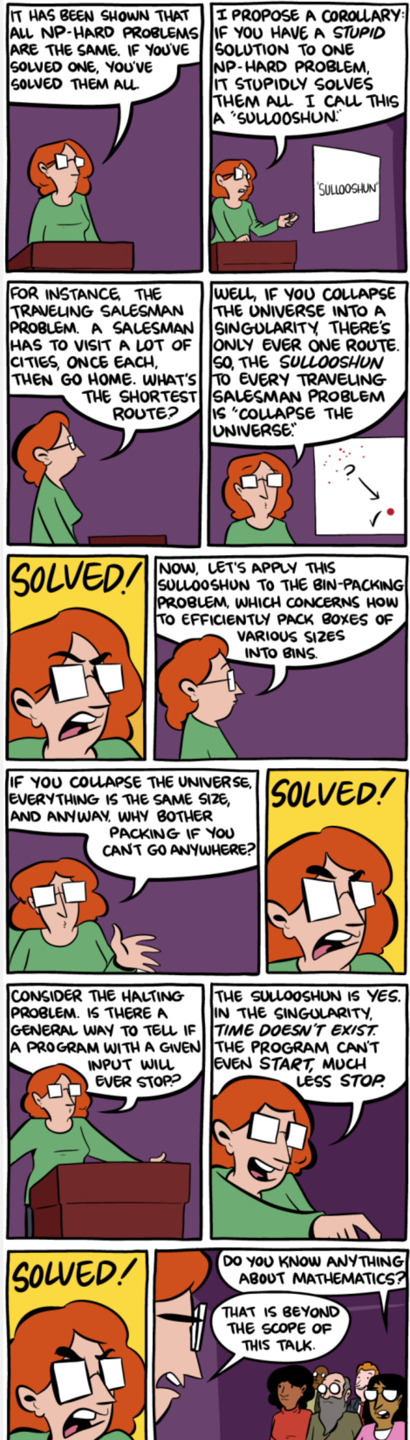
More NP-Complete Problems

Wednesday, May 4, 2022



Announcements

- HW 11 in
  - Due Tues 5/3 11:59pm EST
- HW 12 out tomorrow
  - Due Wed 5/11 11:59pm EST
  - Last HW!
- 3 lectures left!
- Course evals next week

Last Time: **NP**-Completeness

DEFINITION

A language B is ***NP-complete*** if it satisfies two conditions:

1. B is in NP, and
2. every A in NP is polynomial time reducible to B.

Must prove for <u>all</u> langs, not just a single language

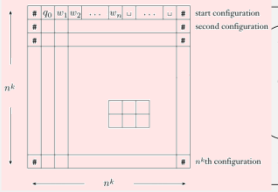
THEOREM

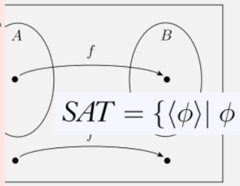
SAT is NP-complete.

It's difficult to prove <u>the first</u> **NP**-complete problem!

	$\langle M_1\rangle$	$\langle M_2\rangle$	$\langle M_3\rangle$	$\langle M_4\rangle$	$\cdots$	$\langle D\rangle$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*
M_3	*reject*	*reject*	*reject*	*reject*		*reject*
M_4	*accept*	*accept*	*reject*	*reject*		*accept*
$\vdots$		$\vdots$			$\ddots$	
D	*reject*	*reject*	*accept*	*accept*		?

(Just like finding <u>the first</u> **undecidable** problem was hard!)





$SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$

But each **NP**-complete problem we prove makes it **easier to prove the next one!**

THEOREM

known

unknown

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

If you're not Stephen Cook or Leonid Levin, **use this theorem to prove a language is NP-complete**

THEOREM

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

3 steps to prove a language C is **NP**-complete:

1. Show C is in **NP**
2. Choose B, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from B to C

To show poly time mapping reducibility:
1. create **computable fn**,
2. show that it **runs in poly time**,
3. then show **forward direction** of mapping red.,
4. and **reverse direction**
   (or **contrapositive** of **reverse direction**)

THEOREM

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

3 steps to prove a language C is **NP**-complete:

1. Show C is in **NP**
2. Choose B, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from B to C

Example:

Let $C = 3SAT$, to prove $3SAT$ is **NP**-Complete:

1. Show $3SAT$ is in **NP**

THEOREM

Last Time: If B is NP-complete and $B \leq_{\mathrm{P}} C$ for C in NP, then C is NP-complete.

3 steps to prove a language *C* is **NP**-complete:

1. Show *C* is in **NP**
2. Choose *B*, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from *B* to *C*

To show poly time mapping reducibility:
1. create **computable fn**,
2. show that it **runs in poly time**,
3. then show **forward direction** of mapping red.,
4. and **reverse direction** (or **contrapositive** of **reverse direction**)

Example:

Let *C* = *3SAT*, to prove *3SAT* is **NP**-Complete:

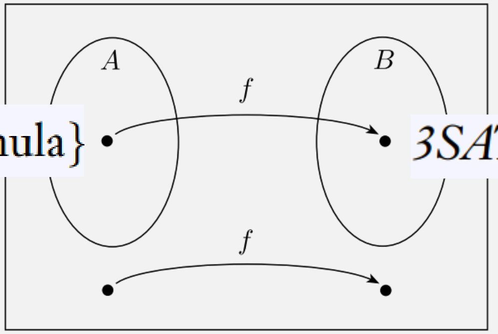
☑ 1. Show *3SAT* is in **NP**

☑ 2. Choose *B*, the **NP**-complete problem to reduce from: *SAT*

3. Show a poly time mapping reduction from *SAT* to *3SAT*

Flashback: *SAT* is Poly Time Reducible to *3SAT*

$SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$



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

Need: poly time computable fn converting a Boolean formula ϕ to 3CNF:

Remaining step: show iff relation holds ...

