**Analysis and comparison of scheduling Algorithms**

Himanshu Pandey, Bhargav Pawar, Nishant Bhat, Abhijit Gawai, Shyam Kawale

Department of Information Technology and MCA

***Abstract :*** The Main purpose of this to compare and make analysis of various scheduling Algorithms in Operating System. We took a survey, and we come across various advantages and disadvantages of different scheduling algorithm. We compared various algorithms on basis of scheduling Criteria. We came across that choice of algorithm depends on factors like CPU utilization, throughput etc. Then we compared various scheduling algorithms on basis of advantages and disadvantages.

1. ***Introduction***

Scheduling refers to a set of policies and mechanisms to control the order of work to be performed by a Computer Operating system. Among all the resources in a computer system that are scheduled before use, the CPU is by far the most important. Multiprogramming is the scheduling of the CPU. the essential idea is to stay the CPU busy the maximum amount as possible by executing a process until it must await an another Process, and then switch to another process. Processes alternate between consuming CPU cycles (CPU burst) and performing I/O (I/O burst).In multiprogramming systems, when there is more than one runnable process, the operating system must decide which one

to activate. The choice is formed by the a part of the OS called the scheduler, by employing a scheduling algorithm .

1. ***CPU Scheduling***

In CPU Scheduling a process will own CPU for execution while another process is on hold. CPU scheduling used to make sure that whenever the CPU remains idle, the OS at least executing one of the processes available in the ready queue. *CPU scheduler* is used for selection of process. It selects one of the processes in memory which is ready for execution.

1. ***Scheduling Main Objectives***

* Fairness to all processes
* Be predictable
* Minimize overhead
* Balance available resources
* Enforcement of priorities
* Achieve balance between response and utilization
* Maximize throughput

1. ***Scheduling Criteria***

Different CPU scheduling algorithms have different properties. In order to choose which algorithm to use in a particular situation, we must consider the properties of the various algorithms.

* *CPU utilization: -*

We want to keep the CPU

as busy as possible.

* *Throughput: -*

One measure of work(CPU executing process) is the number of processes that are completed per time unit, called throughput.

* *Turnaround time: -*

The interval from the time of

submission of a process to the time of

completion is the turnaround time.

* *Waiting time: -*

Time that a process spends waiting in the ready queue for execution.

* *Response time: -*

time from the

submission of a request until the first

response is produced.

Favourable Process need to have -

* Minimized turnaround time
* Minimized waiting time
* Maximized throughput
* Maximized CPU utilization
* Minimized response time

1. ***Preemptive and Non-Preemptive Scheduling***

**Preemptive Scheduling:-**

In preemptive scheduling a process switches from running state to ready state or from waiting state to ready state.

Resources like CPU are allocated to the process only for **limited** time and then is taken away, and the process is placed back in the ready queue again if it still has CPU burst time remaining. The process stays in ready queue till it gets next chance for execution.

If a process with high priority arrives in the ready queue, the current process is interrupted in the middle of execution and is placed in the ready queue till the process with high priority is utilizing resources like CPU cycles.

It is flexible but, increases the overhead of switching the process from running state to ready state and vise-verse.

Algorithms working on preemptive scheduling are Round Robin, Shortest Job First (SJF), Priority scheduling.

**Non-Preemptive Scheduling:-**

In non-preemptive scheduling a process terminates or switches from running to waiting state.

After resources (CPU) is allocated to a process, the process holds the CPU till it gets terminated or reaches a waiting state.

In non-preemptive scheduling a executing process is not interrupted. It waits for the process to complete its CPU burst time and allocates the CPU to another process.

|  |  |
| --- | --- |
| **Preemptive Scheduling** | **Non-Preemptive Scheduling:-** |
| Process can be interrupted in between execution. | Process can not be preempted till it terminates or switches to waiting state. |
| If a high priority process frequently occurs in ready queue, low priority process may starve. | If a process with long burst time is executing on CPU, then another process with less CPU burst time may starve. |
| Preemptive scheduling is flexible. | Non-preemptive scheduling is rigid. |
| The resources are allocated to a process for a limited time. | After resource allocation the process holds it till it completes its burst time or switches to waiting state. |
| Preemptive scheduling is cost associated. | Non-preemptive scheduling is not cost associative. |

1. ***Types of scheduling:***

**First Come First Served (FCFS):-**

First in First out (FIFO) is the simplest

scheduling algorithm. It's easy to understand and implement programmatically, using a Queue data structure, where a new arriving jobs are inserted into the  tail of the queue, and the scheduler selects process from the head of the queue.. FCFS performs better for long jobs. An ideal real world example of FCFS scheduling is buying tickets at ticket counter.

Advantages

* FCFS doesn't include any complex logic, it just puts the method requests in a queue and executes it one after another.
* Eventually, every process will get executed, so starvation doesn't occur in FCFS.

Disadvantages

* There is no chance of pre-emption of a process. If a process gets started, then CPU executes the process until it finishes.
* Because there is no pre-emption in FCFS, if a process executes for a long time than expected, then latter processes in the queue will have to wait for a long time to get a chance to be executed.

**Shortest Job First (SJF):-**

SJF policy selects the job

with the shortest (expected) processing time first. Shorter jobs are always executed before long jobs.

Advantages

* The throughput is increased as more number of processes can be executed in less amount of time.

