Multithreading-Barrier

20307130350 信息安全 陈丹纯 2022/10/18

实验内容

概述

1. read `notxv6/barrier.c`.
2. implement barrier() to achieve the desired barrier which is that each thread blocks in `barrier()` until all `nthreads` of them have called `barrier()`.
3. useful primitives

```
pthread_cond_wait(&cond, &mutex);
pthread_cond_broadcast(&cond);
```

4. skip the assert triggers and pass make grade's barrier test.

原文

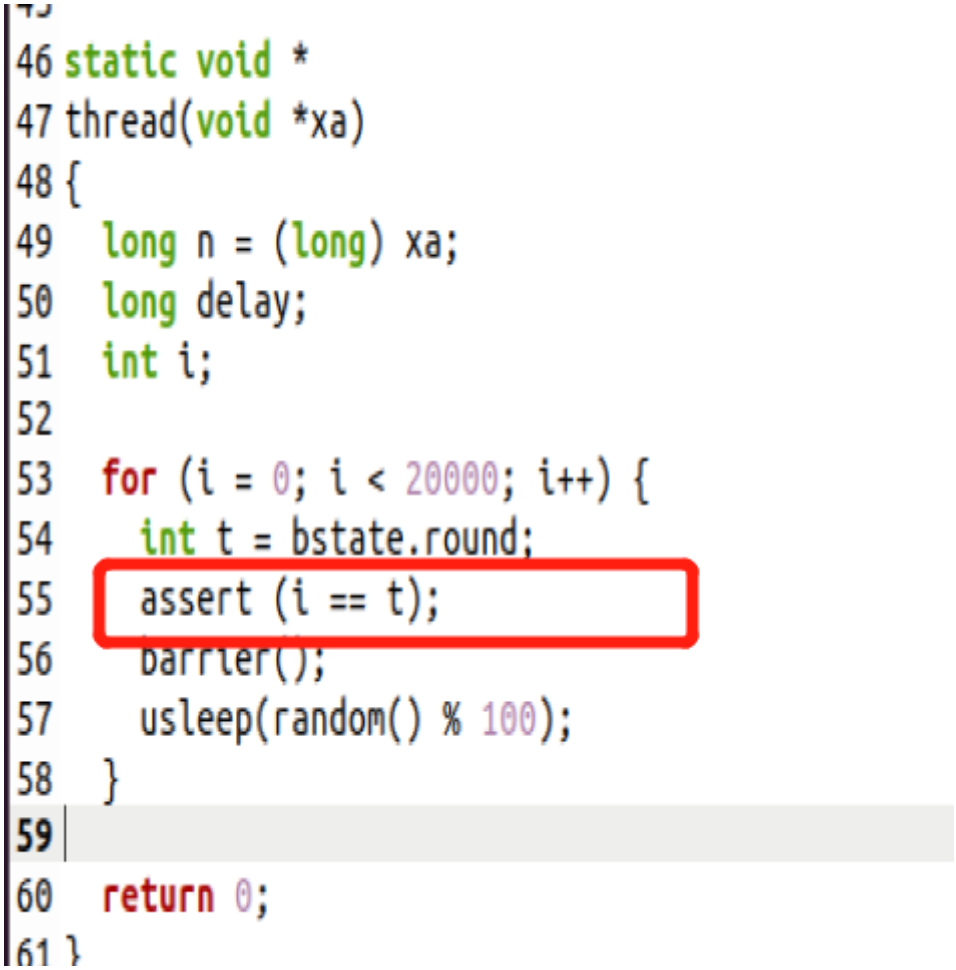
more details in Lab: Multithreading (mit.edu).

实验分析

实验代码上传git。

代码报错原因

Each thread executes a loop. In each loop iteration a thread calls `barrier()` and then sleeps for a random number of microseconds. **The assert triggers, because one thread leaves the barrier before the other thread has reached the barrier.**



```
46 static void *
47 thread(void *xa)
48 {
49   long n = (long) xa;
50   long delay;
51   int i;
52
53   for (i = 0; i < 20000; i++) {
54     int t = bstate.round;
55     assert (i == t);
56     barrier();
57     usleep(random() % 100);
58   }
59
60   return 0;
61 }
```

Solution

therefore, to avoid the `assert(i==t)` triggering, the `barrier()` is supposed to make each thread with the same `bstate.round` when running `thread()'s block` until all threads get the same `bstate.round`, wake up all threads to go on the next loop.

Algorithm

To realize this goal, in the `barrier()`, increase the `nthread` to acount the number of threads in the current round. When `bstate.nthread` is less than the number of all created threads, call `pthread_mutex_wait()`; When equal, increase the `bstate.round` by one and `bstate.nthread` to 0, call `pthread_mutex_broadcast()`;

Note: Don't forget the lock and unlock.

Implement(code)

```
25 static void
26 barrier()
27 {
28   // YOUR CODE HERE
29   //
30   pthread_mutex_lock(&bstate.barrier_mutex);
31   bstate.nthread++;
32   if(bstate.nthread==nthread){
33         bstate.nthread=0;
34         bstate.round++;
35         pthread_cond_broadcast(&bstate.barrier_cond);
36   }else{
37         pthread_cond_wait(&bstate.barrier_cond,&bstate.barrier_mutex);
38   }
39   pthread_mutex_unlock(&bstate.barrier_mutex);
40   // Block until all threads have called barrier() and
41   // then increment bstate.round.
42   //
43
44 }
```

实验结果

```
== Test barrier == make[1]: Entering directory '/home/utegan/xv6-labs-2021'
gcc -o barrier -g -O2 -DSOL_THREAD -DLAB_THREAD notxv6/barrier.c -pthread
make[1]: Leaving directory '/home/utegan/xv6-labs-2021'
barrier: OK (11.8s)
== Test time ==
time: FAIL
    Cannot read time.txt
Score: 24/60
make: *** [Makefile:336: grade] Error 1
utegan@ubuntu:~/xv6-labs-2021$ ./barrier 1
OK; passed
utegan@ubuntu:~/xv6-labs-2021$ ./barrier 2
OK; passed
utegan@ubuntu:~/xv6-labs-2021$ ./barrier 4
OK; passed
utegan@ubuntu:~/xv6-labs-2021$
```