Project #3 – The Mutex Stack

Young-Joon Park

parky8@oregonstate.edu

1. Tell what machine you ran this on

   This was run on a native Ubuntu laptop with an Intel Core i5-8250U CPU.

2. Tell what operating system you were using

   The computer is running a Linux (Ubuntu) operating system.

3. Tell what compiler you used

   g++ compiler was used to run the program.

4. Include, in your writeup, the pieces of code where you implemented the mutexes

```
void
Push( int n )
{   if(USE_MUTEX) {
        omp_set_lock(&Lock);
    }
    StackPtr++;
    Stack[StackPtr] = n;
    if(USE_MUTEX) {
        omp_unset_lock(&Lock);
    }
}
```

   Push's critical section wrapped with mutex lock

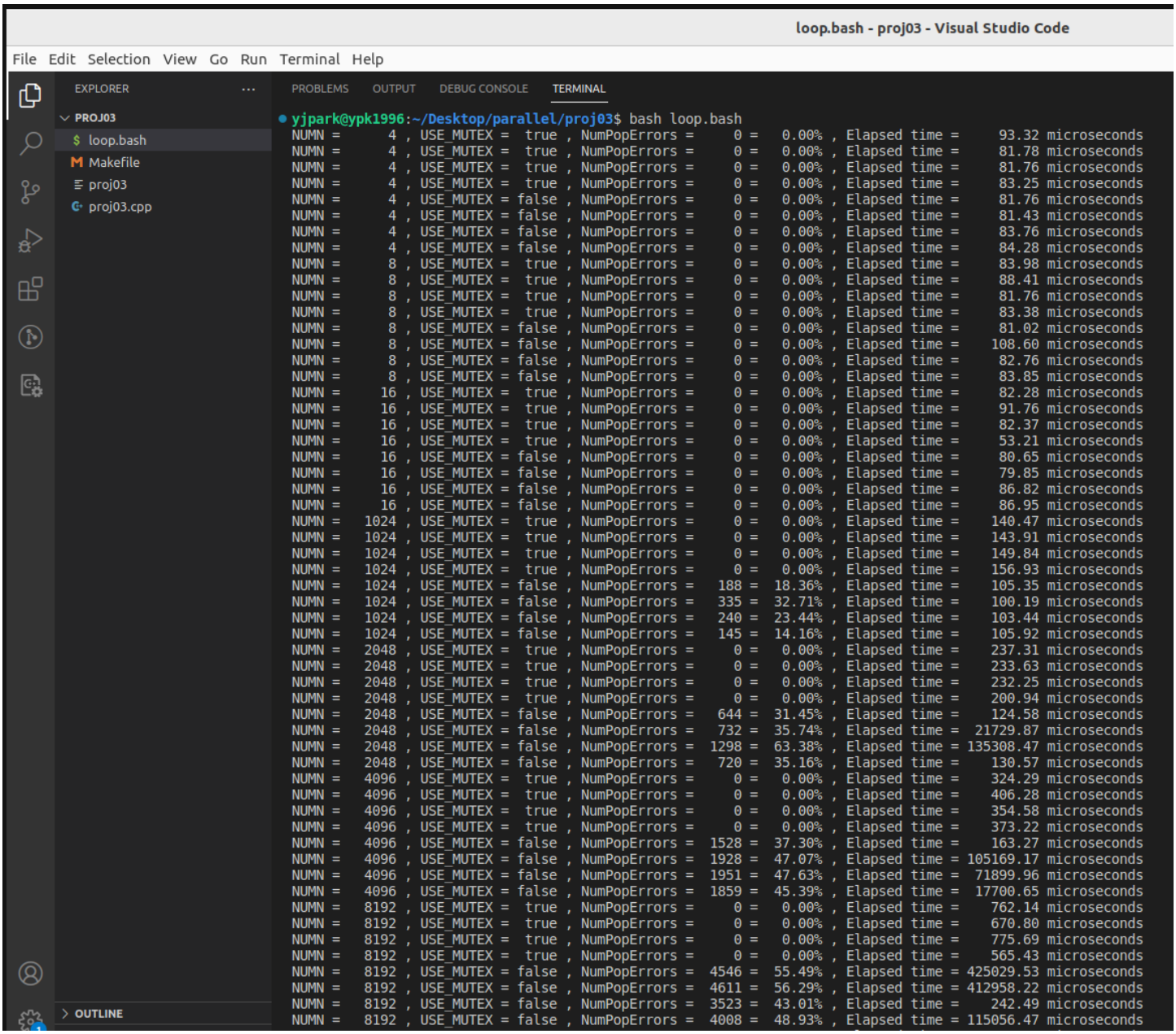
```
int
Pop( )
{
    // if the stack is empty, give the Push( ) function a chance to put something on the stack:
    int t = 0;
    while( StackPtr < 0  &&  t < TIMEOUT )
        t++;
    // if there is nothing to pop, return;
    if(USE_MUTEX) {
        omp_set_lock(&Lock);
    }
    if( StackPtr < 0 )
        return FAILED;

    int n = Stack[StackPtr];
    StackPtr--;
    if(USE_MUTEX) {
        omp_unset_lock(&Lock);
    }

    WasPopped[n] = true;
    return n;
}
```

