The International Journal of Engineering and Information Technology

journal homepage:www.ijeit.misuratau.edu.ly

An Improved Round Robin Scheduling Algorithm For Enhancing CPU Utilization

Asma. Md. Alghwail
Al-asmarya University
Asmamohammed473@gmail.com

Mohamed Sullabi
Misurata University
m.sullabi@it.misurata.edu.ly

*Abstract*_ **CPU scheduling is the first goal and most important section of any operating system. CPU Scheduling algorithms are the methods or strategies which help scheduler to choose process in the ready queue. One of those strategies is Round Robin scheduling algorithm, a big challenge in this algorithm is how to adapt dynamic time quantum to be suitable for all of processes allocated in the ready queue, thus achieve making the best use of CPU. In this paper, a model was built in which a dynamic time quantum calculated by using average and standard deviation formula executed first, then use slice time to perform the first cycle and next cycle until the ready queue becomes empty. By applying this idea, the average turnaround time, waiting time could be minimized and the throughput of the CPU could be maximized and compare the proposal algorithm with three modifications of Round Robin algorithm which modified in recent years. a pre-modified round robin algorithm (DevRR) was modified, a model standard deviation round robin algorithm (SDRR) was developed, where the slice time using statistical measures was calculated and a simulation with Java Eclipse Version 2021-12 (4.22.0) and JDK operating environment (16.0.1) was built, to see whether the experimental results meet my requirements or not? And our modified algorithm using three scheduling criteria was implemented and evaluated, where were the average waiting time, the average turnaround time, and the throughput rate. The experimental result of all data sets indicates which the SDRR algorithm performs best. Moreover, 1000-process data set can give the best result that reduces 19.50% of Average Waiting Time (AWT), 19.51% of Average Turnaround Time (ATT) with 100ms of TQ, and 10-process data set can give the best result that increases 36.80% of THRPT with 29ms of TQ.**

Keywords_ **modified round robin algorithm (SDRR), average waiting time (AWT), an average turnaround time (ATT), throughput (THRPT), high priority queue (HighPQ), medium priority queue (MedPQ), low priority queue (LowPQ).**

I. INTRODUCTION

In the round robin scheduling algorithm, the value of the quantum time has a significant impact on the performance of the system. However, the amount of the quantum time is not easy to adjust.

Received 02 May , 2024; revised 11 May, 2024; accepted 15 Mar 2024. Available online 08 Aug, 2024.

If the quantum time is set too large, the algorithm will deteriorate to behave like the First Come First Served (FCFS) algorithm; That's lead to starvation to the processes stored in the end of queue (convoy effect or indefinite blocking). If the quantum time is set too small, this will cause the number of context switch times to increase, and will extend the waiting time for short tasks, thus increase the system overhead.

In the recent years; a number of improvements on the traditional RR algorithm have been developed; some of them are similar to each other, most of the modifications focus on finding the smart and suitable time quantum for all processes in the ready queue, including those modifications; Arya *et al.* (2018) proposed an algorithm that classifies available processes into two categories are high-priority processes and low-priority processes, one major issue in RR scheduling is selecting the value of time quantum. If the Time Quantum is very long, RR scheduling behaves as FCFS. On the other hand, if the time quantum is short, it imposes significant increase in the number of context switches. their primary goal is to overcome this limitation of traditional RR scheduling algorithm and maximize CPU utilization, further, will lead to a more effective and faster system. Zhang *et al.* (2019) produced a new algorithm called the Median-based Dynamic Round Robin (MDRR) algorithm, which calculating the time quantum for each round, it has the advantages of better performance, less overhead, and less complexity, compared to the traditional RR and other improved round robin scheduling algorithms. In the same year Sohrawordi *et al.* (2019) proposed a new algorithm called a Round Robin with Dynamic Time Quantum (RRDTQ), time quantum is calculated from the burst times for the set of processes which waiting in the ready queue, it gave the lower average waiting time, lower average turnaround time and a fewer number of context switches than of the traditional RR. Paul *et al.* (2019) proposed a new type of round robin where the time quantum is periodically adjusted according to the remaining burst time of the currently executing processes, the proposed algorithm performed much better than some of the algorithms

