Problem 1

Answer. (a) HALT is \( \text{NP} \)-hard. We prove that, for any language \( L \in \text{NP} \), \( L \leq_p \text{HALT} \).

If \( L \in \text{NP} \), there exists a polynomial time TM \( M \) and a polynomial \( p : \mathbb{N} \to \mathbb{N} \) such that for every \( x \in \{0, 1\}^* \),

\[
x \in L \iff \exists u \in \{0, 1\}^{p(|x|)} \text{ s.t. } M(x, u) = 1.
\]

We claim that, for every \( x \), another string \( x' \) can be constructed in polynomial time such that \( x \in L \iff x' \in \text{HALT} \). The construction follows.

On input \( x \), we construct \( x' = \langle S, \alpha \rangle \) where \( \alpha = 00 \) and \( S \) is a TM that runs \( M \) on every \( u \in \{0, 1\}^{p(|x|)} \) to check whether there is a \( u \in \{0, 1\}^{p(|x|)} \) such that \( M(x, u) = 1 \). If yes, then \( S \) will halt and output 1. Otherwise, \( S \) will loop forever. \( S \) can be constructed in polynomial time (note that the reduction function only needs to output the description of \( \langle S, \alpha \rangle \), it does not need to run \( S \)).

Here is the analysis. If \( x \in L \), then there exists \( u \in \{0, 1\}^{p(|x|)} \) such that \( M(x, u) = 1 \). So \( x' \in \text{HALT} \). If \( x \notin L \), \( \forall u \in \{0, 1\}^{p(|x|)} \), \( M(x, u) = 0 \), so \( S \) will loop forever. Thus \( x' \notin \text{HALT} \). This proves that \( L \leq_p \text{HALT} \).

As a result, HALT is \( \text{NP} \)-hard.

(b) No, because HALT is not in \( \text{NP} \). Suppose HALT is in \( \text{NP} \). As \( \text{NP} \subseteq \text{EXP} \), HALT can be decided by a DTM in exponential time. However, HALT is not decidable. This is a contradiction. So HALT is not in \( \text{NP} \).

Problem 2

Answer. By definition of \( \text{NP} \), as \( L_1 \) is in \( \text{NP} \), there exists a polynomial time TM \( M_1 \) and a polynomial \( p_1 : \mathbb{N} \to \mathbb{N} \) such that for every \( x \in \{0, 1\}^* \),

\[
x \in L_1 \iff \exists u \in \{0, 1\}^{p_1(|x|)} \text{ s.t. } M_1(x, u) = 1.
\]

Also, as \( L_2 \) is in \( \text{NP} \), there exists a polynomial time TM \( M_2 \) and a polynomial \( p_2 : \mathbb{N} \to \mathbb{N} \) such that for every \( x \in \{0, 1\}^* \),

\[
x \in L_2 \iff \exists u \in \{0, 1\}^{p_2(|x|)} \text{ s.t. } M_2(x, u) = 1.
\]

(a) The language \( L_1 \cup L_2 \) is in \( \text{NP} \). This can be shown by constructing another verifier \( R \) which is a TM that works as follows. On input \( x \) and \( u = u_1 \circ u_2 \in \)
\{0, 1\}^{p_1(|x|)+p_2(|x|)} (u_1 \text{ has length } p_1(|x|)), R \text{ runs } M_1(x, u_1) \text{ and } M_2(x, u_2). \text{ If at least one of them outputs 1, } R \text{ outputs 1. Otherwise } R \text{ outputs 0.}

We notice that \( R \) runs in polynomial time as both \( M_1 \) and \( M_2 \) run in polynomial time.

