

- 8.1 - Savitch's Thm
- 8.2 - PSPACE, NPSPACE (they are equal) } Ch 8
- 8.3 - We avoid messy details ← "SNEAKY"

Jump to Thm 9.23 Baker-Gill-Solovay

Start with a Sneaky NP-complete language:

Cook-Levin: SAT, 3SAT, 3COLOUR, ... NP-complete
of interest in combinatorial optimization

~~SNEAKY-NTM~~

$$= \left\{ \langle M, w, t \rangle \mid \begin{array}{l} \uparrow \\ \underbrace{\|w\|}_{\text{length } t} \end{array} \right. \left. \begin{array}{l} M \text{ is a non-deterministic TM} \\ \text{that accepts input } w \text{ (along one of} \\ \text{its branches)} \\ \text{in time } t \end{array} \right\}$$

(Practically uninteresting, but to prove that it is NP-complete is easy)

① SNEAKY-NTM \in NP . (usually easy) (more difficult)

② $L \in$ NP, $L \leq_{\text{poly time}}$ SNEAKY-NTM . (usually difficult) (fairly easy)

Prove ②: Say that $L \in$ NP, recognized by a NTM, M , in time n^3 .

Define $f: \Sigma_L^* \rightarrow \Sigma_{\text{SNEAKY-NTM}}^*$ s.t. (2)

- f is poly time

- $w \in L$ iff $f(w) \in \text{SNEAKY-NTM} \quad \forall w \in \Sigma_L^*$

Algorithm for f :

Given w : Now write

- Description of M , w , $\underbrace{1111\dots 1}_{|w|^3}$

$\underbrace{\hspace{10em}}_{\text{constant, indep of } w}$ $\underbrace{\hspace{2em}}_{|w|}$ $\underbrace{\hspace{10em}}_{|w|^3}$ $\leftarrow |w|^3$ (5)

$\leftarrow \text{LENGTH} \uparrow$

This f takes time roughly $\text{const} + |w| + |w|^3$

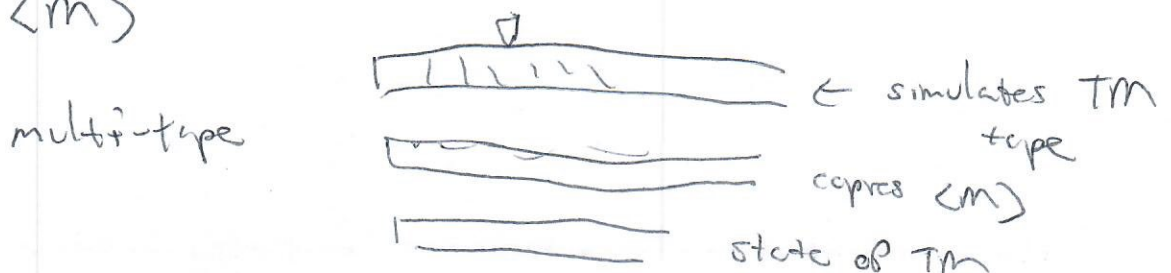
So f is poly-time (depending on wanting to run M for $|w|^3$ steps)

Prove (1): Why is $\text{SNEAKY-NTM} \in \text{NP}$:

$\langle M, w, \underbrace{111\dots 1}_t \rangle$ given: can you build

a non-det TM that recognizes if w is accepted by M in t steps?

- If we are talking about non-determinism, a universal TM would work: claim: each TM step can be simulated size of $\langle M \rangle$



Claim: This will take $\leq O(\text{length } M + \text{Const})$ (3)
 steps per simulated step; total time = $O(t \cdot \text{length } M)$
 $\leq O(\text{input length}^2)$.

- If we are talking about non-det TM, the same thing works...
 (we need to non-deterministically select the computation.)

We say L is PSPACE-complete if

① $L \in \text{PSPACE}$, and

② $L' \in \text{PSPACE}$, then $L' \leq_{\text{poly time}} L$.

$\left\{ \begin{array}{l} \text{PSPACE} = \text{poly space} \\ = \bigcup_{k \in \mathbb{N}} \text{SPACE}(nk) \end{array} \right.$

Claim:

$\text{SINGLE-TAPE-PSPACE} = \left\{ \langle M, w, 1^S \rangle \mid \begin{array}{l} M \text{ is a TM that} \\ \text{accepts } w \text{ within} \\ \text{space } S \end{array} \right\}$

then SINGLE-TAPE-PSPACE is PSPACE-complete.

PSPACE

