Journal of Research in Science and Engineering (JRSE)
ISSN: 1656-1996 Volume-7, Issue-8, August 2025
DOI: 10.53469/jrse.2025.07(08).03

Latency-Aware Round Robin Variants for Heterogeneous Workload Allocation

Naresh Raj K

[1]College of Information Technology and Computer Science, University of the Cordilleras,
Governor Pack Road, Baguio City, Benguet, Philippines
naresh9147@students.uc-bcf.edu.ph

Abstract: ***Operating Systems (OS) performs a variety of tasks, with scheduling being a fundamental function. Scheduling involves making decisions on how to allocate resources among different processes in tasks in order to maximize performance metrics. Some of the popular scheduling algorithms are First Come First Serve (FCFS), Shortest Job First (SJF), Shortest Remaining Time (SRT), Priority Scheduling, and Round Robin (RR), which exhibit differences in performance depending on the circumstances. The primary goal of a scheduling algorithm is to increase the effectiveness, speed, and fairness of the system. Considering these parameters, to improve and be suitable scheduling algorithm, studies have proposed hybrid algorithms. To understand the performance trade-offs between these hybrids, this paper compares different RR-based hybrid algorithms namely, Priority Based Round Robin (PBRR), Round Robin and Short Job First (RRSJF), Efficient Shortest Remaining Time Round Robin (ESRR). Results showed that RRSJF and ESRR performs better than the remaining hybrid algorithm in terms of Average Waiting Time (AWT) and Average Turnaround Time (ATT) with 17.2ms and 13ms AWT 13 and 26.0ms and 21.8ms ATT, respectively.***

Keywords: Hybrid Scheduling, Waiting Time, Turnaround Time, Context Switch

1. Introduction

Scheduling is one of the variety of tasks that operating systems (OS) carry out. This involves making decisions on how to allocate resources among different processes in tasks in order to maximize performance metrics like waiting time, throughput, turnaround, etc. [2], [11]. Scheduling is a crucial function of the OS, as almost all computer resources, including the CPU, are scheduled before use [5].

Multiprogrammed OS rely heavily on CPU scheduling to maximize the utilization of the CPU. In a single-processor system, only one process can run at a time, causing other processes to wait for their turn [13]. Thus, given the CPU's vital role as a computer resource, efficient scheduling algorithms are crucial for managing multiple processes arriving in main memory [18]. Some of the popular scheduling algorithms are First Come First Serve (FCFS), Shortest Job First (SJF), Shortest Remaining Time (SRT), Priority Scheduling, and Round Robin (RR), which exhibit differences in performance depending on the circumstances [2], [5], [14].

RR is one of the popular scheduling algorithms in OS for its simplicity, fairness, and generality in resource allocation [4]. This algorithm is specifically meant for time sharing systems. In this setup, each takes turns having an equal share of something [6]. However, RR has several drawbacks including low throughput, high turnaround time, high waiting time, and large number of context switches [6], [8], [10], [16], [19]. One of the most significant issues is the choice of the time slice length called time quantum (TQ) [2], [7]. A large TQ results in poor response time as it allows long tasks to dominate the processor, causing short tasks to wait [4], [10], [15]. Conversely, in a small TQ, long tasks take a long time to execute, which then causes poor CPU performance [4], [15].

The primary goal of a scheduling algorithm is to increase the effectiveness, speed, and fairness of the system. CPU utilization, context switching, throughput, waiting time, turnaround time, and response time are a few performance metrics [2], [21], [22]. Context switching stores and restores the context (state) of a preempted process, allowing execution to resume at a later time, but it leads to time and memory wastage [9]. CPU utilization measures CPU activity, with the primary objective being to maximize CPU utilization [12]. Throughput is the number of processes completed per unit time, aiming to maximize the work performed [12], [17]. Waiting time is the difference between start time and ready time, aiming to minimize the time spent by a process in a ready queue [12], [17]. On the other hand, turnaround time refers to the time it takes from submission to completion of a process, with the primary objective being to minimize this time [12], [17]. In interactive processes, response time refers to the time between launching a request and receiving the first response, which is typically limited by the output device [12], [17].

