iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** VI **Month of publication:** June 2020

DOI: http://doi.org/10.22214/ijraset.2020.6400

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VI June 2020- Available at www.ijraset.com

A New Hybrid Scheduling Algorithm for Enhancement of CPU Performance

Kolipakula Yuvanaga Venkata Manishankar[1], Rajesh K[2], Sujeeth ES[3], Kakelli Anil Kumar[4]
[1, 2, 3]*Student,* [4]*Associate Professor, Vellore Institute of Technology, Vellore, (Tamil Nadu), INDIA*

Abstract: ***Most of the existing CPU scheduling algorithms are not applicable in real-time environment due to major limitations like high in turnaround time, waiting time, and response time, also high context switches and less throughput. Our main aim is to provide a new hybrid CPU Scheduling algorithm that can enhance the CPU performance in a real-time environment. We have designed hybrid version of algorithm by integrating Highest Response Ratio Next (HRRN) and standard deviation (SD) scheduling algorithms. It combines the strengths of standard deviation, elimination of starvation, and HRRN priority scheduling. It assigns different priorities to processes thereby dealing with aging issue. Thus, our algorithm corrects major limitations of the existing CPU scheduling algorithms. We'll also provided a comparative analysis of our proposed new hybrid algorithm with the prevailing scheduling algorithms based on average turnaround, average waiting time, varying time quantum, and many context switches.***

Keywords: CPU Scheduling, Priority Scheduling, HRRN, Standard Deviation, Hybrid algorithm.

I. INTRODUCTION

In the area of information and communication technologies (ICT), Central processing unit (CPU) scheduling is the procedure by which operating system processes are allowed to access the system resources such as communication, bandwidth, processor cycles and so on.

Therefore, there is a requirement of scheduling algorithms for the computer operating systems to perform multitasking and multiplexing. Scheduling of processors is an important operating system functionality which is used to determine when should the process run, if there are more than one process waiting for the resources. Scheduling algorithm plays an important role in the utilization of the resources.

There are many scheduling algorithms such as Round Robin scheduling, Shortest Job First Scheduling, First Come First Serve, Priority Scheduling, and so on, yet because of various drawbacks these algorithms are not popularly used in real-time operating scenario except for the Round Robin scheduling. The assumptions that are considered in CPU scheduling are as per the following:

A. The task pool comprises of run-able processes waiting for the CPU resources
B. All the processes are considered to be independent. These processes will compete for resources.
C. The important aspect of the scheduler is to allocate the CPU resources to various processes in such a way that can optimize the performance and provides better throughput.

The main function of the scheduler is to select which process to run next. Operating systems consists of three different types of schedulers that is short-term scheduler, medium-term scheduler, and long-term scheduler. The short-term scheduler chooses the processes from the ready queue and provides the CPU to any of them according the criteria. Medium-term scheduler takes out the processes from memory which will be used to decrease the level of multi programming. This type of operations results in exchanging that is a way in which a process can be swapped temporarily to secondary memory from main memory, later again swaps back to the main memory.

Long-term scheduler chooses the process from the execution pool and it will place them into memory for the execution of processes. There are two significant kinds of systems. Hard real-time and Soft real-time systems. In Hard Real-Time System, it must be completed before the specified deadline otherwise, a deadly circumstance may emerge yet in Soft Real-Time System, missing the deadline is undesirable.

In real-time systems, every individual operation or task must be done after the ready time and should be finished before its dead line, a trial has been made to fulfill these limitations. If a real time process having a large CPU burst time will be ending up with the problem of starvation.

II. KEY ELEMENTS FOR EFFECTIVE SCHEDULING

While developing the algorithm, the architect needs to anticipate many requirements. Like different frameworks to be utilized and customer necessaries.

A. High Efficiency or Throughput: Throughput can be defined as the number of jobs completed for unit time. An algorithm throughput must be as max as possible.
B. Remove starvation or blocking: High waiting time for a job may leads to starvation. So, the scheduling algorithm must be robust enough to assign minimum waiting time for jobs or processes and removes starvation.
C. Reduce the overhead: Overhead may lead to the exhaustion of assets. If we utilize the system's resource efficiently, general execution enhances extraordinarily.
D. Best assigning of priorities: The algorithm should assign the priorities viably so that the system can support more executions.
E. Maintain a good relationship between utilization and responses: System must be able to keep the resources occupied.
F. Support jobs that show desirable behavior.

