NP $\equiv$ Polynomial NTM I

- Polynomial verifier $\Leftrightarrow$ polynomial NTM
- Idea:

  "$\Rightarrow$" NTM by guessing certificate

  "$\Leftarrow$" using NTM's accepting branch as certificate
- Proof:
- "$\Rightarrow$": now we have a verifier V in time n^k

NP $\equiv$ Polynomial NTM II

Recall the definition below

$$A = \{w \mid V \text{ accepts } \langle w, c \rangle \text{ for some strings } c\}$$

We have

$$|c| \leq n^k$$

because to handle $\langle w, c \rangle$ in n^k, $|c|$ should be bounded by n^k

- Use an NTM to
  1. nondeterministically select c
  2. run V on $\langle w, c \rangle$

NP $\equiv$ Polynomial NTM III

That is, run c in parallel and each is polynomial

- We have that for any $w \in A$, the NTM accepts it in polynomial time
- "$\Leftarrow$": now w is accepted by a polynomial NTM

  Let c be any accepting branch

  Note that for polynomial NTM, each branch is polynomial
- Then we have a verifier V that handles input $\langle w, c \rangle$ in polynomial time
- Note: the definition of V requires only "some c."
- So finding one is sufficient

SUBSET-SUM I

- Given $x_1, \ldots, x_k$ and t, is sum of a subset $= t$?
- Formally

$$\{\langle s, t\rangle \mid s = \{x_1, \ldots, x_k\} \text{ and } \exists \\ \{y_1, \ldots, y_l\} \subset \{x_1, \ldots, x_k\} \text{ such that } \sum y_i = t\}$$

- Example

$$\langle\{4, 11, 16, 21, 27\}, 25\rangle \text{ OK as } 4 + 21 = 25$$

SUBSET-SUM II

- Note: allow repetition here

$$\langle \{4, 11, 11, 16, 21, 27\}, 25 \rangle$$

- We prove that this problem is NP
- Idea: the subset is the certificate.
- Consider any input

$$\langle \langle s, t \rangle, c \rangle$$

We

1. check if $\sum c_i = t$

SUBSET-SUM III

2. check if all $c_i \in s$

   If both pass, accept; otherwise, reject

- Here

$$\text{length of } c < \text{ length of } s$$

- The verification can be done in polynomial time

P vs. NP I

- Roughly

  P: problems decided quickly

  NP: problems verified quickly
- Question: is $P = NP$?

  This is one of the greatest unsolved problems
- Most believe $P \neq NP$

NP-completeness I

- It has been shown that some problems in NP are related
- For certain NP problems:

  If $\exists$ a polynomial algorithm for one NP $\Rightarrow$ P $=$ NP
- These problems are called NP-complete problems
- They are useful to study the issue of P versus NP
- To prove P $\neq$ NP: only need to focus on NP-complete problems
- To prove P$=$NP: need only polynomial algorithms for an NP-complete problem