CPSC 420 Lecture 17: Today's announcements:

- HW3 is on Gradescope, due Mar 9, 23:59
- Examlet 3 on Mar 17 in class. Closed book & no notes
- Reading: Shor's notes on Lempel-Ziv compression `https://math.mit.edu/~shor/PAM/lempel_ziv_notes.pdf`
  NP-hardness [by Erickson]

Today's Plan

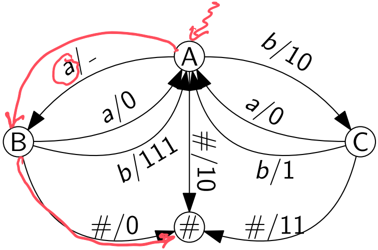
- Compression
  - Lempel-Ziv Compression
- NP-hardness

Compression [Lempel & Ziv 1978]

As good as any finite state compressor

Every text has an algorithm that compresses it well, but the algorithm is as big as the text.

A **finite state compressor** is a finite state machine with output strings on transitions.



input T	Output
$a\#$	0
$aa\#$	010

Require: output of M uniquely determines input T.

1. Show all FSC with s states can't compress better than $r_s(T)$
2. Show $|\text{LZ78}(T)| \leq r_s(T) + o(|T|)$

Compression [Lempel & Ziv 1978]

$s = \#$states

As good as FSC continued

Let $c(T) =$ max number of distinct phrases T can be split into.
Let $c_j = \#$phrases that cause M to output j bits starting from some state in M.
$c_j \leq s^2 2^j$ since $[A, j\text{-bit code}, B]$ uniquely specifies phrase x, where A is state when M starts reading x and B is state when it stops.

If two phrases x and y cause the same output going from A to B then M outputs the same encoding for $wx\# \neq wy\#$, where w takes M to state A.

Assume $c_j = s^2 2^j$ for all $j \leq k$ i.e. use max number of short codes.

$$c(T) = \sum_{j=0}^{k} c_j \leq s^2 \sum_{j=0}^{k} 2^j = s^2(2^{k+1} - 1)$$

Total length of encoding by M:

$$|M(T)| \geq \sum_{j=0}^{k} j c_j = s^2 \sum_{j=0}^{k} j 2^j = s^2((k-1)2^{k+1} + 2) = r_s(T)$$

Compression [Lempel & Ziv 1978]

As good as FSC continued

From before

$$|\mathrm{LZ78}(T)| \leq c(T)\log_2 c(T)$$

and

$$r_s(T) = s^2((k-1)2^{k+1}+2) \geq (c(T)+s^2)\log_2(\frac{c(T)}{4s^2})$$

So

$$|\mathrm{LZ78}(T)| \leq r_s(T) + \underbrace{2c(T) - s^2\log_2 c(T) + (c(T)+s^2)\log_2(4s^2)}_{\text{this is } o(|T|)}$$

Decision Problems

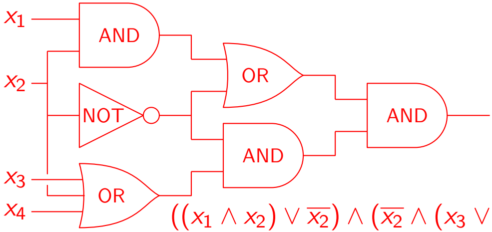
A **decision problem** is a problem (a problem takes an input and produces an output) whose output is either "Yes" or "No".

Low-cost Spanning Tree: Does a given weighted graph G have a spanning tree with total weight less than w?

Primality: Is 13247836221587427 prime?

Sorting: Sort cat dog bee ant

Circuit satisfiability: Is there an input to a given boolean circuit that causes it to output 1?



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

Decision Problems

A **decision problem** is a problem (a problem takes an input and produces an output) whose output is either "Yes" or "No".

Low-cost Spanning Tree: Does a given weighted graph G have a spanning tree with total weight less than w?

Primality: Is 13247836221587427 prime?

~~Sorting: Sort cat dog bee ant~~

Circuit satisfiability: Is there an input to a given boolean circuit that causes it to output 1?

Circuit tautology: Do all inputs to a given boolean circuit cause output 1?

Complexity classes P, NP, co-NP

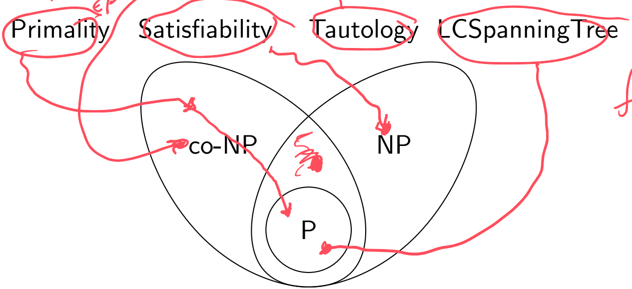
$\mathcal{O}(n^c)$ for constant c n = size

P: Decision problems that can be solved in polynomial time.

