IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Volume-6, Issue-6 E-ISSN: 2347-2693

Mode Based Round Robin Scheduling Algorithm

S.Jain[1*], H. Rohil[2]

[1*] Dept. of Computer Science and Application, CDLU, Sirsa, Haryana, INDIA.
[2] Dept. of Computer Science and Application, CDLU, Sirsa, Haryana, INDIA.

**Corresponding Author:* engishilpa19@gmail.com

Available online at: www.ijcseonline.org

Accepted: 13/Jun/2018, Published: 30/Jun/2018

Abstract— In a multiprogramming environment, the Scheduling technique decides which process will be selected and assigned to the CPU next so that the efficiency of the CPU can be increased. One of the well-known techniques of scheduling is round-robin technique. A number of modifications have been made in the basic round robin scheduling algorithm but still work is going on to make it the ideal one. The performance of this technique mainly depends upon the selected value of time quantum i.e. a fair share of time for which a process can get the CPU and if the process still not completed, it will join the ready queue for completion of the remaining task. For achieving this aim, this paper proposed a new mode based round-robin scheduling algorithm that offers the reduction of average turnaround time as compared to average turnaround time calculated by existing modulus based technique, a best reported similar technique available in the literature. Experimental evaluation is done using C language.

Keywords— CPU Scheduling, Scheduling Algorithm, Round-Robin Scheduling, Turnaround-Time, Time-Quantum.

I. INTRODUCTION

In the process scheduling, the process manager removes the running process from the CPU and selects the next suitable process on the basis of some strategy. Process scheduling is an important part of Multiprogramming operating systems**.** To schedule processes, a technique known as round-robin was invented. Round-robin is a pre-emptive type of scheduling algorithm. At the initial stage, a static time quantum is assigned to each process as a time quantum. A process could take the CPU for its allotted time quantum and has to be pre-empted. As soon as the time quantum completed, it has to leave the CPU sand next process in the ready queue is assigned that CPU. If still, some work is pending for the previous process, it will join the ready queue for completing its pending work. Later, no. of modifications was done to make a better approach than this. To achieve that purpose static time quantum gets changed to dynamic time quantum. Our purpose is to develop a new CPU scheduling technique with dynamic time quantum. Before this, a number of techniques were available with dynamic time quantum e.g. modulus based technique, in which the time quantum is calculated with the help of average, median and some other mathematical functions [1]. The main objective of the proposed scheduling algorithm is to reduce average turnaround time, where Turnaround time is defined as the total time required by a process from its submission till its completion.

The paper is organized in sections. Section 1 discusses the introduction of round robin scheduling. Section 2 covers the research work already done over the round robin scheduling. Section 3 discusses the proposed procedure and the existing procedure which is considered as a base for the current research work. Section 4 covers the numerical analysis of proposed and existing algorithm. Section 5 discusses the comparison among proposed and existing method. Section 6 discusses the final conclusion of the research work.

II. RELATED WORK

To implement round-robin scheduling various static time quantum and dynamic time quantum methods has been developed to date. Due to poor throughput and a large number of context switches, the static time quantum technique was modified to the dynamic one. In other work, modulus based techniques [1], the fuzzy concept [2] etc. are used to predict the length of time quantum. The detailed study is given below:

This is the main research paper with which our proposed work has been compared. The existing algorithm under comparison is the modulus based approach. In this approach concept of the dynamic time quantum is used and the time quantum is calculated by following formula
$\sqrt{((\text{average})^2 + (\text{median})^2)}$, where the average is calculated as the sum of terms/total number of terms. And median is calculated as:

Case 1: If a number of terms is odd, a median element is an element present at the position calculated by ((no. of elements + 1) /2).

Case 2: If number of terms is even, median element is element present at position calculated by sum of (((no. of elements /2) + (no. of elements/2) + 1))/2 [1].

