17 March 2025	NP-Haird	Problems
		· · · · · · · · · · · · · ·
· · · · · · · · · · · · · · · · · · ·		
Plan		
× NP-Handness		
* Announcements		
X NP-Hard Problems		
		· · · · · · · · · · · · · · ·
* Independent Set		
· · · · · · · · · · · · · · · · · · ·		
		· · · · · · · · · · · · · · ·

· ·		1 _0U	νF			;+	 	•		1 7 h	م کرم			•	• •	•	•		•	· ·	•	•	••••	•	• •		• •	· ·	•		• •	•	•	· ·	•	•
· ·		•		ر مر	-e	ہ در د	, 1 , √ , √	, 1 Z	e_	•)v.c	<u>, b</u>	Lei	 	کې د	•	60	ر ج ر	ed	•		, , , , , , ,	1	~~~ ~~~	ہ مر ر		Ēŀ	4 S	Ŷ		H،	A R	D.	•	•
· ·	•	•	• •	• •	•	•	• •		•	•	• •	•	•	•	• •	•	•	•	•	• •	•		th	د میرد	 1 .	c C	NR	· ·	Ţ	0.7	•••	Sc	ہ∫` ۔ `	e	• .	•
	•	•	• •		•	•	• •	•	•	•	• •	•	•	•	• •	•	•	•	•	• •	•	•	• •	•	• •	•	• •		•	•	• •	•	•	• •	•	•
· ·	•	•	• •		•	•	• •	•	•		• •	•		а Ф	• •	•	•	•		• •	•	•	· ·	•	· ·	•	• •	• •	•	•	• •	•	•	• •	•	•
· ·	•	•	· ·	• •	•	•	• •		•	•	• •			•	• •	•	•	•	•	· ·	•	•	· ·	•	· ·		• •	• •	•	•	• •		•	· ·		•
• •	•	•	• •	•	•	•	• •	•	•	•	• •	•	•	•	• •	•	•	•	•	• •	•	•	· ·	•	• •	•	• •	• •	•	•	••••	•	•	• •	•	•
• •	•	•	• •	• •	•	•	• •	•	•	•	• •	•	•	•	• •	•	•	•	•	• •	•	•	· ·	•	• •	•	• •	• •	•	•	••••	•	•	• •	•	•
· ·	•	•	• •	• •	•	•	• •		•	•	• •	•	•	•	• •	•	•	•	•	••••	•	•	· ·	•	• •	•	• •	• •	•	•	••••	•	•	• •	•	•
	•	•		•	•	•	••••	•	•	•	• •	•	•	•	• •	•	•	•	•	• •	•	•	· ·	•		•	• •		•	•	••••	•	•	• •	•	•
• •		•	• •	- • •		•	• •	•	•		• •			•	• •			•		• •		•	• •			•				•	• •	•	•	0 0		
• •	•	•	• •	• •	•	•	• •	•	•	•	••••	•	•	•	• •	•	•	•	•	• •	•	•	· ·	•	· ·	•	• •	· ·	•	•	••••	•	•	• •	•	•
· ·	•	•	• •	• •	•	•	• •	•	•	•	• •	•	•	•	• •	•	•	•	•	• •	•	•	· ·	•	· ·	•	• •	• •	•	•	· ·	•	•	• •	•	•

Complexity Theory
Categorize problems based on how EASY/HARD
they are to solve
P = Problems that can be solved in polynomial time.
NP = Problems that can be verified in polynomial time.
NP-Hard Z Q is NP-Hand if every problem in NP reduces to Q.

· · · · · · · · · · · · · ·		
		NPA a serie
	ST =DIST SHOULTEST-PATH	SAT
MAX BIP	MATCH	
	$\mathcal{F}_{LOW} = \mathcal{F}_{M} = \mathcal{F}_{M$	
	· · · · · · · · · · · · · · · · · · ·	
· · · · · · · · · · · · ·		· · · · · · · · · · · · · · · · · · ·
		· · · · · · · · · · · · · · · · · · ·
· · · · · · · · · · · · · ·		· · · · · · · · · · · · · · · · · · ·

MST EDITEDIST SHOWTEST-PATTH FACTORING SAT
MAX BIP. MINL SUBSET SUM MAX FLOW IND SET
Theorem. Every problem QENP reduces to SAT!
If any efficiently-verifiable problem is hand to solve, then SAT is hand to solve.

