[17] I. M. Ibrahim, "Task scheduling algorithms in cloud computing: A review," *Turkish Journal of Computer and Mathematics Education (TURCOMAT),* vol. 12, no. 4, pp. 1041-1053, 2021.

[18] A. Kazeem Moses, A. Joseph Bamidele, O. Roseline Oluwaseun, S. Misra, and A. Abidemi Emmanuel, "Applicability of MMRR load balancing algorithm in cloud computing," *International Journal of Computer Mathematics: Computer Systems Theory,* vol. 6, no. 1, pp. 7-20, 2021/01/02 2021, doi: 10.1080/23799927.2020.1854864.

❊❊❊❊

[10] T. Balharith and F. Alhaidari, "Round Robin Scheduling Algorithm in CPU and Cloud Computing: A review," in *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*, 1-3 May 2019 2019, pp. 1-7, doi: 10.1109/CAIS.2019.8769534.

[11] D. Biswas and M. Samsuddoha, "Determining Proficient Time Quantum to Improve the Performance of Round Robin Scheduling Algorithm," *International Journal of Modern Education & Computer Science,* vol. 11, no. 10, 2019.

[12] H. Esmael, "Developing Variance Considered Enhanced Round Robin Algorithm in Cloud Computing Environment," ASTU, 2021.

[13] A. Fiad, Z. M. Maaza, and H. Bendoukha, "Improved Version of Round Robin Scheduling Algorithm Based on Analytic Model," *Int. J. Networked Distributed Comput.,* vol. 8, no. 4, pp. 195-202, 2020.

[14] B. H. Shanthan, L. Arockiam, and A. C. Donald, "Task Scheduling algorithm in cloud computing: A Dynamic Variable Quantum Round robin," in *Journal of Physics: Conference Series*, 2020, vol. 1664, p. 012052.

[15] S. K. Mishra, B. Sahoo, and P. P. Parida, "Load balancing in cloud computing: a big picture," *Journal of King Saud University-Computer and Information Sciences,* vol. 32, no. 2, pp. 149-158, 2020.

[16] Y. Moon, H. Yu, J.-M. Gil, and J. Lim, "A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments," *Human-centric Computing and Information Sciences,* vol. 7, no. 1, p. 28, 2017/10/09 2017, doi: 10.1186/s13673-017-0109-2.

computing environments: Analysis, performance evaluation, and future directions," *Simulation Modelling Practice and Theory,* vol. 111, p. 102353, 2021/09/01/ 2021, doi: https://doi.org/10.1016/j.simpat.2021.102353.

[4] D. Tychalas and H. Karatza, "An Advanced Weighted Round Robin Scheduling Algorithm," in *24th Pan-Hellenic Conference on Informatics*, 2020, pp. 188-191.

[5] G. Sinha and D. K. Sinha, "Enhanced weighted round robin algorithm to balance the load for effective utilization of resource in cloud environment," *EAI Endorsed Transactions on Cloud Systems,* vol. 6, no. 18, 2020.

[6] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: A survey," *ACM Computing Surveys (CSUR),* vol. 51, no. 6, pp. 1-35, 2019.

[7] S. A. Alsaidy, A. D. Abbood, and M. A. Sahib, "Heuristic initialization of PSO task scheduling algorithm in cloud computing," *Journal of King Saud University - Computer and Information Sciences,* 2020/11/13/ 2020, doi: https://doi.org/10.1016/j.jksuci.2020.11.002.

[8] A. F. S. Devaraj, M. Elhoseny, S. Dhanasekaran, E. L. Lydia, and K. Shankar, "Hybridization of firefly and improved multi-objective particle swarm optimization algorithm for energy efficient load balancing in cloud computing environments," *Journal of Parallel and Distributed Computing,* vol. 142, pp. 36-45, 2020.

[9] K. F. Ali, A. Marikal, and K. A. Kumar, "A hybrid round robin scheduling mechanism for process management," *International Journal of Computer Applications,* vol. 975, p. 8887, 2020.

tasks. These findings reveal that the proposed SIBRR algorithm is the most suitable algorithm for scheduling of data with outlier tasks.

v. Conclusion

This paper proposed an enhanced scheduling algorithm based on Sum Interquartile boundaries. Results obtained show that the proposed algorithm achieves significant improvements in average waiting times, average turnaround times, and average completion times over baseline algorithm when there are outlier and heterogeneous tasks. These findings reveal that the proposed average Interquartile round robin scheduling (SIBRR) method is the most suitable algorithm for task scheduling algorithm for cloud computing when there are processes with outlier burst time. Further research is needed to enhance the enhanced round robin scheduling algorithm in real world environment and to evaluate the proposed algorithm in real-world experiment.

