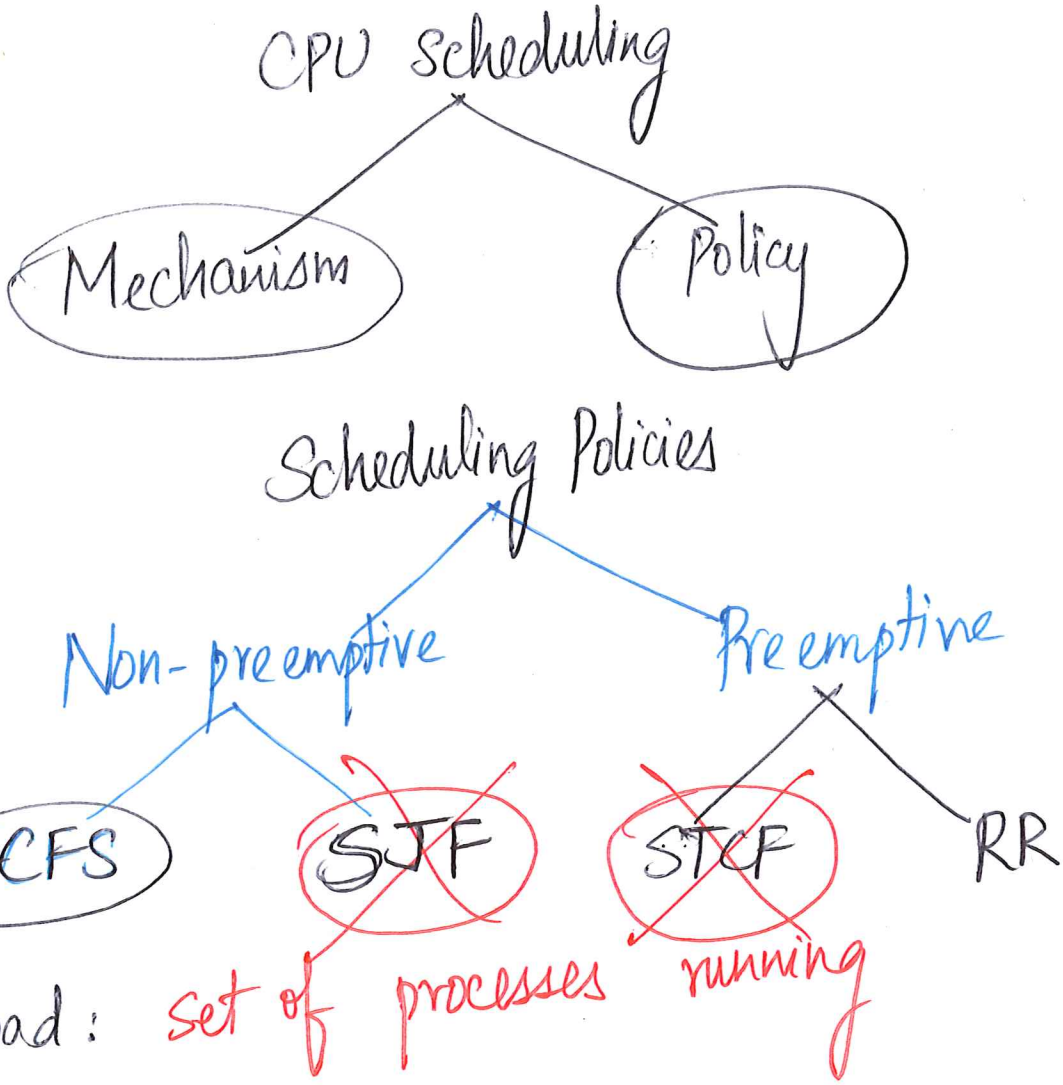
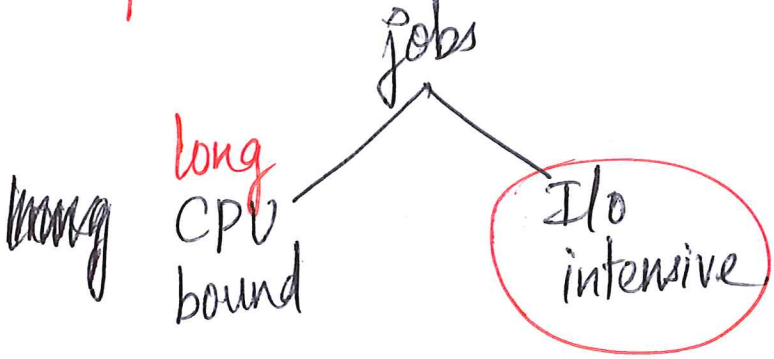


workload : set of processes running

job : process.



