Complexity theory

Decision problem – answers yes/no question

Example: "find maximum size of an independent set in a graph" (search problem)
"is there an independent set of the size $\geq k$" (decision problem)

Size of input – number of bits we need to write the problem input
(effectively proportional to the dimension/size of the problem)

Class P – all decision problems that can be solved in running time $p(n)$
polynomial
size of input

Class NP – all decision problems that have a certificate that can be checked in $p(n)$ time

- What is a certificate?
  e.g. "an answer": the vertices that give the biggest independent set
  [it can be verified quickly but not necessarily constructed]
- $P \subset NP$: an algorithm itself is a certificate

Is $P = NP$? Not known (but not likely :))

Assuming $P \neq NP$, here is the notion of hardness we will look for:

$\boxed{\text{a problem is at least as hard as problems known to be in NP-class}}$

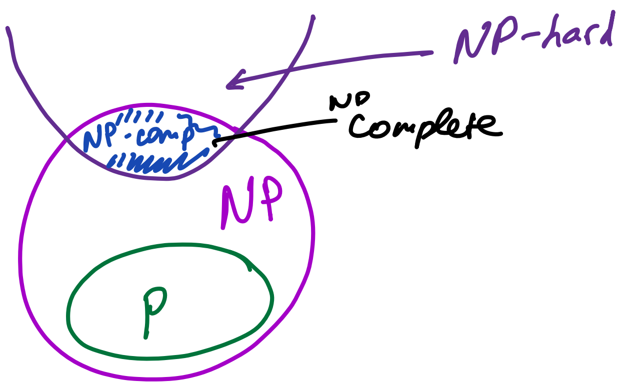
(NP) hard problem

Polynomial time reduction: $A \rightsquigarrow B$ or $A \leq_p B$

A can be reduced to B if arbitrary instances of problem A can be solved using
- polynomial number of steps +
- polynomial number of calls to a problem from B

- A is reducible to B means "A is not harder than B"

Class NP-hard - problem (X) such as there is a reduction from an NP-hard problem to (X).



What does this imply?

$(K) \le_p (X)$

known to be NP-hard (at least as hard as all problems in NP)

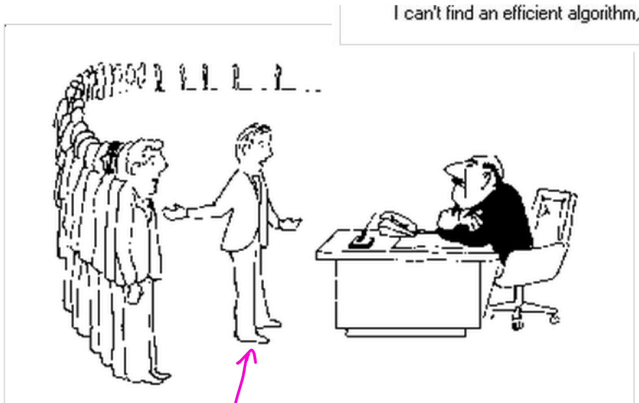
comes from "the first NP-hard problem"

If there is a poly algorithm for (X) $\Rightarrow$ there is a poly algorithm for any NP-problem $\Rightarrow$ P=NP

The value of reductions

I can't find an efficient algorithm, I guess I'm just too dumb.

I can't find an efficient algorithm, because no such algorithm is possible



I can't find an efficient algorithm, but neither can all these famous people.

PRINCETON UNIVERSITY

[AAA slides]

[Garey, Johnson]

If we can solve this problem polynomially, then we can solve all NP-problems polynomially (P=NP)

Reductions can be used to show that a problem is "easy" (in Ⓟ)

Example: LP - known to be in Ⓟ

MAX FLOW: input: directed graph with rational weights on edges (capacities) with selected vertices S (sourse) and T (target)

question: is there a flow of value $\geq k$?