References

[1] F. Alhaidari and T. Z. Balharith, "Enhanced Round-Robin Algorithm in the Cloud Computing Environment for Optimal Task Scheduling," *Computers,* vol. 10, no. 5, p. 63, 2021.

[2] A. Pradhan, S. K. Bisoy, and A. Das, "A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment," *Journal of King Saud University - Computer and Information Sciences,* 2021/01/13/ 2021, doi: https://doi.org/10.1016/j.jksuci.2021.01.003.

[3] H. Singh, S. Tyagi, P. Kumar, S. S. Gill, and R. Buyya, "Metaheuristics for scheduling of heterogeneous tasks in cloud

outlier burst time. In addition, the process arrival times are 0,16, 8, 4, 2. The comparison results shown in Figure 5 show that Summation Interquartile Boundaries Round Robin scheduling (SIBRR) behaves better than the MAXDIFRR[11] algorithm in Case 4 in terms of average waiting times, average turnaround times, and average completion times. Results show that there is a big improvement in the proposed Summation Interquartile Boundaries Round Robin scheduling (SIBRR) algorithm compared to the MAXDIFRR[11] algorithm in terms of average waiting times, average turn-around times, and average completion times.

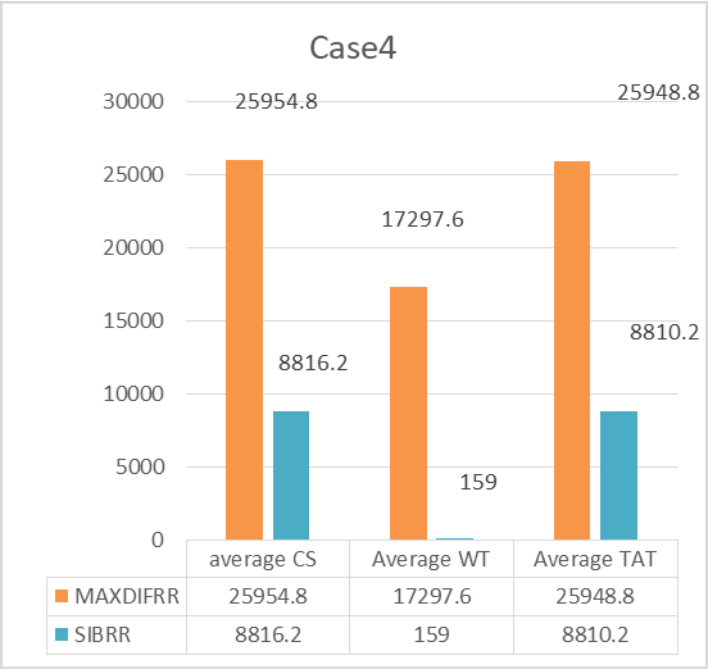

*Figure 5 Comparison of the results of MAXDIFRR[11] and proposed SIBRR for the **fourth** case.*

As observed, experiments show that the proposed SIBRR algorithm significantly outperforms all baseline algorithm on data that contains outlier

- **Case 3: Outlier Burst Time and Zero Arrival Time**: In the third case, five processes (P1, P2, P3, P4 and P5) are used. These processes have distinct burst times of 35, 26, 55, 43,000, and 105. In this case, p2 has an outlier burst time. In addition, all processes have zero arrival times. The comparison results shown in Figure 4 show that SIBRR behaves better than the MAXDIFRR[11] algorithm in Case 3 in terms of average waiting times, average turnaround times, and average completion times. Results show that there is a big improvement in the proposed Summation Interquartile Boundaries Round Robin scheduling (SIBRR) algorithm compared to the MAXDIFRR[11] algorithm in terms of average waiting times, average turn-around times, and average completion times.

