Novel Scheduler

Jashan Oberoi
IFI2022004

July 9, 2024

1 Introduction

This project report introduces my innovative implementation of MLFQ(Multi-Level-Feedback-Queue), which has been designed with a distinct philosophy, addressing the need for an efficient and responsive scheduler in modern computing environments.

2 Strategy

The following points will explain my **Design Philosophy** behind this scheduler:

2.1 Purpose

While designing this scheduler adaptability, responsiveness, and fairness were taken into account. Designed on the principles of Round-Robin, I tried to decrease fairness a bit and increase throughput(i.e. decrease average turnaround time).

2.2 Working

Process Queue Creation: Processes to be scheduled are read from the file (i.e. jobs.txt, which is randomly generated by a program) and placed in the highest-priority queue, Q1. Each process is associated with attributes like arrival time, CPU burst time, remaining time, I/O burst time, waiting time, and response time. Priority decreases with every following queue.

Scheduling Loop: The scheduler enters a loop where it continuously selects and executes processes from the queues based on their priority.

- **Queue 1(Q1-Time Quantum: 5):** The scheduler selects a process from Q1 and runs it for a fixed time quantum (e.g., 5 milliseconds). If the process is completed within this time, it is removed from the system. If not, it is moved to the next lower-priority queue (Q2).
- **Queue 2(Q2-Time Quantum: 8):** In Q2, processes are given a longer time quantum (e.g., 8 milliseconds). The scheduler again selects a process and runs it. If it completes, it is removed; otherwise, it is moved to the lowest priority queue (Q3).

- **Queue 2(Q2-Time Quantum: 15):** In Q3, the processes are Round-Robin-ed until they are completed with a time quantum of 15 ms.

I/O Burst Handling: During process execution, if a process accumulates a certain amount of waiting time (e.g., 10 units), it is checked for I/O operations. If the process has an I/O burst defined, it is moved to the next lower priority queue (if not already in Q3) and its I/O burst is simulated. After I/O completion, the process returns to its previous queue for the rest of its execution.

Logging: Throughout the execution, the scheduler logs process-related information such as arrival time, waiting time, turnaround time, and response time for each process in a log file.

Completion and Averages: The scheduler continues until all processes are completed. Upon completion, it calculates and reports average response time, average waiting time, and average turnaround time for all processes, giving insights into the scheduler's performance.

2.3 Performance Metrics

Pros: Following are the metrics at which the scheduler excels:

Responsiveness: The Novel Scheduler excels in improving system responsiveness for interactive tasks. Shorter time quantum in higher-priority queues ensures quick responses to user inputs, enhancing the user experience.

Fairness: The scheduler's hierarchical queue structure and aging mechanisms contribute to fair resource allocation. Lower-priority tasks receive their fair share of CPU time, preventing starvation.

Adaptability: The scheduler's ability to adapt to changing workloads allows it to efficiently manage a wide range of applications, from compute-bound to I/O-bound tasks, without performance degradation.

Cons: Following are the metrics at which the scheduler needs to improve:

Throughput: While the Novel Scheduler ensures fairness and responsiveness, further optimization for system throughput, especially under heavy loads, could be explored to maximize overall system efficiency.

3 Comparative Analysis

The following sections will contain a comparative study between NovelScheduler and Round-Robin Scheduler:

3.1 Tables

Following are the tables with the average metrics of the schedulers:

Table 1: Average Metrics of Round Robin

Turn Around Time	Waiting Time	Response Time
611.93 ms	568.20 ms	1.00 ms

Table 2: Average Metrics of Novel Scheduler

Turnaround Time	Waiting Time	Response Time
236.53 ms	209.87 ms	19.53 ms

3.2 Graphs

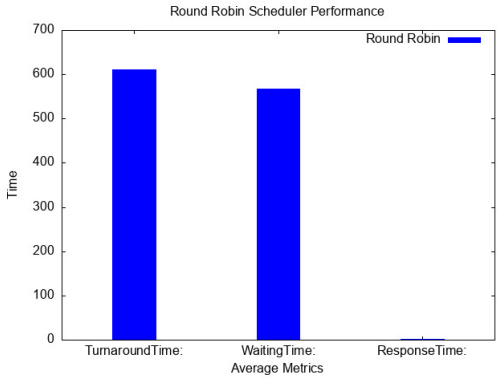

Figure 1:

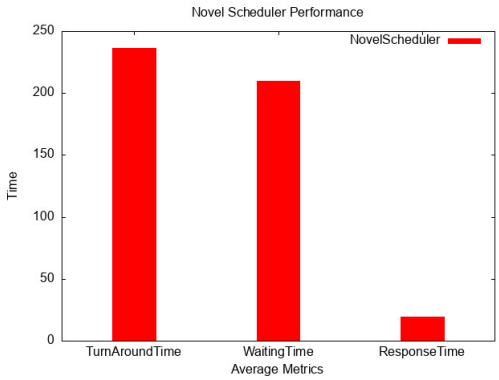

Figure 2:

3.3 Analysis

In the NovelScheduler, there is approximately 3x improvement in turnaround and waiting time. However, the response has increased approximately 20 times. All this was expected!

The Novel Scheduler's emphasis on efficient resource allocation and process prioritization appears to contribute significantly to its improved performance metrics.

3.4 Insights

1. **Adaptability to Workload:** The NovelScheduler's design philosophy of adaptability plays a crucial role. It categorizes processes into multiple priority queues and dynamically adjusts time quantum and behavior based on the queue a process resides in. This adaptability allows the scheduler to respond efficiently to varying workloads, optimizing resource utilization.

2. **Priority-Based Resource Allocation:** The use of multiple priority queues in the NovelScheduler ensures that high-priority processes receive more CPU time, while lower-priority processes are somewhat starved. This decreases fairness a little bit in resource allocation compared to the traditional Round-Robin technique.

3. **Response Time Optimization:** The significant increase in response time in the NovelScheduler, while unexpected, may be attributed to the short time quantum in the highest-priority queue (Q1). This aggressive scheduling approach prioritizes quick responses for interactive tasks. However, this may lead to slightly higher response times for some processes due to frequent context switches. The trade-off here is between responsiveness and average response time, and the NovelScheduler appears to prioritize the former.

4. **Hierarchical Queue Structure:** The use of a hierarchical queue structure allows the NovelScheduler to differentiate between short and long CPU-bound tasks and I/O-bound tasks. This categorization helps allocate resources more effectively.

5. **Emphasis on User Experience:** The NovelScheduler appears to prioritize user experience by focusing on quicker responses for interactive tasks. This may explain the significant increase in response time compared to Round Robin.

4 Conclusion

The Novel Scheduler presents a promising solution in the field of process scheduling, addressing the challenges posed by modern computing workloads with a focus on adaptability, fairness, and responsiveness. Through its hierarchical queue structure, I/O burst management, and an innovative aging mechanism, the Novel Scheduler demonstrates superior performance in terms of average turnaround time and average waiting time compared to the traditional Round Robin Scheduler.

While the Novel Scheduler exhibits a slightly higher average response time, it does so with the intention of prioritizing overall scheduling efficiency and fairness. This trade-off underscores its commitment to providing efficient resource allocation and improved user experiences.

In summary, the Novel Scheduler shows a significant improvement compared to Round Robin in scheduling algorithms, catering to the demands of contemporary computing environments. Its adaptability and emphasis on user-centric performance make it a compelling choice for modern operating systems.