In this research, CPU scheduling done by analyzing the process so that the burst time of the process can be selected, the analysis carried out when the process executed for the first time. The analysis will determine the burst time of the process and according to that burst time, the operating system can adapt the value of the time-slice or time quantum Q. When operating system starts for the first time, it begins with a default time quantum value, which is subject to change after a period of time for that the operating system can identify the burst time for a subset of the programs used by the user. The system will take the short period of time to learn user behavior through the analysis of the burst time of the new processes. The determined time quantum represents real and optimal value because it based on real burst time. When a new process loaded to be executed the operating system tests the status of the program which can be either 1 or 0. When the status equals to 0 this means that the process is either being executed for the first time or it has been modified or updated since the last analysis. The operating system assigns a counter to find the burst time of the process and continues executing on the remaining processes including the new arrival process using the current time quantum Q, otherwise and when status is equal to 1, then the operating system recalculates the time quantum Q depending on the remaining burst time of ready processes including the new arrival process. It was also found that the optimal time quantum can be presented by the median =25 for the set of processes in the ready queue [3].

In this round robin algorithm will be executed in three phase this will help to minimize average turnaround time. The algorithm allocates every process to CPU, a single time by applying Round Robin schedule, an initial time quantum of k units. After the first run, the initial time quantum gets doubled (2k units) and later select the shortest process from the waiting queue and assign it to the CPU. After that, select the next shortest process for execution by excluding the already executed. Above defined steps will be repeated for remaining processes [4].

Here an algorithm is designed where; the jobs are sorted in ascending order of their burst time. Performance of RR algorithm mainly depends upon the size of time quantum. If it is very small, it causes too many context switches. If it is very large, the algorithm degenerates to FCFS. This algorithm takes a dynamic time quantum where the time quantum is repeatedly adjusted according to the remaining burst time of currently running processes. To get the optimal time quantum, the median of the burst time is taken as the time quantum [5].

In this algorithm, the researcher decides the time quantum equals the burst time of the first process, which will change after the end of the first time quantum. The determined time quantum represents real value. Repeatedly, when a new process is loaded into the ready queue, the operating system calculates the average of the sum of the burst times of processes found in the ready queue including the new arrival process and set that value as a time quantum [6].

In this research work, The Fuzzy Inference System is used for finding the time quantum. It got 2 inputs and one output. The first input is the number of user/ processes in the system and the second input is the average burst time of the processes in the ready queue. Time quantum is the output of the Fuzzy inference system. Membership Function gets used here for calculation of inputs and time quantum. In this paper, a method using the fuzzy logic value of time quantum has been decided in such a way that it is neither too large nor too small such that every process has the best throughput without unnecessary context switches [7].

This research work focus on making task set and then applying the adaptive round robin algorithm. According to the arrival time, task sets are being formed. For this one, time quantum value is valid only for a task set. Every time while preparing task set, the algorithm will select the best suitable time quantum value with the help of greedy approach [8].

III. METHODOLOGY

The proposed algorithm is named as the mode based method. In this method, dynamic time quantum (TQ) is calculated by finding the mode of the elements under study and the value of mode and hence the value of TQ will vary with the number of runs. The detailed algorithm is given as:

A) The algorithm for proposed Mode based Round Robin:

i. Enter the number of processes under study.

ii. Enter the maximum and minimum range of the numbers, in which you want to generate the burst time of the processes. The processes are generated randomly.

iii. Sort all the processes in ascending order of their burst time.

iv. For **RUN 1**, TQ is decided by calculating Mode of the available processes under study.

The Mode can be calculated by:

Case 1: An element which is repeated the maximum number of times will be considered as a mode.

Case 2: In case, all elements are repeated the same number of times then the element having the maximum value of time quantum will be the targeted time quantum for the problem under study.

v. For **RUN 2**, TQ is decided by calculating Mode of remaining processes under ready queue.

vi. Repeat steps 4-5 until all processes execution gets completed.

vii. Gantt chart for final execution.

viii. Calculate Average turnaround time for the proposed algorithm by dividing total turnaround time by total no. of processes under study.

