Recall:
A decision problem is a collection of finite inputs (instances) where each instance is either a yes-instance or a no-instance. (Binary output: "yes" or "no")

Π decision problem.
Say that an algo A solves Π if A correctly outputs "yes" or "no" given any instance of Π.

CLIQUE
Instance: A graph G and an integer K.
Question: Does G have a clique of size $\geq K$?

INDEPENDENT SET (IND SET)
Instance: A graph G and an integer K.
Question: Does G have an independent set of vertices of size $\geq k$?
($C \subseteq G.V$ is independent if there are no edge in $G.E$ connecting vertices in C.)

VERTEX COVER (VC)
Instance: A graph G and an integer K.
Question: Does G have a vertex cover of size $\leq K$?
($C \subseteq G.V$ is a vertex cover iff every edge in $G.E$ has at least one endpoint in C.)

SAT (SATISFIABILITY)
Instance: A Boolean formula φ over boolean variables $x_1, \ldots, x_n$.
Question: Is there a truth setting of the variables $x_1, \ldots, x_n$ that makes φ true? (Is φ satisfiable?)
$x_1 \vee (x_2 \wedge \overline{x_1})$ yes
$x_1 \wedge (x_2 \vee \overline{x_1})$ yes

CNF-SAT
Instance: A Bool fmla φ in conjunctive normal form (cnf).
Question: Is φ satisfiable?
cnf: $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_k$ (clauses)
each clause $C = l_1 \vee l_2 \vee \cdots \vee l_n$ (literals)
each literal is either a variable or the negation of a variable.
$\varphi = (\overline{x_1} \vee x_2) \wedge (\overline{x_2} \vee x_3) \wedge (\overline{x_3} \vee \overline{x_1})$
[restriction of SAT: each instance is also an instance of SAT, question follows]

3SAT
Instance: Boolean fmla φ in cnf with exactly 3 literals in each clause.
Question: Is φ satisfiable?
[restriction of CNF-SAT]

Recall:
An algo A runs in polynomial time (ptime) if there is a constant k such that A runs in time $O(n^k)$, where n is the size of the input. ("fast", "efficient")

Π_1, Π_2 decision problems.
A polynomial reduction from Π_1 to Π_2 is a ptime computable function f such that, for every instance x of Π_1, $f(x)$ is an instance of Π_2 with the same answer
{ x is no-instance $\Rightarrow f(x)$ no-inst
" " yes " $\Rightarrow f(x)$ yes-inst
If such an f exists, then Π_1 is polynomially reducible to Π_2
($\Pi_1 \propto \Pi_2$, $\Pi_1 \leq_P \Pi_2$)
Spse $\Pi_1 \propto \Pi_2$ and you have a ptime algo to solve Π_2. Then, very
Contrapositively, if Π_1 is hard (no ptime algo for Π_1) then Π_2 is also hard.

Nov 29-12:29 PM

Polynomially reduce CNF-SAT to 3SAT.

Input: Bool fmla φ in cnf

Goal: Transform φ into some φ' (instance of 3SAT) such that

φ is satisfiable $\iff \varphi'$ is satisfiable

[transformation must be ptime]

Arbitrary cnf fmla φ.

$\varphi = C_1 \wedge \cdots \wedge C_k$ for some k

each $C_i = l_{i_1} \vee l_{i_2} \vee \cdots \vee l_{i_m}$ literals

Construct φ' by replacing each clause of φ. Let C be a clause of φ.

$C = l_1 \vee \cdots \vee l_a$

Case 1: $a=3$. C is a clause of φ' unchanged

Case 2: $a=2$. $C = l_1 \vee l_2$

Introduce a fresh variable x occurring nowhere else

replace C with 2 clauses:

$(l_1 \vee l_2 \vee x) \wedge (l_1 \vee l_2 \vee \bar{x})$

Case 3: $a=1$. $C = l$

Introduce two fresh vars x, y

Replace C with

$(l \vee x \vee y) \wedge (l \vee x \vee \bar{y}) \wedge (l \vee \bar{x} \vee y) \wedge (l \vee \bar{x} \vee \bar{y})$

Case 4: $a \geq 4$.

$C = l_1 \vee l_2 \vee \cdots \vee l_m \quad (m \geq 4)$

Replace C with these clauses:

$(l_1 \vee x_1) \wedge (\bar{x}_1 \vee l_2 \vee x_2) \wedge (\bar{x}_2 \vee l_3 \vee x_3) \wedge \cdots \wedge (\bar{x}_{m-1} \vee l_m)$

ith clause looks like

$(\bar{x}_{i-1} \vee l_i \vee x_i) \quad 1 < i < m$

$x_1, \ldots, x_{m-1}$ are all fresh vars.

Spse l_i is true. Then set

$x_1, \ldots, x_{i-1}$ true

$x_i, \ldots, x_{m-1}$ false

} all replacement clauses satisfied

Converse: If all replacement clauses are satisfiable, then so must C be. equiv. If C not satisfied, then no way to satisfy all replacement clauses.

Suppose then that all $l_1, \ldots, l_m$ are all false.

$(l_1 \vee x_1)$ ← must set $x_1 :=$ true to satisfy

$(\bar{x}_1 \vee l_2 \vee x_2)$ ← must set $x_2 :=$ true to satisfy

$(\bar{x}_{i-1} \vee l_i \vee x_i)$ ← must set $x_i :=$ true to satisfy

$(\bar{x}_{m-1} \vee l_m)$ ← can't satisfy!

Reduction replaces each clause of φ as above to get φ'

φ is sat $\iff \varphi'$ is satisfiable

this reduction is "obviously" ptime (in fact linear-time).

On Thursday: 3SAT $\propto$ VC

Def: A decision problem is easy if there is some ptime (fast) algo that solves it.

P is the class of all easy decision problems

(P is a "complexity class")

NP is the class of all decision problems whose yes-instances are easily verifiable, given extra information ("proof" or "witness")

SAT $\in$ NP

Nov 29-1:07 PM