

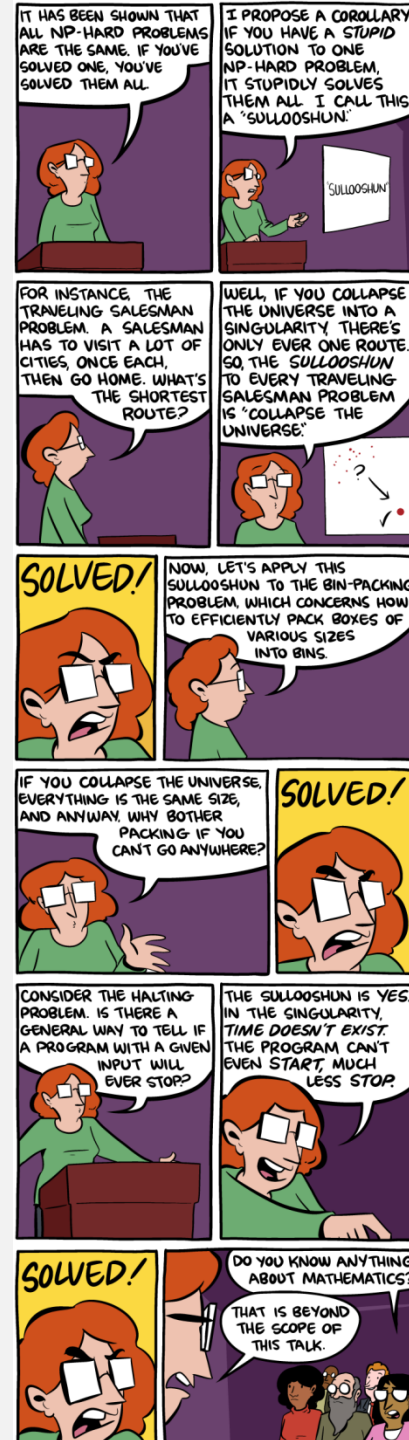
More NP-Complete Problems

Monday, May 10, 2021

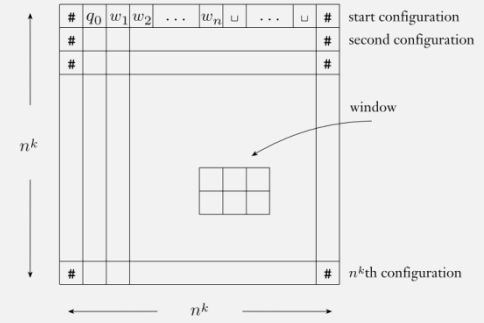


Announcements

- Last HW (HW12) due Wed 11:59pm EST
- HW11 grades will not be returned before Wed
- Taking suggestions for last lecture ...
 - Solutions to specific homework questions?
- Course evaluations coming (check piazza)
 - Fill out during next class

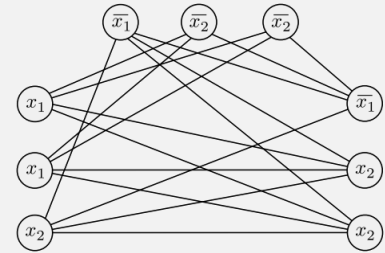


NP-Complete problems, so far



- $SAT = \{\langle \phi \rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$ (Cook-Levin Theorem)

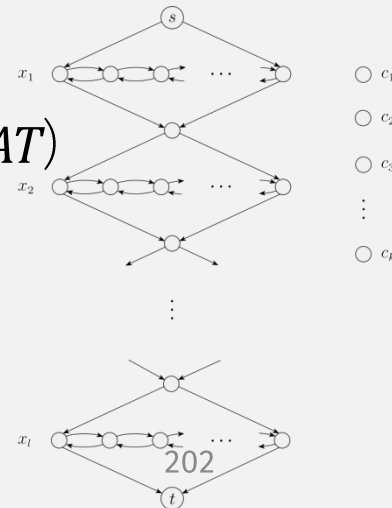
- $3SAT = \{\langle \phi \rangle \mid \phi \text{ is a satisfiable 3cnf-formula}\}$ (reduce from SAT)



- $CLIQUE = \{\langle G, k \rangle \mid G \text{ is an undirected graph with a } k\text{-clique}\}$ (reduce from $3SAT$)

- $HAMPATH = \{\langle G, s, t \rangle \mid G \text{ is a directed graph with a Hamiltonian path from } s \text{ to } t\}$

(reduce from $3SAT$)



NP-Complete problems, TODAY

- $SUBSET-SUM = \{\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t\}$.
 - (reduce from $3SAT$)
- $VERTEX-COVER = \{\langle G, k \rangle \mid G \text{ is an undirected graph that has a } k\text{-node vertex cover}\}$.
 - (reduce from $3SAT$)

Theorem: *SUBSET-SUM* is NP-complete

$SUBSET-SUM = \{ \langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t \}$

THEOREM 7.36
If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.

