Lecture 11

Gidon Rosalki

2024-12-04

1 Multiplying matrices problem

Example 1 (Multiplying 2 matrices)**.** *Input: $n+1$ natural numbers $p_0, \dots, p_n$, representative of the dimensions of the n matrices $A_1, \dots, A_n$, where A_i is of dimensions $p_{i-1} \times p_i$*
Output: Order of multiplying the matrices (inserting brackets) to calculate $B = A_1 \dots A_n$ at the lowest number of calculations.

Solution. We will use $P[1, k]$ to be the optimal price for the calculation of $B_{1,k} = A_1 A_2 \dots A_k$, and $P[k+1, n]$ to be the optimal price for the calculation of $B_{k+1,n} = A_{k+1} \dots A_n$. Therefore the price of the overall problem is $P[1, n]$, and the first recursive formula is $P[1, n] = \min_{1 \leq k \leq n-1} \{P[1, k] + P[k+1, n] + p_0 p_k p_n\}$.
When we continue to split, we will notice that every sub problem we see is of the following style: We will calculate the minimal price, the multiplication $1 \leq i \leq j \leq n$ $B_{i,j} = A_i A_{i+1} \dots A_j$, of which there are $O\left(n^2\right) = \binom{n}{2} + n$. We will also write $P[i, j]$ to be the optimal price of the calculation $B_{i,j}$.
The general recursive formula:

$$P[i, j] = \begin{cases} 0, & \text{if } i = j \\ \min\limits_{i \leq k < j} \{P[i, l] + P[k+1, j] + p_{i-1} p_k p_j\}, & \text{otherwise} \end{cases}$$

□

n	0	2	-
j	0	-	-
1	-	-	-
	1	i	n

Table 1:

Algorithm. We are only interested in the upper left triangle (including the diagonal), since below that is the same work.

We will define the table T of size $n \times n$. We want to write for every $1 \leq i \leq j \leq n$ in the cell $T[i, j]$ the price $P[i, j]$.

1. Initialisation: We will define $\forall i \in [n]$ $T[i, i] = 0$
2. Iteration: We will fill the table in $(n-1)$ iterations, where in the iteration $1 \leq d \leq n-1$ we will fill the cells $T[i, j]$, where $j - i = d$ according to the recursive formula

$$T[i, j] = \min_{1 \leq k \leq j-1} \{T[i, k] + T[k+1, j] + p_{i-1} p_k p_j\}$$

3. Ending: We return $T[1, n]$

Runtime: We will show that filling each cell is $O(n)$. It is sufficient to see that in the filling of the cell $T[i, j]$ (for every $1 \leq i < j \leq n$), the cells that affect them in the recursive formula have already been filled in the previous iteration. It is true that when $k < j$ that $k - i < j - i = d$. Additionally, $j - (k+1) < j - i = d$. There are in total $O\left(n^2\right)$ cells, so therefore the overall runtime will be $O\left(n^3\right)$.

In order to return the correct distribution of brackets, we will save during the filling of each cell the choice of k which enables the minimum in the recursive formula. □