STUDENT-PROJECT ALLOCATION WITH ALLOA

ULRICH KRÄHMER AND MANTE ZELVYTE

ABSTRACT. This manual describes the Python script ALLOA that allocates students to projects, supervisors and second markers.

CONTENTS

1. INTRODUCTION

1.1. **The problem.** ALLOA solves the following allocation problem using an algorithm from the NETWORKFLOWX library [1]:

- There are three sets S, P and A of agents whose elements will be called *students*, *projects*, and *academics*, respectively.
- Each student provides a list of (some of the) projects that she would like to work on. The list is ordered by preference. Similarly, for each project there is a list of academics that could supervise it, again ordered by preference and suitability.
- We also specify the minimal and maximal number of different projects a given student could work on, of different students that could work on a given project, and of different students supervised in total by a given academic.

- One can optionally assign a second academic to each project (e.g. as second supervisor or marker), again giving an ordered list of projects they would like to be assigned to in this function, and a quota specifying the total workload of a given academic. This option is enabled and disabled in the configuration file `alloa.conf` which also controls various other parameters.

ALLOA finds an allocation that minimises unhappiness, prioritising students over academics: it is considered for example more desirable if all students get their fifth choice instead of having one student working on their sixth choice, even if this allowed all others to work on their first choice, and ALLOA chooses from these allocations one which minimises analogously the unhappiness amongst the academics.

For full details, background and explanation we refer to [2].

1.2. The process in practice.

ALLOA takes four main input files which define the agents, their quota and preferences, and one configuration file that specifies file names and other parameters. Follow the quick start guide below to have a first test run with example files whose structure is self-explanatory. Note that the first line in each input file is not used for actual data and will not be read by ALLOA.

In the files about academics and projects, keep old project topics and academics on leave but set their upper quota to 0. In the School of Mathematics and Statistics at the University of Glasgow, these files are updated January-February and then the students are informed which projects will be offered in the following academic year. They have some weeks to think which projects they want to apply for, and a speed dating lunch brings students and academics together and here they can ask all kinds of questions.

The most work intense process is to then collect the student choices. Each student could send an email to an administrator with one line only that gets copied and pasted. We used instead an online sruvey in which they had to log in and select their choices from a menu.

Insist that all students submit a fixed minimum number of choices, those with shorter lists would have an advantage! Also, you need to verify manually that all choices are different.

Make sure you leave enough room in case that many studetns want one and the same topic. We used to handle roughly 100 students choosing from 150 projects, many of which could be taken by up to three students. Requiring seven choices from each student was always sufficient. Usually, everyone got one of their first three choices.

Using student numbers instead of their names avoids having two agents with the same identifier, and also makes it less controversial to simply publish the whole list or send it round by email to all students.

ALLOA could be easily extended to include many more features, such as optional choices and ties, or if desired certain students could be given priority when assigning projects.

2. Quick start guide

2.1. Installation.

ALLOA is freely available at

`http://www.maths.gla.ac.uk/~ukraehmer/alloa.tar`

The archive contains

- `alloa.py` - the script itself
- `alloa.conf` - its configuration file
- `alloa.pdf` - the present manual
- `s.csv,p.csv,a.csv,m.csv` - sample input data

2.2. Running ALLOA.

If you work on a Linux machine with Python and the NETWORKFLOWX library installed, place ithe above files into a directory and run

```
python alloa.py
```

in a shell (being in that directory). Obviously, ALLOA can also be used on any other operating system, but you might need to enter the full paths to all files in `alloa.conf`

2.3. Output files.

ALLOA produces the following output files:

- `allocation_MMDDYY.csv` - the actual allocation that one would publish to students and academics. `MMDDYY` is the date on which the allocation was computed.
- `allocation_profile_MMDDYY.csv` - a file with an overview how successful the allocation was
- `allocation_workloads_capacities_MMDDYY.csv` - a list of how often each project and academic has been assigned
- `allocation_second_markers_MMDDYY.csv` - the outcome of the second marker/supervisor allocation (optional)
- `allocation_markers_workloads_MMDDYY.csv` - the additional workload per academic arising form seocnd marking/supervision
- `allocation_total_workloads_MMDDYY.csv` - the total workload of each academic

2.4. **Troubleshooting.** The following are common reasons for crashes:

- `IndexError: list index out of range`: the csv files defining the agents contain empty lines (usually at the end), or if second markers are allocated their list does not agree with that of the academics.
- `KeyError: 'NAME'`: The agent NAME occurs in a preference list but is not defined (usually it is just misspelled).

References

[1] `https://networkx.github.io/ NetworkX (NX)`

[2] Ulrich Krähmer, Augustine Kwanashie, David Manlove, Mante Zelvyte, *Student/Project Allocation Using Network Flow and Integer Programming*, to appear