$\sum_{n=1}^{\infty} \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \sum_{i=1}^{\infty} \sum_{i$	Given a does there	boolean formu e exist a	ila \$, satisfying assignment?
· · · · · · · ·	· · · · · · · · · · · · ·	$\exists \hat{a} \in \{0, 1\}$	$\hat{\nabla} = s + t + \frac{1}{2} +$
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	\times_1 \vee_1 \times_2 \vee_2 \times_3	$(X_{2}, X_{3}) = (X_{2}, X_{3}) + (X_{3}, X_{3}) + (X_{$	$(1 \times 1) \times 1 \times$
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	· ·
. 	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	
 	· · · · · · · · · · ·	· · · · · · · · · · · ·	. .
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .
. 	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .

SAT CONF	stures P	$\frac{vs}{vs}$. .	· · · · · · · · · · ·
Theorem	T = f + f + f + f + f + f + f + f + f + f	re is a polyn AT, then AT, a then	p = NP	Qgovittur
		SAT		

SAT captures Prs. NP	· · · · · ·
Theorem If there is a polynomial - time algori for SAT, then P=NP. any NP-Hand problem	· · · · · · · · · · · · · · · · · · ·
NP P $SATP = NP$	

Theorem If there is a polynomial - time algorithm for SAT, then P=NP. Proof Suppose A is a poly-time algo that solves SAT * Consider any problem QENP. * We show that QEP => NPEP => P=NP. . . //.

Theorem If there is a polynomial - time algorithm for SAT, then P=NP. Proof Suppose A is a poly-time algo that solves SAT * Consider any problem QENP. * SAT NP-Hard => 3 poly-time reduction R from Q to SAT · · · · · MP· · ·

Theorem If there is a polynomial - time algorithm for SAT, then P=NP. Proof Suppose A is a poly-time algo that solves SAT. * Consider any problem QENP. * SAT NP-Hard \Longrightarrow 3 poly-time reduction R from a contra SATT Algo Q. On input x to Q, SAT . Run Roon X Run A on R(x) Return A(R(x))

Theorem If there is a polynomial - time algorithm for SAT, then P=NP. Proof Suppose A is a poly-time algo that solves SAT. * Consider any problem QENP. * SAT NP-Hard \Longrightarrow 3 poly-time reduction R from Q to SAT Algo Q. On input x to Q, Run Ron X Run A on R(x) Return A(R(x)) Claim. AlgoQ solves Q in poly-time! > QEP.

Most Computer Science	tists Agree
. .	$P = \frac{1}{2} + $
By previous theorem	$\left(\left(ceutvapositive\right)\right)$
$\left \begin{array}{cccccccccccccccccccccccccccccccccccc$	Then SAI does <u>Not</u> have a polynomial - time algorithm,
$\begin{array}{c} \begin{array}{c} \begin{array}{c} \begin{array}{c} \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} \end{array} \\ \end{array} $	Then SAT does Not have a polynomial - time algorithm,

Today Ler	verage SAT to show	other problems must be HARD
To show	problem () is	JP-HARD, show
· · · · · · · · · · ·	SAT reduces fo	Q in poly-time
· · · · · · · · · · · ·	$\sum_{i=1}^{n} A_{i} \sum_{i=1}^{n} A_{i} \sum_{i=1}^{n$	$\leq \stackrel{\circ}{\underset{\rightarrow}{\rightarrow}} p \qquad \qquad$
· · · · · · · · · · · ·	. . <td> </td>	
· · · · · · · · · · ·		. .
· · · · · · · · · · · ·	· ·	
· · · · · · · · · · · ·
· · · · · · · · · · · ·

