Requirements document
Assigning readers to projects

Supervisor: David Manlove
Author: Eimantas Sapoka

October 3, 2014

Chapter 1

Introduction

1.1 Problem definition

The University has a system to assign readers (university staff/lecturers) to projects. Each project must have a supervisor (who is also one of the project's readers) and must have a reader assigned to it. The readers have a limit on how many projects they can be readers/supervisors of (a person with a limit of 10 and supervising 7 projects can only be a reader of no more than 3 projects). Then the readers rank projects they would like to review in order of preference (ideally, the number of rankings is two times the number of projects the reader can be assigned). Each project is then assigned a second reader taking into account the readers' preferences. The current system is not load balanced, meaning the number of projects assigned to readers can be uneven (one reader being assigned his maximum capacity and the second not getting projects at all).

1.2 The task

This project aims to replace the current system by implementing the maximum flow (Ford-Fulkerson), minimum cost algorithm to assign readers to projects based on their preferences and extend it with load balancing. The result must be the optimal reader to project assignment taken into account readers preferences, number of readers limit per project and load balancing. Furthermore, the new system should have a usable user interface and be able to read in and export data.

Chapter 2

Requirements

2.1 Functional requirements

The requirements are categorised using the MoSCoW principle.

2.1.1 Must Have features

- **GUI.** For ease of use, the system must have a usable, user friendly, functional GUI. Weather it is web based or standalone desktop application, it is yet to be decided.
- **File input.** The user must be able to select and input a data file with project and reader data.
- **File output.** The system must be able to export the resulting allocations in text format.
- **Input data altering.** The user must be able to clearly see and alter the input data within the application.
- **Result altering.** The user must be able to preview and alter the results/allocations.
- **Algorithm.** The system must implement the Ford-Fulkerson maximum flow algorithm with minimum cost and load balancing.
- **Information.** The system must display useful information to the user, such as unselected projects, the resulting flow cost, readers with short ranking lists, etc.

2.1.2 Should have features

- **Exporting as Excel.** The system would allow the user to export the results in text and excel formats.

- **Usage manual.** The system would have a usage manual to help with troubleshooting and usage.

2.1.3 Could have features

- **User friendly output.** The system might retrieve information about readers and projects from the current application and map IDs to names and show output in a user friendly format (names rather than IDs).
- **Graphs and visual information.** The system might have visual information such as graphs to reflect and visualise the results.

2.1.4 Would like to have features

These features are most likely not within the scope of the project and would only be possible within certain circumstances.

- **Extending the current application.** The system would gain access to the application holding readers and projects information and access it / extend it so no user input would be required. The system would retrieve data automatically.

2.2 Operational requirements

- **The system should be responsive.** The system should not load for an extensive amount of time and should not crash/fail unexpectedly.
- **The system should be easy to use** The system should be intuitive, not be too complex.
- **The process should be fast.** The algorithm assigning readers to projects should not take more than 20 seconds to run. (This is very slow performance, but the opration is intended to be used once per year, therefore the performance speed is not a huge factor).