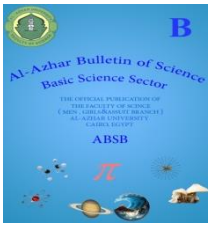
Al-Azhar Bulletin of Science: Section B, Vol. 32, No. 2, (December) 2021, pp.45-54
http://doi.10.21608/absb.2021.99340.1141

Al-Azhar Bulletin of Science: Section B



VORR: A NEW ROUND ROBIN SCHEDULING ALGORITHM

Afaf A. Abdelhafiz *

Department of Mathematics, Division of Computer Science, Faculty of Science, Al-Azhar University (Girls Branch), Cairo, Egypt.
* Corresponding Author: afaf2azhar@azhar.edu.eg

Received: 03 Oct 2021; Revised: 24 Nov 2021; Accepted: 28 Nov 2021; Published: 01 Dec 2021

ABSTRACT

Task scheduling on resources is critical for improving the performance of real-time systems. Enormous efforts have recently and rapidly been proposed by many researchers with improvements in various accessible algorithms of the task scheduling process. Each improvement contributes to the optimization of the system act. The field of enhancing CPU scheduling algorithms (such as the Round Robin algorithm RR) is still open. The Round Robin scheduling has many negatives, such as high waiting time, high turnaround time, or a high number of context switches. Such negatives affect the performance of the algorithm badly. This paper discusses a new approach named, VORR (Variant On Round Robin), which is one of the improvements and enhancements to the Round Robin scheduling algorithm. It effectively exploits the CPU by setting up an effective time quantum based on the median of burst times. The experimental results have demonstrated the worth of the proposed approach in comparison with the traditional Round Robin scheduling algorithm and its recent editions in terms of average waiting time, average turnaround time, and number of context switches. Also, it enhances the response time for some RR algorithms.

Keywords: Round robin algorithm; CPU scheduling; Time quantum; Average waiting time; Average turnaround time.

1. INTRODUCTION

Scheduling tasks and allocating resources are considered one of the main challenges in a system that affects the act of working machines. It determines the execution order of the tasks according to specified criteria [1]. The development of an efficient scheduling algorithm is one of the most vital research fields. A scheduling algorithm optimizes any of the following criteria:

- Turnaround Time: the time between the starting of process and its completion.
- Waiting Time: it is the time that a process waited in the ready queue.
- CPU utilization: using the CPU as much as possible.
- Throughput: indicate the number of processes finished per time unit.
- Response Time: the time from submission of a process to the first response.
- Context Switch: A context switch is the process of storing and restoring context (state) of an interrupted process. So, the execution can be resumed from same point at a later time. Context switching is usually computationally intensive, leading to a wastage of time and memory, which in turn increases the overhead of scheduler, so the design of operating system is to optimize these switches. At any time, there are a lot of ready programs which are required to be executed

simultaneously. The main objective of an operating system is to provide an efficient way to perform a task request. Resource management techniques are utilized for better performance of operating systems. The system scheduler does the scheduling processes to decide which process will access the system resources next. The initial work of the scheduler is to select the next process from the ready queue, dispatch it to the CPU and reduce the starvation problem. Two classes of scheduling algorithms are known: preemptive scheduling and non-preemptive scheduling. In the preemptive scheduling, a process may be interrupted during its execution to perform another highest priority process. In the non-preemptive scheduling, the process is run until its end [2]. The Round Robin (RR) algorithm and its variants are considered preemptive algorithms. The intended approach has been built based on the process scheduling method known as RR. The contributions of the proposed algorithm are:

1- Reducing the average waiting time and average turnaround time

2- Decreasing the number of context switches comparing with RR, IRR, RRRT and Enriched RR algorithms.

3- Enhancing the response time produced by RR and IRR algorithms.

