Alex Adrian Avila

Parallel Computing

Project 2 Map Reduce

**Introduction**

For this project our task was to read many files and count the number appearances of certain words that they appear throughout the whole set of files. This task is not that hard to implement in serial form, but we can improve the run time by using multiple threads to read more files at once and count their appearances and for that we will use the map-reduce algorithm.

**Problems**

While working on the assignment I first read the files in series but then I started implementing the parallel version using the pymp library. When I wanted to implement my algorithm my plan was to split the different files through each thread using iterate so that threads that finish would be able to use other files. However I was not able to run the pymp.iterate function and I ended up having to use the pymp.range function which use uses random splits but it ended up working fine.

**Performance**

Chart, line chart

Description automatically generated

**Analysis**

At first my hypothesis was that because my computer only has 4 cores that times was not going to decrease as I increase the number of threads when use more than for but it ended up being helpful still. I think that the reason of this is because when threads finish, they must wait but that is the chance that other threads to use the available cores.

**Conclusion**

I feel like this assignment just like the other one was not as hard as I thought it was going to be it took me around 3 hours to implement the whole thing in code mainly because of my problems. The map-reduce algorithm was interesting problem to get data from different source like other computers or databases, and I hope I can see more cases where I can use the algorithm.

**cpuDump**

model name : Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz

4 36 220