NP COMPLETE PROBLEMS

REDUCTIONS

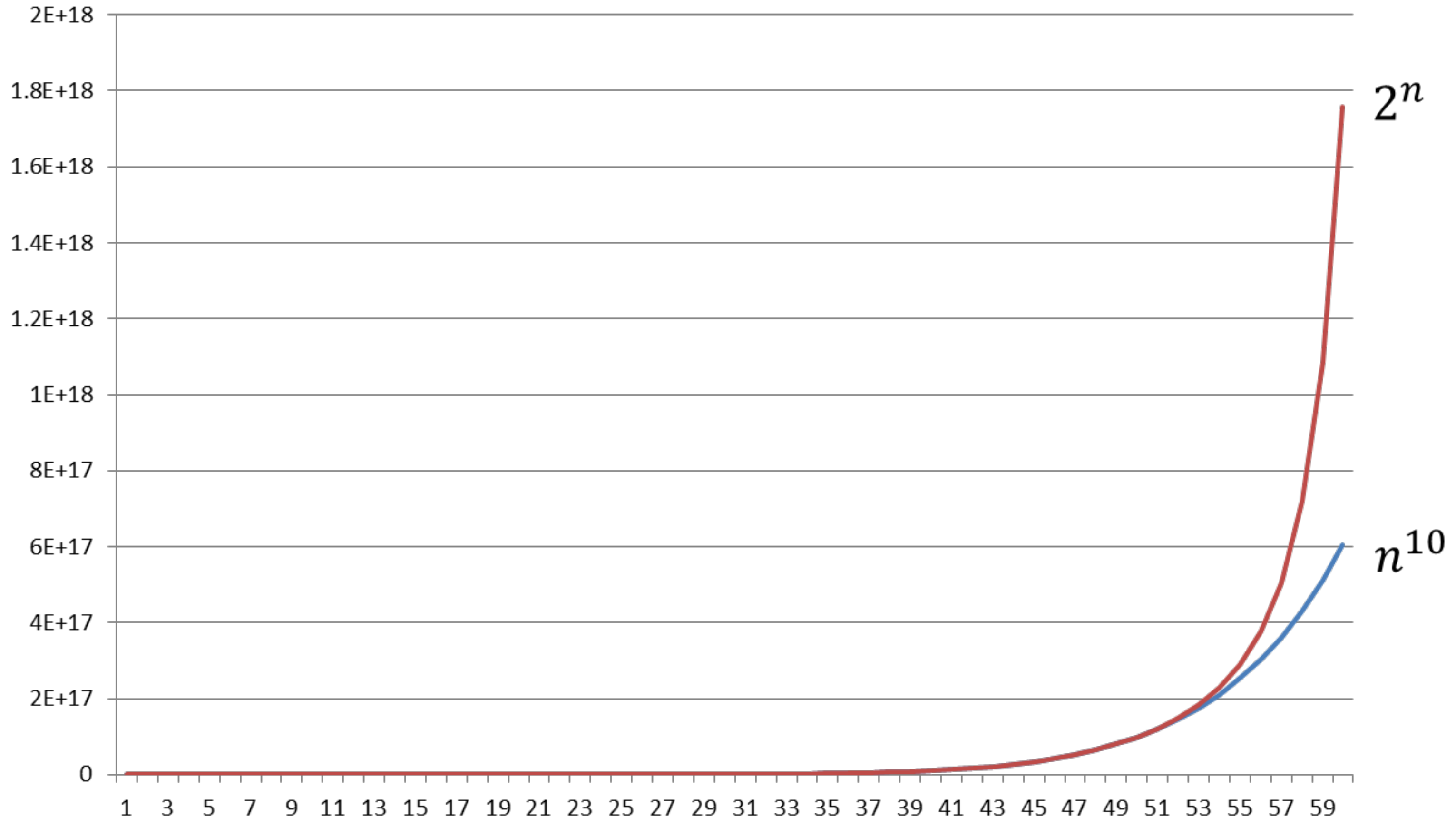
The hardness of a problem : whether an efficient algorithm exists