[assignments of nonnegative flow values at the edges not above the capacities, so that inflow and outflow is the same for every vertex except S and T]

Claim: MAXFLOW can be formulated as an LP feasibility problem

$$\begin{cases} \sum\limits_{v:\, S \to v} f(\overset{\text{sourse}}{S}, v) \geq k \\ \sum\limits_{u \to v} f(u,v) = \sum\limits_{v \to w} f(v,w) \\ 0 \leq f(u,v) \leq c(u,v) \end{cases}$$

So, any instance of MAXFLOW is a particular instance of LP

We can solve any MAXFLOW by solving an LP $\Rightarrow$ MAXFLOW $\leq_p$ LP $\Rightarrow$ MAXFLOW is in Ⓟ

(MAXFLOW reduces to LP)

$\text{MINCUT}_{S\text{-}T}$: input: the same

q: is there a partition into 2 sets S_1 and S_2 so that $S \in S_1$, $T \in S_2$ so that total capacity of the edges between S_1 and S_2 $\leq k$?

Exercise: • $\text{MINCUT}_{S\text{-}T} \leq_p$ MAXFLOW

· MINCUT $\leq_p$ $\text{MINCUT}_{S\text{-}T}$

(via polynomially many calls to the instance of class $\text{MINCUT}_{S\text{-}T}$)

Reductions to show that a problem is hard.

Gameplan: find an NP-hard problem (K), so that we can solve (K) by solving instances of (X) poly times ((K) $\leq_p$ (X))

What are the problems from NP-hard?

First examples

Cook-Levin theorem

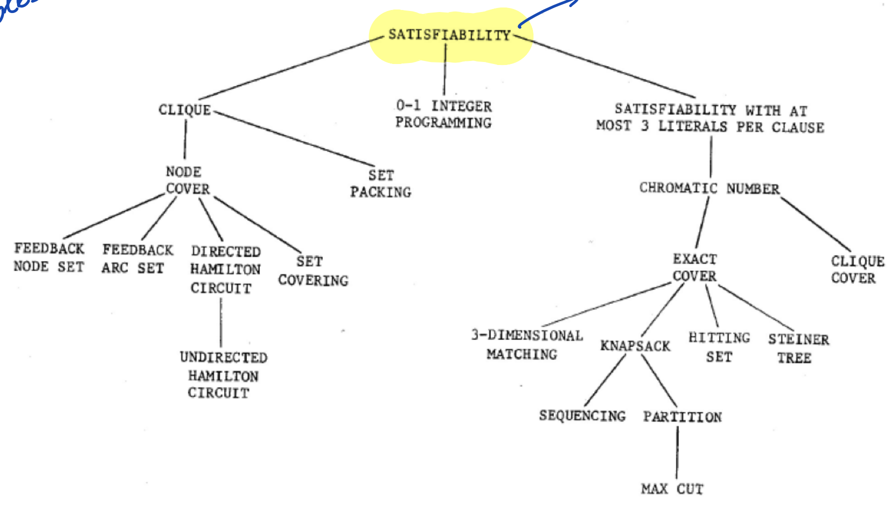


FIGURE 1 - Complete Problems

SAT (Satisfyiability):

input: a boolean formula in a normal form

question: is there a 0-1 assignment that satisfies the formula?

clauses

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

literals

x, y, z - variables 0-1

$x \vee y$ is OR
- 0 0 → 0
- 0 1, 1 1, 1 0 → 1

$x \wedge y$ AND
- 0 0, 0 1, 1 0 → 0
- 1 1 → 1

$\bar{x}$ - NOT
- 0 → 1
- 1 → 0

$(1 \vee 1 \vee 0) \wedge (1 \vee 0) \wedge (1 \vee 1) \wedge (0 \vee 0 \vee 1) = 1 \rightarrow$ satisfied

(every clause must be evaluated to 1)

0-1 INT

$SAT \leq_p$ 0-1 INT

Take an SAT instanse.

$$\begin{cases} x \vee y \vee z = 1 \\ x \vee \bar{y} = 1 \\ y \vee \bar{z} = 1 \\ \bar{x} \vee \bar{y} \vee \bar{z} = 1 \\ x, y, z \in \{0,1\} \end{cases}$$

