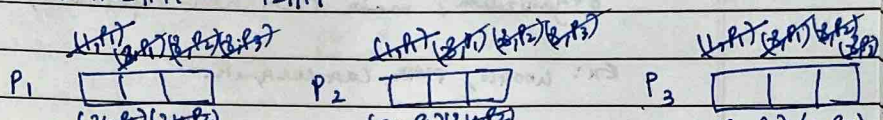
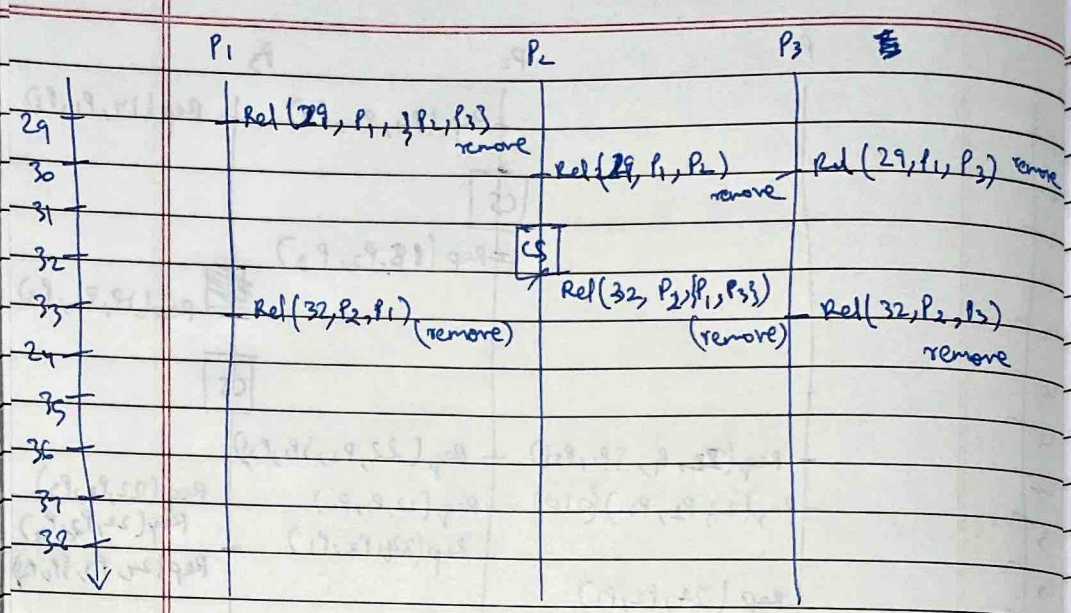
LAMPORT'S DISTRIBUTED MUTEX ALGORITM:

$P_1 \rightarrow P_3 \| P_2 \| P_1 \longrightarrow P_2 \| P_1$

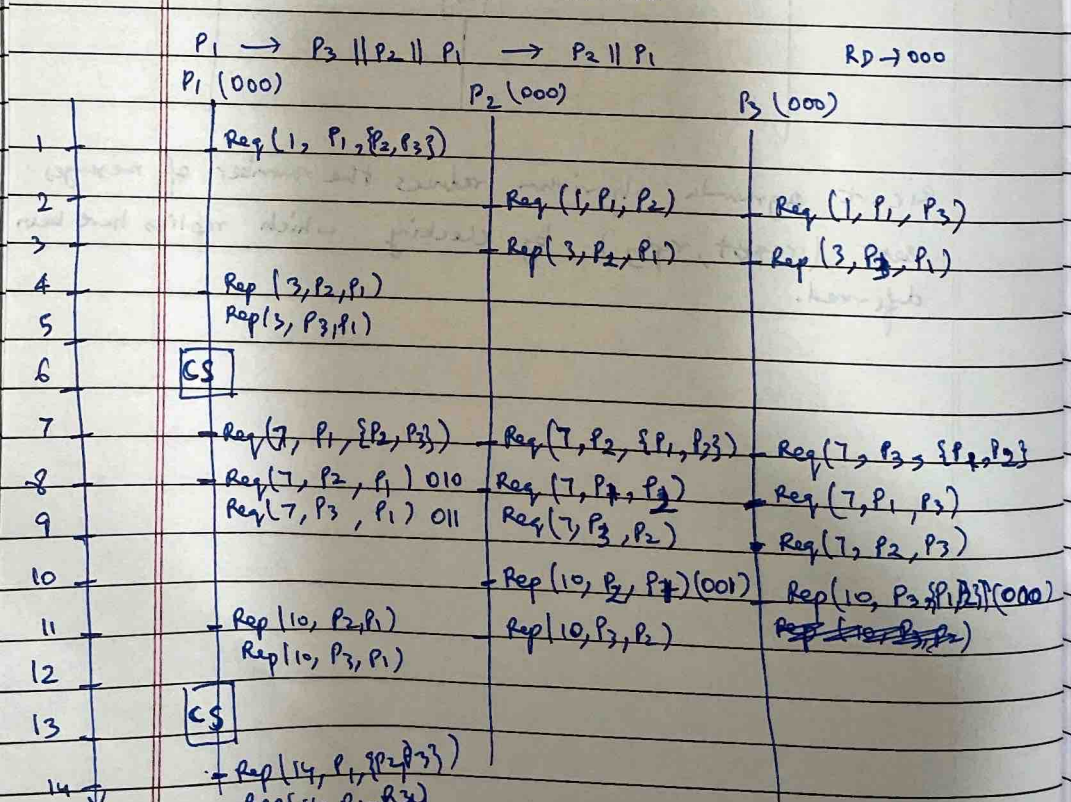


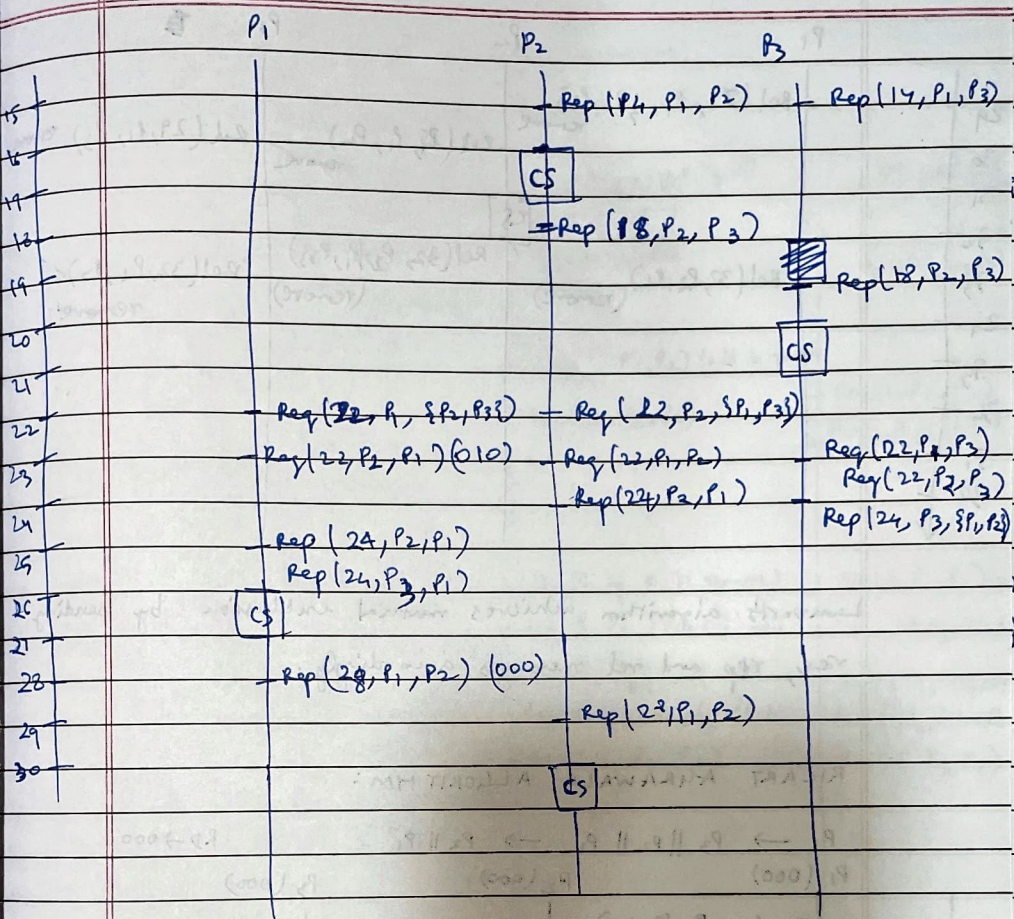
	P_1	P_2	P_3
1	Req(1, P_1, {P_2, P_3}) (Add to queue)		
2		Req(1, P_1, P_2) (add to queue)	
3		Rep(3, P_2, P_1)	Req(1, P_1, P_3)
4	Rep(3, P_2, P_1)		Rep(3, P_3, P_1) (add)
5	Rep(3, P_3, P_1)		
6	CS		
7	Rel(6, P_1, {P_1, P_2}) (remove)	Rel(6, P_1, P_2) (remove)	Rel(6, P_1, P_2) (remove)
8	Req(8, P_1, {P_2, P_3}) (add)	Req(8, P_2, {P_1, P_3}) add	Req(8, P_3, {P_1, P_3})
9	Req(8, P_2, P_1) (add)	Req(8, P_2, P_2) (add)	Req(8, P_1, P_3) (add)
10	Req(8, P_3, P_1) (add)	Req(8, P_3, P_2) (add)	Rel(8, P_2, P_3) add
11		Rep(11, P_2, P_1)	Rep(11, P_3, P_1)
12	Rep(11, P_2, P_1)		
13	Rep(11, P_3, P_1)		
14			
15	CS		
16	Rel(15, P_1, {P_2, P_3}) (remove)	Rel(15, P_1, P_2) remove	Rel(15, P_1, P_3) remove
17			
18		CS	
19	Rel(18, P_2, P_1) remove	Rel(18, P_2, {P_1, P_3}) remove	Rel(18, P_2, P_1) remove
20			
21			CS
22	Rel(21, P_3, P_2) remove	Rel(21, P_3, P_2) remove	Rel(21, P_3, {P_2, P_3}) remove
23			
24	Req(24, P_1, {P_2, P_3}) add	Req(24, P_2, {P_1, P_3}) add	Req(24, P_1, P_3) add
25	Req(24, P_2, P_1) add	Req(24, P_1, P_2) add	Req(24, P_2, P_3) add
26		Rep(26, P_2, P_1)	Rep(26, P_3, P_1)
27	Rep(26, P_2, P_1) Rep(26, P_3, P_1)		
28	CS		



Lamport's algorithm achieves mutual exclusion by sending req, rpp and rel messages accordingly.

RICART AGRAWALA ALGORITHM:





Ricart Agrawala algorithm reduces the number of messages sent (request, reply) by checking which replies have been deferred.