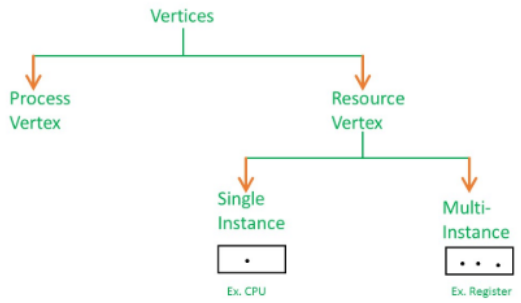
Resource Allocation Graph (RAG) in Operating System

As Banker's algorithm using some kind of table like allocation, request, available all that thing to understand what is the state of the system. Similarly, if you want to understand the state of the system instead of using those table, actually tables are very easy to represent and understand it, but then still you could even represent the same information in the graph. That graph is called **Resource Allocation Graph (RAG)**.

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 –

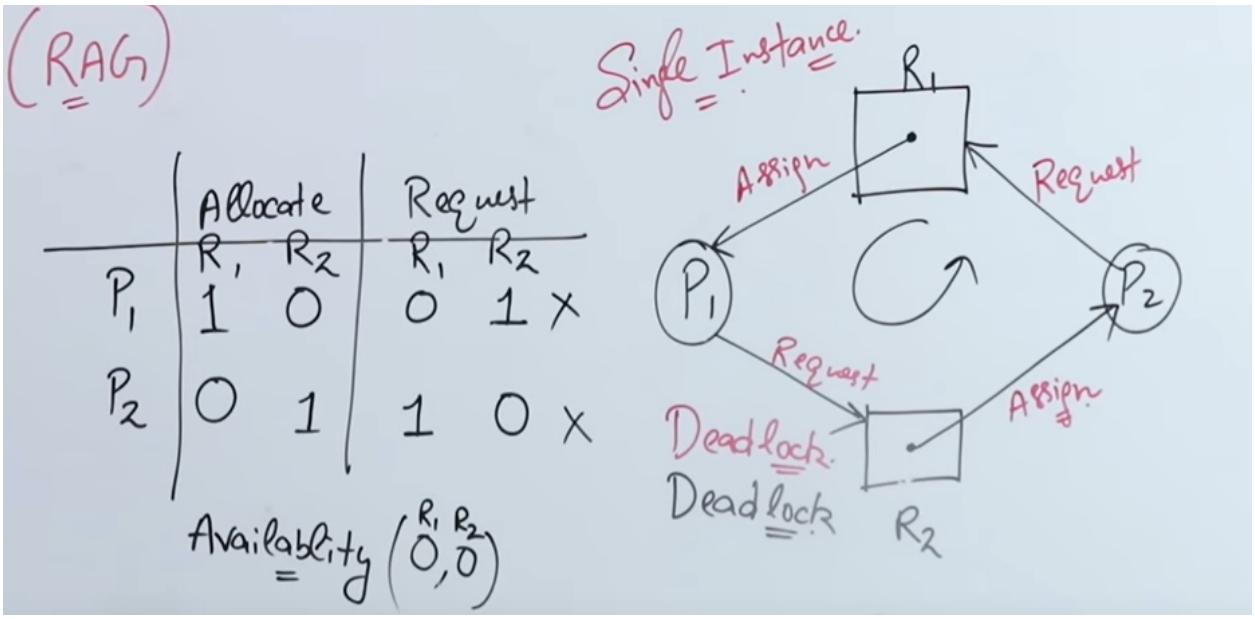
1. **Process vertex –** Every process will be represented as a process vertex.Generally, the process will be represented with a circle.
2. **Resource vertex –** Every resource will be represented as a resource vertex. It is also two type –

- **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.



