3/18/2019

Computational Complexity Theory

[Chapter 8,
K-T 2004]

Have you seen the question:
Is $P \neq NP$?

(a) Yes (b) No.

→ Seen poly time algorithms for variety problems -

→ Seen reduction [implicitly]

(i) Reduce problems to Computing Max-flows in networks.

(ii) Dynamic prog → Shortest path problems.

We will study 'hardness' of solving Computational problems.

Decision Problems.

Problems that admit 'Yes' or 'No' as the only valid outputs.

e.g (i) Shortest-Path : Input: (G, s, t, k)

integer

vertices

Poly time ✓

Output: Yes if there is an s-t path of length at most k.

(ii) Max-Flow : (G, l)

integer

flow network

Poly time ✓

Output : Yes if Max-flow$(G) \geq l$

No o.w.

(iii) Independent Set : (IS) (G, k)

Recall, Def^n: S is an indep set of G if there exists no edges between vertices in S.

Running time ??

Output: Yes if G has an IS

of size at least K.
No o\w.

Polynomial time Karp Reductions.

Let A, B be decision problems.

I_A: be the input instances of A
I_B: " " " " " B

We say $A \leq_p B$ [read: A is Polynomial time Karp reducible to B] if:

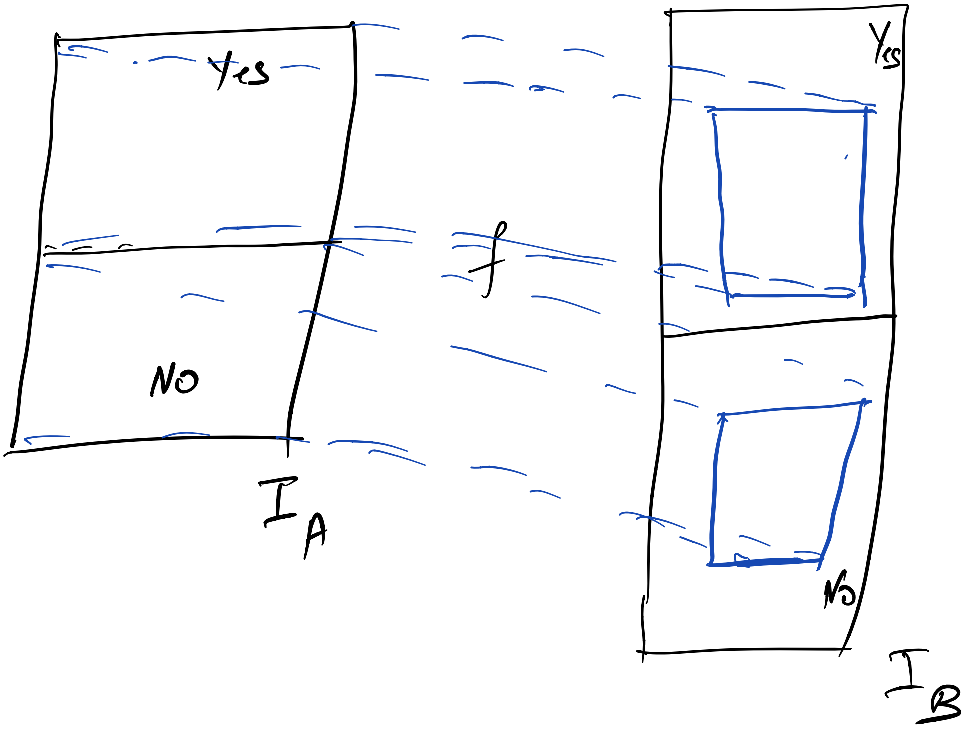
$\exists$ a polynomial time algorithm f that maps I_A to I_B s.t:

(i) if $x \in I_A$ s.t x is a 'Yes' instance of A,

$\Rightarrow f(x) \in I_B$ is a 'Yes' instance of B.

(ii) if $x \in I_A$, x is a 'No' instance of A

$\Rightarrow f(x) \in I_B$ is a 'No' instance of B.



Discussion: Suppose $A \leq_p B$.

(i) If B can be solved in polynomial time, then A can be solved in " ".

(ii) If A is a 'hard problem'

'unlikely to have an efficient solution'.

then B is a 'hard problem'.

Candidate hard problem: Boolean satisfiability.

$x_1, x_2, \ldots, x_n$: variables.

$\neg, \vee, \wedge$: gates.

$\forall i$, $\neg x_i$ is denoted by : $\bar{x}_i$

$\{x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$: literals.

Clause: OR of literals.

eg. $x_1 \vee \bar{x}_2 \vee \bar{x}_5$. ✓

Conjunctive Normal Form (CNF)

Defn: AND of clauses.

eg. $\phi : (x_1 \vee \bar{x}_2 \vee \overset{1}{x_3}) \wedge (\overset{1}{\bar{x}_1} \vee x_2 \vee x_3)$

$\wedge (\overset{1}{\bar{x}_1} \vee x_2 \vee \bar{x}_3)$.

satisfying assignment : $x_1 = 0$, $x_3 = 1$, $x_2 = 0$

Def^n: (K-CNF) : CNF where each clause has exactly K literals.

