Homework 6

Due: Oct. 31, 2017 at 6:00 PM (early)
Nov. 3, 2017 at 6:00 PM (on time)
Nov. 5, 2017 at 6:00 PM (late)

This is a partner homework as usual: work on all problems together; you are responsible for everything you and your partner submit and it is an academic code violation to submit something that is not yours. Check for your partner in the Google doc that was sent out. **If you don't have a partner, please let us know.**

One of the features of good writing style is to say everything once and no more than once. When you are writing up these problems, please find a way to organize your presentation so that similar repeated parts of your argument are instead compressed into a single unit. This will make it easier for you to write and for us to read, and will sound more professional.

Recall the standard way for Local Area Dad to prove that problem P' is NP-hard:

1. First (carefully) pick an NP-hard problem (for example, graph 3-coloring).
2. Then you need to show that Local Area Dad's problem is *at least as hard as graph 3-coloring*: show that if Local Area Dad can magically solve his problem P' (for any input), then you could use him to solve graph 3-coloring:
   (a) Consider an arbitrary instance x of the graph 3-coloring problem.
   (b) For any instance x of the graph 3-coloring problem, show how to translate x into an instance y of Local Area Dad's problem P', making sure that you describe a everything needed to define an instance of Local Area Dad's problem.
   (c) Assume that Local Area Dad solves this instance y of P'.
   (d) Then you need to show that solutions to the two problems are equivalent, that the instance x of graph 3-coloring can be solved **if and only if** the translated instance y of Local Area Dad's problem P' can be solved. You can prove this by showing:
      i. Every valid 3-coloring of x corresponds to a valid solution to the instance y of P'.
      ii. Every valid solution to the instance y of P' corresponds to a valid 3-coloring of x.

      (Parts i and ii are essentially showing that the answer to the 3-coloring instance is yes if and only if the answer to the corresponding instance of P' is yes; there are other ways of proving that "yes maps to yes and no maps to no" besides the method in i and ii.)

Note that most NP-hard problems are phrased as *decision problems*, namely "is a graph 3-colorable", but these are polynomially equivalent to the corresponding *search problems*, "find a 3-coloring of the graph", and this distinction is not going to be emphasized here.

Here are some NP-hard problems relevant to this assignment (google them if you are unfamiliar!): Independent-Set, even when every node has degree at most 3; Set-Packing.

S/NC track: do problems 1, 2, and 4 *and indicate "***S/NC***" on each problem.*

Problem 1

Three things The Guy Who Lived Across From You Freshman Year must do at their party

(20 points total) Winnie, The Guy Who Lived Across From You Freshman Year, is throwing a party for his college friends. He has certain ideas about how his guests should interact. From a list of n potential guests, he knows who is friends with whom, and wants to use this information to figure out who to invite to his party. In each case, determine whether there is a polynomial (Paul-ynomial) time algorithm for Winnie, or whether the problem is NP-hard. State and prove the algorithm, or prove NP-hardness in each case.

1. Winnie wants to ensure that everyone has a group to hang out with at the party: each guest should know at least 4 other guests. He wants to maximize the number of guests at the party.
2. Winnie wants to ensure both that everyone has a group to hang out with *and* that everyone can meet new people: each person should (1) know at least 4 other guests; and (2) *not know* at least 4 other guests. He wants to maximize the number of guests at the party.
3. Winnie wants to help his guests "network": each guest must know *at most 4* other guests at the party. He wants to maximize the number of guests at the party.

Hint: Does changing the number "4" to a smaller number clarify the situation?

Problem 2

Help Little League Coach with this NP-hard problem

Little League Coach has offered to coach n different teams. Each team has a schedule for when the coach would need to show up to train them: the 12th team might say "to train us you need to attend a session from 2:37 PM to 3:19 PM 5 days from now, AND a session from 8:08 AM to 8:51 AM 7 days from now, AND a session from 1 PM to 2 PM 10 days from now." To uphold high coaching standards, the coach cannot miss any of the sessions for a team they want to train. Of course, this means the coach cannot train teams that have overlapping sessions.

