PRIORITY SCHEDULING ALGORITHM

GROUP MEMBER:
SACHIN SHARMA (2021A1R124)
SHIVAM DHAR (2021A1R132)
PREET PAUL SHARMA (2021A1R133)
SUPERB JAIN (2021A1R134)
RANBIR SINGH (2021A1R139)

SUBMITTED TO:
Ms. PRAGTI JAMWAL

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to our operating system lab teacher “Ms. Pragti Jamwal” for their able guidance and support in completing our project.

We came to know about so many new things that we were unaware off.

“PRIORITY SCHEDULING ALGORITHM FOR CPU SCHEDULERS”

➢ ABSTRACT:

In this report we are going to learn about what is priority scheduling algorithm and how it is helpful in CPU scheduling. We will discuss an example of priority scheduling in detail and how schedulers processes on the basis of priority. And also see some of the advantages and disadvantages of priority scheduling. We will discuss about what is static priority and dynamic priority.

➢ INTRODUCTION

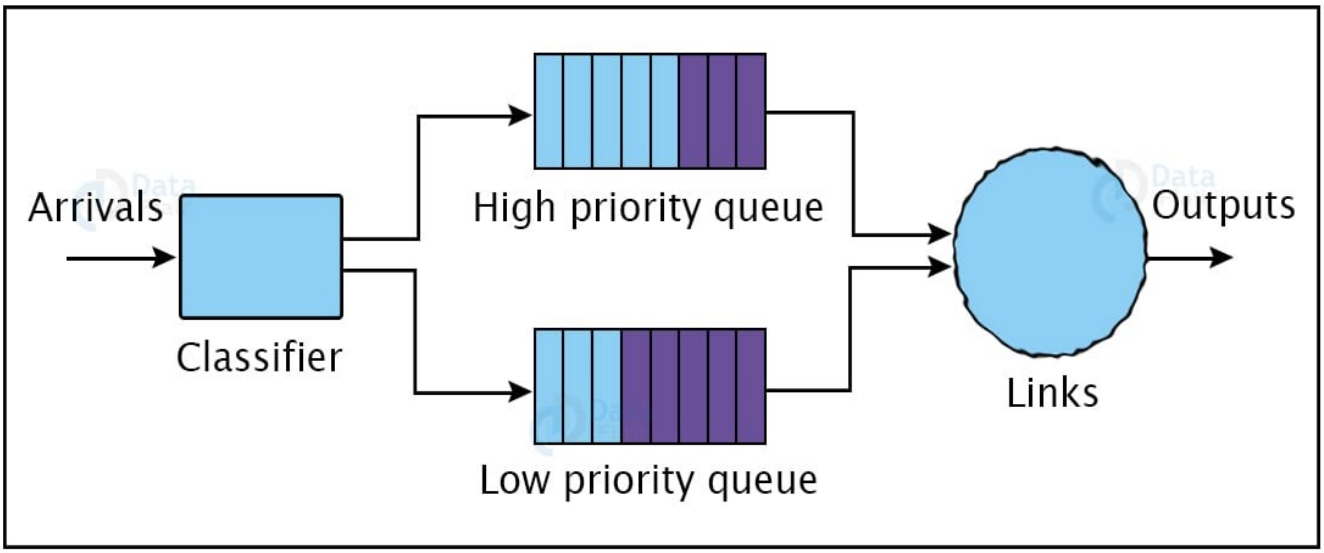
Priority Scheduling is a process scheduling algorithm based on priority where the scheduler selects tasks according to priority. Thus, processes with higher priority execute first followed by processes with lower priorities.

If two jobs have the same priorities then the process that should execute first is chosen on the basis of round-robin or FCFS. Which process should have what priority depends on a process’ memory requirements, time requirements, the ratio of I/O burst to CPU burst, etc.

➢ Types of Priority Scheduling

Following are the two main types of priority scheduling:

1. **Preemptive Scheduling**: Tasks execute according to their priorities. In case a lower priority task is running and a higher priority task arrives in the waiting state then the lower priority task is put on hold. The higher priority task replaces it and once done executing the lower priority task resumes execution from when it was paused. This process requires special hardware like a timer.



2. **Non-Preemptive Scheduling**: The OS allocates the CPU to a specific process that releases the CPU either through context switching or after termination. We can use this method on various hardware platforms because unlike preemptive scheduling it doesn't require any special hardware

.

➢ Characteristics of Priority Scheduling

- Schedules processes on the basis of priority.
- Used to perform batch processes.
- In the case of two processes with similar priorities, we use FCFS and Round-Robin to choose between them.
- A number is given to each process to indicate its priority level.
- Lower is the number assigned, higher is the priority level of a process.

If a higher priority task arrives when a lower priority task is executing, the higher priority task replaces the one with lower priority and the Latter is put on hold until it completes execution.

➢ Example of Priority Scheduling

Following five processes have a unique priority, burst time, and arrival time.

Process	Priority	Burst Time	Arrival Time
P1	1	4	0
P2	2	3	0
P3	1	7	6
P4	3	4	11
P5	2	2	12

Step 0: At time = 0, two processes P1 and P2 arrive. Since P1(with a burst time 4) has higher priority it executes before P2.

Step 1: At time = 1, no new process arrives and execution of P1 continues.

