Mutual Exclusion

UNIT-3:Synchronization in Distributed Systems

- Clock Synchronization
- Mutual Exclusion
- Election Algorithms
- Atomic Transactions
- Deadlocks in Distributed Systems

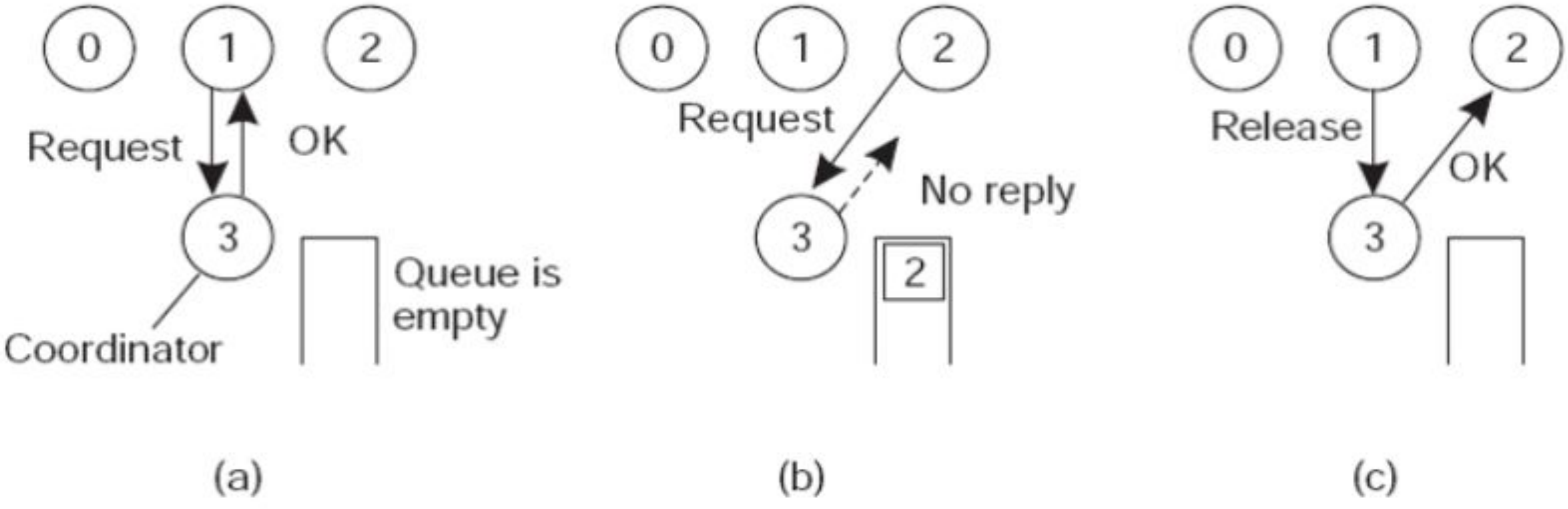
What is Mutual Exclusion ?

- When a process is accessing a shared resource, the process is said to be in a critical section (CS).
- It ensures that no two process can be in the same CS at the same time.
- In uniprocessor systems, CS's are protected using semaphores, monitors, and some similar constructs.
- Different algorithms based on message passing to implement mutual exclusion in distributed systems are

- Centralized algorithms
- Distributed algorithms
- Token Ring algorithms

Centralized Algorithms: (1 Boss)

- One process is elected as the coordinator (highest network address).
- Whenever a process wants to access a shared resources, it sends request to the coordinator to ask the permission.
- Queue is maintained to track the request.



Contd..

Case 1:

Process 1 asked permission to Process 3 to access shared resources (as no other processes is currently in the CS) so, permission granted.

Case 2:

Process 2 asked permission to Process 3, to access some shared resources, but it is already full, so no reply and Process 2 has been pushed into the queue.

Case 3:

Process 1 released the CS, hence shared resources is free.

Dequeue has been done(popping of Process 2), and send OK for accessing it.

Distributed Algorithm

- Whenever a process wants to enter a CS, it builds a message containing the (name of the CS it wants to enter, its process number, and the current time) and sends to all processes, conceptually itself.
- When a process receives a request message from another process; the action it takes depends on its state with respect to the CS named in the message. Three cases have to be distinguished.

