# C++

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Strategie** | **Tip matrice** | **Tip alocare** | **Nr threads** | **Timp executie** |
| **Line** | **N=M=10, n=m=3** | **static** | **1** | **53013** |
| **4** | **602837** |
| **dinamic** | **1** | **1542** |
| **4** | **359922** |
| **N=M=1000, n=m=5** | **static** | **1** | **15603739** |
| **2** | **10197875** |
| **4** | **6741158** |
| **8** | **8401452** |
| **16** | **9839281** |
| **dinamic** | **1** | **17117770** |
| **2** | **9566500** |
| **4** | **7256839** |
| **8** | **5446310** |
| **16** | **4899568** |
| **N=10 M=10000 , n=m=5** | **static** | **1** | **1063639** |
| **2** | **1045843** |
| **4** | **752562** |
| **8** | **1161309** |
| **16** | **2173812** |
| **dinamic** | **1** | **1826212** |
| **2** | **1481195** |
| **4** | **1123177** |
| **8** | **921581** |
| **16** | **1495240** |
| **N=10000 M=10, n=m=5** | **static** | **1** | **48987011** |
| **2** | **30411989** |
| **4** | **24758734** |
| **8** | **31747746** |
| **16** | **50321702** |
| **dinamic** | **1** | **2374243** |
| **2** | **1684312** |
| **4** | **1185115** |
| **8** | **959886** |
| **16** | **1695506** |
| **N=10000 M=10000, n=m=5** | **static** | **1** | **795383256** |
| **2** | **392985151** |
| **4** | **344784166** |
| **8** | **208796203** |
| **16** | **162784043** |
| **dinamic** | **1** | **1584720429** |
| **2** | **838608683** |
| **4** | **534596134** |
| **8** | **391776302** |
| **16** | **226271799** |
| **Column** | **N=M=10, n=m=3** | **static** | **1** | **69442** |
| **4** | **457515** |
| **dinamic** | **1** | **1592** |
| **4** | **443141** |
| **N=M=1000, n=m=5** | **static** | **1** | **11985429** |
| **2** | **8941210** |
| **4** | **5814328** |
| **8** | **10253625** |
| **16** | **11030020** |
| **dinamic** | **1** | **18080006** |
| **2** | **10003626** |
| **4** | **8006977** |
| **8** | **5664485** |
| **16** | **5481601** |
| **N=10 M=10000 , n=m=5** | **static** | **1** | **884933** |
| **2** | **785218** |
| **4** | **771311** |
| **8** | **970369** |
| **16** | **1360517** |
| **dinamic** | **1** | **1890990** |
| **2** | **1117121** |
| **4** | **931899** |
| **8** | **855717** |
| **16** | **1424623** |
| **N=10000 M=10, n=m=5** | **static** | **1** | **56051555** |
| **2** | **44741833** |
| **4** | **41198243** |
| **8** | **45012994** |
| **16** | **90793120** |
| **dinamic** | **1** | **1717631** |
| **2** | **1343022** |
| **4** | **1125109** |
| **8** | **1000951** |
| **16** | **1431616** |
| **N=10000 M=10000, n=m=5** | **static** | **1** | **791146844** |
| **2** | **425313857** |
| **4** | **260965956** |
| **8** | **220011675** |
| **16** | **159804458** |
| **dinamic** | **1** | **1706419683** |
| **2** | **945736378** |
| **4** | **538872235** |
| **8** | **342096840** |
| **16** | **321066021** |
| **Distribution** | **N=M=1000, n=m=5** | **static** | **1** | **12702996** |
| **2** | **13902277** |
| **4** | **12636643** |
| **8** | **10114984** |
| **16** | **11273195** |
| **dinamic** | **1** | **17197574** |
| **2** | **16293873** |
| **4** | **14798194** |
| **8** | **12170798** |
| **16** | **12460999** |
| **N=10 M=10000 , n=m=5** | **static** | **1** | **1121290** |
| **2** | **1650439** |
| **4** | **1445420** |
| **8** | **1525176** |
| **16** | **2116908** |
| **dinamic** | **1** | **2010508** |
| **2** | **2390473** |
| **4** | **1807417** |
| **8** | **1954836** |
| **16** | **1912825** |
| **N=10000 M=10, n=m=5** | **static** | **1** | **48391666** |
| **2** | **28818053** |
| **4** | **40964822** |
| **8** | **30056955** |
| **16** | **44928148** |
| **dinamic** | **1** | **1614712** |
| **2** | **1701356** |
| **4** | **1475238** |
| **8** | **1737025** |
| **16** | **1706966** |
| **N=10000 M=10000, n=m=5** | **static** | **1** | **768183760** |
| **2** | **1582788612** |
| **4** | **888109242** |
| **8** | **834854627** |
| **16** | **784540774** |
| **dinamic** | **1** | **1670261189** |
| **2** | **1922893635** |
| **4** | **1197742721** |
| **8** | **1472205818** |
| **16** | **1334439156** |
| **Block** | **N=M=1000, n=m=5** | **static** | **1** | **11910136** |
| **4** | **5751466** |
| **16** | **8021118** |
| **dinamic** | **1** | **17181618** |
| **4** | **7427663** |
| **16** | **5782860** |
| **N=10 M=10000 , n=m=5** | **static** | **1** | **1066351** |
| **4** | **814517** |
| **dinamic** | **1** | **1796503** |
| **4** | **1039960** |
| **N=10000 M=10, n=m=5** | **static** | **1** | **38773562** |
| **4** | **27982304** |
| **dinamic** | **1** | **1847982** |
| **4** | **1083495** |
| **N=10000 M=10000, n=m=5** | **static** | **1** | **930232490** |
| **4** | **252773233** |
| **16** | **195953890** |
| **dinamic** | **1** | **1552947783** |
| **4** | **603899669** |
| **16** | **300601511** |

