**CSCE 5640: Operating System Design**

**Project Report**

**Name:** Kishan Kumar Zalavadia

**EUID:** 11685261

**Title:** CPU Scheduling Algorithms Implementation and Analysis

**Member name:** Kishan Kumar Zalavadia

**Project GitHub link:** <https://github.com/Kishan-Kumar-Zalavadia/OS_Scheduling_Algorithms>

1. **Introduction**

**Overview**

This project aims to implement and compare five scheduling algorithms: Shortest Job First (SJF), First Come, First Serve (FCFS), Priority Scheduling, Round Robin (RR), and Priority with Round Robin. CPU scheduling is critical in operating systems to ensure all processes are executed with the necessary CPU time based on their requirements. The implementation will also provide a visual comparison of scheduling algorithms based on different metrics and compare to find the best CPU scheduling algorithm.

**Problem Statement**

In modern operating systems, CPU scheduling is very important to manage processes efficiently. The main aim is to have maximum CPU utilization. There are many CPU scheduling algorithms available, but the challenge is to find the best scheduling algorithm that has a high performance in all kinds of workloads. This project aims to analyze all the scheduling algorithms under different workloads and compare their performance using metrics like average waiting time and average turnaround time.

**Importance**

CPU scheduling is important because it impacts system performance and user experience. Understanding CPU scheduling helps in CPU resource optimization. The performance of each algorithm may defer based on their workloads. It is useful for maximizing CPU utilization and ensuring timely response to critical tasks.

1. **Background**

In any multitasking operating system, CPU scheduling is important to manage the execution of multiple processes. By switching the CPU among tasks, the OS can make the computer more productive. The time duration for which a process requires the CPU to execute is the *CPU Burst time*. The number indicates the importance or priority of the process is the *Priority* of that process. In over-test cases, we have priorities from 1 to 15, 15 being the highest priority. To measure the performer, we have used two matrices, which are waiting time and turnaround time. The time that a process spends in a waiting queue is the *waiting time*. The total amount of time from submission to completion is the *turnaround time*. The following are the CPU Scheduling algorithms used.

**FCFS (First-Come-First-Served):**

First come, first serve is the simplest algorithm. In this algorithm, all the processes are executed in the order they arrived. It follows the FIFO(First in, first out) pattern. This is a non-preemptive algorithm, which means when a process is picked by the CPU, the CPU is not released until the process is completed.

**SJF (Shortest-Job-First)**

Shortest Job First is a non-primitive algorithm. In this algorithm, the processes are executed in the order of their burst times. The process having the smallest burst time will execute first, and so on. The processes are executed in the ascending order of their burst times.

**Priority Scheduling**

In this algorithm, the processes are executed based on their priorities. The highest-priority job is executed first, and the lowest-priority job is executed at the end. The process is executed and completed when it is assigned to the CPU.

**Round Robin (RR)**

In this algorithm, all the processes are cyclically assigned a fixed time slot for execution. This is a preemptive algorithm. This algorithm is mainly focused on time-sharing, where each process is executed in a certain amount of time, which is quantum time.

**Priority with Round-Robin**

This is a combination of priority scheduling and round-robin scheduling. We perform priority scheduling first, if two or more processes have the same priority, we use a round-robin for those processes whose priorities are the same.

1. **Implementation**

**Solution Approach**

The project implements five CPU Scheduling algorithms: Shortest Job First (SJF), First Come, First Serve (FCFS), Priority Scheduling, Round Robin (RR), and Priority with Round Robin. Each algorithms are created using Java programming language, which accepts a scheduling file as an input argument. The file contains the process, their priorities, and their burst times in the form [task name] [priority] [CPU burst].

The Java program ScheduleFileGenerator.java generates multiple test files that contain processes in the form[task name] [priority] [CPU burst].

The Java program AnalyzeScheduler.java runs each file present in the folder named “data” for each algorithm. It calculates waiting time and turnaround time for each process separately. It then finds the average waiting time and average turnaround time of each file in each scheduling algorithm. In the end, when all files are executed for all algorithms, it finds the combined average waiting time and turnaround time of each algorithm to evaluate and find the best algorithm. Additionally, it stores all the details in a txt file.

The Python code reads the Txt file and visualizes the results for a clear comparison, and analysis.

**Implementation Details**

* **Programming Language:** Java
* **Operating system:** Linux CSE machine.

**Test cases:**

To test the project, the ScheduleFileGenerator.java file creates the following files:

* 7 test files with 5 processes each
* 7 test files with 10 processes each
* 7 test files with 15 processes each

Each process has a burst time ranging from 5 milliseconds to 50 milliseconds and priority ranging from 1 to 15.

Below are a few examples of the test cases.

Example test case -1:

T1, 14, 21

T2, 9, 21

T3, 1, 25

T4, 10, 36

T5, 12, 7

Example test case -2

T1, 11, 28

T2, 9, 34

T3, 9, 36

T4, 2, 32

T5, 14, 33

T6, 2, 25

T7, 3, 18

T8, 4, 9

T9, 13, 48

T10, 11, 45

Example test case -3

T1, 14, 9

T2, 8, 35

T3, 9, 16

T4, 4, 6

T5, 1, 36

T6, 11, 19

T7, 8, 14

T8, 10, 34

T9, 1, 47

T10, 13, 42

T11, 3, 37

T12, 2, 34

T13, 10, 16

T14, 1, 27

T15, 4, 30

1. **Experimental Results:**

Below is the average waiting time by different scheduling algorithms.

A graph with lines and dots

Description automatically generated

Below is the average turnaround time by different scheduling algorithms.

A graph with colorful lines and text

Description automatically generated

Below is the complete average waiting time and turnaround time of all the test cases by different scheduling algorithms.

A graph of blue and orange bars

Description automatically generated

From the above visualization, we can conclude that the shortest job first is the best algorithm that has the least average waiting time and average turnaround time among all the different algorithms.

Round robin has the lowest performance because the average waiting time and average turnaround time are the highest. This means the process is required to wait longer if the round-robin scheduling algorithm is used.

The performance of Priority scheduling, First come, first serve, and priority with round robin are all similar.

1. **Conclusion**

The project has successfully implemented the five scheduling algorithms and analyzed them using different test cases. By comparing and testing all the CPU scheduling algorithms using multiple test cases, we gained insights, and hence, we conclude that Shortest Job First is the best CPU scheduling algorithm.

**Future Work**

* Implementation of a primitive version of algorithms.
* Addition of an aging mechanism to prevent from the starvation condition.
* Priorities can change dynamically, which again prevents starvation.
* Implementing and comparing even more CPU scheduling algorithms apart from these five.
* Experiment with different quantum values for Round-Robin and Priority with Round-Robin Scheduling algorithms.

1. **References**

<https://www.geeksforgeeks.org/program-for-shortest-job-first-or-sjf-cpu-scheduling-set-1-non-preemptive/>

<https://www.geeksforgeeks.org/program-for-fcfs-cpu-scheduling-set-1/>

[https://www.geeksforgeeks.org/program-for-round-robin-scheduling-for-the-same-arrival-time/#](https://www.geeksforgeeks.org/program-for-round-robin-scheduling-for-the-same-arrival-time/)

<https://www.geeksforgeeks.org/priority-cpu-scheduling-with-different-arrival-time-set-2/>