* Description

For this project, we have to implement two sort of scheduler : a rate monotonic and an earliest deadline first. Also, we only consider synchronous implicit deadline task sets.

We have to implement a tool to visualize the execution of a task set and we also need to show the deadline misses. For that, I used matplotlib.

In this report, we are going to talk about the difficulties of the project, the implementation that I choose and show some graphical examples of rate monitonic and earliest deadline first.

* Running the program

The program is writing in python and it accepts two mandatory arguments : the name of the algorithm you want to use and the name of the task file. You have to use this following command :

**python3 project.py rm|edf <task\_file>**

* Difficulties during this project

La première difficulté que j’ai trouvé dans ce projet était de savoir quelle tâche allait être faite en première. Les tâches avaient souvent une grosse taille mais je le faisais un par un. Par exemple, si nous avions le fichier suivant :

2 4

3 6

J’avais créé une liste qui allait contenir toutes les tâches qui étaient possible de faire par rapport à la feasibilité calculé. Ici, pour le rate monotonic, la feasibilité était de 12. Du coup, nous allions avoir une liste qui ressemblerait a ceci :

[]

Comme on peut le voir, même si la première tâche fait 2 tailles, dans ma liste je la divisais en deux. Ensuite, j’avais une deuxiéme liste qui contenait les premiers jobs de chaque task qui pouvait se faire avant qu’on arrive a un temps x.

Pour le earliest deadline first, la méthode était la même. Au cours de ce projet, on peut même remarquer que les deux algortihmes se ressemblent fortement mais ils sont très différent par rapport à la priorité des tasks. Et que la feasibilité n’est pas toujours la même.

Pour la feasibilité du rate monotonic, j’utilise le LCM mais seulement quand le U(t) < 0,35 mais quand il est plus grand j’utilise le worst case. Pour le earliest monotonic, c’est différent, j’utilise le worst case (faux).

Du coup, revenons vers les priorités, pour le rate monotonic, c’était un peu moins difficile parce qu’il y avait une priorité qui dépendait de voilà et qui restait toujours la même alors que pour le earsliest deadline first, la priorité dépendait de la deadline. Il fallait aussi gérer le cas de quand la deadline était la même. Il y avait plusieurs possibilité mais j’ai choisi de laisser la priorité par rapport à l’index (talk about other possibility).

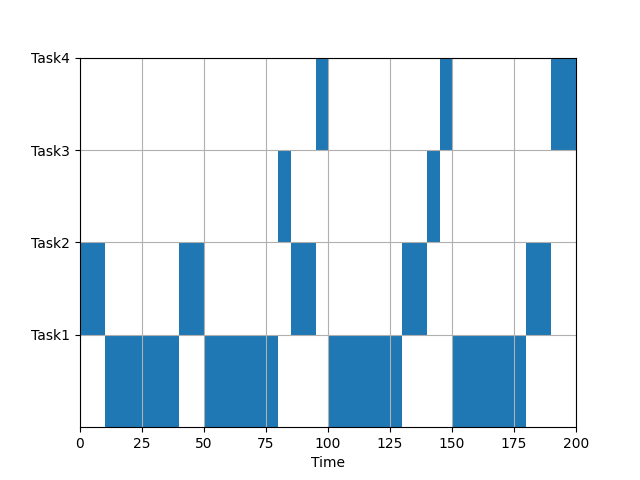
Il y avait une autre petite difficulté a gérer, c’était les deadlines miss et pour ça ma solution était de vérifier s’il y avait une tâche qui n’avait pas été faite avant le temps x qu’il est.

* Implementation

Like i already said, we can see that the rate monotonic and the earliest deadline scheduler are a lit be the same without for the priority task who has very different. I created three class : rm, edf and algorithm. Class rm and class edf are voila by algorithm. That’s why I did that. Also, the feasibility is different that’s why it handle in their own class.

* Example

First example, it’s a rate monotonic is not scheduled but the earliest deadline first is scheduled. We can see that the priority of the task is very important that the system can be scheduled.



Second example, two system is scheduled. We can see that we can have a lot of posibility to handle a system.

