Types of Processor Scheduling, CPU Scheduling Basic concept, Scheduling Criteria, Scheduling Algorithms: FCFS, SJF, SRTF, Priority Scheduling, Round Robin, Highest Response Ratio Scheduling, Multilevel queue scheduling, Multilevel feedback queue scheduling, Traditional Unix scheduling.

WS 9.1 (pg.426- 429)
WS 9.2 (pg.430- 451)
WS 9.3 (pg.450- 454)

Types of Scheduling

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.

Types of Scheduling

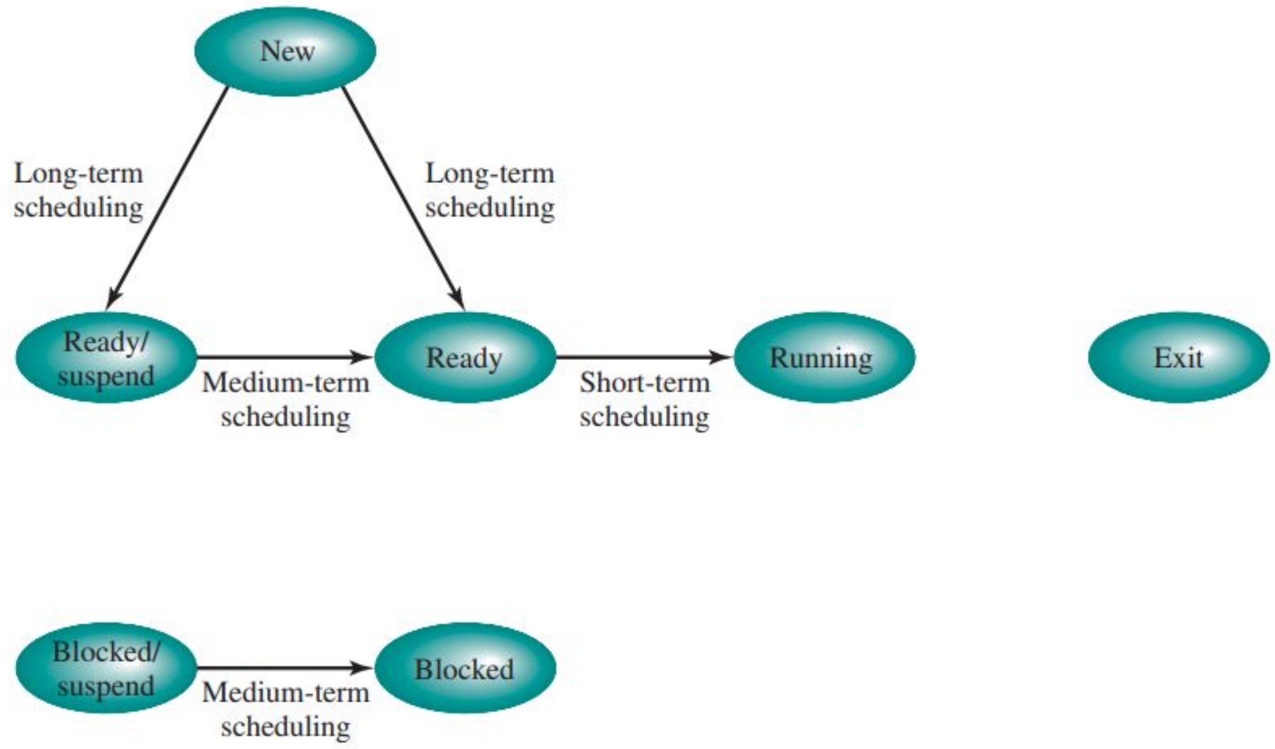

Figure 9.1 **Scheduling and Process State Transitions**

Types of Scheduling

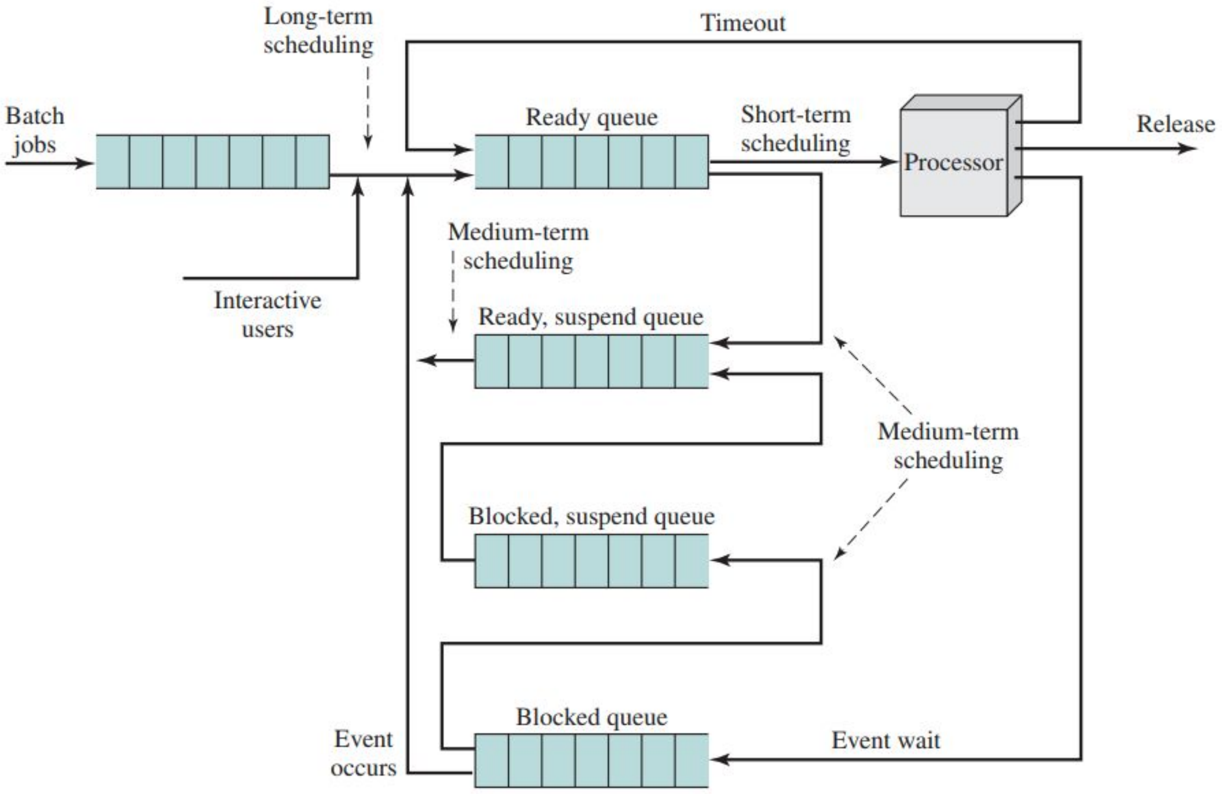

Figure 9.3 **Queueing Diagram for Scheduling**

Short-Term Scheduling Criteria

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.

