RESEARCH ARTICLE OPEN ACCESS

A Comparative Analysis of Various CPU Scheduling Algorithms

Navjot Singh*, Prabhjeet Kaur **, Baljeet Kaur ***, Amandeep Kaur ****
*(Department Of Computer Science Engineering, BFCET, Bathinda- Punjab
** (Department Of Electronics & Communication Engineering, BFCET, Bathinda- Punjab
*** (Department Of Electronics & Communication Engineering, BFCET, Bathinda- Punjab
**** (Department Of Electronics & Communication Engineering, BFCET, Bathinda- Punjab
Corresponding Auther : Navjot Singh

ABSTRACT
Process Scheduling Is A Basic Operating System Function Which Decides The Time And Order In Which The Multiple Designated Processes Have To Be Executed. Scheduling Is Must For The Proper Utilization Of A Processor In Multiprogramming Or Parallel Processing Environment. Several Algorithms Such As First-Come-First-Serve (FCFS), Shortest Job First (SJF), Priority Based, Round Robin, Multilevel Queue And Multilevel Feedback Queue Have Been Proposed To Solve The Scheduling Problem I.E. That Which Task Or A Particular Process Is To Be Allocated To The Processor. It Becomes Very Important To Meet The Operating System Design Goals. In This Paper, We Have Reviewed The Basic Scheduling Algorithms And Explored That Which Algorithm Is Best For A Particular Situation.
Keywords - Operating System, Process, Scheduling, Round Robin

---

Date of Submission: 10-04-2018 Date of acceptance: 24-04-2018

I. INTRODUCTION

Scheduling Is A Fundamental Operating-System Function. There Is A Great Demand For High Speed Computing. So, All The Computer Resources Are Scheduled Before Use. Thus, Their Scheduling Is Central Of Operating System Design. Parallel Computing Is A Form Of Computation In Which Many Jobs Are Carried Out Simultaneously [1]. The Aim Of CPU Scheduling Is To Minimize The Completion Time Of A Parallel Jobs By Properly Allocating Them To The Processors And Also Sequencing The Execution Of The Tasks. When More Than One Process Is Placed In The Ready Queue, The OS Must Decide Which One Is To Run First. That Parts Of The OS Concerned With This Decision Is Called "Scheduler" And The Algorithm It Uses Is Called "Scheduling Algorithm". A CPU Scheduler Is Responsible For Arbitrating Access To The CPU [2]. A Scheduler Selects The Next Job To Be Admitted Into The System And The Next Process To Run. The Objective Is To Schedule N Jobs On The Single Machine Such That A Given Measure Of Performance Is Minimized. The Jobs May Be Independent Or Dependent. Every Processor Needs Operating System To Hide Its Details And Manage Its Resources. The Need For A Scheduling Algorithm Arises From The Requirement For Most Modern Systems To Perform Multitasking And Multiplexing. An OS Has Three Types Of Schedulers:

1. Long-Term Scheduler: The Long-Term Or Job Scheduler Decides Which Jobs Or Processes Are To Be Admitted To The Ready Queue; That Is, When An Attempt Is Made To Execute A Process, Either It Is Authorized To Execute Or Get Delayed By The Long-Term Scheduler [3]. Thus, This Scheduler Dictates What Processes Are To Run On A System And Controls The Degree Of Concurrency At Any Time I.E. Whether A High Or Low Amount Of Processes Are To Be Executed Concurrently, And How The Split Between Input/Output & CPU Intensive Processes. A Long-Term Scheduler Is Responsible For Controlling The Degree Of Multiprogramming. In Modern Operating Systems, Issue Is To Make Sure That Real Time Processes Get Enough CPU Time To Finish Their Tasks. Without Proper Real Time Scheduling, Modern Guis Would Seem Sluggish. The Long-Term Queue Exists In The Hard Disk Or The "Virtual Memory".

2. Medium-Term Scheduler: The Medium-Term Scheduler May Decide To Swap Out A Process Which Has Not Been Active For Some Time, Has A Low Priority, Or Taking Up A Large Amount Of Memory In Order To Free Up Main Memory For Other Processes. That Process Is Allocated Later On. This Process Is Often Called As "Swapping".

3. Short-Term Scheduler: The Short-Term Scheduler (CPU Scheduler) Selects A Process From Pool Of Process Resident In Memory That Are Ready To Execute And Allocates The CPU To One Of Them. Thus, The Short-Term Scheduler Makes Scheduling Decisions Much More Frequent Than

The Long-Term Or Mid-Term Schedulers. A Short-Term Scheduler Can Be Preemptive Or Non-Preemptive.