Step 2: At time = 2, no new process arrives, P2 is in the waiting queue, and execution of P1 continues.

Step 3: At time = 3, no new process arrives, P2 is in the waiting queue, and execution of P1 continues.

Step 4: At time = 4, P1 finishes execution, and P2 starts execution.

Step 5: At time = 5, P2 continues execution as no new process arrives.

Step 6: At time = 6, P3 arrives and since it has a higher priority of 1 the OS preempts P2. P3 starts executing.

Process	Priority	Burst Time	Arrival Time
P1	1	4	0
P2	2	1 of 3 pending	0
P3	1	7	6
P4	3	4	11

P5	2	2	12

Step 7: At time = 7, P3 continues executing as no new process arrives and P2 is on hold in the waiting queue.

Step 8: At time = 8, we continue with P3 as no new process arrives.

Step 9: At time = 9, we continue with P3 as no new process arrives.

Step 10: At time = 10, we continue with P3 as no new process arrives.

Step 11: At time = 11, P4 arrives but P3 continues executing as it has a higher priority than P4.

Process	Priority	Burst Time	Arrival Time
P1	1	4	0
P2	2	1 of 3 pending	0
P3	1	7	6
P4	3	4	11
P5	2	2	12

Step 12: At time = 12, P5 arrives but P3 continues executing as it has a higher priority.

Step 13: At time = 13, P3 is done executing. Now there are three processes P2, P4, and P5 in the ready queue. Among these two processes have higher and similar priorities. These are P2 and P5. Since the arrival time of P2 is less than that of P5, P2 starts execution.

Process	Priority	Burst Time	Arrival Time
P1	1	4	0
P2	2	1 of 3 pending	0
P3	1	7	6
P4	3	4	11
P5	2	2	12

Step 14: At time = 14, P2 completes execution. In the waiting state between P4 and P5, P5 has a higher priority so it starts executing.

Step 15: At time = 15, P5 continues executing.

Step 16: At time = 16, P5 completes execution and since P4 is the only process present in the waiting state it starts executing.

Step 17: At time = 20, no process is left for execution as P5 completes execution.

Step 18: Following is the average waiting time: Waiting Time (Wt) = Start Time (St) – Arrival Time (At) + Waiting Time for next burst

P1 = 0 – 0 = 0

P2 = 4 – 0 + 7 = 11

P3 = 6 – 6 = 0

P4 = 16 – 11 = 5

Thus, the average waiting time is: (0+11+0+5+2)/5 = 18/5 = 3.6

➢ Advantages of priority scheduling in OS

Following are the benefits of priority scheduling method:

- Easy to use.
- Processes with higher priority execute first which saves time.
- The importance of each process is precisely defined.
- A good algorithm for applications with fluctuating time and resource requirements

➢ Disadvantages of priority scheduling

Here, are cons/drawbacks of priority scheduling

- If the system eventually crashes, all low priority processes get lost.
- If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time.
- This scheduling algorithm may leave some low priority processes waiting indefinitely.
- A process will be blocked when it is ready to run but has to wait for the CPU because some other process is running currently.
- If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time

➢ Static and Dynamic Priority

When we were learning about priorities being given to proceses, did you ponder about how the priorities were assigned to the process in the first place? Or rather, when was the priority assigned to the process?

In priority based scheduling, we saw that it could be implemented in two ways – preemptive and non-preemptive with the most common implementation being preemptive priority based scheduling. The same way, it can be categorized again on the basis of the method by which the priorities to processes are assigned.

The two classifications are:

- Static priority
- Dynamic priority

The Static Priority algorithm, assigns priorities to processes at design time, and these assigned priorities do not change, they remain constant for the lifetime of that process. However, in the dynamic priority algorithm, the priorities are assigned to the processes at run time and this assignment is based on the execution

parameters of the processes such as upcoming deadlines.

Of course, the static priority algorithms are simpler than the dynamic priority algorithms. There are other scheduling algorithms, like multilevel queues and multilevel feedback queues scheduling which are important for a system that provides better performance.

➢ Summary:

Priority scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority.

In Priority Preemptive Scheduling, the tasks are mostly assigned with their priorities.

In Priority Non-preemptive scheduling method, the CPU has been allocated to a specific process.

Processes are executed on the basis of priority so high priority does not need to wait for long which saves time

If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time.

Priority scheduling in OS is the scheduling algorithm which schedules processes according to the priority assigned to each of the processes. Higher priority processes are executed before lower priority processes.

➢ Future Work:

In this scheduling algorithm, we notice that if high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an infinite time. So, we want to extend our knowledge to solve this problem.

➢ Reference:

1) https://www.scaler.com/topics/operating-system/priority-scheduling-algorithm/

2) https://www.researchgate.net/publication/337332154_Priority_Class_Scheduling

3) https://www.academia.edu/Documents/in/Priority_Packet_Scheduling

4) https://ieeexplore.ieee.org/document/9591967

5) https://www.academia.edu/53078830/Priority_Scheduling_with_Genetic_Programming

➢ Links of GitHub accounts:

1) https://github.com/ranbirsingh-max

2) https://github.com/NOBODY1819/operatingsystem

3) https://github.com/techshivam-18

4) https://github.com/sachinsharma001?tab=stars

5) https://github.com/Superbjain