Operating Systems
CS4348

Project #3: OS Scheduling Algorithms

Due Date: Saturday, December 3, 2016

I. Project Organization

This project demonstrate six different scheduling algorithms.

You should do the following pieces to complete your project. Each piece is explained below:

- Code 60 points
- Output 30 points
- Summary 10 points

Code

Your code should be nicely formatted with plenty of comments. The code should be easy to read, properly indented, employ good naming standards, good structure, etc.

Output

Output will be graded by running your program. Each algorithm's output is 5 points.

Summary

The summary section should discuss your project experience. You should discuss how you approached the project, including anything difficult or interesting, what was learned, and the end result. This should be at least 1 page in length.

II. Project Description

Language/Platform/Approach

This project must target a Unix platform and execute properly on our CS1 server.
The project must be written in C++ or Java.
You should use the approach described until Details unless you have obtained instructor approval for another approach.

Problem Overview

This project will simulate a scheduler scheduling a set of jobs.

The project will allow the user to choose a scheduling algorithm from among the six presented in the textbook. It will output a representation of how the jobs are executed.

Details

Jobs can be represented as objects. A job needs to store such things as its name, arrival time, and duration. The running of a job will be simulated by calling a run method on the job. A job should have at least two run methods, one that runs it to completion, and one that runs it a given number of time slices. A job can be thought of as encompassing its PCB contents, so it is okay to store execution time on the job.

Schedulers can also be represented as objects. A scheduler will receive a list of job objects when it is created. It will need to do a few basic things. One is to select a job to run from the set of jobs. Another is to execute the selected job. It must also keep track of time and handle arriving jobs.

Note that a scheduler can be preemptive or nonpreemptive. For nonpreemptive schedulers, once a job is chosen, it runs to completion. For preemptive schedulers, once a job is chosen, it runs n time slices depending on the time slice duration, which could be one unit of time or more than one.

To keep things simple, a job will simply print its name once for each unit of its duration. So if a job has duration five, its task is to print its name five times. Its name should just be a letter, such as A-Z. If run preemptively, it will only print its name for the number of time slices passed.

It is expected that you will have six different scheduler classes. Since there is commonality, you should derive these from an abstract Scheduler class then override methods as needed for each specific algorithm.

To get the project running, use an OS class which reads the jobs from the file and creates Job objects, inputs the scheduler to use by menu or command line argument, then creates the scheduler object and passes the list of jobs. The scheduler object should then produce a graph illustrating how they ran. The graph can be text-based or can use graphics.

Sample Output

Below is sample text-based output. For graphical output, you can make the graph look like the ones in the textbook and slides.

FCFS

```
AAA
   BBBBBB
         CCCC
             DDDDD
                  EE
```

Round Robin, quantum=1

```
AA A
  B B B  B   B   B
     C  C   C   C
       D   D   D  DD
          E   E
```

III. Project Guidelines

Submitting

Submit your project on eLearning. Include in your submission the following files:

1) readme.txt. A readme file describing how to compile and run your project
2) summary.doc A Word document for the summary
3) Your source files

Academic Honesty

All work must be your own. If cheating is suspected, you will be referred to the Judicial Affairs Office for further discussion. Copying may be detected in a number of ways, including by software which compares your code with all other students' source code, by comparison with code on the Internet, or by a visual inspection of your source code.

Resources

The slides and textbook describe the six algorithms to be used.