CS 170 Efficient Algorithms and Intractable Problems
Spring 2016 Alessandro Chiesa and Umesh Vazirani

Final Project

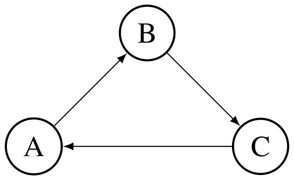
Problem Description

Currently, there are over 93,000 patients on the kidney transplant waiting list waiting for deceased donors. Unfortunately, there are not enough deceased donors to satisfy the needs of our patients. The National Health Service has been suffering from a lack of funding so they have resorted to hiring CS 170 students to work on a new kidney exchange system and algorithm involving living donors to help optimize the network of kidney donations.

In many cases, patients have a relative or friend who is willing to donate a kidney. However, the problem is that the donor's kidney is not compatible with the patient. All such incompatible pairs will enter our new kidney exchange system, where the donor's kidney may be compatible with the patient in another pair.

We can formulate this problem as a graph $G = (V,E)$ where the vertices are each pair P (patient and donor). Consider two such pairs P_1 and P_2. A directed edge exists from P_1 to P_2 if the donor from P_1 has a compatible kidney with the patient from P_2.

Therefore, in order to carry out kidney transplants for these patients, we need to find cycles in the system. For example, suppose that the following cycle exists in the graph:



Using this cycle, we can create a donation chain from A to B, B to C, and C to A. By including a node in a donation chain, that patient is treated successfully. Each node in the graph can be obviously only used once; unless you happen to have three kidneys in your body, you can only donate one kidney.

However, all the transplants must happen at the same time, and it's infeasible to do many operations at once. Therefore, each donation chain must at most 5 in length.

Moreover, the National Health Service has told us that it would prefer to prioritize children over adults receiving transplants. To model this, we receive a penalty of 1 for every untreated adult and a penalty of 2 for every untreated child.

Your goal is to find a set of donation chains that minimizes the penalty of our donation network

Constraints

Suppose that the graph is $G = (V,E)$. We have the following constraints:

- $0 < |V| \leq 500$
- $l(v_1, v_2) = 1$ if v_1 compatible with v_2 patient. Else, 0.

Deliverables

Phase 1: Input Files: Due April 25th 11:59 PM

Generate three hard instances. Make sure to follow the input format exactly, or your file will be discarded.

You will submit your instance files through glookup. In your submission, include your three files, labeled TEAMNAME1.in, TEAMNAME2.in, TEAMNAME3.in. Also, include a README file where the first line is your team name, and the following up to 4 lines are the student ids of people in your group.

Phase 2: Output Files, Code, Write-Up: Due May 2nd 11:59 PM

After Phase 1, we will release the input files generated by each group that are valid. We will post this as a zip file on piazza. After we release all the input files you will run your code, and submit your output file and your code. Make sure you only have a single output file (see output format specification for more details). The input files will be named "i.in", where i is the number of the instance. You can expect around 200-400 input files. See the output specifications for more details on how to format your output file.

Submissions will be done through glookup. In your submission, include your output file, as well a README file where the first line is your team name, and the following up to 4 lines are the student ids of people in your group. This README should be exactly the same as the one you submitted in the first phase.

You will also submit a final report over Gradescope as a team. In half a page, please explain at a high level what strategies and optimizations you used. Please also cite any outside resources you used.

Grading

You will be graded on the following items. The project is worth 2 homeworks and **it cannot be dropped**.

- Valid input files (10 pts)
- Quality of algorithm and results compared to other teams (80 pts)
- Final Report (10 pts)

Miscellaneous Rules and Reminders

- You may work in teams of at most 4.
- You may use any language, publicly available libraries, and research papers, but please make sure to cite your sources if you decide to copy something or use a published method.
- If your input file or output file does not conform to the specifications, you will not get any credit for them. Make sure you carefully read the next sections for more details. Please run the verification scripts before you submit your files to make sure they are correct. There is always one group who doesn't run the script and submits incorrect files, don't be that team.
- The staff is unable to give very involved technical support with specifics on implementation. Choose a language that you are comfortable debugging in. We will help, however, with running through general approaches and strategies if you'd like.