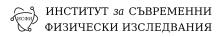
## Използване на OpenMP. Част 2.

Kypc "Паралелно програмиране"



Стоян Мишев

Задача

$$\int_0^1 \frac{4}{1+x^2} dx = \pi$$

Loop Section Single Task

```
#pragma omp parallel
#pragma omp for
      for (I=\dot{O}; I< N; I++){}
             NEAT_STUFF(I);
   double res[MAX]; int i;
 #pragma omp parallel for
  for (i=0;i<MAX; i++){
      res[i] = huge():
```

Schedule

schedule(static [,chunk])	Deal-out blocks of iterations of size "chunk" to each thread.
schedule(dynamic[,chunk])	Each thread grabs "chunk" iterations off a queue until all iterations have been handled.
schedule(guided[,chunk])	Threads dynamically grab blocks of iterations. The size of the block starts large and shrinks down to size "chunk" as the calculation proceeds.
schedule(runtime)	Schedule and chunk size taken from the OMP_SCHEDULE environment variable (or the runtime library).
schedule(auto)	Schedule is left up to the runtime to choose (does not have to be any of the above).

```
double ave=0.0, A[MAX]; int i;
#pragma omp parallel for reduction (+:ave)
  for (i=0;i<MAX; i++){
    ave + = A[i];
}
ave = ave/MAX;</pre>
```

```
double ave=0.0, A[MAX]; int i;
#pragma omp parallel for reduction (+:ave)
  for (i=0;i<MAX; i++){
    ave + = A[i];
}
ave = ave/MAX;</pre>
```

Operator	Initial Value
+	0
*	1
-	0
min	Largest pos num
max	Most neg num

Integral

```
#include <omp.h>
static long num+staps = 100000;
                                     double step;
void main ()
       int i; double x, pi, sum = 0.0;
       step = 1.0/(double) num_steps;
       #pragma omp parallel
          double x;
          #pragma omp for reduction(+:sum)
             for (i=0;i<num_steps; i++){
                   x = (i+0.5)*step;
                   sum = sum + 4.0/(1.0+x*x);
            pi = step * sum;
```

```
#include <omp.h>
static long num+staps = 100000;
                                     double step;
void main ()
       int i; double x, pi, sum = 0.0;
       step = 1.0/(double) num_steps;
       #pragma omp parallel
          double x;
          #pragma omp for reduction(+:sum)
             for (i=0;i<num_steps; i++){
                   x = (i+0.5)*step;
                   sum = sum + 4.0/(1.0+x*x);
            pi = step * sum;
  Threads 1st SPMD 1st SPMD
                               SPMD
                                        Pi Loop
                     Padded
                               Critical
            1.86
                      1.86
                                1.87
                                         1.91
            1.03
                      1.01
                                1.01
                                         1.02
            1.08
                      0.69
                                0.68
                                         0.80
            0.97
                      0.53
                                0.53
                                         0.68
```

```
#pragma omp parallel shared (A, B, C) private(id)
{
    id=omp_get_thread_num();
    A[id] = big_calc(id);
#pragma omp barrier
#pragma omp for
    for(i=0;i<N;i++){C[i]=big_calc3(i,A);}
#pragma omp for nowait
    for(i=0;i<N;i++){ B[i]=big_calc2(C, i); }
    A[id] = big_calc4(id);
}</pre>
```

Master 10

```
#pragma omp parallel
{
          do_many_things();
#pragma omp master
          { exchange_boundaries(); }
#pragma omp barrier
          do_many_other_things();
}
```

```
#pragma omp parallel
{
          do_many_things();
#pragma omp single
          { exchange_boundaries(); }
          do_many_other_things();
}
```

https://www.youtube.com/watch?list= PLLbPZJxtMs4ZHSamRRYCtvowRSOqIwC-I От "Introduction to OpenMP 08 Discussion 3" до "Introduction to OpenMP 11 part 1 Module 6".