**SPRAWOZDANIE**

**z zajęć laboratoryjnych na przedmiocie**

**Programowanie równoległe i rozproszone**

**Wykonane przez:**

* Ivan Prakapets, nr indeksu 295139
* Piotr Jeleniewicz, nr indeksu 291072

**Opis zadania*:***

W ramach laboratorium z przedmiotu Programowanie równoległe i rozproszone, wykonane zostały dwa programy wykonujące następujący scenariusz:

* wymnożenie dwóch macierzy
* wyznaczenie normy Frobeniusa macierzy będącej wynikiem poprzednio wykonanego mnożenia

**Dane wejściowe*:***

|  |  |
| --- | --- |
| Macierz A o wymiarach: 5 x 3  0.196717 0.160657 0.628787 0.846665 0.201070 0.948445 0.325611 0.517757 0.893094 0.203669 0.843018 0.946533 0.061910 0.457823 0.988852 | Macierz B o wymiarach: 3 x 2  0.659330 0.410710 0.947040 0.391100 0.878600 0.714600 |

**Oczekiwane dane wyjściowe*:*** Do sprawdzenia mnożenia dwóch macierzy użyliśmy kalkulatora online dostępnego pod adresem: <https://www.naukowiec.org/macierz.html>

Wynikiem mnożenia macierzy A x B jest macierz C o wymiarach 5 x 2:

0.834302 0.592957  
1.581956 1.104131  
1.489694 0.974431  
1.764280 1.089745  
1.343201 0.911115

Norma Frobeniusa uzyskanej macierzy C wynosi: **3,856502**

Obliczenia normy Frobeniusa wykonane zostało przy użyciu narzędzia: <https://keisan.casio.com/exec/system/15052019544540>

**Opis programu napisanego w języku C:**

Przykładowe wyjście programu zawierające wynik mnożenia macierzy A i B zdefiniowanych powyżej oraz normę Frobeniusa macierzy wynikowej tego mnożenia prezentuje się następująco:

Ilosc N watkow potomnych:8

pierwsza macierz ma wymiar 5 x 3, a druga 3 x 2

Rozmiar C: 5x2

A:

[

0.196717 0.160657 0.628787

0.846665 0.201070 0.948445

0.325611 0.517757 0.893094

0.203669 0.843018 0.946533

0.061910 0.457823 0.988852

]

B:

[

0.659330 0.410710

0.947040 0.391100

0.878600 0.714600

]

Wynik mnozenia C = A \* B:

[

0.834302 0.592958

1.581957 1.104131

1.489694 0.974431

1.764281 1.089746

1.343201 0.911115

]

Suma sum = 11.685816

Norma Frobeniusa = 3.856502

Execution time: 5006 ms

Wyniki obliczania wyniku mnożenia macierzy oraz normy Frobeniusa są zgodne z wynikami uzyskanymi za pomocą narzędzi online, co wskazuje, że program działa poprawnie.

**Wnioski:**

Zależność czasu wykonania obliczeń od liczby wątków dla przykładowych macierzy A i B zdefiniowanych powyżej są na tyle małe, że ich trudno jest jednoznacznie zinterpretować wyniki.

Dlatego poniżej przedstawiono zależność czasu obliczeń mnożenia macierzy od liczby wątków dla dwóch macierzy o wymiarach 500 x 500:

|  |  |
| --- | --- |
| Ilość wątków | Czas [ms] |
| 1 | 2461 |
| 2 | 1938 |
| 4 | 1751 |
| 8 | 1801 |

Pliki zawierające te macierzy nazywają się A.TXT oraz B.TXT i są dołączone do plików zadania.

Z powyższej tabeli można odczytać, że czas działania programu jest najoptymalniejszy dla 4 wątków – wynika to najprawdopodobniej z architektury procesora, na którym przeprowadzane były testy – procesor posiada 4 rdzenie i 4 wątki. Jak widać różnica pomiędzy wykonaniem na 1 wątku, a wykonaniem na 4 wątkach wynosi 710 ms to w tej skali powoduje znaczny zysk przy zastosowaniu obliczeń wielowątkowych. Zysk ten wynika z faktu wykonywania się obliczeń równolegle na każdy z wątków co powoduje znaczne przyspieszenie obliczeń. W przypadku 8 wątków wynik uległ delikatnemu pogorszeniu ze względu na to, że procesor nie był w stanie wykonywać obliczeń na wszystkich wątkach jednocześnie.

**Opis programu napisanego w języku Python:**

Przykładowe wyjście programu zawierające wynik mnożenia macierzy A i B zdefiniowanych powyżej oraz normę Frobeniusa macierzy wynikowej tego mnożenia prezentuje się następująco:

Result of multiplaying:

0.83430228309 0.59295778197

1.58195674425 1.10413105615

1.4896940783100001 0.97443142891

1.76428074229 1.08974571659

1.34320118142 0.9111152706

Multiplication time: 0.0002040863037109375

Frobenius norm = 3.8565022405971203

Frobenius norm calculation time: 0.00020837783813476562

Number of threads: 1

Wyniki obliczania wyniku mnożenia macierzy oraz normy Frobeniusa są zgodne z wynikami uzyskanymi za pomocą narzędzi online, co wskazuje, że program działa poprawnie.

Wątki w języku Python są zależne od GILa, przez w implementacji Pythona Cpython tylko jeden wątek może być wykonywany jednocześnie. Z tego powodu, wykonywanie obliczeń na wielu wątkach, często nie wnosi poprawy wydajności obliczeń.

**Wnioski:**

Poniżej przedstawiono zależność czasu obliczeń mnożenia macierzy od liczby wątków dla dwóch macierzy o wymiarach 500 x 500:

|  |  |
| --- | --- |
| Ilość wątków | Czas [s] |
| 1 | 11,7245 |
| 2 | 12,0007 |
| 4 | 12,6120 |
| 8 | 12.9652 |

Z powyższej tabeli łatwo wywnioskować, że stosowanie zwykłych wątków dostępnych w języku Python nie przynosi korzyści, jeśli w wątkach tych nie występują czasu oczekiwania, w trakcie, których inne wątki mogłyby wykonywać swoją pracę. Czas wraz ze zwiększaniem liczby wątków, także wzrastał co spowodowane było kosztami czasowymi operacji tworzenia wątków. Prawdziwą równoległość w działaniu, można uzyskać w Pythonie stosując np. procesy z biblioteki *multiprocessing*  - wtedy dopiero zauważalne byłby korzyści z posiadania procesora wielordzeniowego/wielowątkowego przy korzystaniu z aplikacji napisanych w Pythonie. Porównując ze sobą wyniki uzyskane w programie napisanym w języku C i w programie napisanym w języku Python, łatwo zauważyć, że język C jest znacznie wydajniejszym językiem w zastosowaniach obliczeniowych, dodatkowo umożliwia równolegle wykonywanie obliczeń