SCHEDULING IN INTERACTIVE SYSTEMS

Module 2.6
COP4600 – Operating Systems
Richard Newman

SCHEDULING IN INTERACTIVE SYSTEMS

- Round-Robin Scheduling
- Priority Scheduling
- Multiple Queues
- Shortest Process Next
- CTSS
- Guaranteed Scheduling
- Lottery Scheduling
- Fair-Share Scheduling

ROUND-ROBIN SCHEDULING

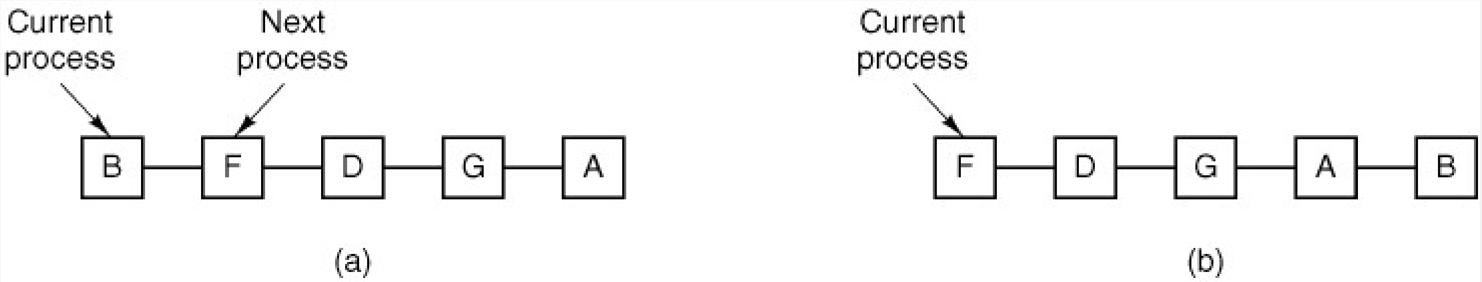


Figure 2-42. Round-robin scheduling.

(a) The list of runnable processes.

(b) The list of runnable processes after B uses up its quantum.

For our problems in scheduling, if process C arrives at the same time that process D uses up its quantum, C will be added to the ready queue before D is added to it.

Tanenbaum & Bos, Modern Operating Systems:4th ed., (c) 2013 Prentice-Hall, Inc. All rights reserved.

ROUND-ROBIN SCHEDULING

Process Name:	A	B	C	D	E
Arrival Time (AT):	0	2	3	5	7
CPU Burst Length (CT):	8	5	1	2	2

