Łukasz Oprych Gr lab.5	Programowanie Równoległe T: OpenMP	7.12.2023r.

Cel ćwiczenia:

Zapoznanie się programami równoległymi tworzonymi w OpenMP, między innymi z dyrektywami i klauzulami.

Przebieg ćwiczenia:

Po pobraniu plików ze strony prowadzącego oraz przygotowanie struktury katalogowej i zaczęto wprowadzać zmiany w pliku openmp_petle_simple.c.

Dokonano zrównoleglenia pętli za pomocą dyrektywy parallel for, w sposób domyślny bez klauzuli schedule, użyto klauzuli default(none), w celu sterowania uzyskaniem ostatecznego wyniku w zmiennej suma_parallel użyto klauzuli reduction oraz w celu wymuszenia kolejności wykonywania operacji użyto klauzuli ordered.

```
loprych@loprych-VirtualBox:~$ export OMP_NUM_THREADS=4
```

```
#pragma omp parallel for default(none) reduction(+:suma_parallel) shared(a) ordered
```

```
for(int i=0;i<WYMIAR;i++) {
  int id_w = omp_get_thread_num();
    suma_parallel += a[i];

#pragma omp ordered
    printf("a[%2d]->W_%1d  \n",i,id_w);
}
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
Suma wyrazów tablicy: 156.060000
a[ 0]->W_0
a[ 1]->W_0
a[ 2]->W_0
a[ 3]->W_0
a[ 4]->W_0
a[ 5]->W_1
a[ 6]->W_1
a[ 7]->W_1
a[ 8]->W_1
a[ 9]->W_1
a[10]->W_2
a[11]->W_2
a[12]->W_2
a[13]->W_2
a[14]->W_3
a[15]->W_3
a[16]->W_3
a[17]->W_3

Suma wyrazów tablicy równolegle (z klauzulą - ....: 156.060000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$
```

Jak widać wątki, dostały dane kolejno oraz możemy zauważyć dekompozycję blokową.

Następnie przetestowano wersje klauzuli schedule i dynamic z użyciem 4 wątków

Pierwszy wariant mamy z użyciem schedule static, z dzielenie danych po 3 kolejno w każdej „paczce” na wątek.

```
  #pragma omp  parallel for default(none) schedule(static, 3) num_threads(4)
reduction(+:suma_parallel) shared(a) ordered
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
Suma wyrazów tablicy: 156.060000
a[ 0]->W_0
a[ 1]->W_0
a[ 2]->W_0
a[ 3]->W_1
a[ 4]->W_1
a[ 5]->W_1
a[ 6]->W_2
a[ 7]->W_2
a[ 8]->W_2
a[ 9]->W_3
a[10]->W_3
a[11]->W_3
a[12]->W_0
a[13]->W_0
a[14]->W_0
a[15]->W_1
a[16]->W_1
a[17]->W_1

Suma wyrazów tablicy równolegle (z klauzulą - ....: 156.060000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$
```

Jak widać każdy wątek kolejno dostał po 3 dane, po przydzieleniu danej porcji danych widać, że ponownie wątki dostają dane aż do czasu, gdy wszystkie dane zostaną obliczone. Można tu zauważyć dekompozycję cykliczną, z podziałem na bloki o rozmiarze 3.

Wariant schedule(static), każdy wątek dostaje kolejno dane blokowo.

Wynik programu:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
Suma wyrazów tablicy: 156.060000
a[ 0]->W_0
a[ 1]->W_0
a[ 2]->W_0
a[ 3]->W_0
a[ 4]->W_0
a[ 5]->W_1
a[ 6]->W_1
a[ 7]->W_1
a[ 8]->W_1
a[ 9]->W_1
a[10]->W_2
a[11]->W_2
a[12]->W_2
a[13]->W_2
a[14]->W_3
a[15]->W_3
a[16]->W_3
a[17]->W_3

Suma wyrazów tablicy równolegle (z klauzulą - ....: 156.060000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$
```

Wariant schedule(dynamic, 2)

Wątki w dynamicznie mają przydzielane dane, w tym przypadku po 2 na przydział.

```
  #pragma omp  parallel for default(none) schedule(dynamic, 2) num_threads(4)
reduction(+:suma_parallel) shared(a) ordered
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
Suma wyrazów tablicy: 156.060000
a[ 0]->W_3
a[ 1]->W_3
a[ 2]->W_0
a[ 3]->W_0
a[ 4]->W_1
a[ 5]->W_1
a[ 6]->W_3
a[ 7]->W_3
a[ 8]->W_0
a[ 9]->W_0
a[10]->W_1
a[11]->W_1
a[12]->W_2
a[13]->W_2
a[14]->W_3
a[15]->W_3
a[16]->W_0
a[17]->W_0

Suma wyrazów tablicy równolegle (z klauzulą - ....: 156.060000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$
```

Jak widać, pierwszy wątek, który otrzymał dane to czwarty wątek i każdy wątek otrzymuje „porcje” 2 danych.

Odpowiedź na pytanie: Wariant schedule(dynamic), wątki domyślnie mają przydzielane pojedynczą porcję danych w kolejności dynamicznej w zależności od tego, który wątek zgłosi się jako pierwszy. Bez klauzuli schedule, wątki są przyjmowane jak w static bez ustalenia porcji.