0 → False
1 → True

Def^n: ϕ is satisfiable if there exists an assignment of $\{0,1\}$-values to the variables s.t ϕ evaluates to 1 (True).

3SAT : Input: ϕ is 3CNF on variables $x_1, \ldots, x_n$.
Output: Yes if ϕ is satisfiable
No o\w.

Assumption: 3SAT is 'hard to solve'.
3SAT is NP-hard.

Our first Reduction.

Goal: Prove IS is NP-hard.
Method: Reduce 3SAT to IS.

IS : (G, k) Yes if G has an indep. set of size

at least K .
NO o/w .

Thm: $3SAT \leq_p IS$.

Pf:

$$\phi \xrightarrow{f} (G_\phi , K)$$

3CNF

ϕ is satisfiable then G_ϕ has IS of size $\geq K$

ϕ is not satisfiable, then largest IS of G_ϕ is of size $< K$.

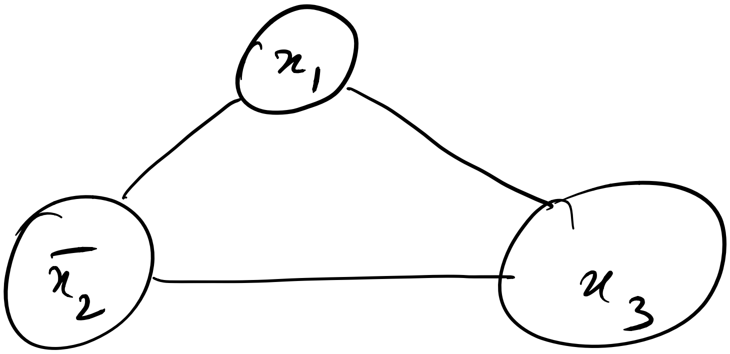
$$\phi : C_1 \wedge C_2 \wedge ---- \wedge C_\ell .$$

$$C_i = x_1 \vee \overline{x_2} \vee x_3$$

For each variable x_i :

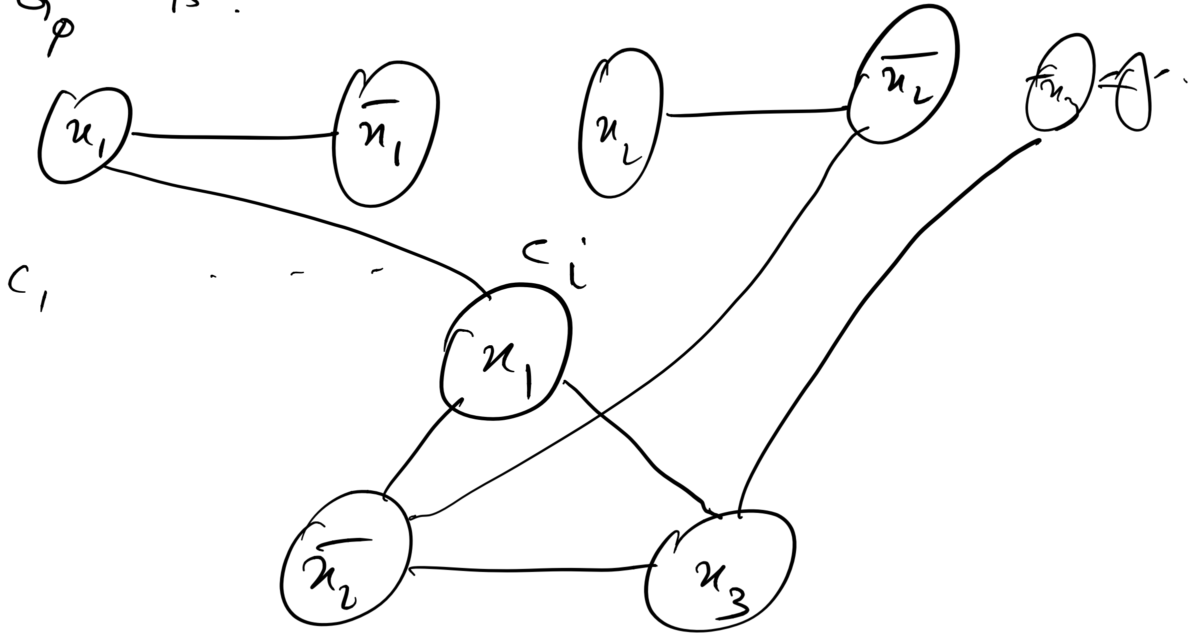
gadget

For each clause, C_i :



given ϕ ;

G_ϕ is :



<u>Lemma</u>: ϕ is satisfiable iff

G_ϕ has an indep set of size $n+l$.

$$\phi \xrightarrow{f} (G_\phi , n+l)$$