B) EXISTING SIMILAR ALGORITHM SELECTED FOR COMPARISON [1]:

The existing algorithm is named as modulus base method [1]. In this method, dynamic time quantum (TQ) concept gets used. This algorithm is selected for comparison with the proposed method because modulus based research work [1] proved that it gives less number of context switches and hence throughput as compared to Shortest remaining burst round robin (SRBRR) technique [5] and AN algorithm [6]. So if proposed method proved itself good as compared to modulus based method [1] then it will be better than both SRBRR and AN algorithm as well.

The algorithm for existing Modulus based Round robin:

1. **i.** Enter the number of processes under study.
ii. Enter the maximum and minimum range of the numbers, in which you want to generate the burst time of the processes. The processes are generated randomly.
iii. Sort all the processes in ascending order of their burst time.
iv. For **RUN 1**,

a. Median can be calculated by:
Case 1: If the number of processes are even in number say n, then median will be a process available at position number calculated by finding an average of $(n/2)^{th}$ and $((n+1)/2)^{th}$ term.
Case 2: If Number of processes n is odd, then the median will be the process available at position number given by ceiling (n/2) in the sorted array.

b. Average can be calculated by:
Average can be calculated by adding burst time of all the processes / Total number of processes and consider the integer part of the calculated term.

c. TQ can be calculated by:
$TQ = \sqrt{[\{(median)^2 + (avg)^2\}/2]}$

d. Draw Gantt chart for RUN 1.

v. For RUN 2:

a. Draw table for pending processes after RUN 1.

b. Median can be calculated by:
Case 1: If the number of remaining processes are even in number say n, then median will be a process available at position number calculated by finding an average of $(n/2)^{th}$ and $((n+1)/2)^{th}$ term.
Case 2: If Number of remaining processes n is odd, then the median will be the process available at position number given by ceiling (n/2) in the sorted array.

c. Average can be calculated by:
Average can be calculated by adding burst time of all the remaining processes / Total number of remaining processes and considers the integer part of the calculated term.

d. TQ can be calculated by:
$TQ = \sqrt{[\{(median)^2 + (avg)^2\}/2]}$

e. Draw Gantt chart for RUN 2.

vi. Repeat step5 until all processes execution gets completed.
vii. Draw Gantt chart for the final execution.
viii. Calculate Average turnaround time for modulus method by dividing total turnaround time by the total number of processes under study.

IV. EXPERIMENTAL EVALUATION

A) Average Turnaround time for proposed Mode based method:

i. The total number of processes entered 5.
ii. Minimum Range of the CPU burst time entered and maximum range of the burst time 18. Processes are generated randomly within the range using random function.
iii. Generated burst time of processes in sorted order:

Table 4.1

Processes	Burst Time in seconds	Sorted Order processes	Sorted order Burst Time
P1	13	**P3**	4
P2	16	**P5**	9
P3	4	**P1**	13
P4	14	**P4**	14
P5	9	**P2**	16

RUN 1:
The value of Mode for Run 1: 16
Therefore, for Run 1, the value of TQ: 16
Gantt chart after RUN 1:

P3	P5	P1	P4	P2

0……..4………13………26……..40………56

RUN 2:
The value of Mode for Run 2: 0
Therefore, for Run 2, value of TQ =0.

Gantt chart for RUN 2:

P3	P5	P1	P4	P2

0…….4………13………26………40………..56

vi. No process remaining

vii. Final Gantt chart :

P3	P5	P1	P4	P2

0…….4………13……..26………40………..56

viii. Average Turnaround time for mode method in above scenario =27.8

B) Average Turnaround time for existing Modulus method:

1. **i.** The total number of processes entered 5.

ii. Minimum Range of the burst time entered 1 and maximum range of the burst time 18. Processes are generated randomly within the range using random function.

