CS 314 – Operating Systems Lab
Lab–4 Report

Name: Kavali sri vyshnavi devi **Roll-No:** 200010023

1 Explanation Of Scheduling Algorithms

In the assignment 2 scheduling algorithms are used namely Shortest Job First (SJF) and Round Robin (RR).

1.1 Setup

- On running **make** generates .out file files namely sjf.out and rr.out.
- Usage ./sjf ¡file¿ or ./rr ¡file¿.

1.2 Shortest Job First

Shortest Job First algorithm selects the process with the shortest execution time to be executed next from ready queue. It is non-preemptive scheduling algorithm, which means that once a process is given control of the CPU, it will run to completion before another process is allowed to execute.

- The process with the shortest burst time is given the highest priority and is executed first, if burst times are equal then jobs are prioritized on arrival times.
- Once it completes, the next process with the shortest burst time is executed, previously executed job is pushed into i/o queue as it goes into blocked state and so on.
- In the algorithm both i/o and cpu processes are executed simultaneously by maintaining current time, in the assignment there are many i/o bursts and cpu bursts for a process so in this case we need to prioritize on the next cpu burst time. In case of i/o as said there is only one device FCFS is used.

```
bool sort_order(const vector<int> &v1, const vector<int> &v2 {
return (v1[2] < v2[2]);
}
bool sort_equal(const vector<int> &v1, const vector<int> &v2) {
return ((v1[2] == v2[2]) && (input_array[v1[0]][0] < input_array[v2[0]][0]));
}
```

1.3 Round Robin

Preemptive Round Robin is a scheduling algorithm used in operating systems to divide CPU time fairly among multiple processes. The algorithm works by dividing the CPU time into equal time slices, called "quantums", and assigning each process a quantum.

- The CPU starts executing the process at front of the queue, and when its quantum expires, the process is moved to the end of the queue. This process continues until all processes have had their turn.
- The advantage of this algorithm is that it ensures that no single process monopolizes the CPU for too long, and that all processes get a fair share of CPU time. However, it can lead to higher overhead and context switching, as processes are constantly being interrupted and moved to the end of the queue.
- Here, the tie breaker condition comes when a process comes back to ready state after completing i/o burst and a process completed it's time slice and coming again to the end of the queue which is dealt by using a bit (0 or 1) to prioritize jobs that comes after completing i/o burst.
- On tie-braking situation if arrival times of the processes are same then newly arrived processes after i/o burst will be given higher priority by using a bit to be set or unset accordingly.

```
bool sort_priority(const vector<int> &v1, const vector<int> &v2) {
return (v1[1] == v2[1] && v1[4] > v2[4]);
}
```

2 Expected Job Characteristics

2.1 Shortest Job First

In SJF (Shortest Job First) scheduling, the job with the shortest execution time is selected from the ready queue and is executed first. The expected job characteristics for SJF scheduling are mentioned below.

- Since the job with the shortest execution time is selected first, jobs with shorter execution times are likely to be executed faster under SJF scheduling.
- SJF scheduling relies on accurate predictions of job execution times, so jobs with highly predictable execution times are well-suited for this scheduling algorithm.
- As the number of processes in ready queue increases, there will be lag in responding to new processes if they are long and leads to starvation and convoy affect.

In SJF algorithm when arrival times slightly differ and burst times vary much then it becomes like the job which arrived first will be executed which may have longer cpu burst time but the job which arrived slightly later with less burst time have to wait for a long time to get scheduled.

2.2 Round Robin

Round Robin algorithm is a preemptive scheduling algorithm which provide fair share of CPU to all the jobs which are in the ready queue by giving a certain time slice to execute the process. The expected job characteristics for Round Robin scheduling are mentioned below

- Since processes are given a small time slice, they are quickly moved to the ready queue, which results in low response time.
- Due to the small time slices given to processes, round robin scheduling may result in lower throughput compared to other scheduling algorithms.
- Round robin scheduling is a fair algorithm as it allocates equal time slices to all processes, independent on the burst time of processes.
- The scheduling algorithm may require a significant amount of overhead to maintain the process queue, which can impact system performance.
- Round robin scheduling is particularly suitable for interactive systems where responsiveness is critical.

