Complexity + Algorithms, Spring 2026

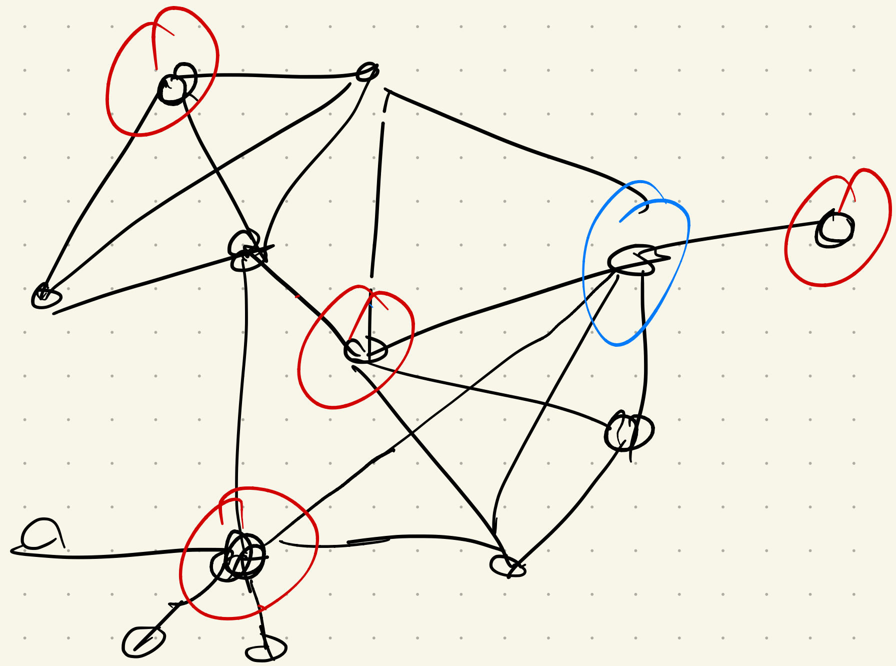
Reductions

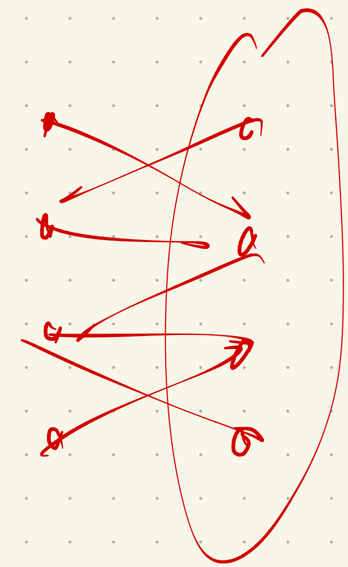
Independent Set:

A set of vertices in a graph with no edges between them:

easy to get small set

↳ maximization



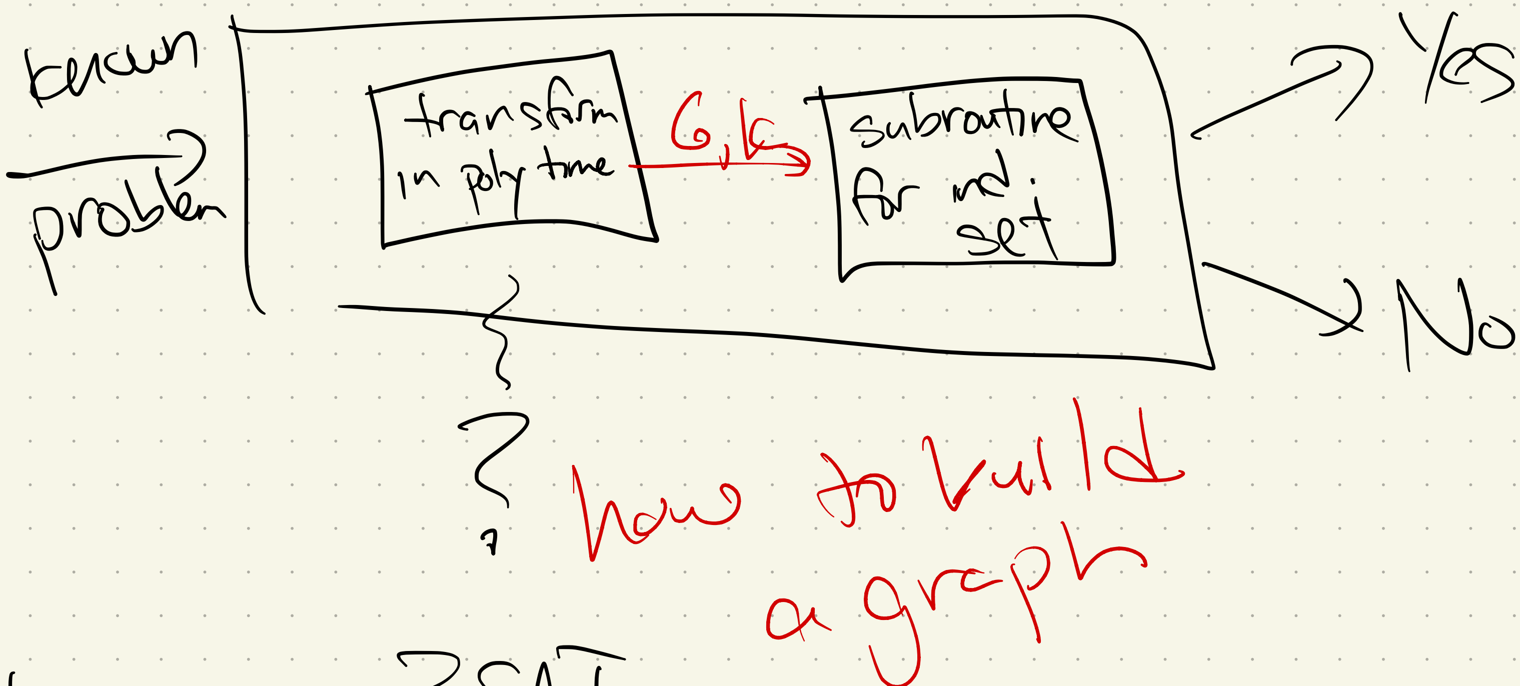


Decision version: Given G & $k \in \mathbb{Z}^+$, does $\exists$ an indep set of size k?

(in NP)

Challenge: No booleans!

But reduction needs to take known NP-Hard problem + build a graph!



We'll use 3SAT

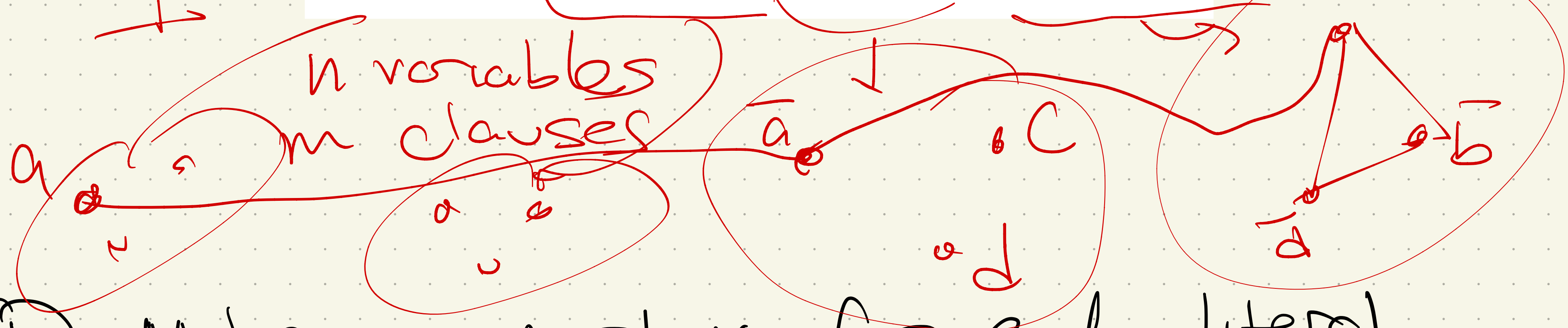
(but stop and marvel a bit first...)

Reduction:

Build G & determine k (k = m)

Input is 3CNF boolean formula

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



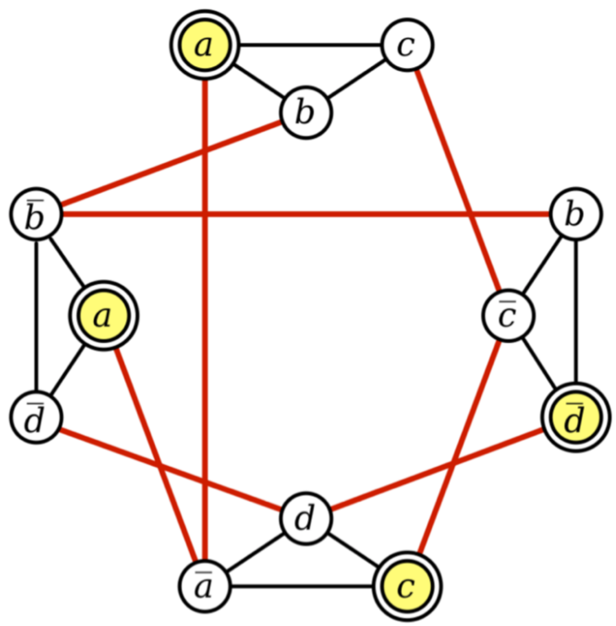
① Make a vertex for each literal in each clause

② Connect two vertices if:

- they are in same clause
- they are a variable & its inverse

Example:

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



A graph derived from a 3CNF formula, and an independent set of size 4.

intuition: choose one T variable per clause in SAT assignment

↳ m vertices

Claim:

Φ formula is Satisfiable

$\Longleftrightarrow$

G has independent set of size $\geq m$

$\Rightarrow$: Assume Φ is satisfiable

↳ choose T/F values so each clause has ≥ 1 true. Pick one per clause.

↳ select correspond $v \in V(G)$:

This set is indep, & size m.

One per clause $\Rightarrow$ 1 vertex per $\triangle$.

and if x is set to true, $\bar{x}$ is not (in Φ), so no clause will choose vertex that is connected outside its $\triangle$.

