Summary/Roadmap

decision problems: answer is YES or NO

solvable in poly-time · poly-time verification · solvable in poly-space · solvable in exponential time

$$P \stackrel{?}{=} NP \stackrel{?}{\subseteq} PSPACE \stackrel{?}{\subseteq} EXPTIME$$

we think $\neq$ $\qquad \neq \qquad \neq$

$P \neq EXPTIME$

generalized checkers $\notin P$

$\underline{Y \leq_p X}$ Y is poly-time reducible to X

$X \in P \longrightarrow Y \in P$

$Y \notin P \longrightarrow X \notin P$

Let $A \in NP$ and $A \notin P$ then $P \neq NP$ and all NP-complete problems are not in P

let B be NP-complete

$A \leq_p B \qquad B \in P$

Suppose $TSP \in P$ then $P = NP$

$A \in NP \qquad A \leq_p TSP \qquad A \in P$

Polynomial-time verification algorithm for decision problem X.

For all x, X(x) = YES ⟷ there is a y (→ poly in size of x) such that X-VERIFY(x,y) = YES

→ given G, does there exist a Ham cycle in G

HC-VERIFY(G, p) ← polynomial in size of G

poly — check that each v in G appears at least once in p

poly — check that each v in G appears at most once in p

poly — for each $(v_i, v_j) \in p$, check that edge (v_i, v_j) exists in G
and edge from last to first exists in G

if all YES, return YES
else return NO

if G has Hamiltonian cycle c, then HC-VERIFY(G, c) = YES
HC(G) = YES

if G has no Hamiltonian cycle, then HC-VERIFY(G, p) = NO for all p
HC(G) = NO

NP = set of problems X s.t. there is a polynomial-time verification algorithm for X

NP′ = set of problems X s.t. there is a non-deterministic (allow coin flips), polynomial-time solution for X

X(x) = YES ⟷ P(X-RANDOM(x) = YES) > 0

HC ∈ NP′ :

HC-RANDOM(G)
guess: randomly permute vertices to get $v_0, \ldots, v_{n-1}, v_0$
check: for i = 0 to n−1
  if no edge $(v_i, v_{(i+1)\%n})$ output NO
output YES

if G has HC c, then with prob $\frac{1}{n!} > 0$ we choose c and output YES

if G has no HC, output NO no matter what we choose in 1st step

NP = NP′

NP ⊆ NP′ : Let X ∈ NP. Then ∃ poly-time verifier X-VERIFY(x,y)

Write non-deterministic alg for X: X-RANDOM(x)
randomly generate y
output X-VERIFY(x,y)

X(x) = YES → ∃ y s.t. X-VERIFY(x,y) = YES → X-RANDOM picks y with prob > 0
→ X-RANDOM outputs Y w/ prob > 0

$X(x) = NO \rightarrow \forall y$, X-VERIFY$(x,y) = NO \rightarrow$ X-RANDOM (x) outputs NO always
$\rightarrow$ X-RANDOM (x) outputs YES w/prob 0

NP′ ⊆ NP: Let $X \in NP'$. Then $\exists$ poly-time non-deterministic X-RANDOM(x).

Write verifier X-VERIFY(x,y)
simulate X-RANDOM(x) using y as random bits

$X(x) = YES \rightarrow P(\text{X-RANDOM}(x) = YES) > 0$
$\rightarrow$ some sequence of random bits y makes X-RANDOM(x) = YES
$\rightarrow$ X-VERIFY(x,y) = YES

$X(x) = NO \rightarrow P(\text{X-RANDOM}(x) = NO) = 0$
$\rightarrow$ all y make X-VERIFY(x,y) = NO

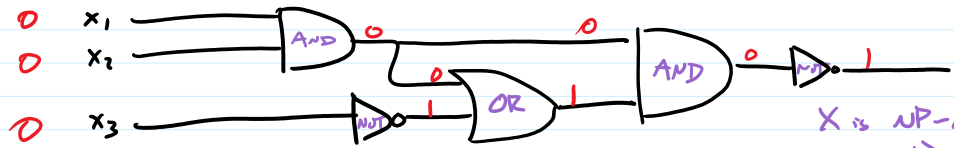
CIRCUIT-SAT and SAT

$\text{CIRCUIT-SAT} \le_p \text{SAT} \le_p \text{3-SAT} \le_p \text{VC} \le_p \text{HC}$

SAT $\in$ NP (done in video) } $\to$ so SAT is NP-complete

CIRCUIT-SAT $\le_p$ SAT — given φ, determine if φ has a satisfying assignment

$\hookrightarrow$ given combinational circuit C, is there an input to make output 1



CIRCUIT-SAT is NP-complete

X is NP-complete means
1) X $\in$ NP
2) for all Y $\in$ NP $Y \le_p X$

want : CIRCUIT-SAT(C)

poly create φ from C $\leftarrow$ φ has a satisfying assignment $\updownarrow$ C has a satisfying input

