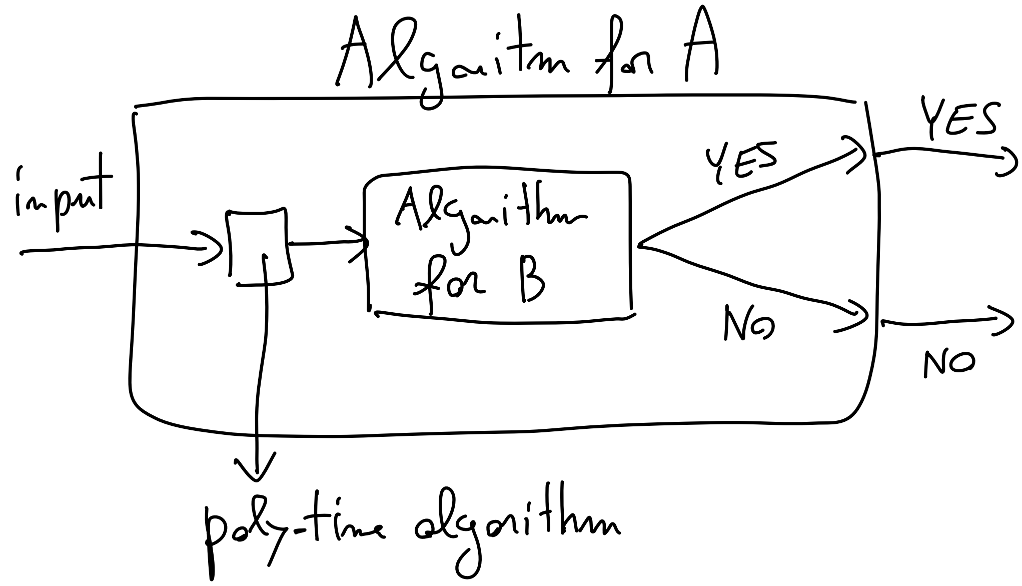
Definition: a problem A <u>reduces in polynomial time</u> to problem B ($A \leq_p B$) if $\exists$ a polynomial-time algorithm that transforms an arbitrary input instance a of A into an input instance b of B such that

usually called "Karp reduction"

1) a is a YES instance of A $\Rightarrow$ b is a YES instance of B

2) b is a YES instance of B $\Rightarrow$ a is a YES instance of A



Observation: it's more restrictive than the general scheme (only one call to B, no postprocessing, only deals with decision problems)

Property: $A \leq_p B$ and $B \leq_p C \Rightarrow A \leq_p C$

NP-hardness (formal definition): a problem is <u>NP-hard</u> if every problem in NP reduces in polynomial time to it.