If \( x \in L_1 \), let \( u'_1 \) be such that \( M_1(x, u'_1) = 1 \). Then \( R \) will output 1 on input \( \langle x, u'_1 \circ 1^{p_2(|x|)} \rangle \). If \( x \in L_2 \), let \( u'_2 \) be such that \( M_2(x, u'_2) = 1 \). Then \( R \) will output 1 on input \( \langle x, 1^{p_1(|x|)} \circ u'_2 \rangle \). Thus if \( x \in L_1 \cup L_2 \), there is a \( u \in \{0, 1\}^{p_1(|x|)+p_2(|x|)} \) such that \( R(x, u) = 1 \). On the other hand, if \( x \notin L_1 \cup L_2 \), we know that \( \forall u_1 \in \{0, 1\}^{p_1(|x|)} \), \( M_1(x, u_1) = 0 \) and \( \forall u_2 \in \{0, 1\}^{p_2(|x|)} \), \( M_2(x, u_2) = 0 \). So \( \forall u \in \{0, 1\}^{p_1(|x|)+p_2(|x|)}, R(x, u) = 0 \). This proves that there exists a polynomial time TM \( R \) s.t.

\[ x \in L_1 \cup L_2 \iff \exists u \in \{0, 1\}^{p_1(|x|)+p_2(|x|)} \text{ s.t. } R(x, u) = 1. \]

So \( L_1 \cup L_2 \) is in \( \text{NP} \).

(b) The language \( L_1 \cap L_2 \) is also in \( \text{NP} \). It can be shown by constructing another verifier \( R \) which is a TM that works as follows. On input \( x \) and \( u = u_1 \circ u_2 \in \{p_1(|x|) + p_2(|x|)\} \) (\( u_1 \) has length \( p_1(|x|) \)), \( R \) runs \( M_1(x, u_1) \) and \( M_2(x, u_2) \). If both of them output 1, \( R \) outputs 1. Otherwise \( R \) outputs 0.

As a result, first we know that \( R \) runs in polynomial time as both \( M_1 \) and \( M_2 \) runs in polynomial time. Second, if \( x \in L_1 \cap L_2 \), there exists \( u_1 \in \{0, 1\}^{p_1(|x|)} \) such that \( M_1(x, u_1) = 1 \) and there exists \( u_2 \in \{0, 1\}^{p_2(|x|)} \) such that \( M_2(x, u_2) = 1 \). So there is a \( u = u_1 \circ u_2 \in \{0, 1\}^{p_1(|x|)+p_2(|x|)} \) such that \( M_1(x, u_1) = 1 \) and \( M_2(x, u_2) = 1 \). Thus \( R(x, u) = 1 \). On the other hand, if \( x \notin L_1 \cap L_2 \), we know that either \( \forall u_1 \in \{0, 1\}^{p_1(|x|)} \), \( M_1(x, u_1) = 0 \) or \( \forall u_2 \in \{0, 1\}^{p_2(|x|)} \), \( M_2(x, u_2) = 0 \). So \( \forall u \in \{0, 1\}^{p_1(|x|)+p_2(|x|)}, R(x, u) = 0 \). Thus there exists a polynomial time TM \( R \) s.t.

\[ x \in L_1 \cap L_2 \iff \exists u \in \{0, 1\}^{p_1(|x|)+p_2(|x|)} \text{ s.t. } R(x, u) = 1. \]

So \( L_1 \cap L_2 \) is in \( \text{NP} \).

\[ \square \]

**Problem 3**

**Answer.** Let \( L \) be an arbitrary language in \( \text{P} \), where \( L \neq \emptyset, L \neq \Sigma^* \).

For every \( L' \in \text{NP} \), we show \( L' \leq_p L \).

As \( L \neq \emptyset \) and \( L \neq \Sigma^* \), there exists a string \( a \) s.t. \( a \in L \) and a string \( b \) s.t. \( b \notin L \).
Since \( P = NP \), \( L' \in P \), assume the decider for \( L' \) is \( M' \) which runs in polynomial time. We define a function \( f \) as follows. For every \( x \in \Sigma^* \), we conduct \( M'(x) \), if \( M'(x) = 1 \) then \( f(x) = a \), otherwise \( f(x) = b \).

As a result, \( \forall x \in \Sigma^*, x \in L' \iff M'(x) = 1 \iff f(x) \in L \)

This shows \( L' \leq_p L \). So \( L \) is \( NP \)-hard. Thus it is \( NP \)-complete.

\[ \square \]

**Problem 4**

**Answer.** We reduce VERTEX COVER to DOMINATING-SET.

For every string \( x \), we will prove that another string \( x' \) can be constructed in polynomial time such that \( x \in VERTEX \text{ COVER} \iff x' \in DOMINATING-SET \).

