ECE 5780/6780 Lab 5

(100 points)

1 Objective

To gain experience in writing and debugging a simulator that implements the most used real-time scheduling algorithms.

2 Logistics

You are to work in teams of 1 or 2 members. You and your teammate should be able to independently describe the solutions.

3 RM, EDF, and LST Simulator

Using C or C++, write a simulator that generates schedules for Rate Monotonic (RM), Earliest Deadline First (EDF), and Least Laxity First (LLF) scheduling algorithms. The grader will use Cygwin and gcc / g++ to compile your code. Your program should read from an input file, which contains the following information:

- Number of tasks
- Total simulation time length (msec)
- For each task, the following attributes are specified:
  - ID (needs to be unique)
  - Execution time (msec)
  - Period (with implicit deadline msec)

See below for input file format.

To an output file, your program will write the corresponding RM and EDF schedules, clearly indicating when a job is being preempted and when a job misses its deadline. The format of the output file is up to you but should be easy to understand. At the end of the simulation, your program will also include a summary, indicating the number of preemptions and deadline misses per task and in total.

The names of the input and output files will be specified by the user and passed in from the command line like these examples:

```
$./scheduler.exe input.txt output.txt
$./scheduler.exe f1.txt f2.txt
```

Un-openable input or output files and mal-formed command lines should be handled gracefully.

4 Handling Aperiodic Tasks

Extend your simulator to include a deferrable server to handle aperiodic requests. Output the average response time of the aperiodic requests in addition to the original printouts.

Add the following information to the input file directly after periodic task definitions:

- Number of aperiodic tasks
- For each aperiodic task, the following attributes are specified:
    - ID (need to be unique)
    - Execution time (msec)
    - Absolute Release time (with implicit deadline of 500 msec)

See below for input file format.

5 Submission

Upload the following files as a single zip file on Canvas.

- The source code (plus any other files necessary for compilation such as a makefile) for part 3
- Example input and output files (5 tasks with different periods and a total system utilization of at least 0.9) for part 3
- The source code (plus any other files necessary for compilation such as a makefile) for part 4
- Example input and output files (5 tasks with different periods, 5 additional aperiodic tasks, and a total system utilization of at least 0.9) for part 4
- 1 page results write-up discussing the differences in the schedules generated using RM, EDF, and LLF in terms of preemptions and deadline misses, along with any other conclusions and observations.

6 Input File Format

```
<num Tasks>\r\n
<sim time in msec>\r\n
<first id>, <execution time e in msec>, <period T in msec>\r\n
...
<last id>, <execution time e in msec>, <period T in msec>\r\n
[optional:
<num aperiodic tasks>\r\n
<first id>, <execution time e in msec>, <release time in msec>\r\n
…
<first id>, <execution time e in msec>, <release time in msec>\r\n]
```

Example File:

```
5
600
A, 10, 100
B, 15, 100
C, 20, 100
D, 30, 100
E, 20, 200
5
V, 5, 55
W, 15, 175
X, 10, 230
Y, 10, 265
Z, 15, 280
```

7 Rubric

The grader will use the following criteria to grade your work

Task	Points
Code is well-structured and documented	10
Correct RM functionality	20
Correct EDF functionality	20
Correct LLF functionality	20
Correct aperiodic task functionality	20
Results and Conclusions	10
Total	100