**Assignment #2, Fork and Join Implementation using PThread**

**Due 11:55PM 10/11, Wednesday**

In this assignment, you will develop a library using PThreads, including PThreads mutex and condition variable, to implement OpenMP parallel construct, barrier construct, single construct and critical construct. The library realizes the basic fork-join execution model of OpenMP and should include at least the following functions:

void kmp\_parallel(int num\_threads, void \*(\*parallel\_func)(void \*), void \* argu);

int kmp\_get\_thread\_num( );

int kmp\_get\_num\_threads( );

void kmp\_barrier( );

int kmp\_single(void \* (\*single\_func)(void \*), void \* argu);

void kmp\_critical(void \* (\*critical\_func)(void \*), void \* argu);

A program, forkjoin.c, is provided for you to add your implementation and to experiment those functions. Your implementation should pass that test program without bugs. For the programming, I recommend you first implement the kmp\_parallel and make sure it works using the parallel\_func\_simple function. Second, you implement kmp\_critical and kmp\_barrier using pthread\_mutex. Thirdly, implement kmp\_single. Lastly, implement kmp\_get\_thread\_num and kmp\_get\_num\_threads and make sure the kmp\_parallel passes the test with the provided parallel\_func\_all test function.

Your submission should include a single source file of the library, and necessary comments should be added in the source code to explain the major data structure and the important logics in your implementation.