Survey on various Scheduling Algorithms

Harshita Jha¹ Subrata Chowdhury² & Ramya.G³
School of Information Technology and Engineering,
VIT University, Vellore - 632014, Tamil Nadu, India¹,³
EXL Techno solutions,Mumbai²

Abstract: The main purpose of this paper is to develop a hybrid CPU scheduling algorithm that combines round robin scheduling with priority scheduling. This approach eradicates the drawbacks of round robin scheduling and shows improvement over the same. It is also found that this scheduling is an improvement over priority scheduling.

1. Introduction

Scheduling refers to a set of policies and mechanisms to control the order of work to be performed by a Computer system. Of 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 basic idea is to keep the CPU busy as much as possible by executing a process until it must wait for an event, 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[2]. The decision is made by the part of the operating system called the scheduler, using a scheduling algorithm.

1. Types of scheduling:
First Come First Served (FCFS):- FCFS, also known as First in First out (FIFO), is the simplest scheduling policy. Arriving jobs are inserted into the tail (rear) of the ready queue and the process to be executed next is removed from the head (front) of the queue. FCFS performs better for long jobs. Relative Importance of jobs measured only by arrival time (poor choice)[1]. A long CPU bound job may hog the CPU and May force shorter (or I/O bound) jobs to wait prolonged periods. This in turn may lead to A Lengthy queue of ready jobs, and thence to the “convoy effect.”

Shortest Job First (SJF):- SJF policy selects the job with the shortest (expected) processing time first. Shorter jobs are always executed before long jobs[1]. One major difficulty with SJF is the need to know or estimate the processing time of each job. Also, long running jobs may starve, because the CPU has a steady supply of short jobs. Round-Robin (RR) reduces the penalty that short jobs suffer with FCFS by preempting running jobs periodically. The CPU suspends the current job when the reserved quantum (time-slice) is exhausted. The job is then put at the end of the ready queue if not yet completed. The critical issue with the RR policy is the length of the quantum. If it is too short, then the CPU will be spending more time on context switching. Otherwise, interactive processes will suffer

II. Scheduling Criteria

Different CPU scheduling algorithms have different properties and the choice of a particular algorithm may favor one class of processes over another.

In choosing 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. Conceptually, CPU utilization can range from 0 to 100 percent. In a real system, it should range from 40 percent (for a lightly loaded system) to 90 percent (for a heavily used system)[3].

- Throughput: - If the CPU is busy executing processes, then work is being done. One measure of work is the number of processes that are completed per time unit, called throughput. For long processes, this rate may be one process per hour; for short transactions, it may be 10 processes per second[3].

- Turnaround time: - From the point of view of a particular process, the important criterion is how long it takes to execute that process. The interval from the time of submission of a process to the time of completion is the turnaround time[3].

- Waiting time: - The CPU scheduling algorithm does not affect the amount of time during which a process executes or does I/O; it affects only the amount of time that a process spends waiting in the ready queue.

- Response time: - In an interactive system, turnaround time may not be the best criterion. Often, a process can produce some output fairly early and can continue computing new results while previous results are being output to the user. Thus,
another measure is the time from the submission of a request until the first response is produced.

Processes have some following characteristics:-

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

### III. 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

#### A. Round Robin Scheduling Algorithm

- The round-robin (RR) scheduling algorithm is designed especially for time-sharing systems. It is similar to FCFS scheduling, but pre-emption is added to switch between processes.
- A small unit of time, called a time quantum or time slice, is defined. A time quantum is generally from 10 to 100 milliseconds. The ready queue is treated as a circular queue[3].
- To implement RR scheduling,
  - We keep the ready queue as a FIFO queue of processes.
  - New processes are added to the tail of the ready queue.
  - The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process.
  - The process may have a CPU burst of less than 1 time quantum.
    a. In this case, the process itself will release the CPU voluntarily.
    b. The scheduler will then proceed to the next process in the ready queue.
  - Otherwise, if the CPU burst of the currently running process is longer than 1 time quantum,
    - The timer will go off and will cause an interrupt to the OS.

- A context switch will be executed, and the process will be put at the tail of the ready queue.
- The CPU scheduler will then select the next process in the ready queue.

