ECE 362 – Embedded Operating Systems

Assignment 4

Total Points: 100

Submission Deadline: March 12 2024

This is an individual assignment. Please do not complete in groups.

This assignment will provide you with an opportunity to experiment with parallel programming and the *pthread* library functions. To complete the assignment, you will need to modify a provided base program to run with multiple threads. *This base program may not be efficient or have any practical application.*

The base algorithm randomly fills a 2-dimensional array with either 0 or 1 values and then searches for horizontal and vertical sequences of 1s at each array positions. For example, consider the following 3x3 array:

0 1 1
0 1 0
1 0 1

- If the search were for sequences of length 3, the algorithm would return 0.
- If the search were for sequences of length 2, the algorithm would return 2. From row 1, column 2 both a horizontal sequence and vertical sequence of length 2 can be found.
- If the search were for sequences of length 1, the algorithm would return 10. There are 5 positions with a 1 value and each begins a horizontal and vertical sequence of length 1.

-r <num>	Number of rows in the array. Default (and max) value is 16000.
-c <num>	Number of columns in the array. Default (and max) value is 16000.
-l <num>	length of sequences to find: Default is 20.
-s <num>	Random seed used to initialize randomization.
You should add a command line option to identify the number of threads the program should use.	
-t<num>	Number of threads: Value must be 1,2,4,8, or 16. Include error checking in your code so that you don't create more than 16.

Deliverables:

a. A revised program that supports running the algorithm with multiple threads(seek.c).

b. A brief report that describes your conclusions on the use of threads from running

experiments with different arguments. To assess how long the program takes to complete each experiment, use the Linux **time** command.

Recommendations:

• Revise the code to split the work up into subtasks based on the -t value. For example, if there are 32 rows and 4 sub tasks, each task would evaluate 8 distinct rows. Confirm your code works before using threads to do the work!

• Create a separate program to experiment with threads. Carefully review the APUE text,

- Your first thread program might create one thread, leveraging the APUE code.
- The program could then be expanded to create some fixed number of threads (for example 4 threads). Please check that exactly the number you specify are created.
- Next, enhance your program to accept a command line parameter indicating how many threads to run. Check this code works without creating threads!

• Regularly run the following command. Especially before logging off.

ps -ef | grep $USER

This will help you determine if runaway processes. If you do, enter the command kill -9 pid, where pid is the process you want to terminate.