CPU SCHEDULING CH5

In the following lectures we will introduce the basic scheduling concepts and present several different CPU scheduling algorithms.

Scheduling Concepts
Scheduling is a fundamental operating system function. Almost all computer resources are scheduled before use. The CPU scheduling is central to operating systems.

CPU-I/O Burst Cycle

The success of CPU scheduling depends on the following observed property of processes: process execution consists of a cycle of CPU execution and I/O wait. Processes alternate between these two states. Process execution begins with a CPU burst. That is followed by I/O burst, then another CPU burst and so on. The last CPU burst will end with a system request to terminate execution.

CPU Scheduler
Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed. The selection process is carried out by the short term scheduler(CPU scheduler). The scheduler selects from amongthe processes in in memory that are ready to execute and allocates the CPU to one of them.

Scheduling Schemes

There are two scheduling schemes can be recognized:
- Preemptive scheduling
- Nonpreemptive scheduling

Under the nonpreemptive scheduling, once the CPU has been allocated to a process, the process keeps the CPU until it release

the CPU either by terminating or by switching to the waiting state. On the other hand Preemptive scheduling occure when the CPU has been allocted to a process and this process is interrupted by higher priority process. At this moment the executing process is stopped and returned back to the ready queue, the CPU is allocated to the higher priority process.

Dispatcher

It is the module that gives control of the CPU to the process selected by the CPU scheduler. This function involves:

- Switching Context
- Switching to user mode
- Jumping to the proper location in the user program to restart the program.

Scheduling Criteria

Many criteria have been suggested for comparing CPU scheduling algorithms. The criteria include the following:

- CPU Utilization
- Throughput
- Turnedaround Time
- Waiting time
- Response time

The optimization criteria are as follow:

- Max CPU utilization
- Max throughput
- Min turnaround time
- Min waiting time
- Min response time

Scheduling Algorithm

Here we will mention some of the CPU scheduling algorithms that are used in different operating systems

1• **First Come First Served (FCFS)**

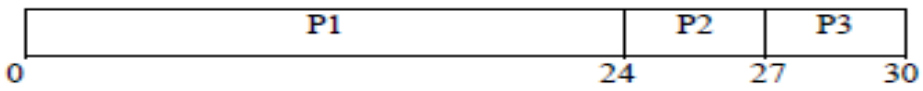
With this algorithm the process that requests the CPU first is allocated the CPU first. The implementation of the FCFS policy is easily managed with FIFO queue. When a process enters the ready queue, its PCB is linked onto the tail of the queue. When the CPU is free, it is allocated to the process at the head of the queue.

The average waiting time under the FCFS policy is often quite long. Consider the following set of processes that arrive at time 0, with the length of CPU burst time given in millisecond:

Example 1:

Process	Burst time
P1	24
P2	3
P3	3

The Gantt Chart is as follows:

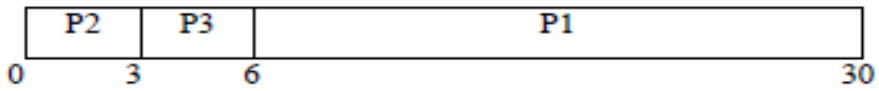


The average waiting time = (0+24+27)/ 3=17 millisecond

The average completion time = (24+27+30)/3 = 27 millisecond

Example 2:

If the processes arrive in the order P2, P3, P1 the result will be shown in the following Gantt Chart:



The average waiting time = (0+3+6)/ 3=3 millisecond

The average completion time = (3+6+30)/3 = 13 millisecond (compared to 27)

Thus the average waiting time under FCFS policy is not the minimal.

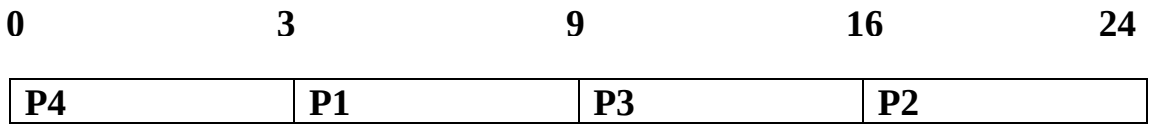
2. Shortest Job First Scheduling (SJF)

This algorithm associate with each process the length of the latter's next CPU burst. When the CPU is available, it is assigned to the process has the smallest next CPU burst. If two processes have the same length , FCFS scheduling is used to break this tie.
As an example consider the following set of processes with the length of the CPU burst given in millisecond:

Example **3** using **SJF:**

Process	Burst time
P1	6
P2	8
P3	7
P4	3

Gantt chart:



W. T p1 = 3-0 = 3
W. T p2 = 16-0 = 16
W. T p3 = 9-0 = 9
W. T p4 = 0-0 = 0

Average wait time = (3 + 16 + 9 + 0) / 4 = 7 milliseconds
The average completion time = (9+24+16+3)/4 =13 milliseconds
The average waiting time in SJF is the optimal that it gives the minimum average waiting time.
The SJF is either preemptive or non-preemptive.

- ***Non preemptive:***once CPU given to the process it cannot be preempted until completes its CPU burst.
- ***Preemptive:*** if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is known as the **Shortest Remaining Time First (SRTF)**.

