UNIVERSIDAD DEL VALLE DE GUATEMALA

Guatemala 25 de agosto de 2024
María José Yee Vidal - 231193

CC3086 - Programación de Microprocesadores
Laboratorio 05

Competencias para desarrollar
Distribuir la carga de trabajo entre hilos utilizando programación en C y OpenMP.

Instrucciones
Esta actividad se realizará individualmente. Al finalizar los períodos de laboratorio o clase, deberá
entregar este archivo en formato PDF y los archivos .c en la actividad correspondiente en Canvas.

Enlace github:
https://github.com/Yee404/Yee-CC3086-2024.git

1. (18 pts.) Explica con tus propias palabras los siguientes términos:
a) private
Crea una copia independiente de una determinada variable para cada hilo dentro de una región paralela, donde cada hilo trabaja con su propia versión de esta variable.

b) shared
Permite que una variable sea accedida y modificada por todos los hilos dentro de una región paralela, por lo que todos los hios operan sobre la misma instancia de la variable.

c) firstprivate
Le asigna a cada hilo una copia privada y única de la variable, la cual se inicializa con ek valor de la variable original que se le asignó antes de entrar a la región paralela.

d) barrier
Es un punto de sincronización donde todos los hilos deben esperar que que los demás hilos hayan llegado a un mismo punto para poder continuar con el código, garantizando que todos los hilos hayan terminado antes de continuar.

e) critical
Asegura que un único hilo puede ejecutar una sección del código a la vez, logrando evitar conflictos al modificar datos/variables compartidos.

f) atomic
Asegura que una operación de lectura-modificación-escritura en una variable compartida sea realizada de manera indivisible y segura sin que le interrumpan otros hilos.

2. (12 pts.) Escribe un programa en C que calcule la suma de los primeros N números naturales utilizando un ciclo for paralelo. Utiliza la cláusula reduction con + para acumular la suma en una variable compartida.
a) Define N como una constante grande, por ejemplo, N = 1000000.
b) Usa omp_get_wtime() para medir los tiempos de ejecución.

3. (15 pts.) Escribe un programa en C que ejecute tres funciones diferentes en paralelo usando la directiva #pragma omp sections. Cada sección debe ejecutar una función distinta, por ejemplo, una que calcule el factorial de un número, otra que genere la serie de Fibonacci, y otra que encuentre el máximo en un arreglo, operaciones matemáticas no simples. Asegúrate de que cada función sea independiente y no tenga dependencias con las otras.

4. (15 pts.) Escribe un programa en C que tenga un ciclo for donde se modifiquen dos variables de manera paralela usando #pragma omp parallel for.
a. Usa la cláusula shared para gestionar el acceso a la variable1 dentro del ciclo.
b. Usa la cláusula private para gestionar el acceso a la variable2 dentro del ciclo.
c. Prueba con ambas cláusulas y explica las diferencias observadas en los resultados.

5. (30 pts.) Analiza el código en el programa Ejercicio_5A.c, que contiene un programa secuencial. Indica cuántas veces aparece un valor key en el vector a. Escribe una versión paralela en OpenMP utilizando una descomposición de tareas recursiva, en la cual se generen tantas tareas como hilos.