Computación Concurrente 2017-2. Práctica 2: Sincronización de Hilos y Paso de Argumentos en Pthreads.

José David Flores Peñaloza Susana Hahn Martín Lunas
Armando Ballinas Nangüelú

Fecha de Entrega: 16 de febrero de 2017.

1. Introducción y Objetivo.

Una vez que se está familiarizado con lo esencial de la biblioteca Pthreads se estudiará una parte fundamental para la implementación de algoritmos concurrentes que es la sincronización de procesos.

Cada biblioteca, ambiente de trabajo o lenguaje de programación ofrece diversos y distintos mecanismos que permiten organizar el trabajo realizado por varios hilos de manera concurrente. Estos mecanismos incluyen: semáforos, candados, monitores, variables de condición, etc.

La biblioteca Pthreads cuenta, en particular con objetos `mutex` que mediante funciones de captura (*lock*) y liberación (*unlock*) permiten resolver el problema de la exclusión mutua y compartir los recursos de manera correcta.

En esta práctica se diseñará un programa que calcule el producto punto de dos vectores en una forma paralela. El producto punto es una función que toma dos vectores en un espacio vectorial de dimensión d y devuelve un elemento del campo. Estos vectores se modelarán mediante arreglos de números de punto flotante de doble precisión y el elemento del campo también será un número de punto flotante de doble precisión.

2. Exclusión Mutua y Sincronización.

Para realizar la práctica se debe utilizar un objeto `mutex` y las funciones de `lock` y `unlock` para capturarlo y liberarlo de manera correcta. Para más detalles acerca de como utilizar los objetos `mutex` deben consultar el tutorial de la práctica anterior:
`https://computing.llnl.gov/tutorials/pthreads/`

3. Paso de argumentos a hilos.

La biblioteca permite solo pasar un argumento de tipo `void *` a un hilo, por lo que se debe idear la manera de que los hilos puedan utilizar todo lo que necesitan, una manera de hacer esto es declarar variables en modo global o englobar todos los atributos necesarios en una estructura.

4. Ejercicios.

Para el envío de los ejercicios deben adjuntarlos todos, junto con un archivo Leeme.txt o Readme.txt en una carpeta con su nombre y apellido y comprimida en .zip o .rar. Debe mandar la carpeta por correo a la dirección armandoballinas@ciencias.unam.mx. El asunto del correo debe ser *Concurrente_p2*. Si el asunto es el correcto y hay archivos adjuntos se les enviará una respuesta automática indicando que se recibió la práctica.

El archivo Leeme.txt debe tener su nombre completo y número de cuenta, así como las respuestas a las preguntas solicitadas.

1. Se debe rehacer el ejercicio del producto punto presentado en el tutorial, de modo que se permitan pasar como argumentos en línea de comandos el número de hilos a crear, así como la dimensión del vector. Este ejemplo utiliza objetos mutex para la sincronización. También utiliza el mecanismo `join` para hacer que el hilo principal espere mientras los demás hilos realizan el trabajo. Realizar este ejercicio en un archivo `dot_example.c`

2. Modificar el ejercicio anterior para permitir que los vectores sean llenados de manera pseudoaleatoria, la semilla del generador de números

pseudoaleatorios debe ser pasada como argumento también. Además en esta versión el hilo principal no debe esperar mediante el mecanismo `join`, sino que los demás hilos deben poder avisarle cuando ya hayan terminado el trabajo. Consejo: considere usar una variable compartida como en el sistema de tickets y hacer que el hilo `main` ejecute un ciclo hasta que la variable compartida tenga cierto valor. Para este ejercicio el hilo principal debe calcular de manera secuencial el mismo producto punto para poder comparar los resultados con la versión en paralelo. Esto lo puede hacer, antes o después de esperar. Realizar este ejercicio en un archivo llamado `dot_product.c`.

3. La biblioteca provee otro mecanismo de sincronización llamado *variables de condición*. Explique en el README qué son y cómo usarlas.