Turnaround Time ↓

Response Time ↓

Scheduling

Workload Assumptions

~~1. Each job runs for the same amount of time.~~

~~2. All jobs arrive at the same time.~~

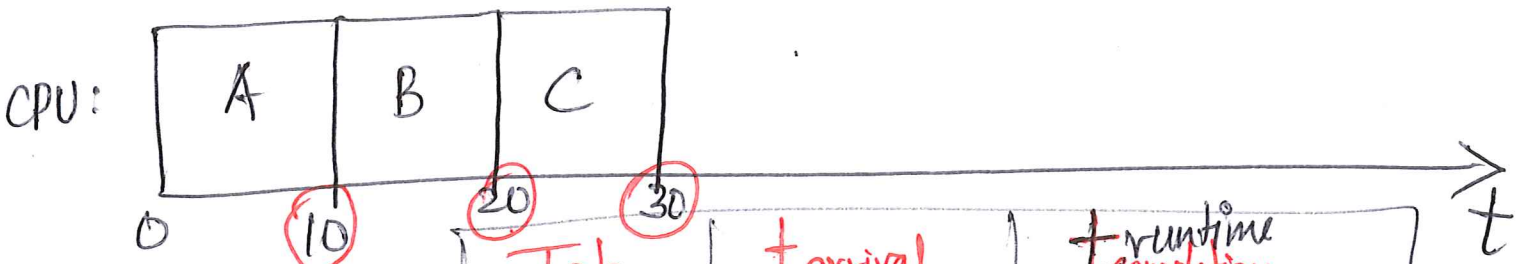
~~3. Once started, each job runs to completion.~~

~~4. All jobs only use the CPU (i.e., they perform no I/O).~~

~~5. The run-time of each job is known.~~

worst

1. FCFS (FIFO)



Job	$t_{arrival}$	~~$t_{completion}$~~ $t_{runtime}$ (CPU burst time)
A	0	10
B	0	10
C	0	10

