© 2017 IJSRST | Volume 3 | Issue 8 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

Customary Methods for CPU Scheduling : A Review

Badal Dave[1], Surendra Yadav[2], Manish Mathuria[3]

[1]*M.Tech Scholar, Department of Computer Science & Engineering, MACERC, Jaipur, Rajasthan, India*
[2] *Prof., Department of Computer Science, Career Point University, Kota, Rajasthan, India*
[3]*Asst. Prof., Department of Computer Science & Engineering, MACERC, Jaipur, Rajasthan, India*

ABSTRACT

Among variety of process the scheduling is one of the most important decisions making practice of an operating system for sharing machine resources in between the several of executable process which persist in memory. Typically, scheduling is a decision making task of an operating system to decide which process will get system resources to complete process in an optimum way. For scheduling a number of approaches have introduced by number of investigators and a lot of works are in progress with optimizing the issues of existing algorithms. This paper deals with the exploration of different accessible scheduling procedures to recognize high efficient algorithm which suits the scheduling goals, minimize average turnaround and waiting time in order to allow as many as possible running processes at all time in order to make best use of CPU.

Keywords: CPU, Scheduling, Operating System, Scheduling Algorithms.

I. INTRODUCTION

In modern digital era, scheduling is a hot research term in the world of real time systems. However, different forms of this technique have been used in dissimilar fields over the past era but with speedy technological escalation it has attains a dramatic progress and became as an essential component of any real time system in present time. Typically, scheduling is a decision making technique that deals with the problem of sharing system resources over the numerous process in an optimum way [1]. A system QOS and optimum resource sharing highly depends on the efficiency of scheduling techniques that has been used in a system at the time of process execution. This practice can be further classified into two main root categories [2] i.e. (i) Static scheduling (ii) Dynamic scheduling. Static scheduling has done at the time of program compilation, tasks has been scheduled before starting the real execution of program and have no option to reschedule the order of tasks or add new tasks at run time [3]. However, due to some rewards this technique confirms its efficiency over the other accessible practice but has suffers due to some of its limitation. Unlike to static scheduling scheme, the technique of dynamic scheduling schedules the task upon their arrival i.e. at run time. Dynamic scheduling is akin to a data flow machine, in which tasks don't scheduled based on the order in which they appear, but manage in a parallel form on base of their arrival time with some dynamic parameters that may change during run time. Apart of unique advantage to manage tasks at run time this technique does not require any recompilation process after adding of new task that would not be visible with compile time scheduling scheme [4]. Following figure has illustrated the basic states of a process.

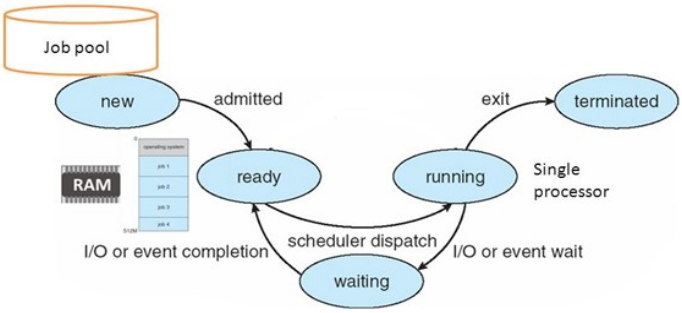

Figure 1. Process State Diagram

Different state of process can be point out as

- New State: - The born stage of process, actuality generated in system.
- Ready State: - Waiting for CPU time, ready for execution but currently not in processing state of employed system.
- Running State: - Process Executed at CPU.
- Waiting State: - Process desires some I/O resource to complete its task.
- Termination State: - Last stage at a process finished its execution.

II. SCHEDULING CRITERION & OBJECTIVES

For an effective performance of system it is must that an employed algorithm has fulfilled above criteria. The employed technique must utilize complete system resources in a good and efficient way with trim down issues of requiring turnaround, waiting and response time. Mostly performance of a system measured on the base of an average case. In literature of an task scheduling a huge amount of investigators have also express in their investigations that minimizing the variance in response time is more imperative in comparisons of minimize the average response time. According to them a system with sensible and expected response time may be considered more attractive than the system that is quicker on the average but is extremely changeable. The main goals of scheduling algorithms can be illustrates as [5].

- **CPU Utilization: -** Keep the CPU busy with useful work as possible for maximum performance.
- **Throughput: -** Enhance number of complete jobs.
- **Turnaround Time: -** Reduce Waiting Time. The amount of time to execute a particular process. It must have minimum value.
- **Waiting time: –** The amount of time a process has been waiting in the ready queue. It must have minimum. From the time of submission to the time of completion, minimize the time batch users must wait for output.
- **Response Time: -** Time from submission till the first response is produced, minimize response time for interactive users. The amount of time it takes from when a request was submitted until the first response is produced, not output (for time-sharing environment). It must have minimum value.
- **Fairness:** Make sure each process gets a fair share of the CPU.