2. CPU scheduling Algorithms

Many efforts have been done by numerous researchers to minimize associated issues of task scheduling algorithms. Authors of [3] proposed an algorithm called AN, based on a new approach called dynamic-time-quantum; the idea of this approach is to make the operating systems adjusts the time quantum according to the burst time of the set of waiting processes in the ready queue. In [4], the authors designed an approach that repeatedly elects an optimal time quantum during the execution of process and sets time quantum actively on the base of process burst time that remains in ready queue. It also reduces the context switching of process in queue or resources.

An algorithm which increases the speed up factor is proposed in [5]. It depends on the pipelining technique for this aim. The analysis of this proposed algorithm showed that its performance is better than the existing scheduling algorithms by 40-50%. In [6], authors presented an improved Round Robin CPU scheduling algorithm to enhance CPU performance using the features of Shortest Job First and Round Robin scheduling with varying time quantum. Another proposed approach in [7] evaluated shortest job first (SJF) algorithm and proposes an algorithm to overcome the shortcomings of SJF with respect to average turnaround time and average waiting time. The algorithm in [8] improved the performance of some RR algorithms and developed a time quantum that achieved stability in terms of the average waiting time and the average turnaround time.

The approach proposed in [9] improved the real time operating system CPU performance. It is based on the combination of round-robin (RR) and Priority based (PB) scheduling algorithms. The algorithm IRR in [10] gives better act than RR concerning the waiting time and turnaround time. It uses a dynamic time quantum calculated from the burst time of the set of waiting processes in the ready queue. In [11], a modified version of the RR algorithm started by clustering the processes into clusters. Every process in a cluster is assigned the same time slice depending on the weight of its cluster and its CPU service period. The experiments showed that the proposed approach gives better results. Authors of [12] proposed an algorithm called hybrid Round Robin scheduling mechanism (HYRR Mechanism) for process management. HYRR Mechanism is an innovative scheduling algorithm with enhanced time quantum. It inherits the properties of Round robin, shortest job first (SJF) algorithm and first come first serve algorithm (FCFS). HYRR decreases ATT, ART and AWT to the desired levels. The technique in [13] partitioned the ready queue into three sub-queues: lowest, medium, and highest priority. The assignment of the TQ to one of those sub-queues depends on a threshold value. All processes in TQ1 should be executed first. Then processes in TQ2 and TQ3,

respectively. The experimental results show that the proposed approach is outperforming some other approaches. The authors in [14] proposed a model which uses a variable time quantum based on an analytic model that takes into consideration different parameters to determine the order of tasks execution. Their algorithm can be applied in any operating system and in the cloud computing environment. In [15], a clustering inspired novel intelligent approach has been developed which used standard deviation and an optimal threshold to create an intelligent sub ready queue. A dynamic time quantum is generated for each cluster. The performance is evaluated on sample data with respect to waiting time, response time, turnaround time and context switching. The results of the developed approach are compared with other existing algorithms and are found better. Authors of [16] proposed an algorithm called Adjustable Time Slice (ATS). It takes the advantage of low overhead of RR algorithm. It also favors short processes allowing them to execute longer time than long processes. The results showed that the proposed algorithm decreases the time cost. Although all of these algorithms have been developed to enhance the original RR algorithm, many algorithms still have a high waiting time and turnaround time. Also, other algorithms may suffer from high number of context switch or high response time. The proposed algorithm VORR reduces the average waiting time, average turnaround time and the context switch comparing with RR, IRR, RRRT and Enriched RR algorithms. Also, VORR algorithm enhances the response time produced by RR and IRR algorithms.