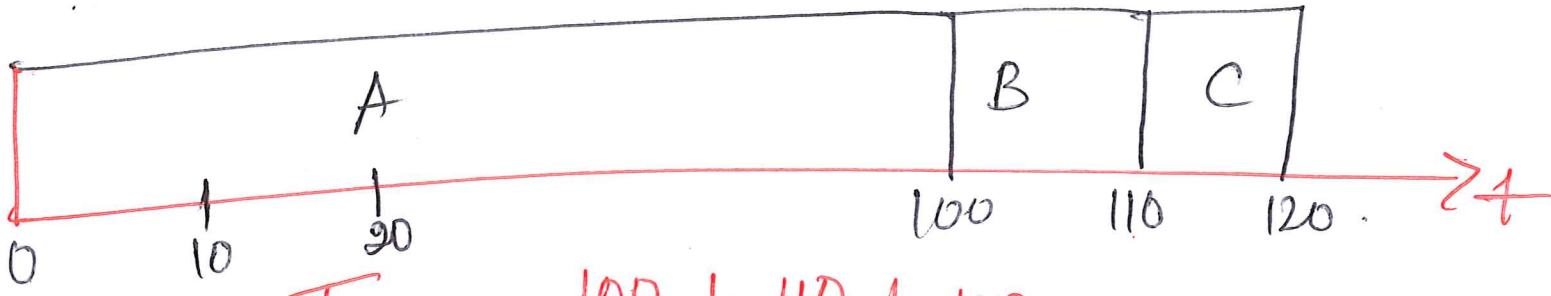
Scheduling Metric: Turnaround Time

$$T_{(T.A)} = T_{completion} - T_{arrival}$$

$$\text{avg. } T_{T.A.} = \frac{(10-0) + (20-0) + (30-0)}{3}$$

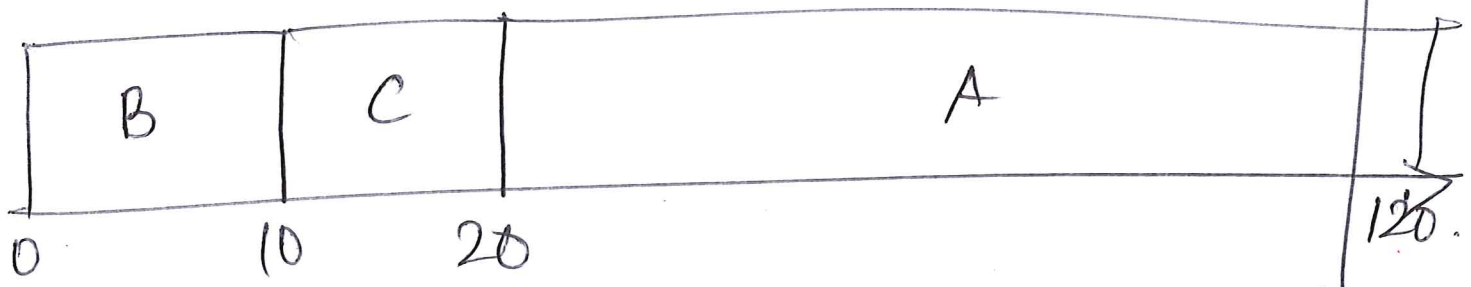
$$= \frac{60}{3} = 20$$

Job	ta	tr
A	0	100
B	0	10
C	0	10



$$T_{T.A.} = \frac{100 + 110 + 120}{3} = \frac{330}{3} = 110$$

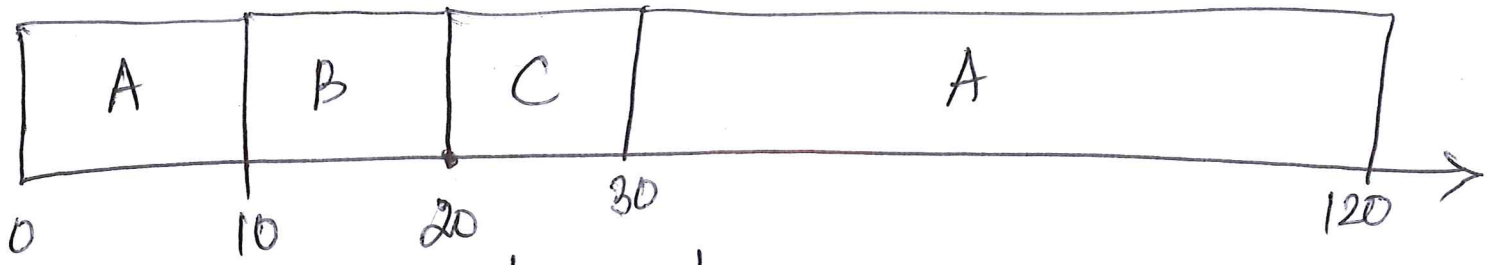
2. SJF



$$T_{T.A} = \frac{10 + 20 + 120}{3} = \frac{150}{3} = 50$$

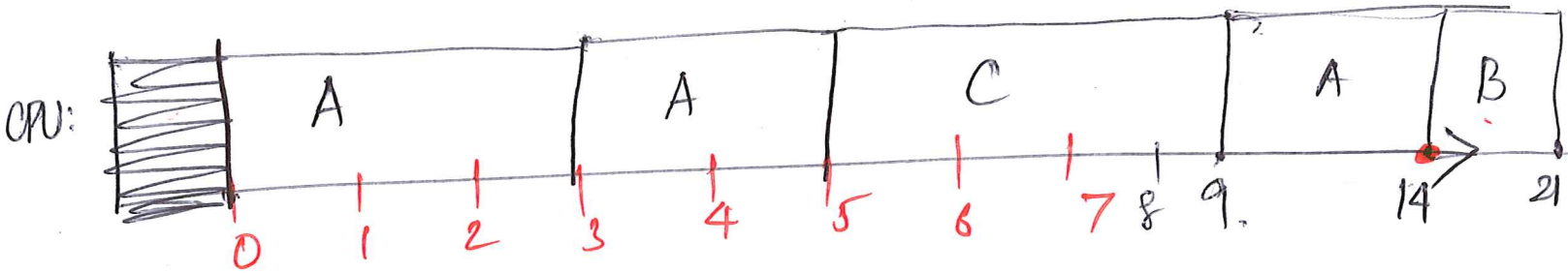
	ta	tr
A	0	100
B	10	10
C	20	10

3. STCF (PSJF)



	ta	tr
A	0	10
→ B	3	7
C	5	4

interactive.