1. Convert ϕ to CNF (an AND of OR clauses)
   a) Use DeMorgan's Law to push negations onto literals

   $\neg(P \vee Q) \Longleftrightarrow (\neg P) \wedge (\neg Q)$ $\quad$ $\neg(P \wedge Q) \Longleftrightarrow (\neg P) \vee (\neg Q)$ $\quad$ $O(n)$

   b) Distribute ORs to get ANDs outside of parens

   $(P \vee (Q \wedge R)) \Leftrightarrow ((P \vee Q) \wedge (P \vee R))$ $\quad$ $O(n)$

... easy for formula conversion: each step is already a known "law"

2. Convert to 3CNF by adding new variables

   $(a_1 \vee a_2 \vee a_3 \vee a_4) \Leftrightarrow (a_1 \vee a_2 \vee z) \wedge (\overline{z} \vee a_3 \vee a_4)$ $\quad$ $O(n)$

THEOREM

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

<u>**3 steps**</u> to prove a language is **NP**-complete:

1. Show *C* is in **NP**
2. Choose *B*, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from *B* to *C*

<u>**Theorem**</u>: *3SAT* is NP-complete

Let *C* = *3SAT*, to prove *3SAT* is **NP**-Complete:

☑ 1. Show *3SAT* is in **NP**

☑ 2. Choose *B*, the **NP**-complete problem to reduce from: *SAT*

☑ 3. Show a poly time mapping reduction from *SAT* to *3SAT* ■

Now have <u>2 known</u> **NP**-Complete languages to use:

- *SAT*
- *3SAT*

THEOREM

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

<u>**3 steps**</u> to prove a language is **NP**-complete:

1. Show C is in **NP**
2. Choose B, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from B to C

<u>**Theorem**</u>: $CLIQUE$ is NP-complete

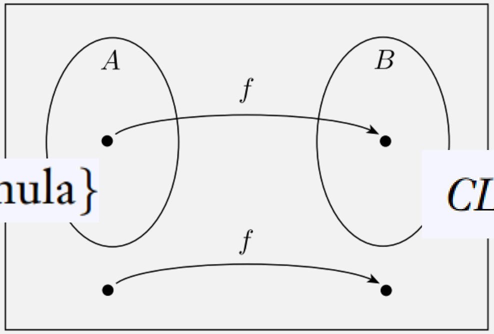
Let C = ~~*3SAT*~~ ***CLIQUE***, to prove ~~*3SAT*~~ ***CLIQUE*** is **NP**-Complete:

?1. Show ~~*3SAT*~~ ***CLIQUE*** is in **NP**

?2. Choose B, the **NP**-complete problem to reduce from: ~~*SAT*~~ ***3SAT***

?3. Show a poly time mapping reduction from B to C

Flashback: *3SAT* is polynomial time reducible to *CLIQUE*.



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

$CLIQUE = \{\langle G, k\rangle \mid G \text{ is an undirected graph with a } k\text{-clique}\}$

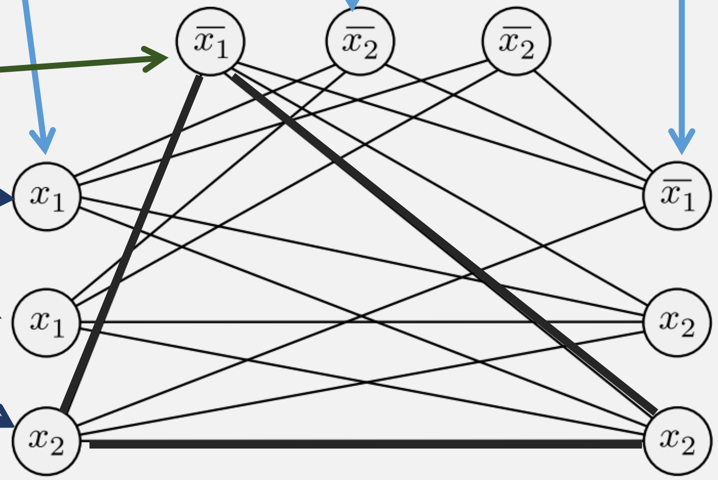
Need: poly time computable fn converting a 3cnf-formula …

Example:

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

- … to a graph containing a clique:
  - Each clause maps to a group of 3 nodes
  - Connect all nodes except:
    - Contradictory nodes
    - Nodes in the same group

Don't forget iff



Runs in **poly time:**
- \# literals = # nodes $O(n)$
- \# edges poly in # nodes $O(n^2)$

⇒ If $\phi \in 3SAT$

- Then each clause has a TRUE literal
  - Those are nodes in the clique!
  - E.g., $x_1 = 0$, $x_2 = 1$

⇐ If $\phi \notin 3SAT$

- For any assignment, some clause must have a contradiction with another clause
- Then in the graph, some clause's group of nodes won't be connected to another group, preventing the clique

THEOREM

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

<u>**3 steps**</u> to prove a language is **NP**-complete:

1. Show *C* is in **NP**
2. Choose *B*, the **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from *B* to *C*

<u>**Theorem**</u>: *CLIQUE* is NP-complete

Let *C* = ~~*3SAT*~~ ***CLIQUE***, to prove ~~*3SAT*~~ ***CLIQUE*** is **NP**-Complete:

☑ 1. Show ~~*3SAT*~~ ***CLIQUE*** is in **NP**

☑ 2. Choose *B*, the **NP**-complete problem to reduce from: ~~SAT~~ ***3SAT***

☑ 3. Show a poly time mapping reduction from *B* to *C*

■

Now have <u>3 known</u> **NP**-Complete languages to use:
- *SAT*
- *3SAT*
- *CLIQUE*

