Operating Systems Project (CLO3)

Complex Engineering Problem Attribute Justification

Attribute	Justification
WP1 Depth of knowledge	Deep knowledge of multi-threaded programming is needed
WP3 Depth of analysis	Detailed analysis of the performance bottlenecks in a multi-threaded application is needed
WP7 Interdependence	The performance of the program needs to be maximized. If a large number of threads are chosen, there would likely be a benefit in CPU-based computations. However, the overhead of having multiple threads and the memory access lately can degrade. Therefore, the tradeoff between threads and performance needs to be analyzed carefully

Objective:

The objective of this project is to familiarize yourself with file handling, multithreaded programming, code analysis and optimization. Your code will create a histogram of characters present in a large text file. You will create two lightweight web crawlers. The main function will read a text file with a single web link per line. Each link will have a book written in plain text. Sample links are given below:

Alice's Adventures in Wonderland: Link
The Picture of Dorian Gray: Link
DRACULA: Link
Pride and Prejudice: Link
The Complete Works of William Shakespeare: Link

Instructions:

1. Your program will have three command line arguments.
    - Argument 1 will be the path of the single text file have single URL per line.
    - Argument 2 will be the total number of threads
    - Argument 3 will be X, an integer. This integer is used in point 4.
2. Your program will have a global data structure called word_char_count with 26 integer variables. Each variable will count the number of words starting with the characters of the english alphabet.
3. Multithreaded Fetching: The URLs will be evenly divided between the threads, to the extent possible. Each thread will read the text from the webpages assigned to it.
4. Each thread will read X words. Aftering reading X words, it will update the global data structure word_char_count. You need to ensure that this update operation in thread safe.
5. The thread will then proceed to reading the next X words, and updating word_char_count. This will need to continue until all text assigned to the thread is processed.

The threads are required to:

- Handle cases like failed HTTP requests, invalid URLs, or timeouts gracefully.
- Use a lightweight library like libcurl for HTTP requests.
- Synchronization primitives (e.g., pthread_mutex_t) for thread safety

Submissions

You will need to submit well commented code. For each code, you need to submit:

1. C file
2. Timing report. Segment your code into reasonable “operations” and profile the different operations to figure out which part of your code is taking the maximum amount of time. The timing analysis should be performed for multiple argument configurations: <number of threads><values of X>
    a. <number of threads> in {1,2,4,8}
    b. <X> in {1, 32, 256, 1024, 4096}
3. Explain and justify the timing differences you observe for the different configurations.
4. Create a graph which has the number of threads on the x-axis and the completion time on the y-axis. Plot this for X=1024
5. Report the steps you took to optimize the performance of your code