Strategy: Use Proof Parts (5):


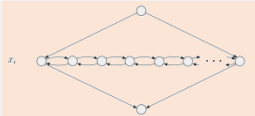
1. Show *SUBSET-SUM* is in **NP** (done in prev class)
2. Choose **NP-complete** problem to **reduce from: 3SAT**

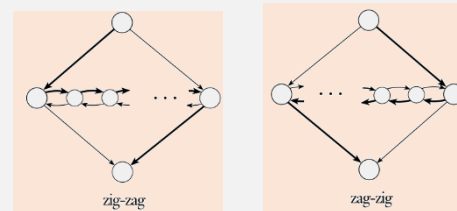


General Strategy: Reducing from 3SAT

NOTE: “gadgets” are not always graphs

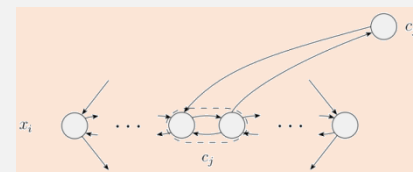
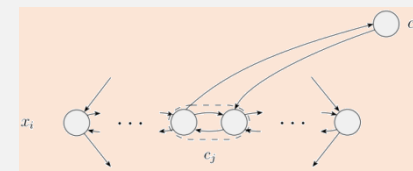
Create a **computable function** mapping formula to “gadgets”:

- **Clause** \rightarrow some “gadget”, e.g.,  c_j
- **Variable** \rightarrow another “gadget”, e.g., 
Gadget is typically used in two “opposite” ways:
 - ZIG when var is assigned **TRUE**, or
 - ZAG when var is assigned **FALSE**



Then connect “gadgets” according to clause literals:

- **Literal x_i in clause c_j** \rightarrow gadget x_i “detours” to c_j
- **Literal \bar{x}_i in clause c_j** \rightarrow gadget x_i “reverse detours” to c_j



Theorem: *SUBSET-SUM* is NP-complete

$SUBSET-SUM = \{\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t\}$

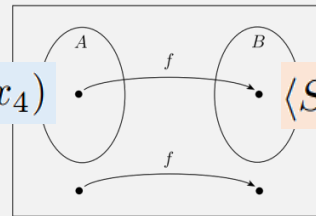
THEOREM 7.36
 If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.

Strategy: Use Proof Parts (5):

- ✓ 1. Show *SUBSET-SUM* is in **NP** (done in prev class)
- ✓ 2. Choose **NP**-complete problem to reduce from: *3SAT*
- ➡ 3. Create the computable function f :

$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$

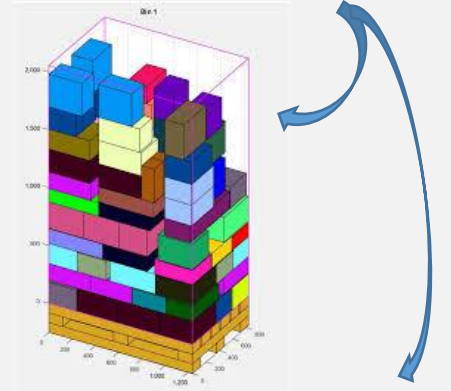
Coming up next!



$\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}$

4. Show it runs in poly time
5. Show Def 7.29 “iff” requirement:

ϕ is a satisfiable 3cnf-formula $\iff f(\langle \phi \rangle) = \langle S, t \rangle$ where some subset of S sums to t



50 kg??