As an example consider the following set of processes with the length of the CPU burst given in millisecond:
Example **4** using **SJF:**

Process	Burst Time	Arrival Time
P0	4	0
P1	4	1
P2	2	2
P3	1	5
P4	3	7

Gantt chart:

0	**4**	**6**	**7**	**10**	**14**
P0	**P2**	**P3**	**P4**	**P1**	

W. T p0 = 0-0 = 0
W. T p1 = 10-1 = 9
W. T p2 = 4-2 = 2
W. T p3 = 6-5 = 1
W. T p4 = 7-7 = 0
Average wait time = (0 + 9 + 2 +1+0) / 5 = 2.4 milliseconds

1. **Priority Scheduling Algorithm**

In this algorithm a priority is associated with each process and the CPU is allocated to the process of the highest priority. We use the low numbers to represent high priority.
As an example consider the following set of processes with the length of the CPU burst given in millisecond
Example **5** using **priority**:

Process	Burst time	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Gantt chart:

P2	**P5**	**P1**	**P3**	**P4**

0 1 6 16 18 19

W. T p1 = 6-0= 6
W. T p2 = 0-0 = 0
W. T p3 = 16-0 = 16
W. T p4 = 18-0 = 18
W. T p5 = 1-0 = 1
Average wait time = (6 + 0 + 16 +18+1) / 5 = 8.2 milliseconds

Example **6** using **non-preemptive priority:**

Process	Burst time	Arrival	Priority
P0	4	0	3
P1	2	3	2
P2	5	4	1
P3	6	10	0
P4	8	11	5

Gantt chart:

0 4 9 11 17 25

P0	**P2**	**P1**	**P3**	**P4**

W. T p0 = 0- 0= 0
W. T p1 = 9-3 = 6
W. T p2 = 4 - 4 = 0
W. T p3 = 11-10 = 1
W. T p4 = 17- 11 = 6

Average wait time = (0 + 6 + 0 +1+6) / 5 = 2.6 millisecond

Example**7**using**Preemptive Priority:**

Process	Burst time	Arrival	Priority

P0	5	0	3
P1	8	2	2
P2	3	3	1
P3	6	4	0
P4	2	8	4

Gantt chart:

0	2	3	4	10	12	19	22	24
P0	**P1**	**P2**	**P3**	**P2**	**P1**	**P0**	**P4**	

W. T p0 = 19-2-0 = 17
W. T p1 = 12 -2-1 = 9
W. T p2 = 10 – 3- 3 = 4
W. T p3 = 4- 4 = 0
W. T p4 = 22- 8 = 14
Average wait time = (17+9+4 + 0 +14) / 5 = 8.8milliseconds

Priority scheduling can be either Preemptive or nonpreemptive, when a process arrives the ready queue, its priority is compared with the priority of the currently running process. A preemptive priority will preempt the CPU if the priority of the newly arrived process is higher than the priority of the currently running process. A nonpreemptive priority scheduling will put the new process with the higher priority than the priority of the currently running process at the head of the ready queue.

4.Round Robin Scheduling Algorithm

The Round Robin algorithm is designed especially for time sharing system. It is similar to FCFS but preemption is added switch between processes. A small unit of time called time quantum (or time slice) is defined. A time quantum is generally from 10 to 100 milliseconds. The ready queue is treated as a circular queue, allocated the CPU to each process for a time interval of up to 1 time quantum.
The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatch the processes. One of two things will then happen. The processes may have a CPU burst of less than 1 time quantum. In this case,

the processes itself will release the CPU. The scheduler will then proceed to the next process in the ready queue. Otherwise, if the CPU burst of the currently running processes is longer than 1 time quantum, the timer will go off and will cause an interrupt to the operating system. A context switch will be executed, and the process will be put at the tail of the ready queue. The CPU scheduler will then select the next process in the ready queue.

As an example consider the following set of processes with the length of the CPU burst given in millisecond:

Example 8 using RR:quantum=4

Process	Burst time
P1	24
P2	3
P3	3

Gantt chart:

0 4 7 10 14 18 22 26 30

P1	**P2**	**P3**	**P1**	**P1**	**P1**	**P1**	**P1**

W. T p1 = 26-4-4-4-4-4 = 6
W. T p2 = 4-0 = 4
W. T p3 = 7-0 = 7
Average wait time = (6 + 4 +7) / 3 = 5.66 milliseconds

Example 9 using **preemptive RR**:quantum=2

Process	Burst time	Arrival time
P0	6	0
P1	8	1
P2	4	7
P3	2	9
P4	10	11

0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30

P0	P1	P0	P1	P2	P3	P4	P0	P1	P2	P4	P1	P4	P4	P4

W. T p0 = 14- 2 – 2- 0= 10
W. T p1 = 22 -2-2-2-1 = 15
W. T p2 = 18 -2-7 = 9
W. T p3 = 10 -9 = 1
W. T p4 = 28 -2-2-2-2-11 = 9

Average wait time = (10 + 15 +9+1+9) / 5 = 8.8 milliseconds

Comparison among Scheduling Algorithms

Algorithms	Policy type	Dis advantages	Advantages
FCFS	Non preemptive	Average waiting time is often quite long.	Easy to implement.
SJF	Non preemptive Or preemptive	Knowing the length of the next CPU request.	Gives minimum average waiting time.
Priority	Non preemptive Or preemptive	Blocking or starvation.	1-Simplicity. 2-support for priority.
R.R	preemptive	If the set time is too long, then the system may become unresponsive, time wasting and would emulate First Come First Served.	It is easy to implement in software