Case 1: If the receiver is not in the CS and doesn't want to enter, it sends back OK message to the sender.

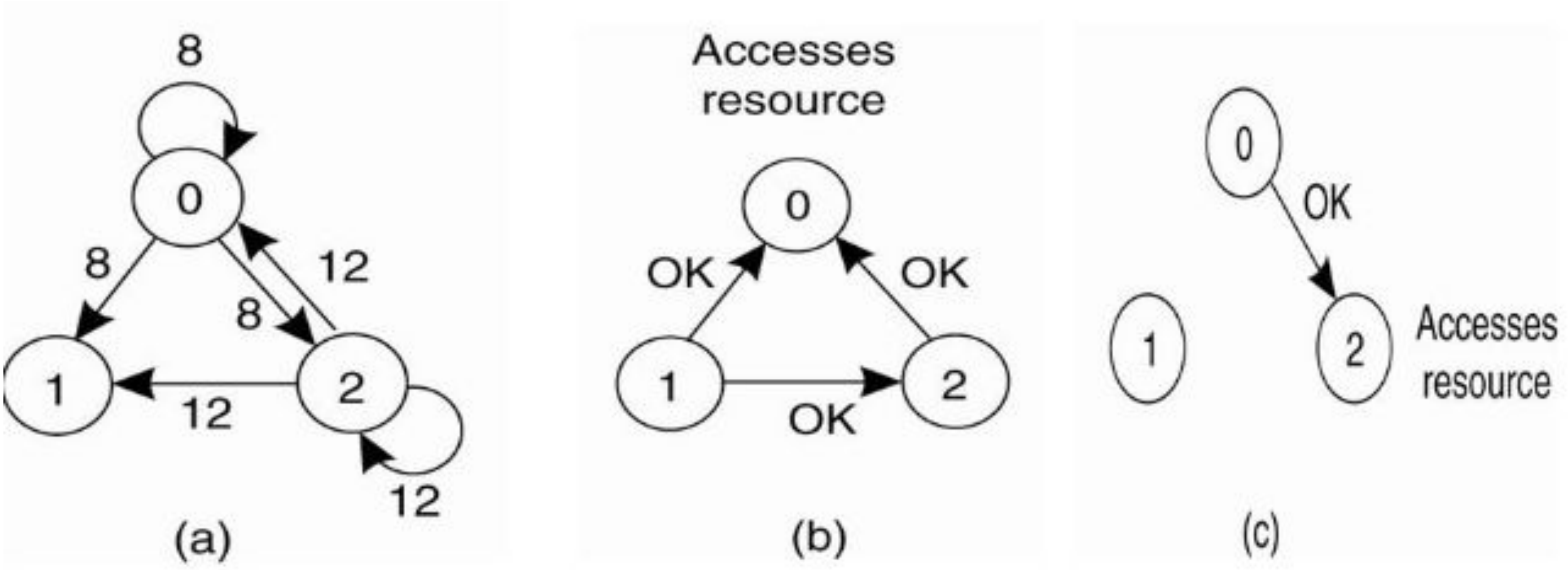
Case 2: If the receiver is already in the CS, it doesn't reply, instead it queues the request.

Contd…

Case 3:

- If the receiver wants to enter the CS but not yet done.
- It compares the timestamp in the incoming message with the one contained in the message that it has sent to everyone.
- Whichever has lowest (wins), and the winner will not say OK, rather the looser will say OK, and the winner will continue in the CS.
- After exists the CS, it sends OK message to all processes on its queue and deletes them all from the queue.

Contd…


8
0
8
12
8
1
2
12
12
(a)
Accesses
resource
0
OK
OK
1
2
OK
(b)
0
OK
1
2
Accesses
resource
(c)

Contd…

a) Process 0 and Process 2 wants to enter the same CS at the same time.

b) Process 0 has the lowest time stamp, so its win.

c) When Process 0 is done, it sends OK also, so Process 2 can now enter CS.

