A Combined Preemptive SJF and Preemptive Priority Algorithm to Enhance CPU Utilization

Asma Joshita Trisha
Lecturer
Premier University
Chattogram, Bangladesh

Senjuthi Bhattacharjee
Lecturer
Premier University
Chattogram, Bangladesh

ABSTRACT

A computer system performs its tasks by executing processes. In a CPU multiplexed system multiple processes can execute concurrently by switching among them. For handling the processes CPU scheduling techniques are used, through which the CPU utilization can be enhanced. There are several CPU scheduling algorithms for deciding which of the processes in the ready queue is to be allocated in the CPU. The existing algorithms have some problems that may lead to huge average waiting time or starvation. The SJF results the minimum average waiting time, but it also introduces starvation for bigger processes. The priority scheduling works on the basis of priority assigned to each process results waiting time that is greater than SJF. Sometimes it is required to apply an algorithm which will give attention to both of these algorithms. The proposed algorithm will work as a bridge between SJF and priority on the basis of their arrival.

General Terms

CPU scheduling algorithms, Shortest Job First algorithm, Priority algorithm.

Keywords

CPU scheduling, SJF, Priority, preemptive, Gantt chart, turnaround time, waiting time, response time.

1. INTRODUCTION

Among various tasks of operating system, process scheduling is the most crucial one. When multiple processes are simultaneously in the ready state, they need to allocate resources to execute their task [1]. This resource allocation is done by the CPU scheduler. In case of multiple process's arrival, one process will allocate resource and the others will be in the waiting state until the resources are released.

To maximize the CPU utilization there must be some procedure running all the time. There are several algorithms to accomplish the task of selecting processes from the ready queue. In choosing among the algorithms in a particular situation, some properties of the various algorithms must be considered such as; average turnaround time, average waiting time, average response time [2]. Some characteristics are used for comparison can make a substantial difference in which algorithm is judged to be best in case of maximize CPU utilization and throughput, and at the same time minimize turnaround time, waiting time and response time [3]. The following criteria are needed to be focused:

- CPU utilization: to keep the CPU busy by allocating processes. The expected CPU utilization rate is 100%. But in a real system it ranges from 40% to 90%.
- Throughput: number of processes completed per hour by the system [4]. The higher the number of completed processes, the better the system is.
- Turnaround time: it is the statically average time from the moment of submission of a process to the moment it is completed.
- Waiting time: sum of time that a process spent in the ready queue.
- Response time: time between issuing a request until the first response is perceived.

The mostly used algorithms that works on the basis of the above criterion are FCFS (first come first serve), SJF (shortest job first), Priority and Round Robin. Amon these the SJF gives the minimum average waiting time but at the same time it introduces starvation in the long processes. On the other hand, the priority scheduling depends only on the given priority number. The lower the number, the higher the priority is.

The SJF algorithm is based only in the burst time, whereas the Priority algorithm schedules the processes according to the priority. In the proposed algorithm, a theory has been developed on a combination of both of the SJF and Priority scheduling algorithm. It gives better result in contrast with the original basic algorithms in the basis of average waiting time and average turnaround time. However, scheduling algorithms can be applied in both preemptive and non-preemptive manner [5].

2. LITERATURE REVIEW

Operating System plays a vital role to allocate CPU according to the arrival of different processes. As the processes arrives randomly with their different types, they require scheduling algorithms for working in the real environment. Varieties of algorithms are available just to make efficient allocation of the CPU so that the CPU can be utilized at its fullest extent and complete its execution in a minimum time. Researches has been done to improve the basic scheduling algorithms. Some of them has reviewed in this section.

Shweta Jain, Dr. Saurabh Jain, "A Review Study on the CPU Scheduling Algorithms". International Journal of Advanced Research in Computer and Communication Engineering", 2016 [6], have discussed about various researches done in the field of CPU scheduling and its performance. In this paper, they have given the review of those different scheduling algorithms that are performed with different parameters, such as turnaround time, burst time, response time, waiting time, throughput, fairness and CPU utilization. It gives a brief overview to the problem of scheduling jobs/processes on the central processing unit (CPU) of the computer system.

Chandra Shekar N, Karthik V, “Analysis of Priority Scheduling Algorithm on the Basis of FCFS & SJF for Similar Priority Jobs”, 2017 [7], proposed a scheduling algorithm, in which processes having equal priority is executed on the basis of burst time, i.e. the process which have shortest burst time will execute first. The SJF based priority algorithm results in reduced average waiting time and turnaround time.