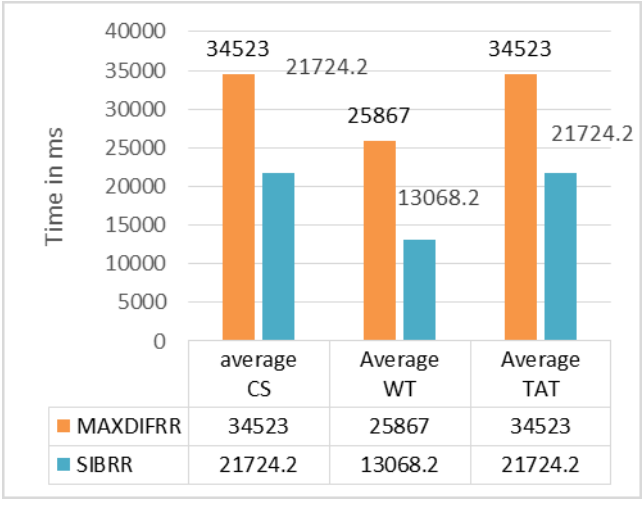

Figure 4 Comparison of the results of MAXDIFRR[11] and proposed SIBRR for the third case.

Case 4: Outlier Burst Time and Different Arrival Time: In In the fourth case, five processes (P1, P2, P3, P4 and P5) are used. Processes have distinct burst times of 95, 26, 43, 000, 60, and 75. In this case, p3 has an

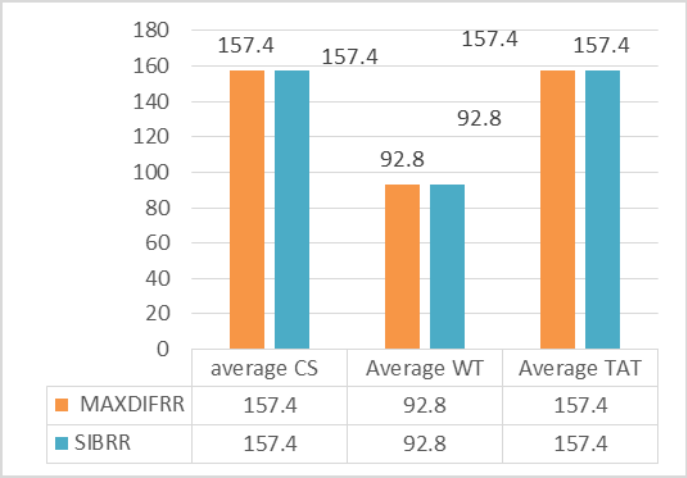

Figure 2 Comparison of the results of MAXDIFRR[11] and proposed SIBRR for the first case.

Case 2: No outlier Burst Time and different arrival time: In the second case, five processes P1, P2, P3, P4 and P5 are used. These processes have distinct burst time 95,26, 43, 60 and 75. In addition, processes arrival times are 0,16, 8, 4, 2. Comparison results shown in Figure 3 show that both algorithms behave similarly on case 2 in terms of average waiting times, average Turn Around Time, and average completion times.

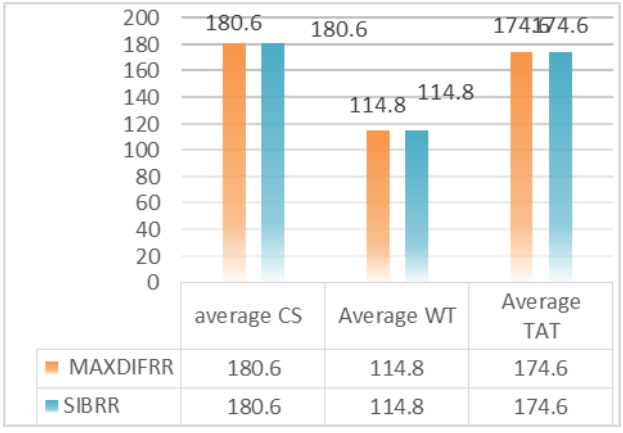

Figure 3 Comparison results of MAXDIFRR[11] and proposed SIBRR for the second case.

III. Experimental setting

Several experiments were conducted to compare the baseline scheduling method MAXDIFRR [11] and the Summation Interquartile Boundaries Round Robin scheduling (SIBRR) algorithm. This work presents the results obtained by baseline and the proposed algorithms on four different cases of processes. To show the success of the proposed methodology and to assure a fair comparison between the proposed algorithm and the existing baseline scheduling method MAXDIFRR [11], this work uses the same tasks, same burst, and arrival time. The four cases of selected lists of processes are shown to show the effectiveness of the proposed algorithm. The proposed algorithm will be compared with the MAXDIFRR [11] algorithm benchmark in terms of average waiting time, average throughput time, and average execution time.

