Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.199

IJCSMC, Vol. 8, Issue. 5, May 2019, pg.176 – 181

COMPARATIVE ANALYSIS BETWEEN FIRST-COME-FIRST-SERVE (FCFS) AND SHORTEST-JOB-FIRST (SJF) SCHEDULING ALGORITHMS

Gideon Dadik Bibu[1]; Gloria Chizoba Nwankwo[2]

[1,2]Department of Computer Science, University of Jos, Nigeria

[1] dadikg@unijos.edu.ng

Abstract— Scheduling is a fundamental operating system function, since almost all computer resources are scheduled before use. The CPU is one of the primary computer resources. Central Processing Unit (CPU) scheduling plays an important role by switching the CPU among various processes. A processor is the important resource in computer; the operating system can make the computer more productive. The purpose of the operating system is to allow the process as many as possible running at all the time in order to make best use of CPU. The high efficient CPU scheduler depends on design of the high quality scheduling algorithms which suits the scheduling goals. In this project, two fundamental CPU scheduling algorithms (First-Come-First-Serve and Shortest Job First) for a single CPU were reviewed and compared to show which algorithm is more suitable for certain processes and how their scheduling are being implemented. A theoretical analysis that subject the algorithms to the same condition is performed through the use of exemplary job processes to determine the best among the algorithms. Average Waiting time and Average Turnaround time scheduling criteria were used to evaluate the algorithms and it was discovered that Shortest Job First scheduling algorithm gives more optimal performance of scheduling processes than the First-Come-First-Serve scheduling algorithm.

Keywords— Scheduling, Shortest Job First, First Come First Serve, Central Processing Unit, Algorithms

I. INTRODUCTION

Over the years, scheduling has been the focus of intensive research, and many different algorithms have been implemented. Today, the emphasis in scheduling research is on exploiting multiprocessor systems, particularly for multithreaded applications, and real-time scheduling.

In multiprogramming systems, multiple processes exist concurrently in main memory. Each process alternates between using a processor and waiting for some event to occur, such as the completion of an I/O operation. Multiprogramming refers to multiple processes from multiple programs while multitasking is reserved to mean multiple tasks within the same program that may be handled concurrently by the operating system (OS). In fact, multiprogramming systems are designed to maximize CPU

usage. Multiprogramming aims at having some process running at all times in order to maximize CPU utilization. that is, keep the CPU busy as much as possible by having it work on more than one program at a time. The processor is kept busy by executing one process while the others wait, hence the key to multiprogramming is scheduling. The CPU, being one of the primary computer resources used in a multiprogramming system needs to be scheduled, and its scheduling is central to the operating-system design [1].

In multiprogramming, the sharing of computing time among programs is controlled by a clock, which interrupts program execution frequently and activates a monitor program. The monitor saves the registers of the interrupted program and allocates the next slice of computing time to another program and so on. Switching from one program to another is also performed whenever a program must wait for the completion of input or output. Thus, although the computer is only able to execute one instruction at a time, multiprogramming creates the illusion that programs are being executed simultaneously, mainly because peripherals assigned to different programs indeed operate in parallel. Scheduling is the method by which threads, processes or data flows are given access to system resources. This is usually done to load balance and share system resources effectively or achieve a target quality of service. The need for a scheduling algorithm arises from the requirement for most modern systems to perform multitasking (executing more than one process at a time) and multiplexing (transmitting multiple data streams simultaneously across a single physical channel) [2].

One of the most important problems in operating systems designing is CPU Scheduling and challenge in this field is to build a program to achieve proper scheduling. In the case of priority scheduling algorithm when similar priority jobs arrive, then FCFS is used and the average waiting and turnaround time is relatively higher. The process that arrives first is executed first, no matter how long it takes the CPU. So in this case if long burst time processes execute earlier then, other processes will remain in waiting queue for a long time. There has not been a clear comparison of scheduling algorithms with an emphasis on user involvement. This is because past researches have attempted to generally compare scheduling algorithms based on some parameters like response time, and throughput. This research work has focused on comparing two very important scheduling algorithms based on average waiting time and average turnaround time scheduling criteria, with as many job population as possible.

II. LITERATURE REVIEW

Scheduling policies of Central Processing Unit (CPU) for computer systems were discussed in [3]. A number of problems were solved to find the appropriate among them. Therefore, based on performance, the shortest job first (SJF) algorithm is suggested for the CPU scheduling problems to decrease either the average waiting time or average turnaround time. Also, the first come first serve (FCFS) algorithm is suggested for the CPU scheduling problems to reduce either the average CPU utilization or average throughput. They recommended that the only accurate method to evaluate a scheduling algorithm is its implementation which was not done.

