Scheduling

Concepts of Scheduling

As cloud computing is serving millions of users simultaneously, it must have the ability to meet all users requests with high performance and guarantee of quality of service (QoS). Therefore, we need to implement an appropriate task scheduling algorithm to fairly and efficiently meet these requests. Task scheduling problem is the one of the most critical issues in cloud computing environment because cloud performance depends mainly on it.

The concept of scheduling in cloud computing refers to the technique of mapping a set of jobs to a set of virtual machines (VMs) or allocating VMs to run on the available resources in order to fulfill users' demands.

Some Scheduling Techniques

- Scheduling is the process of mapping and managing tasks or processes into available resources.

- FCFS
- Shortest Job First
- Min-Min
- Max-Min

FCFS

- CPU gets a lot of processes to handle. Consider a CPU and also consider a list in which the processes are listed as follows:-

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

- Here, Process Number is used instead of the process name
- Arrival Time is the time when the process has arrived the list.
- Burst Time is the amount of time required by the process from CPU.
- Unit of time you can take anything like nano-second, second, minute etc whatever. We consider it as second.

FCFS

- Now for an instance, consider the above list as the ready queue for the CPU, that is the CPU will take processes from this list and will process it.
- Here in FCFS, CPU will take the first process in the list and will process it, then it will take the second process and so on. So this is a very easy thing to do huh ! Ok. So, lets see what CPU is doing,
- At Time 0s Doing 1s time Unit Job of Process-1, so Process-1 has left 2s Time Unit Job
- At Time 1s Doing 1s time Unit Job of Process-1, so Process-1 has left 1s Time Unit Job
- At Time 2s Doing 1s time Unit Job of Process-1, so Process-1 has left 0s Time Unit Job
- At Time 3s Doing 1s time Unit Job of Process-2, so Process-2 has left 1s Time Unit Job
- At Time 4s Doing 1s time Unit Job of Process-2, so Process-2 has left 0s Time Unit Job
- At Time 5s Doing 1s time Unit Job of Process-3, so Process-3 has left 0s Time Unit Job

We can show the above thing as the following time-line

Process-1	Process-1	Process-1	Process-2	Process-2	Process-3
0s	1s	2s	3s	4s	5s

FCFS

```
| Process-1   |            |            |
               Process-2    Process-3
0              3            5            6
```

- A shortened view of the above time-line is as follows,

Pr#	Arr Time	Burst Time
1	0	3
2	1	2
3	2	1

Process-1	Process-1	Process-1	Process-2	Process-2	Process-3

Advantages

- Most popular and simplest scheduling algorithm.
- Fairer than other simple scheduling algorithms.
- Depend on FIFO rule in scheduling task.
- Less complexity than other scheduling algorithms.

Disadvantages

- Tasks have high waiting time.
- Not give any priority to tasks. That means when we have large tasks in the begin tasks list, all tasks must wait a long time until the large tasks to finish.
- Resources are not consumed in an optimal manner.

.

Shortest Job First (SJF)

- This method is quite same as the FCFS but the difference is the in this case the processor will not process the jobs (processes) as they will come.
- Instead, a scheduler after a complete cycle (consider this as a 1 second job done) will check which is the job with the shortest burst time right now and will work on that. Now consider a CPU and also consider a list in which the processes are listed as follows:

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

Shortest Job First (SJF)

AT 0s:

There is only 1 job that is Process-1 with burst time 3. So CPU will do 1 second job of Process-1. Thus Process-1 has 2s more job to be done.

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

AT 1s:

Now there are 2 jobs,

- Process-1 that arrived at 0s and has 2s job to be done.
- Process-2 that arrived at 1s and has 2s job to be done.

So as both of the jobs as equal amount of jobs to be done, CPU will do the job that arrived earlier, that is 1s job of Process-1. So process-1 has more 1s job.

- The scheduler after a complete cycle (consider this as a 1 second job done) will re-evaluate the situation and determine which is the job with the shortest burst time right now and will work on that.
- This comparison of burst time includes the new jobs arrived and the pending jobs which have already started.
- If the duration for any two jobs (time to be completed) are equal, then prefernce is given to the one who arrived earlier.