**Disadvantages of round robin scheduling algorithm:-**

- Very important jobs wait in line.
- Largest job take enough time for completion.
- Setting the quantum too short causes too many context switches and lower the CPU efficiency.
- Setting the quantum too long may cause poor response time and approximates FCFS[2].

#### B. Priority Scheduling

Each process is assigned a priority. The ready list contains an entry for each process ordered by its priority. The process at the beginning of the list (highest priority) is picked first. A variation of this scheme allows preemption of the current process when a higher priority process Arrives. Another variation of the policy adds an aging scheme, where the priority of a process increases as it remains in the ready queue; hence, will eventually execute to completion[2].

**Disadvantages of Priority scheduling: -** Starvation of lower priority process is possible if large number of higher priority process keep arriving continuously. Starvation: Starvation is a resource management problem where a process does not get the resources it needs for a long time because the resources are being allocated to other processes.[UPDATE]: Starvation generally occurs in a Priority based scheduling System. Where High Priority (Lower Number = Higher Priority) requests get processed first. Thus a request with least priority may never be processed[2].

### IV. Hybrid CPU Scheduling Algorithm

The hybrid CPU scheduling algorithm focuses on the drawbacks of simple round robin scheduling which given equal priority to all the process. Because of the drawbacks round robin scheduling is not efficient for process with smaller CPU burst. The result in the increase in response time and waiting time of processes which results decrease the system throughput. Hybrid scheduling algorithm eliminates the defects of implementing simple round robin scheduling. Hybrid scheduling algorithm will be executed in three steps which will help to minimize a number of performance parameters such as average turnaround time and average waiting time. The algorithm performs following steps:

Step1: allocate CPU to every process in round robin fashion.
Step 2: According to the given priority, for given time quantum for all the processes. The processes are executed according to priorities and each process gets the control of the CPU until they finished their execution.

Case Studies
We have taken five (5) processes that have been defined, and also define CPU burst time and their priorities, these five (5) processes are scheduled in round robin and also according to the Hybrid scheduling algorithm. Average turnaround time, average waiting time, has been calculated and the result were compared for doing this we have implemented the priority based CPU scheduling algorithm in c.

Consider five (5) processes P1, P2, P3, P4 and P5. And also given CPU burst time, priorities. We have taken the time quantum is 4ms.

According to the simple round robin scheduling:
Simple round robin does not use priority and five processes has been scheduled using simple round robin scheduling[2]. The time slice or the time quantum of four milliseconds has been used. In round robin algorithm no process is allocated CPU for more than one time slice in a row. If the CPU process exceeds one time slice, the concern process will be preempted and put to the ready queue. The process is preempted after the first time quantum and the CPU is given to the next process which is in the ready queue (P2), similarly schedule the processes. The process time slicing in simple round robin scheduling is show in Gantt chart.

Gantt chart

<table>
<thead>
<tr>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>19</td>
<td>23</td>
<td>27</td>
<td>31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>39</td>
<td>43</td>
<td>47</td>
<td>51</td>
<td>52</td>
<td>56</td>
<td>60</td>
<td>61</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average waiting time: 33.2ms
Average turnaround time: 45.4ms

According to Hybrid scheduling:
We have use priority based round robin CPU scheduling , process with the highest priority is execute first for the time equal to given time quantum or time slice is 4ms. In the same manner other processes are executed according to their priorities for single time quantum.

The sequence of execution for above case is:

<table>
<thead>
<tr>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>19</td>
<td>23</td>
<td>27</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>39</td>
<td>43</td>
<td>47</td>
<td>48</td>
<td>52</td>
<td>56</td>
<td>57</td>
<td>61</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Average waiting time: 29.4ms
Average turnaround time: 41.6ms

V. Conclusion

We have successfully compared both the algorithm simple Round robin scheduling and the hybrid scheduling. That the hybrid scheduling more qualified and capable. Because it has less average turnaround time and average waiting time, as compared to simple round robin, in turn reducing the operating systems overhead and hence dispatch latency. Also it reduces the problem of starvation as the processes with less remaining CPU burst time are assigned with the higher priorities and are executed. Fast in the algorithm performance of time sharing system can be improved with the hybrid Scheduling algorithm and can also be modified to expand the performance of real time system.
References:


