Computational Complexity

Lecture 8
More of the Polynomial Hierarchy
Oracle-based Definition
Recall PH

\[
\begin{align*}
\{x \mid \exists u_1 \forall u_2 \exists u_3 \text{ F}(x, u_1, u_2, u_3)\} & \quad \{x \mid \forall u_1 \exists u_2 \forall u_3 \text{ F}(x, u_1, u_2, u_3)\} \\
\{x \mid \exists u_1 \forall u_2 \text{ F}(x, u_1, u_2)\} & \quad \{x \mid \forall u_1 \exists u_2 \text{ F}(x, u_1, u_2)\} \\
\{x \mid \exists u_1 \text{ F}(x, u_1)\} & \quad \{x \mid \forall u_1 \text{ F}(x, u_1)\} \\
\{x \mid \text{F}(x)\} & \quad \{x \mid \forall u_1 \text{ F}(x, u_1)\}
\end{align*}
\]
Oracle Machines

Recall Oracle Machine
Oracle Machines

Recall Oracle Machine

Writes queries on query-tape, enters and leaves query state, and expects answer from oracle on the tape
Oracle Machines

- Recall Oracle Machine
  - Writes queries on query-tape, enters and leaves query state, and expects answer from oracle on the tape
  - Can run an oracle machine with any oracle
Oracle Machines

Recall Oracle Machine

- Writes queries on query-tape, enters and leaves query state, and expects answer from oracle on the tape
- Can run an oracle machine with any oracle
- Oracle fully specified by the input-output behavior
Oracle Machines

Recall Oracle Machine

- Writes queries on query-tape, enters and leaves query state, and expects answer from oracle on the tape

- Can run an oracle machine with any oracle

- Oracle fully specified by the input-output behavior

- Language oracle: answer is a single bit
Oracle Machines

Recall Oracle Machine

- Writes queries on query-tape, enters and leaves query state, and expects answer from oracle on the tape

- Can run an oracle machine with any oracle

- Oracle fully specified by the input-output behavior

- Language oracle: answer is a single bit

- This is what we consider
Oracle Machines (ctd.)
Oracle Machines (ctd.)

- Non-deterministic oracle machine
Oracle Machines (ctd.)

- Non-deterministic oracle machine
- Can make non-deterministic choices and make oracle queries. (Note: oracles are deterministic!)
Oracle Machines (ctd.)

- Non-deterministic oracle machine
- Can make non-deterministic choices and make oracle queries. (Note: oracles are deterministic!)
- Said to accept if any thread reaches accept state
Oracle Machines (ctd.)

- Non-deterministic oracle machine

- Can make non-deterministic choices and make oracle queries. (Note: oracles are deterministic!)

- Said to accept if any thread reaches accept state

- Equivalently, a deterministic oracle machine which takes a (read-once) certificate $w$ (the list of non-deterministic choices)
Oracle Machines (ctd.)

- Non-deterministic oracle machine
  - Can make non-deterministic choices and make oracle queries. (Note: oracles are deterministic!)
  - Said to accept if any thread reaches accept state
  - Equivalently, a deterministic oracle machine which takes a (read-once) certificate w (the list of non-deterministic choices)
    - Said to accept x if ∃w such that (x,w) takes it to accepting state
Oracle Machines (ctd.)

Non-deterministic oracle machine

Can make non-deterministic choices and make oracle queries. (Note: oracles are deterministic!) All threads reach

Said to accept if any thread reaches accept state

Equivalently, a deterministic oracle machine which takes a (read-once) certificate w (the list of non-deterministic choices)

Said to accept x if there is such that (x, w) takes it to accepting state

NPA
$\text{NP}^A$:

Class of languages accepted by oracle NTMs with oracle for $A$ in poly time.
$\text{NP}^A$

- $\text{NP}^A$ : class of languages accepted by oracle NTMs with oracle for $A$ in poly time
- Certificate version: $\text{NP}^A$ has languages of the form
NP^A

