SISTEMES DISTRIBUITS

Pràctica 1

Albert Cañellas Solé
albert.canellas@estudiants.urv.cat
Laura Romero Huete
Laura.romeroh@estudiants.urv.cat

Problema

El problema que es planteja és la multiplicació de dues matrius a través *IBM-PyWren*, un projecte *open source* que permet executar codi *Python* sobre *IBM Cloud Functions* i executar codi usuari en una plataforma *serverless* sense necessitat de saber com les funcions són invocades i executades. La multiplicació s'ha de poder dividir en submatrius per tal de que cada worker calculi una submatriu de la matriu resultant. Per tant l'arquitectura del sistema consistirà en l'ús d'un o múltiples workers per realitzar les multiplicacions de les submatrius i un *object storage Store* com és el *IBM COS*.

Solució

La solució que presentem esta composta per 4 funcions i la classe main, cada una de les funcions realitza una part necessària per la multiplicació de matrius: Guardar Matriu, Multiplicar les submatrius (map) i Ordenar les submatrius (reduce).

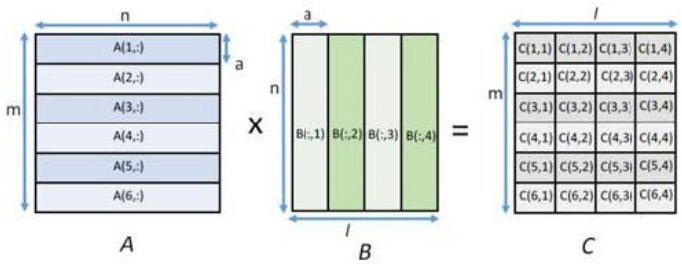
La solució la hem programat els dos junts gràcies al Teams Viewer.

Main()

En el main primerament inicialitzem un rellotge per calcular el temps que triga en realitzar-se la multiplicació. Un cop inicialitzat el rellotge creem els executors del *IBM-PyWren* per poder treballar al cloud i cridem a la funció guardar_matriu() de manera asíncrona per guardar les submatrius al *IBM_COS*. Un cop guardades les submatrius cridem la funció map_reduce, on la funció map multiplicarà les submatrius amb un nombre variable de workers i la funció reduce ordenarà les submatrius resultants. Finalitzada la multiplicació de les matrius s'atura el rellotge i s'imprimeixen els resultats per pantalla.

GuardarMatriu()

La funció principal de guardar matriu es dividir les matrius inicials en files i columnes i guardar-les al *IBM_COS*. Nosaltres ho hem fet en funció de 'a', entenent 'a' com el numero de files o de columnes que tindran les submatrius, anomenarem la primera matriu A i la segona B. Primerament agafem 'a' files de de la matriu A i recorrerem la matriu B per columnes. Guardant les primeres 'a' files de A amb les primeres 'a' columnes de B, i així successivament fins tenir totes les columnes de B. Seguidament agafarem les següents a files de A (si en queden) i tornarem a recorre la matriu B. Cada grup de 'a' files i 'a' columnes es guardarà al cos en un únic fitxer.



Il·lustració 1 Representació de la multiplicació de dues matrius dividies per 'a' (Assignment#1 SD)

Així dons, si tenim una matriu A 4x4 i una matriu B 4x4 i volem fer la seva multiplicació amb 'a'=2, tindríem 4 fitxers al cos.

En el cas de que el número de files o de columnes d'alguna de les matrius no sigui divisible per 'a', l'últim grup de files o columnes que s'agafi serà de mida inferior a 'a'. Cada fitxer portarà com a nom la posició de la matriu resultant (C) on s'ha de col·locar la submatriu.

Multiplicació()

La funció multiplicació s'encarrega de dur a terme la multiplicació de la submatriu de files i la submatriu de columnes que trobem dins del fitxer. Primerament ens descarreguem el fitxer corresponent del *IBM_COS* que conté un array on la primera posició es on troben les files (corresponents a la matriu A) i seguidament llegirà la segona posició on es troben les columnes (corresponents a la matriu B). Per tal de fer la multiplicació de la matriu cal transposar les columnes de la matriu B, ja que al fitxer estan guardades com a files i per fer la multiplicació es necessari que estiguin organitzades com columnes. Un cop transposades es realitza la multiplicació.

La funció multiplicació serà la funció cridada per el map.

