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 : cláusula que se utiliza para especificar que cada hilo debe tener su propia copia local de una variable.

   b) shared : cláusula que indica que una variable será compartida entre todos los hilos en una región paralela. Todos los hilos pueden acceder y modificar esta variable

   c) firstprivate : similar a "private", pero inicializa la copia local de cada hilo con el valor que tenía la variable original antes de entrar en la región paralela.

   d) barrier : punto de sincronización en el código donde todos los hilos deben llegar antes de que cualquiera pueda continuar.

   e) critical : región de código que solo puede ser ejecutada por un hilo a la vez

   f) atomic : directiva que garantiza que una operación específica se realice .

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.

c.

- shared_var: Al ser compartida, todos los hilos acceden y modifican la misma variable. Aunque esto puede llevar a condiciones de carrera, en este caso simple la suma es asociativa y conmutativa, por lo que el resultado final es correcto.
- private_var: Cada hilo tiene su propia copia de esta variable. Las modificaciones se hacen en estas copias locales, que se descartan al finalizar el ciclo paralelo. El valor impreso es el de la variable original en el hilo principal, que no se modificó.

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.

6. **REFLEXIÓN DE LABORATORIO: se habilitará en una actividad independiente.**