3 Test Data To Bring Out The Suitability

3.1 Shortest Job First

For the below mentioned test case one can notice that the first process is executed first as there are no other jobs which arrived at time=0.

```
    0 3 -1
    1 2 -1
    2 1 -1
--------------------OUTPUT------------------
1 exited current time 3
3 exited current time 4
2 exited current time 6
-------------------------analytics------------------
Process ID 1 Response Time 0 Wait Time 0 Burst Time 3 Turn Around Time 3
Penalty Ratio 1
Process ID 2 Response Time 3 Wait Time 3 Burst Time 2 Turn Around Time 5
Penalty Ratio 2.5
Process ID 3 Response Time 1 Wait Time 1 Burst Time 1 Turn Around Time 2
Penalty Ratio 2
-------------------------system averages-----------------
avg res time 1.33 avg wait time 1.33 avg burst time 2
avg turnaround time 3.33333 avg penalty ratio 1.83
Throughput=0.5
```

Shortest Job First scheduling algorithm makes job-3 to execute after job-1 as it need lesser CPU time. This execution reduces turnaround time of all process. So here the average turn around time is 3.33 which is less than that in round robin (3.66 time_slice=3).

3.2 Round Robin

For the below mentioned test case one can notice that the response time is very low which means that round robin is good for interactive processes which need less cpu time.

```
    0 3 -1
    3 6 -1
    6 2 -1
--------------------OUTPUT------------------
1 current time 3 exited
3 current time 8 exited
2 current time 11 exited
-------------------------analytics------------------
Process ID 1 Response Time 0 Waiting Time 0 Turn Around Time 3 Completion Time 3
Arrival Time 0 Cpu Burst Time 3 Total Burst Time 3 Penalty Ratio 1
Process ID 2 Response Time 0 Waiting Time 2 Turn Around Time 8 Completion Time 11
Arrival Time 3 Cpu Burst Time 6 Total Burst Time 6
Penalty Ratio 1.33333
Process ID 3 Response Time 0 Waiting Time 0 Turn Around Time 2 Completion Time 8
Arrival Time 6 Cpu Burst Time 2 Total Burst Time 2 Penalty Ratio 1
-------------------------system averages------------------
Average Response Time 0 Average Turn Around Time 4.33333
Average Waiting Time 0.666667 Average Penalty Ratio 1.11111
Throughput 0.272727
```

Here the average response time is 0 which is very less as compared to SJF in which it is 1 as we are maintaining a quantum, depending on that the processes get faster response as process which is running will be preempted on being executed for time more than quantum.

4 Test Data To Bring Out The Shortcomings

4.1 Shortest Job First

The shortcoming for non-preemptive shortest job first scheduling algorithm is that when a long job executing and in mean time if any new job arrives with lesser remaining time to execute it cannot be scheduled till the currently running job is terminated which leads to convoy affect here too.

```
    0 6 -1
    1 2 -1
```

```
    2 1 -1
--------------------OUTPUT-----------------
1 exited current time 6
3 exited current time 7
2 exited current time 9
-------------------------analytics-----------------
Process ID 1 Response Time 0 Wait Time 0 Burst Time 6 Turn Around Time 6
Penalty Ratio 1
Process ID 2 Response Time 6 Wait Time 6 Burst Time 2 Turn Around Time 8
Penalty Ratio 4
Process ID 3 Response Time 4 Wait Time 4 Burst Time 1 Turn Around Time 5
Penalty Ratio 5
-------------------------system averages-----------------
avg res time 3.33 avg wait time 3.33 avg burst time 3
avg turnaround time 6.33 avg penalty ratio 3.33
Throughput 0.33
```

Here, job-1 is executing as it arrived at time-0 and no other processes are present then, where as in case of job-2 and job-3 which need cpu for lesser time will not be executed until job-1 gets terminated which leads to higher turn around times.