III. HANDY PROCEDURES FOR SCHEDULING

Over past few decades, a good amount of investigators has put their efforts in direction to trim down speed issues in real time system with the concepts of process scheduling technique. There are many different CPU scheduling algorithms. Figure 2 point up some of the popular scheduling algorithms.

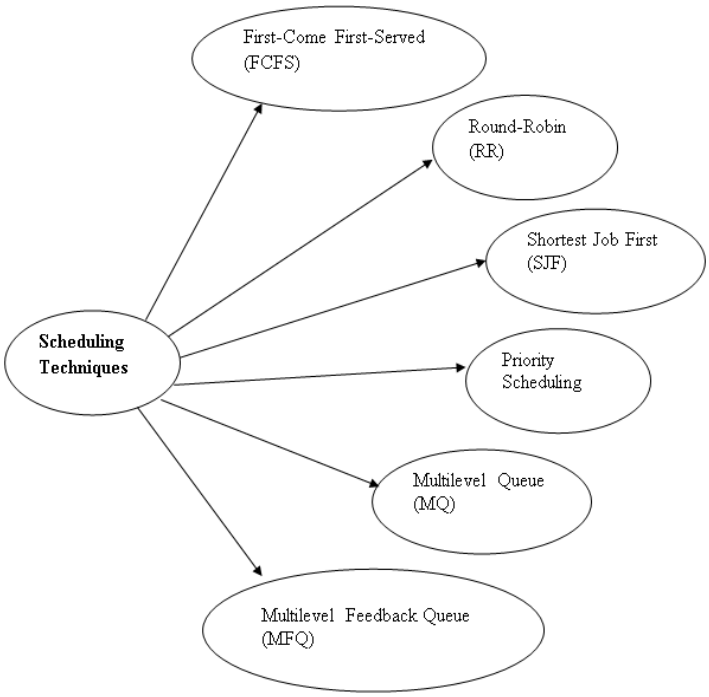

Figure 2. Handy Scheduling Techniques

A. First-Come First-Serve Scheduling (FCFS)

The First-Come-First-Served (FCFS) scheduling scheme is one of the simplest methods among available scheduling solutions [6]. In this scheme process information has store into a queue and has executed in order as they arrives in the system, the process that comes prior to other has executed firstly in comparison of late coming process. Innovative process added at the end position of queue and leaves through the head of the queue. After completion of process it removes from the queue and next task of queue take head position. This is a non-primitive approach, once the process has start execution it will only stop after completing it functionalities. There is no scope to free up or interrupts system resources before completing the functionalities of executed process. Some of the main advantages of FCFS Technique are

- Uncomplicated Scheduling Technique.
- Low Overhead.
- No starvation of system resources.
- Much Better for Long Process.

However, the concepts and working procedure of such scheduling algorithm is simple and easy to understand but this technique associates a number of limitations [7, 8], such as

- Average waiting time is often quite long.
- Non-preemptive Approach.
- Underprivileged performance, low throughout.

B. ROUND ROBIN (RR)

This scheduling technique improve the shortcomings of preceding algorithm i.e. FCFS. Like to working methodology of FCFS this technique has also maintain a queue in FIFO order that consist the process information and executes each process without considering the execution priority of an process but have a dissimilar way of process dispensation. The technique has assigned a fix time interval i.e. quantum time for each job and executes them in a circular order without considering the process priority. Therefore this technique mostly used on timeshared systems. In the execution state if a process complete it execution within allocated time period than system take next process in an account but if an active process has not yet finished its execution than it added at the end of queue for next quantum time turn. However technique improves the performance of scheduling over the FCFS algorithm but have major issue to set the efficient quantum time length [9, 10]. Some of the advantages of this technique can be point out as

- A Simple and fair technique that distributes system resources for equally time frame among the process.
- Recent arrived process added at the end position of queue with assigning a time quantum.
- Starvation free technique.

Limitation of RR approach can be depicts as

- Due to high context switching there is larger waiting time and Response time.
- With the same issue i.e. high context switching, technique present low throughput.
- Degrades performance with setting quantum time too short, main cause of high context switches.
- Due to set higher quantum time range technique produce poor response time, approximates to FCFS technique.

