TORTI Clément

VALENTIN Pierre

Report - Stable Marriage Algorithm

Objective of the project

The aim of this project is to implement the stable marriage algorithm (Gale-Shapley) that we studied in "Graph Theory" lesson in the IT training of Enseeiht, Toulouse. We had to implement it in the context of correspondence between Students and Schools like “Parcoursup” : Students choose between a large variety of schools and the schools has to accept or reject them.

We had to let the user be able to create source files and / or to modify existing ones. A source file contains two groups on which to apply the algorithm, those groups being the Students and the Schools. We also had to let the user choose which group between Students and Schools are doing the serenade. We were free to choose the programming language to develop this algorithm and the technology / syntax of source files.

We decided to go with Python because it's a high-level language which is quite simple to use and very flexible. However, Python isn't robust and performing. If we had to make this algorithm efficient for a high-number of elements, we would choose C++, Rust, or another low-level language.

Regarding source files, we decided to use the YAML language. It is an alternative of XML / JSON. YAML has the advantage of being human-readable. Indeed, its syntax is minimal. Another reason for choosing YAML is the fact that Python has a YAML parser called PyYAML which makes YAML files easy to use in Python.

Description of the stable marriage problem

The stable marriage algorithm responds to the problem of finding a stable matching between two groups of elements. It is required for all elements of a group to rank all elements from the other group. A marriage is qualified as unstable if a member from group 1 prefers a member of group 2 than its current partner, and that the same element from group 2 prefers the same element from group 1 than its partner.

The Gale-Shapley algorithm solves this problem by matching all elements from one group to an element of the other group. Even if there may still be jealous elements, they will not be able to cheat on their spouses.

This algorithm can be used in both IT and economics. One can easily imagine its application to the problem of assigning students to universities (for example, the French platform "Parcoursup" uses a modified version of the Gale-Shapley algorithm). It could also be used to match couples of men and women.

To understand the algorithm, our professor used the metaphor of multiple men doing a serenade under the balcony of a different woman, each day. A woman will choose the man who is ranked highest among all propositions she received, and keep him until another man serenades her. If she prefers this new man, then the first man will have to serenade the next woman in his ranked list. The algorithm stops when all elements are matched with another and there is no possibility of cheating on a spouse. An element can accept multiple elements of the other group as long as its capacity allows it (in this case, it is called a polygamy matching).

In order to make our algorithm "generic" (we didn't want to name groups universities / students, or men / women), we used the terms "Serenades" and "Serenaded" which are not grammatically correct. "Serenades" represents the group who will be doing the serenade, while "Serenaded" is the group that will receive serenades.

How we implemented the algorithm

To implement to final algorithm, we divided it into several subgoals:
- Execute by hand the algorithm in both way: school/student <-> student/school
- Think about the most convenient data structure
- Implement the 1 to 1 version of the algorithm: 1 school per student
- Implement the 1 to Many algorithm (we ended up with one algorithm for each way)
- Merging them to 1 final algorithm

Here is a description of the final program:

1. Main:
- Check the arguments passed to know the direction of the binding
- Execute the algorithm accordingly

2. stableMariageAlgorithm(swap):
- Read the input files and place the data into variables.
- Show the input data in the terminal
- While there are serenades “not married” we loop:
    - We place them in their prefered serenaded that hadn’t refuse them. (serenading)
    - Show the current state in the terminal
    - Every serenaded choose their “capacity” favorite serenades within their current list (`getStableChoices`)
    - The other one’s are removed and defined as “not married”
- At that point, the algorithm is finish, we show the result

How to use the source files

Source files uses the YAML Syntax, here is an example :

```
eleves:
      Pierre:
            ecoles:
               n7
               inp
               mine
               insa
      Clement:
            ecoles:
               n7
               insa
               inp
               mine
      Philippe:
           ecoles:
              n7
              mine
              insa
              inp
      Cedric:
            ecoles:
               n7
               insa
               mine
               inp
      Simon:
            ecoles:
               inp
               n7
               mine
               insa
      Guillaume:
            ecoles:
               insa
               mine
               n7
               inp
      Elias:
            ecoles:
               n7
               mine
               inp
               insa

ecoles:
      n7:
         capacite: 2
         eleves:
            Cedric
            Elias
            Pierre
            Guillaume
            Philippe
            Clement
            Simon
      inp:
         capacite: 2
         eleves:
            Philippe
            Guillaume
            Simon
            Cedric
            Clement
            Pierre
            Elias
      insa:
         capacite: 2
         eleves:
            Guillaume
            Pierre
            Clement
            Philippe
            Simon
            Cedric
            Elias
      mine:
         capacite: 2
         eleves:
            Clement
            Simon
            Elias
            Cedric
            Philippe
            Guillaume
            Pierre
```

How to use the script

This script only works with Python 3.X.

If you do not have pip, just install it with the following command :

```
sudo apt install python3-pip
```

Make sure you use the version 3 of pip. Possibly, replace pip by pip3 in the previous commands. You need to install theses modules :

```
numpy, yaml, random, sys, prettytable
```

You have to pass an argument to the script if you want to switch which group does the serenade (which is named main.py), as in the following example:

`python3 main.py` (Students = Serenades, Schools = Serenaded)
`python3 main.py 1` (Schools = Serenades, Students = Serenaded)