

Instituto Politécnico Nacional

Escuela Superior de Cómputo

UA.Desarrollo de Sistemas Distribuidos

“Tarea 3. Multiplicación de matrices distribuida
utilizando paso de mensajes”

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Alumno:
Márquez León Jorge Luis

Profesor:
Carlos Pineda Guerrero

Grupo:
4CV11

Introducción.

En este reporte se desarrollará un solo programa en Java, el cual calculará el producto de dos matrices cuadradas en forma distribuida sobre cinco nodos.

Sean A,B y C matrices cuadradas con elementos de tipo long, N renglones y N columnas, N par y $C = A \times B$.

Se deberá ejecutar dos casos:

1. $N=10$, desplegar las matrices A,B y C y el checksum de la matriz C.
2. $N=1500$, desplegar el checksum de la matriz C.

$$\text{Checksum} = \sum_{i,j=0}^{n-1} C(i)(j)$$

Se deberá inicializar las matrices de la siguiente manera:

$$\begin{aligned} A[i][j] &= i+2*j \\ B[i][j] &= i-2*j \end{aligned}$$

Donde $A[i][j]$ y $B[i][j]$ son los elementos $A_{i,j}$ y $B_{i,j}$ respectivamente.

El programa deberá ser ejecutado en cinco máquinas virtuales con Ubuntu (1 CPU, 1GB de RAM y disco HDD estándar) en cada máquina virtual se pasará como parámetro al programa el número de nodo, a saber: 0, 1, 2, 3 y 4.

El nombre de cada máquina virtual deberá ser una letra "M", el número de boleta del alumno, un guion y el número de nodo, por ejemplo, si el número de boleta del alumno es 12345678, entonces el nodo 0 deberá llamarse: M12345678-0, el nodo 1 deberá llamarse M12345678-1, y así sucesivamente.

Se eliminarán las máquinas virtuales cuando no las usen, con la finalidad de ahorrar el saldo de sus cuentas de Azure.

Suponga que 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.

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

$$\begin{array}{c}
 \begin{array}{|c|} \hline A1 \\ \hline A2 \\ \hline \end{array} & \times & \begin{array}{|c|c|} \hline B1 & B2 \\ \hline \end{array} & = & \begin{array}{|c|c|} \hline C1 & C2 \\ \hline C3 & C4 \\ \hline \end{array}
 \end{array}$$

Figura 1. Multiplicación de Matrices.

Donde:

$$C1 = A1 \times B1$$

$$C2 = A1 \times B2$$

$$C3 = A2 \times B1$$

$$C4 = A2 \times B2$$

Debido a que las matrices se guardan en memoria por renglones, es más eficiente transponer la matriz B y dividirla de la siguiente manera:

$$\begin{array}{c}
 \begin{array}{|c|} \hline A1 \\ \hline A2 \\ \hline \end{array} & \times & \begin{array}{|c|} \hline B1 \\ \hline B2 \\ \hline \end{array} & = & \begin{array}{|c|c|} \hline C1 & C2 \\ \hline C3 & C4 \\ \hline \end{array}
 \end{array}$$

Figura 2. Multiplicación Efectiva de Matrices.

Ahora supongamos que tenemos cinco nodos identificados con los números 0, 1, 2, 3 y 4.

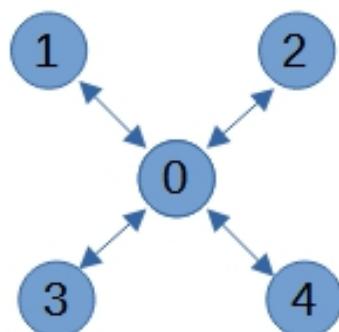


Figura 3. Topología Lógica

Código Fuente del Programa Desarrollado.

