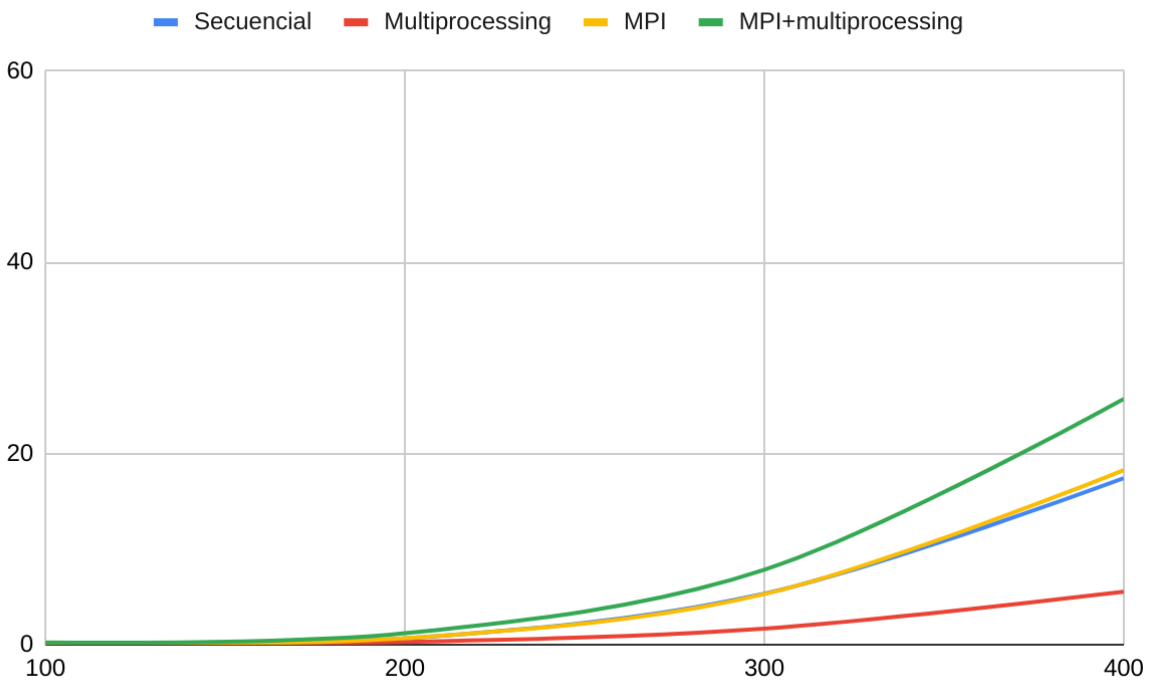
Ejercicio 3, Práctica 3

Guillermo Marquínez

En esta práctica voy a implementar la multiplicación por matrices, utilizando la notación por columnas, y haciendo uso del módulo MPI para que cada procesador realice la multiplicación de cada uno de los cuatro bloques en los que dividimos las matrices. Vamos a comparar tiempos, por un lado haremos la multiplicación en forma secuencial, por otro lado haremos la multiplicación utilizando multiprocessing, después haremos la multiplicación utilizando MPI y por último MPI+multiprocessing. Veremos qué opción es la más eficiente. Utilizando heracles para matrices cuadradas de dimension 100, 200, 300 y 400 los tiempos son los siguientes.



El modelo más eficaz claramente es utilizando multiprocessing. El más lento es utilizando MPI +multiprocessing. Que es lo contrario a lo que debería pasar. Puede que se deba a dos motivos; el primero, cada CPU tiene que crear dos variables de memoria compartida y eso hace ralentizar bastante el programa. El segundo motivo es que al ejecutar el programa cada CPU no tenga la capacidad de realizar en paralelo cuatro subprocesos. En realidad deberíamos coger una CPU formada por 4 cores. En python multiprocessing y multithreading no son lo mismo, por lo que podría ser que para aprovechar los hilos de cada procesador hubiera que usar multithreading y no multiprocessing. Recordamos que multiprocessing crea procesos completamente independientes con memorias independientes. Por lo que podría ser que no aprovechara los hilos de cada procesador.