Universidad Católica del Maule
Facultad de Ciencias de la Ingeniería
Departamento DCI

Profesor: Ricardo J. Barrientos
Fecha: 10/Abril/2025

Evaluación, Unidad 2 (30%)

(K-Means)

Fecha de Entrega	**10/Abril/2025**
Carrera	**Ingeniería Civil Informática**
Académico	**Ricardo J. Barrientos Rojel**
Actividad Curricular	**Sistemas Distribuidos (INF-515 – S2)**

Resultados de Aprendizajes Evaluados:	*Aplicar algoritmos distribuidos de sincronización para coordinar la ejecución de aplicaciones apoyado de documentación técnica en español o inglés, fomentando el trabajo en equipos, juzgando la actuación individual y grupal.*

PUNTAJE MÁXIMO	**6,0**	**PUNTAJE DE CORTE**	**3,6**

INSTRUCCIONES

- Debe desarrollar esta tarea en pareja. Debe comunicar por email a rbarrientos@ucm.cl con quien trabajará, a más tardar el **15 de Abril.** El 16 de Abril se asignarán parejas de manera aleatoria (entre los estudiantes que estén solos).
- Plazo de entrega: Por **LMS** a más tardar **el 27 de Abril del 2025**.
- El producto a entregar es un programa en lenguaje C basado en OpenMP. Recuerde siempre compilar con la opción "-O3"

Restricciones

Se aplicará artículo 67º del reglamento del estudiante, el cual indica que, en caso de sorprender copia parcial o exacta, ya sea entre compañeros o reproducidos de algún medio, lo cual implica un 1,0 para todos los involucrados.

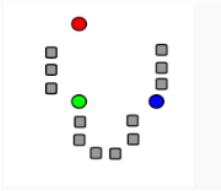
Descripción

Esta tarea consiste en implementar el algoritmo *K-Means* utilizando programación multi-hilo con OpenMP en C o C++.

El algoritmo *K-Means* tiene por objetivo el crear K clúster (o conjuntos), donde cada conjunto está formado por elementos cercanos entre ellos. En nuestro caso, los elementos serán vectores de dimensión 20, es decir, cada vector tendrá 20 coordenadas, donde cada coordenada será un número real (float).

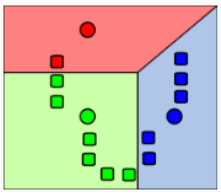
A modo de ejemplo, considere la siguiente imagen donde cada cuadrado representa un vector de la base de datos y los círculos representan los centroides de cada clúster. Las etapas para la implementación del algoritmo son las siguientes (tome en cuenta que el valor de K es conocido):

Paso 1:



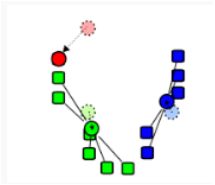
Se eligen K centroides de manera aleatoria. También podrían ser elementos de la base de datos elegidos aleatoriamente.

Paso 2:



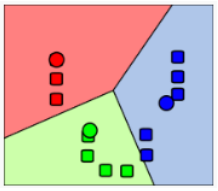
Cada elemento de la base de datos se asocia al clúster más cercano. Un clúster es representado por su centroide.

Paso 3:



Se recalculan los centroides. Cada centroide nuevo es el promedio de los elementos del clúster.

Paso 4:



Se repiten los pasos 2 y 3 hasta converger, es decir, hasta que ningún elemento se cambia a otro clúster en el paso 2.

Usted debe decidir el cómo distribuir el cómputo entre los distintos hilos. Es obligación que T hilos colaboren entre ellos para la implementación del algoritmo, con T > 2.

Utilice las siguientes constantes al inicio de su código:

```
#define DIM 20
#define T 8
#define K 3
```

`DIM` indica la dimensión de los vectores, `T` la cantidad de hilos que utilizará en paralelo y `K` la cantidad de clústers a crear. Utilice la cantidad de hilos que le parezca adecuada.

Su programa, a modo de resultado, debe imprimir por pantalla solamente el número del clúster al que pertenecen los elementos de la base de datos desde el primero al último. Por ejemplo, si imprimiera:

```
3
2
2
1
```

Usted estaría indicando que el primer elemento de la base de datos pertenece al 3er. clúster, el segundo y tercer elemento al 2do. clúster y el cuarto (y último) elemento pertenece al 1er. clúster. Debe imprimir un número por línea.

Usted debe entregar un archivo .c con su código, y también un archivo .txt donde indique 4 datos para cada base de datos indicadas al final de este documento: 1) el tiempo de su algoritmo paralelo (T_P), 2) el tiempo del algoritmo secuencial (T_S), 3) la cantidad de datos de la Base de Datos utilizada, y 4) el speed-up (T_S / T_P). El speed-up es una medición que indica la cantidad de veces más rápido que es el algoritmo paralelo sobre el secuencial.

Debe usar la **distancia euclidiana** como función de distancia entre elementos.

Su tarea será revisada utilizando archivos de prueba con el siguiente formato:

`test.txt`

```
5
0.137 -0.03 0.09 0.10 0.2 -0.1 -0.05 0.08 0.7 0.6 0.01 0.2 0.4 0.03 0.4 0.6 0.2 0.5 0.6 0.9
-0.3 0.9 0.18 0.4 0.1 0.03 0.77 -0.2 1.3 0.1 0.3 0.1 0.02 0.6 0.7 0.1 -0.8 0.8 0.7 0.083
-0.0 0.2 -0.031 0.24 0.1 0.5 0.7 0.8 -0.1 0.5 -0.0 -0.2 0.1 0.5 0.02 0.06 0.96 0.56 0.16 0.30
0.9 0.5 0.26 0.1 -0.9 0.003 -0.8 0.6 0.03 0.7 0.1 -0.9 0.02 0.3 0.3 0.0 0.7 0.65 0.028 -0.044
0.4 -0.1 0.0 0.1 0.5 -0.0 0.8 0.8 -0.6 0.4 0.27 0.4 0.34 0.5 -0.2 0.82 -0.1 0.3 0.043 0.246
```

La primera fila indica la cantidad de elementos (vectores) de la base de datos. Desde la segunda fila en adelante están los vectores que conforman la base de datos. Las coordenadas entre sí están separadas por un espacio. Hay un elemento de la base de datos por fila.

De esta forma, su programa podrá ser ejecutado redirigiendo la entrada desde el teclado al archivo de prueba con el operador "<", de la siguiente manera en caso de ejecución en terminal:

```
./a.out < test.txt
```

Debe descargar archivos de test para probar el desempeño de su algoritmo desde los siguientes enlaces:

- https://ribarrie.cl/temp/Test-BD-100.txt
- https://ribarrie.cl/temp/Test-BD-1000.txt
- https://ribarrie.cl/temp/Test-BD-95000.txt

Para calificar su tarea, se tomará en cuenta los siguientes ítems:

1) Correcta lectura de los datos. (0,5 ptos.)
2) Correcta distribución del cómputo entre los hilos. (3,5 ptos.)
3) Correcta impresión por pantalla del resultado de su programa. (1,5 ptos.)
4) Adecuadas mediciones entregadas en su archivo .txt. (0,5 ptos.)

Nota:

- Esta tarea será revisada con conjunto con los estudiantes de cada grupo. El grupo deberá responder a las dudas planteadas en la revisión, de no hacerlo tendrá -3,0 unidades de nota.
- No puede utilizar `#pragma omp for` o `#pragma omp parallel for`. De hacerlo, tendrá -3,0 unidades de nota.