mentioned in sections of average waiting time, average turnaround time and context switch. Zouaoui *et al.* (2019) in June presented a round robin scheduling algorithm based on Neural Network Models to predict the ideal time quantum length which led to lowest average turnaround time. Ali *et al.* (2020) in February proposed a new approach to the round-robin scheduling algorithm (RR algorithm), that enhanced time quantum-based algorithm, by using dynamic time quantum leads to minimize AWT, ATT, ART and NCS. This approach inherits the properties of Round robin, shortest job first (SJF) algorithm and first come first serve algorithm (FCFS). Mostafa *et al.* (2020) in June introduced a new version of the RR algorithm called Standard Round Robin (SRR) using dynamic time slice. The modified algorithm benefits from clustering technique in clustering processes that similar each other in their features (i.e., burst times, weights, etc). After that Fiad *et al.* (2020) in July produced a modified Round Robin scheduling algorithm using a dynamic time quantum based on an analytical model. The proposed multiparameter analytical model computes the cost of each task. The ready processes list is sorted according to this cost. The new approach uses an optimal time quantum during the execution process based on the burst time of the processes allocating in the ready queue. The results explain that the new approach gives better performance than traditional RR, IRRVQ, and AET-RR algorithms by decreasing the waiting time and TT which help in improving the system performance. Next that, Iqbal *et al.* (2021) they have run out an experimental study where they have developed four versions of RR, each algorithm considers three-time quantum, and the performance of these variations was compared with the traditional RR algorithm. Proposed algorithms are based on time quantum technique relative to the burst time of the processes already in the ready queue. Abdelhaviz. (2021) in December proposed a new version named, VORR (Variant on Round Robin), which is improved and enhanced version of Round Robin scheduling algorithm. which effectively enhance the CPU performance by setting up a time quantum based on the median of burst times. Also, Simon *et al.* (2022) presented a dynamic QT to improve the traditional RR which has a static TQ. where in dynamic RR, more than one TQ used for time slot to occupation processes to the CPU. This work is a proposal that modified HLVQTRR to be 'An Improved Half Life Variable Quantum Time with Mean Time Slice Round Robin CPU Scheduling (ImHLVQTRR)'. In this technique, two quantum time (QT1 and QT2) is calculated. QT1 is the mean of all the processes in the ready queue and it is constant while QT2 is the half of each process burst and it changes depending on which process is in execution. Abdelkader *et al.* (2022) in September produced an enhance modification for the performance of the Round Robin algorithm called a Modified Median Mean Round Robin (MMMRR) algorithm. The new algorithm finds an optimal dynamic time quantum (median+ mean)/2) and generated for each round depending on the remaining burst time of the processes in the ready queue. The system performance was enhanced in terms of waiting time, turnaround time and context switching. The experimental results explain that the proposed algorithm outperforms ADRR, HYRR, EDRR, MARR and MMRR algorithms. Sakshi *et al.* (2022) in April proposed a new Median-Average Round Robin (MARR) scheduling algorithm. In this algorithm, they have modified the time quantum dynamically, the new algorithm with all the metrics gives the best results for Average Turnaround Time (ATT), Average Waiting Time (AWT) in a nutshell, the authors conclude context switches minimizations. Ullah and Shah. (2022) in November produced Novel Resilient Round Robin (NRRR) algorithm uses a dynamic TQ its value is computed using one of the statistical measures proposed in the paper. The new algorithm reduces the average wait time (AWT), the average turnaround time (ATT), the number of context switches (NCS) and enhances CPU utilization criterions. The modified algorithm uses the CPU efficiently and improves system performance. Putra and Purnomo. (2022) in July have three case studies, and the analysis of each case study are given. Comparisons are given about the metrics; average turnaround time and average waiting time, also number of contexts switching between the three case studies. proposed round robin algorithm is a modification from the generic round robin algorithm. In improved round robin algorithm if the remaining burst time is less than the time slice that is allocated, then the currently running process is continuing to be executed. Then finish the currently running process from ready queue and execute the next ready queue.

This paper proposes round robin scheduling algorithm, named SDRR, presents a new idea about calculating value of a time quantum (TQ). SDRR defines a new dynamic time quantum from the average burst time (ABT) and the standard deviation (SD) of all burst time values in the ready queue We emphasize the SD value for a data set of all burst times in each execution round, If the SD is the greater than a half of ABT, it shows that all burst time values are distributed far from the mean (ABT). If the SD is the smaller than a half of ABT, it shows that all burst time values are approaching the mean (ABT). Therefore, SDRR will calculate an appropriate dynamic time quantum for this dataset of the burst times. Below are the objectives that should be achieved by SDRR algorithm:

a. The dynamic time quantum based on average and standard deviation formula.
b. The calculated optimum of the time quantum will be sought, which will be suitable for all the processes in the ready queue.
c. Calculating the best value of dynamic time quantum by choosing between three equations according to the data which is either distributed far or approaching the mean.

Acquisition of the minimum Average Waiting Time (AWT), Average Turnaround Time (ATT) and the maximum of throughput rate (THRPT) through the proposed algorithm.

The rest of the paper contains section (IV) represents related works. In section (V) efficiency of the existing approaches are illustrated. Section (VI) presents the illustration of the proposed algorithm (SDRR). Section (VII) contains the conclusion.

II. RELATED WORKS

Dash et al. (2018) developed a new approach for round robin CPU scheduling algorithm named PBRR, which improves the performance of CPU using the priority factor in real time operating system. The proposed Priority based Round-Robin CPU Scheduling algorithm is based on the integration of round-robin and priority scheduling algorithm. It retains the advantage of round robin in reducing starvation and integrates the advantage of priority scheduling. The proposed algorithm also implements the concept of dynamic time quantum to the processes. The proposed algorithm improves all the drawbacks of round robin CP U scheduling algorithm.

Phorncharoen and Sa-Ngiamvibool. (2018) proposed round robin scheduling algorithm named DevRR that defines new dynamic time quantum calculated by the standard deviation (SD) and the average burst time (ABT) in each execution round. Distribution of a data set of the burst time for all processes in the ready queue is discussed to define the optimized time quantum in each round. DevRR can strongly apply to support an operating system software development on mobiles, other devices, and network operating system (NOS) to enhance performance of CPU utilization at the present and the future.

Shafi et al. (2020) proposed a new CPU scheduling algorithm named as Amended Dynamic Round Robin (ADRR) based on CPU burst time. ADRR gives average waiting time equal to (17 ms), average turnaround time equal to (29.8 ms), and the number of context switches is 4. they take burst time of each process randomly and assume quantum time as 5ms. The processes arrive in order.

