Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

COMPARATIVE ANALYSIS OF SCHEDULING ALGORITHMS IN HARD AND SOFT REAL-TIME SYSTEMS

Ergashev Otabek Mirzapulatovich
Doctor of Philosophy (PhD) Independent Researcher of the National Pedagogical University of Uzbekistan named after Nizami

Abstract
Real-time systems are vital in fields like aerospace, automotive, and healthcare, where timely and predictable task execution is crucial. These systems are classified as hard or soft real-time based on the severity of deadline misses. This paper compares three key scheduling algorithms—Rate Monotonic Scheduling (RMS), Earliest Deadline First (EDF), and Least Laxity First (LLF)—to evaluate their performance under both types of real-time constraints. Through simulation and metrics like deadline miss ratio, CPU utilization, and response time, the study finds that EDF excels in soft real-time settings with dynamic workloads, RMS ensures predictability in hard real-time systems, and LLF, despite its theoretical optimality, is less practical due to high complexity. These findings guide developers in choosing suitable scheduling strategies based on application needs.

Introduction

Real-time systems (RTS) have become fundamental components in a wide range of critical and non-critical applications, from pacemakers and automotive braking systems to video streaming platforms and robotic automation. These systems are defined not only by the correctness of their computational outputs, but also by the timing of their execution. Unlike general-purpose computing, a real-time system must guarantee that operations are completed within specific timing constraints. This distinctive requirement makes task scheduling a central design issue in real-time computing. Real-time systems are broadly categorized into two types: hard real-time systems and soft real-time systems. In hard real-time systems, missing a task deadline is considered a system failure, potentially resulting in catastrophic outcomes. Examples include aircraft control, industrial automation, and medical monitoring systems. Conversely, soft real-time systems allow occasional deadline

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

violations, with the understanding that overall system performance may degrade gracefully—examples include multimedia playback or online gaming systems. The nature of these constraints significantly influences the choice and implementation of scheduling algorithms. Task scheduling in real-time systems determines the order and timing of task executions to ensure deadlines are met. Three classical and widely studied algorithms are Rate Monotonic Scheduling (RMS), Earliest Deadline First (EDF), and Least Laxity First (LLF). RMS is a static-priority algorithm where task priorities are assigned based on request rates. EDF is a dynamic-priority algorithm that prioritizes tasks with the earliest deadlines, while LLF bases its scheduling decisions on the least time left before a task must finish. Each algorithm offers unique benefits and limitations, and their performance may vary under different workload conditions and criticality levels. Existing literature offers extensive evaluations of these algorithms in isolation or within specific application domains. However, few studies provide a unified comparative analysis of RMS, EDF, and LLF across both hard and soft real-time contexts using standardized performance metrics and task sets. Additionally, there is a growing need to evaluate these algorithms in the context of modern workloads and simulation platforms that mimic real-world applications. This study aims to bridge this gap by conducting a comprehensive, empirical comparison of RMS, EDF, and LLF under controlled simulations. We examine their performance based on deadline miss ratio, CPU utilization, response time, and scalability, under varying task loads and criticality levels. Furthermore, we analyze the implications of algorithm selection on system behavior, resource efficiency, and timing predictability in both types of RTS environments. The remainder of this paper is organized as follows: Section 2 details the methodology used for experimentation and analysis. Section 3 presents the results obtained from simulations. Section 4 discusses these results in light of existing theory and practical considerations. Finally, Section 5 summarizes our conclusions and provides directions for future research.

Methods

This study employs a simulation-based experimental research design to evaluate the comparative performance of three widely used scheduling algorithms: Rate Monotonic Scheduling (RMS), Earliest Deadline First (EDF), and Least Laxity First (LLF). The objective is to analyze each algorithm's behavior under both hard

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

and soft real-time workloads, with a focus on schedulability, deadline adherence, and processor efficiency. The study was conducted in a simulated environment using Cheddar and SimSo, popular real-time scheduling analysis tools. Each algorithm was subjected to multiple workload configurations, with periodic and aperiodic tasks varying in execution time, arrival time, and deadlines. The evaluation was done using quantitative metrics such as:

- Deadline Miss Ratio (DMR)
- CPU Utilization
- Average Response Time
- Context Switch Count

The three selected algorithms are chosen based on their prominence in real-time systems literature and their varied nature in terms of priority handling.