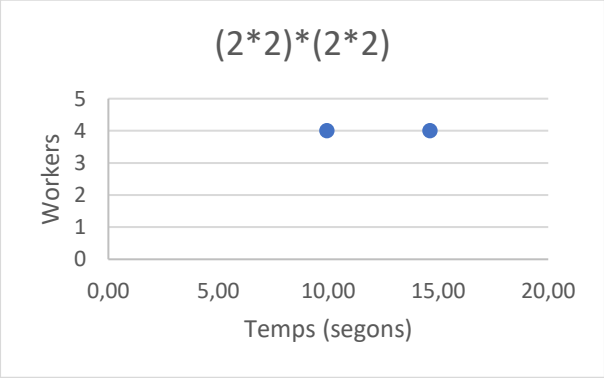
Ordenar()

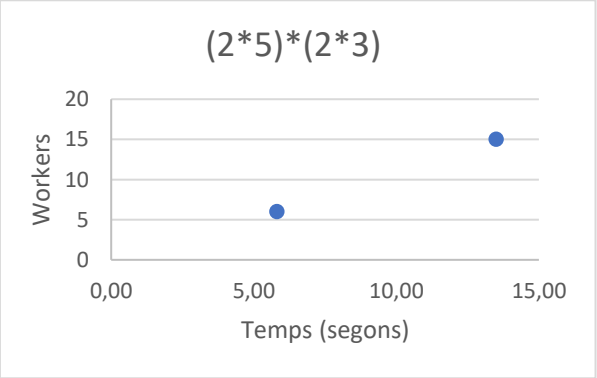
La funció ordenar s'encarrega de llegir totes les submatrius i col·locar-les en una matriu final C. Per tal de fer-ho s'inicialitza una matriu C amb zeros, i a mida que es vagin llegint les submatrius s'aniran substituint els zeros de la matriu C pels valors de les submatrius.

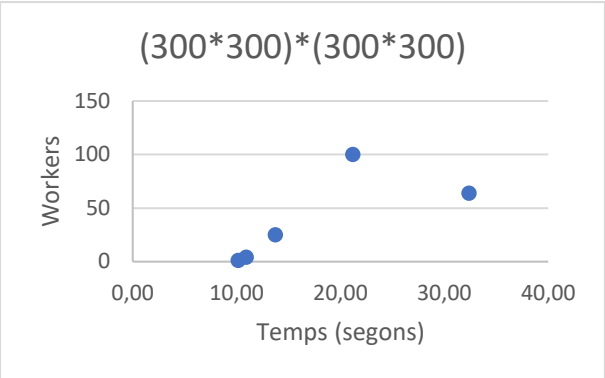
Primerament es llegeixen les primeres submatrius i es concatenen una al costat de l'altre en una matriu transitòria de 'a' files fins que s'han omplert totes les columnes que correspondrien amb la mida de C. Aleshores es copiarà aquesta matriu transitòria a la matriu final C i es buidarà la matriu transitòria. A continuació, si encara queden submatrius, es llegiran la següents submatrius i s'aniran concatenant fins que s'hagin omplert totes les columnes de C. Un cop plena la matriu transitòria es copiaran les files de la matriu transitòria a la matriu C. D'aquesta manera s'anirà formant una matriu final C que contindrà els resultats de les submatrius.

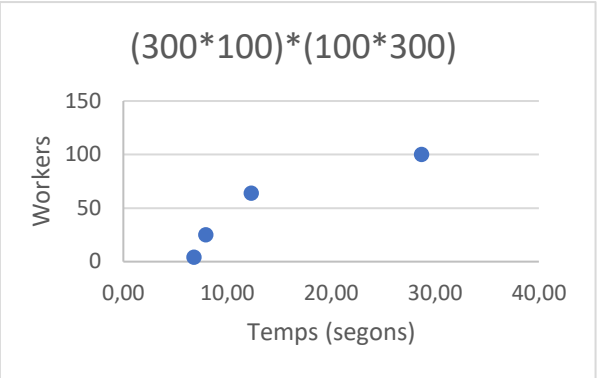
Joc de proves

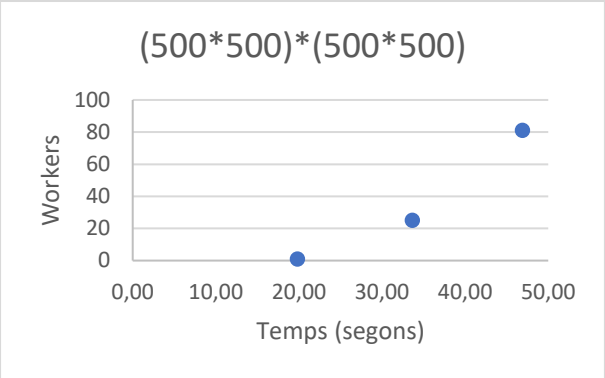
Per tal d'avaluar el bon funcionament del programa s'han realitzat varies proves amb matrius quadrades i rectangulars de diverses mides amb diferent divisions de 'a'.











Conclusió

Un cop realitzats varis jocs de proves, ens adonem que no es redueix el temps realitzant la multiplicació amb més workers sinó que aquest augmenta. Ens esperàvem que el temps disminuís a mida que s'augmentava el numero de workers, però això nomes ens hi hem trobar amb la multiplicació de dues matrius de 300*300 on amb 100 workers el temps d'execució es menor que amb 64 workers.