CS3100

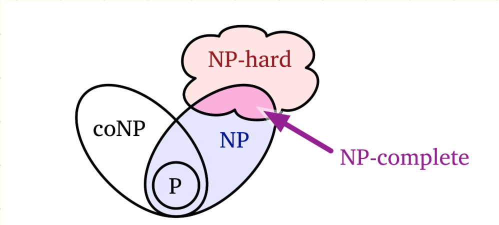
NP-Hardness &
(more) reductions

Announcements

- Next HW - due on paper next Wednesday
- HW after will be oral grading → Friday Nov 17
- Office hours Friday:
  10-11 am
  12-1 pm
- No class Nov. 27
  look for reading assignment

Recall:

A problem is NP-Complete if it is both:

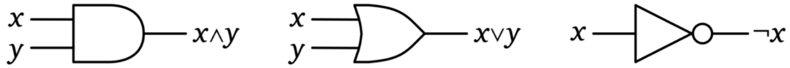


- In NP: A "yes" answer can be checked in polynomial time
- NP-Hard:

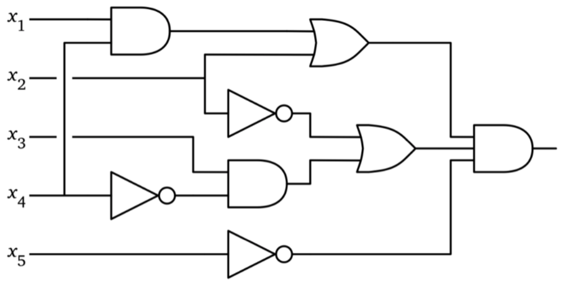
  via reductions

Thm: (Cook-Levine)

Circuit SAT is NP-Complete



An AND gate, an OR gate, and a NOT gate.

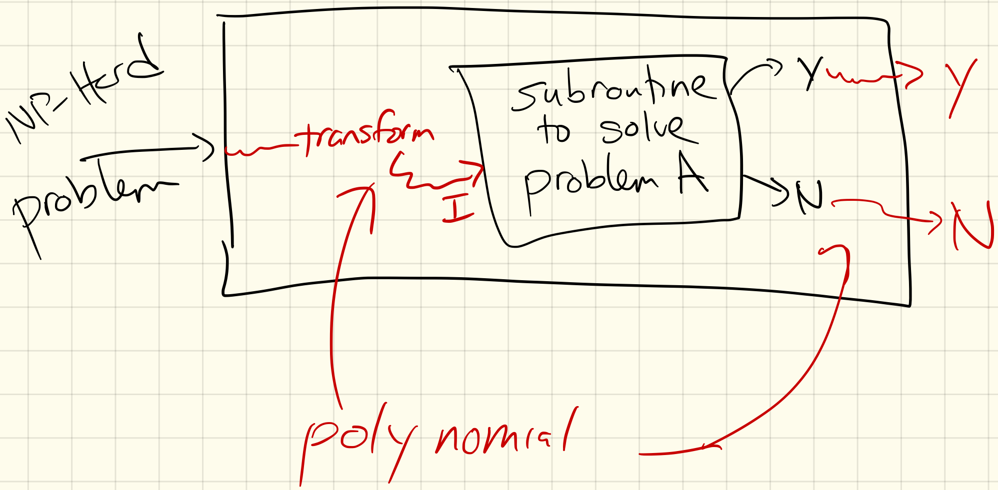


A boolean circuit. inputs enter from the left, and the output leaves to the right.

"Proof": We can turn any Turing machine into a circuit.

To prove any <u>other</u> problem A is NP-Hard, we'll use a reduction:

Reduce a <u>known NP-Hard problem</u> to A.



