IT 540 Operating Systems
ECE519 Advanced Operating Systems

Prof. Dr. Hasan Hüseyin BALIK

(9th Week)

(Advanced) Operating Systems

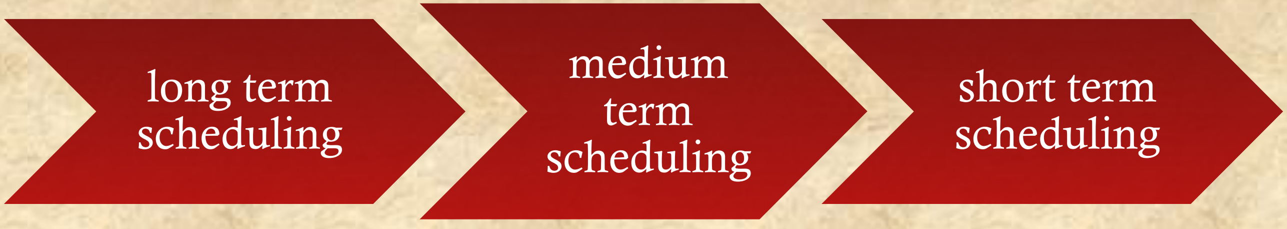
9. Uniprocessor Scheduling

9. Outline

- Types of Scheduling