3. The Proposed Algorithm VORR (Variant On Round Robin)

To optimize the act of a scheduling algorithm, the intended method has put main focus on calculating an effective time quantum. It is assumed that all the processes are available at the beginning. First, the proposed method has set the ready queue increasingly according to burst time of processes and then sets up the time quantum which depends on the median of execution times:

The median is the middle number in a sorted, ascending or descending, list of numbers

- If there is an odd amount of numbers, the median value is the number that is in the middle, with the same amount of numbers below and above.
- If there is an even amount of numbers in the list, the middle pair must be determined, added together, and divided by two to find the median value.
- The proposed algorithm sets up the TQ as follows: TQ=2*median

At beginning, VORR allocates system resource to the first process of ready queue. At completion of either process execution or TQ, the system checked the status of the current process, if it is completed, then it is deleted from the ready queue. Otherwise, it is placed at the end of the ready queue waiting for the next round and so on.

3.1 The pseudo-code of the VORR algorithm

1. Assign all processes to the ready queue
2. Sort all processes in an increasing order of their execution times
3. Set TQ=2*median
4. While (the ready queue is not empty)
5. Assign the first ready process to the processor for a time = TQ
6. If its execution time <= 1 TQ then it is executed till completion and then deleted from the queue

7. Else if its execution time >1 TQ then it is executed for 1 TQ and then appended to the end of the ready queue

8. End while