# JAVA

|  |  |  |  |
| --- | --- | --- | --- |
| **Strategie** | **Tip matrice** | **Nr threads** | **Timp executie** |
| **Line** | **N=M=10, n=m=3** | **1** | 244626 |
| **4** | 3468581 |
| **N=M=1000, n=m=5** | **1** | 45956167 |
| **2** | 37338707 |
| **4** | 36331260 |
| **8** | 40135226 |
| **16** | 47503517 |
| **N=10 M=10000 , n=m=5** | **1** | 17348026 |
| **2** | 28578988 |
| **4** | 40455524 |
| **8** | 49670396 |
| **16** | 51680694 |
| **N=10000 M=10, n=m=5** | **1** | 14646664 |
| **2** | 20069740 |
| **4** | 23731230 |
| **8** | 43680108 |
| **16** | 64443636 |
| **N=10000 M=10000, n=m=5** | **1** | 3970908378 |
| **2** | 2126807852 |
| **4** | 1105753726 |
| **8** | 861830952 |
| **16** | 739559187 |
| **Column** | **N=M=10, n=m=3** | **1** | 246530 |
| **4** | 3136305 |
| **N=M=1000, n=m=5** | **1** | 48126337 |
| **2** | 34363767 |
| **4** | 37616418 |
| **8** | 40141935 |
| **16** | 52774475 |
| **N=10 M=10000 , n=m=5** | **1** | 18538734 |
| **2** | 23743736 |
| **4** | 27741061 |
| **8** | 32646383 |
| **16** | 40672302 |
| **N=10000 M=10, n=m=5** | **1** | 13533222 |
| **2** | 18645130 |
| **4** | 26119081 |
| **8** | 44070048 |
| **16** | 38972098 |
| **N=10000 M=10000, n=m=5** | **1** | 4012877244 |
| **2** | 2183439103 |
| **4** | 1259227104 |
| **8** | 927628204 |
| **16** | 665010974 |
| **Distribution** | **N=M=1000, n=m=5** | **1** | 50110552 |
| **2** | 64806898 |
| **4** | 55506791 |
| **8** | 58686726 |
| **16** | 72139420 |
| **N=10 M=10000 , n=m=5** | **1** | 17179935 |
| **2** | 21465459 |
| **4** | 26001370 |
| **8** | 33809071 |
| **16** | 66393120 |
| **N=10000 M=10, n=m=5** | **1** | 13444248 |
| **2** | 24962861 |
| **4** | 25536833 |
| **8** | 35689752 |
| **16** | 61241735 |
| **N=10000 M=10000, n=m=5** | **1** | 7846793748 |
| **2** | 6816605622 |
| **4** | 5972826577 |
| **8** | 2507805816 |
| **16** | 2785741835 |
| **Block** | **N=M=1000, n=m=5** | **1** | 56940052 |
| **4** | 37395276 |
| **16** | 46361728 |
| **N=10 M=10000 , n=m=5** | **1** | 18361010 |
| **4** | 38705626 |
| **N=10000 M=10, n=m=5** | **1** | 12018670 |
| **4** | 25665247 |
| **N=10000 M=10000, n=m=5** | **1** | 3762873819 |
| **4** | 1408241028 |
| **16** | 741040324 |

# Analiza

Conform rezultatelor, putem observa ca variantele C++ sunt mai rapide in comparative cu Java. De asemenea, se observa ca alocarea statica este mai rapida decat alocarea dinamica, mai putin in cazul in care dimensiunile sunt disproportionate (10x10000, 10000x10); exista de asemenea un overhead in varianta 1000x1000, avand alocate 10000x10000 elemente.

Folosirea mai multor threaduri pentru dimensiuni mici merge mai incet decat folosirea a mai putine, din cauza overheadului cauzat de threaduri. In schimb, folosirea mai multor threaduri duce la o diferenta enorma pentru 10000x10000.

Daca avem mai multe linii decat coloane, impartirea pe linii este mai buna decat cea pe coloane. Invers, impartirea pe coloane este mai buna daca avem mai multe coloane. In rest impartea pe coloane duce la overhead din cauza missurilor la cache. Folosirea impartirii pe blockuri sau cu ajutorul functiei de distributie nu are aceasta problema, dar exista un overhead de la lista de indici transmisa pentru distributie.

Pentru distributia liniara, vedem restul impartii la N la P, si luam blockuri de marimea calculata. In functie de restul impartirii la N la P, primele rest blocuri vor fi cu 1 mai mari. La fel pentru distributia pe coloane, doar ca lucram cu M in loc de N, si blocurile vor fi de coloane.

Pentru distributia block, consideram doar cazul in care se impart exact dimensiunile (dar asta poate fi usor modificat), si impartim dreptunghiul in dreptunghiuri egale mai mici, la fel ca la pe linie/coloane doar ca nested/2D.

Pentru distributia cu functie, liniarizam conceptul matricea, si impartim indecsii cum am facut-o pe linie/coloanal in bucketuri consecutive, iar apoi pentru fiecare thread lucram doar indecsii obtinuti (nu liniarizam efectiv matricea).