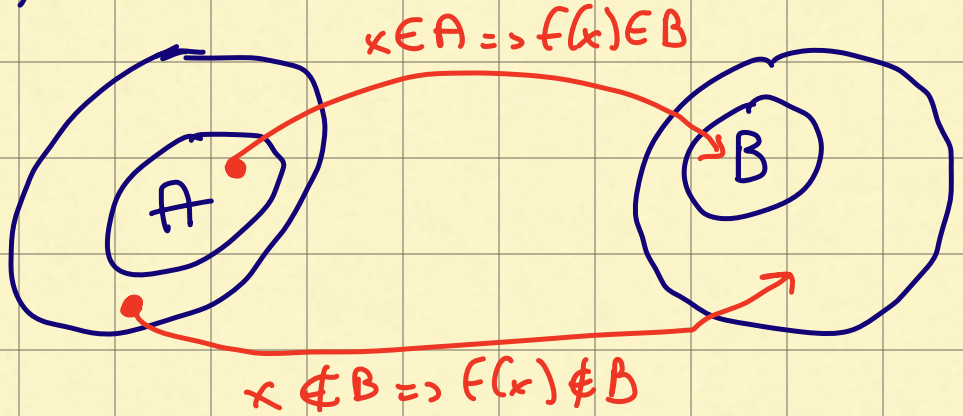
Lecture 22 : Hw5 is due

Sunday @ 11 am

Polytime Mapping reductions

$A \leq_P B$ if there exists a function f computable in polytime such that $x \in A \iff f(x) \in B$



Def: A is NP hard if $\forall L \in NP$

$L \leq_P A$

Def: L is NP complete iff:

(1) L is NP-hard
(2) L is in NP

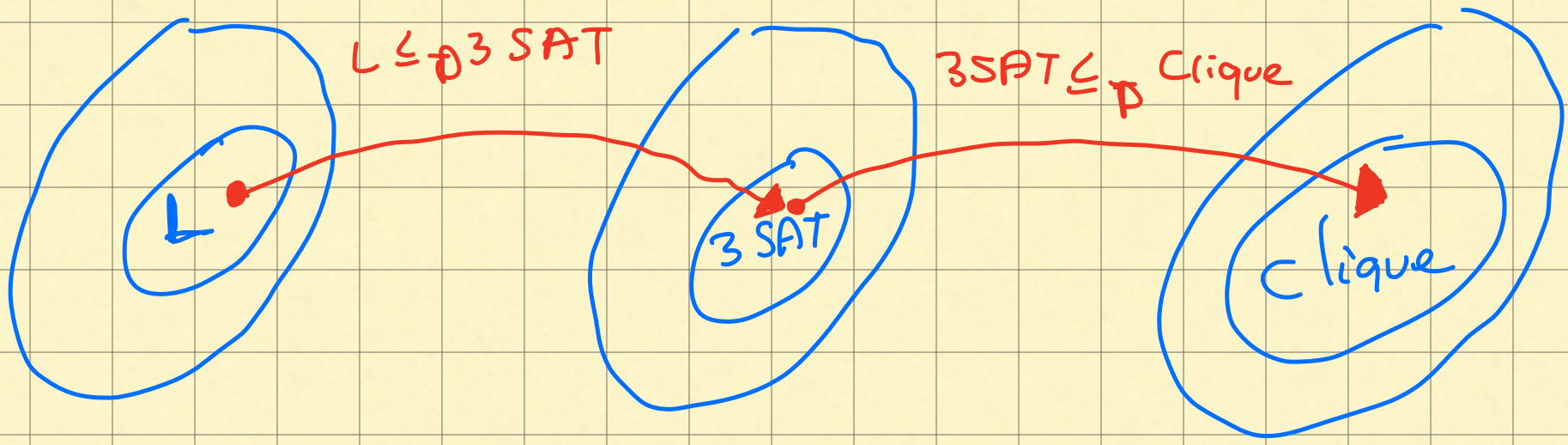
Thm: 3-SAT is NP hard [Cook, Levin]

To prove a language A is NP-hard:

pick NP hard B and show $B \leq_P A$

That's what we did with CLIQUE last time!

$L \in NP$



Ie: for all $L \in NP$, $L \leq_p$ Clique !

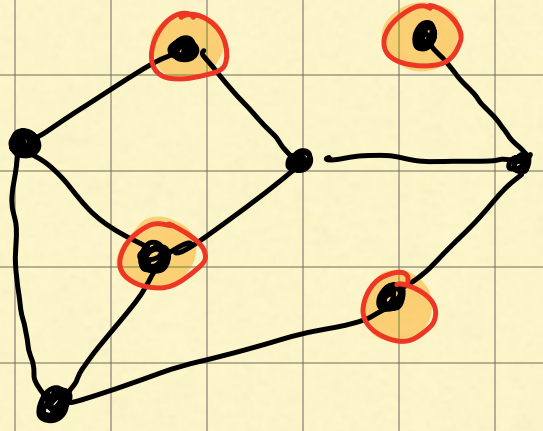
If we could solve Clique in poly time, we have all $L \in NP$ can be solved in poly time !!

(2) Independent Set

$TS = \{ \langle G, k \rangle \mid G$ is an undirected graph with an independent set of size $k \}$

An independent set is k vertices $v_1, \cdots, v_k$ that do not have edges between them

Ex:



An independent set of size 4

Goal: show I.S is NP hard.

(1) I.S $\in$ NP

Proof is similar to k. Clique

NTM: On input $\langle G, k \rangle$

- Nondeterministically pick a subset S of k vertices
- For each pair $(u,v) \in S$:
  If (u,v) is an edge: reject
- Accept

pass iff S is an independent set

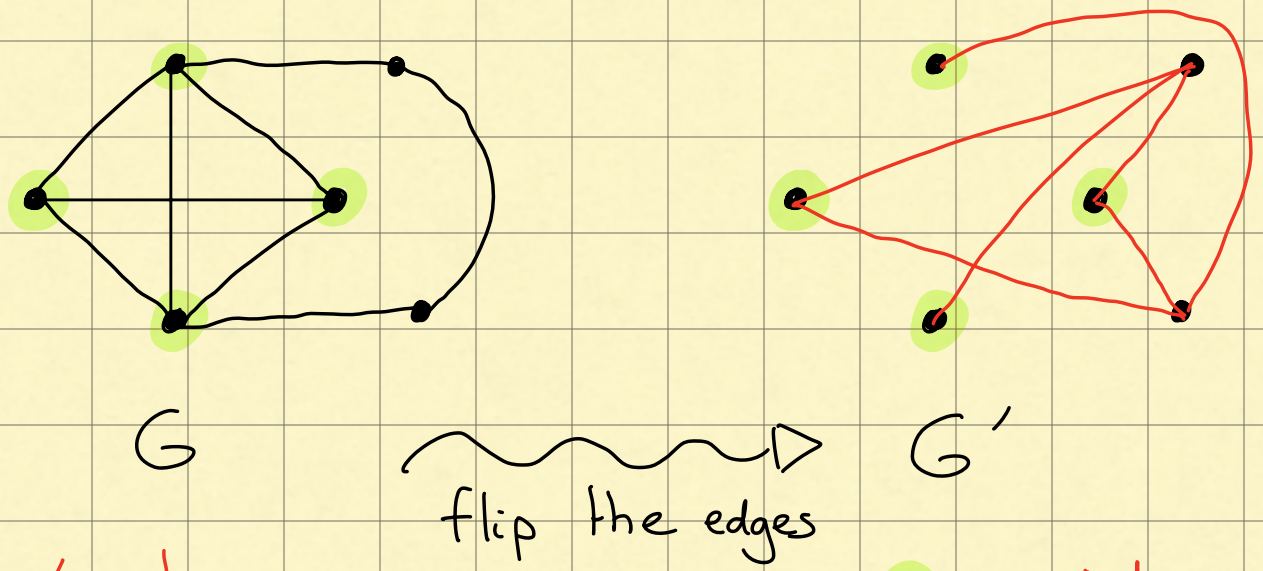
NP hard: we show Clique $\leq_p$ I.S.

Since Clique is NP-hard, IS is NP hard!

The mapping $f(\langle G, k \rangle) = \langle G', k \rangle$ where G' is a graph with the same vertices as G

but : (u,v) is an edge in G' iff

(u,v) is <u>not</u> an edge in G



a 4-clique

an independent set of size 4

In general if S is a clique of size k in G, then S is an indendent set of size k in G'.

(1) can compute f in polytime

(2) Since S is a k-clique in G iff S is an independent set of size k in G'. So $\langle G, k \rangle \in$ Clique iff $f(\langle G, k \rangle) = \langle G', k \rangle \in$ I.S.

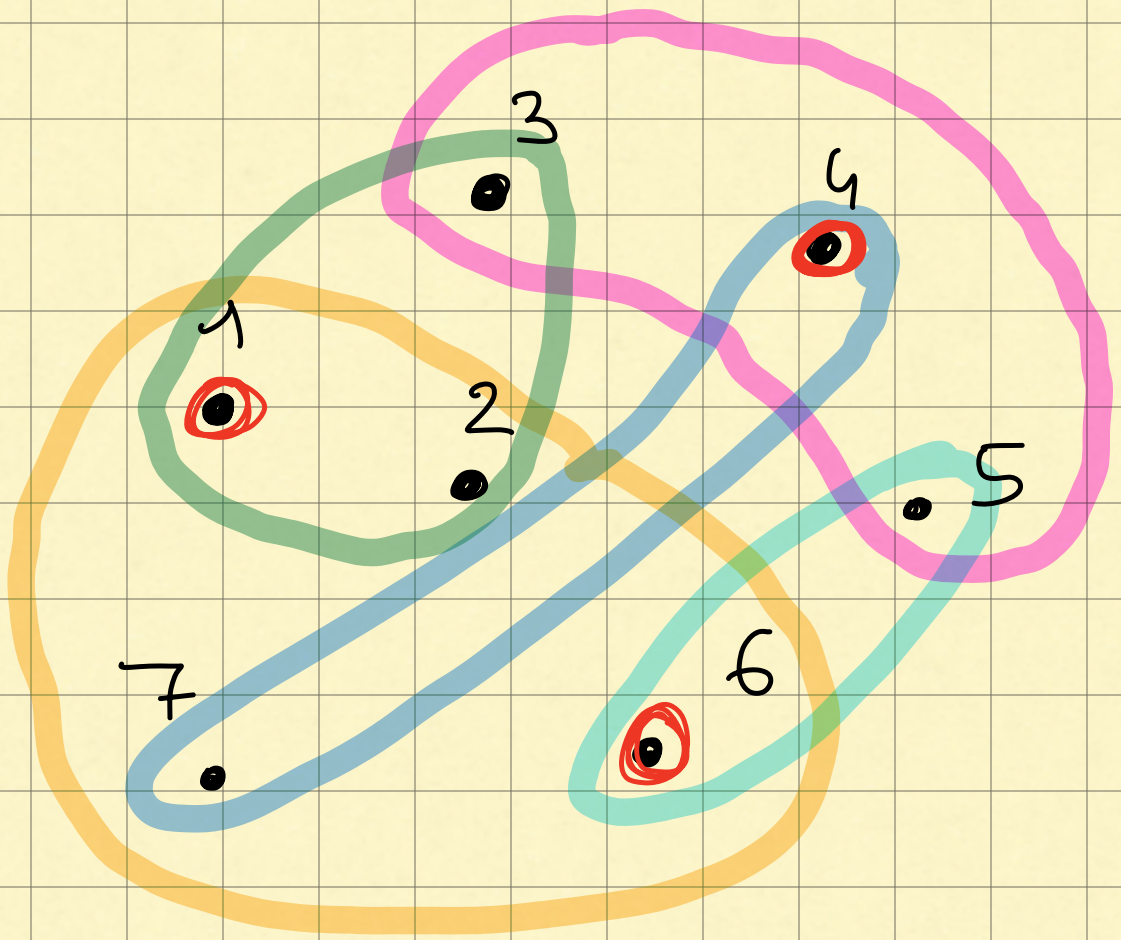
③ Hitting Set:

Input : a set system S over some universe U and K

$S = \{ S_1, \dots, S_m \}$ where $S_i \subseteq U$

A hitting set of size k is a subset $H \subseteq U$, $|H| = k$ and $\forall i$:

$$S_i \cap H \neq \emptyset$$



$S_1 = \{3, 4, 5\}$

$S_2 = \{1, 2, 3\}$

$S_3 = \{1, 2, 6, 7\}$

$S_4 = \{5, 6\}$

$S_5 = \{4, 7\}$

$U = \{1, 2, \dots, 7\}$

$H = \{1, 4, 6\}$: Foreach S_i

$S_i \cap H \neq \emptyset$

What's the smallest hitting set?

H.S. is NP-complete

(1) H.S. $\in$ NP

NTM: On input (S, k)

- Nondeterministically pick $H \subseteq U$
  $|H| = k$
- For each $S_i \in S$:
  If $S_i \cap H = \emptyset$ => reject
- Accept

check H intersects each S_i

Verifier

(C is a set of k elements, check $C \cap S_i \neq \emptyset$ for every S_i

Hitting set is NP-hard:

We show $3SAT \leq_p HS$.

Given $\langle \phi \rangle$ a CNF $C_1 \wedge \dots \wedge C_m$ over n variables, $f(\langle \phi \rangle) = \langle S_\phi, n \rangle$

where S_ϕ is a set system

$\langle \phi \rangle \in 3\text{-SAT}$ iff S_ϕ has a hitting set of size n.

An example:

$\phi = (x_1 \vee x_2 \vee x_3) \wedge$

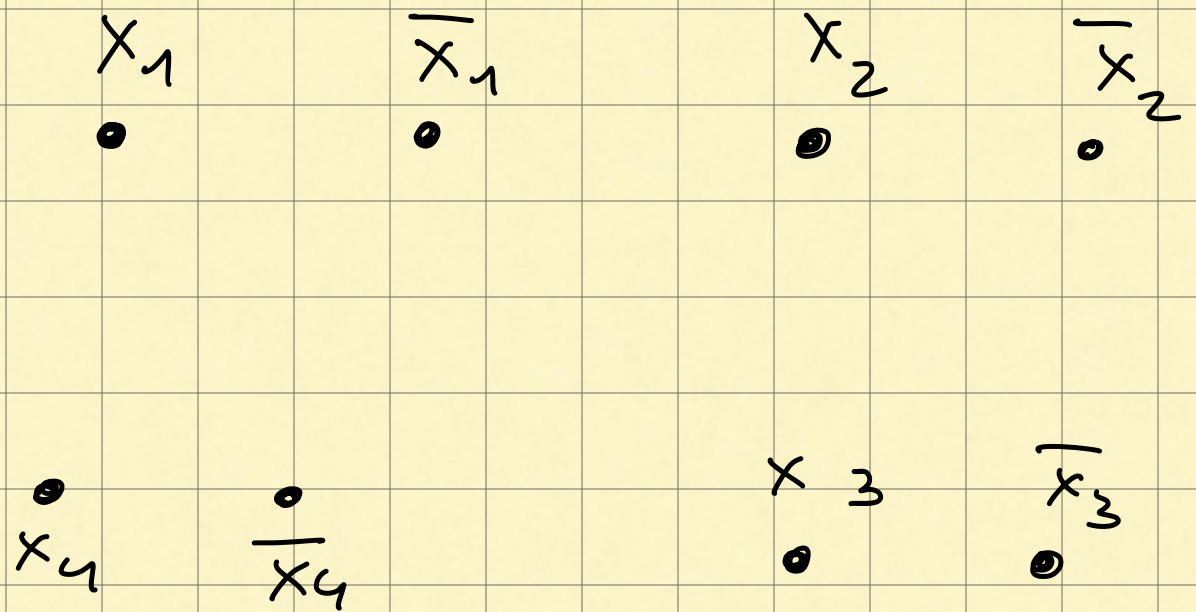
$(x_1 \vee x_2) \wedge$

$(\bar{x}_1 \vee x_4) \wedge$

$(\bar{x}_2 \vee x_3)$

We let

$U = \{x_1, \bar{x}_1, x_2, \bar{x}_2, x_3, \bar{x}_3, x_4, \bar{x}_4\}$



An example:

$\phi = (x_1 \vee x_2 \vee x_3) \wedge$

We let

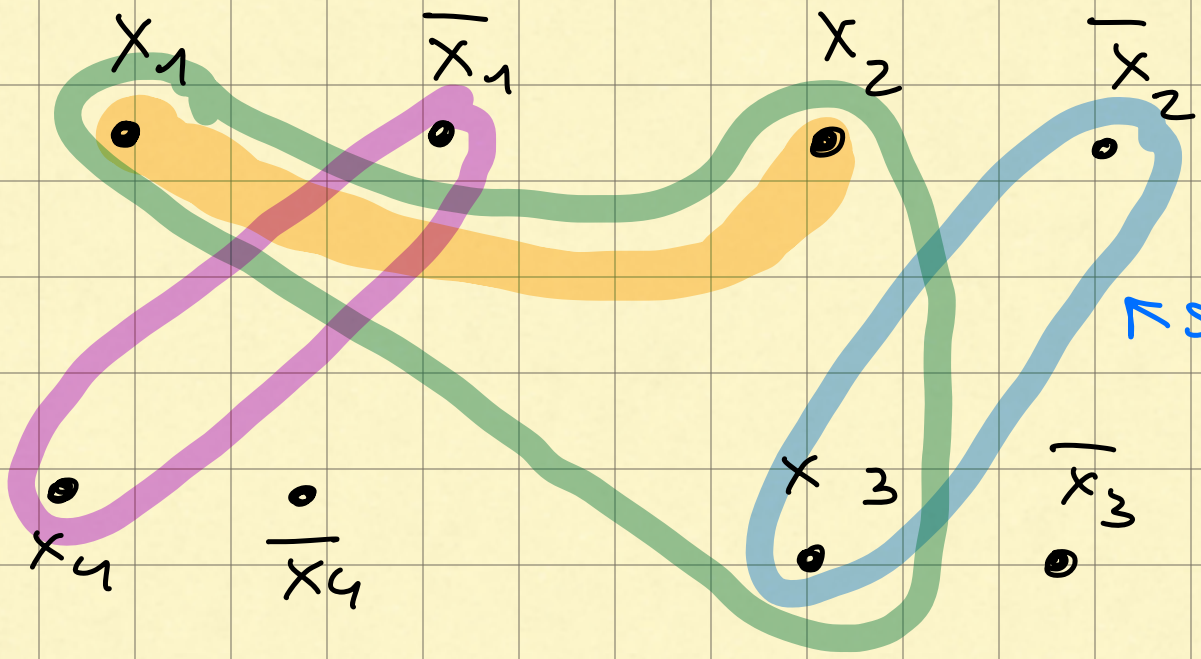
$(x_1 \vee x_2) \wedge$

$U = \{x_1, \bar{x}_1, x_2, \bar{x}_2, x_3, \bar{x}_3, x_4, \bar{x}_4\}$

$(\bar{x}_1 \vee x_4) \wedge$

$(\bar{x}_2 \vee x_3)$

Step 1: Clauses as sets



$\text{set}(C_4) = \{\bar{x}_2, x_3\}$

An example:

$\phi = (x_1 \vee x_2 \vee x_3) \wedge$

We let

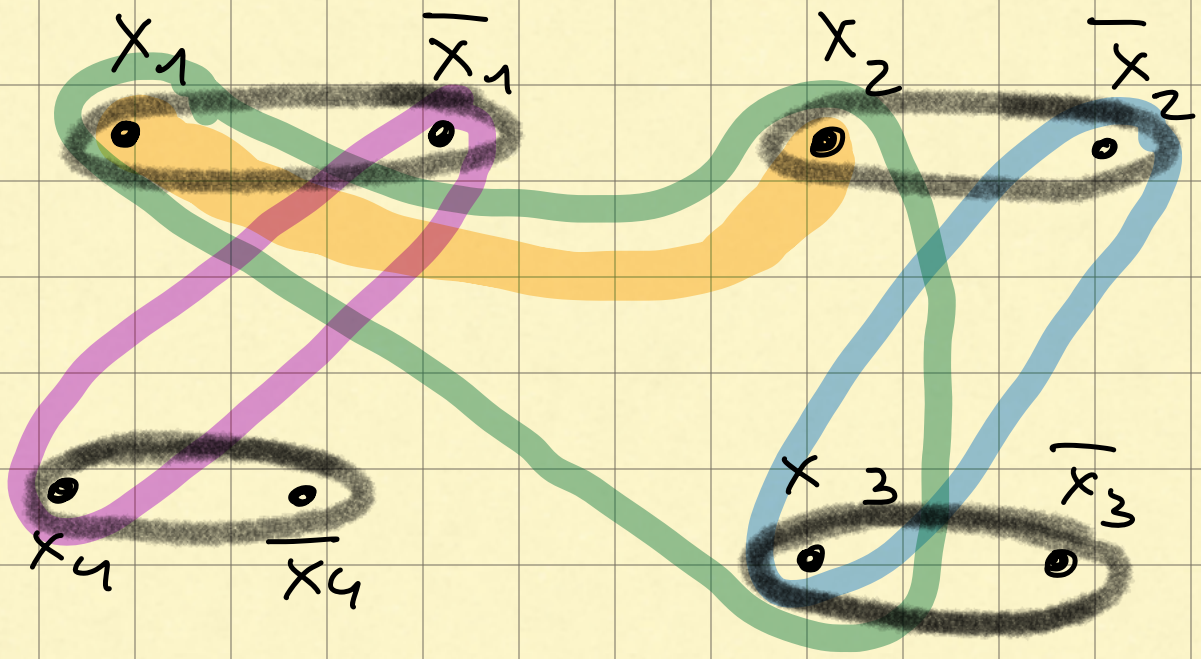
$(x_1 \vee x_2) \wedge$

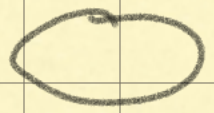
$U = \{x_1, \bar{x}_1, x_2, \bar{x}_2, x_3, \bar{x}_3, x_4, \bar{x}_4\}$

$(\bar{x}_1 \vee x_4) \wedge$

$(\bar{x}_2 \vee x_3)$

Step 2: sets to force variable choice



All  are disjoint. So to build a hitting set of size n => need to pick one element in each  (exactly one)

↳ this corresponds to picking x_i or $\overline{x_i}$ as being true for each variable

An example:

$\varphi = (x_1 \vee x_2 \vee x_3) \wedge$

$(x_1 \vee x_2) \wedge$

$(\bar{x}_1 \vee x_4) \wedge$

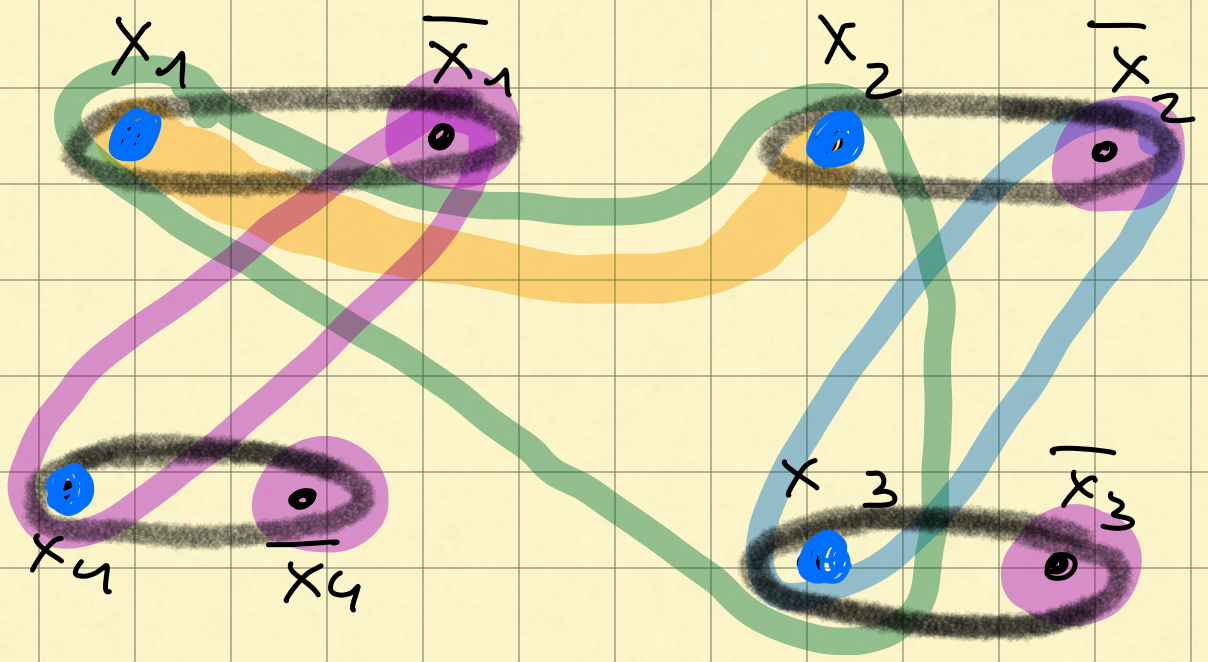
$(\bar{x}_2 \vee x_3)$

What's a satisfying assignment to φ?

$x_1 = 1$ or 0 $\quad x_2 = 1$ or 0 $\quad x_3 = 1$ or 0

$x_4 = 1$ or 0

How to build a H of size n?



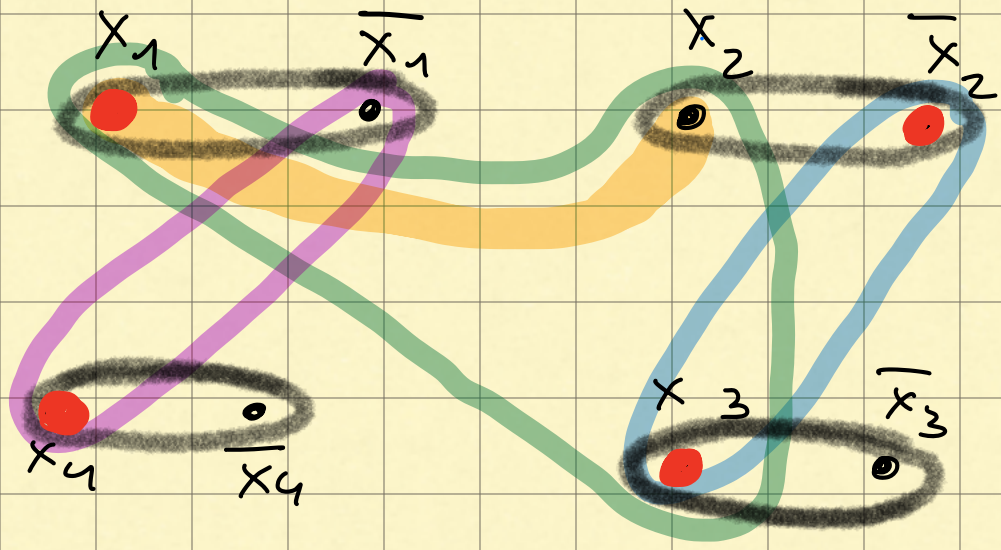
Hitting set of size 4

$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge$

$(\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$

Say H is an hitting set of size n

H contains exactly one of $x_i, \bar{x}_i$



Each clause has some litteral in H

$\alpha = (1, 0, 1, 1)$ 1 if $x_i \in H$, 0 if $\bar{x}_i \in H$

In general: given $\langle \phi \rangle \Rightarrow f(\langle \phi \rangle) = \langle S_\phi, n \rangle$

↳ n variables, m clauses

$U = \{x_1, \bar{x}_1, \cdots, x_n, \bar{x}_n\}$

$S_\phi = \{ \{x_1, \bar{x}_1\}, \cdots, \{x_n, \bar{x}_n\} \}$ ⇐ our

$\cup \{ \text{set}(C_1), \cdots, \text{set}(C_m) \}$ ⇐ encode clause

↳ {set of litterals in C_1}

(1) we can compute f in poly time

(2) We can show ϕ has a satisfying

assignment iff S_ϕ has a hitting set of size n.

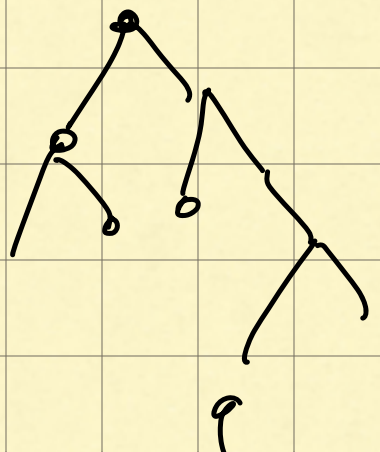
Subset-Sum: $\{\langle S, t\rangle \mid t$ is a positive integer,

S is a set of positive integers

$\exists T \subseteq S$ such that $\sum_{x \in T} x = t\}$

In NP: Sketch:

- On input $\langle S, t\rangle$



- Nondeterministically pick $T \subseteq S$
- Check $\sum_{x \in T} x = t$

each branch is TES

- If yes : accept

  Else reject

Cook Levin Theorem: SAT is NP complete

SAT = $\{\langle \phi \rangle \mid \phi$ is a satisfiable 3 CNF formula$\}$

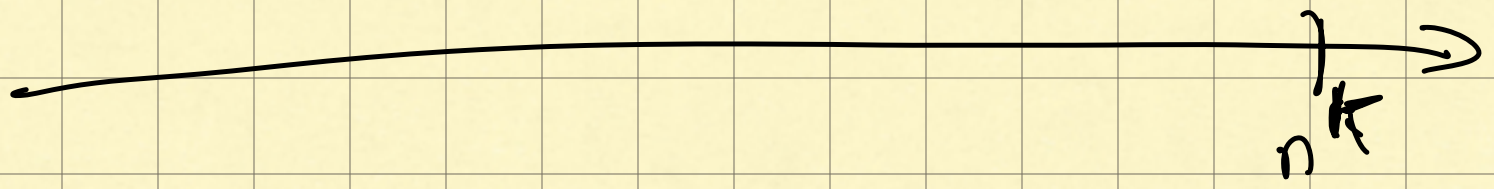
(1) SAT is NP: same as 3-SAT we did it!

(2) SAT is NP-hard

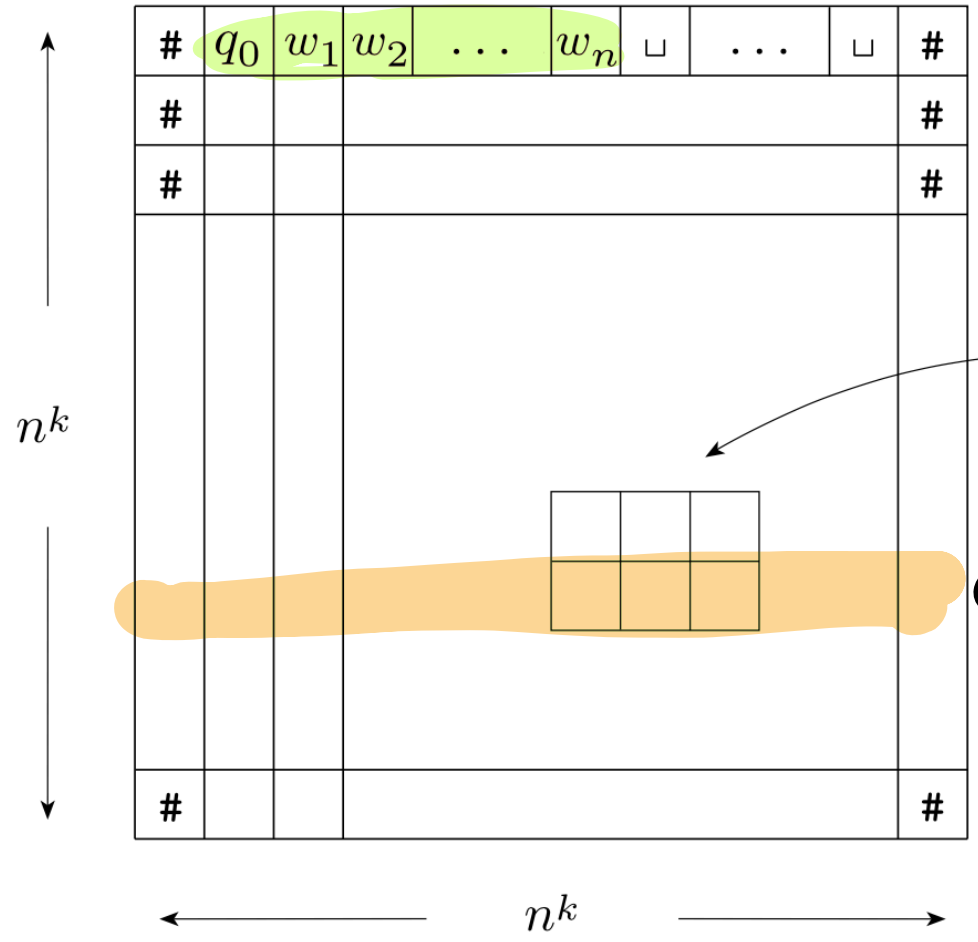
Proof: Let A be in NP, we show $A \leq_p SAT$

Since A is in NP, $\exists$ a NTM N which decides A in time n^k.

↳ so A uses n^k time, but also at most n^k cells in the tape on an input w, $|w|=n$



A tableau of N on w ← w is an input, length $|w|=n$



← N is an accept configuration : N is in state q_{accept}

The rows represent the configurations of N (on some branch) on input w

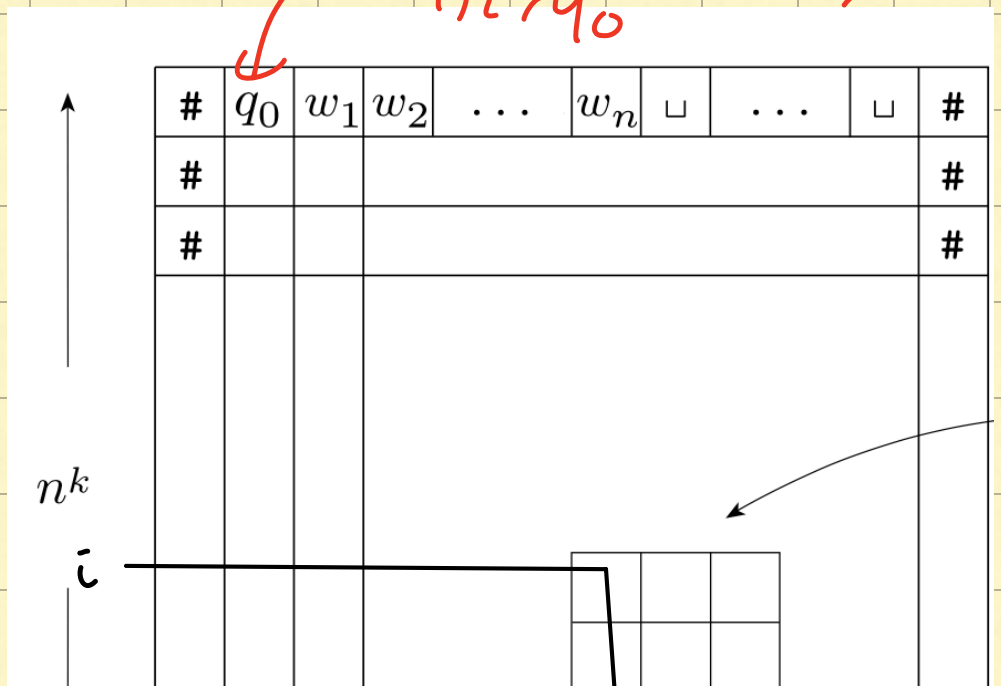
A tableau is accepting if for some time $t \leq n^k$, at time t the NTM is an accept configuration (N is in q_{accept})

Key: There is an accepting tableau of N on w iff N accepts w in some branch:

so this is iff $w \in A$

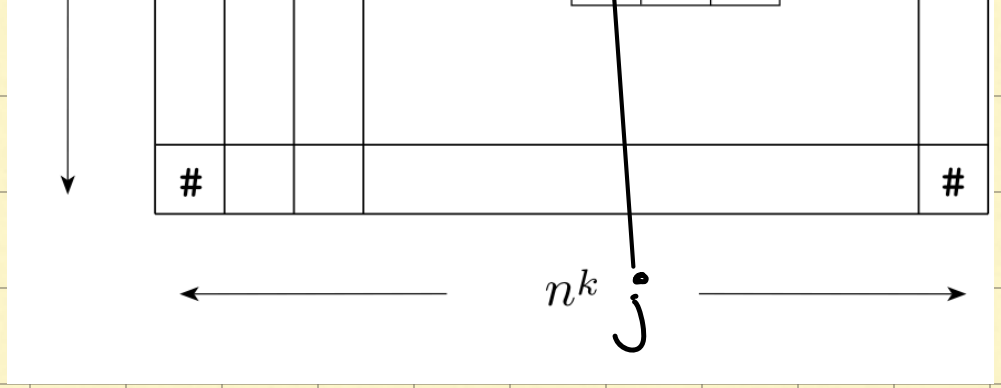
We want a polytime computable $f: \Sigma^* \to \Sigma^*$ such that $\forall w \in \Sigma^*$ w has an accepting tableau on N iff $f(w) \in SAT$. We need to build a CNF ϕ

$x_{1,2,q_0} = 1$, $x_{1,2,\#} = 0$



cell(i,j) has a symbol

$s \in \Gamma \cup Q \cup \{\#\}$



For each cell (i,j) & $s \in P \cup Q \cup \{\#\}$

a variable $x_{i,j,s}$ which is 1 iff

the cell contains s

So we will have $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$

ϕ_{cell}: state that each cell contains exactly one symbol

$$s \in Q \cup \Gamma \cup \{\#\}$$

ϕ_{start}: state the initial configuration is:

$$\# \; q_0 w_1 \cdots w_n \sqcup \sqcup \cdots \sqcup \#$$

ϕ_{accept}: Some cell contains q_{accept}) the tableau must be accepting

ϕ_{move}: Each row of the tableau at time t follows from the row at time $t-1$

by following some transition of N

$\#\, q_0\, w_1\, w_2\, w_3\, \sqcup\, \sqcup\, \sqcup\, \#$

$\#\, w_1\, w_2\, w_3\, q_2\, \sqcup\, \sqcup\, \sqcup\, \#$

follows 1 step of the NTM N

Ex: $\phi_{cell} = \bigwedge_{i,j} \phi_{cell(i,j)}$

$$\phi_{cell(i,j)} = \left(\bigvee_{s \in \Gamma \cup Q \cup \{\#\}} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,s' \\ s \neq s'}} \overline{x_{i,j,s}} \vee \overline{x_{i,j,s'}} \right)$$

each cell has at least 1 symbol

never 2 symbols

False

If $x_{i,j,s} = 1$ and $x_{i,j,s'} = 2$

ϕ_{start}: start config is $\# q_0 w_1 \cdots w_n \sqcup \cdots \sqcup \#$

$\Rightarrow \quad x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n}$

q_0 $\quad$ w

$\wedge x_{1,n+3,\sqcup} \wedge \cdots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$

blanks

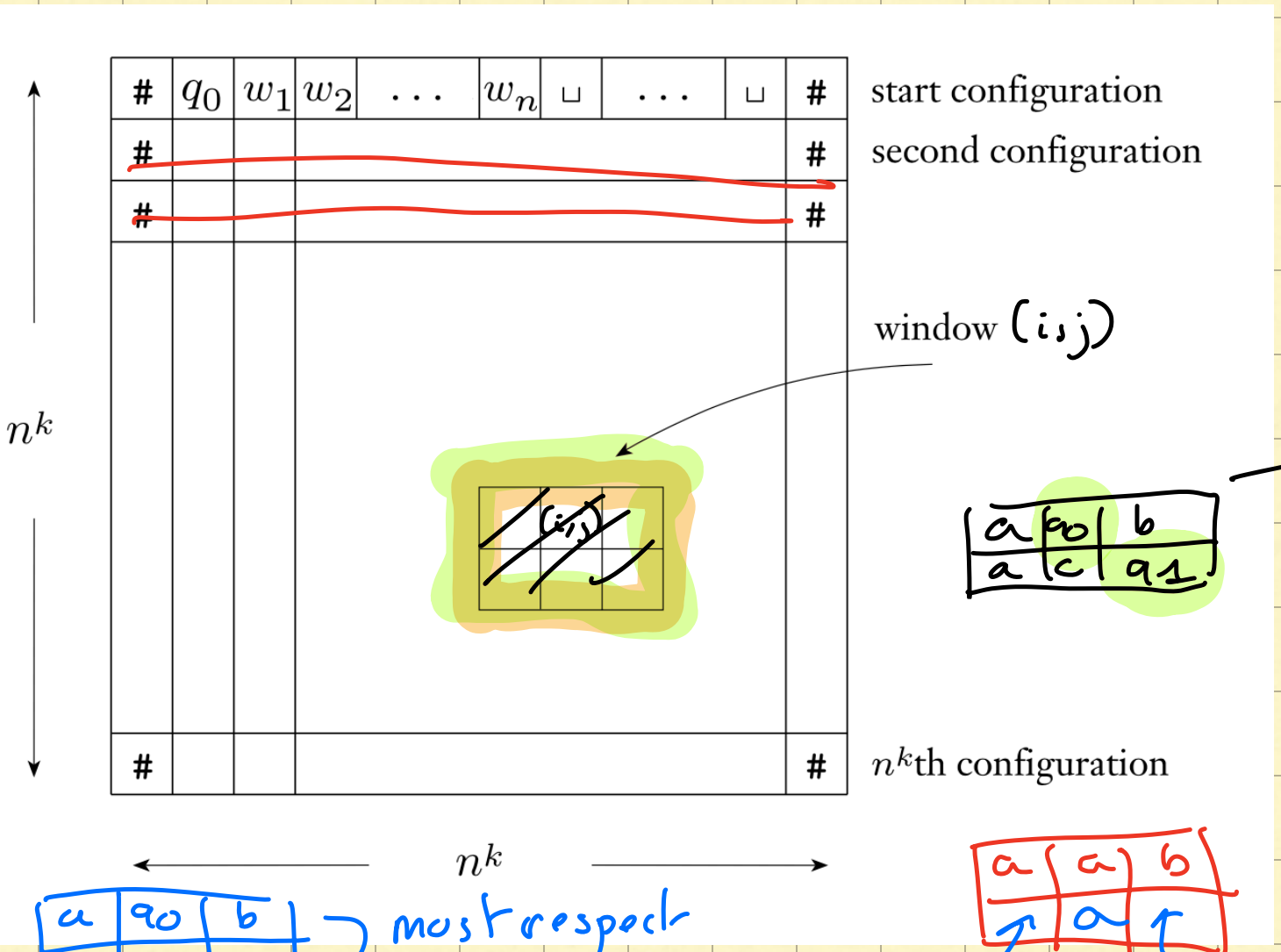
ϕ_{accept} = some cell contains an accept state

$$= \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

some cell (i,j) contains q_{accept}

↳ the tableau is accepting

ϕ move: $\bigwedge_{1 \le i,j < n^c}$ the (i,j) window is legal



a window is a 2x3 rectangle of cell

↳ encode changes in configurations

↳ the TM went L|R, changed state, updated the tape.

For each window : a CNF to encode

it's a legal change in configuration

↳ so each row follows legally from the previous one

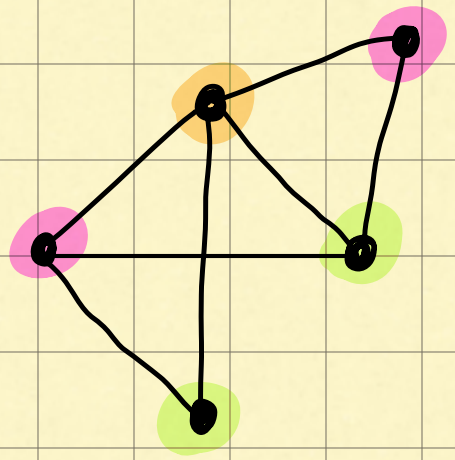
See claim 7.41 + book for details

(1) Building ϕ takes polytime in $|w|$

(2) ϕ is satisfiable iff there is an accepting of N on w.

↑ tableau

Coloring : $\{\langle G, k\rangle \mid$ we can color G using colors $1, \ldots, k$ & no adjacent vertices have the same color $\}$



$\Leftarrow$ 3 colorable

(1) We already showed this was in NP

(2) We show 3-SAT $\leq_T$ Coloring

$f(\phi) = \langle G_\phi, 3 \rangle$, where G_ϕ is a graph

↳ n variables
m clauses

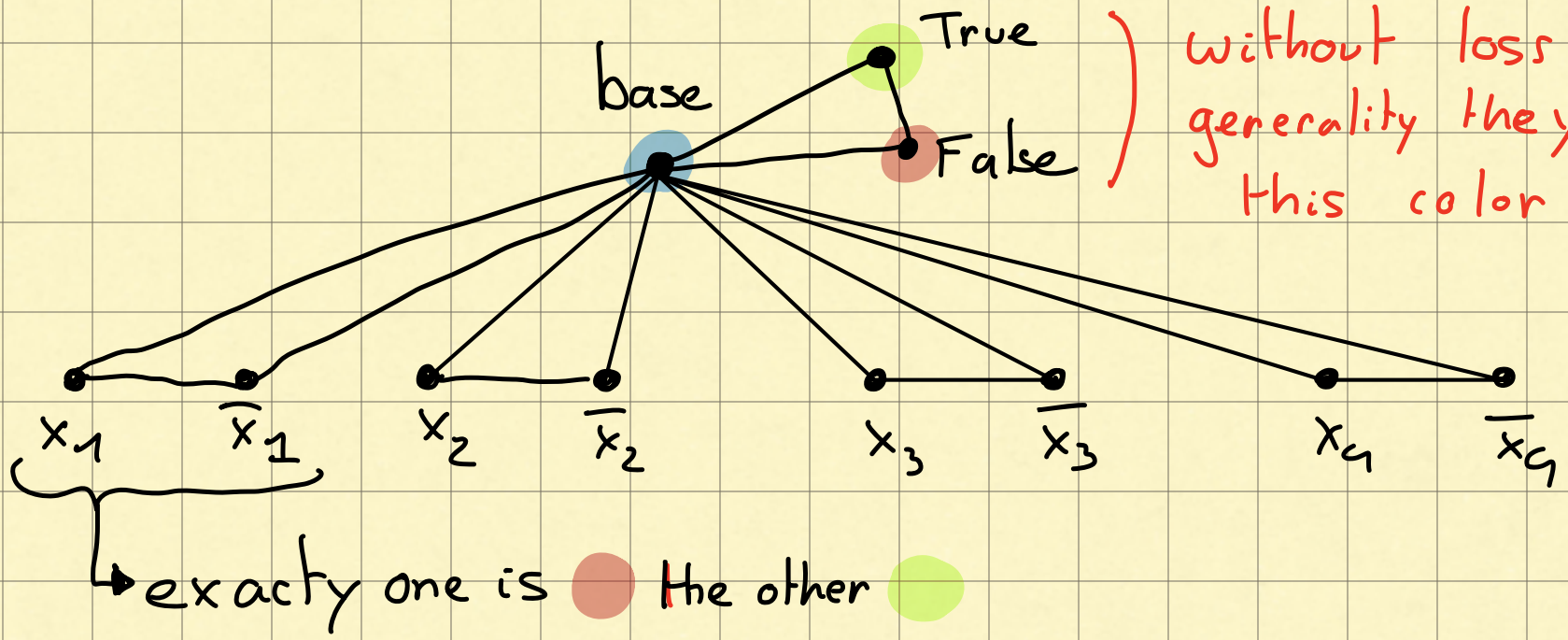
This one is a bit tricky ...

7.29 in the book as an exercise!

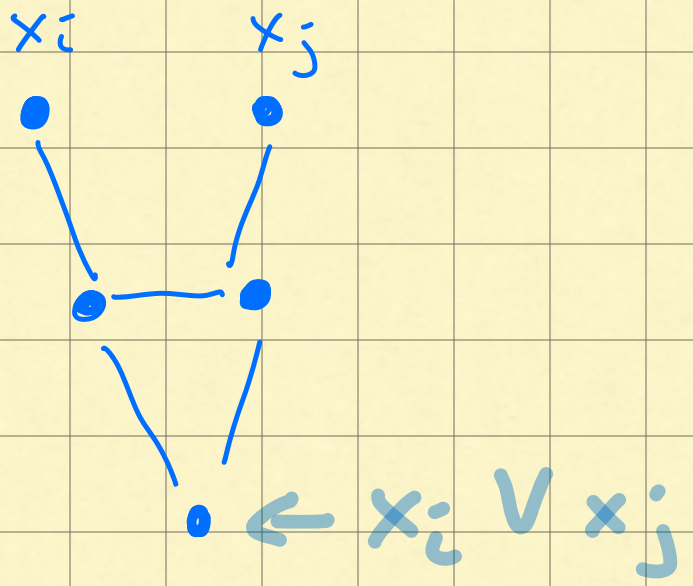
An example:

$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$

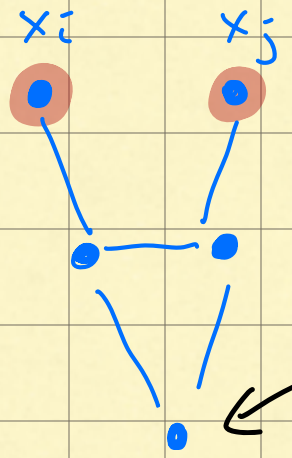
We want to simulate 0/1 assignments to the variables using 3 colors T F N



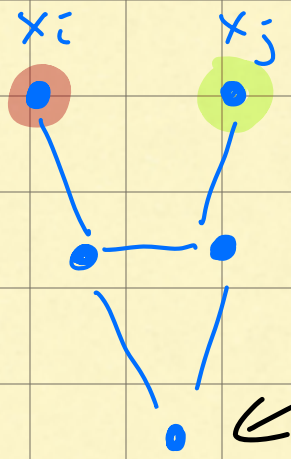
How to encode clauses ?



Both x_i, x_j are

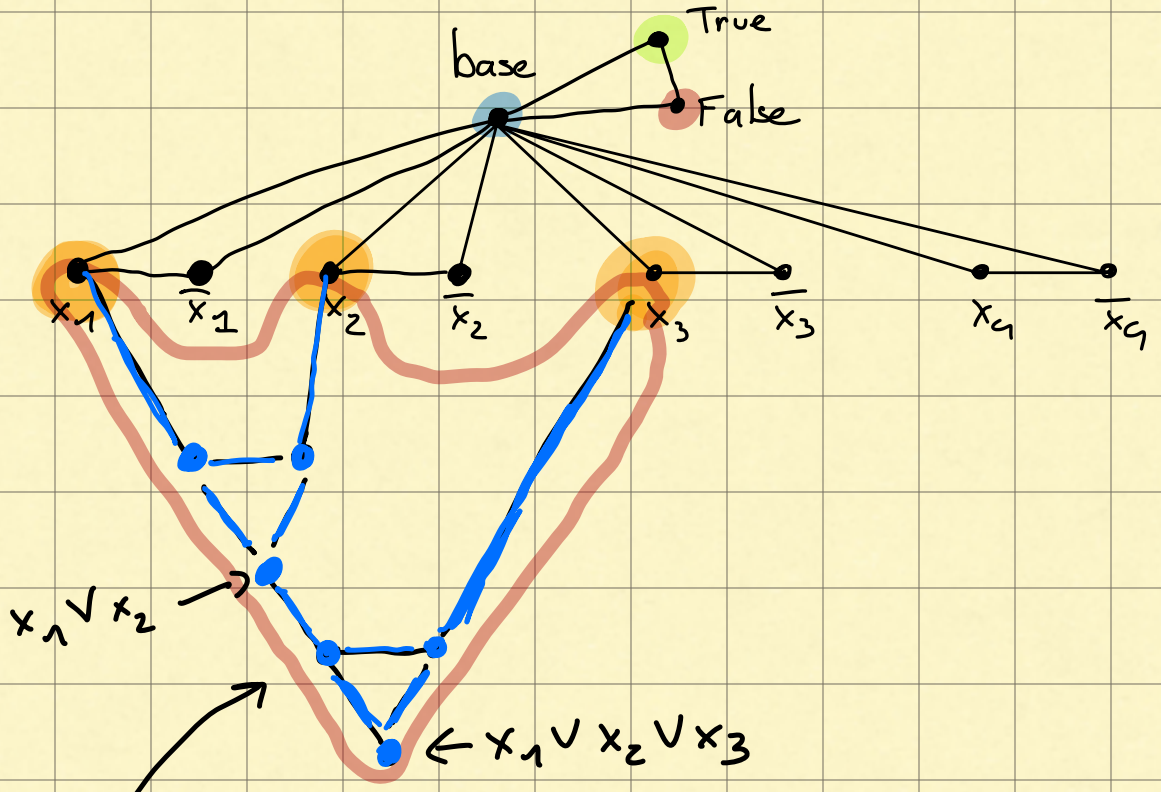


One of x_i, x_j is green



An example:

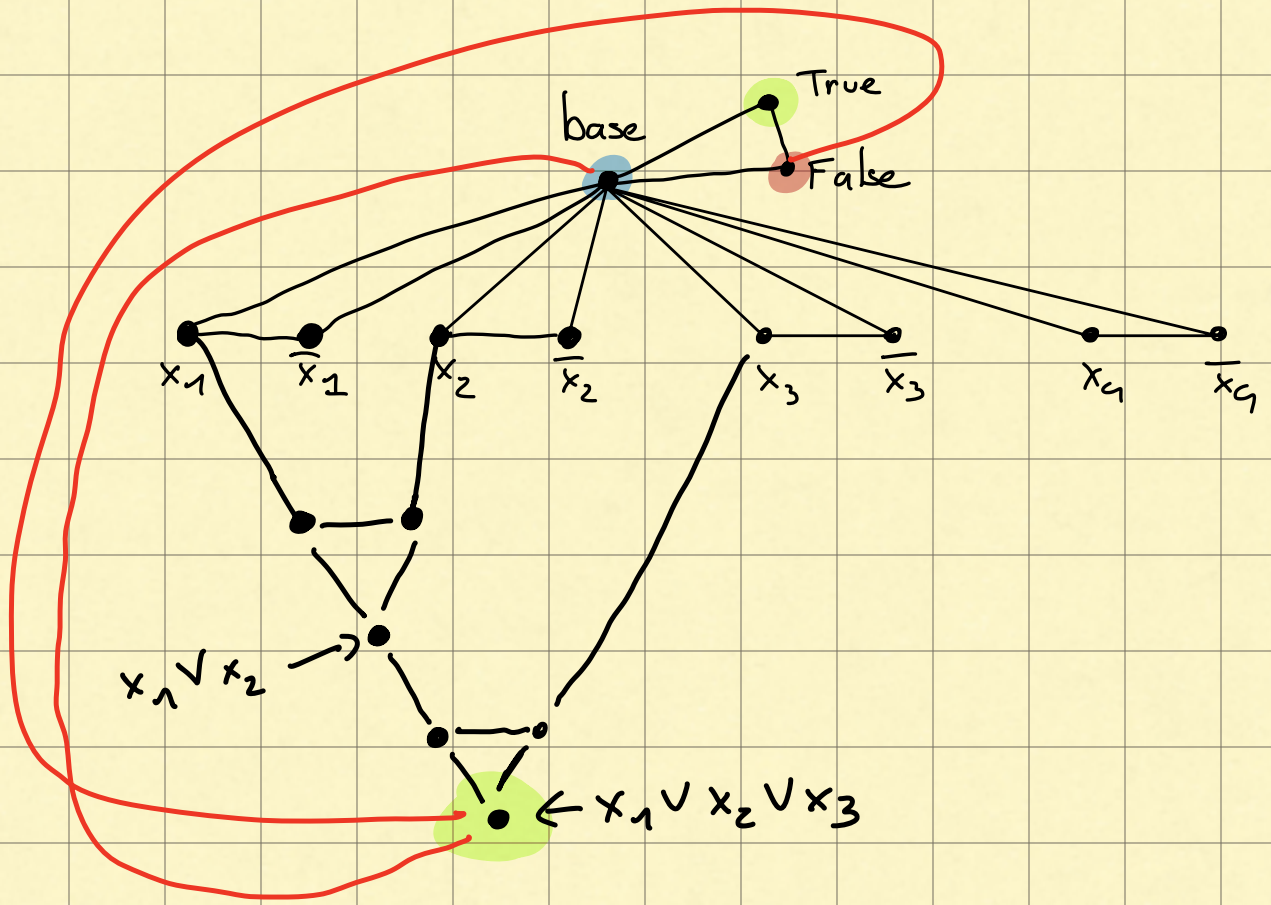
$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$



use OR gadget to encode clauses!

An example:

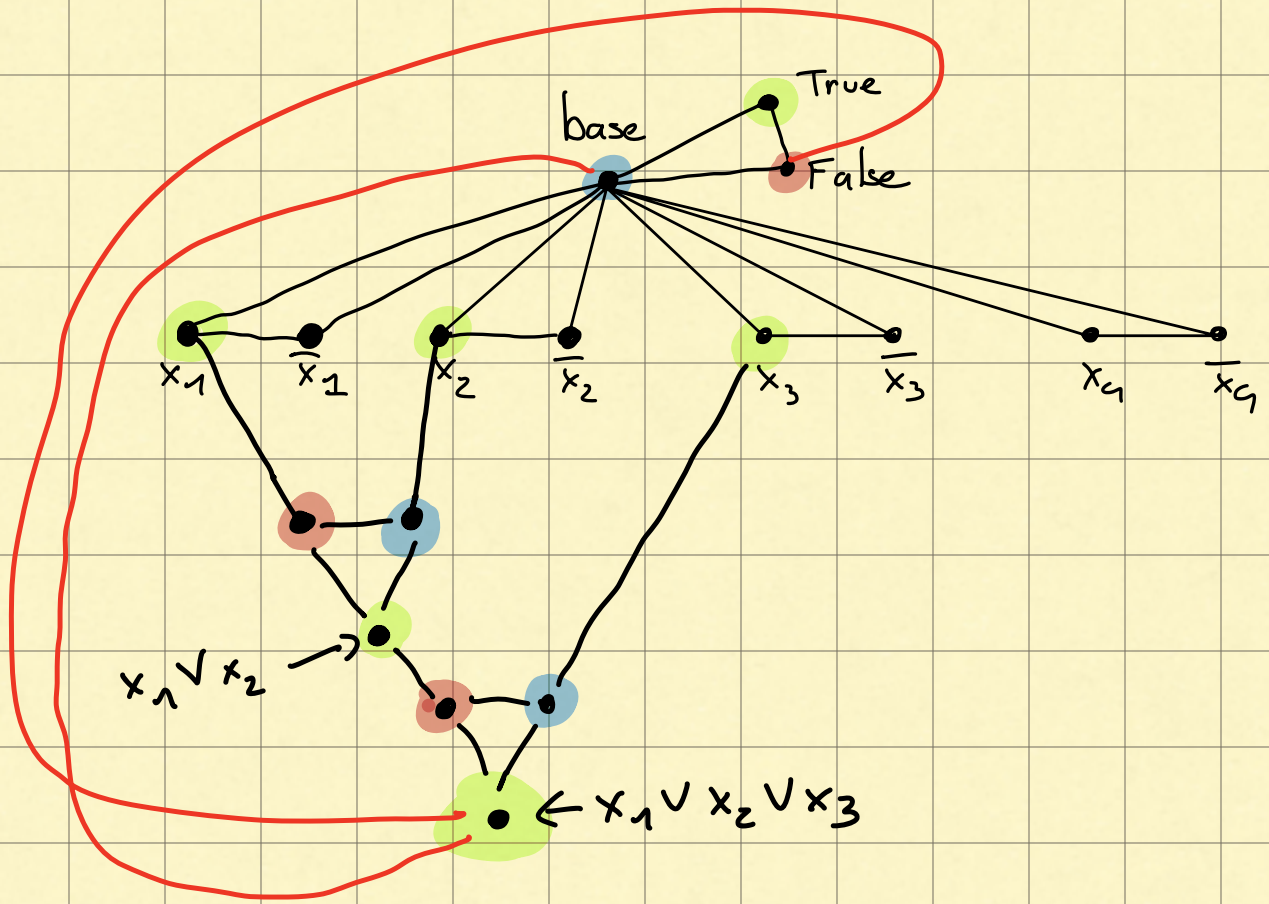
$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$



force $x_1 \vee x_2 \vee x_3$ to be (true

An example:

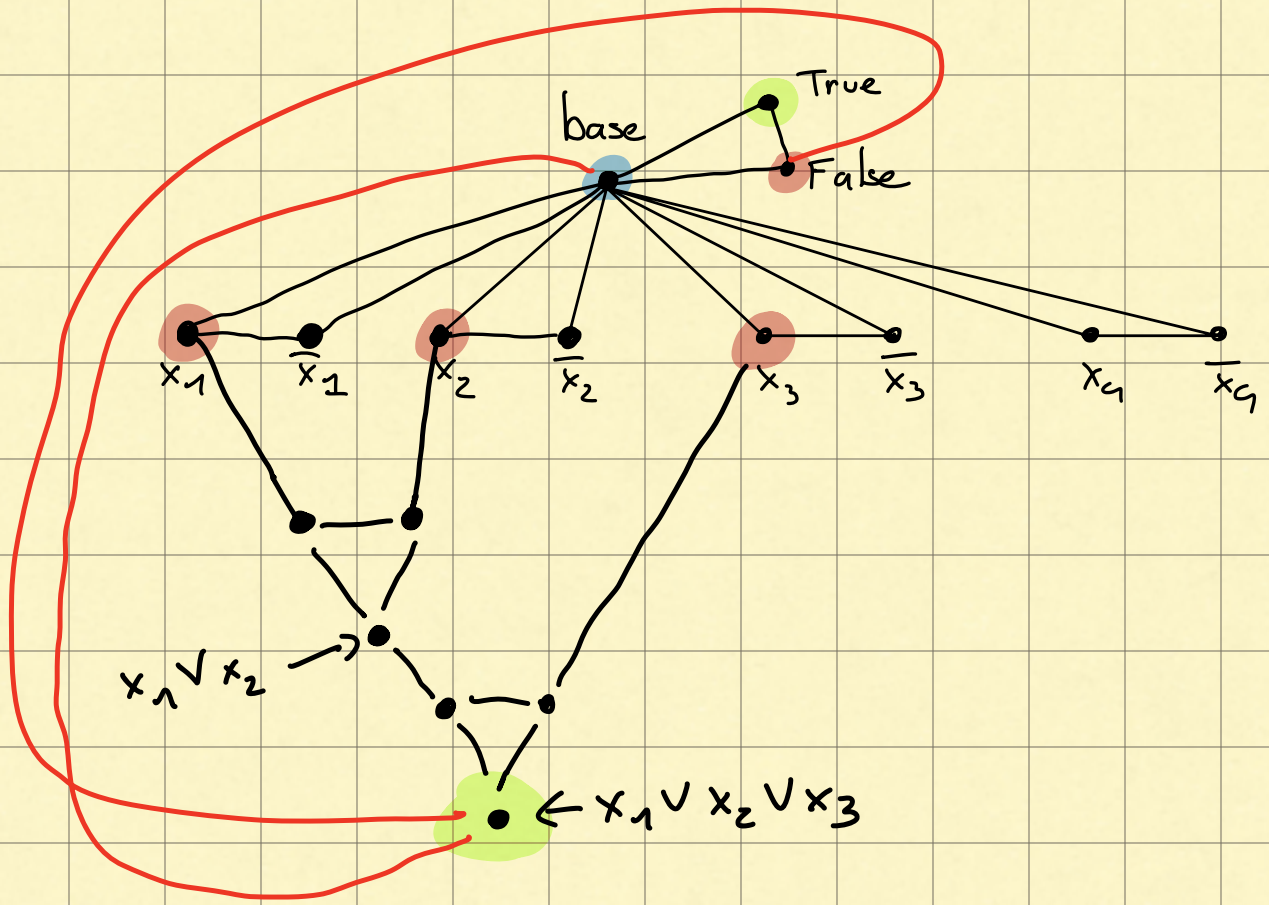
$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$



Say x_1, x_2, x_3 are all true

An example:

$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$



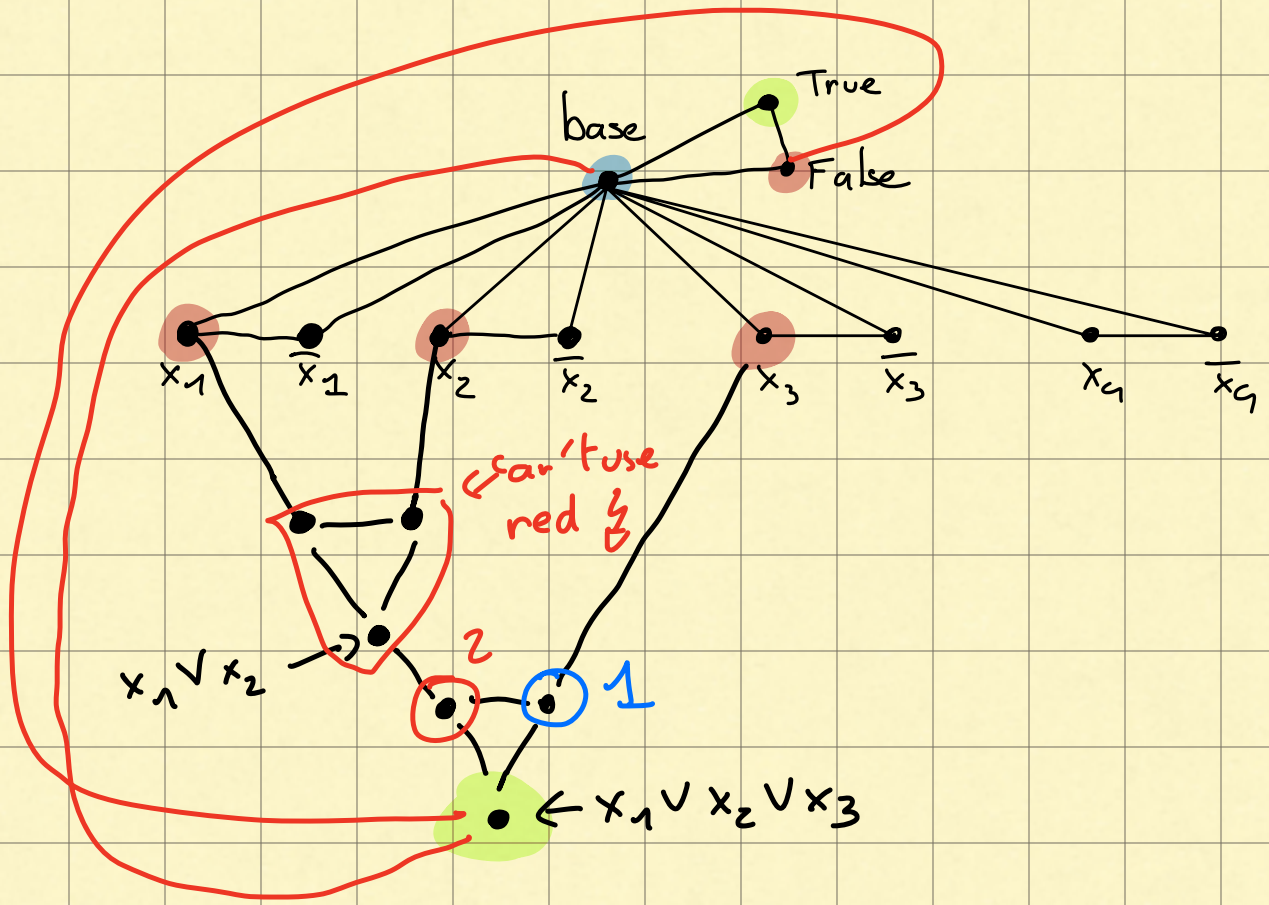
Say x_1, x_2, x_3 are all false...

Impossible!

An example:

For me

$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee x_3)$$



Say x_1, x_2, x_3 are all false...

Impossible!

So use the same gadget trick for each clause... to build G_ϕ

(1) We can build G_ϕ in poly time

(2) We can show that G_ϕ is 3-colorable iff ϕ has a satisfying assignment