Assignments 11 and 12, due June 14th and 21st 2019 Project "Wave"

Problem:

Let $A_1, \ldots, A_n$ be a number of matrices of which the product $A_1 \times \ldots \times A_n$ is to be computed. The size of these matrices is given by $l_1, \ldots, l_{n+1}$, with matrix A_i 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 $A_i \times \ldots \times A_j$ can be computed as

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

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

$$(A_i \times \ldots \times A_k) \times (A_i \times \ldots \times A_k)$$

is to be found.

Resources:

- `week_12/dynamic_programming`
  contains a sequential implementation computing the minimal costs

First assignment (until June 14th)

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?
- 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`.

Solution upload:

- Full source code
- Via e-mail to philipp.gschwandtner@uibk.ac.at – one submission per group only!
  Subject: "`[PS703106] [AS11] GR_## - NAME1, NAME2, NAME3`"
  Solution must be submitted before Friday June 14th, 09:15!

Second assignment (until June 21st)

Assignment:

- Develop an OpenCL implementation exploiting the wave front pattern identified in the first assignment
- Evaluate whether using an out-of-order queue improves performance on your system

Hints:

- Base your OpenCL solution on the tiled / blocked OpenMP version (sample solution will be provided on June 14th)
- Compute every tile / block with an individual kernel call with a single work group
- Synchronize using event dependencies

Solution upload:

- Full source code and performance discussion + illustration
- Via e-mail to philipp.gschwandtner@uibk.ac.at – one submission per group only!
  Subject: "`[PS703106] [AS12] GR_## - NAME1, NAME2, NAME3`"
  Solution must be submitted before Friday June 21st, 09:15!