**Jędrzej Kuczyński**

**Sprawozdanie - zadanie 1**

**Poprawki zaznaczono kolorem czerwonym**

**Wszelkie ćwiczenia i testy wykonano na komputerach w laboratorium 2.6.22 na komputerach z systemem operacyjnym Ubuntu i środowisku MPICH 3.1.**

**1. Ćwiczenie 5**

**Ćwiczenie 5 polegało na napisaniu programu wykorzystującego procesy MPI wraz z przesyłaniem komunikatów do obliczenia liczby pi. Program testowano na jednym komputerze, lecz wielu procesach (wersja jednokomputerowa) oraz w systemie wielokomputerowym.**

**Część 1 - wersja jednokomputerowa**

**LP - liczba procesów. Na przykład: LP3 - program został uruchomiony na 3 procesach.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **LP1** | **LP2** | **LP3** | **LP4** | **LP5** | **LP6** | **LP7** | **LP8** |
| **Czas obliczeń [s]** | 12.0343 | 6.0507 | 4.1380 | 3.1672 | 2.6233 | 2.1864 | 1.8737 | 1.6393 |
| **Przyspieszenie** |  | 1.9889 | 2.9082 | 3.7997 | 4.5874 | 5.5042 | 6.4227 | 7.3412 |

*Tabela 1. Tabela prezentująca czasy obliczeń (w sekundach) i przyspieszenie dla poszczególnych wersji programu liczącego liczbę pi, uruchamianych na jednym komputerze. LP - liczba procesów (LP3 - program został uruchomiony za pomocą 3 procesów)*

**Część 2 - wersja wielokomputerowa**

**Na przykład: LP2/LP1 - program został uruchomiony w systemie wielokomputerowym na dwóch komputerach (2 procesy na pierwszym i 1 na drugim; łącznie 3 procesy)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **LP1/LP1** | **LP2/LP1** | **LP2/LP2** | **LP3/LP2** | **LP3/LP3** | **LP4/LP3** | **LP4/LP4** | **LP5/LP4** | **LP5/LP5** | **LP6/LP5** | **LP6/LP6** | **LP7/LP6** | **LP7/LP7** |
| **Czas obliczeń [s]** | 6.0253 | 4.0180 | 3.0286 | 2.4720 | 2.0672 | 1.8129 | 1.5861 | 1.4585 | 1.3128 | 1.1965 | 1.0952 | 1.0124 | 0.9393 |
| **Przyspieszenie** | 1.9973 | 2.9951 | 3.9736 | 4.8683 | 5.8214 | 6.6382 | 7.5876 | 8.2510 | 9.1672 | 10.0580 | 10.9880 | 11.8874 | 12.8117 |

*Tabela 2. Tabela prezentująca czasy obliczeń (w sekundach) i przyspieszenie dla poszczególnych wersji programu liczącego liczbę pi, uruchamianych w systemie wielokomputerowym   
(2 komputery).*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1/1/1** | **2/1/1** | **2/2/1** | **2/2/2** | **3/2/2** | **3/3/2** | **3/3/3** | **4/3/3** | **4/4/3** | **4/4/4** | **5/4/4** | **5/5/4** | **5/5/5** |
| **Czas obliczeń [s]** | 4.0187 | 3.0281 | 2.4249 | 2.0230 | 1.7726 | 1.5537 | 1.3827 | 1.2701 | 1.1554 | 1.0608 | 1.0092 | 0.9388 | 0.8776 |
| **Przyspieszenie** | 2.9946 | 3.9741 | 4.9629 | 5.9487 | 6.7892 | 7.7456 | 8.7035 | 9.4749 | 10.4159 | 11.3447 | 11.9243 | 12.8188 | 13.7129 |

