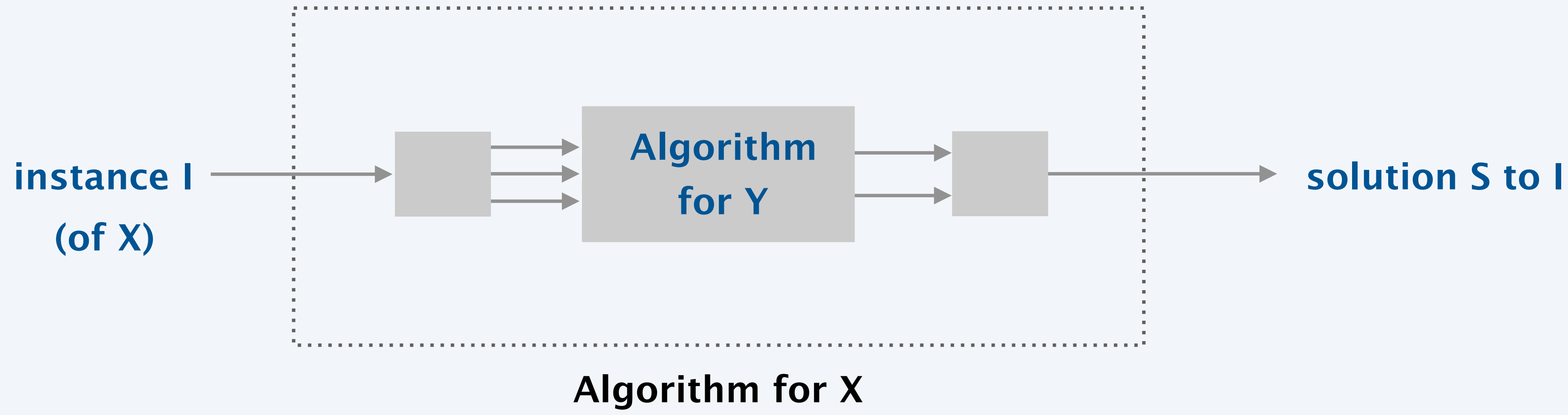
Polynomial-time reductions

Suppose Y in P. What else is in P?

Reduction. Problem X polynomial-time (Cook) reduces to problem Y if arbitrary instances of problem X can be solved using:

- Polynomial number of standard computational steps, plus
- Polynomial number of calls to oracle that solves problem Y.

computational model supplemented by special piece of hardware that solves instances of Y in a single step



Polynomial-time reductions

Suppose Y in P. What else is in P?

Reduction. Problem X polynomial-time (Cook) reduces to problem Y if arbitrary instances of problem X can be solved using:

- Polynomial number of standard computational steps, plus
- Polynomial number of calls to oracle that solves problem Y.

Notation. $X \leq_P Y$.

Note. We pay for time to write down instances sent to oracle $\Rightarrow$ instances of Y must be of polynomial size.

Caveat. Don't mistake $X \leq_P Y$ with $Y \leq_P X$.

Polynomial-time reductions

Design algorithms. If $X \leq_P Y$ and Y can be solved in polynomial time, then X can be solved in polynomial time.

Establish intractability. If $X \leq_P Y$ and X cannot be solved in polynomial time, then Y cannot be solved in polynomial time.

Establish equivalence. If both $X \leq_P Y$ and $Y \leq_P X$, we use notation $X \equiv_P Y$. In this case, X can be solved in polynomial time iff Y can be.

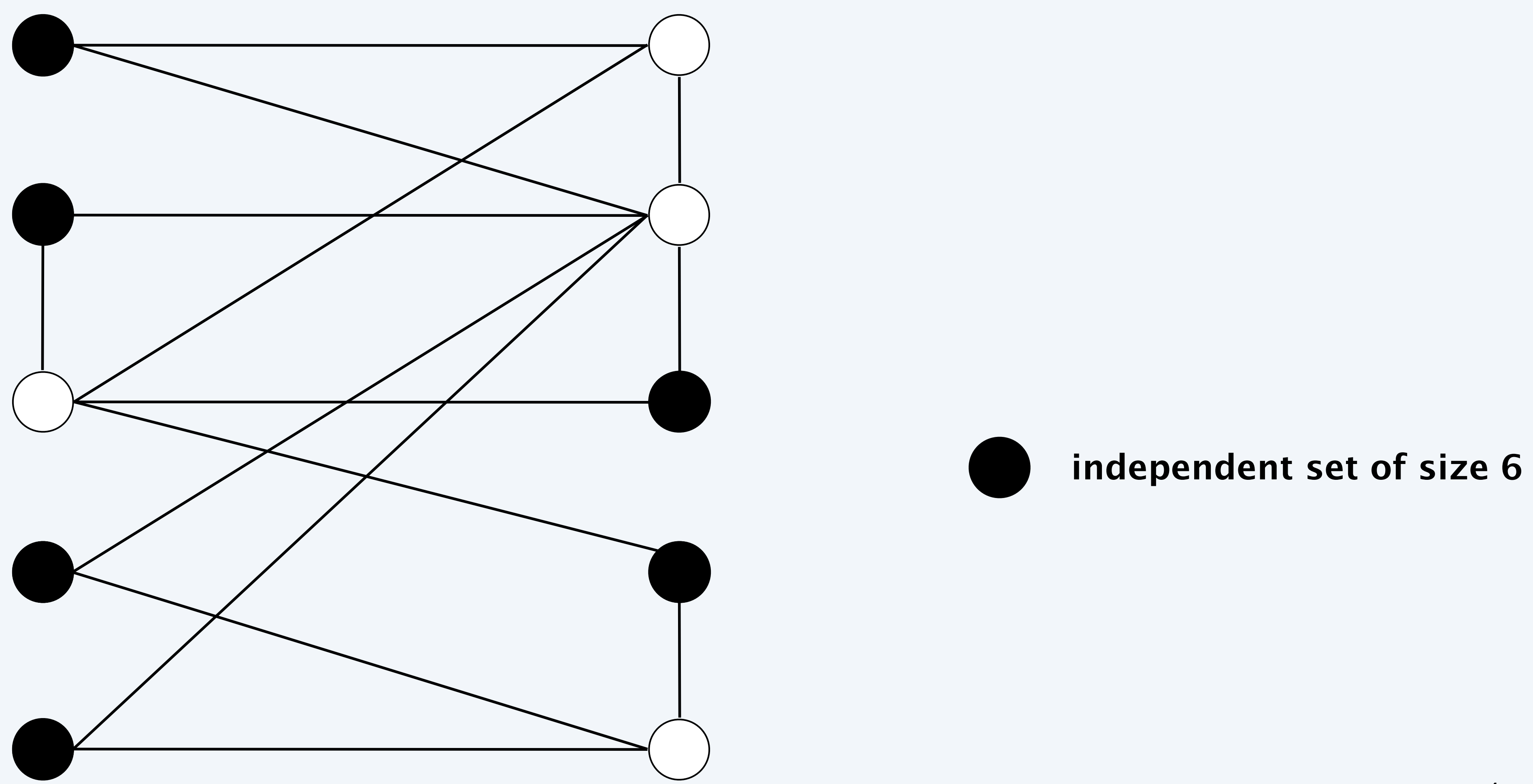
Bottom line. Reductions classify problems according to relative difficulty.

Independent set

INDEPENDENT-SET. Given graph $G = (V, E)$ and integer k, is there subset $S \subseteq V$, with $|S| \geq k$, s.t. no edge contained in S ?

Ex. Is there an independent set of size ≥ 6 ?

Ex. Is there an independent set of size ≥ 7 ?