5000 gold 25 KG	2500 gold 20 KG	10 gold 20 KG	2500 gold 12.5 KG	2500 gold 10 KG
200 gold 10 KG	3000 gold 7.5 KG	500 gold 4 KG	100 gold 1 KG	10 gold 1 KG

Computable Fn: 3cnf $\rightarrow \langle S, t \rangle$

E.g., $(x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee x_3 \vee \dots) \wedge \dots \wedge (\overline{x_3} \vee \dots \vee \dots)$

- Assume formula has:
 - l variables x_1, \dots, x_l
 - k clauses c_1, \dots, c_k
- Computable function f maps:
 - Variable $x_i \rightarrow$
 - Clause $c_j \rightarrow$
 - arranged in a table ...
- Each number has max $l+k$ digits:
 - Literal x_i in clause $c_j \rightarrow$
 - Literal $\overline{x_i}$ in clause $c_j \rightarrow$
- Sum is l 1s followed by k 3s

	1	2	3	4	...	l	c_1	c_2	...	c_k	
y_1	1	0	0	0	...	0	1	0	...	0	
z_1	1	0	0	0	...	0	0	0	...	0	
y_2		1	0	0	...	0	0	1	...	0	
z_2		1	0	0	...	0	1	0	...	0	
y_3			1	0	...	0	1	1	...	0	
z_3			1	0	...	0	0	0	...	1	
\vdots					\ddots	\vdots	\vdots		\vdots	\vdots	
y_l						1	0	0	...	0	
z_l						1	0	0	...	0	
g_1							1	0	...	0	
h_1							1	0	...	0	
g_2								1	...	0	
h_2								1	...	0	
\vdots									\ddots	\vdots	
g_k										1	
h_k										1	
The sum	t	1	1	1	1	...	1	3	3	...	3

y_i and z_i :
 i^{th} digit = 1

y_i : $l+j^{\text{th}}$ digit = 1
if c_j has x_i

z_i : $l+j^{\text{th}}$ digit = 1
if c_j has $\overline{x_i}$

g_j and h_j :
 $l+j^{\text{th}}$ digit = 1,
To help get
the right
sum

The sum

Theorem: *SUBSET-SUM* is NP-complete

$SUBSET-SUM = \{\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t\}$

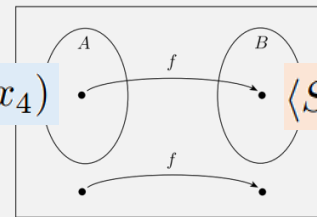
THEOREM 7.36

Strategy: Use If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.

Proof Parts (5):

- ✓ 1. Show *SUBSET-SUM* is in **NP** (done in prev class)
- ✓ 2. Choose **NP**-complete problem to reduce from: *3SAT*
- ✓ 3. Create the computable function f :

$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$



$\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}$

- ➔ 4. Show it runs in poly time
- 5. Show Def 7.29 iff requirement:

ϕ is a satisfiable 3cnf-formula $\iff f(\langle \phi \rangle) = \langle S, t \rangle$ where some subset of S sums to t

Polynomial Time?

E.g., $(x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee x_3 \vee \dots) \wedge \dots \wedge (\overline{x_3} \vee \dots \vee \dots)$ \rightarrow

- Assume formula has:
 - l variables x_1, \dots, x_l
 - k clauses c_1, \dots, c_k
- Table size: $(l + k) * (2l + 2k)$
 - Creating it requires constant number of passes over the table
 - Num variables $l =$ at most $3k$
- Total: $O(k^2)$

	1	2	3	4	...	l	c_1	c_2	...	c_k
y_1	1	0	0	0	...	0	1	0	...	0
z_1	1	0	0	0	...	0	0	0	...	0
y_2		1	0	0	...	0	0	1	...	0
z_2		1	0	0	...	0	1	0	...	0
y_3			1	0	...	0	1	1	...	0
z_3			1	0	...	0	0	0	...	1
\vdots					\ddots	\vdots	\vdots		\vdots	\vdots
y_l						1	0	0	...	0
z_l						1	0	0	...	0
g_1							1	0	...	0
h_1							1	0	...	0
g_2								1	...	0
h_2								1	...	0
\vdots									\ddots	\vdots
g_k										1
h_k										1
t	1	1	1	1	...	1	3	3	...	3

Theorem: *SUBSET-SUM* is NP-complete

$SUBSET-SUM = \{\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t\}$

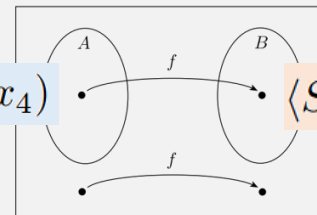
THEOREM 7.36

Strategy: Use If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.