C. SHORTEST JOB FIRST (SJF)

This technique has different approach for the process execution. For minimizing process response time issues this technique has select low execution time process to execute in system priory in comparison of other available process. After completion of each process the technique has again select the next process from ready queue that have required low process time of system and its resources to complete it functionalities. For maintain the process information this technique has also maintain a queue that contain the information of arrived process in an sorted order according to the process execution time requirement so that short programs get to run first and not be held up by long ones. If in the ready queue two processes have the same system time requirement than technique use the concepts of FCFS scheme to select an appropriate process for execution and to safe from starvation. This mechanism has also some unique advantages as

- Easy to implement in system because it does not give exacting notice on the process deadlines. A developer can only attempt to make processes with deadlines as short as possible.
- Have low waiting & turnaround time in comparison of FCFS technique.
- Maximizes process throughput

Limitation of SJF approach can be depicts as

- This technique has mandatory requirement to maintain advance knowledge of process execution time, which may not be feasible for all processes in real time.
- Process with requirement of high burst time wait longer than in FCFS.

D. PRIORITY SCHEDULING

In this algorithm, priority is associated with each process and on the basis of that priority CPU is allocated to the processes. Higher priority processes are executed first and lower priority processes are executed at the end. The low priority is interrupted when the high priority

processes are arrived. If multiple processes having the same priorities are ready to execute, control of CPU is assigned to these processes on the basis of FCFS. Priority Scheduling can be pre-emptive and non-pre-emptive in nature. In pre-emptive priority scheduling the algorithm pre-empt the CPU if the priority of the newly arrived process is higher than the priority of the currently running process. A non-pre-emptive priority scheduling algorithm will simply put the new process at the head of the ready queue. A major problem with priority scheduling algorithms is indefinite blocking, or starvation. A process that is ready to run but waiting for the CPU can be considered blocked. A priority scheduling algorithm can leave some low- priority processes waiting indefinitely. In a heavily loaded computer system, a steady stream of higher-priority processes can prevent a low-priority process from ever getting the CPU. Generally, one of two things will happen. Either the process will eventually be run, or the computer system will eventually crash and lose all unfinished low- priority processes. In [11] a priority scheduling algorithm are describes, the process are schedule based on their antecedence rate and allocate to processor equating with the subsisting programming algorithm based on its duration and resource employment. Some advantages of pre-emptive approach are

- Technique is more appropriate for an environment where process time and resource requirements are varying.
- Simple to implement.

Limitation of pre-emptive approach can be depicts as

- Starvation issues, low priority processes may never execute or high waits for execution.
- It is difficult to validate that all jobs scheduled in a priority-driven manner meet their deadlines when the job parameters vary.

E. MULTILEVEL QUEUE (MQ)

This scheduling technique is one of the most popular solutions for managing variety of processes in system. The working mechanism of this technique separate processes into multiple priority queues. On the base of processes response-time requirements and scheduling needs this scheme maintain foreground or can say interactive processes queue and background or can say batch processes queue. Each queue scheduled with the unique scheduling algorithm for example foreground queue uses Round Robin scheduling, while background uses FCFS scheduling technique. Foreground processes have high priority over background processes [12]. However, this scheme gives preference to process that has required low system time and I/O bound with the restricted scheduling between each queue, higher priority queues must be empty for processes in lower to run. Advantage of this approach can be exemplify as

- Consider processes with distinctiveness burst time.
- Scheduler can assign the priority level on the base of resource requirement, may assign higher priority classes with larger time slice to the process that require resource related to I/O bound.

Some of associated limitation of this approach is

- Often low priority tasks highly wait for execution because they could not run unless all the higher priority queues not go empty.
- Apriori assignment necessity of class to process is not a most efficient way to do things.

F. MULTILEVEL FEEDBACK QUEUE (MFQ)

The mechanism of MFQ is an extraordinary case or can say adaptive version of scheduling algorithm. Unlike Multilevel Queue technique that has assigned process into a queue everlastingly the mechanism of MFQ allows the scheduler to adjust the priority of a process during execution in order to move it from one queue to another based on the process recent behavior. In simple words MFQ is an extended version of MQ scheduling technique that has allows process to move from a priority queue to another queue at execution time i.e. dynamically on the base of process activity [13]. Advantage of this mechanism is

- Employ dynamic priority mechanism,
- Reduce overheads by quickly complete process.

Some of associated limitation of this technique is

- Technique may face a starvation issue in case if new jobs are continuously arriving in the system.

