Exercise Sheet 8: Alternation

David Carral

January 8, 2020

Exercise 1

Describe a polynomial-time ATM solving **EXACT INDEPENDENT SET**.

$$\textbf{ExactIS} = \{(G,k) \mid \; |S| = k \text{ for some independent set } S \text{ in } G \text{ and} \\ |S'| \leq k \text{ for every independent set } S' \text{ in } G\}$$

Find a level of the polynomial hierarchy where this problem is contained in.

Solution.

boolean *Exact-Independent-Set*(graph $G = (V, E)$, integer k)
  if ($|V| < k$) return *FALSE*;
  if ($|V| = k$ and $|E| = \emptyset$) return *TRUE*;
  if ($|V| = k$ and $|E| \neq \emptyset$) return *FALSE*;
  existentially choose $W \subseteq V$ of size k;
  universally choose $U \subseteq V$ of size $k+1$;
  if (W is not independent in G) return *FALSE*;
  if (U is independent in G) return *FALSE*;
  return *TRUE*;

Is the algorithm correct?
Does it run in polynomial time?
Is **EXACT INDEPENDENT SET** in $\Sigma_2\mathrm{P} = \Sigma_2^{\mathrm{P}} = \mathrm{NP}^{\mathrm{NP}}$?

Exercise 2

Consider the Japanese game *go-moku* that is played by two players X and O on a 19x19 board. Players alternately place markers on the board, and the first one to have five of its markers in a row, column, or diagonal wins. Consider the generalised version of go-moku on an $n \times n$ board. Say that a *position* of go-moku is a placement of markers on such a board as it could occur during the game.

Let

$$\mathbf{GM} = \{\langle \mathbf{B} \rangle \mid B \text{ is a position of go-moku where } X \text{ has a winning strategy}\}.$$

Describe a polynomial-time ATM that solves **GM**.

Exercise 2

Let **GM** $= \{\langle \mathbf{B} \rangle \mid B$ is a position of go-moku where X has a winning strategy$\}$.
Describe a polynomial-time ATM that solves **GM**.

Solution.

boolean *xHasWinningStr*(position B) {
    if(*isXWinPos*(B)) return *TRUE*;
    if(*isOWinPos*(B) or *FullBoard*(B)) return *FALSE*;
    if(*isXTurn*(B))
        existentially choose $B' \in next(B)$
            return *xHasWinningStr*(B');
    if(*isOTurn*(B))
        universally choose $B' \in next(B)$
            return *xHasWinningStr*(B'); }

Is the algorithm correct?
Does it run in polynomial time?
Is **GM** in any level of the polynomial hierarchy?

Exercise 3

Show $\mathrm{NP}^{\textbf{SAT}} \subseteq \Sigma_2\mathrm{P}$.

Solution.

1. Let **L** be a language in $\mathrm{NP}^{\textbf{SAT}}$ and let $\mathcal{M}^{\textbf{SAT}}$ be a poly-time NTM that decides **L**.
2. Let $\mathcal{N}$ be the ATM that, on input w, performs the following computation.
   - Start using existential states.
   - Guess an accepting computation path $p = c_1, \ldots, c_n$ of $\mathcal{M}^{\textbf{SAT}}$ on w (note that this path is polynomial in w). Let $y_1, \ldots, y_m$ and $z_1, \ldots, z_\ell$ be all of the oracle queries executed in p that are answered positively and negatively, respectively.
   - For all $i = 2, \ldots, n$, check that c_i is a possible successor configuration of c_{i-1}.
   - For every $i = 1, \ldots, m$, check that $y_i \in \textbf{SAT}$. To do so, guess an assignment α_i for y_i and check that α_i is a satisfying assignment for y_i.
   - Switch to universal states.
   - For every $i = 1, \ldots, \ell$, check that $z_i \notin \textbf{SAT}$. To do so, guess an assignment α_i for z_i and check that α_i is not a satisfying assignment for z_i.
   - The ATM $\mathcal{N}$ accepts if and only if all of the above checks pass.
3. Discuss: $\mathcal{N}$ is a poly-time ATM that decides **L**.
4. Since we only alternate the type of states once and we start with existential states, we have that $\mathbf{L} \in \Sigma_2\mathrm{P}$.

Exercise 4

Show the following implication:
If there is any k with $\Sigma_k^{\mathrm{P}} = \Sigma_{k+1}^{\mathrm{P}}$, then $\Sigma_k^{\mathrm{P}} = \Sigma_j^{\mathrm{P}} = \Pi_j^{\mathrm{P}}$ for all $j > k$ and $\mathrm{PH} = \Sigma_k^{\mathrm{P}}$.

Solution.