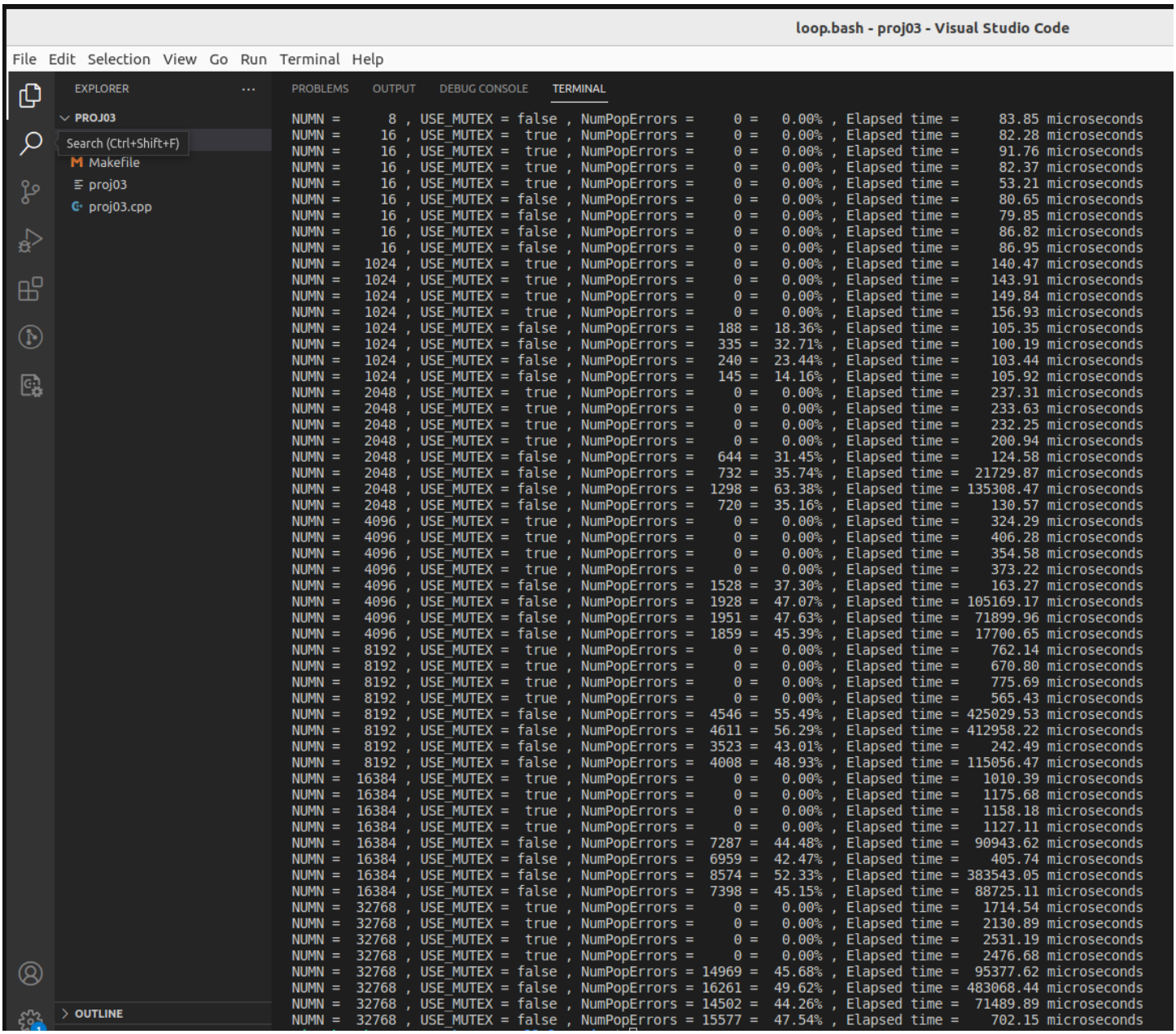
Pop’s critical section wrapped with mutex lock

```
int
main( int argc, char *argv[ ] )
{
#ifndef _OPENMP
    fprintf( stderr, "OpenMP is not supported here.\n" );
    return 1;
#endif

    // this array is here to be sure all the pops actually happened:
    for( int i = 0; i < NUMN; i++ )
    {
        WasPopped[i] = false;
    }

    omp_set_num_threads( 2 );

    // Changed here!!!
    if(USE_MUTEX) {
        omp_init_lock(&Lock);
    }

    double time0 = omp_get_wtime( );
    #pragma omp parallel sections
    {
        #pragma omp section
            PushAll( );

        #pragma omp section
            PopAll( );
    }
    double time1 = omp_get_wtime( );


    NumPopErrors = 0;
    for( int i =0; i < NUMN; i++ )
    {
        if( ! WasPopped[i] )
        {
            if( DEBUG ) fprintf( stderr, "%6d wasn't popped\n", i );
            NumPopErrors++;
        }
    }

    char *useMutexString = (char *)"false";
    if( USE_MUTEX )
        useMutexString = (char *)" true";

    fprintf( stderr, "NUMN = %6d , USE_MUTEX = %s , NumPopErrors = %5d = %6.2f%% , Elapsed time = %9.2lf microseconds\n",
        NUMN, useMutexString, NumPopErrors, 100.*(float)NumPopErrors/(float)NUMN, 1000000.*(time1-time0) );

    return 0;
}
```

