International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Literature Review on Fair Chance Round Robin Arbiter

Anuradha Shenoy[1], Adithya S Chakravarty[2], Anika Kirana Venkat[3], Dr. Smitha Gayathri D[4]

[1,2,3,4]Dept. of Electronics and Communication. Engineering, BNM Institute of Technology, Bengaluru, India

Abstract

The Fair Chance Round Robin (FCRR) arbiter is a scheduling mechanism designed to enhance fairness in resource allocation among competing processes in digital systems. This literature review explores the implementation of FCRR arbiters, focusing on their operational principles, advantages, and performance metrics compared to traditional round-robin and priority-based arbitration methods. The review highlights how FCRR mitigates issues such as starvation and unfair resource distribution by ensuring that each process receives a fair share of resources over time. A comparative analysis with other arbitration techniques reveals the strengths and weaknesses of FCRR in various application scenarios, including real-time systems and multi-core processors. Furthermore, the review discusses future research directions aimed at optimizing FCRR implementations to improve efficiency and adaptability in dynamic environments. By synthesizing current findings, this review provides valuable insights into the ongoing development of fair arbitration mechanisms in computing systems.

I. INTRODUCTION

The implementation of Fair Chance Round Robin (FCRR) arbiters represents a significant advancement in scheduling algorithms, particularly in environments where fairness and efficient resource allocation are paramount. Traditional round-robin scheduling methods, while straightforward and easy to implement, often lead to challenges such as starvation and unequal distribution of resources among competing processes. These limitations can adversely affect system performance, especially in high-demand scenarios where multiple processes vie for limited resources. The FCRR arbiter addresses these issues by introducing a dynamic scheduling mechanism [2][13] that ensures each process receives a fair opportunity to execute, thereby enhancing overall system responsiveness and efficiency.

Fairness in resource allocation is critical in various applications, including real-time systems, multi-core processors, and network scheduling. As systems become increasingly complex and the number of concurrent processes grows, the need for sophisticated arbitration techniques that can adapt to varying workloads becomes evident [7][9]. The FCRR arbiter stands out by not only providing equitable time slices but also by adjusting those slices based on process behaviour and system state. This adaptability allows it to maintain fairness even under fluctuating loads, making it a valuable tool in modern computing environments.

In addition to its fairness benefits, the FCRR arbiter offers improved performance metrics compared to traditional scheduling algorithms. By reducing the likelihood of starvation and ensuring that all processes

receive adequate CPU time, FCRR enhances the overall throughput of the system. Furthermore, its design allows for easier integration into existing systems without significant overhead, making it an attractive option for developers seeking to improve resource management without extensive modifications to their architectures.

II. METHODOLOGY

The methodology for this literature review on the implementation of the Fair Chance Round Robin (FCRR) arbiter and its comparative analysis involved a systematic approach to gather, analyze, and synthesize relevant academic and industrial research. The aim was to capture the latest advancements in scheduling algorithms, particularly focusing on fairness in resource allocation among competing processes.

A. Search Strategy

To ensure comprehensive coverage of the topic, a set of targeted keywords was employed during the literature search. These keywords included:

- Fair Chance Round Robin Arbiter
- Scheduling Algorithms
- Resource Allocation in Computing
- Fairness in Scheduling
- Comparative Analysis of Scheduling Techniques
- Round Robin Scheduling
- Priority-Based Scheduling

These keywords were selected to encompass a broad range of literature related to FCRR and its performance compared to other scheduling methods.

B. Search Filters

The search was conducted using various academic databases, including IEEE Xplore, ScienceDirect, Google Scholar, and SpringerLink. The search was filtered to include papers published between 2010 and 2024 to focus on recent advancements in scheduling algorithms. Additionally, only peer-reviewed journals, conference proceedings, and reputable industry reports were considered to ensure the reliability and quality of the information.

C. Inclusion Criteria

The inclusion criteria for selecting studies were as follows:

- Research focused specifically on the design and implementation of Fair Chance Round Robin arbiters.
- Studies that analyse fairness in scheduling algorithms and their impact on system performance.
- Comparative studies that evaluate FCRR against traditional round-robin and priority-based scheduling techniques.
- Papers addressing practical applications of FCRR in various computing environments, including real-time systems and multi-core processors.

D. Data Extraction and Analysis

Once relevant studies were identified, key data points were extracted, including:

- The design principles of FCRR arbiters.
- Performance metrics such as throughput, latency, and fairness indices.
- Case studies demonstrating the implementation of FCRR in real-world scenarios.

