



MONASH
University

MONASH
INFORMATION
TECHNOLOGY

FIT2100 Semester 2 2020

Lecture 6:
Uniprocessor Scheduling

(Reading: Stallings, Chapter 9)

WEEK 6



Lecture 6: Learning Outcomes

- Upon the completion of this lecture, you should be able to:
 - Discuss the differences among long-, medium-, and short-term scheduling
 - Assess the performance of different scheduling policies
 - Understand the scheduling technique used in traditional Unix*

*Reading from Stallings, Chapter 9 (9.3)

What is the *aim* of processor scheduling?

Processor Scheduling

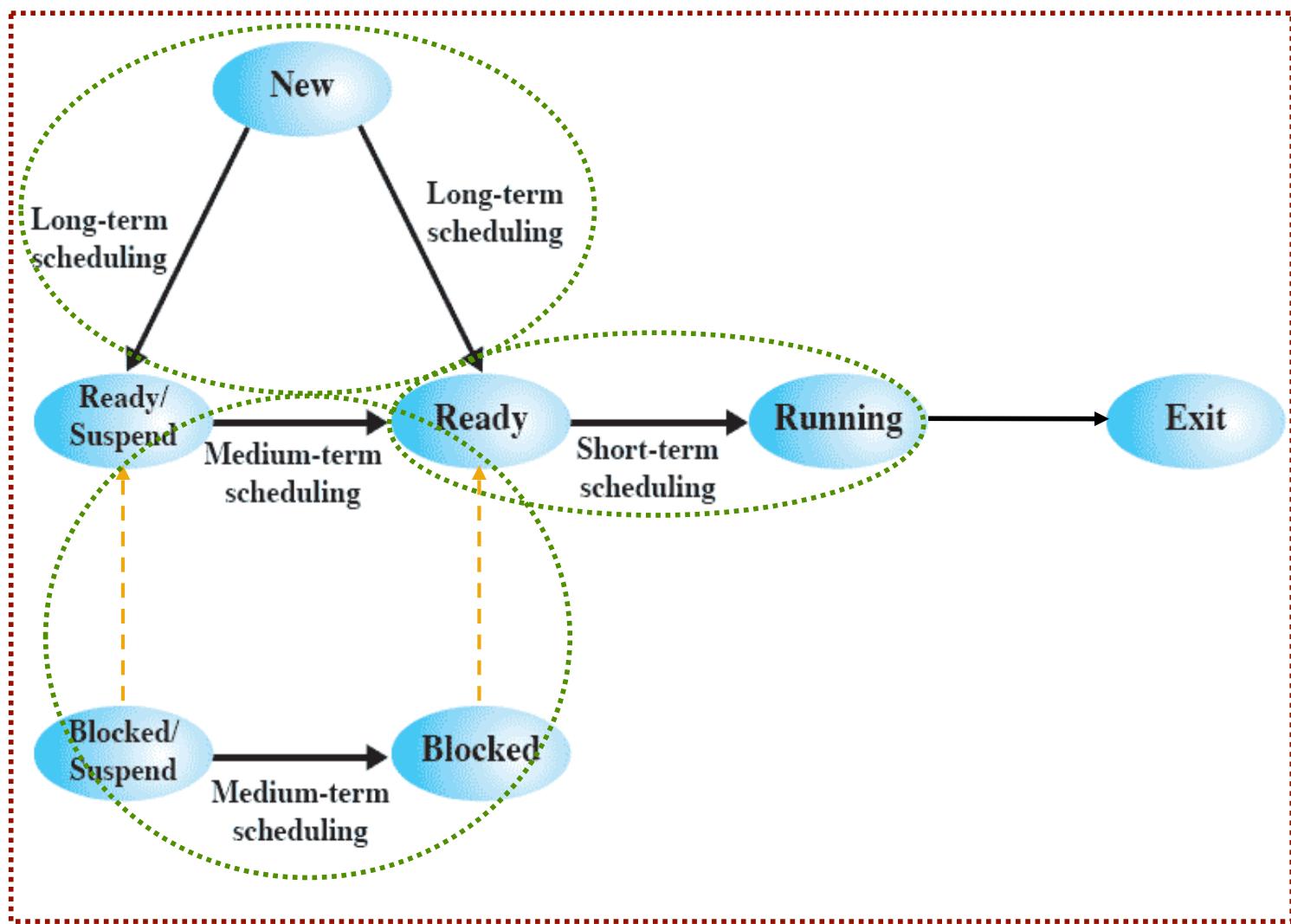
- ❑ To assign processes to be executed by the processor in a way that meets system objectives — such as response time, throughput, and processor efficiency
- ❑ Broken down into three separate functions:



