**Jędrzej Kuczyński**

**Bioinformatyka V rok**

**Sprawozdanie - OpenMP**

**Testy były przeprowadzane na prywatnym komputerze z systemem Linux Ubuntu oraz procesorem Intel(R) Core(TM) i5-4690 3.50GHz (liczba rdzeni: 4, liczba wątków: 4).**

Tabela 1. Czasy procesorów i przetwarzania dla kolejnych wersji wyznaczania liczby pi. Liczba wątków: 2 (połowa fizycznych procesorów). Czasy podane w sekundach.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Pi Serial | Pi Serial -O3 | Pi 2 | Pi 2 -O3 | Pi 3 | Pi 3 -O3 | Pi 4 | Pi 4 -O3 | Pi 5 | Pi 5 -O3 | Pi 6 | Pi 6 -O3 |
| Czas procesorów | 12,4468 | 3,6335 | 21,4708 | 3,6545 | 100,1203 | 85,3206 | 12,3894 | 3,6206 | 12,4042 | 3,6243 | 12,5828 | 3,6203 |
| Czas przetwarzania | 12,4467 | 3,6333 | 11,0841 | 1,8279 | 50,3129 | 43,9819 | 6,2100 | 1,8114 | 6,2036 | 1,8128 | 6,2928 | 1,8141 |

Tabela 2. Wartości przyspieszenia przetwarzania dla poszczególnych wersji wyznaczania liczby pi. Liczba wątków: 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pi 2 -O3 | Pi 3 -O3 | Pi 4 -O3 | Pi 5 -O3 | Pi 6 -O3 |
| Przyspieszenie przetwarzania | 1,99 | 0,08 | 2,01 | 2,00 | 2,00 |

Tabela 3. Czasy procesorów i przetwarzania dla kolejnych wersji wyznaczania liczby pi. Liczba wątków: 4 (liczba fizycznych procesorów). Czasy podane w sekundach.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Pi Serial | Pi Serial -O3 | Pi 2 | Pi 2 -O3 | Pi 3 | Pi 3 -O3 | Pi 4 | Pi 4 -O3 | Pi 5 | Pi 5 -O3 | Pi 6 | Pi 6 -O3 |
| Czas procesorów | 12,4468 | 3,6335 | 24,1906 | 3,8040 | 287,8727 | 281,8412 | 13,0125 | 3,8036 | 13,0003 | 3,7993 | 17,9267 | 3,8068 |
| Czas przetwarzania | 12,4467 | 3,6333 | 6,6213 | 0,9556 | 74,0138 | 74,0974 | 3,2674 | 0,9572 | 3,2642 | 0,9537 | 5,0353 | 0,9551 |

Tabela 4. Wartości przyspieszenia przetwarzania dla poszczególnych wersji wyznaczania liczby pi. Liczba wątków: 4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pi 2 -O3 | Pi 3 -O3 | Pi 4 -O3 | Pi 5 -O3 | Pi 6 -O3 |
| Przyspieszenie przetwarzania | 3,80 | 0,05 | 3,80 | 3,81 | 3,80 |

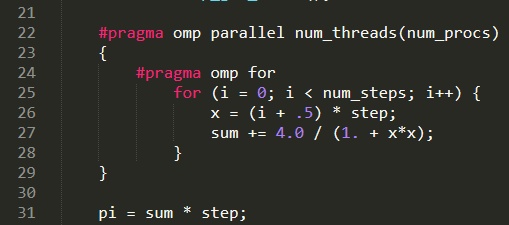
Tabela 5. Czasy procesorów i przetwarzania dla kolejnych wersji wyznaczania liczby pi. Liczba wątków: 16 (liczba procesorów logicznych). Czasy podane w sekundach.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Pi Serial | Pi Serial -O3 | Pi 2 | Pi 2 -O3 | Pi 3 | Pi 3 -O3 | Pi 4 | Pi 4 -O3 | Pi 5 | Pi 5 -O3 | Pi 6 | Pi 6 -O3 |
| Czas procesorów | 12,4468 | 3,6335 | 24,8717 | 3,8033 | 283,4737 | 269,9073 | 13,0013 | 3,8045 | 13,0029 | 3,8046 | 13,0403 | 3,6276 |
| Czas przetwarzania | 12,4467 | 3,6333 | 6,3381 | 0,9575 | 74,0102 | 71,8311 | 3,2669 | 0,9559 | 3,2600 | 0,9579 | 6,5249 | 1,8145 |

Tabela 6. Wartości przyspieszenia przetwarzania dla poszczególnych wersji wyznaczania liczby pi. Liczba wątków: 16.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pi 2 -O3 | Pi 3 -O3 | Pi 4 -O3 | Pi 5 -O3 | Pi 6 -O3 |
| Przyspieszenie przetwarzania | 3,79 | 0,05 | 3,80 | 3,79 | 2,00 |

**2. Wersja druga kodu - Pi 2.**

****

**Jest to niepoprawna wersja kodu zwracająca niepoprawny wynik, ze względu na brak dostępu do współdzielonej sumy w sposób niepodzielny!!!**

