Today: Talk about oracle Thm. and:

there is an oracle A s.t. $P^A = NP^A$

When I was young, I have proof that

$$x^n + y^n = z^n$$

has no solutions for $x, y, z > 0$, $n \geq 3$.

But: My proof actually "worked" for $x, y, z \in \mathbb{R}$

So ...

$x, y, z \in \mathbb{N}$ big difference

conjecture
(now theorem)

Say you want to settle P vs. NP

§9.1 – Classical Thms in Complexity:

e.g. $\mathrm{TIME}(n^k) \neq \mathrm{TIME}(n^{k+0.0001})$

for any $k > 1$, k real.

or not

Problem: $\mathrm{TIME}^{A}(n^k) \neq \mathrm{TIME}^{A}(n^{k+0.0001})$

any oracle.

§9.2 There is an oracle, A, s.t. $P^A \neq NP^A$

" " " " B s.t. $P^B = NP^B$

Homework: Say that we have a subroutine that our T.m. can run to decide SAT. Then given a satisfiable Boolean formula $f = f(x_1, \dots, x_n)$ there is a poly time alg (that can call the subroutine some poly number of times) that gives

$y_1, y_2, \dots, y_n$ = true/false s.t.

$$f(y_1, \dots, y_n) = \text{true}$$

Idea: $f(\text{true}, x_2, \dots, x_n)$ is satisfiable ← run SAT

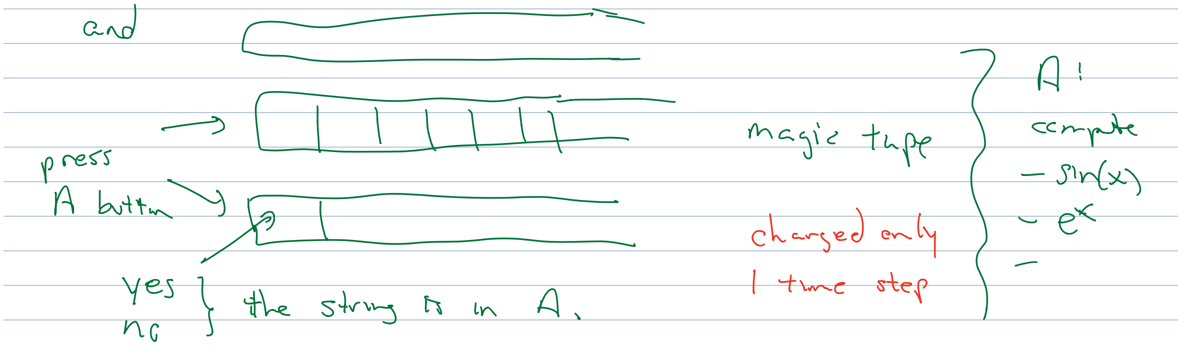
or $f(\text{false}, x_2, \dots, x_n)$ " " ← run SAT

So find y_1 = true, false, keep going ...

Given a language A over some alphabet, a T.m. with oracle A: a Turing machine, M, with an addition "membership in A" subroutine

Run normal T.m (maybe non-deterministic or not)

and



Example: The problem: given $\langle f \rangle$, f Boolean formula, either output: (1) f is not in SAT, or (2) give an assignment of variables true/false s.t. f is true

is in P^{SAT} = poly time if you can call an oracle to decide SAT.

HALT is an interesting oracle.

If you think you have a proof that $P \neq NP$ make sure it doesn't prove $P^A \neq NP^A$

$$P^A = \bigcup_{k=1,2,\dots} TIME^A(n^k)$$

[$TIME^A(f(n))$ = { languages decidable in time $O(f(n))$ with oracle A }]

Why? Let A be PSPACE-complete language.

Claim: $NPSPACE \subset PSPACE \subset P^A \subset NP^A \subset NPSPACE$

Savitch's Thm

By def of PSPACE-complete

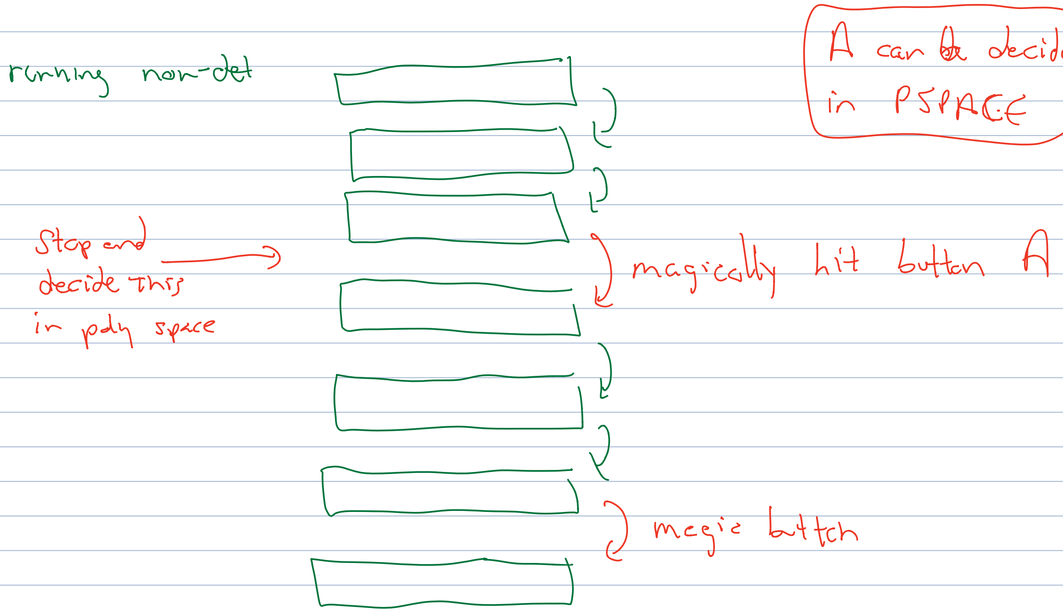
this can handle non-det poly space, and it can take of calls to A in poly space

($PSPACE^A = P^A$) (cheap shot)

Proves: $P^A = NP^A$

NP + button decide some problem in PSPACE

A can be decided in PSPACE


running non-det
Stop and decide this in poly space
magically hit button A
magic button