e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:09/September-2022 Impact Factor- 6.752 www.irjmets.com

COMPREHENSIVE ANALYSIS OF CPU SCHEDULING ALGORITHMS

Dr. Bharathi S*1, Chethan MP*2, Darshan SN*3

*1Professor Of Dr Ambedkar Institute Of Technology, Dept Of MCA,
Bangalore-560060, Karnataka, India.

*2,3Student Of Dr Ambedkar Institute Of Technology, Dept Of MCA,
Bangalore-560060, Karnataka, India.

ABSTRACT

CPU scheduling algorithms play an important role in multiprogramming operating systems. When CPU scheduling is successful, difficult issues can be precisely computed, and the system will continue to function properly. Additionally, CPU scheduling algorithms are the main feature of operating systems that achieve the objective of maximizing CPU use. The purpose of this research is to contrast the features of CPU scheduling techniques. and also discuss its advantages and disadvantages, towards Which algorithm is the most useful for enhancing CPU usage. Numerous scheduling algorithms have been compared, with numerous factors taken into account, including performance, algorithm implementation, flaws, average waiting times, benefits and downsides, allocation criteria, etc. Examining the CPU scheduler in a method that satisfies the scheduling goals is the article's central objective. However, aware of the kinds of algorithm that is most suitable for a specific case by exhibiting its all qualities.

Keywords: Multiprogramming, Turnaround Time, Waiting Time, Throughput, Starvation.

I. INTRODUCTION

Scheduling is a main task of an operating system. The central processor unit's performance is greatly influenced by scheduling methods (CPU), because it controls how the resources are used. There are many algorithms for CPU switching between multiple tasks. The initial goal of scheduling is to verify the equity between processes in the ready queue while increasing throughput and reducing some undesirable aspects like waiting time. Figure 1 shows the process lifecycle [4].

The process attributes and process states are stored in a particular block, that block is called as process control block(PCB). the processes activities and scheduling are controlled by operating system using some CPU scheduling algorithms and PCB[5]. In the scenario of multiprocessing, there is lots of processes are arriving at the same time into main memory. so, there is a need for an efficient scheduling algorithms that will manage all proceses and performance of the system. The most important First come, first served (FCFS), smallest job first (SJF), priority scheduling, and round robin (RR) are the CPU scheduling techniques. [7].

The long term Scheduler determines which processes are to be kept on to get the ready queue so When a user requests the running of a process, it joins the current band of processes that are either approved or suspended by the long-term scheduler. As a result, this scheduler directs which processes should run on a system and directs that the complexity level be handled constantly.

The mid-term scheduler occasionally moves primary memory processes to secondary memory, such as a storage device, from primary memory. This is frequently referred to as processes swapping out or in, while also improperly as processes paging out or in.

Which processes should be performed on the CPU are chosen by the short-term scheduler, commonly known as the CPU scheduler. As well as, it regulates the degree of multi programming for knowing how many processes are in main memory. Preemptive scheduling refers to a schedule that forcibly terminates an active process from the CPU. When the scheduler has been unable to forcefully take the processes off the CPU, it may be non-preemptive.

This paper's goal and inspiration are to fully understand the fundamental CPU scheduling methods. To choose the best algorithm for the processes and system requirements, comparisons between them must take into account a variety of factors. The remainder of the paper is organised as follows: The criteria for CPU scheduling are explained in section 2. Section 3 provides an explanation of scheduling algorithm approaches. Numerous different CPU scheduling strategies are detailed in section 4. The examination of numerous scheduling strategies is reported in section 5. This essay is concluded in Section 6.

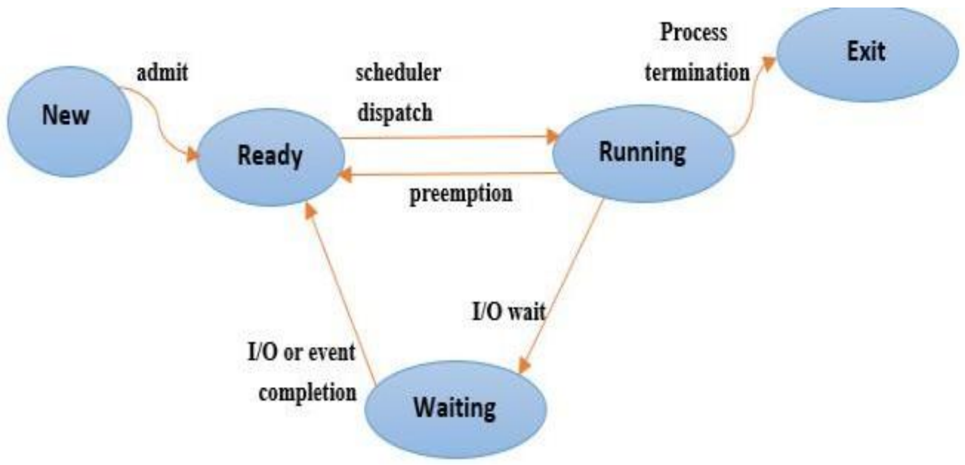

Figure 1. Process life Cycle

II. THE CRITERIA OF CPU SCHEDULING

There are numerous scheduling algorithms, and each one's effectiveness needs to be assessed based on a number of factors. Additionally, each has unique characteristics. Many metrics have been developed to compare CPU scheduling algorithms, some of which are listed below:

- Utilization of CPU: keep the CPU more productive or keep the CPU completely engaged at all times.
- Throughput : is the amount of tasks that are completed in a given amount of time.
- Burst time: Running time of process or the time needed for the execution of process in CPU.
- Completion time: This is the point in time when the process has completed running.
- Turnaround time, It is the total time spent by the process in the main memory.
- Waiting time: The total amount of time a process spent waiting in a line in order to obtain CPU for its execution.
- Time to response: the time taken by the process to give its initial response.
- Equity: making certain that each process receives an equitable share of the CPU.

III. CPU SCHEDULING TECHNIQUES

Preemptive or non-preemptive scheduling techniques are usually used.

A. “Non-preemptive” scheduling

Using a multi-programming system's non-preemptive method, the scheduler allows the process to continue using the CPU until it ends or it wants go for waiting state. In other words, the running process releases the CPU by terminating or switching to the waiting state.

B. Pre-emptive scheduling

In Pre-emptive scheduling, the scheduler allocate the process on to the CPU for a limited time period ,if any interrupts occurs the scheduler will preempt the running process . When a high-priority task enters the ready queue, the currently running process must give the CPU unwillingly.

IV. CPU SCHEDULING ALGORITHMS

When there are multiple executable jobs, scheduling refers to choosing which ones to run. These scheduling techniques can be compared using a variety of metrics. The throughput, turnaround time, and response time are these metrics. In other words, CPU scheduling refers to the process of deciding which task in the queue will receive the CPU's initial allocation.

A. First come, first serve schedule(FCFS)

This scheduling algorithm is non-preemptive. It is a FIFO queue approach because, as implied by the name, any process that comes first will be executed first. Due to the numerous issues associated with this type, subsequent processes that have shorter burst times must wait for a very long time if the process that served first takes too long. The average waiting time will rise as a result of this. Convoy effect is another name for this scenario. Up to its completion or I/O activities, the application does not leave the CPU.

B. Shortest job first, scheduling algorithm (SJFS)

The process chosen in this scheduling technique is the one with the shortest burst time among the incoming processes. because their waiting and turnaround times are kept to a minimal. this scheduling method is comparatively better than the FCFS algorithm . While it has certain limitations, such how difficult it is to determine what is the each processes execution time. this is also one of the non- preemptive type of algorithm. Finally, the most important downside of this type is the process starvation.

C. Longest job first scheduling (LJFS)

This kind of algorithm is the reverse of the SJF algorithm as it is non- preemptive. It works differently because it gives the CPU longer- duration processes first priority over shorter-duration ones. The main benefit is that it is simpler to calculate longer work than shorter jobs. In the engineering field, especially in electro-mechanics, this kind of scheduling is feasible.

D. "Longest remaining time first schedule"(LRFS)

After all of the time intervals have passed, the execution time of every process is calculated and inspected in this kind. Following the completion of a unit, the process with the longest burst duration will be assigned. "Longest job first scheduling" (LJFS) in the preemptive type is another name for it.

E. Shortest job, remaining time first scheduling(SJRF)

In this scheduling algorithm, Based on each process's execution times, the ready queue is created. Preemption causes the process to be split into two divisions, which results in more context switching being produced. After every time unit, the process' burst time is monitored. after completing a unit, check all the arrived process's burst time ,if any processes having minimum burst time then that will be next allocated. in this kind of organizing is also known as the "shortest job first schedule" in preemptive type.

F. Round, Robin(RR)

Standard RR is the name of the preemptive kind of the round robin CPU scheduling method, which allots a period of time known as Quantum Time (TQ). When the TQ is finished, the active process is preempted and moved to the back of the ready queue. Because it allows for an average share of time for each activity, makes the most of the CPU, and provides a rapid response, RR is frequently used in real-time and time-sharing operating systems. Additionally, the Standard RR method has numerous flaws, including low throughput, lengthy turnaround times, and lengthy waiting times. Additionally, there are a lot of context transitions. The Quantum Time is the most crucial component of the RR algorithm. On the one hand, a low QT causes a lot of context shifts, which lowers CPU performance. However, placing a huge QT can result in a slow response time. as well as it can degrade to FCFS.

