Daniel Guo ID: 913290045
Github: yiluun CSC415 Operating Systems

Assignment 4 – Word Blast

Description:
A program that parses a War and Peace text file and returns the top 10 most repeated words in the text file. The program will only use Linux functions and will utilize multithreading to divide the parsing effort.

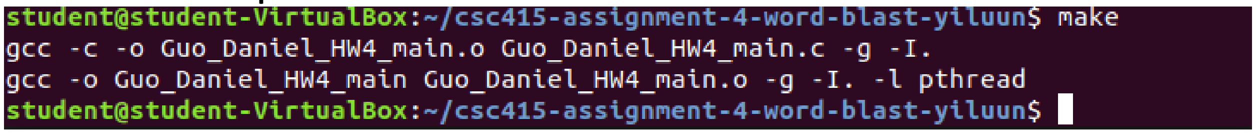
Approach / What I Did:
My approach to this assignment was to first digest and try to visualize what I am trying to solve. The program needs to take in a file and tokenize the text so that we can get all the words sorted later on. I decided to create a word structure that had word and count in it. My idea was that while parsing through the tokens, if any word was greater than 6 characters, we would copy that word into an array of that structure and tally the count attribute. Afterwards, we would have all the words and their counts saved in the global array that we can then process with a sorting algorithm. The sorting function needed me to update counts, and that is where I put my mutex locks around. The instructions told us how to figure out chunk allocation for each thread. We need to figure out the thread count from the command line arguments which tells us how many threads to create. Then, we figure out the size of the file by using lseek. Lseek requires us to get a file descriptor which we can obtain by using open with our file name. After getting this information, we can figure out the chunks each thread is being allocated by dividing file size with thread count. For creating threads, I had to look up the parameters of pthread_create() and figure out how I can get those parameters with my data. Then I would run the function with the data, along with my word counting function. After joining the threads, I would have all the words with 6 or more characters in a word structure array. I then utilized a double for loop and an "a, b, c" type swap algorithm to go through the entire word array and swap around the words until they were in order. Formatting the output text to be the same as the sample output we need to copy was the next step. I also had to iterate through the first ten elements of the array to print out the highest count words in descending order. The final part of the assignment was proper cleanup of the program. I closed the file and destroyed the mutex lock.

Issues and Resolutions:
I had a lot of memory issues, and some of the easy ones were fixed by moving around global/local variables. I am also unsure of if my locks were correctly placed. While I placed them around the word count updates, I am unsure if there is a possibility that while one thread is breaking out of the loop after finding the word isn't in the array (to put the new word in), if there is a moment where another thread puts it in during that lock break. But if I make the lock around the entire process, the runtime is way too long. I didn't figure out if my lock placement was ideal. I ran with increasing numbers of threads and attempts to see if one instance would encounter issues, but it didn't. I also had an issue with incorrect counting. I solved it by moving the condition where the program doesn't find the needed word in the array, outside of the for loop that iterates through the entire array. I had to create a variable outside the loop to keep

track of that information, so that I could use it outside of the loop. But that seemed to solve the problem with proper word count. Another problem that I had issues with that will definitely affect any future building of this program is proper cleanup. While no issues arose, that does not mean that my program is safe. I still need to learn more about how to perform proper cleanup of my program. I learned that while mutex destruction probably won't cause any immediate issues, it does not promise that it won't cause problems with further implementation. This taught me that my program breaking is not a proper way to diagnose improper cleanup, as a lot of the problems won't be seen immediately, but will leave your program and machine vulnerable.

Analysis:
There was definitely a time reduction correlation with number of threads utilized up to a certain point. I went up to a high number of threads and noticed that it would stay around 1.3 to 1.4 seconds. It was difficult because upon running the program more and more with the exact same commands, the time to complete was inconsistent as the threads increased. With 2 threads, the time difference across runs was about have a quarter of a second, but the time difference as I moved to 8 or more threads jumped to about 1 second. The time also did not get much faster as I increased the thread count and it seemed to plateau. I think that this assignment shows us Amdahl's law in effect. A program with parallel processing, like what we are doing right now with parallel parsing, has a limit in which utilizing more processor power and threads does not make the program run faster. That explains the plateau in time reduction. For the inconsistency, I am not sure what may cause that besides just random chance with more and more threads interrupting each other at inopportune times, which causes slowdown. That might explain why when the number of threads increases, it gets more inconsistent, as there are more chances for the threads to interrupt each other.

