# Announcement

Please see Piazza announcement about Quiz 2 problem 2.

# Prove that STRONGLY INDEPENDENT SET is $\mathcal{NP}$-complete

STRONGLY INDEPENDENT SET is in $\mathcal{NP}$:

Cert: $V'$

Verifier:

If $V' \not\subseteq V$ or $|V'| < K$, reject

for each $\{u,v\} \subseteq V'$

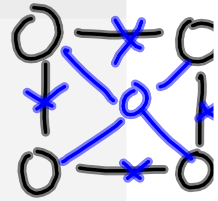
if $\exists e = (u,v) \in E$ or $\exists w \in V$ s.t. $(u,w) \,\&\, (w,v) \in E$

reject

accept

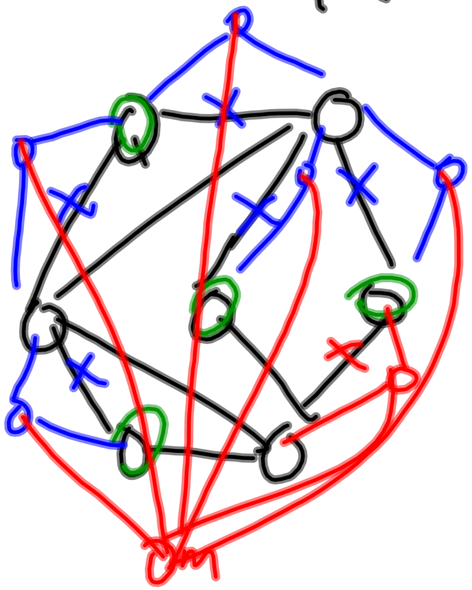
## Prove that STRONGLY INDEPENDENT SET is $\mathcal{NP}$-complete

G



IndSet(G, k)
{
G' ← copy of G. Add vertex m.
for each e=(a,b) ∈ G'.E
delete e
add (a,m), (b,m) to G'.

close...




IndSet (G, k)

