|  |  |  |
| --- | --- | --- |
| Tip Matrice | Kernel Size | Timp executie |
| 10 | Seq | 74.97917 |
| 10 | 2x2 | 77.39739 |
| 10 | 4x4 | 81.87967 |
| 10 | 8x8 | 84.17124 |
| 10 | 16x16 | 85.62264 |
| 1000 | Seq | 123.4865 |
| 1000 | 2x2 | 105.0796 |
| 1000 | 4x4 | 102.2446 |
| 1000 | 8x8 | 91.41077 |
| 1000 | 16x16 | 101.7001 |
| 10000 | Seq | 6657.643 |
| 10000 | 2x2 | 3277.651 |
| 10000 | 4x4 | 2631.138 |
| 10000 | 8x8 | 1592.98 |
| 10000 | 16x16 | 1604.925 |

***Documentatie***

Se poate observa ca timpii scad in functie de numarul de thread-uri folosite. Cu toate acestea, timpii sunt destul de mari, acest lucru se datoreaza faptului ca transpunem 2 matrici pe GPU, una din care citim si una pe care efectuam calcului. Ceea ce intr-adevar este time-consuming este copierea matricilor.

In cazul unei matrici 10x10, se poate observa ineficienta folosirii unui numar mai mare de thread-uri pentru o computatie relativ usoara.

Metoda de abordare este urmatoarea:

* Impartim matricea in (n+p-1)/p blocuri fiecare bloc constituind cate p thread-uri.
* Astfel, am putea spune ca fiecare thread este alocat unui element/mai multe elemente border-box-ul oferit de blockul asignat acelei regiuni din matricea initiala