Arrivals-> A B C D E

Quantum = 1: A A B A C B A D B E A D B E A B A A

A C B A D B E A D B E A B A

B A D B E A D B E A B

A D B E A

Turnaround times: A:18-0, B:16-2, C:5-3, D:12-5, E: 14-7

Average TT = (18+14+2+7+7)/5 = 48/5 = 9.6

Quantum = 2: A A B B A A C B B D D A A E E B A A

A A C C B D D A A E E B B A

C B B D A A E E B B A A

D A E E B B

(PREEMPTIVE) PRIORITY SCHEDULING

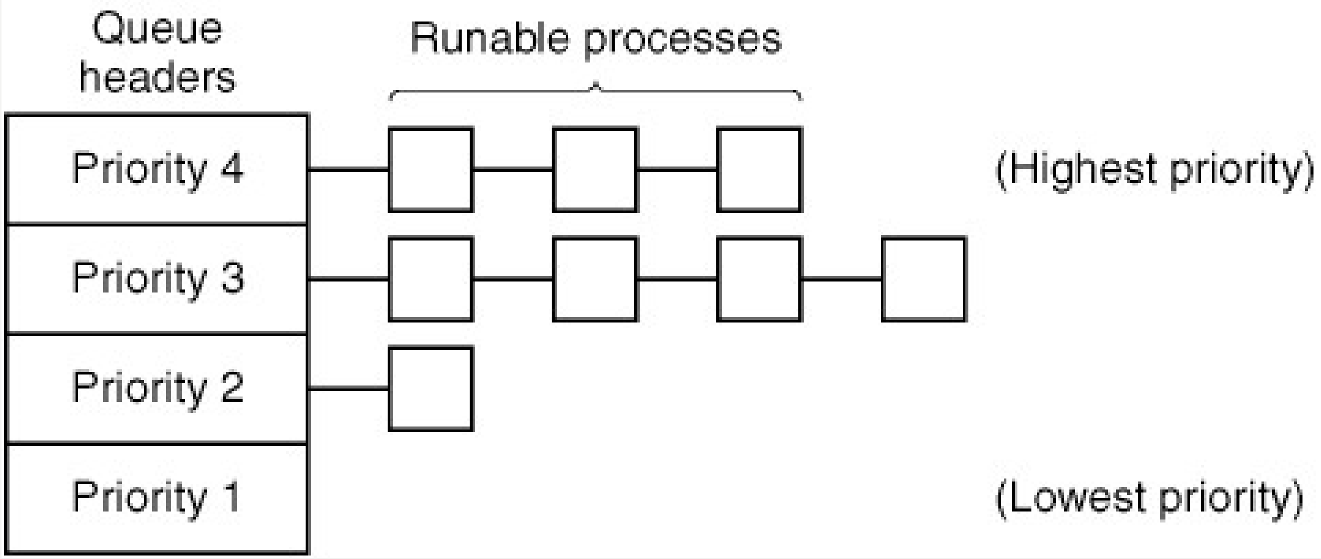


Figure 2-43. A scheduling algorithm with four priority classes.

Run a process at priority i+1 over any process at priority i

Run processes at priority i in round robin (or whatever) order

Preempt process if higher priority process arrives

Tanenbaum & Bos, Modern Operating Systems:4th ed., (c) 2013 Prentice-Hall, Inc. All rights reserved.

PRIORITY SCHEDULING

Process Name:	A	B	C	D	E
Priority:	4	2	3	1	5
Arrival Time (AT):	0	2	3	5	7
CPU Burst Length (CT):	8	5	1	2	2

Priority Schedule:↓A A↓B↓ B B↓D D↓↑ B B↑C↑A A A A A A↑E E↑

Turnaround times: A:16-0, B:9-2, C:10-3, D:7-5, E: 18-7

Average TT = (16+7+7+2+11)/5 = 43/5 = 8.6

Priority is usually preemptive

- Scheduler is run whenever a process arrives
- Scheduler runs whenever a process unblocks

Priority is most general policy

It can model any other policy by priority definition

CTSS SCHEDULING

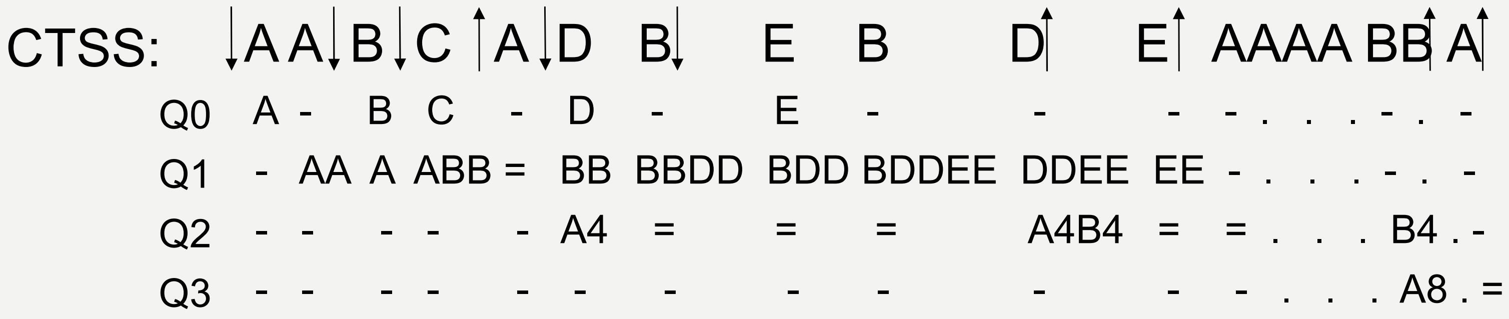
CTSS used by Multics OS – try to approximate SRTF

Multi-level feedback queue –

Enter at queue 0, drop to next queue if TRO

Queue i gets 2^i for time quantum, i = 0, 1, 2, ...

