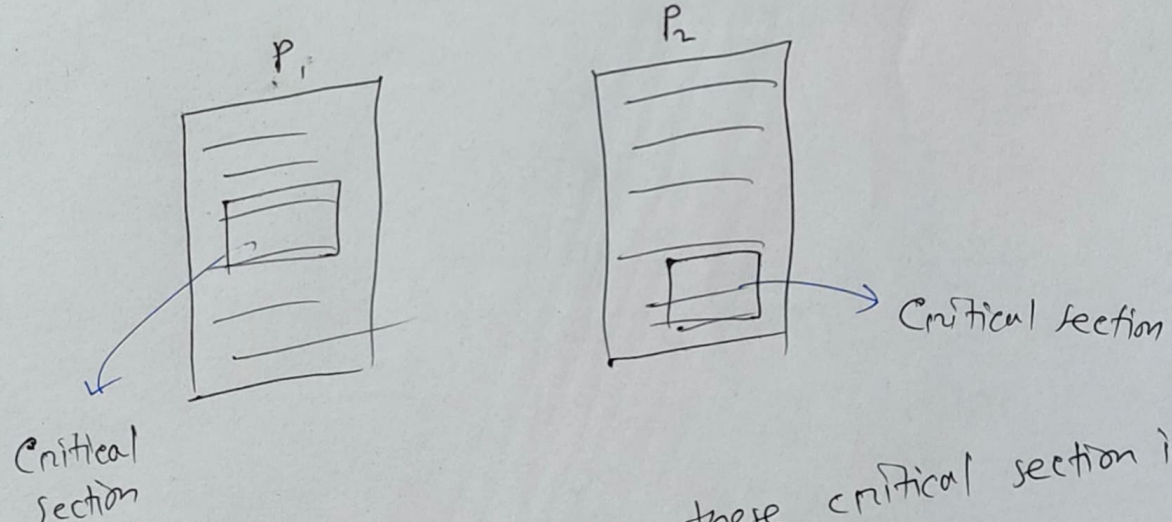
CSE 323 / L-21 / 02.05.2024 /

Midterm-2

L-22 / 11.05.2024 /

Critical Section Problem

If a part of a process is dependend on another part of another process, then we can't run them in parallel. This is known as critical section problem, and the part is known as critical section.



⇒ We can't run these critical section in parallel. But we can run other parts in parallel.

For entering in critical section, process need to ask for permission.

General structure

```
do {
    entry section
        critical section
    exit section
        remainder section
} while (true);
```

Solution to Critical section Problem

Generic Solution

(i) Mutual Exclusion:
if a process executing in critical section, then others process can't execute in their critical section. That means only one process can run in critical section.

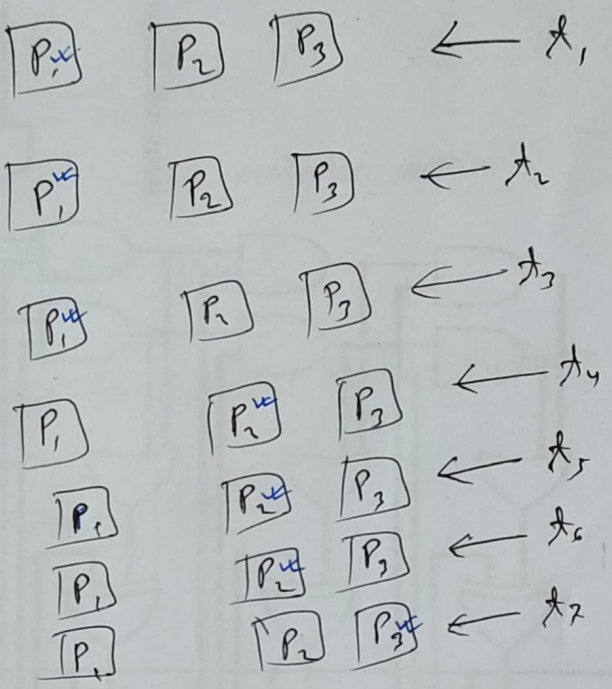
(ii) Progress: If multiple proccess wants to enter their critical section, then we can't ignore all, we must select one from them.

(iii) Bounded Waiting!

There is limit ~~to~~ for a process to enter in critical section in continuow sequence if any other process are waiting for enter in their critical section.

asume limit is 3

Let's say



Peterson's Solutions (Critical Problem)

- load and store instruction are atomic can't be interrupted
- use two variable ⇒ applied between two process only.

P1
P2

int turn ⇒ 1/2 → Second Process
1 → first process

Boolean flag[2]

flag[1] T/F

flag[2] T/F

T = Ready to enter
F = Not ready to enter

P1 ⇒ turn = 1 && flag[1] = T

P2 ⇒ turn = 2 && flag[2] = T

⊗ if P_1 wants to enter into its critical section, then it will change it flags to True.

flag[1] = True

And Turn = 2 (not set as own, it offer to second Process.)

if (flag[2] == True && Turn == 2)

if this statement false, then no process will enter to critical section. P_1 will still wait.

{ the P2 runs its critical section

flag[2] = False

}

P_1 waits

⊗ When P_2 wants to enter, it will offer to ~~second~~ other one process.

flag[2] = True

Turn = 1

if (flag[1] == True && turn == 1)

{ then, P_1 will run in critical section

flag[1] = False

}

P_2 waits.

Project Report - individual

Intro
Theory
working diagram
Code

Slide - 12, 13

Project Show = 16th
Quiz-2 = 23th