CS-3103 : Operating Systems : Sec-A (NB)

Process Synchronization – Questions on Semaphores & bounded buffer ➔ 2

Semaphores Usage

- Operating systems often distinguish between counting and binary semaphores.
- The value of a ***counting semaphore*** can range over an unrestricted domain.
- The value of a ***binary semaphore*** can range only between 0 and 1.
- Counting semaphores can be used to control access to a given resource consisting of a finite number of instances.
- ***The semaphore is initialized to the number of resources available.***

Semaphores Usage

- Each process that wishes to use a resource performs a ***wait()*** operation on the semaphore (thereby decrementing the count).
- When a process releases a resource, it performs a ***signal()*** operation (incrementing the count).
- When the count for the semaphore goes to 0, all resources are being used.
- After that, processes that wish to use a resource will be blocked until the count becomes greater than 0.

Semaphore

***wait* → P operation**
When a process performs a wait operation on a semaphore, the operation checks whether the value of the semaphore is > 0. If so, it decrements the value of the semaphore and lets the process continue its execution; otherwise, it blocks the process on the semaphore

```
procedure wait (S)
begin
    if S > 0
        then S := S–1;
        else block the process on S;
end;

procedure signal (S)
begin
    if some processes are blocked on S
        then activate one blocked process;
        else S := S+1;
end;
```

signal → V operation; signal on semaphore activates a process blocked on the semaphore, if any, or increments the value …

… of the semaphore by 1. Indivisibility of the *wait* and signal operations is ensured by programming language or the OS that implements it.

Figure Semantics of the *wait* and *signal* operations on a semaphore.

Solving synchronization problems using Semaphores

- A semaphore :
  a) is a binary mutex
  b) must be accessed from only one process
  c) can be accessed from multiple processes
  d) None of these

- Answer: c

Solving synchronization problems using Semaphores

- The two kinds of semaphores are : (choose two)
  a) mutex
  b) binary
  c) counting
  d) decimal

- Answer: b and c

Solving synchronization problems using Semaphores

- At a particular time of computation the value of a counting semaphore is 7.Then 20 P operations and 15 V operations were completed on this semaphore.The resulting value of the semaphore is :

  a) 42
  b) 2
  c) 7
  d) 12

P represents Wait (decrement --) and
V represents Signal (increment ++)

- Answer: b

  Explanation: P represents Wait and V represents Signal. P operation will decrease the value by 1 everytime and V operation will increase the value by 1 everytime.

Solving synchronization problems using Semaphores

- A binary semaphore is a semaphore with integer values : (choose two)
  a) 1
  b) -1
  c) 0
  d) 0.5

- Answer: a and c
  Explanation: None

Solving synchronization problems using Semaphores

- Semaphores are mostly used to implement :
  a) System calls
  b) IPC mechanisms
  c) System protection
  d) None of these
  View Answer
- Answer: b

❑The bounded buffer problem is also known as :

a) Readers – Writers problem

b) Dining – Philosophers problem

c) Producer – Consumer problem

d) None of these

Answer: c

❑In the bounded buffer problem, there are the empty and full semaphores that :

a) count the number of empty and full buffers

b) count the number of empty and full memory spaces

c) count the number of empty and full queues

d) None of these

Answer: a

In the bounded buffer in producer-consumer problem:

a) there is only one buffer

b) there are n buffers (n being greater than one but finite)

c) there are infinite buffers

d) the buffer size is bounded **Answer: a, d**

To ensure difficulties do not arise in the readers – writers problem, _____ are given exclusive access to the shared object.

a) readers

b) writers **Answer: b**

c) None of these

The dining – philosophers problem will occur in case of :

a) 5 philosophers and 5 chopsticks

b) 4 philosophers and 5 chopsticks

c) 3 philosophers and 5 chopsticks

d) 6 philosophers and 5 chopsticks

Answer: a

At a particular time of computation, the value of counting semaphore is 7. Then 20 wait operations and 'x' signal operations were completed on this semaphore. If the final value of the semaphore is 5, what is x?

a) 18 b) 13 c) 5 d) 0

Answer: a) s=7; (s-20)+x=5, x = 18

Solving synchronization problems using Semaphores

A process using a semaphore has a start value of 1 for its semaphore. Since the start of execution of the program, 12 signal operations were completed. How many wait operations have been completed so far if the current value of semaphore is 6?

a) 1 b) 5 c) 7 d) 11 **Ans: c)**

Consider the below pseudo code:

```
semaphore   S = 1;
semaphore   E = 1;
If (thread_count++ < 100)  spawnnewthread();
wait(E);
// critical section – begin
----------------------------
----------------------------
// critical section – end
signal(S);
```

Assume that above pseudocode gets called a hundred times, what is the count of semaphore E?

a) 0 b) 1 c) -99 d) -100

Ans: a): The code will run for 100 times, but E will get initialized to 1, then after wait(E) operation, **E will get the value 0**