INFO-F-404: Real-Time Operating Systems

2018 – 2019 Project 1: EDF vs LLF

1 Main goal

Study of the two uniprocessor algorithms EDF and LLF. We will consider systems of n periodic, asynchronous and independent tasks $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$ with implicit hard deadlines on a uniprocessor platform.

2 Project details

EDF and LLF are two optimal algorithms studied in the lectures. Please refer to the definition given in the lectures for both algorithm.

For this project, tasks sets will be described in text files. Each line of a given file describes one task and contains: `Offset; WCET; Period;` which are all integers.

Here is an example: `system.txt`:

```
0; 1; 4
0; 5; 10
1; 2; 6
```

For this project, we ask you to realise the following parts:

EDF feasibility interval

For the considered systems and the EDF scheduler, $[0, O_{\max} + 2 \cdot P)$ is a feasibility interval for a periodic asynchronous taskset τ with implicit deadlines. You are asked to implement a script that, for a system given as a parameter in a file, prints the feasibility interval $[0, O_{\max} + 2 \cdot P]$. For example, with the system contained in the file `system.txt` , the following call:

```
python project.py edf_interval system.txt
```

shall print: `0,121`

Generator Implement a generator of random periodic, asynchronous systems with implicit deadlines. This generator should be able to generate a system with given parameters (utilisation factor and number of tasks), for example:

```
python project.py gen 6 70 tasks.txt
```

have to generate a file `tasks.txt` that describes a system of 6 tasks with an utilisation very close to 70% The format must be similar to `system.txt`.

EDF and LLF Simulator

You are asked to create a script that implements a single processor simulator that simulates the system for a given period, for both scheduling algorithm. The start and stop points of the simulation are given as command line parameters, as well as the file containing the task set.

You should be able to execute your program using the following command line:

```
python project.py <scheduler> <tasksFile> <start> <stop>
```

Where `<scheduler> = llf` or `<scheduler> = edf`. For example,

```
python project.py edf system.txt 4 20
```

will produce the following output:

```
TODO
Schedule from: 4 to: 20 ; 3 tasks
4: Arrival of job T0J1
4-5: T0J1
5-9: T1J0
7: Arrival of job T2J1
8: Arrival of job T0J2
9-10: T0J2
10: Arrival of job T1J1
10-12: T2J1
12: Arrival of job T0J3
12-13: T0J3
13: Arrival of job T2J2
13-15: T2J2
15-16: T1J1
16: Arrival of job T0J4
16-17: T0J4
17-20: T1J1
19: Arrival of job T2J3
20: Job T1J1 misses a deadline
END: 1 preemptions
```

Or,

```
python project.py llf system.txt 0 20
```

will produce the following output:

```
TODO
Schedule from: 0 to: 20 ; 3 tasks
0: Arrival of job T0J0
0: Arrival of job T1J0
```

```
0-1: T0J0
1: Arrival of job T2J0
1-2: T1J0
2-3: T2J0
3-4: T1J0
4: Arrival of job T0J1
4-5: T2J0
5-6: T0J1
6-9: T1J0
7: Arrival of job T2J1
8: Arrival of job T0J2
9-10: T0J2
10: Arrival of job T1J1
10-12: T2J1
12: Arrival of job T0J3
12-13: T0J3
13: Arrival of job T2J2
13-16: T1J1
16: Arrival of job T0J4
16-17: T2J2
17-18: T1J1
18-19: T2J2
19: Arrival of job T2J3
19-20: T0J4
20: Job T1J1 misses a deadline
END: 6 preemptions
```

Your implementation must respect the following guide-lines:

- Time is discrete;
- If a deadline is missed, the job is aborted at its deadline;
- The job with the lowest task ID should have the priority in case of a tie:
  - EDF: if two jobs have the same absolute deadline, the job with the lowest ID has the priority;
  - LLF: if two jobs have the same laxity at the same time, the job with the lowest ID has the priority;
  - If two similar events (job arrival or deadline miss) happen at the same time, the event concerning the job with the lowest ID should be displayed first.
- At a given instant t, the logger should print first the missed deadlines, then the job arrivals then the executed job.
- The preemption count start at <start> and ends at <end>. Preemptions before and after should not be taken into account.

Schedule plotter You are asked to write a program that generates a visual output of the scheduling. The choice of the graphical library is up to you but it shall be available freely. You may choose the format (pdf, png, bmp, avi, *etc.*).

Report
Write a short report that contains at least:

- a description of your code (diagrams) and implementation choices,
- a section where you describe difficulties that you met during this project (and solutions that you found),
- a section stating the library you used for the graphical part, where it can be downloaded and how it should be installed.
- a comparison of the preemption count between EDF and LLF.

The project has to be implemented using *python* programming language.

You are allowed to add more additional (optional) command line parameters to your programs.

3 Submission and planning

This project should be done in groups of 3 maximum, you may choose your partner(s). This project has to be submitted before 23:55 on the 14th of December of 2018. Your project has to work properly under *Linux* in rooms NO3.007, NO4.008 and NO4.009 (ULB, Plaine) except for the graphical part that may require installation of additional software.

You'll submit a folder in a *zip* file that contains at least the following files:

- your python scripts,
- a short report (pdf format, maximum 5 pages plus an appendix if needed, figures and graphics are welcome).

The zip file with your project has to be submitted to the UV: `http://uv.ulb.ac.be`. The name of the folder and of the zip file will be as follows: if Jean Dupont made his project with Albertine Vanderbeken, they should send a file named *dupont-vanderbeken.zip* (that contains a folder of the same name, that contains all your project files).

Please observe that if your project crashes during execution or the report is missing or it is not submitted following the described guidelines then your project will not be graded.

For questions on this project, please refer to **xavier.poczekajlo@ulb.ac.be** with subject "[INFO-F-404] project_1".

Good luck!