CSc 360
Operating Systems
Scheduling Algorithms

Jianping Pan

Summer 2015

We value your feedback

- Course reps (email on connex)
  - Tim Chan (CS)
  - Alice Gibbon (Combined)
  - Cole McGinn (SEng)
  - FCFS and also thanks to many volunteered
- Let us know how we can help you better
  - either directly and welcomed too
  - or via reps: anonymize, aggregate, amplify

Review: CPU scheduling

- CPU scheduler
  - short-term scheduling
- CPU scheduling
  - criteria: CPU, throughput, delay, etc
  - goal: max throughput, min delay
- FCFS
  - “convoy effect”
    - “job size”

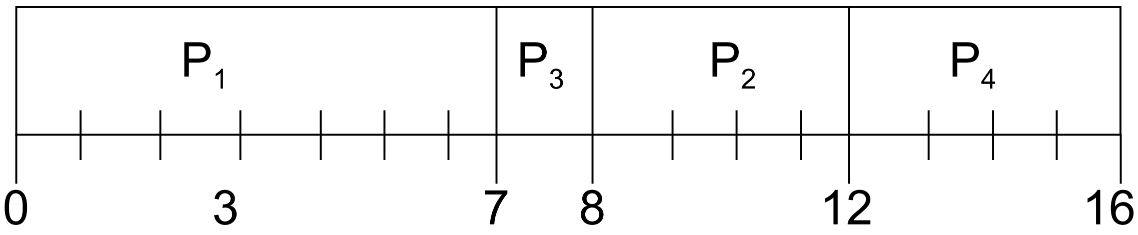
Shortest job first

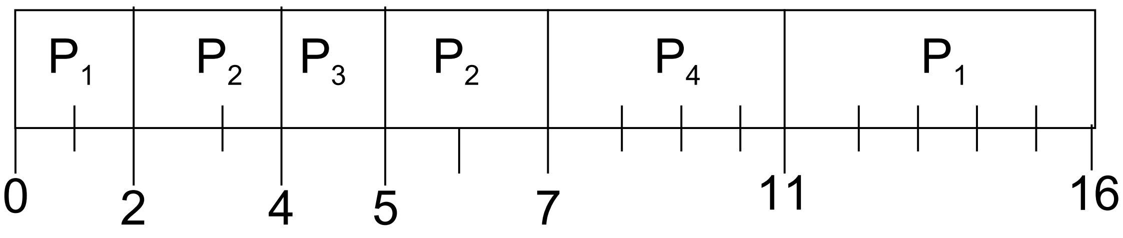
- SJF: based on the length of *next* CPU burst
  - non-preemptive
    - the job with the smallest burst length scheduled first
  - or preemptive
    - i.e., always the shortest remaining time first
- SJF is optimal in average waiting time
  - reduce the total waiting time for all jobs
  - why ***is*** SJF optimal?

Q: possible problems with SJF?

SJF: example

- Example
  - P1: 0 (arrival time); 7 (burst time)
  - P2: 2; 4
  - P3: 4; 1
  - P4: 5; 4
- Non-preemptive
  - P1, P3, P2, P4
- Preemptive
  - P1, P2, P3, P2, P4, P1





Shortest Remaining Time First (SRTF)

Q: turnaround time?