- Comparative results highlighting the advantages and limitations of FCRR relative to other scheduling methods.

The extracted data was then synthesized to provide a cohesive understanding of the current state of research on FCRR arbiters. This synthesis involved identifying trends, gaps in existing literature, and potential areas for future research.

III. MAIN BODY

A. Implementation of Fair Chance Round Robin Arbiter

The Fair Chance Round Robin (FCRR) arbiter is a sophisticated scheduling algorithm designed to enhance fairness in resource allocation among competing processes in digital systems. Unlike traditional round-robin algorithms, which allocate fixed time slices to processes, the FCRR arbiter adapts its time allocation based on the behaviour and needs of each process. This dynamic adjustment not only improves fairness but also optimizes overall system performance by minimizing the risk of starvation for lower-priority tasks.

The implementation of the FCRR arbiter involves several key components. First, a well-defined time slice allocation mechanism is established. In this mechanism, each process is assigned a base time slice, which can be dynamically adjusted based on factors such as process priority and execution history. For instance, if a process consistently requires more CPU time due to its workload, the FCRR arbiter can increase its time slice allocation, allowing it to complete its tasks without unduly delaying other processes. This adaptability is crucial in environments with varying workloads and competing demands.

Additionally, the FCRR arbiter employs an efficient queue management system to maintain the order of processes waiting for CPU access. This system ensures that all processes are treated fairly while preventing any single process from monopolizing resources. The queue is managed using a circular buffer that allows for quick updates and reordering based on real-time performance metrics. By continuously monitoring the state of each process and adjusting their positions in the queue accordingly, the FCRR arbiter can effectively balance resource allocation among competing tasks.

B. Comparative Analysis with Traditional Scheduling Techniques

To evaluate the effectiveness of the FCRR arbiter, a comparative analysis with traditional scheduling methods such as standard round-robin and priority-based scheduling is essential [5]. Traditional round-robin scheduling allocates fixed time slices to each process in a cyclic manner, which can lead to inefficiencies when processes have varying execution requirements. In contrast, priority-based scheduling assigns CPU time based on predefined priorities, potentially resulting in starvation for lower-priority processes.

he FCRR arbiter demonstrates significant advantages over these traditional methods by providing a more equitable distribution of CPU time. Studies have shown that systems utilizing FCRR exhibit improved throughput and reduced waiting times compared to those using standard round-robin scheduling. Furthermore, the dynamic nature of FCRR allows it to adapt to changing workloads more effectively than static priority-based systems, which may struggle under fluctuating demands.

Performance metrics such as average turnaround time, response time, and CPU utilization are critical in assessing the comparative performance of these scheduling techniques. Experimental results indicate that systems implementing the FCRR arbiter achieve lower average turnaround times and improved CPU utilization rates compared to both traditional round-robin and priority-based schedulers[5]. These findings underscore the importance of adopting advanced scheduling mechanisms like FCRR in modern computing

environments where fairness and efficiency are paramount.

C. Applications of Fair Chance Round Robin Arbiter

The Fair Chance Round Robin (FCRR) arbiter has a wide range of applications across various domains, primarily due to its ability to provide fair and efficient resource allocation in systems with competing processes. Below are some key applications where FCRR arbiters are particularly beneficial:

1. Real-Time Systems: In real-time computing environments, where timely execution of tasks is critical, FCRR arbiters ensure that high-priority processes receive the necessary CPU time while still allowing lower-priority tasks to progress. This is essential in applications such as embedded systems, automotive control systems, and industrial automation, where delays can lead to system failures or safety hazards.

2. Multi-Core Processors: The FCRR arbiter is well-suited for managing resource allocation in multi-core processor architectures. By dynamically distributing workloads across multiple cores while maintaining fairness among competing processes [1][3][8], FCRR can enhance overall system throughput and efficiency. This application is particularly relevant in modern computing environments where parallel processing is common.

3. Network Scheduling: In networking, FCRR arbiters can be used to manage bandwidth allocation among multiple data streams. By ensuring that each stream receives a fair share of bandwidth, FCRR helps prevent congestion and improves the quality of service for applications such as video streaming, online gaming, and VoIP communications.

4. Cloud Computing: In cloud environments where multiple users share resources, the FCRR arbiter can effectively manage virtual machine scheduling and resource allocation. By providing fair access to computing resources, it helps optimize performance and ensures that no single user monopolizes the available resources, thereby improving the overall user experience.

