

*Operating
Systems:
Internals
and
Design
Principles*

Chapter 9 Uniprocessor Scheduling

Seventh Edition
By William Stallings

Processor Scheduling

- Aim is to assign processes to be executed by the processor in a way that meets system objectives, such as response time, throughput, and processor efficiency
- Broken down into three separate functions:



Table 9.1

Types of Scheduling

Long-term scheduling	The decision to add to the pool of processes to be executed
Medium-term scheduling	The decision to add to the number of processes that are partially or fully in main memory
Short-term scheduling	The decision as to which available process will be executed by the processor
I/O scheduling	The decision as to which process's pending I/O request shall be handled by an available I/O device

Scheduling and Process State Transitions

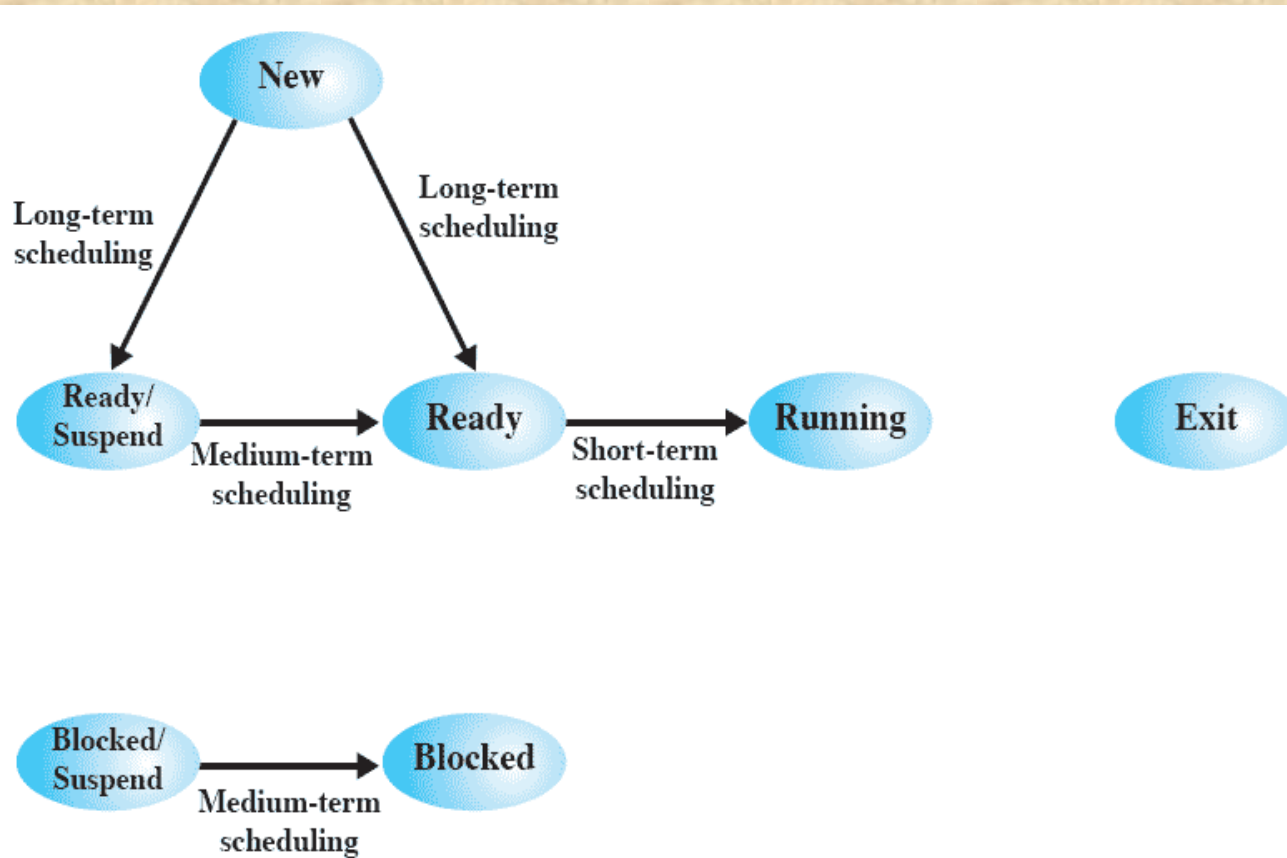
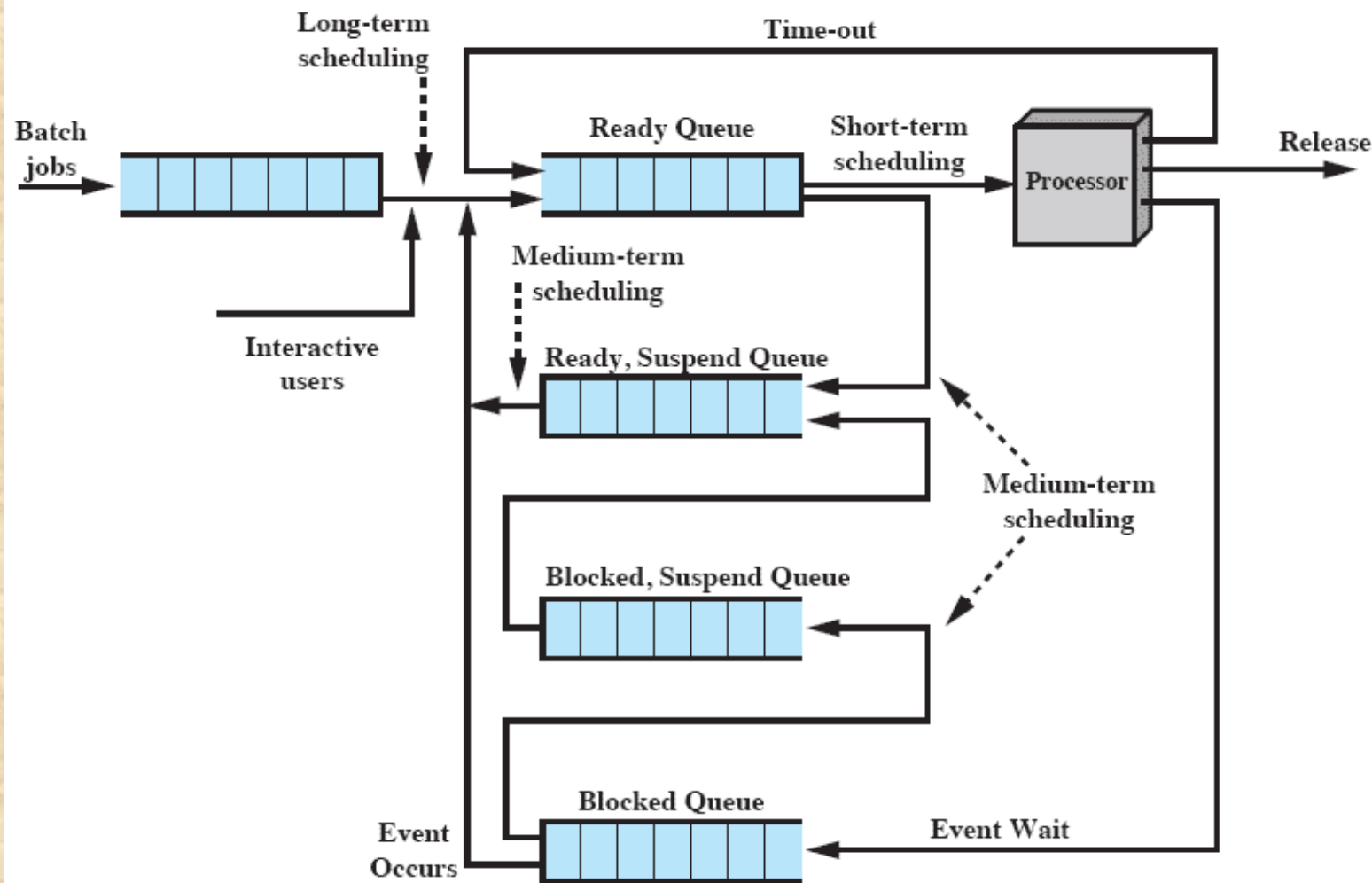