- NP^A: class of languages accepted by oracle NTMs with oracle for A in poly time
- Certificate version: NP^A has languages of the form
  \[ B = \{ x \mid \exists w \ M^A(x, w) = 1 \} \]
NP^A

- NP^A: class of languages accepted by oracle NTMs with oracle for A in poly time
- Certificate version: NP^A has languages of the form
  \[ B = \{ x \mid \exists w \ M^A(x,w) = 1 \} \]
  where M deterministic oracle machine
**$\text{NP}^A$**

- $\text{NP}^A$: class of languages accepted by oracle NTMs with oracle for $A$ in poly time
- Certificate version: $\text{NP}^A$ has languages of the form
  - $B = \{x \mid \exists w \ M^A(x,w) = 1\}$
    - where $M$ deterministic oracle machine
    - $M^A$ runs in $\text{poly}(|x|)$ time and $|w| = \text{poly}(|x|)$
NP\(^A\):

- **NP\(^A\)**: class of languages accepted by oracle NTMs with oracle for \(A\) in poly time.

- Certificate version: **NP\(^A\)** has languages of the form

\[
B = \{x \mid \exists w \ M^A(x,w) = 1\}
\]

where \(M\) deterministic oracle machine

\(M^A\) runs in \(\text{poly}(|x|)\) time and \(|w| = \text{poly}(|x|)\)

i.e., \(B = \{x \mid \exists w \ (x,w) \in L\}\), where \(L\) in \(P^A\)
NP\(^A\)

- **NP\(^A\)**: class of languages accepted by oracle NTMs with oracle for \(A\) in poly time

- Certificate version: **NP\(^A\)** has languages of the form

\[
B = \{x \mid \exists w \ M^A(x,w) = 1\}
\]

  where \(M\) deterministic oracle machine

  - \(M^A\) runs in poly(|x|) time and |w|=poly(|x|)

  i.e., \(B = \{x \mid \exists w \ (x,w) \in L\}\) , where \(L\) in \(P^A\)

- co-(NP\(^A\)) = (co-NP\(^A\))
NP^A

NP^A: class of languages accepted by oracle NTMs with oracle for A in poly time

Certificate version: NP^A has languages of the form

\[ B = \{x \mid \exists w \ M^A(x,w) = 1\} \]

where \( M \) deterministic oracle machine

\( M^A \) runs in poly(|x|) time and |w|=poly(|x|)

i.e., \( B = \{x \mid \exists w \ (x,w) \in L\} \), where L in P^A

\( \text{co-}(NP^A) = (\text{co-NP})^A \)

languages of the form \( \{x \mid \forall w \ (x,w) \in L\} \), where L in P^A
NPA
If $A$ in $P$, $NPA = NP$
If $A$ in $P$, $NP^A = NP$

Can “implement” the oracle as a subroutine
If $A$ in $P$, $NP^A = NP$

Can “implement” the oracle as a subroutine

If $A$ in $NP$?
If \( A \) in \( P \), \( NPA = NP \)

Can “implement” the oracle as a subroutine

If \( A \) in \( NP \)?

Oracle for \( A \) is an oracle for \( A^c \) too! \( NPA = NPA^c \)
\[ \text{If } A \text{ in } P, \text{ then } \text{NP}^A = \text{NP} \]

- Can "implement" the oracle as a subroutine
- If \( A \) in NP?
  - Oracle for \( A \) is an oracle for \( A^c \) too! \( \text{NP}^A = \text{NP}^{A^c} \)
  - \( \text{NP} \cup \text{co-NP} \subseteq \text{NP}^{\text{SAT}} \)
If $A$ in $P$, $NPA = NP$

- Can “implement” the oracle as a subroutine

If $A$ in $NP$?

