Practice Problems Set 6

Fall 25

1. Proof of NP (attributed to CLRS Exercise 34.2-10)

   Prove that if $\mathbf{NP} \neq \mathbf{coNP}$, then $\mathbf{P} \neq \mathbf{NP}$.

Solution:

If $\mathbf{NP} \neq \mathbf{coNP}$, there exists a language L which is in $\mathbf{NP}$ but not in $\mathbf{coNP}$. As $\mathbf{P} \subseteq \mathbf{NP}$ and $\mathbf{P} = \mathbf{coP} \subseteq \mathbf{coNP}$, $L \notin \mathbf{P}$.

2. NP Closure under Concatenation

   Suppose L_1 and L_2 are two languages in the class $\mathbf{NP}$. Prove that their concatenation, written as $L_1 \circ L_2$, is also in $\mathbf{NP}$.

   (The concatenation $L_1 \circ L_2$ is the set of all strings formed by taking a string from L_1 and appending a string from L_2. Formally, $L_1 \circ L_2 = \{xy \mid x \in L_1 \text{ and } y \in L_2\}$.)

Solution:

To prove that $L_1 \circ L_2$ is in $\mathbf{NP}$, we need to show that there exists a non-deterministic Turing machine (NDTM) that can decide it in polynomial time.

Since $L_1 \in \mathbf{NP}$ and $L_2 \in \mathbf{NP}$, we know there are NDTMs, let's call them M_1 and M_2, that decide them in polynomial time. We can construct a new NDTM, M_{cat}, to decide $L_1 \circ L_2$ as follows:

Given an input string w of length n:

(a) **Guess:** Non-deterministically choose a "split point" i anywhere in the string w (where $0 \leq i \leq n$). This splits w into two substrings: a prefix $x = w[0 \dots i-1]$ and a suffix $y = w[i \dots n-1]$.

(b) **Verify:** Run M_1 on the prefix x, and run M_2 on the suffix y. If both M_1 accepts x AND M_2 accepts y, then M_{cat} accepts the original string w. Otherwise M_{cat} rejects.

Time Complexity: There are $n+1$ possible split points to guess, which is a polynomial number of choices. Running M_1 and M_2 each takes polynomial time relative to the length of their inputs, which are at most n. The total time is the sum of these polynomial-time operations, which is still polynomial.

Since we have constructed an NDTM that decides $L_1 \circ L_2$ in polynomial time, we have proven that $L_1 \circ L_2 \in \mathbf{NP}$.

3. Completeness (Adapted from CLRS Exercise 34.3-7)

We have introduced **NP**-completeness in the lecture. In general the completeness can be defined for any language/problem class, e.g., **P**, **NP**, **coNP**. Formally, a language L is C-complete for a language class C if $L \in C$ and $L' \leq_P L$ for all $L' \in C$.

Show that L is **NP**-complete if and only if $\overline{L}$ is **coNP**-complete.

Solution:

We prove both directions of the biconditional.

($\Rightarrow$) Assume L is NP-complete. By definition, this means (a) $L \in$ **NP** and (b) for all $L' \in$ **NP**, $L' \leq_P L$. We need to show that $\overline{L}$ is **coNP**-complete.

(a) **Membership**: Since $L \in$ **NP**, its complement $\overline{L}$ is in **coNP** by definition.

(b) **Hardness**: Let L'' be any language in **coNP**. This means its complement $\overline{L''}$ is in **NP**. By our assumption (b), we can reduce $\overline{L''}$ to L:

$$\overline{L''} \leq_P L$$

The same reduction can be used to reduce the complement of the LHS to the complement of the RHS:

$$\overline{(\overline{L''})} \leq_P \overline{L} \implies L'' \leq_P \overline{L}$$

Thus, any language in **coNP** reduces to $\overline{L}$.

Since $\overline{L}$ is in **coNP** and is **coNP**-hard, it is **coNP**-complete.

($\Leftarrow$) Assume $\overline{L}$ is coNP-complete. The proof is similar and symmetric to the other direction.