NP: Decision problems where if the answer is “Yes” then there is a *proof* that can be checked in polynomial time.

co-NP: Decision problems where if the answer is “No” then there is a *proof* that can be checked in polynomial time.

2002 AKS primality $\in P$



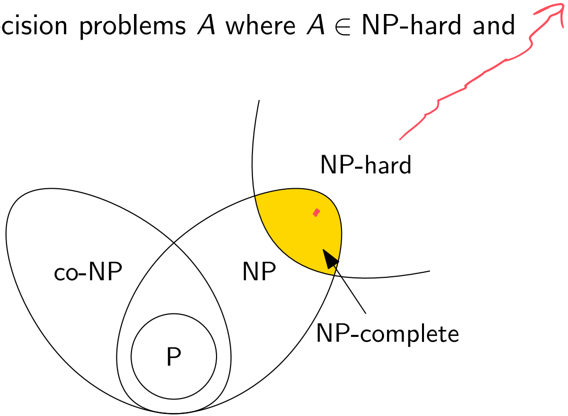
factoring

Our current guess of P vs NP vs co-NP

NP-hard, NP-complete

NP-hard: Problems A where if A can be solved in polynomial time then $\mathrm{P} = \mathrm{NP}$.

NP-complete: Decision problems A where $A \in$ NP-hard and $A \in$ NP.



Our current guess of P, NP, co-NP, NP-hard, NP-complete

Cook-Levin Theorem

Theorem

Circuit satisfiability is NP-hard

Proof.

You don't need to know the proof but the idea is: Show how to encode the execution of any polynomial-time, non-deterministic Turing machine M on an input x as some boolean circuit that is satisfiable if and only if M outputs "Yes" on input x. □

How do we show other problems are NP-hard?

Cook-Levin Theorem

Theorem

Circuit satisfiability is NP-hard

Proof.

You don't need to know the proof but the idea is: Show how to encode the execution of any polynomial-time, non-deterministic Turing machine M on an input x as some boolean circuit that is satisfiable if and only if M outputs "Yes" on input x. □

How do we show other problems are NP-hard?

> To prove problem A is NP-hard, show how to **reduce** (in polynomial time) an NP-hard problem to A.

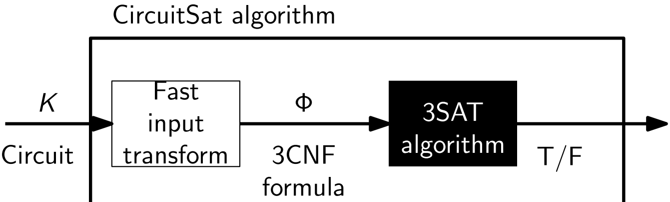
Reduce Circuit Satisfiability to A means "Solve Circuit Satisfiability using an algorithm for A."

3SAT is NP-hard

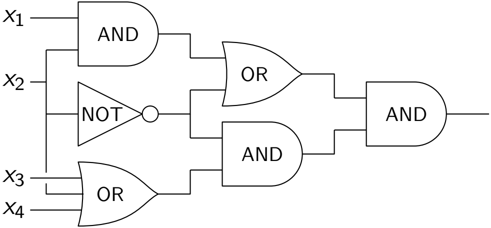
3SAT: Is a given boolean formula, which is the AND of several three-literal OR-clauses, satisfiable?

Example: Is $(a \vee \overline{b} \vee c) \wedge (\overline{a} \vee \overline{b} \vee \overline{c}) \wedge (a \vee \overline{c} \vee \overline{d})$ satisfiable?

Reduction: From CircuitSat to 3SAT

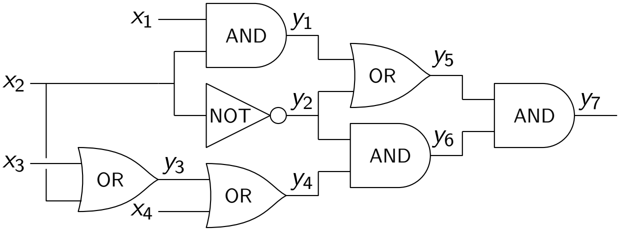


Transform Circuit into 3CNF formula



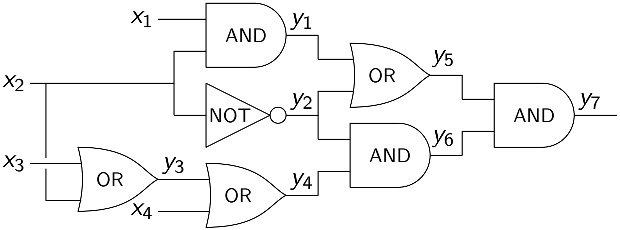
Input to CircuitSAT = Boolean Circuit

Transform Circuit into 3CNF formula



Equivalent circuit where AND and OR gates have two inputs.