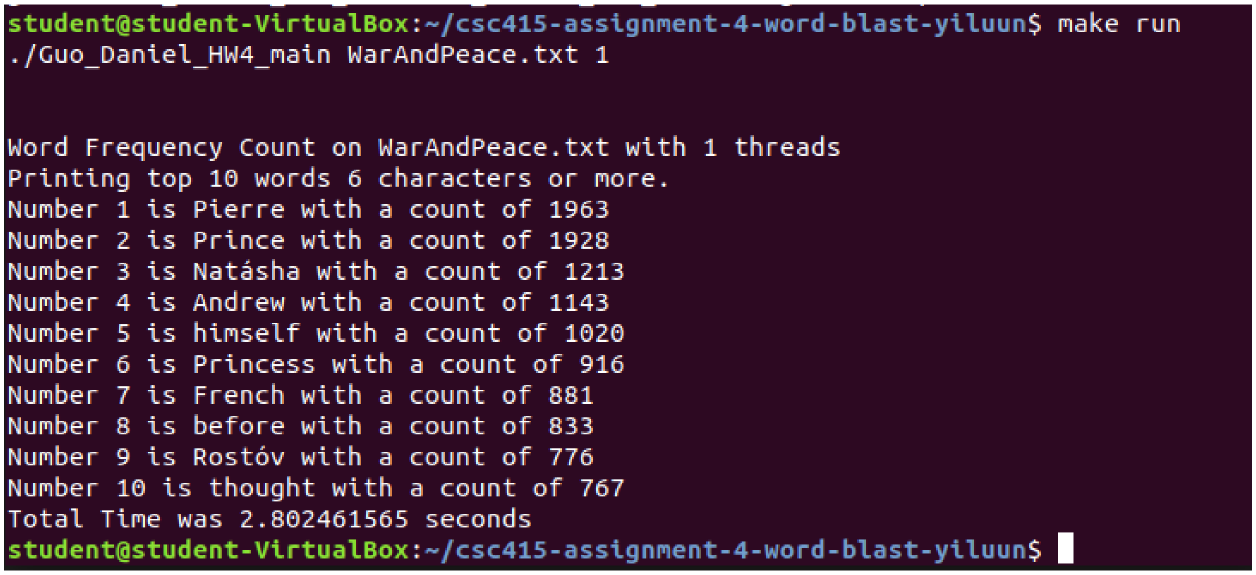
Screen shot of compilation:



```
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$ make
gcc -c -o Guo_Daniel_HW4_main.o Guo_Daniel_HW4_main.c -g -I.
gcc -o Guo_Daniel_HW4_main Guo_Daniel_HW4_main.o -g -I. -l pthread
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$
```

Screen shot(s) of the execution of the program:

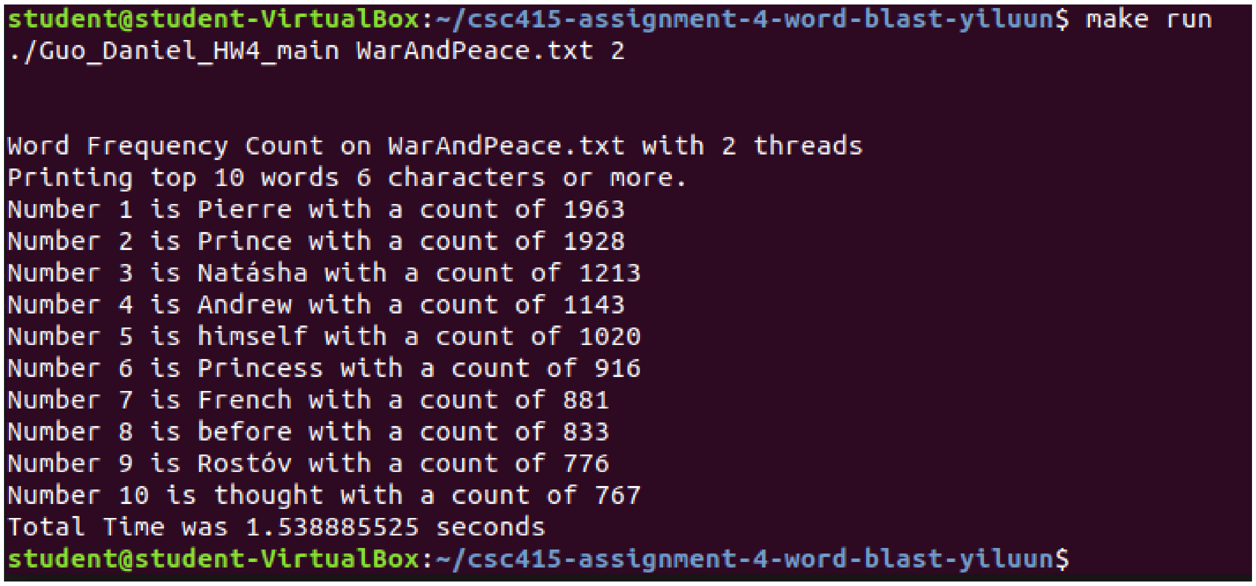
1 thread:



```
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$ make run
./Guo_Daniel_HW4_main WarAndPeace.txt 1


Word Frequency Count on WarAndPeace.txt with 1 threads
Printing top 10 words 6 characters or more.
Number 1 is Pierre with a count of 1963
Number 2 is Prince with a count of 1928
Number 3 is Natásha with a count of 1213
Number 4 is Andrew with a count of 1143
Number 5 is himself with a count of 1020
Number 6 is Princess with a count of 916
Number 7 is French with a count of 881
Number 8 is before with a count of 833
Number 9 is Rostóv with a count of 776
Number 10 is thought with a count of 767
Total Time was 2.802461565 seconds
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$
```

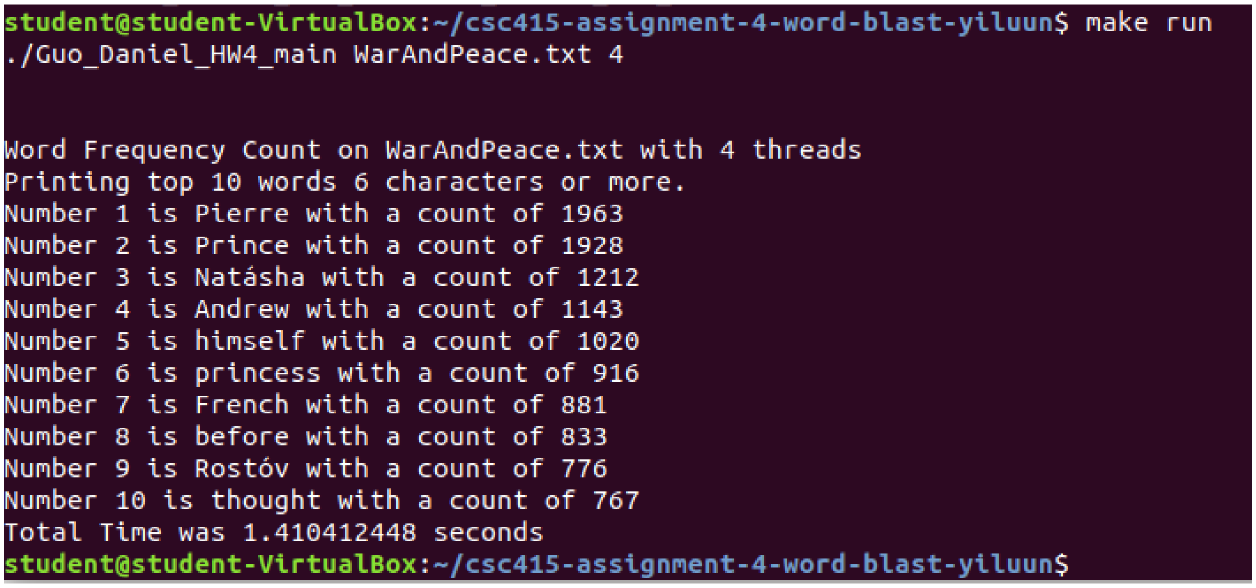
2 threads:



```
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$ make run
./Guo_Daniel_HW4_main WarAndPeace.txt 2


Word Frequency Count on WarAndPeace.txt with 2 threads
Printing top 10 words 6 characters or more.
Number 1 is Pierre with a count of 1963
Number 2 is Prince with a count of 1928
Number 3 is Natásha with a count of 1213
Number 4 is Andrew with a count of 1143
Number 5 is himself with a count of 1020
Number 6 is Princess with a count of 916
Number 7 is French with a count of 881
Number 8 is before with a count of 833
Number 9 is Rostóv with a count of 776
Number 10 is thought with a count of 767
Total Time was 1.538885525 seconds
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$
```