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class Matriz {
    static int nodo;
    static String ip;
    static int N = 10;
    static Object lock = new Object();

    long checksum = 0;

    //Declaramos las matrices originales
    static long[][] A = new long[N][N];
    static long[][] B = new long[N][N];
    static long[][] C = new long[N][N];

    static void imprimeMatriz(long[][] M, int col, int fil) {
        for (int i = 0; i < col; i++) {
            for (int j = 0; j < fil; j++) {
                System.out.print(M[i][j] + "\t");
            }
            System.out.println("");
        }
        System.out.println("");
    }

    static long calcularChecksum(long[][] M, int N) {
        long aux = 0;
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++) {
                aux += M[i][j];
            }
        }
        return aux;
    }
}
```

```

        }
    }
    return aux;
}

static long[][] recibirMatriz(DataInputStream entrada, int fil, int col, int inicioFil,
int inicioCol)
throws Exception {
    long[][] aux = new long[col][fil];

    for (int i = inicioCol; i < col; i++) {
        for (int j = inicioFil; j < fil; j++) {
            aux[i][j] = entrada.readLong();
        }
    }
    return aux;
}

static void enviarMatriz(DataOutputStream salida, long[][] M, int fil, int col) throws
Exception {
    for (int i = 0; i < col; i++) {
        for (int j = 0; j < fil; j++) {
            salida.writeLong(M[i][j]);
        }
    }

    salida.flush();
}

static long[][] dividirMatriz(int inicioCol, int col, int inicioFil, int fil, boolean Matriz)
{
    long[][] aux = new long[N / 2][N];

    for (int i = inicioCol; i < col; i++) {
        for (int j = inicioFil; j < fil; j++) {
            if (Matriz) {

```

```

        aux[i - inicioCol][j] = A[i][j];
    } else {
        aux[i - inicioCol][j] = B[i][j];
    }
}

return aux;
}

static void agregarFragmento(long[][] M, int inicioFil, int inicioCol, int fil, int col) {
    for (int i = inicioCol; i < col; i++) {
        for (int j = inicioFil; j < fil; j++) {
            C[i][j] = M[i][j];
        }
    }
}

static long[][] transponerMatriz(long[][] M, int N) {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < i; j++) {
            long aux = M[i][j];
            M[i][j] = M[j][i];
            M[j][i] = aux;
        }
    }
    return M;
}

static class Worker extends Thread {
    Socket conexion;

    public Worker(Socket conexion) {
        this.conexion = conexion;
    }
}

```

```

public void run() {
    try {
        long[][] auxMA = new long[N / 2][N];
        long[][] auxMB = new long[N / 2][N];
        //Creamos la transmisión de entrada para recibir las matrices
        DataInputStream entrada = new
DataInputStream(conexion.getInputStream());
        //Creamos la transmisión de salida para enviar la matrices
        DataOutputStream salida = new
DataOutputStream(conexion.getOutputStream());
        //Recibe en la variable x el nodo
        int x = entrada.readInt();
        //Enviar las matrices dependiendo el nodo
        if (x == 1) {
            System.out.println("\nSe conecto el nodo 1 envio la matriz A1 y B1");
            auxMA = dividirMatriz(0, N / 2, 0, N, true);
            auxMB = dividirMatriz(0, N / 2, 0, N, false);
        } else if (x == 2) {
            System.out.println("\nSe conecto el nodo 2 envio la matriz A1 y B2");
            auxMA = dividirMatriz(0, N / 2, 0, N, true);
            auxMB = dividirMatriz(N / 2, N, 0, N, false);
        } else if (x == 3) {
            System.out.println("\nSe conecto el nodo 3 envio la matriz A2 y B1 ");
            auxMA = dividirMatriz(N / 2, N, 0, N, true);
            auxMB = dividirMatriz(0, N / 2, 0, N, false);
        } else if (x == 4) {
            System.out.println("\nSe conecto el nodo 4 envio la matriz A2 y B2 ");
            auxMA = dividirMatriz(N / 2, N, 0, N, true);
            auxMB = dividirMatriz(N / 2, N, 0, N, false);
        }
        enviarMatriz(salida, auxMA, N, N / 2);
        enviarMatriz(salida, auxMB, N, N / 2);
    }
}

```

```

synchronized(lock) {
    long[][] matrizAux;
    if (x == 1) {
        matrizAux = recibirMatriz(entrada, N / 2, N / 2, 0, 0);
        agregarFragmento(matrizAux, 0, 0, N / 2, N / 2);
    } else if (x == 2) {
        matrizAux = recibirMatriz(entrada, N, N / 2, N / 2, 0);
        agregarFragmento(matrizAux, N / 2, 0, N, N / 2);
    } else if (x == 3) {
        matrizAux = recibirMatriz(entrada, N / 2, N, 0, N / 2);
        agregarFragmento(matrizAux, 0, N / 2, N / 2, N);
    } else if (x == 4) {
        matrizAux = recibirMatriz(entrada, N, N, N / 2, N / 2);
        agregarFragmento(matrizAux, N / 2, N / 2, N, N);
    }
}