Thm: SAT is NP-Complete

↳ logical sentences

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

pf:

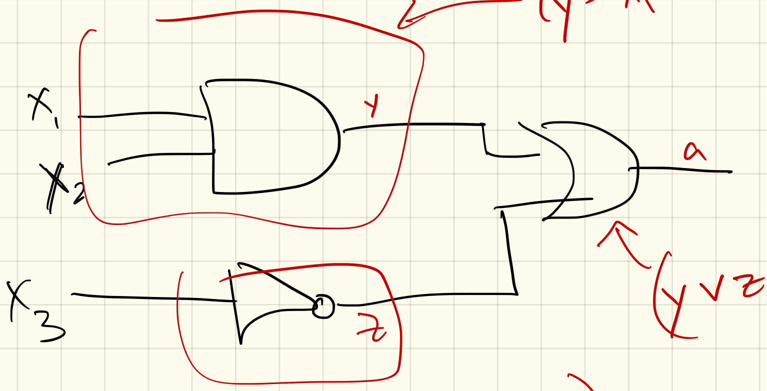
- In NP:

  Given a set of boolean inputs, linear time to compute output

- NP-Hard: reduce CIRCUITSAT to SAT:

For each gate, can write a formula

$(y = x_1 \wedge x_2)$



$(y \vee z = a)$

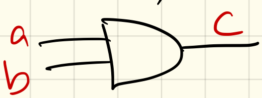
$(z = \neg x_3)$

$\overline{x_3}$

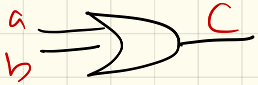
More carefully:

1) For any gate, can transform:

1 clause per gate



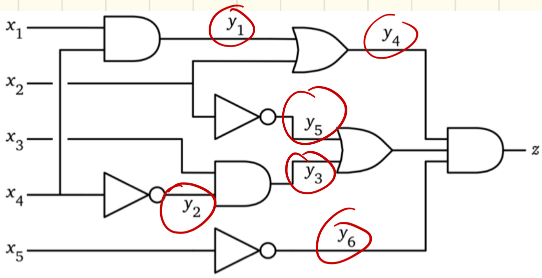
: $(c = a \wedge b)$



: $(c = a \vee b)$

: $(d = \overline{a})$

2) "And" these together, + want final output true:



$$(y_1 = x_1 \wedge x_4) \wedge (y_2 = \overline{x_4}) \wedge (y_3 = x_3 \wedge y_2) \wedge (y_4 = y_1 \vee x_2) \wedge$$
$$(y_5 = \overline{x_2}) \wedge (y_6 = \overline{x_5}) \wedge (y_7 = y_3 \vee y_5) \wedge (z = y_4 \wedge y_7 \wedge y_6) \wedge z$$

Is this poly-size?

Given n inputs + m gates:

Variables: $\left.\begin{array}{l}1 \text{ per input} \\ 1 \text{ per gate}\end{array}\right\} n+m$

Clauses: 1 per gate

Size of SAT formula:

$m+n+m = O(m+n)$

End reduction:

$$\begin{tikzcd}
\boxed{\text{boolean circuit}} \arrow[r, "O(n)"] & \boxed{\text{boolean formula}} \arrow[d, Rightarrow, "\text{SAT}"] \\
\boxed{\textsc{True or False}} & \boxed{\textsc{True or False}} \arrow[l, "\text{trivial}"']
\end{tikzcd}$$

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

(Here, "n" is total input size)

Next: 3SAT:

a restricted version of SAT

Dfn: Conjunctive Normal Form (CNF)

$$\overbrace{(a \vee b \vee c \vee d)}^{\text{clause}} \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{b})$$

"OR"s "and"

3SAT: SAT restricted to be CNF & exactly 3 literals per clause

$$(a \vee b \vee c) \wedge \underbrace{(\bar{a} \vee d \vee \bar{x})}_{\text{3 literals}} \wedge \ldots$$

Thm: 3SAT is NP-Hard

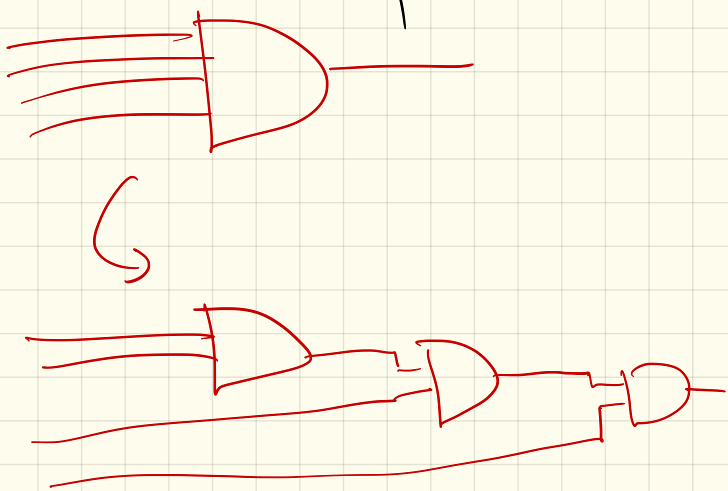
pf: Reduce circuitSAT to 3SAT.