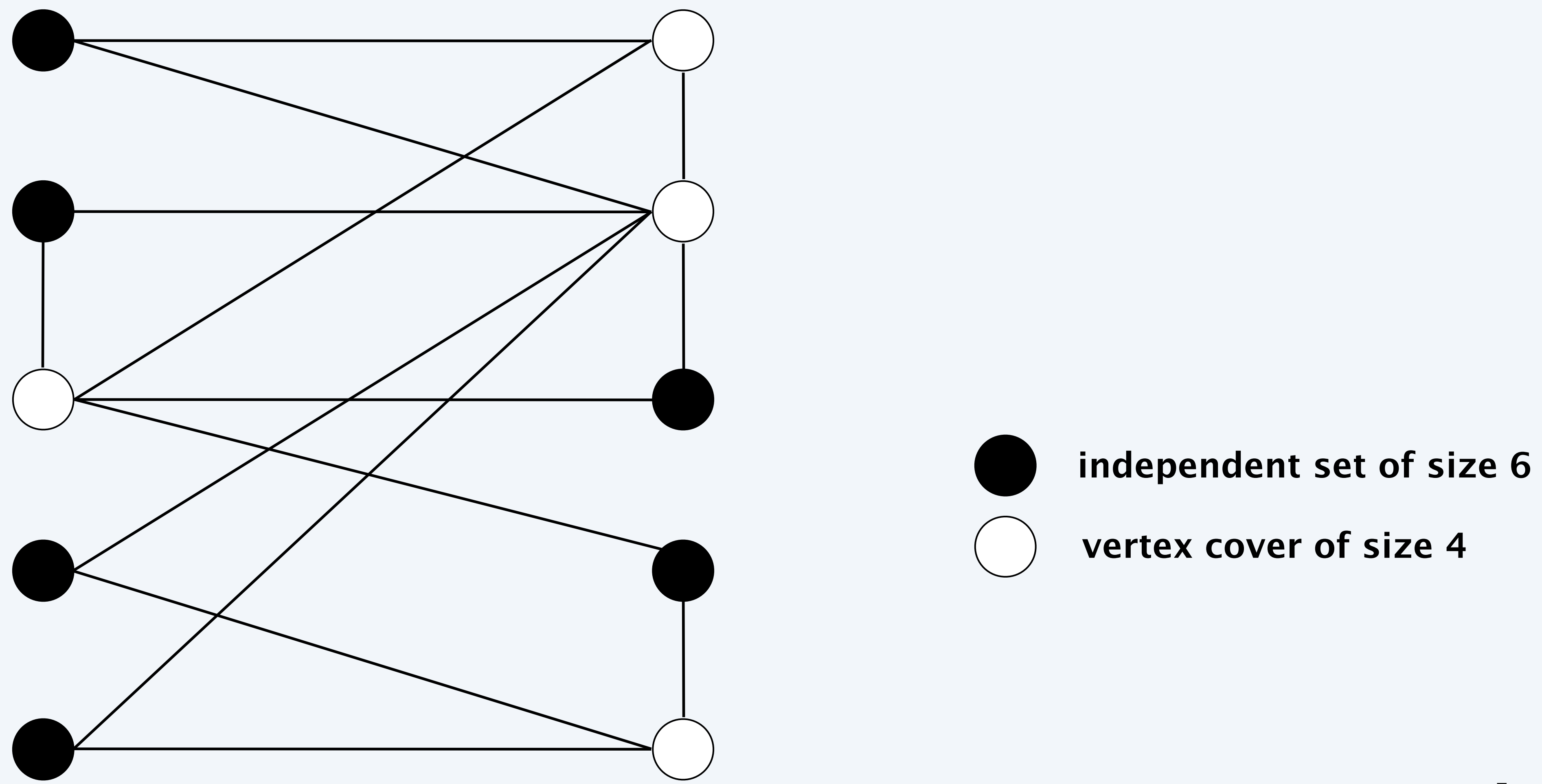


Vertex cover

VERTEX-COVER. Given graph $G = (V, E)$ and integer k, is there $S \subseteq V$ with $|S| \leq k$, s.t. each edge touches S ?

Ex. Is there a vertex cover of size ≤ 4 ?

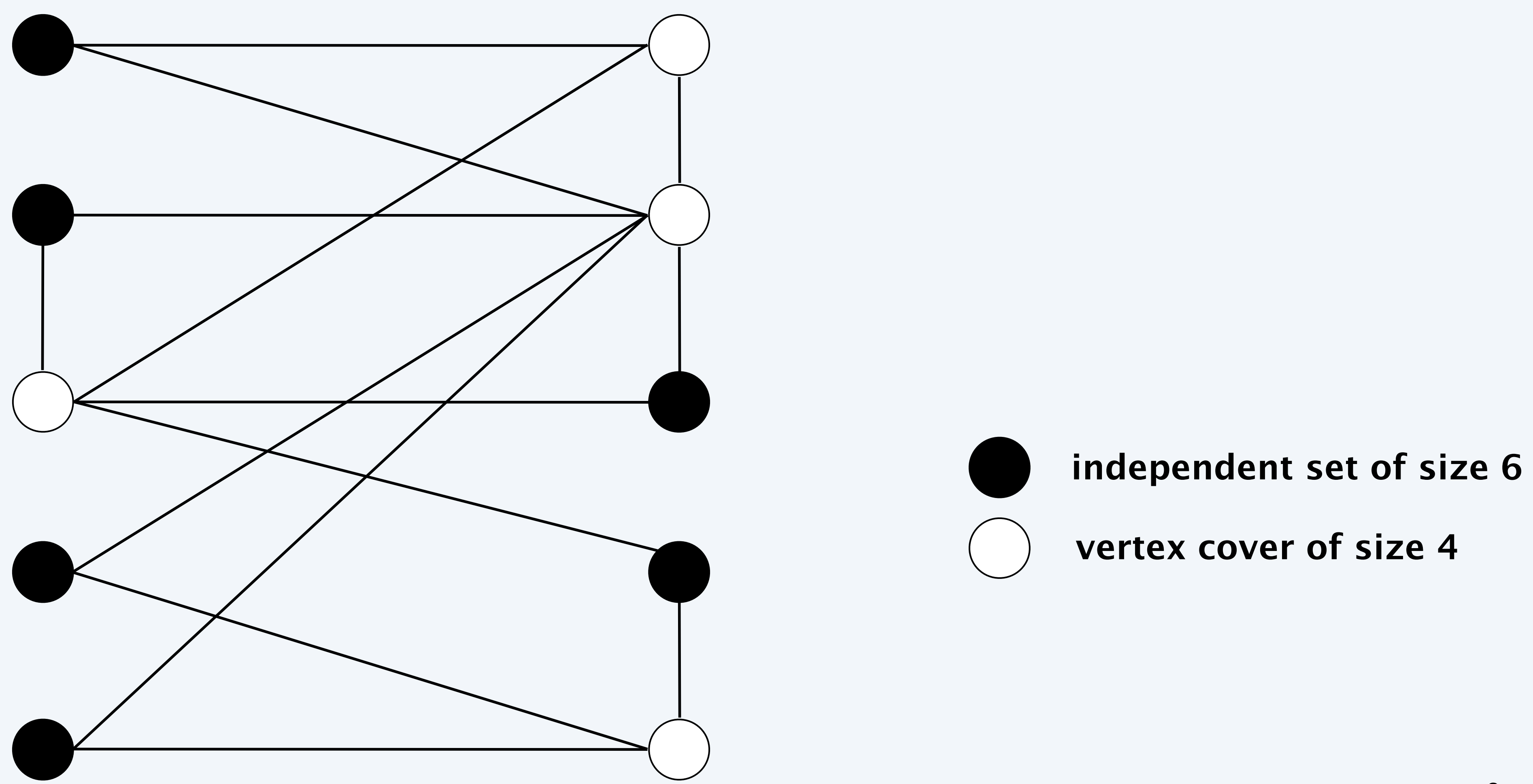
Ex. Is there a vertex cover of size ≤ 3 ?