Four cases are used with two different parameters:

Case 1: no outlier and zero arrival time
Case 2: no outlier and different arrival times
Case 3: outlier and zero arrival time
Case 4: outlier and different arrival time

IV. Results and Discussion

First, Case 1P1, P2, P3, P4, and P5 are the processes used in the first case. These processes have distinct burst times of 35, 43, 55, 85, and 105. In the first case, all processes have a zero arrival time. These processes are first arranged in ascending order after applying the analytical model of using burst time and waiting time. Results in Figure 2 show that both algorithms behave similarly in case 1 in terms of average waiting times, average Turn Around Time, and average completion times.

The time and space complexity of the SIBRR algorithm, given n tasks and m virtual machines. First, the complexity of sorting n tasks is O (n*log n). The space complexity of storing n tasks is O(n) and of storing virtual machines information is O(m). Consequently, the overall time complexity is O (n * log n). However, the real performance of any scheduling algorithm depends on all scheduled process's time, which is quite high compared to the algorithm time complexity n*log n which is quite small. The SIBRR algorithm differs from Remaining Shortest Job First Algorithm (RSJF) and First Come, First Served (FCFS) as each process is provided with a fixed time (quantum time) to execute and the quantum time is updated dynamically when new processes arrived. To show if the processes contain processes with outlier burst time (extreme values) in the tails of the distribution.

$$\text{lower fence: } Q1 - 3(Q3 - Q1)$$

$$\text{upper fence: } Q3 + 3.0(Q3 - Q1)$$

Any processes with burst time smaller than lower fence or larger than upper fence are outlier processes, for example, given ten processes that have the following burst times (10,20,200,500,600,5000,2000,10000,600000 and 200000). First, these processes will be sorted in ascending order. The lower and upper quartile will be calculated (Q1= 110 and Q3 3500). After that the quantum time is calculated TQ=Q3+Q1)= 3610. The quantum time will be fixed and only updated when new processes arrived. In this example, after all ten processes finish their execution, the average waiting time = 48169 and average turnaround time = 130002.

مجلة العلوم الهندسية والتقنية
مجلة علمية محكمة
تصدر عن كلية الهندسة وكلية الحاسبات والمعلوماتية – جامعة ذمار
ISSN: 2958-809X EISSN: 2958-8103

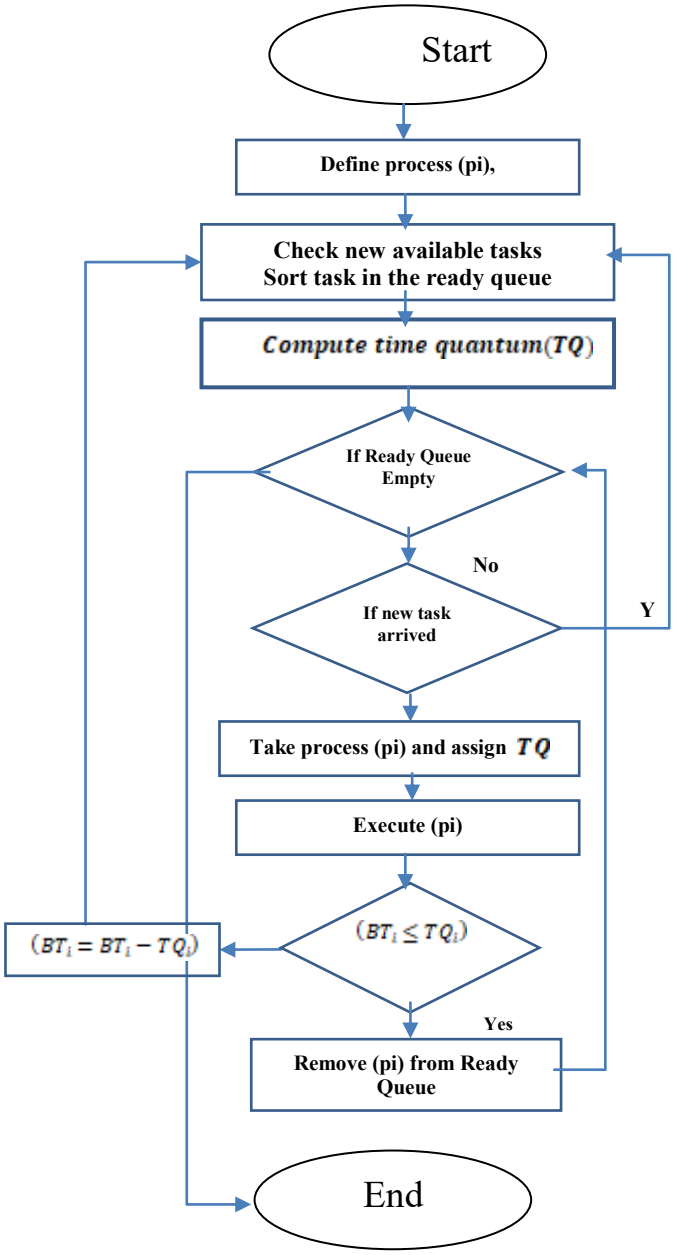

Figure 1. Flow Chart of The SIBRR Algorithm.

