Universidad de Costa Rica

Escuela de Ciencias de la Computación e Informática

Programación Paralela y Concurrente

CI0117

Luis Eduardo Rojas Carrillo - B86875

Semana #9

1-)

Compilar y correr el programa "omp_hello.c".

Salida:

Hello from thread 1 of 5

Hello from thread 0 of 5

Hello from thread 4 of 5

Hello from thread 3 of 5

Hello from thread 2 of 5

Explicar por qué la salida del programa varía.

La salida varía ya que los hilos no tienen un orden, por lo tanto unos pueden terminar antes que los otros hilos en diferentes ejecuciones.

2-)

Compilar y correr el programa "omp_fibo.c".

Salida:

The first n Fibonacci numbers:

0 1

1 1

2 2

3 3

4 5

5 8

6 13

7 21

8 34

9 55

Explique la razón por lo que usted piensa el programa no funciona.

El programa no funciona ya que hay una “condición de carrera”, algunos hilos podrían usar datos del vector que no han sido procesados.

Anote el respaldo teórico de su razonamiento.

```
for (i = 2; i < n; i++)
    fibo[i] = fibo[i-1] + fibo[i-2];
```

En este momento se pueden utilizar casillas del vector que no han sido procesados por los otros hilos.

3-)

Compilar y correr el programa "omp_private.c".

Salida:

3Thread 1 > before initialization, x = 0

Thread 1 > after initialization, x = 4

Thread 2 > before initialization, x = 0

Thread 2 > after initialization, x = 6

Thread 0 > before initialization, x = 0

Thread 0 > after initialization, x = 2

After parallel block, x = 5

Explique el funcionamiento de hacer la variable x privada.

El funcionamiento de hacer las variables privadas permite a cada hilo trabajar con la variable de manera independiente sin variar el valor inicial de la variable “x” en este caso.

4-)

Compilar y correr el programa "omp_trap1.c".

Con critical:

Salida:

Enter a, b, and n

3 4 20

With n = 20 trapezoids, our estimate

of the integral from 3.000000 to 4.000000 = 1.23337500000000e+01

Elimine "#pragma omp critical".

Sin critical:

Salida:

Enter a, b, and n

3 4 20

With n = 20 trapezoids, our estimate

of the integral from 3.000000 to 4.000000 = 1.23337500000000e+01

luis@luis-VirtualBox:~/Escritorio/ipp-source/ipp-source-use/ch5$

Comente sobre el problema encontrado.

En este caso luego de ejecutarlo de diferentes maneras y en varias ocasiones, los valores no variaron, se mantuvieron igual.

5-)

Compilar y correr el programa "omp_trap2a.c".

Salida:

Enter a, b, and n

3 4 20

With n = 20 trapezoids, our estimate

of the integral from 3.000000 to 4.000000 = 1.23337500000000e+01

Anote si el resultado es correcto.

Sí, es correcto.

6-)

Compilar y correr el programa "omp_trap2b.c".

Salida:

Enter a, b, and n

3 4 20

With n = 20 trapezoids, our estimate

of the integral from 3.000000 to 4.000000 = 1.23337500000000e+01

Anote si el resultado es correcto.

Sí, es correcto.

7-)

Compilar y correr el programa "omp_trap3.c".

Salida:

Enter a, b, and n

3 4 20

With n = 20 trapezoids, our estimate

of the integral from 3.000000 to 4.000000 = 1.23337500000000e+01

luis@luis-VirtualBox:~/Escritorio/ipp-source/ipp-source-use/ch5$

Anote si el resultado es correcto.

Sí, es correcto.

8-)

Compilar y correr el programa "omp_pi.c".

Salida:

With n = 300 terms and 3 threads,

Our estimate of pi = 3.13825932951559

pi = 3.14159265358979

Explicar la manera que se realiza la acumulación de la suma.

La suma se realiza en una variable global suma en la cual los hilos tienen un reduction y un private factor mediante el cual previene las condiciones de carrera.

