Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Adaptive Scheduling Heuristic Priority Linear Regression (ASH-PLR): A Novel CPU Scheduling Algorithm using Predictive Priority Levels

Don Harl C. Malabanan [1], Zen Lee D. Foryasen [2], Gem A. Balay-odao [3], Dionisio R. Tandingan Jr. [4]
[1, 2, 3] *College of Information Technology and Computer Science, University of the Cordilleras, Baguio City, Philippines.*
[4] *College of Engineering and Architecture, University of the Cordilleras, Baguio City, Philippines.*
Email: [1] dcmalabanan@uc-bcf.edu.ph, [2] zdforyasen@uc-bcf.edu.ph, [3] gabalay-odao@uc-bcf.edu.ph, [4] drtandingan@uc-bcf.edu.ph
**Corresponding Author: Don Harl C. Malabanan*

ARTICLE INFO

Received: 15 Dec 2024

Revised: 13 Feb 2025

Accepted: 24 Feb 2025

ABSTRACT

Inadequate implementation of parameters such as priority levels can be seen in many common and contemporary scheduling algorithms which can lead to starvation. With the rise of industry 4.0, the advancement of computers and operating systems require more efficient and optimized scheduling algorithms to assess big data. This study aims to explore and develop a novel and heuristic approach to scheduling algorithms by incorporating predictive models in machine learning, more specifically the linear regression model to predict and allocate the most efficient priority level to each process upon execution. The newly developed ASH-PLR algorithm was tested against common and contemporary scheduling algorithms such as the FCFS, AMRR, and the MMRRA in terms of their Average Turnaround Time, Average Waiting Time, and Context Switches. The results indicate that the ASH-PLR is the superior scheduling algorithm when it comes to processes that have shorter burst times and extensively outperforms the FCFS and MMRRA in terms of Average Turnaround Time and Average Waiting Time. ASH-PLR displays the ability of predictive models to be integrated in future algorithms for better optimizations in upcoming new technology.

Keywords: CPU Utilization, Heuristic, Linear Regression, Machine Learning, Novel, Optimization, Priority Scheduling Algorithm.

INTRODUCTION

Scheduling algorithms are algorithms that give the computer instructions on allocating processes to a Central Processing Unit (CPU). Assigning processes to the CPU is a done by the Operating System (OS) which is important for scenarios with concurrent tasks [1]. By scheduling tasks properly and effectively, scheduling algorithms optimize system performance metrics such as the response time of processes, CPU utilization and throughput, and efficiency of the whole program in execution. Scheduling algorithms are important in real-time systems for meeting strict timing requirements and deadlines. Specific algorithms can also adapt to changing workload conditions, dynamically adjusting resource allocation strategies to meet evolving system demands and performance objectives.

With the rapid development of a variety of network services and an increase of internet users over the years, networks have become increasingly busy. This rapidly growing traffic congests the backbone of networks [2]. Due to this, there have been many attempts for inventions of innovative and optimized scheduling algorithms. The Priority Scheduling Algorithm (PSA) is favored among several other CPU scheduling algorithms. However, it can result in a starvation issue where lower-priority processes are deprived of CPU usage because higher-priority processes continuously utilize the CPU. This study carries out relevant research that addresses these issues by optimizing the PSA by integrating linear techniques in machine learning in the algorithm.

The integration of machine learning techniques into various domains has revolutionized problem-solving approaches, particularly in the optimization of algorithms. Among these techniques, multiple linear regression stands

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

out as a robust method for analyzing the relationships between multiple variables and optimizing algorithm performance. This approach leverages the principles of linear regression to model the linear relationship between a dependent variable and two or more independent variables, thereby enabling the prediction of outcomes based on input parameters.

The reusability of multiple linear regression lies in its ability to uncover intricate relationships within datasets comprising numerous variables, allowing researchers to identify the most influential factors affecting algorithmic performance. By quantifying the impact of each variable on the outcome, multiple linear regression facilitates the fine tuning of algorithms to achieve optimal results. Moreover, the interpretability of regression coefficients empowers researchers to glean insights into the underlying mechanisms driving algorithm performance, thereby informing subsequent optimization strategies.

Multiple linear regression serves as a tool that's possibly capable of addressing multiple different challenges across various domains. Whether optimizing computational efficiency, enhancing predictive accuracy, or minimizing error rates, this methodology offers a systematic framework for evaluating algorithm performance and identifying areas for improvement.

The main objective of this research is to develop a new priority scheduling algorithm for CPU optimization using a novel machine learning approach. Specifically, the focus of this research is to achieve the following objectives: to determine the relevant performance metrics of a well-performing CPU scheduling algorithm; to apply machine learning to a CPU scheduling algorithm; and to assess the optimization of the developed novel algorithm against common and contemporary scheduling algorithms.