// Mensaje de término de conexión
System.out.print("Termina conexión con el nodo: " + x);
// Cerramos la conexión y los flujos de entrada y salida
entrada.close();
salida.close();
conexion.close();
} catch (Exception e) {
    e.printStackTrace();
}
}

public static void main(String[] args) throws Exception {
    if (args.length != 2) {
        System.err.println("Se debe pasar como parámetros el número del nodo y la
IP");
        System.exit(1);
    }

    nodo = Integer.valueOf(args[0]);
}

```

```

ip = args[1];

if (nodo == 0) {
    // Inicializar matrices A, B, C
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            A[i][j] = i + 2 * j;
            B[i][j] = i - 2 * j;
            C[i][j] = 0;
        }
    }
}

```

B = transponerMatriz(B, N);

```

// Mensaje del nodo 0 de espera
System.out.println("\nEsperando por conexiones...");
// creamos el servidor
ServerSocket servidor = new ServerSocket(5000);
//Aceptaremos los 4 clientes
Worker[] w = new Worker[4];

for (int i = 0; i < 4; ++i) {
    // Una conexión por cada nodo
    Socket conexion = servidor.accept();
    w[i] = new Worker(conexion);
    w[i].start();
}

// Esperamos a que se ejecute el hilo
for (int i = 0; i < 4; ++i) {
    w[i].join();
}

// Mensaje del nodo 0
System.out.println("\nHe dejado de recibir conexiones... ");
// Cerramos el servidor
servidor.close();

```

```

//Calculamos el checksum
System.out.println("Checksum: " + calcularChecksum(C, N));
//Si N ==10 imprimimos las matrices
if (N == 10) {
    System.out.println("Desplegando matriz C = A x B");
    System.out.println("\nMatriz A:");
    imprimeMatriz(A, N, N);
    System.out.println("\nMatriz B:");
    imprimeMatriz(B, N, N);
    System.out.println("\nMatriz C");
    imprimeMatriz(C, N, N);
}

} else {
    //Creamos la conexión con el socket
    Socket conexion = new Socket(ip, 5000);
    //Creamos las transmisiones de entrada y salida
    DataInputStream entrada = new DataInputStream(conexion.getInputStream());
    DataOutputStream salida = new
    DataOutputStream(conexion.getOutputStream());
    //Enviamos al servidor (nodo 0) el numero de nodo en el que estamos
    salida.writeInt(nodo);
    //Declaramos las matrices donde se recibirán las originales
    long[][] auxMA = recibirMatriz(entrada, N, N / 2, 0, 0);
    long[][] auxMB = recibirMatriz(entrada, N, N / 2, 0, 0);
    //Declaramos la matriz en donde guardaremos el producto de A1 x B1
    long[][] auxMC = new long[N / 2][N / 2];

    //Multiplicar matriz
    for (int i = 0; i < (N / 2); i++) {
        for (int j = 0; j < (N / 2); j++) {
            long suma = 0;
            for (int k = 0; k < N; k++) {
                suma += auxMA[i][k] * auxMB[j][k];
            }
            auxMC[i][j] = suma;
        }
    }
}

```

```

    }

}

// Regresar fragmento

enviarMatriz(salida, auxMC, N / 2, N / 2);

// cerramos las transmisiones de entrada, salida y la conexión

entrada.close();
salida.close();
conexion.close();

