Assignment 3

Parikshit Gehlaut 220150009
Vemula Chandrahaas Reddy 220010062
Soumyadeep Das 220010056

1 Introduction

This report explores the implementation and performance evaluation of several process scheduling algorithms: First In First Out (FIFO), Non-preemptive Shortest Job First (SJF), Preemptive Shortest Job First (SRTF), and Round Robin (RR). The purpose of this assignment is to simulate these scheduling algorithms and analyze their performance based on various metrics such as Makespan, Average and Maximum Completion Time, and Waiting Time. Additionally, the effect of using two processors instead of one is evaluated by comparing the results of both single-core and dual-core simulations.

2 Methodology

The input data consists of workload description files, where each line represents a process and follows this format:

```
<process-arrival-time> <cpu-burst-1-duration> <io-burst-1-duration> ... -1
```

The scheduling simulator reads these files and schedules the processes according to the algorithm specified at runtime.

2.1 Algorithms

We implemented the following scheduling algorithms:

- **FIFO**: Processes are executed in the order of their arrival time.
- **SJF**: The process with the shortest CPU burst is selected for execution (non-preemptive).
- **SRTF**: The process with the shortest remaining time is selected (preemptive).
- **Round Robin**: Each process is given a fixed time quantum, after which it is rotated to the end of the queue.

3 Implementation

The simulator uses queues to manage processes in various states (ready, waiting). For dual processor scheduling, the processes are distributed across two CPUs based on availability.

The key metrics measured include:

- **Makespan**: The total time taken to complete all processes.
- **Completion Time**: The time at which a process completes execution.
- **Waiting Time**: The total time a process spends waiting in the ready queue.
- **Elapsed Time**: The total elapsed time for each algorithm (in milliseconds).

4 Results

The performance of each algorithm in both single-core and dual-core simulations is summarized in Table 1. We also provide visual comparisons through the graphs in Figures 1, 2, and 3.

Table 1: Scheduling Algorithms Metrics (Single-core vs Two-core) wrt process1.dat

Algorithm	Core	Makespan	Avg Completion Time	Max Completion Time	Avg Waiting Time	Max Waiting Time	Elapsed Time (ms)
FIFO	Single	1427	1091	1423	880	1222	590
FIFO	Dual	723	533	723	322	483	447
SJF	Single	1429	755	1419	544	1214	581
SJF	Dual	720	444	720	233	503	475
SRTF	Single	1429	610	1419	399	1214	594
SRTF	Dual	720	363	720	152	505	443
RR	Single	1427	1053	1427	842	1123	699
RR	Dual	745	509	745	298	408	453

4.1 Completion Times

Figure 1 shows the average and maximum completion times across different algorithms for both single-core and dual-core simulations.

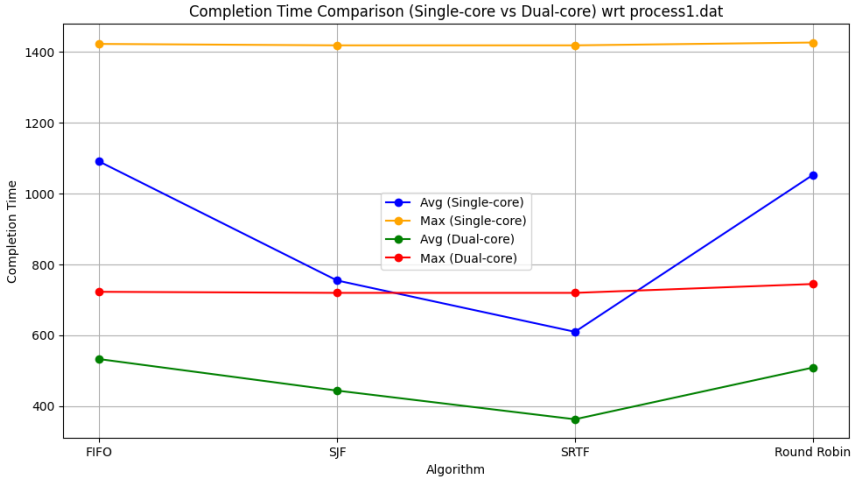

Figure 1: Completion Times Across Algorithms (Single-core vs Dual-core)

4.2 Waiting Times

The average and maximum waiting times are compared in Figure 2 for both core configurations.