Disadvantages

* The time taken by a process must be known by the CPU before execution of queue starts, which is not possible.
* Processes having longer execution time will have more waiting time, eventually they'll suffer starvation.
* Longer processes need more waiting time, so starvation may occur.

**Comparison of FCFS and SJF:-**

|  |  |
| --- | --- |
| Process Time | Execution Time |
| P1 | 24 |
| P2 | 3 |
| P3 | 3 |

FCFS

|  |  |  |
| --- | --- | --- |
| P1 | P2 | P3 |

0 24 27 30

Average time : 0+24+27/3=17ms

SJF

|  |  |  |
| --- | --- | --- |
| P1 | P2 | P3 |

0 3 6 30

Average time : 0+3+6/3=3ms

As SJF gave priority to P3 having expected processing time 3ms, there is huge difference in their Average time.

There might me some cases were FCFS has less average time than SJF.

**Priority Scheduling:-**

In Priority Scheduling each process is assigned with a priority. The queue contains an entry for each process ordered by its priority. The process having higher priority are executed first. Processes with same priority are executed as if they are in FCFS manner.

Types of Priority Scheduling Algorithm:-

* Preemptive Priority Scheduling:

If the new process is received at the ready queue with a higher priority than the currently running process, the CPU is stopped, which means the current process is stopped and therefore the incoming new process having higher priority gets the CPU for its execution.

* Non-Preemptive Priority Scheduling:

In non-preemptive priority scheduling algorithm if a new ready process arrives having higher priority than the currently running process, then incoming process is shifted at the head of the ready queue, therefore after the execution of the current process, new process with higher priority will be executed.

Advantages

* The priority queue is selected based on factors of process like memory requirement, time requirement or user preference.

Disadvantages

* Another scheduling algorithm(FCFS) is also required to schedule the processes having same priority.
* In preemptive priority scheduling, process having higher priority can execute before an already executing lower priority process. If lower priority process keeps waiting for higher priority process completion, starvation may occurs.

**Round Robin Scheduling:-**

A defined time for execution is allotted for every process called *quantum*. Once a process is executed for given amount of time then other process gets executes for given amount of time.

Context switching is used to save states of process having execution completed.

Advantages

* CPU is alloted to every process for fixed amount of time(*quantum*), therefore all processes are given an equal priority.
* Starvation doesn't occur because for every round robin cycle CPU is alloted to every process for fixed amount of time for execution. So, no process is left behind.

Disadvantages

* The throughput in Round Robin depends on the selection of the time(quantum). If expected time (quantum) is longer than needed to completed execution of process, it tends to have the equivalent behavior as FCFS.
* If expected time quantum is shorter than needed, the number of times that CPU switches from one process to different process, increases. This results in decrease in efficiency of CPU.

**Hybrid CPU Scheduling :-**

Hybrid CPU scheduling algorithm mainly focuses on the drawbacks of straightforward round robin scheduling algorithm which gives equal priority to all the process to be executed. That results in the increase in response time and waiting time of further processes which results in decrease of system throughput. Hybrid scheduling algorithm are going to be executed in three steps which is going to help to decrease a number of performance parameters such as average waiting time and average turnaround time. There are some steps of Hybrid CPU scheduling algorithm. The following steps are:

* Step1: Allocation of CPU to each process as of round robin algorithm.
* Step2: In order of the given priority, for given time (quantum) for all the process to be executed.
* Step3: The processes are executed consistent with priorities given and every process gets the control of the CPU for their execution.

Advantages

* Works better with process with smaller CPU burst, than round robin.

1. ***General Comparison of Algorithms***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | Preemptive/ Not | Time quantum | Priority | Queuing Model |
| FCFS | NON | NO | NO | YES |
| SJF | BOTH | WHEN PREMPTIVE | NO | NOT |
| SRJF | YES | YES | NO | NOT |
| Priority Scheduling | BOTH | WHEN PREMPTIVE | YES | NOT |
| Round Robin | YES | YES | NO | YES |
| Hybrid | YES | YES | YES | NOT |

1. ***Conclusion***

We have successfully compared many

scheduling algorithms. The hybrid scheduling has more qualified and capable than Round Robin, Because it has less average waiting time and average turnaround time, as compared to simple round robin, in turn reducing. The operating systems overhead and hence dispatch latency. Also the problem of starvation is reduced as the processes with less remaining CPU burst time are assigned with the higher priorities and are executed first.

First in the scheduling algorithm performance of time sharing system can be improved with the hybrid Scheduling algorithm and can also be modified to expand the performance in real time system.

1. ***References***

<https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/5_CPU_Scheduling.html>

<https://www.sciencedirect.com/science/article/pii/S1110866513000509>

<https://www.semanticscholar.org/paper/AN-IMPROVED-ROUND-ROBIN-CPU-SCHEDULING-ALGORITHM-Mishra-Khan/453c3cf92cdc4f36cdfd2cbc211c8c857bb0ecbf>

<https://www.researchgate.net/publication/251180989_A_New_Operating_System_Scheduling_Algorithm>

<https://www.semanticscholar.org/paper/An-Enhanced-Round-Robin-CPU-Scheduling-Algorithm-Khatri/8d42584f3336606b2a6cac22b75f3941242f2f64>

<https://www.semanticscholar.org/paper/Varying-Response-Ratio-Priority%3A-A-Preemptive-CPU-Singh-Pandey/691f744a2255605394876e4717609d137063d220>