METHODOLOGY

2.1) Literature of Review:

Scheduling stands as an important part in the management of threads or processes assigned to the CPU, constituting a core function handled by the operating system. Its significance becomes noticeable in scenarios where numerous tasks operate concurrently, necessitating efficient resource allocation. This orchestration of tasks is fundamental for ensuring optimal utilization of system resources and maintaining overall system performance. Through the allocation of CPU time to processes, scheduling algorithms maximize the system's efficiency and throughput. One of the key parameters used to evaluate the effectiveness of scheduling algorithms is CPU utilization. This metric measures the extent to which the CPU remains occupied with executing processes, with higher utilization indicative of more efficient resource utilization [3].

Programming languages enable the creation of programs that outline a sequence of tasks. These tasks, collectively referred to as a process, are managed by the operating system, which is important for efficient execution. The operating system has tasks such allocating resources, facilitating communication between processes, managing data structures, and overseeing process execution. Central to this is CPU scheduling, where the operating system employs algorithms to decide which process will utilize the CPU and progress with its execution [3].

Selecting or comparing scheduling algorithms poses a significant challenge due to the diverse nature of operating systems, which may be designed for specific purposes or environments like batch processing, general-purpose computing, or real-time systems. Consequently, there isn't a single standard for evaluating scheduling quality. Nonetheless, it's understood that the main goal of short-term scheduling is to enhance a specific aspect of the system when allocating the CPU to a process. Stallings identifies several crucial criteria for short-term planning, acknowledging the difficulty of optimizing all of them concurrently [3].

The study on CPU scheduling comparisons checks the possible metrics by first stimulating the existing scheduling algorithms by creating a specific program. The program simulates the scheduling of generated processes using each algorithm outlined in the study materials. Moreover, traditional Round Robin and Shortest Job First algorithms are included for comparative purposes alongside the new algorithms. After a process finishes, the program gathers a dataset that can be used to assess the algorithm's efficiency [4].

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The literature introduces a type of machine learning model for optimizations of existing and new systems and programs that can be beneficial to the community and the body of knowledge in the IT and computer science industry. Though there are multiple different machine learning techniques, the literature reviews differ from the data set and system architecture on deciding the machine learning technique [5].

Multiple linear regression offers the capability to manage confounding variables and gauge their influence on the outcome variable. It permits researchers to analyze the effects of numerous independent variables on a single dependent variable concurrently, facilitating thorough and all-encompassing evaluations. This method demonstrates resilience to outliers and missing data, constructing a regression line primarily on the bulk of available data points. Additionally, it exhibits minimal susceptibility to the inclusion of non-informative variables, rendering it apt for exploratory data analysis. Multiple linear regression stands out as one of the prevailing statistical techniques utilized across diverse fields [5].

There have been different ways on how researchers have optimized CPU scheduling algorithms. Some of which optimize specific algorithms such as the RR and the FCFS. notable contribution to the field comes in the form of a hierarchical approach to task prioritization [3]. Some have even integrated multiple individual scheduling algorithm with an already existing scheduling algorithm to form a faster and more efficient algorithm. Some of these optimizations are applicable but some are theoretical in today's technology. a study [6] of outlines using the existing algorithms Average Max Round Robin (AMRR), A New Round Robin Algorithm (ANRR), and Modified Median Round Robin Algorithm (MMRRA) to develop its proposed algorithm. The study explains these in detail. AMRR is where CPU executes processes based on their arrival times, with processes scheduled for execution from the ready queue (RQ), indicating that they are already present in the queue. Drawing insights from various literature sources has provided guidance for addressing other significant challenges. Software specifications required for minimizing context switches are suggested for future research endeavor [5].

Before assessing performance and efficiency, it's important to understand the key characteristics of an ideal scheduling algorithm. [6]. Researchers define metrics differently but usually use ATAT and AWT as a metric for success on if the new optimized algorithm can have a shorter score on ATAT and AWT. The optimization not only include new ways of improvements with new strategies but also rehashing existing systems into contemporary algorithms which leads to a more efficient CPU scheduling algorithm. However, there is little literature showing the usage of machine learning as an optimization algorithm for CPU scheduling [4].

2.2) Methods:

This study uses an experimental research design. This design was chosen for the involvement of independent and dependent variables for testing, as well as measuring the effect factor of the priority level variable in a CPU scheduling algorithm.