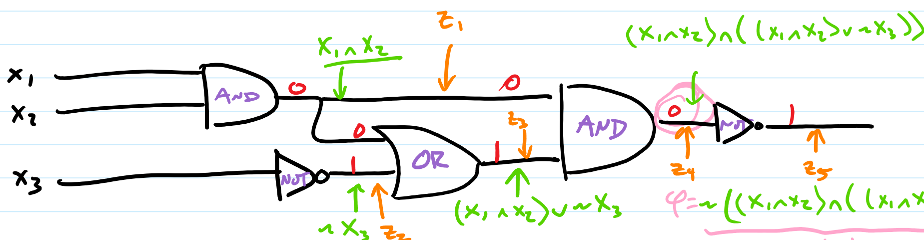
output SAT(φ)

To show X is NP-complete
1) show X $\in$ NP
2) choose NP-complete Z
show $Z \le_p X$

if SAT $\in$ P $\to$ CIRCUIT-SAT $\in$ P

CIRCUIT-SAT $\notin$ P $\to$ SAT $\notin$ P

for all $\underbrace{Y \in NP}_{Z}$, $Y \le_p Z \le_p X$ so $Y \le_p X$



$\varphi = \sim((x_1 \land x_2) \land ((x_1 \land x_2) \lor \sim x_3)))$

but can be exponential in size of circuit

1 subexpression for each gate, each of O(1) size

$\varphi = z_5 \land (z_5 \leftrightarrow \sim z_4) \land (z_4 \leftrightarrow (z_1 \land z_3)) \land (z_3 \leftrightarrow (z_1 \lor z_2)) \land (z_2 \leftrightarrow \sim x_3)$
$\land (z_1 \leftrightarrow (x_1 \land x_2))$

φ is satisfiable $\leftrightarrow$ C is satisfiable

size is linear in size of circuit

SAT and 3-SAT

3-SAT $\in$ NP.

SAT $\leq_p$ 3-SAT $\rightarrow$ Given φ in 3CNF

$\hookrightarrow (x \vee y \vee z) \wedge (\sim x \vee y \vee \sim z) \wedge (x \vee x \vee x) \wedge \ldots$

so 3-SAT is NP-complete

want: given φ, find 3CNF φ' s.t. φ is satisfiable $\Longleftrightarrow$ φ' is satisfiable

$$(x \underset{z_1}{\wedge} y) \underset{z_4}{\wedge} \underset{z_3}{\sim} (x \underset{z_2}{\vee} y)$$

same idea as in CIRCUIT-SAT $\leq_p$ SAT

$$\varphi' = z_4 \wedge (z_4 \Longleftrightarrow (z_1 \wedge z_3)) \wedge (z_3 \Longleftrightarrow \sim z_2) \wedge (z_2 \Longleftrightarrow x \vee y) \wedge (z_1 \Longleftrightarrow x \wedge y)$$

now this is $\sim \wedge \sim \wedge \sim \wedge \sim$

but each $\sim$ is not $(t_1 \vee t_2 \vee t_3)$ so use truth tables to convert

x	y	z_1	$z_1 \Longleftrightarrow x \wedge y$
T	T	T	T
T	T	F	F
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	T
F	F	T	F
F	F	F	T

one clause for each T row in truth table

$$z_1 \Longleftrightarrow x \wedge y \equiv (x \wedge y \wedge z_1) \vee (x \wedge \sim y \wedge \sim z_1) \vee (\sim x \wedge y \wedge \sim z_1) \vee (\sim x \wedge \sim y \wedge \sim z_1)$$

$$(x \wedge y \wedge \sim z_1) \vee (x \wedge \sim y \wedge z_1) \vee (\sim x \wedge y \wedge z_1) \vee (\sim x \wedge \sim y \wedge z_1)$$

close... now is $(\sim \wedge \sim \wedge \sim) \vee (\sim \wedge \sim \wedge \sim) \vee \ldots$

wanted $(\sim \vee \sim \vee \sim) \wedge (\sim \vee \sim \vee \sim) \wedge \ldots$

try again

$$z_1 \Longleftrightarrow x \wedge y \equiv \sim \underbrace{\sim (z_1 \Longleftrightarrow x \wedge y)}_{\text{get this from the F rows in truth table}} \equiv \sim((x \wedge y \wedge \sim z) \vee (x \wedge \sim y \wedge z) \vee (\sim x \wedge y \wedge z) \vee (\sim x \wedge \sim y \wedge z))$$

$$\underset{\text{De Morgan}}{\equiv} \underbrace{(\sim x \vee \sim y \vee z) \wedge (\sim x \vee y \vee \sim z) \wedge (x \vee \sim y \vee \sim z) \wedge (x \vee y \vee \sim z)}_{\text{this is 3CNF; call it } \varphi_1}$$

$$\varphi' = z_4 \wedge \varphi_4 \wedge \varphi_3 \wedge \varphi_2 \wedge \varphi_1$$

conjunction of 3CNF is still 3CNF

each φ_i is $O(1)$ size and can be found in $O(1)$ time

no matter how big φ is, each truth table has only 8 rows

so φ' is linear in # of operators in φ = linear in size of φ and can be created in poly time