1. Assume that $\Sigma_k^{\mathrm{P}} = \Sigma_{k+1}^{\mathrm{P}}$ for some $k \geq 1$ such that.
2. We show via induction that $\Sigma_j^{\mathrm{P}} = \Sigma_{j+1}^{\mathrm{P}}$ for all $j \geq k$.
   - The base case (i.e., $\Sigma_k^{\mathrm{P}} = \Sigma_{k+1}^{\mathrm{P}}$) follows from (1).
   - Induction step: we show that, for any $j > k$, we have that $\Sigma_j^{\mathrm{P}} = \Sigma_{j+1}^{\mathrm{P}}$.
     - $\Sigma_{j-1}^{\mathrm{P}} = \Sigma_j^{\mathrm{P}}$ by induction hypothesis.
     - $\Sigma_j^{\mathrm{P}} = \mathrm{NP}^{\Sigma_{j-1}^{\mathrm{P}}}$ and $\Sigma_{j+1}^{\mathrm{P}} = \mathrm{NP}^{\Sigma_j^{\mathrm{P}}}$ by definition.
     - Therefore, $\Sigma_j^{\mathrm{P}} = \Sigma_{j+1}^{\mathrm{P}}$.
3. By (2): $\Sigma_k^{\mathrm{P}} = \Sigma_j^{\mathrm{P}}$ for all $j \geq k$.
4. By definition, $\Pi_{i+1}^{\mathrm{P}} = \mathrm{coNP}^{\Sigma_i^{\mathrm{P}}}$ for all $i \geq 1$.
5. For any $j > k$, we have that $\Pi_j^{\mathrm{P}} \subseteq \Sigma_{j+1}^{\mathrm{P}}$ and $\Sigma_k^{\mathrm{P}} \subseteq \Pi_{k+1}^{\mathrm{P}} \subseteq \Pi_j^{\mathrm{P}}$ by definition.
   Therefore, $\Pi_j^{\mathrm{P}} = \Sigma_k^{\mathrm{P}}$.

Exercise 4

Show the following implication:
If there is any k with $\Sigma_k^{\mathrm{P}} = \Sigma_{k+1}^{\mathrm{P}}$, then $\Sigma_j^{\mathrm{P}} = \Pi_j^{\mathrm{P}} = \Sigma_k^{\mathrm{P}}$ for all $j > k$ and $\mathrm{PH} = \Sigma_k^{\mathrm{P}}$.

Solution. We show that $\mathrm{PH} = \Sigma_k^{\mathrm{P}}$.

1. $\mathrm{PH} = \bigcup_{i \geq 1} \Sigma_i \mathrm{P}$ by definition.
2. By (1): $\Sigma_k^{\mathrm{P}} \subseteq \mathrm{PH}$.
3. $\Sigma_j^{\mathrm{P}} \subseteq \Sigma_k^{\mathrm{P}}$ for all $j \leq k$ by definition.
4. By (3) from the previous slide: $\Sigma_j^{\mathrm{P}} \subseteq \Sigma_k^{\mathrm{P}}$ for all $j \geq k$.
5. By (3) and (4): $\bigcup_{i \geq 1} \Sigma_i \mathrm{P} = \Sigma_k^{\mathrm{P}}$

Exercise 5

Show that $\mathrm{PH} \subseteq \mathrm{PS{\small PACE}}$.

Solution.

1. Let $\mathbf{L} \in \mathrm{PH} = \bigcup_{i \geq 1} \Sigma_i \mathrm{P}$.
2. By (1), $\mathbf{L} \in \Sigma_i \mathrm{P}$ for some $k \geq 1$.
3. By (2), $\mathbf{L}$ can be solved by a poly-time bounded Σ_i ATM $\mathcal{M}$.
4. We simulate the ATM $\mathcal{M}$ using a space bounded TM $\mathcal{S}$:
   - $\mathcal{S}$ performs a depth-first search of the configuration tree of $\mathcal{M}$
   - The acceptance status of each node is computed recursively (similar to typical $\mathrm{PS{\small PACE}}$ algorithms we have seen before).
   - $\mathcal{M}$ accepts exactly if the root of the configuration tree is accepting
5. By (4), $\mathcal{S}$ decides $\mathbf{L}$ (discuss).
6. By (4), $\mathcal{S}$ is poly-space bounded (discuss).
7. By (5) and (6), $\mathbf{L} \in \mathrm{PS{\small PACE}}$.

Exercise 6

Let $\mathbf{A}$ be a language and let $\mathbf{F}$ be a finite set such that $\mathbf{A} \cap \mathbf{F} = \emptyset$.
Show that

$$\mathrm{P}^{\mathbf{A}} = \mathrm{P}^{\mathbf{A} \cup \mathbf{F}} \quad \text{and} \quad \mathrm{NP}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A} \cup \mathbf{F}}.$$

Infer that there exist infinitely many oracles A and B such that

$$\mathrm{P}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A}} \quad \text{and} \quad \mathrm{P}^{\mathbf{B}} \neq \mathrm{NP}^{\mathbf{B}}.$$

Exercise 6s

Let $\mathbf{A}$ be a language and let $\mathbf{F}$ be a finite set with $\mathbf{A} \cap \mathbf{F} = \emptyset$.
Show that $\mathrm{P}^{\mathbf{A}} = \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$ and $\mathrm{NP}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A} \cup \mathbf{F}}$.