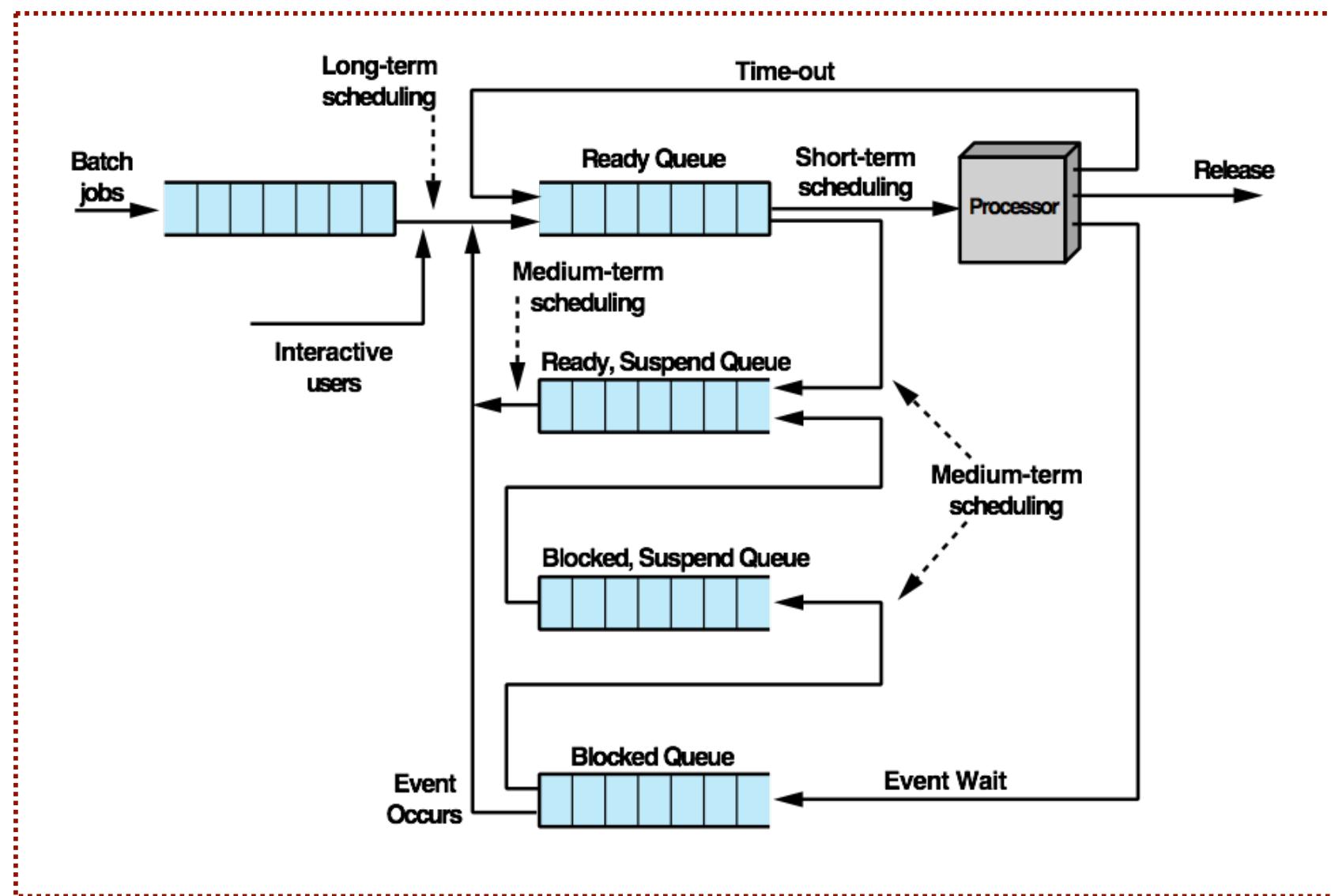
Types of Scheduling

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

Scheduling: Process State Transitions



Scheduling: Queueing Diagram

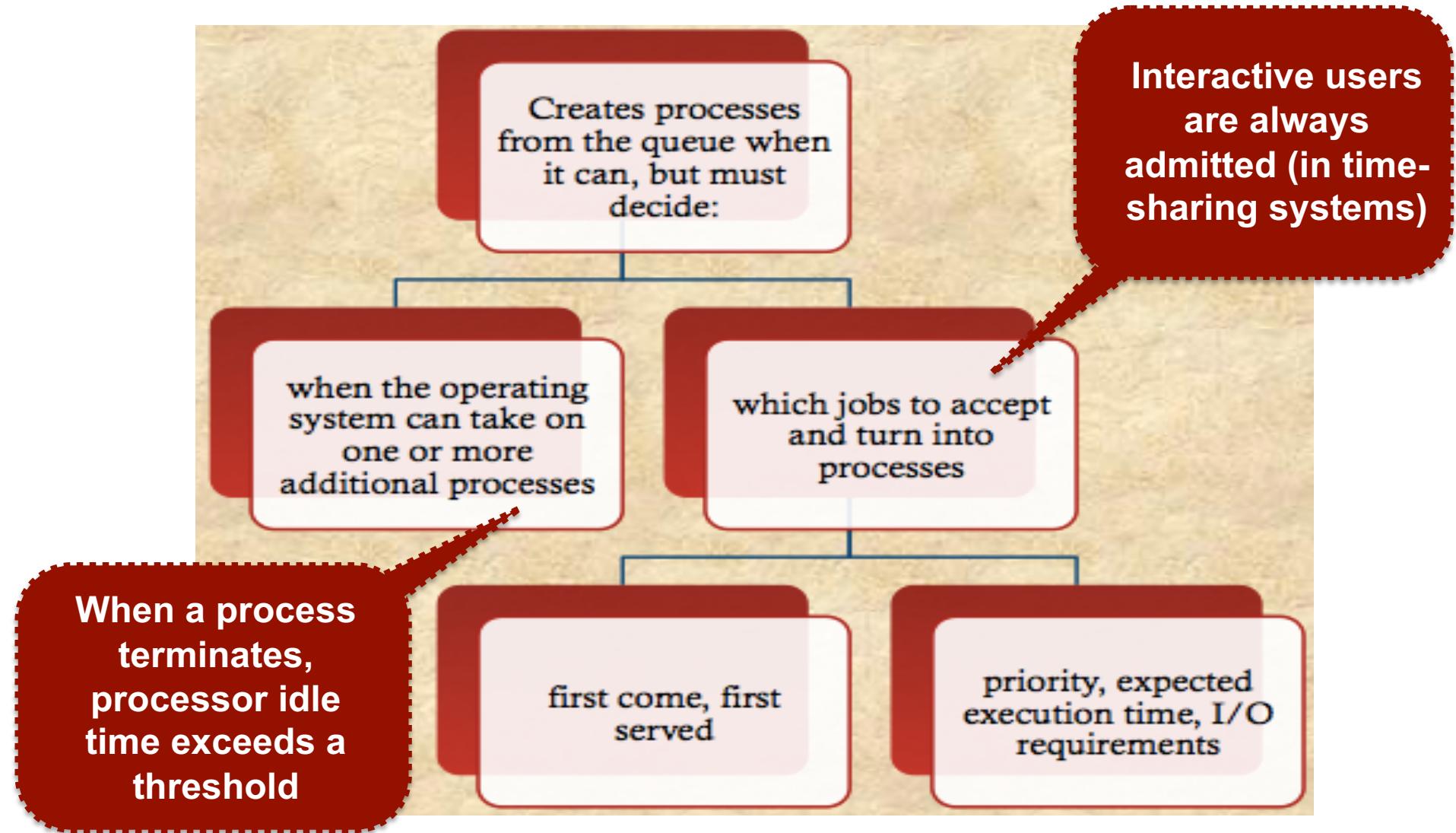


What are the three different scheduling functions?

