15 DE ABRIL DE 2020

Departament d'Enginyeria Informàtica i Matemàtiques

PRÁCTICA 1:

MULTIPLICACIÓN DE MATRICES

SISTEMAS DISTRIBUIDOS 2019-2020

DAMIÁN MALENO GONZÁLEZ
GEOVANNY RISCO CARRERA

1. Decisiones de diseño

La práctica se divide en 3 partes bien diferenciadas: inicialización, map y reduce. De entre éstas, las más compleja es la parte de inicialización, ya que el resultado de ésta será el que indique a las funciones map y reduce qué submatrices multiplicar y a qué lugar de la matriz C corresponde el resultado. Dado que se nos pedía una versión no óptimizada de la multiplicación de matrices, no hemos tenido en cuenta los casos en los que el número de workers fuera mayor que "m" y menor que "m" x "l". A continuación se explica el funcionamiento general de programa:

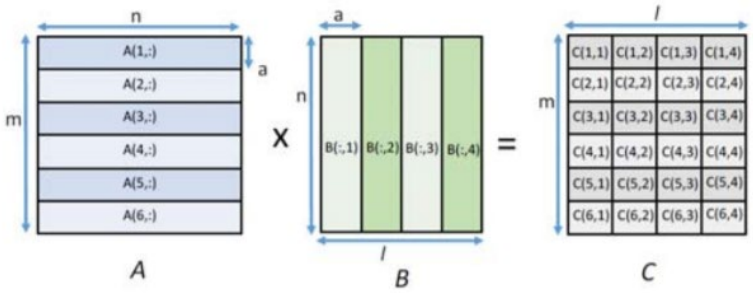


Figura 1. Representación gráfica de la multiplicación de matrices.

Input: Matriz A ∈ $\mathbb{R}^{m\ x\ n}$ y Matriz B ∈ $\mathbb{R}^{n\ x\ l}$ generadas aleatoriamente.
Resultado: Matriz C ∈ $\mathbb{R}^{m\ x\ l}$ como resultado de A x B

1. **Inicialización**.
    - 1.1 Subdivide las matrices A por filas y las sube al IBM COS
    - 1.2 Opción a: Si el nº de workers es m o inferior no se subdivide la matriz B y se sube al IBM COS la matriz B entera.
      Opción b: Si el nº de workers es m x l se subdivide la matriz B por columnas (se procede tal como está en la Figura 1) y se suben las submatrices al IBM COS.
    - 1.3 Se crea el iterdata, el cual es una lista de diccionarios que contiene los identificadores de las submatrices a multiplicar y la posición que ocupará el resultado en la matriz C.
2. **Map.**
    - 2.1 Obtiene la submatriz A y B correspondiente del IBM COS.
    - 2.2 Multiplicarlas.
3. **Reduce**.
    - 3.1 Recibe todos los resultados de cada worker
    - 3.2 Crea una matriz C y añade los resultados en la posición correspondiente.
    - 3.3 Sube al IBM COS la matriz C.

2. Análisis de los gráficos

Para estudiar el comportamiento de nuestro código y poder analizar si existe un *Speed-Up* al realizar la multiplicación de matrices en diferentes hilos respecto a una multiplicación de matrices secuencial, hemos realizado las siguientes ejecuciones:

- Multiplicación de dos matrices de: 10x10, 50x50, 100x100, 250x250, 500x500 y 700x700.
- Para cada tamaño de matriz realizar la ejecución con las cantidades de *workers* más representativas: 1, 2, 5, 10, 25, 50, 100 (para matrices más pequeñas adaptar el número de *workers*).

Como podemos observar en las gráficas presentadas en el anexo de gráficos, obtenemos datos representativos a partir de la multiplicación de matrices de 100x100 (50x50 empieza a mostrar la tendencia), ya que el *workload* es lo suficientemente grande como para poder observar cambios en los tiempos de ejecución.

Tendencia observada: Cuando el *workload* es lo suficientemente grande vemos que la multiplicación secuencial de la matriz tarda más que al realizarla con un número pequeño de *workers*. Observamos que los mayores *Speed-Up* se producen cuando el número de workers bajo, pero superior a 1.