The experimental design tackles the research questions by firstly preprocessing the data set that consists of the variables with the most effect on optimization of CPU scheduling. This is followed by the development of the new scheduling algorithm with the addition of machine learning using a big data set. The design then compares the performance metrics of newly the developed algorithm versus common and contemporary scheduling algorithms to determine its effectiveness and efficiency in different test cases.

Quantitative research method was used as it focuses on the collection of numerical data and uses such data to analyze efficiency of the newly developed scheduling algorithm. This method helps answer the research questions by using existing data for content analysis in the form of data sets as well as multiple test cases to introduce diversity to the experimental analysis for efficiency based on the performance metrics of scheduling algorithms.

The development of the ASH-PLR algorithm was based on a linear regression model that uses existing variables that creates a new PL given to each process based on a mix of features such as process characteristics, arrival times, and potential dynamic adjustments like aging mechanisms. Processes with a higher burst time might have higher priority levels while shorter tasks in turn have lower priority levels.

A total of 100 test cases were generated using Python to assess the efficiency of the ASH-PLR algorithm. These test cases were generated with five processes each with their own AT and BT. The test cases were distributed in specific

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

groups. 25 test cases have low AT and low BT, 25 test cases have low AT and high BT, 25 test cases have high AT and low BT, and the final group has 25 test cases with high AT and high BT. The performance metrics identified in the document analysis and interview was used to gauge the optimization of the ASH PLR algorithm based on the 100 test cases where a random set of test cases was used to compare the performance of the ASH-PLR algorithm versus common and contemporary scheduling algorithms.

Predictive Analysis was utilized to forecast the optimal PL for each process. This is achieved by modeling relationships between the features that are identified to possibly change the outcome of a scheduling algorithm while using a linear regression approach. The underlying hypothesis is that PL is influenced by a combination of features that are typically relevant to OS scheduling algorithms. Linear regression is applied to the existing data set using the JASP software which yields a formula that can be used to predict the new PL. Any feature within the formula that shows a p-value that is higher than 0.05 is removed as it does not show any significant impact within the outcome.

This new PL is applied to the test cases' existing values. A comparison table will then show outcomes of the newly predicted set of PL compared to common and contemporary scheduling algorithms.

The criteria for an excelling scheduling algorithm for an OS depends on the performance metrics of the computer itself. The researchers considered multiple literature citations on possible criteria metrics such as:

- Waiting Time (WT): Refers to how long a process stays in the ready queue before being executed by the CPU. It tracks the delay a process experiences while waiting to be processed.
- Response Time (RT): Measures the time it takes for a system to start responding to a request after it's submitted. It reflects how quickly the system reacts to user inputs.
- Completion Time (CT): Indicates the total time from when a task is submitted to when it is fully processed and completed. It reflects the system's efficiency in handling tasks.
- CPU utilization: Represents the percentage of time the CPU is actively working on tasks compared to the total available time, highlighting how efficiently the CPU is being used.
- Throughput (T): Shows the number of tasks or processes a system completes successfully within a given timeframe, indicating the system's capacity to handle workloads.
- Turnaround Time (TAT): Captures the total time it takes to complete a task, including both the waiting and processing times, providing a measure of task execution efficiency.

The generated data set consists of processes with its own BT, AT, and heuristic PL. The heuristic PL is calculated using the new formula called Adaptive Scheduling Heuristic (ASH) formula:

$$\min\left(10, \max\left(1, int\left(\frac{priority}{max\_bt + \max\_at}\right) * 10\right)\right) \quad (1)$$

where:

$$\text{priority} = \text{burst\_time\_factor} + \text{arrival\_time\_factor}$$

$$\text{burst\_time\_factor} = \text{max\_burst\_time} - \text{burst\_time}$$

$$\text{arrival\_time\_factor} = \text{max\_arrival\_time} - \text{arrival\_time}$$

The burst time factor and arrival time factor applied the ASH formula so that processes with shorter burst times receive higher priority. The idea is that shorter burst times are generally more desirable for quick processing, resulting in a higher priority. Processes that arrive earlier get higher priority. This encourages processes that are ready to run sooner to be prioritized. Linear regression was performed using the generated BT and AT with the inclusion of the heuristic PL. The coefficients with a p-value of less than 0.05 are included in the formula of the linear regression model shown as:

$$yi = \beta 0 + \beta 1xi1 + \beta 2xi2 + \cdots + \beta pxip \quad (2)$$

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Where yi is the dependent variable, xi is the explanatory variable, β0 is the y-intercept, and βp is the slop coefficients for each explanatory variable. The ASH-PLR algorithm uses a heuristic approach and adds machine learning aspects, specifically linear regression as part of its scheduling instructions.

2.3) ASH-PLR Algorithm and Flowchart:

The algorithm follows strict steps assuming that all processes have their corresponding AT and BT. Let R be the ready queue. Let L be the linear regression queue.