Long-Term Scheduling

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

Long-Term Scheduling



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



Memory
management
can be an issue

Short-Term Scheduling

- ❑ Known as **dispatcher** or **CPU scheduler**
- ❑ Executes most frequently
- ❑ Makes the fine-grained decision of which process to execute next
- ❑ Invoked whenever an event occurs
 - May lead to *blocking* of the current process
 - May provide an opportunity to *preempt* a currently running process in favour of another

Short-Term Scheduling

- Known as **dispatcher** or **CPU scheduler**
 - Executes most frequently
 - Makes the fine-grained decision of which process to execute next
 - Invoked whenever an event occurs
 - May lead to *blocking* of the current process
 - May provide an opportunity to *preempt* a currently running process in favour of another
- 
- Clock interrupts,
I/O interrupts,
system calls,
signals (e.g. semaphores)

Short-Term Scheduling: Criteria

- ❑ Main objective: allocate processor time to optimise certain aspects of system behaviour
- ❑ A set of **criteria** is needed to evaluate the scheduling policy

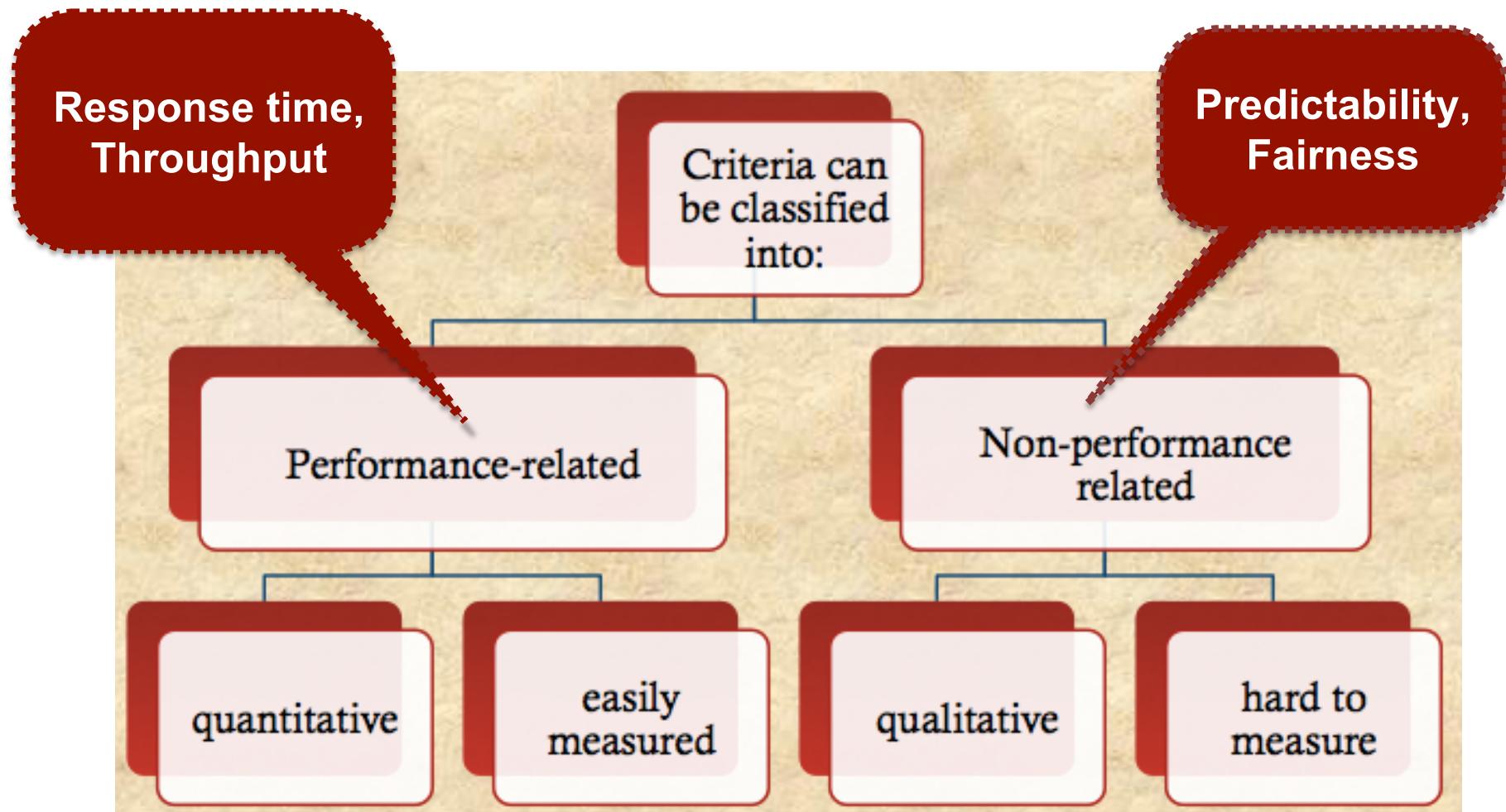
User-oriented Criteria

- Relate to the system behaviour — as perceived by the individual user or process (e.g. response time in an interactive system)
- Important on virtually all systems

System-oriented Criteria

- Focus on effective and efficient utilisation of the processor — rate at which processes are completed (i.e. throughput)
- Generally of minor importance on single-user systems

Short-Term Scheduling Criteria: Performance



Scheduling Criteria: Summary

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.

