Universidad del Valle de Guatemala
Computación paralela y distribuida
Fredy Velásquez
201011

# Hoja de trabajo #1

## Ejercicio #1

```
Hello from thread 1 of 10!
Hello from thread 2 of 10!
Hello from thread 0 of 10!
Hello from thread 3 of 10!
Hello from thread 4 of 10!
Hello from thread 5 of 10!
Hello from thread 7 of 10!
Hello from thread 6 of 10!
Hello from thread 9 of 10!
Hello from thread 8 of 10!
```

**PREGUNTA: ¿Por qué al ejecutar su código los mensajes no están desplegados en orden?**

La intención de usar programación paralela es ejecutar tareas en paralelo para mejorar el rendimiento y la eficiencia, pero esto a menudo puede resultar en un orden de ejecución no determinista.

## Ejercicio #2

```
Feliz cumpleaños número 1 !
Feliz cumpleaños número 21 !
Feliz cumpleaños número 3 !
Saludos del hilo 8
Saludos del hilo 2
Feliz cumpleaños número 13 !
Feliz cumpleaños número 17 !
Feliz cumpleaños número 7 !
Feliz cumpleaños número 5 !
Saludos del hilo 4
Saludos del hilo 12
Feliz cumpleaños número 9 !
Saludos del hilo 10
Saludos del hilo 0
Feliz cumpleaños número 15 !
Saludos del hilo 14
Saludos del hilo 18
Saludos del hilo 20
Saludos del hilo 16
Feliz cumpleaños número 19 !
Saludos del hilo 6
Feliz cumpleaños número 11 !
```

Ejercicio #3

```
[Running] cd "/Users/freddyvelasquez/Downloads/HDTI-Paralela-main/" && gcc Riemann.
Con n = 10000000 trapezoides, nuestra aproximacion
de la integral de 1.000000 a 40.000000 es = 21333.0000000017
```

Ejercicio #4

```
n = 1000000 trapezoides, aproximación de la integral en el intervalo (1.000000, 10.000000) es 57.903977
```

**PREGUNTA: ¿Por qué es necesario el uso de la directiva *#pragma omp critical*?**

En el programa se utiliza la directiva #pragma omp critical para proteger la variable resultado dentro de la región paralela. Dado que varios hilos están actualizando esta variable simultáneamente, existe un riesgo de condición de carrera, donde los hilos podrían pisarse mutuamente y causar resultados incorrectos.

La región crítica en este programa se encuentra dentro del bloque #pragma omp single, que es una construcción de OpenMP que asegura que solo un hilo ejecute el bloque de código dentro de él. Sin embargo, dentro de ese bloque, hay un bucle en el que varios hilos realizan operaciones de suma en la variable pool_threads.

Ejercicio #5

```
Resultado de la integral con n = 1000000 trapezoides, integral de (2.000000, 4.000000) es 10.666667
Resultado de la integral con n = 1000000 trapezoides, integral de (2.000000, 4.000000) es 34.666667
Resultado de la integral con n = 1000000 trapezoides, integral de (2.000000, 4.000000) es -0.785398
```

**PREGUNTA: ¿Qué diferencia hay entre usar una variable global para añadir los resultados a un arreglo?**

Usando una Variable Global:

En este enfoque, se reserva un espacio de memoria global (partial_results) que es accesible por todos los hilos. Cada hilo calcula su resultado parcial y lo almacena en la posición correspondiente del arreglo global. Luego, después de que todos los hilos hayan completado su cálculo, se suma cada uno de los resultados parciales almacenados en el arreglo global para obtener el resultado final.

Ventajas:

- Es relativamente sencillo de implementar.
- Los hilos pueden acceder y actualizar la variable global directamente.

Desventajas:

- Requiere sincronización explícita para evitar condiciones de carrera y garantizar que los hilos no interfieran entre sí al escribir en el arreglo global.
- Puede haber un impacto en el rendimiento debido a la necesidad de sincronización y posible contención entre los hilos al acceder a la variable global.

Sin Usar una Variable Global:

En lugar de utilizar una variable global, cada hilo podría almacenar su resultado parcial en una variable local, y luego sumar estos resultados parciales locales en una sección crítica o utilizar la directiva #pragma omp critical para asegurarse de que solo un hilo a la vez actualice la variable global que almacena el resultado final.

Ventajas:

- Reduce el riesgo de condiciones de carrera, ya que los resultados parciales se almacenan en variables locales separadas para cada hilo.
- Puede mejorar el rendimiento al reducir la necesidad de sincronización global.

Desventajas:

- Requiere un poco más de manejo de sincronización, ya que los hilos deben coordinarse para sumar sus resultados parciales correctamente.

## Repositorio

**https://github.com/fredyvelasquezgt/HT-1-PARALELA/tree/main**