9. Compute the average of both waiting time, turnaround time, response time and the number of context switches.

3.2 The flowchart of the VORR algorithm

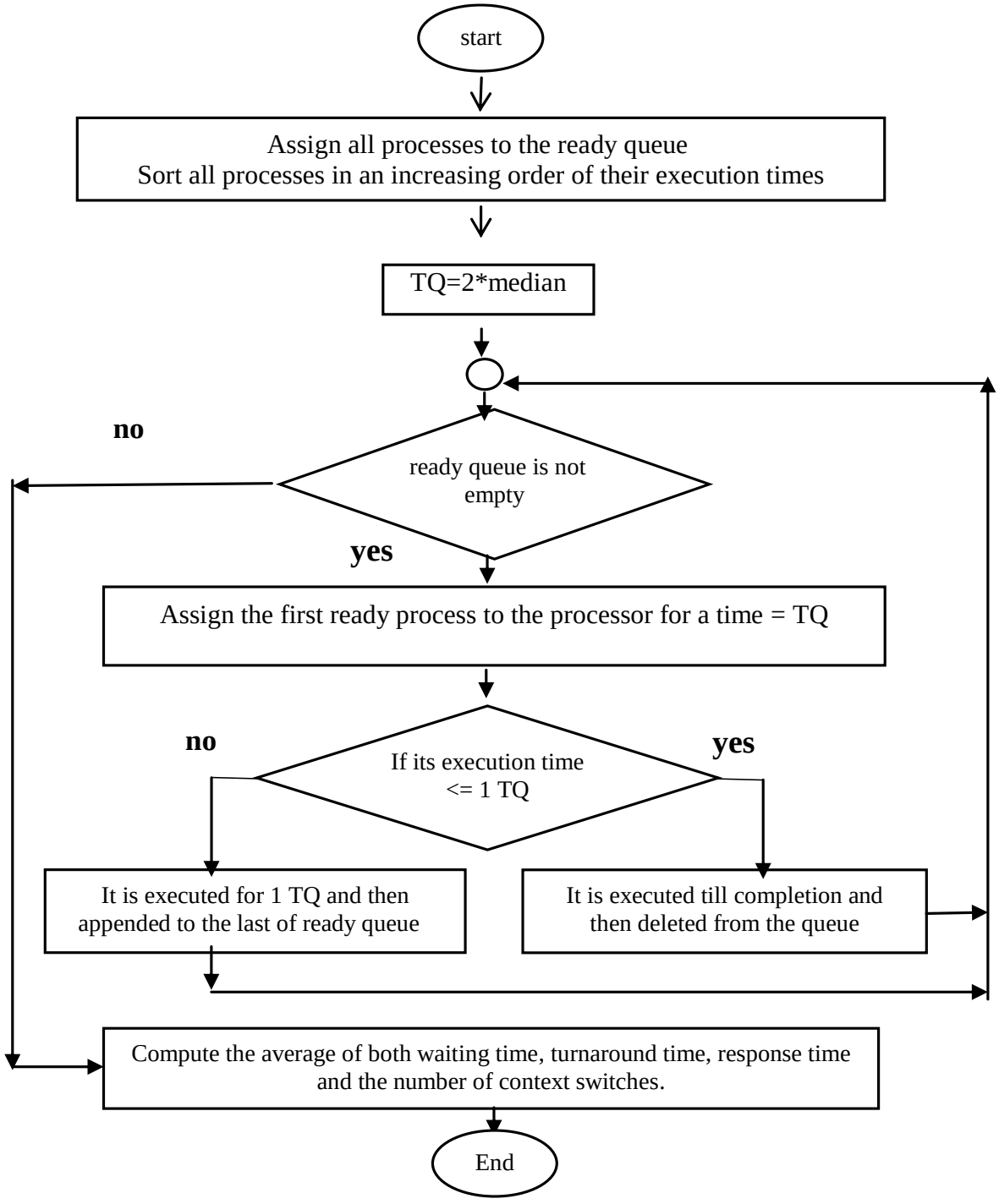


Fig. 1: Flow chart of the VORR Algorithm

4. RESULTS DISCUSSION

To confirm the act of proposed method, Table 1 illustrated the details of 6 processes considered for experimental discussion.

Table 1: Details of 6 processes

Processes IDs	Arrival Time	Burst Time(ms)
p1	0	8
p2	0	3
p3	0	11
p4	0	19
P5	0	2
P6	0	30

Gant chart of proposed approach:

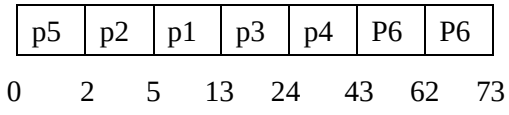


Fig. 2: Gantt chart for the VORR Algorithm

Average Waiting Time=14.5 ms

Average Turnaround Time =26.6 ms

Average response time=14.5 ms

Number of context switches=5

Applying the RR algorithm with TQ=9 ms for these processes yields the following Gantt chart (Fig.3)

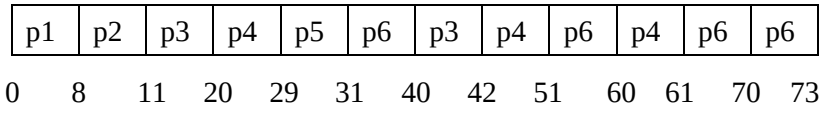


Fig. 3: Gantt chart for RR

Average Waiting Time= 25.5 ms

Average Turnaround Time = 37.6 ms

Average response time=16.5 ms

Number of context switches=10

Applying the RRRT algorithm where its time quantum is calculated as TQ=6 ms for these processes yields the following Gantt chart.

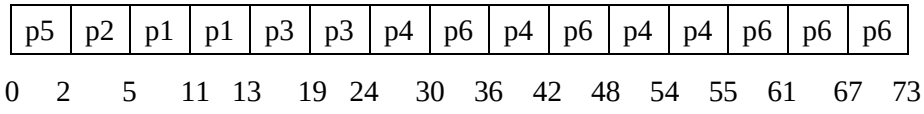


Fig. 4: Gantt chart for RRRT Algorithm

Average Waiting Time=16.5ms

Average Turnaround Time =28.66ms

Average response time=12.3ms

Number of context switches=9

Applying the algorithm Enriched RR where its time quantum is calculated as TQ=9 ms for these processes yields the following chart.

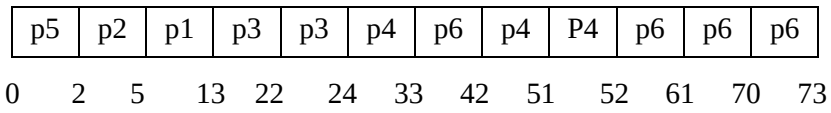


Fig. 5: Chart of Gantt for Enriched RR

Average Waiting Time=16ms

Average Turnaround Time =28.16ms

Average response time=12.8ms

Number of context switches=7

This table summarizes the above results (* mark means best result).

