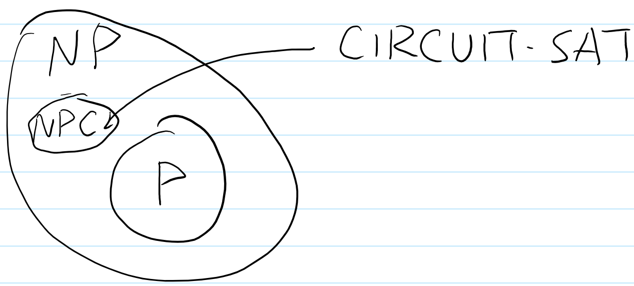
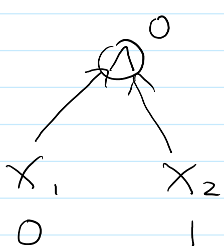
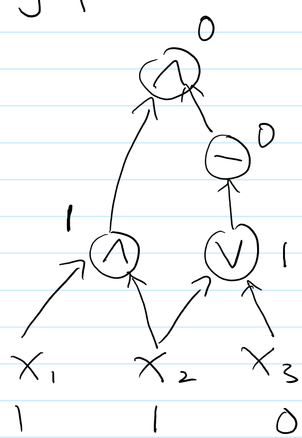
- Cook-Levin Theorem
- Example of Reductions

- Cook-Levin Theorem: For any problem L in NP, there is a polynomial time reduction from L to CIRCUIT-SAT (SAT, 3-SAT).



- CIRCUIT-SAT (circuit satisficbility)
  - boolean circuits
    - 3 basic operations: $\wedge$ and, $\vee$ or, $\overline{x}$ not
    - circuit: Directed acyclic graph whose nodes are "gates"





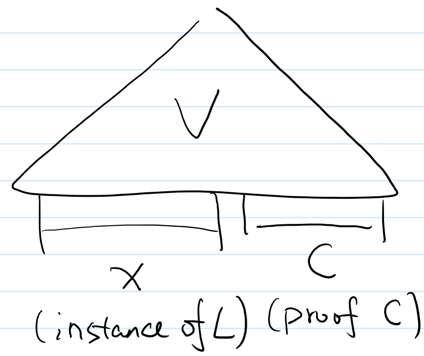
→ not satisfiable.

  - CIRCUIT-SAT: Given a circuit as the input, decide if there is a set of assignments to the input variables that makes the circuit output 1.
  - CIRCUIT-SAT $\in$ NP

    easy. The "proof" C is just one satisfying assignment, verifier will evaluate the circuit, and output 1 if the circuit outputs 1.

- Proof idea of Cook-Levin Theorem.
  - If L is an NP problem, then there is a poly time verifier V

- If the verifier V is actually implemented by a boolean circuit.



- for any instance $x \in L$, fix the input for x, try to see if the circuit is still satisfiable.

  If circuit is sat. $\Longrightarrow$ answer to x is yes

  Circuit is not sat. $\Longrightarrow$ answer to x is no.

- reduction from L to CIRCUIT-SAT.
- Claim: all polynomial time algorithm can be implemented by a circuit of polynomial size.

- reductions
  - To prove L is NP-hard, only need to reduce CIRCUIT-SAT to L.

    $$L \geq \text{CIRCUIT-SAT} \geq \text{any NP problem}$$

  - common starting point: 3-SAT problem
    - A 3-SAT instance has m <u>clauses</u>, each clause is an or of (at most) 3 <u>literals</u>. A literal is a variable or its negation.

$\underline{x_1 \vee x_3 \vee \overline{x_5}}$ $x_2 \vee \overline{x_3} \vee x_6$

clause   literals

clause is satisfied if $x_1 = 1$ or $x_3 = 1$ or $x_5 = 0$

- answer to 3-SAT is yes if all m clauses can be satisfied simultaneously.

