2025 Volume: 01 Issue: 06 www.ijrpa.com Review Article

International Journal Research Publication Analysis

Page: 254-256

STUDY AND COMPARISON OF CPU SCHEDULING ALGORITHMS

Kritika*

New Town, Kolkata.

Article Received: 10 September 2025
Article Revised: 30 September 2025
Published on: 20 October 2025

***Corresponding Author: Kritika**
New Town, Kolkata.

ABSTRACT

CPU scheduling is a critical concept in operating systems. Various scheduling algorithms are implemented to utilize the CPU efficiently and effectively. These algorithms can be broadly categorized into Primitive and Non-Primitive types. This study evaluates different algorithms by calculating turnaround time, waiting time, and response time to assess CPU utilization. Furthermore, it highlights the advantages and disadvantages of each algorithm and provides guidelines on how to optimize system performance.

KEYWORDS: CPU Scheduling, FCFS, SJN, Priority Scheduling, Round Robin, Operating Systems.

1. INTRODUCTION

An operating system (OS) acts as an interface between the user and computer hardware, providing a user-friendly environment for proper system operation. The OS performs various functions, including booting, process management, memory management, disk management, and overall control of system functionality (Silberschatz, Galvin, & Gagne, 2022).

In a uniprocessor system, where only one CPU is considered, resource sharing becomes challenging. With multiple processes waiting for CPU access, multiprogramming environments are necessary to allocate resources effectively and maximize CPU utilization.

CPU scheduling algorithms can be classified into preemptive and non-preemptive types, including: First Come First Serve (FCFS), Shortest Job First (SJF / SJN), Round Robin (RR), and Priority Scheduling.

2. Comparison of CPU Scheduling Algorithms.

Algorithm	Basic Principle	Advantages	Disadvantages	Best Suitable For
FCFS	Executes processes in order of arrival	Simple to understand and design	Shorter jobs may starve	Batch operating systems
SJN / SJF	Executes process with smallest burst time first	Low average waiting time	Requires knowledge of burst time	Batch operating systems
Priority Scheduling	Executes highest priority process first	Handles critical tasks	Low-priority processes may starve	Real-time operating systems
Round Robin (RR)	Executes processes in fixed time quantum	Fair allocation; prevents starvation	Performance depends on quantum size	Interactive operating systems
Multilevel Queue Scheduling	Multiple queues based on priority	Efficient in multiprocessing	Complex to implement	Systems with background/foreground tasks

3. SUMMARY

From the comparison above: FCFS is simple but inefficient for systems with varied process lengths. SJF provides optimal performance for short jobs but requires knowledge of burst time. Priority Scheduling ensures critical tasks get CPU time but may cause starvation. Round Robin offers fairness and responsiveness for time-sharing systems. Multilevel and Feedback Queue Scheduling combine multiple strategies for better adaptability and performance. CPU scheduling is essential in operating system design as it determines the execution order of processes. Efficient scheduling reduces waiting and turnaround time, thereby improving overall system performance.

4. CONCLUSION

To enhance system performance, CPU utilization must be optimized. Lower waiting and turnaround times directly improve system efficiency. No single scheduling algorithm is best for all environments, as the optimal choice depends on specific computing contexts. Hybrid or adaptive strategies are often necessary for achieving balanced performance in diverse systems.

REFERENCES

1. Dash, A. R., Sahu, S. K., & Samantra, S. K. (2016). An optimized round robin CPU scheduling algorithm with dynamic time quantum. arXiv. https://arxiv.org/abs/1605.00362

2. Dash, A. R., Sahu, S. K., Samantra, S. K., & Sabat, S. (2015). Characteristic specific prioritized dynamic average burst round robin scheduling for uniprocessor and multiprocessor environment. arXiv. https://arxiv.org/abs/1511.02498
3. Hajjar, O., & Mekhallalati, M. (2024). Performance assessment of CPU scheduling algorithms: A scenario-based approach with FCFS, RR, and SJF. Journal of Computer Science. https://thescipub.com/abstract/jcssp.2024.972.985
4. Kahu, S. Y. (2025). KernelOracle: Predicting the Linux scheduler's next move with deep learning. arXiv. https://arxiv.org/abs/2505.15213
5. Manyam, R. R., Ganesh, V. V. D. S. S., Lakshmi, S., & Sireesha, Y. (2019). Comparative analysis of CPU scheduling algorithms and their optimal solutions. Proceedings of the 3rd International Conference on Computing Methodologies and Communication (ICCMC). https://www.researchgate.net/publication/335497779
6. Nath, P., Dey, S., Nandi, S., & Nath, S. (2023). An optimized CPU scheduling algorithm with adaptive time quantum approach. Journal of Operating Systems Development & Trends. https://journals.stmjournals.com/article/article=2023/view=90339
7. Samanta, B., & Biswas, A. (2025). Comparative analysis of different CPU scheduling algorithms with respect to average waiting time and average turnaround time. MCSRU Conference Proceedings. https://stm.bookpi.org/MCSRU-V5/article/view/18311
8. Yu, K., Song, K., Liu, X., Liang, L., & Wang, X. (2025). Research on optimization of task scheduling algorithm for embedded real-time systems. Clausius Press. https://www.clausiuspress.com/article/15772.html