A system might finish these objectives in many ways. The scheduler can stall the indefinite blocking of jobs via aging.

III. SCHEDULING CRITERIA

Every CPU scheduling algorithm might be having different properties. If we choose one algorithm may be favorable to one category of processes than others. Hence the scheduling criteria are:

1) *Context Switch:* In order to resume execution at a later point, a pre-empted job is stored and then restored later. Context switch involves lot of computation. This wastes memory, time and thereby maximizing scheduler overhead. So, the operating system should optimize this.
2) *Throughput:* Number of jobs completed for unit time.
3) *CPU Utilization:* Tells the business of the CPU. CPU utilization must be as high as possible.
4) *Turnaround Time (TAT):* Time taken to finish a job. It is the execution time from submission of a job to its completion. TAT can also be achieved by adding execution time on CPU and waiting time.
5) *Waiting Time:* The total waiting time of a process before its execution in ready queue.
6) *Response Time:* Time between task submission to the response received.

Hence an efficient algorithm for scheduling must have: -

a) Less context switches.
b) High CPU utilization.
c) High throughput.
d) Less turnaround time.
e) Less waiting time.

IV. EXISTING SCHEDULING ALGORITHMS

A. Round Robin Scheduling

This algorithm was implemented mainly for time-shared systems not for the real time systems. Time slice or time quantum is defined in case of RR algorithm, which refers to duration for which the process is allocated to the CPU and executed. The processes which have to be executed are kept in a circular queue which has a head and a tail. The scheduler will go through the entire queue and allocate the CPU resource to every process for the required time interval of one quantum yet the problem is that all the processes are arranged according to the FCFS. Arriving processes will be added to the end of the queue. Round robin is more favorable for the process which is having short CPU burst time. The limitations of the round robin scheduling algorithm that affects execution process time shared system are as follows.

1) *Large Waiting And Response Time With High Rates Of Context Switching:* Since Real-time programs must guarantee response within specified time constraints therefore larger waiting, response time and high rates of context switching affect the system's performance and delay the results.
2) *The Throughput Is Very Low:* Number of process completed per time unit is called throughput. When Round robin is implemented in soft real time systems it will be resulted in low throughput that prompts to performance reduction of the system that is due to high context switching. Suppose, there are low context switches then it will result in high throughput. Throughput and context switches are inversely proportional to each other. For this reason, the algorithm is not suitable for real time systems.

B. Priority Scheduling

In this algorithm, each and every process will have a priority number which is assigned by the operating system. According to the order of their priority the algorithm puts the processes in the ready queue. These processes are then allocated to the CPU one by one. Lower priority processes will get disturbed by incoming higher priority ones. Depending on the priority of the processes response time and waiting time are calculated. Higher the priority lesser the waiting time. The process which has early deadlines will have higher priority numbers. There is a chance of starvation of the processes that has low priority by continuous arriving of processes that has higher priority. Therefore, a combination of the above two algorithms will be needed to make an algorithm fit for real time systems.

C. Shortest Job First (SJF)

Any process having shortest burst time should be scheduled first. There are two ways of implementation.
Non-preemptive and other is Preemptive.
Shortest job first -Non-Preemptive Processes are schedules according to the burst time in the ascending order that is the process having shortest burst time will be scheduled first. Shortest job first - Preemptive: In Shortest job first Preemptive after every unit of time the burst times of processes are checked. After finishing of one-unit time, the process having the shortest burst time will be scheduled next. This algorithm is also known as Shortest Remaining Time First.

D. Longest Job First (LJF)

This algorithm is opposite of SJF. The processes having larger burst time are scheduled first. There are two ways of implementation. Longest job first- Non-Preemptive Processes are schedules according to the burst time in the descending order that is the process having largest burst time will be scheduled first. Longest job first- Preemptive In Longest job first Preemptive after every unit of time the burst times of processes are checked. After finishing of one-unit time, the process having the longest burst time will be scheduled next. This algorithm is also known as Longest Remaining Time First.