Vertex cover and independent set reduce to one another

Theorem. VERTEX-COVER $\equiv_P$ INDEPENDENT-SET.

Pf. We show S is an independent set of size k iff $V - S$ is a vertex cover of size $n - k$.



Vertex cover and independent set reduce to one another

Theorem. VERTEX-COVER $\equiv_P$ INDEPENDENT-SET.

Pf. We show S is an independent set of size k iff $V - S$ is a vertex cover of size $n - k$.

$\Rightarrow$

- Let S be independent set.
- Consider edge $\{u, v\}$.
- S independent $\Rightarrow$ either $u \notin S$ or $v \notin S$ (or both)

  $\Rightarrow$ either $u \in V - S$ or $v \in V - S$ (or both).
- Thus, $V - S$ covers $\{u, v\}$.

Vertex cover and independent set reduce to one another

Theorem. VERTEX-COVER $\equiv_P$ INDEPENDENT-SET.

Pf. We show S is an independent set of size k iff $V - S$ is a vertex cover of size $n - k$.

$\Leftarrow$

- Let $V - S$ be vertex cover.
- Consider two nodes $u \in S$ and $v \in S$.
- $\{u, v\} \notin E$ since $V - S$ is a vertex cover $\Rightarrow$ S independent set. ▪

Set cover

SET-COVER. Given a collection $S_1, S_2, \ldots, S_m$ of subsets of $\{1,2,\ldots,n\}$, and an integer k, does there exist $\leq k$ of these sets whose union is equal to U ?

Sample application.

- m available pieces of software.
- Set of n capabilities that we would like our system to have.
- The i^{th} piece of software provides the set $S_i \subseteq U$ of capabilities.
- Goal: achieve all n capabilities using fewest pieces of software.

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_1 = \{ 3, 7 \}$ $\quad$ $S_4 = \{ 2, 4 \}$

$S_2 = \{ 3, 4, 5, 6 \}$ $\quad$ $S_5 = \{ 5 \}$

$S_3 = \{ 1 \}$ $\quad$ $S_6 = \{ 1, 2, 6, 7 \}$

$k = 2$

a set cover instance

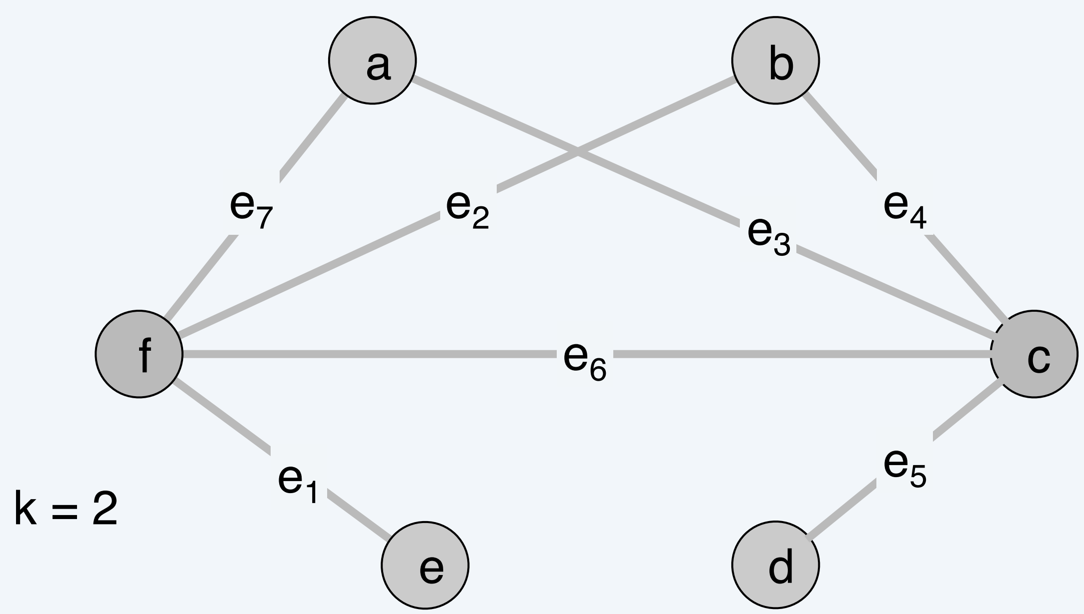
Vertex cover reduces to set cover

Theorem. VERTEX-COVER $\leq_P$ SET-COVER.

Pf. Given VERTEX-COVER instance $G = (V, E)$, we construct a SET-COVER instance that has a set cover of size k iff G has a vertex cover of size k.

Construction.

- Universe = E.
- Include one set for each node $v \in V$: $S_v = \{e \in E : e \text{ incident to } v\}$.



vertex cover instance
(k = 2)

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $\quad$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $\quad$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $\quad$ $S_f = \{ 1, 2, 6, 7 \}$

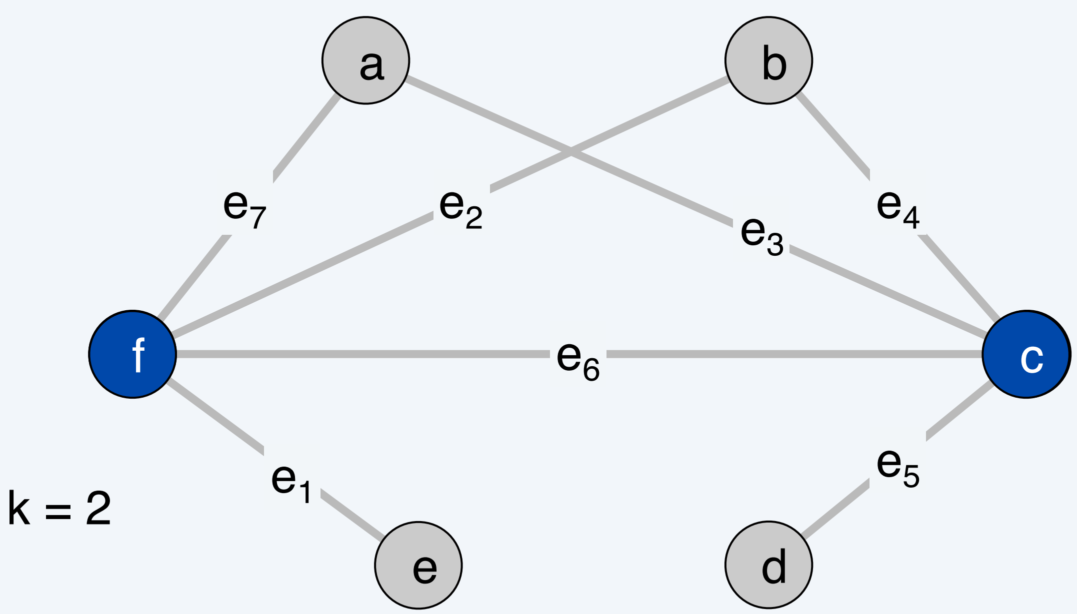
set cover instance
(k = 2)