- Oracle for $A$ is an oracle for $A^c$ too! $NPA = NPA^c$

$NP \cup \text{co-NP} \subseteq NPSAT$

- Can we better characterize $NPSAT$?
NP$_{NP}$ and relatives
\( \text{NP}^{\text{NP}} \) and relatives

\[ \text{NP}^{\text{SAT}} = \bigcup_{A \in \text{NP}} \text{NP}^A \]
$\text{NP}^{\text{NP}}$ and relatives

$\text{NP}^{\text{SAT}} = \bigcup_{A \in \text{NP}} \text{NP}^A$

Oracle for $A$ can be implemented using oracle for SAT in polynomial time (deterministically)
$\text{NP}^{\text{NP}}$ and relatives

$\text{NP}^{\text{SAT}} = \bigcup_{A \in \text{NP}} \text{NP}^A$

Oracle for $A$ can be implemented using oracle for SAT in polynomial time (deterministically)

$\text{NP}^{\text{SAT}}$ also called $\text{NP}^{\text{NP}}$
**NP^{NP} and relatives**

- \( \text{NP}^{\text{SAT}} = \bigcup_{A \in \text{NP}} \text{NP}^A \)
- Oracle for \( A \) can be implemented using oracle for \( \text{SAT} \) in polynomial time (deterministically)
- \( \text{NP}^{\text{SAT}} \) also called \( \text{NP}^{\text{NP}} \)

- \( \text{NP}^{\Sigma_k} = \bigcup_{A \in \Sigma_k} \text{NP}^A = \text{NP}^{\Sigma_k \text{SAT}} \)
**NP^{NP} and relatives**

- \( NP^{SAT} = \bigcup_{A \in NP} NP^A \)
  - Oracle for A can be implemented using oracle for SAT in polynomial time (deterministically)
- \( NP^{SAT} \) also called \( NP^{NP} \)
- \( NP^{\Sigma_k} = \bigcup_{A \in \Sigma_k} NP^A = NP^{\Sigma_k SAT} \)
- Will show \( NP^{\Sigma_k} = \Sigma_{k+1}^P \) (alt. definition for \( \Sigma_{k+1}^P \))
NP^{NP} and relatives

\[ NP^{SAT} = \bigcup_{A \in NP} NP^A \]

Oracle for A can be implemented using oracle for SAT in polynomial time (deterministically)

\[ NP^{SAT} \text{ also called } NP^{NP} \]

\[ NP^{\Sigma_k} = \bigcup_{A \in \Sigma_k} NP^A = NP^{\Sigma_k SAT} \]

Will show \[ NP^{\Sigma_k} = \Sigma_{k+1}^P \] (alt. definition for \( \Sigma_{k+1}^P \))

In particular, \[ NP^{NP} = \Sigma_2^P \]
\[ \Sigma_{k+1} = \mathsf{NP}^{\Sigma_k} \]
\[ \Sigma_{k+1} = \text{NP}^{\Sigma_k} \]

Consider \( L \in \Sigma_{k+1}^P \)
\[ \Sigma_{k+1} = NP^{\Sigma_k} \]

Consider \( L \in \Sigma_{k+1}^P \)

\( L = \{ x | \exists w \ (x, w) \in L' \} \), where \( L' \) in \( \Pi_k^P \)
Consider $L \in \Sigma_{k+1}^P$

$L = \{ x | \exists w (x,w) \in L' \}$, where $L'$ in $\Pi_k^P$

So $L$ in $NP^{L'}$ where $L'$ in $\Pi_k^P$
\[ \Sigma_{k+1} = \mathsf{NP}^{\Sigma_k} \]

Consider \( L \in \Sigma_{k+1}^P \)

- \( L = \{ x | \exists w \ (x,w) \in L' \} \), where \( L' \) in \( \Pi_k^P \)

- So \( L \) in \( \mathsf{NP}^{L'} \) where \( L' \) in \( \Pi_k^P \)