المجلد: الأول
العـــدد: الأول
ديسمبر 2022
مجلة العلوم الهندسية والتقنية
مجلة علمية محكمة
تصدر عن كلية الهندسة وكلية الحاسبات والمعلوماتية – جامعة ذمار
ISSN: 2958-809X EISSN: 2958-8103

Algorithm 1: Summation Interquartile Boundaries Round Robin (SIBRR) Scheduling Algorithm

Input: process (pi), $p = (p1, p2, p3, p4 \dots, pn)$

1. Begin
2. Define process (pi), $p = (p1, p2, p3, p4 \dots, pn)$ with their waiting and Burst Time (BT)
3. Initialize: process $Pi\ (1, 2, 3 \dots . . n)$
4. Check new available tasks in the ready queue and set N=number of Processes
5. Sort processes in increasing order (Processes should be sorted in ascending order).
6. While [$N \geq 0$()]
7. For (counter=0; counter<NP; counter++) {

Execute processes in the queue until new

1. processes are arrived
2. If new processes are arrived
3. Add it ready queue; Break; go to 4
4. Else
5. Execute all the tasks
6. Check $pi = (p1, p2, p3 \dots . pn)$
7. For each pi
8. If (pi ($BT \leq TQ$)
9. Remove: Pi from the ready queue
10. End if
11. Else (pi (BT> TQ)) and Pi remain in ready queue
12. End For
13. Repeat until all $p = (p1, p2, p3 \dots . pn)$ are completed
14. End while

at burst time. The primary goal of this paper is to develop an improved Summation Interquartile Boundaries Round Robin (SIBRR) Scheduling Algorithm for cloud computing in order to reduce waiting time.

The SIBRR is composed of seven main steps, as follows:

Step 1: Check new available tasks in the ready queue and set N=number of Tasks.

Step 2: Arrange the submitted tasks in ascending order based on their burst time.

Step 3: Compute lower quartile (Q1) and upper quartile (Q3)

Step 4: Compute time quantum (TQ) based on Equation TQ=(Q3+Q1).

Step 5: Execute all tasks based on their calculated time quantum. If the task finishes its time quantum, pause the task and insert it into the tail of the ready queue.

Step 6: When a new task arrives, do the following:

1. Sort all tasks in the ready queue in ascending order based on their burst time
2. The lower quartile(Q1), upper quartile (Q3) and time quantum (TQ) for all the tasks in the ready queue will be recalculated.

Step 7: Repeat all the steps until all the tasks are finished.

The details of the main steps involved in the proposed approach are shown in Algorithm 1. Moreover, Figure 1 shows a flowchart describing the followed procedures of the proposed approach.

algorithm. They introduce the Enhanced Weighted Round Robin algorithm that dynamically calculates weights based on the servers' waiting time and can migrate jobs to other resources if a server is overloaded. They conclude that they achieved better performance with homogeneous systems.

In [12], a variance-based enhanced round robin algorithm in

Cloud computing is proposed. They calculate and recalculate the time quantum based on the mean, median, and the variance of the available tasks in the queue. Results show that the proposed algorithm greatly improves the average waiting time, turnaround time, context switch, and response time.

In [4], they propose a dynamic probabilistic algorithm that assigns probabilities to each available resource by taking into account the number of servers, mean service time, and its current utilization. The proposed method performs better than the Weighted Round Robin algorithm method. More specifically, it can reduce Mean Response Time by 8.5% and the Utilization of Remote Fast Resources by 25%.