E. Highest Response Ratio Next (HRRN)

HRRN algorithm executes the "aging priority" schema, in this the priority of the process is boosted according to its waiting time. The priority is calculated using the equation 1. W is the time spent waiting for the process and s is the expected service time

$$Priority = (w + s) / s \ldots (1)$$

F. Multilevel Queue Scheduling

In this algorithm we will divide the load into different queues. Each load will be having its own scheduling criteria. On the basis of the priority of the process, the process which has to be placed in ready queue is decided. Generally, the processes with high priority are kept at the top of the ready queue. The disadvantage is that the processes which are kept at low level ready queue will be experiencing starvation.

G. Multilevel Feedback Scheduling

In Multilevel feedback scheduling, the processes whose execution is not finished in the particular level are placed in next level. The major aim of this scheduling algorithm is to say no to starvation.

H. First Come First Serve

In first come first serve algorithm, the process which enters the ready queue first will be executed first. The process whose arrival time is less when compared to the other processes in the ready queue will gets executed first.

I. Hybrid CPU Scheduling

This algorithm can do in both non-preemptive and preemptive. In Hybrid CPU Scheduling algorithm, we will find a parameter called total elapsed time. This is the addition of both burst time and arrival time. First, we have to calculate total elapsed time and sort the processes from less total elapsed time to high total elapsed time. The processes with low total elapsed time will gets executed first and this process continues till all the processes gets completed. Here due to the consideration of burst time in the algorithm, this algorithm can perform in both non-preemptive and preemptive in nature. This algorithm will increase throughput and also it will decrease both waiting time & turnaround time.

J. SD Scheduling

In this algorithm, first we have to calculate the standard deviation of the processes using the below formula. After this, the processes with nearer to the standard deviation value should be keep in the queue and this process continues till all the processes get into the queue. The process which was having the burst time value nearer to the standard deviation value will gets executed first. The equation 2 for standard deviation scheduling. Here X is burst time of a process and X denotes average burst time value of all processes, N is number of processes.

$$SD = \sqrt{\frac{\sum X - \overline{X)}}{N}} \ldots (2)$$

K. Job Mix Scheduling

In this algorithm, the processes with less burst time are kept first when compared to the processes with high burst time in the separate queue. This algorithm can solve many problems like starvation with higher burst time jobs and gives comparatively better average waiting time than many other algorithms. This algorithm is very easy and so simple for implementing, but there will be increase in overhead due to the usage of another queue.

V. PROPOSED NEW HYBRID ALGORITHM

We have considered two methods one is Standard deviation and another one is Highest Response Ratio Next. First, we have to calculate standard deviation of the processes using the below mentioned formula. Then the process with nearer to the standard deviation value will be assign priority as '1', next nearer value will be assigned as '2'and this process will continue till all the processes gets their assigned priority value as per equation 3.

$$SD = \sqrt{\frac{\sum X - \overline{X)^2}}{N}} \ldots (3)$$

3. Calculate highest response ratio next using the below mentioned formula. Then the process with higher response ratio value will be assign priority as '1' , and the next highest response ratio will be assigned as '2' and this process will continue till all the processes gets their assigned priority value as per the equation 4.

$$R = (w + s) / s \ldots (4)$$

4. In this algorithm we will give equal priority to both the methods by using the below equation 5.

$$Hybrid\ Priority\ (HP) = 0.5 * R' + 0.5 * SD' \ldots (5)$$

R' is the priority number according to R, SD' is priority number according to SD

5. Finally the process with process with lower hybrid priority value will be executed first, then the next lower value will be executed next and this process continues till all the processes finished their execution.

