Stochastic Methods for Resource Allocation that Respect Preferences:

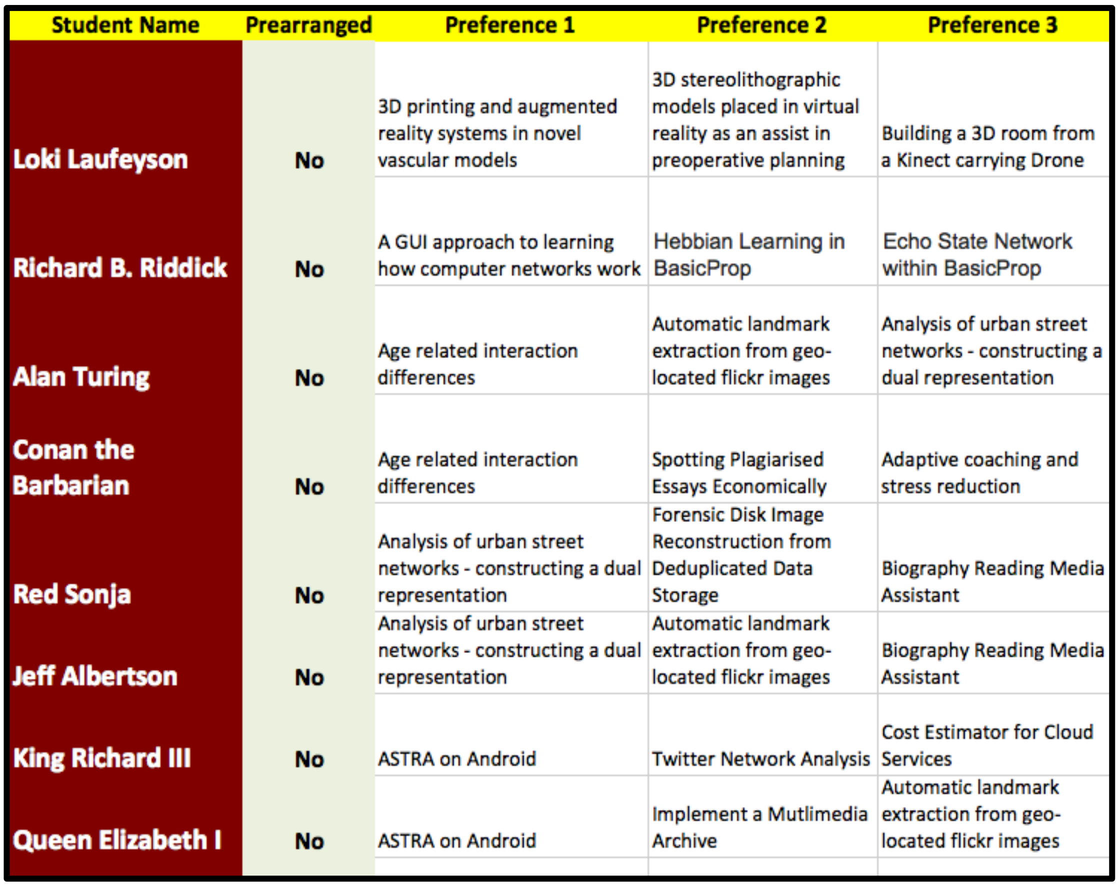
The Case of Project Allocation to Students

Introduction

This project concerns a problem that you will encounter in many parts of industry and in many walks of life: the allocation of limited resources to those who request them, in a way that aims to respect the preferences of those making the requests. Many resources in life are limited, and we can't always get what we want. One such resource is a project specification for a final-year project. These FYP specifications are offered to students who must each choose a different project, though some project specifications will be more popular than others and attract more interest from students. Not everyone can get the project they want. So students are asked to provide an ordered list of preferred projects, from first to last, and the allocation algorithm is tasked with assigning projects to students in a way that maximizes the fit between students and their preferences. Not everyone can get their first preference, but it is preferable to get one's number two or number three preference than one's tenth preference, or to get a project for which no preference at all was expressed.

In this project you are going to work with real data, which has been modified slightly to protect the privacy of the people involved. Though the student names have been replaced (you will notice some very famous alumni in this data) the project names, or their relative popularity, have not. We provide a snapshot of the data on the following page, while the complete dataset is contained in the attached spreadsheet file.

A spreadsheet is simply a table of rows and columns. Each row corresponds to a single student, though the first row is just a list of column labels. If there are N rows there are N-1 students. The first column holds the name of each student. The second indicates whether the student has *pre-arranged* a project with a supervisor in advance. When a project is pre-arranged, then that student is guaranteed to be assigned that project, and so only one preference is expressed.



Student Name	Prearranged	Preference 1	Preference 2	Preference 3
Loki Laufeyson	No	3D printing and augmented reality systems in novel vascular models	3D stereolithographic models placed in virtual reality as an assist in preoperative planning	Building a 3D room from a Kinect carrying Drone
Richard B. Riddick	No	A GUI approach to learning how computer networks work	Hebbian Learning in BasicProp	Echo State Network within BasicProp
Alan Turing	No	Age related interaction differences	Automatic landmark extraction from geo-located flickr images	Analysis of urban street networks - constructing a dual representation
Conan the Barbarian	No	Age related interaction differences	Spotting Plagiarised Essays Economically	Adaptive coaching and stress reduction
Red Sonja	No	Analysis of urban street networks - constructing a dual representation	Forensic Disk Image Reconstruction from Deduplicated Data Storage	Biography Reading Media Assistant
Jeff Albertson	No	Analysis of urban street networks - constructing a dual representation	Automatic landmark extraction from geo-located flickr images	Biography Reading Media Assistant
King Richard III	No	ASTRA on Android	Twitter Network Analysis	Cost Estimator for Cloud Services
Queen Elizabeth I	No	ASTRA on Android	Implement a Mutlimedia Archive	Automatic landmark extraction from geo-located flickr images

The third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh and twelfth columns hold each student's first, second, third, fourth, fifth, sixth, seventh, eight, ninth and tenth project preferences, though a student may express fewer than ten preferences (meaning that some columns will be blank for some students). Each of these columns contains the name of a project, represented as a string. The same string always denotes the same project whenever it is used in the file.

