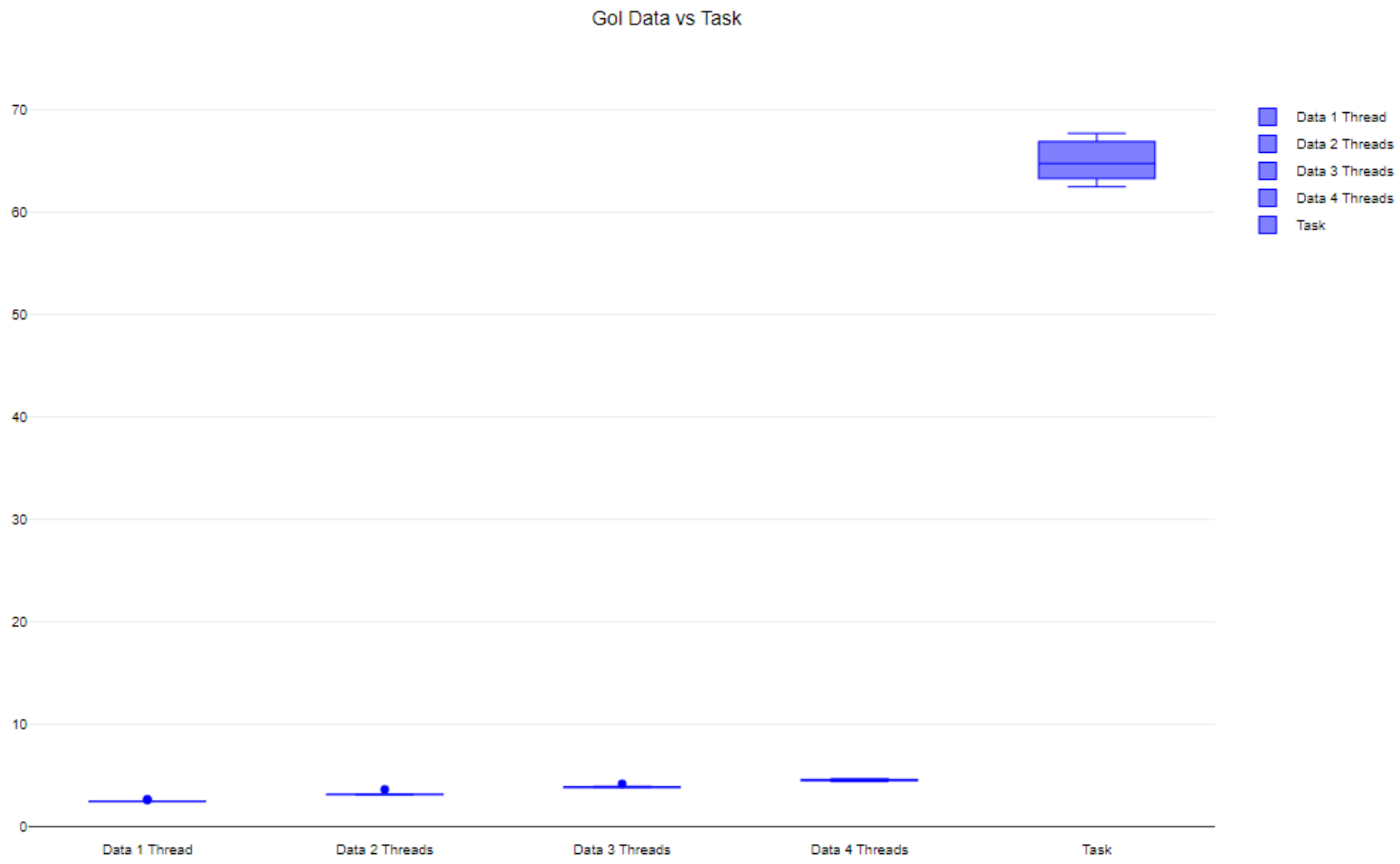
CIS3090 A1 Report

Roman Savelyev
ID: 1106331

cat /proc/cpuinfo | grep processor | wc -l: 8 cores



Prior to beginning I had a feeling that gol_task would take a longer time to complete, simply due to the fact that even with having 3 threads and 1 main thread, a lot of the processes are being repeated, such as the reading/writing of the empty/live cells. Gol_task would write to the queue and then from the queue write to the grid, meanwhile gol_data was writing straight to the grid, and even then was splitting that writing into x number of threads. Not to mention the number of mutexes that are in place in order to

protect the reading/writing of data, causing bigger grids to take a lot longer to generate due to needing to read data atomically (basically one at a time). In regards to gol_data, I was expecting better results for when more threads are added, at the very least see a parabolic pattern. Due to my docker limiting me to 8 cores, as per the command line instruction given to us, the program was using 1/4 of the threads provided, assuming that each core can use up to 2 threads. The more threads we use the more chances that a thread will share a core with another thread, as well as be on standby to be queued to not drain all computer resources. This means that in the worst case scenario, the slowest thread to run, might also be the last to start, and will force all other threads to wait for it to finish.