Performance Evaluation of Dynamic Round Robin Algorithms for CPU Scheduling

Abdulaziz A. Alsulami[1] , Qasem Abu Al-Haija[2], Mohammed I. Thanoon[3] , Qian Mao[4]

Department of Computer Information and Systems Engineering, Tennessee State University,
3500 John A. Merritt Boulevard, Nashville, TN 37209, USA

[1]*aalsula3@my.tnstate.edu,* [2]*qabualha@my.tnstate.edu.sa,* [3] *mthanoon@my.tnstate.edu,* [4]*qmao@tnstate.edu*

***Abstract*—The performance of an operating system (OS) is affected by the algorithm policy that is used by a CPU to schedule the running processes. Thus, a better CPU scheduling algorithm results in faster OS performance using minimal resources over small amounts of time [11]. For that reason, many algorithms were proposed and implemented to enhance the performance of CPU scheduling. Round Robin is considered an efficient and fair algorithm because all processes are given the same amount of time quantum. However, its efficiency depends on the selected time quantum. In this paper, we present a comparative study of four different Round Robin algorithms namely: Adaptive Round Robin Algorithm, Best Time Quantum Round Robin CPU Scheduling, Optimal Round Robin Scheduling Using Manhattan Distance Algorithm, and Improved Round Robin Scheduling Algorithm. We compare these algorithms in terms of four performance factors including: Average Waiting Time (AWT), Average Turnaround Time (ATT), Average Response Time (ART) and Number of Contexts Switching (NCS). The simulation results show that both Adaptive Round Robin and Optimal Round Robin Scheduling Using Manhattan Distance algorithms are more efficient to be adopted as they recorded the minimum values of performance factors.**

***Index Terms*—CPU Scheduling, Round Robin (RR), Adaptive Scheduling, Dynamic Time Quantum, Manhattan Distance.**

I. INTRODUCTION

Due to the significant role of operating systems (OS) in managing computing resources [1] (i.e. hardware and software resources) and providing common services for computer programs, performance efficiency and ability to manage and schedule several concurrent processes and tasks is of the utmost importance for a wide range of applications. Nowadays, computing platforms are designed with multiprocessing capabilities and thus equipped with multitasking operating systems in which a large number of processes can be scheduled to be executed without users noticing. Indeed, increasing the efficiency of CPU scheduling is a challenging issue and thus several scheduling algorithms were proposed to improve CPU scheduling and maximize CPU scheduling performance by controlling and enhancing four different criteria including:

- Average Waiting Time (AWT) which is the average time that all processes wait in the queue.
- Average Turnaround Time (ATT) which refers to the total time between the arrival of a process and its completion time.
- Average Response Time (ART) which refers to the time between the arrival of a process and the time of first response to it.
- Number of Context Switching (NCS) which refers to the number of times that the CPU switches between processes.

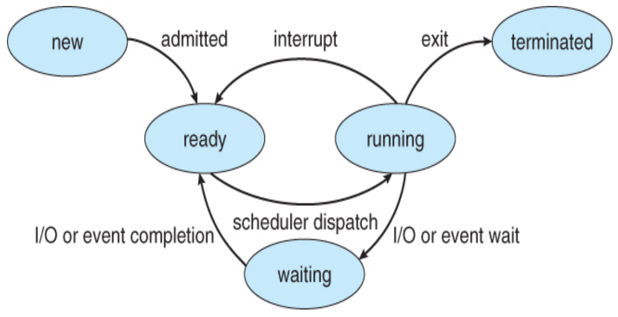

Fig. 1. Waiting to Ready CPU Scheduling Scheme [2]

Generally, in order to improve the efficiency of the CPU, such criteria must be kept to a minimum. The basic CPU scheduling scheme is called ”waiting to ready” and illustrated in Figure 1. In this basic algorithm, if the process is preemptive, it will not wait any longer to get scheduled. It will snatch the chance from other lower-priority process. If the process is having lower priority/non-preemptive, then it will keep waiting for the resources to release , complete the event and then get dispatched through the scheduler. However, a large number of CPU scheduling algorithms that utilize the aforementioned criteria in their execution were proposed and analyzed in the literature, for instance:

978-1-7281-0137-8/19/$31.00 ©2019 IEEE

- First Come First Serve (FCFS): this algorithm executes the jobs based on their arrival times, so any process that arrives first will be executed. This algorithm is the simplest CPU scheduling algorithm however it produces high AWT and ATT.
- Shortest Job First (SJF): this algorithm executes the jobs according to their burst time, so a process with the shortest burst time will be executed first. Hence, this algorithm causes starvation of processes that have longer burst times.
- Round Robin (RR): this algorithm executes jobs with specific amounts of time called time quantum. Therefore, all arrival processes will be executed with equal amount of time quantum slots. If a process burst time is greater than time quantum it will be blocked and moved to the end of the queue and another arrival process will be executed. This algorithm eliminates starvation of processes; however, its efficiency depends on selecting the optimal value of time quantum. For example, if the time quantum is small, NCS will increase; however, if the time quantum is large, then it will behave as FCFS algorithm [3].

There are many researchers developed different solutions to enhance the performance of RR algorithm, therefore there is a need to make a comparative study to select the optimal RR algorithm. Thus, in this paper, we provide a comparative study between four different methods that are used to select optimal time quantum for an RR algorithm that minimizes the performance criterion factors (i.e., AWT, ATT, ART, NCS). The considered RR algorithms include: Adaptive Round Robin Algorithm [4], Best Time Quantum Round Robin CPU Scheduling [6], Optimal Round Robin Scheduling Using Manhattan Distance Algorithm [6], and Improved Round Robin Scheduling Algorithm [7]. We intentionally chose these four RR algorithms because their elapsed time are very close to each other. Also, we provide extensive simulation results to evaluate these algorithms according to the aforementioned factors.

The rest of this paper is organized as follows: Section II presents the related work, Section III describes the Round Robin Scheduling schemes and its various quantum-based approaches. Section IV presents the research methodology and the simulation results as well as the comparison of the four algorithms and factors. Finally, section V concludes the paper

II. Related Work

In the last decade, several approaches and solutions tried to address the efficient utilization of CPU scheduling algorithms for processes and jobs (tasks) of the OS. For instance, authors of [4] provided a survey study that compared between four different algorithms (AAIRR, ERR, AAIRR and ERR). Each algorithm is executed using static and dynamic time quantum. Also, there are three cases of generating burst time in case 1 the burst time was generated randomly, in case 2 burst time was generated randomly but in a decreasing order and in case 3 the burst time was generated randomly but in an ascending order. It is assumed that all processes arrive at the same time (arrival time = 0). The authors use two criteria to examine the performance of all four algorithms which is ART and ATT.

Also, in [9] authors compared between two different algorithms which are the Improved Round Robin (IRR) which has dynamic time quantum and the basic round robin which has static time quantum. Selecting the optimal algorithm was based on three factors which are AWT, ATT and NCS. The experiment has two ceases in case 1 the arrival time is equal to 0 and the burst time is generated in an acceding order, descending order and random. In case 2 the arrival time is different, and the burst time is generated in an acceding order, descending order and random.

There are some limitations in these two researches such as in [4] the authors only use two factors in the compression between the four algorithms which are AWT and ATT however in our research we add two more important factors ART and NCS. In [9] Authors compared between two different algorithms using three criteria AWT, ATT and NCS however the method of generating the burst time and the arrival time was not mentioned. In our research, we used exponential distribution and the reason of using such a model is explained in the next section, Section III, ROUND-ROBIN SCHEDULING ALGORITHMS.

III. Round-Robin Scheduling Algorithms

RR algorithm is a fair-share approach to execute processor's jobs with equal time quantum slots. According to the time quantum, RR algorithms [4] are classified into two categories as follows:

RR algorithm with static time quantum: Static time quantum algorithm means that time quantum will have a fix value that is not changing during the execution of the algorithm which will be preassigned value. This algorithm is simpler than dynamic time quantum algorithm in terms of implementation, but its performance depending on the time quantum that assigned. Usually with high time quantum value this algorithm will behave as FCFS algorithm. However, small time quantum will cause high value of NCS.

