Warm-up Exercises

1. Consider the following problem denoted DNF-SAT:
   Input: A DNF formula ϕ.[1]
   Question: Is ϕ satisfiable?
   Show that DNF-SAT is in P.

2. Define HALT $= \{(M, x) \mid$ the deterministic Turing machine M halts on $x\}$.
   Prove that HALT is NP-hard. Is it NP-complete?

3. Prove that IndSet, Clique and VertexCover are NP-complete.
   Suggestion: Show that 3SAT $\leq_p$ E3SAT $\leq_p$ IndSet $\leq_p$ Clique and IndSet $\leq_p$ VertexCover.

Exercises for Submission

1. We showed in class that NAE-3SAT $\leq_p$ MAX-CUT.
   (a) Complete the hardness proof of MAX-CUT by showing that NAE-3SAT is NP-hard.
   Suggestion: Show that 3SAT $\leq_p$ NAE-4SAT $\leq_p$ NAE-3SAT.
   (b) Show that MAX-CUT remains NP-complete when the input graph is simple.
   (c) Deduce that the following two languages are NP-complete.
      i. MAX-BISECTION= $\{ (G, k) \mid G$ has a cut $(S, V \setminus S)$ of size at least k s.t. $|S| = |V \setminus S|\}$.
      ii. BISECTION-WIDTH= $\{ (G, k) \mid G$ has a cut $(S, V \setminus S)$ of size at most k s.t. $|S| = |V \setminus S|\}$.

2. Recall that a graph is k-colorable if its vertices can be colored using up to k different colors in such a way that any two adjacent vertices have different colors.
   For any $k \in \mathbb{N}$ define the language k-Col $= \{ G \mid G$ is k-colorable $\}$.
   (a) Show that a graph is 2-colorable if and only if it has no cycle of odd length, and deduce that 2-Col is in P.
   (b) Prove that 3-Col is NP-complete.
   Hint: Reduce from NAE-3SAT. Given a formula generate a graph as follows: associate a vertex to each *literal*. Connect all these vertices to a vertex w and connect each variable vertex to its negation. Then, add a triangle for each clause and connect its vertices to the corresponding literals.
   (c) Deduce that the following languages are NP-complete.
      i. 2008-Col.
      ii. Coloring $= \{ (G, k) \mid G$ is k-colorable $\}$.
      iii. CliqueCover $= \{ (G, k) \mid$ the vertices of G can be partitioned into k sets, so that each set induces a clique$\}$.

[1] Recall that a DNF formula is a collection of clauses, each containing literals; we have an $\wedge$ between literals and an $\vee$ between clauses, e.g., $(x \wedge \bar{y} \wedge z) \vee (x \wedge \bar{z})$.

3. Assuming $\mathsf{P} \neq \mathsf{NP}$, can the following problems be solved by a polynomial time algorithm?
   (a) Input: A graph G and a positive integer k.
   Question: Does G contain a vertex of degree at least $\log_2 |V(G)|$ or a clique of size k?[2]
   (b) Input: A list of n positive integer numbers $A_1, \ldots, A_n$ and a number T. All the numbers are given in unary representation (i.e., the number k is represented as 1^k).
   Question: Does exist a subset $S \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{i \in S} A_i = T$?

4. A polynomial-time reduction f from a language $L \in \mathsf{NP}$ to a language $L' \in \mathsf{NP}$ is *parsimonious* if the number of witnesses of x is equal to the number of witnesses of $f(x)$.
   Show a parsimonious reduction from SAT to $\mathsf{3SAT}$ (where by a witness for $\mathsf{SAT}/\mathsf{3SAT}$ we mean a satisfying assignment).

5. A language $L \subseteq \{0,1\}^*$ is *sparse* if there is a polynomial p such that $|L \cap \{0,1\}^n| \leq p(n)$ for every $n \in \mathbb{N}$.
   (a) Show that every sparse language is in P/poly.
   (b) Mahaney's Theorem states that if there exists an NP-complete sparse language then $\mathsf{P} = \mathsf{NP}$.
      i. Notice that unary languages are also sparse, and so the claim we proved in class is the special case of Mahaney's theorem for unary languages. Explain why this proof breaks down if we try to extend it to arbitrary sparse languages.
      ii. Show that if $\mathsf{P} = \mathsf{NP}$ then there are sparse NP-complete languages.

[2] Recall that $V(G)$ denotes the vertex set of the graph G.