*Tabela 3. Tabela prezentująca czasy obliczeń (w sekundach) i przyspieszenie dla poszczególnych wersji programu liczącego liczbę pi, uruchamianych w systemie wielokomputerowym  
 (3 komputery).*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1/1/1/1** | **2/1/1/1** | **2/2/1/1** | **2/2/2/1** | **2/2/2/2** | **3/2/2/2** | **3/3/2/2** | **3/3/3/2** | **3/3/3/3** | **4/3/3/3** | **4/4/3/3** | **4/4/4/3** | **4/4/4/4** |
| **Czas obliczeń [s]** | 3.0151 | 2.4133 | 2.0169 | 1.7327 | 1.5180 | 1.3777 | 1.2447 | 1.1321 | 1.0382 | 0.9781 | 0.9092 | 0.8486 | 0.7955 |
| **Przyspieszenie** | 3.9913 | 4.9866 | 5.9668 | 6.9452 | 7.9280 | 8.7353 | 9.6684 | 10.6305 | 11.5916 | 12.3031 | 13.2362 | 14.1806 | 15.1274 |

*Tabela 4. Tabela prezentująca czasy obliczeń (w sekundach) i przyspieszenie dla poszczególnych wersji programu liczącego liczbę pi, uruchamianych w systemie wielokomputerowym   
(4 komputery).*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1/1/1/1/1 | 2/1/1/1/1 | 2/2/1/1/1 | 2/2/2/1/1 | 2/2/2/2/1 | 2/2/2/2/2 | 3/2/2/2/2 | 3/3/2/2/2 | 3/3/3/2/2 | 3/3/3/3/2 | 3/3/3/3/3 | 4/3/3/3/3 | 4/4/3/3/3 |
| **Czas obliczeń [s]** | 2.4145 | 2.0216 | 1.7337 | 1.5170 | 1.3497 | 1.2160 | 1.1260 | 1.0353 | 0.9593 | 0.8916 | 0.8328 | 0.7951 | 0.7493 |
| **Przyspieszenie** | 4.9842 | 5.9529 | 6.9412 | 7.9328 | 8.9162 | 9.8963 | 10.6872 | 11.6245 | 12.5450 | 13.4979 | 14.4499 | 15.1362 | 16.0611 |

*Tabela 5. Tabela prezentująca czasy obliczeń (w sekundach) i przyspieszenie dla poszczególnych wersji programu liczącego liczbę pi, uruchamianych w systemie wielokomputerowym  
 (5 komputerów).*

**2. Ćwiczenie 6 - kod poprawiony**

**3. Ćwiczenie 7**

Mm - mała macierz 2000x2000

Dm - duża macierz 4000x4000

S - sekwencyjne obliczenia

IJK/IKJ - kolejność pętli

O - poziom optymalizacji przez kompilator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **2000x2000, S, IJK, O Default** | **2000x2000, S, IKJ, O Default** | **4000x4000, S, IJK, O Default** | **4000x4000, S, IKJ, O Default** |
| **Czas obliczeń [s]** | 90,5303 | 28,6854 | 867,0024 | 227,5953 |
| **Prędkość [operacje/s]** | 176 736 361 | 557 775 152 | 147 635 113 | 562 401 732 |
|  | **2000x2000, S, IJK, O3** | **2000x2000, S, IKJ, O3** | **4000x4000, S, IJK, O3** | **4000x4000, S, IKJ, O3** |
| **Czas obliczeń [s]** | 62,3630 | 2,8016 | 642,6791 | 20,8237 |
| **Prędkość [operacje/s]** | 256 562 245 | 5 710 936 658 | 199 166 278 | 6 146 846 728 |

*Tabela 6. Tabela przedstawiająca czas obliczeń i prędkość poszczególnych wersji sekwencyjnego mnożenia Cannon'a.*

Dalsze obliczenia były przeprowadzane na pętli IKJ oraz największym poziomie optymalizacji O3.

Znaczenia nagłówków:

1K/4P - algorytm mnożenia Cannon'a był uruchamiany na jednym komputerze/procesorze i 4 procesach.