Vertex cover reduces to set cover

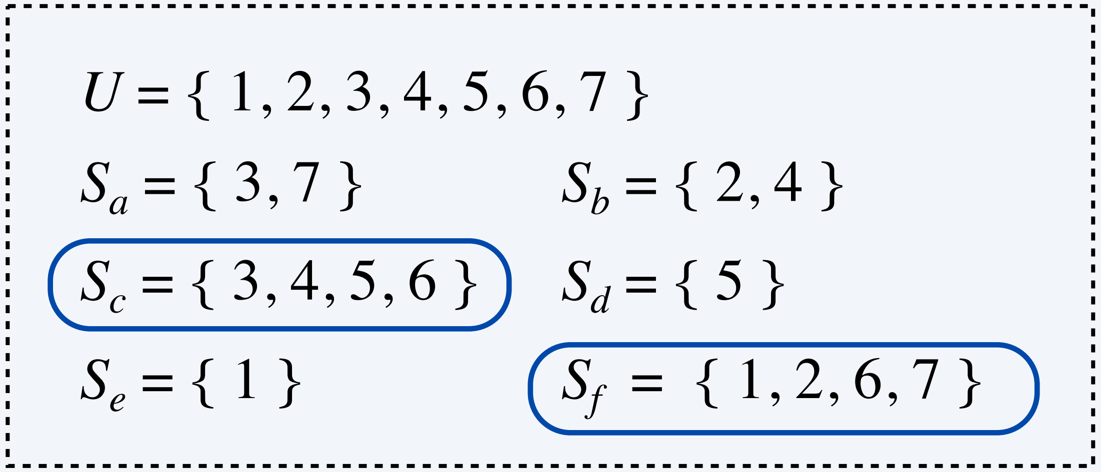
Lemma. $G = (V, E)$ contains a vertex cover of size k iff (U, S) contains a set cover of size k.

Pf. $\Rightarrow$ Let $X \subseteq V$ be a vertex cover of size k in G.

- Then $Y = \{ S_v : v \in X \}$ is a set cover of size k. ▪



vertex cover instance
(k = 2)



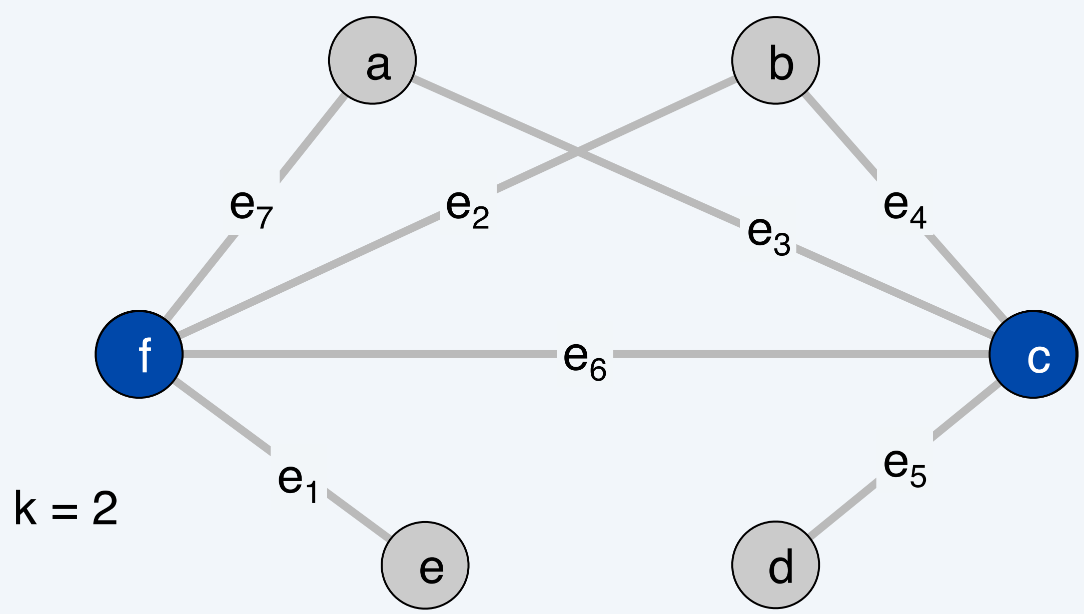
set cover instance
(k = 2)

Vertex cover reduces to set cover

Lemma. $G = (V, E)$ contains a vertex cover of size k iff (U, S) contains a set cover of size k.

Pf. $\Leftarrow$ Let $Y \subseteq S$ be a set cover of size k in (U, S).

- Then $X = \{ v : S_v \in Y \}$ is a vertex cover of size k in G. ▪



vertex cover instance
(k = 2)

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $\quad$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $\quad$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $\quad$ $S_f = \{ 1, 2, 6, 7 \}$

set cover instance
(k = 2)

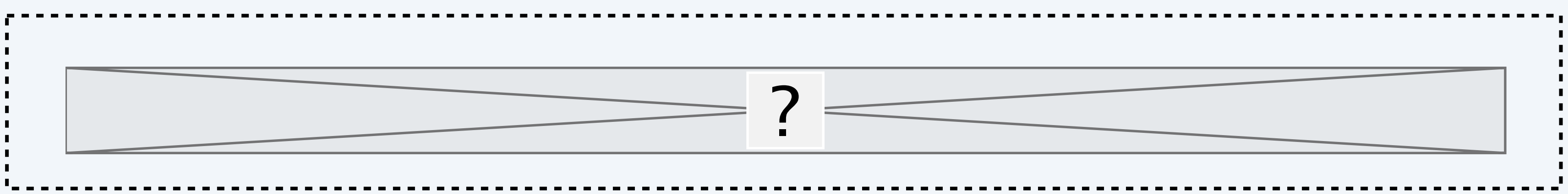
Satisfiability

Literal. A boolean variable or its negation.

Clause. A disjunction of literals.

Conjunctive normal form. A propositional formula Φ that is the conjunction of clauses.

SAT. Given CNF formula Φ, does it have a satisfying truth assignment?
3-SAT. SAT where each clause contains exactly 3 literals (and each literal corresponds to a different variable).



yes instance: x_1 = true, x_2 = true, x_3 = false, x_4 = false

Key application. Electronic design automation (EDA).

3-satisfiability reduces to independent set

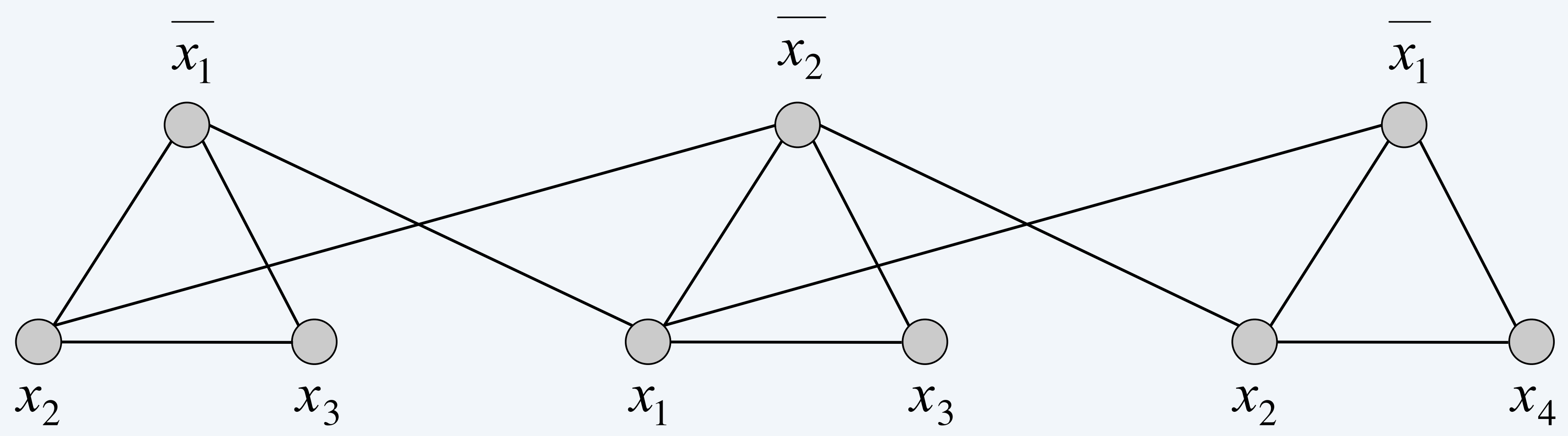
Theorem. 3-SAT $\le_P$ INDEPENDENT-SET.