1. Binary Search, $O(\log n)$, **(supper fast)**
2. Sequential Search, $O(n)$, **(easy and fast)**
3. Sorting, $O(n \log n)$, **(good)**
4. Shortest Path $O(n^2)$, **(smart)**
5. Minimum spanning tree, $O(n^2)$, **(reasonably easy and fast)**
6. Maximum flow (Minimum cut) $O(n^k)$, **(not so bad to have a k)**
7. Eulerian Path $O(n^k)$, **(not so bad to have a reasonable k)**
8. Hamiltonian path **(not good, stuck with $O(e^n)$)**
9. Maximum cut (Minimum flow) **(stuck with $O(e^n)$ again)**
10. Clique **(stuck with $O(e^n)$ again)**
11. Vertex cover **(stuck with $O(e^n)$ again)**
12. Subset Sum **(whoop, it's not as easy as it looks, $O(e^n)$)**
13. SAT **(bad $O(e^n)$)**

What is going on?

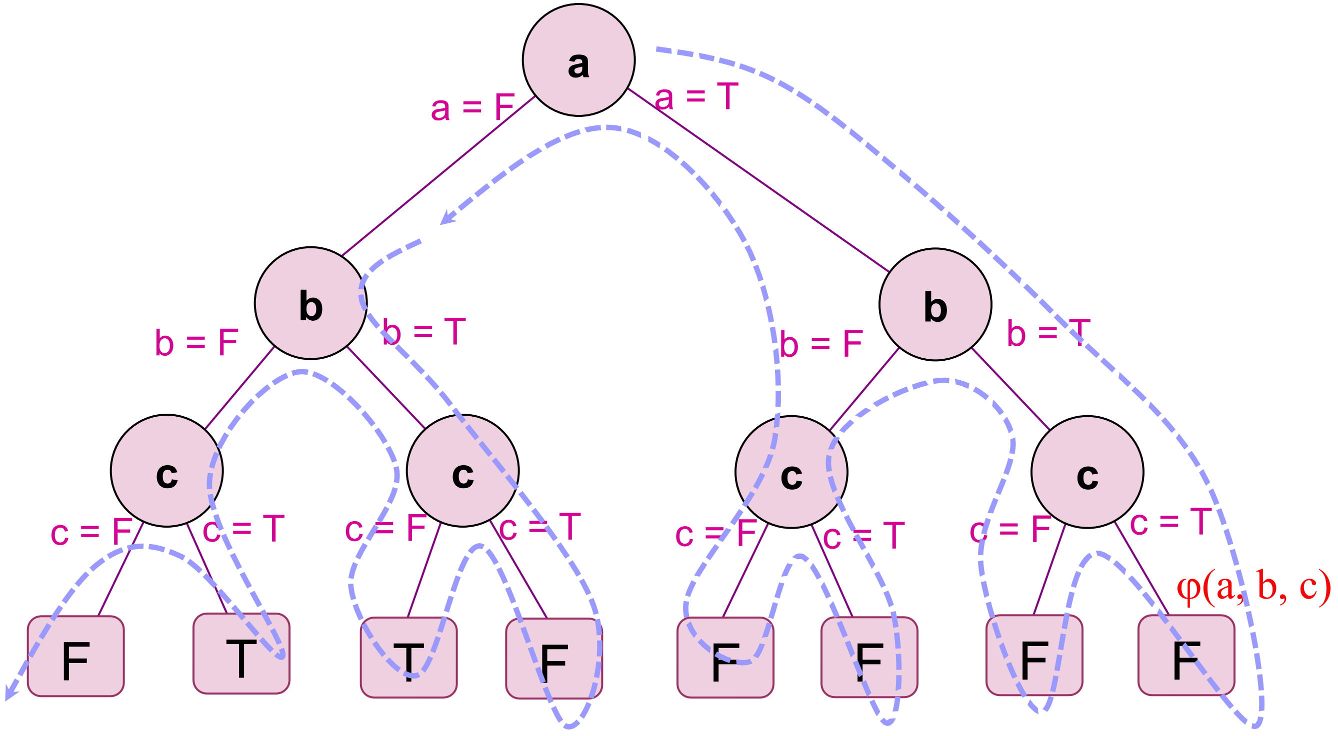
NP-C (NP-complete Problem)

$n^{10} \ll 2^n$ P << Exp



BOOLEAN FORMULA: φ(A, B, C)

Is φ satisfiable?