Ahana Roy, Aspen Olmsted, “An Improved Priority Scheduling Algorithm Using Time Slice to Minimize Response Time”, 2018 [8], propose an algorithm, where the process with highest priority is allocated first and those with the same priorities are scheduled by FCFS policy. Then all the sorted process will be allocated to CPU for a predetermined time slice. This algorithm aids in minimizing some of the performance parameters such as response time, waiting time for lower priority processes ensuring each process is given a fair chance to access resources.

3. OVERVIEW ON EXISISTING SJF AND PRIORITY ALGORITHM

3.1 Shortest Job First (SJF) Algorithm

The working policy of SJF associates the burst time of execution with the process. When a process with smaller burst time arrives in the ready queue, CPU is assigned to that process next. If the burst time of next two consecutive processes become same, then the tie will break applying the FCFS algorithm. SJF can be applied as both preemptive and non-preemptive based on their arrival [9]. In a non-preemptive SJF, the CPU will be assigned to the shorter processes by ignoring their arrival time. But a preemptive SJF proceeds according to their arrival time. When a shorter process is arrived, the currently executing process will be preempted by releasing the CPU for the newly arrived shorter process and stays in the waiting queue [10].

SJF algorithm is possibly optimal. It executes the short process before the long process and thus reduces the waiting time for the short process more than increases waiting time for long process. Consequently, it ends up with a minimum average waiting time compared to the other scheduling algorithms [3].

3.2 Priority Based Algorithm

In the priority scheduling algorithm, each process is associated with a priority and the CPU is allocated based on highest priority. Equal priority processes are executed on first come first serve basis. Priority can be either preemptive or non-preemptive. A preemptive priority algorithm preempts the currently executing process from the CPU if the newly arrived process contains a higher priority [11]. In the non-preemptive priority scheduling algorithm, the currently running process will complete its full task and the newly arrived process are just added to the head of the ready queue.

One drawback of priority scheduling algorithm is the indefinite waiting or starvation. A process is considered as blocking when it is in the ready queue buy waiting for the CPU to become available. Lower priority processes are often undergoing through an indefinite waiting.

4. PROPOSED ALGORITHM

SJF scheduling focuses only in the burst time. Both the preemptive and non-preemptive SJF are based on the CPU burst time. Again, the priority scheduling algorithm concentrate solely in the priority of the processes. When a process with shorter burst time but lower priority arrives in the ready queue it suffers from huge waiting in the ready queue which may cause starvation. In the proposed algorithm, these types of process are kept in front of the ready queue by emphasizing in both the burst time and the priority.

In the proposed algorithm, when a process arrives in the ready queue, it compares both of its burst time and priority with the currently running process. This comparison is accomplished on the basis of the summation of the burst time and priority. If the summation of newly arrived process is less than the summation of the currently running process, then the CPU will be preempted from the running process and assigned it to the new process. In case of equal summation of the burst time & priority, FCFS scheduling will be used as a tie breaker. This comparison will be performed each time when a new process arrives in the ready queue.

Through this proposed algorithm smaller burst time and lower priority will get a privilege to get into the CPU. Moreover, it results better than the original preemptive-priority algorithm and approximately equal result of the preemptive-SJF in the criteria of average turnaround time, average waiting time and average response time. Thus, will give better CPU utilization by eliminating indefinite waiting and starvation.

5. IMPLEMENTATION

The proposed algorithm is compared with preemptive SJF and preemptive priority algorithm for a set of different processes. The comparison has done on the basis of average turnaround time, average waiting time and average response time.

5.1 Case 1

Table 1. Case study no. 1 with four processes

Process	Burst Time	Priority	Arrival Time
P_1	14	3	2
P_2	9	2	0
P_3	20	1	1
P_4	5	4	3

According to the preemptive SJF the Gantt chart is:

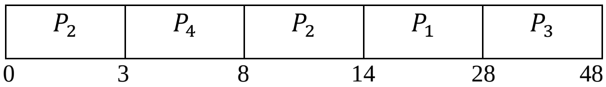

Figure 1. Gantt chart of Preemptive SJF Scheduling

According to the preemptive priority the Gantt chart is:

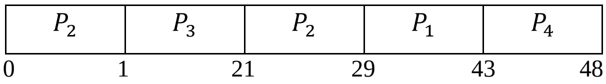

Figure 2. Gantt chart of Preemptive Priority Scheduling

According to the proposed algorithm the Gantt chart is:

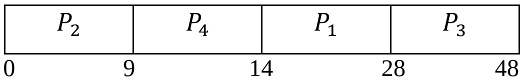

Figure 3. Gantt chart of Proposed Algorithm Scheduling

In the proposed algorithm, at time 3, burst time of P2 is 6 and summation of burst time & priority is 8. Whereas the burst time of P4 is 5 and the summation of burst time & priority is 9. Hence, P2 will continue executing. As all other processes in ready queue at time 9, then the process which has lowest summation of burst time and priority will execute first.

