ENCE360 Assignment: jme161 27033004

Part 5 – Report

Algorithm Analysis

Describe the algorithm found in downloader.c (lines 228-264).

while file line isn't EOF

- Replace the newline character with an EOF character
- Get the number of tasks using that line and the amount of available workers
- Get the max chunk size from the number of workers

for each worker:

- add work and append a new value to the queue

while worker is greater than zero:

- decrement work and wait to get a new task to get from the queue

- merge and remove byte size files

Once al l t asks are complete, free all resources, and finish.

How is this similar to algorithms found in your notes?

This is similar to the work structure which involved using a reading and a writing semaphore in-order to communicate over multiple threads. This could be found in semephor.c.

Any improvements which could be made?

Instead of merging and removing chunk files and removing them every time is not efficient due to removing each byte chunk file after the next. This could be improved by using mmap() instead of using fopen(). Remove chunk files could also be called outside the while loop due to it removing files every iteration.

Also we could fork the process so that the child spawns new threads/workers and the parent processes the lines and stores them in a buffer. Thus allowing the threads to immediately start processing the tasks rather than waiting for there to be enough tasks.

Performance Analysis

The table shows the worker thread ranges from one to 400 for each test file. I tested the my downloader with small.txt and large.txt to Georges downloader. I found that his downloader still passed and processed some files with > 50 threads and found that he would get a connection timeout and a socket file directory error. The results for this are shown in Table 1. These results are dependent on several factors such as connection quality and processor availability as the times would very. By changing the file size this does affect the performance, as the download time increases as the file size increases.

The table below shows the optimal number of worker threads is between 5 – 50. This analysis was performed with the Erskine's lab computers.

Average				
Number of Threads	Small	Large	Small (Sample) (Georges)	Large (Sample) (Georges)
1	3.017	24.277	3.068	19.240
2	3.393	17.943	4.037	19.653
6	4.154	16.519	3.170	15.926
12	4.457	15.981	3.773	15.708
24	5.116	18.232	4.030	18.514
50	7.894	20.801	error	error
100	13.603	31.018	error	error
200	24.945	52.781	error	error
400	>30	>70	error	error

Table 1: Average results for time taken to download