

## High Performance Computing Lab (CS 530):

**Lab 2:** Implement the power of a matrix. Given a large square matrix A (with a dimensionality M), and a number N, implement the power operation  $A^N$ , and assess the running complexity of the algorithm. The matrix multiplication algorithm design and implementation should consider threads in order to parallelize the matrix multiplication.

### Overview

Matrix multiplication is an easy task to accomplish, however, the complexity of the multiplication is polynomial, and for large matrices such an operation can take a long time. Due to the fact that the calculation of each element in the product matrix can be calculated separately, for each operation (element/column/row) one separate thread should be considered.

### Instructions

- Given a square matrix A, and a positive number N, the  $N^{\text{th}}$  power of the matrix A is defined as follows:  $A^N = A^{N-1} \times A$ .
- The dimensionality M of the matrix A should be provided from the keyboard.
- The N parameter should also be provided from the keyboard.
- The program should work for large matrices, therefore dynamic memory allocation should be considered. For extremely large matrices list structures should be considered.
- Due to the size of the matrix, the elements should be populated using random numbers or with an option the identity matrix should be available.
- Calculate the time complexity of the operation.
- Plot a complexity graph for different M sizes and different N values. All kind of plots are accepted (Matlab, R, Excel, etc.). Ex.  $M = \{1, \dots, 1000\}$   $N = \{0, \dots, 10000\}$
- A thread pool with available threads should be created in the beginning.
- Each thread should be responsible to calculate one element in the product matrix.
- Different results should be provided if
  - One element ( $a_{ij}$ ) in the product matrix is calculated by a single thread.
  - One row ( $a_{i,1..N}$ ) in the product matrix is calculated by a single thread.
  - One column ( $a_{1..N,j}$ ) in the product matrix is calculated by a single thread.

### Notes

- Error handling is expected.
- The program should work for large matrices.
- The program should be written in C/C++.
- No special matrix related library should be considered.
- Already existing data structures can be considered to model the matrix.
- Start multiple jobs, and see how the complexity changes.
- The two plots should be attached with the source code.

- Compare different time functions: gettimeofday(), time(), clock(), etc.
- The threads creation time should not be considered. They should be created at the beginning. In the iterations of the matrix power, the same threads should be used, and synchronization is to be expected.

## Rubric

Task	Points
Error handling	1
$A^N$ for large M and N values with thread pool and threads	6
2 plots (images)	2
Comparison of different time functions	1