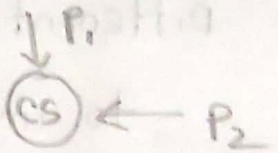
# chapter-8



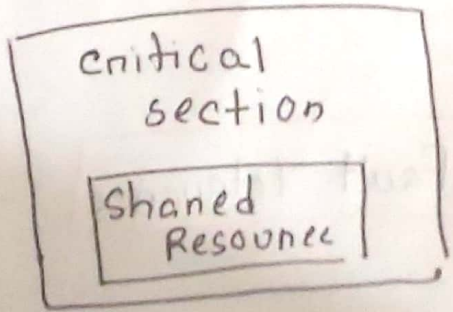

## what is mutual exclusion?

when a process is accessing a shaned variable, the process is said to be in cs (critical section)

→ No two process can be in the same cs at same time :- called Mutual Exclusion

## Distnibuted Mutual Exclusion

→ it ensunes that multiple process on nodes in distnibuted system do not coconnently access a shaned nesounces / critical section



Different Algorithm on message passing to implement mutual exclusion

1. Centralized Algorithm
2. Decentralized Algorithm
3. Token-Ring Algorithm
4. Distributed Algorithm

## Requirement of Mutual Exclusion Algorithm

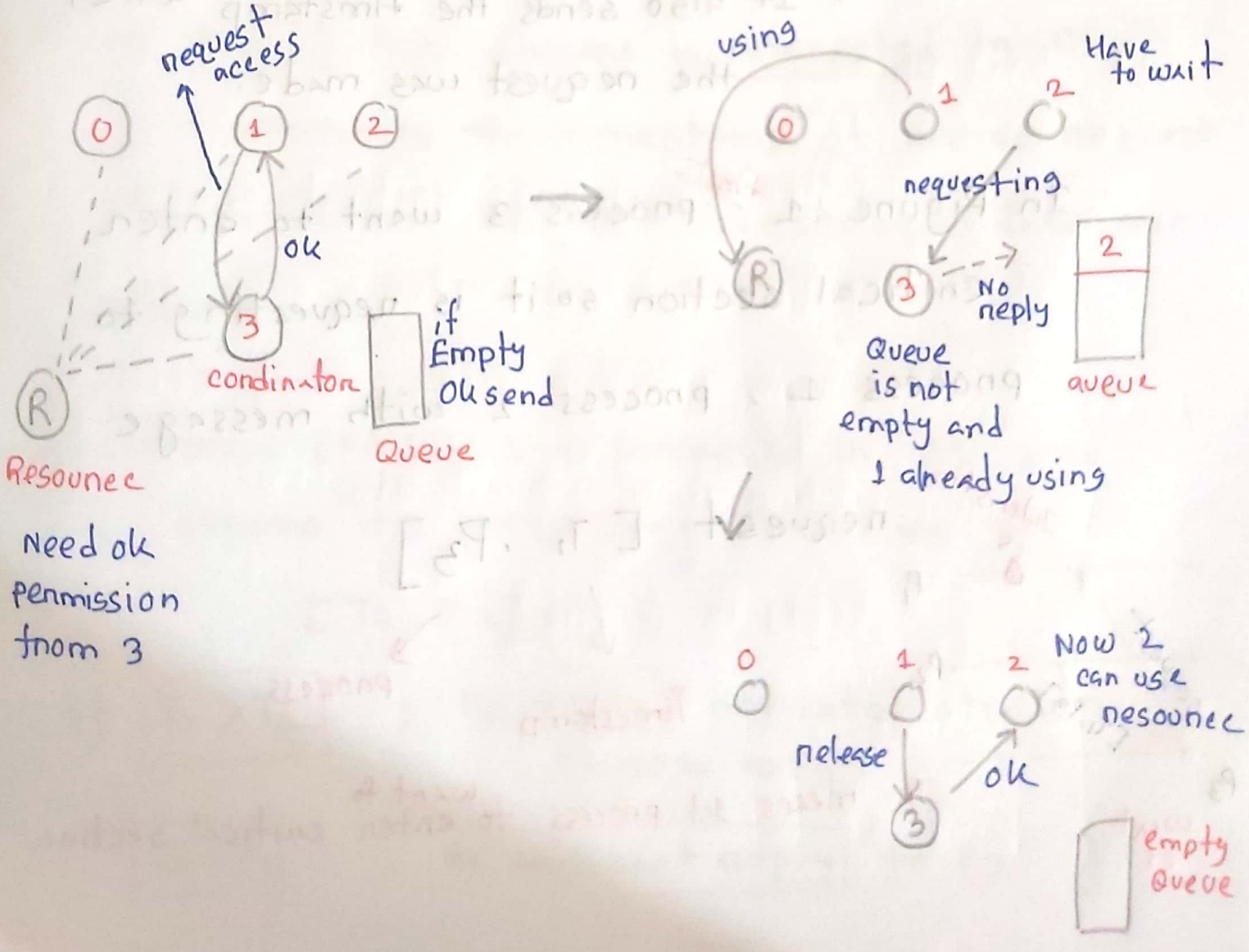
1. **No deadlocks** :- No site should be permanently blocked
2. **No starvation** :- no site have to wait while another getting multiple execution
3. **Fairness** :- request should be maintained inorder
4. **Fault Tolerance** :- algorithm fail to survive one or more sites