In this paper, the proposal of round robin scheduling algorithm (SDRR), presents a new idea about calculating value of a time quantum (TQ). SDRR defines a new dynamic time quantum from the average burst time (ABT) and the standard deviation (SD) of all burst time values in the ready queue We emphasize the SD value for a data set of all burst times in each execution round, If the SD is the greater than a half of ABT, it shows that all burst time values are distributed far from the mean (ABT). If the SD is the smaller than a half of ABT, it shows that all burst time values are approaching the mean (ABT). Therefore, SDRR will calculate an appropriate dynamic time quantum for this dataset of the burst times.

PROPOSED ALGORITHM (SDRR)

SDRR Approach

SDRR defines a new dynamic time quantum from the average burst time and the standard deviation (SD) of all burst time values in the ready queue. It emphasizes the SD value for all burst times in each execution round. If the SD is greater than a half of Average Burst Time (ABT), it shows that all burst time values are distributed far from the mean (heterogeneous). If the SD is smaller than a half of ABT, it shows that all burst time values are approaching the mean (homogeneous). Therefore, SDRR will calculate an appropriate dynamic time quantum for this dataset of the burst times. The relationship between the mean, the SD value, and the burst time of all processes in the ready queue is shown in equations (1), (2), (3):

(1) $TQ = \mu + \sqrt{\sigma}$

(2) $TQ = \left[\frac{\sum_{i=0}^{n} BT_i}{n} + \sqrt{\frac{\sum_{i=0}^{n} (BT_i - \mu)^2}{n-1}} \right] * \mathbf{3}$

(3) $TQ = \left[\frac{\sum_{i=0}^{n} BT_i}{n} + \sqrt{\frac{\sum_{i=0}^{n} (BT_i - \mu)^2}{n-1}} \right] * \mathbf{2.4}$

where: TQ is a time quantum, mean (μ) is Average Burst Time (ABT), (σ) is Variance, √σ is Standard deviation (SD), BT is the burst time, min (BT) is the minimum burst time for each round, n is number of processes in the ready queue (size of queue), and (i) is a process ID.

In the SDRR algorithm concept, the first step is to verify the size of the ready queue. If the ready queue is empty, it finishes the execution. The second step searches for the minimum burst time for the processes in the ready queue. In the third step, the algorithm calculates ABT. In the fourth step, it calculates the variance for a dataset of the burst times in this execution round. The fifth step is calculating the SD value of this dataset. In the last step, it calculates the optimized time quantum under one of two conditions.

First, if the SD is greater than half of the ABT, TQ is set to be the mean added to the SD divided by two to reduce the balance of TQ and avoid the convey effect. In this case, the dataset of the burst times is distributed from the mean. Hence, the TQ must be the larger value as shown in equation (2). Second, if the SD is smaller than half of the ABT, TQ is set to be the mean added to the SD. In this case, it means that this dataset of the burst times is approaching the mean. Therefore, the TQ must be the smaller value as shown in equation (3).

Then the algorithm divides the ready queue into three queues. The first queue is called High Priority Queue (HIGHPQ) which includes all processes with BT < QT and gives It priority = 1. The second queue is called Low Priority Queue (LOWPQ) which includes all processes with BT > QT and gives it priority = 0. The algorithm repeats all steps until the ready queue is empty. After that, it

computes the Average Waiting time (AWT), Average Turnaround Time (ATT), and the throughput (THRPT) to analyze CPU utilization performance.

Pseudo Code of SDRR

This section illustrates the modified algorithm (standard deviation round robin SDRR) and the workflow of the algorithm. A mathematical proof will use to demonstrate the proposed algorithm, apply to the dataset of processes, and compare to the traditional RR. The pseudo code of the proposed SDRR algorithm is as follows:

1. TQ = 0, AWT = 0, ATT = 0, THRPT = 0 //The initial values
2. SizeOfQueue = Queue.getSize()
3. Pr [i]=0
4. Var=0
5. if (SizeOfQueue = 0) go to 45.
6. Sum = 0
7. AverageBurstTime (ABT) = 0 //The mean for each round
8. for (i = 0; i < SizeOfQueue; i++) {
9. BT[i] = Queue.getBurstTime[i]
10. if (BT >=1 and BT < 33) then
11. Pr[i] = 3 elseif (BT >=33 and BT<66) then
12. Pr[i] = 2 elseif (BT >=66) then
13. Pr[i] = 1
14. Sum += BT[i]
15. AverageBurstTime (ABT) = Sum/SizeOfQueue
16. Total = 0
17. // The variance for a data set of the burst time in queue
18. for (i = 0; i < SizeOfQueue; i++) {
19. BT[i] = Queue.getBurstTime[i]
20. BT[i] = BT[i] – AverageBurstTime
21. Total += (BT * BT)
22. }
23. Var =Total/(SizeOfQueue-1)
24. // To calculate the standard deviation (SD)
25. SD = Math.sqrt(Var)
26. // To calculate the new optimized time quantum
27. if (SD > (AverageBurstTime/2))
28. TQ = (AverageBurstTime + SD) *3
29. else
30. TQ = (AverageBurstTime + SD) *2.4
31. HighPQ [SizeOfQueue]=0.
32. MedPQ [SizeOfQueue]=0.
33. LowPQ [SizeOfQueue]=0.
34. For (i = 0; i < SizeOfQueue; i++) {
35. If (Pr[i]=1) then
36. Add BT[i] to HighPQ[i]elseif Pr[i]=2) then
37. Add BT[i] to MedPQ[i]elseif Pr[i]=3) then
38. Add BT[i] to LowPQ[i]
39. Execute processes placed in HighPQ with value of TQ-25.
40. Execute processes placed in MedPQ with value of TQ-10.
41. Execute processes placed in LowPQ with value of TQ+1.
42. Swap execution between the three queues at a rate four processes for HighPQ, two processes for MedPQ, one process for LowPQ respectively.
43. Do steps 1 to step 41 until remaining CPU burst time of processes of queues becomes zero.
44. Go to 1.
45. Store the IN-TIME and OUT-TIME of the process into a table GANTTCHART.
46. Compute AWT, ATT, THRPT
47. Stop.

