4005-800 Algorithms

Homework 5

Christopher Wood

April 27, 2012

PROBLEM 1-a.

Solution.

To find the optimal parenthesization of a matrix chain product whose sequence of dimensions is $< 5, 10, 3, 12, 5, 50, 6 >$, we simply use the *minMuls* and *genParens* functions to find the optimal number of multiplications and then insert the right parentheses, respectively. The steps of the mulMuls algorithm is shown below.

0	-	-	-	-	-
-	0	-	-	-	-
-	-	0	-	-	-
-	-	-	0	-	-
-	-	-	-	0	-
-	-	-	-	-	0

-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-

Table 1: m and c tables for $l = 1$ (base case)

0	150	-	-	-	-
-	0	360	-	-	-
-	-	0	180	-	-
-	-	-	0	3000	-
-	-	-	-	0	1500
-	-	-	-	-	0

-	1	-	-	-	-
-	-	2	-	-	-
-	-	-	3	-	-
-	-	-	-	4	-
-	-	-	-	-	5
-	-	-	-	-	-

Table 2: m and c tables for $l = 2$

0	150	-	-	-	-
-	0	360	-	-	-
-	-	0	180	-	-
-	-	-	0	3000	-
-	-	-	-	0	1500
-	-	-	-	-	0

-	1	-	-	-	-
-	-	2	-	-	-
-	-	-	3	-	-
-	-	-	-	4	-
-	-	-	-	-	5
-	-	-	-	-	-

Table 3: m and c tables for $l = 3$

PROBLEM 1-b. *Show that a fully parenthesization of an n-element expression has exactly $n - 1$ pairs of parentheses.*

Solution. We can prove this fact using induction on the number of matrices in a matrix chain.
Base #1: $n = 1$
By definition, an expression is fully parenthesized if it is a single element. Therefore, since $n = 1$ corresponds to an expression of a single element (a single matrix), then we know it is fully parenthesized with no parentheses. Thus, we have $(1 - 1) = 0$ parentheses for a $n = 1$ element expression.
Base #2: $n = 2$
By definition, a 2-element expression is fully parenthesized if it is the product of the two fully parenthesized elements surrounded by a single pair of parenthesis. Since single elements are fully parenthesized by themselves with no addition parenthesis, we know that an 2-element expression is fully parenthesized if we write it as the product of the two elements surrounded by parentheses. Thus, with this 2-element expression, we can make it fully parenthesized with $2 - 1 = 1$ pair of parentheses.
Induction: $n = 2$
Assume that a full parenthesization of a k-element expression has exactly $k - 1$ pairs of parentheses. Now, let $[A_1, A_2, ..., A_k]A_{k+1}$ be a $k + 1$-element expression. By the induction hypothesis, we know that $[A_1, A_2, ..., A_k]$ is fully parenthesized with $k - 1$ parentheses. Now, since A_{k+1} is a single matrix we know that it is also fully parenthesized with 0 parentheses, so we can make the full expression $[A_1, A_2, ..., A_k]A_{k+1}$ by rewriting it as $([A_1, A_2, ..., A_k]A_{k+1})$ (since it is the product of two fully parenthesized expressions). Now, since $[A_1, A_2, ..., A_k]$ contributed $k - 1$ parentheses and we have just added one more pair of parentheses, the total is now k, which is equal to exactly $(k+1)-1$.

Thus, we can see that a fully parenthesization of an n-element expression has exactly $n - 1$ pairs of parentheses. This can also be argued by observing that a pair of parentheses always wraps two operands with a single operator, and since there are $n - 1$ operators in an n-element expression, there must also be $n - 1$ parentheses if it is fully parenthesized.

PROBLEM 2-a. *Which is a more efficient way to determine the optimal number of multiplications in a matrix-chain multiplication problem: enumerating all the ways of parenthesizing the product and computing the number of multiplications for each, or running the recursive matrix chain algorithm?*

Solution.