$O(2^n) \in \text{Exp}$

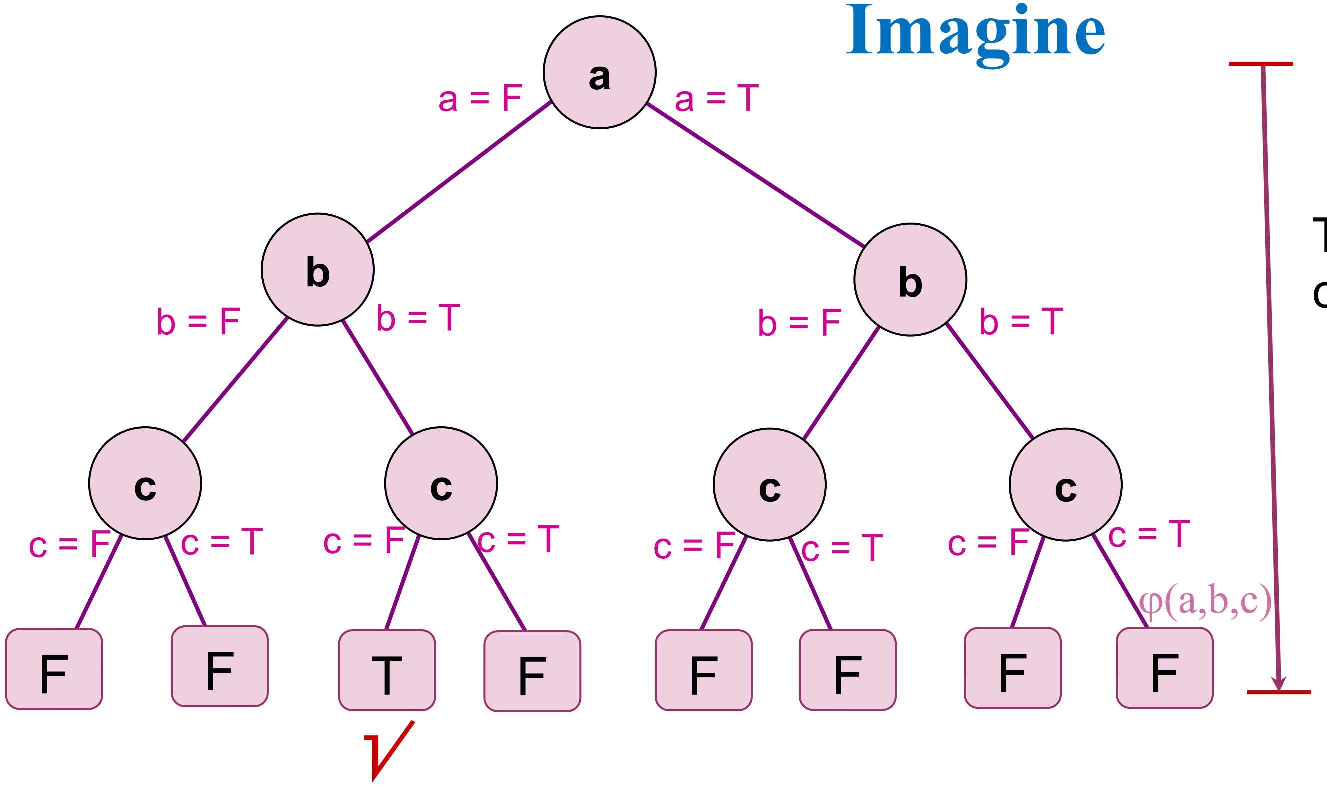
Backtracking algorithm

BOOLEAN FORMULA: $\varphi(A,B,C)$

- Not parallel computing
- Not quantum algorithms