Pf. Given an instance Φ of 3-SAT, we construct an instance (G, k) of INDEPENDENT-SET that has an independent set of size k iff Φ is satisfiable.

Construction.

- G contains 3 nodes for each clause, one for each literal.
- Connect 3 literals in a clause in a triangle.
- Connect literal to each of its negations.

G



k = 3

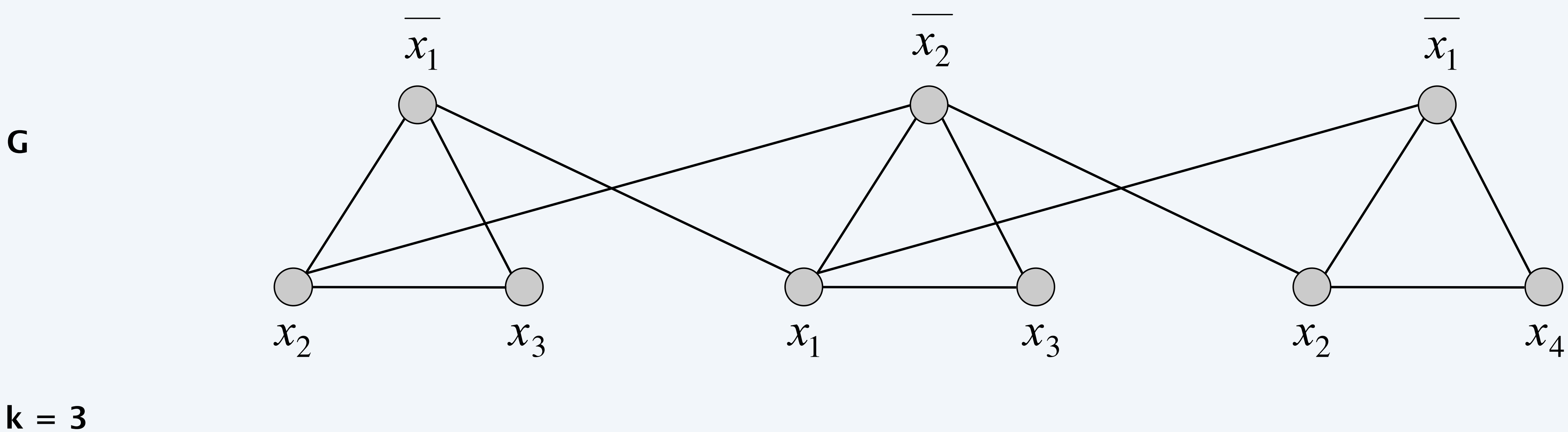
3-satisfiability reduces to independent set

Lemma. G contains independent set of size $k = |\Phi|$ iff Φ is satisfiable.

Pf. $\Rightarrow$ Let S be independent set of size k.

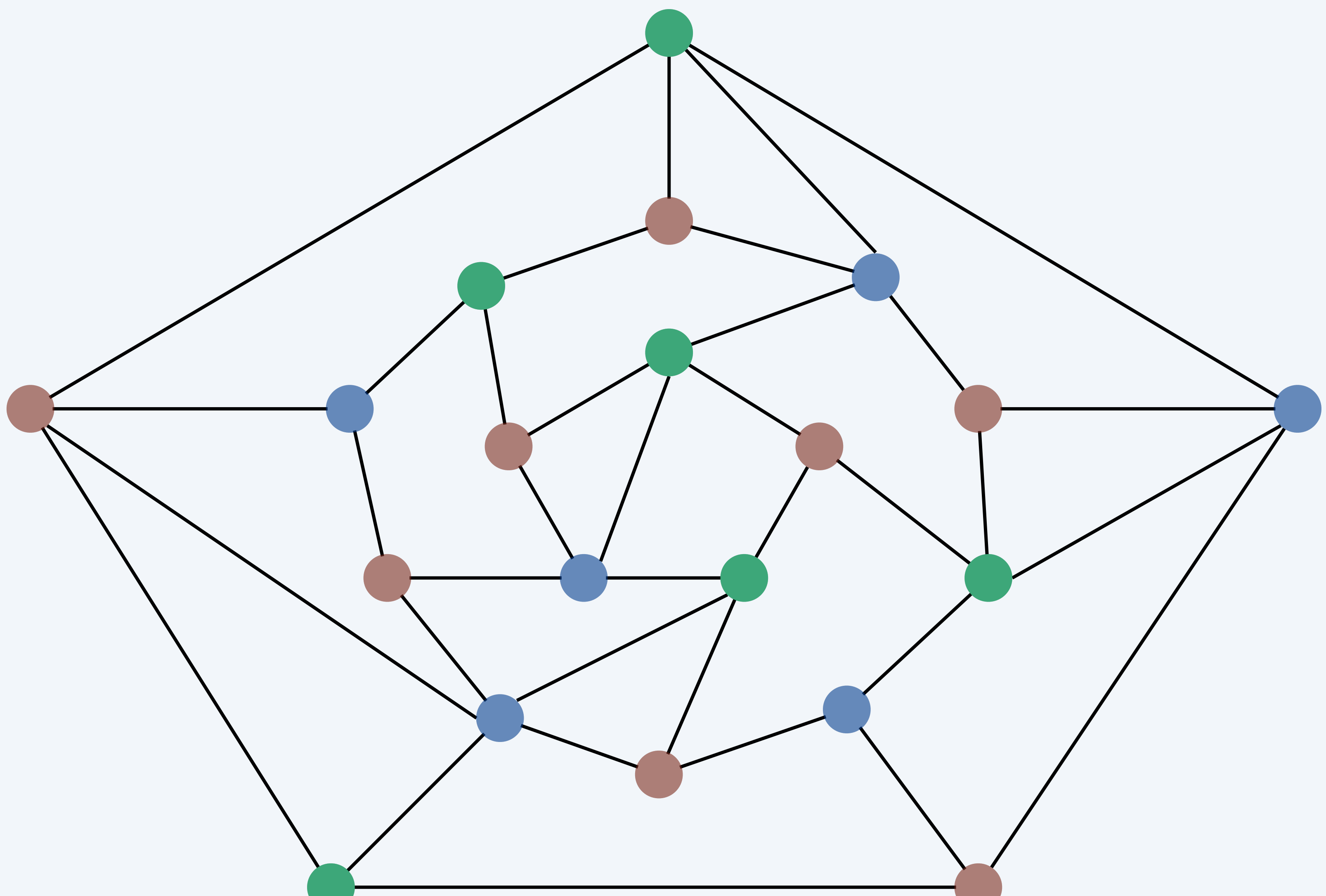
- S must contain exactly one node in each triangle.
- Set these literals to *true* (and remaining variables consistently).
- Truth assignment is consistent and all clauses are satisfied.

Pf $\Leftarrow$ Given satisfying assignment, select one true literal from each triangle. This is an independent set of size k. ▪



3-colorability

3-COLOR. Given an undirected graph G, can the nodes be colored red, green, and blue so that no adjacent nodes have the same color?



yes instance

3-satisfiability reduces to 3-colorability

Theorem. 3-SAT $\leq_P$ 3-COLOR.