(another way to write is $C_1 \wedge C_2 \wedge \cdots \wedge C_m$ is satisfiable)

C_1 ↑ first clause C_m ↑ last clause

3-SAT $\longrightarrow$ CIRCUIT-SAT easy

this reduction does <u>not</u> show 3-SAT is NP-hard.

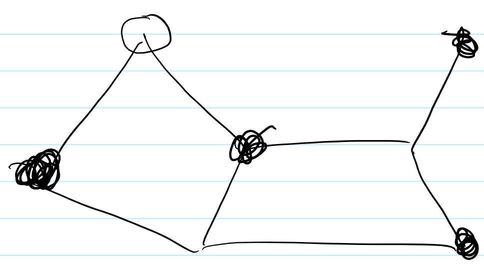
CIRCUIT-SAT $\geq$ 3-SAT

in order to show 3-SAT is NP-hard, need

CIRCUIT-SAT $\longrightarrow$ 3-SAT

3-SAT $\geq$ CIRCUIT-SAT $\geq$ any NP problem

- Example: INDEPENDENT-SET is NP-complete.
  - IND-SET: Given a graph G (undirected), $S \subseteq V$ is an independent set if no two vertices in S are connected by an edge.



IND-SET: (G, k) Decide whether G has an ind-set of size $\geq k$.

- reduction: 3-SAT $\longrightarrow$ IND-SET
  - idea: use "gadgets"

    for each object in 3-SAT $\longrightarrow$ map to some group of objects in IND-SET

    literals (x_i, $\bar{x}_i$) — vertices

    clauses ($C_1, C_2 \cdots$) $\longrightarrow$ edges.

- Literals: for each literal in each clause $\longrightarrow$ map to a vertex

  $x_1 \vee \overline{x_3} \vee x_5 \longrightarrow$ x_1 x_5 $\bar{x}_3$

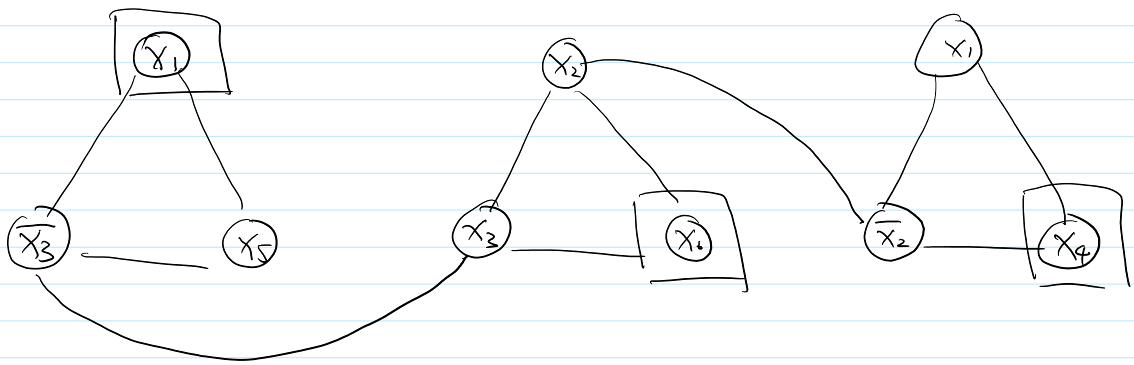
  in solution, $\geq$ one of these three is satisfied. $\longrightarrow$ the satisfied literal will be in ind-set

- edges: u, v are connected $\Longleftrightarrow$ cannot choose both u, v.
  - connect all vertices labeled x_i to all vertices labeled $\bar{x}_i$.

- connect all literals within the same clause.

(want each clause to contribute 1 vertex to IND-SET)

$(x_1 \vee \overline{x_3} \; x_5) \wedge (x_2 \vee x_3 \vee x_6) \wedge (x_1 \vee \overline{x_2} \vee x_4)$



Claim: The 3-SAT instance is satisfiable iff the graph has an ind-set of size m,