Considering these parameters, to improve and be suitable scheduling algorithm, hybrid algorithms were proposed. Results showed better performance compared to the traditional RR, minimizing waiting time, turnaround time, response time, and number of context switches, while maximizing throughput and CPU utilization [1], [20], [23].

Understanding the performance trade-offs between these hybrids is crucial for optimizing system performance in diverse environments. Therefore, this paper aims to answer the ff. questions, comparing the analyzed performance of three (3) RR-based hybrid algorithms, including Priority Based Round Robin (PBRR), Short Job First and Round Robin (SJFRR), and Efficient Shortest Remaining Time Round Robin (ESRR), to confirm their optimality performance:

1) What are the key factors influencing the performance of Round Robin-based hybrid algorithms?

2) What is the performance of Round Robin, Priority Based Round Robin, Round Robin and Short Job First, and Efficient Shortest Remaining Time Round Robin in terms of:
   a) Average Waiting Time;
   b) Average Turnaround Time; and
   c) Number of Context Switches

By identifying the strengths and weaknesses of each hybrid under different workloads and conditions, this study intends to provide valuable insights into selecting the most suitable variant for specific application scenarios.

2. Background Work

2.1 Priority Based Round Robin (PBRR)

This hybrid algorithm incorporates priority-based task management, assigning a priority index to each process and sorting them in the ready queue accordingly. Processes with smaller priority index are placed in the head of the ready queue and so on [23].

2.2 Round Robin and Short Job First (RRSJF)

In this hybrid algorithm, two algorithms are merged namely, Round Robin and Short Job First. It assumes simultaneous arrival of processes and then identifies the value of TQ. Processes are then ranked based on CPU Burst Time in ascending order, taking the smaller value between CPU burst times for the processes. The algorithms grant a process the quantum value, moving to the next process if it finishes, or waiting for the next loop if not finished within the quantum [1].

2.3 Efficient Shortest Remaining Time Round Robin (ESRR)

Shortest Remaining Time First (SRTF) algorithm estimates the processing time required by each process by keeping track of average CPU burst time, while RR operates on the principle of First Come, First Served (FCFS) with preemption functionality to switch between processes. FCFS strictly ordered the list of processes in the ready state based on their arrival in the system [17]. To reduce total waiting time and turnaround time from each individual algorithm, this hybrid algorithm ESRR combines the SRTF and time-sharing concept of RR [20].

3. Experimental Analysis

Prior to submitting the processes to the scheduler for each experiment, the performance evaluation assumes a single processor environment and known burst time for all processes. Each experiment consists of multiple input and output parameters. The number of processes, arrival time, burst time, time quantum, and priority index for PBRR are the input parameters. Conversely, the number of context switches (CS), average waiting time (AWT), and average turnaround time (ATT) are the output parameters. Two experiments were performed to evaluate the performance of the algorithms, with arrival time assumed to be the same for all processes.

Consider five processes (P1, P2, P3, P4, P5) with arrival time = 0 and random burst time (9, 3, 16, 5, 11) as shown in Table 1. RR, PBRR, RRSJF, and ESRR algorithms are then applied to schedule these processes.

Table 1: Five processes with random burst time and priority index

Process	Burst Time *(ms)*	Priority
P1	9	3
P2	3	2
P3	16	5
P4	5	1
P5	11	4

3.1 Case 1

In this case, TQ is assumed to be 3ms. The resulting Gantt charts for the RR, PBRR, RRSJF, and ESRR algorithms are shown in Figures 1, 2, 3, 4, respectively.

