PS 703106
Exercise 11

Group 4:
- Jonas Boutelhik
- Michael Thöni
- Thomas Urban

Hardware:
Intel i7-4710HQ was used for all calulations (Sequential and openMP)
Processor Base Frequency: 2.50 GHz (Quad-Core)

Problem:
Let A_1 , ... , A_n be a number of matrices of which the product $A_1 \times ... \times A_n$ is to be computed. The size of these matrices is given by l_1 , ... , l_{n+1} , with matrix Ai having size $l_i \times l_{i+1}$. Which parenthesis grouping minimizes computational costs assuming a standard matrix multiplication algorithm? There are exponentially many ways of grouping the product terms, rendering an exhaustive search for a larger number of matrices infeasible or impossible. Fortunately, this problem can be solved in polynomial time using dynamic programming.

Algorithm:
The minimal cost $c_{i,j}$ for computing the sub-product $A_i \times ... \times A_j$ can be computed as

$$c_{i,j} = \begin{cases} 0 & i = j \\ \min\limits_{i \le k < j} \left(c_{i,k} + c_{k+1,j} + l_i l_{k+1} l_{j+1} \right) & i < j \end{cases}$$

Hence, for every sub-product $A_i \times ... \times A_j$, the respective k that minimizes the computation

$$(A_i \times ... \times A_k) \times (A_{k+1} \times ... \times A_j)$$

is to be found.

Assignment:
•Sketch the contents of matrix C for n = 5 and illustrate the data dependencies for computing $c_{2,2}$, $c_{1,3}$, and $c_{0,4}$. Can you identify a general pattern? What does it look like?

The dependencies are shown in Fig. 1 and are illustrated as colored triangels.
- $c_{2,2}$ has no dependencies because i = j and so c2,2 = 0 what results in distance d = 0.
- $c_{1,3}$ has a distance of d = 2. So it depends from $c_{1,1}$, $c_{2,2}$, $c_{3,3}$, $c_{1,2}$ and $c_{2,3}$ which is illustrated by the dark blue triangle.
- $c_{0,4}$ has a distance of d = 4. So it depends from all red nodes which are part of the yellow triangle.

→ it looks like a tree-structure with the nodes $c_{2,2}$, $c_{1,3}$ and $c_{0,4}$ as a root.

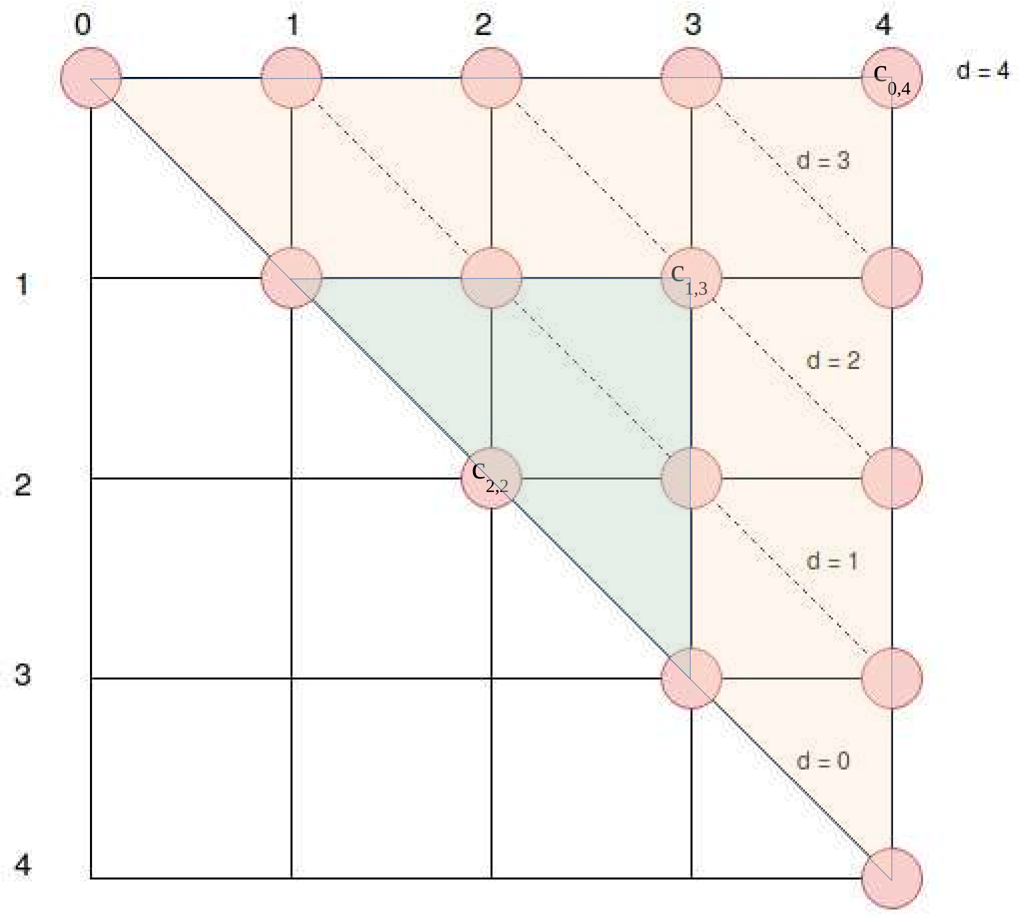

Figure 1: Calculation of minimal cost of matrix $c_{i,j}$ for computing the sub-product $A_i \times \ldots \times A_j$ with their depenencies and d = distance.

•Parallelize the given sequential implementation using OpenMP. Name this implementation dynamic_programming_omp.c.

•Implement a tiled / blocked variant using OpenMP. Ensure that the block size is a compile-time parameter and not hard-coded. Name this implementation dynamic_programming_blocked_omp.c.

Implementation	Minimal costs [FLOPS]	Total processing time [s]
SEQ	4048665	5,191
OMP	4048665	3,583
BLOCK_OMP	2340	0,129

Table 1: Program runtime for sequential, openMP and blocked openMP whereas min. costs = 4048665 FLOPS if the program terminates correct.