El hecho de que observemos una reducción en el tiempo de ejecución cuando aumentamos el número de *workers* (número de *workers* pequeño) y luego veamos un aumento en el tiempo de ejecución (número de *workers* grande) puede deberse a varios motivos. En nuestro caso pensamos que la explicación es la siguiente:

A la disminución en el tiempo de ejecución que nos proporciona realizar el cálculo con varios *workers* hay que restarle el costo temporal que supone trabajar con varios *workers* (el ordenador ha de crear más paquete de información y enviarlos a través de la red). En el caso de trabajar con multiplicaciones de matrices de valores comprendidos entre 0 y 100, suponemos que el coste temporal de transmisión de los paquetes a través de la red suele superar al coste de computo en la plataforma *cos*, por lo que cuando aumentamos mucho el número de *workers* obtenemos peores *Speed-Ups*.

Ilustramos esta explicación con el ejemplo de multiplicación de una matrizA 700x700 * matrizB 700x700:

- N_workers = 1, el coste de ejecutar este cálculo de manera secuencial es costoso por lo que obtenemos un tiempo de ejecución alto.
  T.E. = 8,905 s
- N_workers = 5, el coste de ejecutar este cálculo dividido en 5 hilos es menor al de ejecutarlo de manera secuencial, pese a que el coste de comunicación sea mayor la suma de estos no supera el T.E. de la ejecución secuencial.
  T.E. = 7,190 s
- N_workers = 100, el coste de ejecutar este cálculo dividido en 100 hilos es menor al de ejecutarlo de manera secuencial pero el coste de comunicación es mucho mayor, por lo que el T.E. es muy superior al de trabajar de manera secuencial.
  T.E. = 17,977 s

→ Hemos llegado a esta conclusión ya que observamos las mayores diferencias en los tiempos de ejecución al utilizar una conexión a internet más lenta.

3. Reparto de trabajo

Se utilizó un repositorio en Github para trabajar de manera conjunta. Se puede acceder mediante mediante el siguiente link https://github.com/geovalexis/task1-SD.git, aunque tal vez no tengas acceso porque es un repositorio privado. En caso de que sea necesario lo podemos hacer público o bien darte acceso como contribuidor.

En todo momento se trabajó de forma coordinada discutiendo y trabajando el diseño del programa asi como elaborando el código en Python utilizando *Visual Studio Code*. En general se puede decir que cada uno participó mayoritariamente en las siguientes funciones:

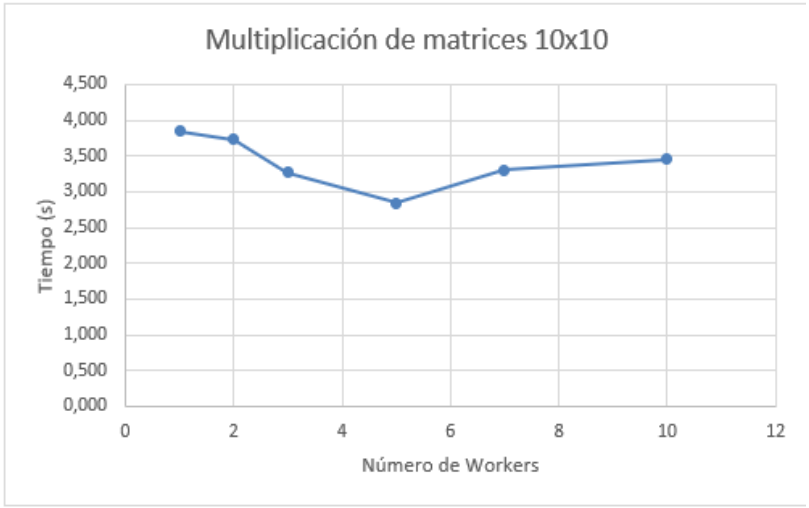
- Damian Maleno: función de inicialización y análisis de resultados.
- Geovanny Risco: funciones map y reduce, y decisiones de diseño.

