Combined Research Paper:

Comparative Study and Future Enhancements of Round Robin CPU Scheduling Algorithms Across Multiple Operating Systems

Abstract

The Round Robin (RR) CPU scheduling algorithm, renowned for its fairness and simplicity in time-sharing systems, continues to evolve in response to modern computing demands. This paper consolidates insights from recent research contributions to propose an integrated analysis of RR scheduling across four major operating systems — Linux, Windows, FreeBSD, and Android. By reviewing improvements like dynamic time quantum adjustments, context-aware scheduling, and AI-assisted decision models, this work highlights both the current state and future scope of RR scheduling algorithms. The study also compares similarities and distinctions in RR implementations and optimizations across these systems.

Introduction

CPU scheduling plays a pivotal role in the effective management of system resources and process execution. Among numerous algorithms, Round Robin (RR) remains a core strategy in time-sharing and real-time systems due to its equitable process handling through cyclic time slices. Recent research papers have introduced various enhancements to the traditional RR algorithm, including dynamic time quantum adjustments, smart scheduling techniques, and AI-integrated scheduling models.

This paper integrates the findings from four notable studies:

- Smart Round Robin CPU Scheduling Algorithm (2020, IEEE)
- Assessing FIFO and RR Scheduling Effects on Data Pipeline Performance (2024, arXiv)

- Improved Version of Round Robin Scheduling Based on Analytic Model (2020, Springer)
- Round Robin Scheduling in CPU and Cloud Computing: A Review (2024)

and extends their ideas into a comparative analysis of RR implementations in four widely used operating systems.

Review of Existing Enhancements to Round Robin

From the four papers:

- **Dynamic Time Quantum Adjustment**: Proposes variable time quanta based on process burst times to minimize context switches and improve turnaround times (IEEE 2020, Springer 2020).

- **AI-Assisted Scheduling**: Suggests predictive models for dynamic quantum management and process prioritization in modern workloads (arXiv 2024, Springer 2020).

- **Energy-Aware Scheduling**: Focuses on reducing energy consumption in RR scheduling within high-load environments like data pipelines and cloud servers (arXiv 2024).

- **Cloud-Optimized RR**: Discusses RR scheduling adaptations for cloud computing and distributed systems (ResearchGate 2024).

Comparative Study of Round Robin in Four Operating Systems

Feature	Linux	Windows	FreeBSD	Android
Primary Scheduler	CFS (uses SCHED_RR for RT)	MLFQ with RR for same-priority	Hybrid MLFQ-RR	CFS with selective SCHED_RR
Time Quantum Management	Configurable for SCHED_RR	Dynamic, load-aware	Kernel-defined, adjustable	Tuned for mobile, short quantum
Real-timeProcess Handling	Static priority in SCHED_RR	Boost/demote policy	Dynamic priority adjustments	Real-time audio/UI threads via SCHED_RR
Energy-Awareness	Limited in RR, better in CFS	Minimal	Limited	Integrated with Android's PowerHAL
Cloud/Distributed Adaptation	Partial via groups & containers	Hyper-V scheduling improvements	Limited	Containerized environments support

Similarities in RR Scheduling Approaches

- All four OS employ Round Robin scheduling for processes at equal priority.
- Time quantum size critically influences performance and context-switching rates in all cases.
- None of the operating systems currently deploy adaptive AI-based RR scheduling as default — though this is proposed in multiple papers.
- All integrate priority-based handling alongside RR for real-time or I/O-sensitive processes.

Key Differences

- **Linux and Android** focus on real-time scheduling through SCHED_RR, while **Windows** uses RR within its MLFQ design.
- **Time quantum tuning** is static in FreeBSD, configurable in Linux, dynamic in Windows, and tightly tuned for responsiveness in Android.
- Android uniquely integrates RR scheduling with **power and battery management frameworks**.
- Linux and Windows offer broader support for **cloud and multi-core optimizations**.

Proposed Future Enhancements

Based on collective insights from the referenced papers:

- **AI-Powered Adaptive Quantum Management** for workload prediction and optimized quantum scaling.
- **Energy-Conscious RR Scheduling** integrating real-time workload awareness, especially for mobile and edge devices.
- **Cloud-native Distributed RR Scheduling Frameworks** that balance CPU time across containers and virtual CPUs.
- **Context-Aware Priority Elevation** for I/O-bound and multimedia processes to improve responsiveness.

Conclusion

Round Robin remains a relevant and widely implemented scheduling strategy, especially for fairness in time-sharing systems. While its core design is preserved, recent research and system-specific implementations reveal opportunities for significant improvements. This paper synthesizes recent advancements and proposes unified, AI-driven, and energy-efficient RR scheduling models for future cross-platform operating systems.

References

https://ieeexplore.ieee.org/document/9020439 https://arxiv.org/abs/2409.15704
https://ieeexplore.ieee.org/document/9114602
https://link.springer.com/article/10.2991/ijndc.k.200804.001