J. Appl. Environ. Biol. Sci., 5(12)42-46, 2015

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

An Optimized Shortest job first Scheduling Algorithm for CPU Scheduling

Muhammad Akhtar[1], Bushra Hamid[1], Inayat ur-Rehman[2], Mamoona Humayun[1], Maryam Hamayun[1] and Hira Khurshid[1]

University institute of information technology, PMAS Arid Agriculture University, Rawalpindi[1]
Department of computer Science, COMSATS institute of information technology, Islamabad[2]

Received: July 3, 2015
Accepted: October 30, 2015

ABSTRACT

Modern operating systems switch from single task environment to multitask environment. Maximizing the resource utilization is the demand of today's computing. There are number of scheduling algorithms are available to schedule CPU resources. These algorithms are not implemented in a real time environment due to high context switching, high response time, large average waiting time, less throughput and large turnaround time. The objective of this paper is to develop a new approach for preemptive shortest job first which helps to improve the efficiency of CPU for real time and time sharing environment. The proposed approach improves the drawbacks of preemptive shortest job first scheduling algorithm. A comparative analysis of proposed algorithm is done with round robin and preemptive SJF algorithms. The comparison results show that the proposed algorithm improve the system performance by decreasing the context switching to a desirable extent.
KEYWORDS: CPU, CPU Scheduling and Operating System

I. INTRODUCTION

CPU scheduling is the basic operation of the operating system. It's referred to in most undergrad book upon concepts of operating system for instance [1].CPU scheduling is a procedure which is used by the processes threads, user process or data gain permissions to the system given assets. The purpose of these scheduling techniques is to increase the system running performance and fair used or available resources. This topic is found in every book of operating system subject. The basic idea of scheduling is to share computer assets among the available no of process to request for system resources. Largely the computer resources are schedule before they use. CPU is the one of most important resource of the computer system resources. It is more important in operating system design to effectively and efficiently to schedule this operating system resource.

CPU scheduling decides which process perform when their execution and acquire how many and which resources. CPU scheduling is essential because it performs a major role in efficient resource utilization and it positively affects the overall system performance. If there are multiple process ready for their execution then it decides which process perform his execution when.

Scheduling belong to the area of operational research topics. In this work we describe a system with multiple processors. Here we see how efficiently the multi-processor systems schedule the multiple jobs for processing. Scheduling of processes on single processor are comparatively an easy problem to schedule multiple job for processing as compared to the problem for a systems with a multiple processors. A few concept of the down sides connected with this particular second option issue could be received through learning "job shop" as well as "flow shop" scheduling difficulties, observe for instance [2][3].

Whenever activities possess timing restrictions, because is actually standard associated with real-time processing techniques, scheduling these types of processes to satisfy their own timing restrictions is actually one of the significant problem which involves thinking's. Scheduling algorithms which are associated with useful worth with regard to real-time processing, types which consider real-world factors into consideration, possess just started to look. Provided the actual huge quantity of function that's been carried out through each the actual procedures investigation as well as pc technology towns within the arranging region, it's not possible to complete a good thorough study from the area. For their growing significance all of us additionally talk about the actual effect associated with quality-timeliness tradeoffs, fault-tolerance restrictions, as well as source reclaiming upon scheduling.

Obviously, a real-time based OS system should have the capability to execute unified scheduling and source distribution. The selections of collaborating responsibilities can acquire the assets they require the correct time in classify to satisfy time limitations. Along with appropriate arrangement technique, possibility involves limited

***Corresponding Author:** Inayat-ur-Rehman, Department of Computer Science, COMSATS institute of information technology, Islamabad, Pakistan.

primitives of OS. While use present paradigms of OS associated with permitting irrelevant waits with regard to assets or even occasions, or even dealing with an activity like an arbitrary procedure won't be achievable later on to satisfy the greater complex group of needs. It's additionally vital that you prevent needing to edit the actual working program for every application area[11].

There are many scheduling algorithms are available for scheduling the system resources. From the existing scheduling algorithms some are simple and easy to understand able for the users and some of them are more complex and can't easily understand able. Some scheduling algorithms run through command line interface and some have graphical user interface. Now we momentarily discus some of the available algorithms. To start we first quickly explain some existing generally used algorithms [1,7,10].Different simulations and models are used to find the efficiency of the existing algorithms [12].

As we know there are no of scheduling algorithms, every algorithm above has some kind of limitations/bottlenecks. Some gives less waiting time, then the other gives good through put. Some have starvation problems, so none of them is fully perfect. I have decided to overcome this problem by developing my own new scheduling algorithm which over comes the existing problem and provides the desired functionality. Proposed algorithm was more efficient and more user friendly. It fulfills the gaps presents in the existing scheduling algorithms. It facilitates users to use and explores the multiprogramming environment of the operating system.

