CS342 Operating Systems – Spring 2023

Project #1 – Processes, IPC, and Threads

Elifsena Öz – 22002245 – Section 3

Onur Uysal – 22002609 – Section 2

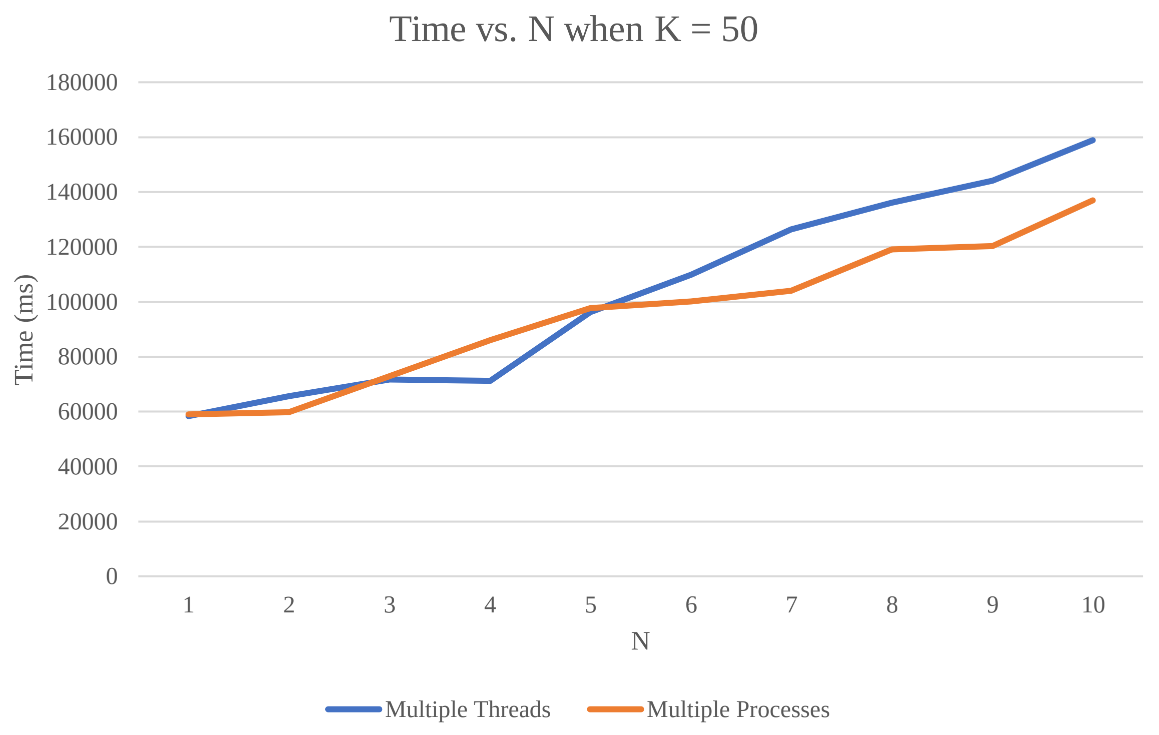
19.03.2023

Part C – Experiments

In Part A of this project, an application to find the most frequent K words in N files was developed with the help of child processes. The parent process allocated shared memory for each of the child processes to analyze exactly one file in order to find most frequent K words in that file. Then, the results from the child processes were analyzed again by the parent process to find the result. In part B of this project, the same operation was conducted using multiple threads. A doubly linked list with nodes that held both the words and their frequencies were used to hold the information in both parts.