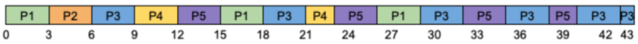

Figure 1: Gantt chart of RR Scheduling with TQ = 3ms

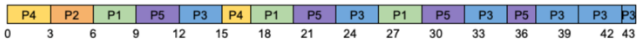

Figure 2: Gantt chart of PBRR Scheduling with TQ = 3ms

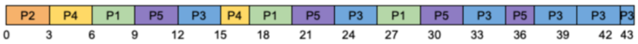

Figure 3: Gantt chart of RRSJF Scheduling with TQ = 3ms

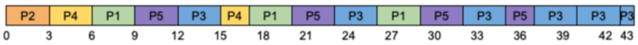

Figure 4: Gantt chart of ESRR Scheduling with TQ = 3ms

3.2 Case 2

In this case, TQ is computed using the ff. formula [3]:

$$TQ = \frac{BT\ of\ \ all\ processes}{No.\ of\ \ processes} \tag{1}$$

With total BT equal to 44, the TQ is 8.8ms ≈ 9ms. Thus, the resulting Gantt charts for the RR, PBRR, RRSJF, and ESRR algorithms are shown in Figures 5, 6, 7, and 8, respectively.

Figure 5: Gantt chart of RR Scheduling with TQ = 9ms

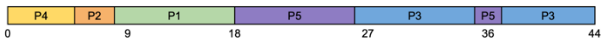

Figure 6: Gantt chart of PBRR Scheduling with TQ = 9ms

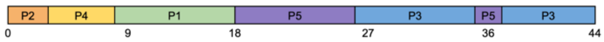

Figure 7: Gantt chart of RRSJF Scheduling with TQ = 9ms

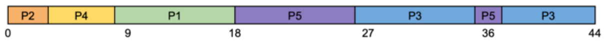

Figure 8: Gantt chart of ESRR Scheduling with TQ = 9ms

4. Comparison of Results

Table 2: Average Waiting Time, Average Turnaround Time, and Number of Context Switches in Case 1

Algorithm	AWT	ATT	CS
RR	19.6	28.4	16
PBRR	17.8	26.6	16
RRSJF	17.2	26.0	16
ESRR	17.2	26.0	16

Table 3: Average Waiting Time, Average Turnaround Time, and Number of Context Switches in Case 2

Algorithm	AWT	ATT	CS
RR	17.8	26.6	7
PBRR	13.4	22.2	7
RRSJF	13.0	21.8	7
ESRR	13.0	21.8	7

5. Conclusion

According to Experimental Analysis, we conclude that RRSJF and ESRR algorithm give better performance than the remaining RR-based hybrid scheduling algorithms in both cases. Since the arrival time of processes are simultaneous, the scheduler's decision-making is more critical. The priority index assigned to processes also influences the performance as higher priority processes may preempt lower priority ones, impacting the waiting and turnaround times of the lower priority processes. The time quantum assigned also affected the number of context switches as smaller time quantum leads to more frequent context switches. Overall, with the hybrid of Round Robin and Short Job First and Round Robin and Shortest Remaining Time First, Average Waiting Time (AWT) and Average Turnaround Time (ATT) can be reduced.

References

[1] R.N. Alhawadi, m.o. nassar, O.A. tarawneh, "Hybrid Scheduling Algorithms Using Round Robin and Short Job First," International Journal of Science and Applied Information Technology, VII (1), pp. 12-16, 2017.

[2] S.B. Bandarupalli, N.P. Nutulapati, P.S. Varma, "A Novel CPU Scheduling Algorithm–Preemptive & Non-Preemptive," International Journal of Modern Engineering Research, II (6), pp. 4484-4490, 2012.

[3] D. Barman, "Dynamic Time Quantum in Round Robin Algorithm (DTQRR) Depending on Burst and Arrival Time of the Processes," International Journal of Innovative Technology and Exploring Engineering, II (4), pp. 60-64, 2013.

[4] A. Fiad, Z.M. Maaza, H. Bendoukha, "Improved Version of Round Robin Scheduling Algorithm Based on Analytic Model," International Journal of Networked and Distributed Computing, VIII (4), pp. 195-202, 2020.

[5] N. Goel, R.B. Garg, "A Comparative Study of CPU Scheduling Algorithms," International Journal of Graphics and Image Processing, II (4), pp. 245-251, (2012).

