|  |  |  |  |
| --- | --- | --- | --- |
| Simulations | Serial | Threading | Multiprocessing |
| 25 | 3.8283 | 3.8751 | 7.1877 |
| 50 | 7.6096 | 7.6245 | 14.0786 |
| 75 | 11.6410 | 11.5004 | 21.1257 |
| 100 | 15.2505 | 15.2661 | 28.2197 |

In theory, threading should be faster than serial and multiprocessing should be faster than threading.

Serial < Threading <Multiprocessing

Computationally bound programs are faster in multiprocessing while I/O bound processes are supposed to be faster in threading.

Processes spawn slower than a thread can, but multiprocessing dodges the global interpreter lock, which is what prevents 2 threads from writing to the same memory at the same time.

I have no idea what is bogging down these programs and have flat run out of time to figure it out. I understand the theory, but I do not understand what is happening inside the python code to cause it to be slower.