The rest of the course:

Talk a lot about → How not to solve P vs. NP ← Baker-Gill-Solovay Thm (Thm 9.20 in [Sip])

→ — " possibly " " " " " " ↳ Circuit Complexity (§9.3)

Cook-Levin Thm again

Ch 8: SPACE complexity : Highlight Savitch's Thm.

(Ch 7: TIME complexity)

Baker-Gill-Solovay: Thm 9.20 in [Sip]: (with oracle A)

(1) There is a language/oracle A s.t. $P^A \neq NP^A$

(2) " " " " " B " $P^B = NP^B$

Upshot: If you solve P vs NP, your proof must have some fact that is not true when TM's are given an oracle.

- E.g. - Set of recognized languages by a TM with a fixed oracle A is countable (Doesn't depend on A)
  - Blah blah _ _ blah TM with oracle A blah blah _ _ (Doesn't depend on A)
  - Other theorems _ _

Another view point:

Fermat's "Last Theorem"

For any integer $n \geq 3$ there are no solutions

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

with $x, y, z > 0$ and x, y, z integers.

and x, y, z real

$FLT^{x,y,z \in \text{integers}} = FLT^{\mathbb{Z}}, FLT^{\mathbb{R}}$

$FLT^{\mathbb{R}}$ is false, $FLT^{\mathbb{Z}}$ = big deal.

I can prove $FLT^{\mathbb{R}}$! ☹

" " " $FLT^{\mathbb{Z}}$; Somewhere in the proof, we must use we are in $\mathbb{Z}$, not $\mathbb{R}$

FLT "relative to $\mathbb{Z}$"
FLT "relative to $\mathbb{R}$" } "relativization"

=

Baker-Gill-Solovay : $P^A = NP^A$ relative to some A

$P^B \neq NP^B$ " " " B

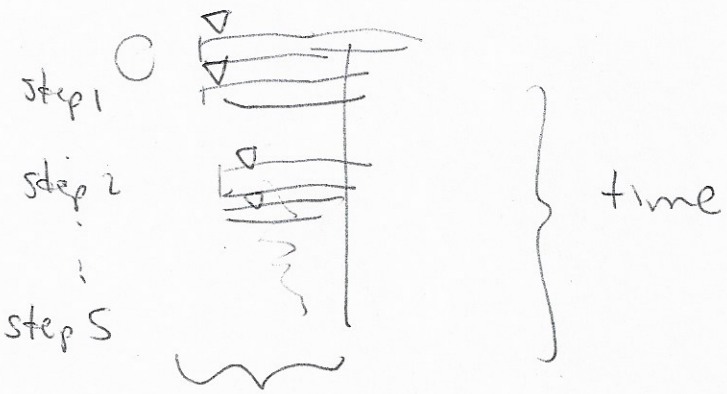
=

More on NP-completeness

Ch 7: Time Complexity

Ch 8: Space "

Recall: We say that TM decides L in Space $f(n)$ if on input $w \in \Sigma^*$, TM uses "space" (i.e. # cells) at most $f(n)$, $n = |w|$.



Space = furthest cell tape heads move to R.

Same def for a multi-tape T.Machine, where space is maximum of tape head movements to the R.

$$SPACE(f(n)) = \{ \text{Languages } L \text{ decided by a T.m. within space } O(f(n)) \}$$

$$POLY\text{-}SPACE = PSPACE = \bigcup_{k=1,2,\ldots} SPACE(n^k)$$ ← Space analogue of P = poly time

$$\text{Nondet Poly-SPACE} = NPSPACE = \bigcup_{k=1,\ldots} NSPACE(n^k)$$ ← Analogue of NP

Ch 8: $PSPACE = NPSPACE$ (Savitch's Theorem)

NP-completeness

NP: — Languages that can be "verified" in poly time

— " " " " " " non-deterministically accepted by a non-det TM (non-det selection = guess = choose)

[Sip] →

→ — Languages where there is a poly-time length proof that an input is in the language

Next: there is a language that clearly is NP-complete ...