COMP4801 Final Year Project

A Web-based Project Allocation System

Project Plan

LEI Wan-hong

Department of Computer Science

The University of Hong Kong

supervised by

Dr. T. W. Chim

October 9, 2016

This page is intentionally left as blank.

Contents

Abstract

This project plan outlines a matching problem between students and projects subject to various constraints. In particular, a web-based system will be implemented to model this problem, together with an algorithm to generate the best matching. Besides, technical aspect of the system is examined. Finally, a tentative schedule is included with distinct milestones to be achieved.

1 Introduction

In many universities, students are generally required to undertake a project before they graduate. Lecturers first propose one or more projects for students to choose from. Students can then take any one of these projects and are supervised by the corresponding lecturers. In addition, different students can work together as a group. Usually, these projects are not expected to be fully taken up by students whilst the number of these are generally sufficient for all students.

1.1 Problem Statement

To reduce the chance of clashing between different students, each student is required to provide a finite number of projects with priority that they are interested in. This is known as the *preference list*. Likewise, each lecturer is required to provide a preference list on students for each project.

The problem in here is to find the best[1] matching between students and projects. It is similar to that of the *Student-Project Allocation Problem* (SPA), which can be further generalised by the *Two-sided Matching problem* [1].

In this project, a web-based system will be developed to stimulate this matching problem. And an algorithm is introduced to solve this problem. Both students and lecturers submit their preference lists on the web-based system and the best matching is returned.

The remainder of this plan proceeds as follows. First, a mathematical description of this problem is introduced together with a current solution. Next, the Project Allocation System is presented with distinct objectives to solve this problem. Then the implementation details are followed. Finally, risks and migrations are discussed and a tentative schedule is included.

[1]Definition will be discussed in Section 2.1.

2 Background

Finding the best matching between students and projects can be done manually when both the size of students and projects are fairly small, possibly within 10. A straightforward approach to construct a matching is individually matched each student to an unallocated project that has a highest priority on the student's preference list. However, when these numbers become substantial, constructing such a matching manually can take up to few months or even impossible.

Besides, this decentralised approach is known to be deficient and cannot guarantee the *stability* property [1]. Therefore a software system that able to automatically construct the best matching is desperately needed.

2.1 Theoretical Aspect

Mathematically, the SPA can be described in a following way. Let

$$\begin{aligned} S &= \{s_1, \cdots, s_n\} \text{ be the set of } n \text{ students,} \\ L &= \{l_1, \cdots, l_m\} \text{ be the set of } m \text{ lecturers and} \\ P &= \{p_1, \cdots, p_q\} \text{ be the set of } q \text{ projects.} \end{aligned}$$

Each lecturer l_i offers a subset of project P_{l_i} which is a partition of P. Assume each project is uniquely offered by one lecturer, the disjoint union of $P_{l_1}, \cdots, P_{l_m}$ is then P. Each student s_i has a preference list α_i which contains a ranked subset of P in a strict order. Similarly, each lecturer l_i has a preference list β_i^j for each project p_j in P_{l_i} (i.e. offered projects) ranked in a strict order. Also, each project p_i has a capacity constraint c_i which limits the maximum number of students in this project. Likewise, each lecturer l_i has a capacity constraint d_i which limits the maximum number of students that the lecturer can undertake.

It follows that a *matching* or an *assignment* M is a subset of $S \times P$ such that [1]:

1. If (s_i, p_j) belongs to M, then p_j belongs to α_i, that is, s_i preference list contains project p_j;

2. For each student s_i, at most one project p_j is matched in M;

3. For each project p_i in P, $|(s_k, p_i)| \leq c_i$, that is, the number of students matched must not greater than c_i; and

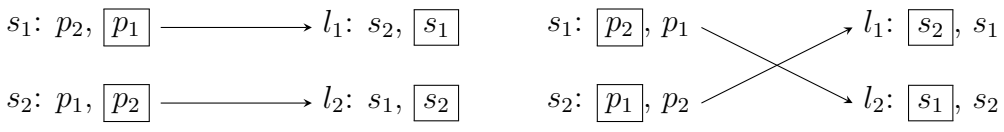
4. For each lecturer l_i in L, $|(s_k, p_j)| \leq d_i$ for all p_j in P_{l_i}, that is, the number of students matched must not greater than d_i.