Shortest Job First (SJF)

AT 2s:

Now there are 3 jobs,

- Process-1 that arrived at 0s and has 1s job to be done.
- Process-2 that arrived at 1s and has 2s job to be done.
- Process-3 that arrived at 2s and has 1s job to be done.

Here, 2 jobs has the shortest amount of job to be done, that is 1s. Both Process-1 and Process-3 has 1s job. CPU will do the job that arrived earlier, that is 1s job of Process-1. So process-1 has more 0s job.

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

Shortest Job First (SJF)

<u>AT 3s:</u>

Now there are 2 jobs,

- Process-2 that arrived at 1s and has 2s job to be done.
- Process-3 that arrived at 2s and has 1s job to be done.

Here, Process-3 has the shortest amount of job to be done, that is 1s. CPU will do 1s job of Process-3. So process-3 has more 0s job.

<u>AT 4s:</u>

There is only 1 job, that is Process-2 with burst time 2. So CPU will do 1 second job of Process-2. Thus Process-2 has 1s more job to be done.

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

Shortest Job First (SJF)

Process No	Arrival Time	Burst Time
1	0	3
2	1	2
3	2	1

AT 5s:

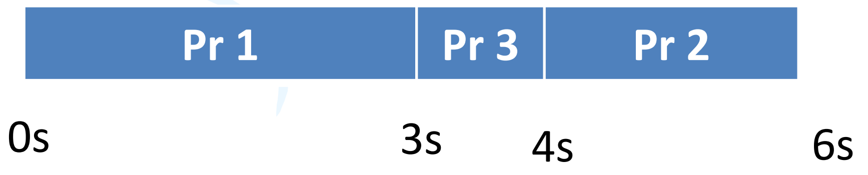
- There is only 1 job, that is Process-2 with burst time 1. So CPU will do 1 second job of Process-2. Thus Process-2 has 0s more job to be done.
- All Job Done.

We can show the above thing as the following time-line

Process-1	Process-1	Process-1	Process-3	Process-2	Process-2

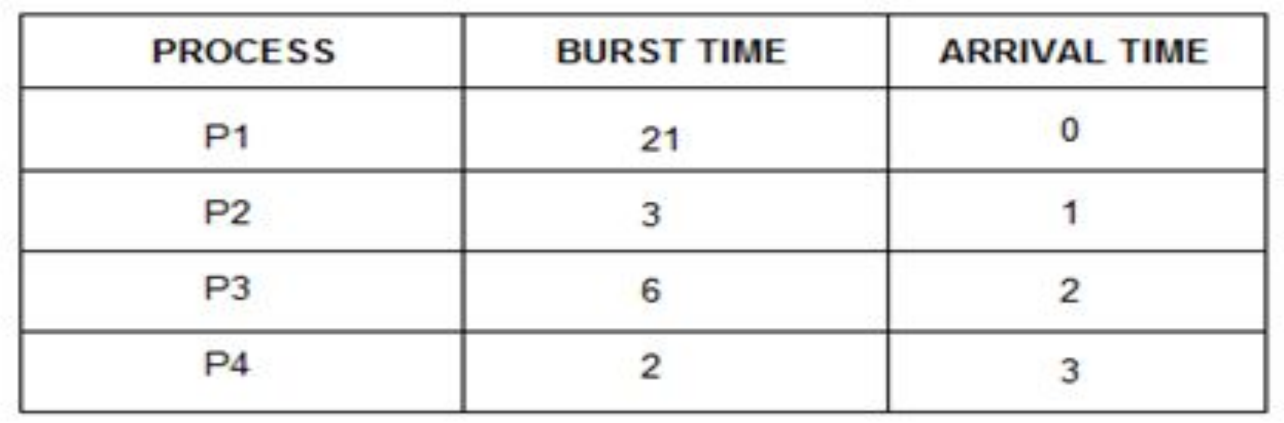
0s 1s 2s 3s 4s 5s

A shortened view of the above time-line is as follows,



Shortest Job First (SJF)

Example-2



PROCESS	BURST TIME	ARRIVAL TIME
P1	21	0
P2	3	1
P3	6	2
P4	2	3

Advantages
Wait time is lower than FCFS.

