CS 4310 Operating Systems
Project
Simulating Job Scheduler and Performance Analysis

Description:

Simulating Job Scheduler of the Operating Systems by programming the following four scheduling algorithms that we covered in the class:

a. First-Come-First-Serve (FCFS)
b. Shortest-Job-First (SJF)
c. Round-Robin with Time Slice = 2 (RR-2)
d. Round-Robin with Time Slice = 5 (PR-5)

The project generates multiple testing cases with inputs of 5 jobs, 10 jobs, and 15 jobs, 20 cases each, for 60 files. The program will read process burst times from a file, job.txt, and execute the four algorithms above. A sample input file of five jobs is given as follows (burst time in ms). The program displays four algorithms job processes in a table format and calculates the average turnover time(ATT). It compares the mean of ATT for each job group.

```
[Begin of job.txt]
Job1
7
Job2
18
Job3
10
Job4
4
Job5
12
[End of job.txt]
```

Note: you can assume that

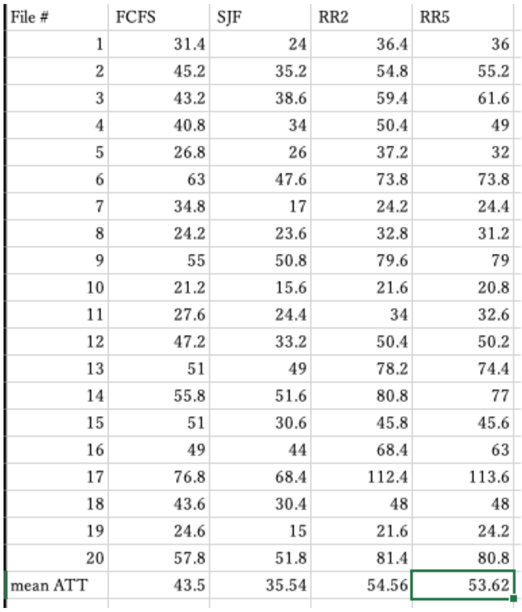
1. There are no more than 30 jobs in the input file (job.txt).
2. Processes arrive in the order they are read from the file for FCFS, RR-2 and RR-5.
3. All jobs arrive at time 0.
4. FCFS uses the order of the jobs, Job1, Job2, Job3, …

Part 3
Performance Analysis (60 points)

a. Run each program with the designed randomly generated input data given in Part 1(c). Generate a table for all the experimental results for performance analysis as follows.

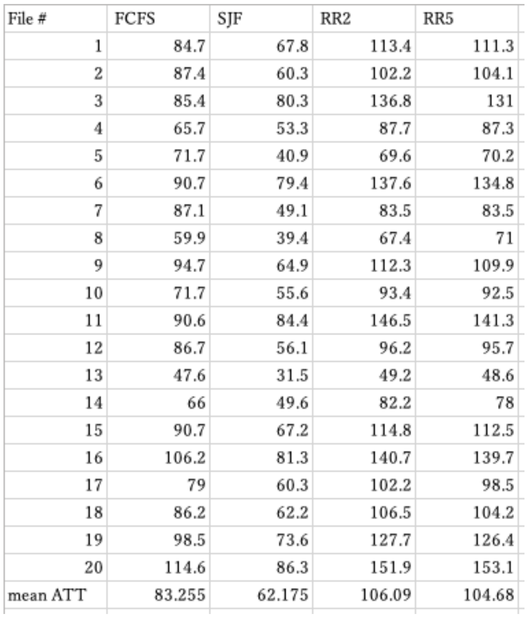
Input Size n jobs	Average of average turnaround times (FCFS Program)	Average of average turnaround times (SFJ Program)	Average of average turnaround times (RR-2)	Average of average turnaround times (RR-5)
5 jobs	43.5	35.54	54.56	53.62
10 jobs	83.255	62.175	106.09	104.68
15 jobs	123.183333	88.7833333	157.69	157.136667

5Jobs Average Turnover Time table.



