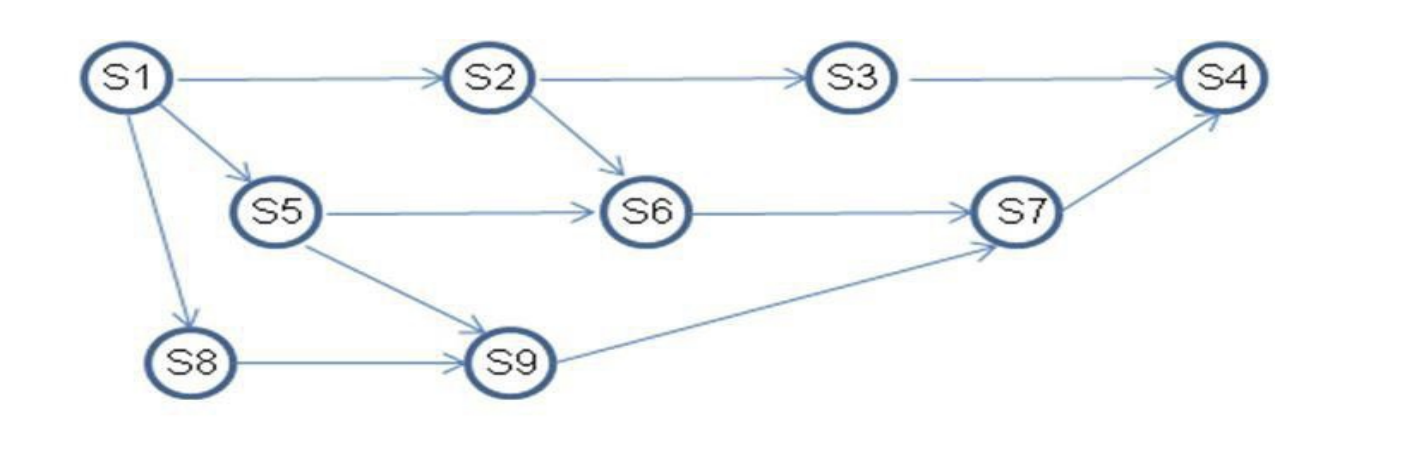
**Operating System - Assignment - 2**

***Q.1 Assume you are given the following wait-graph that represents the relationship between multiple threads (s1,s2,s3,…). An arrow from one thread (Sy) to another (Sx) means that thread Sx must finish its computation before Sy starts. (For example: S1 has to wait for S2,S5,S8 to finish, S2 has to wait for s3,s6 to finish and so on.) Use semaphores to enforce this relationship specified by the graph. Be sure to show the initial values and the locations of the semaphore operations. You will be marked based on finding the best solution with minimum number of semaphores.***



**Answer:** There are minimum 6 semaphores needed to implement scheduling constraints.

We need to look for the process which has multiple processes dependencies.

In this scenario you could use the same semaphore for the predecessors that point to an immediate successor. You just need to iterate the semaphore more than once in such cases.

Following tables show the above steps.

|  |  |
| --- | --- |
| Process | Immediate Predecessor |
| S1 | — |
| S2 | S1 |
| S3 | S2 |
| S4 | S3 AND S7 |
| S5 | S1 |
| S6 | S2 AND S5 |
| S7 | S6 AND S9 |
| S8 | S1 |
| S9 | S5 AND S8 |

Let’s take i, j, k, l, m and n, 6 semaphores initialized to 0.

--------------- : indicates the piece of code for that particular process.

|  |  |  |  |
| --- | --- | --- | --- |
| S4  {  ---------------  ---------------  V(i)  V(i)  } | S7  {  P(i)  ---------------  ---------------  V(j)  V(j)  } | S3  {  P(i)  ---------------  ---------------  V(k)  } | S6  {  P(j)  ---------------  ---------------  V(L)  V(k)  } |
| S2  {  P(k)  P(k)  ---------------  ---------------  V(n)  } | S5  {  P(L)  P(L)  ---------------  ---------------  V(n)  } | S8  {  P(m)  ---------------  ---------------  V(n)  } | S1  {  P(n)  P(n)  P(n)  ---------------  ---------------  } |
| S9  {  P(j)  ---------------  ---------------  V(L)  V(m)  } |  |  |  |

Above is the position of P() and V() operations in all 9 process to enforce the given dependency matrix.

