CS 170 HW 11

Due on 2018-11-11, at 9:59 pm

1 (★) Study Group

List the names and SIDs of the members in your study group.

2 (★★★) 2-SAT and Variants

In this problem we will explore the variant of 3-SAT called 2-SAT, where each clause contains at most 2 literals (hereby called a 2-clause).

(a) Show that 2-SAT is in P. (Hint: Note that the clause $x \vee y$ is equivalent to the implications $\neg x \Rightarrow y$ and $\neg y \Rightarrow x$. If a 2-SAT formula is unsatisfiable, what must be true about the set of all such implications?)

(b) The problem of Max-2-SAT is defined as follows. Let $C_1, \ldots, C_m$ be a collection of 2-clauses and k a non-negative integer. We want to determine if there is some assignment which satisfies at least k clauses.

The problem of Max-Cut is defined as follows. Let G be an undirected unweighted graph, and k a non-negative integer. We want to determine if there is some cut with at least k edges crossing it. Max-Cut is known to be NP-complete.

Show that Max-2-SAT is NP-complete by reducing from Max-Cut. Prove the correctness of your reduction.

3 (★★★) Reduction to (3,3)-SAT

Consider the (3,3)-SAT problem, which is the same as 3-SAT except each literal or its negation appears *at most* 3 times across the entire formula. Notice that (3,3)-SAT is reducible to 3-SAT because every formula that satisfies the (3,3)-SAT constraints satisfies those for 3-SAT. We are interested in the other direction.

Show that 3-SAT is reducible to (3,3)-SAT. By doing so, we will have eliminated the notion that the "hardness" of 3-SAT was in the repetition of variables across the formula. Give a precise description of the reduction and prove its correctness.

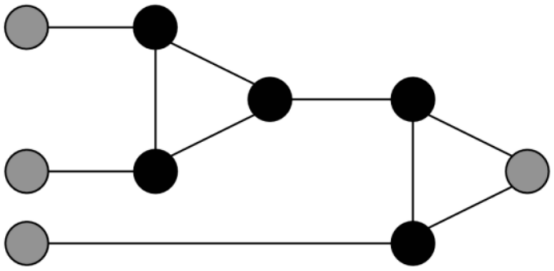
4 (★★★) Reduction to 3-Coloring

Given a graph $G = (V, E)$, a valid 3-coloring assigns each vertex in the graph a color from $\{0, 1, 2\}$ such that for any edge (u, v), u and v have different colors. In the 3-coloring problem, our goal is to find a valid 3-coloring if one exists. In this problem, we will give a reduction from 3-SAT to the 3-coloring problem, showing that 3-coloring is NP-hard.

In our reduction, the graph will start with three special vertices, labelled "True", "False", and "Base", and the edges (True, False), (True, Base), and (False, Base).

(a) For each variable x_i in a 3-SAT formula, we will create a pair of vertices labelled x_i and $\neg x_i$. How should we add edges to the graph such that in any valid 3-coloring, one of $x_i, \neg x_i$ is assigned the same color as True and other is assigned the same color as False?

(b) Consider the following graph, which we'll call a "gadget":



Show that in any valid 3-coloring of this graph which does not assign the color 2 to any of the gray vertices, the gray vertex on the right is assigned the color 1 only if one of the gray vertices on the left is assigned the color 1.

(c) We observe the following about the graph we are creating in the reduction:

(i) For any vertex, if we have the edges (v, False) and (v, Base) to the graph, then in any valid 3-coloring v will be assigned the same color as True.

(ii) Through brute force one can also show that in the gadget, for any assignment of colors to gray vertices such that:

(1) All gray vertices assigned the color 0 or 1

(2) The gray vertex on the right is assigned the color 1

(3) At least one gray vertex on the left is assigned the color 1

There is a valid coloring for the black vertices in the gadget.

Using these observations and your answers to the previous parts, give a reduction from 3-SAT to 3-coloring. Prove your reduction is correct.

5 (★★) Coloring: Two's Company, Three's a Crowd

In the last problem, you proved that 3-coloring is NP-hard. For any $k \geq 2$, consider the k-coloring problem, which is the same as the 3-coloring problem except that there are k colors to choose from. In this problem, we'll see how the hardness of the problem is affected by the value k.

(a) Give an efficient algorithm for the 2-coloring problem. Just a brief description of the algorithm suffices.

(b) For any $k \geq 3$, give a polynomial-time reduction from the k-coloring problem to the $(k+1)$-coloring problem. Just a brief description of the reduction suffices. (By induction, this shows that for any constant $k \geq 3$, the k-coloring problem is NP-hard.)

6 (★★★) Vertex Cover Approximation

In the textbook, we saw that finding a maximal matching and outputting all endpoints of edges in the matching is a 2-approximation, i.e. the size of the solution output by this algorithm is at most twice the size of the best solution. In this problem, we'll give another 2-approximation. For a graph $G(V, E)$, consider the following linear program for the vertex cover problem:

$$\begin{aligned} \min & \sum_{v \in V} x_v \\ \text{s.t. } \forall (u,v) \in E: & \quad x_u + x_v \quad \geq 1 \\ \forall v \in V: & \quad x_v \geq 0 \end{aligned}$$

(In an integer solution, we would set $x_v = 1$ if we include v in our vertex cover and $x_v = 0$ otherwise, and we have the constraint that for every edge we must include one of its endpoints.)

(a) How does the optimal value of this linear program compare to the size of the minimum vertex cover? (Note that the optimal solution to the linear program does not necessarily set every x_v to 0 or 1.)

(b) Describe a procedure which given only the optimal solution to the linear program, outputs a vertex cover of size at most twice the size of the minimum vertex cover. Prove both that your procedure finds a feasible vertex cover and the size guarantee. No runtime analysis needed.

(Hint: Your procedure should output a solution similar to the LP solution, i.e. it should only include vertices for which x_v is sufficiently large)