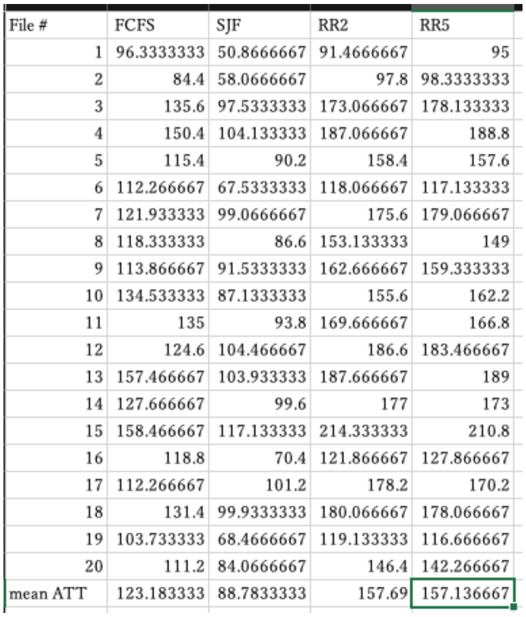
File #	FCFS	SJF	RR2	RR5
1	31.4	24	36.4	36
2	45.2	35.2	54.8	55.2
3	43.2	38.6	59.4	61.6
4	40.8	34	50.4	49
5	26.8	26	37.2	32
6	63	47.6	73.8	73.8
7	34.8	17	24.2	24.4
8	24.2	23.6	32.8	31.2
9	55	50.8	79.6	79
10	21.2	15.6	21.6	20.8
11	27.6	24.4	34	32.6
12	47.2	33.2	50.4	50.2
13	51	49	78.2	74.4
14	55.8	51.6	80.8	77
15	51	30.6	45.8	45.6
16	49	44	68.4	63
17	76.8	68.4	112.4	113.6
18	43.6	30.4	48	48
19	24.6	15	21.6	24.2
20	57.8	51.8	81.4	80.8
mean ATT	43.5	35.54	54.56	53.62

10Jobs Average Turnover Time table.



File #	FCFS	SJF	RR2	RR5
1	84.7	67.8	113.4	111.3
2	87.4	60.3	102.2	104.1
3	85.4	80.3	136.8	131
4	65.7	53.3	87.7	87.3
5	71.7	40.9	69.6	70.2
6	90.7	79.4	137.6	134.8
7	87.1	49.1	83.5	83.5
8	59.9	39.4	67.4	71
9	94.7	64.9	112.3	109.9
10	71.7	55.6	93.4	92.5
11	90.6	84.4	146.5	141.3
12	86.7	56.1	96.2	95.7
13	47.6	31.5	49.2	48.6
14	66	49.6	82.2	78
15	90.7	67.2	114.8	112.5
16	106.2	81.3	140.7	139.7
17	79	60.3	102.2	98.5
18	86.2	62.2	106.5	104.2
19	98.5	73.6	127.7	126.4
20	114.6	86.3	151.9	153.1
mean ATT	83.255	62.175	106.09	104.68

15Jobs Average Turnover Time table.