- So \( \mathsf{NP}^{L'} \subseteq \mathsf{NP}^{\Pi_k} = \mathsf{NP}^{\Sigma_k} \)
\[ \Sigma_{k+1} = NP^{\Sigma_k} \]

Consider \( L \in \Sigma_{k+1}^P \)

\( L = \{ x| \exists w \ (x,w) \in L' \} \), where \( L' \) in \( \Pi_k^P \)

So \( L \) in \( NP^{L'} \) where \( L' \) in \( \Pi_k^P \)

So \( NP^{L'} \subseteq NP^{\Pi_k} = NP^{\Sigma_k} \)

So \( \Sigma_{k+1}^P \subseteq NP^{\Sigma_k} \)
\[ \Sigma_{k+1} = NP^{\Sigma_k} \]

- Consider \( L \in \Sigma_{k+1}^P \)
  - \( L = \{ x | \exists w (x,w) \in L' \} \), where \( L' \) in \( \Pi_k^P \)
  - So \( L \) in \( NP^{L'} \) where \( L' \) in \( \Pi_k^P \)
  - So \( NP^{L'} \subseteq NP^{\Pi_k} = NP^{\Sigma_k} \)
  - So \( \Sigma_{k+1}^P \subseteq NP^{\Sigma_k} \)
- Now to show \( NP^{\Sigma_k} \subseteq \Sigma_{k+1}^P \)
\( NP^{\Sigma_k} \subseteq \Sigma_{k+1} \)
\[ \text{NP}^{\sum_k} \subseteq \sum_{k+1} \]

To show \( \text{NP}^A \subseteq \sum_{k+1}^P \) if \( A \) in \( \sum_k^P \).
$NP^{\Sigma_k} \subseteq \Sigma_{k+1}$

To show $NP^A \subseteq \Sigma_{k+1}^P$ if $A$ in $\Sigma_k^P$

For $B \in NP^A$ poly-time TM $M$ s.t. $B = \{ x | \exists w \ M^A(x,w)=1 \}$
\[ \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \]

To show \( \text{NP}^A \subseteq \Sigma_{k+1}^P \) if \( A \) in \( \Sigma_k^P \)

- For \( B \in \text{NP}^A \) poly-time TM \( M \) s.t. \( B = \{ x \mid \exists w \ M(x,w)=1 \} \)
- i.e., \( B = \{ x \mid \exists w \ \exists \text{ans} \ M^{\text{ans}}(x,w)=1 \text{ and } "\text{ans correct}" \} \)
\( \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \)

To show \( \text{NP}^A \subseteq \Sigma_{k+1}^P \) if \( A \) in \( \Sigma_k^P \)

- For \( B \in \text{NP}^A \) poly-time TM \( M \) s.t. \( B = \{ x | \exists w \ M^A(x, w) = 1 \} \)
- i.e., \( B = \{ x | \exists w \ \exists \text{ans} \ M^{<\text{ans}>}(x, w) = 1 \text{ and "ans correct"} \} \)
- To show \( C = \{(x, w, \text{ans}) | M^{<\text{ans}>}(x, w) = 1 \text{ and "ans correct"} \} \) in \( \Sigma_{k+1}^P \)
\[ \text{NP}^\Sigma_k \subseteq \Sigma_{k+1} \]

To show \( \text{NP}^A \subseteq \Sigma_{k+1}^P \) if \( A \) in \( \Sigma_k^P \)

- For \( B \in \text{NP}^A \) poly-time TM \( M \) s.t. \( B = \{ x \mid \exists w \ M^A(x,w)=1 \} \)
- i.e., \( B = \{ x \mid \exists w \ \exists \text{ans} \ M^{<\text{ans}>}(x,w)=1 \text{ and "ans correct"} \} \)
- To show \( C = \{(x,w,\text{ans}) \mid M^{<\text{ans}>}(x,w)=1 \text{ and "ans correct"} \} \) in \( \Sigma_{k+1}^P \)
- Then \( B \) also in \( \Sigma_{k+1}^P \)
\( \mathcal{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \)
To show $C = \{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \text{ and "ans correct"}\}$ in $\Sigma_{k+1}^P$
\[ \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \]

