**Documentarea performanței**

**C++ pentru laboratorul 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tip matrice** | **Tip alocare** | **Nr threads** | **Timp execuție** |
| N=M=1000  n=m=3 | dinamic | secvențial | 89,82532 |
| 2 | 15,90082 |
| 4 | 8,58016 |
| 8 | 6,34335 |
| 16 | 5,4872 |

**C++ pentru laboratorul 3 – varianta 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tip matrice** | **Nr procese** | **Timp execuție total** | **Timp execuție calcule** |
| N=M=1000  n=m=3 | 5 | 3392.58962 | 1311.69897 |
| 9 | 3531.07461 | 1569.86327 |
| 21 | 3532.54039 | 1669.73427 |

**C++ pentru laboratorul 3 – varianta 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tip matrice** | **Nr procese** | **Timp execuție total** | **Timp execuție calcule** |
| N=M=1000  n=m=3 | 4 | 3374.04862 | 156.9664 |
| 8 | 3380.99782 | 183.94071 |
| 20 | 3401.6776 | 338.42125 |

**Compararea timpilor de execuție pentru diferite valori ale lui p**

* Atât în prima variantă, cât și în cea de a doua, cu cât crește numărul de procese, cu atât crește și timpul de execuție total/calcule. Aceasta se datorează comunicării dintre procese, cu cât mai multe, cu atât mai îndelungată.

**Compararea timpilor de execuție cu timpii obținuți la laboratorul 2**

* Nici nu se pune problema, în cazul multithreading-ului, eficiența este mult îmbunătățită, lucrul cu memoria partajată fiind mai rapid decât comunicarea între procese.

**Compararea timpilor de execuție între cele două variante**

* Varianta a doua este mai rapidă, iar cel mai vizibil se observă aceasta la timpii de calcule. Mecanismul de Scatter este eficient și atunci procesele nu mai așteaptă atât de mult până procesul master împarte zonele din matrice, cum se întâmplă în prima variantă, secvențial. Tot mai este de luat în calcul timpul de trimitere/primire a frontierelor fiecărei zone de interferență între procese.

**Implementare**

* Mecanismele folosite sunt cele descrise în tema de laborator. A se menționa, suplimentar, mecanismul de transmitere a frontierelor:
  + Primul proces care efectuează calcule:
    - Trimite următorului ultima sa linie
    - Primește prima linie a următorului.
  + Următoarele procese până la ultimul (excluzându-l pe acesta):
    - Primesc de la cel de dinainte linia de dinainte de start
    - Trimit celui de dinainte linia start
    - Trimit celui de după ultima linie
    - Primesc de la cel de după prima linie a acestuia din urmă
  + Ultimul proces:
    - Primește ultima linie a penultimului
    - Trimite prima sa line penultimului.