⇐: Suppose G has indep set of size m.
Know at most one vertex per "clause △". Why? (all connected)
Take that vertex ∈ set corresponding variable to be true.
Each clause has 1 var. true, + no var + its negation can both be true
(rest of variables can be either T/F)

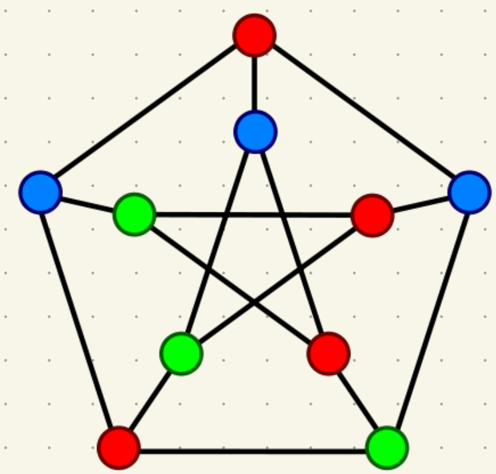
⇒ satisfying assignment.

∎

Next: Graph Coloring

A **k-coloring** of a graph G is a map: $c: V \to \{1, \ldots, k\}$ that assigns one of k "colors" to each vertex so that every edge has 2 different colors at its endpoints

Goal: Use few colors



Aside: this is famous!
Ever heard of map coloring?

Famous theorem: For planar graphs, 4 colors suffice.

Thm: 3-colorability is NP-Complete.

(Decision version: Given G
output yes/no)

In NP:

certificate: color assignment for V

Loop through all edges $e = uv$,
& check that $c(u) \neq c(v)$.

Polytime: $O(E)$

NP-Hard:

Reduction from 3SAT.

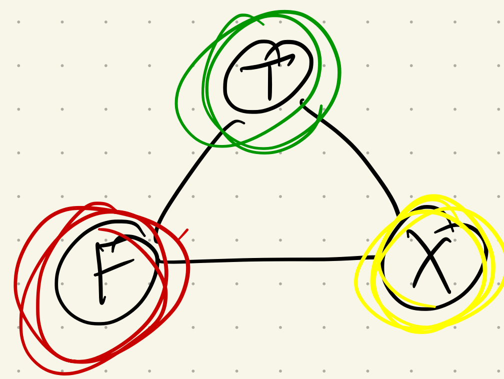
Given formula for 3SAT Φ,

we'll make a graph G_Φ.

Φ will be satisfiable

$\iff G_\Phi$ can be 3-colored.

Key notion: Build "gadgets"!