To show \( C = \{(x,w,\text{ans}) \mid M^{<\text{ans}>}(x,w)=1 \text{ and "ans correct"} \} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not
\( \mathsf{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \)

To show \( C = \{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \text{ and "ans correct"}\} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not

“ans correct”: \( (\text{ans}=1 \land z \in A) \) or \( (\text{ans}=0 \land z \notin A) \)
\[ \mathsf{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \]

To show \( C = \{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \text{ and "ans correct"}\} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not

“ans correct”: \( (\text{ans}=1 \land z \in A) \text{ or } (\text{ans}=0 \land z \notin A) \)

\( C=\{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \land \left[(\text{ans}=1 \land \exists u_1 \forall u_2 \ldots Q_k u_k F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \land \forall v_1 \exists v_2 \ldots Q'_k v_k F(z,v_1,\ldots)=0\right]\} \)
\( \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \)

To show \( C = \{(x,w,\text{ans}) | M^{\text{ans}}(x,w)=1 \text{ and "\text{ans correct}"} \} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not

“ans correct”: \( (\text{ans}=1 \land z \in A) \) or \( (\text{ans}=0 \land z \notin A) \)

\( C = \{(x,w,\text{ans}) | M^{\text{ans}}(x,w)=1 \land [(\text{ans}=1 \land \exists u_1 \forall u_2 \ldots Q_k u_k F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \land \forall v_1 \exists v_2 \ldots Q'_k v_k F(z,v_1,\ldots)=0)] \} \)

\( C = \{(x,w,\text{ans}) | \exists u_1 \forall u_2 v_1 \exists u_3 v_2 \ldots Q_k u_k v_{k-1} Q'_k v_k M^{\text{ans}}(x,w)=1 \land [(\text{ans}=1 \land F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \land F(z,v_1,\ldots)=0)] \} \)
\[ \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \]

To show \( C = \{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \text{ and "ans correct"}\} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not

"ans correct": (\( \text{ans}=1 \land z \in A \)) or (\( \text{ans}=0 \land z \notin A \))

\( C=\{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \land [(\text{ans}=1 \land \exists u_1 \forall u_2 \ldots Q_k u_k \ F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \land \forall v_1 \exists v_2 \ldots Q_k ^' v_k \ F(z,v_1,\ldots)=0)] \}\)

\( C=\{(x,w,\text{ans}) \mid \exists u_1 \forall u_2 v_1 \exists u_3 v_2 \ldots Q_k u_k v_k v_{k-1} Q_k ^' v_k \ M^{\text{ans}}(x,w)=1 \land [(\text{ans}=1 \land F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \land F(z,v_1,\ldots)=0)] \}\)
\[ \text{NP}^{\Sigma_k} \subseteq \Sigma_{k+1} \]

To show \( C = \{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \text{ and "ans correct"}\} \) in \( \Sigma_{k+1}^P \)

Suppose \( M \) makes only one query \( z=Z(x,w) \). \( \text{ans} \) is a single bit saying if \( z \) in \( A \) or not

"ans correct": \( (\text{ans}=1 \wedge z \in A) \) or \( (\text{ans}=0 \wedge z \notin A) \)

\( C=\{(x,w,\text{ans}) \mid M^{\text{ans}}(x,w)=1 \wedge [(\text{ans}=1 \wedge \exists u_1 \forall u_2 \ldots Q_k u_k F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \wedge \forall v_1 \exists v_2 \ldots Q' k v_k F(z,v_1,\ldots)=0)] \} \)

