CMPT 300 — Operating Systems I

Summer 1999

Segment 5:

Processor Scheduling

Melissa O'Neill

Basic Concepts of CPU Scheduling

- Maximum CPU utilization is obtained with multiprogramming
- CPU–I/O Burst Cycle – Process execution consists of a cycle of:
  - CPU execution
  - I/O wait

CPU Scheduler (aka short-term scheduler)

- Selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them.
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state (e.g., I/O wait)
  2. Switches from running to ready state (e.g., interrupt)
  3. Switches from waiting to ready (e.g., I/O completes)
  4. Terminates
- Scheduling under 1 and 4 is nonpreemptive.
- All other scheduling is preemptive.

Class Exercise: What criteria should we use to evaluate different scheduling algorithms?

Scheduling Criteria — User Oriented

Performance-related criteria

- Response time
- Turnaround time
- Deadlines satisfied

Other criteria

- Predictability

Scheduling Criteria — System Oriented

Performance-related criteria

- Throughput
- Processor utilization

Other criteria

- Fairness
- Respect for priorities
- Resource balancing

Optimization Criteria

Maximize

- Throughput
- Processor utilization

Minimize

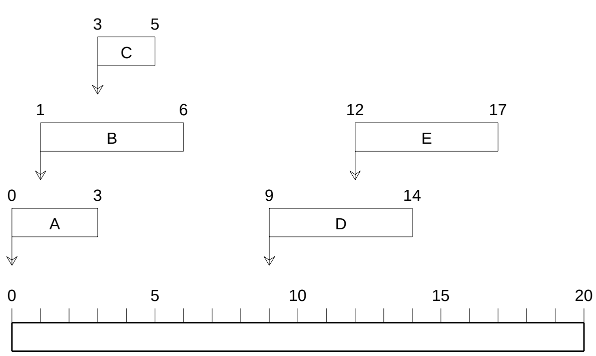
- Turnaround time
- Response time

A Running Example…

Five process, A, B, C, D, E

Process	Burst	Arrival
A	3	0
B	5	1
C	2	3
D	5	9
E	5	12

(B is service time, s, in Stallings)



A Running Example (contd.)

We will examine the following measures for scheduling algorithms:

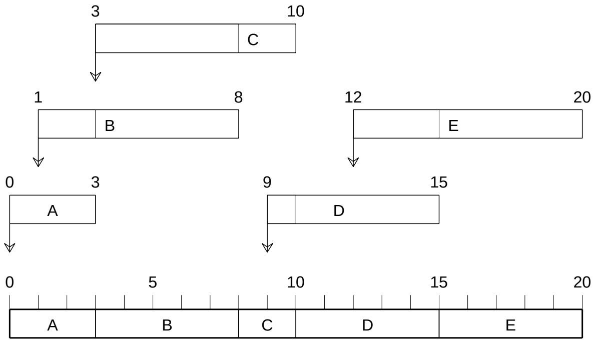
- **S** — start time
- **F** — finish time
- **T** — response time (F – A)
- **M** — missed time (T – B) (w in Stallings)
- **P** — penalty ratio (T / B) (RR in Stallings)

For each scheduling algorithm we will build a table like this:

Process	B	A	S	F	T	M	P
A	3	0					
B	5	1					
C	2	3					
D	5	9					
E	5	12					
Mean							

First-Come, First-Served (FCFS) Scheduling

- Run processes in the order they arrive in the ready queue
- No preemption



First-Come, First-Served (FCFS) Scheduling (contd.)

We find:

Process	B	A	S	F	T	M	P
A	3	0	0	3	3	0	1.0
B	5	1	3	8	7	2	1.4
C	2	3	8	10	7	5	3.5
D	5	9	10	15	6	1	1.2
E	5	12	15	20	8	3	1.6
Mean					6.2	2.2	1.74

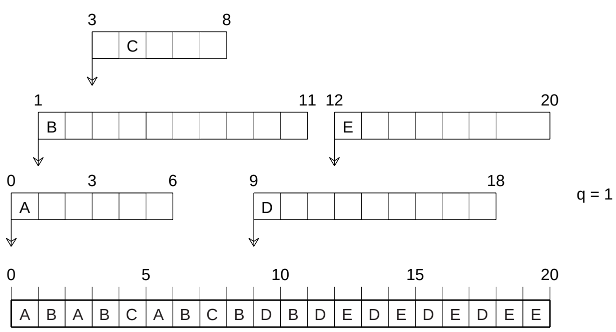
But, consider:

Process	B	A	S	F	T	M	P
W	1	0	0	1	1	0	1.00
X	100	0	1	101	101	1	1.01
Y	1	0	101	102	102	101	102.00
Z	100	0	102	202	202	102	2.02
Mean					101.5	51	28.10

Round Robin (RR) Scheduling

- Divide time between processes on ready queue
- Each process gets one quantum of time before moving on to next process

Quantum = 1:



Round Robin (RR) Scheduling (contd)