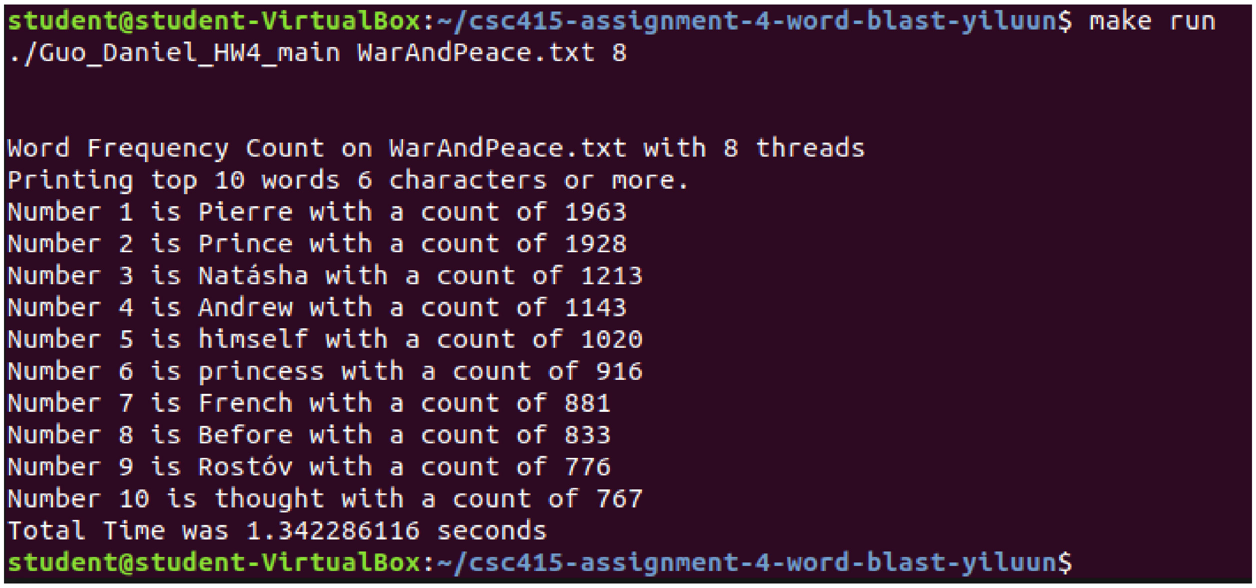
4 threads:



```
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$ make run
./Guo_Daniel_HW4_main WarAndPeace.txt 4


Word Frequency Count on WarAndPeace.txt with 4 threads
Printing top 10 words 6 characters or more.
Number 1 is Pierre with a count of 1963
Number 2 is Prince with a count of 1928
Number 3 is Natásha with a count of 1212
Number 4 is Andrew with a count of 1143
Number 5 is himself with a count of 1020
Number 6 is princess with a count of 916
Number 7 is French with a count of 881
Number 8 is before with a count of 833
Number 9 is Rostóv with a count of 776
Number 10 is thought with a count of 767
Total Time was 1.410412448 seconds
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$
```

8 threads:



```
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$ make run
./Guo_Daniel_HW4_main WarAndPeace.txt 8


Word Frequency Count on WarAndPeace.txt with 8 threads
Printing top 10 words 6 characters or more.
Number 1 is Pierre with a count of 1963
Number 2 is Prince with a count of 1928
Number 3 is Natásha with a count of 1213
Number 4 is Andrew with a count of 1143
Number 5 is himself with a count of 1020
Number 6 is princess with a count of 916
Number 7 is French with a count of 881
Number 8 is Before with a count of 833
Number 9 is Rostóv with a count of 776
Number 10 is thought with a count of 767
Total Time was 1.342286116 seconds
student@student-VirtualBox:~/csc415-assignment-4-word-blast-yiluun$
```