Exercise Sheet 8: Alternation

David Carral

January 8, 2020
Exercise 1

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

**ExactIS** = \{(G, k) | \|S\| = k for some independent set S in G and \|S'\| \leq k for every independent set S' in G\}

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

**Solution.**

boolean $\text{Exact-Independent-Set}(\text{graph } G = (V, E), \text{integer } k)$

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 $\Sigma_2^P = \Sigma_2^P = NP^{NP}$?
Exercise 2

Consider the Japanese game *go-moku* that is played by two players *X* and *O* on a 19×19 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

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

Describe a polynomial-time ATM that solves $\text{GM}$. 
Exercise 2

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

Solution.

```java
boolean xHas WinningStr(position B) {
    if(isXWinPos(B)) return TRUE;
    if(isOWinPos(B) or FullBoard(B)) return FALSE;
    if(isXTurn(B))
        existentially choose \( B' \in \text{next}(B) \)
        return xHasWinningStr(B');
    if(isOTurn(B))
        universally choose \( B' \in \text{next}(B) \)
        return xHasWinningStr(B');
}
```

Is the algorithm correct?
Does it run in polynomial time?
Is \( \text{GM} \) in any level of the polynomial hierarchy?
Exercise 3

Show $NP^{SAT} \subseteq \Sigma_2P$.

Solution.

1. Let $L$ be a language in $NP^{SAT}$ and let $M^{SAT}$ be a poly-time NTM that decides $L$.
2. Let $N$ 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 $M^{SAT}$ 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 $y_i \in SAT$. To do so, guess an assignment $\alpha_i$ for $y_i$ and check that $\alpha_i$ is a satisfying assignment for $y_i$.
   ▶ Switch to universal states.
   ▶ For every $i = 1, \ldots, \ell$, check that $z_i \notin SAT$. To do so, guess an assignment $\alpha_i$ for $z_i$ and check that $\alpha_i$ is not a satisfying assignment for $z_i$.
   ▶ The ATM $N$ accepts if and only if all of the above checks pass.
3. Discuss: $N$ 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 $L \in \Sigma_2P$. 
Exercise 4

Show the following implication:
If there is any $k$ with $\Sigma^P_k = \Sigma^P_{k+1}$, then $\Sigma^P_k = \Sigma^P_j = \Pi^P_j$ for all $j > k$ and $\text{PH} = \Sigma^P_k$.

Solution.

1. Assume that $\Sigma^P_k = \Sigma^P_{k+1}$ for some $k \geq 1$ such that.

2. We show via induction that $\Sigma^P_j = \Sigma^P_{j+1}$ for all $j \geq k$.
   ▶ The base case (i.e., $\Sigma^P_k = \Sigma^P_{k+1}$) follows from (1).
   ▶ Induction step: we show that, for any $j > k$, we have that $\Sigma^P_j = \Sigma^P_{j+1}$.
      ▶ $\Sigma^P_{j-1} = \Sigma^P_j$ by induction hypothesis.
      ▶ $\Sigma^P_j = \text{NP}^P \Sigma^P_{j-1}$ and $\Sigma^P_{j+1} = \text{NP}^P \Sigma^P_j$ by definition.
      ▶ Therefore, $\Sigma^P_j = \Sigma^P_{j+1}$.

3. By (2): $\Sigma^P_k = \Sigma^P_j$ for all $j \geq k$.

4. By definition, $\Pi^P_{i+1} = \text{coNP}^P \Sigma^P_i$ for all $i \geq 1$.

5. For any $j > k$, we have that $\Pi^P_j \subseteq \Sigma^P_{j+1}$ and $\Sigma^P_k \subseteq \Pi^P_{k+1} \subseteq \Pi^P_j$ by definition.
   Therefore, $\Pi^P_j = \Sigma^P_k$. 
Exercise 4

Show the following implication:

If there is any \( k \) with \( \Sigma^P_k = \Sigma^P_{k+1} \), then \( \Sigma^P_j = \Pi^P_j = \Sigma^P_k \) for all \( j > k \) and \( PH = \Sigma^P_k \).

Solution. We show that \( PH = \Sigma^P_k \).

1. \( PH = \bigcup_{i \geq 1} \Sigma_i P \) by definition.
2. By (1): \( \Sigma^P_k \subseteq PH \).
3. \( \Sigma^P_j \subseteq \Sigma^P_k \) for all \( j \leq k \) by definition.
4. By (3) from the previous slide: \( \Sigma^P_j \subseteq \Sigma^P_k \) for all \( j \geq k \).
5. By (3) and (4): \( \bigcup_{i \geq 1} \Sigma_i P = \Sigma^P_k \)
Show that $\text{PH} \subseteq \text{PSPACE}$.

Solution.

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

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

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

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

$$P^A = NP^A \quad \text{and} \quad P^B \neq NP^B.$$
Exercise 6s

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

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

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

Note that we can use the same argument to show that $NP^A \subseteq NP^{A \cup F}$. 
Let $A$ be a language and let $F$ be a finite set with $A \cap F = \emptyset$. Show that $P^A = P^{A \cup F}$ and $NP^A = NP^{A \cup F}$.

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

1. Let $L \in P^{A \cup F}$ and let $M$ be a poly-time OTM such that $M^{A \cup F}$ accepts $L$.

2. Let $N$ be the OTM that is obtained by modifying $M$ in the following manner:
   - Whenever $M$ enters the query state, the machine $N$ first checks if the content of its oracle tape is an element of $F$ (e.g., by running a finite automaton on the content). If so, then $N$ deletes the content of the oracle tape and then enters the query-accepting state; otherwise, $N$ enters the query state.
   - In all other configurations, $N$ behaves exactly like $M$.

3. By (2): $N^A$ accepts $L$, $N^{A \cup F}$ and $N^A$ accept the same language, and $N$ runs in polynomial time (discuss).

4. By (3): $L \in P^A$.

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

Exercise 5

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

1. By the Baker-Gill-Solovay Theorem, we have that there are oracles \( A \) and \( B \) such that \( P^A = NP^A \) and \( P^B \neq NP^B \).
2. Applying the argument from the previous slides, we can modify the oracles \( A \) and \( B \) 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 \( A \) and \( B \).