CSOR 4231: HW 6

Problem 1 Prove Lemma 26.2 in the textbook.

Problem 2 Let G be a graph and f be a maximum flow. Give an algorithm that outputs a series of at most $|E|$ augmenting paths that, when augmented along would give rise to the flow f.

Note that you are not asked to give a new maximum flow algorithm, you asked how, given the maximum flow f, you can recreate a series of augmenting paths.

Problem 3

Show that for any decision problem in NP, there is an algorithm that can solve it that runs in time $2^{O(n^k)}$, for some contant $k > 0$.

Problem 4 Given an integer $m \times n$ matrix A and an integer m-vector b, the *0-1 integer-programming problem* asks whether there exists an integer n-vector x with elements in the set $\{0, 1\}$ such that $Ax \leq b$. Prove that 0-1 integer programming is NP-complete. (Hint: Reduce from 3-SAT).

Problem 5. Bonnie and Clyde have just robbed a bank. They have a bag of money and want to divide it up. For each of the following scenarios, either give a polynomial-time algorithm, or prove that the problem is NP-complete. The input in each case is a list of the n items in the bag, along with the value of each.

1. The bag contains n coins, but only 2 different denominations: some coins are worth x dollars, and some are worth y dollars. Bonnie and Clyde wish to divide the money exactly evenly.
2. The bag contains n coins, with an arbitrary number of different denominations, but each denomination is a nonnegative integer power of 2, i.e., the possible denominations are 1 dollar, 2 dollars, 4 dollars, etc. Bonnie and Clyde wish to divide the money exactly evenly.
3. The bag contains n checks, which are, in an amazing coincidence, made out to "Bonnie or Clyde." They wish to divide the checks so that they each get the exact same amount of money.
4. The bag contains n checks as in part (c), but this time Bonnie and Clyde are willing to accept a split in which the difference is no larger than 100 dollars.

Problem 6 Consider the following algorithm for Vertex Cover. Start with $C = \emptyset$. At each step, choose the vertex v with highest degree, add it to the cover C and then delete v and all its incident edges from the graph. Stop when the graph is empty. Does this algorithm give a 2-approximation for vertex cover? Either argue that it does, or show that it does not.