II. Methodology

Round Robin is one of the most well-known scheduling algorithms. Many research efforts have been made in the literature to improve the Round Robin algorithm, with the objective of determining an appropriate quantum time. Outliers and heterogeneous tasks are the main gaps and shortages in existing round Robin algorithm modifications. The optimal time quantum will be large in this scenario, which in its turn increases waiting time, turnaround time, and the number of context switches. Handling outlier and heterogeneous tasks can result in optimal quantum time, less waiting time, and high performance and throughput. This work proposes a task scheduling algorithm by considering interquartile boundaries among tasks

enhanced Round Robin algorithm is proposed using a dynamic time quantum. The dynamic time quantum is calculated based on the median of task burst time and the smallest burst time in the ready queue. It shows improvements in performance by maximum CPU utilization, throughput, and minimizing waiting time, response time, and the number of context switches.

In [11], an enhanced Round Robin algorithm is introduced, where the quantum time is calculated by computing the maximum difference among the differences of adjacent consecutive tasks in the ready queue. Results show improvement in performance and a reduction in the average turnaround time, average waiting time, and the number of context switches.

In [18], an enhanced task scheduling algorithm is introduced, which combines maximum-minimum and round-robin (MMRR) algorithms. They divided tasks into two types: those with long execution times are allocated using maximum-minimum; and tasks with the lowest execution times will be assigned using round-robin.

In [1],a dynamic round-robin heuristic algorithm is introduced by using round-robin algorithm and tuning its time quantum in a dynamic fashion based on the mean of the time quantum. They used the remaining burst time of the task to decide whether to continue executing the task or not during the current round.

In [14], they propose dynamic variable quantum round robin for the task scheduling algorithm in cloud computing. They calculate the mean of the burst time of tasks to use it as an initial value for time quantum, which is dynamically recalculated in every round for each task.

[5] points out the disadvantages of the round-robin load balancing

The remainder of this paper is organized as follows. Section 2 presents the related work, while Section 3 presents the proposed methodology. In Section 4, the setup of an experiment is presented. Section 5 discusses the experimental results. Finally, we conclude our work and discuss future directions of research in Section 6.

I. Related work

Researchers give increasing attention to designing an efficient task scheduling algorithm as a critical issue in cloud computing that enhances system performance by balancing job loads among virtual machines. Several research and several techniques have been reported in the literature to improve performance and resource use based on load balancing and task scheduling in a cloud computing environment. Task scheduling algorithms can be classified into two main types: (1) traditional (static) algorithms and (2) heuristic and meta-heuristic (dynamic) algorithms. [2, 3, 6, 10, 17] provide a comprehensive survey of task scheduling and load balancing algorithms in cloud computing research work, existing algorithms, and issues and challenges associated with existing algorithms.

Many researchers have conducted several studies and proposed various modifications in order to enhance the Round Robin algorithm and to select the optimal time quantum, which plays a vital role in enhancing the algorithm. They discuss these round robin modifications in [10]. Studies have been classified into two categories: RR based on static quantum, and RR based on dynamic quantum, which is further classified into dynamic per round and dynamic per process. In [13], a modification of the round robin algorithm is proposed based on several parameters, such as the execution time and the task order. The improved model shows effectiveness in improving the average waiting time and average response time. In [9], an

When the timer (time slice) of the task is finished and the task is not completed, the scheduler forces preempts the tasks on the processor and adds them to the queue context switch with the next task in the ready queue. Each context switch contains a processor to save the state of the process and implement the next running process.

High Response Time: The response time is the time from the arrival time to the first response being made. Round robin

system response time causes system performance to be degraded.

Very High Turnaround Time: Turnaround time is the total time between an arrival time and its completion. The Round Robin scheduling algorithm always makes for higher turnaround time.

Several research attempts have been conducted to enhance the Round Robin algorithm, focusing on choosing a suitable quantum time. Several modifications of the Round Robin algorithm are proposed [1, 4, 5, 9–14]. One of the weaknesses/drawbacks of the existing modification of Round Robin is that if there are outlier values and heterogeneous tasks, the optimal quantum time of these existing algorithms will be very large and increase waiting time, turnaround time, and the number of context switches Handling round-robin limitations in dealing with outlier values and heterogeneous tasks can result in optimal quantum time, less waiting time, and high performance and throughput. The main contribution of this paper is to propose an enhanced Round Robin Task-Scheduling algorithm focusing on handling traditional RR algorithm disadvantages in dealing with outlier values and heterogeneous tasks, which provides efficient resource utilization and decreases waiting time, turnaround time, and response time.

happen after adding the process to the queue's tail. The procedure may have a CPU burst that is shorter than the allocated time quantum. In this case, the process will be executed and will freely release the CPU. Hence, the scheduler will select the next process in the FIFO queue. In the second case, if the process's CPU burst is longer than the allocated time quantum, the timer will go off. The operating system will be interrupted. A context switch will occur, and the process will be moved to the end of the ready queue. The CPU schedulers will select and schedule the next process in the ready queue.