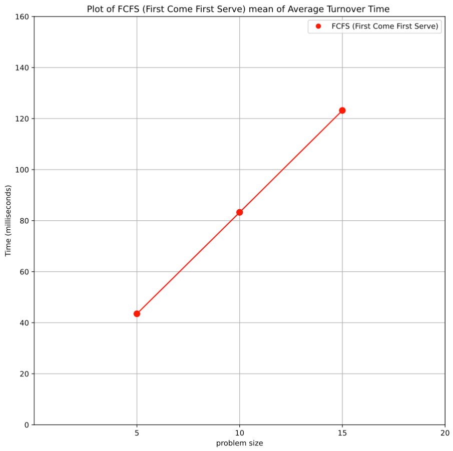


File #	FCFS	SJF	RR2	RR5
1	96.3333333	50.8666667	91.4666667	95
2	84.4	58.0666667	97.8	98.3333333
3	135.6	97.5333333	173.066667	178.133333
4	150.4	104.133333	187.066667	188.8
5	115.4	90.2	158.4	157.6
6	112.266667	67.5333333	118.066667	117.133333
7	121.933333	99.0666667	175.6	179.066667
8	118.333333	86.6	153.133333	149
9	113.866667	91.5333333	162.666667	159.333333
10	134.533333	87.1333333	155.6	162.2
11	135	93.8	169.666667	166.8
12	124.6	104.466667	186.6	183.466667
13	157.466667	103.933333	187.666667	189
14	127.666667	99.6	177	173
15	158.466667	117.133333	214.333333	210.8
16	118.8	70.4	121.866667	127.866667
17	112.266667	101.2	178.2	170.2
18	131.4	99.9333333	180.066667	178.066667
19	103.733333	68.4666667	119.133333	116.666667
20	111.2	84.0666667	146.4	142.266667
mean ATT	123.183333	88.7833333	157.69	157.136667

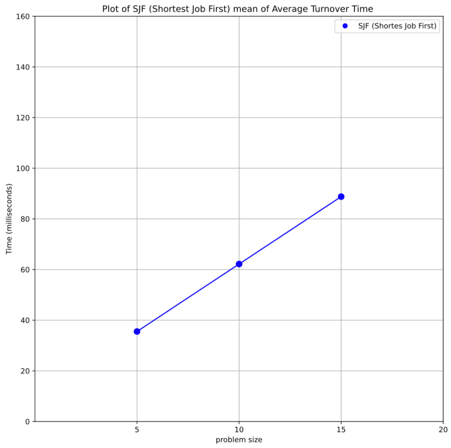
b. Plot a graph of each algorithm, average turnaround time vs input size (# of jobs), and summarize the performance of each algorithm based on its own graph.

<Insert totally four graphs, one for each program, here>

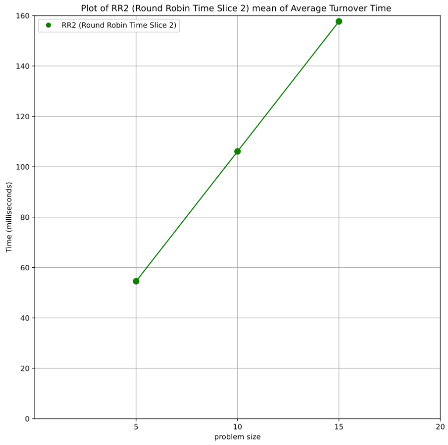
1. FCFS



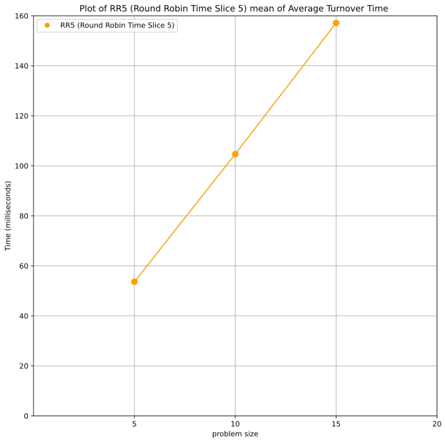
2. SJF



3. RR2



4. RR5



<Write a summary>

1. First Come First Serve (FCFS):

   As the number of jobs increases from 5 to 15, the mean ATT also increases linearly. It can be expected easily since the queue gets longer, and jobs have to wait more. As we discussed in class, FCFS generally has a higher ATT than Shortest Job First (SJF), indicating that jobs with longer processing times can increase the waiting time for subsequent jobs.

2. Shortest Job First (SJF):

   It consistently shows the lowest mean ATT across all scenarios as discussed in class. It reduces the amount of time shorter jobs have to wait. The growing execution time is the slowest among the four algorithms. It may imply SJF is effective when there are many short jobs mixed with longer jobs, as it minimizes the overall waiting time.