$$\text{Avg. } T_{T.A.} = \frac{(14-0)+(21-3)+(9-5)}{3}$$

$$= \frac{14+18+4}{3} = \frac{36}{3} = 12$$

Response Time

Response = $T_{firstrun} - T_{arrived}$

$T_R(A) = 0$

$T_R(B) = 14-3 = 11$

$T_R(C) = 5-5 = 0$

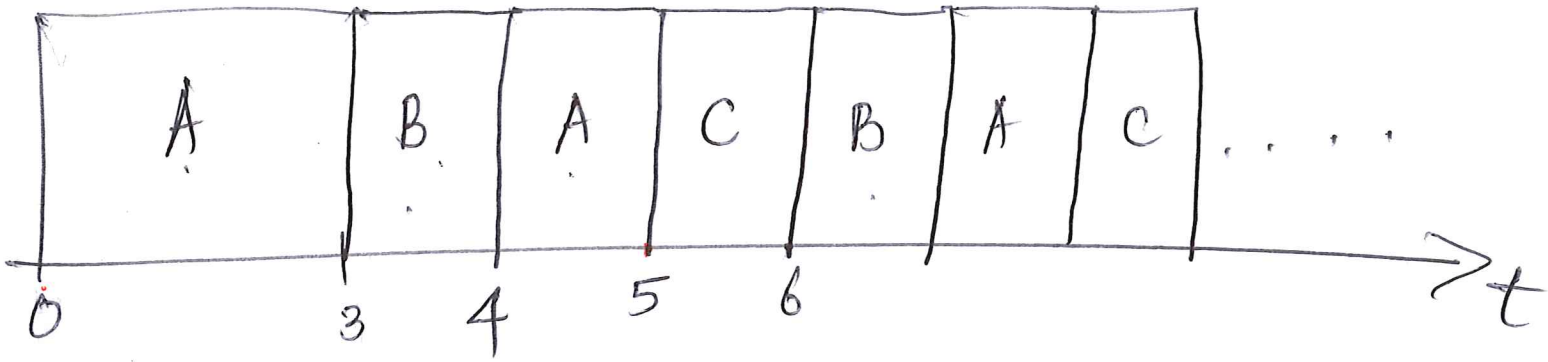
$11/3$ avg. T_R

4. RR

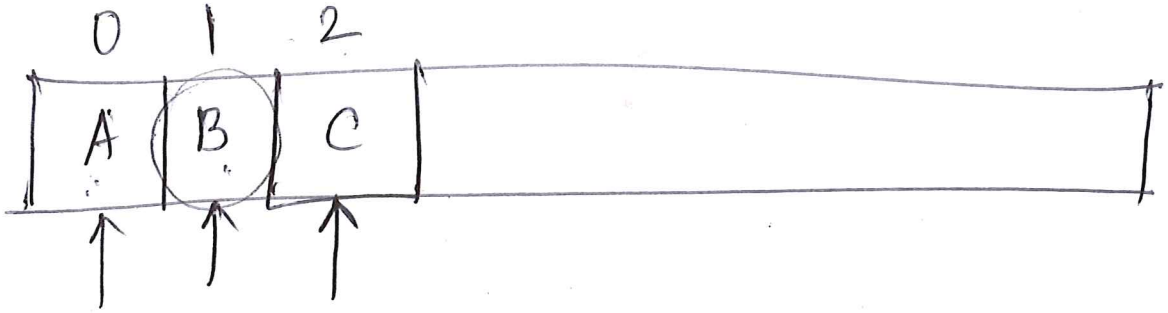
A	0	10
B	3	7
C	5	4

time slice
quantum.

$q = 1$



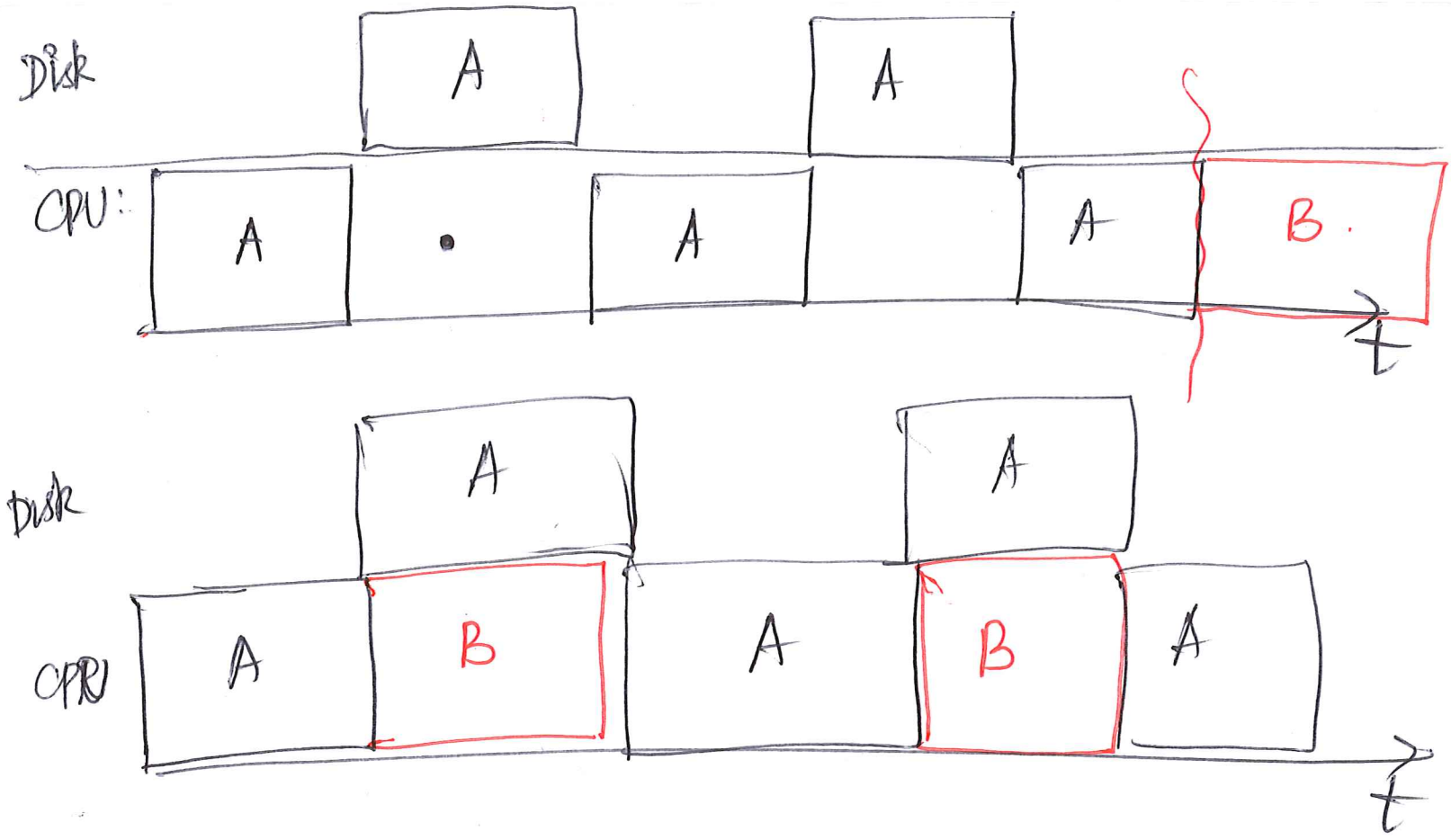
Avg. $T_R = \frac{0+0+0}{3} = 0.$



$q = \infty$?

