Computational Complexity

Take-home exam

Hand in via Canvas before March 27, 2026, at 23:59

`https://canvas.uva.nl/courses/56615/assignments/662594`

Definition 1. Let φ be a propositional formula and let $X \subseteq \text{Var}(\varphi)$ be a subset of its variables. A *circuit projection* of φ onto X is a Boolean circuit C with inputs X such that for every truth assignment $\alpha : X \to \{0,1\}$:

$$C(\alpha) = 1 \quad \Longleftrightarrow \quad \exists\, \beta : \text{Var}(\varphi) \setminus X \to \{0,1\} \text{ such that } (\alpha, \beta) \models \varphi.$$

In other words, C accepts exactly those partial assignments to X that can be extended to a satisfying assignment of φ.

For example, consider the Horn formula $\varphi = (\neg x \vee y) \wedge (\neg y \vee z)$ over variables $\{x, y, z\}$, and let $X = \{x, z\}$. The circuit projection of φ onto X is a circuit C over inputs x, z such that $C(x, z) = 1$ if and only if there exists a value for y making φ true. One can verify that C computes the function $\neg x \vee z$: when $x = 1$, we need $y = 1$ (from the first clause) and then $z = 1$ (from the second clause); when $x = 0$, both clauses are satisfied for any y and z.

We say that a function f *implements circuit projection* if for every propositional formula φ and every $X \subseteq \text{Var}(\varphi)$, the value $f(\varphi, X)$ is a circuit projection of φ onto X.

Question 1 (*4pts; a: 1pt, b: 1pt, c: 2pts*)**.**

(a) Prove that there exists a polynomial-time function implementing circuit projection for the restricted case where φ is a Horn formula. That is: given a Horn formula φ and a set $X \subseteq \text{Var}(\varphi)$, one can compute a circuit projection of φ onto X in polynomial time.

- *Hint:* consider using resolution.
- A Horn clause is a disjunction of literals with at most one positive literal. A Horn formula is a conjunction of Horn clauses.

(b) Prove that if there is a function f that implements circuit projection for arbitrary propositional formulas and that can be computed in polynomial time, then $\mathsf{P} = \mathsf{NP}$.

(c) Prove that if there is a function f that implements circuit projection for arbitrary propositional formulas and that is of polynomial-size, then the Polynomial Hierarchy collapses. A function f is of *polynomial-size* if there exists some polynomial p such that $|f(\varphi, X)| \leq p(|\varphi|)$—i.e., the size of the output circuit is polynomially bounded in the size of the input formula, but there are no restrictions on the time needed to compute f (or whether it is computable at all).

- *Hint:* use the fact that $\mathsf{NP} \subseteq \mathsf{P/poly}$ implies that $\mathsf{PH} = \Sigma_2^{\mathrm{p}}$.
- *Hint:* for different values of $\ell \in \mathbb{N}$, consider the formula:

$$\varphi_\ell = \bigwedge_{1 \leq i \leq (2\ell)^3} (y_i \to c_i),$$

where $c_1, \ldots, c_{(2\ell)^3}$ is an enumeration of all possible clauses of size 3 over the variables $\{x_1, \ldots, x_\ell\}$, and the formula φ_ℓ is over the variables $\{y_1, \ldots, y_{(2\ell)^3}\} \cup \{x_1, \ldots, x_\ell\}$.

Definition 2. An NP-*oracle verifier* for a language $L \subseteq \{0,1\}^*$ is a polynomial-time deterministic Turing machine V that has access to an oracle $O \in \mathsf{NP}$, together with a polynomial p, such that for every $x \in \{0,1\}^*$:

$$x \in L \quad \Longleftrightarrow \quad \exists\, w \in \{0,1\}^{p(|x|)}, \quad V^O(x, w) = 1.$$

The string w is called the *witness* for x.

Question 2 (*2pts; a: 1/2pt, b: 1 1/2pts*).

In this exercise, you will show that the NP-oracle verifier characterization gives us exactly Σ_2^{p}.

(a) Let L be a language that has an NP-oracle verifier. Prove that $L \in \Sigma_2^{\mathrm{p}}$.

(b) Take an arbitrary $L \in \Sigma_2^{\mathrm{p}}$. Prove that L has an NP-oracle verifier.

- *Hint:* use the Σ_2^{p}-complete problem Σ_2SAT: given a propositional formula $\varphi(x_1, \dots, x_n, y_1, \dots, y_m)$, decide whether $\exists x_1, \dots, x_n \, \forall y_1, \dots, y_m \, . \, \varphi(x_1, \dots, x_n, y_1, \dots, y_m)$ is true. Show that Σ_2SAT has an oracle verifier, and argue how this can be used to prove the claim.

Note: for this entire question, you are not allowed to use (without proof) the fact that $\Sigma_2^{\mathrm{p}} = \mathsf{NP}^{\mathsf{NP}}$, because this question in fact asks you to prove this statement. You *may* use the fact that Σ_2^{p} coincides with the class of all decision problems solvable by a nondeterministic polynomial-time Turing machine that has access to an oracle in NP. However, the approach pointed at in the hint (using Σ_2^{p}-completeness of Σ_2SAT) seems to be an easier way towards a solution.

Definition 3. Let $G = (V, E)$ be an undirected graph and let $S \subseteq V$. A vertex $u \in V \setminus S$ is a *private neighbour* of $v \in S$ (with respect to S) if u is adjacent to v but not adjacent to any other vertex in S; that is, $u \in N(v)$ and $u \notin N(w)$ for all $w \in S \setminus \{v\}$.

We call S a *private-neighbour independent set* if S is an independent set in G (no two vertices in S are adjacent) and every vertex $v \in S$ has at least one private neighbour with respect to S.

Consider the following decision problem PNIS (Private-Neighbour Independent Set):

Input: An undirected graph $G = (V, E)$ and a positive integer k, given in unary.

Question: Does G have a private-neighbour independent set of size $|S| \geq k$?

Question 3 (*4pts; a: 1/2pt, b: 2 1/2pts, c: 1pt*).

(a) Prove that there exists an algorithm that solves PNIS in time $n^{O(k)}$, where $n = |V|$.

(b) Prove that PNIS is NP-hard.

(c) Prove that there is no algorithm that solves PNIS in time $n^{o(k)} \cdot m^{O(1)}$, where $n = |V|$ and $m = |E|$, assuming the ETH.

- *Hint:* you may use without proof that, assuming the ETH, there is no algorithm that solves Clique in time $n^{o(k)}$, where n is the number of vertices and k is the clique size.