CMSC 451 Nov 11

NP-Completeness (II)

Decision Problems & **Languages.**

Ans: Yes/no. ↘ set of strings.

x is a "Yes" instance of Prob.

$\Leftrightarrow$ $x \in$ Prob.

P: set of languages (or dec. probs.) **solvable** in **polynomial time**

NP: set of languages **verifiable** in polynomial time

$x \in$ Prob., $V(x, \underline{c}) =$ Yes. ($\underline{c}$: certificate; V: verification algo.)

Polynomial Reduction:

$L_1 \leq_p L_2$ means. $\exists$ **poly-time** function f, s.t.

$\forall x$, $x \in L_1 \Leftrightarrow f(x) \in L_2$.

NP-hard: L is **NP-hard** if $\forall L' \in NP$, $L' \leq_p L$

L is **NP-complete** if $L \in NP$ and L is **NP-hard**

$L_1 \leq_p L_2 \Rightarrow$ if $L_2 \in P$, then $L_1 \in P$.

if any NP-complete problem is solvable in poly-time,
then $P = NP$, all NP problems are solvable in poly-time.

if any problem in NP is not solvable in poly-time,
then no NP-complete problem is solvable in poly-time
$(P \neq NP)$

Q: How to prove a problem is NP-complete?

Observation: if we already have an NP-Complete problem,
then can be done by reducing that known problem to it.

First NPC problem: Boolean Satisfiability (SAT)

Boolean formula:

set of variables: $x, y, z, \ldots$ / $x_1, x_2, x_3, x_4, \ldots$

assign True or False to them.

(logical operators)

operations: $\wedge$ (and) $x \wedge y$ is T iff both x, y are T.

$\vee$ (or) $x \vee y$ is T iff either of them is T.

$\neg$ (not) $\neg x$ is T iff x is F

($\neg x \equiv \bar{x}$).

Boolean Formula:

Example: $F(x,y,z) = (x \wedge (y \vee \bar{z})) \wedge ((\bar{y} \wedge z) \vee \bar{x})$

$x=T,\ y=F,\ z=F$

$F(x,y,z) = (\bar{z} \vee x) \wedge (z \vee y) \wedge (\bar{x} \wedge \bar{y})$

same — contradict

$z=T \Rightarrow x=T \Rightarrow y=F$

$z=F \Rightarrow y=T \Rightarrow x=F$

SAT: Given a formula, is it possible to assign all the variables s.t. the formula is true?

(is it satisfiable?)

Cook-Levin Theorem (1971):

SAT is NP-Complete.

Proof: Long and technical.

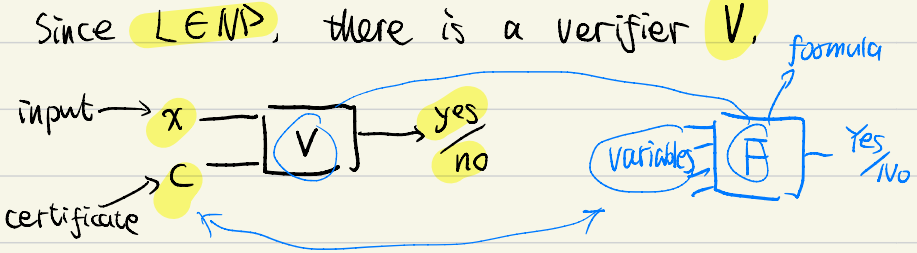
High-level intuition:

① SAT $\in$ NP.

to verify SAT, use a correct assignment as certificate, and verifier check if the formula is true.

② $\forall L \in NP$, $L \leq_p SAT$.

Since $L \in NP$, there is a verifier V.



c can be encoded into a bit string.

$c = 010111\cdots$

$\Rightarrow c_1$ = 1st bit of c, c_2 = 2nd bit of c,

$\cdots$

Then: $V(x, _)$ is a boolean formula of $c_1, c_2, \cdots, c_n$

Because V is poly time, the formula is a poly-size one.

$\Rightarrow x \in L \Leftrightarrow \exists c,\ V(x,c) = \text{yes} \Leftrightarrow F(c_1, \cdots, c_n) = T$ ($\exists c_1, \cdots, c_n$ assignment)

$\Leftrightarrow$ Formula is satisfiable.

$\Rightarrow L \leq_p SAT.$

Stronger Version:

SAT is NP-complete, even if the formula is restricted into 3CNF.

3 – conjunctive normal form (3CNF):

A formula that is the conjunction ("∧"/and) of clauses, each of which is the disjunction ("∨"/or) of 3 literals (variable or its negation)

e.g. $\underbrace{(x_1 \vee x_2 \vee \bar{x}_3)}_{\text{clause}} \wedge \underbrace{(\bar{x}_1 \vee x_3 \vee x_4)}_{\text{clause}} \wedge \underbrace{(x_2 \vee \bar{x}_3 \vee \bar{x}_4)}_{\text{clause}} \wedge \dots$

↳ at least one among $x_1, x_2, \bar{x}_3$ is true.

3SAT: SAT restricted to 3CNF.