Process Name:	A	B	C	D	E
Arrival Time (AT):	0	2	3	5	7
CPU Burst Length (CT):	8	5	1	2	2



CTSS: A A B C A D B E B D E AAAA BB A

Q0 A - B C - D - E - - - - . . . - . -

Q1 - AA A ABB = BB BBDD BDD BDDEE DDEE EE - . . . - . -

Q2 - - - - - A4 = = = A4B4 = = . . . B4 . -

Q3 - - - - - - - - - - - - . . . A8 . =

Turnaround times: A:18-0, B:17-2, C:4-3, D:10-5, E: 11-7

Average TT = (18+15+1+5+4)/5 = 43/5 = 8.6

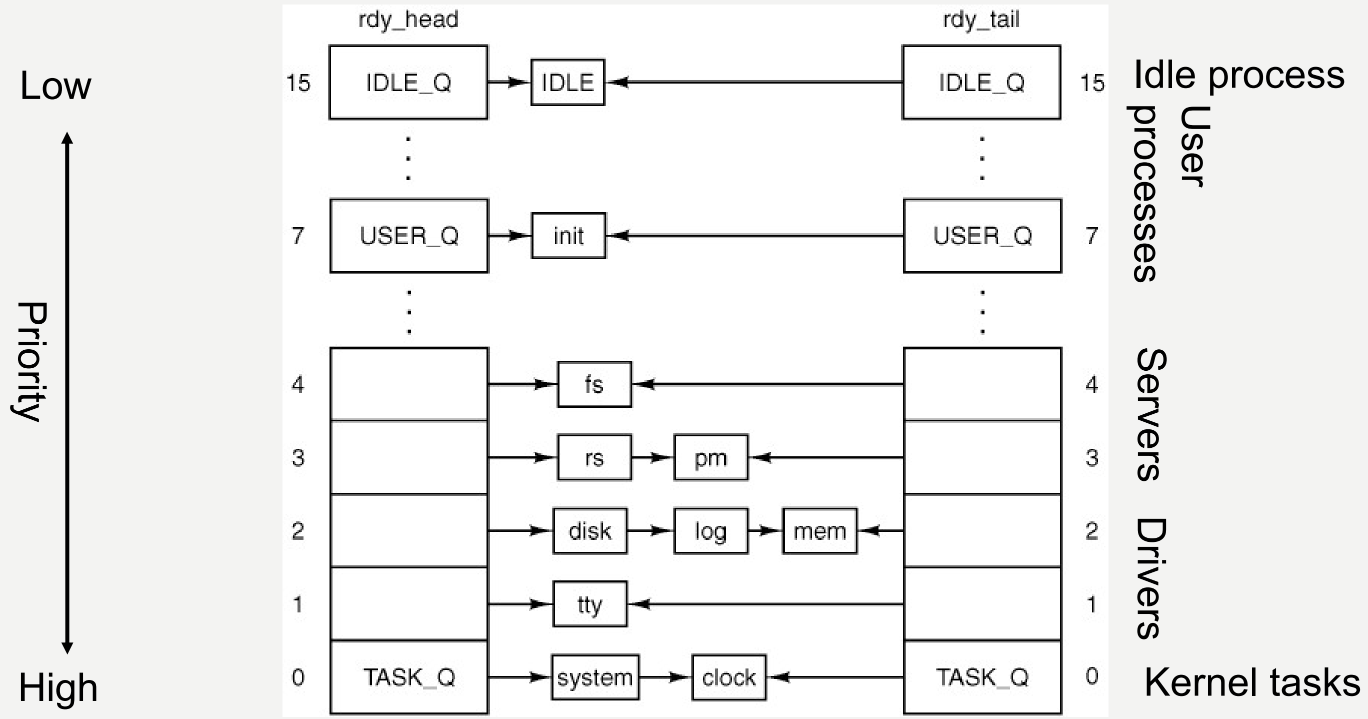
SHORTEST PROCESS NEXT

- Same as SRTF
- Main problem: how to know how much time remains?
- Practically estimate time per CPU burst:
  - Based on observed behavior
  - Count recent behavior more
  - Smooth by “aging”
  - E_0 = initial guess, T_i = i^{th} observed time
  - i^{th} estimate $E_i = aE_{i-1} + (1-a)T_i$
  - Especially easy to compute if a = ½
    - Add new time to current estimate and shift right one bit