- Scheduling Algorithms

Processor Scheduling

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



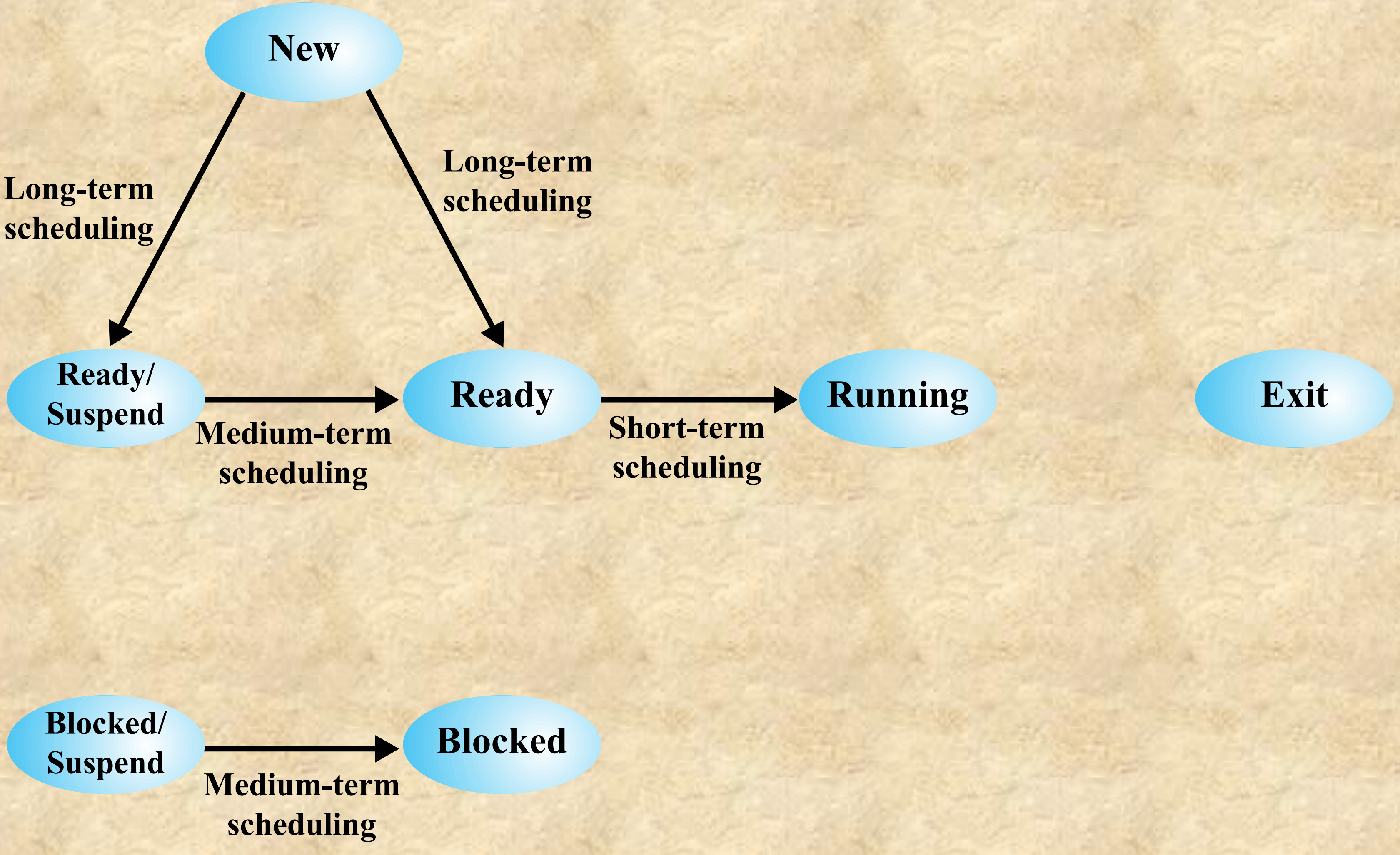
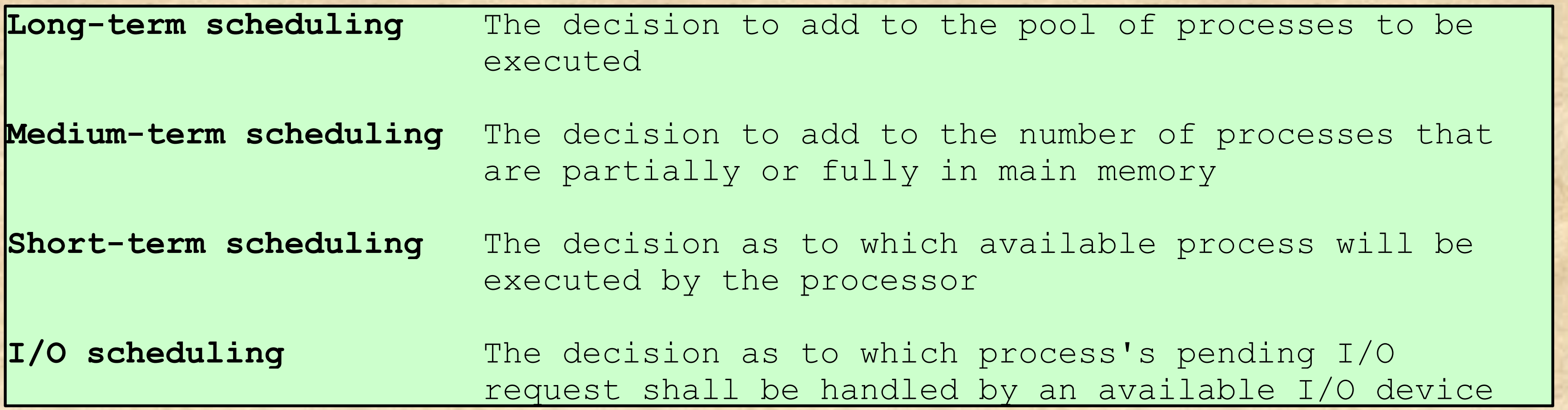
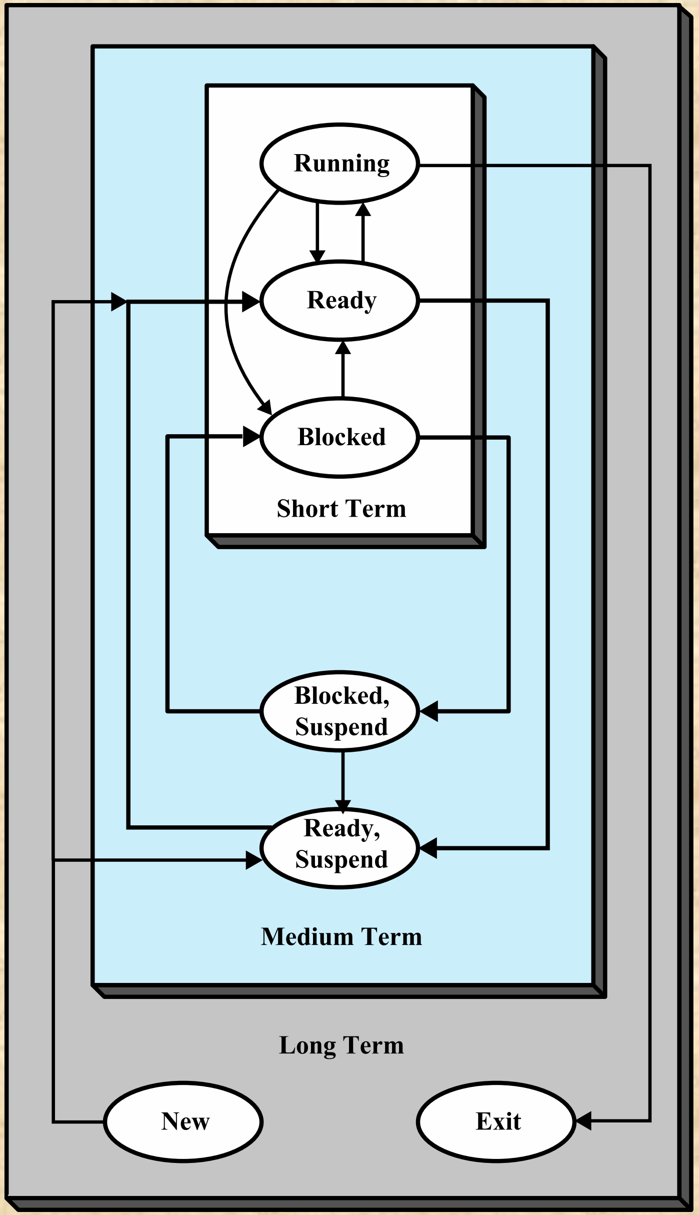


