OS Lab - 4

Abhishek Josyula - 200010021

Usage:

1. `make SJF` compiles SJF `scheduler` to bin/scheduler
2. `make RR` compiles RR `scheduler` to bin/scheduler
3. `make clean` cleans the directory
4. Requirements - g++
5. bin/scheduler data/process.dat prints each process data in RR_process.txt or SJF_process.txt

SHORTEST JOB FIRST:

1. Process having the smallest execution time (CPU burst time) present in the ready queue is chosen for the next execution.
2. Process executes until completion.
3. No preemption

Explanation of scheduling policy:

1. IF the arrival time of the process in Arriving Que is less than the current time it is pushed to readyQ.
2. At each loop scheduler selects the process with lowest burst time.
3. This process runs until completion of its burst.
4. The IO Que follows the FCFS mechanism.
5. When the arriving Que and Ready Queue is empty the scheduler terminates.

OUTPUT by the program:

The output of my program is a text file containing the data for each file.

1. The first line contains pid.
2. The second line contains arrivaltime.
3. The third line contains TotalBurst.
4. The fourth line contains TotalIo.
5. The fifth line contains completion time.
6. The sixth line contains the first executed time.
7. The seventh line contains turn around time.
8. The eighth line contains response time.
9. The ninth line contains penalty ratio.
10. The tenth line contains waiting time.
11. Remaining lines contain time slice start and end.

Round Robin Scheduler:

1. The first job in readyQ is selected and run until its time slice runs out or it is blocked by IO.
2. The scheduler is preemptive in nature and it has the capability to context switch

Explanation of scheduling policy:

1. IF the arrival time of the process in Arriving Que is less than the current time it is pushed to readyQ.
2. Time slice is set at 50ms.
3. Scheduler picks the first job in readyQ and executes it.
4. If job runs out of time slice it is pre empted and pushed back to ReadyQ
5. If a job is blocked by IO it is pushed to IOQ.
6. After the job completes its IO it is pushed back to IOQ.
7. The IO Que follows the FCFS mechanism.
8. The scheduler stops when all IOQ readyQ and ArrivinQ are empty.

Analysis

SJF

Advantages of SJF:

1. Reduces average waiting time
2. Useful when running time are known before.
3. Optimal with regard to average turnaround time.

Disadvantages of SJF:

1. It is necessary to know the job completion time beforehand as it is hard to predict.
2. Non preemptive
3. Starves long running process.
4. Not a real world scheduling policy

RR

Advantages of RR:

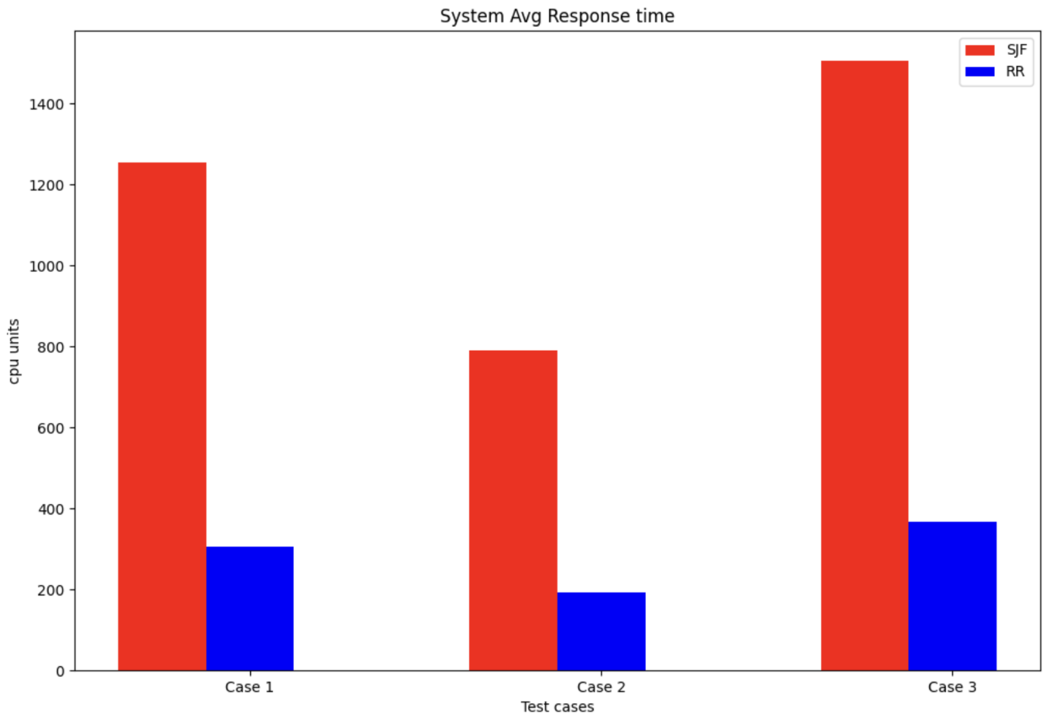
1. No issues of starvation or convoy effect.
2. Every job gets a fair allocation of CPU.
3. Doesn’t depend on burst time and is easily implementable.
4. Best performance in terms of average response time.