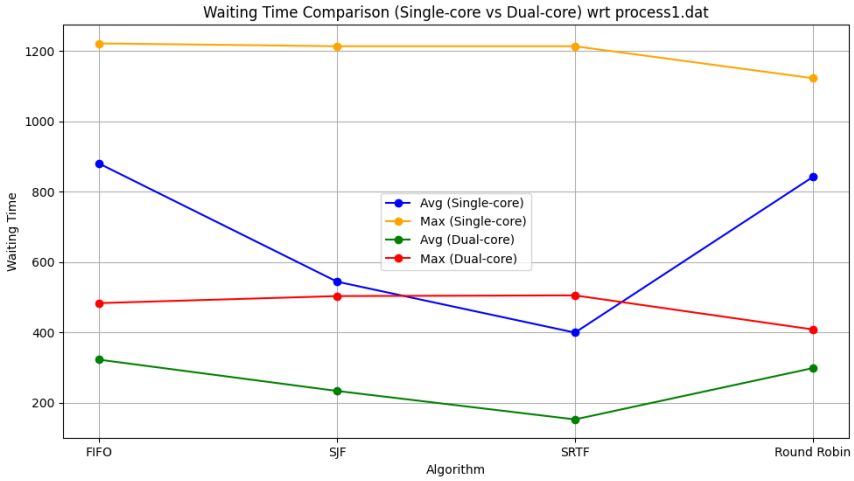

Figure 2: Waiting Times Across Algorithms (Single-core vs Dual-core)

4.3 Makespan

The overall makespan for each algorithm is shown in Figure 3 for single-core and dual-core comparisons.

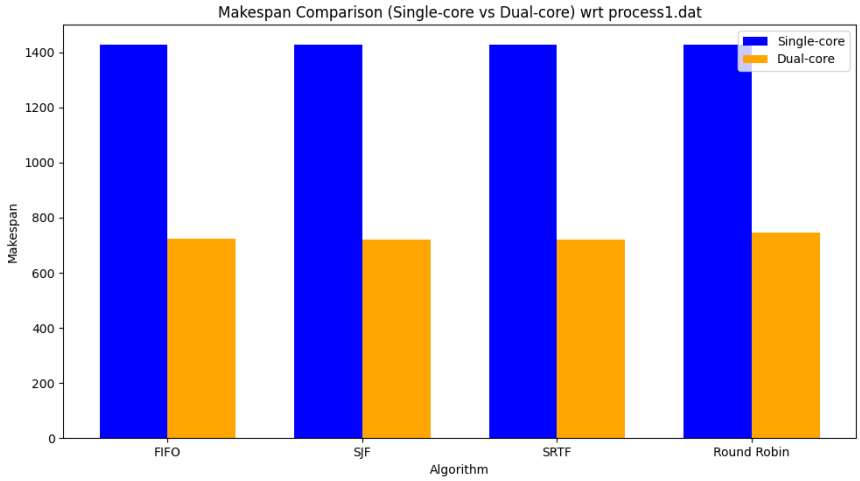

Figure 3: Makespan Across Algorithms (Single-core vs Dual-core)

5 Performance Comparison with Varying Time Quantum

In this section, we analyze the effect of varying time quanta on the performance of the Round Robin (RR) scheduling algorithm. The experiments were conducted with process1.dat, and the results are compared across time quanta of 10, 20, 30, and 40 units. The key metrics compared are the Makespan, Average Completion Time, Maximum Completion Time, Average Waiting Time, and Maximum Waiting Time.

5.1 Single-core Results

The performance of the Round Robin algorithm with varying time quanta in a single-core simulation is summarized in Table.

Table 2: Scheduling Metrics for Varying Time Quantum (Single-core)

Time Quantum	Makespan	Avg Completion Time	Max Completion Time	Avg Waiting Time	Max Waiting Time
10	1431	885	1431	674	1010
20	1429	931	1429	720	1014
30	1429	962	1429	751	1053
40	1429	991	1429	780	1023

As observed, the Makespan remains almost constant with varying time quantum, while the Average Waiting Time increase as the time quantum increases. This is because with a larger quantum, processes spend more time in the CPU before being rotated, which leads to longer waiting times for other processes.

5.2 Dual-core Results

Similarly, the performance of the Round Robin algorithm with varying time quanta in a dual-core simulation is presented in Table.

Table 3: Scheduling Metrics for Varying Time Quantum (Dual-core)

Time Quantum	Makespan	Avg Completion Time	Max Completion Time	Avg Waiting Time	Max Waiting Time
10	760	445	760	234	339
20	757	462	757	251	359
30	750	477	750	266	369
40	747	488	747	277	378

In the dual-core case, the Makespan also remains nearly constant, but there is a noticeable increase in the Average Waiting Time as the quantum increases, though these values are lower compared to the single-core results. This is due to the more efficient use of the two cores, which allows processes to be scheduled concurrently, reducing contention for CPU time.

6 Running The code

- **For part1** make build: Running this command will create executable schedular.out
  make run : Running this command will run schedular for all four algorithms i.e. FIFO, SJF, SRTF and Round Robin for all three test cases.
- **For part2** make build_two: Running this command will create executable schedular.out
  make run : Running this command will run schedular for all four algorithms i.e. FIFO, SJF, SRTF and Round Robin for all three test cases.

7 Conclusion

In this assignment, we simulated four process scheduling algorithms and compared their performance in both single-core and dual-core environments. The results show that using two processors significantly improves performance by reducing waiting times and makespan. SRTF performed best in terms of average waiting time and completion time. Using dual-core scheduling provided considerable improvements in Round Robin scheduling as well.