Figure 9.1 Scheduling and Process State Transitions

Types of Scheduling



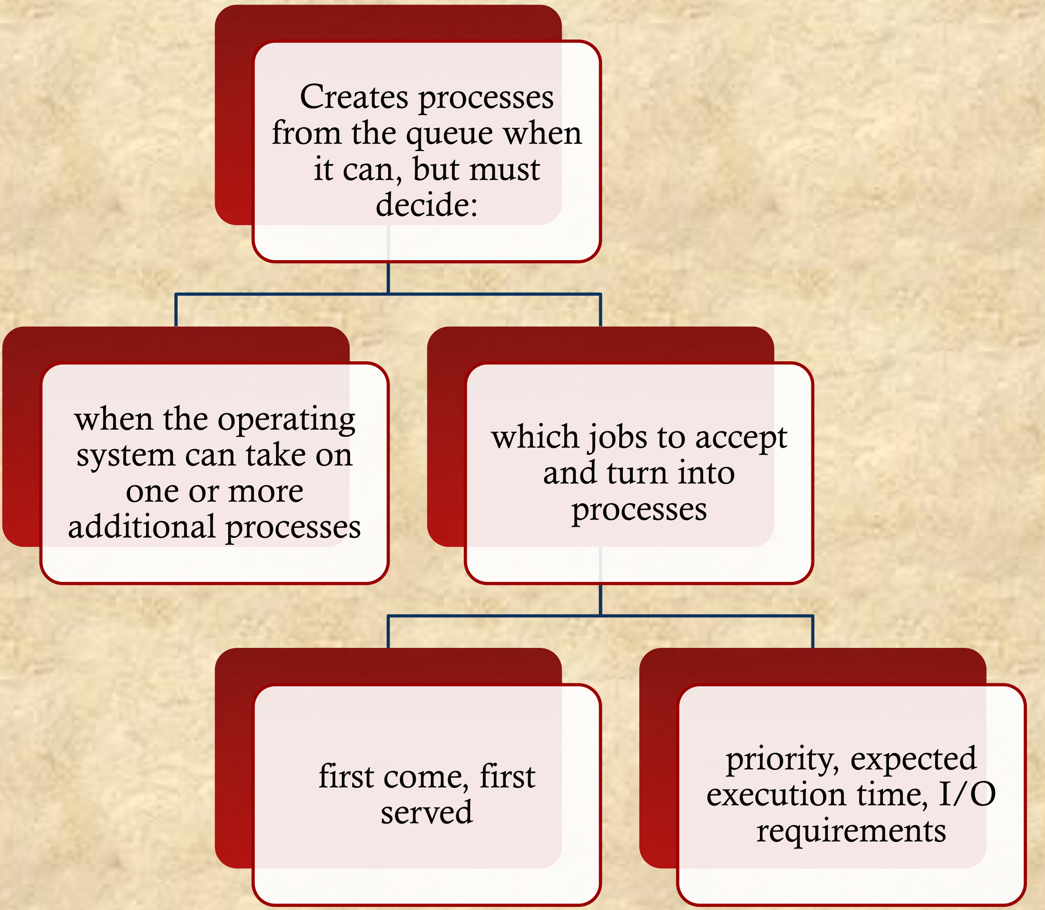
Long-term scheduling	The decision to add to the pool of processes to be executed
Medium-term scheduling	The decision to add to the number of processes that are partially or fully in main memory
Short-term scheduling	The decision as to which available process will be executed by the processor
I/O scheduling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