Figure 9.1 Scheduling and Process State Transitions

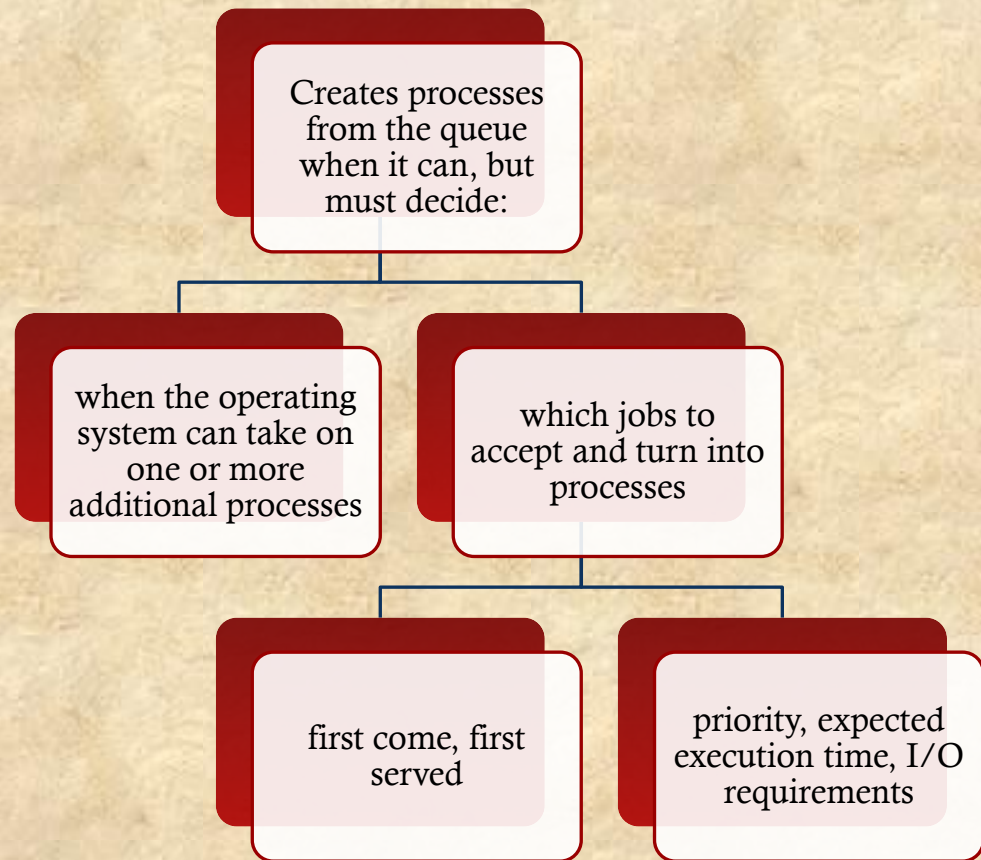


Q D
u i
e a
u g
i r
n a
g m

Figure 9.3 Queuing Diagram for Scheduling

Long-Term Scheduler

- Determines which programs are admitted to the system for processing
- Controls the degree of multiprogramming
 - the more processes that are created, the smaller the percentage of time that each process can be executed
 - may limit to provide satisfactory service to the current set of processes



Medium-Term Scheduling

- Part of the swapping function
- Swapping-in decisions are based on the need to manage the degree of multiprogramming
 - considers the memory requirements of the swapped-out processes

Short-Term Scheduling

- Known as the dispatcher
- Executes most frequently
- Makes the fine-grained decision of which process to execute next
- Invoked when an event occurs that leads to the blocking of the current process or that may provide an opportunity to preempt a currently running process in favor of another

Examples:

- Clock interrupts
- I/O interrupts
- Operating system calls
- Signals (e.g., semaphores)

Short Term Scheduling Criteria

- Main objective is to allocate processor time to optimize certain aspects of system behavior
- A set of criteria is needed to evaluate the scheduling policy

User-oriented criteria

- relate to the behavior of the system as perceived by the individual user or process (such as response time in an interactive system)
- important on virtually all systems

System-oriented criteria

- focus in on effective and efficient utilization of the processor (rate at which processes are completed)
- generally of minor importance on single-user systems

Short-Term Scheduling Criteria: Performance

examples:

- response time
- throughput

Criteria can
be classified
into:

example:

- predictability

Performance-related

Non-performance
related

quantitative

easily
measured

qualitative

hard to
measure

Table 9.2

Scheduling Criteria



User Oriented, Performance Related

Turnaround time This is the interval of time between the submission of a process and its completion. Includes actual execution time plus time spent waiting for resources, including the processor. This is an appropriate measure for a batch job.

Response time For an interactive process, this is the time from the submission of a request until the response begins to be received. Often a process can begin producing some output to the user while continuing to process the request. Thus, this is a better measure than turnaround time from the user's point of view. The scheduling discipline should attempt to achieve low response time and to maximize the number of interactive users receiving acceptable response time.

Deadlines When process completion deadlines can be specified, the scheduling discipline should subordinate other goals to that of maximizing the percentage of deadlines met.

User Oriented, Other

Predictability A given job should run in about the same amount of time and at about the same cost regardless of the load on the system. A wide variation in response time or turnaround time is distracting to users. It may signal a wide swing in system workloads or the need for system tuning to cure instabilities.