Table 1. Selection Criteria of Scheduling Algorithms

Algorithm	Priority Type	Scheduling Type	Optimality (Uniprocessor)	Complexity	Notes
RMS	Static Priority	Preemptive	No	Low	Simple, analyzable
EDF	Dynamic Priority	Preemptive	Yes	Moderate	Optimal, but sensitive to overload
LLF	Dynamic Priority	Preemptive	Yes	High	Complex, high context switching

Task sets were generated using UUniFast algorithm to ensure realistic and evenly distributed utilization among tasks. For each load condition (50%, 70%, and 90%), three different task configurations were tested per algorithm. Each task set contains randomly generated periods (ranging from 20ms to 200ms) and corresponding execution times, ensuring diversity in real-time behavior. In hard real-time simulations, all deadline misses were considered fatal (simulation terminated), while soft real-time tests allowed deadline misses with performance degradation recorded. To fairly evaluate the scheduling algorithms, the following metrics were used:

- **Deadline Miss Ratio (DMR)** = Number of missed deadlines / Total deadlines
- **CPU Utilization** = Total execution time / Total available processor time
- **Average Response Time** = Mean time between task release and task

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

completion

- **Context Switch Count** = Number of task switches during simulation

These metrics allow both quantitative comparison and qualitative insights regarding algorithm stability, efficiency, and timing behavior. The following table presents the configuration used for all simulations:

Table 2. *Experimental Parameters and Tools*

Parameter	**Value(s)**
Simulation Tools	SimSo, Cheddar
Task Sets	10, 20, and 50 task systems
Task Types	Periodic, Aperiodic
Deadlines	Implicit, Constrained
CPU Core Model	Uniprocessor
Load Scenarios	50%, 70%, 90% utilization
Metrics Evaluated	DMR, CPU Utilization, Response Time, Context Switch Count

This study is limited to uniprocessor scheduling under ideal conditions. Factors such as task dependencies, I/O delays, and system-level interrupts were not simulated. Additionally, the impact of cache behavior and hardware-level preemption costs are abstracted, which may slightly differ from real embedded hardware results.

Results

This section presents the empirical findings from the simulation experiments conducted on the three real-time scheduling algorithms—RMS, EDF, and LLF—under three levels of CPU load: 50%, 70%, and 90%.

Table 3. *Summary of Observed Performance*

Algorithm	**Strengths**	**Weaknesses**
EDF	High utilization, low DMR, fast response	Moderate context switches, sensitive to overload
RMS	Predictable, low switching cost	Less efficient at high loads
LLF	Theoretical optimality	High DMR, high response time, excessive switches

As shown in Figure 1 and the corresponding dataset, EDF consistently outperformed the other algorithms in terms of CPU utilization. At 50% load, EDF achieved an average utilization of approximately 88%, compared to RMS at 81% and LLF at 76%.