1) When a process arrives, its heuristic PL is calculated based on its AT and BT using the ASH formula.

2) All process with a heuristic PL goes to L

3) If all processes have arrived at L, linear regression is applied and yields the linear regression formula F.

4) F is applied to all processes in L and receives a new PL.

5) Processes are added to R using PSA incorporating the new PL.

The ASH-PLR flowchart as seen in figure 1(a) and figure 1(b) illustrates the graphical method of the algorithm with the inclusion of the PSA.

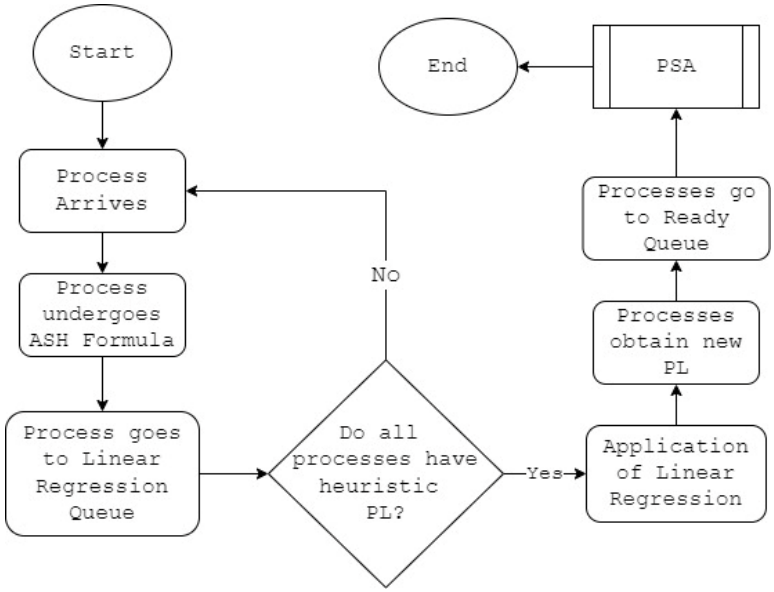

Figure 1(a) Flowchart of the ASH-PLR algorithm.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

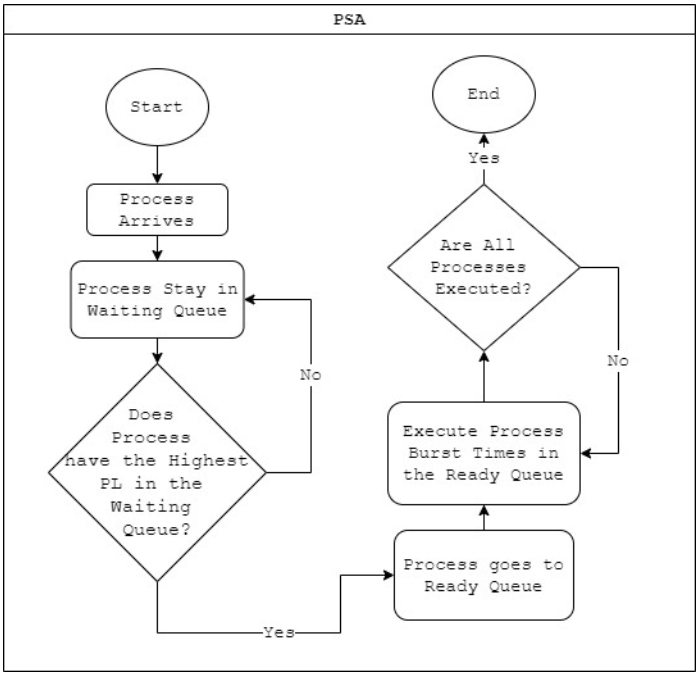

Figure 1(b) Flowchart of the PSA Exploration.

2.4) Illustration of the ASH-PLR algorithm:

The goal of ASH-PLR is to improve process scheduling efficiency by assigning PLs in a way that minimizes the AWT. Table 1 shows processes with its corresponding AT and BT for a sample test case using the ASH-PLR algorithm.

