International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 07 | July 2020 www.irjet.net p-ISSN: 2395-0072

SBFT Scheduling– A New Scheduling Algorithm to Reduce Time Taken to Schedule CPU Processes

Gopi Manoj Vuyyuru[1]

[1]Tata Consultancy Services, Bangalore, Karnataka, India

Abstract- In an operating system, CPU scheduling is one of the important tasks. To schedule processes, there are various algorithms out of that round-robin is widely adopted algorithm for CPU scheduling. But, the major disadvantage of the round-robin algorithm is its quantum size because there is no fixed quantum size. If the time quantum is more, the response time of a process is more. On the other hand, if the time quantum is too less, the overhead of CPU will increase. In this paper, a new approach called SBTF, which takes time quantum based on the shortest remaining time of available tasks in the ready queue. In this process, for each round, a new time quantum is taken based on the shortest remaining time of tasks. In the ending based on various analysis, we have shown that the proposed algorithm reduces the time taken to schedule processes when compared to the round-robin algorithm.

Keywords– Operating Systems, Multi-Tasking, CPU Scheduling, Time Quantum, Round Robin Scheduling.

1. INTRODUCTION

Operating System (OS) is system software that is a bridge between an end-user and hardware of the system. The main important tasks of an operating system include management of storage, memory, processor and devices of the computer. Apart from these managing and scheduling of resources is also a major task involved in multiprogramming operating systems. Scheduling means a set of rules, policies and strategies that are used to allocate resources to various processes. There are two different methods of scheduling. First one is called preemptive scheduling, in which resources are allocated to process for a particular amount of time. The second one is known as non-preemptive scheduling, in which a resource is allocated to a process until the process is terminated. [1][2]

There are various ways to schedule the resources, each algorithm has a different strategy based on which a resource is allocated to a process. In this paper, a new algorithm is proposed which reduces the time taken to allocate a resource to a process compared to that of the round-robin algorithm.

2. ANALYSIS OF ROUND ROBIN ALGORITHM

Round Robin algorithm is one of the easiest scheduling algorithms and also the most widely used process scheduling algorithm. This algorithm follows the preemptive scheduling approach to schedule processes hence it is also called the preemptive version of first come first serve algorithm. Here each process has a fixed time slot, during which it can use resources. Once the slot time is over, the resource is allocated to another process and this fixed time slot is called quantum. The context switching mechanism is used to save the states of the preempted process.

The major drawback of the round-robin algorithm is the size of time quantum. Because there is no fixed time quantum. If time quantum considered is large the response time will be less but the waiting time increases compared to that of waiting time obtained at lower time quantum. Also, if the time quantum considered is very short then it lowers the efficiency of CPU due to overhead. Below, the following example is provided to highlight the drawback of the algorithm.

Consider the following 5 processes and their arrival and burst times.

PROCESS NUMBER	ARRIVAL TIME (AT)	BURST TIME (BT)
P1	0	5
P2	1	6
P3	2	2

P4	4	1
P5	5	3

Table 1: List of processes and their respective Arrival & Burst times

Now, applying round-robin algorithm for above processes with different time quantum's and making a comparison.

TIME QUANTUM	AVG CT	AVG TAT	AVG WT	AVG RT
1	11.4	9	5.6	1.2
2	12.4	10	6.6	2.6
3	12	9.6	6.2	3.6
4	13	10.6	7.2	4.4

Table 2: Comparison of CT, TAT, WT, RT for various time quanta.

Where,

CT = Complete Time,

TAT = Turn Around Time (CT-AT),

WT = Waiting Time (TAT – BT),

RT = Response Time (CPU First time – AT).

From table 2 it is clearly observed that, as time quantum is increasing the averages of CT, TAT, WT, RT kept on increasing. So, to avoid this problem we have introduced a new algorithm in this paper.

3. EXPLANATION OF SBTF SCHEDULING ALGORITHM

The main idea of this algorithm is to reduce the average response time, waiting time and turnaround time taken to allocate resources to various processes compared to that of round-robin algorithm. Initially, the first process (say P1) which arrives into the ready queue is allocated with resource. The process P1, is allowed to utilize the allocated resource until it is terminated. During the execution of process P1, other processes which arrive are moved to ready queue. Once the P1 is finished, then the resource is allocated to the process which have the least burst time and the rest of the processes are also executed for the same amount of time. This process continues until all the processes are executed.

3.1 Working of SBTF Scheduling

Let P1, P2, P3, P4 …. Pn be n number of processes with arrival times AT1, AT2, AT3 …. ATn and burst times BT1, BT2, BT3 …. BTn respectively. Such that process P1 arrives at 0. Now follow the following steps in order to allocate resources to all the processes using SBTF Scheduling algorithm.

i) The process which arrived first i.e. P1, will be allocated with resources, and the processes which comes during this time will be moved to ready queue. Let P2, P3 come during this time, and they are moved to ready queue.