}

}

}

}

```

Capturas de pantalla.

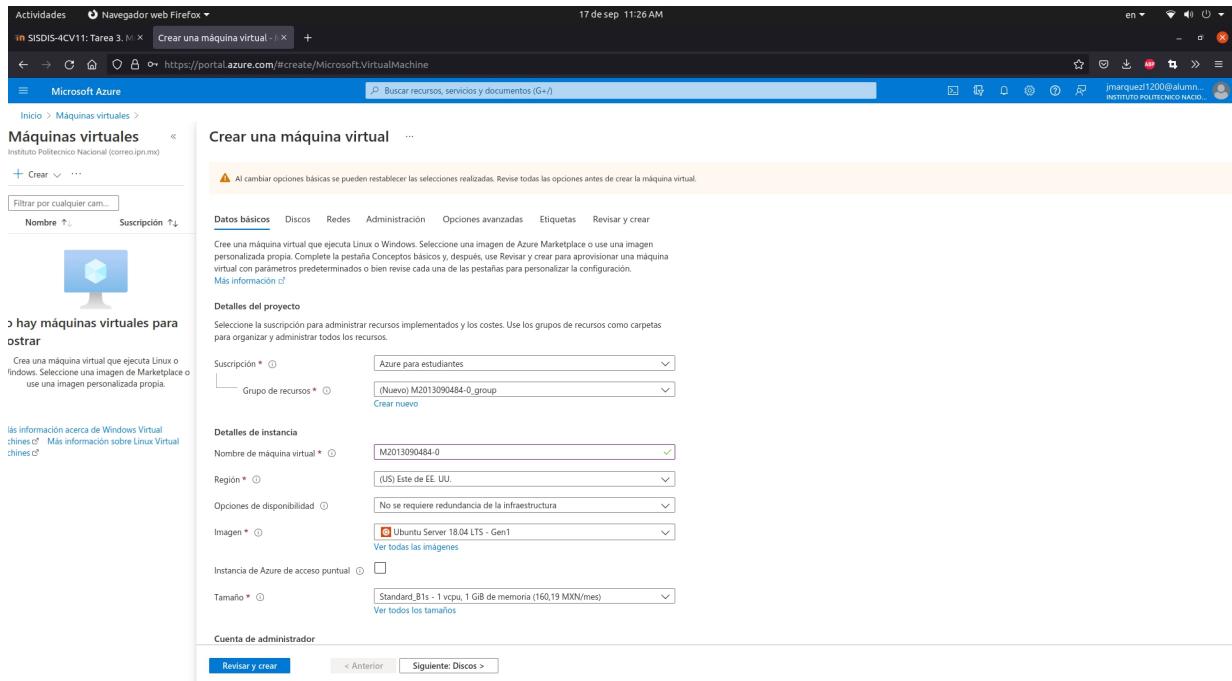


Figura 4. Creación de Máquina Virtual 0

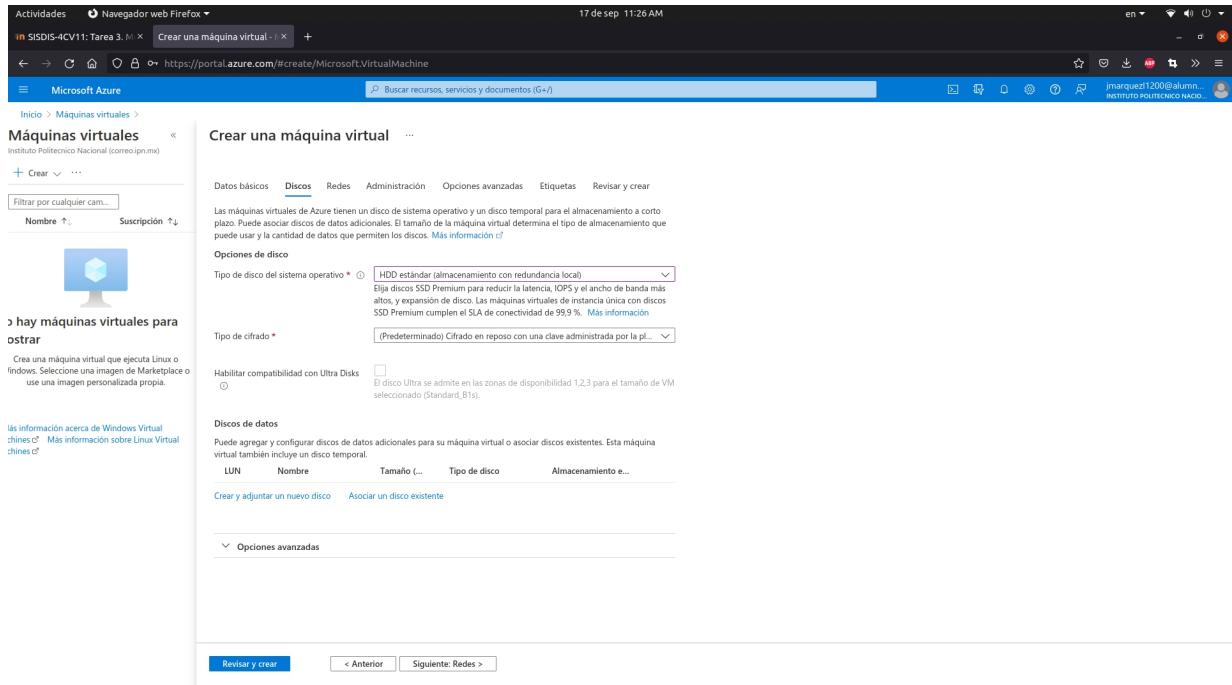