Transform Circuit into 3CNF formula



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

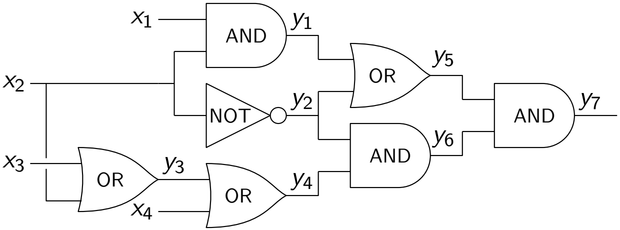
$$a \vee b = c \longmapsto (a \vee b \vee \overline{c}) \wedge (\overline{a} \vee c) \wedge (\overline{b} \vee c)$$

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

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

CNF formula satisfied iff circuit is satisfied.

Transform Circuit into 3CNF formula



$$a \vee b \longmapsto (a \vee b \vee z) \wedge (a \vee b \vee \overline{z})$$

$$a \longmapsto (a \vee w \vee z) \wedge (a \vee \overline{w} \vee z) \wedge (a \vee w \vee \overline{z}) \wedge (a \vee \overline{w} \vee \overline{z}))$$

$$(\overline{x_1} \vee \overline{x_2} \vee y_1) \wedge (x_1 \vee \overline{y_1} \vee z_1) \wedge (x_1 \vee \overline{y_1} \vee \overline{z_1}) \wedge (x_2 \vee \overline{y_1} \vee z_2) \wedge (x_2 \vee \overline{y_1} \vee \overline{z_2}) \wedge (x_2 \vee y_2 \vee z_3) \wedge (x_2 \vee y_2 \vee \overline{z_3})(\overline{x_2} \vee \overline{y_2} \vee z_4) \wedge (\overline{x_2} \vee \overline{y_2} \vee \overline{z_4}) \wedge (x_3 \vee x_2 \vee \overline{y_3}) \wedge (\overline{x_3} \vee y_3 \vee z_5) \wedge (\overline{x_3} \vee y_3 \vee \overline{z_5}) \wedge (\overline{x_2} \vee y_3 \vee z_6) \wedge (\overline{x_2} \vee y_3 \vee \overline{z_6}) \wedge (y_3 \vee x_4 \vee \overline{y_4}) \wedge (\overline{y_3} \vee y_4 \vee z_7) \wedge (\overline{y_3} \vee y_4 \vee \overline{z_7}) \wedge (\overline{x_4} \vee y_4 \vee z_8) \wedge (\overline{x_4} \vee y_4 \vee \overline{z_8}) \wedge (y_1 \vee y_2 \vee \overline{y_5}) \wedge (\overline{y_1} \vee y_5 \vee z_9) \wedge (\overline{y_1} \vee y_5 \vee \overline{z_9}) \wedge (\overline{y_2} \vee y_5 \vee z_{10}) \wedge (\overline{y_2} \vee y_5 \vee \overline{z_{10}}) \wedge (\overline{y_2} \vee \overline{y_4} \vee y_6) \wedge (y_2 \vee \overline{y_6} \vee z_{11}) \wedge (y_2 \vee \overline{y_6} \vee \overline{z_{11}}) \wedge (y_4 \vee \overline{y_6} \vee z_{12}) \wedge (y_4 \vee \overline{y_6} \vee \overline{z_{12}}) \wedge (\overline{y_5} \vee \overline{y_6} \vee y_7) \wedge (y_5 \vee \overline{y_7} \vee z_{13}) \wedge (y_5 \vee \overline{y_7} \vee \overline{z_{13}}) \wedge (y_6 \vee \overline{y_7} \vee z_{14}) \wedge (y_6 \vee \overline{y_7} \vee \overline{z_{14}}) \wedge (y_7 \vee z_{15} \vee z_{16}) \wedge (y_7 \vee \overline{z_{15}} \vee z_{16}) \wedge (y_7 \vee z_{15} \vee \overline{z_{16}}) \wedge (y_7 \vee \overline{z_{15}} \vee \overline{z_{16}})$$

3CNF formula satisfied iff circuit is satisfied.