Nondeterministic Algorithms

Imagine



The depth of the computational tree

$O(n) \in$ P

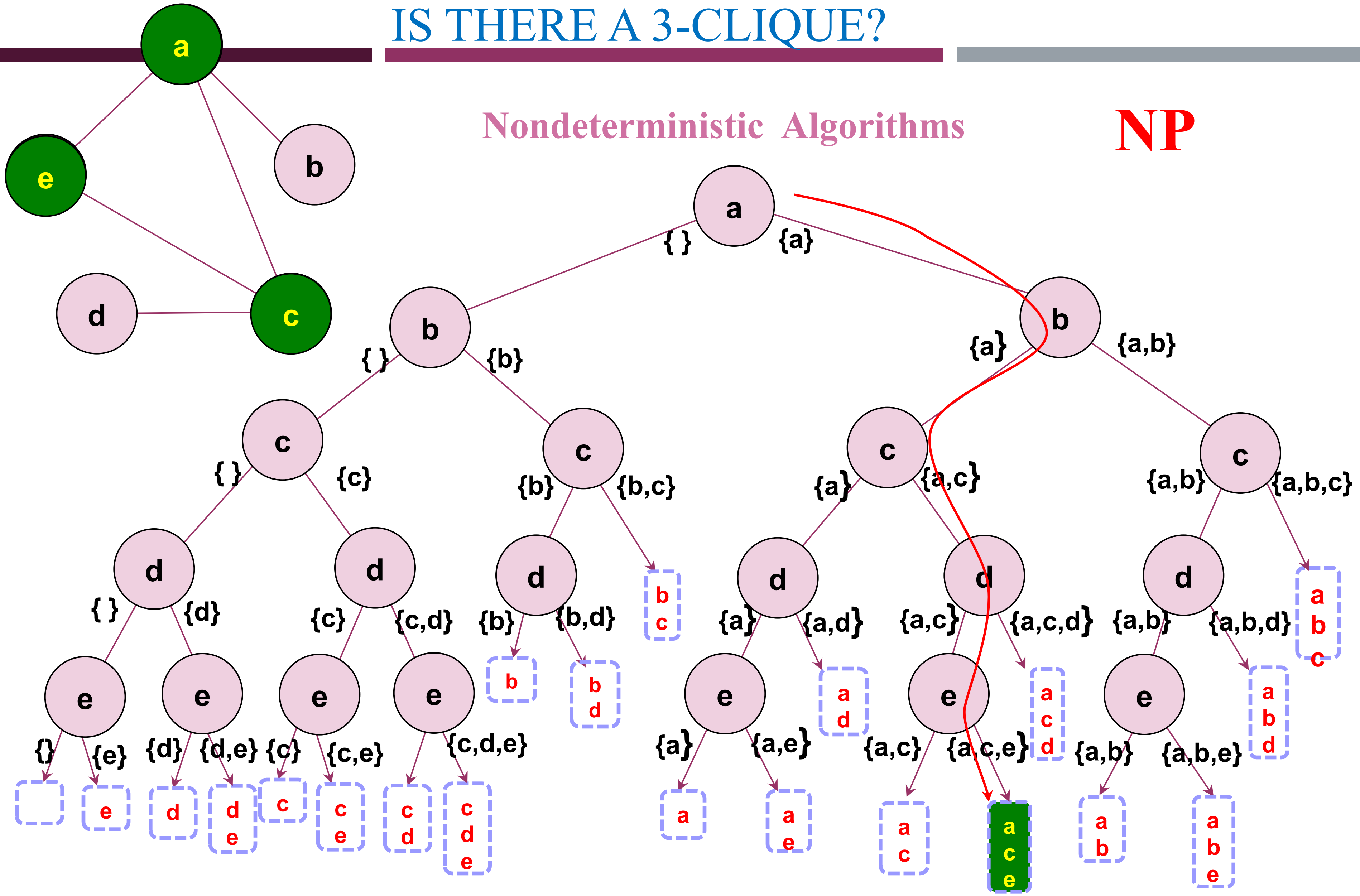
Nondeterministic Polynomial time computable

NP

IS THERE A 3-CLIQUE?