S4 will execute first as it doesn’t have any direct dependency on any other process, and it will enable S3 and S7 to run, For S2, S3 and S6 should run. S3 ran after S4, S6 and S9 dependent on S7, So S7 will enable the S6 and S9 to run and S6(along with S3 which ran earlier) will further allow S2 to run. S6(along with S9) will also allow S5 to run. S9 will allow S8 to run. At the end, S1 will run after allowing by S2, S5 and S8.

***Q.2 What is the difference between Mesa and Hoare scheduling for monitors?***

Answer:

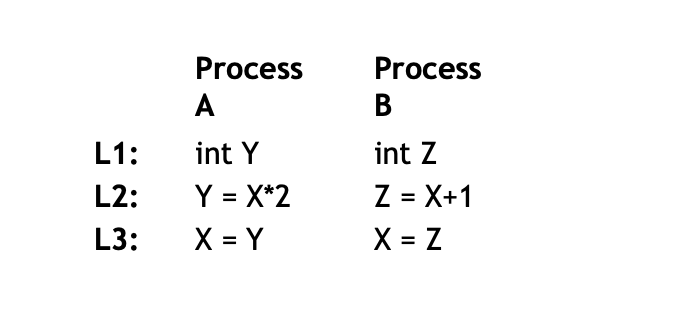
**Mesa monitors :**

When the waiter is woken, it must contend for the lock with other threads Hence, it must re-check the condition.

**Hoare monitors:**

Signaling thread gives up lock, hence Woken-up waiter acquires lock, Signaling thread re-acquires lock after waiter unlocks it.

***Question 3: (20 points) The following pair of processes share a common variable X:***

******

***is set to 5 before either process begins execution. As usual, statements within a process are executed sequentially, but since no assumptions can be made regarding each process’s speed of execution, statements in either process may execute in any order with respect to statements in the other process.***

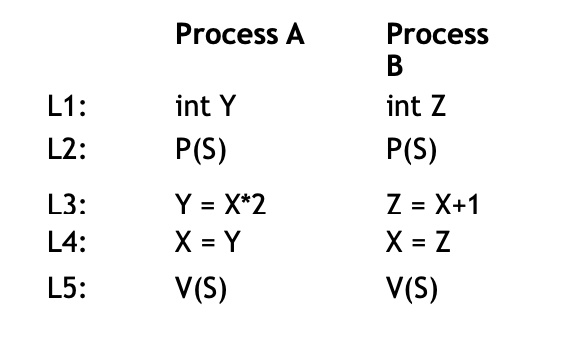
***(a) How many different values of X are possible after both processes finish executing?***

Answer:

There are four cases :

X=11, 12, 6 and 10.

***(b) Suppose the programs are modified as follows to use a shared binary semaphore S***:



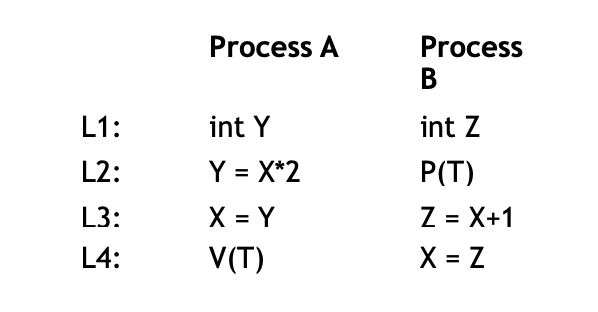
***S is set to 1 before either process begins execution or, as before, X is set to 5. Now, how many different values of X are possible after both processes finish executing?***

Answer:

There are two cases:

X= 11 and 12.

***(c) Finally, suppose the programs are modified as follows to use a shared binary semaphore T:***

******

***T is set to 0 before either process begins execution or, as before, X is set to 5. Now, how many different values of X are possible after both processes finish executing***?

Answer:

Only one case, X = 11;