Introduction to Theoretical Computer Science, Fall 2024
Assignment 9 Solutions

Q1. Since $A \in \mathcal{P}$, A is decided by some deterministic Turing machine M_A with polynomial running time.

Construct a deterministic Turing machine $M_{\overline{A}}$ as follows.

$M_{\overline{A}}$ = on input w:
1. Run M_A on w
2. If M_A accepts w
3.   Reject w
4. Else (M_A rejects w)
5.   Accept w

It is easy to see that $M_{\overline{A}}$ decides $\overline{A}$ in polynomial time. Therefore, $\overline{A} \in \mathcal{P}$.

Q2. Since $A \in P$, there is some polynomial-time Turing machine M_A that decides A. We construct a Turing Machine M_{A^*} to decide A^* as follows. We use $dp[i]$ to record whether or not $y_1 \cdots y_i \in A^*$.

M_{A^*} = on input "$y = y_1 \cdots y_n$":
1. For $i = 1, ..., n$:
2.   Run M_A on $y_1 \cdots y_i$,
3.   If M_A accepts y_i:
4. $dp[i] = 1$.
5.   For $j = 1, \ldots, i-1$:
6.   If $dp[j] = 1$ and M_A accepts $y_{j+1} \cdots y_i$:
7. $dp[i] = 1$.
8. If $dp[n] == 1$:
9.   accept.
10. Else:
11.   reject.

Since M_A runs in $poly(n)$ time, M_{A^*} runs in $O(n^2 poly(n))$ time.

Q3. By the conclusion of Q1, we know that $A \in \mathcal{P}$ implies that $\overline{A} \in P$. Since $\mathcal{P} \subseteq \mathcal{NP}$, we have $A \in \mathcal{NP}$ and $\overline{A} \in \mathcal{NP}$. Therefore, $A \in \mathcal{NP} \cap$ co-$\mathcal{NP}$.

Q4. It is easy to see that DOUBLE-SAT is in NP. We give a reduction from SAT to DOUBLE-SAT as follows.

Given an instance F of SAT with m clauses, say $C_1 \wedge \cdots \wedge C_m$, we do the following: We let y be a new Boolean variables, and construct an equivalent instance of DOUBLE-SAT by adding a new clause $y \vee \bar{y}$, that is, $F' = F \wedge (y \vee \bar{y})$. If F' has at least two satisfying assignments, it is straightforward that F must be satisfiable. If F is satisfiable, then by letting $y = 0$ or $y = 1$, F' has at least two satisfying assignments. So DOUBLE-SAT is NP-Complete.

Q5. It is easy to see that DOMINATING-SET is in NP. We give a reduction from VERTEX COVER problem to DOMINATING-SET as follows.

Given an instance $(G = (V, E), k)$ of VERTEX COVER, we construct an equivalent instance $(G' = (V \cup V_E, E \cup E_V), k)$ of DOMINATING-SET, where $V_E = \{v_e : e \in E\}$, $E_v = \{(u, v_e) :$

$e = (u, v) \in E\}$. That is, for each edge $e = (u, v)$ of G, we construct a new vertex v_e and two edges (u, v_e) and $e_2 = (v, v_e)$.

Now we prove that G' has a dominating set with k nodes if and only if G has a vertex cover with k nodes. If G has a vertex cover S with k nodes, it is easy to see that S is also a dominating set for G'. Suppose that G' has a dominating set S' with k nodes. S' may contain vertices from V_E and therefore, may not be a vertex cover for G. But we can replace any node $v_e \in S' \cap V_E$ with an endpoint of e. The resulting set would be a vertex cover for G. So DOMINATING-SET is NP-Complete.