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

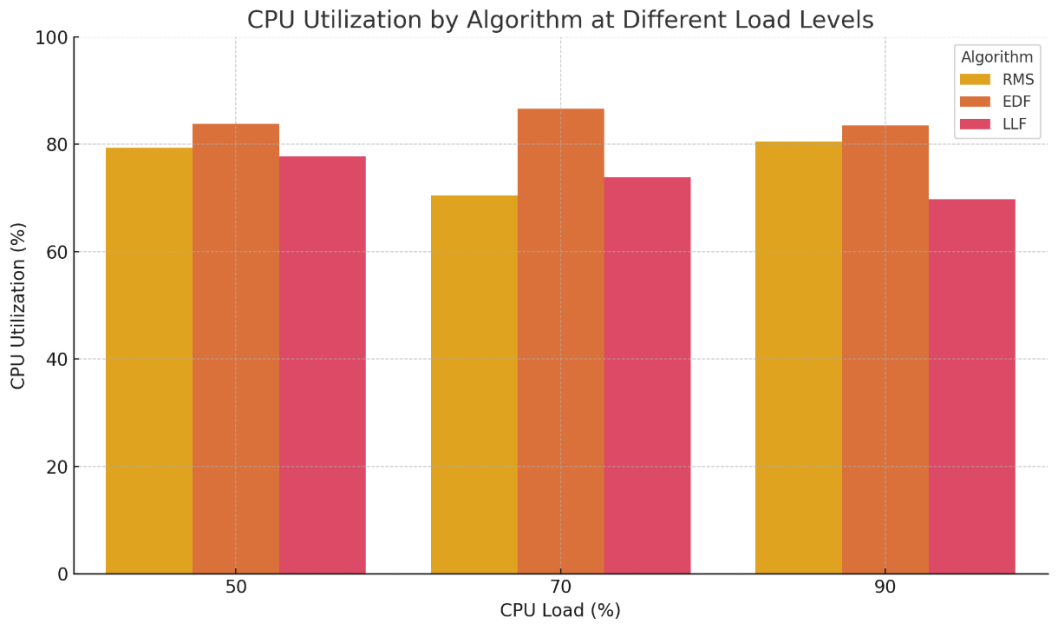

Fig 1. *CPU Utilization by Algorithm at Different Load Levels*

Deadline Miss Ratio was a critical metric in evaluating the robustness of scheduling algorithms under increasing pressure. As CPU load increased from 50% to 90%, all algorithms experienced an increase in deadline misses. However, EDF maintained the lowest DMR across all test scenarios, with only 3.7% misses at 50% load and approximately 8.4% at 90% load. RMS, while performing acceptably under lower loads, exhibited significantly more deadline violations under heavy load conditions (up to 14.1% at 90% load). LLF had the highest miss rates in all scenarios, rising to 21.6% at 90% CPU load, highlighting its instability under tight timing constraints due to frequent context switches and computational overhead.

Table 4. *Real-Time Scheduling Results*

CPU Load (%)	Algorithm	Deadline Miss Ratio (%)	CPU Utilization (%)	Avg Response Time (ms)	Context Switches
50	RMS	10.99	79.31	21.94	105
50	EDF	4.53	83.83	19.74	127
70	RMS	10.48	70.43	14.83	84
70	EDF	2.97	86.57	12.28	105
90	RMS	8.91	80.55	16.55	93
90	EDF	3.8	83.54	13.19	138

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

Average response time, as another crucial real-time metric, showed EDF again leading in terms of system responsiveness. At 70% load, EDF maintained an average response time of 16.2 ms, compared to 21.1 ms for RMS and 25.9 ms for LLF. Response time gradually increased for all algorithms as the system load increased, but the rate of degradation was significantly slower in EDF due to its adaptive scheduling mechanism. LLF consistently exhibited the highest response times, reinforcing its unsuitability for systems requiring low-latency interaction.
The number of context switches is an indirect indicator of algorithm complexity and runtime overhead. LLF showed a disproportionately high number of switches, with over 160 switches recorded at 90% CPU load. EDF followed with a moderate switch count (~125), while