According to this, the performance of the Round Robin Scheduling Algorithm depends on the value of the time quantum. At one extreme, if the time quantum has an extremely large value, this will result in a shorter response time and the round robin will behave in the same way as FCFS. On the other hand, if the time quantum is exceedingly small, the number of context switches will be large, which will result in lower CPU efficiency.

The main drawbacks of the Round Robin Scheduling Algorithm can be summarized as follows: **Increased Average Waiting Time:** In round robin, waiting time is the time the process spends in the waiting room waiting to be executed. With a large time quantum, completing the process in the Round Robin Scheduling Algorithm will result in a high average waiting time.

Low Throughput: Throughput is the number of processes finished per time unit. Execution of processes in a circular way means more context switches and lower throughput, and this means lower overall performance. On the other hand, if the number of context switches is low, this means high throughput.

Accordingly, designing an efficient task scheduling algorithm that enhances system performance by balancing job loads among virtual machines is a critical issue in cloud computing. Several task scheduling algorithms have been proposed. In a cloud computing environment, task scheduling algorithms can be classified into two main types: (1) traditional algorithms, such as first come, first serve (FCFS), shortest job first (SJF), largest job first (LJF), and round-robin (RR)[1, 4, 5, 9–14], and (2) heuristic and meta-heuristic algorithms, such as Min-Min, Max-Min[15], particle swarm optimization (PSO)[2, 7], Grey Wolf and Whale Optimization, and ant colony optimization (ACO)[3, 16].

Among all these algorithms, Round Robin can be considered the most well-known and used scheduling algorithm. The Round Robin Scheduling Algorithm is based on time-sharing, which means sharing a computing resource among

many users. The Round Robin Scheduling Algorithm provides support for multiprogramming and multitasking as it allows many users to interact at the same time with a single computer. The time-sharing characteristic of the Round Robin Scheduling Algorithm dramatically lowers the cost of providing computing capability. The algorithm behaves like a First Come, First Served (FCFS) Scheduling Algorithm, but RR differs from FCFS because preemption is added to switch between processes. A compact unit of quantum time is usually calculated where quantum time value depends on task burst and arrival times. The CPU scheduler checks processes in the ready queue and assigns the CPU to each process for a time interval of up to the calculated time quantum. A FIFO queue is used as a ready queue for processes. New processes will be added to the tail of the FIFO queue. A timer will be set to the allotted time quantum. One of two things will

مجلة العلوم الهندسية والتقنية
مجلة علمية محكمة
تصدر عن كلية الهندسة وكلية الحاسبات والمعلوماتية – جامعة ذمار
ISSN: 2958-809X EISSN: 2958-8103

Introduction

With the rapid growth of cloud computing and Internet infrastructure, a growing number of businesses are turning to cloud computing to enhance productivity, achieve their goals, and meet consumer demands at a lower cost [1-6]. Cloud computing is one of the most popular and leading technologies in the information technology area, and it has had a significant impact on today's computing. In fact, cloud computing plays an important role in providing and offering three different types of technology services, namely infrastructure, platform, and software services through the internet [7, 8]. First, infrastructure as a service (IaaS), where cloud computing provides infrastructure services such as storage and computation resources. Second, cloud computing provides platform as a service (PaaS), where a customer can build their applications on top of the platform. Third, software as a service (SaaS) is provided by cloud computing, where users can use software in the cloud without having to install it.

Given a huge number of requested tasks in a finite time, task scheduling is required to ensure optimal allocation of resources to improve the overall performance of cloud computing and finally achieve the desired quality of service (QoS). Task scheduling is the most critical issue in cloud computing as it is a primary determinant of other performance factors such as availability and scalability. Task Scheduling helps to take maximum advantage of available resources; and it accelerates networks and resources. In general, high performance in cloud computing can be achieved by allocating workloads among all resources effectively, resulting in minimum waiting time, execution time, maximum throughput, and optimal resource utilization.