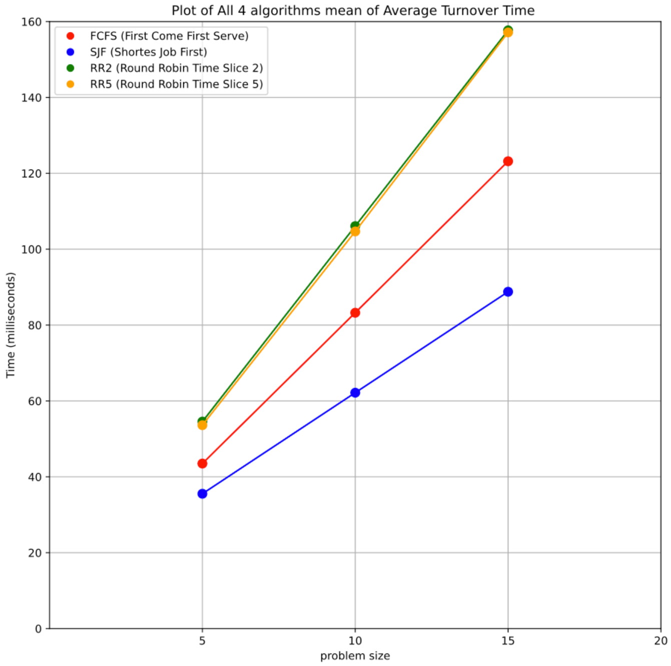
3. Round Robin (RR) with Time Slice 2:

   The mean ATT is higher compared to SJF and FCFS, especially as the number of jobs increases. It can be due to the overhead of context switching and the fact that longer jobs will take more cycles to complete. The time slice of 2 might be too short for some jobs, leading to increased overhead.

4. Round Robin (RR) with Time Slice 5:

   I expected RR5 would be more efficient than RR2 with a larger time slice. Jobs have more time to execute before being preempted, which can be more efficient for longer jobs. But it can also lead to longer waiting times for shorter jobs if they arrive behind longer jobs. The mean ATT for RR with a time slice of 5 is slightly lower than that of a time slice of 2, indicating a possible reduction in context switching overhead. However, it's still higher than SJF and FCFS.

Plot all four graphs on the same graph and compare the performance of all four algorithms. Rank four scheduling algorithms. Try giving the reasons for the findings. <Insert four-graphs-in-one graph here> <Write about explaining the results>



From the observations above and the compassion four algorithms' execution time on the same plot, we can infer the following: SJF is the most efficient in terms of average turnaround time across the board. It may be effective when many short jobs are mixed with longer ones. FCFS is predictable, and short-term jobs would be stuck waiting behind long-term jobs. Round Robin is generally fairer regarding CPU time distribution but can have higher average turnaround times. But we cannot see much difference between time slices 2 and 5.

RR2 and RR5 show a steeper ATT increase compared to SJF and FCFS. This suggests that as the problem size increases, the effect of context switching causes inefficiency in handling longer jobs and higher ATT. Comparing RR2 and RR5, we can see that RR5 generally has a slightly lower ATT than RR2. It might come from the longer time slice in RR5, which reduces the frequency of context switches, which can be particularly costly in terms of time, and improves the ATT slightly. However, both versions of Round Robin have higher ATT than SJF and FCFS as the problem size increases.

As the problem size grows from 5 to 15, the gap in performance between the algorithms also increases. It shows that the choice of algorithm becomes more critical as the number of jobs scales up. The consistent performance of SJF across different problem sizes highlights its efficiency.

The plot shows that SJF is the most efficient algorithm in this circumstance. However, the result highly depends on the experimental environment, such as hardware components, number of jobs, and time length of jobs. The choice of scheduling algorithm should be based on the specific characteristics of the workload.

c. Conclude your report with the strength and constraints of your work. At least 100 words.

(Note: It is reflection of this project. If you have a change to re-do this project again, what you like to keep and what you like to do differently in order get a better quality of results.)

<Write a conclusion about strength and constraints of your work here.>

Implementing and analyzing the scheduling algorithms discussed in our class has been a significant experience. The plots show that while SJF records are the most efficient in our simulations, their performance may vary in different computational environments or with different job arrival patterns. The project did not include priority-based scheduling algorithms, which could add another interpretation to algorithm analysis.

In future iterations of this project, I would include a variety of round-robin time slices, such as 50 and 100, to examine how the length of time slice affects algorithm performance further. Also, I could generate job batches that consist exclusively of short or long tasks to investigate how job length influences scheduling efficacy. These modifications and varying the mix of job lengths in the input files can project additional insights and enhance our understanding of scheduling algorithm characteristics.