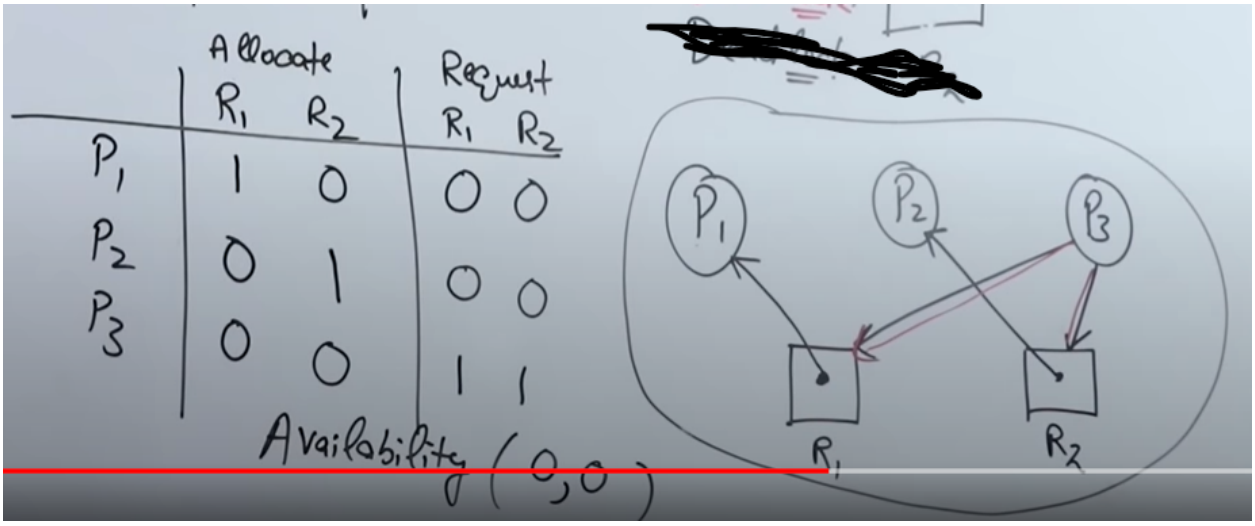
Single Instance: **Single instance system + Circular wait= Deadlock

Example 1:



Current Avail.= (0,0)

Example 2:



Current avail. =(0,0)

P1: IN EXECUTION, TERMINATE-→ R1 released

Current avail. =(1,0)

P2: IN EXECUTION, TERMINATE-→ R2 released

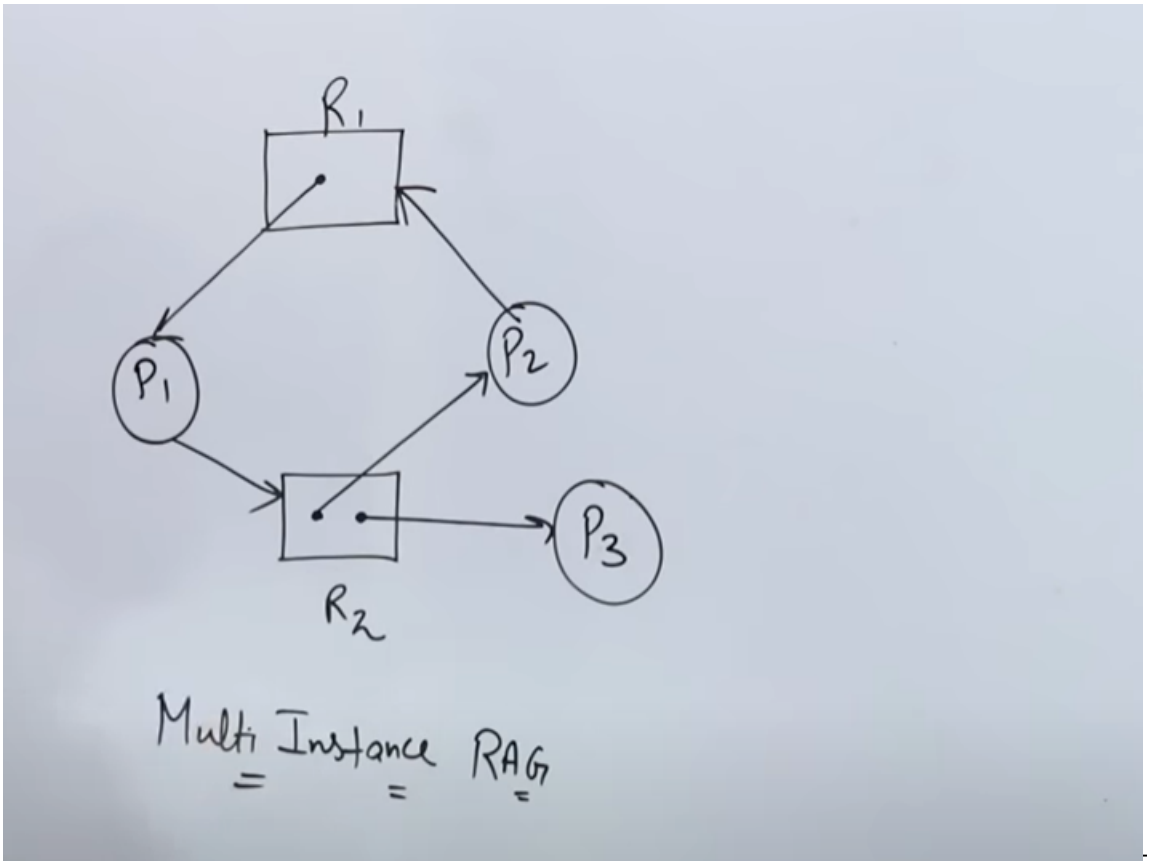
Current avail. =(1,1)