Figura 5. Creación de Máquina Virtual 0

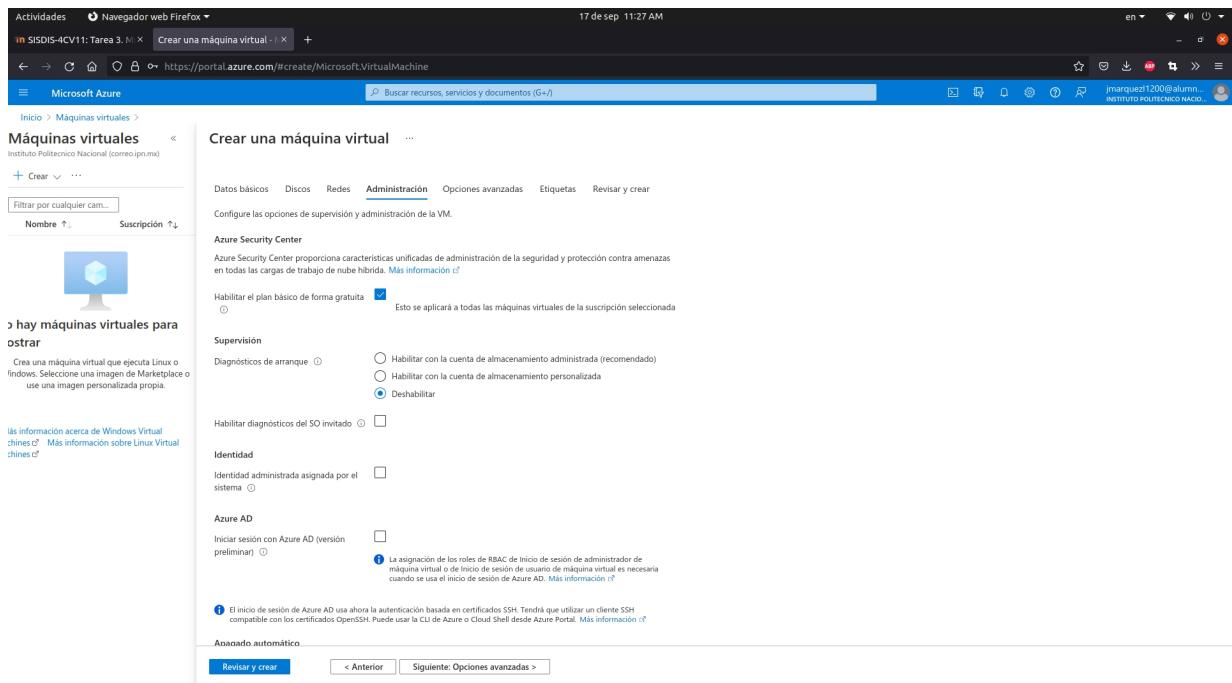


Figura 6. Creación de Máquina Virtual 0