System Oriented, Performance Related

Throughput The scheduling policy should attempt to maximize the number of processes completed per unit of time. This is a measure of how much work is being performed. This clearly depends on the average length of a process but is also influenced by the scheduling policy, which may affect utilization.

Processor utilization This is the percentage of time that the processor is busy. For an expensive shared system, this is a significant criterion. In single-user systems and in some other systems, such as real-time systems, this criterion is less important than some of the others.

System Oriented, Other

Fairness In the absence of guidance from the user or other system-supplied guidance, processes should be treated the same, and no process should suffer starvation.

Enforcing priorities When processes are assigned priorities, the scheduling policy should favor higher-priority processes.

Priority Queuing

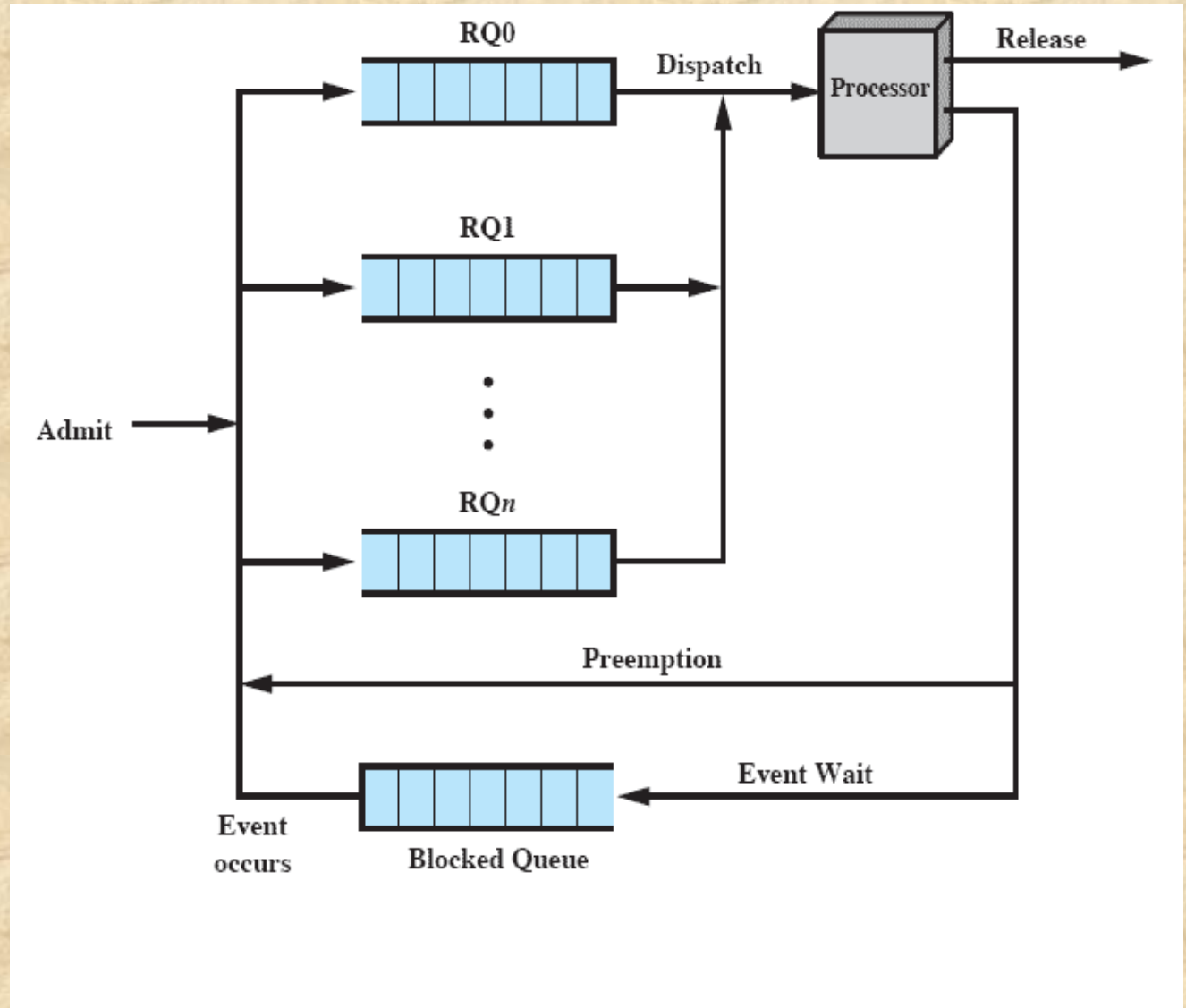


Figure 9.4 Priority Queuing

Selection Function

- Determines which Ready process is dispatched next
- May be based on priority, resource requirements, or the execution characteristics of the process
- If based on execution characteristics, some factors to consider are
 - w = time spent in system so far, waiting
 - e = time spent in execution so far
 - s = total service time required by the process, including e ; (estimated by system or user)

