

Tarea3 Multiplicacion de Matrices de forma Distribuida

Pulido Bejarano Raymundo

12 de Octubre del 2020

1. Introducción

En esta Tarea se desarrollo un pequeño sistema distribuido donde se realizara la multiplicacion de matrices de forma distribuida.

En donde las matrices seran cuadradas es decir $N \times N$, donde $N = 4$ y $N = 1000$, en diferentes pruebas.

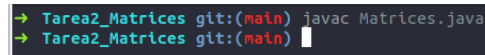


Figura 1: Topologia del Sistema

Donde:

- $C1 = A1 \times B1$
- $C2 = A1 \times B2$
- $C3 = A2 \times B1$
- $C4 = A2 \times B2$

2. Capturas de Pantalla



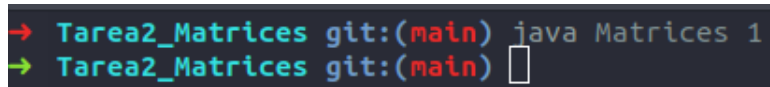
```
→ Tarea2_Matrices git:(main) javac Matrices.java
→ Tarea2_Matrices git:(main) █
```

Figura 2: Compilación

En la figura 2 se muestra la correcta compilación del archivo *Matrices.java*, en el cual está escrito el programa solución al problema planteado.

Esta compilación muestra el mismo resultado ya sea para $N=1000$ como para $N=4$.

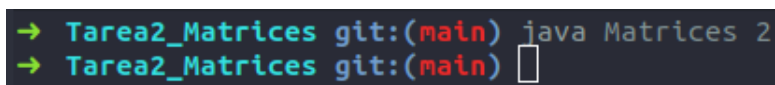
2.1. $N = 1000$



```
→ Tarea2_Matrices git:(main) java Matrices 1
→ Tarea2_Matrices git:(main) █
```

Figura 3: Ejecución Nodo 1

En la figura 3, se muestra la correcta operación del programa usando de parámetro el *numero 1*, que nos dice que hará como el nodo 1, esto quiere decir que realizara el computo de C1 descrito anteriormente.



```
→ Tarea2_Matrices git:(main) java Matrices 2
→ Tarea2_Matrices git:(main) █
```

Figura 4: Ejecución Nodo 2

En la figura 4, se muestra la correcta operación del programa usando de parámetro el *numero 2*, que nos dice que hará como el nodo 2, esto quiere decir que realizara el computo de C2 descrito anteriormente.

```
→ Tarea2_Matrices git:(main) java Matrices 3
→ Tarea2_Matrices git:(main) □
```

Figura 5: Ejecución Nodo 3

En la figura 5, se muestra la correcta operación del programa usando de parámetro el *numero 3*, que nos dice que hará como el nodo 3, esto quiere decir que realizara el computo de C3 descrito anteriormente.

```
→ Tarea3_Matrices git:(main) java Matrices 4
→ Tarea3_Matrices git:(main) □
```

Figura 6: Ejecución Nodo 4

En la figura 6, se muestra la correcta operación del programa usando de parámetro el *numero 4*, que nos dice que hará como el nodo 4, esto quiere decir que realizara el computo de C4 descrito anteriormente.

```
→ Tarea3_Matrices git:(main) X java Matrices 0
El checksum de la matriz C es: 625312338132448
→ Tarea3_Matrices git:(main) X □
```

Figura 7: Ejecución Nodo 0

En la figura 7, se muestra la correcta operación del programa usando de parámetro el *numero 0* y de una forma funcional, hará de servidor central para gestionar las peticiones y operar los datos suministrados por los clientes, que en este casos son los distintos nodos, a su vez este recolectara la informacion de las distintas partes de $C = C1, C2, C3, C4$, como tambien calcula el *Checksum* de la dicha matriz.

2.2. $N = 4$

Como se comento anteriormente la compilacion muestra el mismo resultado en $N = 1000$ como en $N = 4$, esto mismo ocurre al procesar los nodos 1,2,3 y 4, por lo que en esta sub seccion omitire la ejecucion de esto dichos nodos, para unicamente mostrar los resultados devueltos por el nodo 0.

```
→ Tarea3_Matrices git:(main) X java Matrices 0
Matriz A:
0 1 2 3
2 3 4 5
4 5 6 7
6 7 8 9
Matriz B:
0 2 4 6
-1 1 3 5
-2 0 2 4
-3 -1 1 3
Matriz C:
28 22 16 10
52 38 24 10
76 54 32 10
100 70 40 10
El checksum de la matriz C es: 592
→ Tarea3_Matrices git:(main) X
```

Figura 8: Ejecución Nodo 0

En la figura 8, se muestra la correcta operación del programa usando de parámetro el *numero 0* y de una forma funcional, hará de servidor central para gestionar las peticiones y operar los datos suministrados por los clientes, que en este casos son los distintos nodos, a su vez este recolectara la informacion de las distintas partes de $C = C_1, C_2, C_3, C_4$, como tambien calcula el *Checksum* de la dicha matriz y a su vez mostrar las matrices A,B y C para su entendimiento mas grafico.