# Centralized Algorithm

→ one process is elected as coordinator

→ when a process want to access a shared resource, it send request to the coordinator to ask permission

→ coordinator may queue request



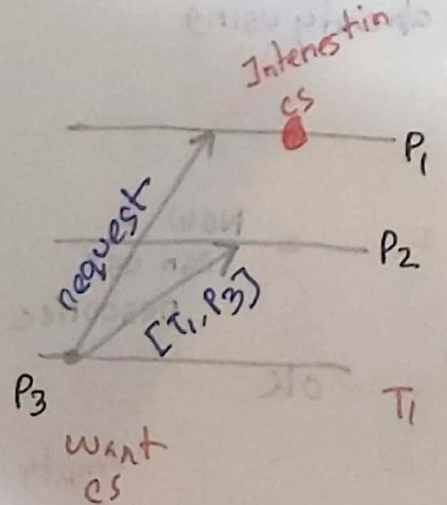
# Richant Agrawala Algorithm

→ To achive mutual exclusion

① **Request phase :-** when a process want to enter the critical section, it sends a message to all other process in the system.

→ It also sends the timstamp of when the request was made.

∴ In figure : 1 first process 3 want to enter critical section so it is requesting to process 1, process 2 with message request [ $T_1$ , $P_3$ ]




$T_1$ → Timestamp

$P_3$ → process

$T_1$ means 1st process want to enter critical section

## (2) Receiving Phase / Reply Phase :-

→ after neciving a nequest, othen pnocesses nesponds a neply message if it is not connently intenested in cnitical section

[ pnocess 2 is not intenested in cs, so it will neply no intenest to pnoces 3 ]
in cs

→ But if the pnocess is intenested in cs, it companes the timestamp of necive nequest with its own nequest timestamp to detenmine pnionity.

[ suppose pnocess 1 is intenested in cs. and assume it's nequesting Timestamp $T_2$

[ $T_2$, P 1]

if $T_1 > T_2$ ; $P_1$ Reply not intenested in cs now to $P_3$

$T_2 > T_1$ ; $P_1$ will not neply to $P_3$ ]

