15-455
Relativization
FINITE STATE CONTROL

INFINITE TAPE

$q_0$

Is $\phi$ in SAT?

INPUT

INFINITE TAPE
FINITE STATE CONTROL

INFINITE TAPE

q₀

Is φ in SAT?

Yes

OUTPUT

INPUT

INFINITE TAPE

ORACLE TMs

Is φ in SAT?

Yes

GANDALF
ORACLE MACHINES

An oracle is a set \( A \) to which the TM may pose membership questions and always receive correct answers after one step of time.
\( \mathcal{M}^A \) denotes the machine \( M \) with access to an oracle for \( A \)

\[ \mathcal{P}^A = \{ L \mid L \text{ can be decided by a poly-time machine } M \text{ that uses an oracle for } A \} \]

\[ \mathcal{P}^{\text{SAT}} = \text{all languages that can be decided in deterministic polynomial time with an oracle for SAT} \]
Is $\text{NP} \subseteq \text{P}^{\text{SAT}}$?
Is $\text{NP} \subseteq \text{P}^{\text{SAT}}$?

Yes!

Since $\text{SAT}$ is NP-complete, any NP language can be polynomial time encoded in SAT.

So given $L \in \text{NP}$, testing membership in $L$ can be polytime reduced to testing membership in SAT.
Is coNP ⊆ P_{SAT}? 

Yes!

Given $L \in \text{coNP}$, test for membership in $\neg L \ (\in \text{NP} \subseteq P_{\text{SAT}})$ and flip the answer.
Is NP = NP^{SAT}?

Nobody knows!

We will give an example of language in NP^{SAT} but not known to be in NP.
Two Boolean formulas $\phi$ and $\psi$ over the variables $x_1,\ldots,x_n$ are equivalent if they have the same value on any assignment to the variables.

Are $x$ and $x \lor x$ equivalent? Yes

Are $x$ and $x \lor \neg x$ equivalent? No

Are $(x \lor \neg y) \land \neg(\neg x \land y)$ and $x \lor \neg y$ equivalent?

A Boolean formula is minimal if no smaller formula is equivalent to it.

**NON-MIN-FORMULA** = \{ $\phi$ | $\phi$ is not minimal \}
Theorem: \( \text{NON-MIN-FORMULA} \in \text{NP}^{\text{SAT}} \)

Proof:

\[
\text{EQUIV} = \{ (\phi, \psi) \mid \phi \text{ and } \psi \text{ are equivalent} \}
\]

\[
\text{EQUIV} \in \text{coNP} \ (\subseteq \text{P}^{\text{SAT}})
\]

So \( \text{EQUIV} \) can be decided in poly time with an oracle for \( \text{SAT} \).

\( \text{NP}^{\text{SAT}} \) machine for \( \text{NON-MIN-FORMULA} \):

Guess an equivalent smaller formula and test equivalency in poly time using oracle for \( \text{SAT} \).

\( \text{NON-MIN-FORMULA} \) is not known to be in \( \text{NP} \).
Theorem:

(1) An oracle B exists where $P^B = NP^B$

(2) An oracle A exists where $P^A \neq NP^A$

Proof of (1):

RECALL

$TQBF = \{ \phi \mid \phi \text{ is a true fully quantified Boolean formula} \}$

Theorem: $TQBF$ is PSPACE-complete
Theorem:

(1) An oracle B exists where $P^B = NP^B$

(2) An oracle A exists where $P^A \neq NP^A$

Proof of (1):

Let $B = TQBF$

Then:

$$NP^{TQBF} \subseteq NPSPACE \subseteq PSPACE \subseteq P^{TQBF}$$
Theorem:

(1) An oracle B exists where $P^B = NP^B$

(2) An oracle A exists where $P^A \neq NP^A$

Proof of (1):

Let $B = TQBF$

Then:

$$NP^{TQBF} \subseteq NPSPACE \subseteq PSPACE \subseteq P^{TQBF}$$

1. Can convert an NP machine with oracle TQBF to an equivalent NPSPACE machine that computes answers to queries about membership in TQBF
Theorem:

(1) An oracle B exists where $P^B = NP^B$

(2) An oracle A exists where $P^A \neq NP^A$

Proof of (1):

Let $B = TQBF$

Then:

1. $NP^{TQBF} \subseteq NPSPACE \subseteq PSPACE \subseteq P^{TQBF}$

2. By Savitch’s Theorem.

3. Since TQBF is PSPACE-complete
Theorem:

(1) An oracle B exists where \( P^B = NP^B \)

(2) An oracle A exists where \( P^A \neq NP^A \)

Proof of (2):

IDEA!

Construct A and a language \( L_A \in NP^A \) so that no polytime machine with oracle for A decides \( L_A \)

The construction will consider every polytime machine with oracle for A in turn and ensure that each fails to decide \( L_A \)
(2) An oracle $A$ exists where $P^A \neq NP^A$

For oracle $A$, define $L_A = \{ \text{strings } w \mid \exists x \in A \ [ |x| = |w| ] \}$

Notice that: $L_A \in NP^A$

We show how to construct $A$ so that $L_A \not\in P^A$

Let $M_1, M_2, \ldots$ be a list of all poly-time oracle TMs

We can assume that $M_i$ runs in time at most $n^i$ (WHY?)

We will construct $A$ in stages

Stage $i$ guarantees that $M_i^A$ doesn’t decide $L_A$
In STAGE $i$:

A finite number of strings have been assigned to $A$ so far.

Pick $n$ greater than the length of any string in $A$ so far and such that $n^i$ is smaller than $2^n$.

Run $M_i$ on $1^n$ and respond to its queries as follows:

If $M_i$ queries about a string $y$ whose status has already been determined, respond consistently.

Otherwise, respond NO and declare $y$ to be out of $A$.

Continue on until $M_i$ halts.
If $M_i$ accepts $1^n$ we declare all (remaining) strings of length $n$ are out of $A$, so $1^n \notin L_A$.

If $M_i$ rejects we find a string of length $n$ that $M_i$ has not queried and declare that string to be in $A$, so $1^n \in L_A$ (such string exists since $M_i$ runs for $n^i$ steps and there are $2^n$ strings and $n^i < 2^n$).

Hence $M_i$ doesn’t decide $L_A$.

PROCEED to Stage $i+1$.

At end, declare any string whose status remains undetermined by all stages is out of $A$.

No poly time machine with oracle $A$ decides $L_A$.

QED