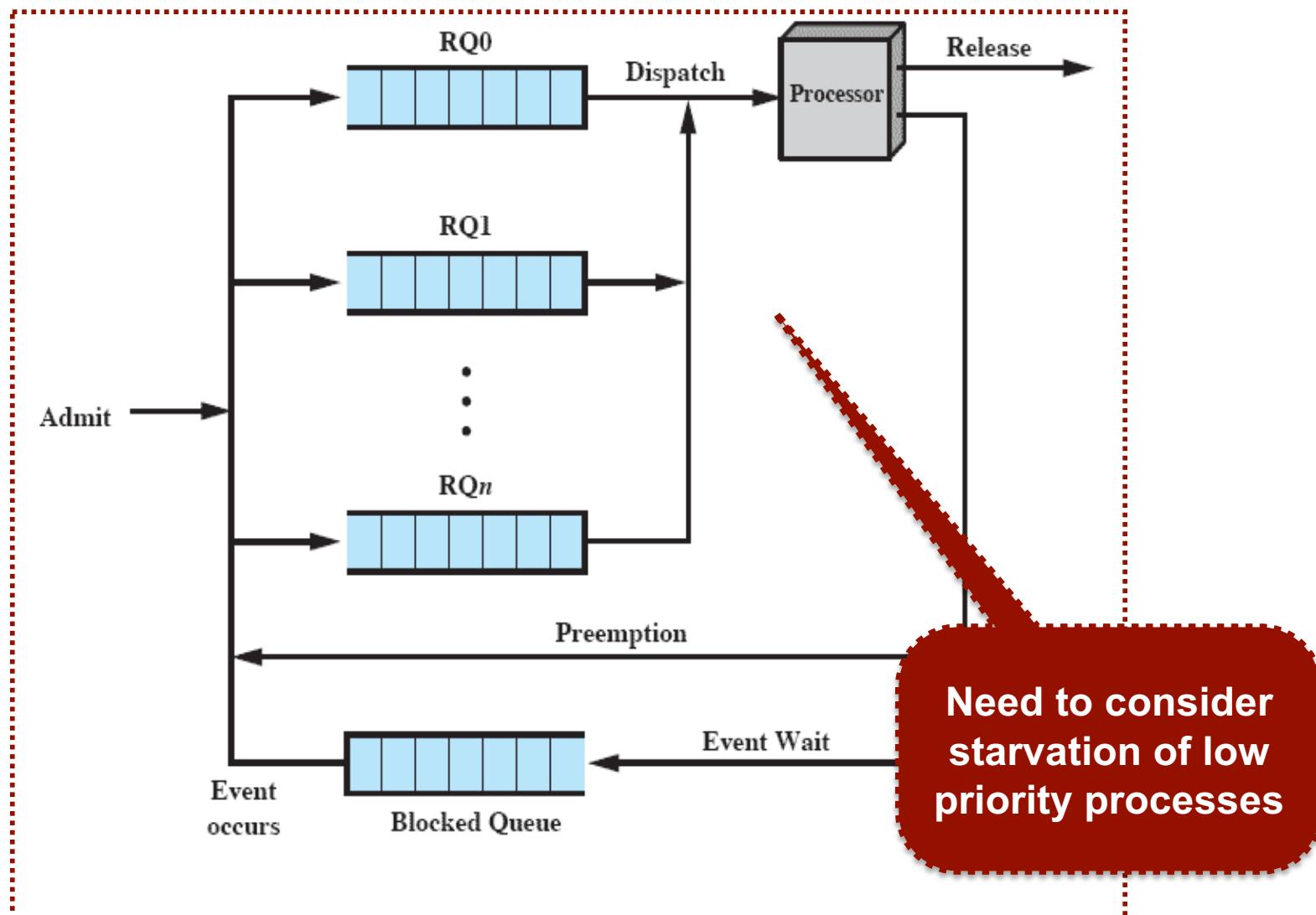
System Oriented, Other

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.

Balancing resources The scheduling policy should keep the resources of the system busy. Processes that will underutilize stressed resources should be favored. This criterion also involves medium-term and long-term scheduling.

Priority Queueing



What are the different scheduling policies?

(Short-Term) Scheduling Policies

- ❑ **FCFS** First Come First Served (**FIFO** First-In First-Out)
- ❑ **RR** Round Robin
- ❑ **SPN** Shortest Process Next (**SJF** Short Job First)
- ❑ **SRT** Shortest Remaining Time
- ❑ Feedback Scheduling

Selection Function

- Determines which process, among ready processes, is selected next for execution
- May be based on priority, resource requirements, or the execution characteristics of the process
- If based on **execution characteristics** — important quantities 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



Must be estimated or supplied by the user

Decision Mode

- Specifies the instants in time at which the **selection function** is exercised
- Two categories
 - **Non-preemptive**
 - **Preemptive**

Non-Preemptive vs Preemptive

Non-Preemptive

- Once a process is in the running state — it will continue until:
 - the process **terminates** or **blocks** itself for I/O and other OS services

Preemptive

- Currently running process may be ***interrupted*** and moved to Ready state by OS
- **Preemption** may occur — when new process arrives, on an interrupt, or periodically (due to time-out)

Process Scheduling: Example

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

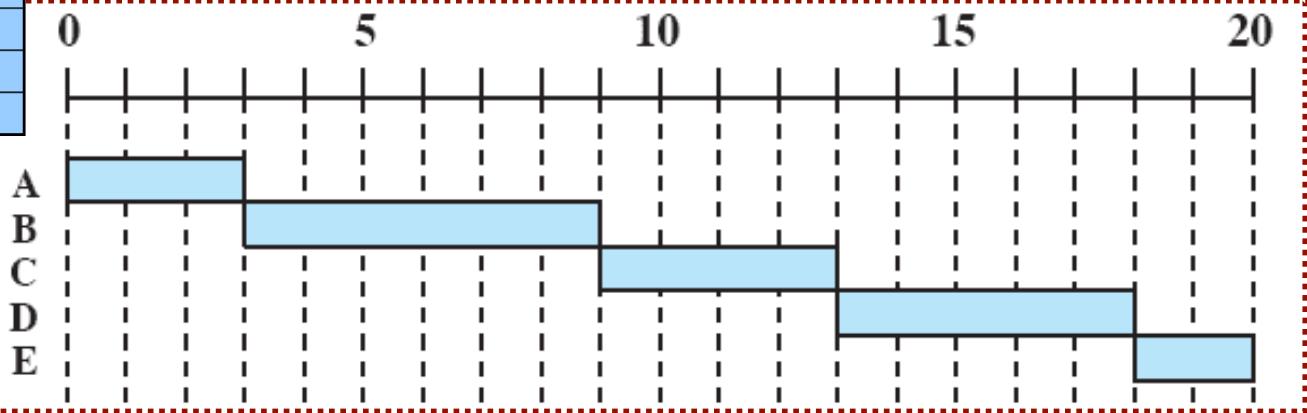
First-Come-First-Served: FCFS

- Simplest scheduling policy
- 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 **favour processor-bound processes over I/O-bound processes**

First-Come-First-Served: FCFS

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

First-Come-First
Served (FCFS)



FCFS						
Finish Time	A = 3	B = 9	C = 13	D = 18	E = 20	Mean
Turnaround Time (Tr) = Completion_time – Arrival_time	A = 3-0=3	B= 9-2=7	C= 13-4=9	D= 18-6=12	E= 20-8=12	8.60
ServiceTime (Ts)	3	6	4	5	2	4
Turnaround Time / ServiceTime = Tr/Ts	1.00	1.17	2.25	2.40	6.00	2.56

