CSCE 750
11/28/2023

NP-completeness (cont.)

Defs of P, NP, NP-hard, NP-complete

Polynomial Reductions ☆

①

Last time:

G a graph. An independent ~~sent~~ set in G is a set $I \subseteq G.V$ such that no two ~~elem~~ vertices in I are adjacent.

Note: I is an indep set iff $G.V - I$ is a vertex cover.

IS
Instance: A graph G and a ~~number~~ integer k.
Question: Does G have an independent set of size $\geq k$?

G has an indep set of size $\geq k$ iff
G has a v.c. of size $\leq |G.V| - k$

HAMILTONIAN CIRCUIT (HC)

Instance: A graph G.

Question: Is there a cycle in G that visits every vertex exactly once? (I.e., does G have a Hamiltonian circuit?)

[This is "equivalent" to VC and to IS.]

CNF-SAT (SAT = SATISFIABILITY)

<u>Instance</u>: A Boolean formula φ in conjunctive normal form (cnf).

<u>Question</u>: Is φ satisfiable?

A <u>Boolean formula</u> is a formula made from

1) variables ($x_1, \ldots, x_n$, say) taking on values 0 or 1 (<u>false</u> or <u>true</u>) (Boolean variables)

2) Logical connectives $\wedge$ (AND), $\vee$ (OR), and $\neg$ (NOT).

optional [3) Boolean constants 0 & 1]

A Bool. fmla. φ is in <u>conjunctive normal form</u> (cnf)

if $\varphi = C_1 \wedge \cdots \wedge C_k$ (<u>clauses</u> C_j for $1 \le j \le k$)

where each clause C_j is of the form

$$C_j = l_1 \vee \cdots \vee l_m \quad (l_j \text{ are } \underline{\text{literals}})$$

and each <u>literal</u> is either a variable or the negation of a variable (say x or $\overline{x}$)

same as $\neg x$

<u>Ex</u>: $\varphi = (x \vee \bar{y} \vee z) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$

A <u>truth assignment</u> is a setting of Boolean values (0 or 1) to the vars occurring in the formula.

A <u>satisfying assignment</u> is a truth assignment that makes the formula true (1).

φ is <u>satisfiable</u> if a sat. assmt. exists.

For a cnf φ, a sat. assmt. is one that makes at least one literal true (1) in every clause.

<u>Def</u>: P is the class of all decision problems that can be solved in <u>polynomial time</u>, i.e., there is an algo that on any instance returns the correct answer in time $O(n^k)$ for some constant k, where n represents the size of the instance.

[polynomial time $\equiv$ "fast" $\equiv$ "efficient"]

<u>Def</u>: Given a decision problem Π, a <u>verifier</u> V for Π is an algorithm that takes two inputs:

1) An instance x of Π
2) A string y

and either accepts or rejects, and
$V(x,y)$ runs in time polynomial in $|x|$
($O(|x|^k)$ some constant k)
and ~~behave~~ the following is true:

1) If x is a yes-instance, then there exists a string y (of length polynomial in $|x|$) such that $V(x,y)$ accepts.
2) If x is a no instance, then $V(x,y)$ rejects, for any y.

In (1), y is a proof (easily verifiable) that x is a yes-instance (y is a certificate for x)
$V(x,y)$ verifies this fact (if true).

"nondeterministic polynomial time"

Def: NP is the class of decision problems that have (poly-time) verifiers.

All problems mentioned so far are in NP.

Ex: CNF-SAT: For satisfiable φ, a proof y making $V(\varphi,y)$ accept would be a satisfying assignment.

Fact ("easy"): $P \subseteq NP$.

Pf: Given $\Pi \in P$ let A be an algo deciding Π in polytime. A verifier V for Π:

$V(x,y)$: Ignore y and run A on input x,
If A says "yes" then accept
else reject.

Shows that $\Pi \in NP$. $\blacksquare$

Notorious Open: $NP \subseteq P$? [equiv. is $P = NP$?]

General belief: $P \neq NP$.

Good ~~of~~ evidence: There are so many different types of decision problems in NP that nobody has found any ptime decision algo for any ~~of~~ of them despite intense study for (literally) millenia.

Def: A polynomial reduction from decision problem Π_1 to decision problem Π_2 is a ptime algorithm f that takes an instance of Π_1 as input and outputs an instance of Π_2 with the same answer (no-instance $\mapsto$ no-instance, yes-instance $\mapsto$ yes-instance).

$x \in \Pi_1$ has answer "yes" iff $f(x) \in \Pi_2$ has answer "yes"

Say $\Pi_1 \leq_p \Pi_2$ if such an f exists. (Π_1 is <u>p-time reducible to Π_2</u>)

<u>Prop</u>: Let f be a p-reduction from Π_1 to Π_2.

1) If $\Pi_2 \in P$ then $\Pi_1 \in P$

2) If $\Pi_2 \in NP$ then $\Pi_1 \in NP$

<u>Proof</u> (next time)