CS 170 Fall 2017 — Efficient Algorithms and Intractable Problems — Prasad Raghavendra and Sanjam Garg — Project

Release date: November 3, 2017. Updated as of: November 3, 2:45 PM.

Introduction

You are at another wizard party hosted by Prasad and Sanjam. As usual, wizards are shy about discussing their age. From conversations at the party, you are only given information in the following form: some wizard's age is not between the ages of two other wizards. Since wizard parties are also notoriously long (an entire month!), you decide to use your time at the party to write a solver and figure out the true ordering of wizards by their ages.

Grade-wise, this project alone is worth two homework grades and cannot be dropped. You may work in groups of 1-3. We encourage you to form groups; grading expectations will not scale with the number of people in your group. Your grade for Phase I and Phase II will be curved separately. Your grade will be shared with your partner(s). All due dates are at 11:59 PM, PST.

Phase I: Input Files

Overview

Due date: **November 10th**
Deliverables: Each group must submit three input files.

We will collect all of your input files and release a subset of the hardest problems publicly after the phase I due date. The difficulty of your input files will determine part of your grade for this phase.

Details

You will submit three input files. These files must be called `input20.in`, `input35.in` and `input50.in`. These files must be zipped inside a folder called `inputs` and the zip must be submitted to Gradescope on the Phase I assignment. Throughout this project, only one group member needs to submit to Gradescope, however, remember to **add your partners on Gradescope**.

We are providing you with a Python Script `input_validator.py` to validate your input files locally. The script runs in Python3, and it must be used as follows:

```
python instance_validator.py [path to input file] [20, 35 or 50]
```

Each input file must adhere to the following specifications.

- The first line of each input file must be a single integer W, the number of wizards in your problem. This number may be between 1 and (20, 35 or 50), depending on which input file you're working on. For instance, `input20.in` must contain at most 20 wizards.
- The second line of each input file must contain the W unique, alphanumeric wizard names, separated by a space. Each wizard name must be an alphanumeric string of maximum 10 characters. These space-separated wizard names must be an ordering of the wizards (according to their relative ages)
- The third line of each input file must be a single integer C, the number of constraints in your problem. This number may be between 1 and 500.
- The next C lines of your input file must contain constraints of the form:
  `<Wizard1 Wizard2 Wizard3>`. Each constraint specifies that the age of `Wizard3` is not in between the ages of `Wizard1` and `Wizard2`. **Note: this does not mean `Wizard1` is necessarily younger than `Wizard2`.**

If your input files do not satisfy these requirements, then your input is considered invalid, and you will not receive any credit for this portion of the project. Although Gradescope will autograde your inputs, and print out relevant error messages for badly formatted input files, it is still your responsibility to write correctly formatted input files.

Sample Input

```
4
Harry Hermione Severus Albus
3
Hermione Harry Severus
Severus Harry Albus
Severus Albus Hermione
```

Grading

This entire phase is worth 15% of your project grade based on the difficulty of each input file itself. For each file, the grading follows as below:

5% (full credit): at least 80% of student submissions cannot return the optimal ordering
4%: 60-80% of submissions cannot return the optimal ordering
3%: 40-60% of submissions cannot return the optimal ordering
2%: 20-40% of submissions cannot return the optimal ordering
1%: 0-20% of submissions cannot return the optimal ordering

Phase II: Algorithm Implementation and Write-Up

Overview

Release date: November 13th
Due date: **December 1st**
Deliverables: Output files for each input file released, your solution code, and a write-up of your methods.

You will design an algorithm to figure out the relative ordering of as many wizards as possible and run it on the given pool of input files. Of course, a perfect ordering that satisfies all constraints does exist, since the group that submitted the input would have provided it to us. Your task is to find such a perfect ordering, or get as close as you can. You will submit your solution for each problem as an output file. The top 5% of teams will receive extra credit. You must also submit your code and a write-up (see more details below).

Details

After all input files have been submitted, we will release a subset of the harder problems in a zip file for you to run your algorithm on. We have released starter code to parse input files in the Python Script `solver.py`.

In this phase, you will submit output files for each input file released. Each output file must be named `output#.out` where # is the number of the corresponding input file. For instance, the output to the file `input13.in` must be named `output13.out`. All of your output files must be zipped in a folder named `outputs` and uploaded to Gradescope. We will provide you with a Python script `output_validator.py` that will validate the format of your output file, and print how many constraints you have satisfied for that problem. In addition, we will be running a live anonymous leader board for groups to see how they are doing in comparison to other groups in the class.

Your one-page write-up should describe your algorithm clearly. If you use non-standard libraries, you need to explicitly cite the libraries you used, instructions to install them, and explain why you chose to use those libraries. You must submit your code and write-up on Gradescope, along with the output files to receive any points for this phase. Be sure to **include your partners on Gradescope**.

You may use any programming language you wish. However, your code must be able to run on only **one** machine and complete in a reasonable amount of time. Note that this rule prohibits you from using multiple machines in Soda or Cory Hall too (such as the Hive or Ashby machines). You are responsible for terminating your running process before starting a new one. Leaving processes running in the background is not an excuse to violate this policy. You will receive no credit for the project if you violate these requirements. You may **only** use free services (academic licenses are fine). We will link a Google form to anonymously report academic dishonesty on the project Piazza post.

Note that we reserve the right to run the code submitted by any team, and that the code of the top 10 teams will certainly be run by us.

Sample Output

```
Harry Hermione Severus Albus
```

Grading

Your grade will be dependent on how many constraints you satisfy in a file. We will normalize each file to be worth the same amount of points.

This phase is worth 70% of your grade total. 65% of your project grade will be dependent on your performance relative to the rest of the class. The final 5% of your grade will be dependent on your code, which will be graded on style, and the corresponding write up.

Phase III: Peer-grade input files

Overview

Release date: November 17th
Due date: **December 7th**
Deliverables: Output files for each input file assigned.

In this phase, each group will be randomly assigned some *other* input files to run their solver on. The output in this phase will be used to grade the difficulty of the input files submitted by other students. You are not expected to modify your algorithm or code from Phase II, but instead to run your existing solver. Note that these input files will be released to you one week into Phase II, and you are free to work and submit outputs to these files during Phase II too. The output format in this phase is the same as Phase II, and you will name and zip the files in the same manner as in Phase II but will submit on an assignment corresponding to Phase III. Once again, **remember to add your partners to the submission on Gradescope**.

Grading

Submitting output files in this phase is worth 15%. Each input file you are assigned will be worth an equal amount. For instance, if you are assigned 40 input files, each will be worth $(\frac{15}{40})\%$. You will **not** be tested on your performance on the files in this phase, however, do note that it is in your favor to make an attempt to solve the assigned input files, since your input files are graded relative to those of other groups.