Computational Complexity

Lecture 5: Relativization and the Baker-Gill-Solovay Theorem

Ronald de Haan
me@ronalddehaan.eu

University of Amsterdam

April 15, 2025

Recap

What we saw last time..

- Diagonalization arguments
- Time Hierarchy Theorems
- $\mathsf{P} \neq \mathsf{EXP}$

What will we do today?

- Can we use diagonalization to attack $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$? (Spoiler: no.)
- Limits of diagonalization
- Relativizing results
- Oracles

Diagonalization

- One concrete interpretation of *diagonalization proofs*:

  any proof technique that depends on the following properties of TMs:

  (I) effective representation of TMs by strings

  (II) ability of one TM to simulate another efficiently

Diagonalization

- One concrete interpretation of *diagonalization proofs*:

  any proof technique that depends on the following properties of TMs:

  (I) effective representation of TMs by strings

  (II) ability of one TM to simulate another efficiently

- We will see some limits of these proof techniques.

Oracles

Oracles

- Black-box machine that can solve a decision problem O in a single time-step

Oracle Turing machines

Definition

An *oracle Turing machine* is a TM $\mathbb{M}$ that has a special (read-write) tape that we call the *oracle tape* and three special states $q_{\mathsf{query}}, q_{\mathsf{yes}}, q_{\mathsf{no}} \in Q$.

To execute $\mathbb{M}$, we specify some $O \subseteq \{0,1\}^*$ that is used as the *oracle* for $\mathbb{M}$.

Whenever during the execution, $\mathbb{M}$ is in the state q_{query} the machine (in the next step) enters the state q_{yes} if $w \in O$ and the state q_{no} if $w \notin O$—where w denotes the current contents of the special oracle tape.
The tape contents and tape heads do not change/move.

$\mathbb{M}^O(x)$ denotes the output of $\mathbb{M}$ on input x with oracle O.

Oracle Turing machines

Definition

An *oracle Turing machine* is a TM $\mathbb{M}$ that has a special (read-write) tape that we call the *oracle tape* and three special states $q_{\mathsf{query}}, q_{\mathsf{yes}}, q_{\mathsf{no}} \in Q$.

To execute $\mathbb{M}$, we specify some $O \subseteq \{0,1\}^*$ that is used as the *oracle* for $\mathbb{M}$.

Whenever during the execution, $\mathbb{M}$ is in the state q_{query} the machine (in the next step) enters the state q_{yes} if $w \in O$ and the state q_{no} if $w \notin O$—where w denotes the current contents of the special oracle tape.
The tape contents and tape heads do not change/move.

$\mathbb{M}^O(x)$ denotes the output of $\mathbb{M}$ on input x with oracle O.

- An oracle TM knows how to use *any* oracle $O \subseteq \{0,1\}^*$

Relativized complexity classes

Definition

Let $O \subseteq \{0,1\}^*$ be a decision problem.

- P^O is the set of all decision problems that can be decided by a polynomial-time deterministic TM with oracle access to O.
- NP^O is the set of all decision problems that can be decided by a polynomial-time nondeterministic TM with oracle access to O.

- We will use similar notation for variants of other complexity classes that are based on Turing machines with bounds on the running time, e.g., EXP^O.

Diagonalization

- One concrete interpretation of *diagonalization proofs*:

  any proof technique that depends on the following properties of TMs:

  (I) effective representation of TMs by strings

  (II) ability of one TM to simulate another efficiently

Diagonalization

- One concrete interpretation of *diagonalization proofs*:

  any proof technique that depends on the following properties of TMs:

  (I) effective representation of TMs by strings

  (II) ability of one TM to simulate another efficiently

- We will see some limits of these proof techniques.