General Block Diagram of The System

The general block diagram explains overall modelling flowchart; beginning with an arrival and burst time and priority generation for each process then, definition classes, objects and lunch threads for selected round robin algorithm, after that, run and compare the results of four modifications of RR scheduling algorithm as shown in figure (1).

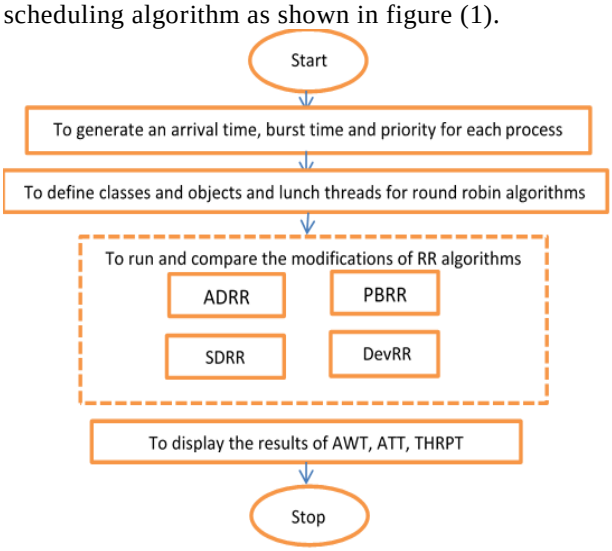

Fig. 1. explains the general block diagram of our modified algorithm (SDRR)

Flowchart of SDRR

The next figure (2) explains a flowchart of the modified algorithm (SDRR); which combining features of three CPU scheduling algorithms are round robin, priority, and multi-level queues scheduling are merged together for better performance.

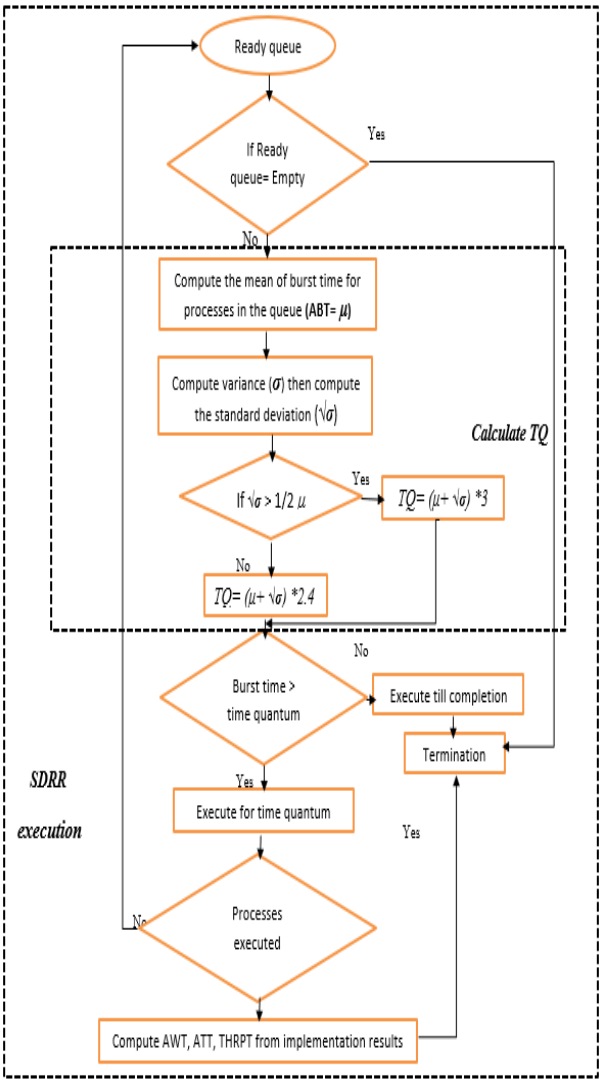

Fig.2. explains the flow chart of SDRR

III. RESULTS AND DISCUSSION

Hardware scifications

the experiments were carried out using a computer with the following specifications (Intel(R) Core (TM) i7-5600U CPU @ 2.60GHz processor and Java Eclipse Version 2021-12 (4.22.0) and JDK operating environment (16.0.1)).

Text Files (Datasets)

Datasets used in our paper token from Google Cloud Jobs Dataset (GoCJ) website. The GoCJ dataset is comprised of different files. Each file contains the sizes of a specified number of jobs or processes. The name of the file determines the number of jobs within the file i.e.; GoCJ_Dataset_100 has the sizes of 100 jobs. Moreover, the GoCJ dataset also (i. e named as Google-like realistic dataset, Hussain et al. (2018).