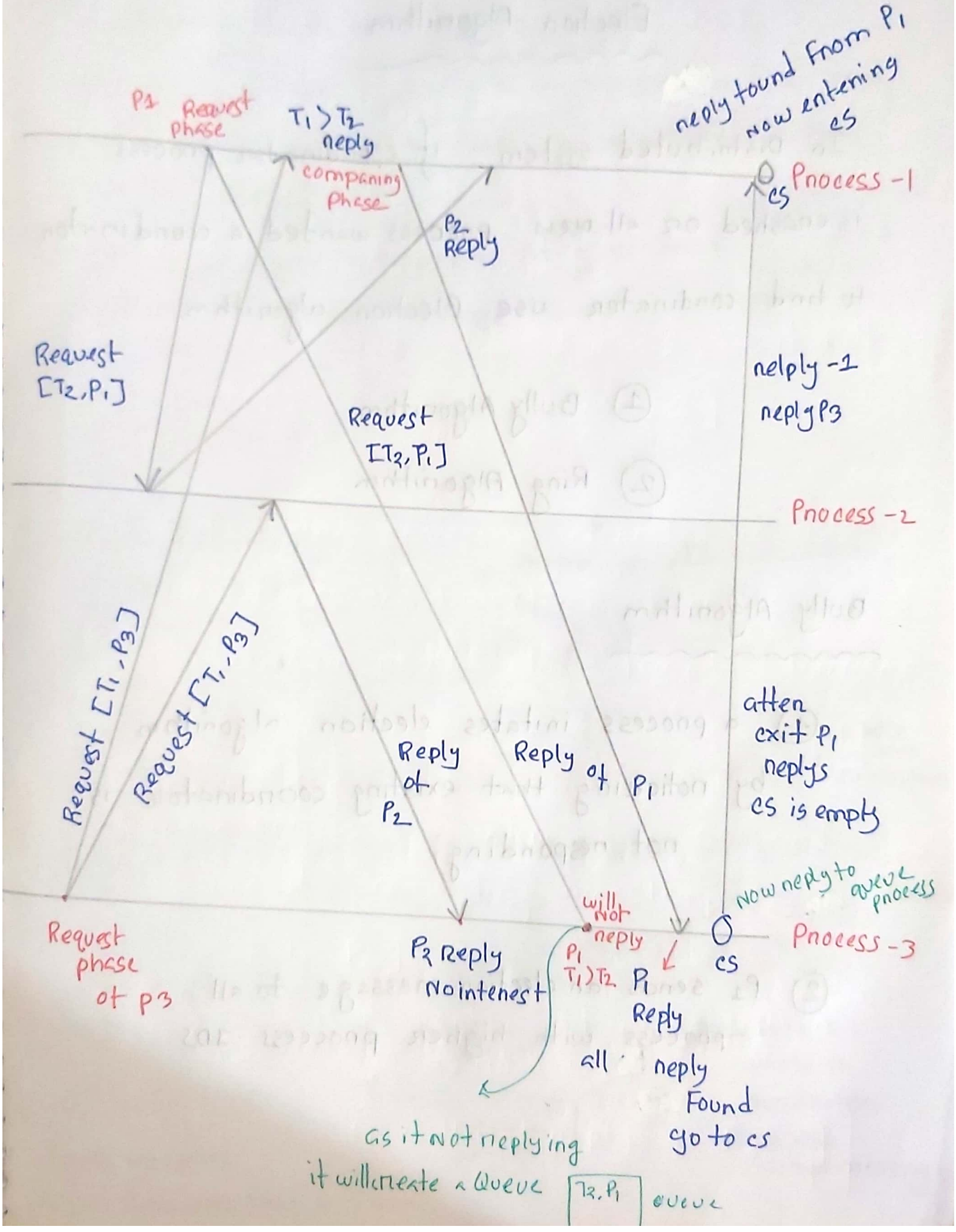
(3) Critical section Entry:-

a process can enter critical section only when it has received reply message from all other process and has highest priority

(4) Exit phase:-

after completing critical section the process send release message to inform other processes that critical section is now available

Reply to Queue


P1 Request phase
T1 > T2 reply
comparing phase
reply found from P1 now entering cs
cs
Process -1
P2 Reply
Request [T2, P1]
Request [T1, P3]
Request [T1, P3]
Request [T2, P1]
nelply -1 neplyP3
Process -2
Reply of P2
Reply of P1
atten exit P1 neplys cs is emply
now neply to queue process
will Not neply
Request phase of p3
P2 Reply Nointenest
P1 T1 > T2
P1 Reply
cs
Process -3
all neply Found go to cs
cs it not replying it will create a Queue
T2, P1
queue

# Election Algorithm

In Distributed system, if coordinator process is crashed or all process wanted a coordinator to find condinator use Election algorithm

