Assignment 2

Introduction to Complexity Theory (CS 721)

Instructor: Vishwas Bhargava
IIT Bombay

This assignment looks long, but that is only because each part is spelled out in full. Most parts want a short answer: for Questions 1, 3, 4, 5 a single paragraph of five or six lines per part is what is expected, not more. The reductions in Question 2 will naturally run a little longer. Aim for precision over volume.

1. A zoo of reductions.
Throughout the course we have written $A \leq_p B$ to mean "A reduces to B", i.e. B is *at least as hard* as A. There are, however, several distinct notions of reduction, differing in *what* they preserve. Below, $\mathrm{Sol}(x)$ denotes the set of witnesses / solutions of an instance x.

Type	**Notation**	**What it preserves**
Karp	$A \leq_p B$	$x \in A \iff f(x) \in B$, for a poly-time f (many-one).
Levin	$A \leq_p^L B$	A Karp f *together with* poly-time maps g, h that transport witnesses: g sends $w \in \mathrm{Sol}(x)$ to $\mathrm{Sol}(f(x))$, and h sends back.
Parsimonious	$A \leq_p^{\#} B$	A Karp f with $\lvert\mathrm{Sol}(x)\rvert = \lvert\mathrm{Sol}(f(x))\rvert$ (indeed a bijection of witnesses).
Cook (Turing)	$A \leq_p^T B$	Decide A by a poly-time machine with oracle access to B (adaptive, many queries).

Note the implications $A \leq_p^{\#} B \Rightarrow A \leq_p^L B \Rightarrow A \leq_p B \Rightarrow A \leq_p^T B$: each notion is a special case of the one after it.

For each item below, state which of the four types the reduction is (name the *strongest* that applies), and justify in one or two lines.

(a) The reductions NAE-3SAT $\leq$ Max-2SAT and 3SAT $\leq$ IndSet seen in class / Assignment 1. In each case: can you recover a witness of the source instance from a witness of the target? Do the witness counts match exactly?

(b) The Cook–Levin reduction of an arbitrary $L \in \mathbf{NP}$ to SAT. Which type is it? Using your answer, argue that an oracle for SAT lets you *find* a certificate for any $L \in \mathbf{NP}$ (not merely decide membership).
(Hint: combine the witness-recovery map h with the self-reducibility of SAT*: fix variables one at a time using the oracle to read off a satisfying assignment.)*

(c) Building on (b), can you produce an *explicit* proper 3-colouring of a given 3-colourable graph using a decision 3-colour oracle? (*You might have to add 3 more vertices and how would you add new edges to signify partial colouring?*)

2. An NP-completeness reduction (*do exactly one* of the two).
Prove that *one* of the following problems is **NP**-complete. In both cases, membership in **NP** is the easy direction; the substance is a reduction *from* 3SAT. Both appear in Sipser, *Introduction to the Theory of Computation* (3rd ed.); a PDF is here if you want to consult the gadget constructions.

(A) Solitaire. You are given an $m \times m$ board; each of the m^2 cells holds a blue stone, a red stone, or nothing. A *move* removes a stone from the board. You *win* if you can reach a position in which every column contains stones of only a single colour, and every row still contains at least one stone. Let

$$\textsc{Solitaire} = \{\, \langle G \rangle \mid G \text{ is a winnable initial configuration} \,\}.$$

Prove that SOLITAIRE is **NP**-complete. *(Suggested reduction: one column per variable, one row per clause; let a blue stone in a variable-column encode* TRUE *and red encode* FALSE*, and place stones so that each clause-row can be cleared iff the clause is satisfied. Make the "single colour per column" constraint force a consistent assignment.)*

(B) Hamiltonian path. HAMPATH is the set of $\langle G, s, t \rangle$ where G is a directed graph containing a Hamiltonian path from s to t (a path visiting every vertex exactly once). Prove that HAMPATH is **NP**-complete by a reduction from 3SAT. *(Suggested reduction: the "diamond"/variable-gadget construction, with one traversable diamond per variable whose left-to-right vs. right-to-left direction encodes the truth value, with a separate node per clause that the path may detour to from any diamond whose literal satisfies that clause.)*

In whichever you pick, prove *both* directions of correctness and check the reduction is polynomial-time, and mention what kind of reduction it is.

3. Hierarchy, undecidability, and separation. *(All parts.)*

(a) **Reading the time hierarchy theorem.** Recall the (deterministic) Time Hierarchy Theorem: if f is time-constructible and $f(n)\log f(n) = o\big(g(n)\big)$, then $\mathbf{DTIME}(f(n)) \subsetneq \mathbf{DTIME}(g(n))$. For each pair below, decide whether the two classes are equal or one is strictly contained in the other, and justify:
(i) $\mathbf{DTIME}(2^n)$ vs. $\mathbf{DTIME}(2^{n+1})$;
(ii) $\mathbf{DTIME}(2^n)$ vs. $\mathbf{DTIME}(2^{2n})$.

(b) **An undecidable timing property.** Show that the following language is undecidable:

$$\big\{\, \langle M \rangle \;\big|\; M \text{ is a machine that halts within } 100n^2 + 200 \text{ steps on every input of length } n \,\big\}.$$

(Don't diagonalize. Reduce from the halting problem: given (M, w), build a machine that ignores its input, simulates M on w, and blows past the bound exactly when M halts.)

(c) **"Superiority".** Say that a class $\mathcal{C}_1$ is *superior* to a class $\mathcal{C}_2$ if there is a machine M_1 in $\mathcal{C}_1$ such that for every machine M_2 in $\mathcal{C}_2$ and every large enough n, there is an input of size between n and n^2 on which M_1 and M_2 answer differently. Is $\mathbf{DTIME}(n^{1.1})$ superior to $\mathbf{DTIME}(n)$? Prove your answer.

4. A verifier view of NEXP. *(All parts.)*

(a) Recall the certificate (verifier) definition of **NP** from Definition 2.1 of Arora–Barak. Give an analogous definition of **NEXP** that does *not* mention nondeterministic machines. Does such a definition exist? Say why or why not, and identify exactly which quantities in the **NP** definition change.

(b) We call a language **NEXP**-complete if it is in **NEXP** and every language in **NEXP** is polynomial-time reducible to it. Describe one **NEXP**-complete language (you may cite a completeness result proved elsewhere rather than reproving it here).

(c) Prove that if this problem lies in **EXP**, then **NEXP** = **EXP**.

5. Oracle TM

(a) Define the unique-sat problem to be

$$\text{USAT} = \{\langle\phi\rangle \mid \phi \text{ is a Boolean formula that has a single satisfying assignment}\}.$$

Show that USAT $\in \mathrm{P}^{\mathrm{SAT}}$.

(b) Define $\mathrm{P}^{A,k}$ to be the collection of languages that are decidable by polynomial-time k-query oracle (i.e. oracle can only be used k-times) Turing machines with an oracle for A.

i. Show that $\mathrm{NP} \cup \mathrm{coNP} \subseteq \mathrm{P}^{\mathrm{SAT},1}$.

ii. Assume that $\mathrm{NP} \neq \mathrm{coNP}$. Show that $\mathrm{NP} \cup \mathrm{coNP} \subsetneq \mathrm{P}^{\mathrm{SAT},1}$.