Relativizing results

- Regardless of the choice of $O \subseteq \{0,1\}^*$, properties (I) and (II) also hold for oracle TMs
- *Relativizing results* are results that depend only on (I) and (II)
  - E.g., $\mathsf{P} \subsetneq \mathsf{EXP}$
- Relativizing results also hold when you add *any* oracle $O \subseteq \{0,1\}^*$
  - E.g., $\mathsf{P}^O \subsetneq \mathsf{EXP}^O$, for each $O \subseteq \{0,1\}^*$

Theorem (Baker, Gill, Solovay 1975)

There exist $A, B \subseteq \{0,1\}^*$ *such that* $\mathsf{P}^A = \mathsf{NP}^A$ *and* $\mathsf{P}^B \neq \mathsf{NP}^B$.

- So no proof that $\mathsf{P} = \mathsf{NP}$ or $\mathsf{P} \neq \mathsf{NP}$ can be relativizing.

Oracle A such that $\mathsf{P}^A = \mathsf{NP}^A$

- Let $A = \{ \ (\alpha, x, 1^n) \mid \mathbb{M}_\alpha$ outputs 1 on input x within 2^n steps $\}$.
- Then $\mathsf{EXP} \subseteq \mathsf{P}^A \subseteq \mathsf{NP}^A \subseteq \mathsf{EXP}$.

Oracle A such that $\mathsf{P}^A = \mathsf{NP}^A$

- Let $A = \{\ (\alpha, x, 1^n) \mid \mathbb{M}_\alpha$ outputs 1 on input x within 2^n steps $\}$.
- Then $\mathsf{EXP} \subseteq \mathsf{P}^A \subseteq \mathsf{NP}^A \subseteq \mathsf{EXP}$.
- $\mathsf{EXP} \subseteq \mathsf{P}^A$ *(idea)*:
    - With one oracle query to A you can do exponential-time computation in one step.

Oracle A such that $\mathsf{P}^A = \mathsf{NP}^A$

- Let $A = \{\ (\alpha, x, 1^n) \mid \mathbb{M}_\alpha$ outputs 1 on input x within 2^n steps $\}$.
- Then $\mathsf{EXP} \subseteq \mathsf{P}^A \subseteq \mathsf{NP}^A \subseteq \mathsf{EXP}$.
- $\mathsf{EXP} \subseteq \mathsf{P}^A$ *(idea)*:
  - With one oracle query to A you can do exponential-time computation in one step.
- $\mathsf{NP}^A \subseteq \mathsf{EXP}$ *(idea)*:
  - Simulate computation of NP^A machine in exponential time.
    - Enumerate all sequences of nondeterministic choices.
    - Compute answer to each (polynomial-size) oracle query.

Oracle B such that $\mathrm{P}^B \neq \mathrm{NP}^B$

- For any $B \subseteq \{0,1\}^*$, let $U_B = \{\ 1^n \mid \text{there is some } x \in \{0,1\}^n \text{ such that } x \in B\ \}$.

Oracle B such that $\mathsf{P}^B \neq \mathsf{NP}^B$

- For any $B \subseteq \{0,1\}^*$, let $U_B = \{\ 1^n \mid \text{there is some } x \in \{0,1\}^n \text{ such that } x \in B\ \}$.
- Then $U_B \in \mathsf{NP}^B$.
  - On any input 1^n, we use nondeterminism to guess $x \in \{0,1\}^n$, and query the oracle B to check if $x \in B$.

Oracle B such that $\mathsf{P}^B \neq \mathsf{NP}^B$

- For any $B \subseteq \{0,1\}^*$, let $U_B = \{\ 1^n \mid \text{there is some } x \in \{0,1\}^n \text{ such that } x \in B\ \}$.
- Then $U_B \in \mathsf{NP}^B$.
  - On any input 1^n, we use nondeterminism to guess $x \in \{0,1\}^n$, and query the oracle B to check if $x \in B$.
- We construct some $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$.
  - Using diagonalization. :-)

Construct $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$

- We gradually build up B in stages. Start with $\emptyset$. One stage for each $i \in \{0,1\}^*$.

