Analysis of Algorithms

Homework 2

Abraham Murciano

November 22, 2020

1 Matrix Multiplication

We are given the following four matrices to be multiplied, as well as their sizes.

Matrix	Size
$\mathbf{A}_1$	10×30
$\mathbf{A}_2$	30×5
$\mathbf{A}_3$	5×60
$\mathbf{A}_4$	60×10

We are to apply the algorithm described in class in order to find the order in which to apply the associative matrix multiplications such that the number of scalar multiplications is minimal.

When multiplying two matrices, suppose these are of sizes $r \times s$ and $s \times t$, the resulting matrix would be of size $r \times t$, meaning $r \cdot t$ dot products must be calculated. And each of those dot products would be a sum of s scalar multiplications. Therefore the total number of scalar multiplications for these two matrices is $r \cdot s \cdot t$.

When multiplying a chain of n matrixes, we can say that whatever the optimal order of performing the multiplications, if the last two matrices to be multiplied are

$$(\mathbf{A}_1 \times \cdots \times \mathbf{A}_k) \times (\mathbf{A}_{k+1} \times \cdots \times \mathbf{A}_n)$$

then the way to multiply $\mathbf{A}_1 \times \cdots \times \mathbf{A}_k$ must be optimal, and the same can be said about $\mathbf{A}_{k+1} \times \cdots \times \mathbf{A}_n$.

We denote by $m_{i,j}$ the minimal number of scalar multiplications required to multiply matrices $\mathbf{A}_i$ through $\mathbf{A}_j$, where the dimensions of matrix $\mathbf{A}_i$ is denoted by $d_i \times d_{i+1}$. Now we can define $m_{i,j}$.

$$m_{i,j} = \begin{cases} 0 & i = j \\ \min\limits_{i \le k < j} (m_{i,k} + m_{k+1,j} + d_i \cdot d_{k+1} \cdot d_{j+1}) & i < j \end{cases}$$

Let $s_{i,j}$ be the value of k which gives the minimal number of multiplications in the definition of $m_{i,j}$. Now we are ready to compute the initial question by calculating $m_{1,4}$, $s_{1,4}$, and all the intermediate results, as shown in table 1.

$i \backslash j$	1	2	3	4
1	0	1500	4500	5000
2	-	0	9000	4500
3	-	-	0	3000
4	-	-	-	0

$i \backslash j$	1	2	3	4
1	-	1	2	2
2	-	-	2	2
3	-	-	-	3
4	-	-	-	-

Table 1: The computed final and intermediate results of $m_{1,4}$ (left) and $s_{1,4}$ (right)

Thus we have shown that the optimal way to multiply $\mathbf{A}_1 \times \mathbf{A}_2 \times \mathbf{A}_3 \times \mathbf{A}_4$ is $((\mathbf{A}_1)(\mathbf{A}_2))((\mathbf{A}_3)(\mathbf{A}_4))$ with a total of 5000 scalar multiplications.

2 Longest Non-Decreasing Subsequence

The problem. Given a sequence of natural numbers $S = (s_1, s_2, \ldots, s_n)$, we are to create an algorithm to find the longest monotonically non-decreasing subsequence as well as its length.

The optimal substructure. Suppose T is a solution; i.e. it is a non-decreasing subsequence of S of maximal length. Suppose T' is a prefix of T such that $|T'| = m$ and the last value of T' is s_a.

We can be certain that T' is the longest non-decreasing subsequence of the prefix of S of length a (i.e. the first a terms of S), such that the last element of the subsequence is less than or equal to t_{m+1}.

The proof. Suppose T' is not the longest subsequence within the conditions specified earlier. Meaning there exists U which is a longer subsequence of the prefix of S of length a. Then we would be able to substitute the prefix T' for the U in the subsequence T, and that would give a longer subsequence of S, contradicting the maximality of T.

The formula.

$$T_{a,}$$