P = class of decision problems <u>solvable</u> in ptime

NP = class of decisions problems <u>verifiable</u> in ptime (yes instances only)

Open question: $P \stackrel{?}{=} NP$

$P \subseteq NP$.

Wide belief: $P \neq NP$

<u>Reduction</u>: $\Pi_1 \propto \Pi_2$

if you can easily map any instance of Π_1 into an an instance of Π_2 with the same answer.

<u>Fact</u>: If $\Pi_1 \propto \Pi_2$ and $\Pi_2 \in P$, then $\Pi_1 \in P$.

Equiv: ~~If~~ $\Pi_1 \propto \Pi_2$ and $\Pi_1 \notin P$, then $\Pi_2 \notin P$.

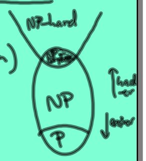
<u>Def</u>: A problem Π is <u>NP-hard</u> if every $\Pi' \in NP$ polynomially reduces to Π.
(Π is at least as hard as any problem in NP)

If Π is NP-hard and $\Pi \in P$, then $P = NP$.

<u>Tool</u>: Showing a problem to NP-hard is good evidence that the problem has no easy solution.

<u>Def</u>: Π is <u>NP-complete</u>

If <u>$\Pi \in NP$ and</u> Π is NP-hard



<u>Thm</u> (Cook, Levin)
CNF-SAT is NP-complete.

~~Want~~ Given some NP problem Π. We want evidence that Π is hard. We can show that Π is NP-complete (maybe).

To show NP-completeness, find a known NP-complete problem Π' and a polynomial reduction from Π' to Π.

Reducibility is transitive:

$\Pi_1 \propto \Pi_2 \propto \Pi_3 \Rightarrow \Pi_1 \propto \Pi_3$

Any $\Pi'' \in NP$

$$\Pi'' \propto \underset{\substack{\uparrow \\ \Pi' \text{ is NP-complete}}}{\Pi'} \propto \Pi$$

now we know that Π is NP-complete

CNF-SAT is NPC (Cook, Levin)
Tuesday: CNF-SAT $\propto$ 3SAT
<u>3SAT is NPC</u> (3SAT $\in$ NP)

VC: Given graph G & integer K, does G have a v.c. of size $\leq K$?

Reduce 3SAT to VC
3SAT $\propto$ VC

Given a 3SAT instance
$\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$
each clause has 3 literals (exactly)
Let $x_1, x_2, \ldots, x_n$ be the vars occurring in φ.

Build the following graph:

1) For each variable x_i $(1 \leq i \leq n)$ we have a graph

n many of these, disjoint "truth-setting components"

So far:

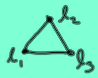
<u>Note</u>: An v.c. must include ≥ 1 vx from each truth-setting component.

2) Satisfaction testing components:

Dec 1-12:28 PM

For each clause C_j $(1 \le j \le m)$
add a triangle:

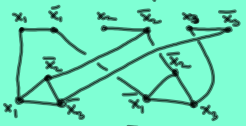
$C_j = l_1 \vee l_2 \vee l_3$



3) Connect via an edge, each literal in each triangle with the identically-labeled vx in the truth-setting $\underline{\underline{\text{components}}}$ graph

Output this graph & $K = \underset{\substack{\uparrow \\ \#\text{vars}}}{n} + 2\underset{\substack{\uparrow \\ \#\text{clauses}}}{m}$.

$\varphi = (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3)$



$K = 3 + 2 \cdot 2 = 7$

<u>Show:</u> graph G_φ. φ is satisfiable iff G_φ has a v.c. of size $\le \underline{n+2m}$

Any such v.c. must have exactly one vx of each truth-setting component and exactly 2 vx's from each satisfaction-testing component.

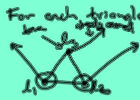
($\Longrightarrow$) Suppose φ is satisfiable. Let $a_1, a_2, \ldots, a_n$ be a satisfying truth assignment for φ.

[I mean: $a_1, \ldots, a_n \in \{0,1\}$]
and setting $\left.\begin{array}{l} x_1 := a_1 \\ x_2 := a_2 \\ \vdots \\ x_n := a_n \end{array}\right\}$ makes φ true.

Here is a vx cover C of G_φ of size $n+2m$:

a) From each truth-setting component, add the true literal to C (under the satisfying assignment)
n vertices added to C
$2m$ left to add.

b) For each triangle,



If, say, l_3 is true, then its connecting edge is already covered at the other end.

we can afford to leave l_3 out of C.
Put l_1, l_2 into C, covering their connecting edges as well.

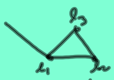
For each C:
$n+2m$ vertices that cover every edge. ($\Longrightarrow$ proved)

($\Longleftarrow$) Suppose G_φ has a v.c. C of size $n+2m$. [Show that φ is satisfiable)

Know that C has exactly one vx in each t.s. component and exactly 2 in each triangle.

Consider the truth assignment that makes each literal true iff it is in the cover (in the truth-setting component)

Consider any clause $C_j = l_1 \vee l_2 \vee l_3$



Know that exactly one of these is not in C. Say l_1.
Then C must cover l_1's connecting edge at the other end.
$\therefore$ l_1 is true
Each clause is satisfied by this assignment. $\blacksquare$

<u>INDEPENDENT SET</u> (IS)

Given graph G and integer K, does G have an i.s. of size $\ge K$

Reduce VC $\propto$ IS.
$\therefore$ IS is NP-hard.

Dec 1-12:58 PM

Fact: G graph.
$C \subseteq G.V.$
C is a vertex cover of G
iff $G.V - C$ is an independent set.

Give instance (G, K) of VC,
output $(G, |G.V| - K)$.

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

iff $(G, |G.V| - K)$ is a yes-instance of IS.

Reduce IS $\propto$ CLIQUE

Def: Let G be a graph.
$\overline{G}$ (complement of G)
is the graph with the same set of vertices as G, and
for any $u, v \in G.V$, $u \neq v$,
$\{u, v\} \in \overline{G}.E$ iff $\{u, v\} \notin G.E$

Map instance (G, K) of IS
to instance $(\overline{G}, K)$ of CLIQUE
G has an i.s. of size $\leq K$ iff
$\overline{G}$ has a clique of size $\leq K$
(same set).

Resources
- Chap 34 of CLRS
- Garey & Johnson
"Computers & Intractability: A Guide to the Theory of NP-completeness"
- Look at past Qualifying exams in Algorithms (from my home page)