$x + y + z \geq 1$

$x + (1-y) \geq 1$

3-SAT

every clause has exactly 3 members, clearly $3\text{-}SAT \leq_p SAT$

other direction? SAT

$$\varphi = (x \vee y \vee z) \wedge (x \vee \bar{y}) \wedge (y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee w)$$

$$(x \vee \bar{y} \vee z) \wedge (x \vee \bar{y} \vee \bar{z}) = 1$$

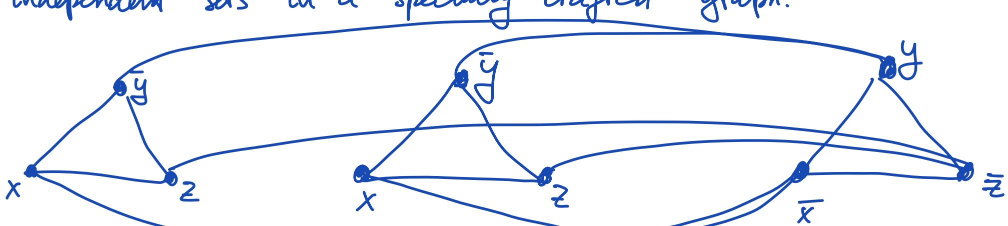
INDEPENDENT SET

$3SAT \leq_p$ IND SET

Take any 3SAT input

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

We will solve its satisfiability via detecting independent sets in a specially crafted graph:



Test: is there an independent set of size $\geq k$? (k clauses)

Yes $\iff \varphi$ is satisfiable.

Q: Can we go away with regular SAT here?

Feasibility of quadratic optimization

IND SET $\leq_p$ FEAS-QUAD

Consider arbitrary G $\quad \alpha(G) \geq k$

$$\Updownarrow$$

$$\begin{cases} \sum_{i=1}^{n} x_i - k = s^2 \\ x_i x_j = 0 \quad i \leftrightarrow j \\ x_i(1-x_i) = 0 \quad i=1,\dots,n \end{cases} \quad \text{is feasible}$$

Polynomial positivity

3SAT $\leq_p$ POLYPOS (deg 6)

Given a polynomial, $p(x)$ is there an x $p(x) \leq 0$?

Take any 3SAT input

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

$$\begin{aligned} p(x) = \; & [x(1-x)]^2 + [y(1-y)]^2 + [z(1-z)]^2 \\ & + [(x+y+z-1)(\cancel{x+y+z}-2)(\cancel{x+y+z}-3)]^2 \\ & + [(x+(1-y)+z-1)(\cancel{x+(1-y)+z}-2)(\cancel{x+(1-y)+z}-3)]^2 \\ & + [((1-x)+y+(1-z)-1)(\cancel{(1-x)+y+(1-z)}-2)(\cancel{(1-x)+y+(1-z)}-3)]^2 \end{aligned}$$

Right assignment $= 0$

So, testing positivity of this polynomial = testing satisfiability

degree 6

Can we get lower degree?

Idea: each clause can be evaluated to at most 1.

1-in-3 SAT

Is there an assignment of 3SAT so that exactly 1 literal in each clause evaluates to 1?

$\hookrightarrow$ 3SAT $\leq_p$ 1-3SAT

Take any 3SAT input

$\varphi = \underbrace{(x \vee y \vee z)} \wedge (x \vee \bar{y} \vee z) \wedge (y \vee \bar{z} \vee \bar{x})$

Need to rewrite each clause so that it is 1-in-3 satisfiable if and only if the original one is satisfiable

$(x \vee y \vee z)$

$\overbrace{(\bar{x} \vee a \vee b) \wedge (b \vee y \vee c) \wedge (c \vee d \vee \bar{z})}$

000 1 0 1 $\Rightarrow$ $a, b, c, d = 0$ $\Rightarrow$ middle clause is 0
not 1-3 SAT

all other cases are fine
(either b or c can be ~~1~~ as $y = 1$)

1-In-3 SAT $\leq_p$ POLYPOS dep 4.