Construct $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$

- We gradually build up B in stages. Start with $\emptyset$. One stage for each $i \in \{0,1\}^*$.
- In stage i:
  - For only finitely many strings x we chose whether $x \in B$ or $x \notin B$.
    Let n be larger than the length of any such x.

Construct $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$

- We gradually build up B in stages. Start with $\emptyset$. One stage for each $i \in \{0,1\}^*$.
- In stage i:
  - For only finitely many strings x we chose whether $x \in B$ or $x \notin B$.
    Let n be larger than the length of any such x.
  - Run $\mathbb{M}_i$ on input 1^n for $2^n/10$ steps.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for strings for which we already determined if $x \in B$ or $x \notin B$, use the same answer.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for new strings, answer that $x \notin B$.

Construct $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$

- We gradually build up B in stages. Start with $\emptyset$. One stage for each $i \in \{0,1\}^*$.
- In stage i:
  - For only finitely many strings x we chose whether $x \in B$ or $x \notin B$.
    Let n be larger than the length of any such x.
  - Run $\mathbb{M}_i$ on input 1^n for $2^n/10$ steps.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for strings for which we already determined if $x \in B$ or $x \notin B$, use the same answer.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for new strings, answer that $x \notin B$.
  - Ensure that $\mathbb{M}_i$'s answer on 1^n after $2^n/10$ steps is wrong.
    - If $\mathbb{M}_i$ accepts 1^n, for all strings $x \in \{0,1\}^n$, let $x \notin B$.
    - If $\mathbb{M}_i$ rejects 1^n, take some yet unqueried $x \in \{0,1\}^n$, and let $x \in B$.

Construct $B \subseteq \{0,1\}^*$ such that $U_B \notin P^B$

- We gradually build up B in stages. Start with $\emptyset$. One stage for each $i \in \{0,1\}^*$.
- In stage i:
  - For only finitely many strings x we chose whether $x \in B$ or $x \notin B$.
    Let n be larger than the length of any such x.
  - Run $\mathbb{M}_i$ on input 1^n for $2^n/10$ steps.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for strings for which we already determined if $x \in B$ or $x \notin B$, use the same answer.
    - If $\mathbb{M}_i$ queries "$x \in B$?" for new strings, answer that $x \notin B$.
  - Ensure that $\mathbb{M}_i$'s answer on 1^n after $2^n/10$ steps is wrong.
    - If $\mathbb{M}_i$ accepts 1^n, for all strings $x \in \{0,1\}^n$, let $x \notin B$.
    - If $\mathbb{M}_i$ rejects 1^n, take some yet unqueried $x \in \{0,1\}^n$, and let $x \in B$.
- Each TM is represented by infinitely many i, and every polynomial is smaller than $2^n/10$ for large enough n. So no TM can decide U_B in polynomial time with oracle access to B.

No relativizing results for P vs. NP

- Suppose that we have a relativizing proof that $\mathsf{P} = \mathsf{NP}$
- Then also $\mathsf{P}^B = \mathsf{NP}^B$, contradicting $\mathsf{P}^B \neq \mathsf{NP}^B$.

No relativizing results for P vs. NP

- Suppose that we have a relativizing proof that $\mathsf{P} = \mathsf{NP}$
- Then also $\mathsf{P}^B = \mathsf{NP}^B$, contradicting $\mathsf{P}^B \neq \mathsf{NP}^B$.

- Suppose that we have a relativizing proof that $\mathsf{P} \neq \mathsf{NP}$
- Then also $\mathsf{P}^A \neq \mathsf{NP}^A$, contradicting $\mathsf{P}^A = \mathsf{NP}^A$.

Recap

- Limits of diagonalization, relativizing results
- Oracles
- There exist $A, B \subseteq \{0,1\}^*$ such that $\mathsf{P}^A = \mathsf{NP}^A$ and $\mathsf{P}^B \neq \mathsf{NP}^B$.

Next time

- Space-bounded computation
- Limits on memory space