Table 1. ASH-PLR Sample Processes

PID	**Parameters**					
	BT	***AT***	***ASH PL***	***ASH-PLR PL***	***TAT***	***WT***
P1	5	11	4	4.362	13	8
P2	10	17	1	0.797	19	9
P3	3	9	6	5.658	10	7
P4	8	8	4	4.368	8	0
P5	2	19	3	2.731	7	5
P6	5	11	4	4.362	13	8
Average					11.4	5.8

The ASH-PLR using the sample test case follows the algorithmic procedure after the ASH PL and ASH-PLR PL are calculated. Since no processes arrive at time 0, the queue is empty until a process arrives. Process 4 arrives at time 8 thus entering the ready queue and executing 8 units of time and finishes at time 16. By this time, process 3 and process 1 arrive respectively. Since process 3 has a higher PL of 5.658, it goes in the queue first and executes 3-time units finishing at time 19. During this time, process 1, process 2, and process 5 are in the waiting queue. Process 1 is executed first as it as a higher PL with 4.362 finishing at time 24. Process 5 is executed next with its PL of 2.731 finishing at time 26. Finally process 2 is executed at last having the lowest PL of 0.797 finishing at time 36 ending the program. The ATAT computed is 11.4 and the AWT computed is 5.8.

RESULT

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

ASH-PLR was benchmarked against FCFS and two optimized RR variations, AMRR and MMRRA. FCFS was chosen to highlight potential trade-offs in showing the justification for complexity in ASH-PLR over simpler scheduling algorithms. AMRR and MMRRA were chosen to check the comparison of performance against new innovative computations. This comparison ensures ASH-PLR is evaluated alongside both traditional and contemporary counterparts.

All test cases in this study are based on the assumptions defined in the scope and delimitations. The test cases are organized and grouped according to the duration of BT and the intervals between AT namely group A, B, C, and D. The ATAT and AWT of the FCFS, AMRR, and MMRRA algorithms were compared to those of the ASH-PLR algorithm. The outputs of each test case can be seen through tables 2 through 9.

Table 2. Group A ATAT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	15.8	15.8	15.2	18.2
2	10.4	11.6	5.8	13.4
3	10.8	13.2	9.2	13.6
…	…	…	…	…
24	12.6	15.4	7.6	16.4
25	10	12.8	8.4	13.4

Table 3. Group A AWT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	9	9	13.4	12.8
2	5.2	6.4	3.8	10.6
3	5.2	7.6	6	9.4
…	…	…	…	…
24	7.6	10.4	5.4	13.4
25	5.2	8	6.6	10

Group A generated 25 test cases with processes having short BT and short AT. Table 2 and Table 3 show the comparison of the ATAT and AWT of all stated scheduling algorithms.

Table 4. Group B ATAT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	63.4	75.2	35.6	92.8
2	32	43.4	9	53.4
3	63	84.2	47.2	92.8
…	…	…	…	…
24	25.4	27.4	20.8	27.4
25	62.4	70	61.2	75.4

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 5. Group B AWT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	37.8	49.6	25.6	79.2
2	16.8	28.2	5.4	47.4
3	35	56.2	32.2	72.2
…	…	…	…	…
24	10.6	12.6	15.6	20.2
25	38.2	45.8	47.6	57

Group B generated 25 test cases with processes having long BT but short AT. Table 4 and Table 5 show the comparison of the ATAT and AWT of all stated scheduling algorithms.

Table 6. Group C ATAT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	8.2	8.2	5.8	8.6
2	3.8	3.8	2.8	8
3	6.4	6.4	5	9.4
…	…	…	…	…
24	9	9	10	11
25	10	9.4	6.6	17.8

Table 7. Group C AWT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	3.2	3.2	3	4.8
2	0.2	0.2	1	5.4
3	2	2	3.6	7.4
…	…	…	…	…
24	2.4	2.4	6.2	6
25	3.4	2.8	1.6	12.8

Group C generated 25 test cases with processes having short BT but long AT. Table 6 and Table 7 show the comparison of the ATAT and AWT of all stated scheduling algorithms.

Table 8. Group D ATAT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	55.8	52.8	33	52.4
2	35.8	46.8	17	50.8
3	80.8	99.4	54.2	126.8

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

…	…	…	…	…
24	31.6	31.6	23.2	35
25	38.4	44.6	36.2	44.6

Table 9. Group D AWT Test Case Results

Test Case	ASH- PLR	FCFS	AMRR	MMRRA
1	32.8	29.8	27.8	35.8
2	15.4	26.4	11.2	42
3	44.8	63.4	42.4	108.8
…	…	…	…	…
24	31.6	31.6	23.2	35
25	38.4	44.6	36.2	44.6

Group D generated 25 test cases with processes having long BT and long AT. Table 8 and Table 9 show the comparison of the ATAT and AWT of all stated scheduling algorithms.

The ASH-PLR algorithm performed relatively well compared to FCFS and MMRRA with AMRR being the only scheduling algorithm that can sometime have a short ATAT and AWT. Figure 3 shows the graphical representation of the comparative analysis of the scheduling algorithms in regards to their resulting ATAT and AWT for the test cases in group A.