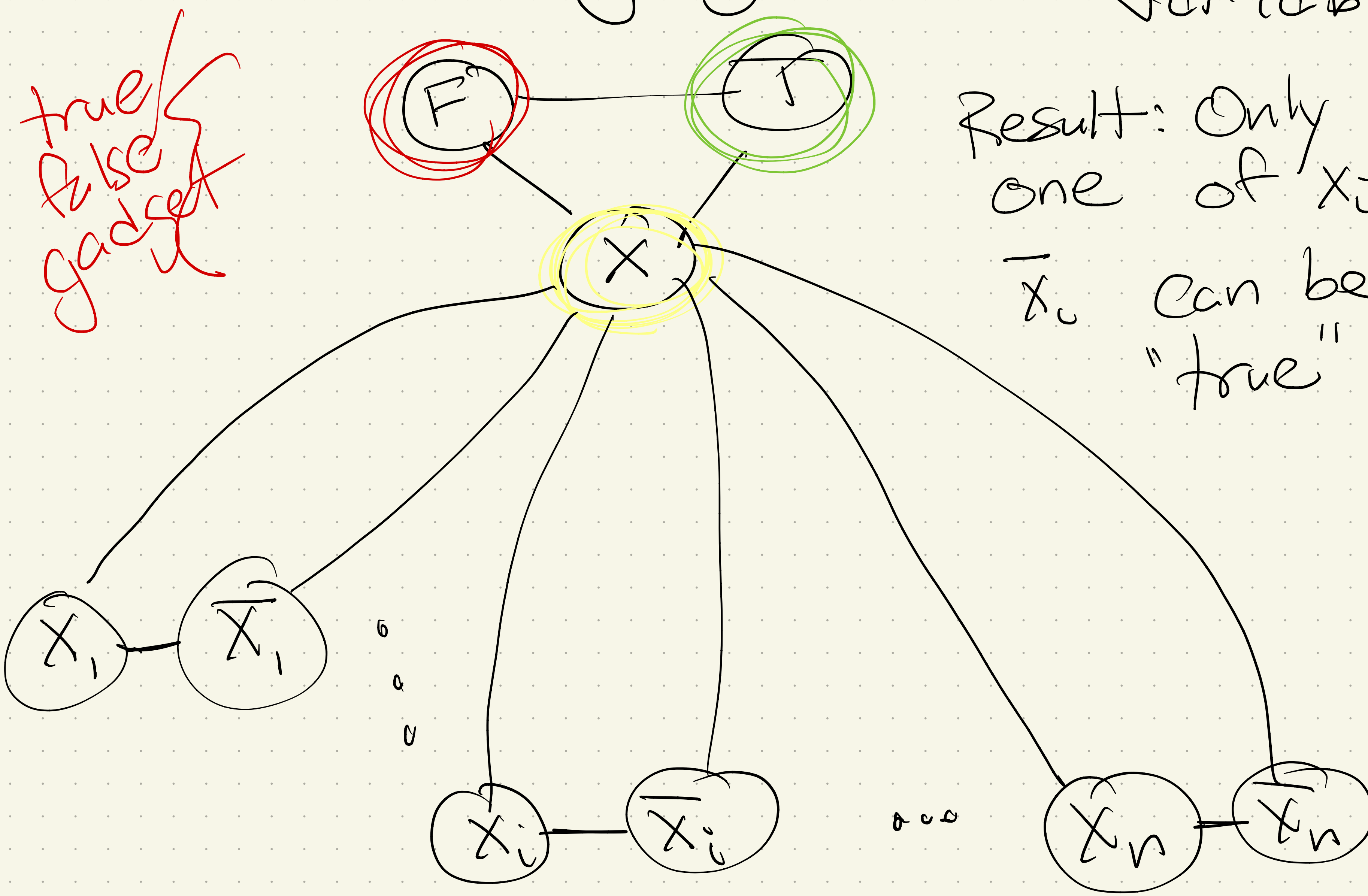


(1) Truth gadget -

Must use 3 colors -

so establishes a "true" color.

(2) Variable gadget: one per variable

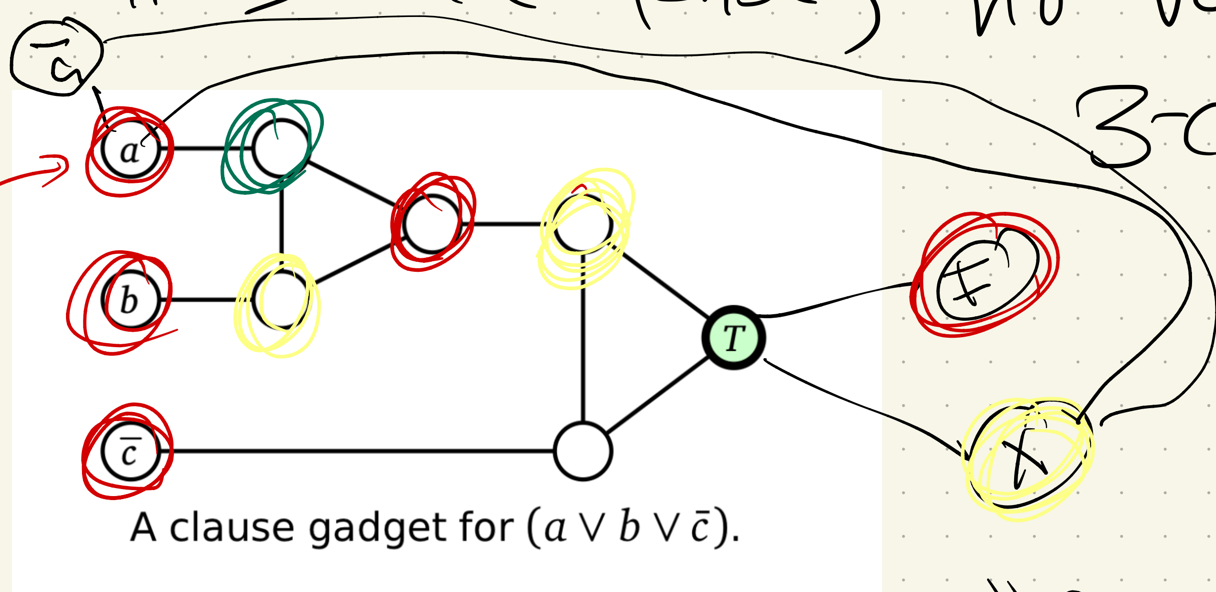


Result: Only one of X_i + $\overline{X_i}$ can be "true"

③ Clause gadget:

For each clause, join 3 of the variable vertices to the "true" vertex from the truth gadget.