$q = 0$?

$q = nt$ → timer interrupt (10ms)


Disk
A
A
CPU:
A
A
A
B
t
Disk
A
A
CPU
A
B
A
B
A
t

MLFQ

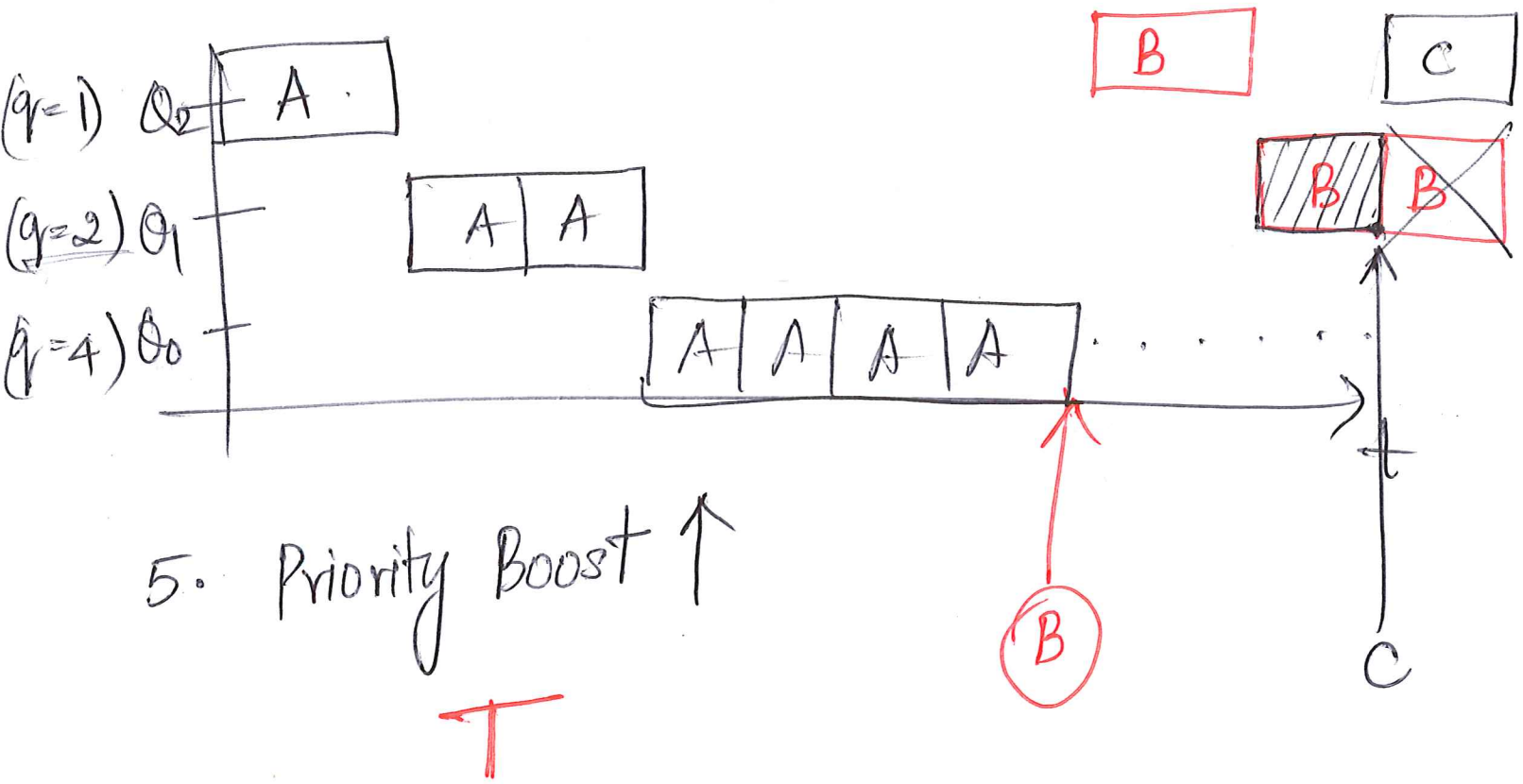