Decision Mode

- When/under what circumstances is the selection function is exercised?
- Two categories:
 - Nonpreemptive
 - Preemptive



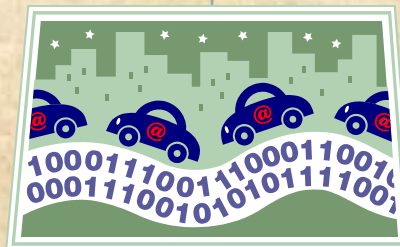
Nonpreemptive vs Preemptive

Nonpreemptive

- once a process is in the running state, it will continue until it terminates or blocks itself for I/O

Preemptive

- currently running process may be interrupted and moved to ready state by the OS
- preemption may occur when a new process arrives, on an interrupt, or periodically



Alternative Scheduling Policies

Table 9.3 Characteristics of Various Scheduling Policies

	FCFS	Round robin	SPN	SRT	HRRN	Feedback
Selection function	max[w]	constant	min[s]	min[s - e]	$\max\left(\frac{w + s}{s}\right)$	(see text)
Decision mode	Non-preemptive	Preemptive (at time quantum)	Non-preemptive	Preemptive (at arrival)	Non-preemptive	Preemptive (at time quantum)
Throughput	Not emphasized	May be low if quantum is too small	High	High	High	Not emphasized
Response time	May be high, especially if there is a large variance in process execution times	Provides good response time for short processes	Provides good response time for short processes	Provides good response time	Provides good response time	Not emphasized
Overhead	Minimum	Minimum	Can be high	Can be high	Can be high	Can be high
Effect on processes	Penalizes short processes; penalizes I/O bound processes	Fair treatment	Penalizes long processes	Penalizes long processes	Good balance	May favor I/O bound processes
Starvation	No	No	Possible	Possible	No	Possible

Table 9.4

Process Scheduling

Example



Process	Arrival Time	Service Time
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2

Performance Comparison

- Any scheduling discipline that chooses the next item to be served independent of service time obeys the relationship:

$$\frac{T_r}{T_s} = \frac{1}{1 - \rho}$$

where

T_r = turnaround time or residence time; total time in system, waiting plus execution

T_s = average service time; average time spent in Running state

ρ = processor utilization

Table 9.5

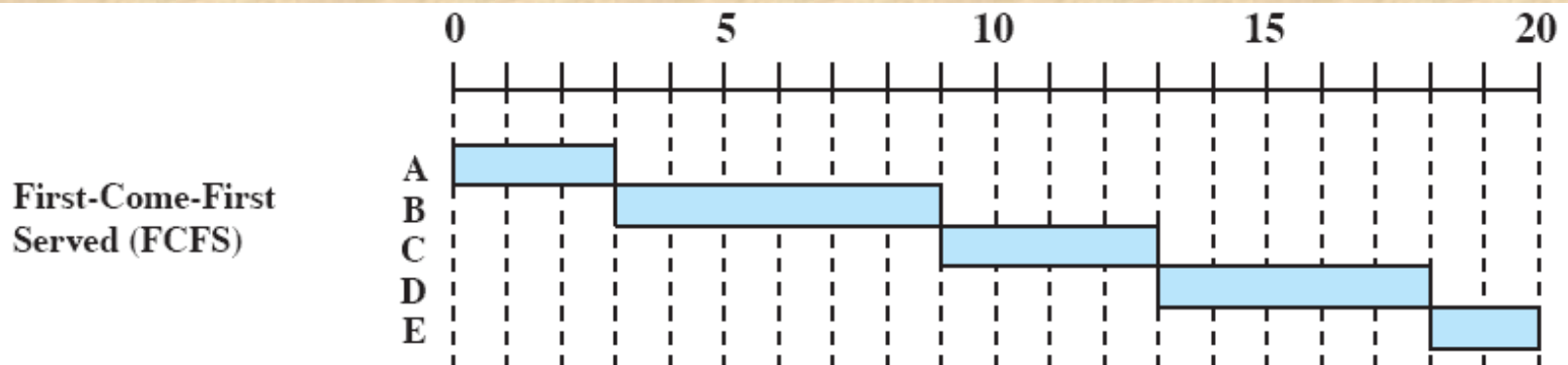
Comparison of Scheduling Policies

(Assumes no process blocks itself, for I/O or other event wait.)

