1. Show that \( \mathsf{NP} \) is closed under union and intersection. In other words,

(a) If \( L_1 \in \mathsf{NP} \), \( L_2 \in \mathsf{NP} \), then \( L_1 \cup L_2 \in \mathsf{NP} \). \[2\]
(b) If \( L_1 \in \mathsf{NP} \), \( L_2 \in \mathsf{NP} \), then \( L_1 \cap L_2 \in \mathsf{NP} \). \[2\]

\textbf{Answer:} Given the assumption, there are TMs \( M_1 \) and \( M_2 \) such that

\[ x \in L_i \iff \exists y, \; M_i(x, y) = 1, \]

where \( y \) has a length bounded by a polynomial. We then give two verifying algorithms for \( L_1 \cup L_2 \) and \( L_1 \cap L_2 \).

(a) If \( x \in L_1 \cup L_2 \), then there is a certificate \( y \) which works for either \( M_1 \) or \( M_2 \) (or both, but that doesn’t matter). The TM \( M_{\cup} \) for \( L_1 \cup L_2 \) would treat the certificate \( y \) as a concatenation of two certificates \( y_1 \) and \( y_2 \). Then \( M_{\cup} \) simulates \( M_1 \) on \( (x, y_1) \) and \( M_2 \) on \( (x, y_2) \), and accepts if one of them accepts.

(b) If \( x \in L_1 \cap L_2 \), then there are certificates \( (y_i)_{i=1,2} \) so that \( y_i \) works for \( M_i \). Similarly to the case above, the TM \( M_{\cap} \) for \( L_1 \cap L_2 \) treat the certificate \( y \) as a concatenation of two certificates \( y_1 \) and \( y_2 \). Then \( M_{\cap} \) simulates \( M_1 \) on \( (x, y_1) \) and \( M_2 \) on \( (x, y_2) \), and accepts if both of them accept.

2. Show that if \( \text{Sat} \leq_p \text{Taut} \), then \( \mathsf{PH} = \mathsf{NP} \). \[4\]

\textbf{Answer:} We know that \( \text{Taut} \in \mathsf{coNP} \), and \( \text{Sat} \) is \( \mathsf{NP} \)-complete. Hence, \( \text{Sat} \leq_p \text{Taut} \) implies that \( \mathsf{NP} \subseteq \mathsf{coNP} \). Notice that this implies \( \mathsf{NP} = \mathsf{coNP} \) (by the same reason explained in the proof of Theorem 2 from Lecture 9). By Theorem 2 from Lecture 9, it implies that \( \mathsf{PH} = \mathsf{NP} = \mathsf{coNP} \).

3. Recall that TQBF is the problem of determining the validity of a totally quantified Boolean formula. Show that \( \text{TQBF} \notin \mathsf{L} \). \[4\]

\textbf{Answer:} We have seen that TQBF is \( \mathsf{PSpace} \)-complete. Thus, if \( \text{TQBF} \in \mathsf{L} \), it implies that \( \mathsf{PSpace} = \mathsf{L} \). However, this contradicts to the space hierarchy theorem. Therefore, \( \text{TQBF} \notin \mathsf{L} \).

4. Let \( \oplus(\cdot) \) be the parity function. Namely, \( \oplus(x) = \sum_{i=1}^{n} x_i \mod 2 \) for a vector \( x = \{x_1, \ldots, x_n\} \) and for all \( i \in [n], x_i \in \{0, 1\} \).

Show that to express \( \oplus(\cdot) \) as a CNF or DNF formula, at least \( 2^{n-1} \) clauses are required. \[4\]

\textbf{Hint:} first show that every clause must have size \( n \).
What about expressing $\oplus(\cdot)$ as an arbitrary (not necessarily CNF or DNF) formula? For simplicity, you may only give the construction for $n$ being a power of 2. Try to minimize the size of your construction.