Round Robin: RR

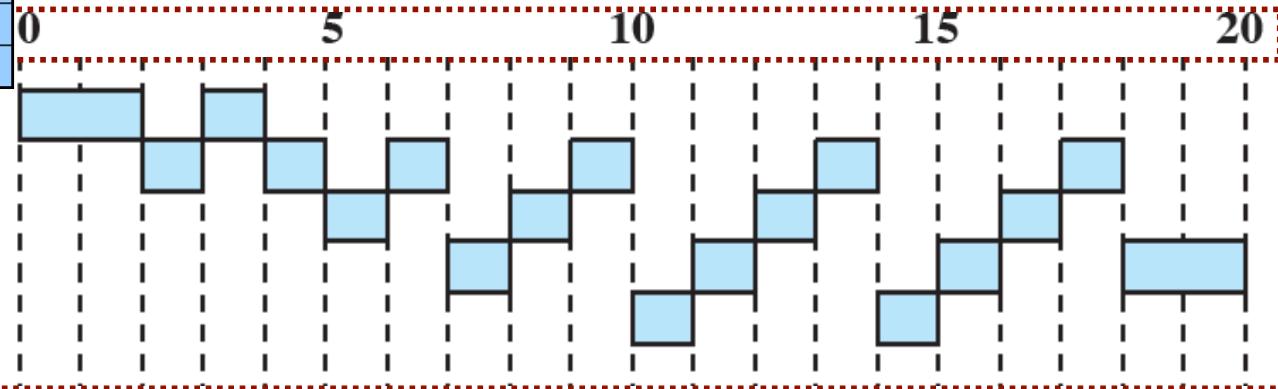
- Uses **preemption** based on a clock
- Also known as **time slicing** — each process is given **a slice of time before being preempted**
- Principal design issue — is the length of the time quantum, or slice, to be used?
- Particularly effective in a general-purpose time-sharing system or transaction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

Time quantum should be slightly greater than the time required for a typical interaction or process function

Round Robin: RR

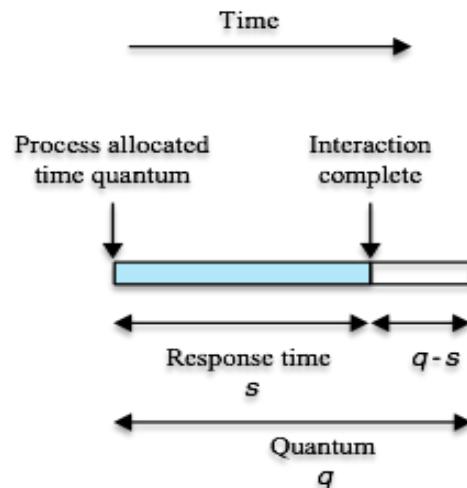
Process	Arrival Time	Service Time
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2

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

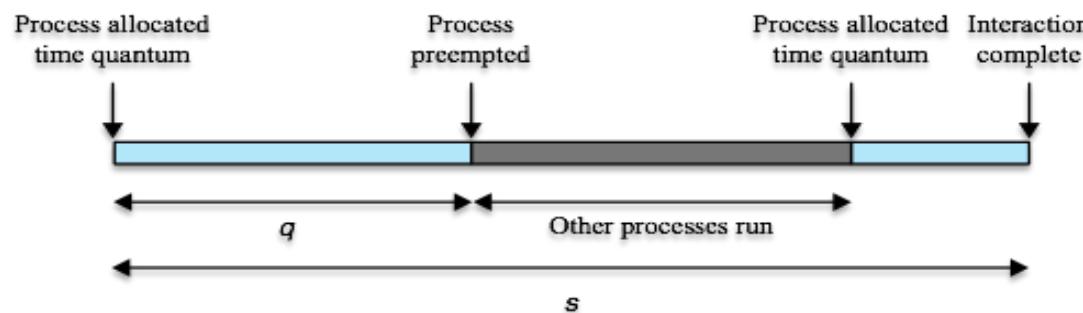


RR $q = 1$ (quantum q)						
Finish Time	4	18	17	20	15	Mean
Turnaround Time (Tr)	4	16	13	14	7	10.80
Tr/Ts	1.33	2.67	3.25	2.80	3.50	2.71
RR $q = 4$						
Finish Time	3	17	11	20	19	Mean
Turnaround Time (Tr)	3	15	7	14	11	10.00
Tr/Ts	1.00	2.5	1.75	2.80	5.50	2.71