Goal: If all 3 are false, no valid 3-coloring

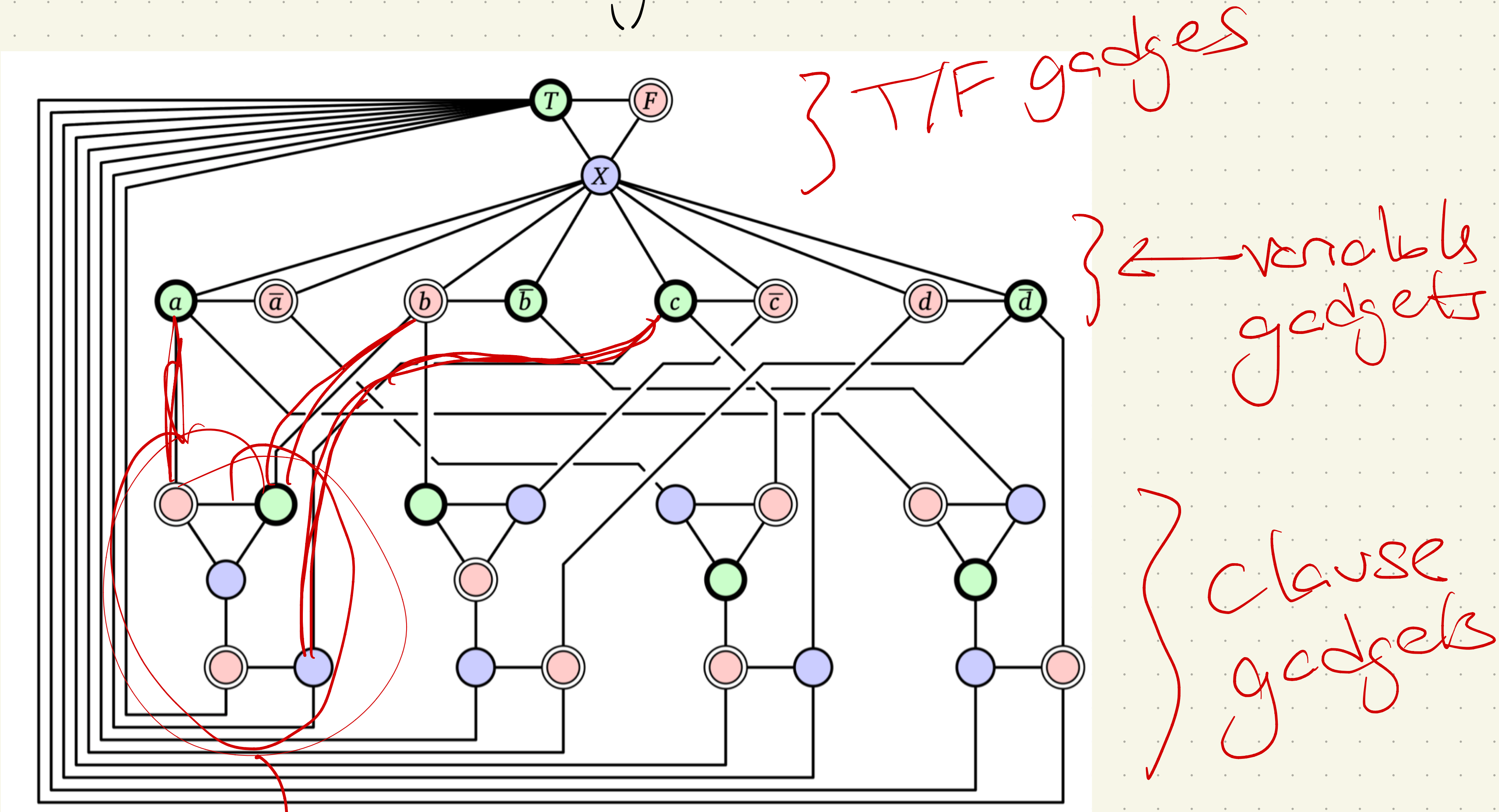


A clause gadget for $(a \vee b \vee \bar{c})$.

if all variable are red cannot 3color

Why?? try to color all "false"

Final reduction image:



A 3-colorable graph derived from the satisfiable 3CNF formula

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

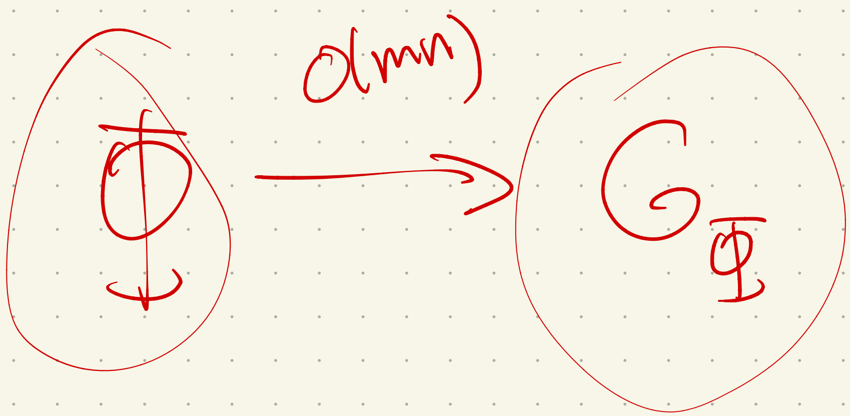
Now, need reduction proof:

Construction is polynomial:

G has: $3 + 2n + 5m$ vertices $= |V|$

& $\leq |V|^2$ edges: loop through clauses & add

$\Rightarrow$ poly size, & takes poly time to build



3 colorings of G_Φ $\Longleftrightarrow$ Φ is satisfiable

Pf:

$\Rightarrow$: Consider a 3-coloring of G:

↳ one vertex per clause must be green.

