操作系统(D)

第 6 次作业

李子龙 518070910095

2021 年 3 月 22 日

6.8 Race conditions are possible in many computer systems. Consider anonline auction system where the current highest bid for each itemmust be maintained. A person who wishes to bid on an item calls thebid(amount) function, which compares the amount being bid to thecurrent highest bid. If the amount exceeds the current highest bid, thehighest bid is set to the new amount. This is illustrated below:

```
void bid(double amount){
    if (amount > highestBid)
        highestBid = amount;
}
```

Describe how a race condition is possible in this situation and whatmight be done to prevent the race condition from occurring.

6.13 The first known correct software solution to the critical-section problemfor two processes was developed by Dekker. The two processes,P0andP1, share the following variables:

```
boolean flag[2]; /* initially false */
int turn;
```

The structure of process P_i(`i` == 0 or 1) is shown in Figure 6.18. Theother process is P_j(`j`== 1 or 0). Prove that the algorithm satisfies all three requirements for the critical-section problem.

6.21 A multithreaded web server wishes to keep track of the number ofrequests it services (known as *hits*). Consider the two following strategies to prevent a race condition on the variablehits. The first strategy is to use a basic mutex lock when updating `hits`:

```
int hits;
mutex_lock hitlock;
hit_lock.acquire();
hits++;
hit_lock.release();
```

A second strategy is to use an atomic integer:

```
atomic_t hits;
atomic_inc(&hits);
```

Explain which of these two strategies is more efficient.

7.8 The Linux kernel has a policy that a process cannot hold a spinlock while attempting to acquire a semaphore. Explain why this policy is in place.

7.11 Discuss the tradeoff between fairness and throughput of operationsin the readers–writers problem. Propose a method for solving thereaders–writers problem without causing starvation.

7.16 The C programstack-ptr.c(available in the source-code download)contains an implementation of a stack using a linked list. An example ofits use is as follows:

```
StackNode *top = NULL;
push(5, &top);
push(10, &top);
push(15, &top);

int value = pop(&top);
value = pop(&top);
value = pop(&top);
```

This program currently has a race condition and is not appropriate fora concurrent environment. Using Pthreads mutex locks (described inSection 7.3.1), fix the race condition.