**Wersja sekwencyjna: Czas procesorów: 3,6335**; **Czas przetwarzania: 3,6333**

**2 wątki: Czas procesorów: 3,6545; Czas przetwarzania: 1,8279; Przyspieszenie: 1,99**

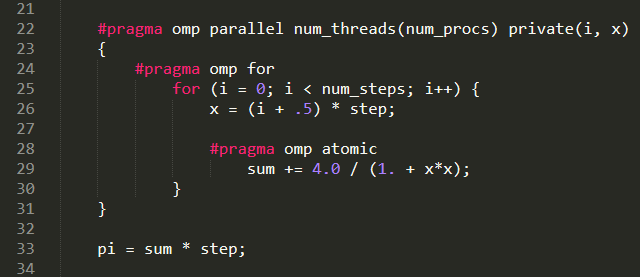
**4 wątki: Czas procesorów: 3,8040**; **Czas przetwarzania: 0,9556; Przyspieszenie: 3,80**

**16 wątków: Czas procesorów:** **3,8033**;  **Czas przetwarzania: 0,9575; Przyspieszenie: 3,79**

Unieważnianie kopii linii pamięci podręcznej nie występuje, ponieważ wątki pracują na współdzielonej sumie, która jest pojedynczą zmienną.

Zmienne prywatne: 'i' oraz 'x'. Zmienne współdzielone: "sum" (współdzielenie dotyczy odczytu i zapisu).

**3. Wersja trzecia kodu - Pi 3.**

****

**Wersja sekwencyjna: Czas procesorów: 3,6335**; **Czas przetwarzania: 3,6333**

**2 wątki: Czas procesorów: 85,3206; Czas przetwarzania: 43,9819; Przyspieszenie: 0,05**

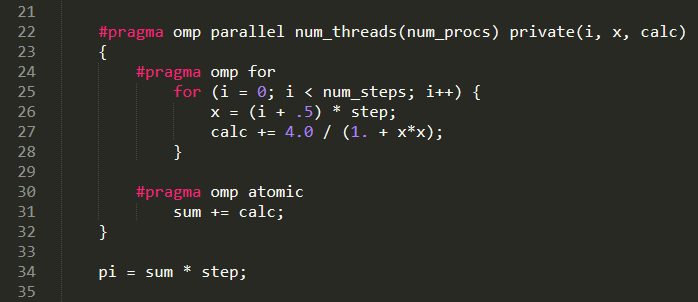
**4 wątki: Czas procesorów: 281,8412; Czas przetwarzania: 74,0974; Przyspieszenie: 0,05**

**16 wątków: Czas procesorów: 269,9073; Czas przetwarzania: 71,8311; Przyspieszenie: 0,05**

Unieważnianie kopii linii pamięci podręcznej nie występuje, ponieważ wątki pracują na współdzielonej sumie, która jest pojedynczą zmienną.

W przypadku tej wersji kodu można zaobserwować diametralnie dłuższe czasy procesorów i przetwarzania (jest to wersja wolniejsza od sekwencyjnej). Jest to spowodowane występowaniem w kodzie dyrektywy "atomic" na współdzielonej przez wątki sumie, której odczyt/zapis następuje w każdej iteracji pętli (gdzie iteracji jest sumarycznie miliard). Dyrektywa "atomic" zapewnia niepodzielność operacji aktualizacji sumy (w związku z czym wynik zwracany przez program jest już poprawny), jednakże robi to poprzez wykluczanie dostępu innych wątków do zmiennej. Powoduje to występowanie powtarzającej się sytuacji, w której pozostałe wątki czekają na uzyskanie dostępu do współdzielonej sumy i wykonanie na niej własnych operacji, co wydłuża działanie algorytmu.

**4. Wersja czwarta kodu - Pi 4.**

****

**Wersja sekwencyjna: Czas procesorów: 3,6335**; **Czas przetwarzania: 3,6333**

**2 wątki: Czas procesorów: 3,6206; Czas przetwarzania: 1,8114; Przyspieszenie: 2,01**

**4 wątki: Czas procesorów: 3,8036; Czas przetwarzania: 0,9559; Przyspieszenie: 3,80**

**16 wątków: Czas procesorów: 3,8045; Czas przetwarzania: 0,9559; Przyspieszenie: 3,80**

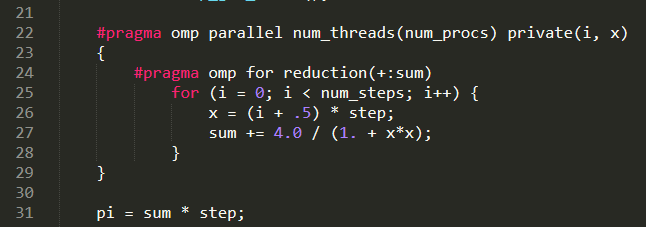
Unieważnianie kopii linii pamięci podręcznej nie występuje, ponieważ wątki pracują na współdzielonej sumie, która jest pojedynczą zmienną.

