Computación Paralela

Profesor Responsable: Héctor Fco Migallón Gomis hmigallon@umh.es			
Profesor de Laboratorio: Héctor Fco Migallón Gomis hmigallon@umh.es			
Departamento: INGENIERÍA DE COMPUTADORES			
Área de Conocimiento: Arquitectura y Tecnología de Computadores			
Curso: 3º	**Docencia:** 1Sem.	**Tipo:** Obligatoria	**Créditos: 6** ECTS (60 + 90 horas)
Página web de la asignatura: (institucional)			

• PRACTICA 1

El objetivo de esta práctica es programar con el paradigma MPI para sistemas de memoria distribuida. Se deben realizar dos tareas y analizar su comportamiento mediante ejecuciones variando el número de procesos. Las dos tareas son:

Se leerá una imagen en tonos de grises almacenada en formato raw, es decir los datos serán de tipo unsigned char y el tamaño del fichero será de altura x anchura x 1byte. La altura y la anchura podrán ser diferentes.

La imagen leída se almacenará en una matriz bidimensional por el proceso root, las imágenes procesadas se almacenarán en ficheros binarios en formato raw, proceso realizado por el proceso root.

Procesados a implementar (se escogerá uno pasando parámetro):

Filtrado por media (3 x 3 elementos)
Filtrado por mediana (3 x 3 elementos)
Detección de bordes (SOBEL)

C

-1	0	1
-2	0	2
-1	0	1

F

-1	-2	-1
0	0	0
1	2	1

$$F_{ij} = \text{sqrt}\ (C^2 + F^2)$$

El fichero texto de salida, generado por el cero tendrá la siguiente información: nombre de fichero de entrada, tamaño de la imagen, nombre de fichero de salida, número de procesos utilizado, tiempo paralelo.

Aquellos elementos que no disponen de los 8 vecinos SI serán considerados, realizando extensión simétrica tanto para filas (la fila -1 será igual a la 1, la fila N será igual a la N-2, siendo N el número de filas) como para columnas.

Los nombres de los ficheros y resto de parámetros necesarios serán pasados como argumentos en la sentencia de ejecución.

Como se ha dicho, el proceso “root” será el encargado de leer y escribir ficheros, el proceso “root” almacenará toda la información y distribuirá un conjunto de filas a cada proceso “slave”, recibirá los datos de cada proceso “slave” y la escribirá en fichero. Se realizará un análisis de speed-up y eficiencia con imágenes de 4096x4096.

Tarea 2. Sistema iterativo

Desarrollar un programa que implemente el siguiente esquema iterativo para k=0..m. Teniendo en cuenta que el vector x^0 será el vector unidad (= 1,1,....1). El valor del número de iteraciones (m) se especificará como argumento. El tamaño de los vectores será igual a N=15000.

Iteración 1:	$x^1 = M\ x^0$
Iteración 2:	$x^2 = M\ x^1$; Convertir x^2 a valores entre [-1,1], buscar el mayor valor absoluto de x^2, y dividir todos los elementos por dicho valor.
Iteración 3:	$x^3 = M\ x^2$; Convertir x^3 a valores entre [-1,1], buscar el mayor valor absoluto de x^3, y dividir todos los elementos por dicho valor.

…

Los valores de M, que es una matriz cuadrada de tamaño NxN con elementos iguales a 1 en la diagonal y el resto se generarán aleatoriamente o pseudoaleatoriamente y estarán comprendidos TODOS excepto la diagonal entre -50.0<x<50.0, de forma que los elementos de la mitad triangular inferior serán positivos y de la triangular superior negativos. Y podrán estar almacenados en un fichero binario.

El número de iteraciones (m), el nombre del fichero de salida (fichero texto) y el nombre del fichero donde estará almacenada la matriz M serán pasados en sentencia de ejecución. M será una matriz reservada fila a fila, es decir puntero doble. Si el fichero donde está la matriz M no existe se genera y después de generarla se guardará en un fichero para poder ser reutilizada.

Los elementos de la matriz y los vectores son de tipo double. Sólo el proceso root lee y escribe ficheros.

El fichero texto de salida, generado por el "root" tendrá la siguiente información: nombre de fichero de entrada o salida, nombre de fichero de salida, elementos de mayor valor absoluto obtenidos, número de procesos utilizado, tiempo paralelo considerando tratamiento de ficheros, comunicaciones iniciales y ejecución, tiempo paralelo considerando comunicaciones iniciales y ejecución, tiempo paralelo considerando sólo ejecución. Se realizará un análisis de speed-up y eficiencia

IMPORTANTE: Usar los recursos (memoria) estrictamente necesarios.

ENTREGA EN AMBOS CASOS:

Entregar un fichero comprimido que incluya:

Ficheros .c del código o códigos (COMPLETAMENTE COMENTADOS)
Ficheros texto de salida de todas las opciones (media, mediana, sobel, iterativo con m=5, para 1 y 4 procesos). Son 8 ficheros en total.
Fichero texto con la sentencia de compilación (un fichero por tarea).
Fichero texto con el script de lanzamiento (un fichero por tarea)
Tabla de resultados de análisis de speed-up y eficiencia (Tiempo, speed-up y eficiencia de 1 a 4 procesos) de todas las opciones (media, mediana, sobel, iterativo con m=5).