SJF has minimum average waiting time among all tasks scheduling algorithms.

Disadvantages
Unfairness to some tasks when tasks are assigned to VM, due to the long tasks tending to be left waiting in the task list while small tasks are assigned to VM.

Taking long execution time and TFT.

Min-Min Heuristic

- For each task, determine its minimum completion time over all machines.
- Over all tasks, find the **minimum completion time (min of all mins)**
- Assign that task to the machine that **gives this completion time(min-min)**
- Iterate till all the tasks are scheduled

Max-Min Heuristic

- For each task, determine its minimum completion time over all machines
- Over all tasks, find the **maximum completion time (max of all mins)**
- Assign the task to the machine that gives this completion time
- Iterate till all the tasks are scheduled

Example of Min-Min

	T1	T2	T3
M1	140	20	60
M2	100	100	70

Stage 1:

T1-M2 = 100 (min)

T2-M1 = **20** (min)

T3-M1 = 60 (min)

Assign **T2** to **M1(min-min)**

	T1	T3
M1	140+**20** =160	60+**20** =80
M2	100	70

Stage 2:

T1-M2 = 100 (min)

T3-M2 = **70** (min)

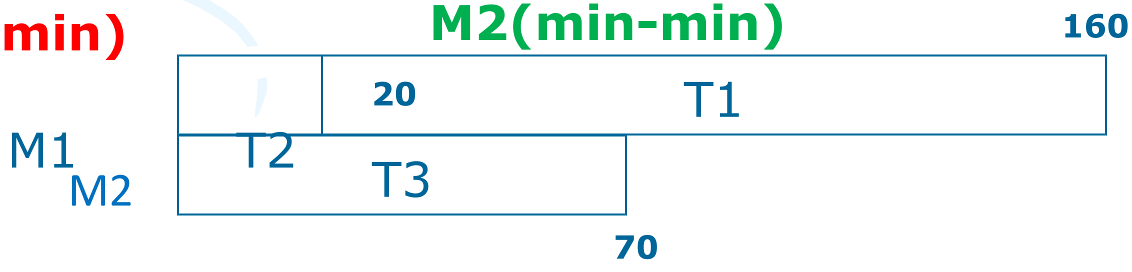
Assign **T3** to **M2(min-min)**

	T1
M1	160
M2	100+**70** =170

Stage 3:

T1-M1 = 160 (min)

Assign T1 to M1



Example of Max-Min

	T1	T2	T3
M1	140	20	60
M2	100	100	70

	T2	T3
M1	20	60
M2	100+**100** =200	70+**100** =170

	T2
M1	20+**60** =80
M2	200

Stage 1:

T1-M2 = **100(min)**

T2-M1 = 20(min)

T3-M1 = 60(min)

Assign **T1** to **M2(max-min)**

Stage 2:

T2-M1 = 20 (min)

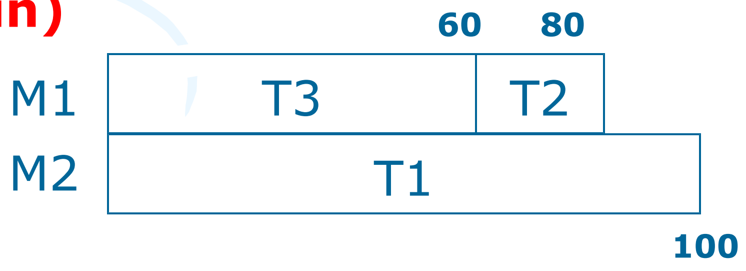
T3-M1 = **60** (min)

Assign **T3** to **M1(max-min)**

Stage 3:

T2-M1 = 80 (min)

Assign T2 to M1


60
80
M1
T3
T2
M2
T1
100