Experiment 1

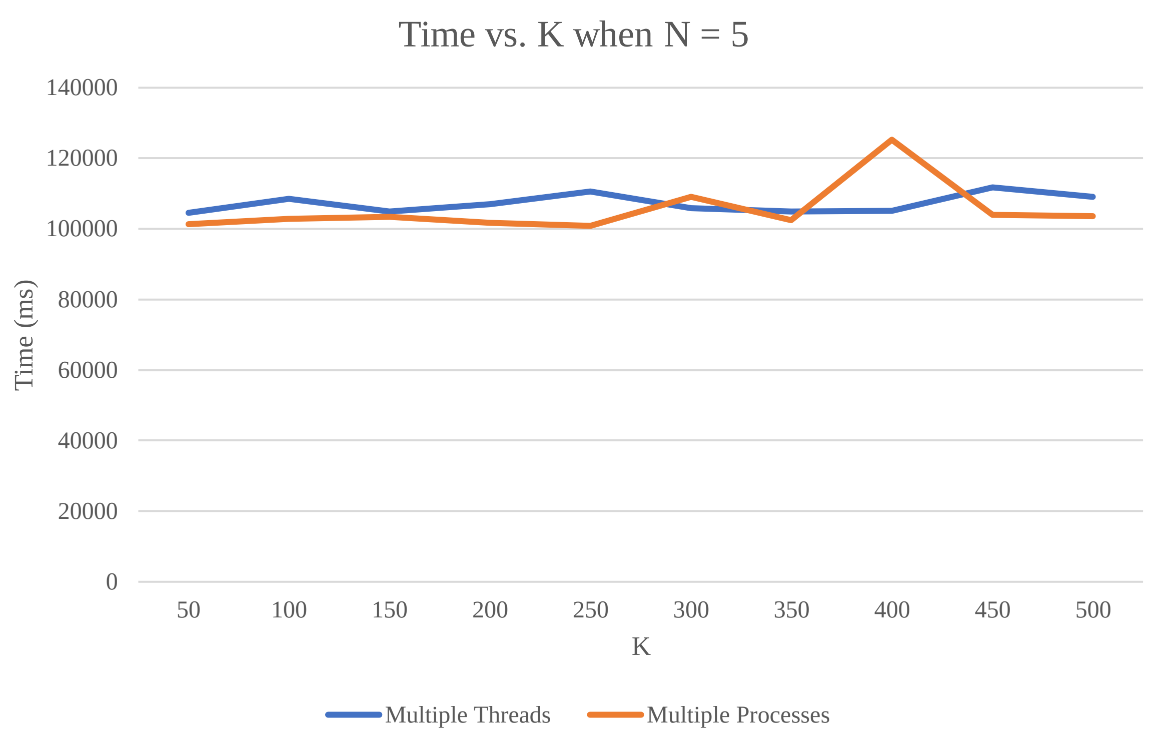
In the first experiment, the value of K was set to be 50, and the number of files were integers from 1 to 10 (N = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10).



Looking at this graph, it can be seen that the time needed to find most frequent 50 words in different numbers of files increases linearly. Until the number of files comes up to 5, Part A and Part B has performed similarly in terms of time consumed. However, it can be observed that if more than 5 files are processed multiple threads (Part B) takes more time than multiple processes (Part A). It can also be observed that running time for Part B increases more than Part A in total, as the input size grows.

Experiment 2

In the second experiment, the number of files were fixed to be 5 (N = 5) and the number of most frequent words were 50, 100, 150, 200, 250, 300, 350, 400, 450, 500 (K) respectively.



Observing this graph, it can be concluded that the number of most frequent words do not have a significant effect on the running time of the program except small irregularities. Between 50 words and 300 words multiple processes (Part A) has performed a little faster than multiple threads (Part B). Until 450 words, there seems to be some irregularity on the graph. Then, the graph continues with the same pattern. It can be concluded that processes are a little faster than threads in regular conditions.

Conclusion

From the graphs that have been plotted regarding these experiments, it can be concluded that although there was no major difference, multiple processes run faster than multiple threads in most cases. Since the threads are simply a segment of their process, their performance declines as the number of threads increases. Although a similar decline in performance of processes is seen in the first experiment, from the graph it can be concluded that the performance decline is smaller for processes. Therefore, as the complexity and the number of tasks increase, it is better to use processes for performance, while it is better to use threads when there are small number of tasks that are lightweight.