2. **iii.** Generate burst time of processes in sorted order:

Table 4.2

Processes	Burst Time in seconds	Sorted Order processes	Sorted order Burst Time
P1	13	**P3**	4
P2	16	**P5**	9
P3	4	**P1**	13
P4	14	**P4**	14
P5	9	**P2**	16

RUN 1:

a. Median for Run 1:
Total number of elements in sorted list = 5
Median (Element at position) = Ceiling (5/2) =3
Therefore, median= 13.

b. Average for Run1:
Floor {(4+9+13+14+16)/5} = 56/5 = floor (11.2)
Average: 11.

c. Time Quantum for Run 1:
TQ = int [[√{ $(13)^2$ + $(11)^2$ } / 2] = 12

d. Gantt Chart for Run 1:

P3	P5	P1	P4	P2

0……..4……..13……….25……..37……….49

RUN 2:

a. Table for pending processes after Run 1:

Table 4.3

P2	4	**P3**	1
P3	1	**P4**	2
P4	2	**P2**	4

b. Median for Run 2:
Total number of elements in sorted list=3
Median (Element at position) = Ceiling (3/2) =2
Therefore, median= 2.

c. Average for Run2:
Floor (4+1+2)/3 = 7/3 =floor (2.3)
Average: 2.

d. Time Quantum for Run 2:
TQ = int [√ {$(2)^2$ + $(2)^2$ }] =2

e. Gantt Chart for Run 2:

P3	P5	P1	P4	P2	P1	P4	P2

0…..4…..13….25…..37…..49……50….52…54

vi. **RUN 3:**

a. Table for pending processes after Run 2:

Table 4.4

P2	2	**P2**	2

b. Median for Run 3:
Total number of elements in sorted list = 1
Median (Element at position) = ceiling (1/2) =1
Therefore, median= 2.

c. Average for Run 3:
Floor (2/10) = floor (2)
Average: 2.

d. Time Quantum for Run 3:
TQ = int [[√{$(2)^2$ + $(2)^2$} / 2]] = 2

e. Gantt Chart for Run 3:

P3	P5	P1	P4	P2	P1	P4	P2	P2

0…..4…13….25…..37…49…50…..52…..54…56

vii. No process is left.

viii. Final Gantt Chart:

P3	P5	P1	P4	P2	P1	P4	P2	P2

0…..4…..13….25….37…..49…..50…52…54…56

ix. Average Turnaround time for modulus method for above scenario =35.

V. RESULT AND DISCUSSION

Table 5.1 shows the average turnaround time calculated using the existing algorithm (Modulus based Round Robin scheduling) vs. proposed algorithm (Mode based Round Robin scheduling) for various ranges of CPU burst time. Details are as under:

TABLE 5.1
AVERAGE TURNAROUND TIME

Sr. No.	**The Range of CPU Burst Time**	**Average Turn Around Time**	
		Proposed Mode based Round Robin Method	**Existing Modulus based Round Robin Method**
1.	1-10	12.00	13.62
2.	10-20	63.86	73.29
3.	20-30	115.43	130.91
4	30-40	104.279	113.279
5	40-50	134.8999	146.789

The experiment shown above is conducted for different ranges of CPU burst time. Here, 100 experiments were conducted in each range of burst time. For each experiment under one range, 5 processes were taken. CPU burst time for these processes was generated using RAND () function in C language. Then the average of turnaround times of these 100 experiments was calculated using proposed and existing technique. Similarly next 100 experiments were conducted for CPU burst time ranges from 10-20 and the process continues for remaining ranges. Hence, a total of 500 experiments in 5 ranges were conducted. Each row of the table 5.1, gives an average of turnaround time of 100 experiments for a given range of CPU burst time for proposed method and for the existing method. The result produced by the proposed method is far better than existing approach under comparison