4.2 Round Robin

In this case, when time slice is high(let it be 10) then response time will be less and later it's execution becomes similar to FCFS which leads to convoy affect, if time slice is less then context-switches increase it leads to high overhead and more complexity in restoring state when it returns back to a process. Here time slice value is 10.

```
    0 6 -1
    1 1 -1
    2 3 -1
--------------------OUTPUT-----------------
0 current time 6 exited
1 current time 7 exited
2 current time 10 exited
-------------------------analytics-----------------
Process ID 1 Response Time 0 Wait Time 0 Burst Time 6 Turn Around Time 6
Penalty Ratio 1
Process ID 2 Response Time 5 Wait Time 5 Burst Time 1 Turn Around Time 6
Penalty Ratio 6
Process ID 3 Response Time 5 Wait Time 5 Burst Time 3 Turn Around Time 8
Penalty Ratio 2.66
-------------------------system averages-----------------
avg res time 3.33 avg wait time 3.33 avg burst time 3.33
avg turnaround time 6.66 avg penalty ratio 3.22
Throughput 0.3
```

In Round-Robin algorithm turn around times increase if there are processes with uniform burst times also. Here as time slice is high so they are executed in the order of arrival time and so job-1 is executed first, later job-2 is scheduled and then job-3 is executed.

5 Performance Analysis

5.1 Performance Analysis

From the below picture which is showing system averages one can say that response time of round robin algorithm is very less as compared to outputs from sjf algorithm where as turn around times of round robin algorithm are high as round robin compensates it with lower response time.

```
      sjf
------------------

PROCESS 1 - Average Response Time 324.429 Average Turn Around Time 755.286 Average Waiting Time 551.429 Average Penalty Ratio 7.59108

PROCESS 2 - Average Response Time 118.611 Average Turn Around Time 215.222 Average Waiting Time 162.056 Average Penalty Ratio 10.1165

PROCESS 3 - Average Response Time 428.333 Average Turn Around Time 912.833 Average Waiting Time 738.75 Average Penalty Ratio 5.82961


      rr
--------------------
PROCESS 1 - Average Response Time 7.28571 Average Turn Around Time 874.143 Average Waiting Time 670.286 Average Penalty Ratio 5.03014

PROCESS 2 - Average Response Time 7.28571 Average Turn Around Time 874.143 Average Waiting Time 670.286 Average Penalty Ratio 5.03014

PROCESS 3 - Average Response Time 14.5833 Average Turn Around Time 1245.33 Average Waiting Time 1071.25 Average Penalty Ratio 9.01827
```

Figure 1: Results

5.2 Graphs

From the graphs generated shown one can notice that the round robin scheduling algorithm have less response time as compared to response time of shortest job first algorithm as round robin scheduling is preemptive which context switches to another process on completion of time slice.

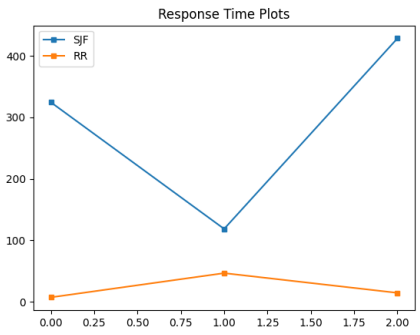

Figure 2: Response Time Graphs

Coming to waiting time plots one can notice that waiting time for round robin algorithm is higher as if there are processes with uniform burst times then depending on the time slices there waiting around time increases as if one process completes it's execution then it comes to the front of the queue which and all the other processes gets their chance to execute later so the waiting time of process which got executed just now increases and it leads to higher waiting time. In sjf algorithm if a process is executing then it will not be interrupted in between and so the waiting time is according to their burst times.

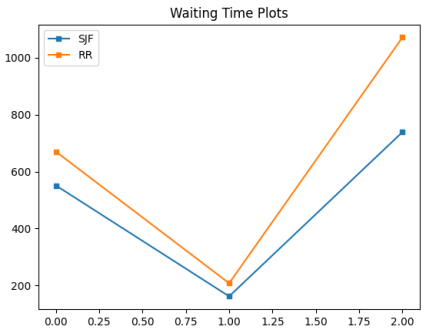

Figure 3: Waiting Time Graphs

By delivering a faster average response time, RR provides a fix to the fairness problem. However, that usually results in a longer turnaround time on average. In the result, there is a trade-off between fairness in terms of response time and the average turnaround time. RR (time slice is 3) is implemented to avoid starvation of process which we encounter in sjf.

