- FCFS is the simplest scheduling time.
- SJF is ~~the~~ mostly used in industry.

lecture-6
BDU
01/02/18

SJF - better in production environment than development environment.
- For new programs, we can't ~~do~~ apply SJF. So first time we use FCFS and SJF afterwards.
- We allow shortest CPU burst time job first.

$$\tau_{n+1} = \alpha t_n + (1-\alpha)\tau_n$$

τ_{n+1} : current history of CPU burst
τ_n : past history of CPU burst
t_n : contains most recent information
α : constant that controls the relative weight of recent and past history for the prediction of CPU burst. $0 \leq \alpha \leq 1$

$\tau_{n+1} = \tau_n$ ← • When $\alpha = 0$, the recent prediction has no effect

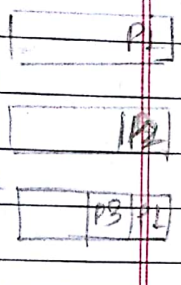
$\tau_{n+1} = t_n$ ← • When, $\alpha = 1$, only the most recent CPU burst matters.

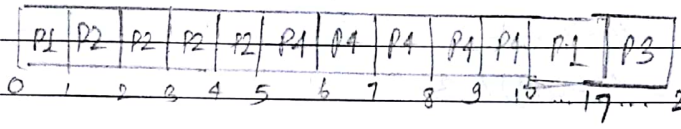
$\tau_{n+1} = \frac{1}{2}t_n + \frac{1}{2}\tau_n$ • When $\alpha = 1/2$, the recent and past history of CPU burst equally matter

Preemptive SJF / Shortest remaining time (SRT)

Process	Arrival time	Burst time
P_1	0	8 7 → at time interval 1
P_2	1	4 3 → this is smaller than all left. So, P2 continues
P_3	2	9
P_4	3	5 4

(Since burst time of P2 is less than remaining time of P1 (7>4) so we context switch P2 with P1.





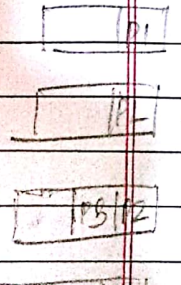
$T_{TRND}(P1) = 17 - 0 = 17$ (When it finished - when it arrived at ready list)
$T_{TRND}(P2) = 5 - 1 = 4$
$T_{TRND}(P3) = 26 - 2 = 24$
$T_{TRND}(P4) = 10 - 3 = 7$

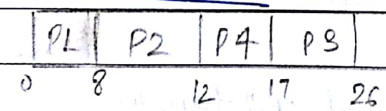
Avg. $T_{TRND} = (17+4+24+7)/4 = 13$

$T_w(P1) = (10-1) - 0 = 9$ (When it got CPU - when it arrived at ready list)
$T_w(P2) = (1-1) = 0$
$T_w(P3) = 17 - 2 = 15$
$T_w(P4) = 5 - 3 = 2$

Avg. $T_w = (9+0+15+2)/4 = 6.5$

Non-preemptive SJF for comparision —





$T_{TRND}(P1) = 8 - 0 = 8$
$T_{TRND}(P2) = 12 - 1 = 11$
$T_{TRND}(P3) = 26 - 2 = 24$
$T_{TRND}(P4) = 17 - 3 = 14$

Avg. $T_{TRND} = (8+11+24+14)/4 = 14.25$

$T_w(P1) = 0 - 0 = 0$
$T_w(P2) = 8 - 1 = 7$
$T_w(P3) = 17 - 2 = 15$
$T_w(P4) = 12 - 3 = 9$

Avg. $T_w = (0+7+15+9)/4 = 7.75$

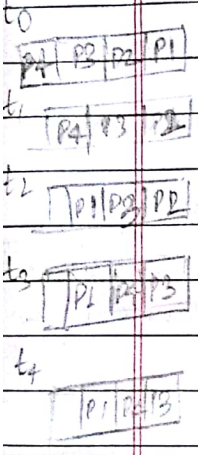
Round Robin Algorithm

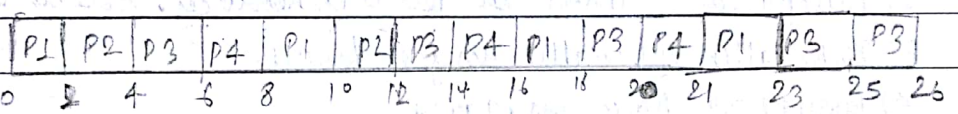
- Suitable for time sharing environment.
- FCFS but preemptive scheduling
- The process release the CPU based on time quanta or according to its completion.

Process	CPU burst	~~Arrival time~~
P_1	8 6 4	0
P_2	4 2 0	1
P_3	9 7	2
P_4	5 3	3

(Time quantum = 2)

→ No arrival time means all arrive at once at t_0





Lecture-7

BDU 05/02/18

$T_{TRND}(P1) = 23$
$T_{TRND}(P2) = 12$
$T_{TRND}(P3) = 26$
$T_{TRND}(P4) = 21$

Avg. $T_{TRND} = 20.5$

Avg $T_W = 13.5$

$T_W(P1) = (8-2) + (16-10) + (21-18)$
$= 6 + 6 + 3$
$= 15 \rightarrow T_{TRND}$ - CPU burst

$T_W(P2) = 2 + (10-4)$
$= 8 = T_{TRND}$ - CPU burst

$T_W(P3) = 4 + (12-6) + (18-14) + (23-20)$
$= 4 + 6 + 4 + 3$
$= 17 = T_{TRND}$ - CPU burst

$T_W(P4) = 6 + (14-8) + (20-16)$
$= 6 + 6 + 4 = 16 = T_{TRND}$ - CPU burst

* If the time quantum is smaller, context-switching overhead increase.
* If the time quantum is larger, it becomes plain FCFS

Both cases doesn't improve turn around time optimally.