PROPORTIONATE SCHEDULING

- Guaranteed Scheduling
  - Use actual vs. guarantee to prioritize
  - E.g., fair share fraction of CPU
- Lottery Scheduling
  - Grant each process some number of tickets
  - Periodically pick a ticket at random
  - Odds of winning a quantum proportional to number of tickets held
- Fair-Share Scheduling
  - Allocate fraction of CPU to user, not process
  - Can use lottery approach
  - Can prioritize users by number of tickets

SCHEDULING IN MINIX



Low

Priority

High

Idle process

User processes

Servers

Drivers

Kernel tasks

Figure 2-43. The scheduler maintains sixteen queues, one per priority level. Shown here is the initial queuing process as MINIX 3 starts up.

Tanenbaum & Bos, Modern Operating Systems:4th ed., (c) 2013 Prentice-Hall, Inc. All rights reserved.

RESTART

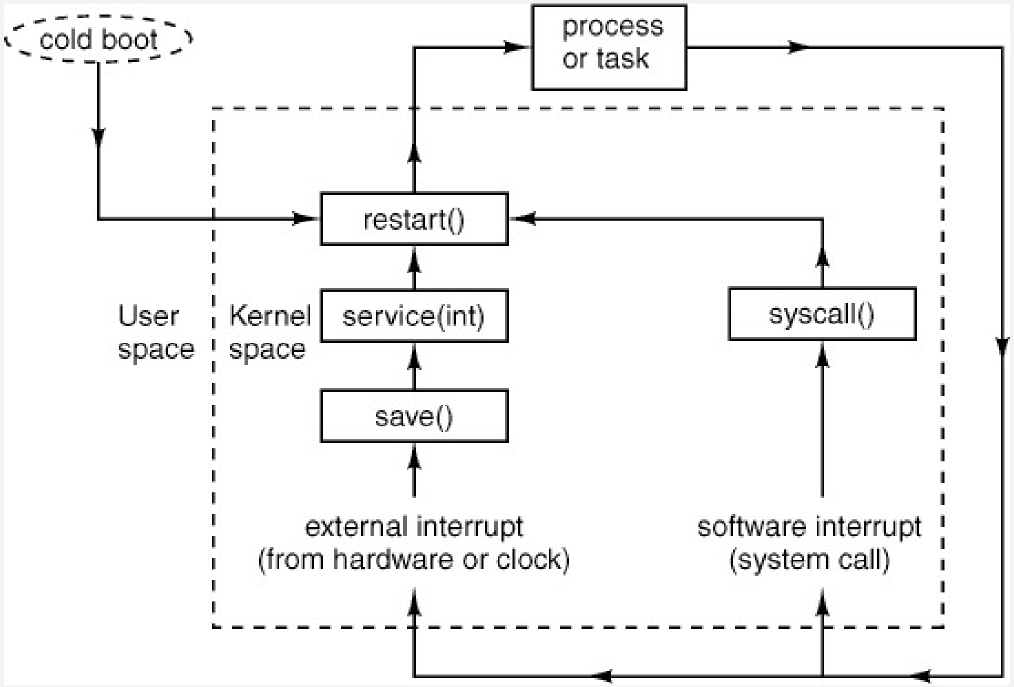


Figure 2-41. Restart is the common point reached after system startup, interrupts, or system calls. The most deserving process (which may be and often is a different process from the last one interrupted) runs next. Not shown in this diagram are interrupts that occur while the kernel itself is running.

Tanenbaum & Bos, Modern Operating Systems:4th ed., (c) 2013 Prentice-Hall, Inc. All rights reserved.

SUMMARY

- Round-Robin
- Priority
- Multiple Queues
- CTSS
- Shortest Process Next
  - Estimating Compute Burst Times
- Proportionate Scheduling
- Scheduling in Minix3