Process	A	B	C	D	E	
Arrival Time	0	2	4	6	8	
Service Time (T_s)	3	6	4	5	2	Mean
FCFS						
Finish Time	3	9	13	18	20	
Turnaround Time (T_r)	3	7	9	12	12	8.60
T_r/T_s	1.00	1.17	2.25	2.40	6.00	2.56
RR $q = 1$						
Finish Time	4	18	17	20	15	
Turnaround Time (T_r)	4	16	13	14	7	10.80
T_r/T_s	1.33	2.67	3.25	2.80	3.50	2.71
RR $q = 4$						
Finish Time	3	17	11	20	19	
Turnaround Time (T_r)	3	15	7	14	11	10.00
T_r/T_s	1.00	2.5	1.75	2.80	5.50	2.71
SPN						
Finish Time	3	9	15	20	11	
Turnaround Time (T_r)	3	7	11	14	3	7.60
T_r/T_s	1.00	1.17	2.75	2.80	1.50	1.84
SRT						
Finish Time	3	15	8	20	10	
Turnaround Time (T_r)	3	13	4	14	2	7.20
T_r/T_s	1.00	2.17	1.00	2.80	1.00	1.59
HRRN						
Finish Time	3	9	13	20	15	
Turnaround Time (T_r)	3	7	9	14	7	8.00
T_r/T_s	1.00	1.17	2.25	2.80	3.5	2.14
FB $q = 1$						
Finish Time	4	20	16	19	11	
Turnaround Time (T_r)	4	18	12	13	3	10.00
T_r/T_s	1.33	3.00	3.00	2.60	1.5	2.29
FB $q = 2i$						
Finish Time	4	17	18	20	14	

First-Come-First-Served (FCFS)

- Simplest scheduling policy
- Also known as first-in-first-out (FIFO) or a strict queuing scheme
- When the current process ceases to execute, the longest process in the Ready queue is selected
- Performs much better for long processes than short ones
- Tends to favor processor-bound processes over I/O-bound processes





Round Robin

- Uses preemption based on a clock
- Also known as **time slicing** because each process is given a slice of time before being preempted
- Principal design issue is the length of the time quantum, or slice, to be used
- Particularly effective in a general-purpose time-sharing system or transaction processing system
- One drawback is its relative treatment of processor-bound and I/O-bound processes

Round-Robin
(RR), $q = 1$

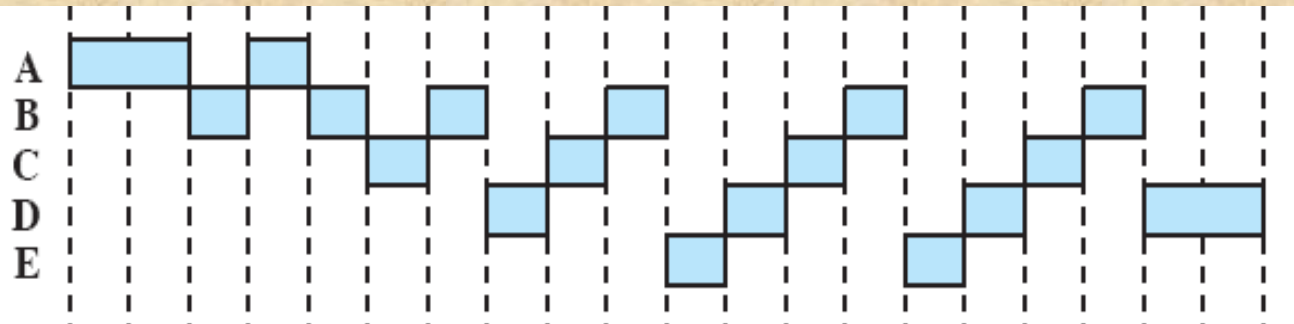
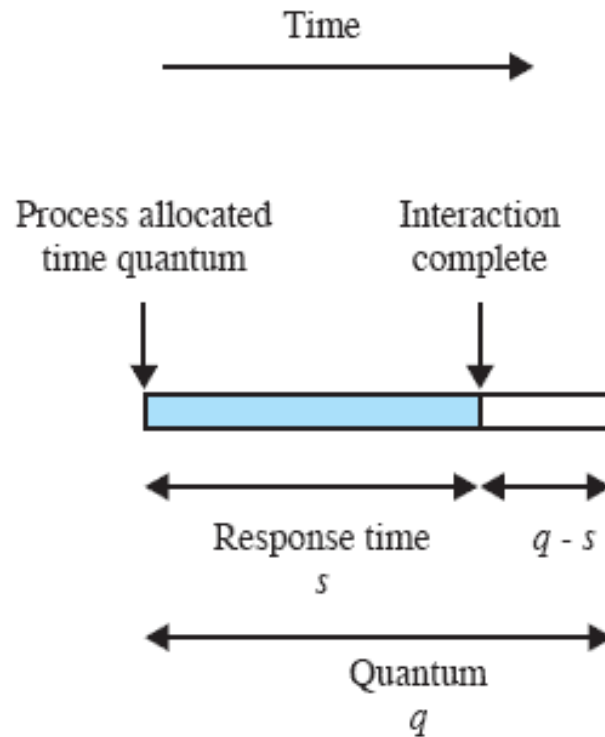


