Algorithms

NP-Hardness (cont)

Recap

- Last 2 HWs up
  - one due next Monday
  - final one due last Wed. of classes
  - final worksheet (ungraded) will be up after break

- Final exam: Monday of finals week

Review sometime that Friday

P, NP, + co-NP

$P \subseteq NP$

Consider only decision problems:
so Yes/No output

P: Set of decision problems that can be solved in polynomial time.

Ex: - Is x in the ~~list~~? $O(n)$ or $O(\log n)$
- Is there a cut in G of size 100? ⟶ F-F: $O(VE)$

NP: Non-determinate poly time

Set of problems such that, if the answer is yes & you hand me proof, I can verify/check in polynomial time.

Ex: CircuitSAT: hand me inputs I can check in $O(n+m)$ time

Co-NP: If answer is no, I can check that in poly time.

Dfn: NP-Hard

X is NP-Hard

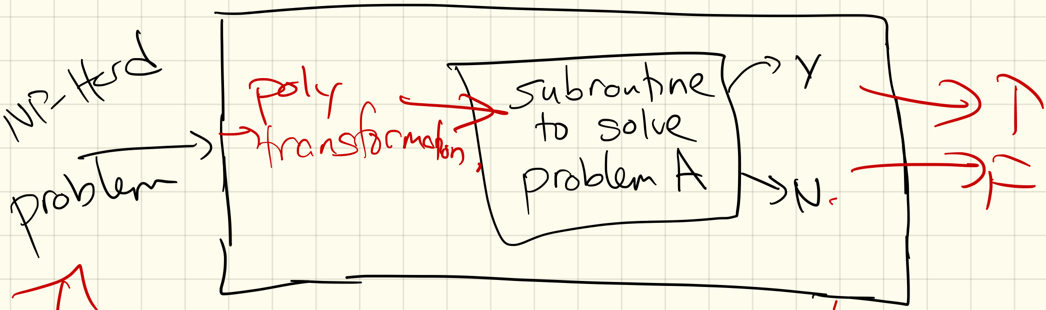
$\iff$

If X could be solved in polynomial time, then $P = NP$.

So if <u>any</u> NP-Hard problem could be solved in polynomial time, then all of NP could be.

To prove NP-Hardness of A:

Reduce a known NP-Hard problem to A.



circuitSAT

If tranformation + subroutine for A is poly time, then could solve CircuitSAT in that time.

So far:

1. CircuitSAT : Cook-Levine
   (only direct proof)

2. SAT :
   Take circuit + build formula
   circuit is true $\Longleftrightarrow$ form. is SAT

3. 3SAT : 3CNF formula:

$$(x \vee y \vee z) \wedge (\_\ \_\ \_) \wedge (\quad) \wedge (\quad)$$

Take circuitSAT + reduce to 3SAT

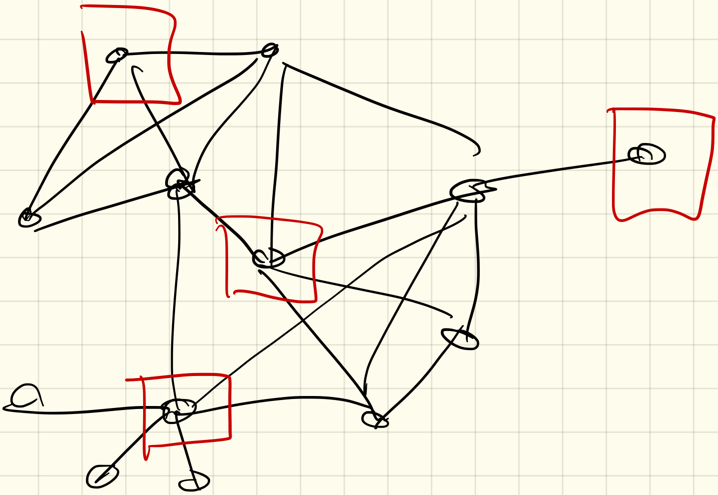
circuit has inputs yield T $\Longleftrightarrow$ 3CNF formula is satisfiable...

Today: More!
(plus general pattern)

Next Problem:

Independent Set:

A set of vertices in a graph with no edges between them:



decision version: input: G & k

Does G have ind set of size k?