Running
Ready
Blocked
Short Term
Blocked, Suspend
Ready, Suspend
Medium Term
Long Term
New
Exit

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



Creates processes from the queue when it can, but must decide:

when the operating system can take on one or more additional processes

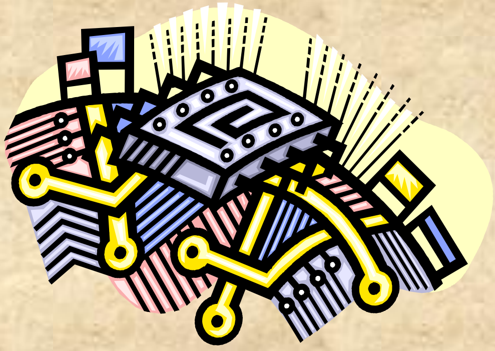
which jobs to accept and turn into processes

first come, first served

priority, expected execution time, I/O requirements

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

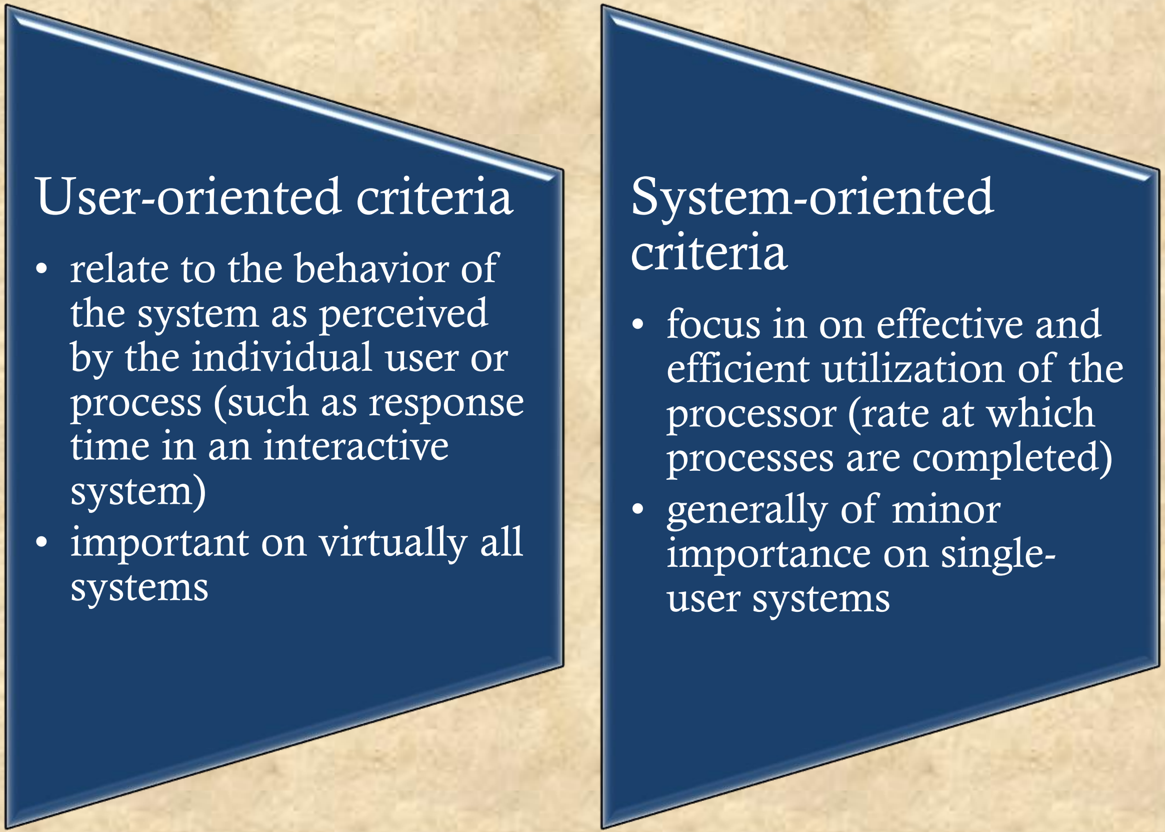
- Known as the dispatcher
- Executes most frequently
- Makes the fine-grained decision of which process to execute next
- Invoked when an event occurs that may lead to the blocking of the current process or that may provide an opportunity to preempt a currently running process in favor of another

