Advanced Operating Systems (CSEN 383)
Project - 2: Process Scheduling Algorithms

Group - 2
Rishi Dixit, Julianna Dietrich, Bhoomika Ganapuram, and Chandan Dhamande

Intro

This project explores turnaround time, waiting time, response time, and throughput for several CPU process scheduling algorithms, including First-come First-served (FCFS), Shortest Job First (SJF), Shortest Remaining Time (SRT), Round Robin (RR), and both non-preemptive and preemptive Highest Priority First (HPF). Each algorithm was evaluated using simulated workloads over multiple runs to compare their efficiency, responsiveness, and fairness.

Requirements

The objective of this project is to design, implement, and evaluate multiple CPU scheduling algorithms through simulation. The system must generate a fixed workload of processes and measure scheduling performance using standard operating system metrics. The following requirements were implemented:

1. **Scheduling Algorithms**
   The simulator must implement the following six scheduling algorithms:
   - First Come First Serve (FCFS), non-preemptive
   - Shortest Job First (SJF), non-preemptive
   - Shortest Remaining Time (SRT), preemptive
   - Round Robin (RR) with a time quantum of 1
   - High Priority First (HPF), non-preemptive, using four priority queues
   - High Priority First (HPF), preemptive, using four priority queues with Round Robin (time quantum = 1) within each queue
2. **Process Generation**
   Each simulation run must generate a set of processes with randomly assigned arrival times, service times, and priority levels. All scheduling algorithms must operate on identical process sets to ensure fair comparison.

3. **Performance Metrics**
   For each scheduling algorithm, the simulator must compute:
   - Average Turnaround Time
   - Average Waiting Time
   - Average Response Time
   - Throughput
4. Each metric is averaged over five independent simulation runs.
5. **Priority Queue Statistics**
   For both HPF algorithms, the simulator must additionally compute average performance metrics and throughput for each priority queue, as well as overall system averages.
6. **Correctness and Consistency**
   All performance metrics must be computed using consistent definitions across all algorithms, and simulations must reset process state between runs to prevent cross-run interference.

Calculated Statistics Explained

Turnaround Time (TAT)
Turnaround time represents the total time a process spends in the system, measured from its arrival to its completion. It is computed as:

$$TAT = Completion\ Time - Arrival\ Time$$

Waiting Time (WT)
Waiting time is the total time a process spends waiting in the ready queue without executing on the CPU. It is derived from turnaround time as:

$$WT = Turnaround\ Time - Service\ Time$$

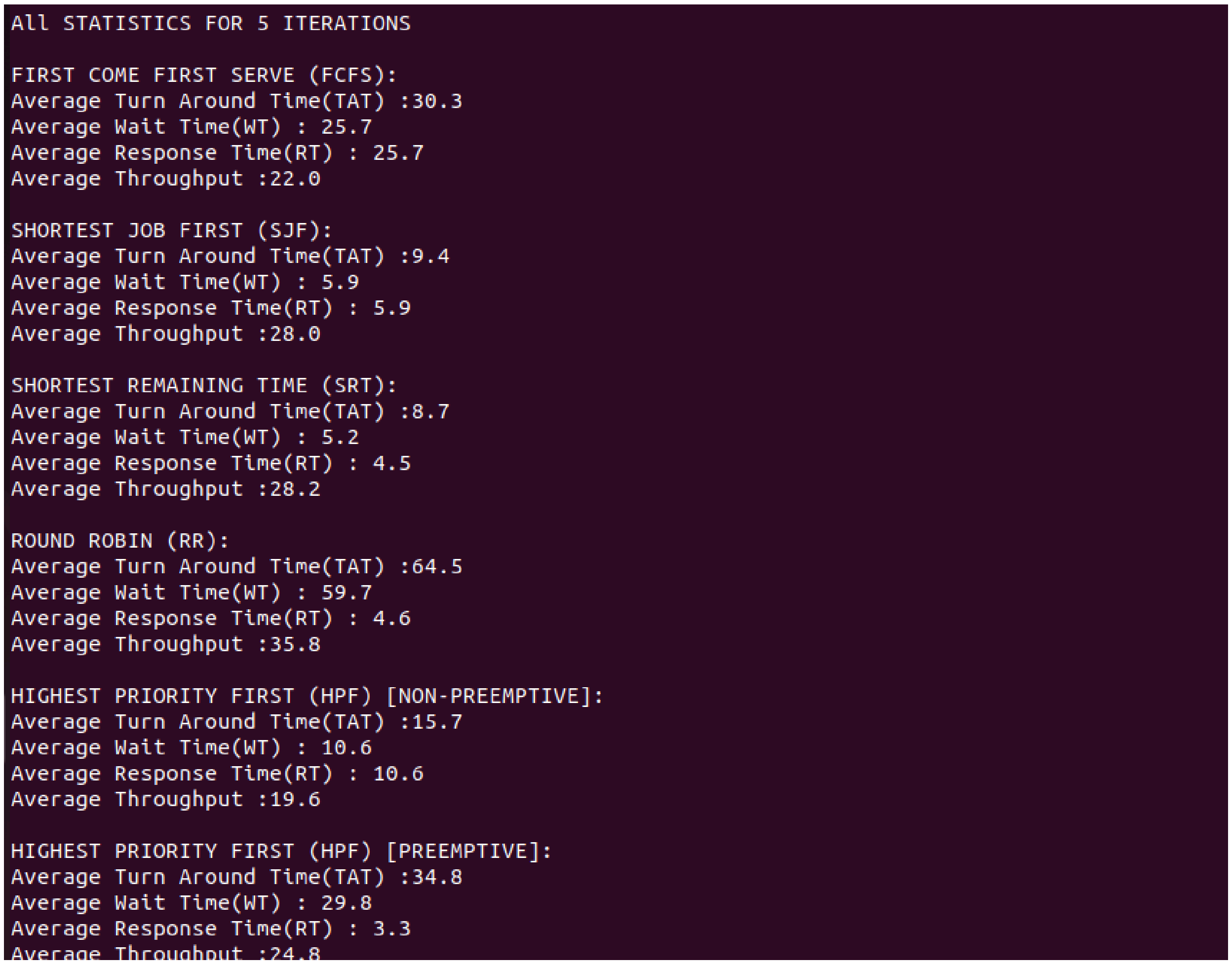
Response Time (RT)