```
  #pragma omp  parallel for default(none) schedule(dynamic) num_threads(4)
reduction(+:suma_parallel) shared(a) ordered
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
Suma wyrazów tablicy: 156.060000
a[ 0]->W_1
a[ 1]->W_3
a[ 2]->W_0
a[ 3]->W_1
a[ 4]->W_3
a[ 5]->W_0
a[ 6]->W_1
a[ 7]->W_3
a[ 8]->W_0
a[ 9]->W_1
a[10]->W_3
a[11]->W_0
a[12]->W_1
a[13]->W_3
a[14]->W_0
a[15]->W_1
a[16]->W_3
a[17]->W_0

Suma wyrazów tablicy równolegle (z klauzulą - ....: 156.060000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle_simple
```

W każdym w przypadków jak widać obliczenia wykonane sekwencyjnie i równolegle są sobie równe.

Odpowiedź na pytanie prowadzącego: Wątki przy każdym uruchomieniu w schedule static zawsze dostają te same iteracje, a w dynamic zależnie od zgłoszenia się wątku.

Następnie skopiowano ze strony prowadzącego openmp_petle.c i przystąpiono do edycji.

Dekompozycja wierszowa, dokonano zrównoleglenie pętli zewnętrznej (po i), uzyskano sumę stosując reduction, ustalono schedule(static, 2)

```
  #pragma omp parallel for default(none) reduction(+:suma_parallel)  schedule(static, 2)
shared(a) private(j) ordered
   for(i=0;i<WYMIAR;i++) {
    int id_w = omp_get_thread_num();
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle
Suma wyrazów tablicy: 913.500000
(0,0)-W_0 (0,1)-W_0 (0,2)-W_0 (0,3)-W_0 (0,4)-W_0 (0,5)-W_0 (0,6)-W_0 (0,7)-W_0 (0,8)-W_0 (0,9)-W_0
(1,0)-W_0 (1,1)-W_0 (1,2)-W_0 (1,3)-W_0 (1,4)-W_0 (1,5)-W_0 (1,6)-W_0 (1,7)-W_0 (1,8)-W_0 (1,9)-W_0
(2,0)-W_1 (2,1)-W_1 (2,2)-W_1 (2,3)-W_1 (2,4)-W_1 (2,5)-W_1 (2,6)-W_1 (2,7)-W_1 (2,8)-W_1 (2,9)-W_1
(3,0)-W_1 (3,1)-W_1 (3,2)-W_1 (3,3)-W_1 (3,4)-W_1 (3,5)-W_1 (3,6)-W_1 (3,7)-W_1 (3,8)-W_1 (3,9)-W_1
(4,0)-W_2 (4,1)-W_2 (4,2)-W_2 (4,3)-W_2 (4,4)-W_2 (4,5)-W_2 (4,6)-W_2 (4,7)-W_2 (4,8)-W_2 (4,9)-W_2
(5,0)-W_2 (5,1)-W_2 (5,2)-W_2 (5,3)-W_2 (5,4)-W_2 (5,5)-W_2 (5,6)-W_2 (5,7)-W_2 (5,8)-W_2 (5,9)-W_2
(6,0)-W_3 (6,1)-W_3 (6,2)-W_3 (6,3)-W_3 (6,4)-W_3 (6,5)-W_3 (6,6)-W_3 (6,7)-W_3 (6,8)-W_3 (6,9)-W_3
(7,0)-W_3 (7,1)-W_3 (7,2)-W_3 (7,3)-W_3 (7,4)-W_3 (7,5)-W_3 (7,6)-W_3 (7,7)-W_3 (7,8)-W_3 (7,9)-W_3
(8,0)-W_4 (8,1)-W_4 (8,2)-W_4 (8,3)-W_4 (8,4)-W_4 (8,5)-W_4 (8,6)-W_4 (8,7)-W_4 (8,8)-W_4 (8,9)-W_4
(9,0)-W_4 (9,1)-W_4 (9,2)-W_4 (9,3)-W_4 (9,4)-W_4 (9,5)-W_4 (9,6)-W_4 (9,7)-W_4 (9,8)-W_4 (9,9)-W_4
Suma wyrazów tablicy równolegle: 913.500000
```

Jak widać wątki są przydzielane wierszowo, np. pierwsze 2 wiersze dostał wątek 0. Schedule ustala cykliczne przypisywanie wierszy.

Dekompozycja kolumnowa, zrównoleglenie pętli wewnętrznej (po j), ustalono schedule(dynamic)