Given these potential schedules, you want to find the maximum number of teams the coach can train.

1. (10 points) Show this problem is NP-hard.
2. (10 points) Suppose that the coach is now allowed to miss one training session, total, among all the teams. That is, the coach's schedule may contain one overlap. Show that this problem is still NP-hard.

Problem 3

Ever wonder why crosswords are so hard?

(20 points) In this problem, you will show that crossword puzzles are NP-hard. We consider a simplified variant of crosswords where there are no clues, and also no "black squares": a crossword solver is given a rectangular board, and a dictionary, and needs to find a way to fill the board with letters such that each row and each column is a valid word from the dictionary. (The "dictionary" is an input to the problem, and thus is not meant to be English, but just an arbitrary list of sequences of letters that you, the puzzle maker, get to choose. Remember that the dictionary is part of the input to the problem, so cannot be too large.) Show that this problem is NP-hard.

A few hints:

1. Remember, you need to show how to embed *some* NP-hard problem in the crossword problem, and in particular, your construction might only consider crosswords of a very particular form. (You get to write the dictionary.)
2. Consider a rectangular board (i.e., not square) so that in your dictionary: 1) words that can appear as a row and 2) words that can appear as a column form distinct sets.
3. For an $n \times m$ crossword, consider having the only valid length-n words be the $n + 1$ words consisting of A's and B's with *at most one B*.

Problem 4

Want to know the best way to get rid of competitors?

(20 points) A reality TV series wants to have k contestants, where *every pair* Valiantly competes in front of a judge to see who is the better singer, and each of these "sing-offs" will be broadcast to their eager audience. At the end, the TV show will produce a ranking of the k contestants, where the best contestant beat all the others in sing-offs; the second best contestant lost to the best contestant but beat all the remaining contestants, etc.

However, the producers actually filmed this with $n \geq k$ contestants (filmed all n-choose-2 pair sing-offs), and found the following issue: sometimes contestant A beats contestant B, who beats contestant C, etc. who beats contestant Z, who beats contestant A. This means that the producers cannot possibly produce a consistent final ranking.

What the producers have to do in this case is "edit out" some of the n contestants, so that the k contestants who remain can be ordered as desired. Their goal is to remove *as few contestants as possible*, so that the contestants that remain can be ranked unambiguously.

You have been hired as a consultant to the show, to figure out which contestants to "edit out of existence". You soon realize that finding the *optimum* number is NP-hard, and instead you decide to come up with a *3-approximation*. Namely, your task is to find a polynomial time algorithm that removes at most 3 times as many people as the optimal scheme would.

Hint: Show that if there is an "inconsistent sequence" $\mathrm{A}, \mathrm{B}, \mathrm{C}, \ldots, \mathrm{Z}$ where each one beats the one after her and the last one beats the first one, then there *must* exist an inconsistent sequence of length 3.

Problem 5

University launches full-scale investigation to determine if scheduling is NP-hard

(20 points) Find either a polynomial time algorithm or a proof of NP-hardness for the following problem (essentially a restricted version of problem 2.1, but a generalization of problem 1.1 from the previous homework): There are n classes you want to take, and each class meets **twice** per week, between some arbitrary times (Tuesdays from 11:13 AM to 12:54 PM, and Fridays from 4 PM to 8:57 PM, for example). Find the maximum number of classes you can take that do not overlap.

Important new rules for this problem: This problem is harder than usual, and we will email out hints over the course of the week. However, *you* will have to come up with the hints! The game is as follows: when you think you have a significant idea for how to make progress on this problem, send a brief email to the TA list outlining the idea. If you email us an idea worth sharing, you and your partner will get **extra credit**! The **earlier** you send us the hint, the **more credit** you get! (If multiple groups email us the same idea before we send it out, they will all get credit; each partner group can receive extra credit at most once for this problem.) Start thinking about this problem **early**.