SOFT400227: Operating System 2022-Fall

Homework-**5** Solutions

软件 2101 杨豪 学号: 2206213297

2022 年 10 月 15 日

Honor Code: I promise that I finished the homework solutions on my own without copying other people's work.

Synchronous and Mutual Exclusion

Problem 1. 6.4

Because the spinlock requires busy waiting, so there have to be another process executing that waits for the signal about interrupt condition.

In single-processor systems, if the method's resource is the processor and doesn't release it, other processes won't be able to obtain interrupt signal.

In multiprocessor systems, processes can execute on other processors and obtain the signal so that the resources can be released.

Problem 2. 6.9

If two same operations are executed on a semaphore at the same time and they are not performed atomically, both operations might simultaneously operate on the semaphore value, which violates mutual exclusion.

Problem 3. 6.11

```
Semaphore come_in = 1, customer = 0, baber = 1;
int available = N;

New_Customer{
    P(come_in);
    if(available != 0){
        V(customer);
        available --;
    }
    V(come_in);
}
```

```

Baber{
    P(customer);
    wake(baber);
    P(baber);
    while(++ available < N){
        P(customer);
    }
    V(baber);
    sleep(baber);
}
```

Problem 4.

1. Yes, easy to lead to starvation;

2. The mechanism of given abnormal PV operations is a stack but we need a queue. According to idea in Peterson's Algorithm, we can set N 1-length stack to simulate queue as below:

```
    Semaphore S[N - 1];
    for(int i = 0; i < N - 1; i ++) S[i] = N - i- 1;
        for(int i = 0; i < N - 1; i ++) P(S[i]);
            // Critical Section
        for(int i = n - 2; i >= 0; i --) V(S[i]);
```

The algorithm above take O(n) space. Sadly, although this algorithm seems so easy and waste of characteristics, I still have no idea about how to optimize it.

However, it is said that it has proved that there is a method using only O(log(N)) space, amazing!

Problem 5.

We need 3 semaphore:

- When there is a writer waiting, no more readers are allowed so `Writer()` should control this sequence by a semaphore which is a crucial part of `Reader()` —— `preRead` . And this kind of control should be kept in a mutual exclusion environment, so we need another semaphore —— `mutex` .

- When there is a reader reading, writer should waiting until it finished so as above, `Reader ()` need to guarantee this rule with a semaphore too. —— `Writer` . and in the `mutex` also.
- There is always at most one writer writing at some time —— `Writer` is adequate to deal with it.

```
Semaphore mutex = 1, write = 1, preRead = 1;
int writeCount = 0, readCount = 0;

Reader(){
    P(preRead);
    if(readCount==0) P(write);
    readCount++;
    V(preRead);

    // Reader Critical Section.

    P(mutex);
    readCount--;
    if(readCount==0) V(write);
    V(mutex);
}

Writer(){
    P(mutex);
    if(writeCount == 0) P(preRead);
    writeCount ++;
    V(mutex);
    P(write);

    // Writer Critical Section.

```

```
        P(mutex);
        writeCount --;
        if(writeCount == 0) V(preRead);
        V(write);
        V(mutex);
    }
```

Problem 6.

We need 3 semaphore:

- The door is only allowed for one person to get in and get out, so we need `door` to deal with this mutual exclusion.
- There are N students inside the classroom, exam should begin, so we need `exam` to deal with this synchronous.
- When the exam begin and the teacher handout papers, the door should be closed. After the teacher handout papers, anybody who have handed can leave the classroom, so we need to open the door.We can use `door` to control this flow.
- When all papers handed in the teacher can leave the classroom and the exam ended. Students can hand in papers at the same time, so we just need one new semaphore, `handin_done` , to control this synchronous.

```
    Semaphore door = 1, exam = 0, handin_done = 0;
    int inCount = 0, paperCount = 0;
    In(){
        P(door);
        inCount ++;
        if(inCount == N) V(exam);
        V(door);
    }
    Out(){
        P(door);
        inCount --;
        V(door);
    }
    Exam(){
        P(exam);
        P(door);

        // Exam Critical Section: hand out papers.

        V(door);
        P(handin_done);
        // Teacher packaging papers;
        V(exam);
    }
    Handin(){
        paperCount ++;
        if(paperCount == N) V(handin_done);
    }
```

Other things

LaTeX code refer to these things and was complied on texlive2020.

- UCB-CS70's given homework template.
- A free website useful to edit LaTeX formula code.
- A free website helpful to generate LaTeX table.

Some description refer to *Operating System Concepts 10th*, Wikipedia and Professor.Tian's PPT.

The purpose of writing in English is to adapt to bilingual teaching and to improve my poor English writing skills in preparation for a possible future exchange program.

Thanks for your correcting and grading :).