Problem Set 7

This problem set is due at **11:59 pm** on **Wednesday, November 4th, 2015**. The exercises are **optional**, and should not be submitted.

Exercise 7- 1: (Optional) You are given two decision problems L_1 and L_2 in **NP**. You are given that $L_1 \leq_p L_2$. For each of the following statements, state whether it is true, false, or an open question. It might be helpful to consider the implications of each statement if $\mathbf{P} = \mathbf{NP}$ and if $\mathbf{P} \neq \mathbf{NP}$. Prove your answers.

(a) If $L_1 \in \mathbf{P}$, then $L_2 \in \mathbf{P}$.

(b) If $L_1 \in \mathbf{NP}$-complete, then $L_2 \in \mathbf{NP}$-complete.

(c) If $L_2 \in \mathbf{P}$, then $L_1 \in \mathbf{P}$.

(d) If $L_2 \in \mathbf{NP}$-complete, then $L_1 \in \mathbf{NP}$-complete.

(e) Suppose L_2 is solvable in $O(n)$ time. Then L_1 is also solvable in $O(n)$ time.

(f) If $L_2 \leq_p L_1$, then L_1 and L_2 are **NP**-complete.

Problem 7- 1: Solving a Geology Problem Set

You and two of your friends, Alice and Bob, decide to take a Geology elective together. You split the 3 problems of the first problem set among yourselves. Below are the descriptions of each of your assigned problems:

- Alice's Problem - BOX-PACKING(A, m) : Alice is given n rocks of weights $A = [a_1, a_2, \ldots, a_n]$ units and m boxes that can support up to 1 unit of weight each. Alice's job is to pack the rocks into the boxes, such that all rocks are packed and no box is over its capacity.
- Bob's Problem - EQUAL-WEIGHT(B) : Bob is given n rocks of weights $B = [b_1, b_2, \ldots, b_n]$ units. Bob's job is to divide the n rocks into two piles, B_1 and B_2, of equal weight, such that every rock is either in pile B_1 or pile B_2.
- Your Problem - DESIRED-WEIGHT(C, w) : You are given n rocks of weights $C = [c_1, c_2, \ldots, c_n]$ and a target weight w. Your job is to find a set $G \subseteq C$ such that the sum of the weights of rocks in G is exactly equal to w.

(a) Show that each of the above problems is in **NP**.

(b) After struggling for a long time with your problem, DESIRED-WEIGHT, you realize it is **NP**-complete (equivalent to SUBSET-SUM). Your friends have been unable to solve their problems and would like to show that their problems are also NP-complete. Give a reduction from your DESIRED-WEIGHT problem to Alice's BOX-PACKING problem. **Hint**: Consider two steps. First show DESIRED-WEIGHT reduced to EQUAL-WEIGHT. Then show EQUAL-WEIGHT reduced to BOX-PACKING.

(c) You now consider ways to modify your problem to make it easier to solve. Consider the same DESIRED-WEIGHT problem, but with the assumption that every rock weight is of integer value and bounded by constant k. ($\forall i, c_i < k$). Describe an algorithm to solve this K-BOUNDED-INTEGER-DESIRED-WEIGHT problem in time polynomial in n and k. Include running time analysis and brief argument for correctness.

(d) Consider another simplification of your DESIRED-WEIGHT where the inputted rock weights C is sorted and has the property that each rock is greater than twice the weight of the rock before it.

$$c_i > 2c_{i-1} \forall i$$

Describe a polynomial time algorithm to solve the SUPERINCREASING-DESIRED-WEIGHT problem. Include running time analysis and brief argument for correctness. **Hint**: First show $c_i > \sum_{j=1}^{i-1} c_j$

(e) Say you have a black-box algorithm for DECISION-DESIRED-WEIGHT that runs in $O(1)$ time. This algorithm takes in C and w and returns YES if a valid subset G exists, and NO otherwise. Now consider the SEARCH-DESIRED-WEIGHT problem that requires the valid subset G to be specified. Describe a polynomial time algorithm using the DECISION-DESIRED-WEIGHT black box to solve the SEARCH-DESIRED-WEIGHT problem. Include running time analysis and brief argument for correctness.

Problem 7- 2: Variants of Max-Flow

In this problem, we examine NP-hard variants of the max-flow problem. In each variant, the input incldues a directed graph $G = (V, E)$ with source s and sink t.

(a) **SWAP-FLOW** In this variant, you are allowed to swap the capacities of outgoing edges for each vertex. Formally, you are given a function C which maps each vertex u to a set $\{k_1, k_2, \ldots, k_n\}$ of size n, where each k_i is a positive integer, and n is the outdegree of u. Call c a valid capacity function if for all vertices u, $\bigcup_{v|(u,v)\in E}\{c(u,v)\} = C(u)$. Find the maximum flow across all valid capacity functions.

Prove that SWAP-FLOW is NP-hard by reducing 3SAT to SWAP-FLOW.

(b) **ALL-OR-NONE-FLOW** In this variant, you must either fully utilize an edge or not use it at all. Formally, you are given a capacity function c, and you want to find the max-flow f such that for each edge (u, v), $f(u, v)$ is equal to either 0 or $c(u, v)$.

Prove that ALL-OR-NONE-FLOW is NP-complete by proving it is in NP and then proving it is NP-hard by reducing the EQUAL-WEIGHT problem from problem one to ALL-OR-NONE-FLOW.

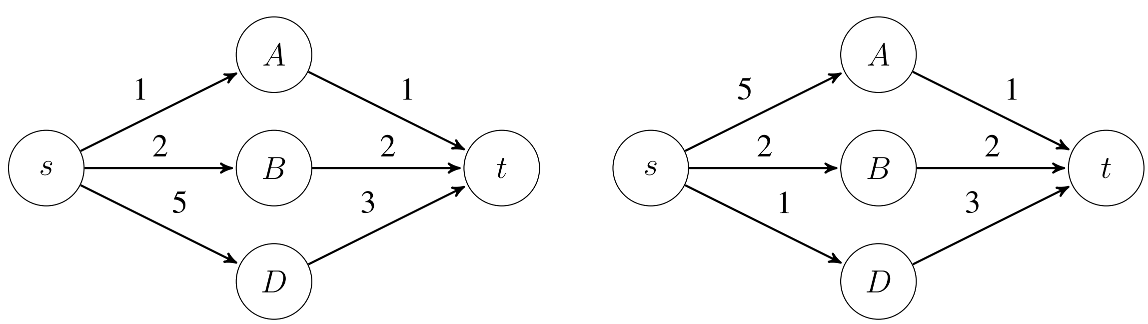

Figure 1: Two graphs with capacities drawn for each edge. Both graphs have a valid capacity function for a SWAP-FLOW instance with $C(s) = \{1, 2, 5\}, C(A) = \{1\}, C(B) = \{2\}$, and $C(D) = \{3\}$. In this example, the max flow over all capacity functions is 6, which can be achieved with the capacity function in the left graph, but not the right.