5. In solving the matrix-chain multiplication problem, the pseudocode returns a full parenthesisation X such that X involves minimum scalar multiplications in computing the product $A_1 * A_2 *, ... * A_n$. Modify the pseudocode in such a way that it returns a full parenthesisation Y such that Y involves maximum scalar multiplications in computing the product $A_1 * A_2 *, ... * A_n$. Analyse the modified pseudoocde with all the components required for the analysis.

6. The dynamic programming pseudocode for MCM returns an optimal solution along with the optimal value (that is, the code returns a full parenthesization along with the minumum number of scalar multiplications). Modify the dynamic programming pseudocodecode of MCM in such a way that the modified code returns all the possible full parenthesizations that involves minimum number of scalar multiplications in computing the product $A_1 * A_2 *, ... * A_n$.

7. Two subproblems of the same problem are independent if they do not share the same resources for solving the subproblems. Two subproblems are said to overlap if they share the same resources for solving the subproblems, The dynamic programming code for MCM involves subproblems which are over-lapping. Modify the code in such a way the modified code for MCM considers only the subproblems which are independent. Observe your modified code and conclude whether the modified code exhibits the divide-conquer- combine strategy. If possible, identify atleast one sequence of matrices $A_1, A_2, ..., A_n >$ which when given as an input to the pseudocode of MCM and the modified pseducode MCM, returns the same optimal value.

8. Consider a matrix-chain of four matrices $< A_1, A_2, A_3, A_4 >$. With the chosen matrix-chain, illustrate the functionality of dynamic-programming pseudocde (bottom-up approach) and dynamic programming pseudocde (top-down approach). You are expected to illustare each line of both the pseudocodes with the matrix-chain chosen by you. Based on the marix-chain chosen by you, conclude which pseudocode returns the optimal value fast.

9. Design a pseudocode to output any two full parenthesizations (say), X_1, X_2 of $< A_1, A_2, ..., A_n >$ such that the total number of scalar multiplications involved in X_1 and X_2 are same. If the given matrix-chain does not have two parenthesisations with same number of scalar multiplications, your pseudocode should retrun 0.

10. You have designed many pseudocodes for computing the minimum number of scalar multiplications in a matrix-chain multiplication problem. With that designing experience, identify a matrix-chain of n square matrices such that the minimum number of scalar multiplications in the matrix-chain multiplication is a multiple of the dimension (number of rows or the number of columns) of the matrices involved in the chain. Your answer should be supported with a proper justification.