1. Bully Algorithm
2. Ring Algorithm

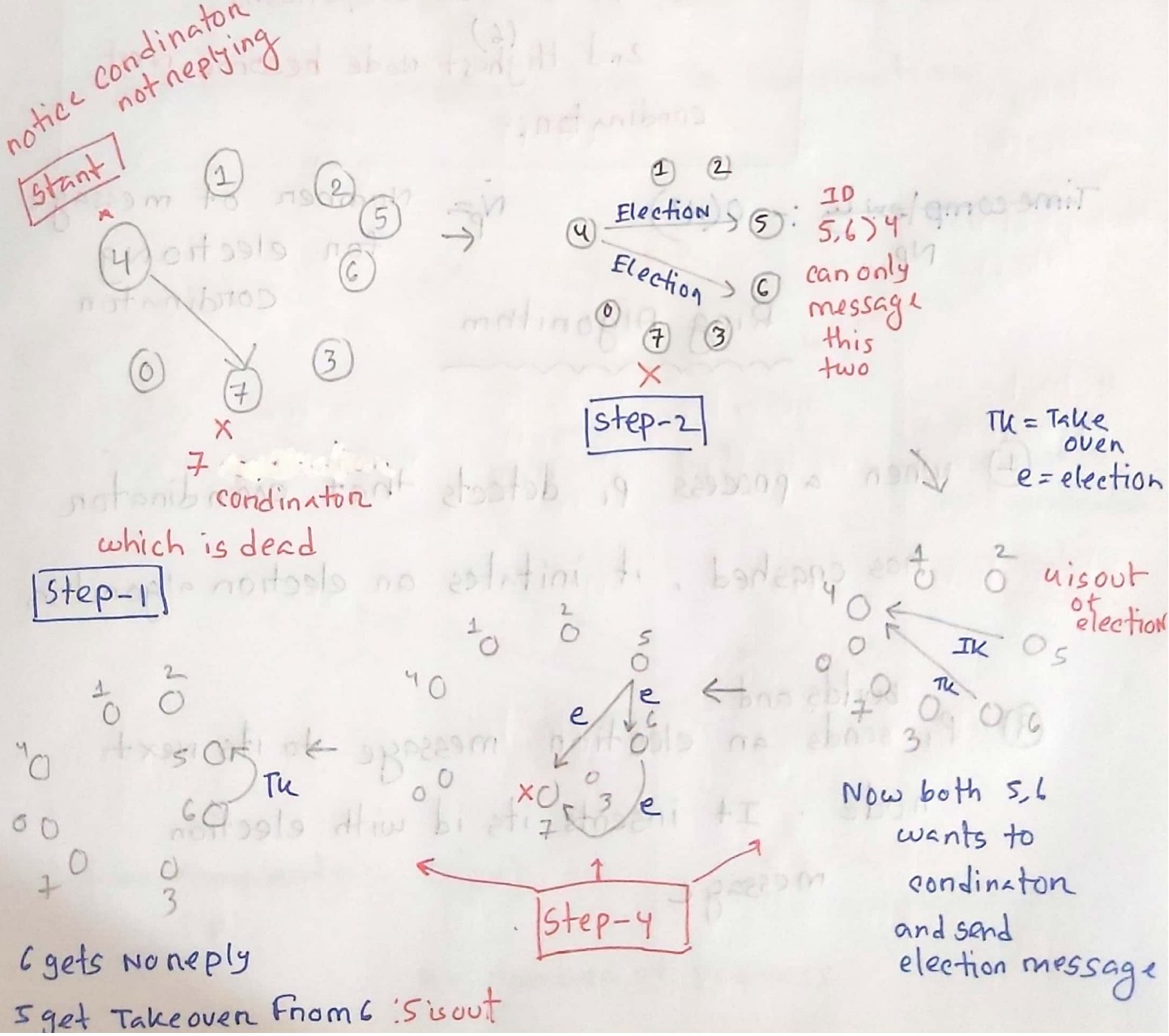
## Bully Algorithm

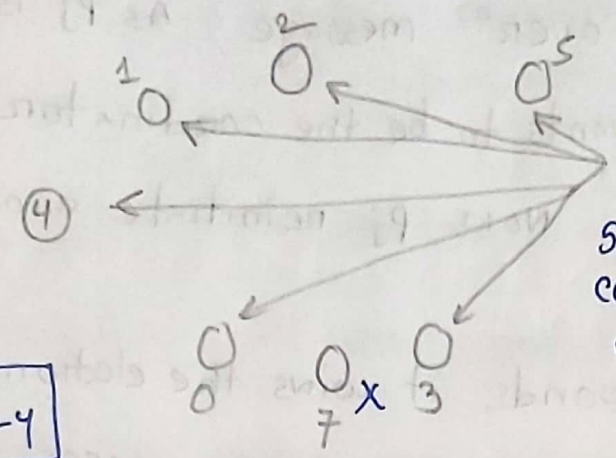
1. a process initates election algorithm by noticeing that existing coordinator is not responding

2. $P_1$ sends an election message to all process with higher processs IDs

③ when process Pj with J > i receives the message it responds with "take over" message. As PJ is Higher than Pi Now Pj wants to be the cordinator and Pi is no mone in contest. Now Pj reinitiate another call for election

④ But if Pi get no responds, it wins the election and send condinaton message to every process






6 becomes condinaton as Noone neply and win the election

step-4

N.T :- if a (7) node die/cnash, the 2nd Highest (6) Node become next condinaton :-

~~Time complexity~~ :- $O(N^2)$.