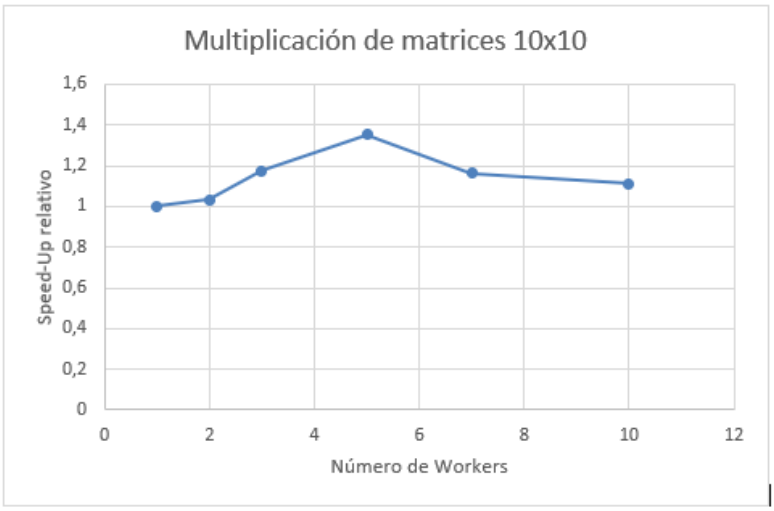
4. Referencias

- Funcionamiento e implementación de pywren: https://github.com/pywren/pywren-ibm-cloud
- Programación en Python: https://runestone.academy/runestone/books/published/pythonds/index.html
- Librería de Python para las operaciones matemáticas: https://docs.scipy.org/doc/numpy/index.html
- Diapositivas, fórum y apuntes de clase.

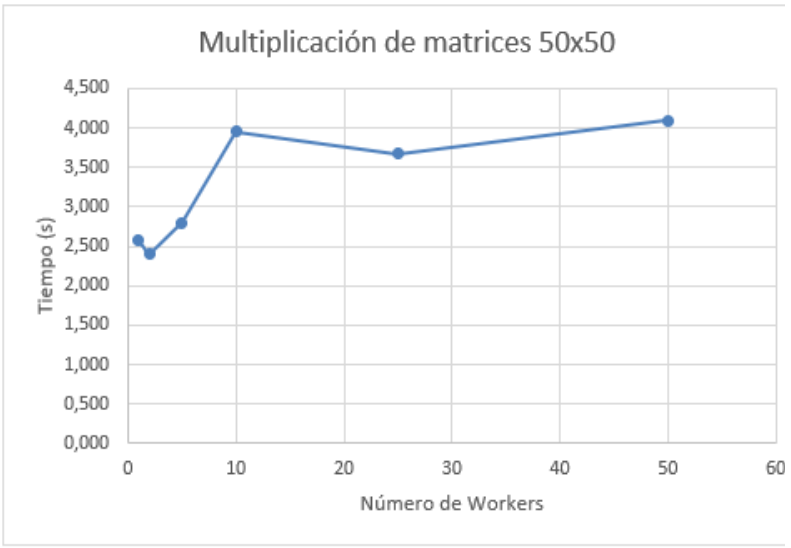
ANEXO I: Gráficos

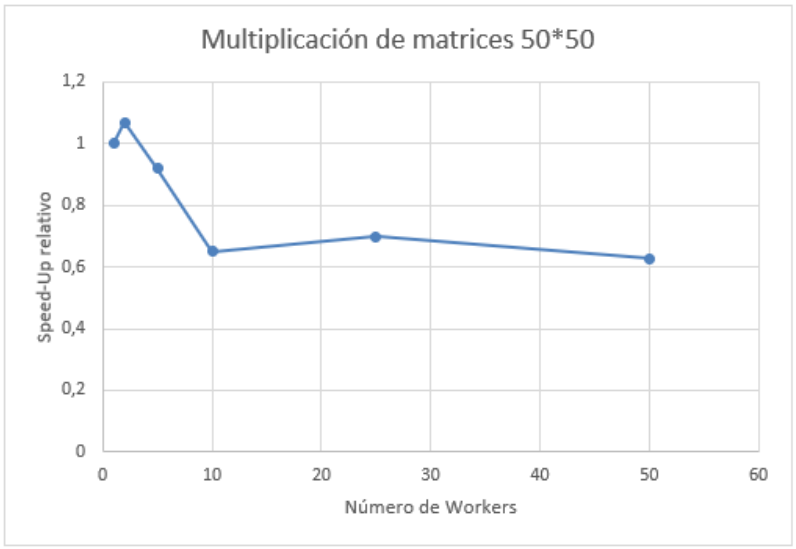
Multiplicación de matrizA 10x10 * matrizB 10x10



