Complexité avancée - TD 5

Benjamin Bordais

October 21, 2020

Exercise 1 Family of circuits

Definition 1 A boolean circuit with \( n \) inputs is an acyclic graph where the \( n \) inputs \( x_1, \ldots, x_n \) are part of the vertices. The internal vertices are labeled with \( \land \), \( \lor \) (with 2 incoming edges) or \( \neg \) (with 1 incoming edge), with an additional distinguished vertex \( o \) that is the output (with no exiting edge). The size \( |C| \) of a circuit \( C \) is its number of vertices (excluding the input ones). For a word \( x \in \{0,1\}^* \), the notation \( C(x) \) refers to the output of the circuit \( C \) if the input vertices of \( C \) are valued with the bits of \( x \).

Definition 2 For a function \( t : \mathbb{N} \rightarrow \mathbb{N} \), a family of circuit of size \( t(n) \) is a sequence \( (C_n)_{n \in \mathbb{N}} \) such that: \( C_n \) is an \( n \)-input circuit and \( |C_n| \leq t(n) \).

Definition 3 A language \( L \subseteq \{0,1\}^* \) is decided by a family of circuit \( (C_n)_{n \in \mathbb{N}} \) if for all \( n \in \mathbb{N} \), for all \( w \in \{0,1\}^n \), we have: \( C_n(w) = 1 \iff w \in L \).

Definition 4 For a function \( t : \mathbb{N} \rightarrow \mathbb{N} \), we define \( \text{SIZE}(t) := \{L \subseteq \{0,1\}^* \mid L \text{ is decided by a family of circuits of size } O(t(n))\} \).

Definition 5 \( \text{P/poly} := \cup_{k \in \mathbb{N}} \text{SIZE}(n^k) \)

1. Show that any language \( L \subseteq \{0,1\}^* \) is in size \( \text{SIZE}(n \cdot 2^n) \).
2. Show that for all function \( t(n) = 2^{o(n)} \), there exists \( L \notin \text{SIZE}(t(n)) \).
3. Show that every unary language is in \( \text{P/poly} \).
4. Exhibit a undecidable language that is in \( \text{P/poly} \).
5. Show that \( \text{P/poly} \) is not countable.

Exercise 2 Some alternation

1. Exhibit a polynomial time alternating algorithm that solves QBF.
2. Let \( \text{ONE} – \text{VAL} \) be the problem of deciding whether a boolean formula is satisfied by exactly one valuation. Show that \( \text{ONE} – \text{VAL} \in \Sigma_2^p \).
3. A boolean formula is minimal if it has no equivalent shorter formula – where the length of the formula is the number of symbols it contains. Let \( \text{MIN} – \text{FORMULA} \) be the problem of deciding whether a boolean formula is minimal. Show that \( \text{MIN} – \text{FORMULA} \in \Pi_2^p \).
Exercise 3 Collapse of \( \text{PH} \)

1. Prove that if \( \Sigma^P_k = \Sigma^P_{k+1} \) for some \( k \geq 0 \) then \( \text{PH} = \Sigma^P_k \). (Remark that this is implied by \( P = \text{NP} \)).

2. Show that if \( \Sigma^P_k = \Pi^P_k \) for some \( k \) then \( \text{PH} = \Sigma^P_k \) (i.e. \( \text{PH} \) collapses).

3. Show that if \( \text{PH} = \text{PSPACE} \) then \( \text{PH} \) collapses.

4. Do you think there is a polynomial time procedure to convert any QBF formula into a QBF formula with at most 10 variables?

Exercise 4 Oracles

Consider a language \( A \). A Turing machine with oracle \( A \) is a Turing machine with a special additional read/write tape, called the oracle tape, and three special states: \( q_{\text{query}}, q_{\text{yes}}, q_{\text{no}} \). Whenever the machine enters the state \( q_{\text{query}} \), with some word \( w \) written on the oracle tape, it moves \textbf{in one step} to the state \( q_{\text{yes}} \) or \( q_{\text{no}} \) depending on whether \( w \in A \).

We denote by \( \text{P}^A \) (resp. \( \text{NP}^A \)) the class of languages decided in by a deterministic (resp. non-deterministic) Turing machine running in polynomial time with oracle \( A \). Given a complexity class \( \mathcal{C} \), we define \( \text{P}^\mathcal{C} = \bigcup_{A \in \mathcal{C}} \text{P}^A \) (and similarly for \( \text{NP} \)).

1. Prove that for any \( \mathcal{C} \)-complete language \( A \) (for logspace reductions), \( \text{P}^\mathcal{C} = \text{P}^A \) and \( \text{NP}^\mathcal{C} = \text{NP}^A \).

2. Show that for any language \( A \), \( \text{P}^A = \overline{\text{P}}^A \) and \( \text{NP}^A = \overline{\text{NP}}^A \).

3. Prove that if \( \text{NP} = \text{P}^{\text{SAT}} \) then \( \text{NP} = \overline{\text{NP}} \).

4. Show that there exists a language \( A \) such that \( \text{P}^A = \overline{\text{NP}}^A \).

5. We define inductively the classes \( \text{NP}_0 = \text{P} \) and \( \text{NP}_{k+1} = \text{NP}_{\text{NP}_k} \). Show that \( \text{NP}_k = \Sigma^P_k \) for all \( k \geq 0 \).

\[^{1}\text{In fact, there also exists a language} \overline{B} \text{such that} \overline{\text{P}}^B \neq \overline{\text{NP}}^B, \text{which does not prove that} \text{P} \neq \text{NP}.\]