RR algorithm with dynamic time quantum: Dynamic time quantum algorithm means that time quantum will not have a fixed value instated it will have variant time quantum. Therefore, the time quantum will be changed during the execution of the algorithm. Calculating time quantum is usually based on the burst time of processes that are in ready queue.

The efficiency of dynamic time quantum is better than static time quantum algorithm because in dynamic time quantum algorithm time quantum will be assigned based on known burst time. However, the performance of dynamic time quantum algorithm depends on the method that is used to calculate the time quantum in each cycle. Indeed, the dynamic time quantum concept has been proved to develop optimized RR scheduling algorithms [9] where each the dynamic time quantum assigned to the processes waiting in the ready queue. In this paper, we consider dynamic time quantum based RR algorithms. Specifically, we consider the following four dynamic algorithms:

A. *Optimal Round Robin Using Manhattan Distance:*

This method selects the optimal time quantum based on Manhattan Distance Method. This is done by finding differences between the highest burst time with the lowest burst time. Finally, the optimal time-quantum will be selected using the following algorithmic procedure:

```
Result: Time Quantum for RR Scheme
Initialization;
Generate processes with their bursts time.;
Assign the process that arrive in the queue. ;
if Process arrives to queue based on its arrival time then
    while queue is not empty do
        Find maximum and minimum burst times for processes that are presented in the queue;
        Calculate the difference between the maximum burst time and minimum burst time;
        Set Time Quantum = max (bursts time) - min (bursts time);
    end
end
```

Algorithm 1: Optimal Round Robin Scheduling Using Manhattan Distance Algorithm

B. *Improved Round Robin Scheduling Algorithm:*

In this method, the optimal time-quantum will be selected using the following algorithmic procedure:

```
Result: Time Quantum For Round Robin Scheme
Initialization;
Generate processes with their bursts time.;
if Process arrives to queue based on its arrival time then
    while queue is not empty do
        Sort processes that are already in the queue based on their burst time increasingly;
        Find the maximum burst time of processes that in the queue;
        Find the median burst time of processes that in the queue;
        Set Time Quantum = Ceil (sqrt (median x highest Burst time) ;
    end
end
```

Algorithm 2: Improved Round Robin Scheduling Algorithm

C. *Adaptive Round Robin Algorithm:*

In this method, the optimal time-quantum will be selected using the following algorithmic procedure:

```
Result: Time Quantum For RR Scheme
Initialization;
Generate processes with their bursts time;
if Process arrives to queue based on its arrival time then
    while queue is not empty do
        Sort processes that are already in the queue based on their burst time increasingly;
        Calculate the median of the sorted burst times;
        Set Time Quantum = median (sort (bursts time));
    end
end
```

Algorithm 3: Adaptive Round Robin Algorithm

D. *Best Time Quantum RR Algorithm:*

In this method, the optimal time-quantum will be selected using the following algorithmic procedure:

```
Result: Time Quantum For RR Scheme
Initialization;
Generate processes with their bursts time.;
if Process arrives to queue based on its arrival time then
    while queue is not empty do
        Sort processes that are already in the queue based on their burst time increasingly;
        Calculate the mean of the sorted burst times;
        Calculate the median of the sorted burst times;
        Set Time Quantum = [mean + median]/2;
    end
end
```

Algorithm 4: Best Time Quantum Round Robin CPU Scheduling Algorithm

IV. Simulation Results

In this section, we present our simulation and experimental results of comparison between the aforementioned methods based on AWT, ATT, NCS and ART. Unlike other research works [6] that compare the performance of different RR algorithms by assuming that all processes arrive at the same time (arrival time =0). However, we assumed that every process has a random burst time and also has a random arrival time that are generated randomly using exponential distribution. The reason why exponential distribution is used in this paper because it gives accurate result as is shown in [10] for timing factors since the CPU processes usually arrive at different burst time. Also, we have simulated all the methods using MATLAB assuming that there are 10,000 processes generated along with their arrival time and burst time. The reason why we run 10,000 processes is that we need the simulation to run longer. Moreover, all methods used the same test data for fairness. Finally, following is a summary steps of executing the simulation:

1) *Generate arrival time and burst time*
2) *Execute Method 1: Adaptive Round Robin Algorithm.*
3) *Calculate AWT, ATT, NCS and ART of method 1*
4) *Execute Method 2: Best Time Quantum Round Robin.*
5) *Calculate AWT, ATT, NCS and ART of method 2*
6) *Execute Method 3: Optimal Round Robin using Manhattan Distance Algorithm.*
7) *Calculate AWT, ATT, NCS and ART of method 3*
8) *Execute Method 4: Improved Round Robin Scheduling Algorithm.*
9) *Calculate AWT, ATT, NCS and ART of method 4*

TABLE I
EVALUATION SUMMERY OF ALL METHODS

RR Method	AWT	ATT	NCS	ART
Method 1	799.19	819.92	19480.00	413.37
Method 2	806.51	827.24	17104.00	474.72
Method 3	812.21	832.95	11113.00	794.07
Method 4	807.45	828.18	12389.00	667.82

Table 1 along with Figures 2-5 provide the simulation results obtained for all studied methods and criteria while executing the simulation for 10 times. Specifically, Figure 2 shows the results for the average waiting times (AWT) recorded from the extensive simulation runs for the four RR based schemes, Figure 3 shows the results for the average turnaround time (ATT) recorded from the extensive simulation runs for the four RR based schemes, Figure 4 shows the results for the average response time (ART) recorded from the extensive simulation runs for the four RR based schemes, and Figure 5 shows the results for the average number of context switching (NCS) recorded from the extensive simulation runs for the four RR based schemes. Accordingly, the results obtained for the four methods are tightly close in terms of both AWT and ATT while every method recorded different figures in term both NSC and ART. The highest number of NSC which is equal to 19480 belongs to Adaptive Round Robin Algorithm while Optimal Round Robin based Manhattan Distance Algorithm recorded the smallest number of NSC among all methods with 11113 context switches. Moreover, the highest figure of ART time results has been listed for Optimal Round Robin based Manhattan Distance Algorithm with 794 time units on average whereas Adaptive Round Robin Algorithm registered the smallest ART number with 413 time units.