The data used is dataset are divided into seven classes, according to the size of data increasing gradually to monitor performance development progressively; where it is divided as following Hussain et al.(2018): (10,50,100,500,1000,5000,10000 process), selected in implementation for testing from the dataset manually; where classification the data in dataset into seven classes makes the evaluation of the system efficient.

Used Tools for Performance Evaluation

There are many scheduling criteria that can be used for round robin algorithm performance evaluation as performance evaluation measures, three of them were used to compare the results of SDRR, PBRR, ADRR and DevRR; these measures are:

1. Average Turnaround Time: is the total amount of time spent by the process from arriving in the ready state to its completion. Which calculated by equation:
   Average Turnaround Time (ATT)= [Completion Time (CT) – Arrival Time (AT)].
2. Average Waiting time: is the total time spent by the process in the ready state waiting for CPU. Which calculated by equation:
   Average Waiting Time (AWT) = Average Turnaround Time (ATT) – Burst Time (BT).
3. Throughput: is defined as number of processes completed per unit time. Which calculated by equation:
   Throughput (THRPT) = [Total of Processed Burst Times (BT) / Number of processes].

In general, those are some scheduling criteria used to evaluate any modification or enhancement of round robin CPU scheduling algorithm.

IV. THE MODIFIED ALGORITHM (SDRR) EVALUATION METHOD

A modified algorithm will be tested and show the results. Also, analysis and discuss about them.

To check retrieval effectiveness of our modified algorithm; we will be testing by selecting the classes dataset in ascending order to monitor performance development gradually, then, I have to evaluate the performance by comparison our modified algorithm with three of latest versions of improved round robin, those are (DevRR produced in 2018, PBRR produced in 2018, ADRR produced in 2020).

DevRR, PBRR and ADRR have been programmed belongs to the modified algorithm (SDRR); to implement them with same of dataset and same time quantum that generated by data analysis in our modified algorithm (SDRR) to show more accurate results, then to compare the performance and note the change in results. The following experiments illustrate the obtained results:

Table (1) shows discussion the results of all datasets of processes

Dataset	Algorithm	TQ (ms)	AWT			ATT			THRPT		
			(ms)	%	Rank	(ms)	%	Rank	(ppu)	%	Rank
10	DevRR	13.20	34.3	%14.36	2nd	46.5	%16.16	2nd	8.714	%26.28	3rd
	ADRR	29.28	72.2	%30.22	3rd	84.4	%29.34	3rd	11.091	%33.45	2nd
	PBRR	29.28	101.5	%42.49	4th	113.7	%39.52	4th	1.151	%3.47	4th
	SDRR	29.28	30.9	%12.93	1st	43.1	%14.98	1st	12.2	%36.80	1st
50	DevRR	16.78	391.58	%20.98	3rd	407.36	%21.11	3rd	10.662	%25.02	3rd
	ADRR	37.872	390.16	%20.90	2nd	405.94	%21.04	2nd	15.471	%36.30	2nd
	PBRR	37.872	728.70	%39.04	4th	744.48	%38.58	4th	1.013	%2.38	4th
	SDRR	37.872	355.92	%19.07	1st	371.70	%19.26	1st	15.473	%36.31	1st
100	DevRR	26.54	1001.59	%16.98	2nd	1027.13	%17.11	2nd	16.803	%25.41	3rd
	ADRR	61.2960	1465.51	%24.84	3rd	1491.05	%24.84	3rd	25.287	%38.24	1st
	PBRR	61.2960	2443.35	%41.41	4th	2468.89	%41.13	4th	1.027	%1.55	4th
	SDRR	61.2960	989.52	%16.77	1st	1015.06	%16.91	1st	23.009	%34.80	2nd
500	DevRR	42.61676	10884.599	%20.82	2nd	10976.487	%20.91	2nd	27.184	%24.82	3rd
	ADRR	99.880	10934.87	%20.92	3rd	10976.49	%20.91	2nd	41.534	%37.93	1st
	PBRR	99.880	20502.46	%39.22	4th	20544.08	%39.14	3rd	1.0012	%0.91	4th
	SDRR	99.880	9951.82	%19.04	1st	9993.44	%18.41	1st	39.790	%36.33	2nd
1000	DevRR	42.61676	22330.011	%21.21	3rd	22371.627	%21.22	3rd	27.166	%24.81	3rd
	ADRR	99.880	21333.16	%20.27	2nd	21374.78	%20.27	2nd	41.575	%37.97	1st
	PBRR	99.880	41075.25	%39.02	4th	41116.87	%39.00	4th	1.0	%0.91	4th
	SDRR	99.880	20527.45	%19.50	1st	20569.06	%19.51	1st	39.752	%36.31	2nd
5000	DevRR	42.61676	113962.80	%21.52	3rd	114004.41	%21.52	3rd	27.152	%24.80	3rd
	ADRR	100	104589.68	%19.75	1st	104631.29	%19.76	1st	41.608	%38.00	1st
	PBRR	100	205727.76	%38.85	4th	205769.38	%38.85	4th	1.0	%0.91	4th
	SDRR	100	105195.912	%19.87	2nd	105237	%19.87	2nd	39.722	%36.28	2nd
10000	DevRR	42.61676	228597.59	%21.56	3rd	228639.21	%21.56	3rd	27.150	%24.80	3rd
	ADRR	100	208754.21	%19.69	1st	208795.83	%19.69	1st	41.613	%38.01	1st
	PBRR	100	411637.30	%38.82	4th	411678.92	%38.83	4th	1.0	%0.91	4th
	SDRR	100	211124.555	%19.92	2nd	211166	%19.92	2nd	39.718	%36.28	2nd