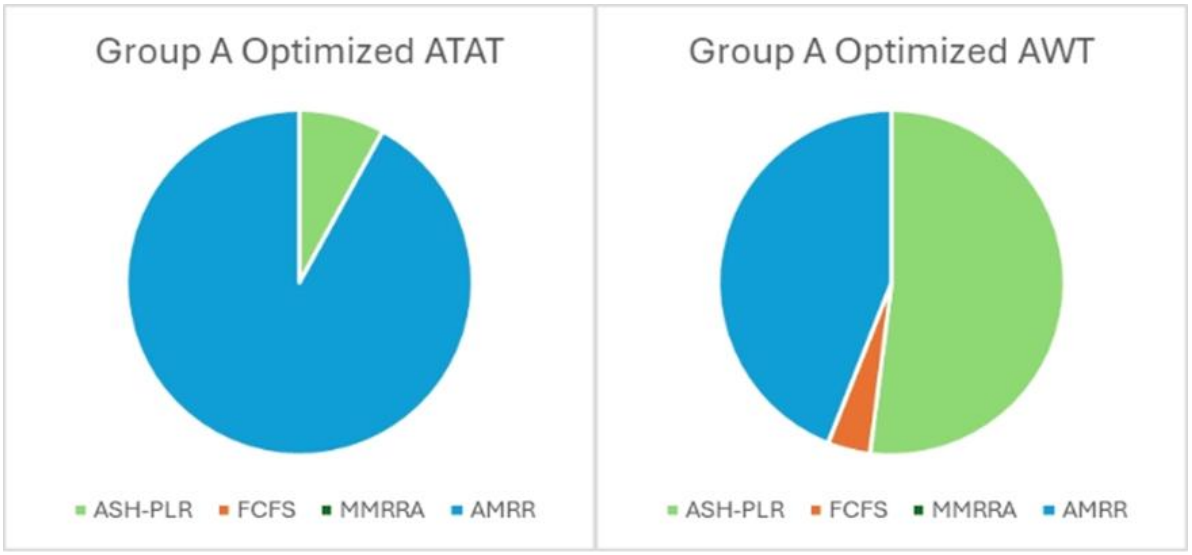

Figure 2. ATAT and AWT Comparison Pie Chart for Group A.

The ASH-PLR algorithm yielded the most optimized ATAT only in 2 test cases compared to AMRR's 23. However, ASH-PLR showed a better AWT optimization for 13 out of the 25 test cases.

Figure 4 shows the graphical representation of the comparative analysis of the scheduling algorithms in regards to their resulting ATAT and AWT for the test cases in group B.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

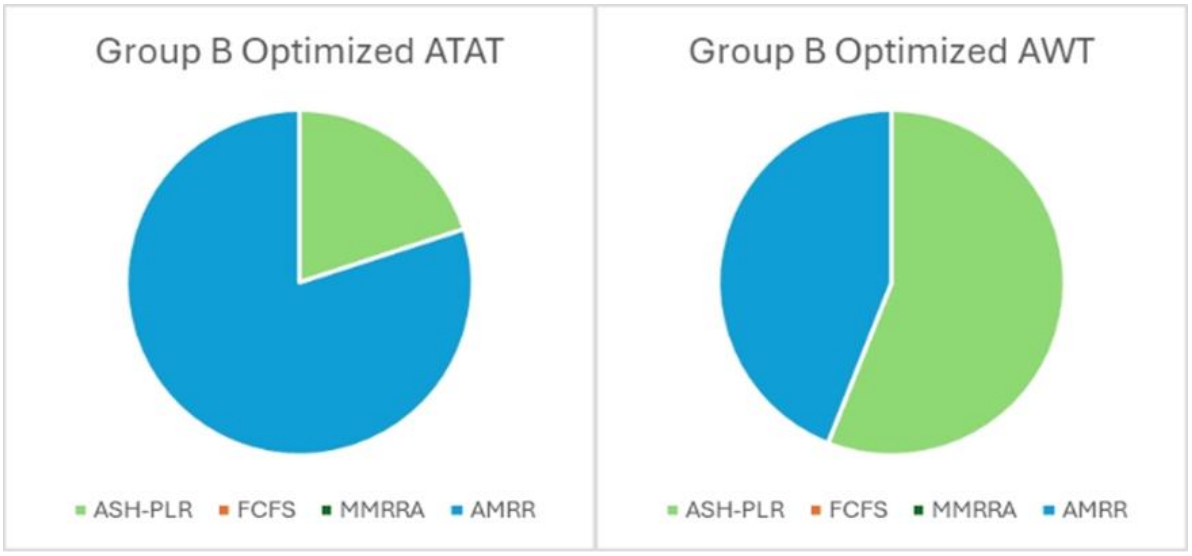

Figure 3. ATAT and AWT Comparison Pie Chart for Group B.