Then (s_i, p_j) means that s_i is assigned to p_j. If p_j is offered by l_k then it also means l_k is assigned to s_i.

It follows that the stability of a matching may be defined as follows [2]:

Definition 1 *A matching M is called stable if it does not exist two students s_1 with project p_1 and s_2 with projects p_2, that s_1 prefers p_2 to p_1 and s_2 prefers p_1 to p_2.*

An example is illustrated as below. Suppose lecturer l_1 offers project p_1 and that of l_2 offers project p_2. Student s_1 has a preference list of $\alpha_1 = \{p_2, p_1\}$ such that p_2 is more preferable than p_1. This priority, as defined, applies to preference lists of all students and lecturers.



(a) Unstable matching (b) Stable matching

Figure 1: An example to illustrate (a) Unstable matching and (b) Stable matching, with their assigned project or student boxed.

An *unstable* matching M' happens when $M' = \{(s_1, p_1), (s_2, p_2)\}$ (see Figure 1(a)). When such a pair exists, a *stable* matching $M = \{(s_1, p_2), (s_2, p_1)\}$ (see Figure 1(b)) can always obtain by swapping their projects. This stable matching is better than that of the unstable.

There are potentially many stable matchings existed in a SPA instance. However, among of all these matchings, the *optimal* one is prefrred [2]:

Definition 2 *A stable matching is called **optimal** if every student is at least as well off in this matching as in any other stable matchings.*

This definition is saying that a matching is optimal if and only if it matches each student with a project that has a highest priority on the student's preference list than that of the other projects.

If a matching satisfies both *stability* and *optimality*, then it is uniquely existed as the *best matching* [2].

2.2 Current Solution

A linear time algorithm has been proposed to solve the original SPA using appropriate data structures [1]. However, this algorithm does not perfectly solve the problem described in this project. Specifically, it does not allow students work in a group. A group of student may be defined as follows. Let $G = \{G_1, \cdots, G_r\}$ be r groups. Each group G_i in G is a subset of S such that each student s_k in G can exist once only (i.e. a student cannot belong to 2 different groups). It follows that $S \setminus G$ is a subset of students who work individually.

3 Objectives

This project aims to develop and implement a web-based system called the *Project Allocation System* (PAS) to generate the best matching between students and projects. Students and lecturers can submit their preference lists on the system. The PAS then run an algorithm to find the best matching.

3.1 Scope

The PAS contains following major functions:

1. *Student registration.* Each student can register as a user in the system.
2. *Ranking projects/students.* Users are allowed to rank a finite number of projects or students.
3. *Generate the best matching.* After the system collected preference lists fro all users, the best matching is generated.
4. *Administrator panel.* For administrator to manage the system. One major function is to approve generated matching.

3.2 Goals

The ultimate goal of this project is to reduce the time and cost in constructing the best matching between students and projects which is primarily subject to their preference lists. Particularly, the administrative cost can be remarkably cut down. The web-based

nature of the PAS allows users to access the system from range of devices (e.g., laptop, mobile) without having to develop another system on a specific platform.

4 Methodology

4.1 Software Development Process

An unified software development process is used in this project. It features 4 distinct phases with each phase containing iterative and incremental cycles. These phases are Inception, Elaboration, Construction and Transition. The last phase, Transition will be omitted in this project because of the limited time on deploying the system into real-life situation. The details of each phase are given in Table 2.

4.2 Technical Aspect

As the PAS is designed to be a web-based system, both web server and database management system are required. A high-level system architecture diagram is illustrated as below. Essentially, the PAS is based on these free and open source software packages.