NP-Complete problems, so far

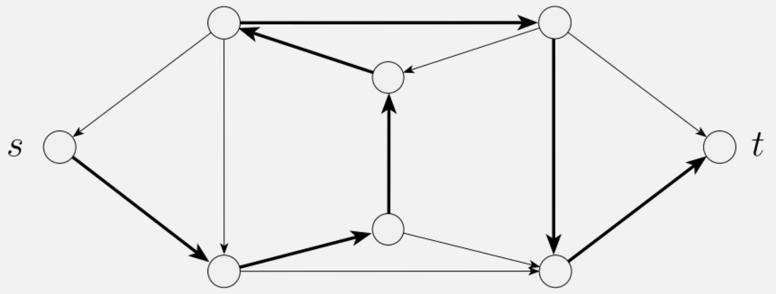
- $SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$ (Cook-Levin Theorem)
- $3SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable 3cnf-formula}\}$ (reduced *SAT* to *3SAT*)
- $CLIQUE = \{\langle G, k\rangle \mid G \text{ is an undirected graph with a } k\text{-clique}\}$ (reduced *3SAT* to *CLIQUE*)

We now have 3 options to choose from when proving the next **NP**-complete problem

Flashback: The *HAMPATH* Problem

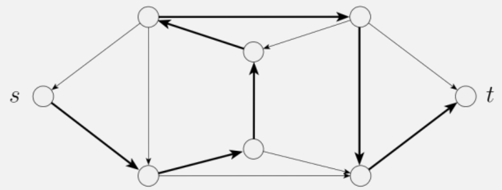
$$HAMPATH = \{\langle G, s, t\rangle \mid G \text{ is a directed graph with a Hamiltonian path from } s \text{ to } t\}$$

- A Hamiltonian path goes through every node in the graph



- The **Search** problem:
  - Exponential time (brute force) algorithm:
    - Check all possible paths of length $\boldsymbol{n}$
    - See if any connects s and t: $O(\boldsymbol{n!}) = O(2^n)$
  - Polynomial time algorithm:
    - Unknown!!!
- The **Verification** problem:
  - Still $O(\boldsymbol{n}^2)$, just like *PATH*!
- So *HAMPATH* is in **NP** but not known to be in **P**

Theorem: *HAMPATH* is NP-complete



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

THEOREM

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

3 steps to prove a language is **NP**-complete:

1. Show C is in **NP**
2. Choose B, the known **NP**-complete problem to reduce from
3. Show a poly time mapping reduction from B to C

Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑ 1. Show *HAMPATH* is in **NP** (left as exercise)

☑ 2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

3. Show a poly time mapping reduction from *B* to *HAMPATH*

Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑ 1. Show *HAMPATH* is in **NP** (left as exercise)

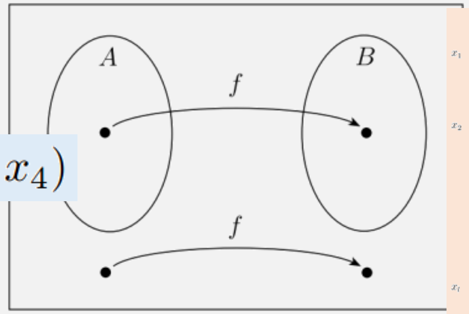
☑ 2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

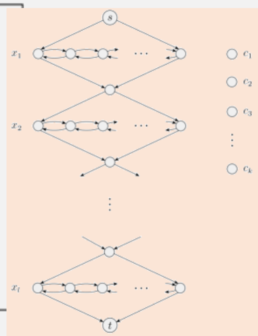
? 3. Show a poly time mapping reduction from *3SAT* to *HAMPATH*

To show poly time mapping reducibility:
1. create **computable fn**,
2. show that it **runs in poly time**,
3. then show **forward direction** of mapping red.,
4. and **reverse direction** (or **contrapositive** of **reverse direction**)

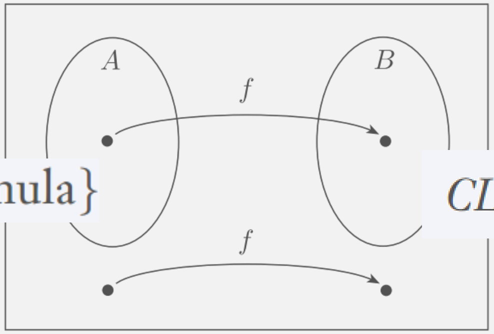
???

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





Flashback: $3SAT$ is polynomial time reducible to $CLIQUE$.



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

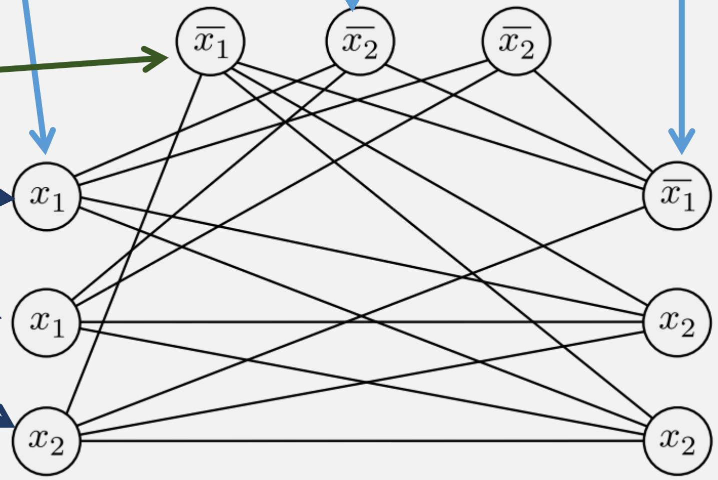
$CLIQUE = \{\langle G, k\rangle \mid G \text{ is an undirected graph with a } k\text{-clique}\}$

Need: poly time computable fn converting a 3cnf-formula …

Example:

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

- … to a graph containing a clique:
  - Each **clause** maps to a group of 3 **nodes**
  - **Connect** all **nodes** except:
    - Contradictory nodes
    - Nodes in the same group

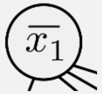


Do conversion piece by piece …

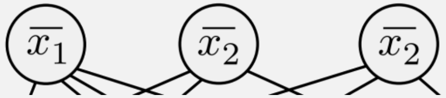
General Strategy: Reducing from *3SAT*

Create a **computable function** mapping formula to "**gadgets**":

- Variable → "gadget", e.g.,



- Clause → "gadget", e.g.,

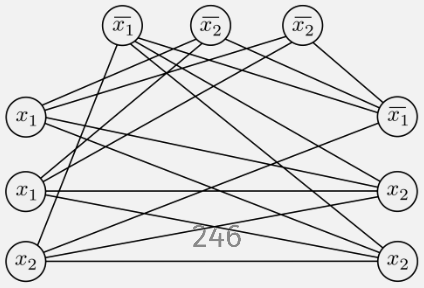


  Gadget is typically "used" in two "opposite" ways:
  1. "something" when var is assigned TRUE, or
  2. "something else" when var is assigned FALSE

NOTE: "gadgets" are not always graphs; depends on the problem

Then **connect variable** and **clause "gadgets" together:**

- Literal x_i in clause c_j → gadget x_i "connects to" gadget c_j
- Literal $\overline{x_i}$ in clause c_j → gadget x_i "connects to" gadget c_j
- E.g., **connect each node to node not in clause**



Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑1. Show *HAMPATH* is in **NP** (in HW9)

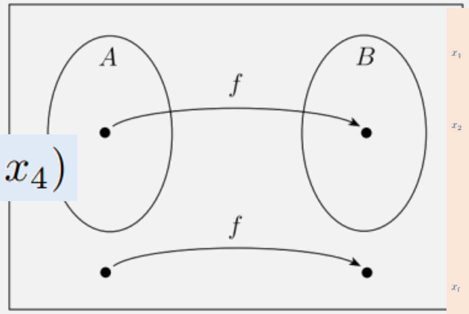
☑2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

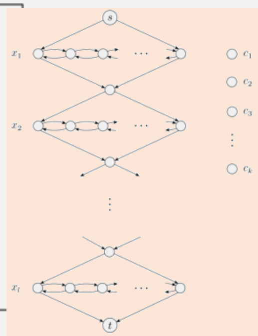
? 3. Show a poly time mapping reduction from *3SAT* to *HAMPATH*

To show poly time mapping reducibility:
1. create **computable fn**,
2. show that it **runs in poly time**,
3. then show **forward direction** of mapping red.,
4. and **reverse direction** (or **contrapositive** of **reverse direction**)

???

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





Computable Fn: Formula (blue) → Graph (orange)

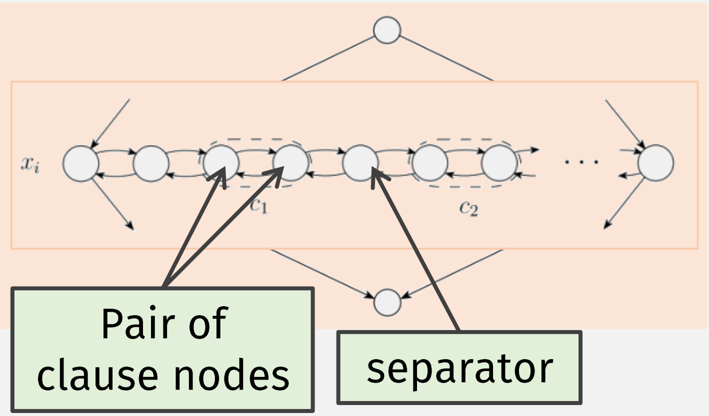
variable clause

Example input: $\phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \cdots \wedge (a_k \vee b_k \vee c_k)$

k = # clauses

(extra)

- Clause → (extra) single nodes, Total = k
- Variable → diamond-shaped graph "gadget"
  - Clause → 2 "connector" nodes + separator
  - Total = $3k$+1 "connector" nodes per "gadget"



Computable Fn: Formula (blue) → Graph (orange)

Example input: $\phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \cdots \wedge (a_k \vee b_k \vee c_k)$

k = # clauses

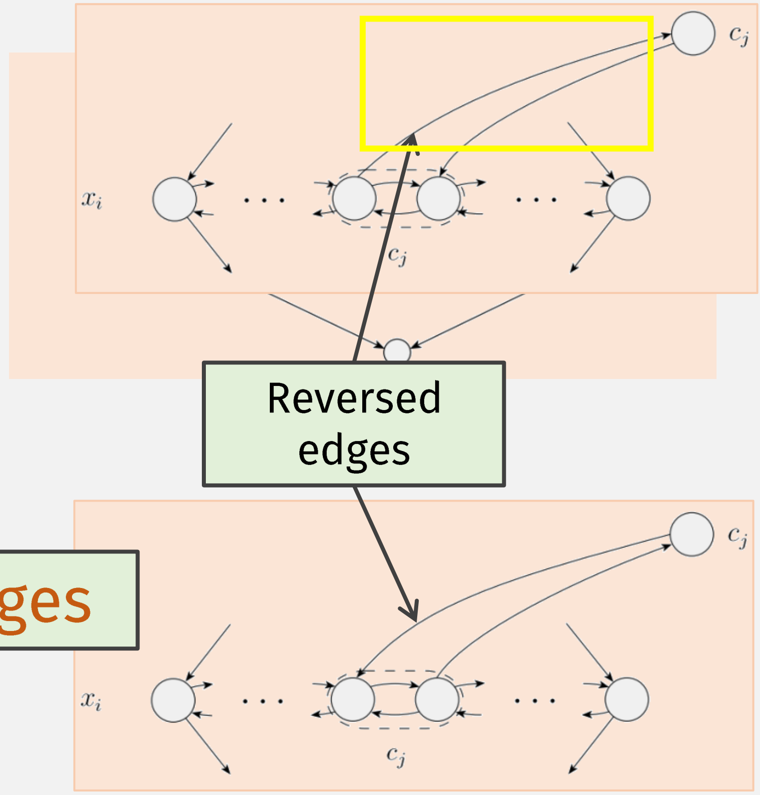
- Clause → (extra) single nodes, Total = k
- Variable → diamond-shaped graph "gadget"
  - Clause → 2 "connector" nodes + separator
  - Total = $3k$+1 "connector" nodes per "gadget"

Literal = variable or negated variable

- Lit x_i in clause c_j → c_j node edges in gadget x_i

Each extra c_j node has 6 edges

- Lit $\overline{x_i}$ in clause c_j → c_j edges in gadget x_i (rev)



Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑ 1. Show *HAMPATH* is in **NP**

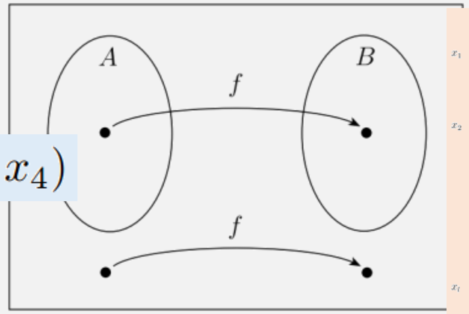
☑ 2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

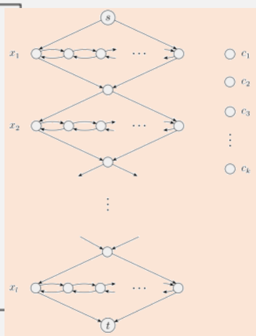
? 3. Show a poly time mapping reduction from *3SAT* to *HAMPATH*

To show poly time mapping reducibility:

☑ 1. create **computable fn**,
2. show that it **runs in poly time**,
3. then show **forward direction** of mapping red.,
4. and **reverse direction** (or **contrapositive** of **reverse direction**)

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





Polynomial Time?

TOTAL:
O($\boldsymbol{k}^2$)

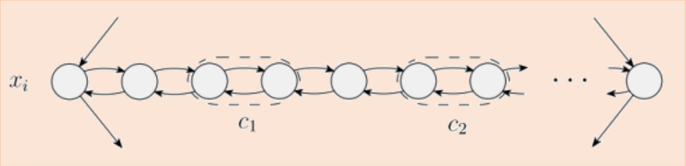
Example input: $\phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \cdots \wedge (a_k \vee b_k \vee c_k)$

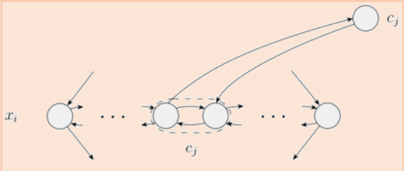
k = # clauses = at most $\boldsymbol{3k}$ variables

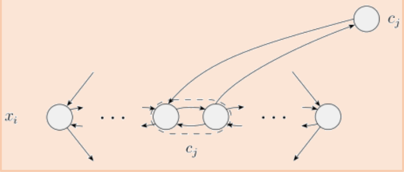
- Clause → (extra) single nodes 

O($\boldsymbol{k}$)

- Variable → diamond-shaped graph "gadget" O($\boldsymbol{k}^2$)
    - Clause → 2 "connector" nodes + separator
    - Total = $3k$+1 "connector" nodes per "gadget"



- Lit x_i in clause c_j → c_j node edges in gadget x_i  O($\boldsymbol{k}$)

- Lit $\overline{x_i}$ in clause c_j → c_j edges in gadget x_i (rev)  O($\boldsymbol{k}$)

Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑ 1. Show *HAMPATH* is in **NP**

☑ 2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

? 3. Show a poly time mapping reduction from *3SAT* to *HAMPATH*

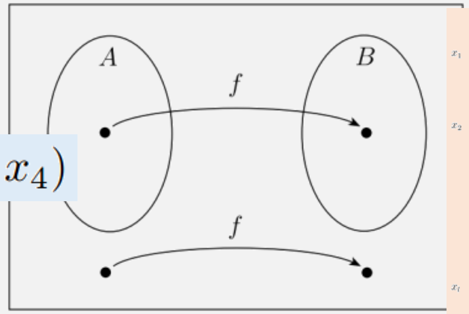
To show poly time mapping reducibility:

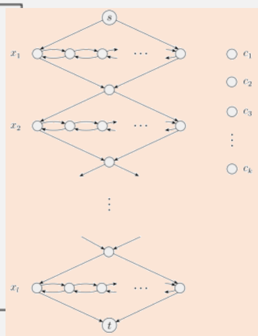
☑ 1. create **computable fn**,

☑ 2. show that it **runs in poly time**,

3. then show **forward direction** of mapping red.,
4. and **reverse direction** (or **contrapositive** of **reverse direction**)

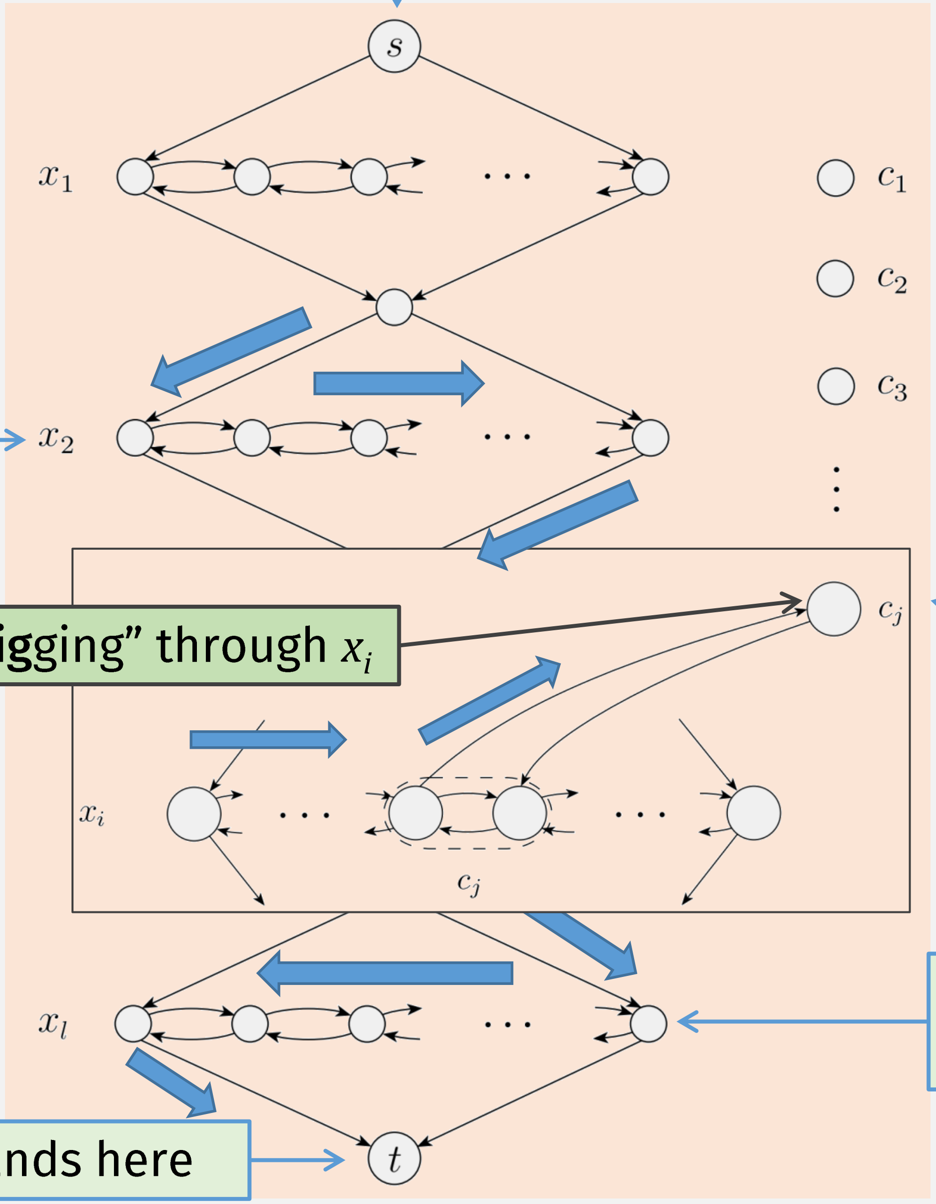
$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$





A **Ham. Path** (must touch all nodes) **through this graph:**

1. Starts here



3a. and either **“zig-zags”** …

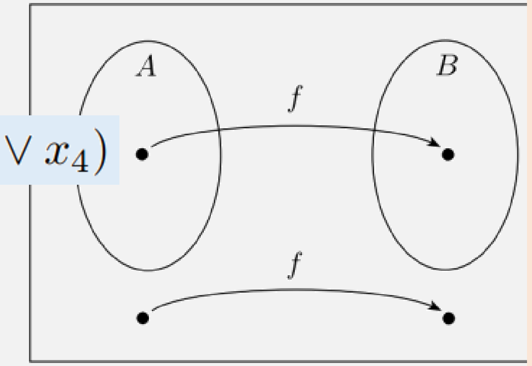
Can only go to this c_j if “**zig**ging” through x_i

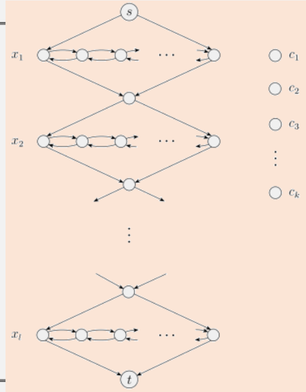
4. and **must “detour” to these clause gadgets** (at least once, has 3 chances)

3b. or “**zag-zigs**” through each variable gadget

2. Ends here

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



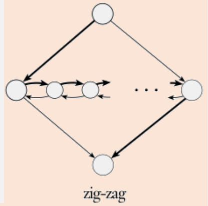


Want: **Satisfiable 3cnf formula** ⇔ **graph with Hamiltonian path**

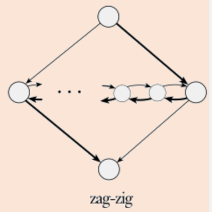
⇒ If **there is satisfying assignment,** then **Hamiltonian path exists**

These hit all nodes except extra c_js

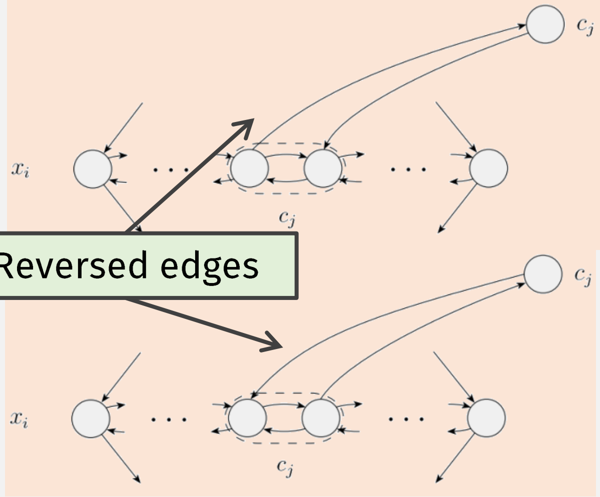
- x_i = TRUE → Hampath "zig-zags" gadget x_i



- x_i = FALSE → Hampath "zag-zigs" gadget x_i



- Lit x_i makes clause c_j TRUE → "detour" to c_j in gadget x_i
- Lit $\overline{x_i}$ makes clause c_j TRUE → "detour" to c_j in gadget x_i

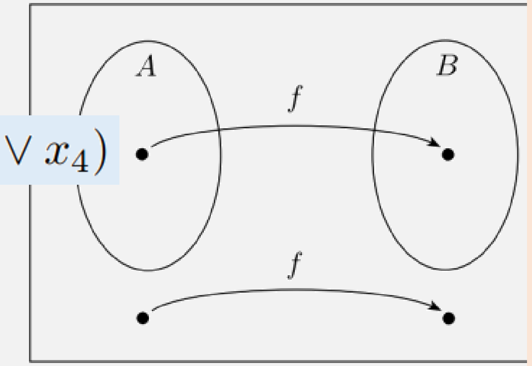


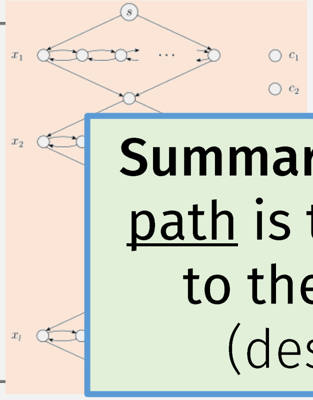
Now path goes through every node

Every clause must be TRUE so path hits all c_j nodes

- And edge directions align with TRUE/FALSE assignments

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

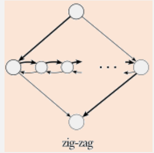




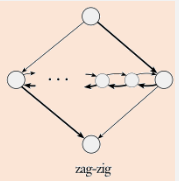
Summary: the only possible Ham. path is the one that corresponds to the satisfying assignment (described on prev slide)

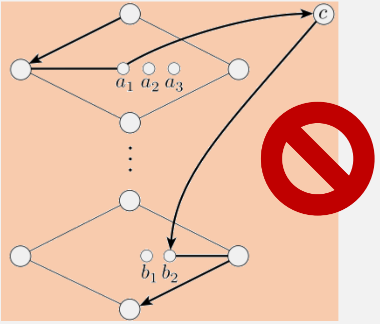
Want: **Satisfiable 3cnf formula ⇔ graph with Hamiltonian path**

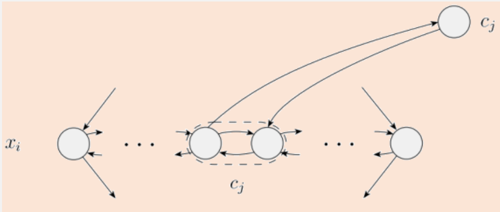
⇐ if **output has Ham. path,** then **input had Satisfying assignment**

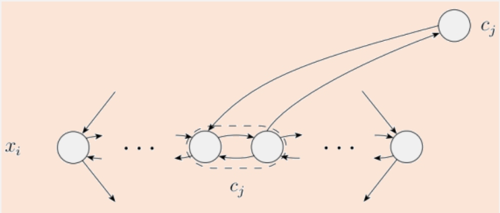


- A Hamiltonian path must choose to either zig-zag or zag-zig gadgets
- Ham path can only hit "detour" c_j nodes by coming right back
- Otherwise, it will miss some nodes





- 

gadget x_i "detours" from left to right → x_i = TRUE
- 

gadget x_i "detours" from right to left → x_i = FALSE

Theorem: *HAMPATH* is NP-complete

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

To prove *HAMPATH* is **NP**-complete:

☑1. Show *HAMPATH* is in **NP**

☑2. Choose *B*, the **NP**-complete problem to reduce from *3SAT*

☑3. Show a poly time mapping reduction from *3SAT* to *HAMPATH*

To show poly time mapping reducibility:

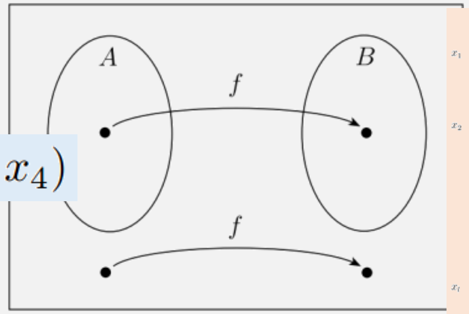
☑ 1. create **computable fn**,

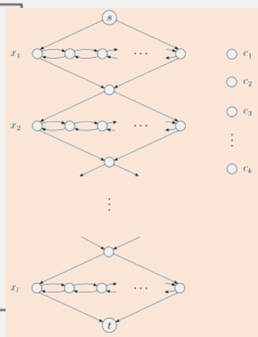
☑ 2. show that it **runs in poly time**,

☑ 3. then show **forward direction** of mapping red.,

☑ 4. and **reverse direction**
(or **contrapositive** of **reverse direction**)

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





Theorem: *UHAMPATH* is NP-complete

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

To prove *UHAMPATH* is **NP**-complete:

☑ 1. Show *UHAMPATH* is in **NP**

→ 2. Choose the **NP**-complete problem to reduce from *HAMPATH*

3. Show a poly time mapping reduction from *???* to *UHAMPATH*

Theorem: *UHAMPATH* is NP-complete

un

$UHAMPATH = \{\langle G, s, t\rangle |\ G$ is a directed graph with a Hamiltonian path from s to $t\}$

To prove *UHAMPATH* is **NP**-complete:

- ☑ 1. Show *UHAMPATH* is in **NP**
- ☑ 2. Choose the **NP**-complete problem to reduce from *HAMPATH*
- → 3. Show a poly time mapping reduction from *HAMPATH* to *UHAMPATH*

<u>Theorem</u>: *UHAMPATH* is NP-complete

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

Need: **Computable function from *HAMPATH* to *UHAMPATH***

Naïve Idea: **Make all directed edges undirected?**

- But **we would create some paths that didn't exist before**

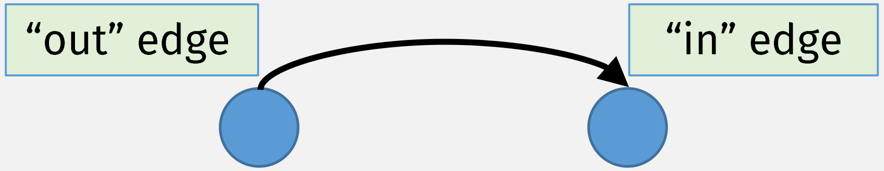
- **Doesn't work!**

Theorem: *UHAMPATH* is NP-complete

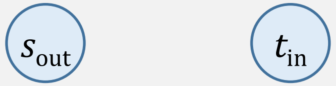
$UHAMPATH = \{\langle G, s, t\rangle \mid G$ is a undirected graph with a Hamiltonian path from s to $t\}$

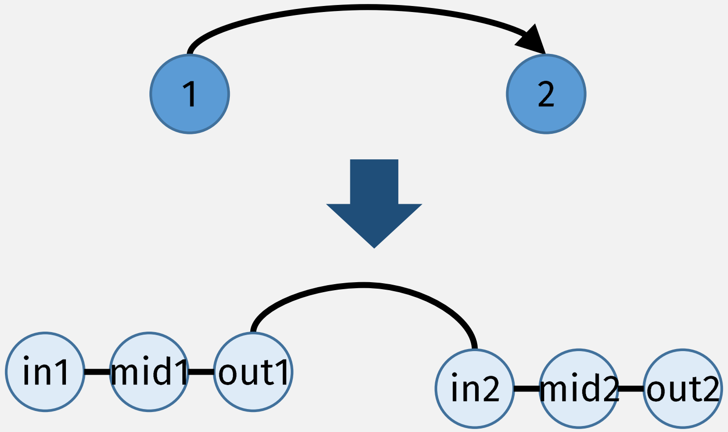
Need: **Computable function from *HAMPATH* to *UHAMPATH***

Better Idea:



- **Distinguish "in"** vs **"out" edges**
- **Nodes** (directed) → **3 Nodes** (undirected)**: in/mid/out**
  - Connect in/mid/out with edges
  - Directed edge (u, v) → (u_{out}, v_{in})
- Except: s → s_{out}, t → t_{in} **<u>only!</u>**





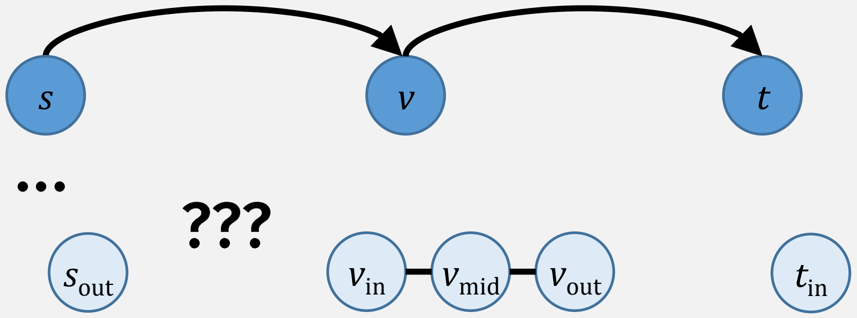
<u>Theorem</u>: *UHAMPATH* is NP-complete

$UHAMPATH = \{\langle G, s, t\rangle \mid G$ is a undirected graph with a Hamiltonian path from s to $t\}$

Need: **Computable function from *HAMPATH* to *UHAMPATH***

⇒ If **there is a directed path from s to t ...**

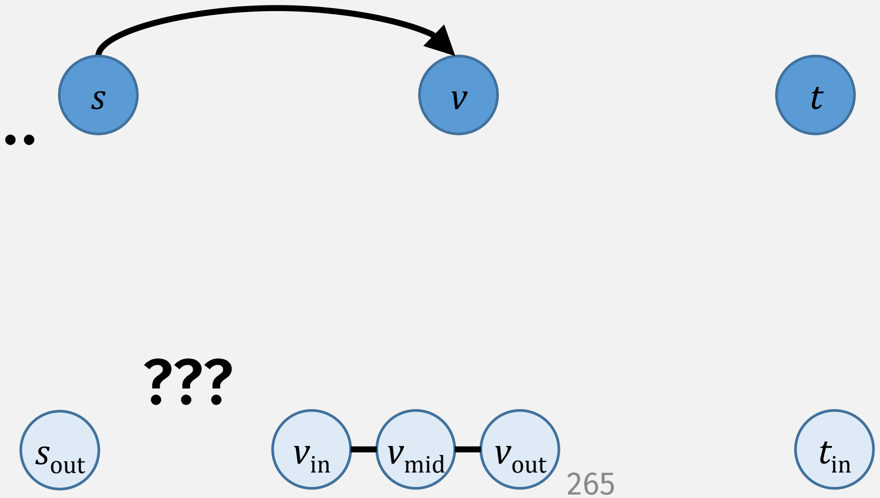
- ... then **there must be an undirected path ...**
- Because ...



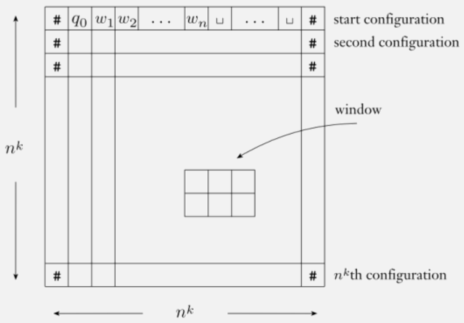
⇐ If **there is <u>no</u> directed path from s to t ...**

- ... then **there is <u>no</u> undirected path ...**
- Because ...

Finish this proof for HW 12

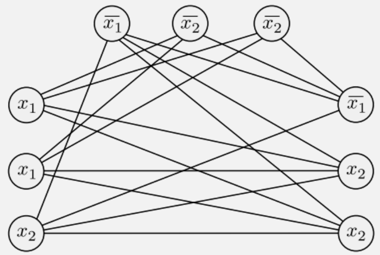


NP-Complete problems, so far

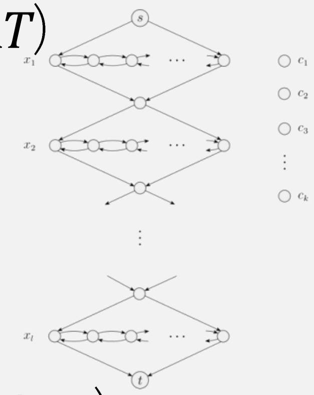


- $SAT = \{\langle\phi\rangle \mid \phi$ is a satisfiable Boolean formula$\}$ (Cook-Levin Theorem)

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



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



- $HAMPATH = \{\langle G, s, t\rangle \mid G$ is a directed graph with a Hamiltonian path from s to $t\}$ (reduce from 3*SAT*)

- $UHAMPATH = \{\langle G, s, t\rangle \mid G$ is a undirected graph with a Hamiltonian path from s to $t\}$ (reduce from *HAMPATH*)

Quiz 5/4