~~P1~~	P2	P3

Figure 1: Ready queue, where process P1 is completed and P2, P3, P4 are waiting for resources.

ii) Now, the processes P2, P3 are waiting in ready queue and their burst times are BT2, BT3. Now we have to compare the burst times of these processes and the process with shortest burst time will be executed first. The rest of the processes present in

ready queue will be executed with the same amount of time equal to the shortest burst time. The following are the cases we have for P2, P3.

a) If (B2 < B3)

Execute P2 until it is terminated and then execute P3 for B2 time and the burst time of P3 becomes

B3 = B3 – B2

b) If (B3 < B2)

Execute P3 until it is terminated and then execute P2 for B3 time and the burst time of P2 becomes

B2 = B2 – B3

c) If (B2 = B3)

If two processes have same burst time then execute the process which is first in the ready queue and execute rest of the processes. In this case since P2 is first in ready queue, it is executed first followed by P3.

The processes which come during the above step are moved to ready queue and the processes which are not finished in this first are also moved into ready queue.

iii) Repeat the above step until all the processes are allocated with resources and executed.

3.2 Explanation of algorithm with an example

Consider the following 4 processes and their arrival and burst times.

PROCESS NUMBER	ARRIVAL TIME (AT)	BURST TIME (BT)
P1	0	3
P2	1	3
P3	2	2
P4	4	1

Table 3: List of processes and their respective Arrival & Burst times

Step 1: At t = 0 the ready queue is as follows:

P1					

Figure 2: Ready queue at t=0.

Since at t = 0 we have P1 in ready queue, so we allocate resource to this process and execute it until it is terminated. So the running queue is as follows:

P1	P1	P1			
t=0	t=1	t=2	t=3		

Figure 3: Running queue at t=3.

The process P1 is executed until 3 seconds since BT1 = 3. During this time P2, P3 will come into ready queue.

Step 2: At t = 3 the ready queue is as follows:

~~P1~~	P2	P3			

Figure 4: Ready queue at t=3.

Since at t = 3 we have P2, P3 in ready queue, so we compare the burst times of two processes burst times. Since P3 is less than P2. So, we will execute P3 till it is terminated and P2 will be executed for 2 seconds and the burst time will become

B2 = B2 – B3 = 3 – 2 = 1

P1	P1	P1	P3	P3	P2	P2	
t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7

Figure 5: Running queue at t=7

During the execution of P3, P4 comes into ready queue.

Step 3: At t = 7 the ready queue is as follows:

~~P1~~	~~P2~~	~~P3~~	P4	P2	

Figure 6: Ready queue at t=7.

Since at t = 7 we have P4, P2 in ready queue, so we compare the burst times of two processes burst times. Since P3 and P2 have same burst times. So, we will execute P4 till it is terminated and then we execute P2.

B2 = B2 – B4 = 1 – 1 = 0 // Process P2 is completed.

P1	P1	P1	P3	P3	P2	P2	P4	P2
t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8

Figure 7: Running queue at t=9.

So, here all the processes have been successfully executed and the gantt chart is as follows:

P1	P3	P4	P4
t=0	t=3	t=5	t=8 t=9

Figure 8: Gantt chart for the given processes

4. ROUND-ROBIN V/S SBTF ALGORITHM

Let's take an example and make comparison between round-robin scheduling algorithm and SBTF.

PROCESS	ARRIVAL TIME	BURST TIME
P1	0	4
P2	1	2
P3	2	5
P4	3	6
P5	4	3
P6	5	1

Table 4: List of processes and their respective Arrival & Burst times

Now we will compute the averages of turnaround time, waiting time, response time for both round-robin as well as SBTF algorithms.

TIME QUANTUM	AVG CT	AVG TAT	AVG WT	AVG RT
1	13.83	11.33	7.83	1.67
2	13.83	11.33	7.83	3.5
3	14.5	12	8.5	4.67
4	14.17	11.67	8.17	6
5	13.5	11	7.5	6.83
6	13.17	10.67	7.17	7.17

Table 5: Comparison of CT, TAT, WT, RT for various time quantum's using Round-Robin algorithm