The Key Challenge

Given a spreadsheet organized as above, generate a 1-to-1 mapping of students to projects that, as far as is possible, gives each student their highest preference of project. As many students will not get their top preference, we shall have to define what is meant by a "valid" mapping and the "best" mapping. A valid mapping is one in which each student gets one, and just one, of their expressed preferences, while no project is allocated to more than one student. A valid mapping also respects the pre-assignment of projects, which is to say, it obeys the settings given in the second column of the spreadsheet. The “best” mapping is the one that minimizes the disappointment of the student body as a whole. We can say that a student that gets their top-rated (first) preference experiences *zero* disappointment, while a student who gets their second rated preference experiences *one* unit of disappointment, and a student who gets their third-rated preference experiences *two* units of disappointment, and so on, so that a student who is assigned their lowest-ranked (tenth) preference experiences *nine* units of disappointment. By summing the disappointment of all students in a mapping we can calculate an overall disappointment value for the mapping as a whole. In the perfect (and very rare) situation where the total value of a valid mapping is zero units of disappointment, then every student has been assigned their first-preference. Naturally, we are only interested in evaluating valid mappings when a valid mapping is possible. We shall consider cases where it is not shortly.

Searching for the “Best” Valid Mapping: Two Approaches

We can use brute-force to consider all valid mappings of projects to students, measure the total disappointment of each valid mapping we generate, and return the mapping that causes the least total disappointment as a solution. But this is rarely a sensible use of resources, and it may not be workable at all for large spreadsheets. So in your system you will use two stochastic techniques as discussed in class, *Simulated Annealing* (SA) and a *Genetic Algorithm* (GA). These are alternate approaches that should both be implemented by you as part of your system. Each approach requires you to generate random solutions as a starting point. SA is a greedy approach that uses just one random solution, and works to improve this solution incrementally. In contrast, a GA generates a whole population of random solutions and aims to improve the overall fitness of the population by mating the solutions with the highest fitness and culling the solutions with the lowest fitness. In the case of SA you will need to experiment with different *temperature schedules*, and in the case of your GA you will need to experiment with different population sizes, mating policies and culling policies. In the case of SA you will need to define an *energy function* that reflects the inadequacy of any mapping, so that SA can gradually reduce this energy and improve the mapping as much as it can (clearly, energy will be a function of overall disappointment). For your GA, you will need to define a *fitness function* that reflects the adequacy of a solution. Clearly, fitness will be inversely related to overall disappointment. You are free to develop your energy and fitness functions as you see fit, but naturally they should reflect the basic intuitions about preferential (and valid) project allocation as we have discussed them here.

Invalid Mappings

It may not be possible to give every student one of their preferences, especially if some students attempt to "game" the system by expressing only one preference. We can deal with such situations by assigning a student with no assigned project one of the left-over projects that has not been assigned to anyone else. Naturally this a most undesirable situation, but one we must be willing to consider. So that this situation is avoided whenever possible, we can assign a high disappointment score to any student that is assigned a project they did not express a preference for. How high should this disappointment score be? It should be related to the number of preferences that the student *did* express. A student who expresses 10 preferences has made a solid effort to engage with the system, so we can assume that this student experiences 10x10 units of justifiable disappointment, or 100 – that is, 10 multiplied by the number of preferences expressed by the student. If a student unwisely expresses only one preference (and it is not a pre-assigned project) then we can estimate the student's righteous disappointment at just 10x1 units, or just 10 units. This means that our system is more likely to assign a random left-over project to a student who states only one preference than it is to to a student who expresses 10.

Your Target Solution

Your system should be able to read in a spreadsheet of preferences, like the one displayed previously. You may assume that the spreadsheet has been saved as a *.tsv* (tab-separated values) text file, so you can simply use text-processing I/O in Java to read the file line by line, each line representing a different row, and use an instance of Java's StringTokenizer class (from *java.util*) to obtain each column value from each row/line. For the SA solution, run your algorithm 10 times and return the "best" solution (that is, the one with the lowest energy). For the GA solution, run your algorithm for N generations and return the solution with the highest fitness. Report to the user whether the solution is a valid solution. If it is not, report to the user which student or students have been assigned projects they did not ask for. In every case, offer an assessment of the solution quality.

Note: Your code for this project should be entirely your own. Do not use code for existing SA or GA solutions, even if these are available as third-party libraries. You may use libraries to add a GUI to your system, but do not use boiler-plate code from the Web for any part of your solution.

What To Submit, and When

You should submit the following via Moodle

- The complete source code for your project
- A README file showing how to run your code. The more convenient you make this, the more likely you are to be marked well
- Your final report, outlining how your solution works, what its key features are, or any other points you would like your graders to consider.