In group B, the ASH-PLR algorithm yielded the most optimized ATAT in 5 test cases compared to AMRR's 20. ASH-PLR performed a better AWT optimization result for 14 out of the 25 test cases.

Figure 5 shows the graphical representation of the comparative analysis of the scheduling algorithms in regards to their resulting ATAT and AWT for the test cases in group C.

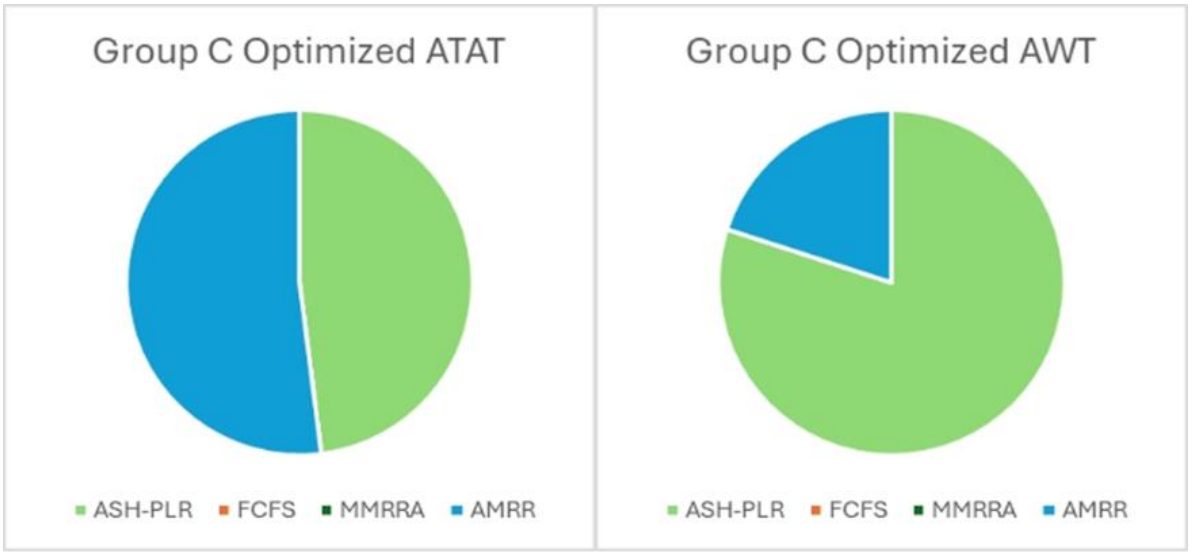

Figure 4. ATAT and AWT Comparison Pie Chart for Group C.

For the test cases where there are shorter BT and longer AT, the ASH-PLR algorithm performed almost equally with AMRR showing the best optimization in 12 out of the 25 test cases. the ASH-PLR algorithm performed significantly better against the other scheduling algorithms when comparing the resulting AWT with 20 out of the 25 test cases being more optimized.

Figure 6 shows the graphical representation of the comparative analysis of the scheduling algorithms in regards to their resulting ATAT and AWT for the test cases in group D.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

