CS3000: Algorithms & Data — Spring 2019 — Paul Hand

Homework 8
Due Wednesday 4/17/2019 at 2:50pm via Gradescope

Name:
Collaborators:

- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Wednesday 4/17/2019 at 2:50pm via Gradescope. No late assignments will be accepted. Make sure to submit something before the deadline.
- Solutions must be typeset in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly forbidden.

Problem 1. *Seating Arrangements*

Several families are having a dinner party. To be more social, they would like to arrange seating so that no two members of the same family are at the same table. There are p families with family i having f_i members, and there are q tables with each table j having t_j seats. The goal is to determine if there exists a seating arrangement where no two members of the same family are at the.

More precisely, we have the following inputs and correct outputs:

- **Input:** The values $p, f_1, \ldots, f_p$ and $q, t_1, \ldots, t_q$.
- **Output:** If possible, an assignment of guests to tables[1] such that (1) every guest is assigned, (2) no family has more than one member assigned to any table, and (3) no table has too many guests. Otherwise, $\perp$ meaning that no such assignment is possible.

Show how to obtain a polynomial time algorithm for the seating arrangement problem by reducing it to the integer maximum flow problem that we know how to solve.

(a) How do you map an input to the seating arrangement problem into an input into the maximum flow problem?

Solution:

(b) How do you map an output for the maximum flow problem to an output for the seating arrangement problem? Be sure to state what properties of the output you are relying on.

Solution:

(c) Give a rigorous argument that your algorithm correctly solves the seating arrangement problem provided that you get a correct output to the integer maximum flow problem you created in part (a).

Solution:

(d) What is the running time of the entire algorithm, including the time to compute the maximum flow? You may assume you have access to a state-of-the-art maxflow solver that runs in time $O(\text{nodes} \times \text{edges})$.

Solution:

[1] Note that it doesn't matter *which* member of a family is assigned to a table, so you only need to specify information like "the 3 members of family 1 are assigned to tables 2, 5, and 12."