$G'$ = copy of G

m = new vertex add to $G'$

for each $e = (u, v)$ in $G'$

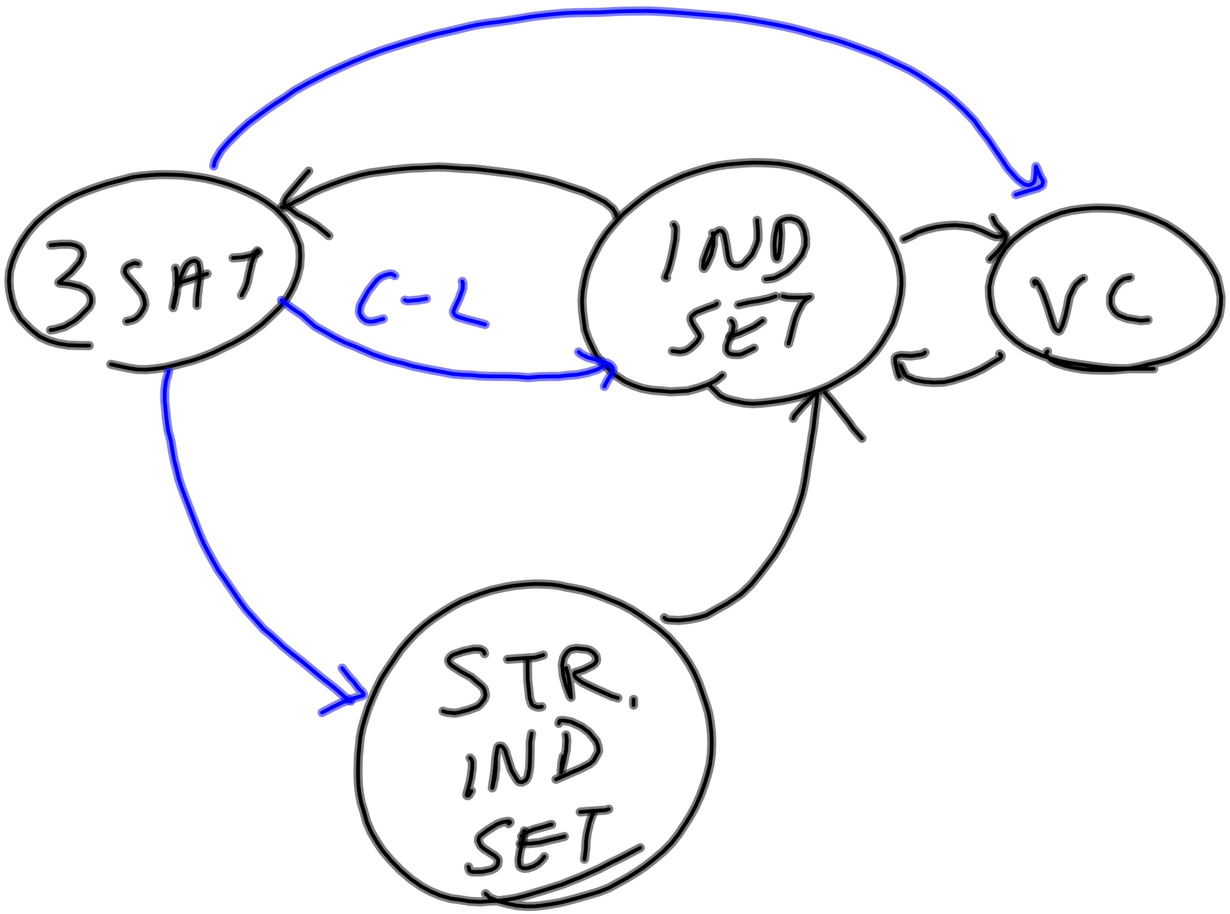
delete e

create "uv" vertex to $G'$

add edges (u, uv) (v, uv) to $G'$

connect (m, uv) w/ edge

return Str. Ind Set ($G'$, k)



3SAT
C-L
IND SET
VC
STR. IND SET

3-SAT($\frac{1}{2}$)

- Select arbitrary T.V.A.
- count true clauses
- if $> \frac{1}{2}$ are false
  invert the T.V.A.

- return that T.V.A.

## Prove that 3-SAT($\frac{15}{16}$) is $\mathcal{NP}$-complete

3-SAT($\frac{15}{16}$) is in $\mathcal{NP}$:

As per proof 3-SAT $\in \mathcal{NP}$,
except
- count true clauses
- compare that to $\frac{15k}{16}$.

## Prove that 3-SAT$(\frac{15}{16})$ is $\mathcal{NP}$-complete

3-SAT( )
for each 8 clauses
Create 3 new vars
Create ALL 8 clauses on new vars

Call 3-SAT$(\frac{15}{16})$ w/ ~~new~~ amended vars / clauses

~~Call 3-SAT$(\frac{15}{16})$ on new clause set~~

"all 8 clauses?"

$A, B, C$:

| | |
|---|---|
| $A \vee B \vee C$ | $\bar{A} \vee B \vee C$ |
| $A \vee B \vee \bar{C}$ | $\bar{A} \vee B \vee \bar{C}$ |
| $A \vee \bar{B} \vee C$ | $\bar{A} \vee \bar{B} \vee C$ |
| $A \vee \bar{B} \vee \bar{C}$ | $\bar{A} \vee \bar{B} \vee \bar{C}$ |

# Prove that Hitting Set is $\mathcal{NP}$-complete

Hitting Set is in $\mathcal{NP}$:

Certificate: $H$

Verifier:

1. for each $B_i$:
   if $H \cap B_i = \emptyset$
   reject
2. if $|H| \neq k$, reject
3. If $H \not\subseteq A$, reject
4. accept

VC verifier

for ea $e = (u,$
if $u, v$ not. ch
rej

## Prove that HITTING SET is $\mathcal{NP}$-complete

VertexCover(G, k)

for each $e = (u, v)$
create $B_e = \{u, v\}$

$A = V$
return HittingSet(A, B, K)

# Hamiltonian Paths

In a graph, a HAMILTONIAN PATH is a simple path that includes every *vertex*.