Lock initialized in main

5. Tell us what you discovered by doing this:


loop.bash - proj03 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
PROJ03
loop.bash
Makefile
proj03
proj03.cpp
OUTLINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
yjpark@ypk1996:~/Desktop/parallel/proj03$ bash loop.bash
NUMN = 4 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 93.32 microseconds
NUMN = 4 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 81.78 microseconds
NUMN = 4 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 81.76 microseconds
NUMN = 4 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 83.25 microseconds
NUMN = 4 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 81.76 microseconds
NUMN = 4 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 81.43 microseconds
NUMN = 4 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 83.76 microseconds
NUMN = 4 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 84.28 microseconds
NUMN = 8 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 83.98 microseconds
NUMN = 8 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 88.41 microseconds
NUMN = 8 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 81.76 microseconds
NUMN = 8 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 83.38 microseconds
NUMN = 8 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 81.02 microseconds
NUMN = 8 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 108.60 microseconds
NUMN = 8 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 82.76 microseconds
NUMN = 8 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 83.85 microseconds
NUMN = 16 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 82.28 microseconds
NUMN = 16 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 91.76 microseconds
NUMN = 16 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 82.37 microseconds
NUMN = 16 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 53.21 microseconds
NUMN = 16 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 80.65 microseconds
NUMN = 16 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 79.85 microseconds
NUMN = 16 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 86.82 microseconds
NUMN = 16 , USE_MUTEX = false , NumPopErrors = 0 = 0.00% , Elapsed time = 86.95 microseconds
NUMN = 1024 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 140.47 microseconds
NUMN = 1024 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 143.91 microseconds
NUMN = 1024 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 149.84 microseconds
NUMN = 1024 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 156.93 microseconds
NUMN = 1024 , USE_MUTEX = false , NumPopErrors = 188 = 18.36% , Elapsed time = 105.35 microseconds
NUMN = 1024 , USE_MUTEX = false , NumPopErrors = 335 = 32.71% , Elapsed time = 100.19 microseconds
NUMN = 1024 , USE_MUTEX = false , NumPopErrors = 240 = 23.44% , Elapsed time = 103.44 microseconds
NUMN = 1024 , USE_MUTEX = false , NumPopErrors = 145 = 14.16% , Elapsed time = 105.92 microseconds
NUMN = 2048 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 237.31 microseconds
NUMN = 2048 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 233.63 microseconds
NUMN = 2048 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 232.25 microseconds
NUMN = 2048 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 200.94 microseconds
NUMN = 2048 , USE_MUTEX = false , NumPopErrors = 644 = 31.45% , Elapsed time = 124.58 microseconds