Proof Parts (5):

- ✓ 1. Show *SUBSET-SUM* is in **NP** (done in prev class)
- ✓ 2. Choose **NP**-complete problem to reduce from: *3SAT*
- ✓ 3. Create the computable function f :

$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$



- ✓ 4. Show it runs in poly time
- ➔ 5. Show Def 7.29 iff requirement:

ϕ is a satisfiable 3cnf-formula $\iff f(\langle \phi \rangle) = \langle S, t \rangle$ where some subset of S sums to t

ϕ is a satisfiable 3cnf-formula $\iff f(\langle\phi\rangle) = \langle S, t \rangle$ where some subset of S sums to t

Each column:
 - At least one 1
 - At most 3 1s

=> If formula is satisfiable ...

- Sum $t = l$ 1s followed by k 3s
- Choose for the subset ...
 - y_i if $x_i = \text{TRUE}$
 - z_i if $x_i = \text{FALSE}$
 - and some of g_i and h_i to make the sum t
- ... **Then** this subset of S must sum to t bc:
 - Left digits:
 - only one of y_i or z_i is in S
 - Right digits:
 - Top right: Each column sums to 1, 2, or 3
 - Because each clause has 3 literals
 - Bottom right:
 - Can always use g_i and/or h_i to make column sum to 3

S only includes one

	1	2	3	4	...	l	c_1	c_2	...	c_k
y_1	1	0	0	0	...	0	1	0	...	0
z_1	1	0	0	0	...	0	0	0	...	0
y_2		1	0	0	...	0	0	1	...	0
z_2		1	0	0	...	0	1	0	...	0
y_3			1	0	...	0	1	1	...	0
z_3			1	0	...	0	0	0	...	1
...				
y_l						1	0	0	...	0
z_l						1	0	0	...	0
g_1							1	0	...	0
h_1							1	0	...	0
g_2								1	...	0
h_2								1	...	0
...								
g_k										1
h_k										1
t	1	1	1	1	...	1	3	3	...	3

g_j and h_j : help get the correct sum

So each column sum (for left digits) is 1

ϕ is a satisfiable 3cnf-formula $\iff f(\langle\phi\rangle) = \langle S, t \rangle$ where some subset of S sums to t

Subset must have some number with 1 in each right column

\Leftarrow If a subset of S sums to t ...

- It can only include either y_i or z_i
 - Because each left digit column must sum to 1
 - And no carrying is possible
- Also, since each right digit column must sum to 3:
 - And only 2 can come from g_i and h_i
 - Then for every right column, some y_i or z_i in the subset has a 1 in that column
- ... **Then** a satisfying assignment is:
 - $x_i = \text{TRUE}$ if y_i in the subset
 - $x_i = \text{FALSE}$ if z_i in the subset
- This is satisfying because:
 - Table was constructed so 1 in column c_j for y_i or z_i means that variable x_i satisfies clause c_j
 - We already determined, for every right column, some number in the subset has a 1 in the column
 - So all clauses are satisfied

S only includes y_i or z_i

	1	2	3	4	...	l	c_1	c_2	...	c_k	
y_1	1	0	0	0	...	0	1	0	...	0	
z_1	1	0	0	0	...	0	0	0	...	0	
y_2		1	0	0	...	0	0	1	...	0	
z_2		1	0	0	...	0	1	0	...	0	
y_3			1	0	...	0	1	1	...	0	
z_3			1	0	...	0	0	0	...	1	
\vdots					\ddots	\vdots	\vdots		\vdots	\vdots	
g_1							1	0	0	...	0
h_1							1	0	0	...	0
g_2								1	0	...	0
h_2									1	...	0
\vdots										\ddots	\vdots
g_k											1
h_k											1
t	1	1	1	1	...	1	3	3	...	3	

