CS341 #32 - Scheduling & Scheduling Algorithms

wait for I/O, sync primitive

Why might a process be placed on the ready queue?

time quanta expired

What is 'wait time'? Total wait time, or ~~the first waiting before it is scheduled the first time?~~

Write a formula for the wait time based on arrival time, execution time(=duration) and completion time

duration 时长 wait time = $(t_{complete} - t_{arrival}) - t_{exec}$ when it happen 时刻

Scheduling. Some terms... How shall I compare thee?

"Turnaround time"

"Waiting time"

"Response time" stop at "first byte" appear

"Throughput"

"Latency"

"Starvation? Waiting in queue for my icecream ...

Through put

Good for Batch? Good use of CPU/IO resources?

Good for Interactive? good response time

Good for real-time systems? → hard constrain exists!

scheduling algorithms

FCFS (aka.....) First Come First served

SJF vs Pre-emptive SJF → SJF, but pre-emptive if shorter job appear

→ will interrupt things

Shortest Job First → hard to know how "short" it is

RR Round Robin → CPU goes to each process circularly tedious

→ no starvation

Priority-scheduling Priority queue

Choosing an appropriate time-quantum. What does scheduler does Linux use?

1 ms ?

completely fair scheduler

keeps track of history CPU usage

Which schedulers can suffer from starvation?

Which schedulers are appropriate for batch jobs?

Which schedulers are appropriate for interactive jobs?

What scheduler does Linux use?

Determine the scheduling sequence and calculate the average wait time of the following schedulers
Tie-break #1: Schedule the earliest arriving job. Tie break #2: P4 is placed on ready queue first

Round robin (quanta = 10ms)

Process	Arrival Time(ms)	Burst Time(ms)	Wait Time (ms)
P1	0	30	50
P2	0	20	
P3	0	20	
P4	10	10	20

0..10	.. 20	..30	..40	..50	.. 60	.. 70	..80
P1	P2	P3	P4	P1	P2	P3	P1

Shortest Job First

Process	Arrival Time(ms)	Burst Time(ms)	Wait Time (ms)
P1	0	30	
P2	0	20	
P3	0	20	
P4	10	10	

0..10	20	30	40	50	60	70	80

First Come First Served (assume arrive in order P1,P2,P3)

Process	Arrival Time(ms)	Burst Time(ms)	Wait Time (ms)
P1	0	30	
P2	0	20	
P3	0	20	
P4	10	10	

0..10	..20	30	40	50	60	70	80

Pre-emptive Shortest Job First (assume interrupted jobs are placed at the front of the queue)

Process	Arrival T	Burst T	Wait T
P1	0	30	
P2	0	20	
P3	0	20	
P4	10	10	

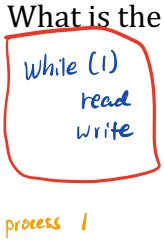
0..10	..20	30	40	50	60	70	80

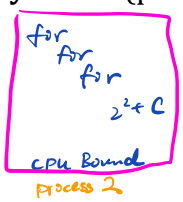
Pre-emptive Priority (higher value = higher priority)

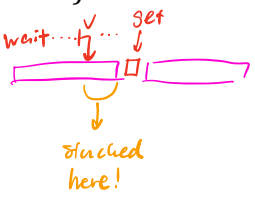
Process	Arrival	Burst	Priority	Wait
P1	0	30	2	
P2	0	20	4	
P3	0	20	1	
P4	10	10	3	

0..10	..20	30	40	50	60	70	80

What is the **Convoy Effect** (poor I/O parallelism)?







Round Robin

Process	Arrival Time(ms)	Burst	Wait
P1	0	30	50
P2	0	20	40
P3	0	20	50
P4	10	10	20

0..10	20	30	40	50	60	70	80
P1	P2	P3	P4	P1	P2	P3	P1

Wait = (End-Arrival) - Execution duration
(P1:) 50 + (P2:) 40 + (P3:) 50 + (P4:) 20 = 160ms. Average Wait = 40 ms

Shortest Job First (**Not** shortest remaining time)

Process	Arrival Time(ms)	Burst	Wait
P1	0	30	50
P2	0	20	0
P3	0	20	30
P4	10	10	10

0..10	20	30	40	50	60	70	80
P2	P2	P4	P3	P3	**P1**	P1	P1

Total Wait = 50 + 30 + 0 + 10 = 90 ms. Average wait = 90/4 = 22.5 ms

First Come First Served (assume arrive in order P1,P2,P3)

Process	Arrival Time(ms)	Burst	Wait
P1	0	30	0
P2	0	20	30
P3	0	20	50
P4	10	10	60

0..10	20	30	40	50	60	70	80
P1	P1	P1	P2	P2	P3	P3	**P4**

Total Wait = 0 + 30 + 50 + 60 = 140 ms. Average wait = 35 ms

Pre-emptive Shortest Job First

Process	Arrival Time(ms)	Burst
P1	0	30
P2	0	20
P3	0	20
P4	10	10

0..10	20	30	40	50	60	70	80
P2	**P4**	P2	P3	P3	P1	P1	P1

Total Wait = 50 + 10 + 30 + 0 = 90 ms. Average wait = 22.5 ms

Pre-emptive Priority (higher value = higher priority)

Process	Arrival (ms)	Burst (ms)	Priority
P1	0	30	2
P2	0	20	4
P3	0	20	1
P4	10	10	3

0..10	20	30	40	50	60	70	80
P3	P3	P1	P1	P1	P4	P2	P2

Total Wait = 20 + 60 + 0 + 40 = 120 ms. Average wait = 30.0 ms

Which scheduler has poor I/O parallelism (suffers from the "Convoy Effect")?
FCFS (Processes that could be using I/O have to queue behind long-running CPU job). Note, you could also make a similar argument for non-premptive SJF.
Which schedulers can suffer from starvation?
Pre-emptive SJF (long jobs may never be scheduled); Pre-emptive priority (low priority jobs may never be scheduled)
Which schedulers are appropriate for batch jobs? Ans: Depends on your requirements!
What scheduler does Linux use? What about threads? What does *nice* do?
Completely Fair Scheduler ("Stride scheduler"; inspired from similar network flow scheduling – gives additional time to processes that are in the waiting state more often than the executing state "If you only took small sips in the recent past, you can take longer drink now")