مجلة العلوم الهندسية والتقنية
مجلة علمية محكمة
تصدر عن كلية الهندسة وكلية الحاسبات والمعلوماتية – جامعة ذمار
ISSN: 2958-809X EISSN: 2958-8103
المجلد: الأول
العـدد: الأول
ديسمبر 2022

الرباعيات الأدنى والأعلى لخوارزمية Robin المستديرة المحسنة لجدولة المهام الخارجية في الحوسبة السحابية

د.منير عبد الله سعيد هزاع المخلافي* Muneer_hazaa@yahoo.com
نشوان ناجي صالح مصلح الماربي** nashwan.almarbi@gmail.com

ملخص:

تعد الحوسبة السحابية واحدة من أفضل التقنيات الناشئة ذات إمكانات السوق والمؤسسات الضخمة؛ لأنها توفر الوصول عند الطلب على الإنترنت إلى موارد الحوسبة المشتركة على نطاق واسع. وتعد جدولة المهام واحدة من أهم القضايا في الحوسبة السحابية من أجل تعزيز الأداء واستخدام الموارد مع تقليل التكاليف. نظرًا لبساطتها وإنصافها، فإن خوارزمية روبن المستديرة هي خوارزمية جدولة المهام الأكثر مثالية، على الرغم من أنها تعاني من تعقيد الوقت ولا يمكنها التعامل مع المهام الخارجية. وقد تم إدخال عدد من التعديلات على خوارزمية روبن المستديرة لتعزيز تعقيد الوقت. ولضمان التعامل مع تعقيد الوقت والمهام الخارجية، يقدم هذا البحث خوارزمية إرشادية مستديرة محسّنة جديدة من خلال استخدام خوارزمية روبن المستديرة، وتحديث وقتها الكمي بشكل ديناميكي، بناءً على الأرباع السفلية والعليا من الوقت الكمي لجميع المهام في قائمة الانتظار الجاهزة. وقد أظهرت النتائج التجريبية على أربع مجموعات بيانات أن الخوارزمية المقترحة تفوقت بشكل كبير على خوارزميات خط الأساس من حيث متوسط وقت الانتظار ووقت التحول ووقت الاستجابة. وأن الخوارزمية المقترحة تعزز تعقيد الوقت بنسبة 50٪ مع مجموعة البيانات التي تحتوي على مهام عشوائية وخارجية مقارنة بخوارزميات خط الأساس.

الكلمات المفتاحية: الحوسبه السحابيه، جدوله المهام، رويند روبن، كميه الوقت.

* أستاذ مشارك ، كلية الحاسبات والمعلوماتية، جامعة ذمار، اليمن
** طالب ماجستير، كلية الحاسبات والمعلوماتية ، جامعة ذمار، اليمن

المجلد: الأول
العـــدد: الأول
ديسمبر 2022

مجلة العلوم الهندسية والتقنية
مجلة علمية محكمة
تصدر عن كلية الهندسة وكلية الحاسبات والمعلوماتية – جامعة ذمار
ISSN: 2958-809X EISSN: 2958-8103

Lower and Upper Quartiles Enhanced Round Robin Algorithm for Scheduling of Outlier Tasks in Cloud Computing

Dr. Muneer Abdullah Saeed Al-Mekhlafi* muneer_hazaa@yahoo.com

Nashwan Nagi Saleh Al-Marbe** nashwan.almarbi@gmail.com

***Abstract*— Cloud computing is one of the top emerging technologies with huge market and enterprise potential as it provides on-demand, -based access to large-scale shared computing resources. Task scheduling is one of the most important issues in cloud computing in order to enhance performance and resource utilization while minimizing costs. Because of its simplicity and fairness, the round-robin algorithm is the ideal task scheduling algorithm, although it suffers from time complexity and cannot handle outlier tasks. Several modifications of Round Robin have been introduced to enhance time complexity. To ensure sufficient deal with time complexity and outlier tasks, this paper introduces a novel enhanced round-robin heuristic algorithm by utilizing the round-robin algorithm and updating its time quantum dynamically based on the lower and upper quartiles of the time quantum for all the tasks in the ready queue. The experimental results on four datasets showed that the proposed algorithm significantly outperformed baseline algorithms in terms of the average waiting time, turnaround time, and response time. The results show that, when compared to the baseline algorithm in cases 3 and 4, the proposed algorithm enhances the average waiting time's time complexity by 50% with datasets containing random and outlier tasks.**

Keywords: Cloud Computing; Task Scheduling; Round-Robin; Quantum Time

* *Associate Professor, Faculty Of Computer Science and Information System, Thamar University, Yemen*

** *Master Student, Faculty Of Computer Science and Information System, Thamar University, Yemen*