```

Actividades Terminal 17 de sep 12:36 PM
SISDIS-4CV11: Tarea 3. M X M2013090484-4 - Microsoft Edge (5) Como abrir puertos d Cómo usar SFTP para tra + 
https://m4gm.com/moodle/mod/assign/view.php?id=142
m4gm Español (Méjico) (es_mx) Jorge Luis Marquez Leon
SISDIS-4CV11
Participantes
Insignias
Competencias
Calificaciones
General
1. Introducción
2. Sincronización y coordinación
3. Sistemas basados en objetos distribuidos
4. Servicios de nombres, archivos y replicación
5. Cómputo en la nube
Tablero
Página inicial del sitio
Calendario
Archivos privados
Mis cursos
abrir puerto
17 de sep 12:36 PM
M2013090484-0@M2013090484... ~ M2013090484-1@M2013090484... ~ M2013090484-2@M2013090484... ~ M2013090484-3@M2013090484... ~ M2013090484-4@M2013090484... ~
Adding debian:Amazon Root CA_3.pem
Adding debian:emSign ECC_Root_CA_-C3.pem
Adding debian:Autoridad_de_Certificación_FirmaProfesional_CIF_A62634068.pem
Adding debian:emSign Root CA_-C1.pem
Adding debian:Hellenic_Academic_and_Research_Institutions_ECC_RootCA_2015.pem
Adding debian:SSL_com_EV_Root_Certification_Authority_ECC.pem
Adding debian:Microsoft_RSA_Root_Certificate_Authority_2017.pem
Adding debian:emSign_ECC_Root_CA_-G3.pem
done.
Setting up default-jre (2:1.11-68ubuntu1~18.04.1) ...
Setting up openjdk-11-jdk-headless (11.0.1+9~Ubuntu18.04.1) ...
update-alternatives: using /usr/lib/jvm/java-11-openjdk-headless/bin/jconsole to provide /usr/bin/jconsole (jconsole) in auto mode
Setting up default-jdk (2:1.11-68ubuntu1~18.04.1) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for ureadahead (0.100.0-21) ...
Processing triggers for libgc-bin (2.27-3ubuntu1.4) ...
Processing triggers for systemd (237-3ubuntu0.51) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ca-certificates (20210119-18.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
done.
N2013090484-0@M2013090484-0:~$ javac Matriz.java
N2013090484-0@M2013090484-0:~$ ls
'MatrizWorker.class'  Matriz.class  Matriz.java
N2013090484-0@M2013090484-0:~$ 
```

entrega
Estatus de calificación No calificado

Figura 6. Compilación del Programa en el Nodo0