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



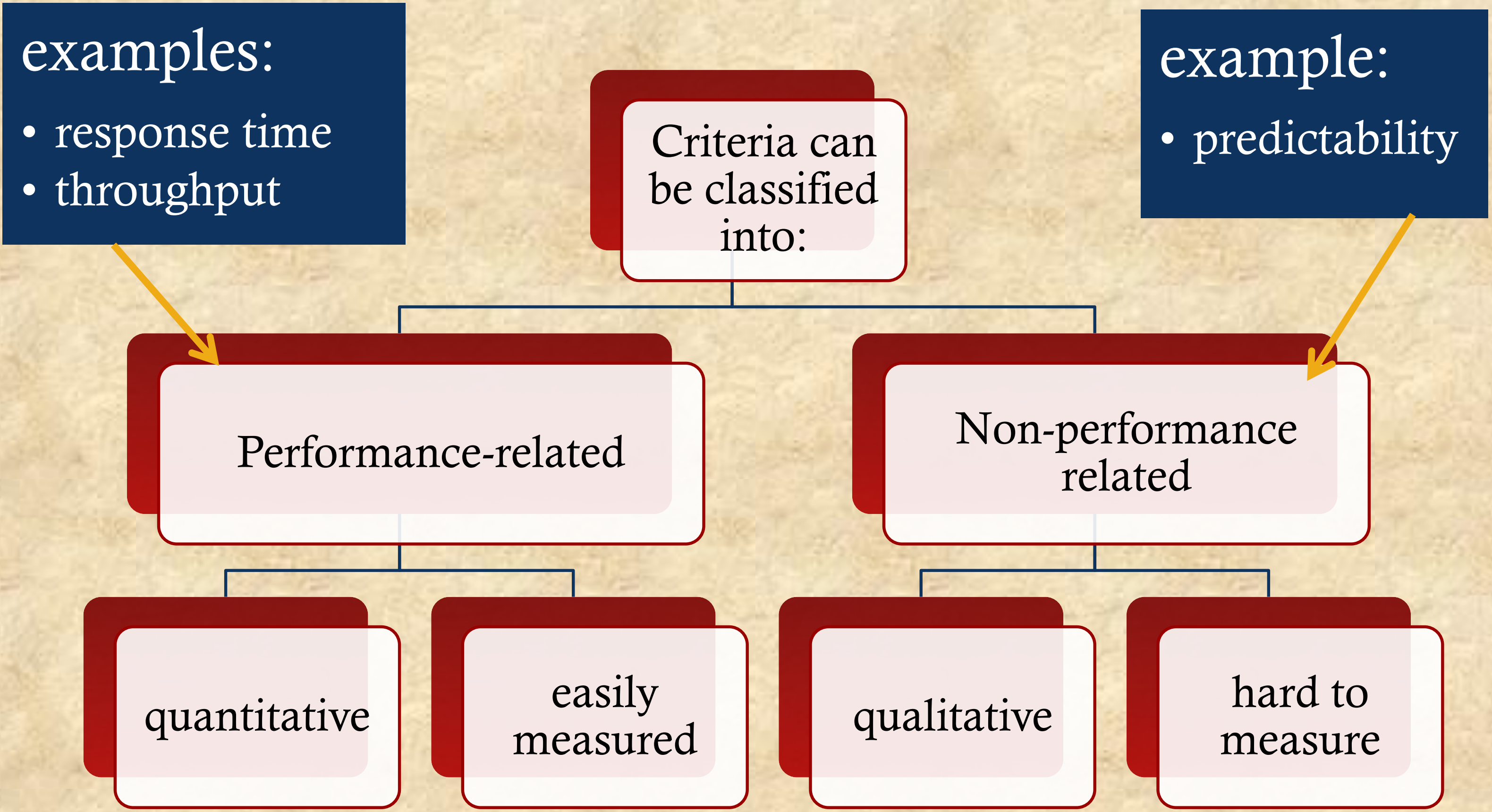
User-oriented criteria

- 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

example:

- predictability

Criteria can be classified into:

Performance-related

Non-performance related

quantitative

easily measured

qualitative

hard to measure

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, then 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*; generally, this quantity must be estimated or supplied by the user

Decision Mode

- Specifies the instants in time at which 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 new process arrives, on an interrupt, or periodically

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

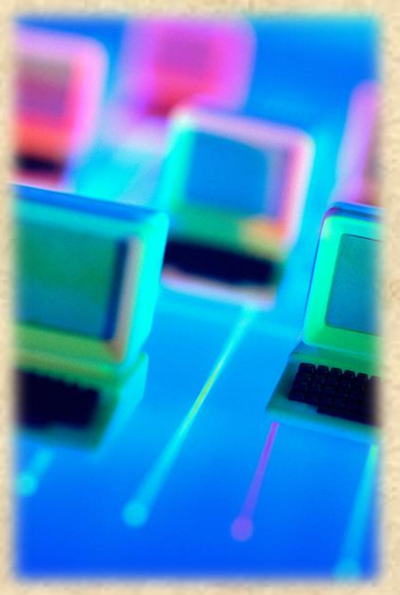
Shortest Process Next (SPN)

- 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 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

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

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

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

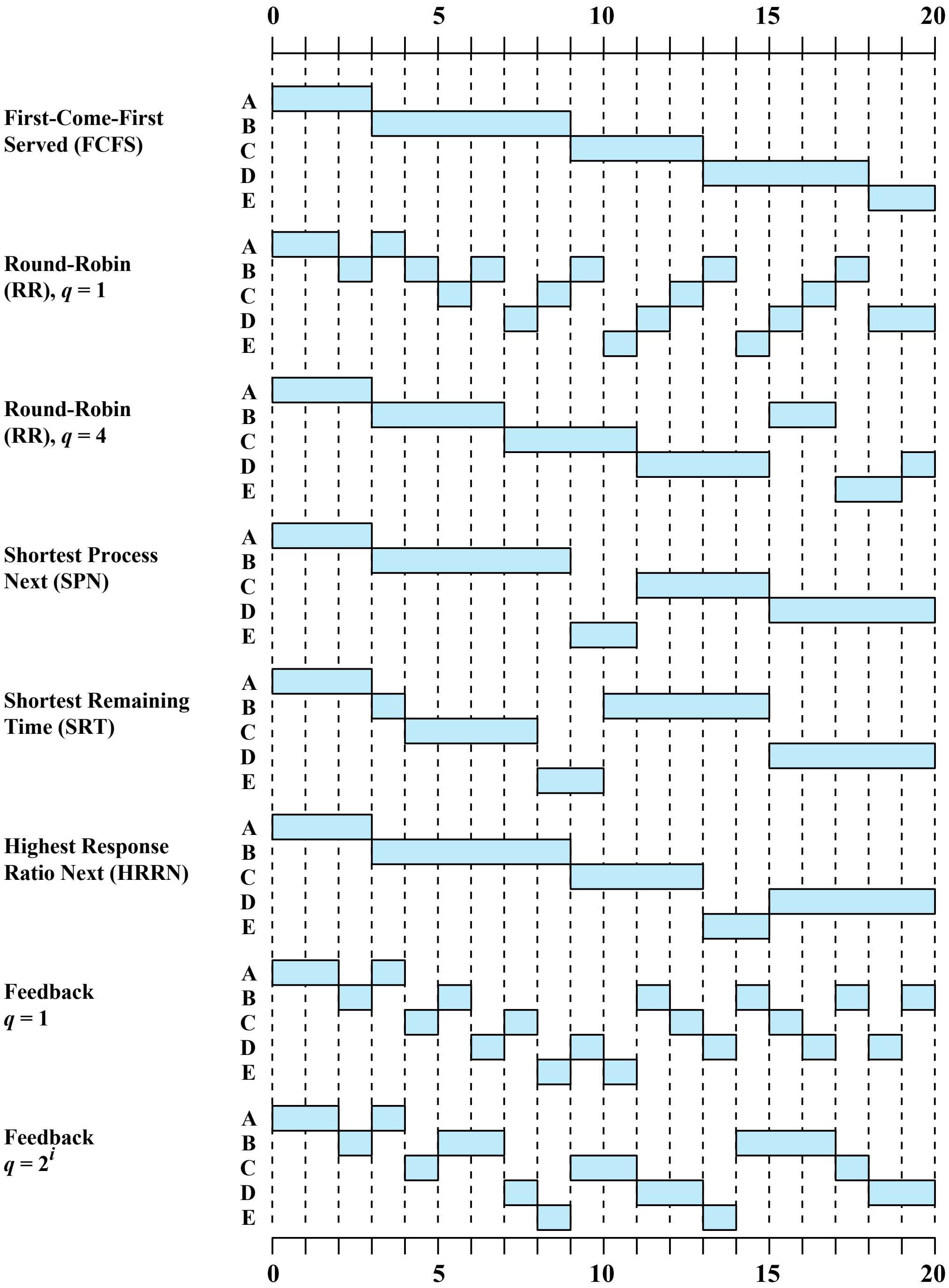


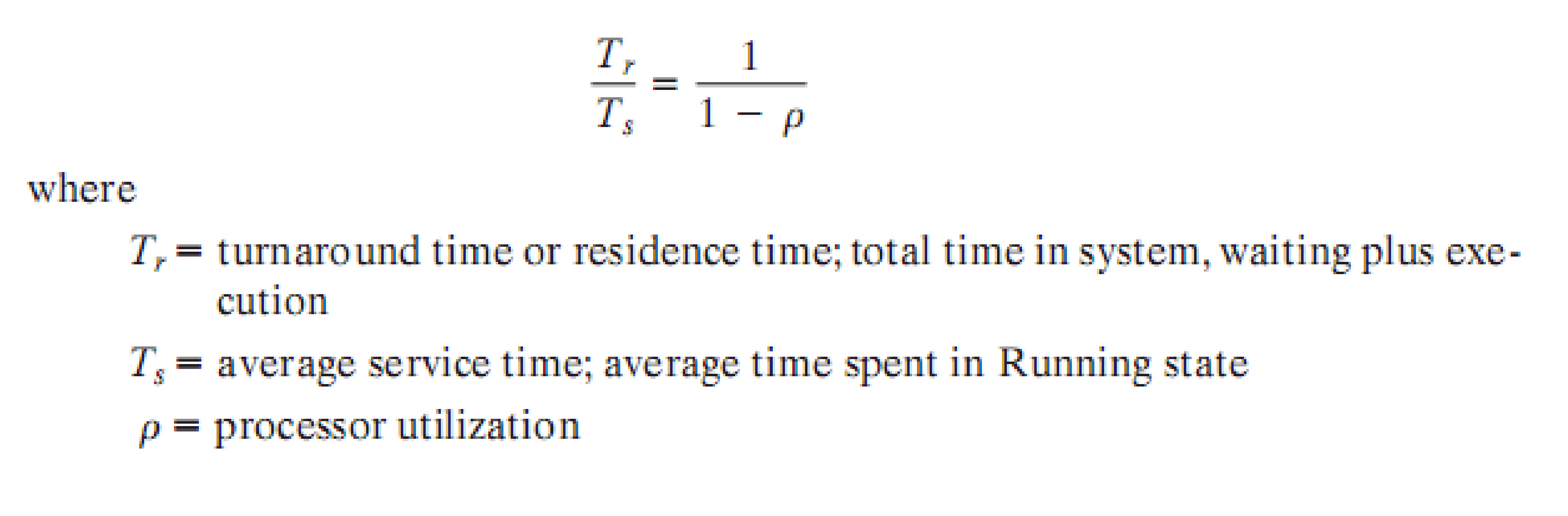
Figure 9.5 A Comparison of Scheduling Policies

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)
Through-Put	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

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

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