- We will start with directed graphs, and seeking a path.
- This is also true for undirected graphs.
- It is also true for seeking a cycle (in a directed or undirected).

## Prove that HAMILTONIAN PATH is in $\mathcal{NP}$

Certificate: $P$, a permutation of $V$

Verifier:

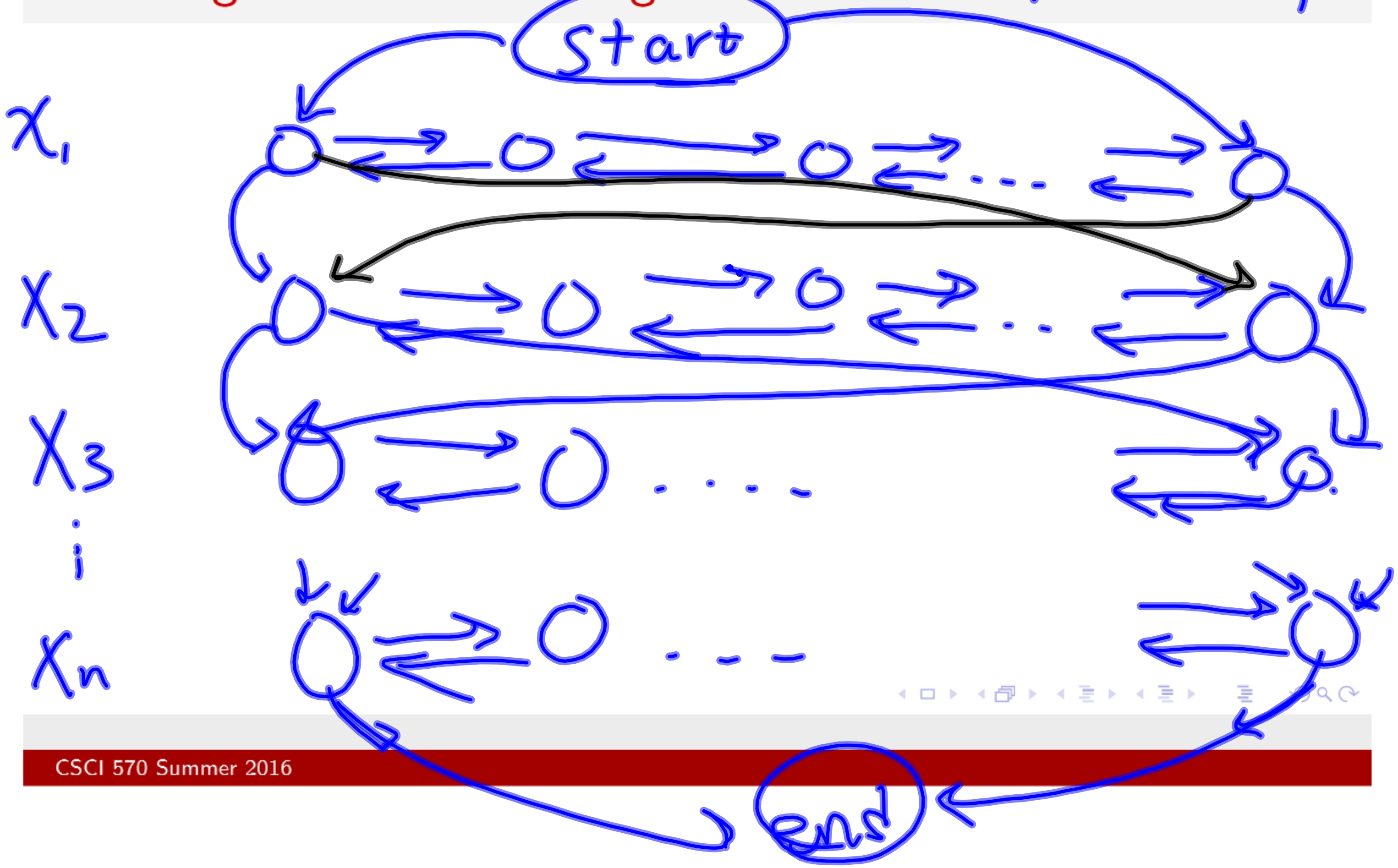
1. If any vertex omit or $2^+$x, reject
2. for $i=1$ to $n-1$
   if $e=(P_i, P_{i+1}) \notin G.E$
   reject
3. accept