Nondeterministic Algorithms

NP



NP $\equiv$ **The set of Nondeterministic Polynomial Time Computable Problems (i.e problems that can be solved in polynomial time using *nondeterministic* algorithms.)**

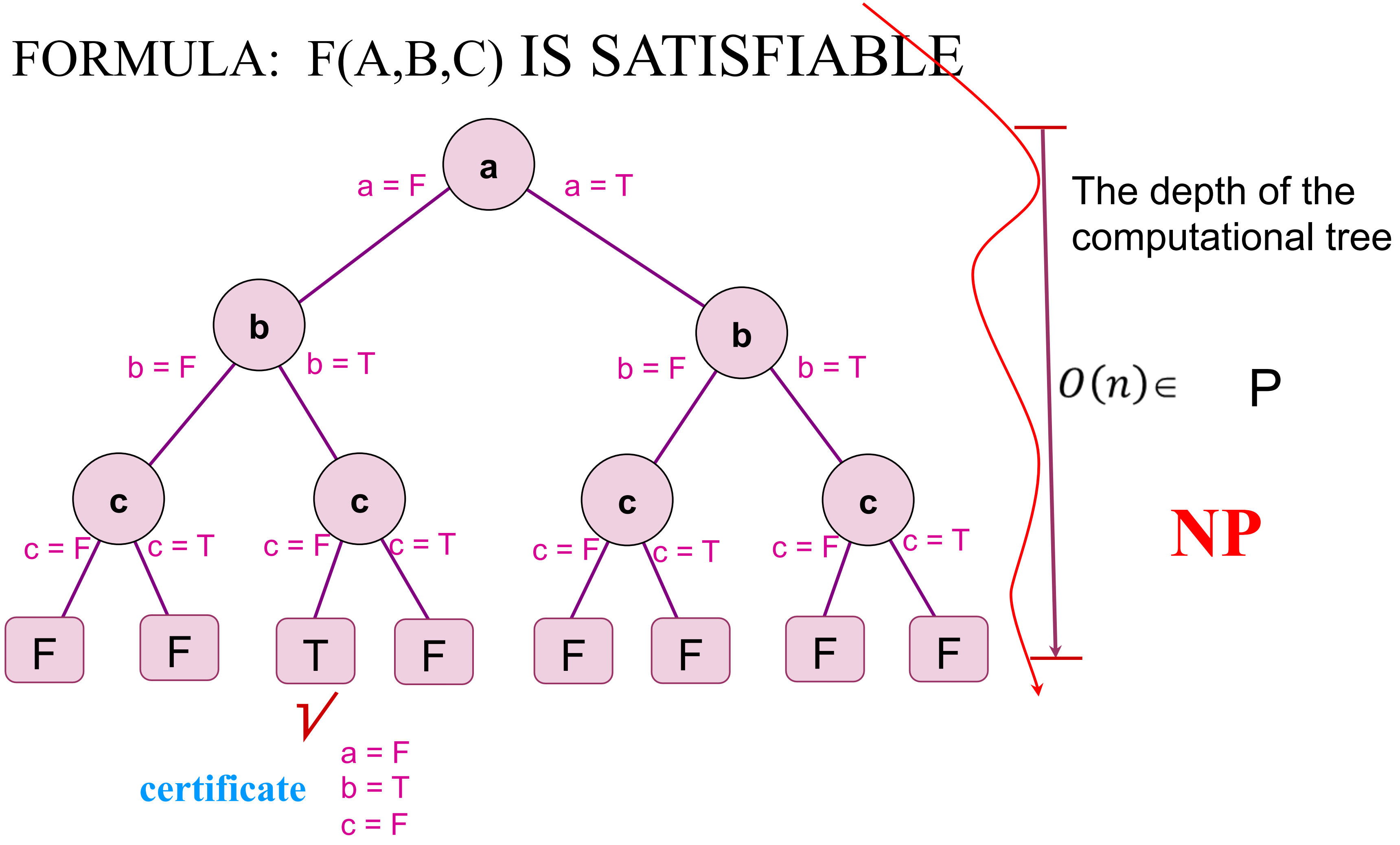
1. Binary Search, O(log n), **(supper fast)**
2. Sequential Search, O(n), **(easy and fast)**
3. Sorting, O(n log n), **(good)**
4. Shortest Path $O(n^2)$, **(smart)**
5. Minimum spanning tree, $O(n^2)$, **(reasonably easy and fast)**
6. Maximum flow (Minimum cut) $O(n^k)$, **(not so bad to have a k)**
7. Eulerian Path $O(n^k)$, **(not so bad to have a reasonable k)**
8. Hamiltonian path **(not good, stuck with $O(e^n)$)**
9. Maximum cut (Minimum flow) **(stuck with $O(e^n)$ again)**
10. Clique **(stuck with $O(e^n)$ again)**
11. Vertex cover **(stuck with $O(e^n)$ again)**
12. Subset Sum **(whoop, it's not as easy as it looks, $O(e^n)$)**
13. SAT **(bad $O(e^n)$)**