```
   #pragma omp  parallel for default(none) reduction(+:suma_parallel)  schedule(dynamic)
shared(a,i)  ordered
    for(j=0;j<WYMIAR;j++) {
      suma_parallel += a[i][j];
      #pragma omp ordered
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle
Suma wyrazów tablicy: 913.500000
(0,0)-W_5 (0,1)-W_3 (0,2)-W_1 (0,3)-W_2 (0,4)-W_0 (0,5)-W_5 (0,6)-W_3 (0,7)-W_1 (0,8)-W_2 (0,9)-W_0
(1,0)-W_5 (1,1)-W_0 (1,2)-W_5 (1,3)-W_0 (1,4)-W_5 (1,5)-W_0 (1,6)-W_5 (1,7)-W_0 (1,8)-W_5 (1,9)-W_0
(2,0)-W_5 (2,1)-W_3 (2,2)-W_2 (2,3)-W_1 (2,4)-W_0 (2,5)-W_5 (2,6)-W_3 (2,7)-W_2 (2,8)-W_1 (2,9)-W_0
(3,0)-W_1 (3,1)-W_3 (3,2)-W_2 (3,3)-W_0 (3,4)-W_1 (3,5)-W_3 (3,6)-W_5 (3,7)-W_2 (3,8)-W_0 (3,9)-W_1
(4,0)-W_0 (4,1)-W_1 (4,2)-W_2 (4,3)-W_0 (4,4)-W_1 (4,5)-W_2 (4,6)-W_0 (4,7)-W_1 (4,8)-W_2 (4,9)-W_0
(5,0)-W_0 (5,1)-W_2 (5,2)-W_5 (5,3)-W_4 (5,4)-W_3 (5,5)-W_0 (5,6)-W_2 (5,7)-W_5 (5,8)-W_1 (5,9)-W_4
(6,0)-W_4 (6,1)-W_1 (6,2)-W_4 (6,3)-W_1 (6,4)-W_4 (6,5)-W_1 (6,6)-W_4 (6,7)-W_1 (6,8)-W_4 (6,9)-W_1
(7,0)-W_5 (7,1)-W_4 (7,2)-W_3 (7,3)-W_0 (7,4)-W_1 (7,5)-W_5 (7,6)-W_4 (7,7)-W_3 (7,8)-W_0 (7,9)-W_1
(8,0)-W_0 (8,1)-W_2 (8,2)-W_4 (8,3)-W_5 (8,4)-W_1 (8,5)-W_3 (8,6)-W_0 (8,7)-W_2 (8,8)-W_4 (8,9)-W_5
(9,0)-W_3 (9,1)-W_4 (9,2)-W_5 (9,3)-W_3 (9,4)-W_0 (9,5)-W_4 (9,6)-W_1 (9,7)-W_5 (9,8)-W_3 (9,9)-W_0
Suma wyrazów tablicy równolegle: 913.500000
```

Odpowiedzi na pytania: Każdy wiersz jest iterowany przez watek główny, a następnie osobno jest dzielony po kolumnach, któryś z wątków może nie brać udział. Podział danych przez dynamic z domyślną porcją wygląda, że wątki dostają po jednej danej (cyklicznie). Podział kolumn pomiędzy wątki nie jest taki sam, co wiersz zmienia się.

Dekompozycja kolumnowa, zrównoleglenie pętli zewnętrznej (pętla po kolumnach jest pętlą zewnętrzną, a pętla po wierszach wewnętrzną, doszło do transpozycji)

```
#pragma omp parallel default(none) shared(a, suma_parallel)
{
double suma_tmp = 0.0;
#pragma omp for schedule(static) ordered private(i)
for(j=0;j<WYMIAR;j++) {
  int id_w = omp_get_thread_num();

  for(i=0;i<WYMIAR;i++) {
    suma_tmp += a[i][j];
    #pragma omp ordered
```

Dla ułatwienia zrozumienia wyników zmieniono formę wypisania wyniku, na graficzną:

```
switch(omp_get_thread_num()){
 case 0:
   printf("#");
 break;
 case 1:
   printf("&");

 break;
 case 2:
   printf("$");
 break;
 }
}
```

Wynik:

```
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ gcc -fopenmp openmp_
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$ ./openmp_petle
Suma wyrazów tablicy: 913.500000
##########
##########
&&&&&&&&&&
&&&&&&&&&&
$$$$$$$$$$
$$$$$$$$$$



Suma wyrazów tablicy równolegle: 913.500000
loprych@loprych-VirtualBox:~/Desktop/PR_lab/lab_9/petle$
```

Zastosowany wariant klauzuli schedule daje wątkom, przypisanie danych blokowo i podzielono kolumnowo na wątki.