Oracle Reductions
Oracle Turing Machines

Is \((M, w)\) in \(A_{TM}\)?

Yes

Finite State Control

Infinite Rewritable Tape

INPUT...
Oracle Turing Machines

An oracle Turing machine $M$ that can ask membership queries in a set $B \subseteq \Gamma^*$ on a special “oracle tape” [Formally, $M$ enters a special state $q_?$.]

The TM receives an answer to the query in one step [Formally, the transition function on $q_?$ is defined in terms of the entire oracle tape: if the string $y$ written on the oracle tape is in $B$, then state $q_?$ is changed to $q_{YES}$, otherwise $q_{NO}$.]

This notion makes sense even if $B$ is not decidable!
How to Think about Oracles?

A black-box subroutine. In terms of Turing Machine pseudocode:
An oracle Turing machine $M$ with oracle $B \subseteq \Gamma^*$ lets you include the following kind of branching instructions:

“if $(z \in B)$ then <do something> else <do something else>”

where $z$ is some string defined earlier in pseudocode.

By definition, the oracle TM can always check the condition $(z \in B)$ in one step.

This notion makes (mathematical) sense even if $B$ is not decidable.
Definition: A is recognizable with B if there is an oracle TM \( M \) with oracle \( B \) that recognizes \( A \)

Definition: A is decidable with B if there is an oracle TM \( M \) with oracle \( B \) that decides \( A \)

Language A “Turing-Reduces” to B

\( A \leq_T B \)
A\textsubscript{TM} is decidable with HALT\textsubscript{TM} (A\textsubscript{TM} \leq\textsubscript{T} HALT\textsubscript{TM})

We can decide if M accepts w using an ORACLE for the Halting Problem:

On input (M,w),

If (M,w) is in HALT\textsubscript{TM} then
    run M(w) and output its answer.
else REJECT.
HALT$_{TM}$ is decidable with $A_{TM}$ ($HALT_{TM} \leq_T A_{TM}$)

On input $(M,w)$, decide if $M$ halts on $w$ as follows:

1. If $(M,w)$ is in $A_{TM}$ then ACCEPT

2. Else, switch the accept and reject states of $M$ to get a machine $M'$. If $(M',w)$ is in $A_{TM}$ then ACCEPT

3. REJECT
Theorem: If $A \leq_m B$ then $A \leq_T B$

Proof (Sketch):

If $A \leq_m B$ then there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$, where for every $w$,

$$w \in A \iff f(w) \in B$$

To decide $A$ on the string $w$, just compute $f(w)$ and “call the oracle” for $B$

Theorem: $\neg \text{HALT}_{TM} \leq_T \text{HALT}_{TM}$

Theorem: $\neg \text{HALT}_{TM} \not\leq_m \text{HALT}_{TM}$

Why?
Limitations on Oracle TMs!

The following problem cannot be decided by any TM with an oracle for the Halting Problem:

\[ \text{SUPERHALT} = \{ (M,x) \mid M, \text{with an oracle for the Halting Problem, halts on } x \} \]

We can use the proof by diagonalization!
Assume \( H \) (with \( \text{HALT} \) oracle) decides \( \text{SUPERHALT} \)

Define \( D(X) := \text{"if } H(X,X) \text{ (with } \text{HALT} \text{ oracle) accepts then LOOP, else ACCEPT.} \) (D uses a \( \text{HALT} \) oracle to simulate H)

But \( D(D) \) halts \( \iff \) \( H(D,D) \) accepts \( \iff \) \( D(D) \) loops...

(by assumption) (by def of D)
```
Limits on Oracle TMs

“Theorem” There is an infinite hierarchy of unsolvable problems!

Given ANY oracle $O$, there is always a harder problem that cannot be decided with that oracle $O$

SUPERHALT$^0 = \text{HALT} = \{(M,x) \mid M \text{ halts on } x\}.$

SUPERHALT$^1 = \{(M,x) \mid M, \text{ with an oracle for } \text{HALT}_{TM}, \text{ halts on } x\}$

SUPERHALT$^n = \{(M,x) \mid M, \text{ with an oracle for } \text{SUPERHALT}^{n-1}, \text{ halts on } x\}$
```
Parting thoughts:
Turing reductions, more powerful than mapping reductions. An oracle that is always wrong is just as useful. For every hard problem, there is a harder one.