Advanced OS (COEN383) Assignment - 2
Group #4

Robin Lee
Anand Santosh Kulkarni
Justin Li
Nikhil Kavuru
Pralhad Kulkarni

Statistics for CPU Scheduling Algorithms (over 5 runs)				
Scheduling Algorithm	Average Turnaround Time	Average Waiting Time	Average Response Time	Throughput (# jobs done in 99 time slices)
First-come first-served (FCFS) [non-preemptive]	51.56	45.0	45.0	7.23
Shortest job first (SJF) [non-preemptive]	19.72	13.17	13.17	14.96
Shortest remaining time (SRT) [preemptive]	19.27	12.72	11.58	14.96
Round robin (RR) [preemptive]	30.5	23.81	3.62	14.85
Highest priority first (HPF) [non-preemptive]	23.27	16.72	16.72	14.40
Highest priority first (HPF) [preemptive]	22.84	16.29	16.00	14.20

HPF algorithms were implemented with aging with 5 quanta interval

Observation of Scheduling Algorithms:

<u>**First-come first-served (FCFS) [non-preemptive] :**</u>

Advantages :
- Since the process that arrives first runs first, every process gets a chance to execute eventually.
- Deterministic behavior : Execution order of processes is predictable and follows FIFO.
- Simple and easy to understand.

Disadvantages :
- Convoy effect : Short processes must wait for the long one to complete, leading to inefficient resource utilization.
- Low throughput : FCFS doesn't take into account the burst time of processes, and it may lead to longer average waiting times compared to other scheduling algorithms.
- Longer turnaround times, as processes are executed in the order of arrival.
- Poor response times for interactive tasks, as they have to wait for long-running processes to finish before getting CPU time.

Analysis :
FCFS is most suitable for scenarios where simplicity and determinism are more critical than optimizing resource utilization or response time. It can be used effectively in embedded systems, single-user systems, and scenarios where processes have predictable and similar execution times.

<u>**Shortest job first (SJF) [non-preemptive]:**</u>

Advantages :
- It is optimal theoretically and probably gives the least waiting times as compared to FCFS
- It improves the output by offering processes with smaller burst times, which should be executed first and have a shorter turnaround time
- It is useful for batch processing of the jobs

Disadvantages :
- It may cause starvation as processes with smaller burst times keeps on coming and a process with larger burst time does not execute or has to delay its execution for longer time. This can be solved by aging
- It's practically impossible to implement as we don't know the burst times of the processes prior to their execution

Shortest remaining time (SRT) [preemptive]:

Advantages :

- **Optimal Turnaround Time**: SRTF minimizes the average turnaround time. It selects the shortest job that is ready to execute, which means that shorter jobs are given preference, leading to faster job completions on average.
- **Improved Responsiveness**: SRTF provides better responsiveness in interactive systems. Short jobs can get executed quickly, which can lead to a more interactive and responsive user experience.
- **Efficient Utilization of CPU**: SRTF maximizes CPU utilization by selecting the job with the shortest remaining time, which minimizes idle time. This results in efficient use of system resources.

Disadvantages :

- Implementing SRTF can be complex, especially in a multiprogramming environment. Maintaining a list of ready-to-execute jobs and constantly reevaluating their remaining times can be resource-intensive and may require efficient data structures.
- Longer jobs may suffer from starvation in SRTF. If a shorter job continually arrives, it can keep preempting longer jobs, preventing them from ever completing. This can lead to unfairness in resource allocation.

Round robin (RR) [preemptive]:

For this policy, all the processes get an equal time slice for execution. Due to this, all processes in queue get CPU for limited time and completion of long processes does not take place after CPU allocation in a single turn. This majorly increases the wait time of all processes, as well as the turnaround time. However, its main strength is in its response time, as the average is much lower than all of the other algorithms due to the constant switching between processes.

Highest priority first (HPF) [non-preemptive]:

According to the observations above, this policy is pretty decent in all categories. This algorithm lets higher-priority processes have a lower response time (better), but as a result, can starve lower-priority processes. Theoretically, this algorithm would also have a very high response time (bad) because it would starve processes. This problem was mitigated by adding aging to the algorithm, therefore making the algorithm look pretty good.

Highest priority first (HPF) [preemptive]:

Preemptive HPF does slightly better on every metric when compared to non-preemptive HPF except on turnaround time. This algorithm also had aging implemented. Overall, this is a pretty decent algorithm.

Observational Conclusions:

For best turnaround time, the Shortest remaining time (SRT) [preemptive] algorithm was the best with an average TAT of 19.27 time slices. For the best real-world implementable algorithm, the Highest priority first (HPF) [preemptive] algorithm was best with an average TAT of 22.84 time slices.

For best waiting time, the Shortest remaining time (SRT) [preemptive] algorithm was the best with an average waiting time of 12.72 time slices. For the best real-world implementable algorithm, the Highest priority first (HPF) [preemptive] algorithm was best with an average TAT of 16.29 time slices.

For best response time, the Round robin (RR) [preemptive] was the best with an average response time of 3.62 time slices.

For best overall throughput, the Shortest job first (SJF) [non-preemptive] and Shortest remaining time (SRT) [preemptive] algorithms tied for the best with an average throughput of 14.96 jobs done in the first 99 time slices. For the best real-world implementable algorithm, the Round robin (RR) [preemptive] algorithm was best with an average throughput of 14.85 jobs done in the first 99 time slices.