1.1 Terminologies Used In Scheduling Algorithms:

1. CPU Utilization*:* It Is The Average Fraction Of Time During Which The Processor Is Kept Busy [4]. It Is Measured In Clock Ticks/Seconds. It Is Useful To Measure CPU Time As A Percentage Of The CPU's Capacity, Which Is Called The CPU Usage.
2. Throughput: It Refers To The Amount Of Work Completed Per Unit Of Time Or Processes Executed In Specific Time. The Higher The Number Of Processes Executed, Higher Is The Throughput.
3. Waiting Time: The Average Period Of Time A Process Spends In Waiting Due To Unavailability Of Required Resources.
4. Turnaround Time: The Interval From The Time Of Submission Of A Process To The Time Of Completion Is The Turnaround Time.
5. Response Time: Response Time Is The Time From Submission Of A Request Until The First Response Is Produced.
6. Priority: Each Process Is Assigned A Unique Priority And Given The Preferential Treatment Based Upon This Priority. Lesser Will Be Number, Higher Will Be Priority.
7. Fairness: Avoid The Process From Starvation. All The Processes Must Be Given Equal Opportunity To Execute [5].

1.2 Overview Of Existing CPU Scheduling Algorithms

1. First Come First Serve Scheduling: The Most Intuitive And Simplest Technique Is To Allow The First Process Submitted To Run First. This Approach Is Called As First-Come, First-Served (FCFS) Scheduling. In Effect, Processes Are Inserted Into The Tail Of A Queue When They Are Submitted. The Next Process Is Taken From The Head Of The Queue When Each Finish Running. Such A Technique Is Fair In The Case Of Smaller Processes But Is Quite Unfair For Long Ones. FCFS Does Not Involve Context Switching And Has Minimal Overhead. But May Results In High Waiting Time, Turnaround Time And Response Time.

2. Shortest Job First Scheduling: The Criteria Of This Algorithm Are Which Process Having The Smallest CPU Burst, CPU Is Assigned To That Process Next [6]. If Two Processes Having The Same CPU Burst Time FCFS Is Used To Break Up The Tie. A Scheduler Arranges The Processes With The Least Burst Time In Head Of The Queue And Longest Burst Time In Tail Of The Queue. SJF Can Be Worked As Preemptive And Non– Preemptive In Nature Based On The Arrival Time And Burst Time Of The Processes. SJF Reduces Average Waiting Time Of The Processes As Compared To FCFS.

3. Priority Based Scheduling: In This Algorithm, Priority Is Associated With Each Process And On The Basis Of That Priority CPU Is Allocated To The Processes. Lower Priority Processes Get Interrupted By Incoming Higher Priority Processes. It Provides A Good Mechanism Where The Relative Importance Of Each Process May Be Well Defined But Lead To The *"Starvation"*. Priority Algorithm Can Be Also Implemented As Preemptive And Non– Preemptive [7].

4. Round Robin Scheduling: It Is A Preemptive Scheduling Algorithm. It Is Designed Especially For Time Sharing Systems. In This Algorithm, A Fixed Execution Time Called As The Time Quantum Is Assigned To Each Process. When The Time Quantum Expired, The CPU Is Switched To Another Process. Performance Of This Algorithm Is Totally Depending On The Size Of The Predefined Time Quantum [8].

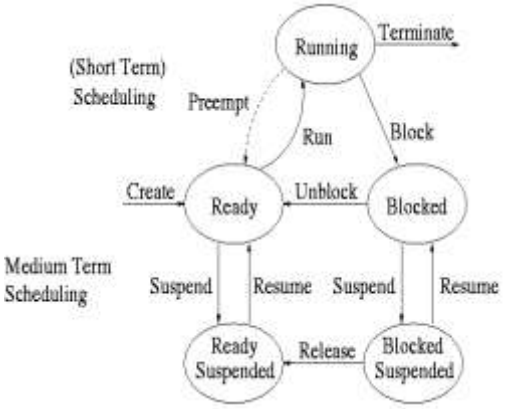

Fig.1 CPU Scheduling

5. Multilevel Queue Scheduling*:* First Composite Algorithm That Partitions The Processes Into Different Queues And Each Queue Has Its Own Scheduling Algorithm (As Appropriate For That Queue) Such As Interactive Processes Use RR In One Queue And Batch Processes Use FCFS In Another Queue.

