Advanced Operating Systems

COEN 383 Project - 5

Group 5

Snehal Nikam
Sarthak Patel
Bharadwaj Parthasarathy
Raksha Narasegowda
Smit Patel

Objective

This project primarily aims at exploring and evaluating various process scheduling methodologies. We delved into and coded the following algorithms in C:

- First come, first served (FCFS) [non-preemptive]
- Round robin (RR) [preemptive]
- Shortest job first (SJF) [non-preemptive]
- Shortest remaining time (SRT) [preemptive]
- Highest priority first (HPF) [preemptive]
- Highest Priority First (HPF) [Non-Preemptive]

Key Considerations

- Process attributes like Arrival Time, Expected Execution Duration, and priority are generated through a random algorithm.
- No consideration of I/O time in this model.
- We will be running each algorithm for a total of 100 quanta.
- Arrival time for the processes are set to be between 0 to 99.
- Time slice for round robin set to 1 quantum.
- A single process queue is in use.
- There are 4 priority queues used for HPF .

After applying the algorithms on a dataset of 52 processes, we derived statistical averages from five tests per algorithm. Below are the summarized outcomes:

The average of the 5 runs of all algorithms is as follows:

ALGORITHM 01: FIRST COME FIRST SERVE:

Average Response Time(RT) : 29.3
Average Wait Time(WT) : 29.7
Average Turn Around Time(TAT) :34.5
Average throughput(tr) :19.0

ALGORITHM 02: ROUND ROBIN [PREEMPTIVE]:

Average Response Time(RT) : 9.4
Average Wait Time(WT) : 107.0
Average Turn Around Time(TAT) :111.8
Average throughput(tr) :45.0

ALGORITHM 03: SHORTEST JOB FIRST [NON PREEMPTIVE]:

Average Response Time(RT) : 9.0
Average Wait Time(WT) : 9.4
Average Turn Around Time(TAT) :12.3
Average throughput(tr) :31.0

ALGORITHM 04: SHORTEST REMAINING TIME FIRST [PREEMPTIVE]:

Average Response Time(RT) : 8.0
Average Wait Time(WT) : 9.2
Average Turn Around Time(TAT) :12.0
Average throughput(tr) :31.0

```
ALGORITHM 05: HIGHEST PRIORITY FIRST [PREEMPTIVE]:

Average Response Time(RT) : 6.3
Average Wait Time(WT) : 8.4
Average Turn Around Time(TAT) :10.1
Average throughput(tr) :52.0
```

```
ALGORITHM: HIGHEST PRIORITY FIRST [NON PREEMPTIVE]:

Average Response Time(RT) : 17.6
Average Wait Time(WT) : 17.8
Average Turn Around Time(TAT) :21.3
Average throughput(tr) :17.0
```

Insights Derived

ALGORITHM 01: First Come First Serve (FCFS)

The First Come First Serve algorithm is based on the principle of serving processes in the order they arrive. As a straightforward scheduling technique, it is easily understandable and implementable. However, it may suffer from the "convoy effect," leading to longer average wait times for processes that arrive later. The primary advantage of FCFS is its simplicity, but it might not be efficient in scenarios with diverse process bursts. Compared to other algorithms like Round Robin or SJF, FCFS might yield longer wait times, especially if a lengthy process arrives early.

ALGORITHM 02: Round Robin (RR) [Preemptive]

Round Robin is a preemptive scheduling algorithm that assigns a fixed time quantum to processes in cyclic order. It is especially effective in time-sharing systems, ensuring each process gets a fair share of CPU time. However, the efficiency of RR is highly dependent on the length of the time quantum. A smaller time quantum can lead to excessive context switching, while a larger one might degrade RR to perform similarly to FCFS. When comparing with algorithms like SJF or SRTF, RR might exhibit higher average wait times but offers more predictability in execution.

ALGORITHM 03: Shortest Job First (SJF) [Non-Preemptive]

The Shortest Job First algorithm prioritizes tasks based on their execution time, processing the shortest tasks initially. By design, SJF can significantly reduce the average waiting time, especially in systems with diverse burst durations. However, a notable drawback is the potential for process starvation, especially for longer tasks in a queue dominated by shorter ones. The strength of SJF lies in its efficiency in

scenarios with a varied mix of short and long tasks. In comparison to algorithms like RR or FCFS, SJF might offer better average response and wait times, though at the risk of process starvation.

ALGORITHM 04: Shortest Remaining Time First (SRTF) [Preemptive]

SRTF is a preemptive variation of SJF, where the scheduler constantly reassesses the queue, prioritizing tasks with the shortest remaining execution time. This dynamic nature optimizes response and waiting times, adjusting swiftly to incoming processes. However, like SJF, it's susceptible to the risk of starving longer processes. While SRTF is good at real-time task management, it demands more computational overhead due to frequent recalibrations. When evaluated alongside non-preemptive counterparts like SJF, SRTF offers more agility but at the cost of increased complexity.

ALGORITHM 05: Highest Priority First (HPF) [Preemptive]

HPF operates on a hierarchical system, prioritizing processes based on predefined urgency levels. In its preemptive variant, it might preempt a running low-priority task if a higher-priority task enters the system. This ensures critical tasks are attended swiftly but might lead to starvation or extended wait times for lower-priority tasks. HPF's strength is in scenarios where processes have clear priority distinctions. In a comparative spectrum, the preemptive HPF is more dynamic and responsive than its non-preemptive version, ensuring critical tasks are not delayed.

ALGORITHM 06: Highest Priority First (HPF) [Non-Preemptive]

The non-preemptive version of HPF serves tasks based on priority without preemption. Once a process begins, it runs to completion regardless of the priority of incoming tasks. This approach offers predictability in execution but might not be optimal when high-priority tasks arrive during the execution of a lower-priority task. While the non-preemptive nature reduces overhead from context switches, it might delay critical processes. In comparison to its preemptive counterpart, non-preemptive HPF offers more stability but may not be as agile in responding to urgent tasks.

Conclusion

The choice of CPU scheduling algorithms often depends on the desired system output: Round Robin ensures fairness among processes, while Highest Priority First focuses on urgency-driven performance. Each algorithm has its strengths and trade-offs in terms of responsiveness, efficiency, and fairness. The challenge lies in selecting an algorithm that best matches a system's specific needs, optimizing for factors such as wait times, throughput, and overall system responsiveness.

From our execution we have seen that for the given random processes with multiple test (after executing code multiple time) the HIGHEST PRIORITY FIRST [PREEMPTIVE] algorithm has given the high throughput in all the test and provided low turnaround time, response time and wait time most of test runs.