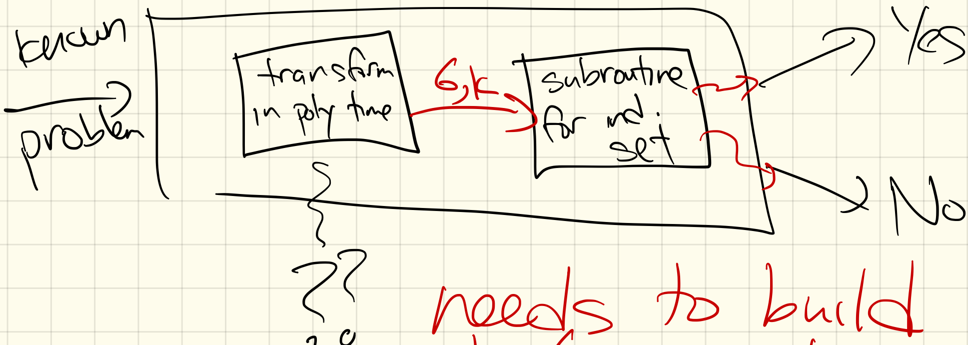
output: T/F

(wait - didn't we see this already!?)

solved in paths or trees

Challenge: No booleans!

But reduction needs to take known NP-Hard problem + build a graph:



needs to build graph G + pick a number k

We'll use 3SAT

(but stop and marvel a bit first...)

Reduction:

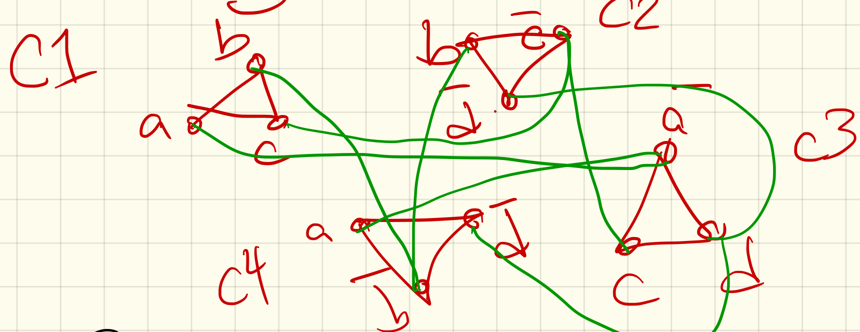
Input is 3CNF boolean formula: n variables & m clauses

$$(a \vee b \vee c) \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{b} \vee \bar{d})$$

$$\underbrace{(a \vee b \vee c)}_{C1} \wedge \underbrace{(b \vee \bar{c} \vee \bar{d})}_{C2} \wedge \underbrace{(\bar{a} \vee c \vee d)}_{C3} \wedge \underbrace{(a \vee \bar{b} \vee \bar{d})}_{C4}$$

(1) Make a vertex for each literal in <u>each</u> clause

3m vertices:

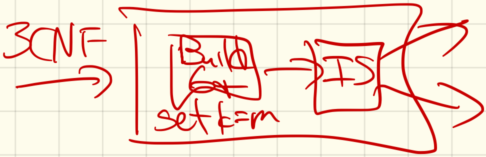


(2) Connect two vertices if:

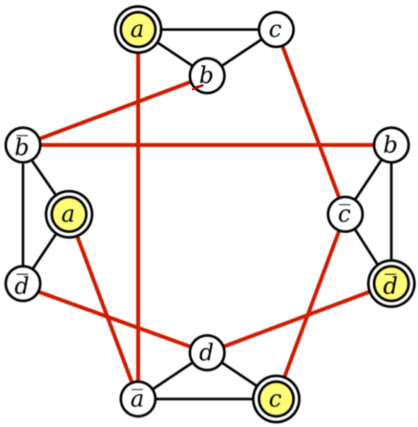
- they are in same clause
- they are a variable & its <u>inverse</u>

For loop (or two) ⟹ $O((n+m)^2)$ time

Example:



$$(a \lor b \lor c) \land (b \lor \bar{c} \lor \bar{d}) \land (\bar{a} \lor c \lor d) \land (a \lor \bar{b} \lor \bar{d})$$



A graph derived from a 3CNF formula, and an independent set of size 4.

Input to IndSet subroutine

- this graph
- K = m

Claim:

formula is Satisfiable $\Longleftrightarrow$ G has independent set of size m.

Proof:

$\Rightarrow$: Suppose formula is satisfiable.

Set of inputs $x_1 \ldots x_n$ s.t. whole thing is true.

Since 3CNF, I know at least 1 variable in each clause must be true.

$$\underbrace{(\underset{F}{x_i} \vee \underset{F}{x_j} \vee \underset{F}{\overline{x_k}})}_{F} \wedge \ \_\_\_\_\_\_\_\_\_\_\_\_$$

Build ind. set in G:

Pick vertex corr. to a true value in clause.

This forms an IS in G.

Pf (cont)

$\Leftarrow$: Spps G has ind set I' of size m.

Since I built G, I know each clause made a Δ.

So ind set gets at most 1 vertex per clause.

So if I pick T for all values in IS, I get 1 per clause.

Why valid?

A var & its negation can't both be in IS (b/c I put an edge!)

So:

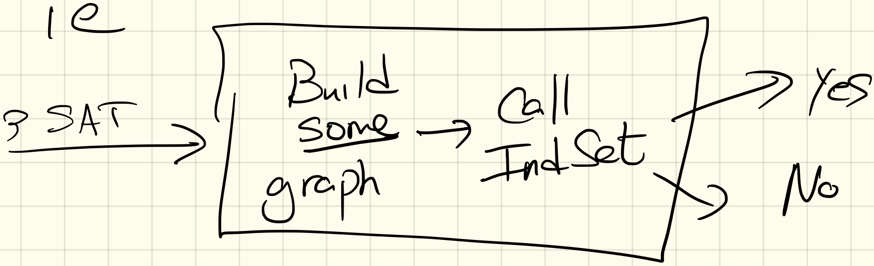
$$\begin{tikzcd}[column sep=large] \text{3CNF formula with } \cancel{k}\ m \text{ clauses} \arrow[r, "\cancel{O(n)}\ O((m+n)^2)"] & \text{graph with } \cancel{3k}\ 3m \text{ nodes} \arrow[d, Rightarrow, "\textsc{MaxIndSet}"] \\ & \text{\sout{maximum independent set} size} \\ \textsc{True} \text{ or } \textsc{False} & \text{ind set of size } m \arrow[l, "O(1)"'] \end{tikzcd}$$

$$T_{3\text{SAT}}(n) \le O(n) + T_{\textsc{MaxIndSet}}(O(n)) \quad \Longrightarrow \quad T_{\textsc{MaxIndSet}}(n) \ge T_{3\text{SAT}}(\Omega(n)) - O(n)$$

The Pattern:

1) Find an NP-Hard problem, & solve it using unknown problem as a subroutine

ie



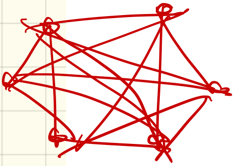
Proof:

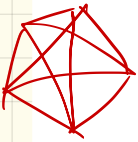
Need if & only if!

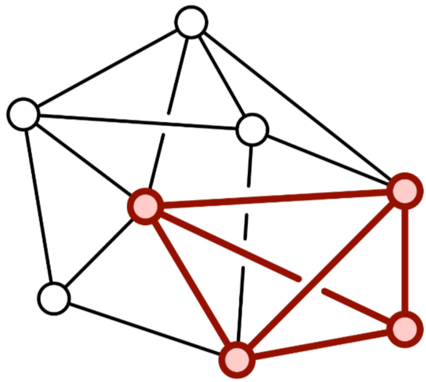
(ie might be some wierd indep. set that doesn't make a SAT)

Next one: Clique

A clique in a graph is a subgraph which is complete – all possible edges are present.







A graph with maximum clique size 4.

Try $\binom{n}{k}$ possible cliques

How could we check if G has a clique of size k?

Decision version: Does G have a clique of size k?

Input:

Output:

This is NP-Complete:

1 In NP. Why?

2 NP-Hard:

What should we reduce to k-Clique?