**Hint:** use a recursive construction.

**Answer:** Let us break it into two parts.

(a) First, assume that $\oplus(\cdot)$ is written as a CNF formula $\varphi$. If one of the clauses of $\varphi$ has length $\leq n - 1$, then there is at least one variable, say $x_n$, left out. Thus, there are assignments $\sigma_0$ and $\sigma_1$, that differ only on $x_n$, and both do not satisfy $\varphi$. However, we know that $\sigma_0$ and $\sigma_1$ have different parity, and thus one of them must make $\oplus 1$. It implies that all clauses of $\varphi$ must have length $n$. Each clause of length $n$ forbids one assignment out of $2^n$. The parity function $\oplus(\cdot)$ has $2^{n-1}$ assignments mapping to 0. Hence, we need $2^n - 1$ clauses to forbid all of them.

The argument for a CNF formula is similar.

(b) We use a recursive construction, and first assume that $n = 2^k$ for some $k$. The basic idea is that, to compute the parity of two inputs, we can use the following formula

$$\varphi_2(x, y) = (x \land \neg y) \lor (\neg x \land y).$$

Next we just need to “scale it up”.

Suppose that $\varphi_i$ computes the parity function $\oplus(\cdot)$ with input of length $2^i$, for all $i \leq k - 1$. Define:

$$\varphi_k(x_1, \ldots, x_{2^k}) := (\varphi_{k-1}(x_1, \ldots, x_{2^{k-1}}) \land \neg \varphi_{k-1}(x_{2^{k-1}+1}, \ldots, x_{2^k}))$$

$$\lor (\neg \varphi_{k-1}(x_1, \ldots, x_{2^{k-1}}) \land \varphi_{k-1}(x_{2^{k-1}+1}, \ldots, x_{2^k})).$$

It is easy to see that this computes $\oplus(\cdot)$.

Let $C_k$ be the size of $\varphi_k$. Then it is easy to have the following recurrence:

$$C_k = 4C_{k-1},$$

with $C_0 = 1$. Solving it yields that $C_k = 4^k = n^2$ (recall that $n = 2^k$). Hence our construction has size $O(n^2)$.

If $n$ is not a power of 2, then we find the closest power of 2 larger than $n$, and then project down to $n$. This gives a formula of size at most $(2n)^2 = 4n^2$. So the size is still $O(n^2)$.

5. (a) Show that if a function $f : \{0, 1\}^n \to \{0, 1\}$ has a support of size $k$, then it can be decided by a circuit of size $O(nk)$. [2]

(The support of $f$ is the set of inputs that map to 1, namely $\text{Supp}(f) = \{x \mid f(x) = 1\}$.)

(b) Show that, for sufficiently large $n$, there is a language that can be decided by circuits of size $n^3$ but not $n^2$. [4]

**Hint:** use a counting argument.

**Answer:**
(a) We can simply consider a DNF formula for \( f \). Each clause of the DNF permits one assignment, and we just need \( k \) clauses to express \( f \), since its support has size \( k \).

Each clause of the DNF contains at most \( n \) literals, and thus we need to use \( O(n) \) gates per clause, resulting in \( O(nk) \) gates in total.

(b) Due to the first question, we can realize any function with support of size \( n^{3/2} \) using at most \( O(n^{5/2}) \leq n^3 \) gates, for sufficiently large \( n \). For a language \( L \), we realize it this way by considering its equivalent function \( f_L \). The total number of these languages is

\[
\left( \frac{2^n}{n^{3/2}} \right)^{n^{3/2}} \geq 2^{n^{9/4}},
\]

for sufficiently large \( n \).

On the other hand, a circuit of size \( \leq n^2 \) can be described using only \( O(n^2 \log n) \) bits. (Consider the adjacency matrix of the digraph, which is sparse.) Thus, the total number of languages computed by a circuit of size at most \( n^2 \) is at most \( 2^{O(n^2 \log n)} \).

For sufficiently large \( n \),

\[
2^{n^{9/4}} \geq 2^{O(n^2 \log n)},
\]

which implies that there must be a language that can be computed by a circuit of size at most \( n^3 \) but not \( n^2 \).