Then, to prove that a problem X is NP-hard reduce a known NP-hard problem Y to X

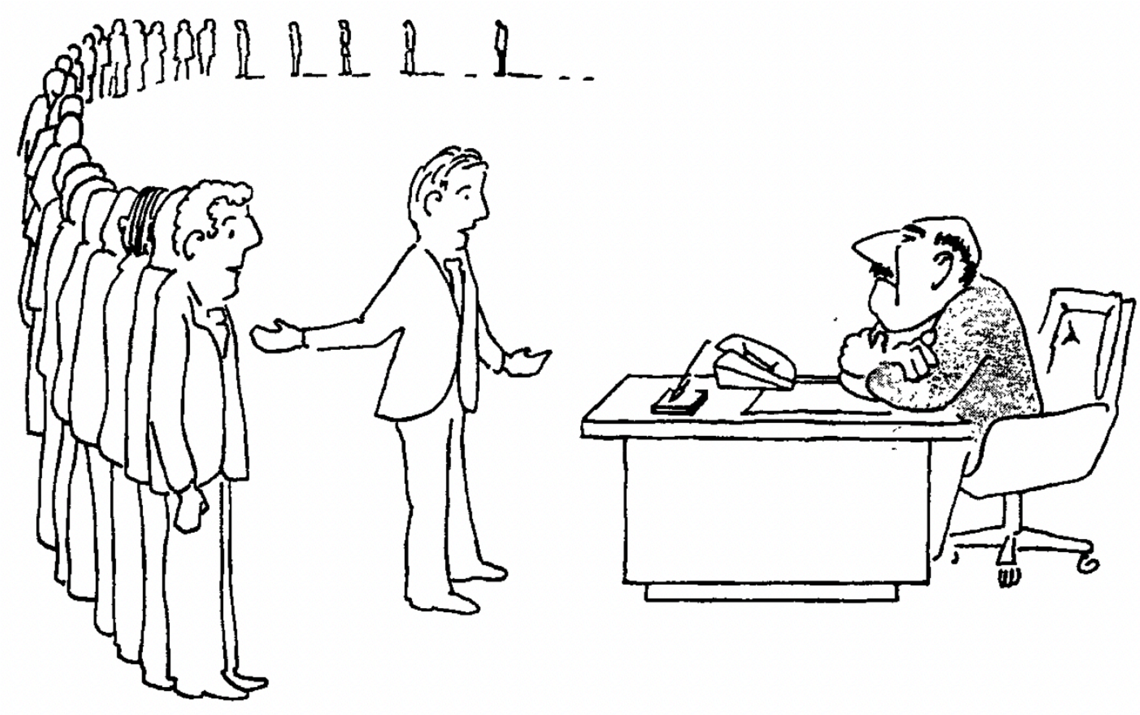
Let's emphasize: the reduction is FROM Y TO X

I already know it's NP-hard

the "new" problem

rookie mistake: do a reduction in the wrong direction

Again, NP-hardness doesn't mean the problem is not in P, but it does provide strong evidence for that:



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

(picture from the book of Garey & Johnson, 1979)

Library of NP-hard problems:

3-SAT

one of the 21 problems shown NP-hard by Karp '72

Hamiltonian Circuit

Our first NP-hardness proof:

Theorem: TSP is NP-hard

Proof: reduction from Hamiltonian Circuit to TSP

$\text{Ham} \leq_p \text{TSP}$

Wait a minute: TSP is not a decision problem!

No problem:

TSP: input: $G = (V, E)$ complete, undirected, weighted

$K \in \mathbb{R}$

output: $\exists$ in G a Ham. Circuit of cost $\leq k$?

Pick an arbitrary input instance $G = (V, E)$ for Ham. Circuit; create the following input for TSP:

$G' = (V, E')$ complete, undirected, weighted

with

$$w(e \in E') = \begin{cases} 1 & \text{if } e \in E \\ +\infty & \text{otherwise} \end{cases}$$

$$K = n$$

this reduction takes poly-time ($O(n^2)$)

Then

1) if G has a Ham. circuit, then the TSP algorithm run on G' returns a Ham. circuit of cost n

2) if G doesn't have a Ham. circuit, then any Ham. circuit in G' must have ≥ 1 edge not in G $\Rightarrow$ in this case a TSP alg. run on G' returns a Ham. circuit of cost $> n$

More problems

Definition: given a graph $G = (V, E)$, an <u>independent set</u> in G is a subset $I \subseteq V$ with no edges between them.

<u>(Maximum) Independent Set problem</u>: compute an ind. set of maximum size.