Explain why they are all in NP?

Another definition of NP

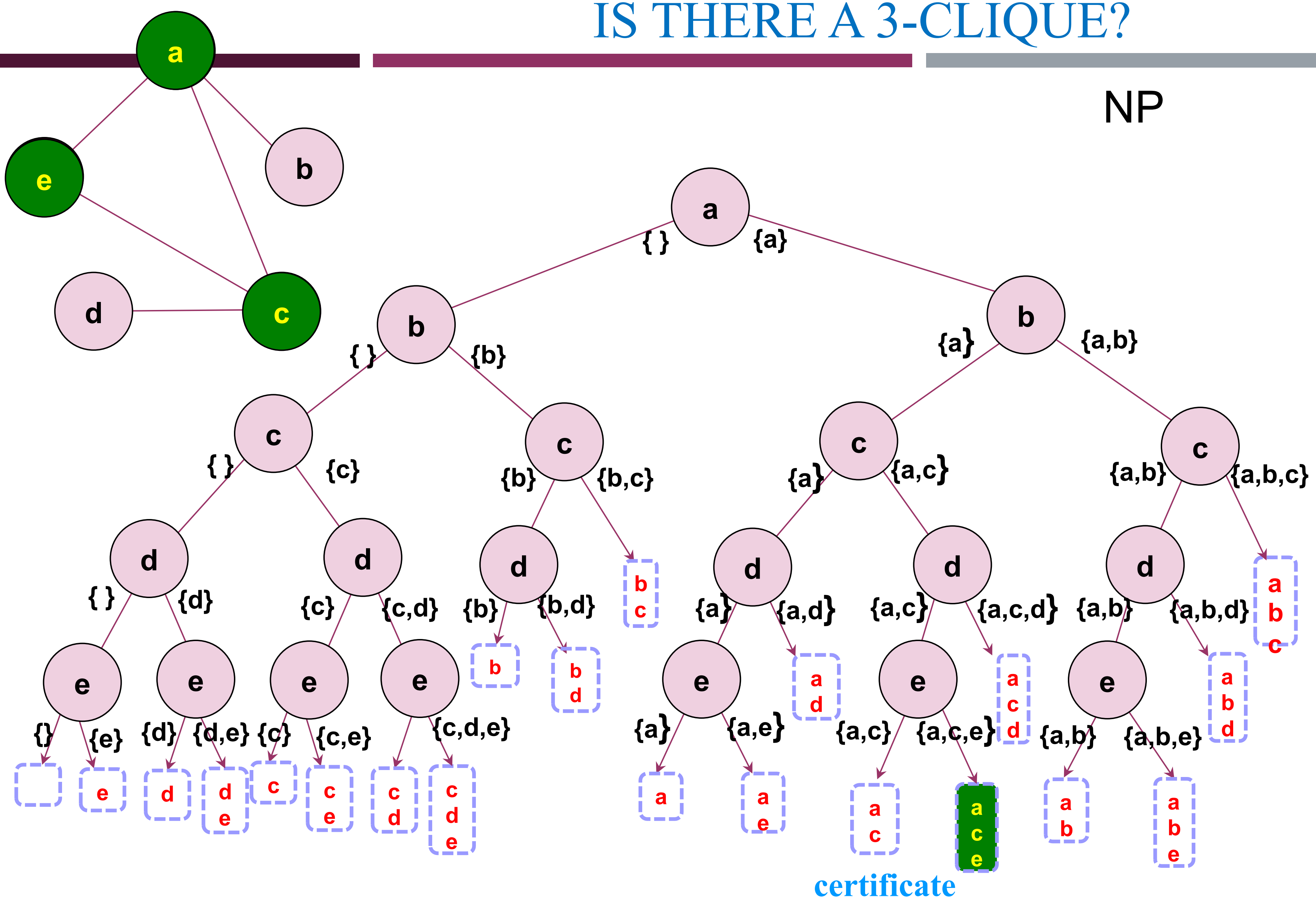
NP $\equiv$ **The set of problems their *positive* answers can be verified in polynomial time (i.e., having polynomial time computable certificates)**