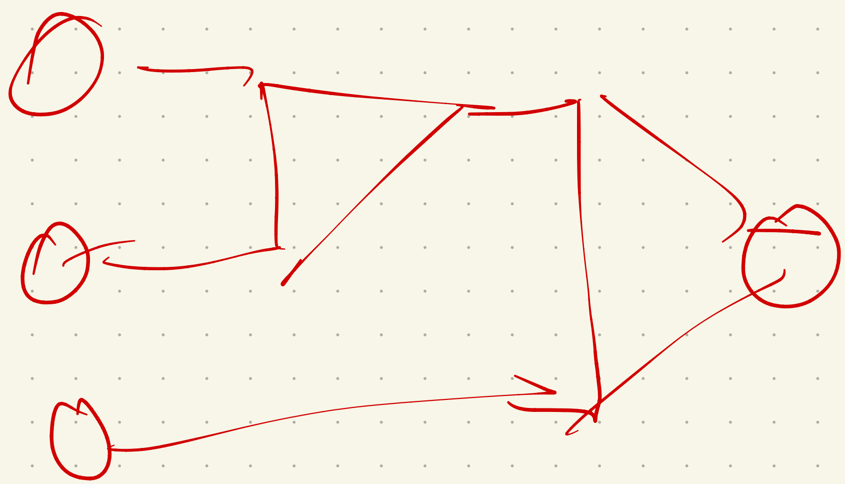
So set variables T/F according to red/green.

Know one per clause will be true.

$\Leftarrow$ Consider a satisfying assignment to Φ:

Color variables x_i & $\overline{x_i}$ in G_Φ accordingly.

& then 7 cases show extend coloring through clause gadgets.



$\blacksquare$

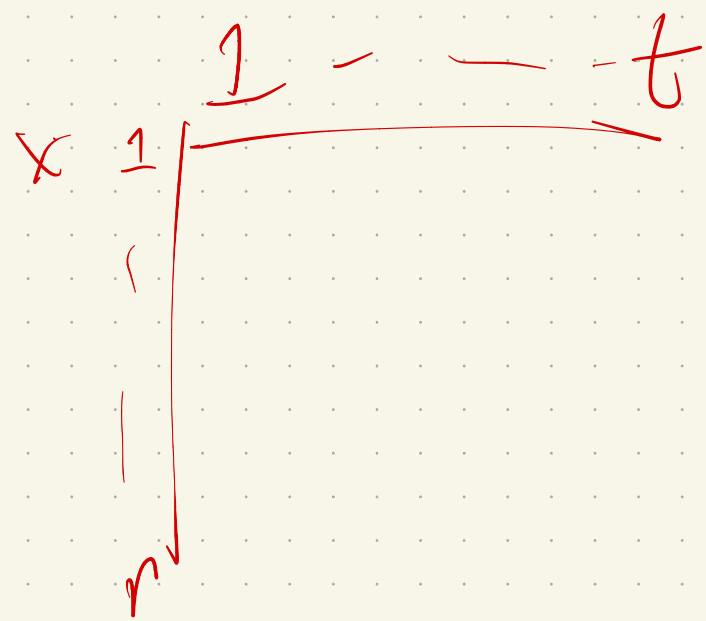
Subset Sum:

Given a set of numbers $X = \{x_1, \ldots, x_n\}$ and a target t, does some subset of X sum to t?

Ex: Actually did this one!
See lecture from Ch. 2

Runtime: $O(nt)$

not polynomial in input: $t \to \log t$



Subset Sum is NP-Hard.

Reduction: Vertex Cover

Input: Graph G & size k

Goal: find k vertices, such that every edge in G is incident to at least one vertex in set

Challenge: Construct a set of numbers, s.t. we can hit a target value $\iff$ G has vertex cover of size k

Recall: base 4

$(32012)_4 = 2 + 1 \cdot 4^1 + 0 \cdot 4^2 + 2 \cdot 4^3 + 3 \cdot 4^4 =$

Idea: Use base 4:
force a target T that requires you to use only vertices, but to "cover" edges

Number edges $0 \ldots E-1$, & create a number for subset sum with E digits:

e_0: $b_0 = 00 \cdots 001$

e_1: $b_1 = 00 \cdots 010$

⋮

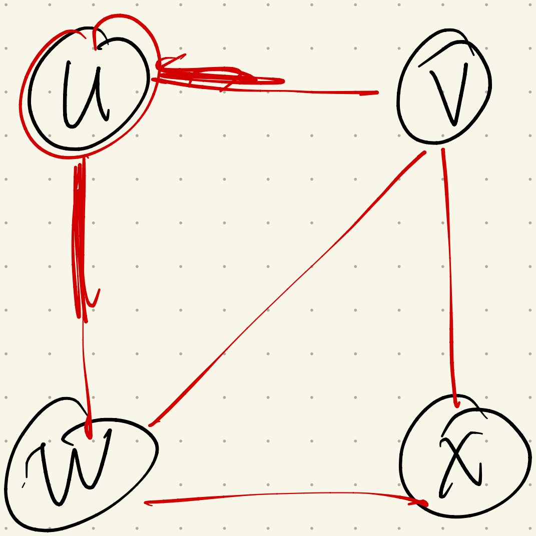
e_{E-1}: $b_{E-1} = 010 \cdots 0$

For each vertex, make another #:

$$a_v := \underset{E}{1}\ \underset{E-1}{\_\_}\ \underset{E-2}{1}\ \cdots\ \underset{2}{\_\_}\ \underset{1}{\_\_}\ \underset{0}{\_\_}$$

↳ any edge incident, put 1 in that spot

Think of base 4 representation



$$\begin{aligned} a_u &:= 111000_4 = 1344 \\ a_v &:= 110110_4 = 1300 \\ a_w &:= 101101_4 = 1105 \\ a_x &:= 100011_4 = 1029 \end{aligned} \qquad \begin{aligned} b_{uv} &:= 010000_4 = 256 \\ b_{uw} &:= 001000_4 = 64 \\ b_{vw} &:= 000100_4 = 16 \\ b_{vx} &:= 000010_4 = 4 \\ b_{wx} &:= 000001_4 = 1 \end{aligned}$$

#s: E digits in base 4

one per vertex

1 ___ (0's + 1's)

E's

4^E

1 ___

one per edge

0 ___ (exactly 1 1's)

E

?

