ELC-467 Operating Systems
Spring 2020

Problem Set (2): Process Interaction and Communications

Study Questions

[Q1] To insert an item into a linked list requires two pointers to be changed. Suppose a thread is context-switched out after changing only one of them. If the next thread to run tries to traverse the same list, what will happen?

[Q2] A program that runs in a multi-programmed system has a critical section in which two shared variables are accessed. Two semaphores s1 and s2 are used to guarantee exclusive access to the two variables. However, the programmer made an error in the following program segment:

```
down(s1); down(s2); Critical_section; up(s1); up(s1);
```

Explain the problem(s) that this error can cause?

[Q3] Consider the producer/consumer problem with a bounded buffer. The solution of this problem using two counting semaphores and an additional semaphore for mutual exclusion was explained in the lecture. Assume that two or more processes can write in the buffer, and each of these writing processes was written exactly as the writer code used before, and that the reader code was not modified. Will the system operate correctly?

[Q4] How can threads be used to speed up the addition of a large number of scalar variables in a multiprocessor system? Assume n variables and p processors with n being an integer multiple of p. How will semaphores be useful in this application?

[Q5] In some operating systems, when a process enters a critical section using a binary semaphore, its priority is raised to that of the highest priority process blocked by down() operation on the same semaphore. What is the advantage of this method?

Problems:

[P1] Consider the following three processes that run concurrently at unknown speeds. Assume that initially m=3 and n=0.

Process 1

```
….
while (1)
  { down (m);
    printf ("A ");
    up (n);}
```

Process 2

```
….
while (1)
  { down (n);
    printf ("B ");
    printf ("C ");
    up (n);}
```

Process 3

```
….
while (1)
  { down (n);
    printf ("D ");}
```

If the three processes run until they are all blocked:

a) How many A's are displayed?
b) How many D's are displayed?
c) What is the minimum number of B's that might be displayed?
d) Is A B C B A D C A B C D D a possible output sequence? Explain.

[P2] Discuss the following Client /Server model using semaphores.

<u>Server process</u>

```
while TRUE do
begin
      down(s);
      perform_service;
end
```

<u>Client process</u>

```
up(s);
request_service;
```

[P3] Two processes are required to run on a multitasking system such that a function f1 in the first process must run simultaneously with a function f2 in the second process. Show how to achieve this synchronization using semaphores.

[P4] A thread is required to be blocked until *n* other threads end. These other threads can run in any order. Show how this can be done using the semaphores method.

[P5] Two threads are required to access a shared array `A[ ]`, as shown below:

<u>Thread 1</u>

```
…..
for (i=0; i<100; i++)
A[i]= B[i];
…..
```

<u>Thread 2</u>

```
…..
for (i=0; i<100; i++)
C[i]= A[i];
…..
```

Show how to use semaphores such that thread 2 will not read each element of the array `A` before thread 1 writes the value of this element.

[P6] A multitasking system uses round-robin scheduling with q=1. Three processes start at the same time with process times of 6, 7, and 8. Calculate the average process waiting time if the three processes access a common data item and use semaphores to avoid the race condition. Assume that critical regions are as shown below:

```
                      ---- critical----
1st process  |---|---|---|---|---|---|

                           -critical
2nd process  |---|---|---|---|---|---|---|

                                 ---critical---
3rd process  |---|---|---|---|---|---|---|---|
```