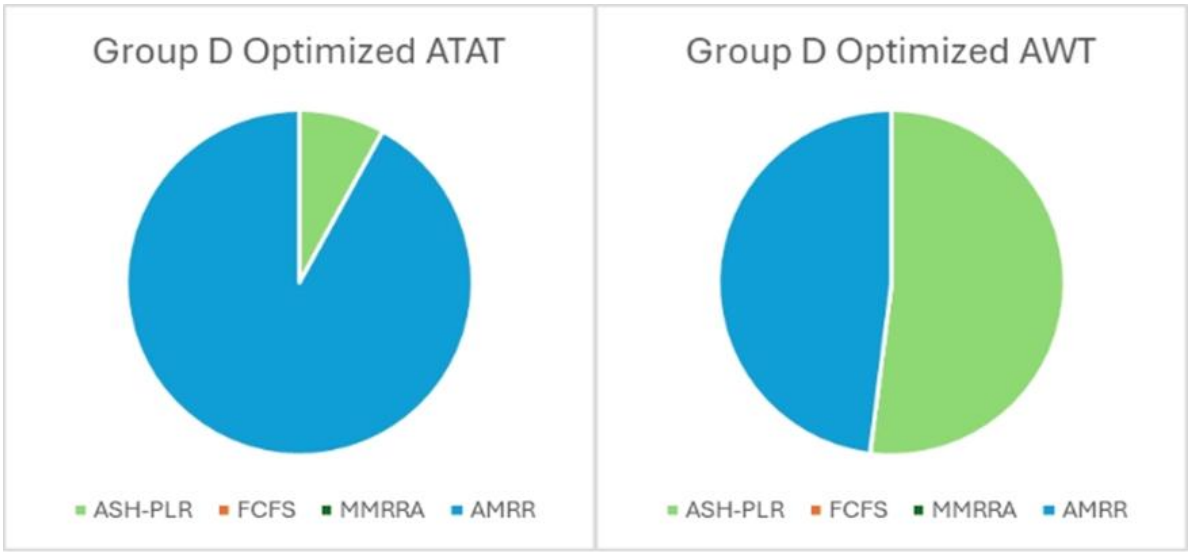

Figure 5. ATAT and AWT Comparison Pie Chart for Group D.

The ASH-PLR algorithm underperformed in group D where both the AT and BT of each process is considered to be long for the ATAT optimization. For the AWT optimization, the ASH-PLR algorithm performed the best with a majority of 13 optimized test cases out of the 25 test cases.

Another metric tested for the 100 test cases was the context switch of each algorithm in each group. ASH-PLR and FCFS yielded the most optimized context switches due to the non-preemptive nature of the algorithms.

The novel approach in heuristic ASH formula and linear regression model of the ASH-PLR algorithm shows a significant optimization factor compared to FCFS, AMRR, and MMRRA as seen in all test cases. Based on the test cases, the ASH-PLR performs better when processes to be executed arrive with a shorter BT.

CONCLUSION

The relevant performance metrics of a well-performing CPU scheduling algorithm are the average turnaround time, average waiting time, and context switches. These metrics measure the specific aspects of performance and efficiency such as the responsiveness and fairness among the arriving and executing processes.

Linear regression was applied to a scheduling algorithm as a predictive agent for calculating the new priority levels of each process using historical data or processes that enter the ready queue such as their arrival time and their burst time. This machine learning technique optimized the heuristic priority level output of the ASH formula by giving each process a more accurate priority level.

The ASH-PLR algorithm can compete with common and contemporary scheduling algorithms such as the FCFS, MMRRA, and AMRR in terms of the average turnaround time, average waiting time, and context switches of processes. ASH-PLR is more efficient for process fairness compared to FCFS and MMRRA and is equally as efficient against AMRR. The algorithm also seems to be more efficient in specific cases where the processes have shorter burst times.

Acknowledgment:

We thank Don Harl Malabanan, Gem Balay-Odao, and Paul Terese Catala for their contributions to this work. Special thanks to Dionisio Tandingan Jr. for their assistance and the University of the Cordilleras for financial support.

REFERENCE

Research Article

[1] Moseley, B., & Vardi, S. (2022). The efficiency-fairness balance of Round Robin scheduling. Operations Research Letters, 50(1), 20–27. https://doi.org/10.1016/j.orl.2021.11.008
[2] Yang, L., Pan, C., Zhang, E., & Liu, H. (2012). A new class of priority based weighted fair scheduling algorithm. Physics Procedia, 33, 942 948. https://doi.org/10.1016/j.phpro.2012.05.158
[3] Chandiramani, K., Verma, R., & Sivagami, M. (2019). A modified priority preemptive algorithm for CPU scheduling. Procedia Computer Science, 165, 363–369. https://doi.org/10.1016/j.procs.2020.01.037
[4] González-RodrıGuez, M. L., Otero-Cerdeira, L., González-Rufino, E., & Rodríguez-Martínez, F. J. (2024). Study and evaluation of CPU scheduling algorithms. Heliyon, https://doi.org/10.1016/j.heliyon.2024.e29959 10(9), e29959.
[5] Huang, Y., Xu, W., Sukjairungwattana, P., & Yu, Z. (2024). Learners' continuance intention in multimodal language learning education: An innovative multiple linear regression model. Heliyon, e28104. https://doi.org/10.1016/j.heliyon.2024.e28104
[6] Sakshi, Sharma, C., Sharma, S., Kautish, S., Alsallami, S. a. M., Khalil, E. M., & Mohamed, A. W. (2022). A new median-average round Robin scheduling algorithm: An optimal approach for reducing turnaround and waiting time. Alexandria Engineering Journal /Alexandria Engineering Journal, 61(12), https://doi.org/10.1016/j.aej.2022.04.006
[7] Malabanan,D. H.,Valdez,M. M.,&Tandingan,D.Jr.(2024).POT-AVL: A novel CPU scheduling algorithm based on AVL trees and postorder traversal.InternationalJournalofComputingSciencesResearch,8,3298-3310. https://doi.org/10.25147/ijcsr.2017.001.1.211