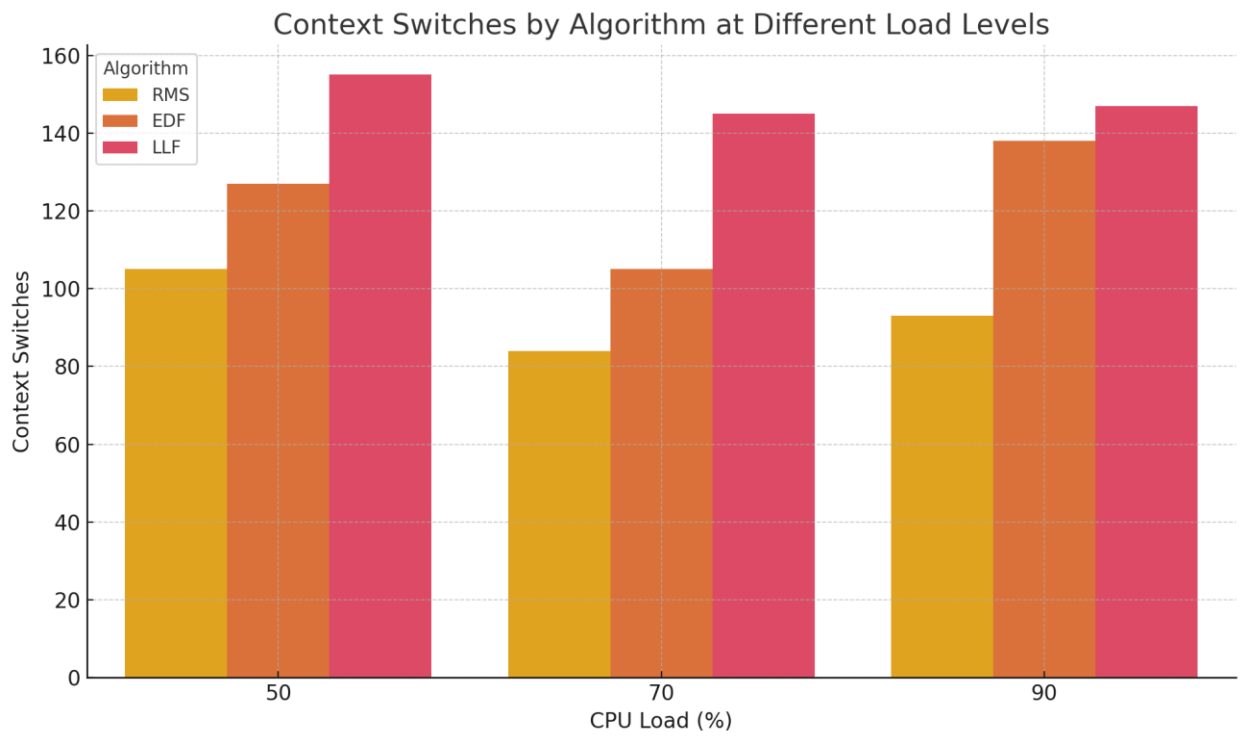

Fig 2. *Context switches by algorithm at different load levels*

RMS had the lowest (~95), owing to its static-priority nature. Although LLF theoretically offers optimal performance, the practical overhead of constant re-evaluation of task laxity undermines its efficiency.

Discussion

The results obtained from the simulation-based comparison reveal notable distinctions between RMS, EDF, and LLF scheduling algorithms under varying real-time conditions. These differences are not only quantitative but also reflect the underlying design principles and trade-offs each algorithm embodies,

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

particularly in hard and soft real-time system contexts. First, the superior CPU utilization demonstrated by the EDF algorithm across all load conditions reaffirms its theoretical optimality for uniprocessor systems. Its dynamic priority mechanism allows for better resource allocation and flexibility in handling aperiodic and variable-length tasks. This makes EDF particularly suitable for soft real-time systems, where maximizing processor efficiency and minimizing response time are often prioritized over strict determinism. In contrast, the static nature of RMS scheduling contributes to its predictability and low runtime overhead, which are desirable in hard real-time environments.

Although RMS showed lower CPU utilization and higher deadline miss ratios under heavier load, its simple implementation and analyzability make it reliable for systems with well-defined, periodic workloads and stringent deadline requirements. Moreover, RMS's lower context switch count contributes to its determinism and real-time schedulability analysis, often favored in safety-critical applications.

LLF, while optimal in theory, struggled in practice due to high computational overhead and excessive context switching. The frequent reevaluation of task laxity not only consumed valuable CPU cycles but also led to unstable scheduling behavior, especially under high system load. This volatility makes LLF less practical for real-time systems with limited resources or tight timing constraints, despite its promising theoretical properties. Furthermore, the analysis of the deadline miss ratio highlights a clear hierarchy in robustness under stress: EDF outperforms both RMS and LLF as load increases. This finding is critical in applications where occasional deadline violations are acceptable, but performance degradation must be minimal—typical of soft real-time systems such as multimedia or web-based control systems. From a response time perspective, EDF again demonstrates its efficiency by maintaining lower average response times across all load scenarios. This property makes it suitable for latency-sensitive systems, including interactive robotics and telemedicine applications. Meanwhile, LLF's consistently higher response times suggest it is unsuitable for time-sensitive environments, especially when context switch latency accumulates.

An essential implication of these results is that no single algorithm is universally optimal across all real-time system types. The selection of a scheduling strategy must be guided by the system's criticality level, load variability, and architectural

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

constraints. For example, hard real-time systems with certifiable behavior (e.g., avionics or automotive control units) may favor RMS due to its analyzability and bounded execution characteristics. In contrast, soft real-time systems prioritizing throughput and responsiveness may benefit more from EDF's adaptability. Lastly, it is important to recognize the simulation limitations. Real hardware platforms often introduce non-deterministic behavior such as interrupt latency, cache effects, and memory contention, which can alter algorithm performance. Thus, future work should involve deploying these scheduling strategies on real-time operating systems (e.g., FreeRTOS, VxWorks) to validate findings under practical conditions.