# Creating a truth value assignment

- Design a graph so that any Hamiltonian Path in it will correspond to a truth value assignment on $n$ variables. Do not (yet) worry about *satisfying* assignments.
- How many Hamiltonian Paths are in your graph?
- What does any given Hamiltonian Path mean as a truth-value assignment?

# Creating a truth value assignment

row: L → R : "true"
R → L : "false"

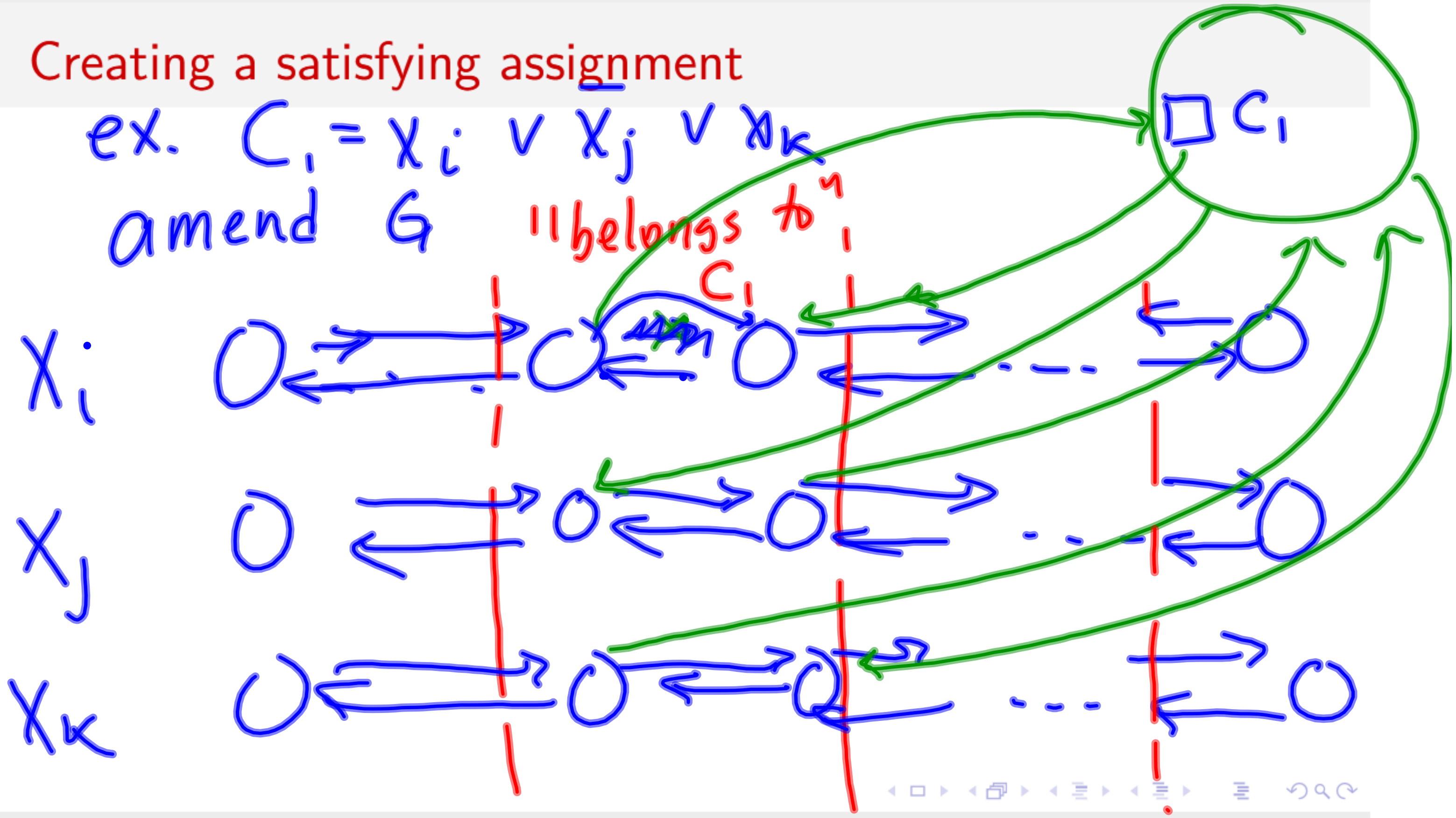


# Creating a satisfying assignment

ex. $C_1 = x_i \vee \bar{x}_j \vee x_k$

amend G

"belongs to" $C_1$



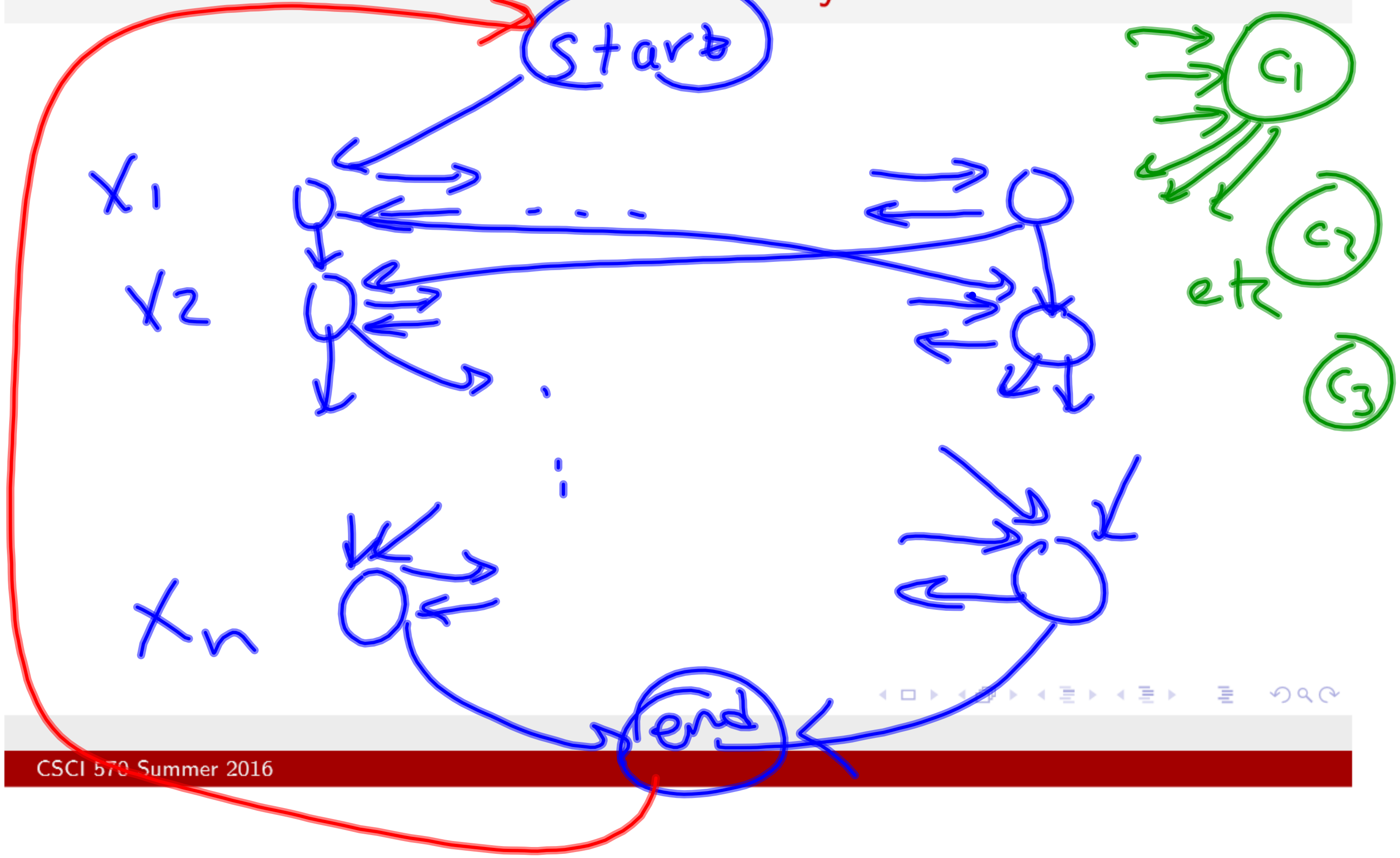
# Putting it together...