II. RELATED WORK

A large no of papers are published on the topic of operating system scheduling algorithms in the recent years. This paper is the extension of the already published research papers [4][12]. Many Researchers develop their own scheduling algorithms which are published in their articles. These different algorithms have worked differently according to their model which is used to develop these algorithms. Some of the scheduling algorithms which are related to some extinct to this are discussed below,

The first come first served is a one of the easiest scheduling technique. Ready queue of this algorithm is like ready queue present in FIFO. The process selects from the head of ready queue and execution starts[3]. When the running process completes execution the process is terminated and removes from the process list. The new process is select for execution from the ready queue.

In priority based scheduling technique the available processes are arranged by system given priorities. The processes with high priority are selected first and perform their execution [5]. Here the process can be arranged with ascending or descending order depending on the system given preferences. When the running process completes its execution the process is terminated and removes from the process list. The new process is select for execution from the ready queue head.

The Round Robin is another scheduling technique used to schedule the processes. The ready queue is arranged like as the ready queue of FIFO. This is a preemptive scheduling technique and this algorithm use the concept of time quantum. The time quantum is the amount of time which is set and this time quantum is same for all the available process. Process selected from front of the queue and assigned to CPU for their execution. When quantum expires running routine is swap out and positioned in the last of the queue. When a routine completes its operations in amount of time given then the process is terminated and it removes from process list. The new process is select for execution from ready queue [6][12].

The "Short Remaining Time First Schedule" is a scheduling algorithm. In this scheduling algorithm the ready queue is organized according to the burst times of the processes. The routines which requires small amount of time to its completion are placed in front of the queue. This algorithm is also a preemptive scheduling algorithm. The process with smallest burst time is selected and assigned to CPU. If a process with lower burst time as compared to process which is running is comes in queue then the process which is running is preempted and the new process with small burst time starts its execution. When a process successfully completes its execution then the process is terminated and removes from the waiting process list. New process is select for execution from the front of ready queue [8][9].

In smallest jobs scheduled first (SJF) scheduling algorithm the ready queue is arranged in order to burst duration of CPU. In these algorithms the activities with shortest burst time are placed in front or the queue [4]. On the basis of this algorithm the process is select and transmits to CPU for execution. When the running process completes its execution the process is terminated and removes from the process list. The new process is selected for execution from the ready queue head.

Existing literature shows that SJF is an optimal scheduling algorithm as its average waiting time is less than other algorithms. It's through put is quite better than the other scheduling algorithms. Besides its numerous

advantages, it shows the problem of high context switching. Context switching is also an overhead. It decreases the system performance. So there is a need to minimize the context switching to maximize the system performance.

III. Proposed Algorithm

The proposed algorithm are hybrid scheduling algorithm. This algorithm consists of two algorithms, the first one is shortest job first (SJF) algorithms and the other one is the constrained on remaining burst time of running process. This algorithm belongs to category of preemptive scheduling algorithms. The concept behind this algorithm is that on start it selects the job having shortest burst time and starts the execution of this process. As it is declared above that it is preemptive scheduling algorithm, so when a new job arrives in ready queue it decide whether to preempt running process or not. If the remaining execution time of running process is less than or equal to burst time of newly arrived process than running process will not preempt. There are two possibilities, if the remaining execution time of running process is greater than burst time of newly arrived process. These possibilities are given below,

1. *If half of the remaining execution time of running process is less than the burst time of newly process, then running process will not preempt.*

2. *If half of the remaining execution time of running process is greater than the burst time of newly process, then we will preempt running process and assign processor to newly arrived process.*

Proposed approach is very helpful in optimizing the system performance by minimizing the context switching. Proposed approach provides sufficient reduction in context switching with a slightly change in average waiting time as compare to SJF. But this change in average waiting time is very small. But average waiting time of proposed approach is always less than the other scheduling techniques.

IV. Case study

We select five processes with their arrival time and burst time. These processes are scheduled by Round Robin, Preemptive SJF and proposed algorithm. We calculate and compare average waiting time, contest switching and turnaround time of each scheduling algorithm. We execute many processes sets and calculate and compare them. The example is shown below,

Table 1:A set of five processes with arrival time and burst time

Processes	Time of Arrival	Burst Duration (ms)
P1	0	4.0
P2	2	7.0
P3	5	5.0
P4	6	8.0
P5	8	9.0

1. Round Robin with quantum = 5

Gantt chart:

P1	P2	P3	P4	P5	P2	P4	P5

No. of context switches = 7
Average waiting time= 11.4 ms
Average turnaround time= 18ms

2. Shortest Remaining Job First:

Gant chart:

P1	P2	P3	P2	P4	P5	

No. of context switches = 5
Average waiting time = 6.6 ms
Average turnaround time = 13.2ms

3. Proposed Algorithm

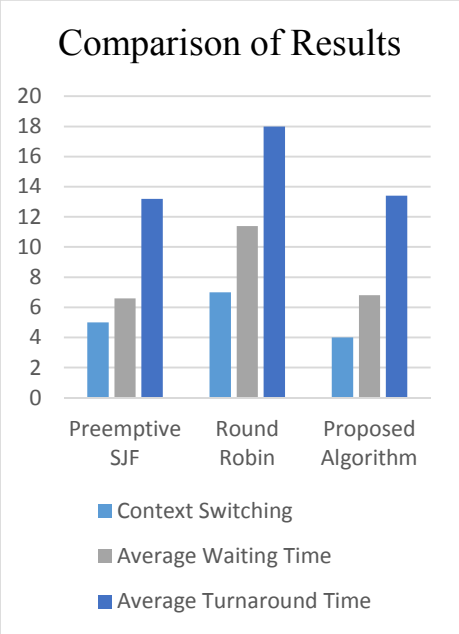
Gantt chart:

P1	P2	P3	P4	P5

No. of context switches = 4
Average waiting time= 6.8 ms
Average turnaround time= 13.4ms

V. RESULTS AND DISCUSSION

To check the running performance of the newly developed scheduling algorithm, we give the same set of processes to different scheduling algorithms. Comparison result shows that proposed approach has less context switching as compare to round robin and preemptive SJF. As we know that the context switching is an overhead and it minimize the system performance. Applicability of proposed approach can sufficiently maximize the system performance by minimizing the context switching. Performance evaluation of proposed algorithm as compared to round robin and preemptive SJF is shown in chart given below,



The table 2 given below shows the performance analysis of different scheduling algorithms is given below,

Table 2: Performance Analysis of Scheduling Algorithms

	Round Robin	FCFS	Preemptive SJF	Proposed Algorithm
Decision Method	Preemptive	Non Preemptive	Preemptive	Conditional Preemptive
Response Spell	Low	High	Low for short processes	Low for short processes
Waiting Spell	Small	High	Low for short processes	Low
Throughput	Less for small quantum	High	High	High
Starvation	No	No	Can possible	Can possible
Context Switching	Large	No	Large	Small

VI. Conclusion

As we see the previous results these shows that when we use First Come First Serve its computational overhead is small, but we notice that it produce low through put hence its performance was poor. The other drawback of FCFS is that it has high waiting time. The Shortest Job First served is an ideal scheduling algorithm to reduce the average waiting time of the given set of available process. Thought SJF gives less average waiting time for all available process but it has more waiting time for process which requires more time to complete their execution when we compare it with FCFS algorithm. In case short process arrive continuously then the starvation for long process will be possible. The round robin is the algorithm which provides the faire CPU sharing to all the available processes. In round robin a fix quantum is assigned to all the process in the ready line. The processes with short burst time can complete their execution in single given time quantum, it shows relatively better response time. In case of processes with long burst time it gives comparatively long turnaround time and their waiting time also increase. There is also a possibility for processes with small time quantum can wait for long time for their turn.

When we talk about the priority based scheduling, we can see that starvation is also appears here. The process with high priority will execute first so the process with low priority may never get the chance for their execution,

this creates the starvation for lower priority processes. The algorithm witch we proposed is an extension of preemptive SJF. We have already seen that SJF is an optimal scheduling algorithm as its average waiting time is less than other algorithms. It's through put is quite better than the other available algorithms. Proposed approach provides sufficient reduction in context switching with a slightly change in average waiting time as compare to SJF. But this change in average waiting time is very small. So by performing fewer contexts switching this algorithms increase the system performance.

REFERENCES

[1] Silberschatz A, Galvin P, Gagne G. "Operating system concepts". 7th Edition. Hoboken, NJ: J. Wiley & Sons; 2005.

[2] Błażewicz J, Domschke W, Pesch E. "The job shop scheduling problem: Conventional and new solution techniques". European Journal of Operational Research. 1996.

[3] Rajendran C, Holthaus O. "A comparative study of dispatching rules in dynamic flowshops and jobshops". European Journal of Operational Research. 1999.

[4] Shah, S. N. M., Mahmood, A. K. Bin, & Oxley, A. (2009). Hybrid Scheduling and Dual Queue Scheduling.2009 2nd IEEE International Conference on Computer Science and Information Technology.

[5] MilenkovicM."Operating System Concepts and Designs".2nd Edition. McGraw Hill International 1992.

[6]…....http://siber.cankaya.edu.tr/OperatingSystems/ceng328/node125

[7] Lawrence A.W, Badre A, Stasko J. "Empirically Evaluating the Use of Animations to Teach Algorithms", Proceedings of the 1994 IEEE Symposium on Visual Languages, 1994.

[8] Hong C, Caesar M, Godfrey P. Finishing flows quickly with preemptive scheduling. SIGCOMM Comput Commun Rev. 2012.

[9]…....http://www.gitam.edu/eresource/comp/gvr(os)/5.3.htm

[10] Beck L., L. "Software System", Ed, 3rd, Addison Wesley, Issue in 1997

[11] Ramamritham K, Stankovic J. Scheduling algorithms and operating systems support for real-time systems. Procedings of IEEE. 1994.

[12] Singh, A., Goyal, P., Batra, S. "An Optimized Round Robin Scheduling Algorithm for CPU Scheduling". International Journal on Computer Science and Engineering Vol. 02, 2010.

[13]. Ur-Rehman, I., Elahi, M., & Mohsin, S. (2015). Learning Technologies and Developing Countries. J. Appl. Environ. Biol. Sci, 5(4), 7-14.