1. In Extra Practice 4 there was a problem where we gave a Turing Machine for this language: $L_1 = \{a^n b^n c^n : n \geq 0\}$.

   The answers posted provided a Turing Machine with the following behavior:

   See an a at the beginning, cross it off, move forward to find a b, cross it off, move forward to find a c, cross it off, rewind to an a, repeat. If you don't find a b or c on the move forward, reject (this is done implicitly by not having a transition, although explicit is permitted of course). If when we're rewinding we eventually find a space, move forward and check that everything is crossed off, and then accept when we find the space on the other side.

   Give the running time for this Turing Machine in big-$\mathcal{O}$ notation.

2. In an earlier lecture, we saw a Turing Machine that recognizes $L_3 = \{a^i b^j c^k : i \times j == k \text{ and } i, j, k \geq 1\}$. Give the running time in $\mathcal{O}$-notation in terms of n, the length of the provided input tape. Is this a reasonable description of the running time's growth rate?

3. Consider the following boolean satisfiability problem:

   $\phi = (\overline{x_2} \vee x_3 \vee \overline{x_4})(\overline{x_1} \vee x_3 \vee x_5)(\overline{x_2} \vee x_3 \vee \overline{x_5})(\overline{x_2} \vee x_4 \vee x_5)(\overline{x_1} \vee x_4 \vee \overline{x_5})(\overline{x_2} \vee x_4 \vee x_5)(x_2 \vee x_3 \vee \overline{x_4})(\overline{x_1} \vee \overline{x_3} \vee x_4)(\overline{x_1} \vee x_3 \vee x_5)(\overline{x_1} \vee \overline{x_2} \vee x_4)(\overline{x_1} \vee x_3 \vee \overline{x_5})(x_1 \vee \overline{x_2} \vee x_3)$

   (a) Find a satisfying truth value assignment (I promise there is one).

   Note: the following problems can get tedious. I encourage you to work on these (like any other extra practice problem) with your group. I also encourage you to ***stop*** *once you "get the point" – where finishing the problem is more paperwork than adding understanding.*

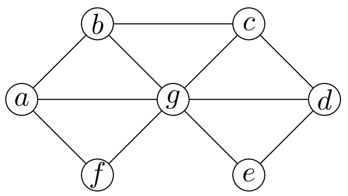
   (b) Build the graph that we would use to solve this, if we had a solver for Independent Set readily available.

   (c) Build the graph that we would use to solve this, if we had a solver for 3-Color available to us.

   (d) Suppose we had a Subset Sum solver available to us. Give the set S and the value t that we would provide that solver.

   (e) Suppose we had a Hamiltonian Path solver available to us. Draw a graph such that a Hamiltonian Path in that graph would correspond to a valid truth value assignment for ϕ.

4. Does the following graph have a Hamiltonian Path? Either explain why it does not or provide a Hamiltonian Path. Note that this graph is undirected, while the graph from lecture was directed.



5. In lecture, we saw that SUBSET SUM is in $\mathcal{NP}$ (we also saw it is $\mathcal{NP}$-complete, but that isn't the point here).

   Suppose you had a polynomial-time algorithm (or Turing Machine, however you prefer to think of this) for the decision version of the SUBSET SUM problem. Note that this means the algorithm only outputs "yes" or "no" – and you can't inspect the source code or schematic to see how it works! Just the same, show how you can use this to create a deterministic, polynomial-time algorithm that determines which subset, if any, of the input set adds to the target value. If multiple subsets do, you need only provide one of them.

   How many calls to the decider does your function make?

   Note: If this were for credit, solutions that do not follow directions would get zero credit; this includes the dynamic programming algorithm for this problem that you probably saw in CompSci 161.

$\mathcal{NP}$-complete proofs

6. The following is a version of the Independent Set Problem. You are given a graph $G = (V, E)$ and an integer k. For this problem, we will call a set $I \subset V$ *strongly independent* if, for any two nodes $v, u \in I$, the edge (v, u) does not belong to E, and there is also no path of two edges from u to v; that is, there is no node w such that both $(u, w) \in E$ and $(w, v) \in E$. The STRONGLY INDEPENDENT SET problem is to decide whether or not G has a strongly independent set of size at least k. Prove that STRONGLY INDEPENDENT SET is $\mathcal{NP}$-complete.

7. The SET PACKING problem is as follows. We are given n sets $S_1, S_2, \ldots S_n$ and an integer k. Our goal is to select k of the n sets such that no selected pair have any elements in common. Prove that this problem is $\mathcal{NP}$-complete.