Preemption Time Quantum: Effect

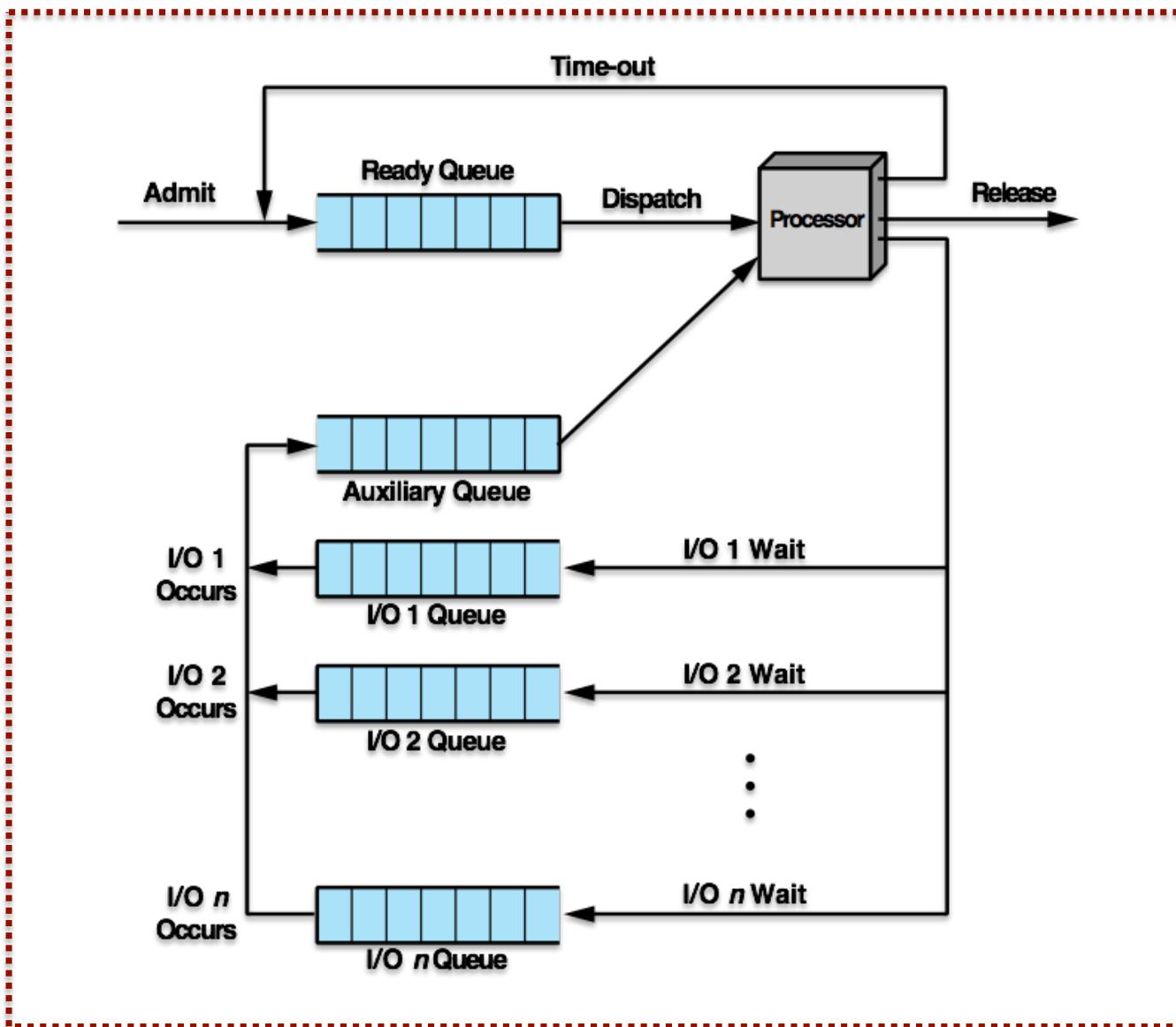


(a) Time quantum greater than typical interaction



(b) Time quantum less than typical interaction

Virtual Round Robin: VRR



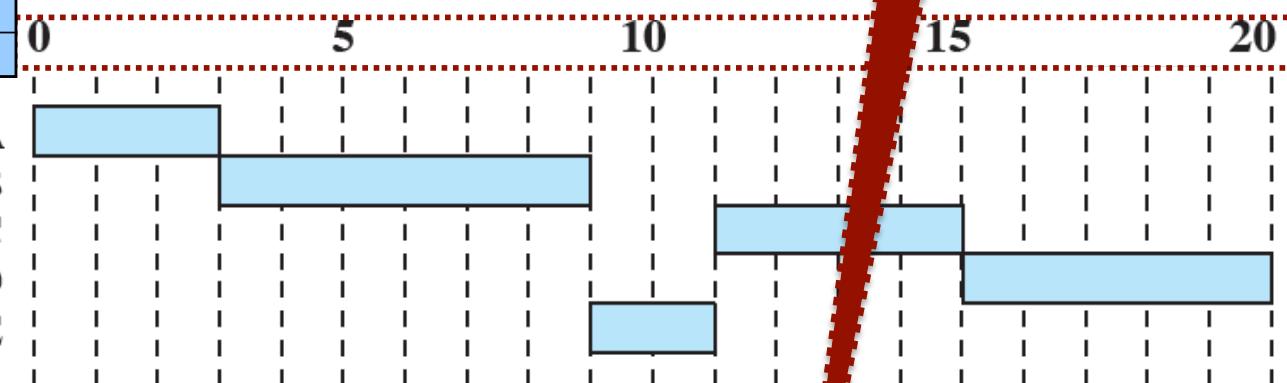
Shortest Process Next: SPN

- **Non-preemptive** policy
 - 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
- Difficulty: 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

When there is a steady supply of short jobs

Shortest Process Next: SPN

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



1.50 compared to
6.0 for FCFS

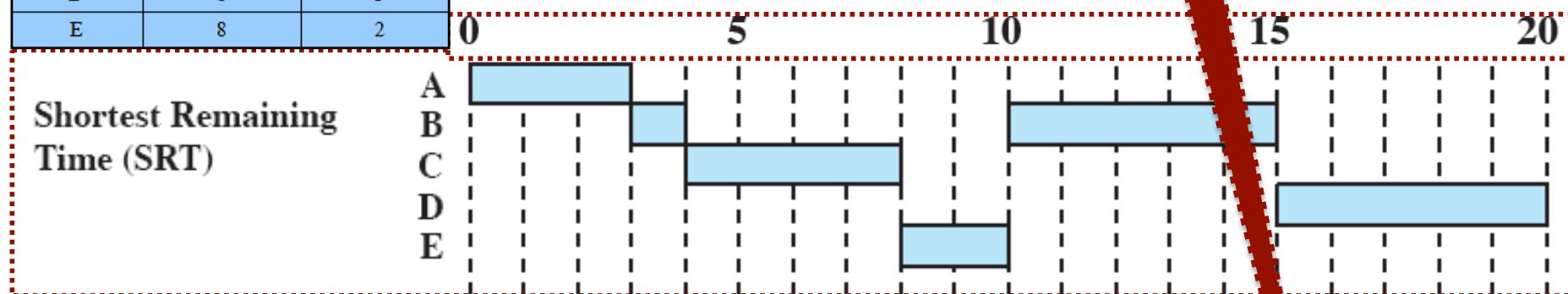
SPN						
Finish Time	3	9	15	20	11	Mean
Turnaround Time (Tr)	3	7	11	14	3	7.60
Tr/Ts	1.00	1.17	2.75	2.80	1.50	1.84

Shortest Remaining Time: SRT

- ❑ Preemptive version of SPN (Shortest Process Next)
 - ❑ 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

