Laborator 2 PPD

* **Goal**

Divide a simple task between threads. The task can easily be divided in sub-tasks requiring no cooperation at all. See the effects of false sharing, and the costs of creating threads and of switching between threads.

Requirement: write two problems: one for computing the sum of two matrices, the other for computing the product of two matrices.

Divide the task between a configured number of threads (going from 1 to the number of elements in the resulting matrix). See the effects on the execution time.

* **Architecture**



* **Implementation details**

The matrices are randomly generated and the result matrix is initialised with zeros.

The total number of elements m of the matrix is divided by the number of threads, and the remainder is then equally divided to all threads.

The matrixes are considered to be linearized and the linking between indexes is made by the formula k=i\*m+j, where k is the 'fictional array' s index and i,j are the indexes of the matrix's line, respectively column.

* **Test cases**

Sockets: 1

Cores: 2

Logical Processors: 4

**PRODUCT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Matrix dimension** | **Thread no** | **Execution time** | **Obs.** |
| 5x5 | 2 | 3072597 (0.003073 s) |  |
| 20x20 | 3 | 31827073 (0.031827 s) |  |
| 20x20 | 7 | 221527316(0.221527s) |  |
| 1000x1000 | 8 | 2125716662(2.12571s) |  |
| 1000x100 | 4 | 23633483(0.023633 s) |  |
| 120x120 | 3 | 47909601 (0.047910 s) |  |

**SUM**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dimensiune Matrice** | **Nr de threaduri** | **Timp executie** | **Obs.** |
| 10x10 | 3 | 1701870(0.001702 s) |  |
| 10x10 | 2 | 1221176(0.001221 s) |  |
| 100x100 | 6 | 4224980(0.004225 s) |  |
| 100x100 | 2 | 3590037(0.003590 s) |  |
| 1000x1000 | 2 | 16298305(0.016298 s) |  |
| 1000x1000 | 8 | 24870042(0.024870 s) |  |