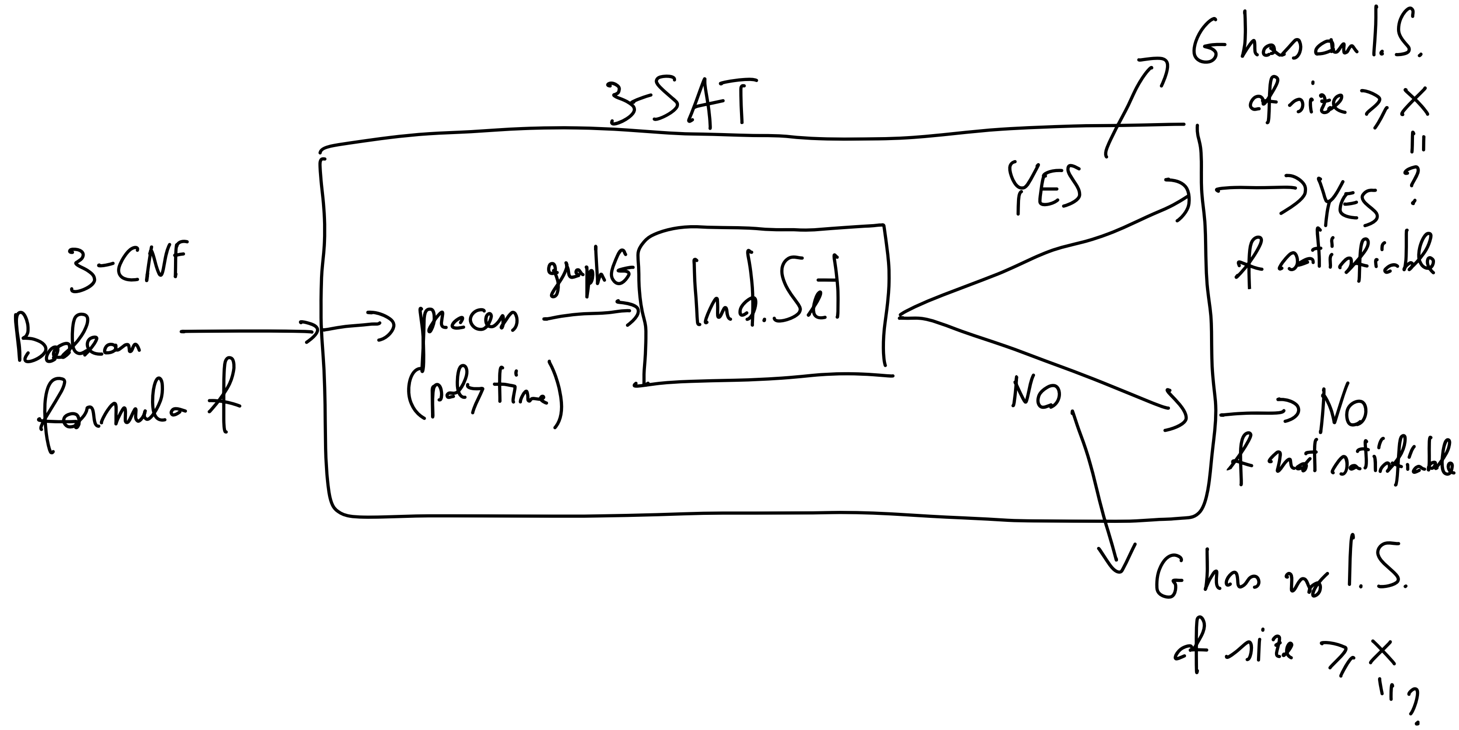
Theorem: Independent Set is NP-hard

Proof: reduction from 3-SAT to Ind. Set

3-SAT ↓ logic

Ind. Set ↓ graphs

they seem totally unrelated problems!