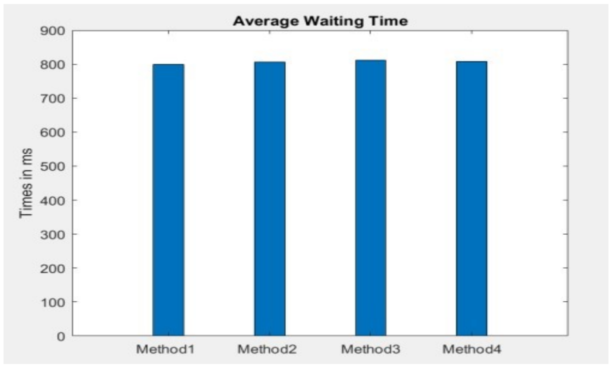

Fig. 2. Result of Average Waiting Time of All Methods

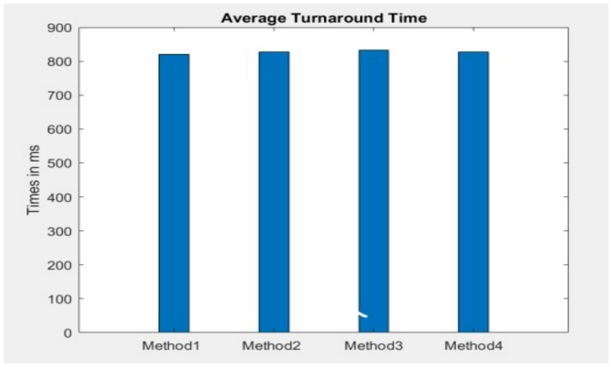

Fig. 3. Result of Average Turnaround Time of All Methods

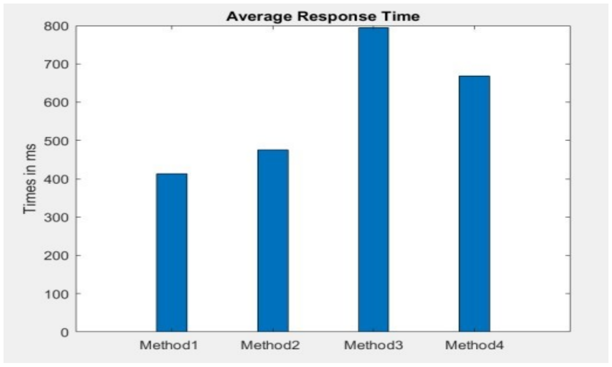

Fig. 4. Result of Average Response Time of All Methods

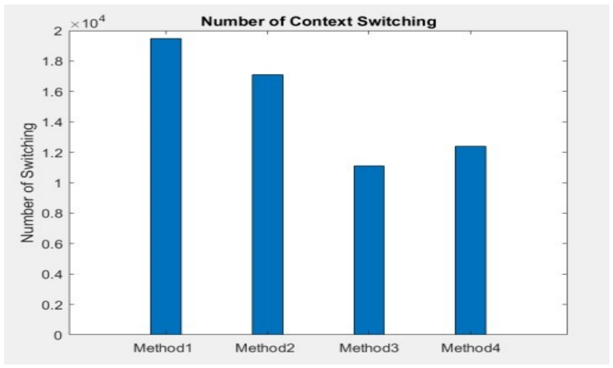

Fig. 5. Number of Context Switching

V. Conclusions and Remarks

In conclusion, many Round Robin algorithms were developed to maximize the efficiency of the CPU scheduling. Therefore, there is a necessity to compare between them by evaluating their performance using some criteria such as AWT, ATT, ART and NCS. For that reason, this paper compared between four different types of RR algorithm. The simulation results showed that both Adaptive Round Robin Algorithm and Optimal Round Robin based Manhattan Distance Algorithm are more efficient CPU scheduling techniques than Best Time Quantum Round Robin CPU Scheduling and Improved Round Robin Scheduling Algorithm since they recorded the best performance results.

References

[1] Stallings, W. (2012). Operating systems. 7^{th} edition, New Jersey: Prentice Hall, pp.108-124

[2] Silberschatz, A, Galvin, P.B, G. Gagne. (2013). Operating systems Concepts. 9^{th} edition, John Wiley & Sons Inc.

[3] Mora, H., Abdullahi, S. and Junaidu, S. (2017). Modified Median Round Robin Algorithm (MMRRA). Abuja: International Conference on Electronics, Computer and Computation, pp.1-7

[4] ElDahshan, K., Abd Elkader, A. and Ghazy, N. (2017). Round Robin based Scheduling Algorithms, A Comparative Study. Automatic Control and System Engineering, 17(2), pp.29-30

[5] Khokhar, D. and Kaushik, A. (2017). Best Time Quantum Round Robin Cpu Scheduling Algorithm. international Journal of Scientific Engineering and Applied Science (IJSEAS), 3(2395-3470), pp.213216.

[6] Sharma, A. and Kakhani, G. (2015). Analysis of Adaptive Round Robin Algorithm and Proposed Round Robin Remaining Time Algorithm. International Journal of Computer Science and Mobile Computing, 4(12, December 2015), pp.139-147

[7] Srilatha, N., Sravani, M. and Divya, Y. (2017). Optimal Round Robin CPU Scheduling Algorithm Using Manhattan Distance. International Journal of Electrical and Computer Engineering (IJECE), 7(6), p.3664

[8] Ahmed, W. and Muquit, S. (2018). Improved Round Robin Scheduling Algorithm with Best Possible Time Quantum and Comparison and Analysis with The Rr Algorithm. International Research Journal of Engineering and Technology (IRJET), 03(03), pp.1357- 360.

[9] D. Nayak, S. K. Malla, and D. Debadarshini(2012). Improved Round Robin Scheduling using Dynamic Time Quantum. International Journal of Computer Applications, Vol. 38, No. 5, 2012, 34-38.

[10] Abur, M., Mohammed, A., Danjuma, S. and Abdullahi, S. (2018). A Critical Simulation of CPU Scheduling Algorithm using Exponential Distribution. International Journal of Computer Science Issues, 8(6), pp.201-205

[11] M. U. Farooq, et. al. (2017). An Efficient Dynamic Round Robin algorithm for CPU scheduling. IEEE 2017 International Conference on Communication, Computing and Digital Systems (C-CODE)