Multi Level Feedback Queue

highest priority Q_2 — A B

Q_1 — C

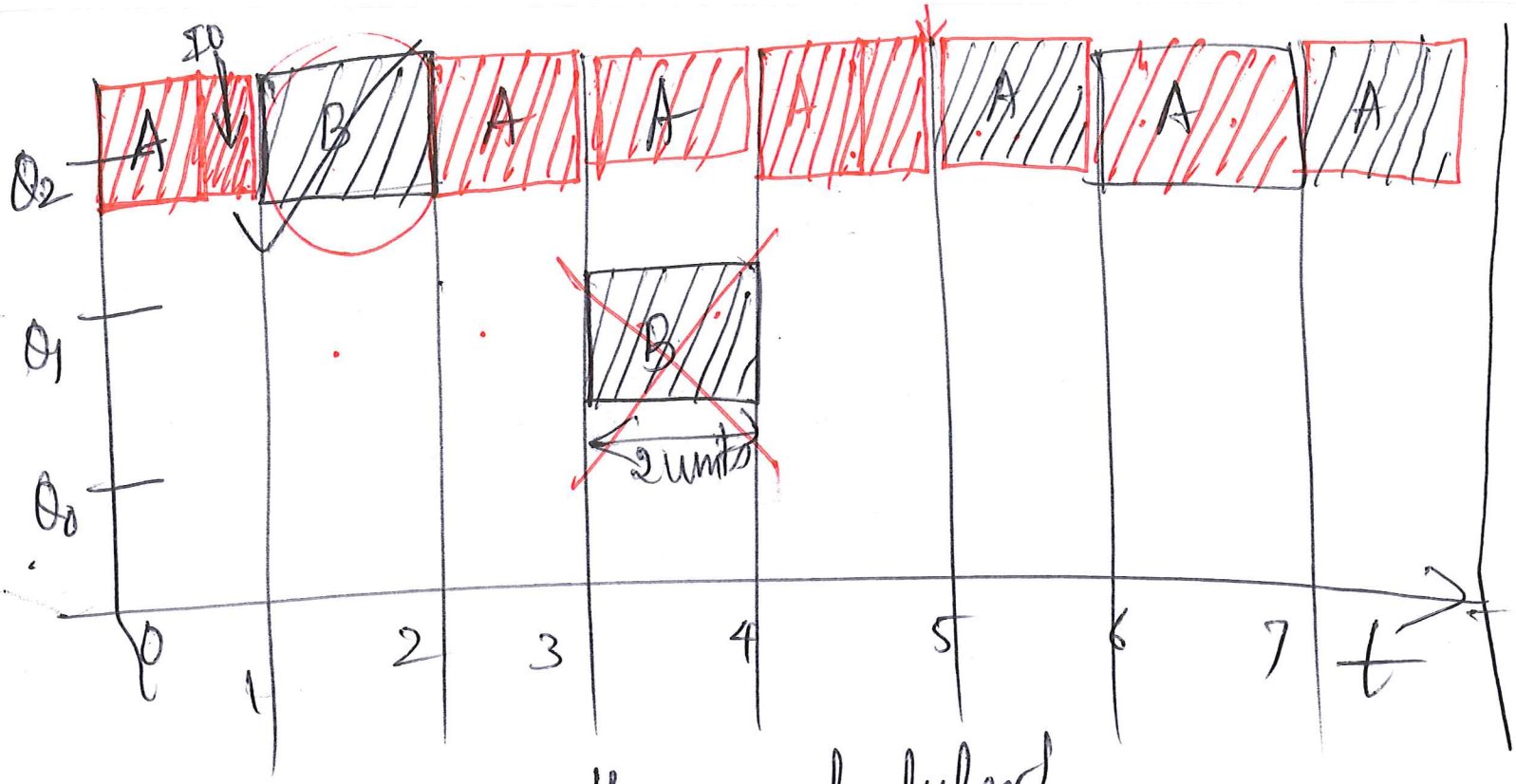
lowest priority Q_0 — D E

CPU: A B A B A B A B . . . . . .