Shortest Remaining Time: SRT

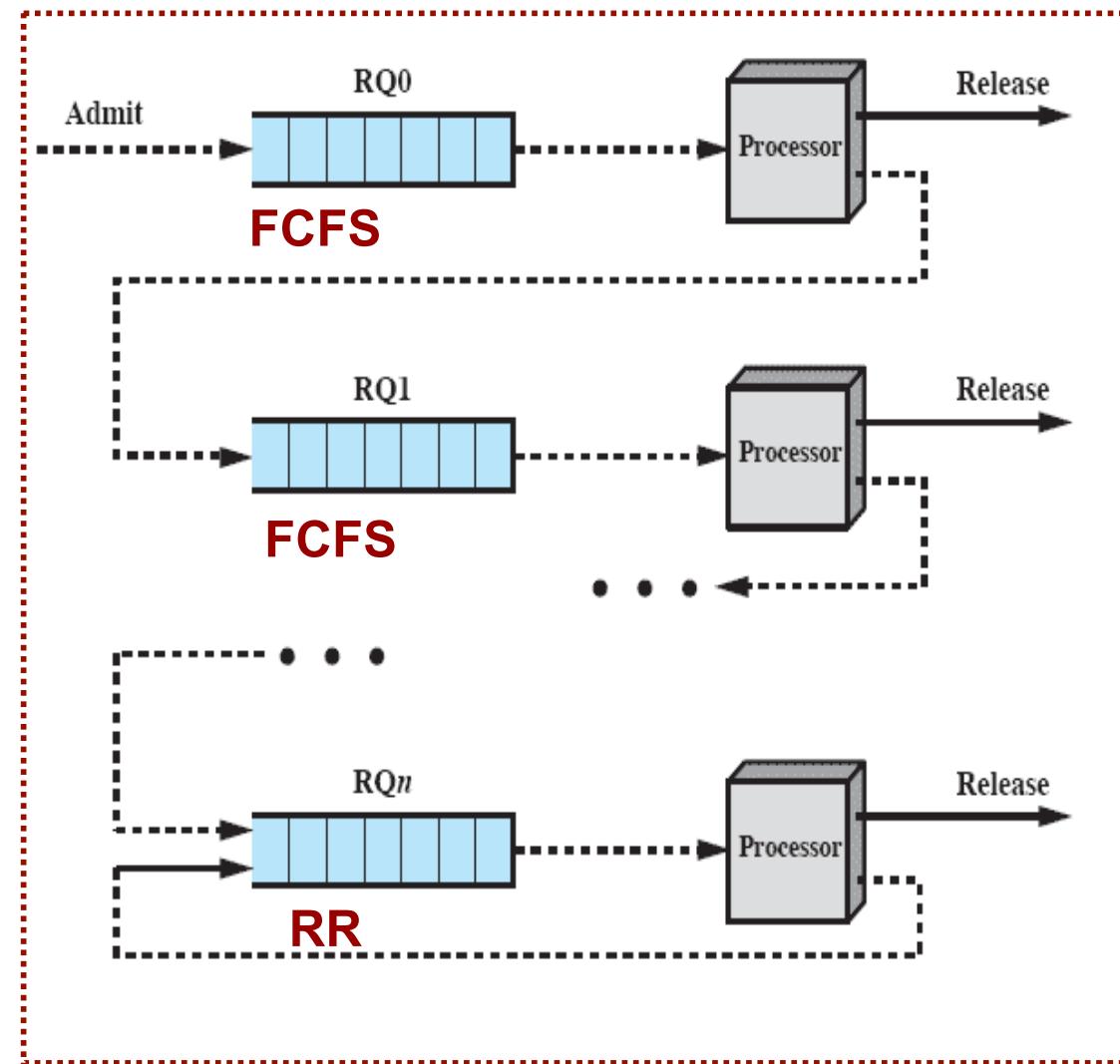
Process	Arrival Time	Service Time
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2



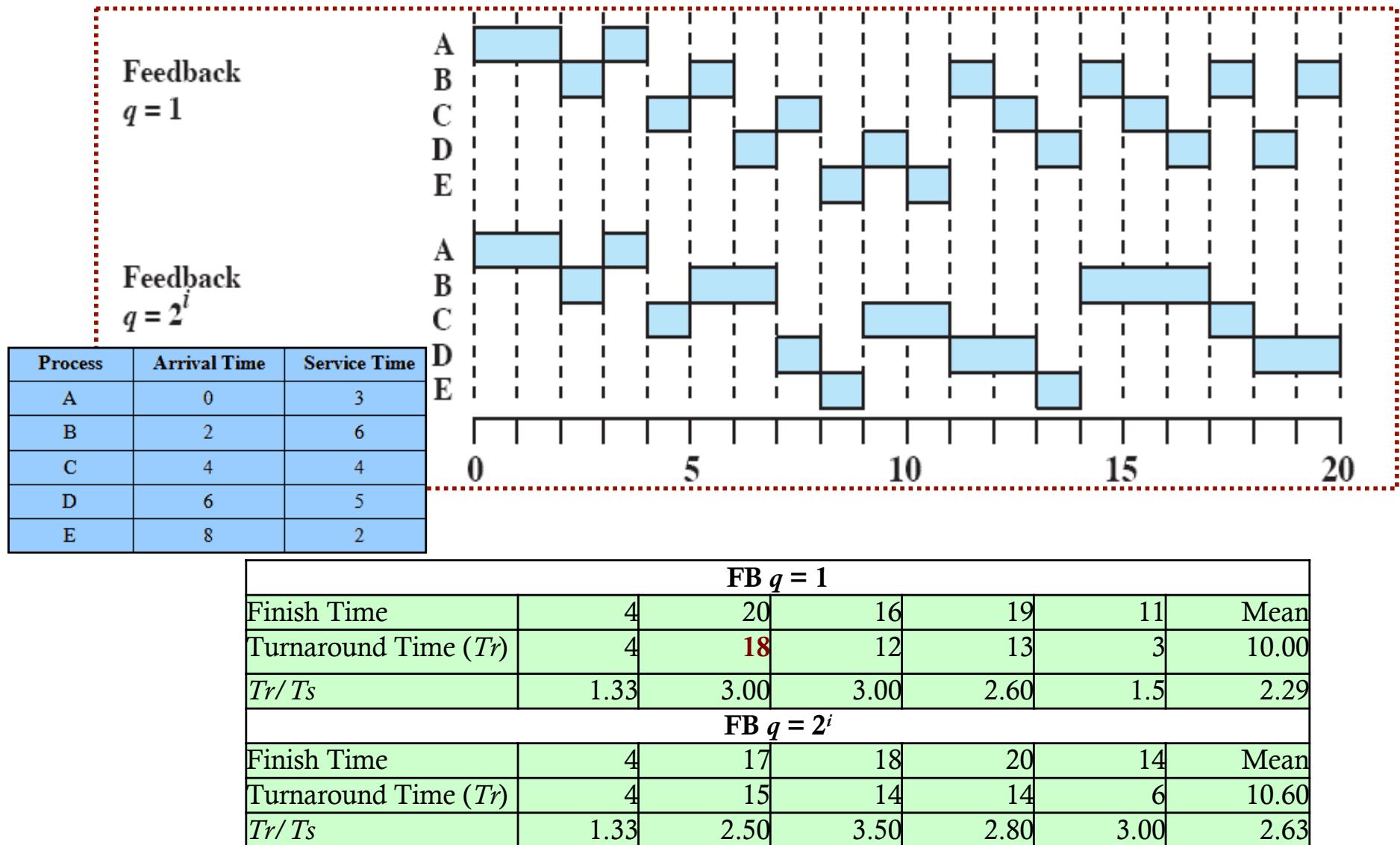
SRT						
Finish Time	3	15	8	20	10	Mean
Turnaround Time (Tr)	3	13	4	14	2	7.20
Tr/Ts	1.00	2.17	1.00	2.80	1.00	1.59