Theorem: 3SAT is NP-complete

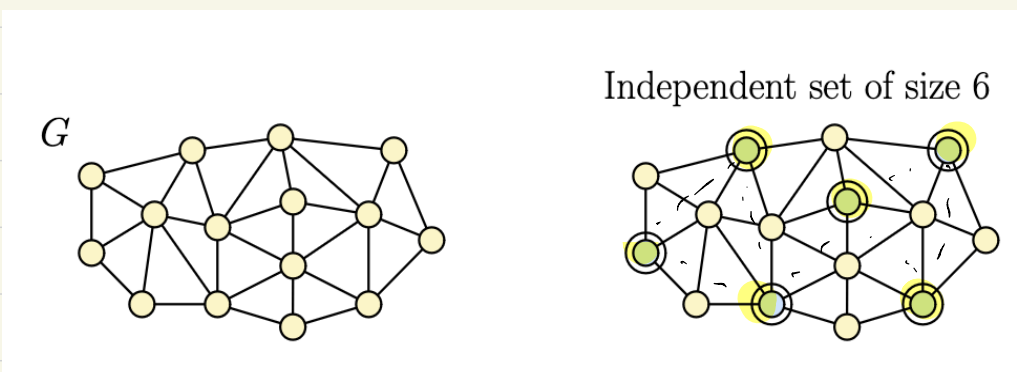
Prove a prob. is NP-complete.

Reduce an NPC problem into it.

Independent Set (IS):

Given a graph G, does it have a independent set of size $\geq k$?

Independent set: set of vertices s.t. no edges among them.



Claim: IS is NP-complete.

① IS $\in$ NP,

Certificate: an actual ind. set of size $\geq k$

Verifier: check if it is indeed an ind. set.

② IS is NP-hard, want to show 3SAT $\leq_p$ IS.

Given an instance of 3SAT, find translation function f, (in poly-time), s.t.

$x \in$ 3SAT $\iff$ $f(x) \in$ IS.

f : (formula)

for each clause $(x_i \vee x_j \vee x_k)$

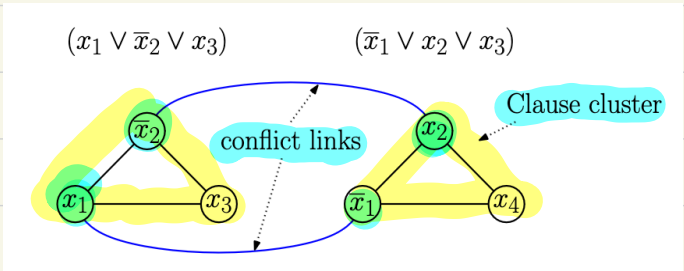
Create 3 vertices, labelled x_i x_j & x_k.

Create 3 edges among them.

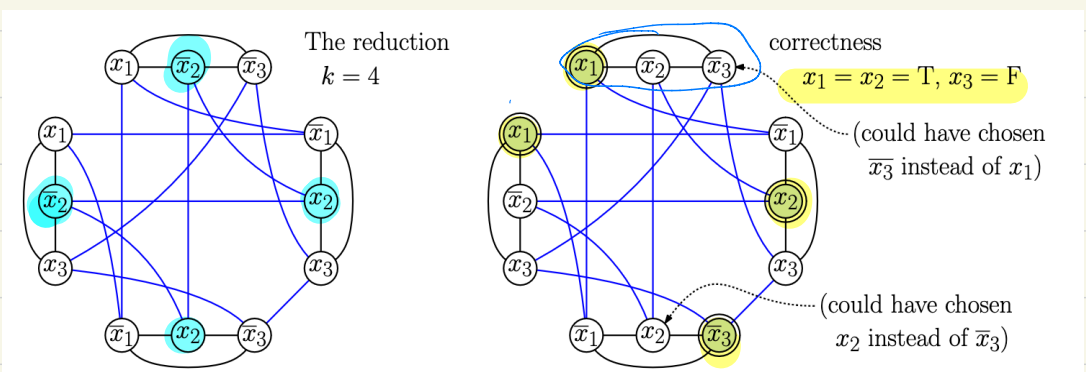
for each variable x_i,

create edges between vertices labelled x_i & vertices labelled $\overline{x_i}$

let k = # of clauses.



- f is poly-time (straight forward)
  |V| in G = 3 · # clauses.
- f is faithful, i.e. $x \in 3SAT \iff f(x) \in IS$



$$F(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$

Intuition: Finding assignment ⇒ for each clause, pick one literal, assign it to be true, and make sure no contradiction across clauses.

"$\Rightarrow$" if $F \in 3SAT$, then $\exists$ assignment C, $F(C) = True$.

- $\forall$ clause $F_1, F_2, \ldots, F_k$ in F, $F_i(C) = True \Rightarrow$ at least one literal is true.

- select one of the true literal in each clause cluster. into the ind. set. (detail that it is indeed an ind. set)
- IS of size k found.

"$\Leftarrow$" if $f(F)$ has a IS of size $\geq k$.

- each clause cluster, at most one can be included in IS
- only k such clusters.
- must select exactly one for each cluster.

- because conflict links,

  x_i $\overline{x_i}$ cannot both be in IS.

  x_i in IS $\Rightarrow$ x_i = True

  $\overline{x_i}$ ... False

  neither ... can be either.

$\Rightarrow$ the assignment makes $F(x)$ = true

(details)

$\Rightarrow$ $F \in$ 3SAT.

Prove a Prob. is NP-complete.

△ ① Prob $\in$ NP

② find a known NP-C prob. A,

prove $A \leq_p$ Prob.

<1> construct an f.

△ <2> f in poly time

<3> $x \in A \Leftrightarrow f(x) \in$ Prob.

△ $\Rightarrow$

△ $\Leftarrow$