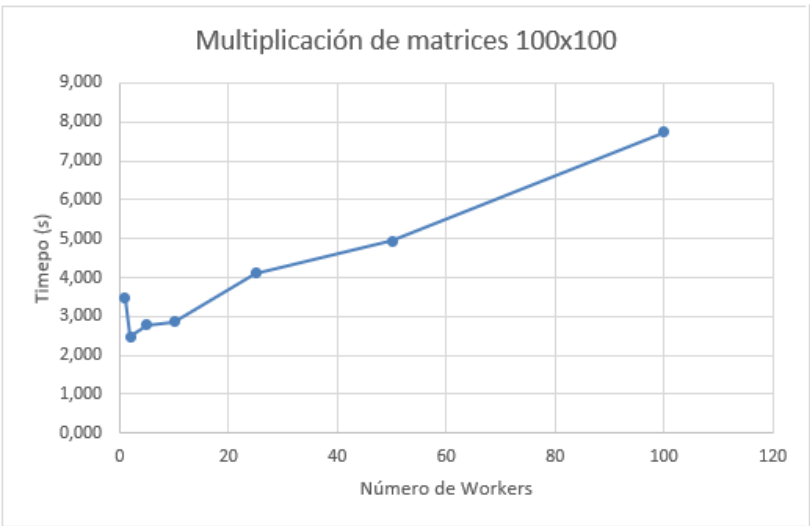


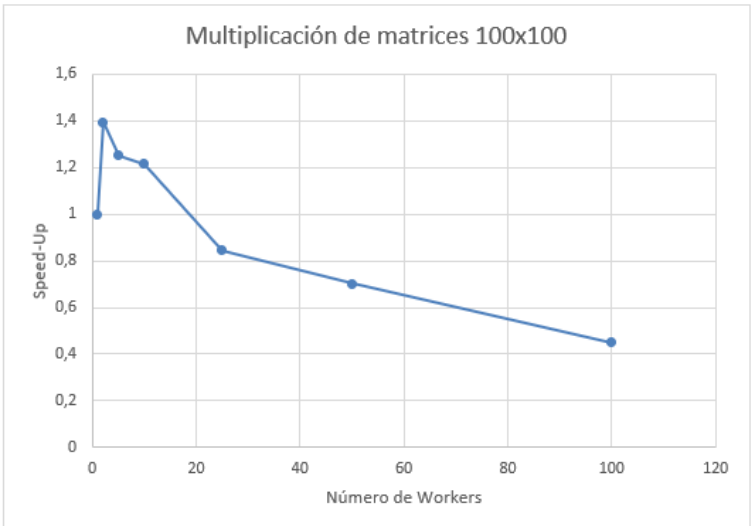
Multiplicación de matrizA 50x50 * matrizB 50x50