$N_p$ = Number of message for electing condinaton

$N_p$

## Ring Algonithm

(1) when a process $P_i$ detects that coondinaton has cnashed, it initates an election algonithm

(2) $P_i$ builds and sends an election message to its next node. It insents its id with election message

(3) when process Pj recives the message, it add its id also and fonwand to next node

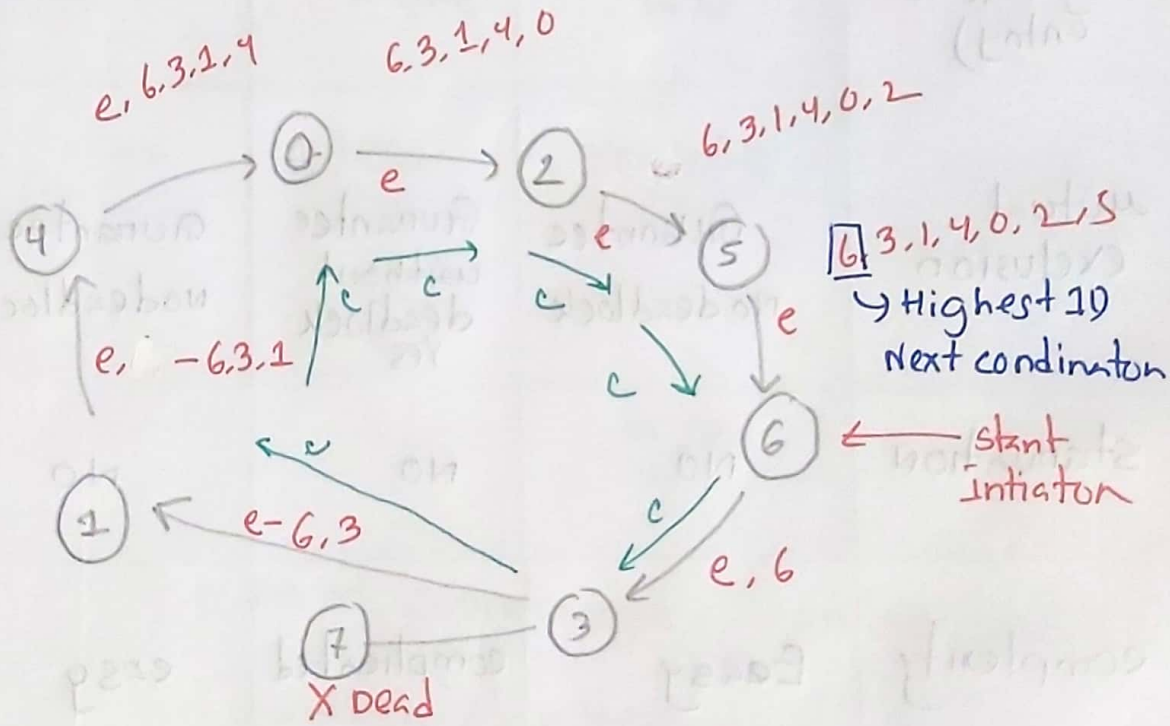
→ if next node is cnashed, Pj finds Next alive node

(4) when the message gets back to the process who stanted the election

→ it selects the highest id from list as condinaton

→ and changes message type to condination

Ring Topology Must Need




~~Time complexity~~ :- $O(2N)$

Np

e = election

c = condination

n = number of process

| | centralized | Distributed | Token Ring | De centralized |
|---|---|---|---|---|
| election | one process is elected as coondinator | Total ordering of all events in system | used token for entry in critical section | Ricart Agrawala algorithm |
| messages per entry/ exit | 3 | 2(n-1) | 1 to ∞ | 3mk |
| Delay in message times (Delay before entry) | 2 | 2(n-1) | 0 to n-1 | 2m |
| mutual exclusion | Guarantee No deadlock | Gunantee ~~without~~ deadlock Yes | Gurantee Nodeadlock | Gunante Nodeadlock |
| starvation | NO | NO | NO | yes |
| complexity | Easy | complicated | easy | medimum to hard |

| | | | | |
|---|---|---|---|---|
| used for | general allocation | small group of process with fixed membership | Process in Ring configuration | system which do not rely on central authority |
| problems | entire system can go to down if condinator fails (bottleneck) | N points of failure | lost token process is difficult | starvation low efficency |
| expense | less | more | less | more |
| robust / fault tolenance | more | less | more | more |
| Synchronizatio delay | 2 | 1 | 1 to n-1 | 2m |

m = no of coordinators contracted

k = number of attempts

Sychronization delay:-

refers to the time ~~to the~~ it takes to coordinate and sychronize activites among different process in distributed system