So:

$$
\begin{tikzcd}
\boxed{\text{graph } G} \arrow[r, "O(n)"] & \boxed{\text{complement graph } \overline{G}} \arrow[d, Rightarrow, "\textsc{MaxClique}"] \\
\boxed{\text{largest independent set}} & \boxed{\text{largest clique}} \arrow[l, "\text{trivial}"']
\end{tikzcd}
$$

Next: Vertex Cover:

A set of vertices which touches every edge in G.

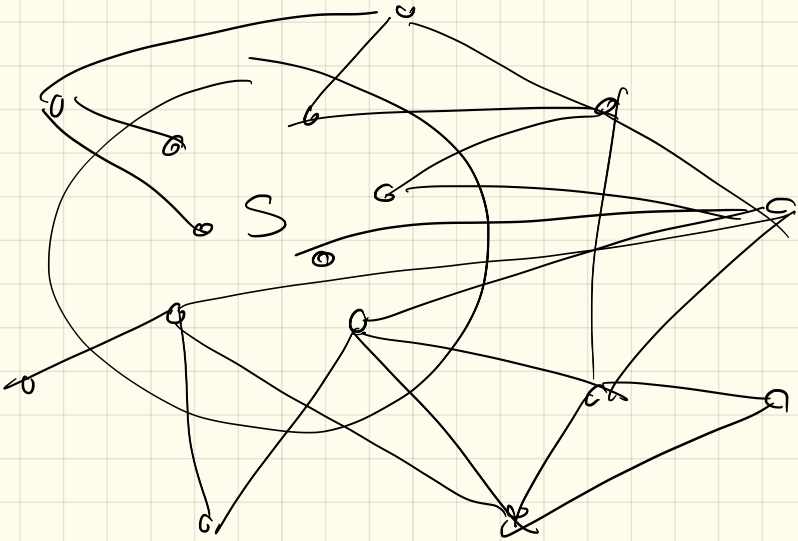
k-Vertex cover (decision version):

In NP:

NP-Hardness: reduce what?

(propably clique or ind set!)

Key: If S is independent set, what is $V-S$?



So simple reduction!

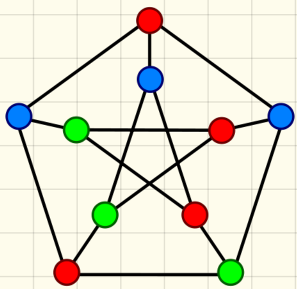
Given G & k to indep. set, ask if $\exists$ vertex cover of size $n-k$.

$$\begin{tikzcd}
\boxed{\text{graph } G=(V,E)} \arrow[r, "\text{trivial}"] & \boxed{\text{graph } G=(V,E)} \arrow[d, Rightarrow, "\textsc{MinVertexCover}"] \\
\boxed{\text{largest independent set } V\setminus S} & \boxed{\text{smallest vertex cover } S} \arrow[l, "O(n)"']
\end{tikzcd}$$

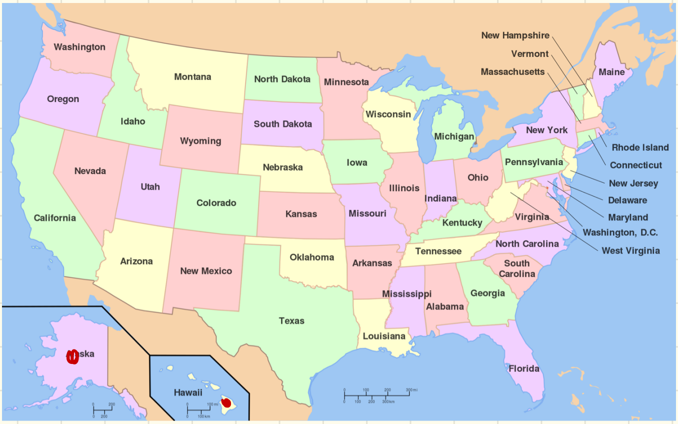
Next: Graph Coloring

A **k-coloring** of a graph G is a map: $c : V \to \{1, \ldots, k\}$ that assigns one of k "colors" to each vertex so that every edge has 2 different colors at its endpoints.

Goal: Use few colors



Aside: this is famous!
Ever heard of map coloring?



Famous theorem:

Next time:

More involved reduction...