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 Scheduling

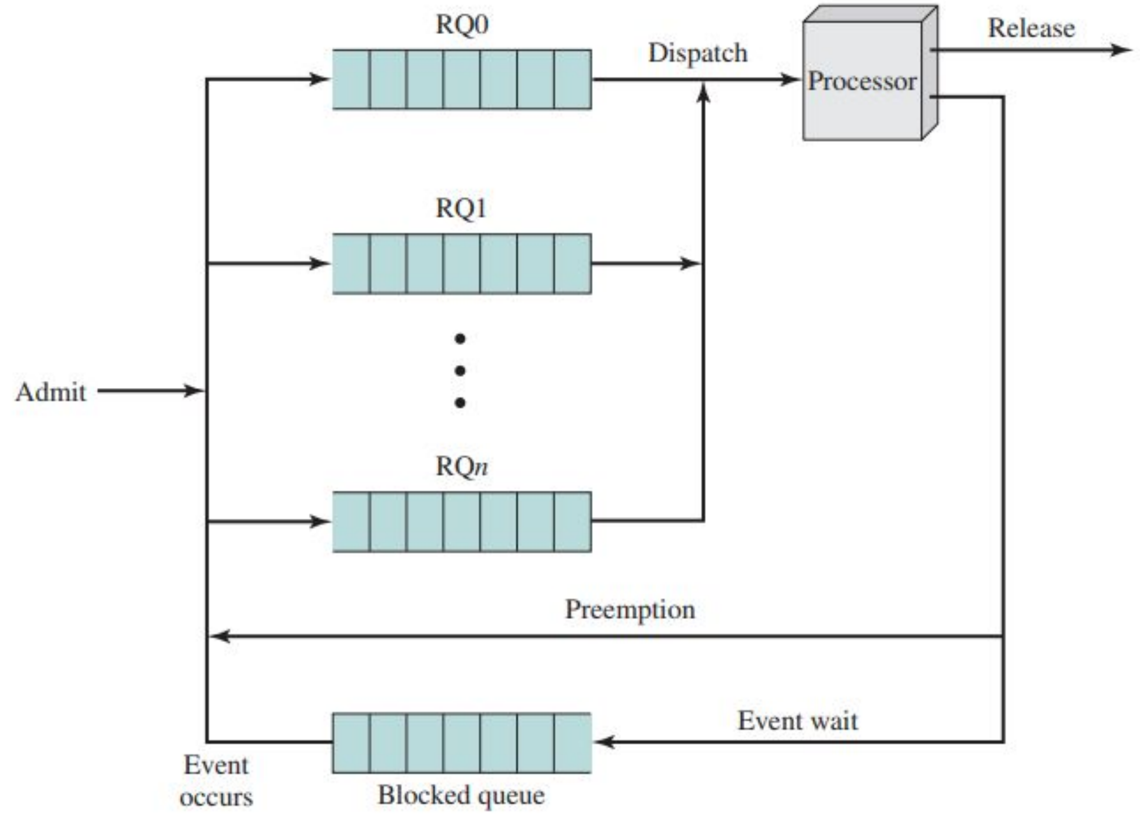

Figure 9.4 **Priority Queueing**

Characteristics

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	Preemp-tive (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