[6] A. Gupta, I. Nagpal, "Revised Round Robin Scheduling Algorithm," International Journal of Computer Science and Information Technology Research, III (3), pp. 164-168, (2015).

[7] R.N.D.S.S. Kiran, P.V. Babu, B.B.M. Krishna, "Optimizing CPU Scheduling for Real Time Applications Using Mean-Difference Round Robin (MDRR) Algorithm," ICT and Critical Infrastructure: Proceedings if the 48th Annual Convention of Computer Society of India, pp. 713-721, 2014.

[8] A. Muraleedharan, N. Antony, R. Nandakumar, "Dynamic Time Slice Round Robin Scheduling Algorithm with Unknown Burst Time," Indian Journal of Science and Technology, IX (8), pp. 1-6, 2016.

[9] S. Negi, P. Kalra, "A Comparative Performance Analysis of Various Variants of Round Robin Scheduling Algorithm," International Journal of Information & Computation Technology, IV (7), pp. 765-772, 2014.

[10] A. Noon, A. Kalakech, S. Kadry, "A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average," International Journal of Computer Science Issues, VIII (3), pp. 224-229, 2011.

[11] T.O. Omotehinwa, "Examining the developments in scheduling algorithms research: A bibliometric approach," Heliyon, VIII (5), 2022.

[12] E.O. Oyetunji, A.E. Oluleye, "Performance Assessment of Some CPU Scheduling Algorithms," Research Journal of Information Technology, I (1), pp. 22-26, 2009.

[13] M. Pirani, D. Ranpariya, M. Vaishnav, "A Comparative Review of CPU Scheduling Algorithms," International Journal of Scientific Research & Engineering Trends, VII (4), 2448-2452, 2021.

[14] H.B. Parekh, S. Chaudhari, "Improved Round Robin CPU scheduling algorithm: Round Robin, Shortest Job First and priority algorithm coupled to increase throughput and decrease waiting time and turnaround time," International Conference on Global Trends in Signal Processing, Information Computing and Communication, pp. 184-187, 2016.

[15] I. Qureshi, "CPU Scheduling Algorithms: A Survey," International Journal of Advanced Networking and Applications, V (4), pp. 1968-1973, 2014.

[16] I.S. Rajput, D. Gupta, "A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems," International Journal of Innovations in Engineering and Technology, I (3), 2012.

[17] M.G. Rodriguez, L.O. Cerdeira, E.G. Rufino, F.J.R. Martínez, "Study and evaluation of CPU scheduling algorithms," Heliyon, X (9), 2024.

[18] B. S, C. MP, D. SN, "Comprehensive Analysis of CPU Scheduling Algorithms," International Research Journal of Modernization in Engineering Technology and Science, IV (9), pp. 180-185, 2022.

[19] A. Singh, P. Goyal, S. Batra, "An Optimized Round Robin Scheduling Algorithms for CPU Scheduling," International Journal on Computer Science and Engineering, II (7), pp. 2383-2385, 2010.

[20] P. Sinha, B. Prabadevi, S. Dutta, N. Deepa, N. Kumari, "Efficient Process Scheduling Algorithm using RR and SRTF," 2020 International Conference on Emerging Trends in Information Technology and Engineering, pp. 1-6, 2020.

[21] P.T. Sonagara, V.D. Sanghvi, "A Comparative Parameters Analysis of Different Round Robin

Scheduling Algorithm using Dynamic Time Quantum," International Journal of Science and Research, III (12), pp. 2045-2047, 2014.

[22] K. Vayadande, S. Patil, S. Chauhan, R. Thakur, T. Baware, S. Naik, "A Survey Paper on CPU Process Scheduling," International Conference on Recent and Future Trends in Smarts Electronic System and Manufacturing, pp. 40-47, 2023.

[23] S. Zouaoui, L. Boussaid, A. Mtibaa, "Priority based round robin (PBRR) CPU scheduling algorithm," International Journal of Electrical and Computer Engineering, IX (1), pp. 190-202, 2019.