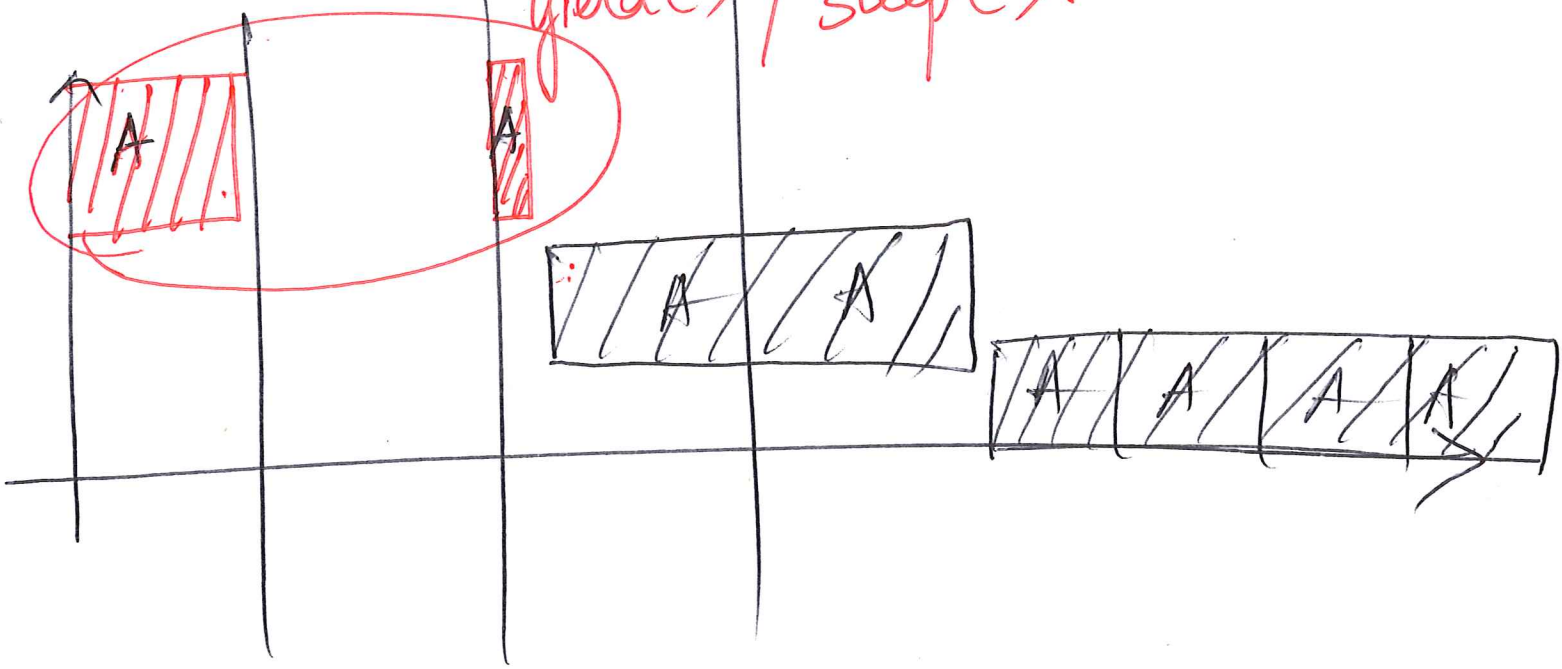
5. Priority Boost ↑

T



Gaming the scheduler!

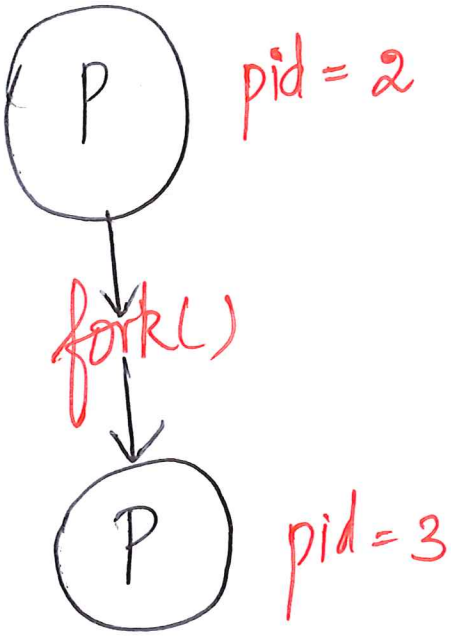
yield() / sleep().