Pf. Given 3-SAT instance Φ, we construct an instance of 3-COLOR that is 3-colorable iff Φ is satisfiable.

3-satisfiability reduces to 3-colorability

Construction.

(i) Create a graph G with a node for each literal.

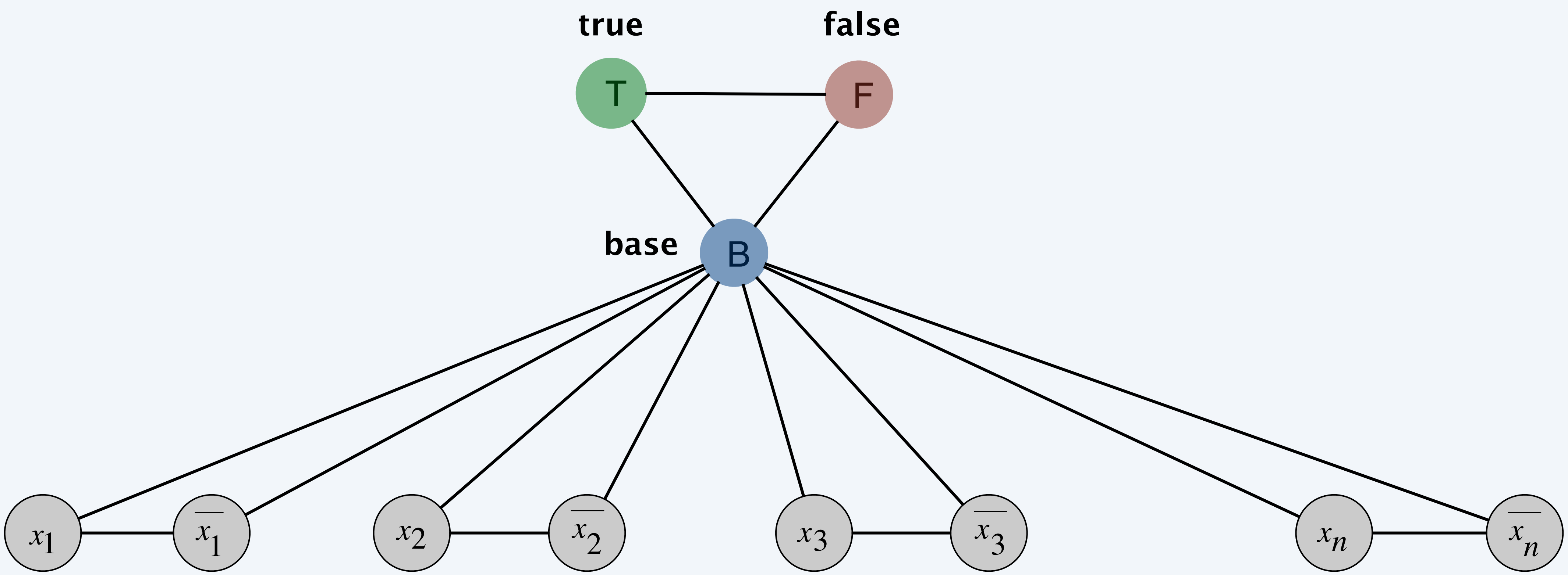
(ii) Connect each literal to its negation.

(iii) Create 3 new nodes T, F, and B; connect them in a triangle.

(iv) Connect each literal to B.

(v) For each clause C_j, add a gadget of 6 nodes and 13 edges.

to be described later

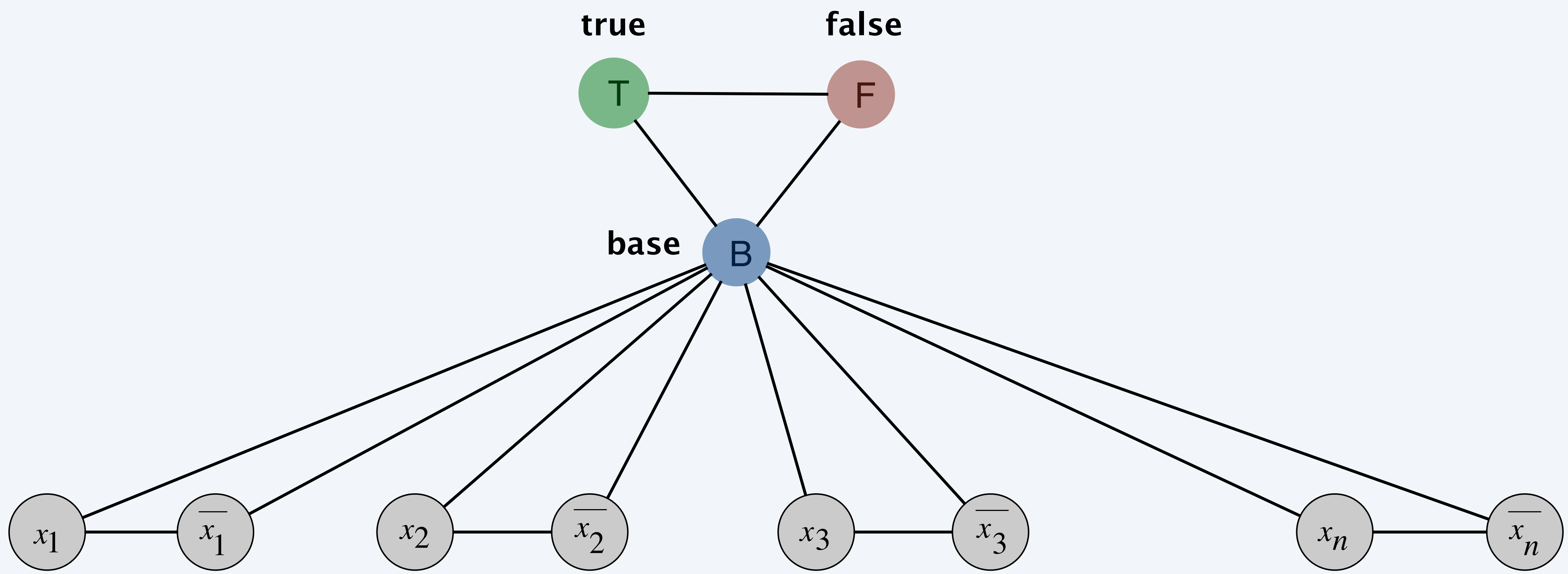


3-satisfiability reduces to 3-colorability

Lemma. Graph G is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph G is 3-colorable.

- Consider assignment that sets all T literals to true.
- (iv) ensures each literal is T or F.
- (ii) ensures a literal and its negation are opposites.

