

PRACTICA CON SPARK Y WORKERS

1. Cambia el tamaño de la matriz N y compara los tiempos

Estos tiempos son con 3 worker activos.

The screenshot shows the Spark Master UI at `spark://192.168.56.1:7077`. It displays cluster statistics: 3 Alive, 0 Dead, 0 Decommissioned, 0 Unknown cores in use (24 total, 0 used), 44.8 GiB memory in use (Total 0.0 B used), and 0 applications running. Below this, the 'Workers (3)' section lists three workers with their addresses, states, cores, and memory usage. The 'Running Applications (0)' and 'Completed Applications (0)' sections are empty.

Worker Id	Address	State	Cores	Memory
worker-20260102121221-192.168.56.1-51734	192.168.56.1:51734	ALIVE	8 (0 Used)	14.9 GiB (0.0 B Used)
worker-20260102121227-192.168.56.1-51742	192.168.56.1:51742	ALIVE	8 (0 Used)	14.9 GiB (0.0 B Used)
worker-20260102121232-192.168.56.1-51756	192.168.56.1:51756	ALIVE	8 (0 Used)	14.9 GiB (0.0 B Used)

Matriz N = 128

```
# --- Generar matrices aleatorias ---
N = 128
A = np.random.randint(0, 10, (N, N))
B = np.random.randint(0, 10, (N, N))
C = C - N // 2
● Tiempo total: 14.19 segundos
```

PS C:\Users\MARIA> & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:23:13 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
● Tiempo total: 14.19 segundos

Matriz N = 256

```
# --- Generar matrices aleatorias ---
N = 256
A = np.random.randint(0, 10, (N, N))
B = np.random.randint(0, 10, (N, N))
C = C - N // 2
● Tiempo total: 23.95 segundos
```

PS C:\Users\MARIA> & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:24:18 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
● Tiempo total: 23.95 segundos

Matriz N = 512

```

50  # --- Generar matrices aleatorias ---
51  N = 512
52  A = np.random.randint(0, 10, (N, N))
53  B = np.random.randint(0, 10, (N, N))
54
55  L = N // 2

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

```

PS C:\Users\MARIA & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:16:38 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
⌚ Tiempo total: 69.09 segundos

```

2. Cambiar el número de workers locales

Aquí cambia a 1 activo

Spark Master at spark://192.168.56.1:7077

URL: spark://192.168.56.1:7077

Workers: 1 Alive, 2 Dead, 0 Decommissioned, 0 Unknown

Cores in use: 0 Used

Memory in use: 14.9 GiB Total: 0.0 B Used

Resources in use:

Applications: 0 Running, 7 Completed

Drivers: 0 Running (0 Waiting), 0 Completed (0 Killed, 0 Failed, 0 Error, 0 Relaunching)

Status: ALIVE (Environment, Log)

Workers (3)

Worker Id	Address	State	Cores	Memory
worker-20260102121221-192.168.56.1-51734	192.168.56.1:51734	ALIVE	8 (0 Used)	14.9 GiB (0.0 B Used)
worker-20260102121227-192.168.56.1-51742	192.168.56.1:51742	DEAD	8 (0 Used)	14.9 GiB (0.0 B Used)
worker-20260102121232-192.168.56.1-51756	192.168.56.1:51756	DEAD	8 (0 Used)	14.9 GiB (0.0 B Used)

3. Sustituye el algoritmo por np.dot(A,B)

Numpy es más rápido que Spark. Numpy ejecuta operaciones vectorizadas directamente en memoria RAM utilizando bibliotecas de bajo nivel altamente optimizada

```

✓ Multiplicación Spark completada.
⌚ Tiempo Spark: 17.88 segundos

--- Ejecutando NumPy dot (Local) ---
✓ Multiplicación NumPy completada.
⌚ Tiempo NumPy: 0.017 segundos

```

4. Observa en el Spark Web UI el DAG y las etapas

Spark DAG

StrassenFinal application |

Spark Jobs (7)

User: MARIA
Started At: 2026/01/02 12:35:33
Total Uptime: 34 s
Scheduling Mode: FIFO
Active Jobs: 1

Event Timeline

Executors: Added (Blue), Removed (Red)

Jobs: Succeeded (Blue), Failed (Red), Running (Green)

Timeline: 35, 40, 45, 50, 55, 60 (2026/01/02 12:35 to 2026/01/02 12:36)

Active Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	collect at c:\Users\MARIA\Desktop\spark\strassen_spark.py:66 collect at c:\Users\MARIA\Desktop\spark\strassen_spark.py:66	2026/01/02 12:35:37	30 s	0/1	0/4 (4 running)

Spark stage

5. Mide el speedup. Tiempo con 1 worker / tiempo con 3 worker

En el ejercicio 1 esta el tiempo trabajando con 3 worker

Realizo la prueba para los tiempo trabajando con 1 worker

Matriz N = 128

```

50  # --- Generar matrices aleatorias ---
51  N = 128
52  A = np.random.randint(0, 10, (N, N))
53  B = np.random.randint(0, 10, (N, N))

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

PS C:\Users\MARIA> & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
PS C:\Users\MARIA> & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:42:42 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
● Tiempo total: 11.51 segundos

```

Matriz N = 256

```

50  # --- Generar matrices aleatorias ---
51  N = 256
52  A = np.random.randint(0, 10, (N, N))
53  B = np.random.randint(0, 10, (N, N))

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

CORRECTO: el proceso con PID 10768 (proceso secundario de PID 25224)
ha sido terminado.
● & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:40:47 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
● Tiempo total: 18.02 segundos

```

Matriz N = 512

```

50  # --- Generar matrices aleatorias ---
51  N = 512
52  A = np.random.randint(0, 10, (N, N))
53  B = np.random.randint(0, 10, (N, N))
54
55  k = N // 2

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

PS C:\Users\MARIA> & C:/Users/MARIA/AppData/Local/Programs/Python/Python311/python.exe c:/Users/MARIA/Desktop/spark/strassen_spark.py
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
26/01/02 12:38:03 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
✓ Multiplicación completada.
● Tiempo total: 78.28 segundos

```

En este ejercicio me di cuenta de que el verdadero cuello de botella no fue la potencia de los ordenadores, sino el tiempo que se pierde organizando todo. Al usar tres workers, el sistema tarda más tiempo preparando las piezas de la matriz y mandándolas por la red que lo que tarda realmente en hacer la multiplicación. Básicamente, para una matriz de este tamaño, sale más a cuenta que un solo worker haga todo el trabajo a que Spark pierda tiempo repartiendo tareas y esperando a que los demás terminen.