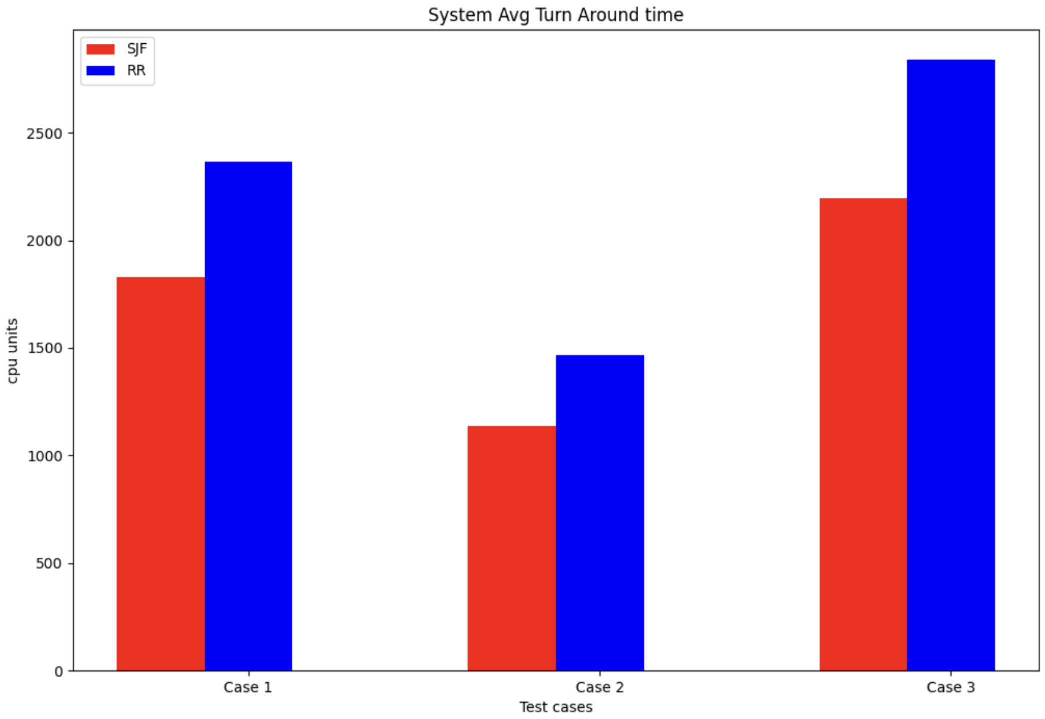
Disadvantages of RR:

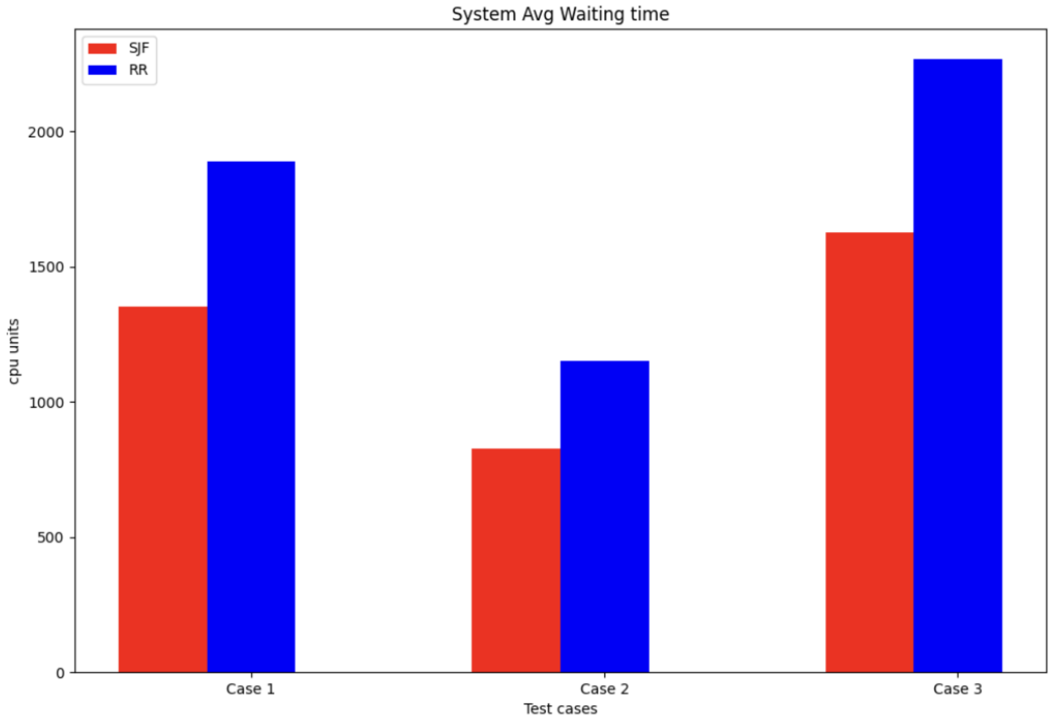
1. Low slicing time reduces processor output
2. Processes don’t have priorities.
3. Context Switch Overhead
4. Difficult to find a correct time quantum

Observations

1. SJF Optimal with regard to average turnaround time.
2. SJF Reduces average waiting time
3. RR doesnot have starvation.
4. RR has best performance in terms of average response time.




System Avg Turn Around time
SJF
RR
cpu units
2500
2000
1500
1000
500
0
Case 1
Case 2
Case 3
Test cases


System Avg Waiting time
SJF
RR
cpu units
0
500
1000
1500
2000
Case 1
Case 2
Case 3
Test cases