```

Actividades Terminal 17 de sep 12:40 PM
SISDIS-4CV11: Tarea 3. M X M2013090484-4 - Microsoft Edge (5) Como abrir puertos d Cómo usar SFTP para tra + 
https://m4gm.com/moodle/mod/assign/view.php?id=142
m4gm Español (Méjico) (es_mx) Jorge Luis Marquez Leon
SISDIS-4CV11
Participantes
Insignias
Competencias
Calificaciones
General
1. Introducción
2. Sincronización y coordinación
3. Sistemas basados en objetos distribuidos
4. Servicios de nombres, archivos y replicación
5. Cómputo en la nube
Tablero
Página inicial del sitio
Calendario
Archivos privados
Mis cursos
abrir puerto
17 de sep 12:40 PM
M2013090484-0@M2013090484... ~ M2013090484-1@M2013090484... ~ M2013090484-2@M2013090484... ~ M2013090484-3@M2013090484... ~ M2013090484-4@M2013090484... ~
'MatrizWorker.class'  Matriz.class  Matriz.java
N2013090484-0@M2013090484-0:~$ java Matriz 0 20.185.226.249
Esperando por conexiones...
Se conecto el nodo 1 envio la matriz A1 y B1
Termina conexion con el nodo: 1
Se conecto el nodo 2 envio la matriz A1 y B2
Termina conexion con el nodo: 2
Se conecto el nodo 3 envio la matriz A2 y B1
Termina conexion con el nodo: 3
Se conecto el nodo 4 envio la matriz A2 y B2
Termina conexion con el nodo: 4
He dejado de recibir conexiones...
Checksum: -44259
Desplegado matriz C = A x B
Matriz A:
0   2   4   6   8   10  12  14  16  18
1   3   5   7   9   11  13  15  17  19
2   4   6   8   10  12  14  16  18  20
3   5   7   9   11  13  15  17  19  21
4   6   8   10  12  14  16  18  20  22
5   7   9   11  13  15  17  19  21  23
6   8   10  12  14  16  18  20  22  24
7   9   11  13  15  17  19  21  23  25
8   10  12  14  16  18  20  22  24  26
9   11  13  15  17  19  21  23  25  27
Matriz B:
0   1   2   3   4   5   6   7   8   9
1   2   3   4   5   6   7   8   9   10
2   3   4   5   6   7   8   9   10  11
3   4   5   6   7   8   9   10  11  12
4   5   6   7   8   9   10  11  12  13
5   6   7   8   9   10  11  12  13  14
6   7   8   9   10  11  12  13  14  15
7   8   9   10  11  12  13  14  15  16
8   9   10  11  12  13  14  15  16  17
9   10  11  12  13  14  15  16  17  18
Matriz C:
0   2   4   6   8   10  12  14  16  18
1   3   5   7   9   11  13  15  17  19
2   4   6   8   10  12  14  16  18  20
3   5   7   9   11  13  15  17  19  21
4   6   8   10  12  14  16  18  20  22
5   7   9   11  13  15  17  19  21  23
6   8   10  12  14  16  18  20  22  24
7   9   11  13  15  17  19  21  23  25