Similarly, [1] presents a state diagram that depicts the comparative study of various scheduling algorithms for a single CPU and showed which algorithm is best for the particular situation. Their paper gave a representation on what is going on inside the system and why a different set of processes is a candidate for the allocation of the CPU at different times. In a nutshell, the objective of their paper was to analyse the high efficient CPU scheduler on design of the high quality scheduling algorithms which suits the scheduling goal. However, they were not able to arrive at a definite conclusion as to which algorithm is best.

In their paper, [4] made a comparison of existing scheduling algorithm on the basis of seven parameters namely pre-emption, complexity, allocation, application, waiting time, usability, and type of system. They also observed that there is no one algorithm satisfying all the seven basis parameters chosen for analysis and suggested needs for improvement. However, no direction for the improvement was provided and has been referred to as future works.

Reference [2] concentrated on the scheduling algorithms utilised for scheduling processes in a multiprogramming system and made a correlation of a number of scheduling algorithms on the premise of some parameters. Different papers were also reviewed that demonstrated no scheduling algorithm is perfect in fulfilling the conditions and inferred that further reviews which enhance scheduling algorithms need to be done.

Comparative analysis of various scheduling algorithms were presented in [5] including First Come First Serve, Shortest Job First, Round Robin, Priority, Longest Job First, Multilevel queue, multilevel feedback queue and priority scheduling with their pre-emptive as well as non-pre-emptive modes. From their study, they deliberated that scheduling is one of the utmost significant responsibilities of the operating system. Depending upon the type and requirements of the operating system, to make the most appropriate system a good scheduling algorithm must be chosen. All the necessary factors that affect the performance of the computer system i.e. waiting time, response time, turnaround time must be considered to make that decision. First come first serve is the simplest scheduling algorithm to implement. It is most suitable for interactive and batch systems. The shortest job first algorithm works in both pre-emptive and non-pre-emptive manner and the length of next CPU request must be known. Round Robin algorithm treats all process equally and allocate the same time quantum to every process but it's very hard to decide the proper time quantum. It's best suited for time sharing environment. The multilevel queue scheduling algorithm faced the problem of starvation to remove it multilevel queue is transformed to multilevel feedback queue scheduling algorithm.

III. METHODOLOGY

In order to evaluate the performance of the job scheduling algorithms, the scheduling criteria, average turnaround time, and the average waiting time parameters were used. The test cases used to evaluate the scheduling strategies are also shown.

A. Choosing a Scheduling Algorithm.

Different CPU scheduling algorithms have different properties, and the choice of a particular algorithm may favour 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. The criteria include the following:

1) CPU utilization: This is the percentage of time that the CPU is busy. CPU utilization refers to a computer's usage of processing resources, or the amount of work handled by a CPU. Actual CPU utilization varies depending on the type of managed computing tasks. Certain tasks require heavy CPU time, while others require less because of non-CPU resource requirements. CPU utilization may be used to gauge system performance.

2) Throughput: This is the number of processes completed in a unit of time. It is the average rate of successful processes completed over a specified period of time. It is measured in bits per second (bps).

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. Turnaround time is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.

4) Waiting time: This is the total amount of time that a process is in the ready queue waiting in order to be executed. Sometimes the waiting time could be short and at other times, it could be long.

5) Response time: In an interactive system, turnaround time may not be the best criterion. Often, a process can produce some output fairly early 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. This measure, called response time, is the time it takes to start responding, not the time it takes to output the response. The turnaround time is generally limited by the speed of the output device.

It is desirable to maximize CPU utilization and throughput and minimize turnaround time, waiting time, and response time. In most cases, the average measure is optimized. However, under some circumstances, it is desirable to optimize the minimum or maximum values rather than the average. For example, to guarantee that all users get good services, we may want to optimize the maximum response time [6].

B. The Proposed System

The proposed system compares the performance of the scheduling algorithms after they have been implemented. The system provides a clean and convenient way to test the given data and do the analysis of CPU scheduling algorithms. For the purpose of analysis and testing, the user first specifies the required information such as number of simulations and processes, and then the algorithms can be executed producing output in appropriate readable format.