5. Distributed Systems: In distributed computing environments, where processes may be spread across multiple nodes, the FCRR arbiter can help coordinate resource allocation among these nodes. This ensures that all processes receive adequate resources regardless of their location within the network, enhancing system reliability and performance.

6. Gaming and Simulation: In gaming applications, where multiple players or entities compete for limited resources (e.g., CPU time for game logic), the FCRR arbiter can ensure that all players have a fair chance to participate without experiencing significant delays or performance degradation.

7. Data Processing Applications: For applications involving data processing tasks that require equitable access to computational resources—such as in big data analytics or machine learning training—the FCRR arbiter can help balance workloads efficiently among various processing units.

IV. DISCUSSION/ANALYSIS

The implementation of the Fair Chance Round Robin (FCRR) arbiter presents a significant advancement in the field of scheduling algorithms, particularly in addressing the inherent limitations of traditional round-robin and priority-based scheduling methods. One of the primary advantages of FCRR is its ability to dynamically allocate resources based on real-time process behaviour, which enhances overall system fairness and efficiency. This adaptability is crucial in environments where processes exhibit varying execution requirements and priorities, as it minimizes the risk of starvation for lower-priority processes while still accommodating the needs of higher-priority tasks.

Empirical studies comparing FCRR with standard round-robin scheduling have demonstrated notable improvements in performance metrics such as throughput, average waiting time, and response time. For

instance, systems utilizing FCRR have shown a reduction in average waiting times by up to 30% compared to traditional methods [5]. This improvement can be attributed to the arbiter's dynamic time slice allocation, which allows for more efficient use of CPU resources [18][19][20]. Additionally, FCRR's design facilitates better responsiveness in multi-tasking environments, where processes may require varying amounts of CPU time based on their current state and workload.

However, the implementation of FCRR is not without challenges. The complexity introduced by dynamic time slice adjustments necessitates a robust monitoring mechanism to track process behaviour effectively. This requirement can lead to increased computational overhead, which may offset some of the performance gains achieved through enhanced fairness. Furthermore, while FCRR provides significant improvements in fairness and efficiency, its performance can be influenced by system load and specific characteristics of the processes involved. For example, under heavy load conditions with many competing processes, the benefits of FCRR may diminish as the arbiter struggles to maintain equitable resource distribution.

The comparative analysis with other scheduling techniques reveals that while FCRR excels in fairness and adaptability, it may not always outperform priority-based scheduling in scenarios where strict priority enforcement is crucial. In applications requiring guaranteed execution times for critical tasks, priority-based methods may still be preferable despite their potential for unfair resource allocation among lower-priority tasks. Therefore, selecting an appropriate scheduling algorithm should consider the specific requirements and constraints of the application domain.

In conclusion, the Fair Chance Round Robin arbiter represents a promising approach [6] to resource allocation in digital systems, offering significant advantages in fairness and efficiency over traditional scheduling methods. However, careful consideration must be given to its implementation challenges and performance trade-offs when applied in real-world scenarios. Future research should focus on optimizing FCRR's dynamic adjustment mechanisms to reduce computational overhead while maintaining its core advantages. Additionally, exploring hybrid approaches that combine elements of FCRR with other scheduling techniques could yield further enhancements in performance across diverse computing environments.

V. CONCLUSION

The implementation of the Fair Chance Round Robin (FCRR) arbiter represents a significant advancement in scheduling algorithms [21], addressing the limitations of traditional methods by promoting fairness and efficiency in resource allocation. Through dynamic time slice allocation, the FCRR arbiter ensures that all processes receive equitable access to CPU time, thereby minimizing issues such as starvation and improving overall system performance. The comparative analysis with standard round-robin and priority-based scheduling techniques highlights the advantages of FCRR in various performance metrics, including throughput, average waiting time, and response time.

Despite its benefits, the FCRR arbiter does present certain challenges, particularly regarding the complexity of its implementation and the computational overhead associated with dynamic adjustments. These factors can impact system performance under high load conditions. However, the adaptability of FCRR to varying workloads positions it as a superior choice for modern computing environments where fairness is essential.

The applications of FCRR arbiters span diverse domains, including real-time systems, multi-core processors, and network scheduling, demonstrating its versatility in enhancing resource management

strategies. As digital systems continue to evolve and the demand for equitable resource allocation grows, the adoption of FCRR arbiters is likely to increase.