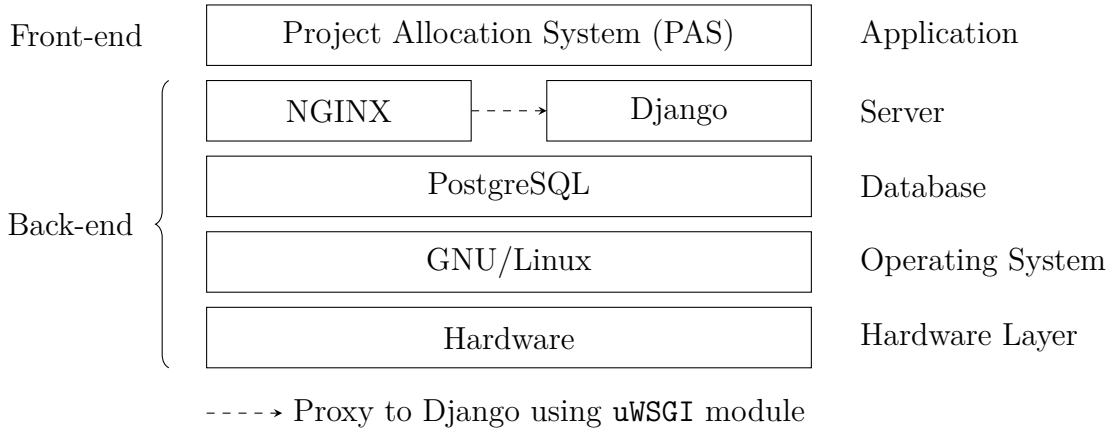


Figure 2: A high-level system architecture diagram.

4.2.1 Back-end Layer

The PAS will deploy under the GNU/Linux operating system which offers a robust and security platform for software to run.

The PAS uses the Django web framework as its back-end system. The scripting language for the Django is Python. With comprehensive built-in functions in Django, it provides a rapid development platform. For the database system, PostgreSQL is chosen. It is an object-relational database system (ORDS) that is renowned for its reliability and data integrity.

To handle users' requests, NGINX is chosen for the web server. It is a lightweight and yet a highly scalable server compare with Apache. By adding `uWSGI` module, it serves as a proxy server to redirect requests to the Django web framework so as to provide dynamic web contents. Together with the PostgreSQL database system, the performance of the PAS will outperform the traditional LAMP stack which comprise of GNU/Linux, Apache web server, MySQL database system and PHP.

4.2.2 Front-end Layer

In order to implement a web-based system, 3 core web technologies are used in the PAS. These are HTML, CSS and JavaScript which are all run on the client side, that is, the browser. The details of each technology are as follows:

Name	Full name	Major functions on a web page
HTML	Hypertext Markup Language	Outline the structure of a web page, it serves as the building blocks
CSS	Cascading Style Sheets	Set the visual style to provide a better user interface
JavaScipt	N/A	Provide interactive elements

Table 1: A table listing 3 core technologies used in the PAS.

5 Risk, Challenges and Mitigation

Despite the project is mainly about developing the PAS, there are still several challenges. First, the variant SPA introduced in this project might not be able to solve in polynomial time, that is, no polynomial algorithm for this problem. This will greatly affect the efficiency of the PAS. Some assumptions (e.g. work as a group) might have to abandon if it happened.

Second, since only web interface will be implemented, there might have discrepancy between different devices. This will lead to difficulties for users to use the system. It is hoped that by using some well-written web front-end frameworks, the user interface can maintain its consistency.

6 Schedule

The tentative schedule to design and implement the PAS is given in Table 2.

7 Conclusion

The PAS we introduced in this project aims solve the SPA in real-life scenario. The web-based nature of the system allows great flexibility across devices. After obtaining the preference lists from users, the system can construct the best matching efficiently.

Date	Tasks
4 October 2016	Deliverables of Phase 1 (Inception) • Detailed project plan • Project website
October 2016	Requirements gathering and analysis • Collect user requirements • Further study on SPA based on the requirements • Implement a prototype to solve the SPA
November 2016 to December 2016	Development of the demo system • Implement a working web-based demo PAS • Design the web-based user interface
22 January 2017	Deliverables of Phase 2 (Elaboration) • Preliminary implementation of the demo system • Detailed interim report
February 2017 to March 2017	System Design and Construction • Refine the demo PAS • Implement the administrator user interface
April 2017	Depolyment and Testing • Final testing and optimisation • Deploy the PAS
17 April 2017	Deliverables of Phase 3 (Construction) • Finalised implementation • Final report

Table 2: Tentative schedule.

References

[1] D. J. Abraham, R. W. Irving, and D. F. Manlove. Two algorithms for the student-project allocation problem. *Journal of Discrete Algorithms*, 5(1):73–90, 2007. 1.1, 2, 2.1, 2.2

[2] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962. 2.1, 2.1, 2.1