Let \( x = \langle G, k \rangle \) where \( G = (V, E) \). we can construct \( x' \) in the following way. First construct another graph \( G' = (V', E') \) using \( G \). Without lost of generality, assume that \( G \) does not have vertices which has degree 0. Let \( V' = V \cup V_E \) where \( V_E \) is a set of new vertices such that each vertex \( v_e \in V_E \) if and only if \( e \in E \). On the other hand, let \( E' = E \cup E' \). Here \( E' \) includes the edges \((v_e, u)\) and \((v_e, w)\) for every \( v_e \in V_E \) where \( e = (u, w) \). Let \( x' = \langle G', k \rangle \).

Next we will show that \( x \in VERTEX \text{ COVER} \iff x' \in DOMINATING-SET \).

If \( G \) has a vertex cover \( C \) of size \( k \), then \( G' \) has a dominating set of size \( k \). To see this, we claim the dominating set of \( G' \) is exactly \( C \). Let’s consider every vertex in \( G' \). For every \( v \in V \), as we have assumed that \( v \) has degree at least 1, it is connected with an edge of \( E \). As \( C \) is a vertex cover, this edge is connected with a vertex of \( C \). So either \( v \) is in \( C \) or \( v \) is adjacent to a vertex in \( C \). For every \( v_e \in V_E \), as the edge \( e \in E \) is connected with a vertex in \( C \), according to our definition of \( v_e \), we know that \( v_e \) is also adjacent to that vertex. As a result, \( C \) is a dominating set of \( G' \).

On the other hand, if \( G' \) has a dominating set which is \( D \) of size \( k \), we claim that a vertex cover \( C \) for \( G \) of size \( k \) can be found. Next we describe an algorithm to find \( C \).

First let \( C \) be the empty set. Then for each vertex \( v \) in \( D \), if \( v \in V \), put \( v \) into \( C \). If \( v \in V_E \), assuming it corresponds to the edge \( e = (u, w) \), put either \( u \) or \( w \) into \( C \).

As a result, \( |C| \leq k \). Consider every edge \( e = (u, w) \in E \). We know \( v_e \) is adjacent to a vertex \( v' \) of \( D \). This vertex can only be \( v_e, u \) or \( w \). In any case, one of \( u \) and \( w \) is in \( C \) which covers the edge \( e \). As a result, \( C \) covers all the edges. This means it is a vertex cover of size at most \( k \) for \( G \).

So we have that DOMINATING-SET is \( NP \)-hard.
Next we show that it is in \( \text{NP} \). For \( x = (G = (V, E), k) \) the certificate is a set \( D \) of vertices such that \( D \subseteq V \). Thus \( |D| \leq |V| \). The verifier TM \( M \) checks whether \( D \) is a dominating-set of size \( k \) for \( G \). This checking can be accomplished in polynomial time as we can check the vertices one by one to make sure that each vertex is either in \( D \) or adjacent to a vertex in \( D \). According to the definition of \( \text{NP} \), \( \text{DOMINATING-SET} \) is in \( \text{NP} \).

As a result, \( \text{DOMINATING-SET} \) is \( \text{NP} \)-complete. \( \square \)

**Problem 5**

*Answer.* Part (a). If \( \text{P} = \text{NP} \), then \( \forall L \in \text{NP} \), we have \( L \in \text{P} \). Thus \( \overline{L} \in \text{P} \subseteq \text{NP} \) which implies that \( L \in \text{coNP} \). Therefore \( \text{NP} \subseteq \text{coNP} \).

Similarly, \( \forall L \in \text{coNP} \), we have \( \overline{L} \in \text{NP} \) and thus \( \overline{L} \in \text{P} \). Hence \( L \in \text{P} \subseteq \text{NP} \).

Therefore \( \text{coNP} \subseteq \text{NP} \).

This implies that \( \text{NP} = \text{coNP} \).

Part (b). Suppose \( \text{coNP} \subseteq \text{NP} \), we will show that \( \text{NP} \subseteq \text{coNP} \) and this implies that \( \text{NP} = \text{coNP} \).

To see this, consider \( \forall L \in \text{NP} \), then by definition \( \overline{L} \in \text{coNP} \subseteq \text{NP} \). Thus again by definition we have \( L \in \text{coNP} \). Therefore we have \( \text{NP} \subseteq \text{coNP} \). \( \square \)