The table above explains results for all datasets using three optimized of round robin CPU scheduling algorithm (ADRR, PBRR, DevRR) in addition to the modified algorithm (SDRR) and three criteria were used are average waiting, turnaround time and throughput (AWT, ATT, THRPT) for evaluation with three attributes; the first shows (ms or ppu) the average and throughput which obtained of the scheduling process, the second shows (%) the average in percentage and the third (rank) shows the order from best to worst according to results were given for each algorithm. The table shows that SDRR is considered the best in terms of results and is compatible with datasets of all sizes except THRPT, where it decreases by a small percentage with larger volume of processes or tasks entering the processor, this is due to the limitation of the value of the time quantum, and to the low throughput value of the Round Robin algorithm itself. Where the percentage of SDRR when dataset1 were (12.93%, 14.98%, 36.80%) of three criteria respectively (the lower of waiting and turnaround time the better of results), vice versa for THRPT. The algorithm so continued with best results for the datasets (2, 3, 4, 5) until it reached to dataset6 that contains 5000 where the ADRR was the best results (19.75%, 19.76%, 38.00%) with little variance, followed by SDRR, and so was the case for the dataset7 (10000 processes). It should be clarified that the time quantum is generally between to these values (10-100 ms).

Through the previous table, shows that the results of the algorithm SDRR is superior, where is has the lowest waiting time as shown as below, compared to other algorithms (ADRR, PBRR, DevRR), (3) shows flowchart results of Average waiting time (AWT) for all datasets.

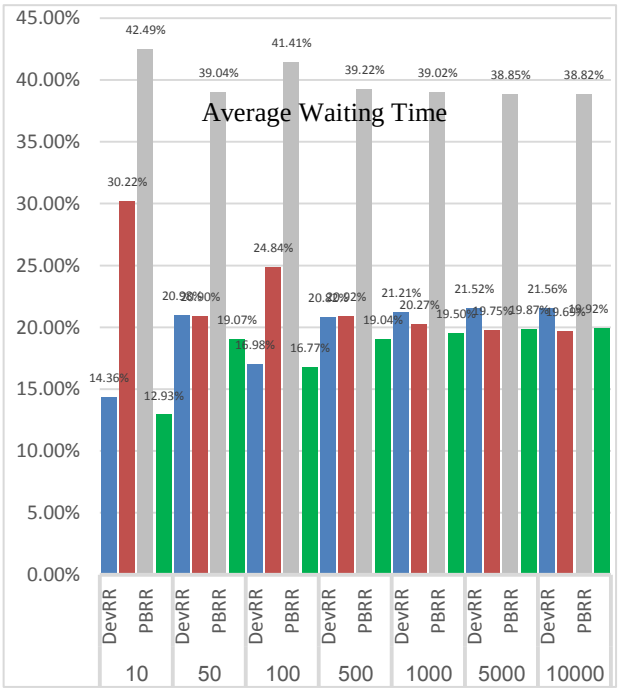

Fig .3. shows results Average waiting time for all data sets

Figure (3) shows a comparison of AWT values for ADRR, PBRR and DevRR algorithm compared to the SDRR. The best performance algorithm is SDRR that can compute AWT as 20527.45ms (19.50% reduction) for 1000-process data set, 355.92ms (19.07% reduction) for 50-process data set, and 9951.82% (19.04% reduction) for 500-process data set respectively. While the reason for the slight decrease in performance of AWT with the dataset (5000,10000-process) is due to the

restriction of the time quantum value in the range (10-100ms).

Figure (4) shows flowchart results of average Turnaround Time (ATT) for all data sets, where SDRR has the best results, from the chart below, we note SDRR has the least turnaround time.

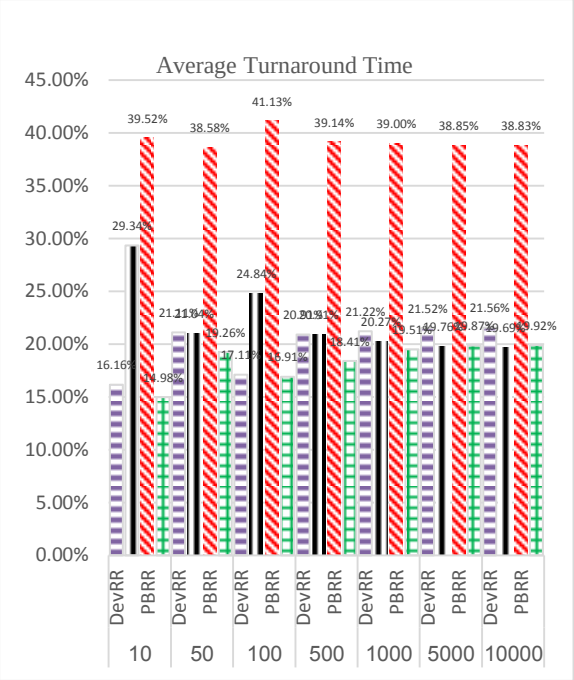

Fig .4. shows results Average Turnaround Time for all datasets

Figure (4) shows a comparison of ATT values for ADRR, PBRR, and DevRR algorithm compared to the SDRR. The best performance algorithm is SDRR that can compute ATT as 20569.06ms (19.51% reduction) for 1000-process data set, 371.70ms (19.26% reduction) for 50-process dataset, and 9993.44ms (18.41% reduction) for 500-process dataset respectively. While the reason for the slight decrease in performance of ATT with the dataset (5000,10000-process) is due to the restriction of the time quantum value in the range (10-100ms).