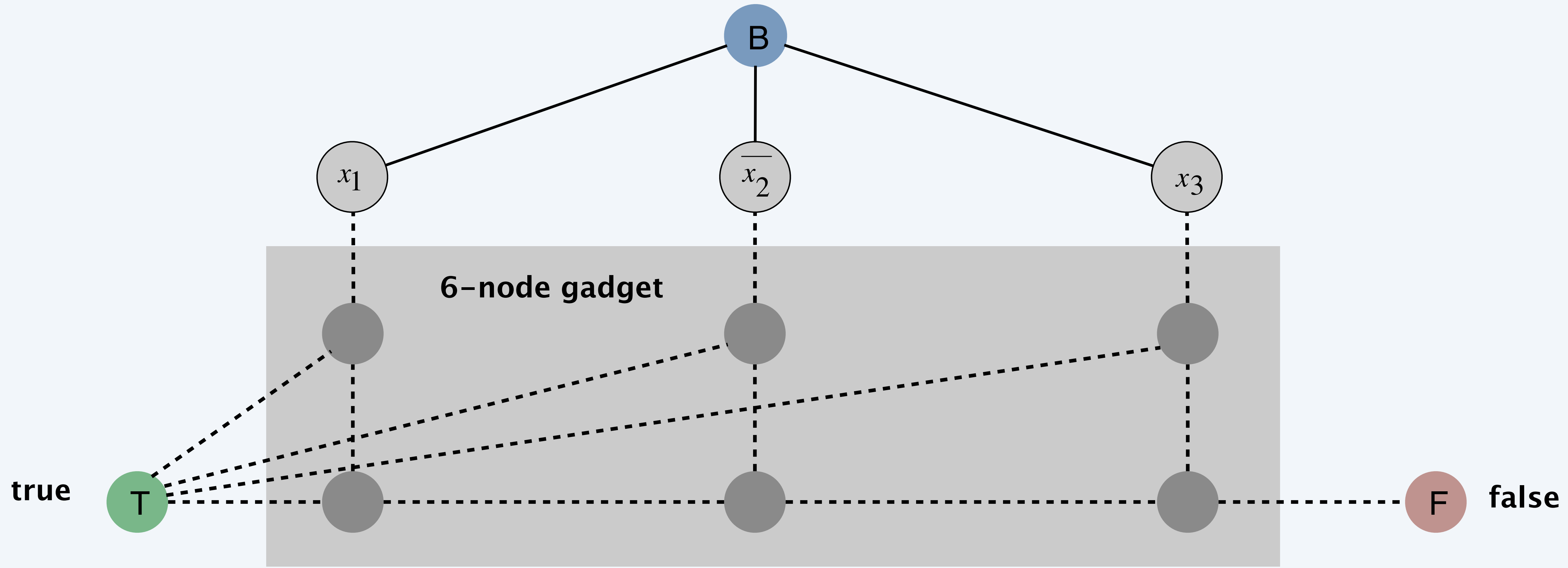


3-satisfiability reduces to 3-colorability

Lemma. Graph G is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph G is 3-colorable.

- Consider assignment that sets all T literals to true.
- (iv) ensures each literal is T or F.
- (ii) ensures a literal and its negation are opposites.
- (v) ensures at least one literal in each clause is T.

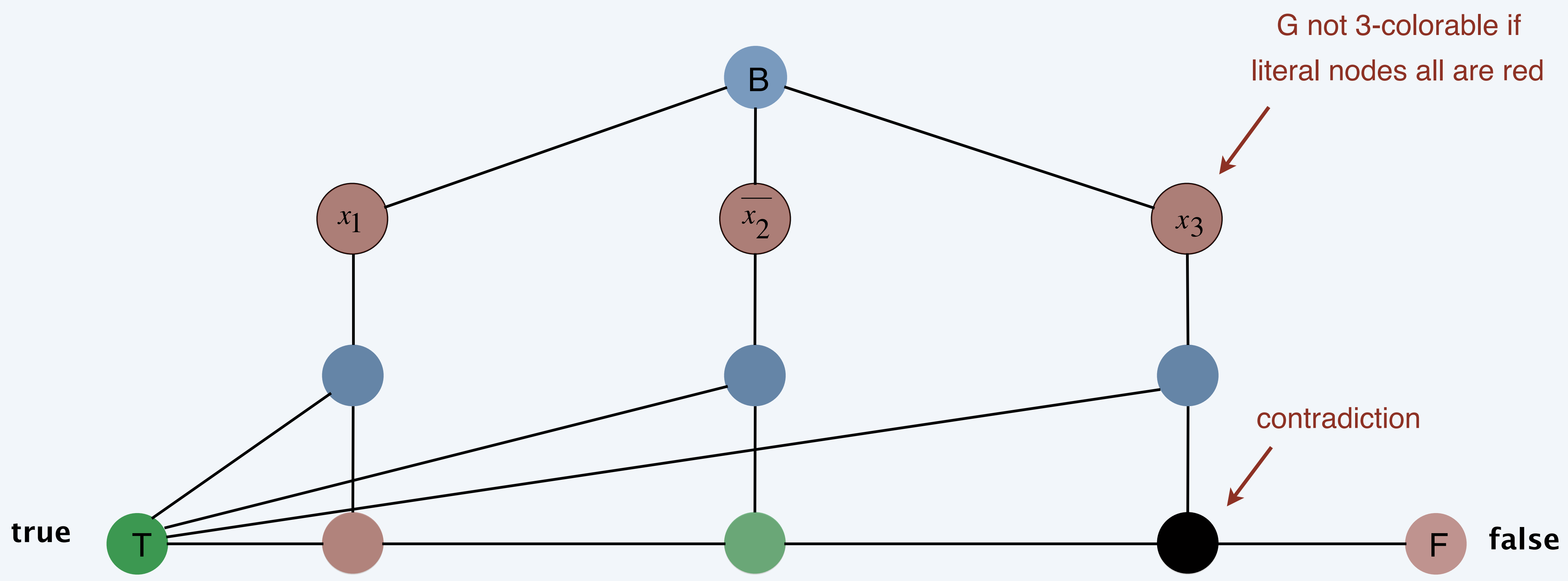


3-satisfiability reduces to 3-colorability

Lemma. Graph G is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph G is 3-colorable.

- Consider assignment that sets all T literals to true.
- (iv) ensures each literal is T or F.
- (ii) ensures a literal and its negation are opposites.
- (v) ensures at least one literal in each clause is T.



3-satisfiability reduces to directed hamilton cycle

DIR-HAM-CYCLE: Given a digraph $G = (V, E)$, does there exist a simple directed cycle Γ that contains every node in V ?

Theorem. 3-SAT $\leq_P$ DIR-HAM-CYCLE.