Solution. We first show that $\mathrm{P}^{\mathbf{A}} \subseteq \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$.

1. Let $\mathbf{L} \in \mathrm{P}^{\mathbf{A}}$ and let $\mathcal{M}$ be a poly-time OTM such that $\mathcal{M}^{\mathbf{A}}$ accepts $\mathbf{L}$.
2. Let $\mathcal{N}$ be the OTM that is obtained by modifying $\mathcal{M}$ in the following manner:
   - Whenever $\mathcal{M}$ enters the query state, the machine $\mathcal{N}$ first checks if the content of its oracle tape is an element of $\mathbf{F}$ (e.g., by running a finite automaton on the content). If so, then $\mathcal{N}$ deletes the content of the oracle tape and then enters the query-rejection state; otherwise, $\mathcal{N}$ enters the query state.
   - In all other configurations, $\mathcal{N}$ behaves exactly like $\mathcal{M}$.
3. By (2): $\mathcal{N}^{\mathbf{A}}$ accepts $\mathbf{L}$, $\mathcal{N}^{\mathbf{A}}$ and $\mathcal{N}^{\mathbf{A} \cup \mathbf{F}}$ accept the same language, and $\mathcal{N}$ runs in polynomial time (discuss).
4. By (3): $\mathbf{L} \in \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$.

Note that we can use the same argument to show that $\mathrm{NP}^{\mathbf{A}} \subseteq \mathrm{NP}^{\mathbf{A} \cup \mathbf{F}}$.

Exercise 6

Let $\mathbf{A}$ be a language and let $\mathbf{F}$ be a finite set with $\mathbf{A} \cap \mathbf{F} = \emptyset$.
Show that $\mathrm{P}^{\mathbf{A}} = \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$ and $\mathrm{NP}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A} \cup \mathbf{F}}$.

Solution. Then, w show that $\mathrm{P}^{\mathbf{A} \cup \mathbf{F}} \subseteq \mathrm{P}^{\mathbf{A}}$.

1. Let $\mathbf{L} \in \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$ and let $\mathcal{M}$ be a poly-time OTM such that $\mathcal{M}^{\mathbf{A} \cup \mathbf{F}}$ accepts $\mathbf{L}$.
2. Let $\mathcal{N}$ be the OTM that is obtained by modifying $\mathcal{M}$ in the following manner:
   - Whenever $\mathcal{M}$ enters the query state, the machine $\mathcal{N}$ first checks if the content of its oracle tape is an element of $\mathbf{F}$ (e.g., by running a finite automaton on the content). If so, then $\mathcal{N}$ deletes the content of the oracle tape and then enters the *query-accepting* state; otherwise, $\mathcal{N}$ enters the query state.
   - In all other configurations, $\mathcal{N}$ behaves exactly like $\mathcal{M}$.
3. By (2): $\mathcal{N}^{\mathbf{A}}$ accepts $\mathbf{L}$, $\mathcal{N}^{\mathbf{A} \cup \mathbf{F}}$ and $\mathcal{N}^{\mathbf{A}}$ accept the same language, and $\mathcal{N}$ runs in polynomial time (discuss).
4. By (3): $\mathbf{L} \in \mathrm{P}^{\mathbf{A}}$.

Note that we can use the same argument to show that $\mathrm{NP}^{\mathbf{A} \cup \mathbf{F}} \subseteq \mathrm{NP}^{\mathbf{A}}$.

Exercise 5

Let $\mathbf{A}$ be a language and let $\mathbf{F}$ be a finite set with $\mathbf{A} \cap \mathbf{F} = \emptyset$.
Show that $\mathrm{P}^{\mathbf{A}} = \mathrm{P}^{\mathbf{A} \cup \mathbf{F}}$ and $\mathrm{NP}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A} \cup \mathbf{F}}$.
Show that there are infinitely many oracles $\mathbf{A}$ and $\mathbf{B}$ with $\mathrm{P}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A}}$ and $\mathrm{P}^{\mathbf{B}} \neq \mathrm{NP}^{\mathbf{B}}$.

1. By the Baker-Gill-Solovay Theorem, we have that there are oracles $\mathbf{A}$ and $\mathbf{B}$ such that $\mathrm{P}^{\mathbf{A}} = \mathrm{NP}^{\mathbf{A}}$ and $\mathrm{P}^{\mathbf{B}} \neq \mathrm{NP}^{\mathbf{B}}$.
2. Applying the argument from the previous slides, we can modify the oracles $\mathbf{A}$ and $\mathbf{B}$ in a finite manner preserving the relationships among the classes.
3. Since there are infinitely many such modifications, we obtain that there exist infinitely many such oracles $\mathbf{A}$ and $\mathbf{B}$.