VI. CONCLUSION

Round robin scheduling is a pre-emptive fair share scheduling method. It is free from starvation problem. The success of this scheduling algorithm depends upon the selection of the right value of time quantum. The proposed technique named as mode based round robin scheduling algorithm also tried to find the suitable value of time quantum so that average turnaround time of the system could be minimized. It was compared with the existing modulus based technique, the best reported similar approach available in literature. The existing and the proposed algorithms were implemented in C language. In each experiment under range 1-10 of CPU burst time, 5 processes were generated randomly using rand () function of C language and turnaround time was calculated for the proposed approach and the existing approach. This step was repeated 100 times for one range. The average of these 100 turnaround times was calculated for both approaches under range 1-10. The same experiment was repeated for remaining ranges and the results were entered in the table 5.1. From above experiment, it was concluded that the proposed approach gives better average turnaround time as compared to existing modulus technique.

REFERENCES

[1] S. Arif, S. Rehman, F. Riaz, "*Design of A Modulus Based Round Robin Scheduling Algorithm*", 9th Malaysian Software Engineering Conference, pp. 230-235, Dec. 2015.
[2] M. Ghazizadeh and M. Naghibzadeh et al., "*Fuzzy Round Robin CPU Scheduling (FRRCS) algorithm*", International Conference on Systems, Computing Sciences and Software Engineering (SCSS), Part of the International Joint Conferences on Computer, Information and Systems Sciences and Engineering, pp: 348-353, 2007.
[3] R. Matarneh, "*Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Processes*"**,** American Journal of Applied Sciences, pp. 1831-1837, 2009.
[4] A. Singh et. al., "*An Optimized Round Robin Scheduling Algorithm for CPU Scheduling"*, International Journal on Computer Science and Engineering, Vol. 02, No. 07, pp. 2383-2385, 2010.
[5] R. Mohanty, H. Behera, K. Patwari et al. "*Design and Performance Evaluation of a New Proposed Shortest Remaining Burst Round Robin (SRBRR) Scheduling Algori thm*"
[6] A. Noon, A. Kalakech, S. Kadry, "*A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average* ", IJCSl lnternational Journal of Computer Science, Issues, Vol. 8, Issue 3, No. I,2011.
[7] B. Alam, M. Doja, R. Biswas,"*Finding Time Quantum of Round Robin CPU Scheduling Algorithm Using Fuzzy Logic",* International Conference on Computer and Electrical Engineering, pp. 795-798, 2008.
[8] N. Kumar, A. Kumar, "*A Task set Based Adaptive Round Robin (TARR) scheduling algorithm for improving performance*", 1st International conference on futuristic trend in computational analysis and knowledge management, pp. 347-352, 2015.

Authors Profile

S. Jain- Shilpa Jain is M.Tech. in Computer Science and engineering (2011) from Ch. Devilal University, Sirsa, Haryana. I have completed my B.Tech. in Computer Science and Engineering (2008) from Kurukshetra University, Kurukshetra. I have qualified Gate (2010), GATE (2017), UGC NET (DEC 2014), UGC NET (DEC 2015), UGC NET (JAN 2017). I have been working as an assistant professor, Department of Computer Science and Applications, Ch. Devilal University, Sirsa, Haryana, India since 2015. I am having a teaching experience of approximately 7 years. Published a research paper named as "Text-based emotion detection" authored by Poonam Arya and Shilpa Jain in International Journal of Computer Engineering & Technology (IJCET) (ISSN Print: 0976-6367) recently. A total of 10 Research papers have been published in various national, international Journals.

H. Rohil Harish Rohil started his career as Asst. Professor at Dept. of CSA, Ch. Devi Lal University, Sirsa (Haryana) in 2004. He worked as Associate Professor and has been founder officiating Registrar in addition to other charges of Dean, Faculty of Physical Sciences and Chairperson, Dept. of CSA at Ch. Ranbir Singh University, Jind (Haryana). He obtained his Ph. D. from Department of Computer Sc. & Applications, Kurukshetra University, Kurukshetra in 2012 and his M.Tech(CSE) from Guru Jambheshwar University of Sc. & Technology, Hisar in 2004. He has published 70 research papers. His research interests include software reuse, data mining, data structure and, operations research.