**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

ING. SISTEMAS COMPUTACIONALES

DESARROLLO DE SISTEMAS DISTRIBUIDOS

PINEDA GUERRERO CARLOS

**TAREA #3**

**MULTIPLICACIÓN DE MATRICES UTILIZANDO**

**PASO DE MENSAJES**

FECHA DE REALIZACIÓN: 15/03/2021

FECHA DE ENTREGA: 19/03/2021

GRUPO: 4CM3

ELABORÓ:

**PÉREZ FEDERICO JOSÉ JOEL**

**DESCRIPCIÓN**

En esta tarea el alumno deberá desarrollar un solo programa en Java, el cual calculará el producto de dos matrices cuadradas en forma distribuida sobre cinco nodos.

Sean A y B dos matrices cuadradas con elementos de tipo int, N el tamaño de las matrices. La matriz C con elementos de tipo int. Se deberá ejecutar dos casos:

1. N=4, desplegar las matrices A, B y C=AxB y el checksum de la matriz C.
2. N=1000, desplegar el checksum de la matriz C.

Se deberá inicializar las matrices de la siguiente manera:

A[i][j]= i - 2\*j

B[i][j] = i + 2\*j

Donde A[i][j] y B[i][j] son los elementos Ai,j y Bi,j respectivamente.

El checksum de la matriz C se calculará como la suma de todos elementos de la matriz. Para calcular la sumatoria se deberá declarar una variable "checksum" de tipo long.

El programa deberá ser ejecutado en cinco ventanas de Sistema (Windows) o en cinco terminales (Linux o MacOS). En cada ventana se pasará como parámetro al programa el número de nodo, a saber: 0, 1, 2, 3 y 4.

Para realizar la multiplicación de matrices en forma distribuida se divide la matriz A en las matrices A1 y A2. El tamaño de las matrices A1 y A2 es N/2 renglones y N columnas.

La matriz B se divide en las matrices B1 y B2. El tamaño de matrices B1 y B2 es N renglones y N/2 columnas, pero debido a que las matrices se guardan en memoria por renglones, es más eficiente transponer la matriz B y dividirla de la siguiente manera:



Figura 0. División de Matrices

Entonces la matriz C=AxB se compone de las matrices C1, C2, C3 y C4, tal como se muestra en la Figura 0.

Donde:

C1 = A1 x B1

C2 = A1 x B2

C3 = A2 x B1

C4 = A2 x B2

Para multiplicar las matrices A y B se implementan los siguientes algoritmos:

Nodo 0

1. Inicializar las matrices A y B.

2. Transponer la matriz B.

3. Enviar la matriz A1 al nodo 1.

4. Enviar la matriz B1 al nodo 1.

5. Enviar la matriz A1 al nodo 2.

6. Enviar la matriz B2 al nodo 2.

7. Enviar la matriz A2 al nodo 3.

8. Enviar la matriz B1 al nodo 3.

9. Enviar la matriz A2 al nodo 4.

10. Enviar la matriz B2 al nodo 4.

11. Recibir la matriz C1 del nodo 1.

12. Recibir la matriz C2 del nodo 2.

13. Recibir la matriz C3 del nodo 3.

14. Recibir la matriz C4 del nodo 4.

15. Calcular el checksum de la matriz C.

16. Desplegar el checksum de la matriz C.

17. Si N=4 entonces desplegar la matriz C

Nodo 1

1. Recibir del nodo 0 la matriz A1.

2. Recibir del nodo 0 la matriz B1.

3. Realizar el producto C1=A1xB1 (renglón por renglón).

4. Enviar la matriz C1 al nodo 0.

Nodo 2

1. Recibir del nodo 0 la matriz A1.

2. Recibir del nodo 0 la matriz B2.

3. Realizar el producto C2=A1xB2 (renglón por renglón).

4. Enviar la matriz C2 al nodo 0.

Nodo 3

1. Recibir del nodo 0 la matriz A2.

2. Recibir del nodo 0 la matriz B1.

3. Realizar el producto C3=A2xB1 (renglón por renglón).

4. Enviar la matriz C3 al nodo 0.

Nodo 4

1. Recibir del nodo 0 la matriz A2.

2. Recibir del nodo 0 la matriz B2.

3. Realizar el producto C4=A2xB2 (renglón por renglón).

4. Enviar la matriz C4 al nodo 0.

**COMPILACIÓN Y EJECUCIÓN**

La Figura 1, muestra la compilación del programa MD.java.

