EECS 336: Introduction to Algorithms	Lecture 17
P vs. NP (cont.)	CIRCUIT-SAT, 3-SAT

Reading: 8.1-8.4

Last time:

- 3-SAT $\leq_{\mathcal{P}}$ INDEP-SET

Today:

- NP $\leq_{\mathcal{P}}$ CIRCUIT-SAT $\leq_{\mathcal{P}}$ 3-SAT

Notorious Problem: NP

input:

- decision problem verifier program VP.
- polynomial $p(\cdot)$.
- decision problem instance: x

output:

- "Yes" if exists certificate c such that $VP(x, c)$ has "verified = true" at computational step $p(|x|)$.
- "No" otherwise.

Problem 4: 3-SAT

input: boolean formula $f(\mathbf{z})$

- in conjunctive normal form (CNF)
- three literals per or-clause
- or-clauses anded together.

output:

- "Yes" if assignment $\mathbf{z}$ with $f(\mathbf{z}) = T$ exists
- "No" otherwise.

Note: 2 steps to NP-completeness

1. $X \in \mathcal{NP}$
2. X is $\mathcal{NP}$-hard (via reduction)

3 steps to reduction

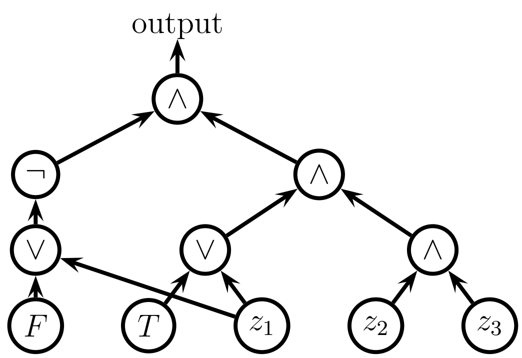
1. construction
2. runtime of construction
3. correctness of construction (iff)

Note: algorithms in reductions:

```
            3-SAT          INDEP-SET
input:        f        =>       G,D
output:       z       <=>        S
```

Circuit Satisfiability

Example:



Problem 4: CIRCUIT-SAT

input: boolean circuit $Q(\mathbf{z})$

- directed acyclic graph $G = (V, E)$
- internal nodes labeled by logical gates:

  "and", "or", or "not"

- leaves labeled by variables or constants

  $T, F, z_1, \ldots, z_n$.

- root r is output of circuit

output:

- "Yes" if exists $\mathbf{z}$ with $Q(\mathbf{z}) = T$
- "No" otherwise.

Lemma: CIRCUIT-SAT is $\mathcal{NP}$-hard.

Proof: (reduce from NP)

- goal: convert NP instance (VP, p, x) to CIRCUIT-SAT instance Q
- $VP(\cdot, \cdot)$ polynomial time

  $\Rightarrow$ computer can run it in poly steps.

- each step of computer is circuit.
- output of one step is input to next step
- unroll $p(|x|)$ steps of computation

  $\Rightarrow \exists$ poly-size circuit $Q'(\mathbf{x}, \mathbf{c}) = VP(x, c)$

- hardcode $\mathbf{x}$: $Q(\mathbf{c}) = Q'(\mathbf{x}, \mathbf{c})$
- Conclusion: Q is sat iff exists c with $VP(x, c) =$ "verified".

QED

LE3-SAT

"CIRCUIT-SAT $\leq_{\mathcal{P}}$ LE3-SAT $\leq_{\mathcal{P}}$ 3-SAT"

Problem 5: LE3-SAT

"like 3-SAT but <u>at most</u> 3 literals per or-clause"

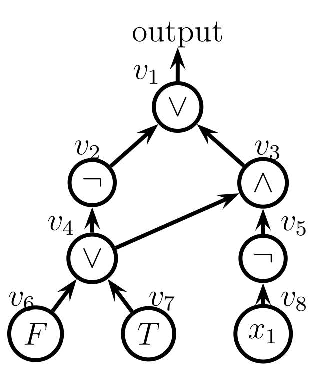
Note: $\leq_{\mathcal{P}}$ is transitive: if $Y \leq_{\mathcal{P}} X$ and $X \leq_{\mathcal{P}} Z$ then $Y \leq_{\mathcal{P}} Z$.