NUMN = 2048 , USE_MUTEX = false , NumPopErrors = 732 = 35.74% , Elapsed time = 21729.87 microseconds
NUMN = 2048 , USE_MUTEX = false , NumPopErrors = 1298 = 63.38% , Elapsed time = 135308.47 microseconds
NUMN = 2048 , USE_MUTEX = false , NumPopErrors = 720 = 35.16% , Elapsed time = 130.57 microseconds
NUMN = 4096 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 324.29 microseconds
NUMN = 4096 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 406.28 microseconds
NUMN = 4096 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 354.58 microseconds
NUMN = 4096 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 373.22 microseconds
NUMN = 4096 , USE_MUTEX = false , NumPopErrors = 1528 = 37.30% , Elapsed time = 163.27 microseconds
NUMN = 4096 , USE_MUTEX = false , NumPopErrors = 1928 = 47.07% , Elapsed time = 105169.17 microseconds
NUMN = 4096 , USE_MUTEX = false , NumPopErrors = 1951 = 47.63% , Elapsed time = 71899.96 microseconds
NUMN = 4096 , USE_MUTEX = false , NumPopErrors = 1859 = 45.39% , Elapsed time = 17700.65 microseconds
NUMN = 8192 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 762.14 microseconds
NUMN = 8192 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 670.80 microseconds
NUMN = 8192 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 775.69 microseconds
NUMN = 8192 , USE_MUTEX = true , NumPopErrors = 0 = 0.00% , Elapsed time = 565.43 microseconds
NUMN = 8192 , USE_MUTEX = false , NumPopErrors = 4546 = 55.49% , Elapsed time = 425029.53 microseconds
NUMN = 8192 , USE_MUTEX = false , NumPopErrors = 4611 = 56.29% , Elapsed time = 412958.22 microseconds
NUMN = 8192 , USE_MUTEX = false , NumPopErrors = 3523 = 43.01% , Elapsed time = 242.49 microseconds
NUMN = 8192 , USE_MUTEX = false , NumPopErrors = 4008 = 48.93% , Elapsed time = 115056.47 microseconds



1. Does the non-mutex way of doing this *ever* work? If so, how often?

   The non-mutex way works if there are very few iterations to perform (i.e. 4, 8, 16). Once it gets beyond even a slightly considerable number in the 3 digits, the non-mutex version does not work. Once race conditions are involved due to both threads having (relatively) a lot of work to perform, mutexes become necessary for correct results.

2. Does changing NUMN make any difference in the failure percentage?

   At lower NUMNs, the failure percentage is relatively low (at 10 to 25 percent), but as the trials increase, the failure percentage levels off at between 45 to 55 percent. One can note that once the treads are working in their full capacity (relatively higher NUMN and higher NUMNs than that threshold), the error rate stays at around 50%. Before that, the error rate may be slightly lower (under that NUMN threshold).

3. Is there a difference in elapsed execution time between mutex and non-mutex? Why do you suppose this is? (Ignore the very large elapsed times -- these are a reult of the TIMEOUT being used up.)

   Theoretically, mutexes should cause a slowdown in the program as it is an additional overhead, and it also blocks threads from accessing a shared resource when one thread is working with it. The timeouts make the data above hard to make out in the larger NUMNs, but if you look at NUMN = 1024 trials, you can clearly see that the use of mutexes have considerably slowed the program down (150 microseconds with mutex lock vs 100 microseconds without).

6. No tables or graphs are needed for this project.