Table 2: Summary of previous results

Criteria	VORR	RR	RRRT	Enriched RR
Average Waiting Time	14.5*	25.5	16.5	16
Average Turnaround Time	26.6*	37.6	28.66	28.16
Average response time	14.5	16.5	12.3*	12.8
Number of context switches	5*	10	9	7

The following figure shows that the proposed method VORR have attained 14.5 ms average waiting time which is the smallest value against RR, IRR, RRRT and Enriched RR algorithms.

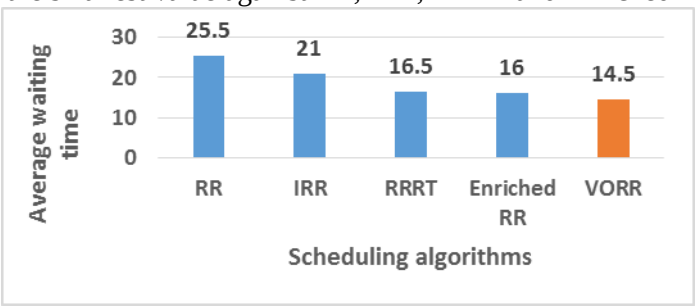


Fig. 6: Comparing the average waiting time for proposed method and other algorithms

The figure below shows that the proposed method have attained 26.6 ms average turnaround time which is the smallest value against RR, IRR, RRRT and Enriched RR algorithms.

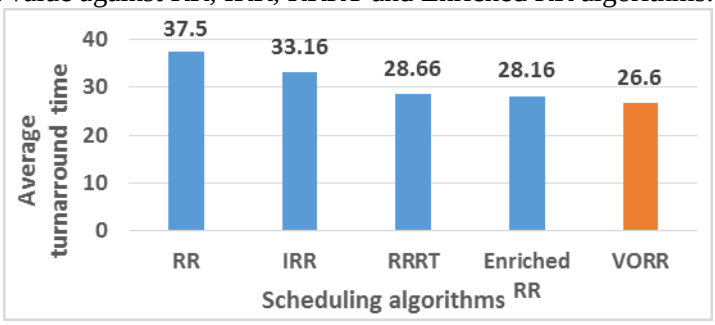


Fig. 7: Comparing the average turnaround time for proposed method and other algorithms

The number of context switches is depicted in the following figure which shows that the proposed method have attained 5 context switches which is the smallest value against RR, IRR, RRRT and Enriched RR algorithms.

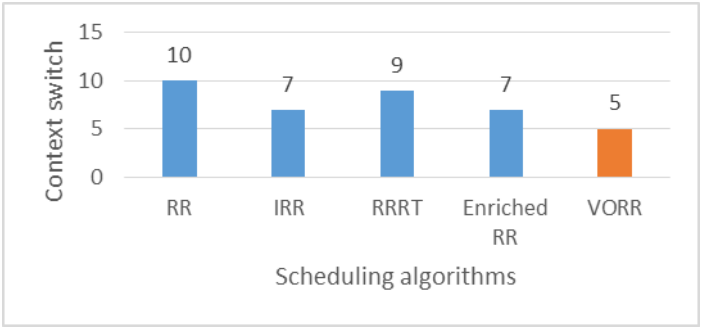


Fig. 8: Comparing the number of context switch for proposed method and other algorithms

The following figure describes that the proposed method has attained 14.5 ms average response time which is smaller value than that of RR and IRR algorithms. Since the RRRT and Enriched RR algorithms cut processes many times in such a way that causes a reduction in the response time, the VORR algorithm has larger average response time than RRRT and Enriched RR.