W tej wersji kodu, w porównaniu do poprzedniej, zastosowano prywatną dla każdego wątku zmienną "calc", która przechowuje liczoną sumę przez wątek. Synchronizacja wyników poprzez dyrektywę "atomic" została przeniesiona poza pętlę, co nadal zapewnia atomowość uaktualniania współdzielonej sumy, jednakże drastycznie zmniejsza czas spędzony przez wątki na czekaniu na dostęp do zmiennej.

Dla 2 i 4 wątków można zaobserwować odpowiednio prawie dwukrotne i czterokrotne przyspieszenie (sytuacja zbliżona do idealnej), czego można się spodziewać, ze względu na użyty do obliczeń czterordzeniowy procesor, w którym każdy rdzeń może operować na 4 wątkach (w przypadku obliczeń z wykorzystaniem 4 wątków w sytuacji idealnej każdy rdzeń otrzyma jeden wątek). Stąd spodziewany stopień lokalności dostępu do zmiennych jest duży, a jednocześnie występuje efektywny podział pracy. To wszystko wpływa na krótszy czas obliczeń.

W przypadku przeprowadzania obliczeń na 16 wątkach sytuacja prezentuje się dokładnie tak samo, jak w przypadku obliczeń na 4 wątkach. Można to wyjaśnić tym, że ilość pracy, którą należy wykonać nie zwiększyła się, a każdy wątek dostał jej mniejszą część. W związku z tym wątki spędzają mniej czasu na obliczeniach, ale więcej czasu na synchronizacji. Jednakże czas zaoszczędzony na obliczeniach wyrównuje straty synchronizacji. Stąd taki sam czas obliczeń jak w przypadku obliczeń przeprowadzanych przy użyciu 4 wątków.

**5. Wersja piąta kodu - Pi 5.**



**Wersja sekwencyjna: Czas procesorów: 3,6335; Czas przetwarzania: 3,6333**

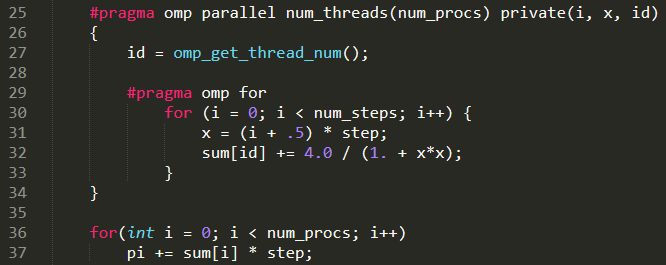
**2 wątki: Czas procesorów: 3,6243; Czas przetwarzania: 1,8128; Przyspieszenie: 2,00**

**4 wątki: Czas procesorów: 3,7993; Czas przetwarzania: 0,9537; Przyspieszenie: 3,81**

**16 wątków: Czas procesorów: 3,8046; Czas przetwarzania: 0,9579; Przyspieszenie: 3,79**

W przypadku tej wersji sytuacja prezentuje się tak samo jak w przypadku wersji 4.

**6. Wersja szósta kodu - Pi 6.**

****

**Wersja sekwencyjna: Czas procesorów: 3,6335; Czas przetwarzania: 3,6333**

**2 wątki: Czas procesorów: 3,6203; Czas przetwarzania: 1,8141; Przyspieszenie: 2,00**

**4 wątki: Czas procesorów: 3,8068; Czas przetwarzania: 0,9551; Przyspieszenie: 3,80**

**16 wątków: Czas procesorów: 3,6276; Czas przetwarzania: 1,8145; Przyspieszenie: 2,00**

W przypadku kompilacji kodu z opcją "-O3" sytuacja prezentuje się tak samo, jak w przypadku wersji 4 czy 5. Jednakże, gdy z kompilujemy kod bez tej opcji możemy zauważyć wpływ "false sharing" w przypadku obliczeń przy wykorzystaniu 4 i 16 wątków (różnica o około 2-3 sekundy):

**Wersja sekwencyjna: Czas procesorów: 12,4468; Czas przetwarzania: 12,4467**

**2 wątki: Czas procesorów: 12,5828; Czas przetwarzania: 6,2928; Przyspieszenie: 1,98**

**4 wątki: Czas procesorów: 17,9267; Czas przetwarzania: 5,0353; Przyspieszenie: 2,47**

**16 wątków: Czas procesorów: 13,0403; Czas przetwarzania: 6,5249; Przyspieszenie: 1,91**

Efekt ten jest spowodowany korzystaniem przez wątki ze współdzielonej tablicy i modyfikacji sąsiednich słów tablicy. Takie działanie powoduje wymóg ciągłego aktualizowania linii pamięci podręcznej procesora. Możliwe, że efekt ten nie występuje w przypadku obliczeń na 2 wątkach, ponieważ oba wątki operują w ramach tego samego rdzenia procesora.

**7. Eksperyment**