Could this produce *false negatives*?
That is, could there be an instance of 3-SAT that has a satisfying assignment, but with the corresponding HAMILTONIAN PATH (that we create) causing a "false"?

# Putting it together...

Could that create *false positives*?
That is, if HAMILTONIAN PATH returns true, do we really know that the corresponding 3-SAT instance has a satisfying assignment?

# What about directed Hamiltonian Cycle?



# What about undirected graphs?

# Prove that Bipartite Directed Hamiltonian Cycle is $\mathcal{NP}$-complete

Bipartite Directed Hamiltonian Cycle is in $\mathcal{NP}$:

Same as DHC

## Prove that Bipartite Directed Hamiltonian Cycle is $\mathcal{NP}$-complete

Dir H.Cyc(G)
{
// Create bipartite G' that has
// HC iff G does
for each v ∈ G
"split" v in G':

$$v \Longrightarrow v_{in} \rightarrow v_{OUT}$$

all incoming

all out edges

# Prove SUBSET SUM is in $\mathcal{NP}$

# Prove 3-Color is in $\mathcal{NP}$

# Use SUBSET SUM to get a truth value assignment on $n$ variables

$x_1$ true

$x_1$ F

| | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| $v_1$ | 1 | 0 | 0 |
| $v_1'$ | 1 | 0 | 0 |
| $v_2$ | 0 | 1 | 0 |
| $v_2'$ | 0 | 1 | 0 |
| $v_3$ | 0 | 0 | 1 |
| $v_3'$ | 0 | 0 | 1 |
| T | 1 | 1 | 1 |