\( C=\{(x,w,\text{ans}) \mid \exists u_1 \forall u_2 v_1 \exists u_3 v_2 \ldots Q_k u_k v_{k-1} Q' k v_k \ M^{\text{ans}}(x,w)=1 \wedge [(\text{ans}=1 \wedge F(z,u_1,\ldots)=1) \text{ or } (\text{ans}=0 \wedge F(z,v_1,\ldots)=0)] \} \)

Changes for \( t \) queries: \( z=Z(x,w) \rightarrow (z^{(1)} \ldots z^{(t)}) = Z(x,w,\text{ans}), \ u_i \rightarrow u_i^{(1)} \ldots u_i^{(t)}, \ v_i \rightarrow v_i^{(1)} \ldots v_i^{(t)}, \) and use conjunction of \( t \) checks (for \( j=1,\ldots,t \)) of the form \[ (\text{ans}^{(j)}=1 \wedge F(z^{(j)},u_1^{(j)},\ldots)=1) \text{ or } (\text{ans}^{(j)}=0 \wedge F(z^{(j)},v_1^{(j)},\ldots)=0) \]
Oracle Version
Oracle Version

\[ \Sigma_{k+1}^P = \text{NP}^{\Sigma_k} \text{ (with } \Sigma_0^P = P) \]
Oracle Version

\[ \Sigma_{k+1}^p = \text{NP}^{\Sigma_k} \text{ (with } \Sigma_0^p = P) \]

\[ \Pi_{k+1}^p = \text{co-NP}^{\Pi_k} \text{ (with } \Pi_0^p = P) \]
Oracle Version

\[ \Sigma_{k+1}^P = \text{NP}^{\Sigma_k} \text{ (with } \Sigma_0^P = P) \]

\[ \Pi_{k+1}^P = \text{co-NP}^{\Pi_k} \text{ (with } \Pi_0^P = P) \]

\[ \Pi_{k+1}^P = \text{co-(NP}^{\Sigma_k}) = \text{co-NP}^{\Sigma_k} = \text{co-NP}^{\Pi_k} \]
$\Delta_{k^P}$
$\Delta_k^p$

$\Delta_{k+1}^p = p^{\Sigma_k} = p^{\Pi_k}$
\[ \Delta_k^p \]

\[ \Delta_{k+1}^p = p^{\Sigma_k} = p^{\Pi_k} \]

\[ \Delta_1^p = p \]
\( \Delta_k^p \)

- \( \Delta_{k+1}^p = p^{\Sigma_k} = p^{\Pi_k} \)
- \( \Delta_1^p = p \)
- \( \Delta_2^p = p^{\NP} \)
\[ \Delta_k^p \]

- \[ \Delta_{k+1}^p = \pi_S^k = \pi^p_k \]
- \[ \Delta_1^p = \pi \]
- \[ \Delta_2^p = \pi^{NP} \]

Note that \[ \Delta_2^p = \text{co-}\Delta_2^p \]
\[ \Delta_k^p \]

- \( \Delta_{k+1}^p = p^{\Sigma_k} = p^{\Pi_k} \)
- \( \Delta_1^p = p \)
- \( \Delta_2^p = p^{NP} \)

Note that \( \Delta_2^p = \text{co-}\Delta_2^p \)

- \( \Delta_{k+1}^p \supseteq \Sigma_k^p \cup \Pi_k^p \)
$$\Delta_k^P$$

- $$\Delta_{k+1}^P = p^{\Sigma_k} = p^{\Pi_k}$$
- $$\Delta_1^P = p$$
- $$\Delta_2^P = p^{NP}$$

Note that $$\Delta_2^P = \text{co-}\Delta_2^P$$

- $$\Delta_{k+1}^P \supseteq \Sigma_k^P \cup \Pi_k^P$$
- $$\Delta_{k+1}^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P$$ (why?)
\[ \Delta_k^P \]

- \[ \Delta_{k+1}^P = P^{\Sigma_k} = P^{\Pi_k} \]
  - \[ \Delta_1^P = P \]
  - \[ \Delta_2^P = P^{NP} \]

Note that \[ \Delta_2^P = \text{co-}\Delta_2^P \]

- \[ \Delta_{k+1}^P \supseteq \Sigma_k^P \cup \Pi_k^P \]
- \[ \Delta_{k+1}^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P \] (why?)