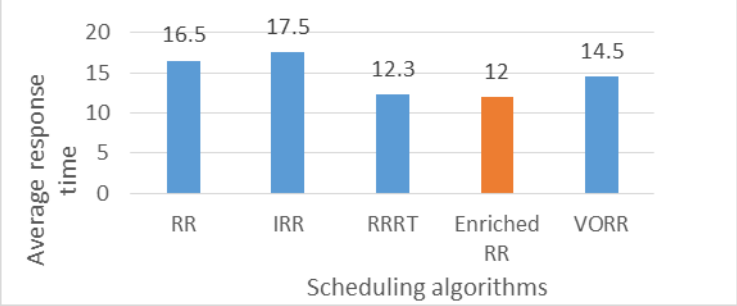


Fig. 9: Comparing the average response time for proposed method and other algorithms

Implementations for the RR, IRR, RRRT, Enriched RR, and VORR algorithms are carried out to examine the efficiency of each one. The model for these implementations consists of a number of processes equal to n where n ranges from 100 to 1000. The execution times of processes are randomly generated from 1 to 2000 using the predefined C++ function; rand().

To show the benefits of VORR compared to the RR, IRR, RRRT, Enriched RR; the figure below schemes the average waiting time versus the number of processes for both the VORR algorithm and

other algorithms. It is observed that the VORR algorithm has better average waiting time than other algorithms.

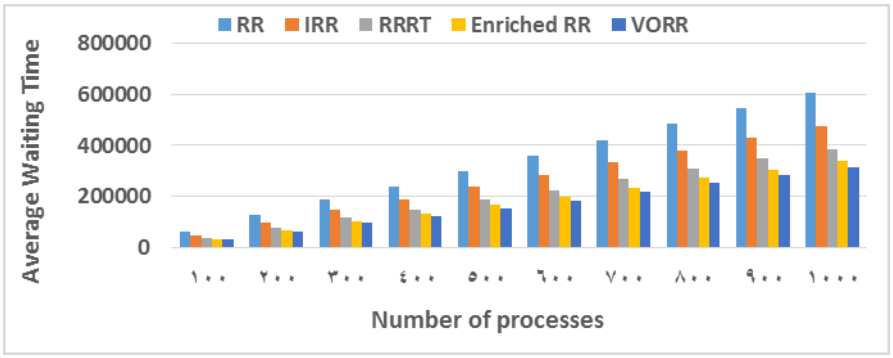


Fig. 10: Average waiting time for VORR and other algorithms

The following figure schemes the average turnaround time for VORR and other algorithms. It is clear from this figure that the VORR algorithm has better average turnaround time than other algorithms.

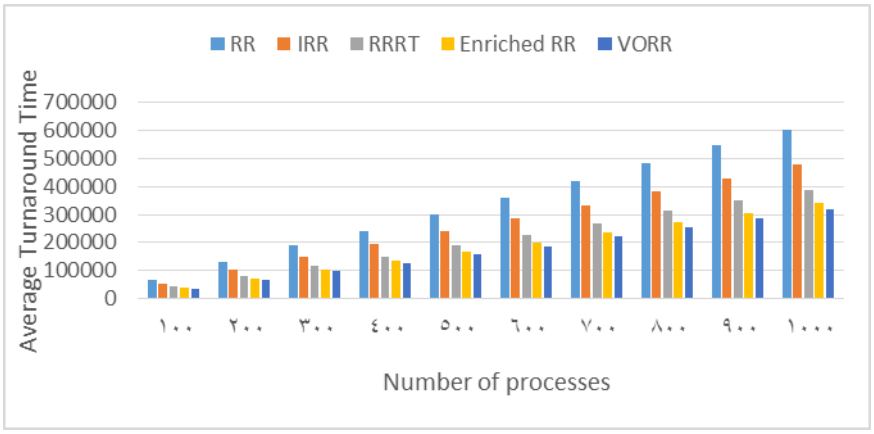


Fig. 11: Average turnaround time for VORR and other algorithms

The number of context switches for VORR and other algorithms is depicted in the figure below. The figure shows that the VORR algorithm has smaller number of context switches than other algorithms.