8. The STINGY-SAT problem has input that looks like an instance of the 3-SAT problem, but the literals are *never negated*. It is possible to satisfy all clauses by simply setting all literals to true. We are additionally given a number k, and are asked to determine whether we can satisfy all clauses while setting *at most* k literals to be true. Prove that this problem is $\mathcal{NP}$-complete.

9. In a graph G, a DOMINATING SET in a subset $V' \subseteq V$ of the vertices such that every vertex either is in V' or is adjacent to one that is. Prove that DOMINATING SET is $\mathcal{NP}$-complete.

   Note that this is NOT the same problem as VERTEX COVER

10. Consider the problem of PARTITION: we are given a set of numbers (if you prefer, you may think of these as positive integers; however, that is not necessarily part of the problem statement). We want to answer the boolean question: is there a subset of these elements whose sum is exactly half of the total?

    (a) Express this as a language recognition problem.

    (b) Prove that your language from the previous part is in $\mathcal{NP}$.

(c) Observe that we could decide this language by making a call to Subset Sum's decider with parameter $t = (1/2)\sum x_i$. Why does this not prove anything about the difficulty of this problem?

(d) Prove that this problem is $\mathcal{NP}$-complete.

Challenge Problems

The following two problems are more difficult than the above.

11. The Steiner Tree problem is as follows. Given an undirected graph $G = (V, E)$ with nonnegative edge costs and whose vertices are partitioned into two sets, R and S, find a tree $T \subseteq G$ such that for every $v \in R$, $v \in T$ with total cost at most C. That is, the tree that contains every vertex in R (and possibly some in S) with a total edge cost of at most C.

12. In the Min-Cost Fast Path problem, we are given a graph $G = (V, E)$ along with non-negative integer times $t(e)$ and non-negative costs $c(e)$ on each edge. Our goal is to determine if there is a path P from s to t such that $\sum_{e \in P} t(e) \leq T$ and also that $\sum_{e \in P} c(e) \leq C$, for some given integers T and C. Prove that this problem is $\mathcal{NP}$-complete.

Complexity Problems from CompSci 260P

When I teach CompSci 260P, at the midterm, students are given two problems and have to identify which of the two is $\mathcal{NP}$-complete and which is polynomial-time solvable. They then need to the former with a proof and the latter with a brief explanation. What follows are some samples of questions from real or diagnostic exams for that class.

For each numbered problem, the difficulty (for students) is identifying which is which, then completing the proof. Once each has been correctly categorized, the complexity proof is appropriate to 162 level difficulty.

I *will not* ask you to distinguish which is $\mathcal{NP}$-complete and which is polynomial time solvable in 162 for credit, although it's a good exercise.

13. Consider the following two problems that may arise with respect to loading potentially hazardous cargo into trucks. We have a warehouse with n canisters, each of which have some form of hazardous material. We also have m trucks, each of which can hold up to k containers.

    - Version One : For each canister, there is a specified subset of trucks that may carry it. Is there a way to load all n canisters into m trucks such that no truck is overloaded and each container goes into a truck that can carry it?
    - Version Two : In this version, any canister can be placed on any truck. However, there are some pairs of containers that cannot be placed on the same truck. Is there a way to load all n containers onto the m trucks such that no truck is overloaded and no two containers are placed on the same truck when they cannot be?

14.
- The FOUR-SUM problem asks: given a vector of n distinct integers, each of which may individually be positive, negative, or zero, and a target integer T, is there a subset of *at most* four elements from n that add up to T?
- The CYCLE LENGTH problem is as follows. You are given a directed graph $G = (V, E)$ with weights w_e on the edges. Weights can be positive or negative or zero. You are also given a target value T. The cycle length problem returns true if and only if there is a simple cycle in G whose weights add up to exactly T.

15.
- BOUNDED DEGREE SPANNING TREE : is there a spanning tree of this graph such that each vertex in the tree has degree at most δ, for some input integer δ?
- BOTTLENECK SPANNING TREE : is there a spanning tree of this (weighted) graph such that each edge in the tree has cost at most B, for some input number B?

16.
- 4-COLOR: Given an undirected graph G, determine whether there is a mapping of vertices in G to 4 distinct colors such that no edge is monochromatic.
- 4-INDEPENDENT SET: Given a graph G, determine if there is some subset of the vertices V', $|V'| \geq 4$ such that no two vertices in V' share an edge.