Figure (5) shows flowchart results of the Throughput of the CPU for all data sets, where SDRR has results closed to ADRR, and PBRR has the lowest of throughput.

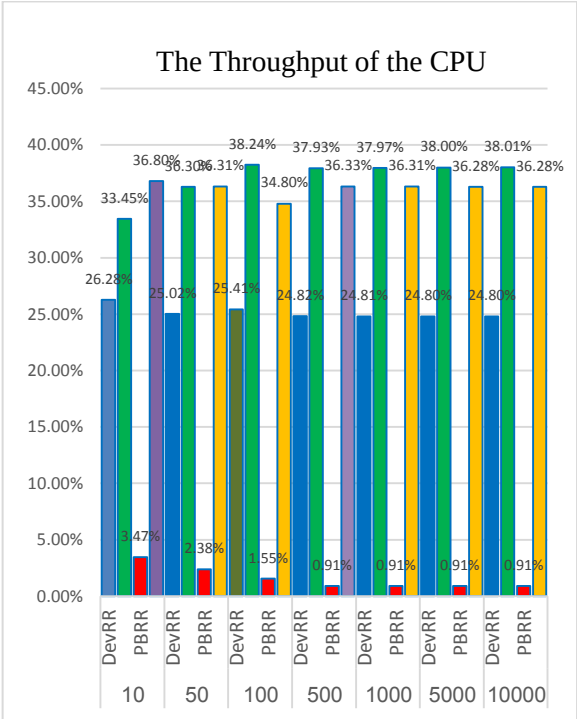

Fig .5. shows results throughput for all datasets

Figure (5) shows a comparison of THRPT values for ADRR, PBRR, and DevRR algorithm compared to the SDRR. The best performance algorithm is SDRR that can compute THRPT as 12.2p/ms (36.80% reduction) for 10-process data set, 39.790p/ms (36.33% reduction) for 500-process data set, and 39.752p/ms (36.31% reduction) for 1000-process data set respectively. While the reason for the slight decrease in performance of THRPT with the other datasets is due to the restriction of the time quantum value in the range (10-100ms).

V. CONCLUSION

In this paper. The problem of finding the most appropriate time quantum in round robin CPU scheduling algorithm has been studies, we modified a pre-modified round robin (DevRR), And we named it (SDRR), where we calculated the slice time using statistical measures and built a simulation program with Java Eclipse Version 2021-12 (4.22.0) and JDK operating environment (16.0.1), to see whether the experimental results meet our requirements or not? And we implemented and evaluated our modified algorithm using three scheduling criteria, which are the average waiting time, the average turnaround time, and the throughput rate. The experimental result of all data sets indicates that the best performance algorithm is SDRR. Moreover, 1000-process data set can gain the best result that can reduce 19.50% of Average Waiting Time (AWT), 19.51% of Average Turnaround Time (ATT) with 100ms of TQ, and 10-process data set can gain the best result that can reduce 36.80% of THRPT with 29ms of TQ.

VI. FUTURE WORKS

for developing our algorithm (SDRR) in future we suggest:

SDRR algorithm where can combine it with SJF algorithm and produce suitable algorithm can resolve starvation algorithm efficiently.

Simulation can be performed with C language.

Put the processes in ascending after finding the TQ by standard deviation.

REFERENCES

[1] Abdelhafiz, Afaf A. (2021). " VORR: A NEW ROUND ROBIN SCHEDULING ALGORITHM". Available at Egyptian Knowledge Bank (EKP) Journal Homepage: https://absb.journals.ekb. Vol. 32, No. 2, (December) 2021, pp.45-54.

[2] Abdelkader, Afaf; Ghazy, Nermeen; Zaki, Mervat S.; ElDahshan, Kamal A. (2022). " MMMRR: A Modified Median Mean Round Robin Algorithm for Task Scheduling ". International Journal of Intelligent Engineering and Systems, Vol.15, No.6, 2022 DOI: 10.22266/ijies2022.1231.53.

[3] Abu-Dalbouh, Hussain Mohammad. (2022). " A New Combination Approach to CPU Scheduling based on Priority and Round-Robin Algorithms for Assigning a Priority to a Process and Eliminating Starvation ". (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 13, No. 4, 2022.

[4] Ali, Khaji Faizan; Marika, Abhijeet; Kumar, PhD.Kakelli Anil. (2020). " A Hybrid Round Robin Scheduling Mechanism for Process Management". International Journal of Computer Applications (0975 – 8887) Volume 177 – No. 36:14-19.

[5] Arya, Govind Prasad; Nilay, Kumar; Prasad, Devendra. (2018). "An Improved Round Robin CPU Scheduling Algorithm based on Priority of Process". International Journal of Engineering & Technology, 7 (4.5): 238-241.