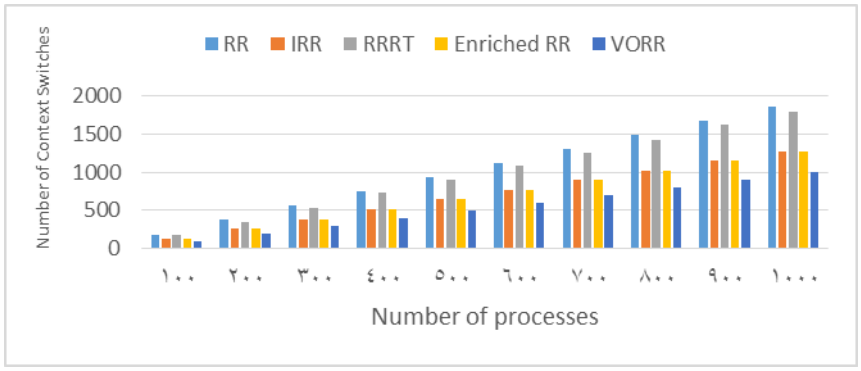


Fig. 12: Number of context switches for VORR and other algorithms

The average response time for VORR and other algorithms is depicted in the following figure. It is showed from this figure that the VORR algorithm has better average response time than only IRR algorithm.

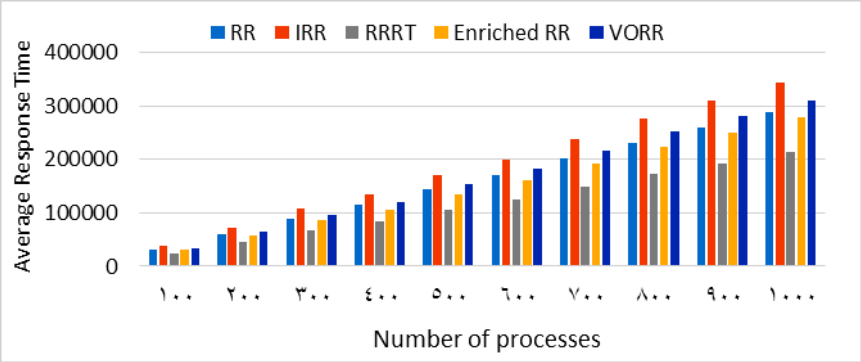


Fig.13: Average response time for VORR and other algorithms

5. CONCLUSION

The key for enhancing RR algorithms is to choose an intelligent time quantum. This paper proposed a variant on the classical RR scheduling algorithm based on an efficient time quantum value to enhance the act of a system. Choosing this value for the time quantum reduces the average waiting time, the average turnaround time and the context switches for the RR, IRR, RRRT and Enriched RR algorithms. The average response time is also enhanced for the VORR algorithm comparing with IRR algorithm. A number of simulations and comparisons have been done which have evidently shown the efficiency of the proposed algorithm. It is intended in the future to search for more effective time quantum value aiming to further reduction in waiting time, turnaround time, response time and number of context switches.

REFERENCES

[1] Mahmoud H, Thabet M, Khafagy MH, Omara FA. An efficient load balancing technique for task scheduling in heterogeneous cloud environment. Cluster Comput [Internet]. 2021;24(4):3405–3419. Available from: https://doi.org/10.1007/s10586-021-03334-z

[2] Rofifah D. Operating Systems Internal and Design Principles. Paper Knowledge . Toward a Media History of Documents. 2020. 12–26 p.