In each right column, g_i and h_i can account for at most 2

Because each column sum (for left digits) is 1

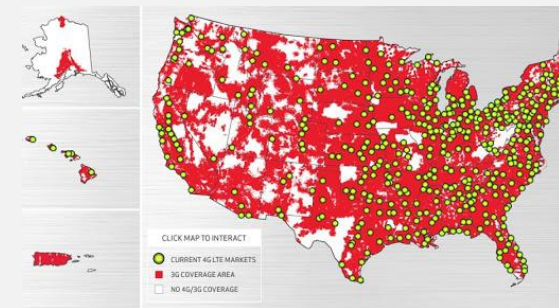
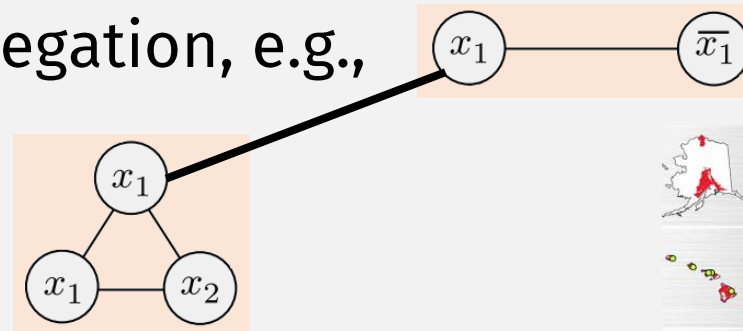
NP-Complete problems, TODAY

- ✓ • $SUBSET-SUM = \{\langle S, t \rangle \mid S = \{x_1, \dots, x_k\}, \text{ and for some } \{y_1, \dots, y_l\} \subseteq \{x_1, \dots, x_k\}, \text{ we have } \sum y_i = t\}.$
 - (reduce from $3SAT$)
- $VERTEX-COVER = \{\langle G, k \rangle \mid G \text{ is an undirected graph that has a } k\text{-node vertex cover}\}.$
 - (reduce from $3SAT$)

Theorem: *VERTEX-COVER* is NP-complete.

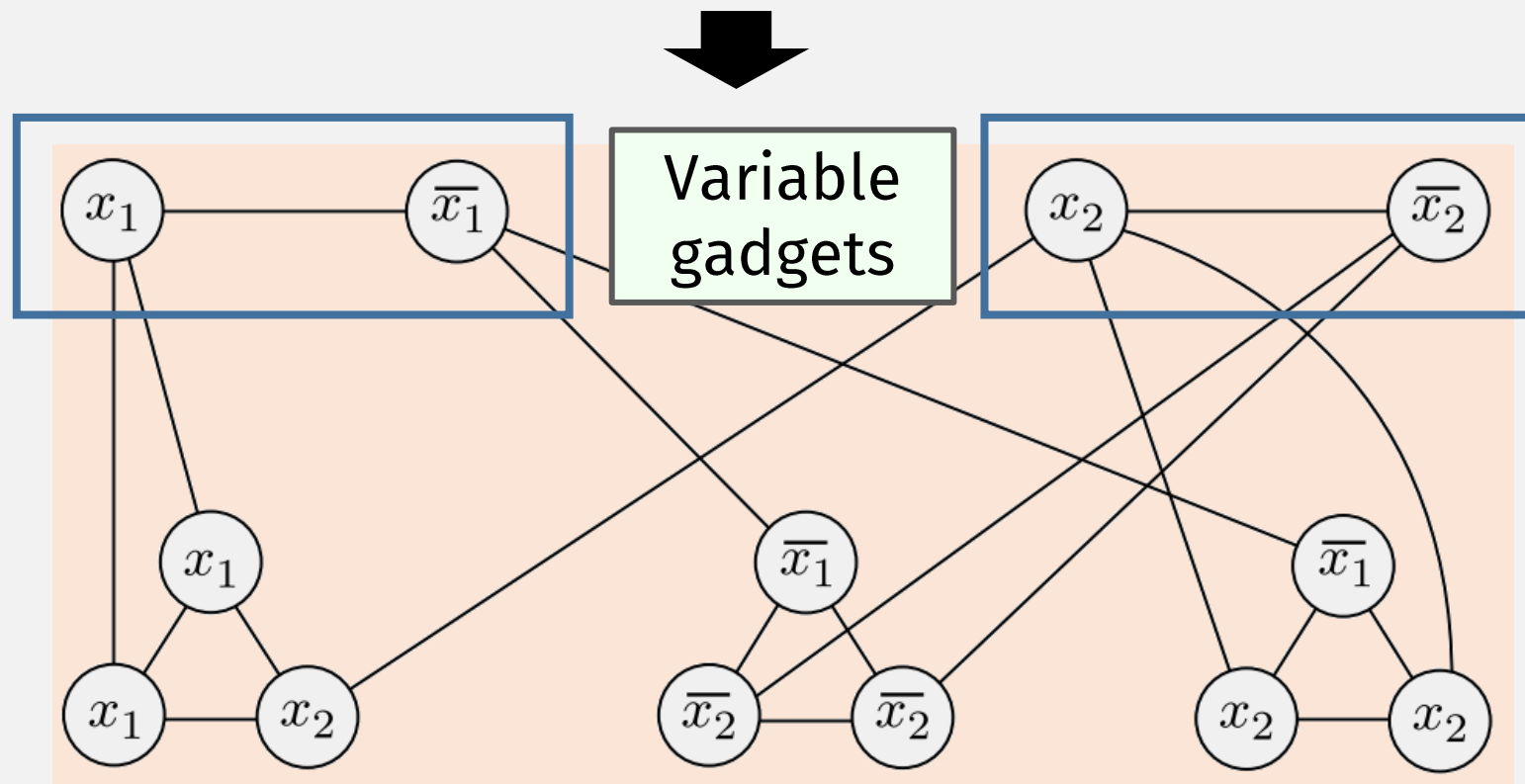
$VERTEX-COVER = \{ \langle G, k \rangle \mid G \text{ is an undirected graph that has a } k\text{-node vertex cover} \}$