Recall: NP $\leq_{\mathcal{P}}$ CIRCUIT-SAT

Plan: CIRCUIT-SAT $\leq_{\mathcal{P}}$ LE3-SAT $\leq_{\mathcal{P}}$ 3-SAT

Lemma: CIRCUIT-SAT $\leq_{\mathcal{P}}$ LE3-SAT

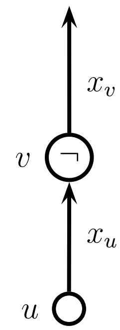
Example:



Proof: (reduce from CIRCUIT-SAT)

Step 1: convert CIRCUIT-SAT instance Q into 3-SAT instance f

- variables x_v for each vertex of Q.
- encode gates
  - **not**: if v not gate with input from u



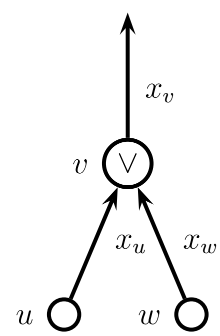
need $x_v = \bar{x}_u$

$x_v \setminus x_u$	0	1
0	0	1
1	1	0

$\Rightarrow$ add clauses $(x_v \vee x_u) \wedge (\bar{x}_v \vee \bar{x}_u)$

- **or**: if v is or gate from u to w

need $x_v = x_u \wedge x_w$



$x_v \setminus x_u x_w$	00	01	11	10
0	1	0	0	0
1	0	1	1	1

$\Rightarrow$ add clauses $(\bar{x}_v \vee x_u \vee x_w) \wedge (x_v \vee \bar{x}_u) \wedge (x_v \vee \bar{x}_w)$

- **and:** if v is and gate from u to w

$\Rightarrow$ add clauses $(x_v \vee \bar{x}_u \bar{x}_w) \wedge (\bar{x}_v \vee x_u) \wedge (\bar{x}_v \vee x_w)$.

- **0:** if v is 0 leaf.

need $x_v = 0$

$\Rightarrow$ add clause $(\bar{x}_v)$

need $x_v = 1$

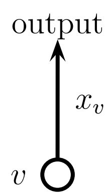
- **1:** if v is 1 leaf.

  $\Rightarrow$ add clause (x_v)

- **literal:** if v is literal z_j

  $\Rightarrow$ do nothing

- **root:** if v is root



need $x_v = 1$

$\Rightarrow$ add clause (x_v).

Step 2: construction is polynomial time.

- at most 3 clauses in f per node in Q.

Step 3: construction is correct (i.e., Q is sat iff f is sat.)

- f constrains variables v_i to "proper circuit outcomes".

- if exists $\mathbf{z}$ s.t. $f(\mathbf{z})$ is T,

  then can read $\mathbf{x}$ from $\mathbf{z}$ and $\mathbf{z}$ encodes proper circuit outcome to make Q output T for this $\mathbf{x}$.

- if Q outputs T for some $\mathbf{x}$

  then can map $\mathbf{x}$ and values at nodes to variables $\mathbf{z}$ such that $f(\mathbf{z})$ is true.

QED

Lemma: LE3-SAT $\leq_{\mathcal{P}}$ 3-SAT

Step 1: convert LE3-SAT instance f' into 3-SAT instance f

- $f \leftarrow f'$

- add variables w_1, w_2

- add w_i to 1- and 2-clauses

  $(l_1) \Rightarrow (l_1 \vee w_1 \vee w_2)$.

  $(l_1 \vee l_2) \Rightarrow (l_1 \vee l_2 \vee w_1)$.

- ensure $w_i = 0$ add variables y_1, y_1 and clauses:

  $(\bar{w}_i \vee y_1 \vee y_2)$

  $(\bar{w}_i \vee \bar{y}_1 \vee y_2)$

  $(\bar{w}_i \vee y_1 \vee \bar{y}_2)$

  $(\bar{w}_i \vee \bar{y}_1 \vee \bar{y}_2)$

Step 2: construction is polynomial time.

Step 3: f is sat iff f' is sat.

- given satisfying assignment $(\bar{z}, w_1, w_2, y_1, y_2)$ to f,

  $\Rightarrow$ $w_i = F$ by construction.

  $\Rightarrow$ $f(\bar{z}, F, F, y_1, y_2) \overset{\text{simplify}}{\Longrightarrow} f(\bar{z})$

  $\Rightarrow$ f is sat.

- given satisfying assignment $\bar{z}$ to f',

  - $f(\bar{z}, w_1, w_2, y_1, y_2) \overset{\text{simplify}}{\Longrightarrow}$ "clauses with only w_i and y_i"

  - set $w_i = F$ and $y_i = F$ (or anything) to satisfy. **QED**