G. "Priority scheduling" for preemptive and non-preemptive techniques

This priority scheduling technique categorises processes according to a priority based on the type of processed data. A series of measures determine the process's priority, and each process that adds itself to the ready queue does so by prioritising its "importance". Only such priority number determines which process will receive the CPU allocation so that the "high value" priority will be sent to the CPU first or last. There are two different kinds of techniques for this algorithm: preemptive and non- preemptive. When a high priority process keeps entering the ready queue in the preemptive type of this kind of method, the lowest priority processes might starve .

H. Multilevel queue, scheduling Algorithm(MLQ)

Depending on where they are, the processes might be divided into various segments. For example, the process can be categorized into three types, such as system process, foreground process and background process. among these the system process has been given to the highest importance , next importance is foreground process later background process. Varying expectations and scheduling constraints are present in these process situations. As a result, the ready queue gets split up into different queues, and each queue has its own scheduling mechanism. For example, the one queue will have Roundrobin Scheduling and the another queue will have Fcfs scheduling type. Major priority processes are often positioned at the front of the ready queue stage, whereas lower - priority processes are positioned at the bottom of the ready queue level. When this strategy is used, the processes there at lower level of the ready queue experience a starving issue.

The state diagram for "multilevel queue scheduling (MLQ)" is shown in Figure2.

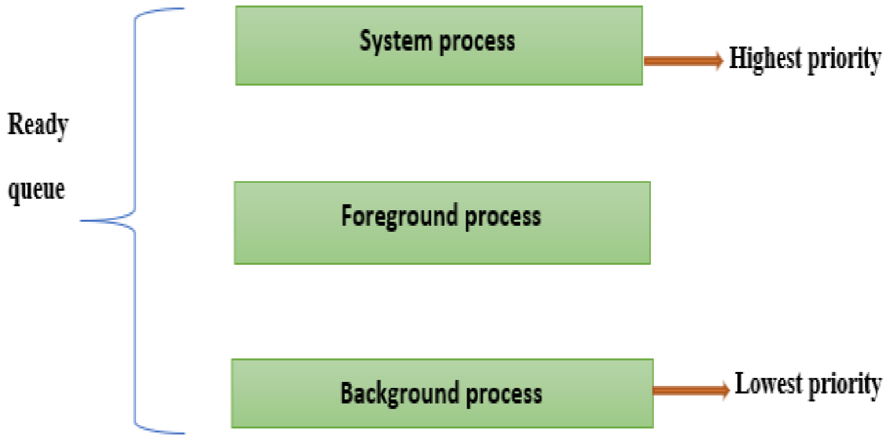

Figure 2. Multi level queue

V. MULTILEVEL "FEEDBACK"QUEUE SCHEDULING ALGORITHM(MLFQ)

This scheduling idea is almost same as multilevel queue scheduling, In order to solve the starving issue, processes which do not complete their operation at the top level are preempted and moved to the next phase of the ready queue. The main issues with this method of scheduling, however, include firstly: "How to calculate and pick the best queue numbers for scheduling?," Second, "how long is the quantum time for every queue?," and third, "how is the priority set for each Process?". Therefore, there will not be any starving. For interactive jobs, major priority queues are often assigned to tiny quantum, and lower - priority queues have been assigned to long-time quantum. As a result, the duration of QT varies between several queues. The MLFQ demands that the system become interactive in order to reduce reaction time, and this technique attempts to minimise the intermediate turnaround time.

VI. CONTRAST BETWEEN DIFFERENT SCHEDULING ALGORITHMS

The following Tables 1-3, which include a number of notable criteria connected to each algorithm, are used to explain all the properties of the scheduling algorithms listed above.

Table 1. Algorithms preemptive and Performance analysis

No	Algorithms	Preemption	Performance
1	FCFS	no	Performance is Slow.
2	SJF	no	Average Waiting Time is minimum.
3	LJFS	no	Turn-around time is more.
4	LRTF	yes	The preference is given to the longer burst time.
5	SRTF	yes	The preference is given to the shorter burst time.
6	RR	no	A fixed time slice is given to each processes.
7	PR preemptive	yes	Good performance but it has a starvation problem.
8	PR non-preemptive	no	Most useful with batch system.
9	MLQ	no	Performance is good but contains a starvation problem.
10	MFLQ	no	Performance is fast.

Table 2. Al location Criteria

No.	Algorithms	Allocations
1	FCFS	The scheduler will allocates the processes to CPU in which order they arrived.
2	SJF	Whichever processes having lowest burst time will allocate first.
3	LJFS	Processes having highest CPU execution time will allocate first.
4	LRTF	It is same as LJFS but it is in preemptive type.
5	SRTF	Minimum CPU burst time processes have allocated initially.
6	R R	Order of arrival time of processes but time slice is given to all processes.
7	PR preemptive	Based on the priority, whichever having highest priority.
8	PR “non-Preemptive”	According to priority, but it is in non-preemptive technique. whichever having higher priority will allocated first.
9	“MLQ”,	System processes have higher priority, also depends upon the bigger priority queue.
10	“MLFQ”,	Based on the process of a “higher priority queue”.