VI. ALOGORITHM

Table 1. The proposed new hybrid algorithm

1. Begin
2. Read the process id, arrival time and burst times of the all the processes
3. Calculate standard deviation of all the processes using the eq (1) and assign the priorities to all the processes where SD assigns priority as '1' and continues.
4. Calculate the highest response ratio next of all the processes from eq (2) and assign the priorities to all the processes i.e., the process with higher response ratio assigns priority as '1' and continues.
5. Calculate Hybrid priority using eq (3).
6. If two processes having same hybrid priority value?
If true, did processes having same hybrid priority value arrives at a particular time?
If true, the process with lower process id will be executed first.
Else, did processes having same hybrid priority value arrives at different time?
If true, the process with less arrival time will be executed first.
7. Processes executes according to the hybrid priority of it.
8. If burst time is zero then process leaves the ready queue and calculate waiting time and turnaround time of the process
9. If ready queue is empty, then calculate average waiting time, average turnaround time and average response time of all the processes
10. End

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VI June 2020- Available at www.ijraset.com

VII. THE COMPARATIVE ANALYSIS OF EXSITINGTECHNIQUES

The performance of the proposed new hybrid scheduling algorithm has been analyzed and compared with other prominent algorithms as shown below in Tables 2, 3, 4 and 5 and also the figures 1, 2, 3and 4.

Table 2. The comparative analysis various algorithms with proposed Hybrid algorithm

S.No	Algorithm	Implementation Complexity	Preemption	Starvation	Prior knowledge of priorities	Performance
1	First Come First Serve	Easy	No	No	No	large average waiting time
2	Shortest Job First	Easy to implement in Batch system	No	Yes	No	Minimum average waiting time
3	Shortest Remaining Time First	Impossible to implement the interactive or real time systems	Yes	Yes	No	Short jobs are given preference
4	Round Robin	Easy	No	No	No	Fixed time to each process
5	Longest Remaining Time First	Impossible to implement the interactive or real time systems	Yes	Yes	No	Longer job is given preference
6	Longest Job First	Easy	No	Yes	No	Large average turnaround time
7	Priority Non-Preemptive	Mildly	No	Yes	Yes	Best for batch systems
8	Priority Preemptive	Complex	Yes	Yes	Yes	Good but problem of Starvation
9	Multi-Level Queue	Complex	No	Yes	Yes	Good but problem of Starvation
10	Multi-Level Feed Back Queue	More Complex	No	No	Yes	Starvation problem is removed
11	Hybrid scheduling	Middle Complex	No	No	No	Best for batch systems
12	Standard deviation	Easy	No	Yes	No	Suffer from Starvation
13	Job Mix	Mildly complex	No	No	No	Increase in over head
14	HRRN	Easy	No	Yes	Yes	Increase in Throughput
15	SD Based HRRN	Easy	No	Yes	Yes	both waiting Time and priority

VIII. RESULTS AND ANALYSIS

Table 3. The Arrival Time and Burst time of New Hybrid Algorithm

Process Number	Arrival Time	Burst time
1	1	3
2	3	9
3	5	6
4	2	14
5	4	10
6	6	7

Table 4. The comparative analysis of the New Hybrid Algorithm and others w.r.to Av. TAT and Av. WT

S. No	Algorithm	Average TAT	Average WT
1	First Come First Serve	22.67	14.5
2	Shortest Job First	20.33	12.33
3	Shortest Remaining Time First	20.66	12.50
4	Round Robin	30.66	22.50
5	Longest Remaining Time First	44	13.8333
6	Longest Job First	32.66	24.5
7	Priority Non-Preemptive	26.33	18.16
8	Priority Preemptive	26.33	18.16
9	Hybrid scheduling	20.66	13
10	Standard deviation	20.67	12.5
11	Job Mix	25	16.833
12	HRRN	24.33	16.16
13	Proposed New Hybrid SD Based HRRN	21.16	13