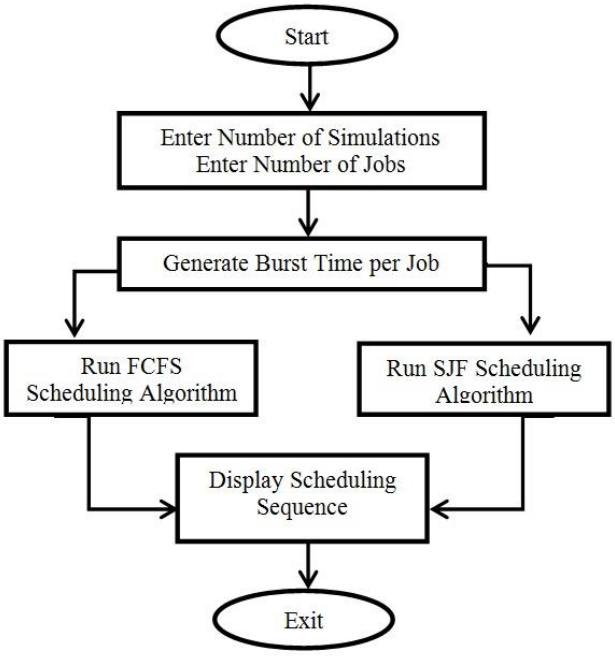

Fig. 1 System block diagram of the system

Fig. 1 illustrates the system in which the principal parts or functions of the system is represented by blocks connected by lines that show the association between the blocks. The user would be required to enter the number of simulations and the number of jobs. The system generates the processes and computes the FCFS and SJF algorithms. The output is then displayed to the user.

Fig. 2 illustrates the activities that will be carried out in the system with emphasis on the sequence and conditions of the flow. The system works with the user input and computes and displays the average waiting time and average turnaround time of the scheduling algorithms based on the user input and a bar chart is generated and displayed afterward showing a clear comparison of the two algorithms.

1) *First Come First Serve scheduling algorithm:* The most intuitive and simplest technique is to allow the first process submitted to run first. This approach is called as first-come, first-served (FCFS) scheduling. In effect, processes are inserted into the tail of a queue when they are submitted. The next process is taken from the head of the queue when each finishes running. This idea is illustrated in the four state diagram of Fig. 3.

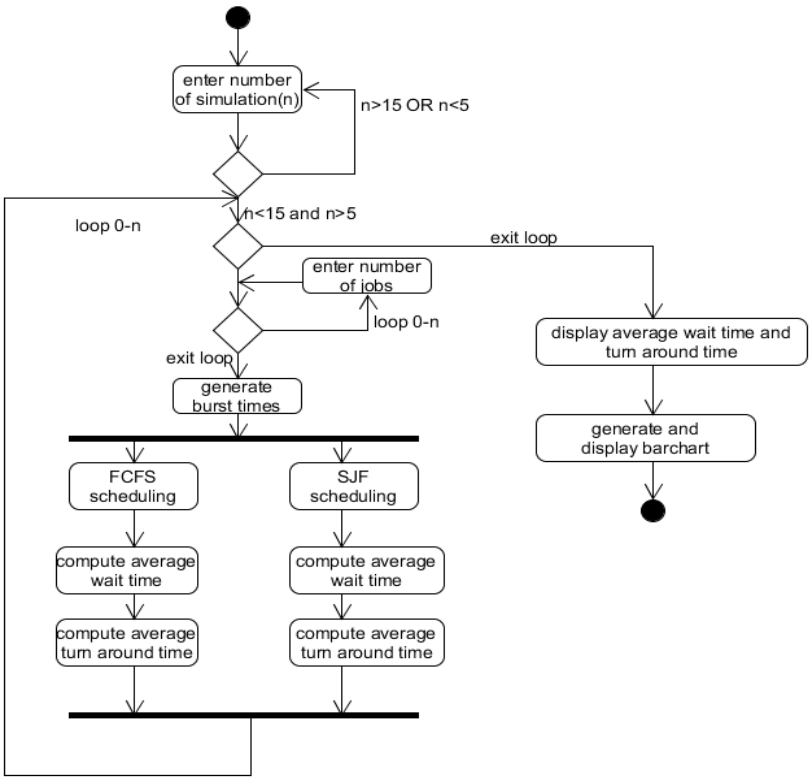

Fig. 2 Activity Diagram of the System

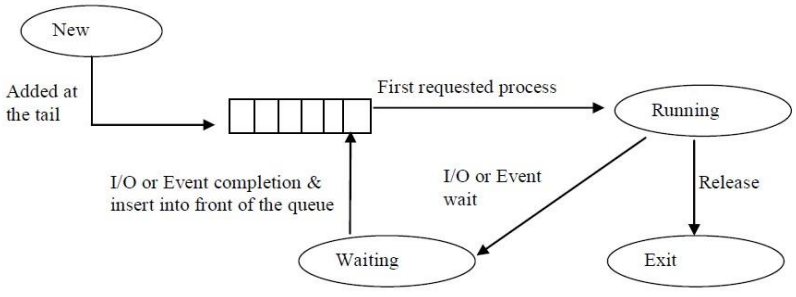

Fig. 3 State diagram showing the first come first serve scheduling

2) Shortest Job First scheduling algorithm: The process is allocated to the CPU which has least burst time. A scheduler arranges the processes with the least burst time in head of the queue and longest burst time in tail of the queue. This requires advanced knowledge or estimations about the time required for a process to complete. This idea is illustrated in the four state diagram of Fig. 4.