Table 3. Algorithms Implementation, Strengths and Weaknesses

Algorithms	Implementation	Strengths	Weaknesses
FCFS.	Very easy toimplement.	The executionof process is done in whichorder they are arrived.	1- Minimum throughput. 2- It may lead toconvoy effect.
SJF.	Implementation is complex than the FCFS.	1- Minimum waiting time in non- preemptive systems. 2- I/O bound jobs have higher prioritythan the CPU bound jobs.	It is not practically implementable asCPU burst time is not known earlier.
LJFS.	Easy implementation, as it is oppositeof SJF.	Easier to implement.	It Dominates the CPU.
SRTF.	It is difficult toapply in interactive system.	Minimum waiting time as compared to all the algorithms.	Problem of starvationoccurs.
RR.	Quantum time plays the most significant role.	1- CPU is shared to all the processes equally. 2.It eliminates the problem ofstarvation.	1- Tough to maintain Quantum time. 2- There is a chance that RR can be degrade to FCFS ifTQ is more.
Priority preemptive .	Difficult as compared to non-preemptivetype.	Waiting timestep by step growths for “ equal priorityProcess”.	Starvation may occurs.
“Priority(non-	Implementationis	Higher priority processes	Become complex when larger

preemptive)".	medium as compared to other algorithms.	are completed muchfaster.	processesgot high priority.
"MLQ",	It is difficult to understand.	Processes areallocated permanently into the queue.	Suffers from starvation..
"MLFQ",	Implementationis complicated as compared to MLQ.	Problem of starvation may solved.	Extra context switching is required.

switching. According to the attributes of the specified algorithms in the tables, choosing the optimal scheduling algorithm relies on the scenario and the type of system. however, many things that may occur in upcoming days for obtaining the best performance with minimum cost. Further researches may find the supreme scheduling algorithms and provide a best possible solution in this area.

VII. CONCLUSION

One of the important functions of Operating system is CPU scheduling. Response time, average wait time, turn - around time, and throughput are all crucial scheduling characteristics that influence the choice of scheduling algorithms. moreover, Consider all the other elements as playing a significant role in execution. In this study, ten key properties of scheduling algorithms are described and addressed. Additionally, a quick examination of eight significant evaluation metrics and criteria that fall within the ten algorithm types mentioned above is provided. For example, For each type of algorithm, efficiency, implementations, strengths, flaws, and other criteria are shown.. The results of this study show that there are numerous ways for CPU scheduling, but none of them are perfect in all the dimensions. Because each algorithms have its own flaws. For instance, Fcfs has convoy effect, Rr has a maximum "average waiting time". nevertheless, SJF, LJFS, LRTF, and SRTF may suffer from issue of starvation. MLFQ requires more context.

VIII. REFERENCES

[1] M. R. Reddy, V. V. D. S. S. Ganesh, S. Lakshmi and Y. Sireesha, "Comparative Analysis of CPU Scheduling Algorithms and Their Optimal Solutions," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), 2019, pp. 255-260, doi: 10.1109/ICCMC.2019.8819679.

[2] Goel, Neetu, and R. B. Garg. "A comparative study of cpu scheduling algorithms." arXiv preprint arXiv:1307.4165 (2013).

[3] Singh, Pushpraj, Vinod Singh, and Anjani Pandey. "Analysis and comparison of CPU scheduling algorithms." International Journal of Emerging Technology and Advanced Engineering 4.1 (2014): 91-95.

[4] S. B. Bandarupalli, N. P. Nutulapati and P. S. Varma, "A novel 'CPU' Scheduling Algorithm, Preemptive & Non Preemptive," International Journal of Modern Engineering Research ("IJMER"), vol. 2, no. 6, pp. 4484-4490, Nov- Dec 2012.

[5] Omar, Hoger K., Kamal H. Jihad, and Shalau F. Hussein. "Comparative analysis of the essential CPU scheduling algorithms." Bulletin of Electrical Engineering and Informatics 10.5 (2021): 2742-2750.

[6] Qureshi, Imran. "Cpu scheduling algorithms: A survey." International Journal of Advanced Networking and Applications 5.4 (2014): 1968.

[7] Ali, Shahad M., et al. "A Review on the CPU Scheduling Algorithms: Comparative Study." International Journal of Computer Science & Network Security 21.1 (2021): 19-26.

[8] Yousra Ahmed Fadil, "CPU SCHEDULING SIMULATION", DJES, vol. 2, no. 2, pp. 39–52, Dec. 2009.(IEEE).