Figure 9.6a

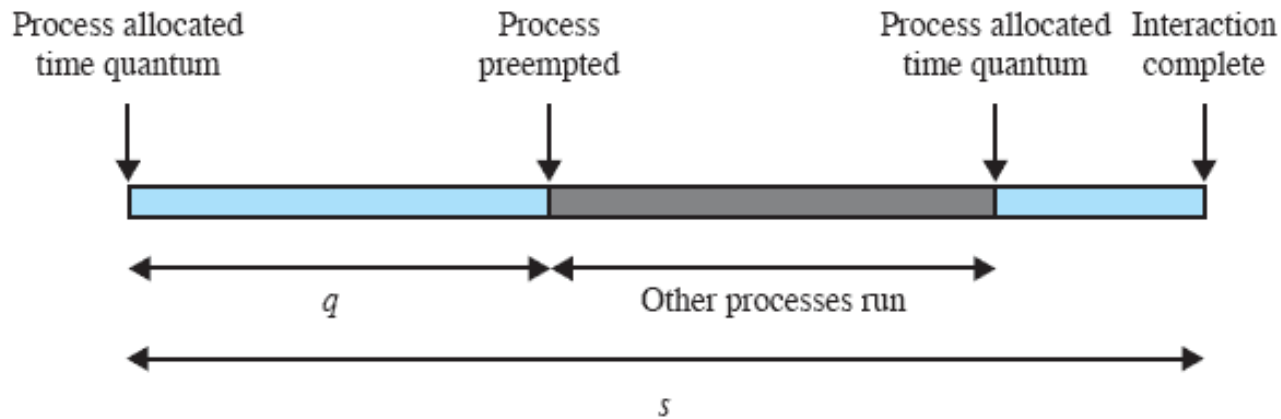
Effect of Size
of
Preemption
Time
Quantum



(a) Time quantum greater than typical interaction

Figure 9.6b

Effect of Size of Preemption Time Quantum



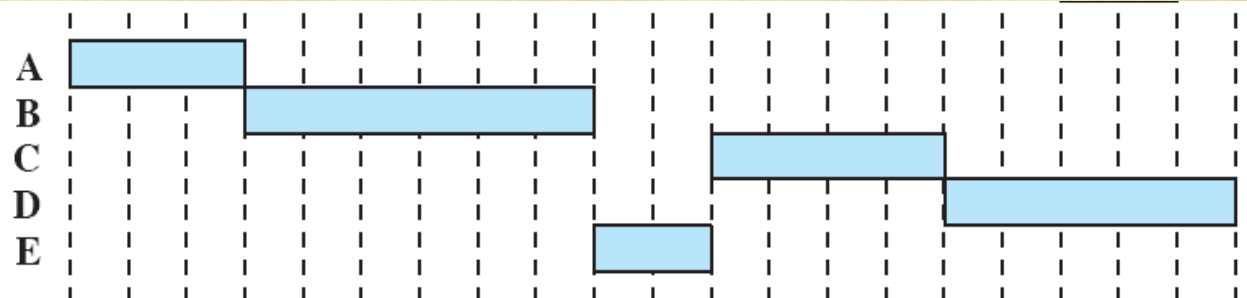
(b) Time quantum less than typical interaction

Figure 9.6 Effect of Size of Preemption Time Quantum

Shortest Process Next (SPN)