8   10  12  14  16  18  20  22  24  26
9   11  13  15  17  19  21  23  25  27

```

entrega
Estatus de calificación No calificado

Figura 7. Ejecución del Programa Con N=10

The screenshot shows a Moodle assignment page titled "SISDIS-4CV11: Tarea 3". The assignment is named "M2013090484-0@M2013090484-0". The task is "Desplegando matriz C = A x B". The page displays two matrices, A and B, and their product C. Matrix A is a 9x9 matrix with values from 0 to 9. Matrix B is a 9x9 matrix with values from -18 to 0. Matrix C is a 9x9 matrix with values ranging from -1050 to 900. The Moodle interface includes a sidebar with course navigation and a footer with search and filter options.

Figura 8. Ejecución del Programa Con $N=10$

This screenshot shows the same Moodle assignment page as Figure 8, but with a different set of matrices. Matrix A is a 9x9 matrix with values from 9 to 27. Matrix B is a 9x9 matrix with values from -18 to 0. Matrix C is a 9x9 matrix with values ranging from -2265 to 570. The Moodle interface is identical to Figure 8, including the sidebar and footer.

Figura 9. Ejecución del Programa Con $N=10$

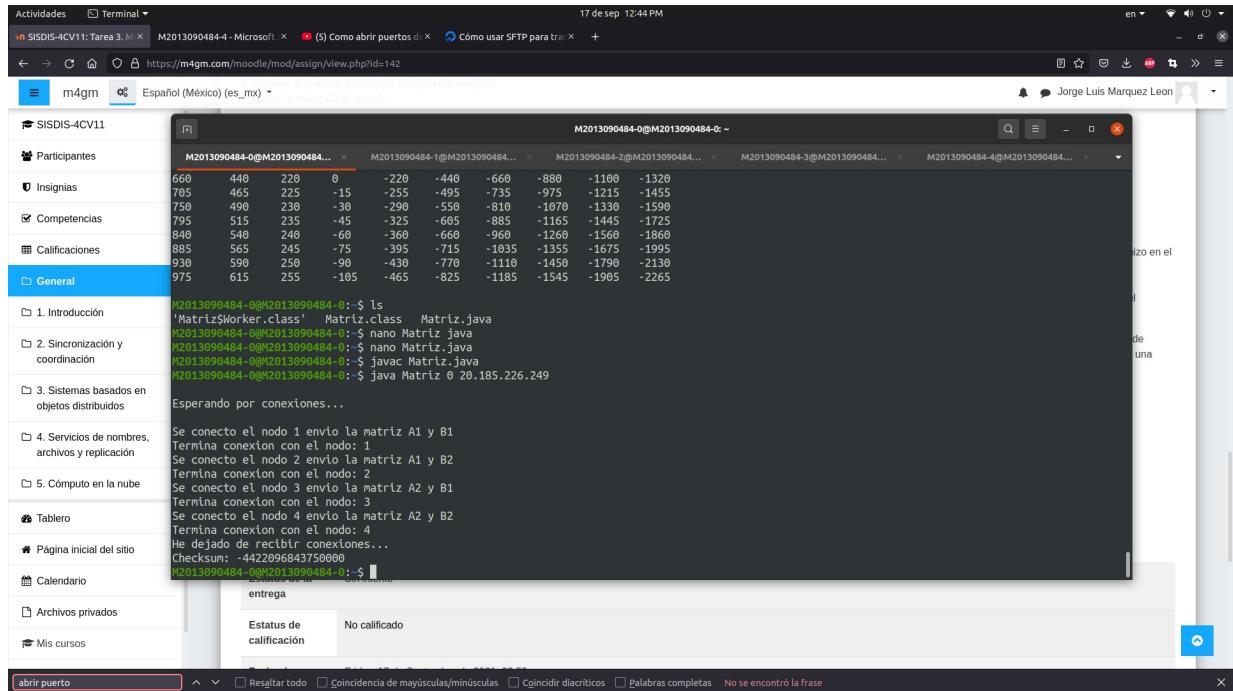


Figura 10. Ejecución del Programa Con $N=1500$

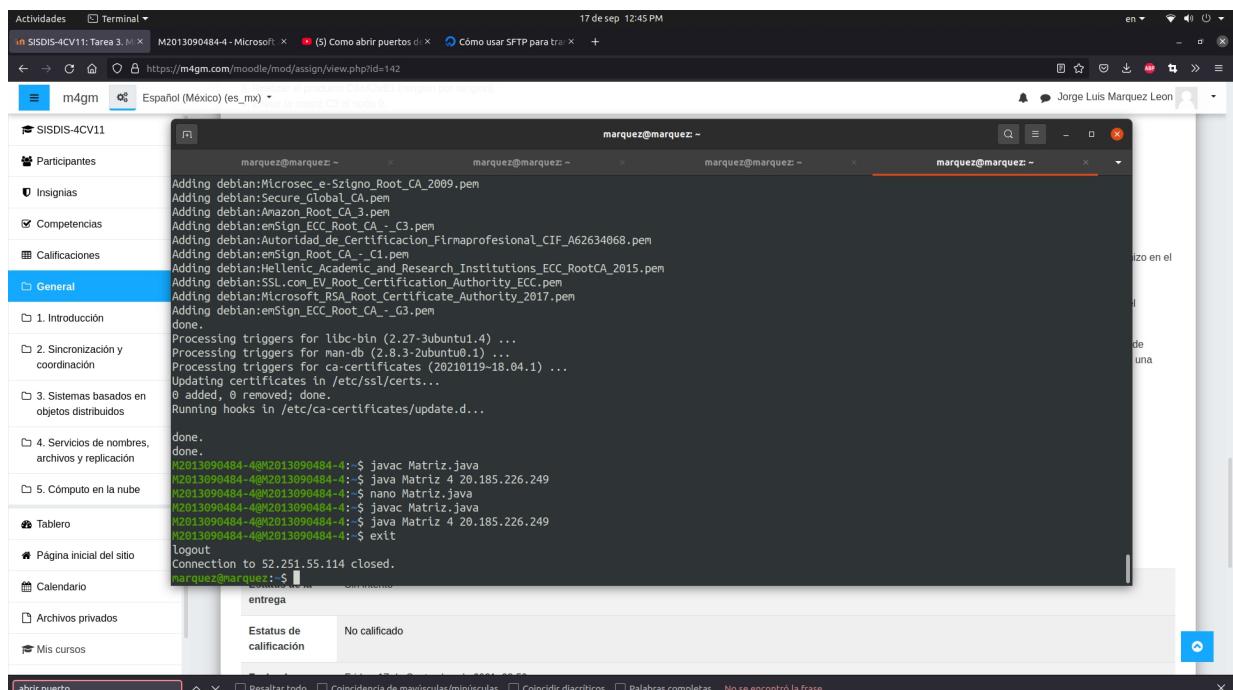


Figura 11. Desconexión de las máquinas virtuales en la nube.

The screenshot shows the Microsoft Azure portal interface. The left sidebar has a tree view with 'M2013090484-0_group' selected under 'Grupos de recursos'. The main content area lists various resources: M2013090484-0-ip, M2013090484-0-nsg, m2013090484-0634, M2013090484-0_disk1_f1b81dc7f8a4a9cad4b1f7faf98e82c, M2013090484-0_group-vnet, M2013090484-1, M2013090484-1-ip, M2013090484-1-nsg, m2013090484-1470, M2013090484-1_disk1_054e9900d50d4b16ab897013bd7398a5, M2013090484-2, and M2013090484-2-in. The right side shows a 'Notificaciones' (Notifications) panel with a log of recent actions, including the creation of security rules named 'Port_5000'.

Figura 12. Eliminación del Grupo de Recursos

Conclusión.

Es muy útil el uso de maquinas en la nube para distribuir un problema matemático que necesita de cantidades de procesamiento altas, esta vez se ejecutó vía Secure Shell usando computadoras de Microsoft Azure, el uso de terminal no difiere en demasía con mi máquina host ya que uso Ubuntu por lo que el uso de los comandos no se me dificultó.