Dylan Kapustka (dlk190000)

Professor Ozbirn

CS 4348.001 – S21

04/31/2021

Project Summary

**Implementation**

This program was written in Java in order to utilize the object-oriented nature of the language. I decided upon the implementation of five main classes:

Main.java

MyJob.java

FCFS.java

SPN.java

HRRN.java

The main class, Main.java, handles reading and parsing the input file (jobs.txt) and instantiating a new instance of each algorithm. The input file is parsed as a new instance of MyJob.

MyJob.java is an object created to hold the data we gather from jobs.txt, such as arrival time, duration, and job name. This class also handles printing the output graph. For simplicity, each individual algorithm handles the printing of its name above the output graph.

The FCFS.java class implements the First Come, First Served algorithm, which is very similar to a first-in, first-out queue; the selection function is done strictly by finding the max wait time and ordering jobs accordingly.

The SPN.java class implements the Shortest Process Next algorithm and works by calculating which jobs have the shortest expected processing time; the selection function is done by finding the minimum service time required.

The HRRN.java class implements the Highest Response Ratio Next algorithm and works by calculating the ratio of turnaround time to actual service time; the selection function is done by computing the max of the wait time plus the service time divided by the service time.

**Personal Experience**

This project helped expand my understanding of different scheduling algorithms and what is required to utilize them.

The hardest part of this project was aligning the arrival times and duration accordingly with each algorithm. While the FCFS algorithm was easy since it is purely a queuing scheme, the other two algorithms were trickier since we had to keep track of jobs jumping around to the head of the queue past longer jobs.

Something I found interesting with this project was the realization that programmers will not always have access to the information needed to implement these algorithms. For example, the SPN algorithm requires the user to already know the required processing time of each process. Another example is the HRRN algorithm, where the service time needs to be approximated.

**End Result**

Given a jobs.txt file containing the following jobs (the example in the book) in Figure 1, we are given the output in Figure 2.

A 0 3

B 2 6

C 4 4

D 6 5

E 8 2

**Figure 1.** The jobs.txt file contents.

Text

Description automatically generated

**Figure 2.** The output after running my implementation of each algorithm.

Our output aligns with the expected output from the textbook. We see that all schedulers are functioning as expected.