Today Leverage SAT to show other problems un	st be HARD
	· · · · · · · · · · ·
To show problem Q is NP-HARD, show	
SAT reduces to Q in poly-time	· · · · · · · · · · ·
5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5 +	· · · · · · · · · ·
· · · · · · · · · · · · · · · · · · ·	· · · · · · · · · · ·
Conclude. No polynomial-time algorithm for Q (unless	P = NP
 	· · · · · · · · · · ·
	· · · · · · · · · ·
· · · · · · · · · · · · · · · · · · ·	

Announcements
* HWS ongoing
\times HWG
- Solutions Wednesday 26 March
× Prelim 2. 27 March, 7:30 - 9p
La Location: Same as Prelim 1.
L' Coverage: Hworzh HWG
- Max Flow / Min Cut
- NP- Completeners
Le Practice Exam veleased Wednesday.
· · · · · · · · · · · · · · · · · · ·

$\sum_{n=1}^{\infty} \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \sum_{i=1}^{\infty} \sum_{i$	Given a does there	boolean formu e exist a	ila \$, satisfying assignment?
· · · · · · · ·	· · · · · · · · · · · · ·	$\exists \hat{a} \in \{0, 1\}$	$\hat{\nabla} = s + t + \frac{1}{2} +$
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	\times_1 \vee_1 \times_2 \vee_2 \times_3	$(X_{2}, X_{3}) = (X_{2}, X_{3}) + (X_{3}, X_{3}) + (X_{$	$(1 \times 1) \times 1 \times$
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	· ·
. 	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	
 	· · · · · · · · · · ·	· · · · · · · · · · · ·	. .
· · · · · · · ·	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .
. 	· · · · · · · · · · · ·	· · · · · · · · · · · · ·	. .

SAT. Given a boolean formula \$,
does there exist a satisfying assignment.
$\exists \vec{a} \in \{0,1\}^{n} \text{s.t.} \varphi(\vec{a}) = 1?$
$ \cdot \cdot$
$\Phi_{1} = \left($
$\cdots \cdots $
$\frac{k-CNT}{t}$
Conjunction (AND) of m clauses
Disjunction (OR) of k literals
one of n variables or negation
· · · · · · · · · · · · · · · · · · ·

<u>3SAT</u>	Given a	3-CNF	formula \$	b does 7 à
· · · · · · · · ·		 . f		ς . t . $\phi(\vec{a}) = 1$
· · · · · · · · ·				
· · · · · · · · ·			· · · · · · · · · · · ·	
· · · · · · · · ·	Every	c (au se	$has \leq 3$	liferals
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				
· · · · · · · · ·				

<u>3SAT</u>	Given a 3-	CNF formula ¢	does 7 à
· · · · · · · · ·		· · · · · · · · · · · · · · · · · · ·	$s.t. \phi(\vec{a}) = 1$
· · · · · · · · ·		· · · · · · · · · · · · · · · · ·	· · · · · · · · · · · · · · · · · · ·
· · · · · · · · ·	Every clo	$mse has \leq 3$	liferals
· · · · · · · · · ·		· · · · · · · · · · · · · · · · · · ·	
Theorew	y (Cook-Levin	$ \left(\begin{array}{cccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 &$	NP-Hand
· · · · · · · · · ·	<pre> · · · · · · · · · · · · · · · · ·</pre>	· · · · · · · · · · · · · · · ·	· · · · · · · · · · · · · · · · · · ·
· · · · · · · · ·	
. .			
. .			

		$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	satisficble	3CNF formula
· · · · · · ·	Represent	YES/NO	Decision 7 Subsets	of valid inputs
· · · · · ·		set of	YES inste	mces
· · · · · ·	· ·	· · · · · · · · · ·	· · · · · · · · · · · ·	· ·
· · · · · ·	. .	· · · · · · · · ·	· · · · · · · · · · ·	. .
 	. .	· · · · · · · · ·	· · · · · · · · · · ·	. .
· · · · · ·		· · · · · · · · · ·	· · · · · · · · · · · ·	

3SAT = 2		satisficble	3CNF formula
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	esent YES/NO	Decision P Subsets	of valid inputs
· · · · · · · · · · · · · · · · · · ·	the set of	TES inste	$M \xrightarrow{c} \underbrace{c} \underbrace{s} \underbrace{s} \underbrace{s} \underbrace{s} \underbrace{s} \underbrace{s} \underbrace{s} s$
Réduction	maps YES No	instances	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Reduction	YES NO	instances	No. instances

Maximum Independent Set $G_{iven} = G_{iven} + G_{iven}$ SEV is an independent set if for all $\mathsf{veS} \mid \mathsf{veS} \mid \Rightarrow \mathsf{veS} \models \mathsf{E}$ INDSET = $\{G, k\}$: G contains on independent set $\{G, k\}$: of condinality $\geq k$

Maximum Independent Se	• • • • • • • • • • • • • • • • • • •
Criven a graph G=($(\mathcal{V}, \mathcal{E})$
$5 \leq \sqrt{10}$ is $0 \leq 10$	independent set if for all
$\mathcal{W} \in \mathcal{S} $	\mathbb{S}
$INDSET = \left\{ \left\langle G_{1}, k \right\rangle \right\}$	G contains on independent set Z of condinality > k
Fact. INDSET E NP.	
. .	. .

Maximum Independent Set Criven a graph G = (V, E), SEV is an independent set if for all $\mathsf{ueS} | \mathsf{veS} \Rightarrow (\mathsf{u},\mathsf{v}) \notin \mathsf{E}$ ≠SENP ⇒ poly-time Verifier Verifier Gik, S Chechs 151 Z k Huires chech (u,v) & E. INDSET = $\{G, k\}$: G contains on independent set $\{G, k\}$: of condinality $\geq k$ Fact. INDSET E NP.

Maximum Independent Set
Criven a graph $G = (V, E)$,
SEV is an independent set if for all
$u \in S v \in S \Longrightarrow (u, v) \notin E$
INDSET = $\{G, k\}$: G contains on independent set $\{G, k\}$: of cardinality $\geq k$
Theorem. 3SAT <pre>p INDSET.</pre>
Covollary. INDSET is NP-HARD.

Reduction from 3SAT to INDSET.	· · · ·
Design polynomial-time algorithm: * Given \$\$	
\times Networks (9, \sim)	
Satisfiable (Satisfiable) of candinality	
· · · · · · · · · · · · · · · · · · ·	· · · · ·
· · · · · · · · · · · · · · · · · · ·	
· · · · · · · · · · · · · · · · · · ·	
· · · · · · · · · · · · · · · · · · ·	
	· · · · ·

Reduction from 3SAT to	INDSET
Design polynomial-time al	zor i Hun :
* Returns (G, K)	· ·
	<pre></pre>
ϕ Satisfia	ble (=> G has IS of candinality >k
$\phi \in 3SA^{-1}$	$r \iff \langle G, h \rangle \in I \lor D S \in T$
· · · · · · · · · · · · · · · · · · ·	. .
· · · · · · · · · · · · · · · · · · ·	
· · · · · · · · · · · · · · · · · · ·	

Reduction from 3SAT to INDSET.
Design polynomial - time algorithm:
\times \times Given ϕ
* Returns (G, K)
s.t.
P Satistiable <-> of candinality
\cdots
$\phi \in 3SAT \iff \langle G, h \rangle \in INDSET$
· · · · · · · · · · · · · · · · · · ·
Want produce a graph where
large IS "selects" assignment that satisfies all clauses.

Variable Assignment Gadgets				
For each $i=1$ - n	$X_i = \alpha_i \in Z$			
	assighed a	Single .	valne.	
$(\chi_{1} = 0)$	· · · · · · · · · · ·	$\left(\begin{array}{c} \chi \\ \chi \\ \chi \\ \chi \\ \chi \\ \eta - \eta \end{array} \right) = \left(\begin{array}{c} \chi \\ \chi \\ \eta \end{array} \right)$	(1) (2)	
	· · · · · · · · · · · · · · · · · · ·			
$\left(\begin{array}{c} x_{1} \\ x_{2} \end{array} \right) = \left(\begin{array}{c} x_{1} \\ x_{2} \end{array} \right) = \left(\begin{array}{c} x_{2} \\ x_{3} \end{array} \right) = \left(\begin{array}{c} x_{2} \\ x_{3} \end{array} \right) = \left(\begin{array}{c} x_{1} \\ x_{2} \end{array} \right) = \left(\begin{array}{c} x_{1} \\ x_{3} \end{array} \right) = \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) = \left(\begin{array}{c} x_{1} \end{array} \right) = \left(\begin{array}{c} x_{1} \end{array} \right) = \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) = \left(\begin{array}{c} x_$	· · · · · · · · · · ·	$(X_{1})_{n-1} = 1$	$\left(\begin{array}{c} x \\ x $	
· · · · · · · · · · · · · · · · · · ·		· · · · · · · · · ·		
			· · · · · · · · · · ·	
			· · · · · · · · · · · ·	
	· · · · · · · · · · ·		· · · · · · · · · · ·	
	· · · · · · · · · · ·		· · · · · · · · · · ·	

Variable Assighment Gade	yets	· · · · · · · · ·	· · · · · · · · · · ·
For each i=1	$X_2 = Q_2 \in {\mathbb{Z}}$ assigned a	Single	
$(\chi_{1}=0)$. .	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	$ \begin{array}{cccccccccccccccccccccccccccccccccccc$
$ \left \begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$		$\begin{array}{cccccccccccccccccccccccccccccccccccc$
$ + \frac{1}{2} = 1 \sqrt{N} $			
~ Create two vertices	$X_{10} = (X_{1} = 0)$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	· · · · · · · · · · · · · · · · · · ·
$-Add (V_{20}, V_{21}) \in E$			
· · · · · · · · · · · · · · · · · · ·		· · · · · · · · · ·	

Clause Satisfaction Gadgets	· · · · · · · · · · ·
For each j=1,, ~ some literal in G eve	aluates to 1
	unsatisfiable)
$C_{j} = \left(\begin{array}{c} l_{j_{1}} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ $	
\cdots	
· · · · · · · · · · · · · · · · · · ·	
(1) (1) (1) (1) (1) (1) (1) (1) (1) (1)	

Clause Satisfaction Gadgets
For each j=1,, m some literal in Cj evaluates to 1
(else, unsatisfiable)
$C_{j} = \left(\begin{array}{c} l_{j} \\ l_{j^{2}} \end{array} \right)$
$\left(\begin{array}{c} \begin{array}{c} \begin{array}{c} \\ \\ \\ \end{array}\right)^{\prime} \\ \end{array}\right)^{\prime} \\ \end{array}$
$\forall j = 1 - m$ -Create 3 vertices $l_{j_1}, l_{j_2}, l_{j_3} \in V$
-Add edges between each (lj, lj2), (lj2, lj3), (lj3, lji) EE
· · · · · · · · · · · · · · · · · · ·

Clause / Assignment Consistency $(X_1 = 0)$ $(X_2 = 0)$ $(X_q = 0)$ $\left(\begin{array}{c} X_{n-1} \\ X_{n-1} \end{array} \right)$ $\left(\begin{array}{c} X_{n} \\ X_{n-1} \end{array} \right)$ $(X_1 = 1) \qquad (X_2 = 1)$ $\begin{pmatrix} X_{n-1} \\ A_{n-1} \end{pmatrix} \begin{pmatrix} X_{n-1} \\ X_{n-1} \end{pmatrix} \begin{pmatrix} X_{n-1} \\ X_{n-1}$ $\left(X_{q}=1\right)$ Qmi (\mathcal{Q}_{1}) Q 21 l12) (Q13) J23 lm2 Lm3

Clause Assignment Consistency $(X_1 = 0)$ $(X_2 = 0)$ $(X_2 = 0)$ $(X_q = 0)$ $X = \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \end{array} \right) \left(\begin{array}{c} x_{1} \\ x_{1} \end{array} \right) \left($ $\begin{pmatrix} X_{n-1} \\ n-1 \end{pmatrix} = \begin{pmatrix} X_{n-1} \\ n-1 \end{pmatrix}$ $\begin{array}{c} \mathbf{x} \\ \mathbf{$ $l_{ij} = X_i + | \cdot | \cdot |$ Q_{m1} $\phi = (x_1, \vee \neg X_2 \vee X_q) \wedge (\neg X_q \vee X_{n-1} \vee \neg X_n) \wedge \neg -$ If literal lj= = Xi, add edge (lj+, Xio) EE If liferal ljt = TXi, add edge (ljt, Xi)) EE

Clause Assignment Consistency $\begin{pmatrix} X_{n-1} \\ X_{n-1} \end{pmatrix} = \begin{pmatrix} X_{n-1} \\ X_{n-1} \end{pmatrix} \begin{pmatrix} X_{n-1} \\ X_{n-1$ $(X_1 = 0)$ $(X_2 = 0)$ $X_q = 0$ $Q_{22} = X_{m-1}$ $\begin{array}{cccc} x & x & y \\ x & z & z \\ z &$ $l_{ij} = \chi_{ij}$ $l_{23} = 7 \times n$ For each clause Cj, for each literal literal literal literal If literal $2j_{+} = X_{i}$, add edge $(2j_{+}, X_{i0}) \in E$ If literal $2j_{+} = \forall X_{i}$, add edge $(2j_{+}, X_{i1}) \in E$

Clause / Assignment Consistency $(X_1 = O)$ $\left(X_{n}=0\right)$ $(X_2 = O)$ $X_q = 0$ $(X_n = 1)$ $(X_2 = I)$ $(X_1 = 1)$ $(X_q = 1)$ $\left(X = 1 \right)$ lm2 LIS J_22 Jis L12 (Lm3) Claim. O is satisfiable () G has IS of an condinality Z N+M

ϕ satisfiable \Longrightarrow IS of size $n + m$.
* Consider a satisfying assignment à EZO, 13h.
X For each $i = 1 \infty$, add X_{ib} to S for $a_i = b$.
* For each j=1, add literal ljt to S
for some 2jt = 1 under a
$\frac{C(nim)}{S} = n + m$
Claim 1. S is an independent set.
<pre>- · · · · · · · · · · · · · · · · · · ·</pre>
· · · · · · · · · · · · · · · · · · ·

\$ satisfiable => IS of size n+m.
* Consider a satisfying assignment à EZO, 13h.
\times For each $i = 1 \infty$, add x_{ib} to S for $a_i = b$.
* For each j=1, add literal ljt to S for some ljt =1 under a
$\frac{C(\alpha)m}{\partial} = n + m$
Claim 1. S is an independent set.
* only select 1 vertex per variable gadget. 1 vertex per clause gadget.
* $(l_{jE}, x_{ib}) \in E$ only if setting $x_i = b$ causes $l_{jE} = 0$ $\Rightarrow (l_{jE}, x_{ib}) \notin E$.

ϕ ϕ ψ ϕ ψ ϕ ϕ	$tistiable \Longrightarrow Every IS has cardinality < n+M$
* Every	IS has < n vertices from variable gadgets & < m vertices from clause gadgets
* * Every	assignment $\overline{a} \in \overline{\{0\}}, \overline{\{1\}}^n$ results in some clause C_j with every literal $Q_{j_1} = Q_{j_2} = Q_{j_3} = 0$
. .	· ·
	. .
· · · · · · · · · · ·	. .

ϕ unsatisfiable \Longrightarrow Every IS has cardinality $< n+M$
* Every IS has < n vertices from variable gadgets & < m vertices from clause gadgets
* Every assignment $\overline{a} \in \overline{\{0\}}, \overline{\beta}^n$ results in some clause C_j^n with every literal $2j_1 = 2j_2 = 2j_3 = 0$.
* Consider the set S_{a} of n vertices associated w/ $\bar{x} = \bar{a}$. L> For each vertex l_{j_1} , l_{j_2} , l_{j_3} , there is some neighboring vertex $x_{ib} \in S_{a}$
Thus, IS cannot have I vertex per variable and I rentex per clarse.

	. .	NP SAT INDSET		
INDSET INDSET	is NP-H	$c_{\rm even} d$	IND SET	ENP-Complete
· ·	· · · · · · · · · · · · · · ·	· ·	· · · · · · · · · · ·	· ·