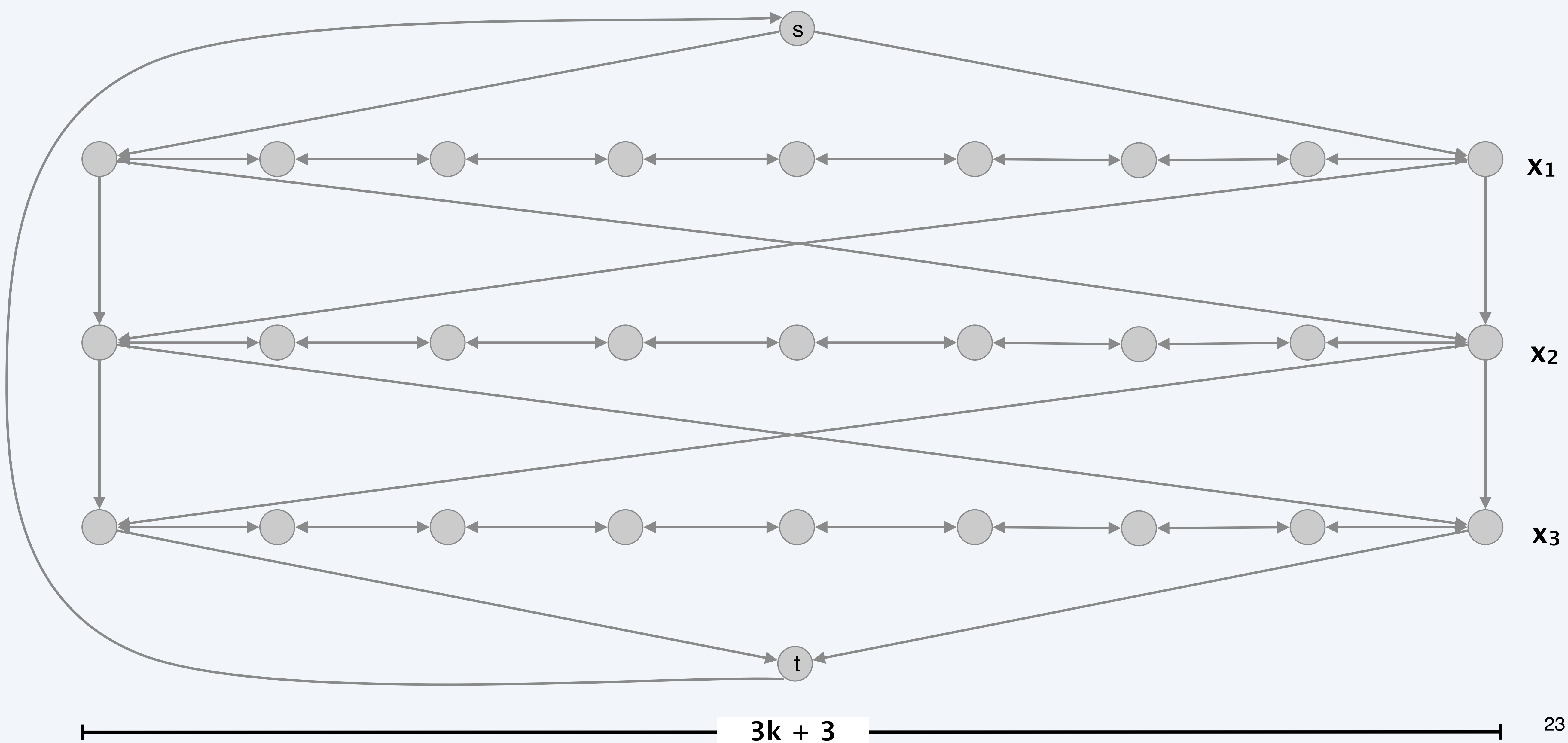
Pf. Given an instance Φ of 3-SAT, we construct an instance of DIR-HAM-CYCLE that has a Hamilton cycle iff Φ is satisfiable.

Construction. First, create graph that has 2^n Hamilton cycles which correspond in a natural way to 2^n possible truth assignments.

3-satisfiability reduces to directed hamilton cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

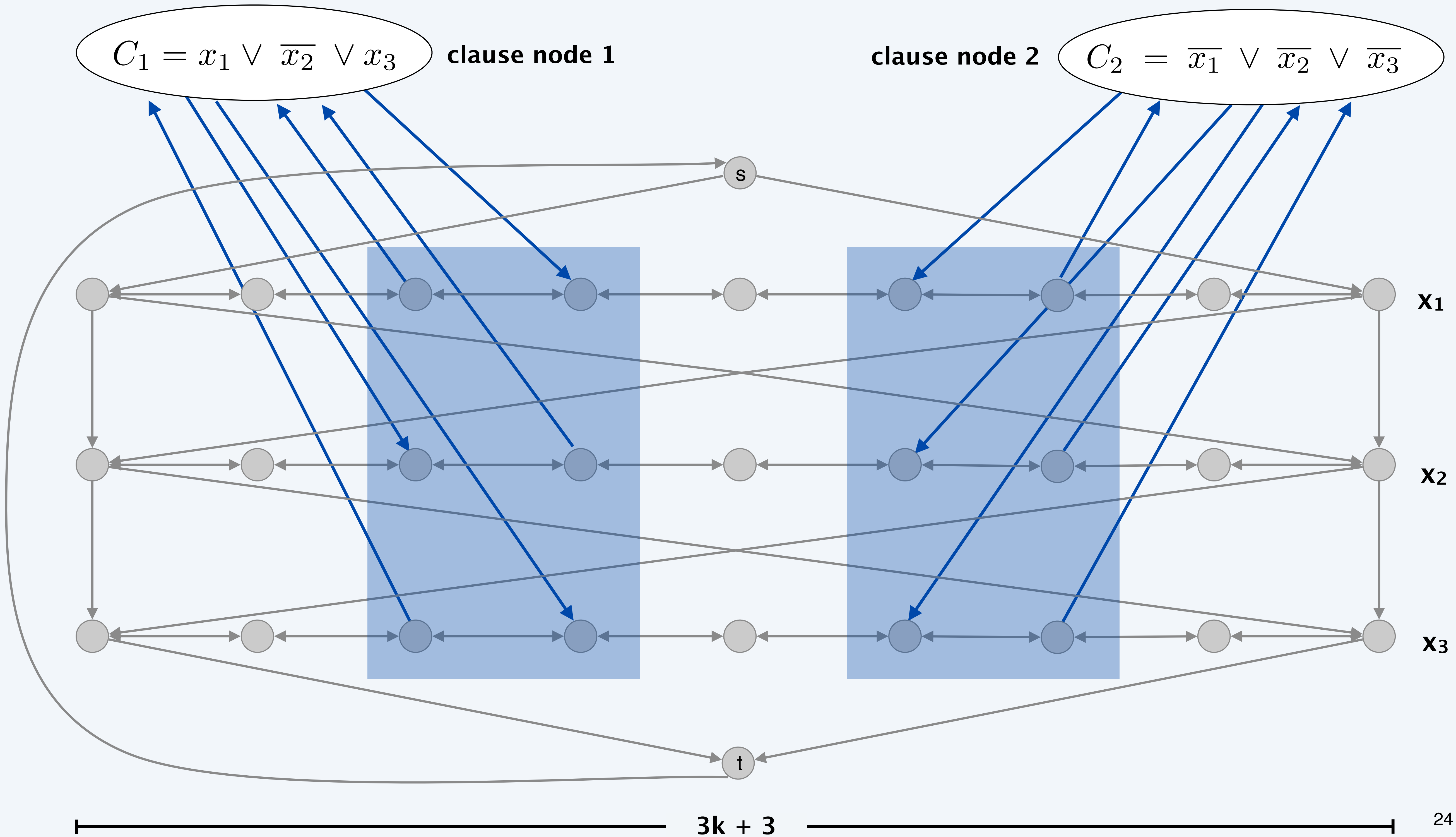
- Construct G to have 2^n Hamilton cycles.
- Intuition: traverse path i from left to right $\Leftrightarrow$ set variable $x_i = true$.



3-satisfiability reduces to directed hamilton cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

- For each clause, add a node and 6 edges.



3-satisfiability reduces to directed hamilton cycle

Lemma. Φ is satisfiable iff G has a Hamilton cycle.

Pf. $\Rightarrow$

- Suppose 3-SAT instance has satisfying assignment x^*.
- Then, define Hamilton cycle in G as follows:
  - if $x^*_i = true$, traverse row i from left to right
  - if $x^*_i = false$, traverse row i from right to left
  - for each clause C_j , there will be at least one row i in which we are going in "correct" direction to splice clause node C_j into cycle (and we splice in C_j exactly once)

3-satisfiability reduces to directed hamilton cycle

Lemma. Φ is satisfiable iff G has a Hamilton cycle.

Pf. $\Leftarrow$

- Suppose G has a Hamilton cycle Γ.
- If Γ enters clause node C_j , it must depart on mate edge.
  - nodes immediately before and after C_j are connected by an edge $e \in E$
  - removing C_j from cycle, and replacing it with edge e yields Hamilton cycle on $G - \{ C_j \}$
- Continuing in this way, we are left with a Hamilton cycle Γ' in $G - \{ C_1 , C_2 , \ldots, C_k \}$.
- Set $x^*_i = true$ iff Γ' traverses row i left to right.
- Since Γ visits each clause node C_j , at least one of the paths is traversed in "correct" direction, and each clause is satisfied. ▪

Final