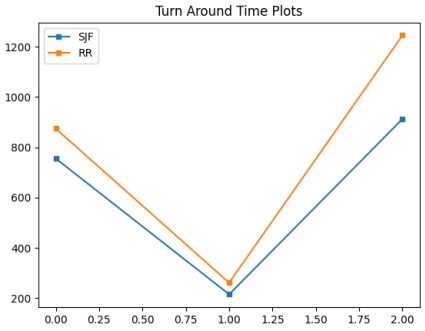

Figure 4: Turn Around Time Graphs

In Round Robin (RR), the penalty ratio can be high if the time quantum (time slice) used is small. In the assignment the time slice used is 3 which lead to high penalty ratios in process2.data and process3.dat files. In the 1st test case processes have varying processing times, and shorter processes arrive frequently so the penalty ratio for rr is less as compared to sjf.

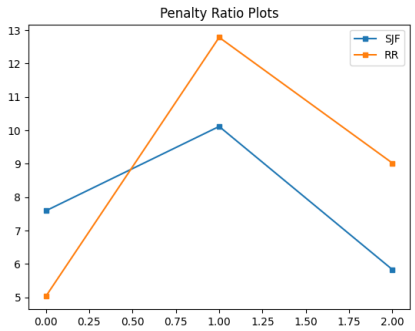

Figure 5: Penalty Ratio Graphs

The throughput for Round Robin (RR) and Shortest Job First (SJF) can be similar because both algorithms have their strengths and weaknesses. RR can provide good utilization of CPU time by allocating equal time slices to each process. However, it may not be as efficient as SJF in terms of completion time because it is based on fixed time slices and can result in longer wait times for shorter processes. SJF, on the other hand, has the advantage of optimizing the completion time by executing the shortest process first. However, it can result in longer wait times for longer processes and can be susceptible to process starvation if a new, shorter process arrives while a longer process is running.

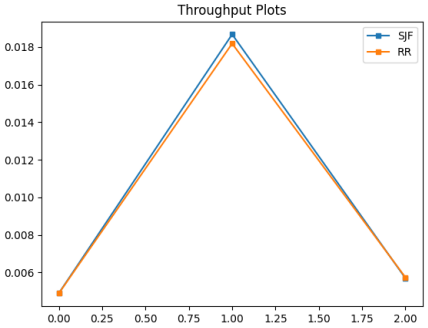

Figure 6: Throughput Graphs

Here, the test cases shows that turn around of sjf is less compared to rr can be observed by executing the test case created which is named as **sjf_good_test.dat** which can also be observed from the below plot shown.

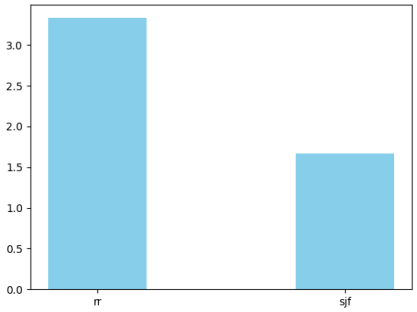

Figure 7: Sjf less turn around time

Here, the fact that response time of rr is less compared to sjf can be observed by executing the test case created which is named as **rr_good_test.dat** which can also be observed from the below plot shown.

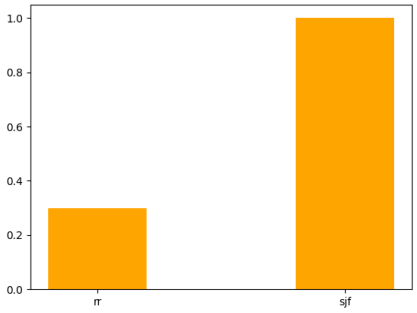

Figure 8: Rr less response time