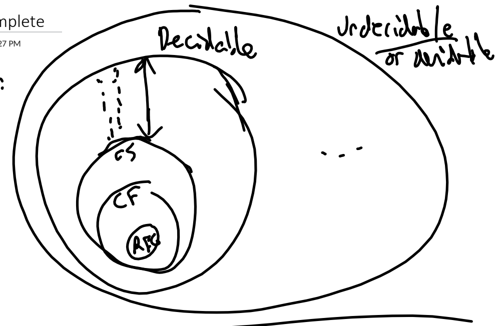
NP and NP-complete

Thursday, April 23, 2020 3:27 PM

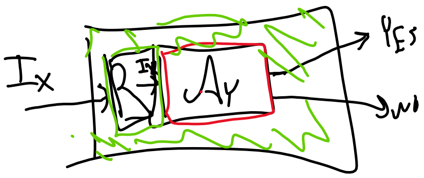
Before:



Recall: poly time reductions.

$X \le_p Y$ polytime alg solution $Y \Rightarrow$ polytime solution to X

e.g. CLIQUE



- R runs in polytime
- ~~go~~ YES instances of I_X become YES instances of Y.
   - Same for NO
- SAT $\le_p$ 3SAT SAT $\approx_p$ 3SAT.
  3SAT $\le_p$ SAT
- MIS $\approx_p$ CLIQUE

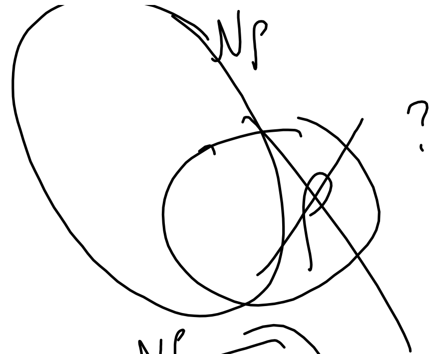
P vs. NP: ~~"NP means not polynomial time"~~

P: decision problem solved in poly time.

NP: solution can be checked in polytime.

Some decision problems have efficiently checkable "proofs" or "certificates" for YES answers

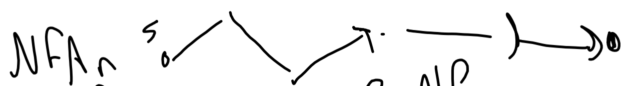
- COMPOSITES:
  given number X, is this composite?
  nontrivial factor $p \mid X$, where $1 < p < X$
  proof: p the ~~factor~~ where ↑
  check: Divide X by p and check remainder.
- IS: is there a k ind set?
  proof: ind set itself, S
  check: check $|S| \ge k$,
  and $\forall u, v \in S$ $u \not\leftrightarrow v \in G$.



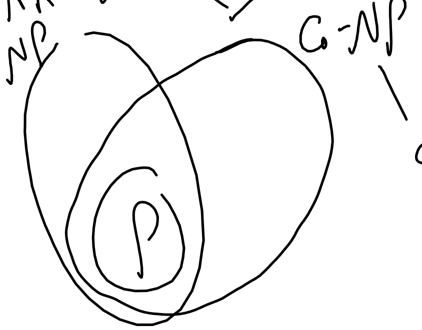
suppose
$X \in P$, ...
proof: ∈
check:
$X \in P$ ∃ solution A_X
$A_X(I_X) \to$ YES/NO

$P \subseteq NP$

NP: nondeterministic polynomial time



$x \in L$
implies
∃ path from s to an accept state in n.



co-NP : "NO" instances can be checked in poly time.

EXP: exponential time solvable.
$\exists A_X$ that solves X in $O(2^{|X|})$

$P \le EXP$.

PRIME ⇒ Co-NP.

$NP \subseteq EXP$?

In other words, any problem X that YES instances can be checked in poly time, then it can be solved (both YES & NO) in exp-time.

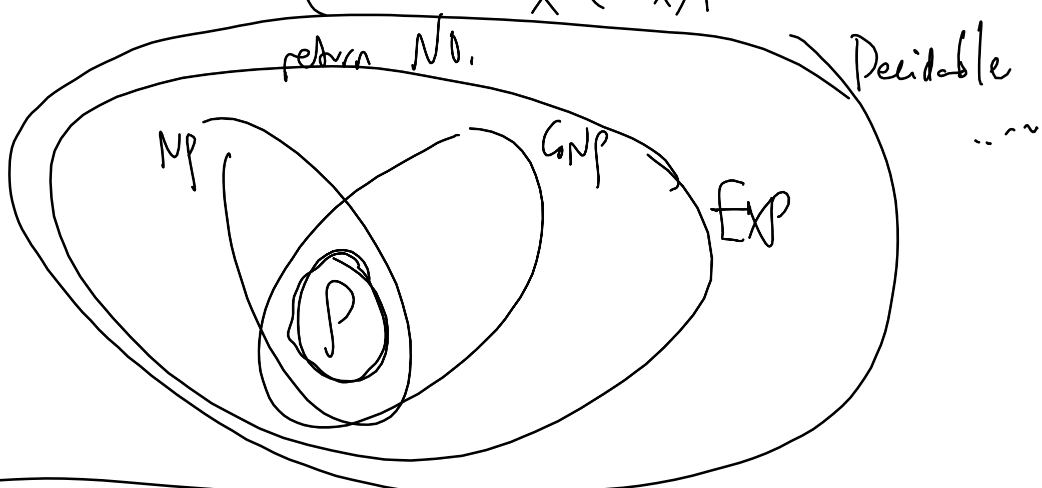
$X \in NP$ means,
$\exists\, C_X(I_X \text{ proof})$,

C_X runs polytime,
$\exists\, p(n)$, $\exists$ proof, st $C_X(I_X, \text{proof}) = 1 \iff I_X$ is a YES instance

A_X solves in Exp. time:

$A_X(I_X)$:
  $i = 0$
  for i counting from 0 to $p(n)$:
    foreach proofs that are i bits long:
      if $C_X(I_X, \text{proof}) = 1$, return YES
  return NO.

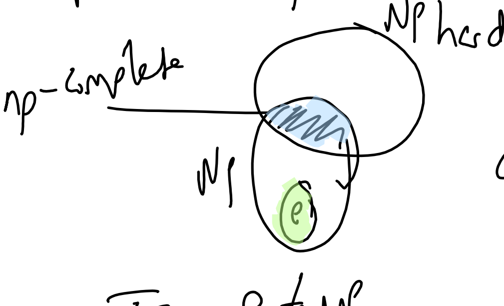
Exp time



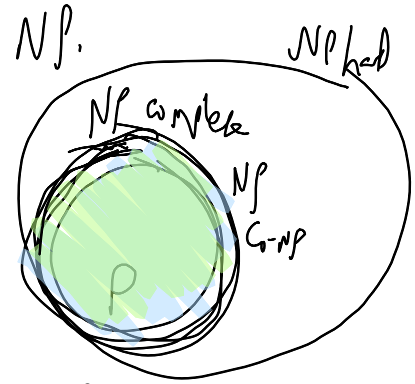
NP-hard, NP-complete

- Y is NP-hard iff it's at least as hard as any in NP.
  solution to Y can be used to solve X.
  equiv. $\forall X \in NP,\ X \le_p Y$
  equiv. a solution to $Y \Rightarrow$ solution to any NP problem.

- NP-complete: NP-hard and also in NP.



could be



If $P \neq NP$ $P = NP$

$P = NP$
"collapsed hierarchy"

$P \neq NP \Rightarrow$ there are problems that are hard to solve but easy to check.

$P = NP \Rightarrow$ all easy to check problems are also easy to solve.

eg. all composite numbers can be factored
$\Rightarrow$ no cryptography would work.

Usually we assume $P \neq NP$.

Proving problems are NP-hard & NP-complete.

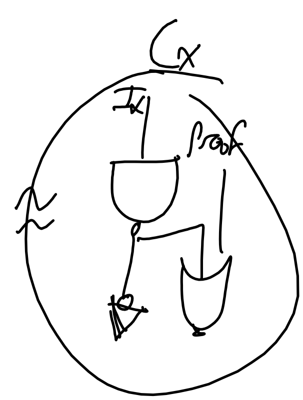
- To show $\in NP$, prove YES instances can be checked.
- To show NP-hard:

1. Cook-Levin Theorem. Direct way.

$\forall X \in NP$, we can encode X into SAT.

$X \in NP \Rightarrow C_X(I_X, \text{proof})$

CNF clause1 $(X_1 \vee \neg X_2 \vee X_3)$
$\wedge$ clause2
$\wedge$
$\wedge$
$\wedge$
$\wedge$



$\boxed{2.}$ Show that some NP hard problem reduces to Y.

e.g. $3SAT \leq_p Y$.

$\Rightarrow \forall X, X \leq_p SAT \leq_p 3SAT \leq_p Y$

by 1. by yesterday

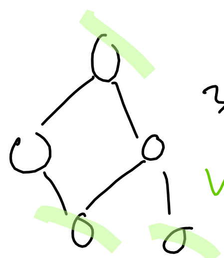
Skill: prove a problem is NP-hard by reducing from a known NP-hard problem.

Ex. $3SAT \le_p k\text{-}IS$. each clause has 3 literals.

$\wedge (x_1 \vee \neg x_2 \vee \neg x_4)$

$\wedge ($

$($



3-is? ✓

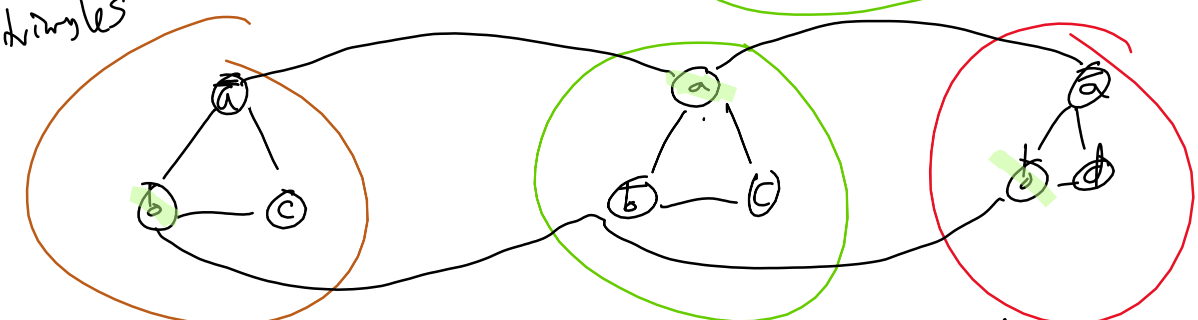
$\overset{?}{\longleftrightarrow}$

show "encode any instances of 3SAT into k-IS"

k clauses in

e.g. $(\bar{a} \vee b \vee c) \wedge (a \vee \neg b \vee c) \wedge (\bar{a} \vee b \vee d)$

K triangles



- A ind set select at most 1 from each triangle.
- A k-IS must select exactly 1

→ I. Show transformation is polytime.

→ II. Good instances of 3SAT ⟹ good instances of k-IS.

→ III. Good instances of k-IS ⟹ good instances of 3SAT.

or Bad instances of 3SAT ⟹ bad instances of k-IS.