Response time measures how quickly a process receives CPU service after arrival. It is defined as the time between a process's arrival and its first execution on the CPU:

$$RT = \text{First Run Time} - \text{Arrival Time}$$

Throughput
Throughput is the rate at which the system completes processes, expressed as the number of processes finished per unit of time. Higher throughput indicates greater system efficiency under a given workload

Results



```
All STATISTICS FOR 5 ITERATIONS

FIRST COME FIRST SERVE (FCFS):
Average Turn Around Time(TAT) :30.3
Average Wait Time(WT) : 25.7
Average Response Time(RT) : 25.7
Average Throughput :22.0

SHORTEST JOB FIRST (SJF):
Average Turn Around Time(TAT) :9.4
Average Wait Time(WT) : 5.9
Average Response Time(RT) : 5.9
Average Throughput :28.0

SHORTEST REMAINING TIME (SRT):
Average Turn Around Time(TAT) :8.7
Average Wait Time(WT) : 5.2
Average Response Time(RT) : 4.5
Average Throughput :28.2

ROUND ROBIN (RR):
Average Turn Around Time(TAT) :64.5
Average Wait Time(WT) : 59.7
Average Response Time(RT) : 4.6
Average Throughput :35.8

HIGHEST PRIORITY FIRST (HPF) [NON-PREEMPTIVE]:
Average Turn Around Time(TAT) :15.7
Average Wait Time(WT) : 10.6
Average Response Time(RT) : 10.6
Average Throughput :19.6

HIGHEST PRIORITY FIRST (HPF) [PREEMPTIVE]:
Average Turn Around Time(TAT) :34.8
Average Wait Time(WT) : 29.8
Average Response Time(RT) : 3.3
Average Throughput :24.8
```

Discussion

Turn Around Time (Best: SRT)

Shortest Remaining Time (SRT) achieves the best average Turn Around Time (TAT) because it is preemptive and always schedules the process with the shortest remaining CPU burst. This allows short jobs to complete quickly, even if they arrive while a longer process is executing. As a result, SRT improves responsiveness to short processes and outperforms Shortest Job First (SJF), its non-preemptive counterpart, as well as the other scheduling algorithms in terms of average turnaround time.

Wait Time (Best: SRT)

SRT also achieves the lowest average waiting time (WT). Its preemptive nature ensures that processes with shorter remaining execution times are not forced to wait behind longer jobs. This reduces the overall time processes spend waiting in the ready queue. By continuously favoring jobs closer to completion, SRT minimizes unnecessary waiting and provides the most efficient use of CPU time among the evaluated algorithms.

Response Time (Best: HPF (preemptive))

The best response time is achieved by preemptive High Priority First (HPF) scheduling. In this algorithm, newly arriving higher-priority processes immediately preempt lower-priority ones, allowing them to receive CPU service with minimal delay. Additionally, the use of Round Robin scheduling with a small time quantum within each priority queue ensures that processes of the same priority also receive prompt CPU access.

Non-preemptive HPF provides improved response time compared to FCFS and SJF, but it cannot match the responsiveness of preemptive HPF because once a process begins execution, it cannot be interrupted. This highlights the importance of preemption in latency-sensitive systems.

Throughput (Best: RR)

The highest throughput is observed in SRT and SJF. These algorithms prioritize shorter jobs, allowing more processes to complete in a given time interval. By minimizing average turnaround time, they maximize the rate at which jobs finish. In contrast, Round Robin and HPF introduce additional overhead through frequent context switching or priority enforcement, which slightly reduces throughput. introduce additional overhead through frequent context switching or priority enforcement, which slightly reduces throughput.

Tradeoffs: Fairness vs. Efficiency

Each scheduling algorithm exhibits different tradeoffs between fairness, efficiency, and responsiveness. SRT and SJF are highly efficient but may be unfair to long-running processes, which can experience starvation if short jobs continue to arrive. Round Robin improves fairness by ensuring that all processes receive CPU time, but this comes at the cost of higher waiting and turnaround times. HPF scheduling enforces priority fairness, ensuring that important tasks are serviced quickly, but may disadvantage lower-priority processes.

Starvation considerations

Starvation is most prominent in HPF scheduling, particularly in the preemptive variant. If higher-priority processes continue to arrive, lower-priority processes may experience indefinite postponement. SRT may also cause starvation for long processes in workloads dominated by short jobs. While starvation was not explicitly prevented in this project, real-world systems often mitigate this issue using aging techniques that gradually increase the priority of waiting processes.

Incorporating priority-based scheduling allowed for deeper analysis of fairness and starvation tradeoffs. Observing per-priority queue statistics in HPF demonstrated how preemptive priority enforcement improves responsiveness for critical tasks while negatively impacting lower-priority throughput. This reinforced the importance of balancing responsiveness and fairness when designing real-world schedulers.