P3: IN EXECUTION, TERMINATE-→ R1 & R2 released

Current avail. =(1,1)

Multiple Instance:

Example 1:



	ALLOC		REQ	
	R1	R2	R1	R2
P1	1	0	0	1
P2	0	1	1	0
P3	0	1	0	0

Current avail. = (0,0)

P3: IN EXECUTION, TERMINATE-→ R2 released

Current avail. =(0,1)

P1: IN EXECUTION, TERMINATE-→ R1 released

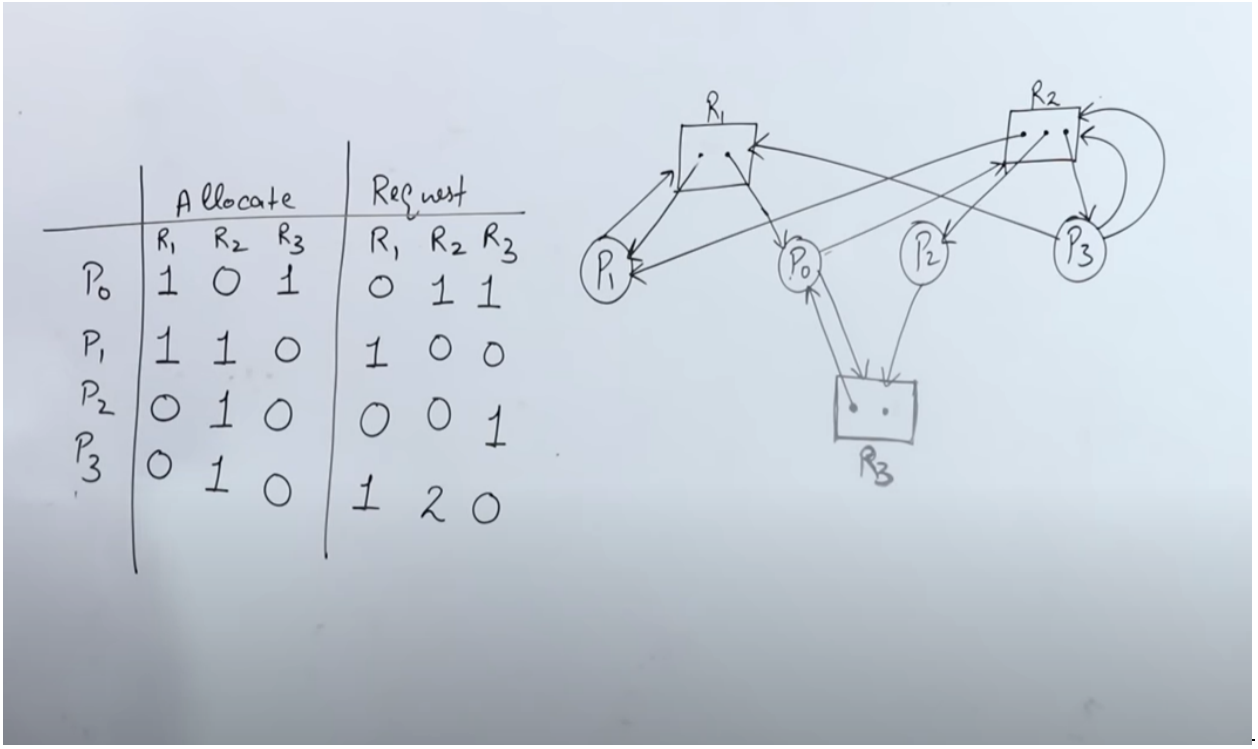
Current avail. =(1,1)

P2: IN EXECUTION, TERMINATE-→ R2 released

Current avail. =(1,2)

This system is not in DEADLOCK!

Example 2:



	Allocate			Request		
	R_1	R_2	R_3	R_1	R_2	R_3
P_0	1	0	1	0	1	1
P_1	1	1	0	1	0	0
P_2	0	1	0	0	0	1
P_3	0	1	0	1	2	0

Current avail. =(0,0,1)

P2: IN EXECUTION, TERMINATE-→ R2 released

Current avail. =(0,1,1)

P0: IN EXECUTION, TERMINATE-→ R1 & R3 released

Current avail. =(1,1,2)

P1: IN EXECUTION, TERMINATE-→ R1 & R2 released

Current avail. =(2,2,2)

P3: IN EXECUTION, TERMINATE-→ R2 released

Current avail. =(2,3,2)

The system is not in Deadlock!