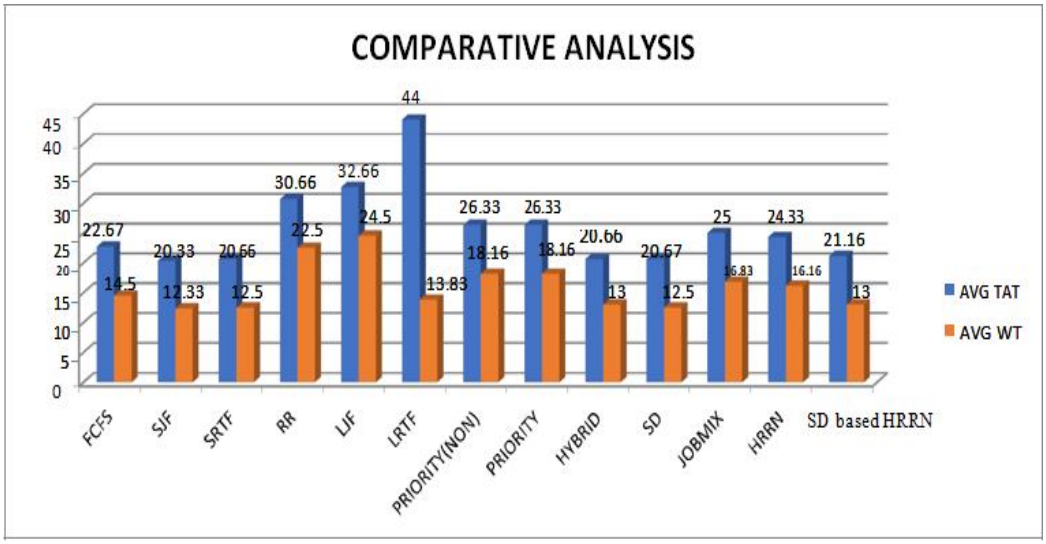


Figure 1. The comparative Analysis of Hybrid (SD based HRRN) and other Algorithms

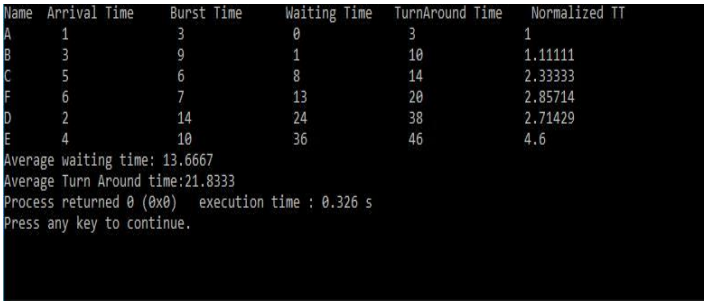


Figure 2. The input and output of Hybrid (SD based HRRN) and other Algorithms

Table 5. The performance analysis of the New Hybrid Algorithm (SD based HRRN) w.r.to Av. TAT and Av. WT

S. No	Algorithm	Average TAT	Average WT
1	First Come First Serve	26	17.83
2	Shortest Job First	23	14.83
3	Shortest Remaining Time First	22.83	14.66
4	Round Robin	35.83	27.66
5	Longest Remaining Time First	47.5	39.34
6	Longest Job First	27.5	27.17
7	Priority Non-Preemptive	26.33	19.33
8	Hybrid scheduling	23.84	15.667
9	Standard deviation	23.84	15.667
10	Job Mix	28.5	30.25
11	HRRN	23.66	15.5
12	Proposed New Hybrid SD Based HRRN	23.5	15.33

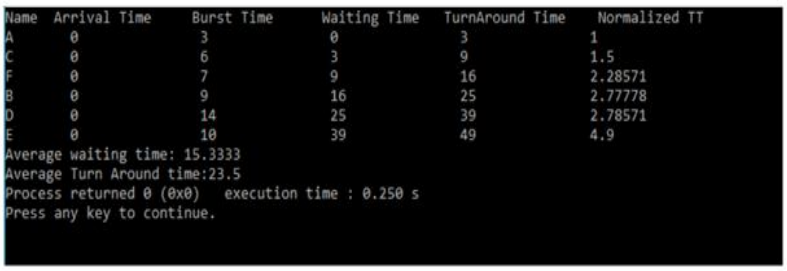


Figure 3. The input and output of new Hybrid Algorithm (SD based HRRN)