IV. HITCHES WITH HANDY PROCEDURES

First come first serve scheduling algorithm (FCFS) is one of the simple and easy to working algorithm in comparison of all other classical algorithm. However technique is fit for batch processing system where waiting time is large but not most suitable for an interactive systems and face huge waiting time issues if low burst time process arrives after the requested process that require long execution time in system. Another process scheduling practice known as Round Robin algorithm has trim down the issues of FCFS technique. However, the process execution of this technique is same as the FCFS scheme but has process preemption functionality by using the concept of quantum time, allocates a fixed quantum time for each process in cycled form to complete it working within system. Therefore this technique is most suitable for time sharing systems. Shortest Job First Scheduling (SJF) technique trims down the issues of average waiting time by executing process in a way that processes which requires low system time for its execution has executed priory in comparison of others. Like the name one of popular scheduling technique, priority scheduling algorithm has executed system process in order of priorities. This scheme face a starvation issue due to start the execution with higher priority associated process therefore lowest priority job highly waits in queue for execution. This inadequacy of technique makes it unsuitable for interactive system. In multilevel queue scheduling, processes are permanently assigned to a queue depending upon its nature and no process in the lower priority queue could run unless the higher priority queues were empty. Also, it is preemptive in nature. Multilevel feedback queue scheduling is also preemptive in nature and it allows the processes to move between the queues depending upon the given time quantum.

V. CONCLUSION

This paper deals with an investigation over the handy methodology of CPU scheduling. Instead of simple illustration of available methods this paper focus on finding an associated issues of each and every technique which may helpful for new comers researchers to understand related issues in easy way and encourage them to design more effective method to overcome such issues and enhance the power of employed system.

VI. REFERENCES

[1] V. SureshBabu, and D.Prabakar "Circular Wait Scheduling Scheme for Wireless Sensor Network" International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE) ISSN: 0976-1353 Volume 7 Issue 1 –March 2014.

[2] Haluk T, Salim H, MinYou Wu, "Performance Effective and Low-Complexity Task Scheduling for Heterogeneous Computing", IEEE Transactions on Parallel and Distributed Systems, Vol. 13, 2002.

[3] M. I. Daoud, N. Kharma, "A High Performance Algorithm for Static Task Scheduling in Heterogeneous Distributed Computing Systems", Journal of Parallel and Distributed Computing, pp. 399-409, 2007.

[4] Kamolov Nizomiddin Baxodirjonovich, Tae-Young Choe "Dynamic Task Scheduling Algorithm based on Ant Colony Scheme" International Journal of Engineering and Technology (IJET), Vol 7 No 4 Aug-Sep 2015.

[5] Reena Sharma, Antima Saxena "Processes Scheduling & Accessible Scheduling Algorithms: A Review" International Research Journal of Engineering and Technology (IRJET)**,** Volume: 03 Issue: 09 | Sep-2016

[6] Doaa M. Abdelkader, Fatma Omara "Dynamic task scheduling algorithm with load balancing for heterogeneous computing system", Egyptian Informatics Journal, Production and hosting by Elsevier (2012) 13, 135–145

[7] Saleem, U., and Muhammad Younus Javed. Simulation of CPU scheduling algorithms. In TENCON. Proceedings, vol. 2, pp. 562-567. IEEE, 2000.

[8] Jyotirmay Patel, A.K.Solanki. CPU Scheduling: A Comparative Study. .Proceedings of the 5th National Conference, INDIACom, Computing For Nation Development, March 10-11, 2011 Bharti Vidyapeeth's Institute of Computer Applications and Management, New Delhi.

[9] Masoud Nosrati, Ronak Karimi, Mehdi Hariri "Task Scheduling Algorithms Introduction" World Applied Programming, Vol (2), Issue (6), pp. 394–398, June 2012.

[10] Abbas Noon1, Ali Kalakech2, Seifedine Kadry "A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average" May 2011, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1. ISSN (Online): 1694-0814(pp)

[11] Weifeng Sun, Yudan Zhu; Zhiyuan Su, Dong Jiao, Mingchu Li, "A Priority-Based Task Scheduling Algorithm in Grid" Parallel Architectures, Algorithms and Programming (PAAP), 2010 Third International Symposium on , vol., no., pp.311,315,18-20 , doi: 10.1109/PAAP.2010.24.

[12] G. Siva Nageswara Rao, N. Srinivasu, S.V.N. Srinivasu, G. Rama Koteswara Rao, "Dynamic Time Slice Calculation for Round Robin ProcessScheduling Using NOC", International Journal of Electrical and Computer Engineering (IJECE) Vol. 5, No. 6, December 2015, pp. 1480~1485 ISSN: 2088-8708.

[13] Becchetti, L., Leonardi, S. and Marchetti S.A., "Average-Case and Smoothed Competitive Analysis of the Multilevel Feedback Algorithm" 2006, Mathematics of Operation Research Vol. 31, No. pp. 85–108.(pp)