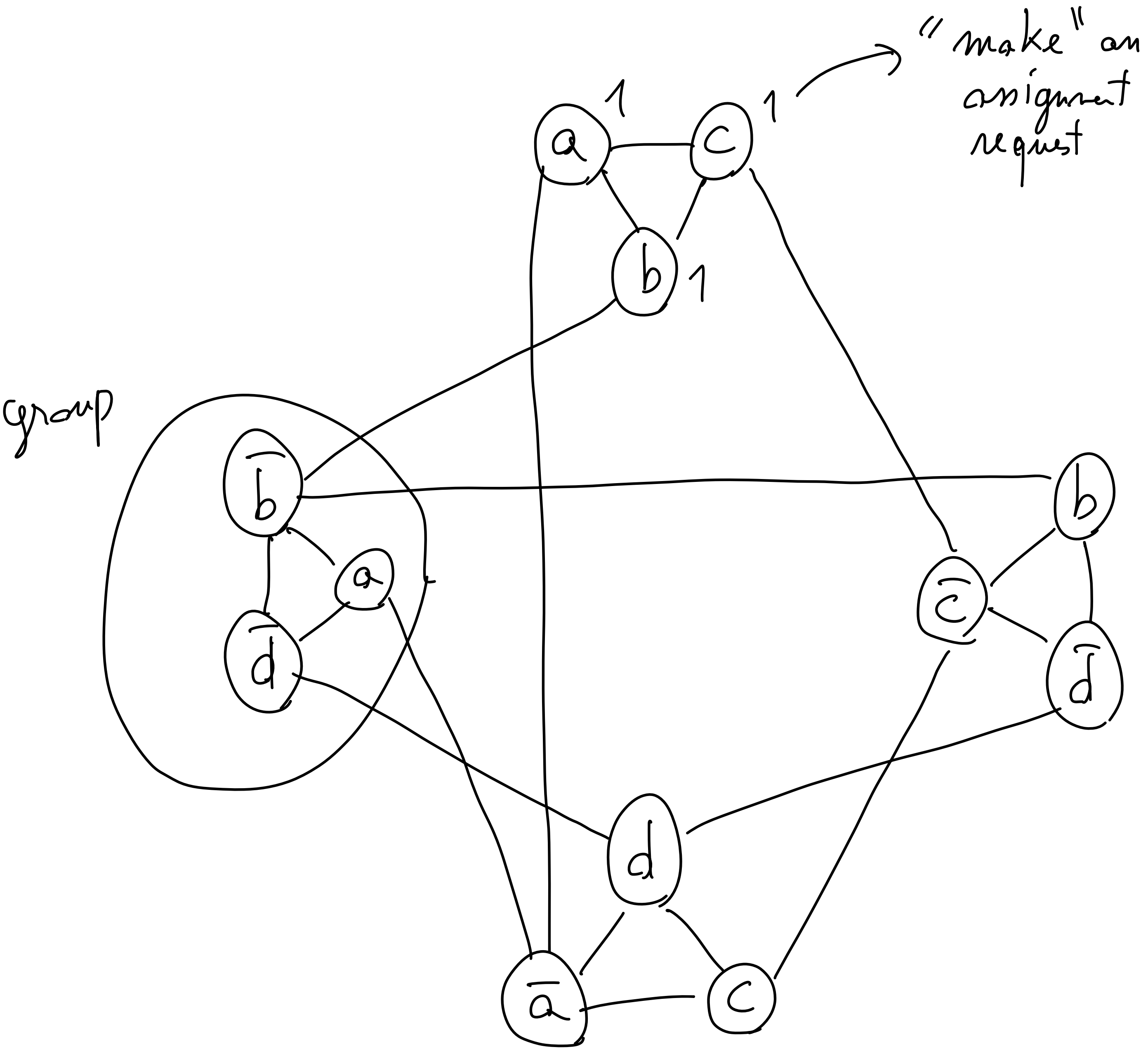


Let's see the main ideas, step by step

- pick an arbitrary 3-CNF Boolean formula f with K clauses

$$f = (a \vee b \vee c) \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{b} \vee \bar{d})$$

- vertices : each vertex represents one literal in f



- edges :

1) idea : ind. set represents <u>conflicts</u> $\Rightarrow$ add an edge between every pair of vertices making requests that are inconsistent

(asking for opposite assignments to the same variable)

Obs.: an ind. set with ≥ 1 vertex in each group gives a satisfying truth assignment $\rightarrow$ should look for ind. sets of size $\geq K$ to say "YES, f is satisfiable"

Issue: an ind. set now is free to choose <u>multiple</u> vertices from a group

$\Rightarrow$ I might output "YES, f is satisfiable" even if this is not true!

$\Rightarrow$ idea: force the choice of exactly <u>one</u> vertex per group $\Rightarrow$

$\Rightarrow$ 2) add one edge between every pair of vertices that are in the same group.

Claim: G contains an ind. set of size K

$\iff$ the formula f is satisfiable