COS3043
System Fundamentals

Lab 3

Task

Develop and execute a program using any thread library to create number of threads specified by the user, each thread independently generate a random integer as upper limit. Then it computes and prints the prime numbers list where the length of the list is less than or equal to that upper limit.

Description of Basic Functions Required for The Program

What is a Thread?

- Technically, a thread is defined as an independent stream of instructions that can be scheduled to run as such by the operating system. But what does this mean?
- To the software developer, the concept of a "procedure" that runs independently from its main program may best describe a thread.

What are Pthreads?

- Pthreads are defined as a set of C language programming types and procedure calls, implemented with a pthread.h header/include file and a thread library - though this library may be part of another library, such as lab, in some implementations.

Why Pthreads?

- In the world of high performance computing, the primary motivation for using Pthreads is to realize potential program performance gains.
- When compared to the cost of creating and managing a process, a thread can be created with much less operating system overhead. Managing threads requires fewer system resources than managing processes.
- All threads within a process share the same address space. Inter-thread communication is more efficient and in many cases, easier to use than inter-process communication.

The Pthreads API:

The original Pthreads API was defined in the ANSI/IEEE POSIX 1003.1 - 1995 standard. The POSIX standard has continued to evolve and undergo revisions, including the Pthreads specification. The subroutines which comprise the Pthreads API can be informally grouped into four major groups:

1. Thread management: Routines that work directly on threads - creating, detaching, joining, etc. They also include functions to set/query thread attributes (joinable, scheduling etc.)

2. Mutexes: Routines that deal with synchronization, called a "mutex", which is an abbreviation for "mutual exclusion". Mutex functions provide for creating, destroying, locking and unlocking mutexes.

3. Condition variables: Routines that address communications between threads that share a mutex. Based upon programmer specified conditions. Functions to set/query condition variable attributes are also included.

4. Synchronization: Routines that manage read/write locks and barriers.

Thread Management

pthread_create (thread, attar, start routine, rag):

- Initially, your main () program comprises a single, default thread. All other threads must be explicitly created by the programmer.
- pthread_create creates a new thread and makes it executable. This routine can be called any number of times from anywhere within your code.

pthread_create (thread, attar, start routine, rag):

- pthread_create arguments:
  - Thread: An opaque, unique identifier for the new thread returned by the subroutine.
  - Attr: An opaque attribute object that may be used to set thread attributes. You can specify a thread attributes object, or NULL for the default values.
  - Start_routine: the C routine that the thread will execute once it is created.
  - Arg: A single argument that may be passed to start_routine. It must be passed by reference as a pointer cast of type void. NULL may be used if no argument is to be passed.

pthread_exit (): There are several ways in which a thread may be terminated:

- The thread returns normally from its starting routine. Its work is done.
- The thread makes a call to the pthread_exit subroutine - whether its work is done or not.
- The entire process is terminated due to making a call to either the exec() or exit()
- If main() finishes first, without calling pthread_exit explicitly itself.

Joining: pthread_join (threadid, status)

- The pthread_join () subroutine blocks the calling thread until the specified threadid thread terminates.
- The programmer is able to obtain the target thread's termination return status if it was specified in the target thread's call to pthread_exit ().
- A joining thread can match one pthread_join () call. It is a logical error to attempt multiple joins on the same thread.

Detaching: The pthread_detach () routine can be used to explicitly detach a thread even though it was created as joinable.

pthread_self (): Pthread_self returns the unique, system assigned thread ID of the calling thread.

General Idea

- A function to check if an argument is a prime number.
- A function to print the list of prime numbers generated.
- A thread function to generate a random number.
- A main function to get input from users of how many threads to be generated.