- \[ P^{\Sigma_k} \subseteq NP^{\Sigma_k} \cap coNP^{\Sigma_k} \]
PH

Diagram showing the relationships between complexity classes such as $\Sigma^p_3$, $\Pi^p_3$, $\Sigma^p_2$, $\Pi^p_2$, $\text{NP}$, $\text{coNP}$, and $\text{P}$. The diagram illustrates the hierarchy and containment relationships among these classes.
PH

Diagram showing the relationships between complexity classes such as \( \Sigma_3^P \), \( \Pi_3^P \), \( \Sigma_2^P \), \( \Pi_2^P \), NP, coNP, and P.
PH

\[ \Sigma_3^P \rightarrow \Sigma_2^P \rightarrow \Sigma_1^P \rightarrow \text{NP} \rightarrow \text{P} \]

\[ \Pi_3^P \rightarrow \Pi_2^P \rightarrow \Pi_1^P \rightarrow \text{coNP} \rightarrow \text{P} \]

\[ \text{NP} \rightarrow \text{coNP} \rightarrow \text{P} \]

\[ \text{NP}^\text{NP} \rightarrow \Sigma_2^P \rightarrow \Pi_2^P \rightarrow \text{coNP}^\text{NP} \]
PH

\[ \Sigma_2^P \rightarrow \Sigma_3^P \rightarrow \Pi_3^P \rightarrow \Sigma_2^P \]

\[ \Pi_2^P \rightarrow \Pi_3^P \rightarrow \Pi_2^P \]

\[ NP \rightarrow \Sigma_2^P \rightarrow NP \]

\[ NP \rightarrow \Pi_2^P \rightarrow NP \]

\[ NP \rightarrow \Sigma_3^P \rightarrow NP \]

\[ NP \rightarrow \Pi_3^P \rightarrow NP \]

\[ \text{coNP} \rightarrow \Sigma_2^P \rightarrow \text{coNP} \]

\[ \text{coNP} \rightarrow \Pi_2^P \rightarrow \text{coNP} \]

\[ \text{coNP} \rightarrow \Sigma_3^P \rightarrow \text{coNP} \]

\[ \text{coNP} \rightarrow \Pi_3^P \rightarrow \text{coNP} \]

\[ P \rightarrow \Sigma_2^P \rightarrow P \]

\[ P \rightarrow \Sigma_3^P \rightarrow P \]

\[ P \rightarrow \Pi_2^P \rightarrow P \]

\[ P \rightarrow \Pi_3^P \rightarrow P \]
PH

\[ \Sigma_3^P \rightarrow \Pi_3^P \]
\[ \Sigma_2^P \rightarrow \Pi_2^P \]
\[ NP \rightarrow coNP \]
\[ NP \rightarrow coNP \]
\[ P \rightarrow coNP \]
$\textbf{PH}$
Today

Today, more PH
Today

Today, more PH

Oracle-based definitions (in particular $\text{NP}^{\text{NP}} = \Sigma_2^P$)
Today

- Today, more PH
- Oracle-based definitions (in particular $\text{NP}^\text{NP} = \Sigma_2^P$)
- Next lecture, more PH
Today

- Today, more PH
  - Oracle-based definitions (in particular $\text{NP}^{\text{NP}} = \Sigma_2^P$)
- Next lecture, more PH
  - Alternating TM-based definitions
Today

- Today, more PH
  - Oracle-based definitions (in particular $\text{NP}^\text{NP} = \Sigma_2^P$)
- Next lecture, more PH
  - Alternating TM-based definitions
  - Time-Space tradeoffs