5.2 Case 2

Table 2. Case study no. 2 with four processes

Process	Burst Time	Priority	Arrival Time
P_1	8	3	2
P_2	4	1	1
P_3	9	2	0
P_4	5	4	3

According to the preemptive SJF the Gantt chart is:

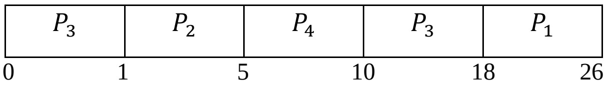

Figure 4. Gantt chart of Preemptive SJF Scheduling

According to the preemptive priority the Gantt chart is:

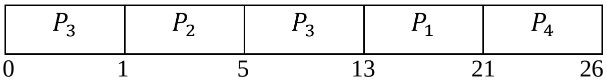

Figure 5. Gantt chart of Preemptive Priority Scheduling

According to the proposed algorithm the Gantt chart is:

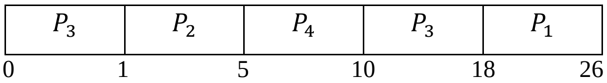

Figure 6. Gantt chart of Proposed Algorithm Scheduling

In the proposed algorithm, the process which has lowest summation of burst time and priority will execute first.

5.3 Case 3

Table 3. Case study no. 3 with four processes

Process	Burst Time	Priority	Arrival Time
P_1	10	3	2
P_2	1	1	3
P_3	2	4	1
P_4	5	2	0
P_5	7	5	4

According to the preemptive SJF the Gantt chart is:

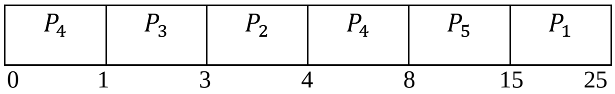

Figure 7. Gantt chart of Preemptive SJF Scheduling

According to the preemptive priority the Gantt chart is:

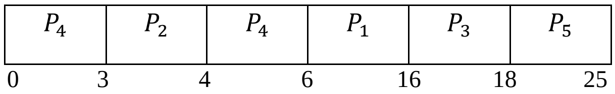

Figure 8. Gantt chart of Preemptive Priority Scheduling

According to the proposed algorithm the Gantt chart is:

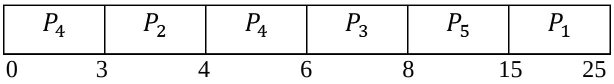

Figure 9. Gantt chart of Proposed Algorithm Scheduling

In the proposed algorithm, at time 1, there are P4 & P3 in the ready queue consisting the same summation of burst time & priority which is 6. As P4 has arrived at time 0, which is prior to the arrival of P3, hence P4 will continue executing according to FCFS algorithm.

Again, at time 3, the summation of burst time & priority of P4 is 2, whereas the summation of burst time & priority of P2 is 1. Therefore, P4 will be preempted from the CPU & P2 will continue is execution. After that the process which has lowest summation of burst time and priority will execute first.

6. RESULT

Table 4. Comparison of Preemptive SJF, Preemptive Priority and Proposed algorithm based on Case study no.1

Scheduling Algorithms Scheduling Criteria	Preemptive SJF	Preemptive Priority	Proposed Algorithm
Average Turnaround Time	23	33.75	23.25
Average Waiting Time	11	21.75	11.25
Average Response Time	9.75	16.75	11.25

Table 5. Comparison of Preemptive SJF, Preemptive Priority and Proposed algorithm based on Case study no.2

Scheduling Algorithms Scheduling Criteria	Preemptive SJF	Preemptive Priority	Proposed Algorithm
Average Turnaround Time	13.25	14.75	13.25
Average Waiting Time	6.75	8.25	6.75
Average Response Time	4.5	7.25	4.5

Table 6. Comparison of Preemptive SJF, Preemptive Priority and Proposed algorithm based on Case study no.3

Scheduling Algorithms Scheduling Criteria	Preemptive SJF	Preemptive Priority	Proposed Algorithm
Average Turnaround Time	9	11.8	9.6
Average Waiting Time	4	6.8	4.6
Average Response Time	3.5	6.6	4.4

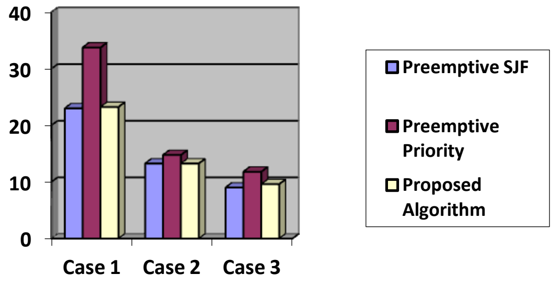

Figure 10: Graph showing comparative result of preemptive SJF, preemptive priority and proposed algorithm based on the average turnaround time.