Need to show any circuit can be transformed to CNF form

(so last reduction fails)

Steps:

(1) Rewrite so each gate has 2 inputs:



(2) Write formula, like in SAT:

3 types:

$$\boxed{\begin{aligned} y &= a \vee b \\ y &= a \wedge b \\ y &= \overline{a} \end{aligned}}$$

(3) Now, change to CNF:

go back to truth tables

$$a = b \wedge c \quad \longmapsto \quad (a \vee \bar{b} \vee \bar{c}) \wedge (\bar{a} \vee b) \wedge (\bar{a} \vee c)$$

$$a = b \vee c \quad \longmapsto \quad (\bar{a} \vee b \vee c) \wedge (a \vee \bar{b}) \wedge (a \vee \bar{c})$$

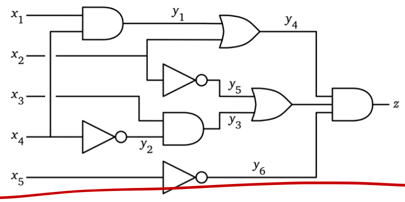
$$a = \bar{b} \quad \longmapsto \quad (a \vee b) \wedge (\bar{a} \vee \bar{b})$$

(4) Now, need 3 per clause:

$$a \quad \longmapsto \quad \overset{1}{(a \vee x \vee y)} \wedge \overset{2}{(a \vee \bar{x} \vee y)} \wedge \overset{3}{(a \vee x \vee \bar{y})} \wedge \overset{4}{(a \vee \bar{x} \vee \bar{y})}$$

$$a \vee b \quad \longmapsto \quad (a \vee b \vee x) \wedge (a \vee b \vee \bar{x})$$

Note: Bigger!



$$(y_1 = x_1 \wedge x_4) \wedge (y_2 = \overline{x_4}) \wedge (y_3 = x_3 \wedge y_2) \wedge (y_4 = y_1 \vee x_2) \wedge$$
$$(y_5 = \overline{x_2}) \wedge (y_6 = \overline{x_5}) \wedge (y_7 = y_3 \vee y_5) \wedge (z = y_4 \wedge y_7 \wedge y_6) \wedge z$$

A boolean circuit with gate variables added, and an equivalent boolean formula.