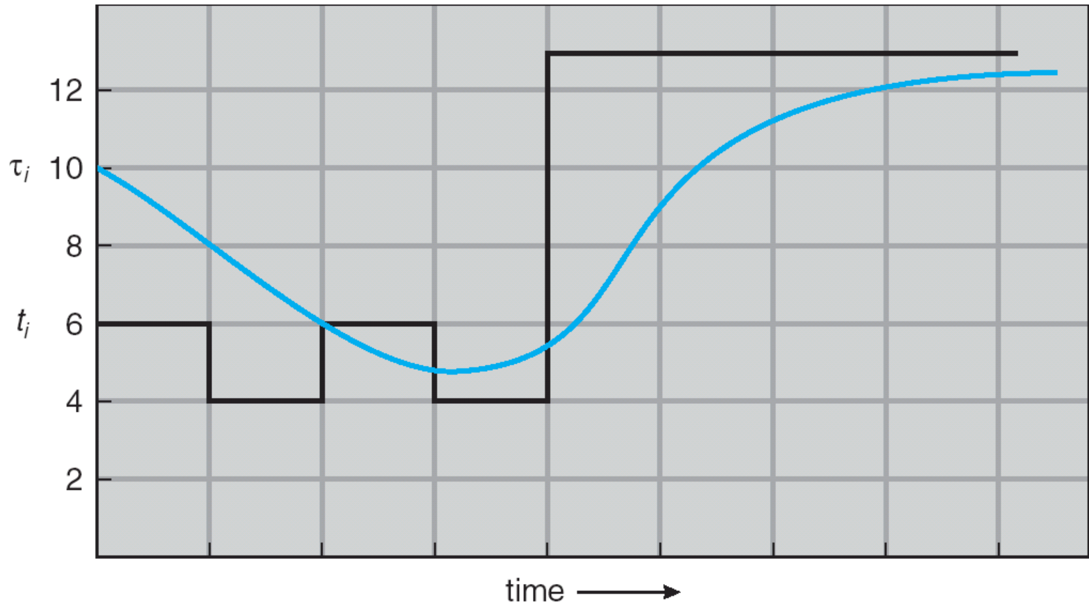
SJF: more

- Determine the *next* burst length
  - how to predict the future?
    - if history is of any indication …
  - use the last burst length
  - use the average so far
  - use the moving average
  - use the weighted moving average
  - the exponentially weight moving average
    - i.e. $\tau_{n+1} = \alpha t_n + (1 - \alpha)\tau_n .$

Q: storage requirement?

EWMA: example

- Exponentially weighted moving average
  - tau_0=10
  - alpha = 0.5
    - normally (0,1)



CPU burst (t_i)		6	4	6	4	13	13	13	…
"guess" (τ_i)	10	8	6	6	5	9	11	12	…

Q: when \alpha=0 or 1?

Priority scheduling

- Priority
  - the job with the highest priority scheduled first
    - SJF: shorter CPU burst, higher priority
    - FCFS: arrival earlier, higher priority
  - static priority: starvation
    - e.g., SJF
  - dynamic priority
    - e.g., aging
- Non-preemptive vs preemptive

Round-robin scheduling

- Discrete processor sharing
  - CPU time quantum
    - usually 10~100 ms
  - for a process
    - either yield after a CPU burst
    - or be preempted after using up a time quantum
  - a FIFO queue
    - all ready processes
- Weighted round-robin

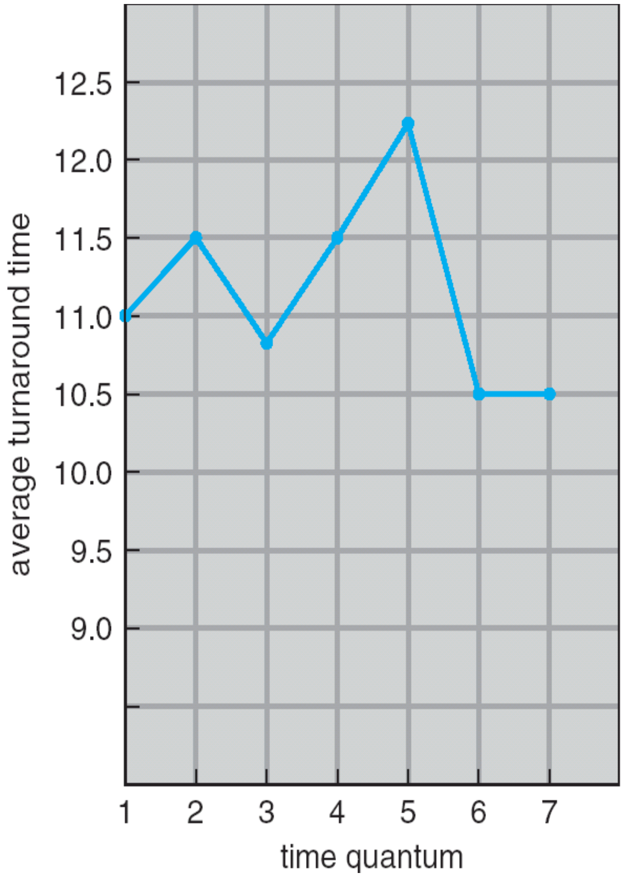
RR: example

- Example
  - P1: 0 (arrival time); 7 (burst time)
  - P2: 2; 4
  - P3: 4; 1
  - P4: 5; 4
- Time quantum
  - e.g., 1 quantum = 1 time unit
  - how about 1 quantum = 4 time units

Q: turnaround time?

Time quantum

- Large quantum
  - => FCFS
- Small quantum
  - better responsiveness
  - be aware of overhead
    - context switching
  - "80%" rule



process	time
P_1	6
P_2	3
P_3	1
P_4	7

This lecture

- Scheduling algorithms
  - FCFS
  - SJF, priority, RR
  - preemptive and non-preemptive
- Explore further
  - evaluate average waiting time, average turnaround time per unit job for these algorithms
  - Q's on slides and in the textbook

Next lecture

- Next lecture
  - more on scheduling