#### Clase del día – 14/04/2021

En la tarea 3 desarrollamos un programa que multiplica matrices cuadradas en forma distribuida usando paso de mensajes.

Como pudimos ver, la programación de un sistema distribuido utilizando el paso de mensajes es complicada, ya que se debe controlar explícitamente la serialización, el envío y la des-serialización de los datos, además de la lógica particular del sistema.

En la clase de hoy vamos a ver cómo desarrollar un programa distribuido que calcule el producto de matrices cuadradas utilizando Java RMI.

**Partición de los datos**

Dadas las matrices A y B de tamaño NxN, el producto C=AxB se obtiene dividiendo la matriz A en las matrices A1 y A2, y dividiendo la **transpuesta** de la matriz B en las matrices B1 y B2.

El tamaño de las matrices A1, A2, B1 y B2 es N/2 renglones y N columnas

Entonces, la matriz C se compone de las matrices C1, C2, C3 y C4, tal como se muestra en la siguiente figura:

****

Las matrices C1, C2, C3 y C4 se calculan de la siguiente manera:

C1 = A1 x B1

C2 = A1 x B2

C3 = A2 x B1

C4 = A2 x B2

**Multiplicación de matrices utilizando objetos distribuidos**

Para multiplicar matrices utilizando objetos distribuidos, escribiremos un servidor RMI que ejecute un método remoto llamado multiplica\_matrices(), este método recibe como parámetros dos matrices de tamaño (N/2)xN y regresa como resultado una matriz cuadrada de tamaño (N/2)x(N/2).

Ahora debemos escribir un cliente RMI que inicialice las matrices, transponga la matriz B, invoque el método multiplica\_matrices() y acomode las matrices C1, C2, C3 y C4 para formar la matriz C.

Consideremos el método multiplica\_matrices() (este método ejecutará en el servidor RMI):

public int[][] multiplica\_matrices(int[][] A,int[][] B) throws RemoteException

{

int[][] C = new int[N/2][N/2];

for (int i = 0; i < N/2; i++)

for (int j = 0; j < N/2; j++)

for (int k = 0; k < N; k++)

C[i][j] += A[i][k] \* B[j][k];

return C;

}

Cuando el método multiplica\_matrices() se invoca localmente, recibe como parámetros las referencias a las matrices A y B y regresa una referencia a la matriz C.

Cuando el método es invocado en forma remota, entonces la capa RMI serializa las matrices A y B en el cliente y las des-serializa en el servidor. De la misma forma, la capa RMI serializa la matriz C en el servidor y la des-serializa en el cliente.

Ahora veamos el método separa\_matriz() el cual utilizaremos para obtener las matrices A1, A2, B1 y B2:

static int[][] separa\_matriz(int[][] A,int inicio)

{

int[][] M = new int[N/2][N];

for (int i = 0; i < N/2; i++)

for (int j = 0; j < N; j++)

M[i][j] = A[i + inicio][j];

return M;

}

El método separa\_matriz() recibe como parámetros la matriz a dividir y el renglón inicial. El método regresará una matriz de tamaño (N/2)xN.

Entonces, podemos obtener las matrices A1, A2, B1 y B2 de la siguiente manera:

int[][] A1 = separa\_matriz(A,0);

int[][] A2 = separa\_matriz(A,N/2);

int[][] B1 = separa\_matriz(B,0);

int[][] B2 = separa\_matriz(B,N/2);

Dadas las matrices A1, A2, B1 y B2, podemos obtener las matrices C1, C2, C3 y C4 utilizando el método multiplica\_matrices():

int[][] C1 = multiplica\_matrices(A1,B1);

int[][] C2 = multiplica\_matrices(A1,B2);

int[][] C3 = multiplica\_matrices(A2,B1);

int[][] C4 = multiplica\_matrices(A2,B2);

Veamos ahora el método acomoda\_matriz(), el cual permite construir la matriz C a partir de las matrices C1, C2, C3 y C4:

static void acomoda\_matriz(int[][] C,int[][] A,int renglon,int columna)

{

for (int i = 0; i < N/2; i++)

for (int j = 0; j < N/2; j++)

C[i + renglon][j + columna] = A[i][j];

}

El método acomoda\_matriz() recibe como parámetros la matriz C, la sub-matriz a acomodar, y la posición (renglón,columna) en la matriz C donde se va a colocar la sub-matriz.

Finalmente para obtener la matriz C podemos hacer lo siguiente:

int[][] C = new int[N][N];  
acomoda\_matriz(C,C1,0,0);  
acomoda\_matriz(C,C2,0,N/2);  
acomoda\_matriz(C,C3,N/2,0);  
acomoda\_matriz(C,C4,N/2,N/2);

#### Actividades individuales a realizar

Desarrollar un programa en Java que multiplique dos matrices cuadradas de tamaño NxN en forma local, utilizando las funciones separa\_matriz(), multiplica\_matrices() y acomoda\_matriz().

Principio del formulario

Final del formulario

#### Clase del día - 16/04/2021

En la clase de hoy vamos a jugar un kahoot en la modalidad de "challenge".

Para jugar el kahoot deberán ingresar al siguiente enlace:

[Desarrollo de sistemas distribuidos - Objetos distribuidos con Java RMI](https://kahoot.it/challenge/01649177?challenge-id=3882dbfd-4f85-4e86-9fa2-018882d19a9a_1618171740901)

Es necesario que los alumnos y alumnas ingresen su "nickname" como su nombre y apellido (por ejemplo JuanLopez), de manera que sea posible identificar a los ganadores de puntos extra.

La hora límite para jugar este kahoot es 11:00 PM del 16 de abril.