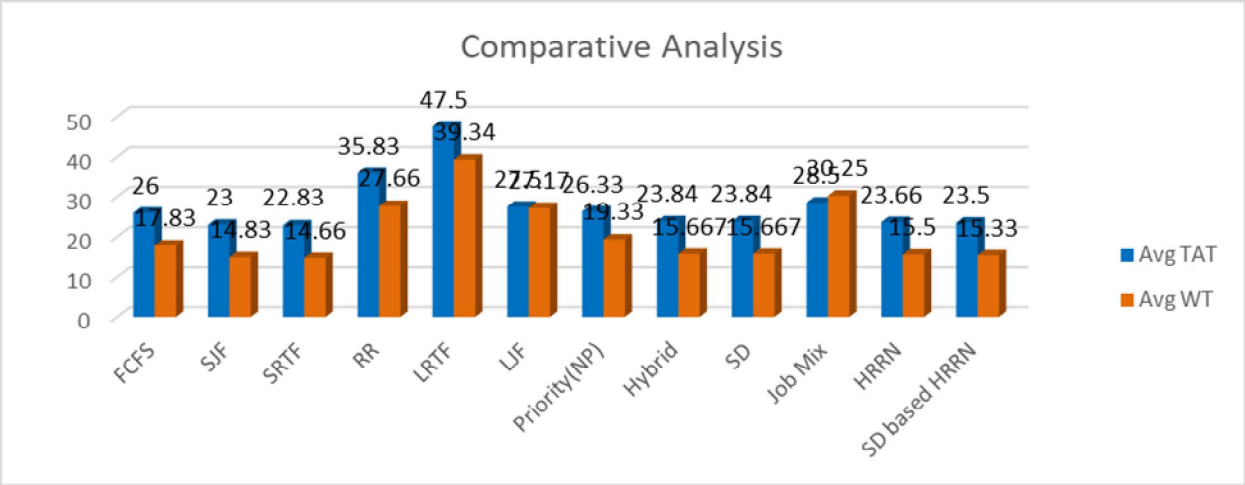


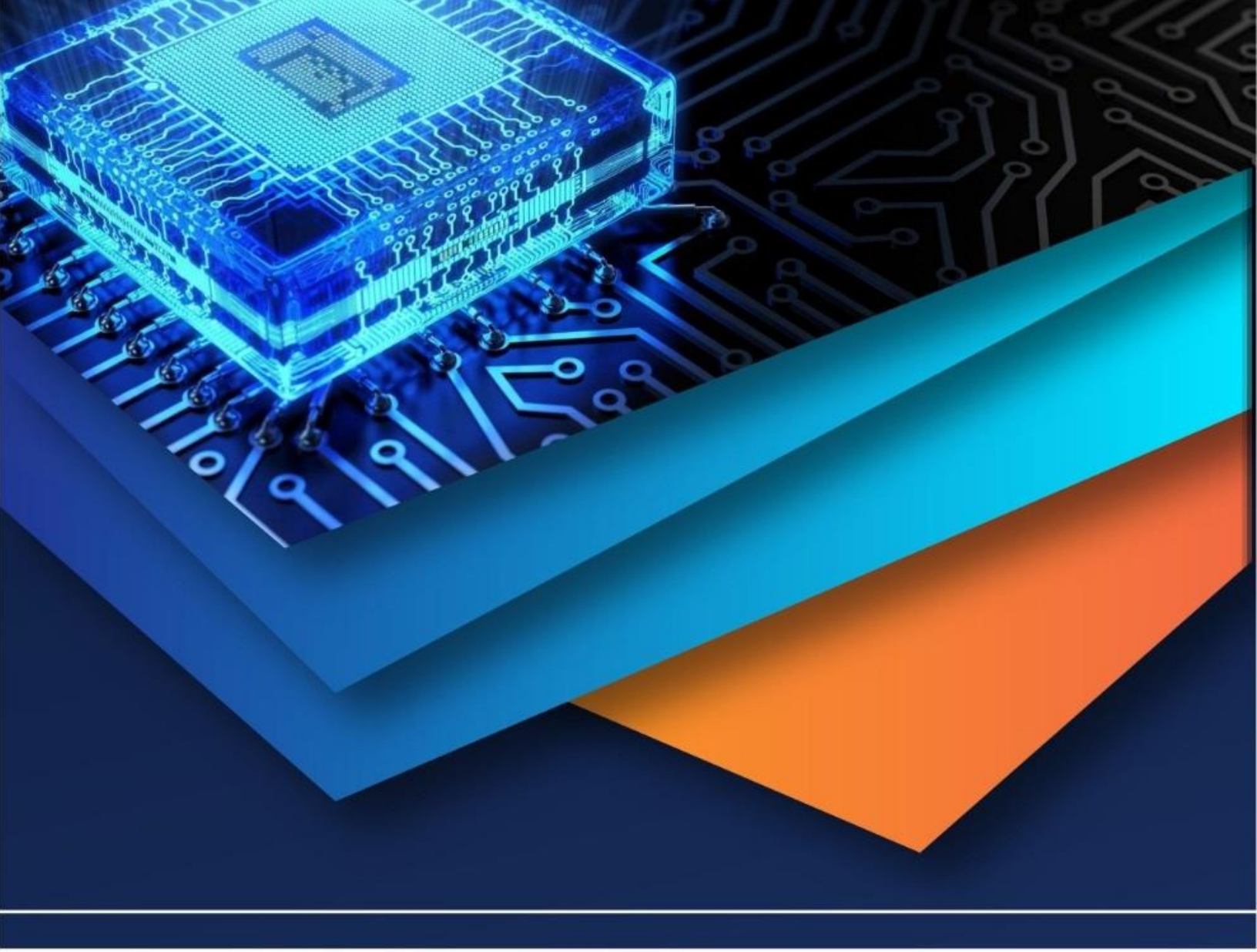
Figure 4. The comparative analysis of new Hybrid Algorithm (SD based HRRN)

