Homework 2

due Thursday, Jan 20, 2022

Directions: Do all problems.

Transitivity of Reductions

Problem 1. Let $f, g, h : \{0,1\}^* \to \{0,1\}$ be functions such that $f \leq g$ and $g \leq h$; prove that $f \leq h$. Deduce that if computing f is $\mathsf{NP}-$hard then so is computing g and h. Similarly, deduce that if h can be computed in polynomial time, then so can f and g.

Some More $\mathsf{NP}-$complete Problems

Problem 2. Prove that three of the following languages are $\mathsf{NP}-$hard.

- **Rules:** For each language you choose, you must give a polynomial time reduction from a known $\mathsf{NP}-$complete language. These can include any of the languages we proved $\mathsf{NP}-$complete in class, as well as any language in the following list (other than the language itself; *i.e.*, you cannot prove L $\mathsf{NP}-$complete by proving $\mathsf{L} \leq \mathsf{L}$).

- <u>CIRCUIT-SAT</u> $= \{\mathsf{C}$ boolean circuit : $\exists\, \mathbf{x} \in \{0,1\}^n$ st $\mathsf{C}(\mathbf{x}) = 1\}$.

- <u>QUAD-EQ</u> $= \{(\varphi_1, \ldots, \varphi_m)$ satisfiable 3-local quadratic equations over n boolean variables$\}$, where a 3-local quadratic equation over the variables $x_1, \ldots, x_n$ has the form

$$x_i x_j - x_k \equiv 1 (\text{mod } 2)$$

  for some $i, j, k \in \{1, \ldots, n\}$. We say that a sequence of 3-local quadratic equation over n boolean variables is satisfiable if there exists a $0/1$ assignment to each of the variables $x_1, \ldots, x_n$ which simultaneously satisfies all the equations.

  Hint: Reduce from $\mathsf{CIRCUIT-SAT}$; you may assume circuits can be constructed entirely of NAND gates.

- <u>VERTEX-COVER</u> $= \{(\mathsf{G}, k) :\ \exists\, S \subset \mathsf{V}$ st $|S| = k$ and $\forall\, (v, v') \in \mathsf{E},$ either $v \in S$ or $v' \in S\}$.

- <u>dHAMPATH</u> $= \{\mathsf{G}$ directed graph : $\exists$ Hamiltonian cycle$\}$, where a Hamiltonian cycle in a graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ is a list of vertices $v_0, v_1, \ldots, v_n$ such that $v_0 = v_n$ and each other vertex in V appears exactly once in the list and moreover $(v_{i-1}, v_i) \in \mathsf{E}$ for all $i = 1, 2, \ldots, n$.

- <u>HAMPATH</u> $= \{\mathsf{G}$ undirected graph : $\exists$ Hamiltonian cycle$\}$, where a Hamiltonian cycle in a graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ is a list of vertices $v_0, v_1, \ldots, v_n$ such that $v_0 = v_n$ and each other vertex in V appears exactly once in the list and moreover $(v_{i-1}, v_i) \in \mathsf{E}$ for all $i = 1, 2, \ldots, n$.

Hint: Reduce from $\mathsf{dHAMPATH}$.

- $\underline{\mathsf{IPROG}}$ $= \{S$ integer linear system of inequalities : $\exists$ satisfying assignment$\}$, where an integer linear system of inequalities, S, over the variables $\{x_1, \ldots, x_n\}$ is a set of inequalities of the form $a_1x_1 + \cdots + a_nx_n \geq b$ where $a_i, b \in \mathbb{Z}$; and we say S has a satisfying assignment if there exist integer values for the x_i such that each inequality in S is satisfied.

$\mathsf{2SAT} \in \mathsf{P}$

Problem 3. Describe a polynomial time algorithm which decides $\mathsf{2SAT}$. Deduce that $\mathsf{2SAT} \in \mathsf{P}$.

EXP vs. NEXP

The complexity classes EXP and NEXP are the analogues of P and NP but for exponential time, rather than polynomial time. So specifically,

$$\mathsf{EXP} = \bigcup_{c \geq 1} \mathsf{DTIME}(2^{n^c}); \ \mathsf{NEXP} = \bigcup_{c \geq 1} \mathsf{NTIME}(2^{n^c}).$$

Problem 4. Prove that if $\mathsf{P} = \mathsf{NP}$ then $\mathsf{EXP} = \mathsf{NEXP}$.

A Search-to-Decision Reduction for SAT

So far in class, we have been interested in Turing Machines which *decide* SAT and other languages in NP (*i.e.*, M takes a formula Φ and outputs 1 or 0 based on whether Φ is satisfiable or not; we say M *decides* SAT or equivalently that M solves the *decision version* of SAT). One could ask for more. We might want M to output a satisfying assignment for Φ if $\Phi \in \mathsf{SAT}$. In this case we say that M solves the *search version* of SAT.

Problem 5. Suppose we have oracle access to the decision function $f_{\mathsf{dSAT}} : \{\mathsf{CNFs}\} \to \{0, 1\}$ where $f_{\mathsf{dSAT}}(\Phi) = 1$ if and only if Φ is satisfiable. Describe a deterministic polynomnial time $f_{\mathsf{dSAT}}-$oracle algorithm which takes as input a CNF formula over n boolean variables $\{x_1, \ldots, x_n\}$ and by making $2n-1$ oracle calls to f_{dSAT} outputs a satisfying assignment to the $\{x_i\}$ whenever there is one. Deduce that solving the search version of SAT is no harder than solving the decision version. Such results are called *search to decision* reductions.