Now, set $T = k \cdot 4^E + \sum_{i=0}^{E-1} 2 \cdot 4^i$

Why? forces choosing k vertices

If select 2 endpts of edge, you have $2 \cdot 4^i$ edge i

If not → choose edge i #

Proof: size k VC ⟺ sum to T

⇒ VC: ∃ k vertices $v_1, v_2, \ldots, v_k$

s.t. $\forall e \in E$, e is incident to some $v_i \in \{v_1, \ldots, v_k\}$

Take those k #s

have $k \cdot 4^{E}$

& either $1 \cdot 4^{i}$ or $2 \cdot 4^{i}$

for each $i < E$

b/c every edge has at least

one endpoint in cover

If edge i has only one

endpoint, add b_{ei} to subset.

T

$\Rightarrow$ (cont)

Pick a subset:

must $= k 4^{\bar{t}} + \sum_{i=0}^{\bar{t}-1} 2 \cdot 4^{i}$

$\hookrightarrow$ can only get $k \cdot 4^{\bar{t}}$ if we have k vertex #s

$\Leftarrow$: suppose some subset of #s sums to T. options?

Recall: $T = k \cdot 4^E + \sum_{i=0}^{E-1} 2 \cdot 4^i$

+ a_r = 1 1 1 1

+ b_e = 1

Plus:

Each digit position has only 3 1's across all #s:

no carrying!

So!

those k vertices must touch all edges

Time to convert: $G=(V,E)$

$V+E$ numbers, each has

E digits

& T has $\le k+E$ digits

Partition:

Given $X = \{x_1, \ldots, x_n\}$, can we partition X into A & B (so $A \cup B = X$, $A \cap B = \emptyset$, & $A, B \neq \emptyset$)

s.t.

$$\sum_{x_i \in A} x_i = \sum_{x_j \in B} x_j \text{ ?}$$

Reduction? From subset sum

Input: $X = \{x_1, \dots, x_n\}$ + target T

Build set of #s s.t.

$\exists$ subset $= T$ in X

$\iff$ Partition the new #s.

Idea: somehow add target in set

Let $W = \sum_i x_i$

$x_1, \dots, x_n, T, W-T$

add 2 #s

Subset of the x_i:

$= T$

Rest of the x_i:

$W-T$

$W-T$ and T together:

$= W$

Proof:

Some fun examples

arXiv.org > cs > arXiv:1203.1895

Search or Article ID inside arXiv | All papers | Broaden your search usin

(Help | Advanced search)

Computer Science > Computational Complexity

Classic Nintendo Games are (Computationally) Hard

Greg Aloupis, Erik D. Demaine, Alan Guo, Giovanni Viglietta

(Submitted on 8 Mar 2012 (v1), last revised 8 Feb 2015 (this version, v3))

We prove NP-hardness results for five of Nintendo's largest video game franchises: Mario, Donkey Kong, Legend of Zelda, Metroid, and Pokemon. Our results apply to generalized versions of Super Mario Bros. 1-3, The Lost Levels, and Super Mario World; Donkey Kong Country 1-3; all Legend of Zelda games; all Metroid games; and all Pokemon role-playing games. In addition, we prove PSPACE-completeness of the Donkey Kong Country games and several Legend of Zelda games.

Comments: 36 pages, 36 figures. Fixed some typos. Added NP-hardness results (with proofs and figures) for American SMB2 and Zelda 2
Subjects: **Computational Complexity (cs.CC)**; Computer Science and Game Theory (cs.GT)
Cite as: arXiv:1203.1895 **[cs.CC]**
(or arXiv:1203.1895v3 **[cs.CC]** for this version)

Submission history

From: Alan Guo [view email]
[v1] Thu, 8 Mar 2012 19:37:20 GMT (627kb,D)
[v2] Thu, 6 Feb 2014 18:24:15 GMT (3330kb,D)
[v3] Sun, 8 Feb 2015 19:45:26 GMT (3425kb,D)

Which authors of this paper are endorsers? | Disable MathJax (What is MathJax?)

Link back to: arXiv, form interface, contact.

Dow
- PD
- Otl

(license)

Curre
cs.CC
< prev
new | r

Chan
cs
cs.(

Refer
- NA

6 blog

DBLF
listi
Gre
Eril
Ala

Book

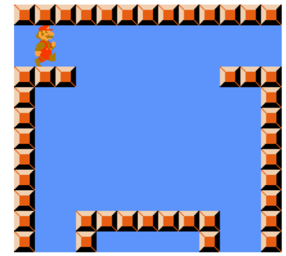


Figure 10: Variable gadget for Super Mario Bros.

shes until it is collected by Mario.