IX. CONCLUSION

From the above results, our proposed new hybrid SD based HRRN algorithm is highly efficient over other many other algorithms. The hybrid algorithm has given better results in terms of average waiting time and average turnaround time therefore decreasing the overhead, saving of memory and also the issue of starvation is resolved, which is caused in Priority scheduling. Standard Deviation (SD) has more starvation when compared to our new proposed algorithm whereas hybrid scheduling is complexity in terms of implementation. Thus by considering all the cases, the proposed new hybrid scheduling algorithm will fulfil the present demands of a real-time software systems.

REFERENCES

[1] Gupta, A. K., Yadav, N. S., and Goyal, D. 2019. Design and Performance Evaluation of Smart Job First Dynamic Round Robin (SJFDRR) Scheduling Algorithm with Smart Time Quantum. American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS).
[2] Manuel, Jezreel Ian & Baquirin, Rey & Guevara, Kier & Tandingan, Dionisio. (2019). Fittest Job First Dynamic Round Robin (FJFDRR) scheduling algorithm using dual queue and arrival time factor: a comparison. IOP Conference Series: Materials Science and Engineering. 482. 012046.
[3] B. Fataniya and M. Patel, Dynamic Time Quantum Approach to Improve Round Robin Scheduling Algorithm in Cloud Environment, vol. 4, no. 4, pp. 963-969, 2018.
[4] Hina Gull, Sardar Zafar Iqbal, Saqib Saeed, Mohammad Abdulrahman Alqatani, Yasser Bamarouf "Design and Evaluation of CPU Scheduling Algorithms Based on Relative Time Quantum: Variations of Round Robin Algorithm." Journal of Computational and Theoretical Nanoscience, 15 (2018) Volume 6, Issue 8, August 2017.
[5] Chowdhury, S. 2018. Survey on various Scheduling Algorithms. "Imperial Journal of Interdisciplinary Research" (IJIR). ISSN: 2454-1362
[6] Ul Sabha, S. (2018). A Novel And Efficient Round Robin Algorithm With Intelligent Time Slice And Shortest Remaining Time First. Materials Today: Proceedings, 5(5), 12009–12015.
[7] Arpaci-Dusseau, Remzi H., and Andrea C. Arpaci-Dusseau. Operating systems: Three easy pieces. Arpaci-Dusseau Books LLC, 2018.
[8] Berhanu, Y., Alemu, A., Kumar, M.: Dynamic time quantum based round robin CPU scheduling algorithm. Int. J. Comput. Appl. 167(13), 48–55 (2017)
[9] Yosef H. Jbara, "A new visual tool to improve the effectiveness of teaching and learning CPU scheduling algorithms", 2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), 2017.
[10] Siva G., Rao Nageswara, Srinivasu S.V.N., Srinivasu N. and Ramakoteswara Rao G. 2015 Enhanced Precedence Scheduling Algorithm with Dynamic Time Quantum (EPSADTQ) Research Journal of Applied Sciences, Engineering and Technology



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY