Starvation

Starvation problem - disadvantage

: process in the ready state waiting indefinitely to get to CPU.
└ event occurence keeps being postponed

: any priority based scheduling algorithm has a chance of starvation (ex. SJF, SRTF)
= low priority process can be starving

Convoy effect

convoy effect - disadvantage

: a smaller process (process with very small execution time)
waiting for one big process to get off (release) CPU. (ex. FCFS, SJF)
⇒ higher waiting time, higher TAT

Practical implementation

· SJF, SRTF가 FCFS에 비해 throughput이 낫다. throughput ↑ faster, more efficient computer

· 그러나 SJF, SRTF를 실질적으로 적용하기는 어렵다. 모든 process의 burst time을 알아야하기 때문
: burst time (execution time)은 복합적인 요소로 인해 정해짐. ex. system power 등

Throughput

: number of processes executed per unit time.

How will you get the maximum throughput? burst time이 작은 것을 먼저 실행하는 것이 좋다.
(조건: processes are infinite)

① SRTF ② SJF ③ FCFS

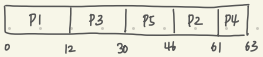
LJF Longest Job First scheduling algorithm

: process with the longest burst time will be scheduled first # non-preemptive # priority based

* disadvantages
  1) starvation
  2) convey effect
  3) low throughput
  4) practically very difficult to implement

(ex)

process Id	1	2	3	4	5
arrival time	0	1	2	3	4
burst time	12	15	18	2	16



CT	12	61	30	63	46
TAT	12	60	28	60	42
WT	0	45	10	58	26
RT	0	45	10	58	26

schedule length = 63

throughput = 5/63

LRTF Longest Remaining Time First Scheduling algorithm

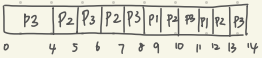
: process with the longest burst time will be scheduled first # preemptive # priority-based

* disadvantages
  1) starvation
  2) convey effect
  3) low throughput
  4) practically very difficult to implement

(ex)

process Id	1	2	3
arrival time	0	0	0
burst time	2	~~4~~ 3	~~8~~ ~~4~~ 2 ~~2~~

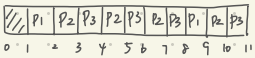
(2번) burst time이 같다면 arrival time 빠른, arrival time이 같다면 process Id 빠른



(ex)

process Id	1	2	3
arrival time	1	2	3
burst time	2	4	4

(2번) burst time이 같다면 arrival time 빠른, arrival time이 같다면 process Id 빠른



CT	9	10	11
TAT	8	8	8
WT	6	4	4
RT	0	0	0

schedule length = 11 - 1 = 10

throughput = 3/10

RR Round Robin Scheduling algorithm

windows, mac 등에서 많이 사용되는 algorithm

Time Quantum : Maximum allowable time a process can run without getting preempted.

RR : TQ + FCFS - FCFS이지만 일정 time unit 동안 실행

- works on basis of a particular time quantum
- uses queue data structure
- very popular & used in most of the OS today

(ex)

process Id	1	2	3	4	5	6
arrival time	0	1	2	3	4	5
burst time	~~4~~ ~~3~~ 1 ~~1~~ 0	~~3~~ 0	~~1~~	~~6~~ ~~4~~ 2	~~3~~ 1	

* time quantum : 2

Queue: P1 P2 P3 P1 P4 P5 P2 P6 P5 P4 P6 P5

Gantt Chart: P1 | P2 | P3 | P1 | P4 | P5 | P2 | P6 | P5 | P2 | P6 | P5

0 2 4 6 8 9 11 13 15 17 18 19 21

* time quantum : 3

Queue: P1 P2 P3 P4 P1 P5 P6 P2 P5

Gantt Chart: P1 | P2 | P3 | P4 | P1 | P5 | P6 | P2 | P5

3 6 8 9 10 13 16 18 21

⇒ as time quantum increases, the number of context switching may decrease

⇒ 〃 , the response time increases

Advantages

- No starvation : not a priority based
- No convoy-effect
- Practically implementable
- Response time ↓

Limitation

- Throughput is good but not as good as SJF, SRTF