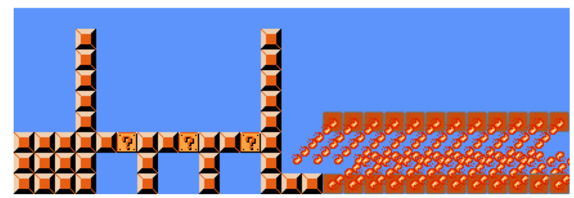


Figure 11: Clause gadget for Super Mario Bros.

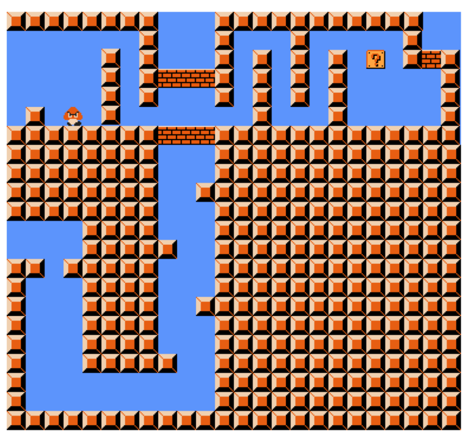
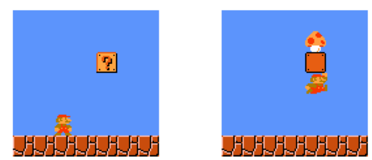


Figure 12: Crossover gadget for Super Mario Bros.



Left: Start gadget for Super Mario Bros. Right: The item block contains a

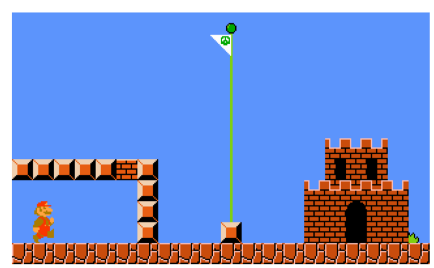
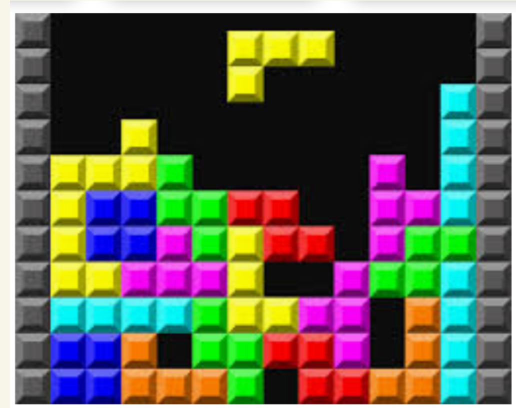


Figure 9: Finish gadget for Super Mario Bros.

Another: Tetris



NP-Hard: Reduce 3-partition

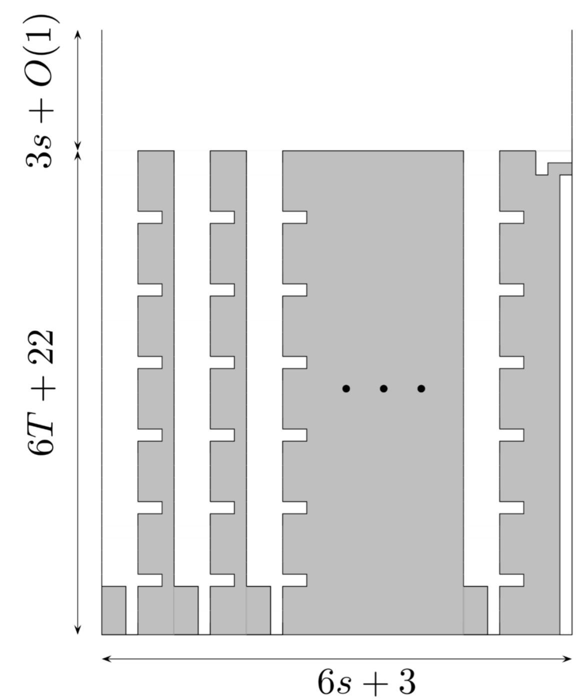


Fig. 2. *The initial gameboard for a Tetris game mapped from an instance of* 3-PARTITION.

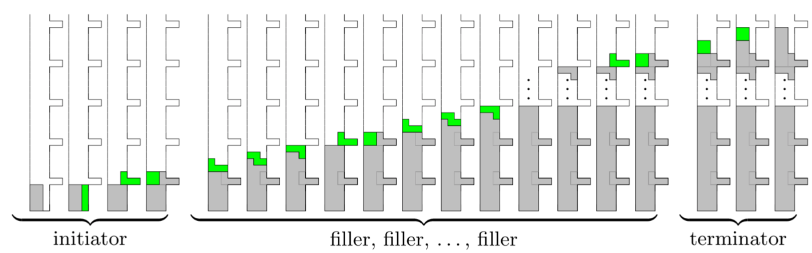


Fig. 3. *A valid sequence of moves within a bucket.*

Next week: some hardness of approximation, plus more reductions.

(Readings posted.)