- Suppose Halt can be solved by using an alien's machine! Does it mean that every problem can be solved? How do we formalize such questions?

Oracles (& Relativizing proofs)

Defn: We call a TM M an *oracle TM using a language O* if M has

- three special states q_{query}, q_{yes}, q_{no}
- a special oracle-tape,

such that when M enters q_{query} with a string y on the oracle-tape, in the next step it is in q_{yes} (resp. q_{no}) if $y \in O$ (resp. $y \notin O$).

Defn: • $P^O := \{L \mid L$ has a poly-time oracle TM using $O\}$.

• $NP^O := \{L \mid L$ has a poly-time oracle NDTM using $O\}$.

Proposition: (1) $\overline{O} \in P^O$.

(2) If $O \in P$ then $P^O = P$.

(3) Let $\text{Expcom} := \{(M, x, 1^n) \mid$ TM M accepts x in $\leq 2^n$ steps$\}$. Then,

$$P^{\text{Expcom}} = EXP = NP^{\text{Expcom}}.$$

Proof:

(1) Negate the answer of O.

(2) Ignore the oracle-tape; instead use the poly-time TM.

(3) Show the easy consequences,

$$EXP \subseteq P^{\text{Expcom}} \subseteq NP^{\text{Expcom}} \subseteq EXP$$

□

Defn: A proof about complexity classes, $C_1 = C_2$ (resp. $C_1 \neq C_2$), is said to be <u>relativizing</u> if $\forall O$, $C_1^O = C_2^O$ (resp. $C_1^O \neq C_2^O$) also follows.

- $C_1 = C_2$ may not mean $C_1^O = C_2^O$!

▷ Diagonalization proofs till now are all relativizing.

Pf: Properties (1) & (2) before. $\square$

<u>$P \stackrel{?}{=} NP$ requires a non-relativizing proof</u>

Theorem (Baker, Gill, Solovay, 1975): $\exists$ languages A & B s.t. $P^A = NP^A$ & $P^B \neq NP^B$.

Proof:
- We have already seen $A := \text{Expcom}$.
- Now we design B via diagonalization!
- For any B, the related unary language $U_B := \{1^n \mid \exists x \in B, |x| = n\} \in NP^B$.

- So, we design a B s.t. $U_B \notin P^B$ by recursion.

- Let $\{M_i \mid i$ is an oracle TM description$\}$ be an enumeration of oracle TMs in the increasing order of i.

- We incrementally construct B. In the i-th stage we ensure that M_i^B <u>does not</u> decide U_B in (2^n-1) steps.

  Initially, $B = \phi$.

- In the i-th stage:

  We have declared only a finite number of strings in/out of B. Choose n_i to be larger than the length of those strings.

  Run M_i^B on 1^{n_i} for $(2^{n_i}-1)$ steps as:

(1) If M_i queries B on strings whose status is <u>undetermined</u>, we declare them <u>not</u> in B.

(2) If M_i queries B on strings whose status is determined, then be consistent.

(3) If, eventually in $(2^{n_i}-1)$ steps, M_i^B accepts 1^{n_i}, then declare all strings of length n_i out of B.

Else, we put a string of length n_i in B that has not been queried by M_i. (There exists such a string!)

▷ At the i-th stage, M_i^B does not decide U_B in (2^n-1) steps.

• So, if $U_B \in P^B$ we can consider a large enough j s.t. M_j^B decides U_B in poly-time. This gives a contradiction.

$\Rightarrow U_B \notin P^B$

$\Rightarrow \quad P^B \neq NP^B$.

$\square$