TRABAJO GLOBAL INTEGRADOR*

En grupo de 2 o 3 integrantes, elegir uno de los siguientes problemas* (al final del listado de consignas se define brevemente cada uno de estos temas):

1) Multiplicación de matrices
2) Algoritmo de ordenamiento Merge Sort
3) Algoritmo de ordenamiento Quick Sort

Una vez determinado el tema, desarrollar los incisos que se proponen a continuación a fin de desarrollar el ciclo de vida completo de la versión paralela, de acuerdo a las siguientes consignas:

a) Diseñar e implementar la versión secuencial. Analizar el algoritmo elegido e identificar las posibles porciones de código o datos que podrían ser factibles de paralelizarse. Realizar la entrega correspondiente al análisis realizado, y la versión secuencial, a través del Aula Abierta.
b) Diseñar y desarrollar un algoritmo paralelo que resuelva el problema elegido, documentando las estrategias y decisiones de diseño e implementación consideradas. Validar el funcionamiento del programa. Realizar la entrega correspondiente al diseño paralelo, a través del Aula Abierta.
c) Realizar el diseño experimental correspondiente. Efectuar la entrega correspondiente al diseño experimental, a través del Aula Abierta.
d) Explicar las características de la solución de software realizada. Incluir análisis de **speedup**, **eficiencia**, **balanceo de carga** y **escalabilidad**. Presentar un informe que contenga estos resultados y análisis, a través del Aula Abierta.
e) (*Opcional*) Iterar sobre el diseño e implementación alcanzados, en base al resultado de la evaluación de rendimiento, a fin de implementar alguna mejora. Documentar las mejoras y la comparativa, y realizar la entrega a través del Aula Abierta.
f) (*Opcional*) Preparar un artículo científico con los desarrollos alcanzados para su presentación en congreso. Realizar la entrega a través del Aula Abierta.
g) Realizar una presentación oral del desarrollo alcanzado y los resultados, de acuerdo a las pautas y fechas que oportunamente se indicarán. Entregar una versión final del informe siguiendo el formato que oportunamente se indicará, que condense todo el proceso de desarrollo realizado. La entrega se realizará a través del Aula Abierta y asimismo en versión impresa.

TEMA 1: Desarrollar un algoritmo paralelo que resuelva la multiplicación de matrices, con la forma C = A∗B, con A y B no necesariamente cuadradas.

> *Conceptualmente, la multiplicación de matrices funciona de la siguiente manera: Dadas una matriz A(MxR) de M filas y R columnas, donde cada uno de sus elementos se denota* $\boldsymbol{a_{ij}}$ *con* $1 \leq \boldsymbol{i} \leq \boldsymbol{M}$, y $1 \leq \boldsymbol{j} \leq \boldsymbol{R}$; y una matriz B($R \times N$) *de R filas y N columnas, donde cada uno de sus elementos se denota* $\boldsymbol{b_{ij}}$ *con* $1 \leq \boldsymbol{i} \leq \boldsymbol{R}$, *y* $1 \leq \boldsymbol{j} \leq \boldsymbol{N}$; *la matriz C resultante de la operación de*

multiplicación de las matrices **A y B, C = A × B**, *es tal que cada uno de sus elementos que se denota como* c_{ij} *con* $1 \leq i \leq M$, y $1 \leq j \leq N$, *y se calcula de la siguiente manera:*

$$c_{ij} = \sum_{k=0}^{R-1} a_{ik} b_{kj}$$

TEMA 2: Desarrollar un algoritmo paralelo que resuelva el ordenamiento denominado Merge Sort para una lista de elementos. Cada elemento debe ser representado mediante una estructura con varios campos, uno de los cuales se considera campo "clave", es decir que será el campo a considerar a la hora de efectuar el ordenamiento. Desarrollar un programa que implemente dicho algoritmo paralelo.

Conceptualmente, el ordenamiento ***Merge Sort*** *funciona de la siguiente manera:*

1. *Si la longitud de la lista es 0 ó 1, entonces ya está ordenada. En otro caso:*
2. *Dividir la lista desordenada en dos sublistas de aproximadamente la mitad del tamaño.*
3. *Ordenar cada sublista recursivamente aplicando el Merge Sort.*
4. *Mezclar las dos sublistas en una sola lista ordenada.*

TEMA 3: Desarrollar un algoritmo paralelo que resuelva el ordenamiento denominado **Quick Sort** para una lista de elementos. Cada elemento debe ser representado mediante una estructura con varios campos, uno de los cuales se considera campo "clave", es decir que será el campo a considerar a la hora de efectuar el ordenamiento. Desarrollar un programa que implemente dicho algoritmo paralelo.

Conceptualmente, el algoritmo ***Quick Sort*** *trabaja de la siguiente manera:*

1. *Elegir un elemento de la lista de elementos a ordenar, al que llamaremos pivote.*
2. *Resituar los demás elementos de la lista a cada lado del pivote, de manera que a un lado queden todos los menores que él, y al otro los mayores. Los elementos iguales al pivote pueden ser colocados tanto a su derecha como a su izquierda, dependiendo de la implementación deseada. En este momento, el pivote ocupa exactamente el lugar que le corresponderá en la lista ordenada.*
3. *La lista queda separada en dos sublistas, una formada por los elementos a la izquierda del pivote, y otra por los elementos a su derecha.*
4. *Repetir este proceso de forma recursiva para cada sublista mientras éstas contengan más de un elemento. Una vez terminado este proceso todos los elementos estarán ordenados.*

***NOTA: Cada grupo deberá elegir uno de los temas propuestos, o bien puede proponer un tema alternativo, cuyo tenor deberá ser evaluado por el cuerpo docente. Debe minimizarse la repetición de temas entre los distintos grupos.**