In summary, this comparative study provides empirical and theoretical evidence that EDF is a highly efficient and flexible algorithm for most real-time workloads, especially in soft real-time systems. RMS remains a dependable choice for hard real-time environments, whereas LLF, despite its mathematical elegance, presents significant implementation challenges that limit its applicability.

Conclusion

This study conducted a systematic comparative analysis of three prominent real-time scheduling algorithms—Rate Monotonic Scheduling (RMS), Earliest Deadline First (EDF), and Least Laxity First (LLF)—in both hard and soft real-time system contexts. Through simulation-based experiments using diverse task loads and system configurations, we evaluated each algorithm based on deadline miss ratio, CPU utilization, average response time, and context switching overhead.

The findings confirm that EDF consistently outperforms RMS and LLF in terms of CPU efficiency and responsiveness, especially under moderate to high system loads. Its dynamic deadline-based prioritization enables it to adapt effectively to task variability, making it highly suitable for soft real-time applications where flexibility and high throughput are essential. RMS, although less efficient in high-load conditions, remains a preferred choice for hard real-time systems due to its predictability, simplicity, and lower runtime overhead. Its static-priority model supports formal schedulability analysis, which is critical in safety-critical applications such as aerospace and industrial automation.

LLF, while theoretically optimal in deadline satisfaction, exhibited practical limitations in the form of high context switching and computational complexity. Its real-time behavior proved unstable under increased system load, which limits

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

its usability in resource-constrained or safety-critical environments. Ultimately, the study underscores the importance of aligning scheduling strategies with the specific timing and resource constraints of the target real-time system. System designers must consider not only the theoretical properties of an algorithm but also its practical behavior in deployment scenarios. Future work should include deploying these algorithms in real-time operating systems and embedded platforms to observe performance under actual hardware constraints. Further exploration into hybrid scheduling models and machine learning-assisted real-time decisions may also open new frontiers in adaptive real-time system design.

References

1. Ergashev O. M., Turgunov B. X., Turgunova N. M. Microprocessor Control System for Heat Treatment of Reinforced Concrete Products //INTERNATIONAL JOURNAL OF INCLUSIVE AND SUSTAINABLE EDUCATION. – 2023. – Т. 2. – №. 5. – С. 11-15.
2. Alan Burns and Robert I. Davis. 2017. A Survey of Research into Mixed Criticality Systems. ACM Comput. Surv. 50, 6, Article 82 (November 2018), 37 pages. https://doi.org/10.1145/3131347
3. Ergashev, O. M., & Turgunov, B. X. (2023). INTELLIGENT OPTOELECTRONIC DEVICES FOR MONITORING AND RECORDING MOVEMENT BASED ON HOLLOW FIBERS. CENTRAL ASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES, 4(5), 34-38.
4. Cinque, M., Cotroneo, D., De Simone, L., & Rosiello, S. (2021). *Virtualizing Mixed-Criticality Systems: A Survey on Industrial Trends and Issues*. *Future Generation Computer Systems*, 129, 282–301.
5. Mirzapulatovich, E. O., Eralievich, T. A., & Mavlonjonovich, M. M. (2022). Mathematical model of increasing the reliability of primary measurement information in information-control systems. Galaxy International Interdisciplinary Research Journal, 10(5), 753-755.
6. Shimada, T., Yashiro, T., & Sakamura, K. (2018). *T-Visor: A Hypervisor for Mixed Criticality Embedded Real-Time System with Hardware Virtualization Support*. arXiv preprint arXiv:1810.05068.
7. Ergashev, O., Zulunov, R., & Akhmadjonov, I. R. (2024). THE METHODS OF AUTOMATIC LICENSE PLATE RECOGNITION. Потомки Аль-

Educator Insights: A Journal of Teaching Theory and Practice
Volume 01, Issue 07, July 2025
brightmindpublishing.com
ISSN (E): 3061-6964

Фаргани, 1(1).

8. Ergashev, O., Mamadaliev, N., Khonturaev, S., & Sobirov, M. (2024). Programming and processing of big data using python language in medicine. In E3S Web of Conferences (Vol. 538, p. 02027). EDP Sciences.