4K/4P - algorytm mnożenia Cannon'a był uruchamiany w systemie wielokomputerowym na 4 komputerach i łącznej liczbie procesów równej 4 (po jednym procesie na komputer/procesor)

4K/16P - algorytm mnożenia Cannon'a był uruchamiany w systemie wielokomputerowym na 4 komputerach i łącznej liczbie procesów równej 16 (po 4 procesy na komputer/procesor)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **2000x2000 1K/4P** | **4000x4000 1K/4P** | **2000x2000 4K/4P** | **2000x2000 4K/16P** | **4000x4000 4K/4P** | **4000x4000 4K/16P** |
| **Czas obliczeń [s]** | 0,8227 | 11,3530 | 0,4649 | 0,2527 | 4,3923 | 2,4195 |
| **Prędkość [operacje/s]** | 19 449 000 000 | 11 275 000 000 | 34 414 000 000 | 63 327 000 000 | 29 142 000 000 | 52 904 000 000 |
| **Przyspieszenie** | 3,4056 | 1,8342 | 6,0259 | 11,0887 | 4,7410 | 8,6067 |
| **Efektywność** | 0,8514 | 0,4586 | 1,5065 | 0,6930 | 1,1852 | 0,5379 |
| **Koszt zrównoleglenia** | 0,4890 | 24,5882 | -0,9419 | 1,2409 | -3,2547 | 17,8880 |
| **Względny koszt zrównoleglenia** | 0,1745 | 1,1808 | -0,3362 | 0,4429 | -0,1563 | 0,8590 |

*Tabela 7. Tabela przedstawiająca czasy obliczeń, prędkość i inne miary opisujące efektywność poszczególnych wersji algorytmu mnożenia Cannon'a*

Kryteria efektywnego przetwarzania:

a) prędkość - liczba operacji zmiennoprzecinkowych na sekundę, zależna od wymiaru macierzy/instancji N zgodnie ze wzorem 2\*N\*N\*N. Ogólnie będzie wzrastać wraz z liczbą procesów. Jednakże będziemy obserwować większą prędkość, gdy liczba procesów będzie równomiernie rozłożona na poszczególne procesory w systemie.

b) przyspieszenie (Sp = T\*s / Tp; gdzie T\*s - czas przetwarzania najlepszego algorytmu sekwencyjnego, Tp - czas przetwarzania równoległego) - będzie zmieniać się tak samo jak prędkość.

c) efektywność (E = Sp / p; Sp - przyspieszenie, p - liczba procesów) - określa część czasu przetwarzania, w której procesory są efektywnie wykorzystane. Będzie największa przy równomiernym rozłożeniu procesów na procesory oraz zredukowaniu komunikacji i synchronizacji między procesami. Ponadto większy rozmiar instancji obniża efektywność, ponieważ wiąże się z dużymi kosztami komunikacji - należy przesłać więcej wartości.

Wnioski:

a) Macierz 2000x2000:

- dla 4 procesów liczących w ramach jednego komputera obserwujemy prawie 3.5-krotne (3.4) przyspieszenie działania algorytmu, a w przypadku 16 procesów liczących na 4 komputerach (po 4 procesy na komputer) 11-krotne w porównaniu do algorytmu sekwencyjnego i trochę ponad 3-krotne w porównaniu do konfiguracji 1K/4P.

- sytuacja prezentuje się szczególnie w konfiguracji 4K/4P - obliczenia prowadzone na 4 komputerach i 4 procesach, po jednym na komputer. Przyspieszenie jest 6-krotne (większe od liczby uczestniczących w obliczeniach procesów) a efektywność > 1 (1.5). Taka niezwykła sytuacja może być związana z faktem, że procesy mają dostęp do całej pamięci podręcznej procesora i dane potrzebne do obliczeń kopiowane są jednokrotnie na początku działania programu. W związku z tym procesy nie tracą czasu na wielokrotne odwołania do pamięci.

b) Macierz 4000x4000:

- dla 4 procesów liczących w ramach jednego komputera obserwujemy prawie 2-krotne (1.83) przyspieszenie działania algorytmu, a w przypadku 16 procesów liczących na 4 komputerach (po 4 procesy na komputer) 8.6-krotne przyspieszenie w porównaniu do algorytmu sekwencyjnego i około 4.7-krotne przyspieszenie w porównaniu do konfiguracji 1K/4P.

- Tak jak w przypadku macierzy 2000x2000 sytuacja prezentuje się analogicznie w konfiguracji 4K/4P (po jednym procesie na komputer): przyspieszenie 4.74 a efektywność 1.18. Oczywiście niższe wartości tych miar wiążą się z większym rozmiarem instancji.

**4. Ćwiczenie 8 - kod opisany**