Paul Kummer
Dr. Hanku Lee
Operating Systems CSIS 430
09/22/2021

two_threads.c review

Overview of assignment 03

Assignment 03 is a C program that uses threads to accomplish two different tasks of computing a factorial and approximating pi. To do this, the main function takes in two command line arguments. The first argument is used for the number that will be computed for factorial and the second argument is the number of iterations done to compute pi more accurately. In the main function, two threads are created with their own thread identifications. In the process of creating the threads, each thread will execute a different function of calculating a factorial or calculating pi with a void pointer that points to a variable of type double as its parameter. The main process will wait for each thread to accomplish their task before continuing to its last statement that returns the int zero, indicating it is finished.

There are a total of four functions in the program, excluding the main function. Two of the functions return void pointers and are used exclusively by a thread and are “approx_pi” and “factorial”. The other two functions, “fact” and “pow”, are used within the threads because the math.h library is not used, and some mathematical operations were needed for the threads to accomplish their tasks.

The function “factorial” is used by a thread and computes the factorial of whatever number is passed as an argument. The argument is passed as a void pointer and must be cast as a pointer to the data type of double. Then the pointer is dereferenced, resulting in the actual number used as an argument in the “fact” function. Finally, when the fact function returns its result, it is displayed to the terminal window.

The other function used by a thread is “approx_pi”. This function uses the Gregory-Leibniz series to compute the value of pi by summing a mathematical formula a number of times specified by the argument passed into the function. The mathematical formula uses the other math functions of “math” and “fact” during the computation. After the number is summed the specified number of times, it is multiplied by four. Then the result is displayed to the terminal. With every additional iteration of the summing, the result of pi will become closer to its actual value. For instance, if the number five is passed as an argument, its result will be farther away from the real value of pi than if the argument of one hundred million would have been used.

To avoid the “math.h” library and reuse some code for another function, the function “fact” is created that accepts one argument as a double. That argument will be used in a for loop

to multiply a resulting number by the next incremented value from one to the value of the argument. The result of the multiplications will be returned to the caller.

Another function used to compute a mathematical operation is “pow”. It will multiply a number times itself a specified number of times. The first argument is the number that will be multiplied by itself, and the second argument is the number of times to multiply the number. An iterative loop is used to do the multiplications. Then the result is returned to the caller.

Lessons Learned

This assignment helped reinforce the syntax of the C language and how to use multiple threads in a program. It taught me how to pass parameters into functions when a new thread is created. I also learned how to change a void pointer into useful data, which directly correlates to passing a parameter into a function with threads. Another important concept that was reinforced was the use of “pthread_join” to wait for a thread to finish so the program doesn’t terminate before a thread can accomplish its task.

Furthermore, I learned about different ways that pi is calculated. Some of the ways pi can be calculated can result in numbers overflowing, resulting in bad values. Instead of using values that can become very large with factorial or powers, I used a formula that kept the values smaller so I didn’t have to try to make mathematical functions as good as the standard library’s math.h file.