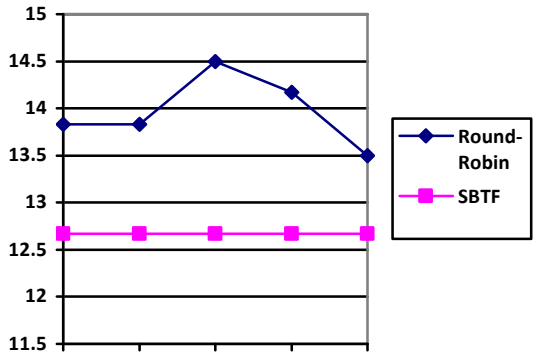
AVG CT	AVG TAT	AVG WT	AVG RT
12.67	10.17	6.67	4.17

Table 6: Comparison of CT, TAT, WT, RT using SBFT algorithm

From the above two tables we can clearly see that the completion time, turnaround time, waiting time of proposed algorithm i.e SBTF algorithm is less than that of any time quantum of that of Round-Robin algorithm. Also, the response time of the SBTF algorithm is very less than that of the average response time of many time quanta of that of Round-Robin algorithm.

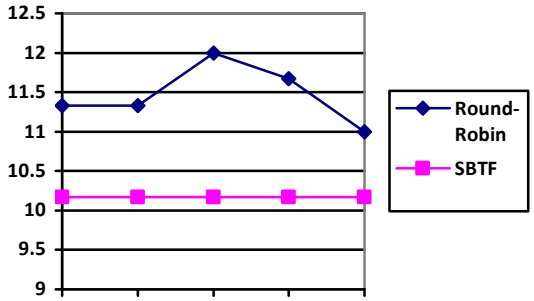
5. GRAPHICAL COMPARISION BETWEEN ROUND-ROBIN & SBTF ALGORITHMS

Now, we make the graphical comparison between the algorithms based on values from above example. Initially, we make comparison between the Completion times between algorithms.



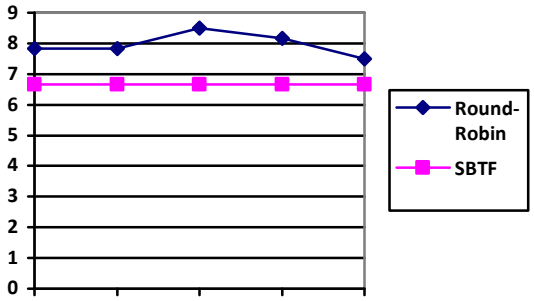
Graph 1: Comparison of Completion Times

Now comparing turnaround times between algorithms:



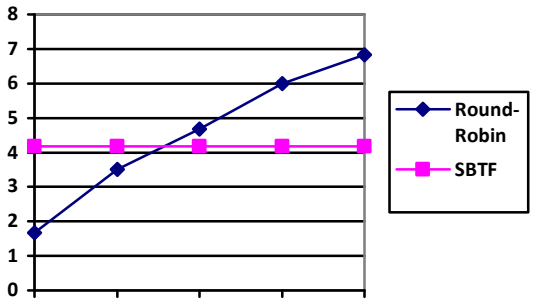
Graph 2: Comparison of Turnaround Times

Now comparing waiting times between algorithms:



Graph 3: Comparison of Waiting Times

Finally, comparing response times between algorithms:



Graph 4: Comparison of Response Times

From the above it is clear that the proposed algorithm SBTF has less completion time, turnaround time, waiting time and response time is less than that of Round-Robin algorithm.

6. CONCLUSION

In this paper a new scheduling method is presented which reduces the time taken to allocate the resources to the processes compared to that of round-robin algorithm. It is shown that we consider shortest job from the ready queue and execute it and execute rest of the processes for same amount of time in each step. An example is considered to

REFERENCES

[1] Shweta Jain, Dr. Saurabh Jain, "A Review Study on the CPU Scheduling Algorithms", IJARCCE, Vol. 5, Issue. 8, Aug 2016

[2] Noon, Abbas & Kalakech, Ali & Kadry, Seifedine, "A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average", IJCSI, Vol. 8, Issue 3, No. 1, May 2011.

[3] Y. H. Jbara, "A new Improved Round Robin-Based Scheduling Algorithm-A comparative Analysis," 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 2019, pp. 1-6.

[4] S. M., R. R. and V. P., "An Optimum Multilevel CPU Scheduling Algorithm," 2010 International Conference on Advances in Computer Engineering, Bangalore, 2010, pp. 90- 94.

[5] S. A, S. K R, Fiza, P. M S and V. D C, "A Study on Different Types of Scheduling Algorithm," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2019, pp. 721-724.

[6]"Round-robin scheduling," Wikipedia, 29-Mar-2020. [Online]. Available: https://en.wikipedia.org/wiki/Roundrobin_scheduling. [Accessed: 01-Apr-2020].

[7] Operating System Concepts by Abraham Silberschatz, Peter B. Galvin, Greg Gagne (first published 1982 by Wiley), 8th edition.