

## Homework 4, CS520, Spring '09, due Thursday, March 26

### Modified 3/20 12:15 to add a problem about the simplex algorithm.

I've modified how the text suggests you write up your answers to make it easy to show you understand the solutions without having to write out all the tedious details.

1. The problem of 34-3: showing that graph coloring is NP-hard.
  - (a) Satisfy yourself about the result of part (e). You don't need to write anything.
  - (b) To show you understand the full reduction, draw the graph you get from  $(x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$
2. 35-2: Evidence that a constant approximation ratio for maximum clique is a long shot.

For part (a), briefly describe a mapping from sets of vertices of  $G$  to sets of vertices of  $G^{(k)}$  such that cliques of  $G$  map to cliques of  $G^{(k)}$  and non-cliques of  $G$  map to non-cliques of  $G^{(k)}$ . If your mapping maps each set  $S$  to a set of size  $|S|^k$  you are done with part (a). Keep in mind that a vertex can appear multiple times in the tuple.

3. The problem of 35-4: maximum matching.

A solution to part b is to iteratively pick an arbitrary edge and kill off its two endpoints, until there are no more edges. The killed-off vertices are the set  $T$  mentioned in part d. Obviously, the vertices not in  $T$  are an independent set.

For your written solution to 35-4, just argue that that  $|T|$  is an upper bound on the size of a maximum matching, and use this to get your approximation bound.

4. 35-5: parallel machine scheduling.
  - First show that the problem is NP-hard.
  - Work parts (a), (b), and (d).
5. The simplex algorithm.

Review the book's example of how to run the simplex algorithm. Then show how the algorithm works on the following linear program:

$$\text{Maximize } 5x_1 + 4x_2 + 3x_3$$

Subject to:

$$2x_1 + 3x_2 + x_3 \leq 5$$

$$4x_1 + x_2 + 2x_3 \leq 11$$

$$3x_1 + 4x_2 + 2x_3 \leq 8$$

$$x_1, x_2, x_3 \geq 0.$$

Each time you perform a pivot, you must select a variable whose increase increases the value of the objective function, and swap it with one of the basic variables. Sometimes you will have a choice of such variables; choose the one with the smallest subscript