Operating System Concepts

Sessional Assignment Spring 2020

Marks : 20

Student Name: Shayan khan **Student ID#: 6833**

Class : BS(SE) **Submitted To : Sir Daud Khan**

Semester: 8th **Subject: Operating System**

---

1. **Explain the necessary conditions that may lead to a deadlock situation. 2. What are the various methods for handling deadlocks?**

Answer:

A deadlock situation can arise if and only if the following four conditions hold simultaneously in a system-

Mutual Exclusion: At least one resource is held in a non-sharable mode that is only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.

Hold and Wait: There must exist a process that is holding at least one resource and is waiting to acquire additional resources that are currently being held by other processes.

No Preemption: Resouces cannot be preempted; that is, a resource can only be released voluntarily by the process holding it, after the process has completed its task.

Circular Wait: There must exist a set {p_0, p_1,.....p_n} of waiting processes such that p_0 is waiting for a resource which is held by p_1, p_1 is waiting for a resource which is held by p_2,..., p_{n-1} is waiting for a resource which is held by p_n and p_n is waiting for a resource which is held by p_0.

Methods for Handling Deadlocks

- **Deadlock Prevention.**
  - Disallow one of the four necessary conditions for deadlock.
- **Deadlock Avoidance.**
  - Do not grant a resource request if this allocation have the potential to lead to a deadlock.
- **Deadlock Detection.**

    - Always grant resource request when possible. Periodically check for deadlocks. If a deadlock exists, recover from it.

- **Ignore the problem...**
    - Makes sense if the likelihood is very low.

2. **Is it possible to have a deadlock involving only one single process? Explain your answer.**

Answer:

Deadlock with one process is not possible. Here is the explanation.

A deadlock situation can arise if the following four conditions hold simultaneously in a system.

- Mutual Exclusion.
- Hold and Wait.
- No Preemption.
- Circular-wait.

It is not possible to have circular wait with only one process, thus failing a necessary condition for Circular wait. There is no second process to form a circle with the first one. So it is not possible to have a deadlock involving only one process.

3. **Consider a system consisting of 4 resources of the same type that are shared by 3 processes, each of which needs at most 2 resources. Show that the system is deadlock free.**

   Answer:

   Suppose the system is deadlocked. This implies that each process is holding one resource and is waiting for one more. Since there are three processes and four resources, one process must be able to obtain two resources. This process requires no more resources and, therefore it will return its resources when done.

4. **What is a resource allocation graph? How do you obtain a wait-for graph from it? Explain their uses.**

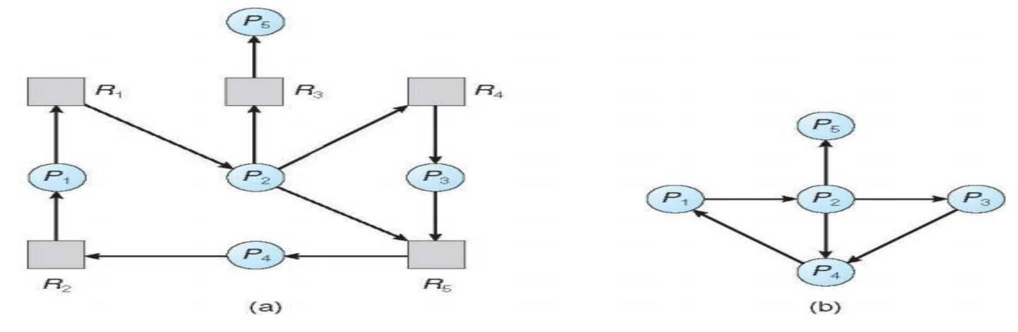
Answer:

A resource allocation graph tracks which resource is held by which process and which process is waiting for a resource of a particular type. It is very powerful and simple tool to illustrate how interacting processes can deadlock. If a process is *using* a resource, an arrow is drawn from the resource node to the process node. If a process is *requesting* a resource, an arrow is drawn from the process node to the resource node.

If there is a cycle in the Resource Allocation Graph and each resource in the cycle provides only one instance, then the processes will deadlock. For example, if process 1 holds resource A, process

2 holds resource B and process 1 is waiting for B and process 2 is waiting for A, then process 1 and process 2 will be deadlocked.

Resource-Allocation Graph and Wait-for Graph

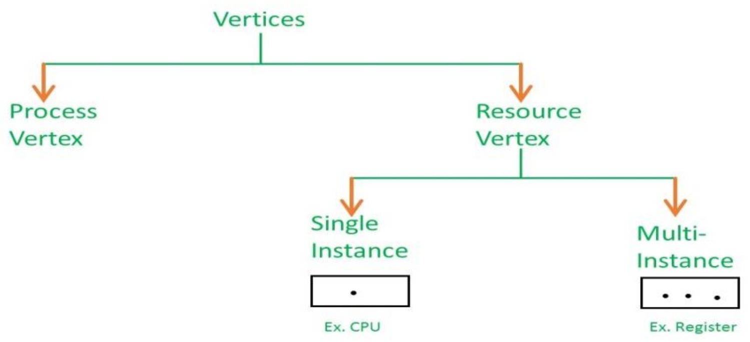


Resource-Allocation Graph Corresponding wait-for graph

<u>There Uses:</u>

So, resource allocation graph is explained to us what is the state of the system in terms of **processes and resources**. Like how many resources are available, how many are allocated and what is the request of each process. Everything can be represented in terms of the diagram. One of the advantages of having a diagram is, sometimes it is possible to see a deadlock directly by using RAG, but then you might not be able to know that by looking at the table. But the tables are better if the system contains lots of process and resource and Graph is better if the system contains less number of process and resource.
We know that any graph contains vertices and edges. So RAG also contains vertices and edges. In RAG vertices are two type –
1. Process vertex – Every process will be represented as a process vertex.Generally, the process will be represented with a circle.
2. Resource vertex – Every resource will be represented as a resource vertex. It is also two type –

- **Single instance type resource –** It represents as a box, inside the box, there will be one dot.So the number of dots indicate how many instances are present of each resource type.
- **Multi-resource instance type resource –** It also represents as a box, inside the box, there will be many dots present.