9-)

Compilar y correr el programa "bubble.c".

Salida:

1000 g

real 0m0,004s

user 0m0,004s

sys 0m0,000s

Usando valgrind.

Salida:

==2164== Memcheck, a memory error detector

==2164== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.

==2164== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info

==2164== Command: ./bubble 10 g

==2164==

==2164==

==2164== HEAP SUMMARY:

==2164== in use at exit: 0 bytes in 0 blocks

==2164== total heap usage: 2 allocs, 2 frees, 1,064 bytes allocated

==2164==

==2164== All heap blocks were freed -- no leaks are possible

==2164==

==2164== For counts of detected and suppressed errors, rerun with: -v

==2164== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

Interprete la salida y anote los errores que encuentre.

Establece un resumen del uso del heap, muestra los dos alloc y los dos frees de estos dos allocs y menciona que todos los bloques de memoria en el heap fueron liberados.

10-)

Compilar y correr el programa "omp_odd_even1.c".

Salida:

10 1000 g

Elapsed time = 7.419503e-02 seconds

real 0m0,075s

user 0m0,010s

sys 0m0,029s

Con 1 hilo.

Salida:

Result = 1.88892327977904e+00

Check = 1.88892327977966e+00

With n = 10000 terms, the error is 6.23945339839338e-13

Elapsed time = 1.517374e+00 seconds

real 0m2,841s

user 0m2,719s

sys 0m0,000s

Con 2 hilos.

Result = 1.88892327977898e+00

Check = 1.88892327977905e+00

With n = 10000 terms, the error is 7.28306304154103e-14

Elapsed time = 1.324470e+00 seconds

real 0m2,496s

user 0m2,420s

sys 0m0,000s

Speedup.

1.4s

OMP_SCHEDULE="dynamic".

Result = 1.88892327977904e+00

Check = 1.88892327977905e+00

With n = 10000 terms, the error is 1.24344978758018e-14

Elapsed time = 1.430016e+00 seconds

real 0m2,840s

user 0m2,815s

sys 0m0,000s

OMP_SCHEDULE="auto".

Result = 1.88892327977902e+00

Check = 1.88892327977905e+00

With n = 10000 terms, the error is 2.53130849614536e-14

Elapsed time = 1.257185e+00 seconds

real 0m2,430s

user 0m2,349s

sys 0m0,000s

OMP_SCHEDULE="static".

Result = 1.88892327977910e+00

Check = 1.88892327977905e+00

With n = 10000 terms, the error is 4.79616346638068e-14

Elapsed time = 1.518467e+00 seconds

real 0m2,750s

user 0m2,552s

sys 0m0,000s

Diferencias:

OMP_SCHEDULE="auto" fue el que realizó la tarea de una manera más eficaz en comparación con los otros dos.

12-)

Compilar y correr "omp_msgps.c".

Thread 0 > received -1 from 0

Thread 1 > received 0 from 0

Thread 1 > received -2 from 0

Thread 1 > received -3 from 0

Thread 1 > received -4 from 0

Thread 1 > received -5 from 0

Thread 1 > received -2 from 1

Thread 1 > received -3 from 1

Thread 1 > received -5 from 1

Thread 0 > received 0 from 1

Thread 0 > received -1 from 1

Thread 0 > received -4 from 1

Analice la salida y describa que está sucediendo.

Los hilos se envían como mensaje los datos ingresados anteriormente aleatoriamente.

Busque e indique la funcionalidad de "barrier" y "atomic" en este programa.

El barrier permite que los procesos esperen a los otros procesos, generando como un estilo de pausa hasta que todos lleguen al mismo punto.

El atomic permite accesar a la memoria de manera automática, siendo más eficiente para evitar condiciones de carrera.

13-)

Compilar y correr el programa "omp_mat_vect.c".

Salida 1.

10 100 200

Elapsed time = 3.856080e-04 seconds

Salida 2.

10 100 100

Elapsed time = 2.397090e-04 seconds