[3] Noon A, Kalakech A, Kadry S. A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average. 2011;(November 2011):224-229. Available from: http://arxiv.org/abs/1111.5348

[4] Dave B, Yadev S, Mathuria M, Sharma YM. Optimize task scheduling act by modified round robin scheme with vigorous time quantum. Proc Int Conf Intell Sustain Syst ICISS 2017. 2018;(Iciss):905–910.

[5] Arora H, Arora D, Goel B, Jain P. An Improved CPU Scheduling Algorithm. Int J Appl Inf Syst. 2013;6(6):7–9.

[6] Kumar Mishra M, Rashid F. An Improved Round Robin CPU Scheduling Algorithm with Varying Time Quantum. Int J Comput Sci Eng Appl. 2014;4(4):1–8.

[7] Garg S, Kumar D. A K-Factor CPU Scheduling Algorithm. 2018 9th Int Conf Comput Commun Netw Technol ICCCNT 2018. 2018;1–6.

[8] ElDahshan K, Abd A, Ghazy N. Achieving Stability in the Round Robin Algorithm. Int J Comput Appl. 2017;172(6):15–20.

[9] Zouaoui S, Boussaid L, Mtibaa A. Priority based round robin (PBRR) CPU scheduling algorithm. Int J Electr Comput Eng. 2019;9(1):190-202.

[10] a S. a Modified Round Robin Cpu Scheduling Algorithm With Dynamic Time Quantum. Int J Adv Res. 2019;7(2):422–429.

[11] Mostafa SM, Amano H. Dynamic round robin CPU scheduling algorithm based on K-means clustering technique. Appl Sci. 2020;10(15).

[12] Faizan K, Marikal A, Anil K. A Hybrid Round Robin Scheduling Mechanism for Process Management. Int J Comput Appl. 2020;177(36):14–19.

[13] Arif KI, Morad M, Mohammed MA, Subhi MA. AN EFFICIENT THRESHOLD ROUND-ROBIN SCHEME for CPU SCHEDULING (ETRR). J Eng Sci Technol. 2020;15(6):4048–4060.

[14] Fiad A, Maaza ZM, Bendoukha H. Improved version of round robin scheduling algorithm based on analytic model. Int J Networked Distrib Comput. 2020;8(4):195–202.

[15] Sharma PS, Kumar S, Gaur MS, Jain V. A novel intelligent round robin CPU scheduling algorithm. Int J Inf Technol [Internet]. 2021; Available from: https://doi.org/10.1007/s41870-021-00630-0

[16] Mostafa SM, Idris SA, Kaur M. ATS: A novel time-sharing cpu scheduling algorithm based on features similarities. Comput Mater Contin. 2022;70(3):6271–6288.

خوارومية روند روبن جديدة VORR

عفاف عبد القادر عبد الحفيظ

قسم الرياضيات- كلية العلوم (فرع البنات) – جامعة الازهر-القاهرة-مصر

الملخص:

ان جدولة المهام على المصادر مهم جدا لتحسين كفاءة انظمة الوقت الحقيقية. يتقدم الباحثون بجهود كثيرة و سريعة لتحسين خوارزميات جدولة المهام. كل تحسين يشارك فى رفع كفاءة النظام. ان مجال تحسين جدولة وحدة اللتشغيل المركزية (مثل خوارزمبة ال روند روبن) ما زال مفتوحا. ان خوارزمية ال روند روبن لها العديد من السلبيات مثل وقت انتظار طويل ، الوقت الكلى لتنفيذ المهمة طويل او عدد مرات التبديل بين المهام كثير. مثل هذه السلبيات تؤثر بطريقة سيئة على كفاءة الخوارزمية. هذا البحث يقدم خوارزمية جديدة كاحد التحسنات على خوارزمية ال روند روبن عن طريق اقتراح قيمة فعالة لشريحة الوقت التى تعطى لكل مهمة. اثبتت النتائج التجريبية ان الخوارزم الجديد يفضل خوارزمية روند روبن و الاصدارات الحديثة منها.