5. **Can a system detect that some of its processes are starving? If you answer "yes," explain how it can. If you answer "no," explain how the system can deal with the starvation problem.**

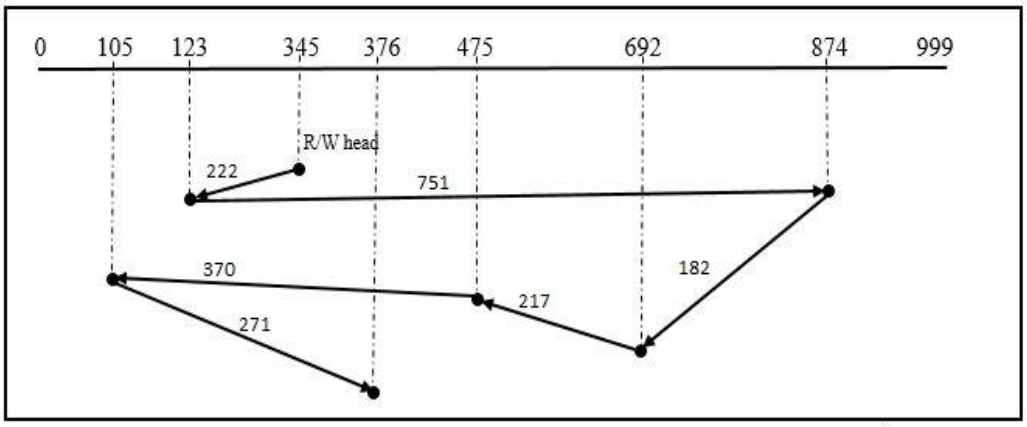
Answer:

Starvation is a difficult topic to define as it may mean different things for different systems. For the purposes of this question, we will define starvation as the situation whereby a process must wait beyond a reasonable period of time perhaps indefinitely before receiving a requested resource. One way of detecting starvation would be to first identify a period of time that is considered unreasonable. When a process requests a resource, a timer is started. If the elapsed time exceeds T, then the process is considered to be starved. One strategy for dealing with starvation would be to adopt a policy where resources are assigned only to the process that has been waiting the longest. For example, if process Pa has been waiting longer for resource X than process Pb , the request from process Pb would be deferred until process Pa ’s request has been satisfied. Another strategy would be less strict than what was just mentioned. In this scenario, a resource might be granted to a process that has waited less than another process, providing that the other process is not starving. However, if another process is considered to be starving, its request would be satisfied first.

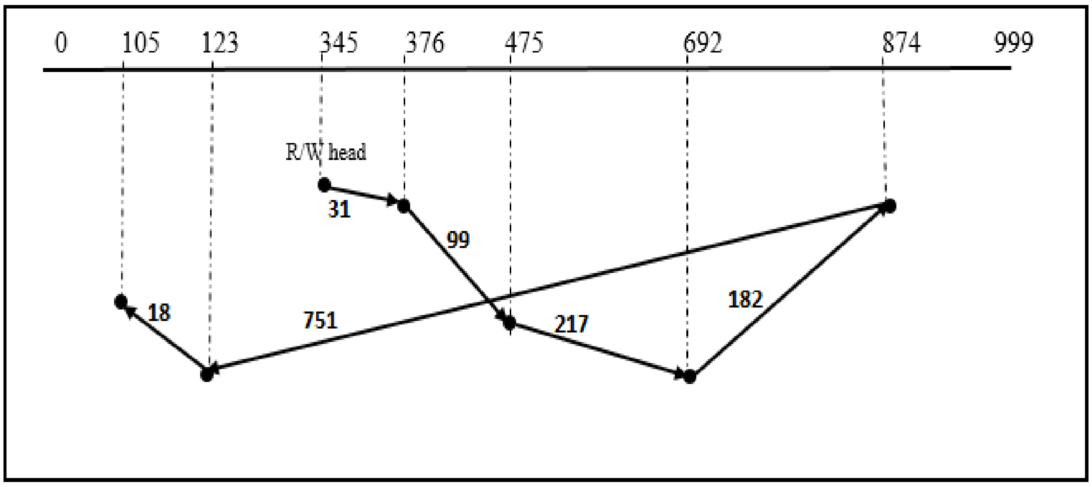
6. **On a disk with 1000 cylinders, number 0 to 999, compute the number of tracks the disk arm must move to satisfy all the requests in the disk queue. Assume the last request serviced was at track 345 and the head is moving toward track 0. The queue in FIFO order contains requests for the following tracks: 123, 847, 692, 475, 105, 376. Perform the computations for the following disk scheduling algorithms:**
    - **FCFS**
    - **SSTF**

Answer:

- **FCFS**



- **SSTF**



<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<END>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>