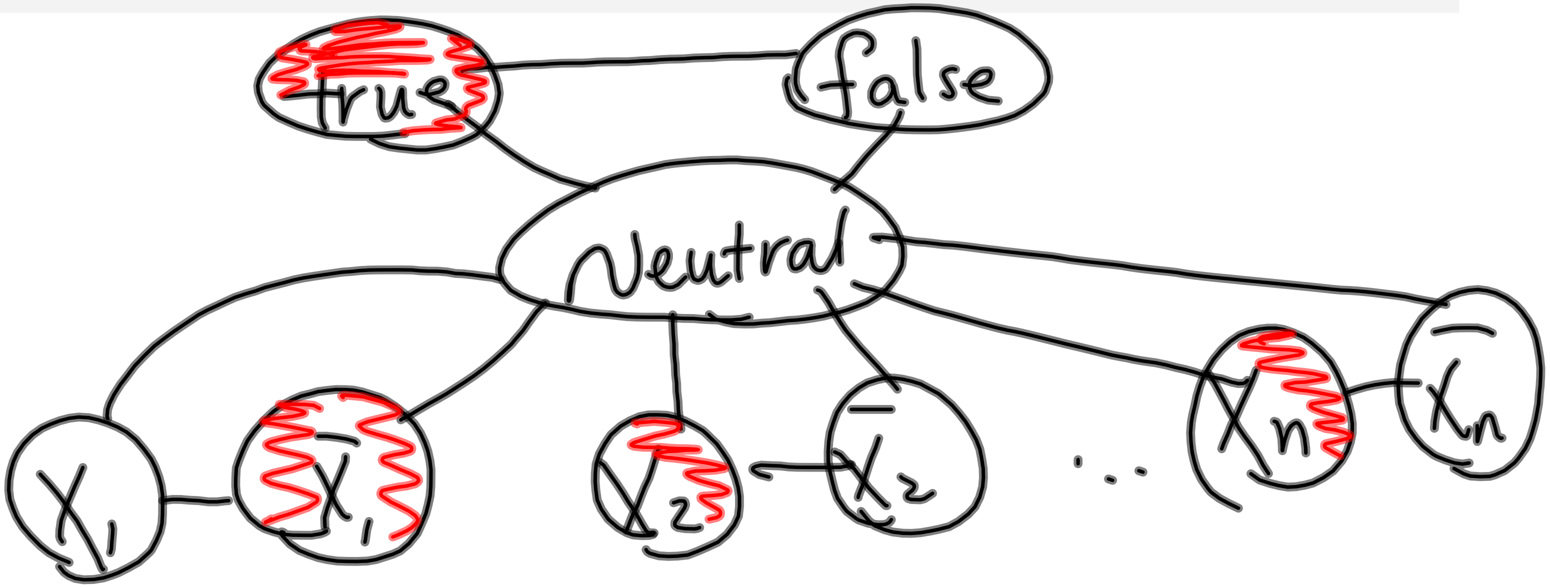
as binary,

$S = \{ 4_{A'}, 4_{B'}, 2_{A'}, 2_{B'}, 1_{A'}, 1 \}$

B

$T = 7$

# Use 3-COLOR to get a truth value assignment on $n$ variables



Amend the SUBSET SUM usage to get a satisfying truth value assignment on $n$ variables

$\phi = (x_1 \vee \overline{x_2} \vee \overline{x_3})(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})(\overline{x_1} \vee \overline{x_2} \vee x_3)(x_1 \vee x_2 \vee x_3)$

Suppose instead of each clause having $\vee$ it was $\oplus$.

| | $x_1$ | $x_2$ | $x_3$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ |
|---|---|---|---|---|---|---|---|
| $v_1$ | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $v_1'$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| $v_2$ | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| $v_2'$ | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| $v_3$ | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| $v_3'$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| T | 1 | 1 | 1 | X | X | X | X |

red: $\vee$ not $\oplus$ "1 or 2 or 3"

Amend the SUBSET SUM usage to get a satisfying truth value assignment on $n$ variables

$\phi = (x_1 \vee \overline{x_2} \vee \overline{x_3})(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})(\overline{x_1} \vee \overline{x_2} \vee x_3)(x_1 \vee x_2 \vee x_3)$

"slack" But then again, each clause has $\vee$, not $\oplus$.

see prev

| | $x_1$ | $x_2$ | $x_3$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ |
|---|---|---|---|---|---|---|---|
| $s_1$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $s_1'$ | 0 | 0 | 0 | 2 | 0 | 0 | 0 |
| $s_2$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $s_2'$ | 0 | 0 | 0 | 0 | 2 | 0 | 0 |
| $s_3$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $s_3'$ | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| $s_4$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $s_4'$ | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| T | 1 | 1 | 1 | 4 | 4 | 4 | 4 |

"1–3"

# Putting it together...

Remember, our algorithm is:

- We are given an instance of 3-SAT
- We create the corresponding $2n$ "boolean variables"
- We create $2k$ "clause variables"
- We create a value $T$
- We call any correct implementation of SUBSET SUM

Could this produce *false negatives*?
That is, could there be an instance of 3-SAT that has a satisfying assignment, but with the corresponding SUBSET SUM (that we create) causing a "false"?

# Putting it together...

Could that create *false positives*?
That is, if SUBSET SUM returns true, do we really know that the corresponding 3-SAT instance has a satisfying assignment?

How big is that Subset Sum?

$|S| = 2n + 2K$

how many bits ea?

If treat as base 8

$3(n+K)$

T is also $3n+3K$ bits

value $\approx 2^{3n+3K}$

DP: $\Theta(|S| \cdot T) \Rightarrow$ in terms of orig 3SAT?

$\Theta\left((n+K)\, 2^{3n+3K}\right)$