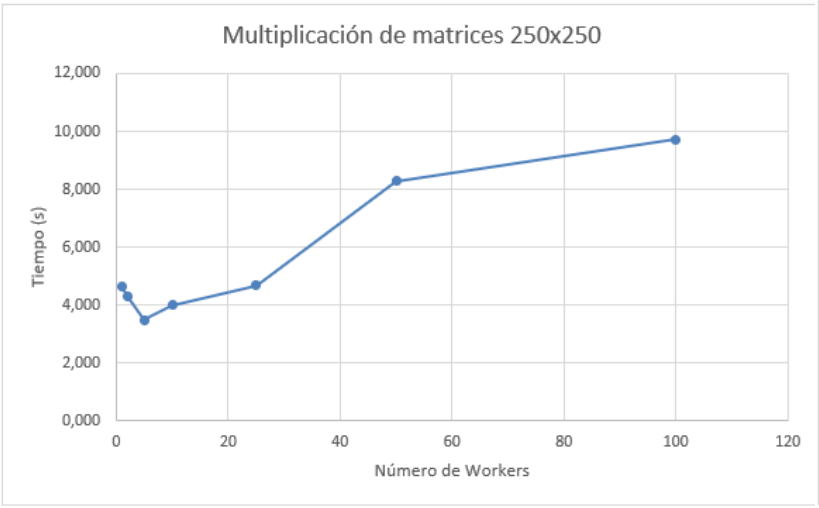


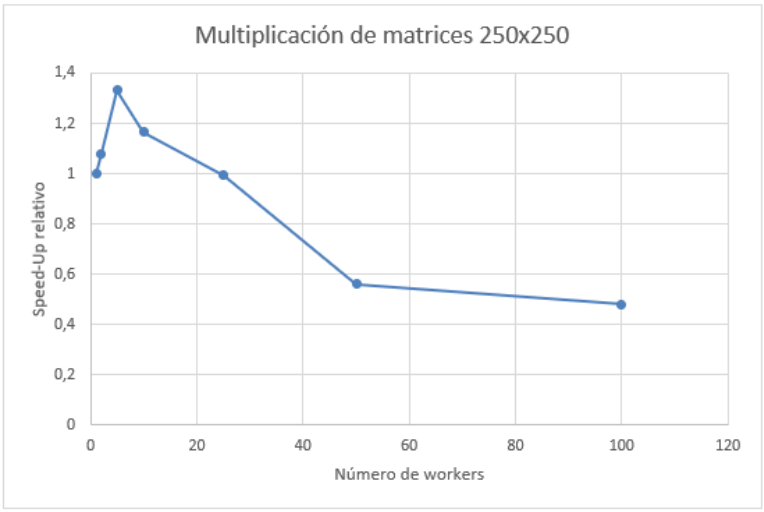
Multiplicación de matrizA 100x100 * matrizB 100x100



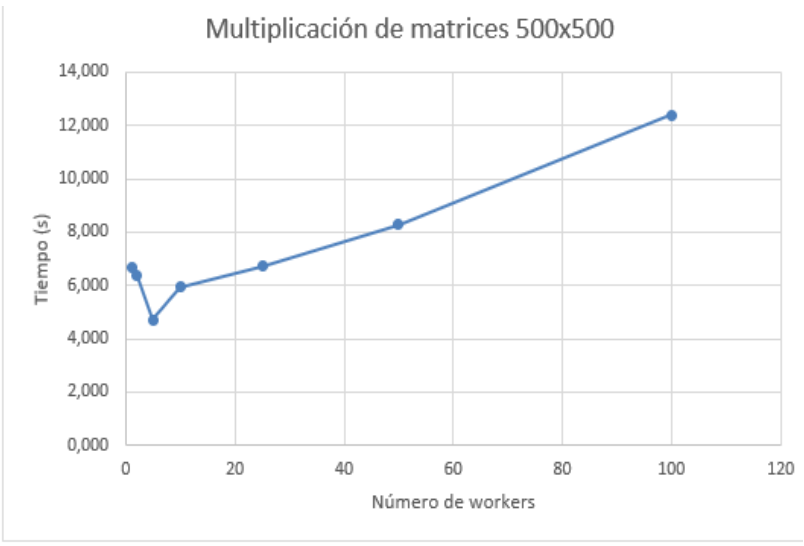


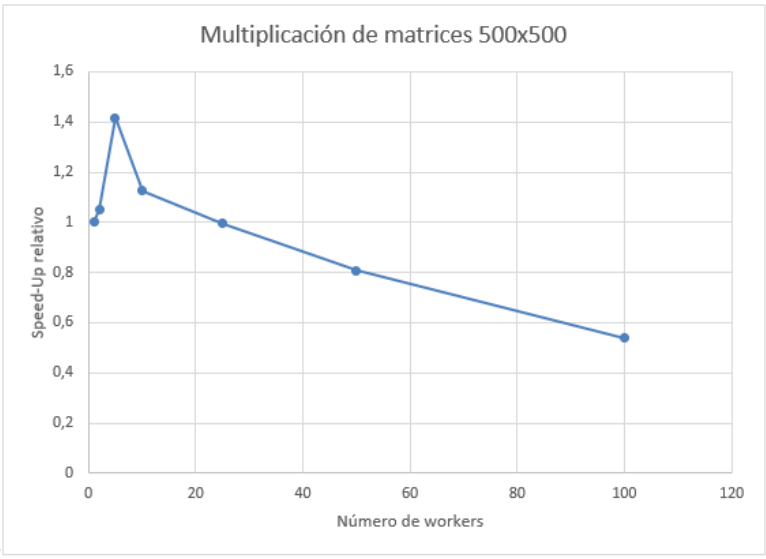
Multiplicación de matrizA 250x250 * matrizB 250x250





Multiplicación de matrizA 500x500 * matrizB 500x500





Multiplicación de matrizA 700x700 * matrizB 700*700