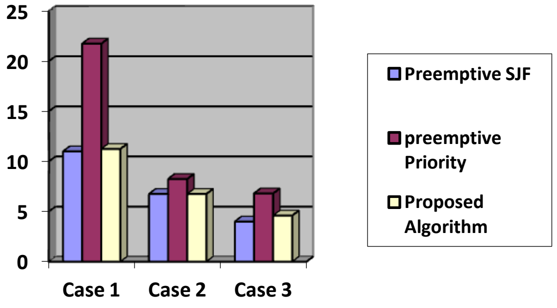

Figure 11: Graph showing comparative result of preemptive SJF, preemptive priority and proposed algorithm based on the average waiting time.

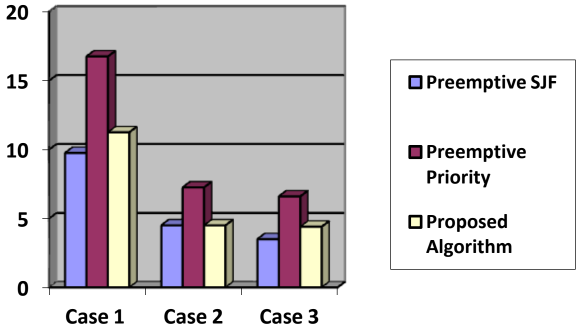

Figure 12: Graph showing comparative result of preemptive SJF, preemptive priority and proposed algorithm based on the average response time.

From the above comparisons between preemptive SJF, preemptive priority and proposed algorithm it can be observed that the proposed algorithm gives almost equal result as the preemptive SJF and definitely a better result than preemptive priority in terms of average turnaround time, average waiting time and average response time.

7. CONCLUSION

The intention is primarily to enhance the CPU utilization; however, the difference with other algorithms is in the approach. An alternative way has proposed to the steps where processes are swapped symmetrically in order to avoid starvation for low priority processes. The proposed hypothesis of calculating the summation of burst time and priority for the time slice value is less time consuming than preemptive priority algorithm and it clearly shows maximum CPU utilization and efficient handling of resources.

8. REFERENCES

[1] Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, “Operating System Concepts”, John Wiley & Sons Inc, 8th ed, Chapter No.5.

[2] Basit Shahzad, Muhammad Tanvir Afzal, “OPTIMIZED SOLUTION TO SHORTEST JOB FIRST BY ELIMINATING THE STARVATION”. Jordanian International Electrical Engineering and Electronic Conference, November 2005.

[3] William Stallings, “Operating System”, Pearson Prentice Hall, 7th ed, Chapter No. 9.

[4] Andrew S. Tanenbaum, “Modern Operating System”, Pearson Prentice Hall, 3rd ed, Chapter No.2.

[5] Sukumar Babu Bandarupalli, Neelima Priyanka Nutulapati, Prof. Dr. P. Suresh Varma, “A Novel CPU Scheduling Algorithm–Preemptive & Non-Preemptive”, International Journal of Modern Engineering Research, Vol.2, Issue.6, Nov-Dec. 2012 pp-4484-4490.

[6] Shweta Jain, Dr. Saurabh Jain, “A Review Study on the CPU Scheduling Algorithms”. International Journal of Advanced Research in Computer and Communication Engineering”, Vol. 5, Issue 8, August 2016.

[7] Chandra Shekar N, Karthik V, “Analysis of Priority Scheduling Algorithm on the Basis of FCFS & SJF for Similar Priority Jobs”, International Journal of Engineering Research in Computer Science and Engineering, Vol. 4, Issue 3, March 2017.

[8] Ahana Roy, Aspen Olmsted, “An Improved Priority Scheduling Algorithm Using Time Slice to Minimize Response Time”, DOI: 10.13140/RG.2.2.20543.56489, November 2018.

[9] Harshita Jha, Subrata Chowdhury, Ramya.G, “Survey on various Scheduling Algorithms”, Imperial Journal of Interdisciplinary Research, Vol-3, Issue-5, 2017.

[10] Ms. Rukhsar Khan, Mr. Gaurav Kakhani, “Analysis of Priority Scheduling Algorithm on the Basis of FCFS & SJF for Similar Priority Jobs”, International Journal of Computer Science and Mobile Computing, Vol. 4, Issue. 9, September 2015, pg.324 – 331.

[11] Mohammed A.F. Al-Husainy, 2007, “Best-Job-First CPU Scheduling Algorithm”. Information Technology Journal, 6: 288-293.