- A vertex cover of a graph is ...
 - ... a subset of its nodes where every edge touches one of those nodes
- Proof Sketch: Reduce *3SAT* to *VERTEX-COVER*
- The reduction maps:
- **Variable $x_i \rightarrow 2$ connected nodes**
 - corresponding to the var and its negation, e.g.,
- **Clause $\rightarrow 3$ connected nodes**
 - corresponding to its literals, e.g.,
- Additionally,
 - connect var and clause gadgets by ...
 - ... connecting nodes that correspond to the same literal



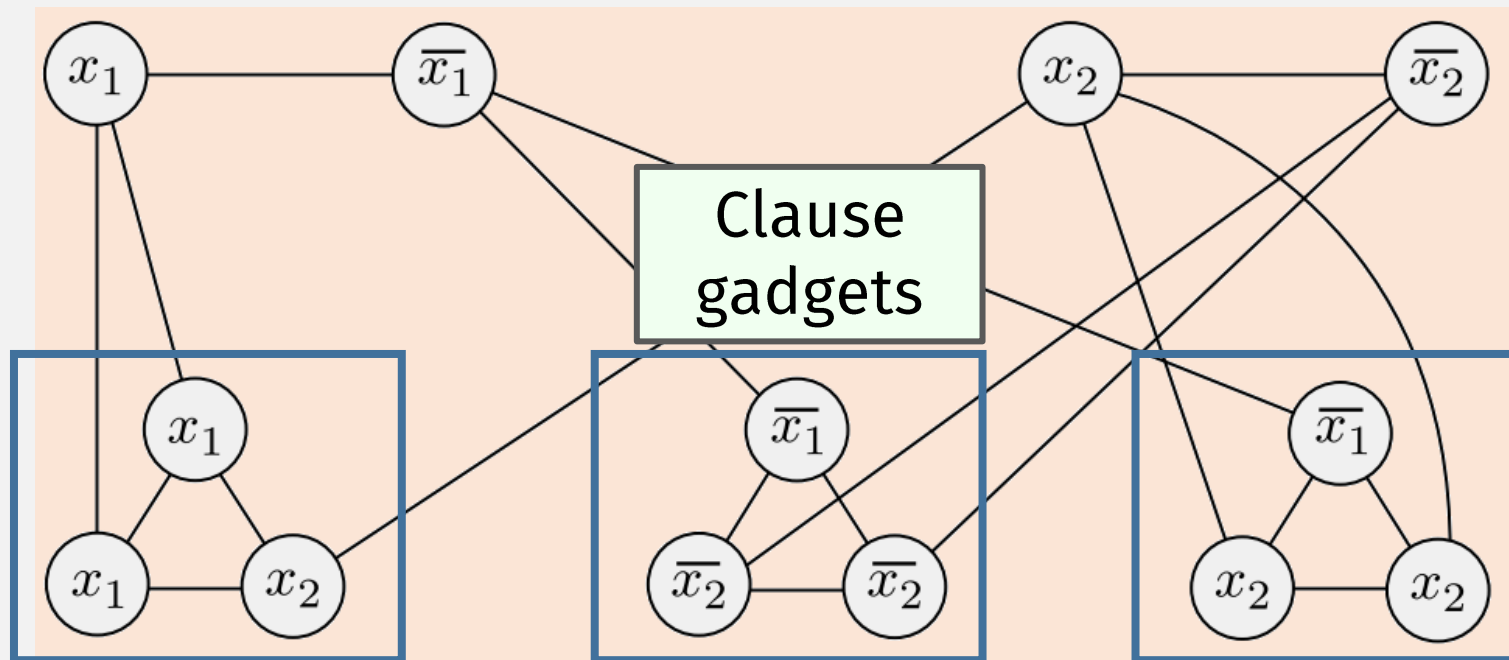
VERTEX-COVER example

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



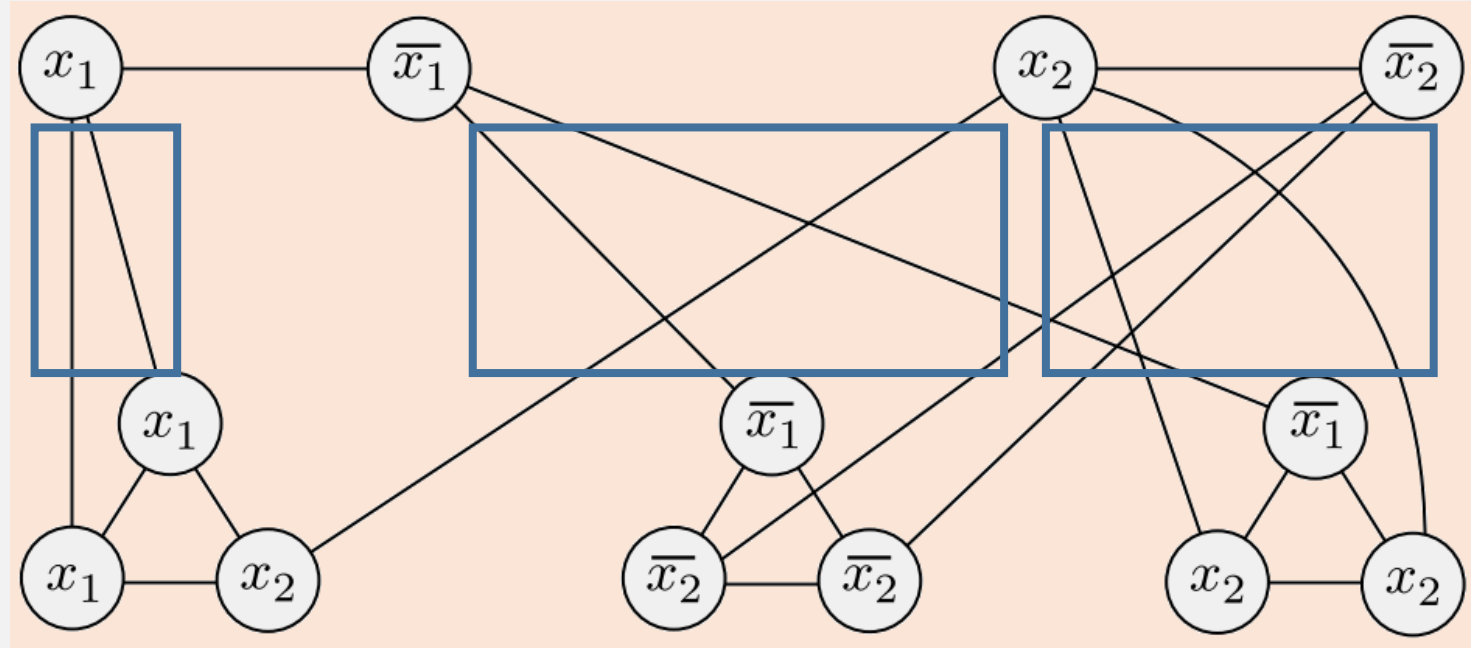
VERTEX-COVER example