Feedback Scheduling

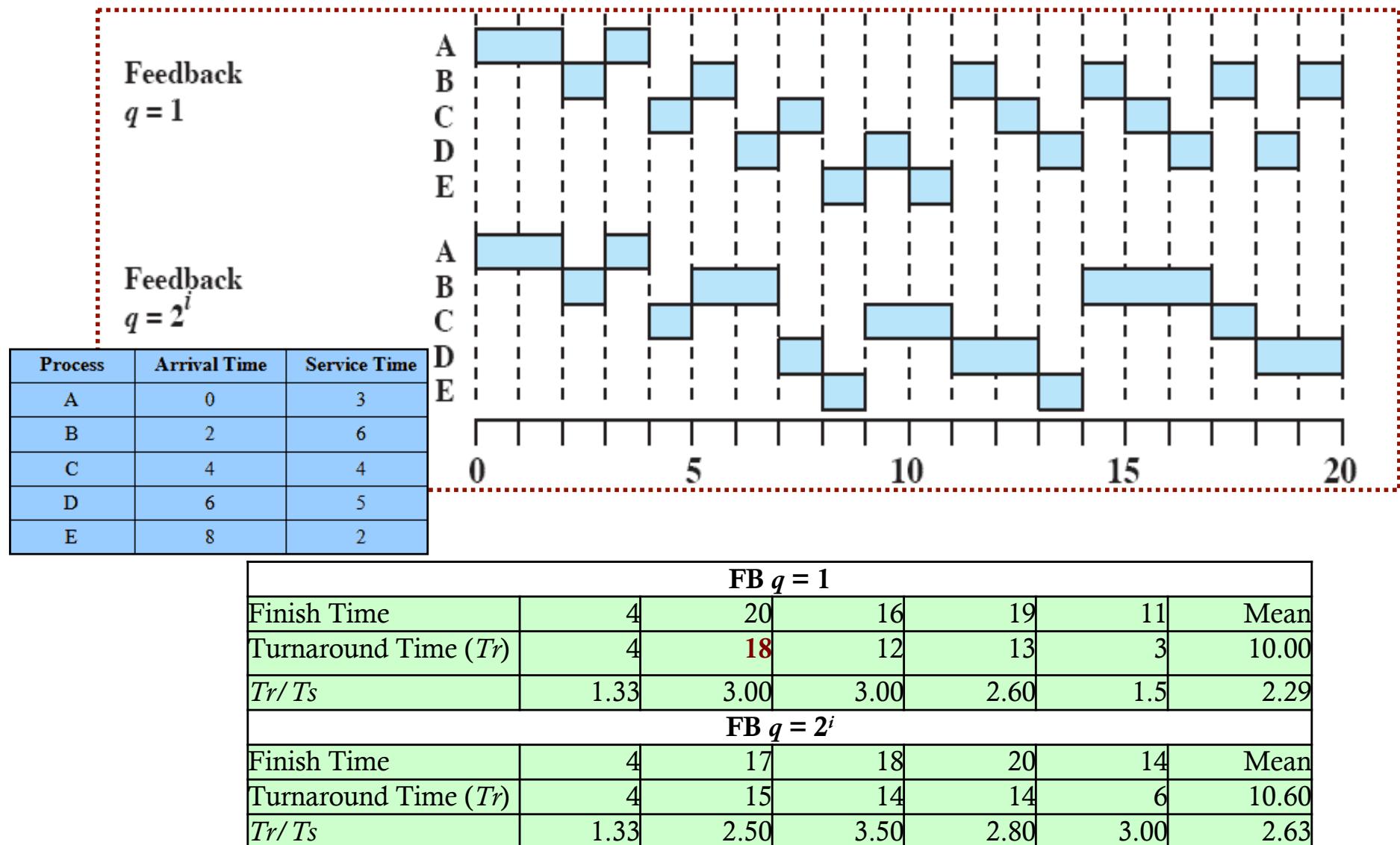
- ❑ Scheduling is done on a **preemptive** (at time quantum) basis
- ❑ **Dynamic priority** mechanism is used
- ❑ Focus on the time spent on CPU rather than predicting how much CPU time a process will use



Feedback Scheduling



Feedback Performance



Scheduling Policies: Comparison (1)

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

Scheduling Policies: Comparison (2)

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
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 $q = 2^i$						
Finish Time	4	17	18	20	14	
Turnaround Time (T_r)	4	15	14	14	6	10.60
T_r/T_s	1.33	2.50	3.50	2.80	3.00	2.63

Scheduling Policies: Characteristics

	FCFS	Round robin	SPN	SRT	Feedback
Selection function	$\max[w]$	constant	$\min[s]$	$\min[s - e]$	(see text)
Decision mode	Non-preemptive	Preemptive (at time quantum)	Non-preemptive	Preemptive (at arrival)	Preemptive (at time quantum)
Throughput	Not emphasized	May be low if quantum is too small	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	Not emphasized
Overhead	Minimum	Minimum	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	May favor I/O bound processes
Starvation	No	No	Possible	Possible	Possible

Summary of Lecture 6

- OS 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.
- Scheduling algorithms: **FCFS, Round Robin, SPN, SRT, Feedback, Fair Share**

Reading from Stallings, Chapter 9: 9.1, 9.2, 9.3