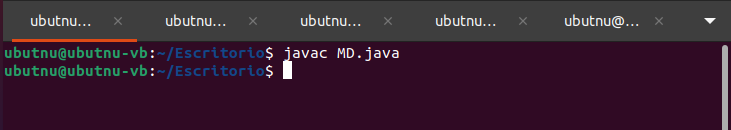


Figura 1: Compilación de MD.java

Las Figuras 2, 3 4 y 5 muestran la ejecución de los nodos 1, 2, 3 y 4 para el cálculo de la multiplicación de las matrices.

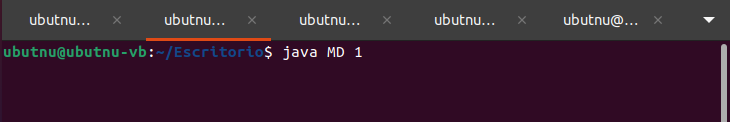


Figura 2. Ejecución del nodo 1

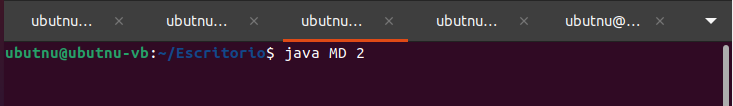


Figura 3. Ejecución del nodo 2

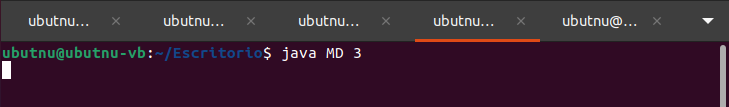


Figura 4. Ejecución del nodo 3

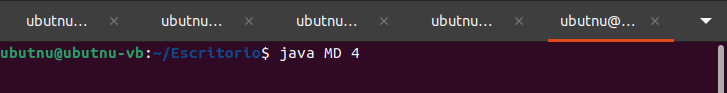


Figura 5. Ejecución del nodo 4

A continuación, se presenta el resultado de la ejecución del nodo 0 para N=4 en la Figura 6 y N=1000 en la Figura 7.

Para el caso de N=4, se imprimen los elementos de ambas matrices, el resultado de su multiplicación y el checksum.

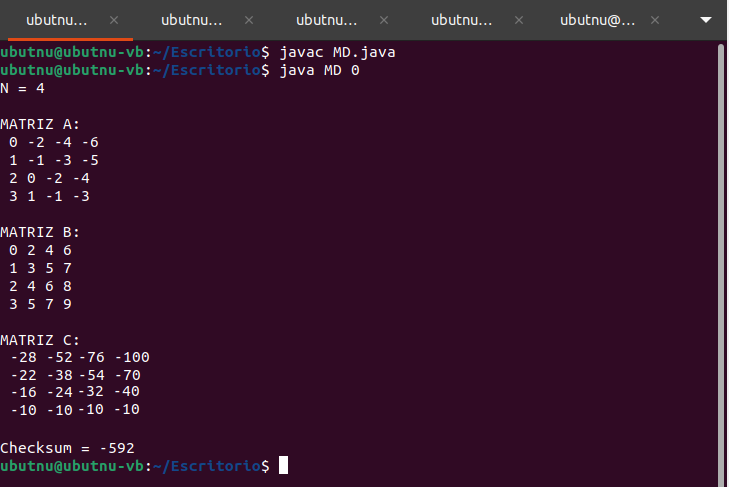


Figura 6. Resultados en nodo 0 para N=4

Para el caso de N=1000, únicamente se imprime el checksum.

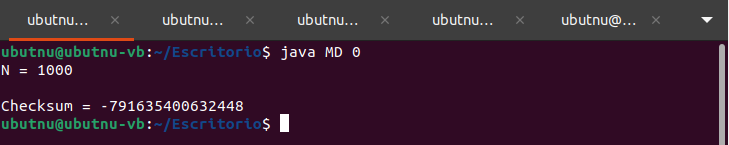


Figura 7. Resultados en nodo 0 para N=1000

**CONCLUSIONES**

Como observe con los programas MultiplicaMatriz.java y MultiplicaMatriz2.java de la tarea anterior, cuando se accede a información en memoria, se puede mejorar la eficiencia del algoritmo considerando que el tiempo de recuperación de la información es más rápido si tiene localidad espacial y/o temporal. Para esta tarea, la implementación del cálculo de la multiplicación de forma distribuida nos ha ayudado a descomponer el problema inicial y fragmentarlo en problemas más pequeños y fáciles de calcular esto con el objetivo de obtener una mejora en el tiempo de ejecución.

Para la ejecución del programa, se utilizó una máquina virtual con Ubuntu, y aunque el tiempo de ejecución fue bueno, comparándolo con la ejecución de los programas de la tarea anterior lo cuales se ejecutaron en una maquina ‘normal’, no se alcanza percibir mucha diferencia.