ARQO

Memoria Práctica 4

Roberto Martín Alonso

Diego Forte Jara

Pareja 11

ÍNDICE

Ejercicio 1……………………………………………………………………………………...…………4

Ejercicio 2……………………………………………………………………………….…….…………5

Ejercicio 3……………………………………………………………………………….……………….6

Ejercicio 4……………………………………………………………………………….……………….7

Ejercicio 5……………………………………………………………………………….……………….8

# Aclaraciones previas:

La práctica actual se va a desarrollar haciendo uso del subsistema de Windows para Linux (WSL), con las siguientes especificaciones:

Texto

Descripción generada automáticamente

Los ficheros entregados están distribuidos de la siguiente forma:

* En el directorio “Codigo” están todos los ficheros fuente base de los programas desarrollados en la práctica, ficheros .c, .h y makefile.
* En los directorios Ex (Donde x es el número de ejercicio) se encuentran los ficheros .c, .dat, .png y los scripts correspondientes a cada ejercicio. **Para ejecutar un script se debe pasar dicho script al interior del directorio “Codigo” junto a los ficheros .c (solo aplica para los ejercicios 4 y 5)**
* En el directorio “Docs” se encuentra la memoria de la práctica.

# Ejercicio 0: Información sobre la topología del sistema

Tras ejecutar el comando “cat /proc/cpuinfo” por terminal y volcarlo al fichero de texto “cpuinfoE0.txt” se obtienen los siguientes resultados:

Texto

Descripción generada automáticamente

Aquí se puede apreciar que el número de cores físicos es 8 (campo cpu cores), el número de cores lógicos es 16 (campo siblings) y su frecuencia es 4200.16MHz (campo cpu MHz). Podemos concluir que el hyperthreading está activado en el sistema debido a que el número de cores lógicos es mayor que el de cores físicos.

NOTA: En la carpeta Docs de la entrega se adjunta el fichero “cpuinfoE0.txt” donde se puede ver toda la información completa tras ejecutar el comando “cat /proc/cpuinfo”

# Ejercicio 1: Información sobre la topología del sistema

# 1.1-

El sistema en el que se realiza la práctica dispone de 16 cores lógicos (16 hilos máximo). Se prueba a lanzar el programa “omp1.c” con 32 hilos y el programa funciona, por lo que es posible lanzar más hilos que cores lógicos.

Con respecto a si tiene sentido hacerlo, en general no debido a que el sistema operativo debe gestionar el cambio de contexto entre hilos y produce sobrecarga en el sistema, pero en situaciones donde se produzcan un gran número de bloqueos si debido a que mientras que haya hilos bloqueados puede haber otros trabajando.

# 1.2-

El número de hilos a utilizar depende de la tarea que se quiera realizar. Se podría comenzar usando el número máximo de hilos del sistema ( “” para el ordenador del laboratorio y 16 hilos para el ordenador propio) y posteriormente experimentar con más hilos y medir el rendimiento para ver si hay mejoras.

# 1.3-

Se modifica el programa “omp1.c” tal y como se pide en el enunciado y se ejecuta. A partir de esta ejecución se deduce que la prioridad del número de hilos es, de forma descendente:

nº hilos cláusula -> nº hilos con función -> nº hilos variable de entorno

NOTA: Si no se indica número de hilos mediante cláusula o función, OpenMP utiliza el valor de la variable de entorno OMP\_NUM\_THREADS, por lo que para que se ejecute con el valor de dicha variable de entorno no se le asigna número de hilos en la ejecución con el valor de la variable de entorno

# 1.4-