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



VERTEX-COVER example

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

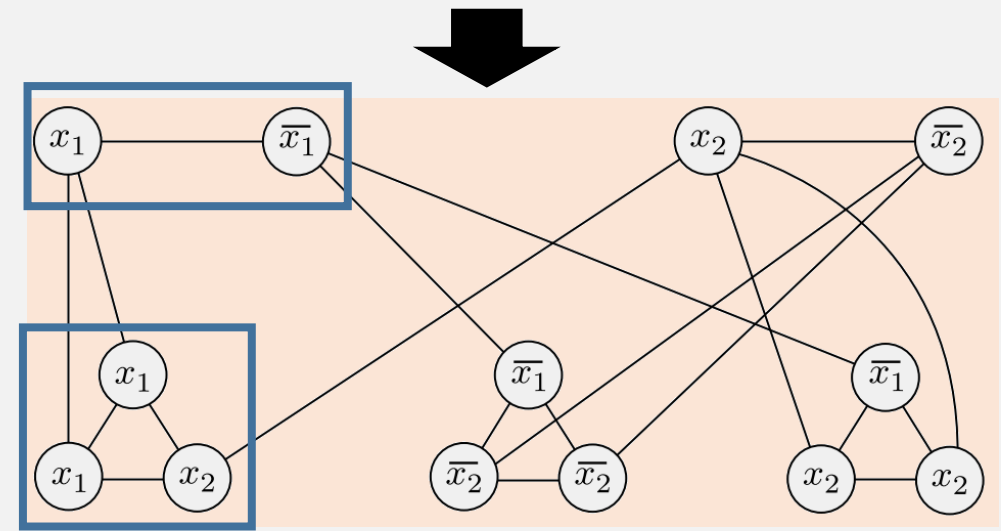


Extra edges connecting variable and clause gadgets together

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

VERTEX-COVER example

- If formula has ...
 - $m = \#$ variables
 - $l = \#$ clauses
- Then graph has ...
 - $\#$ nodes = $2m + 3l$



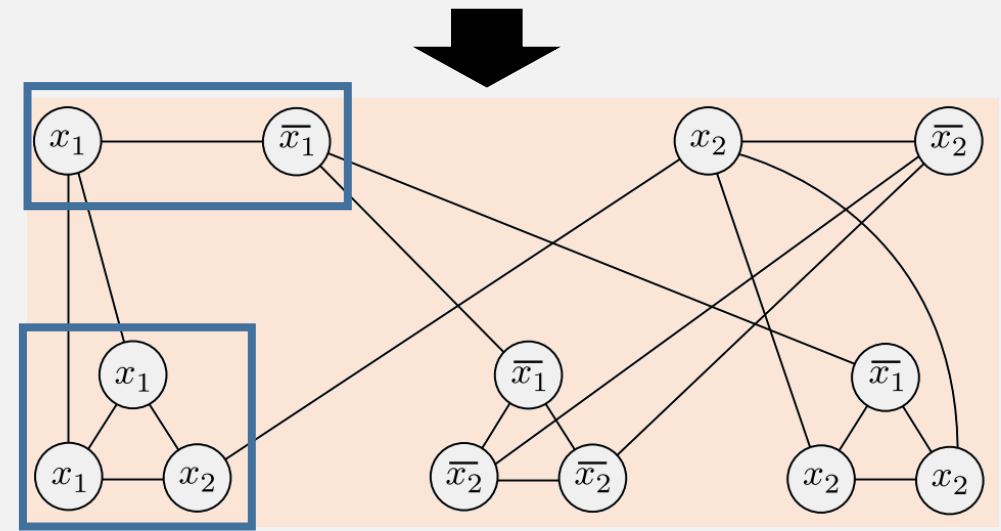
\Rightarrow If satisfying assignment, then there is a k -cover, where $k = m + 2l$

- Nodes in the cover:
 - In each of m var gadgets, choose 1 node corresponding to TRUE literal
 - For each of l clause gadgets, ignore 1 TRUE literal and choose other 2
 - Since there is satisfying assignment, each clause has a TRUE literal
 - Total = $m + 2l$

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

VERTEX-COVER example

- If formula has ...
 - $m = \#$ variables
 - $l = \#$ clauses
- Then graph has ...
 - $\#$ nodes = $2m + 3l$



\Leftarrow If there is a k -cover, then there is a satisfying assignment

- The k -cover must include ...
 - 1 node from each of “var” gadgets
 - 2 nodes from each “clause” gadget
- So the satisfying assignment is:
 - Assign $x_i = \text{TRUE}$ if node x_i from x_i var gadget is in the cover set
 - Else $x_i = \text{FALSE}$

NO Quiz 5/10!