1. Explain Maekawa's mutual exclusion algorithm. How does it avoid deadlock? Create request sets if there are 10 nodes in the system.

Maekawa's Algorithm uses quorum based approach to work. In this algorithm any of the node which wants to seek permissions for a process need to only approach the nodes which are in the subset (or Quorum) of that node.

Requirements which are necessary to satisfy in the algorithm:-

-> For all i and j, Si∩Sj ≠ NULL, i ≠ j, 1≤i,j≤N -- For the resolution of request for Mutual exclusion.

->For all i, Node(i) ∈ S, 1≤i≤N -- Nodes invloved in communication in a subset.

-> For all i,|Si|=K ,1≤i≤N -- Here the size of subset is K among which message passing will happen.

-> For all j, Node(j) must be K times present in K*Si's ,1≤j≤N.

Condition for number of Nodes within a quorum:-

N=K(K-1)+1 where N is the total number of Nodes and K= O(√N)

Algorithm:-

Request Phase:- Any site which wants to enter critical section sends REQUEST messages to all other nodes in the request set of that node. And when a site recieves REQUEST message, it sends a REPLY message to the requested site if it hasn't sent a REPLY message to any other site since the reciept of last RELEASE message. If that happens then REQUEST is queued for later consideration.

CS Execution Phase:- If a site has succesfully recieved REPLY message from all the sites within subset, Then it starts executing critical section.

CS Release Phase:- When site completes its CS execution phase it sends REPLY message to every site within subset. And when a site receives a Release message from a site, it sends REPLY message to other site which was waiting in the queue and deletes its entry from the queue or If its queue is empty then site updates itself that it hasn't send any REPLY message to other site since the reciept of last RELEASE message.

Deadlock Handling:-

As Dedalock can happen in Maekawa's Algorithm, It must be handled. So, deadlock handling requires 3 types of messages:-

Failed -- A Failed message from site Si to site Sj indicates that Si cannot grant Sj's request because it has currently granted permission to a site with a higher priority request.

Inquire -- An Inquire message from Si to Sj indicates that Si would like to find out from Sj if it has succeeded in locking all the sites in its request set.

Yield -- A Yield message from site Si to Sj indicates that Si is returning the permission to Sj (to yield to a higher priority request at Sj).

By adding extra messages we can handle deadlock, all these messages work as follows:-

Suppose a node Si sends REQUEST message containing timestamps to site Sj for CS permission, And Sj has already granted permission to Sk then if the priority of Si message is lower then Sj sends it FAILED message otherwise it INQUIRE to site Sk.

If Sk gets a INQUIRE message it sends YIELD message to Sj if it has recieved FAILED message from any site or if it has sent a YIELD to any site but not recieved a new REPLY from it.

If site has recieved YIELD message it places Sk request to the queue in appropriate position and sends reply to the top Request site in the queue.

Request Sets of 10 node System:-

As there are total 10 Nodes so K=4, then all 10 request sets are:-

S1 -- 4 , 5 , 10 , 6

S2 -- 10, 2, 7, 3

S3 -- 1, 2, 3, 4

S4 -- 5 , 1, 6, 7

S5 -- 2 , 5, 8, 1

S6 -- 6 , 2, 9, 10

S7 -- 9, 4, 7, 8

S8 -- 7, 3, 8, 6

S9 -- 3, 5, 9, 4

S10 -- 8, 1, 9, 10

```
In [ ]:
```