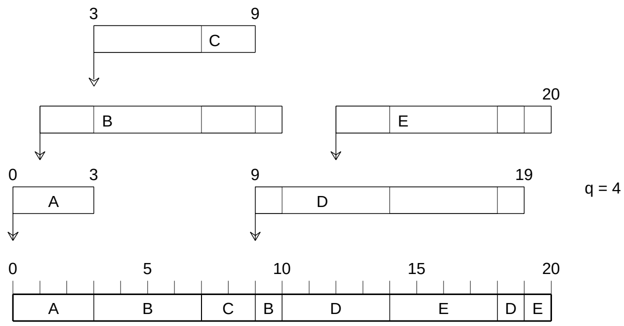
We find:

Process	B	A	S	F	T	M	P
A	3	0	0	6	6	3	2.0
B	5	1	1	11	10	5	2.0
C	2	3	4	8	5	3	2.5
D	5	9	9	18	9	4	1.8
E	5	12	12	20	8	3	1.6
Mean					7.6	3.6	1.98

Round Robin (RR) Scheduling

- Very small quantum approximate a *processor sharing* policy

Quantum = 4:



Round Robin (RR) Scheduling (contd)

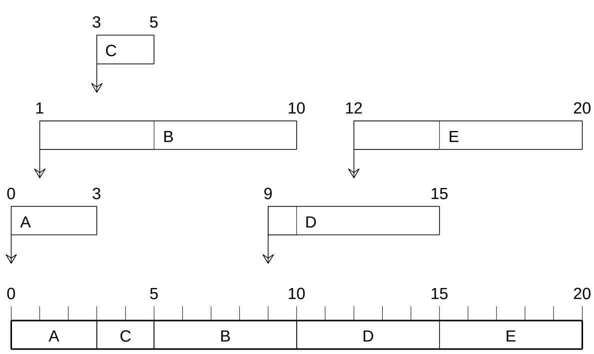
We find:

Process	B	A	S	F	T	M	P
A	3	0	0	3	3	0	1
B	5	1	3	10	9	4	1.8
C	2	3	7	9	6	4	3.0
D	5	9	10	19	10	5	2.0
E	5	12	14	20	8	3	1. 6
Mean					7.2	3.2	1.88

- Very large quantum approximate a *FCFS* policy

Shortest Burst Next (SPN)

- Try to have advantages of RR, without cost of preemption
- Run the process that will have the shortest burst of CPU next.



Shortest Burst Next (SPN) (contd.)

We find:

Process	B	A	S	F	T	M	P
A	3	0	0	3	3	0	1.0
B	5	1	5	10	9	4	1.8
C	2	3	3	5	2	0	1.0
D	5	9	10	15	6	1	1.2
E	5	12	15	20	8	3	1.6
Mean					5.6	1.6	1.32

Better than FCFS, but:

- Long bursts are still a problem, due to lack of preemption

Preemptive Shortest Burst Next (PSPN (aka SRT))

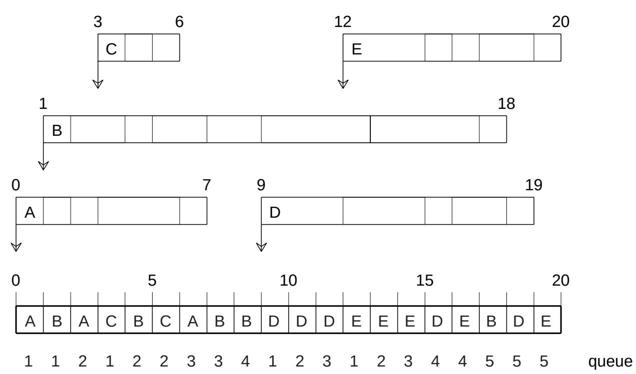
- Pre-empt when a shorter-burst process becomes ready
- Achieves best achievable penalty ratio

Highest Penalty Ratio Next (HPRN (aka HRRN))

- *Choose based on penalty ratio, rather than burst time.*
- Avoids starving long bursts, but performance appears worse than SPN.

Mutlilevel Feedback (FB)

- Penalize jobs that have been running a long time.
- Multiple ready queues, take from the topmost queue that has items in it (thus queues have priority)
- When a process has spent "too long" on one queue, it is bumped down to the next lowest queue



Mutlilevel Feeback (FB)

We find:

Process	B	A	S	F	T	M	P
A	3	0	0	7	7	4	2.3
B	5	1	1	18	17	12	3.4
C	2	3	3	6	3	1	1.5
D	5	9	9	19	10	5	2.0
E	5	12	12	20	8	3	1.6
Mean					9	5	2.16

Many variations:

- *Give lower priority ready queues larger quanta.*
- *Different criteria for what constitutes "too long" in a queue*
- *Move "starving" processes in low priority queues back up into higher priority queues*
- *Share CPU time between queues according to some allocation strategy, rather than always preferring the top queue*

Segment Review

You should be able to:

- Draw *Extended Gantt Charts* to describe scheduling algorithms
- Describe and contrast a variety of scheduling algorithms, including FCFS, RR, SPN, SRT, FB