Round-Robin

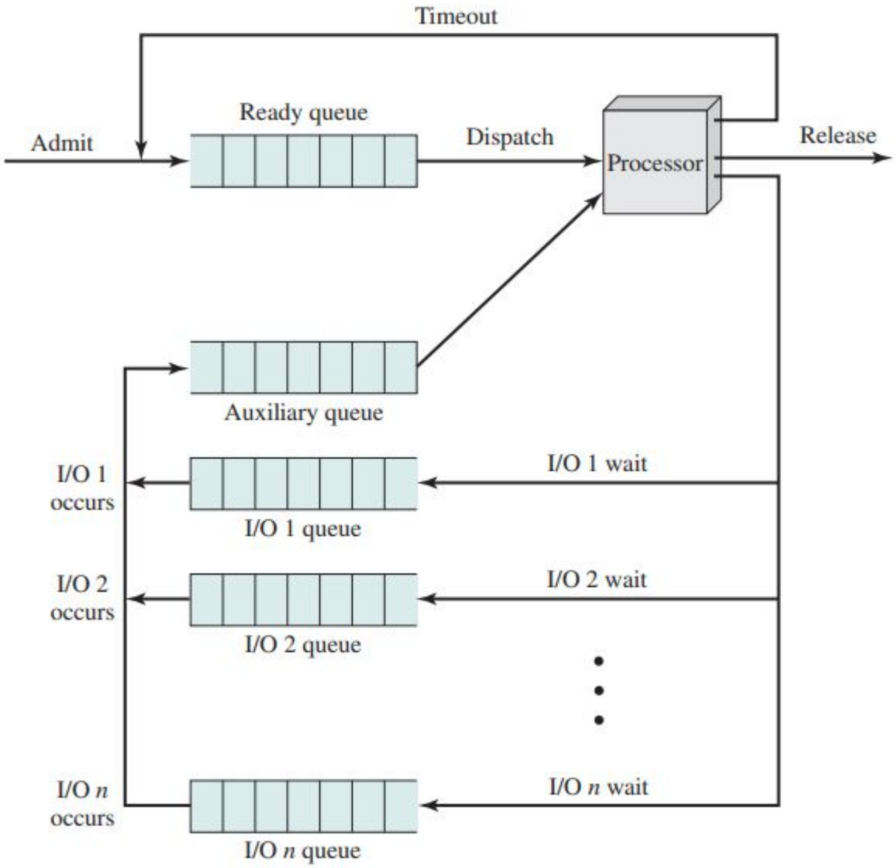

Figure 9.7 **Queueing Diagram for Virtual Round-Robin Scheduler**

Round-Robin

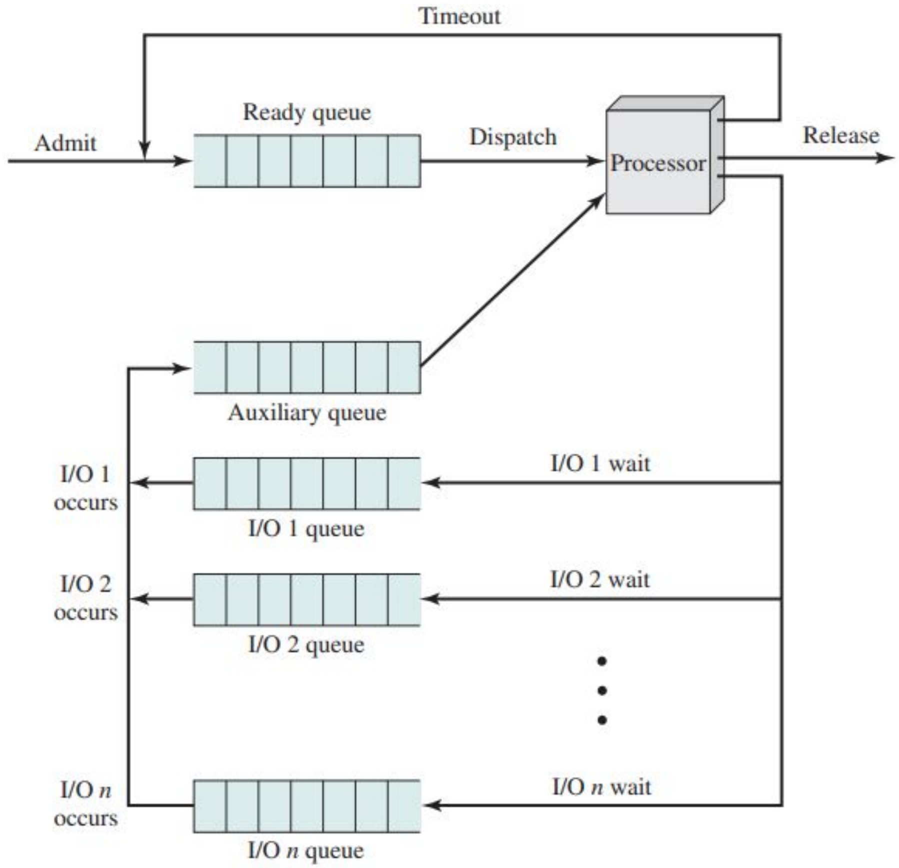

Figure 9.7 **Queueing Diagram for Virtual Round-Robin Scheduler**

Highest Response Ratio Next, Feedback Queue Scheduling

$$R = \frac{w + s}{s}$$

where

R = response ratio,
w = time spent waiting for the processor, and
s = expected service time.

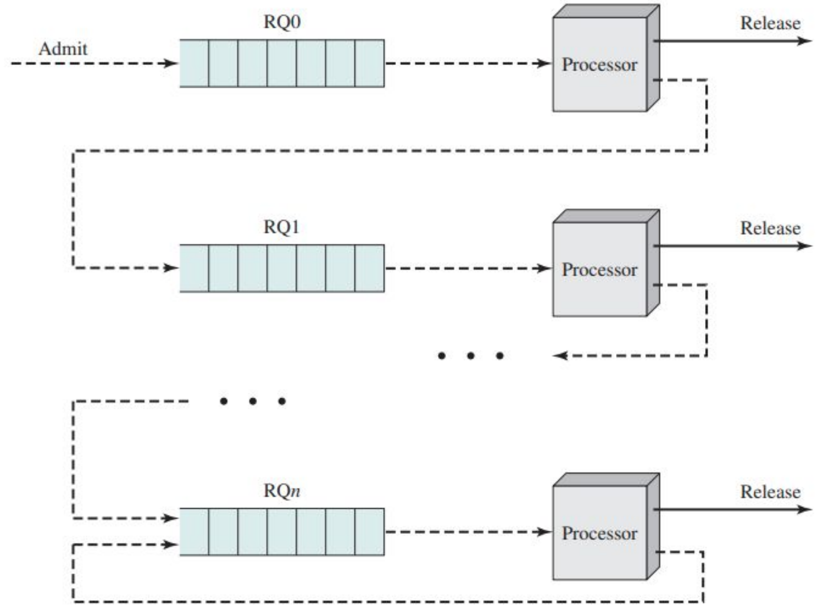

Figure 9.10 **Feedback Scheduling**

TRADITIONAL UNIX SCHEDULING

The traditional UNIX scheduler employs multilevel feedback using round robin within each of the priority queues. The system makes use of one-second preemption. That is, if a running process does not block or complete within one second, it is preempted. Priority is based on process type and execution history. The following formulas apply:

$$CPU_j(i) = \frac{CPU_j(i-1)}{2}$$

$$P_j(i) = Base_j + \frac{CPU_j(i)}{2} + nice_j$$

where

$CPU_j(i)$ = measure of processor utilization by process j through interval i,

$P_j(i)$ = priority of process j at beginning of interval i; lower values equal higher priorities,

$Base_j$ = base priority of process j, and

$nice_j$ = user-controllable adjustment factor.

Summary

A variety of algorithms have been developed for making the short-term scheduling decision among all ready processes:

- **First-come-first-served:** Select the process that has been waiting the longest for service.
- **Round robin:** Use time slicing to limit any running process to a short burst of processor time, and rotate among all ready processes.
- **Shortest process next:** Select the process with the shortest expected processing time, and do not preempt the process.
- **Shortest remaining time:** Select the process with the shortest expected remaining process time. A process may be preempted when another process becomes ready.
- **Highest response ratio next:** Base the scheduling decision on an estimate of normalized turnaround time.
- **Feedback:** Establish a set of scheduling queues and allocate processes to queues based on execution history and other criteria.

The choice of scheduling algorithm will depend on expected performance and on implementation complexity.