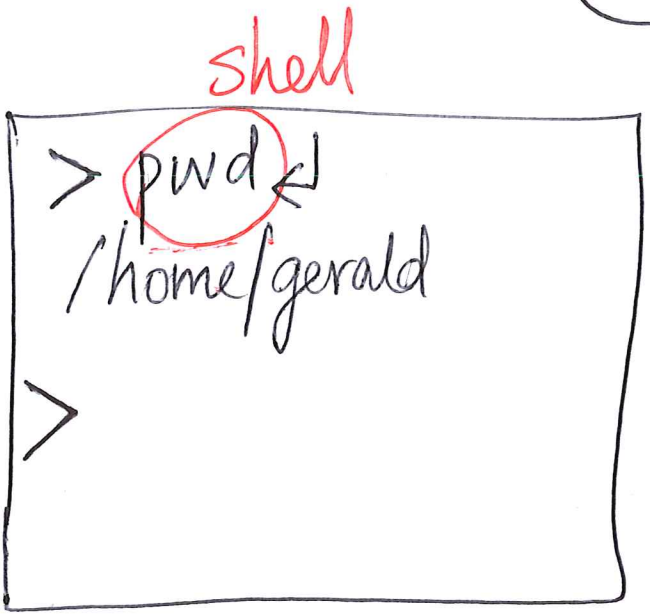


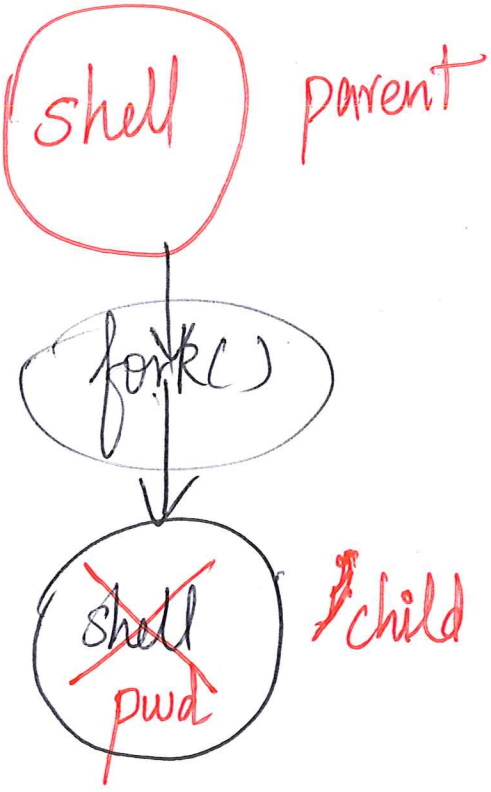
Process APIs

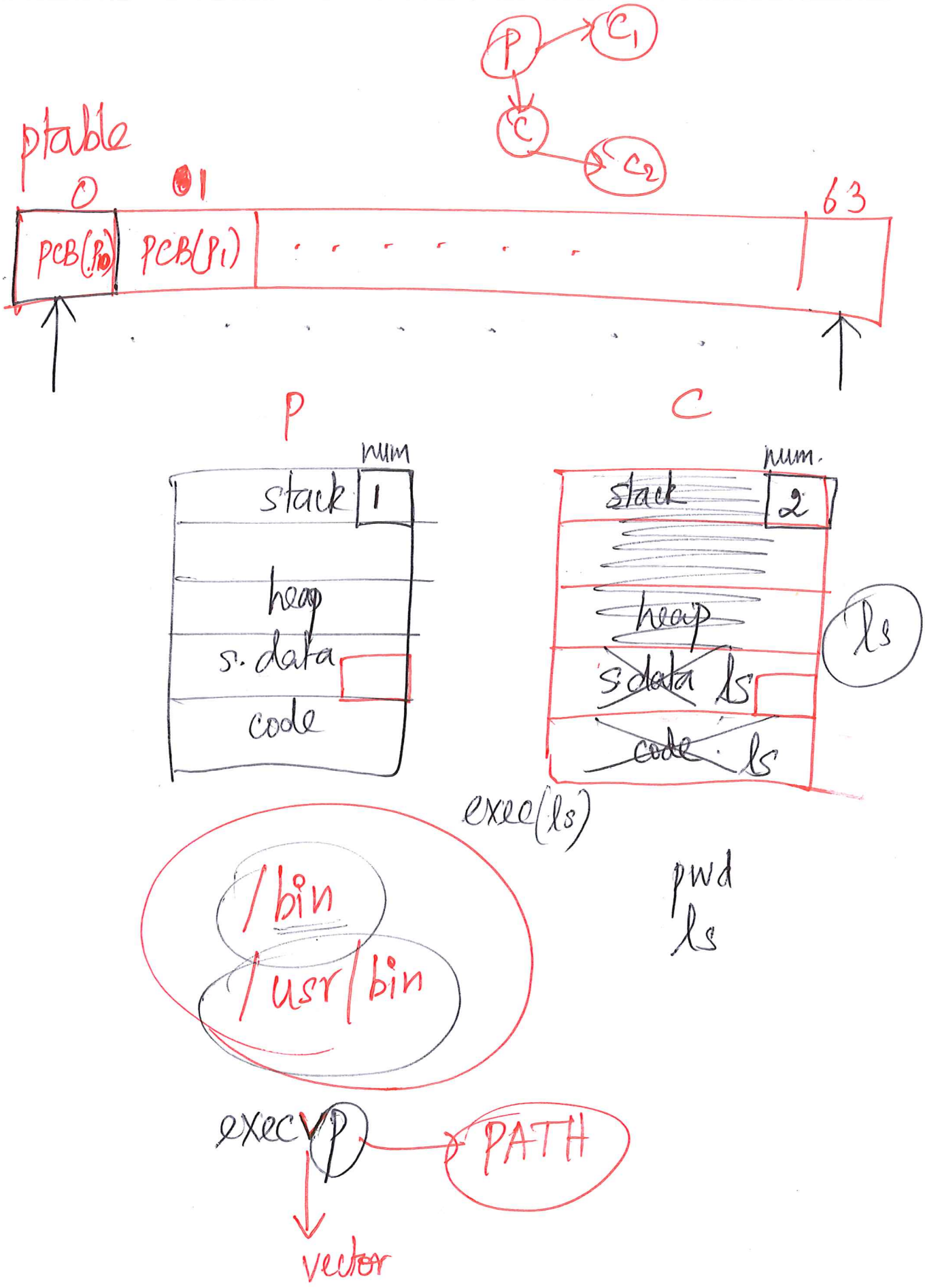
1. create?

fork()








P
C1
C
C2
ptable
0
1
63
PCB(P0)
PCB(P1)
P
C
num
stack
1
heap
s. data
code
num.
stack
2
heap
s data ls
code ls
ls
exec(ls)
/bin
/usr/bin
pwd
ls
execvp
PATH
vector

MLFQ

Basic Rules

1. if Priority(A) > Priority(B), A runs.
2. if Priority(A) = Priority(B), A & B run in RR.

How to change priority?

3. when a job enters the system, it is placed at the highest priority (topmost queue).

4a. If a job uses up an entire time slice while running, its priority is reduced.

4b. If a job gives up the CPU before the time slice is up, it stays at the same priority level.

MLFQ

Basic Rules: 1, 2

How to change priority: 3, 4a, 4b.

Priority Boost

Rule 5: After some time period S, move all the jobs in the system to the topmost queue.

Better Accounting

Rule 4 (new): Once a job uses up its time allotment at a given level, its pribrity is reduced.