Future research should focus on optimizing the dynamic adjustment mechanisms of the FCRR arbiter to further reduce computational overhead while maintaining its core advantages. Additionally, exploring hybrid approaches that integrate FCRR with other scheduling techniques may yield further improvements in performance and adaptability. Overall, the Fair Chance Round Robin arbiter stands out as a promising solution for achieving fair and efficient resource allocation in increasingly complex computing environments.

REFERECES

1. Aung, Toe. "Design and Verification of a Round-Robin Arbiter." null (2018).
2. Zhizhou Fu and Xiang Ling, "The design and implementation of arbiters for Network-on-chips," 2010 2nd International Conference on Industrial and Information Systems, Dalian, China, 2010, pp. 292-295, doi: 10.1109/INDUSIS.2010.5565854
3. Prateek Karanpuria. "Fair Chance Round Robin Arbiter" International Journal of Computer Applications.81, 14(November 2013), 36-40. DOI=10.5120/14187-2446
4. Tarun Kumar Gauttam, Rekha Agrawal, Sandhya Sharma, "Design and Simulation of Router Using WWF Arbiter and Crossbar", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075 (Online), Volume-3 Issue-3, August 2013
5. N.L, Venkataraman & Subramanian, Sumithra & Ramaiah, Purushothaman & Kumar, Subramani & Kokulavani, Kathiresan & Gowri, Velankanni. (2023). "Design of matrix, distributive round robin, ping pong and enhanced ping lock arbiter for shared resources systems", Indonesian Journal of Electrical Engineering and Computer Science. 32. 1337. 10.11591/ijeecs.v32.i3.pp1337-1345.
6. G. Xiaopeng, Z. Zhe and L. Xiang, "Round Robin Arbiters for Virtual Channel Router," The Proceedings of the Multiconference on "Computational Engineering in Systems Applications", Beijing, China, 2006, pp. 1610-1614, doi: 10.1109/CESA.2006.4281893.
7. Li, Yihan & Panwar, Shivendra. (2002). The dual round robin matching switch with exhaustive service". 58 - 63. 10.1109/HPSR.2002.1024209.
8. H. J. Chao, C. H. Lam, and X. Guo, "A Fast Arbitration Scheme for Terabit Packet Switches," Proceedings of IEEE Global Telecommunications Conference, 1999, pp. 1236-1243.
9. H. J. Chao and J. S. Park, "Centralized Contention Resolution Schemes for a Larger-capacity Optical ATM Switch," Proceedings of IEEE ATM Workshop, 1998, pp. 11-16.
10. Si Quing Zheng and Mei Yang, "Algorithm Hardware Code sign of Fast Parallel Round Robin Arbiters," IEEE transactions on Parallel and distributed Systems, January 2017
11. M. J. Karol, M. Hluchyj, and S. Morgan, "Input versus output queuing on a space-division packet switch," IEEE Trans. on Communications, vol.35, pp. 1347-1356, 1987.
12. Eung S Chin,Vincent J. Mooney III and George F Riley, "Round Robin Arbiter Design and Generation," ISSS'02, October 2-4 2002 Kyoto, Japan.
13. N. Mckeown, P. Varaiya, and J. Warland, "The iSLIP Scheduling Algorithm for Input-Queued Switch," IEEE Transaction on Networks, 1999, pp. 188-201.
14. Nick McKeown,"The iSLIP Scheduling Algorithm for Input-Queued Switches" IEEE/ACM transactions on networking, vol. 7, no. 2, April 1999.

15. P. Gupta and N. Mckeown, “Designing and Implementing a Fast Crossbar Scheduler,” IEEE Micro, 1999, pp. 20-28.
16. Y. -L. Lee, J. M. Jou and Y. -Y. Chen, "A high-speed and decentralized arbiter design for NoC," 2009 IEEE/ACS International Conference on Computer Systems and Applications, Rabat, Morocco, 2009, pp. 350-353, doi: 10.1109/AICCSA.2009.5069347.
17. Kavitha, T. & Shiyamala, Subramani & Nagarajan, P. (2016). “FPGA implementation of arbiters algorithm for network-on-chip” 11. 11451-11456.
18. Ruma Deb, Dr Rajrajan, “Speed efficient implementation of round robin arbiter design using Verilog” International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463 Vol. 2 Issue 9, September-2013, pp: (1-9)
19. R. Kamal and J. M. Moreno Arostegui, "RTL implementation and analysis of fixed priority, round robin, and matrix arbiters for the NoC's routers," 2016 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 2016, pp. 1454-1459, doi: 10.1109/CCAA.2016.7813949