**SPRAWOZDANIE**

Przedmiot: Programowanie komputerów wielordzeniowych

Laboratorium: 2

Student: Artur Ziemba

Grupa: C

Celem sprawozdania było sprawdzenie czasu pracy programu wykonującego operacje mnożenia macierzy w zależności od ilości wykorzystanych wątków systemu operacyjnego przy użyciu języka c lub c++ oraz biblioteki Openmp.

Wyniki zostały uzyskane na komputerze z systemem Windows 7 (64bit), 16GB RAM oraz procesorze Intel(R) Core(TM) i7-7700K CPU 4.20 GHz, 4 korowym, (8 procesorów logicznych). Do kompilacji programu wykorzystano kompilator g++. Każdy test został powtórzony 20 razy, a uzyskane wyniki prezentują wartość średnią.

1. Implementacja macierzy przy użyciu dynamicznej tablicy dwuwymiarowej:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | iterowanie przez kolumny, następnie przez wiersze | | iterowanie przez wiersze następnie przez kolumny | | Algorytm Strassena | |
| ilość wątków | 50x50 | 500x500 | 50x50 | 500x500 | 50x50 | 500x500 |
| 1 | 0.006296 | 7.507960 | 0.006241 | 7.296810 | 0.006121 | 7.084263 |
| 2 | 0.006226 | 7.417120 | 0.006207 | 7.259180 | 0.006052 | 7.058364 |
| 3 | 0.006188 | 7.659780 | 0.006290 | 7.284000 | 0.006103 | 7.078346 |
| 4 | 0.006188 | 7.477020 | 0.006732 | 7.302260 | 0.006034 | 7.083563 |
| 5 | 0.006167 | 7.516840 | 0.006520 | 7.294970 | 0.006195 | 7.087363 |
| 6 | 0.006186 | 7.636640 | 0.006281 | 7.229660 | 0.006173 | 7.023522 |
| 7 | 0.006217 | 7.508680 | 0.006221 | 7.223110 | 0.006137 | 7.028384 |
| 8 | 0.006158 | 7.527710 | 0.006465 | 7.230380 | 0.006523 | 7.027246 |

1. Implementacja macierzy na kolekcji std::vector:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | iterowanie przez kolumny, następnie przez wiersze | | iterowanie przez wiersze następnie przez kolumny | | Algorytm Strassena | |
| rozmiary macierzy / ilość wątków | 50x50 | 500x500 | 50x50 | 500x500 | 50x50 | 500x500 |
| 1 | 0.104251 | 104.447000 | 0.106069 | 104.506000 | 0.094208 | 90.836384 |
| 2 | 0.103661 | 104.555000 | 0.105964 | 103.855000 | 0.094462 | 90.375036 |
| 3 | 0.103543 | 104.198000 | 0.105958 | 104.741000 | 0.094937 | 90.482645 |
| 4 | 0.103489 | 104.474000 | 0.105985 | 104.770000 | 0.094936 | 90.472594 |
| 5 | 0.103940 | 104.664000 | 0.105981 | 105.559000 | 0.094163 | 91.036245 |
| 6 | 0.103826 | 104.561000 | 0.105886 | 103.976000 | 0.097300 | 91.736293 |
| 7 | 0.103549 | 104.403000 | 0.106135 | 103.795000 | 0.094837 | 92.983332 |
| 8 | 0.103597 | 104.949000 | 0.108377 | 104.073000 | 0.096724 | 90.926212 |

Duża wartość czasowa uzyskana dla macierzy 500x500 dla tej kolekcji jest spowodowana sposobem zapisu kolekcji w pamięci (musi zostać dopasowana do ciągłego obszaru pamięci). W związku z tym znaczący okres pracy zajęło alokowanie zasobów.

1. Podsumowanie wyników:

Uzyskane wyniki pokazują, że największa sprawność została uzyskana dla rozwiązania bazującego na tablicy dwuwymiarowej oraz algorytmie Strassena. Względem programu jednowątkowego dla macierzy 50x50 najlepszą sprawność uzyskano na czterech wątkach - przyśpieszenie o 0.000087 sekundy (1.42%), a dla macierzy 500x500 najlepszą sprawność uzyskano na sześciu wątkach – przyśpieszenie o 0.060741 sekundy (0.86%)