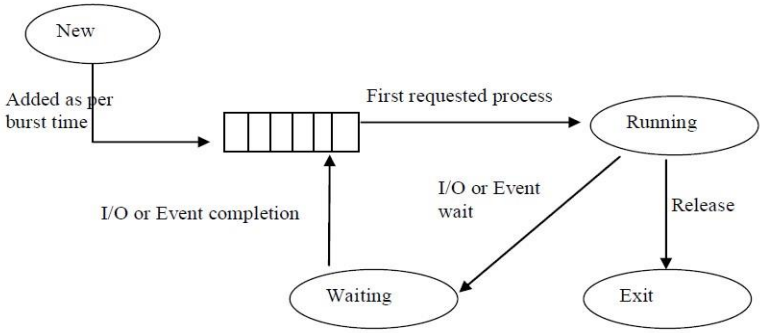

Fig. 4 State diagram showing the shortest job first scheduling

IV. Discussion of Results

The result obtained is shown using a bar chart for five set of simulations tests as shown in Fig. 5.

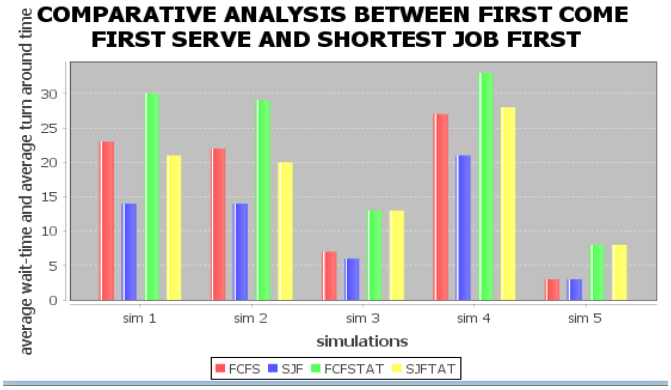

Fig. 5 Bar Chart showing test result of comparison of FCFS and SJF Scheduling Algorithms

Table 1 shows the comparative analysis of the result generated after computation of the average waiting time and average turnaround time of both scheduling algorithms.

From Fig. 5 showing the bar chart generated by the average values of the wait times and turnaround times of both FCFS and SJF for each of the test cases, it can be seen that Shortest Job First (SJF) performs better than First-Come-First-Serve, because it is obvious from the chart that smaller jobs are starved in FCFS thereby, affecting the overall average performance of FCFS.

In some cases where the number of jobs is limited, FCFS performs approximately equal to SJF as it is seen in test cases 3 and 5. So based on the finding of this work, it is concluded that Shortest Job First (SJF) gives the best result and is suggested for CPU scheduling problems to decrease either the waiting time or the average turnaround time.

The discussed result is summarized in the table 1.

TABLE I
COMPARATIVE ANALYSIS OF RESULT GENERATED

Test Cases	Average Waiting Time		Average Turnaround Time	
	FCFS	SJF	FCFS	SJF
Test Case 1	23	14	30	21
Test Case 2	22	14	29	20
Test Case 3	7	6	13	13
Test Case 4	27	21	33	28
Test Case 5	3	3	8	8

V. CONCLUSION

This work has succeeded in comparing FCFS and SJF scheduling algorithms based on only the parameters of average waiting time and average turnaround time.

One of the limitations of this research work is its stochastic nature as it contains probabilistic elements i.e. real life data were not worked with and multiple runs were required to analyse the output.

Under different circumstances of probably more resources and time, this research work would have also considered using more parameters for the comparison and designing an algorithm to reduce the Average waiting time and Average Turnaround time of CPU.

REFERENCES

[1] N. Goel and R.B. Garg, "A Comparative Study of CPU Scheduling Algorithms", *Internal Journal of Graphics & Image Processing*, Vol. 2, Issue 4, November 2012.

[2] Y. A. Adekunle, Z. O. Ogunwobi, A.S. Jerry, B. T. Efuwape, S. Ebiesuwa, and J. Ainam, "A Comparative Study of Scheduling Algorithms for Multiprogramming in Real-Time Systems", *International Journal of Innovation and Scientific Research* ISSN 2351-8014, *Vol.12, No.1*, 180-185, 2014.

[3] J. Patel and A. K. Solanki, "CPU Scheduling: A Comparative Study", *Proceedings of the 5th National Conference, IndiaCom-2011*, Computing for Nation development, 2011.

[4] B. Ankur, S. Rachhpal and Gaurav, "Comparative Study of Scheduling Algorithms in Operating System", *International Journal of Computers and Distributed Systems, Vol. No. 3*, Issue I, 2013

[5] S. K. Nager and S. G. Nasib, "Comparative Study of Various CPU Scheduling Algorithm", *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Vol 6*, Issue 2, 2017.

[6] S. Robbins, A disk head scheduling simulator. In *ACM SIGCSE Bulletin* (Vol. 36, No. 1, pp. 325-329). ACM, March 2004.