- Nonpreemptive policy in which the process with the shortest expected processing time is selected next
- A short process will jump to the head of the queue
- Possibility of starvation for longer processes
- One difficulty is the need to know, or at least estimate, the required processing time of each process
- If the programmer's estimate is substantially under the actual running time, the system may abort the job

Shortest Process
Next (SPN)

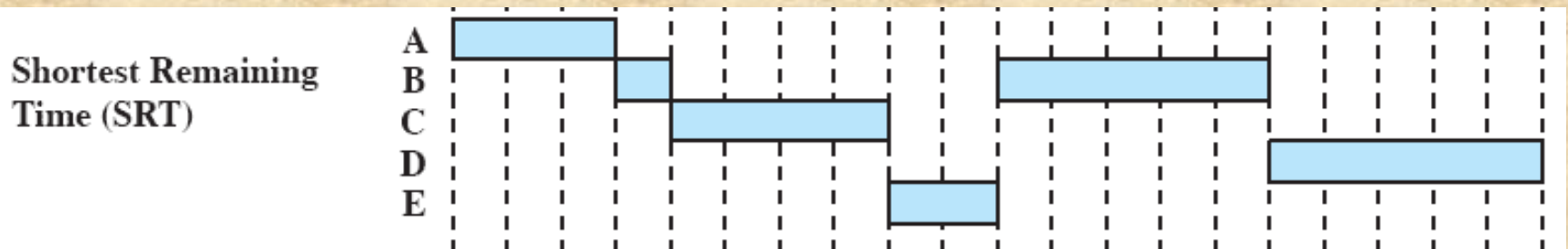


Shortest Process Next (SPN)

- Problem: Estimating execution time
- OS may collect statistics and use process history to estimate run time
 - e.g., for processes in a production environment
- Problem: avoiding starvation for long processes
- Problem: not suitable for timesharing or transaction processing due to no preemption.

Shortest Remaining Time (SRT)

- Preemptive version of SPN
- Scheduler always chooses the process that has the shortest expected remaining processing time
- Risk of starvation of longer processes
- Should give superior turnaround time performance to SPN because a short job is given immediate preference to a running longer job
- Still depends on having accurate service time estimates.

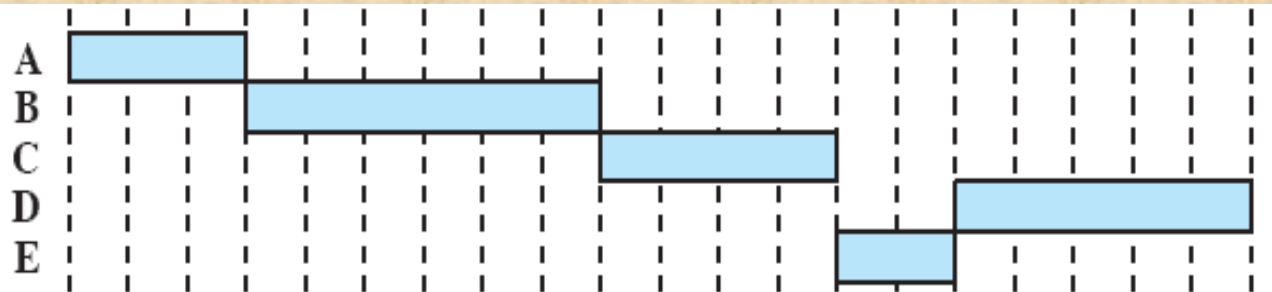


Highest Response Ratio Next (HRRN)

- Chooses next process with the greatest ratio
- Attractive because it accounts for the age of the process
- While shorter jobs are favored, aging without service increases the ratio so that a longer process will eventually get past competing shorter jobs

$$\text{Ratio} = \frac{\text{time spent waiting} + \text{expected service time}}{\text{expected service time}}$$

Highest Response Ratio Next (HRRN)



Fair-Share Scheduling

- Scheduling decisions based on the process sets
- Each user is assigned a share of the processor
- Objective is to monitor usage to give fewer resources to users who have had more than their fair share and more to those who have had less than their fair share



Summary

- The operating system must make three types of scheduling decisions with respect to the execution of processes:
 - Long-term – determines when new processes are admitted to the system
 - Medium-term – part of the swapping function and determines when a program is brought into main memory so that it may be executed
 - Short-term – determines which ready process will be executed next by the processor
- From a user's point of view, response time is generally the most important characteristic of a system; from a system point of view, throughput or processor utilization is important
- Algorithms:
 - FCFS, Round Robin, SPN, SRT, HRRN, Feedback