[6] Dash, S. S.; Bakhara, N.; Agrawalla, P.; Behera, P. P. (2018). " A New Proposed Dynamic Quantum for Priority Based Round Robin Scheduling Algorithm". International Research Journal of Engineering and Technology (IRJET), Volume: 05 Issue: 02: 1333-1337.

[7] Fiad, Alaa; Maaza, Zoulikha Mekkakia; Bendoukha, Hayat. (2020)." Improved Version of Round Robin Scheduling Algorithm Based on Analytic Model ". International Journal of Networked and Distributed Computing Vol. 8(4); December (2020), pp. 195–202.

[8] Hussain, Altaf and Aleem, Muhammad (2018, Sep26). GOCJ: Google Cloud Jobs Dataset. [online] Available at: https://data.mendeley.com/datasets/b7bp6xhrcd/1.[Accessed 27 Jan, 2021].

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR), Int. J. Adv. Res. 7(2):422-429.

[9] Iqbal, Sardar Zafar; Gull, Hina; Saeed, Saqib; Saqib, Madeeha; Alqahtani, Mohammed; Bamarouf, Yasser A.; Krishna, Gomathi; Aldossary, May Issa. (2021). " Relative Time Quantum-based Enhancements in Round Robin Scheduling ". Computer Systems Science & Engineering DOI:10.32604/csse.2022.017003.

[10] Joshi, Ankush; Goyal, S. B.; Sharma, Kalpana. (2020). "A Modified Version of Round Robin Algorithm "Modulo Based Round Robin Algorithm ". international Journal of Advanced Science and Technology Vol. 29, No. 11s:576-578.

[11] Mostafa, Samih M. and Amano, Hirofumi. (2020). "Dynamic Round Robin CPU Scheduling Algorithm Based on K-Means Clustering Technique". applied science. MDPI.

[12] Paul, Tithi; Faisal, Rahat Hossain; Samsuddoha, Md. (2019). "Improved Round Robin Scheduling Algorithm with Progressive Time Quantum". International Journal of Computer Applications (0975 – 8887) Volume 178 – No. 49:30-36.

[13] Phorncharoen, Sarayut and Sa-Ngiamvibool, Worawat. (2018). "A proposed round robin scheduling algorithm for enhancing performance of CPU utilization". PRZEGLĄD ELEKTROTECHNICZNY, ISSN 0033-2097, R. 94 NR 4/2018:26-29.

[14] Putra, Tri Dharma; Purnomo, Rakhmat. (2022). " Case Study: Improved Round Robin Algorithm". Sinkron: Jurnal dan Penelitian Teknik Informatika Volume 7, Number 3, July 2022.

[15] Sakshi; Sharma, Chetan; Sharma, Shamneesh; Kautish, Sandeep; Alsallami, Shami A. M.; Khalil, E.M.; Mohamed; Ali Wagdy. (2022). " A new median-average round Robin scheduling algorithm: An optimal approach for reducing turnaround and waiting time ". 1110-0168 4, 2022 THE AUTHORS. Published by Elsevier BV on behalf of Faculty of Engineering, Alexandria University.

[16] Shafi, Uferah; Shah, Munam; Wahid, Abdul; Abbasi, Kamran; Javaid, Qaisar; Asghar, Muhammad; Haider, Muhammad. (2020). " A Novel Amended Dynamic Round Robin Scheduling Algorithm for Timeshared Systems". The International Arab Journal of Information Technology, Vol. 17, No. 1:90-98.

[17] Simon, Ashiru; Dams, Gabriel Lazarus; Danjuma, Salome. (2022). " AN IMPROVED HALF LIFE VARIABLE QUANTUM TIME WITH MEAN TIME SLICE ROUND ROBIN CPU SCHEDULING (IMHLVQTRR) ". Science World Journal Vol. 17 (No2) 2022 www.scienceworldjournal.org ISSN: 1597-6343 (Online), ISSN: 2756-391X (Print).

[18] Sohrawordi, Md.; U. A. Md.; Ehasn Ali, Md.; Palash Uddin Md.; Hossain, Mahabub. (2019). "A MODIFIED ROUND ROBIN CPU SCHEDULING ALGORITHM WITH DYNAMIC TIME QUANTUM".

[19] Sonia Zouaoui, Lotfi Boussaid, Abdellatif Mtibaa. (2019)." Priority based round robin (PBRR) CPU scheduling algorithm", International Journal of Electrical and Computer Engineering (IJECE) Vol. 9, No. 1, February 2019.

[20] Ullah, Wali; Shah, Munam Ali. (2022). " A NOVEL RESILENT ROUND ROBIN ALGORITHM BASED CPU SCHEDULING FOR EFFICIENT CPU UTLILIZATION ". Authorized licensed use limited to: COMSATS INSTITUTE OF INFORMATION TECHNOLOGY. Downloaded on November 23,2022 at 10:28:28 UTC from IEEE Xplore. Restrictions apply.

[21] Zhang, Chunhong; Luo, Ping; Zhao, Yuye; Ren, Jianqiang. (2019). "An Efficient Round Robin Task Scheduling Algorithm Based on a Dynamic Quantum Time". INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING Volume 13:197-204.

[22] Zouaoui, Sonia; Boussaid, Lotfi; Mtibaa, Abdellatif. (2019). " Improved time quantum length estimation for round robin scheduling algorithm using neural network". Indonesian Journal of Electrical Engineering and Informatics (IJEEI) Vol. 7, No. 2:190-202.