6. **Multilevel Queue Feedback Scheduling*:*** Almost Similar To The Previous Algorithm But With Major Difference That Multiple Queues Can Be Transfer The Jobs Among Them During The Execution
Of A Process. It Provides The Maximum Chances To Run All The Jobs Having Different Levels Of Priorities. By Implementing This Algorithm, Average Turnaround Time Can Be Decreased But An Overhead May Be Increased Due To Switching Of The Jobs Among Queues Arranged At Different Levels.

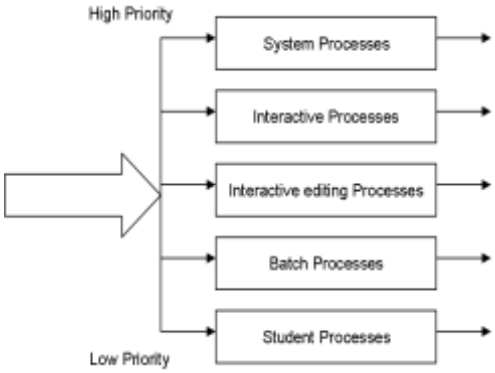

Fig.2 Multilevel Queue Scheduling

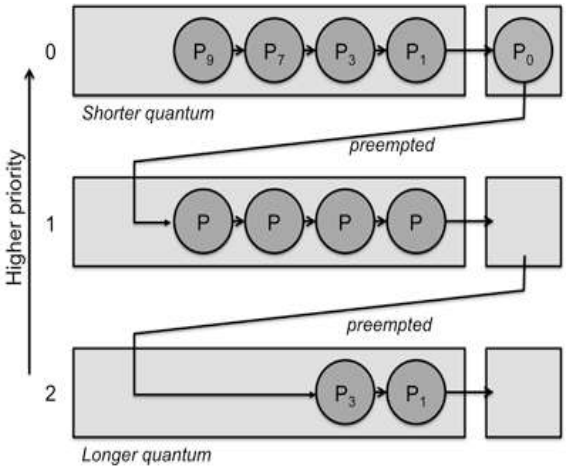

Fig.3 Multilevel Queue Feedback Scheduling

II. CONCLUSION

The Performance And Efficiency Of Any Kind Of Scheduling Algorithm Is Measured Through Various Performance Parameters Such As Average Waiting Time And Throughput Etc. Each Algorithm Is Specially Designed To Meet One Of These Parameters. Treatment Of Shortest Process In SJF Scheduling Tends To Result In Increased Waiting Time For Long Processes, Though It Produces Minimum Average Waiting Time And Average Turnaround Time. To Evaluate A Scheduling Algorithm, It Should Be Handed To Operating System And Its Proper Working And Capabilities Can Be Measured. But, The Main Concern Is That Performance Of An Algorithm Merely Depends Upon The Certain Conditions Under Which It Is Supposed To Perform The Scheduling.

REFERENCES

[1] Silberschatz, A. P.B. Galvin And G. Gagne, "Operating System Concepts" (8th Edition, Wiley India, 2012)

[2] Sabrian, F., C.D. Nguyen, S. Jha, D. Platt And F. Safaei, "Processing Resource Scheduling In Programmable Networks", 2005

[3] Sun Huajin, Gao Deyuan, Zhang Shengbing, Wang Danghui, "Design Fast Round Robin Scheduler", ©IEEE, 2012

[4] Mr. Umar Saleem And Dr. Muhammad Younus Javed, "Simulation Of CPU Scheduling Algorithm", ©IEEE, 2000

[5] Mr. Sindhu M, Mr. Rajkamal R And Mr. Vigneshwaran P, "An Optimum Multilevel CPU Scheduling Algorithm", ©IEEE, 2010

[6] Mahima Shrivastava, "Analysis And Review Of CPU Scheduling Algorithms", International Journal Of Scientific Engineering And Research, 2014

[7] Nikhil Selvaraj, Yeshwanth Raja, R. Ajay Adithya, Arjun Narendran, "Comparison Of The Various CPU Scheduling Algorithms", International Journal Of Computer Science, 2014

[8] Pinal Salot, "A Survey Of Various Scheduling Algorithms In Cloud Computing Environment", IJRET, 2013

Navjot Singh "A Comparative Analysis of Various CPU Scheduling Algorithm "International Journal of Engineering Research and Applications (IJERA) , vol. 8, no. 4, 2018, pp. 33-35