1 Problem Set 6

Problem 1 Prove or disprove the following statements:

1. If \( A, B \in \mathcal{NP} \), then \( A \cap B \in \mathcal{NP} \) and \( A \cup B \in \mathcal{NP} \).
2. If \( A \) and \( B \) are two \( \mathcal{NP} \)-complete languages, then \( A \cap B \) is \( \mathcal{NP} \)-complete.
3. If \( A \) and \( B \) are two \( \mathcal{NP} \)-complete languages, then \( A \cup B \) is \( \mathcal{NP} \)-complete.

Solution.
1. Answer: TRUE. \( A \cap B \in \mathcal{NP} \) and \( A \cup B \in \mathcal{NP} \).

If \( A \) and \( B \) are in \( \mathcal{NP} \), then, by the definition, there exist polynomial time algorithms \( M_A \) and \( M_B \) such
\[
x \in A \text{ if and only if } \exists w_A \ M_A(x, w_A) = 1;
\]
\[
x \in B \text{ if and only if } \exists w_B \ M_B(x, w_B) = 1.
\]

Let us construct an algorithm that decides whether \( "x \in A \cap B" \), given a witness \( w \).

| Input: x and a witness w. The algorithms expects w to be a pair \((w_A, w_B)\), where \(w_A\) is a witness for \(x \in A\); \(w_B\) is a witness for \(x \in B\). |
| Output: 1 – accept; or 0 – reject |
| 1. Let \( w_A \) and \( w_B \) be the first and the second components of the pair \( w \); that is, \( (w_A, w_B) = w \) (if \( w \) is not a pair of words, then Reject.) |
| 2. if \( M_A(x, w_A) = 1 \) and \( M_B(x, w_B) = 1 \), then Accept; else Reject |

If \( x \in A \cap B \), then the algorithm accepts \( x \) with the witness \( w = (w_A, w_B) \), since \( M_A(x, w_A) = 1 \) and \( M_B(x, w_A) = 1 \).
If \( x \notin A \cap B \), then \( x \notin A \) or \( x \notin B \). Assume without loss of generality that \( x \notin A \). Hence for every \( w_A, M_A(x, w_A) = 0 \). Therefore, the if-condition is false and the algorithm rejects \( x \). Similarly, we can prove that \( A \cup B \in \mathcal{NP} \).
2. **Answer: FALSE.** There exist $\mathcal{NP}$-complete languages $A$ and $B$ such that $A \cap B$ is not $\mathcal{NP}$-complete. Example:

\[
A = \{1\#x : x \in SAT \} ; \\
B = \{0\#x : x \in SAT \} .
\]

**Remark:** $\#$ denotes concatenation e.g. $0\#10111 = 010111$.

The languages $A$ and $B$ are $\mathcal{NP}$-complete (why?). On the other hand, $A \cap B$ is the empty set; and thus it is not $\mathcal{NP}$-complete.

3. **Answer: FALSE.** There exist $\mathcal{NP}$-complete languages $A$ and $B$ such that $A \cup B$ is not $\mathcal{NP}$-complete. Example:

\[
A = \{1\#x : x \in SAT \} \cup \{0\#x : x \in \{0,1\}^* \} ; \\
B = \{0\#x : x \in SAT \} \cup \{1\#x : x \in \{0,1\}^* \} .
\]

The languages $A$ and $B$ are $\mathcal{NP}$-complete (prove it). On the other hand, $A \cup B$ contains all binary strings (i.e. $A \cup B = \{0,1\}^*$); and thus it is not $\mathcal{NP}$-complete.

**Definition 1 (Circuit Minimization Problem).** Given a circuit $C$ determine if there exists a smaller circuit that computes the same function as $C$.

**Problem 2** Prove that if the SAT problem is in $\mathcal{P}$, then the Circuit Minimization Problem is solvable in polynomial time.

**Solution.** We will show that

1. The Circuit Minimization Problem is in $\Pi_2 = \text{co-}\Sigma_2$;

2. If $\mathcal{P} = \mathcal{NP}$, then $\Sigma_2 = \mathcal{P}$.

Therefore, if SAT $\in \mathcal{P}$, then $\mathcal{P} = \mathcal{NP}$ (since SAT is $\mathcal{NP}$-complete) and the Circuit Minimization Problem is in $\Pi_2 = \mathcal{P}$.

Recall, that a language $L$ is in $\Sigma_2$ (by the definition) if there exists a polynomial algorithm $A$ such that

\[
x \in L \text{ if and only if } \exists w_1 \forall w_2 \ A(x, w_1, w_2) = 1 . \tag{1}
\]

Here the witnesses $w_1$ and $w_2$ are of polynomial size.

I. A circuit $C$ is not minimal, if there exists a smaller circuit $C'$ that is equivalent to $C$. In other words, $C$ is not minimal if there exists a circuit $C'$ such that for every input $x$:

- $C'(x) = C(x)$ (that is, $C'$ is equivalent to $C$);
- $\text{size}(C') < \text{size}(C)$.  


From this characterization, we get that the complement to the Circuit Minimization Problem is in $\Sigma_2$. Thus the problem itself is in co-$\Sigma_2 = \Pi_2$.

II. We now need to show that if $P = NP$, then $\Sigma_2 = P$. Consider an arbitrary language $L$ in $\Sigma_2$ defined as follows:

$$x \in L \text{ if and only if } \exists w_1 \forall w_2 A(x, w_1, w_2) = 1.$$  \hspace{1cm} (2)

Define a new language $L'$:

$$L' = \{(x, w_1) : \forall w_2 A(x, w_1, w_2) = 1\}.$$

Now rewrite (2) in a slightly different way:

$$x \in L \text{ if and only if } \exists w_1 \text{ s.t. } (x, w_1) \in L'.$$  \hspace{1cm} (3)

Observe, that $L'$ is in co-$NP$. Thus there exists a polynomial time algorithm $B$ deciding the language $L'$ (we assume that $P = NP$). Hence (3) is equivalent to

$$x \in L \text{ if and only if } \exists w_1 \text{ s.t. } B(x, w_1).$$

But this is an $NP$-statement, thus the problem can be solved in polynomial time (again, we assume that $P = NP$). \hfill \Box