BLG 372E - Analysis of Algorithms, Spring 2014

Project 1 – Stable Matching Problem

Total Worth : 10% of your grade

Handed Out : 04.03.2011 Tuesday

Due : 18.03.2007 - Wednesday- 23:00

Overview

Your project is about a lecturer-assistant assignment problem. Lecturers have preference lists for assistants. Assistants have preference list for the courses. **One lecturer can have more than one course in the term. However an assistant can have only one course in each term.**

Note: This is not a straightforward stable matching problem. As you have noticed, assistants prefer courses while lecturers prefer assistants. You need to evolve this problem into a stable matching problem.

Code (60 points)

You have three type of input files.

CourseOfLecturers.txt file contains the courses of each lecturer. i^{th} line of the file contains the courses of the i^{th} lecturer.

LecturersPL.txt file contains lecturer's preference list for assistants. i^{th} line of the file contains the preference list of the i^{th} lecturer for assistants.

AssistantsPL.txt file contains teaching assistants' preference list for courses. i^{th} line of the file contains the preference list of the i^{th} assistant for courses.

1. [30] Gale-Shapley algorithm: Use the implementation outlined in the slides. Lecturers offer to assistants. (Lecturer optimal solution) See file GS_out.txt for an example.

2. [15] Gale-Shapley algorithm: Modify your solution in (1) so that assistants offer instead of lecturers.

3. [15] You assign weights to lecturers and assistants depending on their matching.
   - ✓ For example there are n assistants and n courses. If the assistant A gets the most favorable course on her preference list after the termination of algorithm, it sets Weight[A]= n^2. If the assistant A gets the second favorable course on her preference list, it sets Weight[A]= $(n-1)^2$. If the

assistant A gets the least favorable course on her preference list, it sets Weight[A]= 1. So on.

- ✓ You will apply same procedure to find weight of each lecturer.
- ✓ Firstly you get total weights of all lecturers and all assistants, when lecturers offer to assistants. Write the total weight of lecturers and total weight of assistants for lecturer-optimal GS.
- ✓ Secondly, you get total weights of all lecturers and all assistants, when assistants offer to the lecturers. Write the total weight of lecturers and total weight of assistants for assistant optimal GS.

- ✓ The compilation command is: g++ yourStudentID.cpp –o GS
- ✓ The running command must be:

./GS –i CourseOfLecturers.txt LecturerPL.txt AssistantsPL.txt –o GS_out.txt

You must get an output file as in GS_output.txt

Report (40 points)

1. [15] Your program

- ✓ How do you construct preference lists of Gale-Shapley algorithm. Explain with figures or tables.
- ✓ How does your algorithm work? Write your pseudo code.
- ✓ Explain your classes and your methods. What are their purposes?

2. [10] Show complexity of your algorithm on pseudo-code
3. [15] According to weights, compare lecturer optimal and assistant optimal solution. What is the reason of the difference between lecturer and assistant optimal stable matching? Explain in details.

Submission

You should be aware that the Ninova system clock may not be synchronized with your computer, watch, or cell phone. Do not e-mail the teaching assistant or the instructors your submission after the Ninova site submission has closed. If you have submitted to Ninova once and want to make any changes to your report, you should do it before the Ninova submission system closes. Your changes will not be accepted by e-mail. Connectivity problems to the Internet or to Ninova in the last few minutes are not valid excuses for being unable to submit. You should not risk leaving your submission to the last few minutes. After uploading to Ninova, check to make sure that your project appears there.

Policy: You may discuss the problem addressed by the project at an abstract level with your classmates, but you should not share or copy code from your classmates or from the Internet. You should submit your own, individual project. Plagiarism and any other forms of cheating will have serious consequences, including failing the course.

Submission Instructions: Please submit your homework through Ninova. Please zip and upload all your files using filename HW1_ studentID.rar. In the zipped file, you must include your completed report_StudentId file and all your program and header files.

All your code must be written in C++, and we must be able to compile and run on it on ITU's Linux Server (you can access it through SSH) using g++. You should supply one yourStudentID.cpp file that calls necessary routines for all questions (Multiple files are acceptable, as long as you state the compilation instructions in your report).

When you write your code, try to follow an object-oriented methodology with well-chosen variable, method, and class names and comments where necessary**. Your code must compile without any errors; otherwise, you may get a grade of zero on the assignment.**

If a question is not clear, please let the teaching assistant know by email (ozens@itu.edu.tr).