Note:

- In centralized algorithm, mutual exclusion is guaranteed without deadlock and starvation.
- In distributed algorithm ,the number of messages required per entry is now 2(n-1), where the total number of process in the system is ‘n’, no single point failure exists.
- But unfortunately single point failure has been replaced by n points of failure. If any process crashes, it fails to respond to requests.

Contd…

- Therefore, when a request comes in, the receiver should always sends a reply, either granting or denying permission.

- Whenever either a request or a reply is lost, the sender times out and keeps trying until either a reply comes back or the sender concludes that the destination is dead.

- This algorithm is slower, more complicated, more expensive, and less robust than the original centralized one.

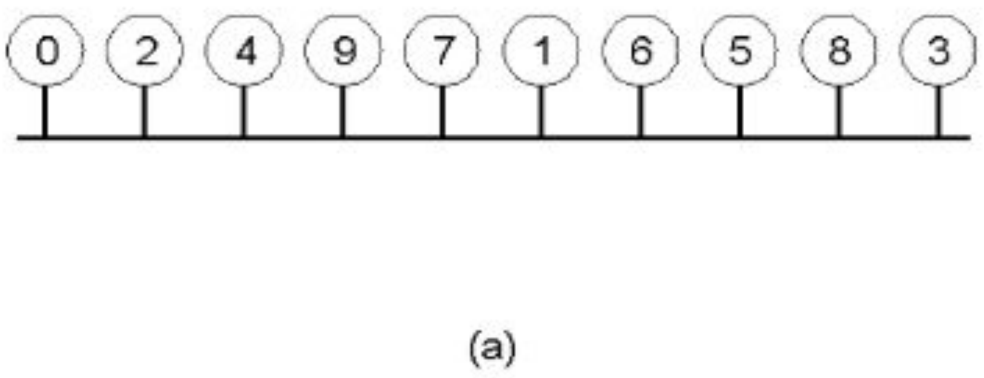
Token Ring Algorithm

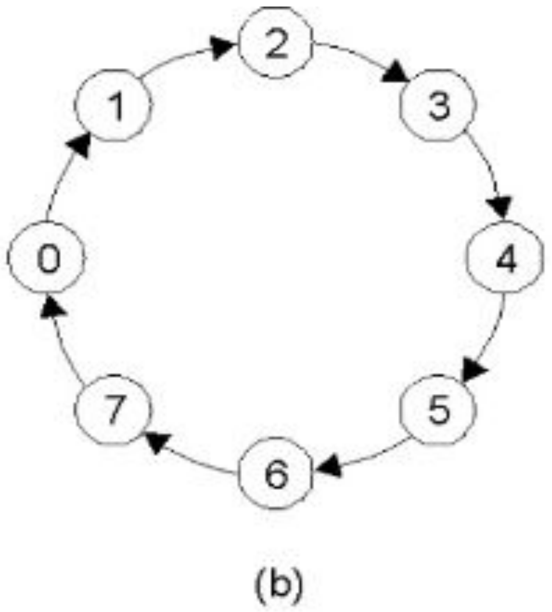
- A logical ring is constructed in which each process is assigned a position in the ring.
- It doesn't matter what the ordering is, but each process knows who is next in line after itself.
- When the ring is initialized, process 0 is given a token.
- The token circulates among the ring, as it passed from process K to process K+1 (modulo the ring size) in point to point messages.
- When a process acquires the token, it checks whether it needs to enter CS, if yes, the process enters, does all the work, and leaves the CS.
- After exited, it passes the token along the ring.

Continue…

- Not allowed to enter the second CS, with the same token.
- If the process is handed the token by its neighbour but doesn’t need it for CS, it just passes it along .
- So, when no process wants to enter any CS, the token just circulates at a speed around the ring.
- Only one process can have the token at any instant and only one process can be in a CS.
- No starvation.
- Once a process decides it wants to enter CS, at worst, it will has to wait for every other process to enter and leave one CS.

Continue…





An unordered group of processes on a network.

A logical ring constructed in software.

Comparison of the three Algorithms

Algorithms	Message per entry/exit	Delay before entry (in message times)	Problems
Centralized	3	2	Coordinator crash
Distributed	2(n-1)	2(n-1)	Crash of any process
Token ring	1 to ∞	0 to n-1	Lost token, process crash

Q&A

Thank You!