$\Downarrow$

$$\begin{aligned}
&(y_1 \vee \overline{x_1} \vee \overline{x_4}) \wedge (\overline{y_1} \vee x_1 \vee z_1) \wedge (\overline{y_1} \vee x_1 \vee \overline{z_1}) \wedge (\overline{y_1} \vee x_4 \vee z_2) \wedge (\overline{y_1} \vee x_4 \vee \overline{z_2}) \\
&\wedge (y_2 \vee x_4 \vee z_3) \wedge (y_2 \vee x_4 \vee \overline{z_3}) \wedge (\overline{y_2} \vee \overline{x_4} \vee z_4) \wedge (\overline{y_2} \vee \overline{x_4} \vee \overline{z_4}) \\
&\wedge (y_3 \vee \overline{x_3} \vee \overline{y_2}) \wedge (\overline{y_3} \vee x_3 \vee z_5) \wedge (\overline{y_3} \vee x_3 \vee \overline{z_5}) \wedge (\overline{y_3} \vee y_2 \vee z_6) \wedge (\overline{y_3} \vee y_2 \vee \overline{z_6}) \\
&\wedge (\overline{y_4} \vee y_1 \vee x_2) \wedge (y_4 \vee \overline{x_2} \vee z_7) \wedge (y_4 \vee \overline{x_2} \vee \overline{z_7}) \wedge (y_4 \vee \overline{y_1} \vee z_8) \wedge (y_4 \vee \overline{y_1} \vee \overline{z_8}) \\
&\wedge (y_5 \vee x_2 \vee z_9) \wedge (y_5 \vee x_2 \vee \overline{z_9}) \wedge (\overline{y_5} \vee \overline{x_2} \vee z_{10}) \wedge (\overline{y_5} \vee \overline{x_2} \vee \overline{z_{10}}) \\
&\wedge (y_6 \vee x_5 \vee z_{11}) \wedge (y_6 \vee x_5 \vee \overline{z_{11}}) \wedge (\overline{y_6} \vee \overline{x_5} \vee z_{12}) \wedge (\overline{y_6} \vee \overline{x_5} \vee \overline{z_{12}}) \\
&\wedge (\overline{y_7} \vee y_3 \vee y_5) \wedge (y_7 \vee \overline{y_3} \vee z_{13}) \wedge (y_7 \vee \overline{y_3} \vee \overline{z_{13}}) \wedge (y_7 \vee \overline{y_5} \vee z_{14}) \wedge (y_7 \vee \overline{y_5} \vee \overline{z_{14}}) \\
&\wedge (y_8 \vee \overline{y_4} \vee \overline{y_7}) \wedge (\overline{y_8} \vee y_4 \vee z_{15}) \wedge (\overline{y_8} \vee y_4 \vee \overline{z_{15}}) \wedge (\overline{y_8} \vee y_7 \vee z_{16}) \wedge (\overline{y_8} \vee y_7 \vee \overline{z_{16}}) \\
&\wedge (y_9 \vee \overline{y_8} \vee \overline{y_6}) \wedge (\overline{y_9} \vee y_8 \vee z_{17}) \wedge (\overline{y_9} \vee y_8 \vee \overline{z_{17}}) \wedge (\overline{y_9} \vee y_6 \vee z_{18}) \wedge (\overline{y_9} \vee y_6 \vee \overline{z_{18}}) \\
&\wedge (y_9 \vee z_{19} \vee z_{20}) \wedge (y_9 \vee \overline{z_{19}} \vee z_{20}) \wedge (y_9 \vee z_{19} \vee \overline{z_{20}}) \wedge (y_9 \vee \overline{z_{19}} \vee \overline{z_{20}})
\end{aligned}$$

How big? If exponential, no good

Still polynomial:

For each gate:

turned into ≤ 3 clauses (wrong sizes)

$\rightarrow \leq 4$ clause

$\Rightarrow \leq 12$ clauses per gate

<u>So:</u> $\Rightarrow O(m+n)$ size

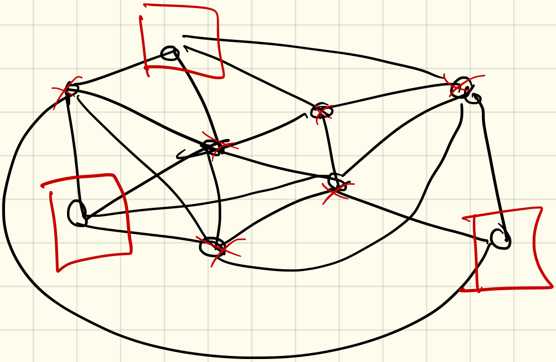
$$
\begin{tikzcd}
\boxed{\text{boolean circuit}} \arrow[r, "O(n)"] & \boxed{\text{3CNF formula}} \arrow[d, Rightarrow, "\text{3SAT}"] \\
\boxed{\textsc{True or False}} & \boxed{\textsc{True or False}} \arrow[l, "\text{trivial}"']
\end{tikzcd}
$$

$$T_{\text{CSAT}}(n) \leq O(n) + T_{\text{3SAT}}(O(n)) \quad\Longrightarrow\quad T_{\text{3SAT}}(n) \geq T_{\text{CSAT}}(\Omega(n)) - O(n)$$

Next Problem:

Independent Set:

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



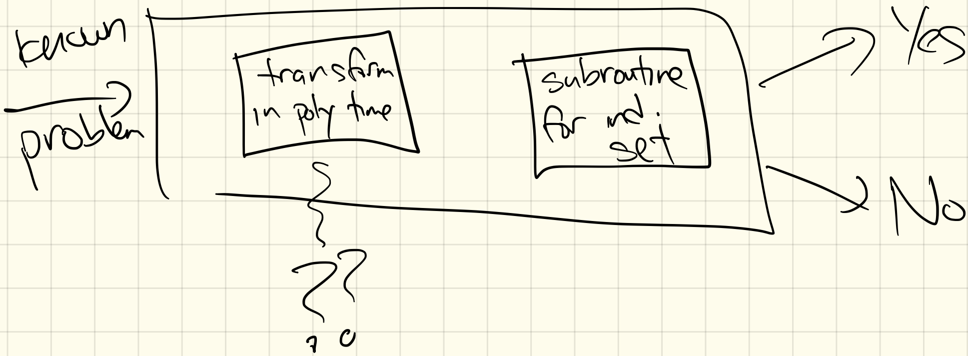
decision version:

Given G & #k, is there an indep set of size $\geq k$?

In NP: Given k vertices, check if any edges b/t them.

Challenge: No booleans!

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



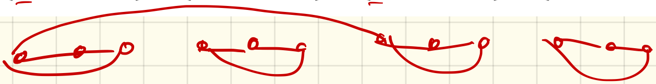
We'll use 3SAT

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

Reduction:

Input is 3CNF boolean formula

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



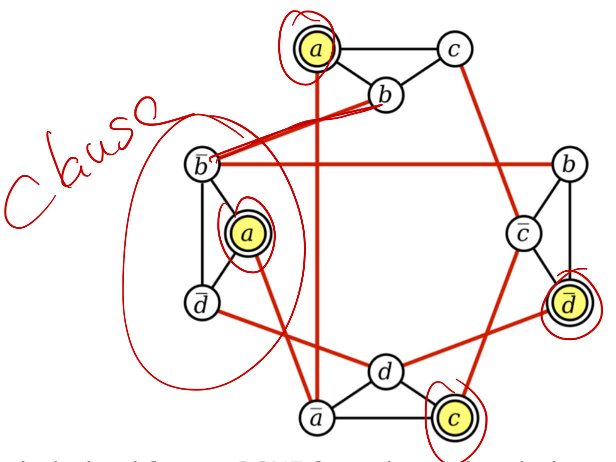
① Make a vertex for each literal in <u>each</u> clause

② Connect two vertices if:

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

Example:

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



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

Claim:

formula is Satisfiable

$\iff$

G has independent set of size n (= # input ~~variables~~ clauses)

pf: $\Rightarrow$ Say have a satisfying assignment.

Each clause has at least 1 true variable.

Choose the corresponding vertex in G to be in an independent set:

1 per "triangle" (clause)

know no 2 are connected, since satisfying assignment means x is true (so $\bar{x}$ won't be)

So, must have indep set of size exactly = # of clauses.

$\Leftarrow$: Start w/ indep set in G, of size m.

G has Δ's: means indep set choose ≤ 1 per triangle.

$\Rightarrow$ exactly 1 per Δ.

Since indep set, never choose both x + $\bar{x}$ in different clauses.

Build sat. assignment by marking all vertices in indep. set as true. (others don't matter)

So:

$$
\begin{tikzcd}
\boxed{\text{3CNF formula with } \underline{k} \text{ clauses}} \arrow[r, "O(n)"] & \boxed{\text{graph with } 3k \text{ nodes}} \arrow[d, Rightarrow, "\textsc{MaxIndSet}"] \\
\boxed{\textsc{True} \text{ or } \textsc{False}} & \boxed{\cancel{\text{maximum}} \text{ independent set size } \geq k} \arrow[l, "O(1)"']
\end{tikzcd}
$$

$$T_{3SAT}(n) \leq O(n) + T_{\textsc{MaxIndSet}}(O(n)) \implies T_{\textsc{MaxIndSet}}(n) \geq T_{3SAT}(\Omega(n)) - O(n)$$

#clauses

\# inputs

$\rightarrow O(k + n)$

Know:

- 3SAT
- Circuit SAT
- independent set of size k

$\rightarrow$ are NP-Hard