PROGRAMACIÓN AVANZADA

Tecnológico de Monterrey, Campus Querétaro

Actividad colaborativa - Manejo de threads

En esta actividad, debes desarrollar una versión miltihilos de la multiplicación de matrices. Esta versión usa hilos para dividir el trabajo necesario para calcular el producto de dos matrices. Existen varias formas de mejorar el rendimiento utilizando multihilos. Un ejemplo, dividir el cálculo en la dimensión de los renglones entre varios hilos. Es decir, si estamos calculando $A * B$, siendo A y B matrices cuadradas de `10 x 10`, la solución debe usar hilos para dividir el cálculo de los renglones del producto (que resulta ser una matriz final `10 x 10`). Si contáramos con 5 hilos, los renglones 0 y 1 del producto se calculan con el hilo 0, los renglones 2 y 3 se calculan con el hilo 1, ... y los renglones 8 y 9 se calculan con el hilo 4.

Si la matriz del producto tuviera 100 renglones, entonces cada "segmento"de la matriz tendría 20 renglones para asignar a un hilo. Esta forma de paralelización se denomina "descomposición en tira"(strip decomposition) de la matriz. Ten en cuenta que el número de hilos puedo no dividir el número de renglones uniformemente. Por ejemplo, si tenemos una matriz del producto de 20 renglones y estamos usando 3 hilos, alguno deberá calcular más renglones que otros. En este caso, habría 2 renglones extra. Una buena solución es asignar 7 renglones a un hilo, otras 7 a otro hilo y 6 renglones al último. Ten en cuenta que para esta asignación se debe pasar las matrices A y B a cada hilo.

Escribe el programa `multiply` que multiplica dos matrices. El programa recibe de línea de comandos:

```
$ .\mutiply -a matrix_file_a.txt -b matrix_file_b.txt -t thread_amount
```

Los archivos de entrada son archivos de texto que tiene el siguiente formato: la primera linea tiene dos números, `N` renglones y `M` columnas. Las siguientes `N` línea contienen `M` números flotantes. Por ejemplo:

```
3
1.0 2.0 3.0
4.0 5.0 6.0
7.0 8.0 9.0
```

Puedes utilizar el programa `create_matrix.c` para generar matrices de prueba.

Tu programa deberá imprimir el resultado de la multiplización en pantalla. La matriz resultante debe ser impresa en el mismo formato que se utiliza en los archivos de texto. Tu solución también debe ser cronometrada. Si tu implementación utiliza correctamente varios hilos, se debe esperar una mejora en el tiempo requerido para hacer el cálculo completo. Por ejemplo, en alguna maquina ejecutamos:

```
$ .\mutiply -a a1000.txt -b b1000.txt -t 1
```

quizás se complete en 8.8 segundos cuando utilizamos los archivos de entrada a1000.txt y b1000.txt, mientras que si ejecutamos:

```
$ .\mutiply -a a1000.txt -b b1000.txt -t 2
```

quizás se complete en 4.9 segundos. Estos resultados los debes reportar como comentario en tu envío.

Rúbrica de evaluación:

Ponderación	
+5 puntos	Verifica la cantidad correcta de parámetros. En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -2 como resultado de su ejecución.
+10 puntos	Verifica que parámetros sean los correctos (`-a`, `-b`, `-t`). En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -3 cuando falte `-a`, regresando -4 cuando falte `-b`, regresando -5 cuando falte`-t`.
+5 puntos	Verifica que loas archivos existan. En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -6 para el `matrix_file_a` y -7 para el `matrix_file_b`.
+5 puntos	Verifica que `threads_amount` sea un número entero valido (mayor a 0). En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -8 como resultado de su ejecución.
+5 puntos	Verifica el tamaño de las matrices sean idénticos. En caso de que no sea así, el programa despliega un mensaje adecuada y termina, regresan -9 como resultado de su ejecución.
+50 puntos	Se imprime el resultado correcto en pantalla.
+10 puntos	El reporte de los tiempos de ejecución vienen en los comentarios.