Universidad del Valle de Guatemala
Facultad de Ingeniería
Departamento de Ciencias de la Computación
CC3086 Programación de microprocesadores

Ciclo 2 de 2,024
Temario A

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.

1. **(18 pts.)** Explica con tus propias palabras los siguientes términos:

   a) **private:** Esta cláusula hace que cada hilo tenga su propia copia de una variable. Así, cada hilo trabaja de forma independiente con su copia, sin interferir con las copias de otros hilos.

   b) **shared:** Con esta cláusula, todos los hilos comparten la misma variable. Esto puede ser útil, pero también arriesgado porque varios hilos pueden intentar modificar la variable al mismo tiempo.

   c) **firstprivate:** Es similar a private, pero con la diferencia de que cada hilo comienza con una copia de la variable inicializada al valor que tenía antes de entrar en la región paralela.

   d) **barrier:** Una barrera es un punto donde todos los hilos se detienen hasta que todos hayan llegado. Esto asegura que ningún hilo avance hasta que todos los hilos hayan terminado las tareas previas.

   e) **critical:** Uso esta directiva para proteger secciones de código que no deben ser ejecutadas por más de un hilo al mismo tiempo. Así evito problemas de concurrencia.

   f) **atomic:** Similar a critical, pero más eficiente para operaciones simples como sumar o restar. Asegura que solo un hilo realiza la operación en un momento dado, evitando conflictos.

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.