BOOLEAN FORMULA: F(A,B,C) IS SATISFIABLE



IS THERE A 3-CLIQUE?

NP

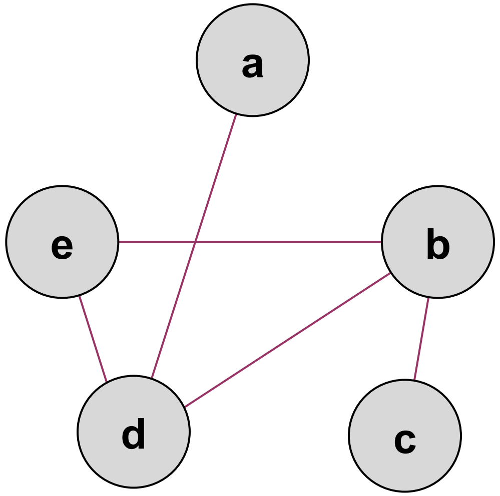


Another definition of NP

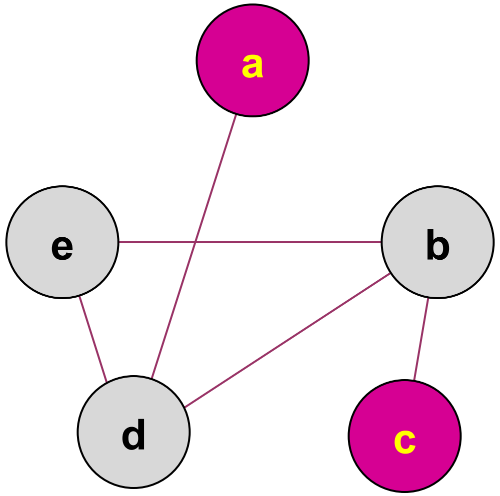
NP $\equiv$ **The set of problems their position answers can be verified in polynomial time (i.e., having polynomial time computable certificates)**

1. Binary Search, O(log n), **(supper fast)**
2. Sequential Search, O(n), **(easy and fast)**
3. Sorting, O(n log n), **(good)**
4. Shortest Path $O(n^2)$, **(smart)**
5. Minimum spanning tree, $O(n^2)$, **(reasonably easy and fast)**
6. Maximum flow (Minimum cut) $O(n^k)$, **(not so bad to have a k)**
7. Eulerian Path $O(n^k)$, **(not so bad to have a reasonable k)**
8. Hamiltonian path **(not good, stuck with $O(e^n)$)**
9. Maximum cut (Minimum flow) **(stuck with $O(e^n)$ again)**
10. Clique **(stuck with $O(e^n)$ again)**
11. Vertex cover **(stuck with $O(e^n)$ again)**
12. Subset Sum **(whoop, it's not as easy as it looks, $O(e^n)$)**
13. SAT **(bad $O(e^n)$)**

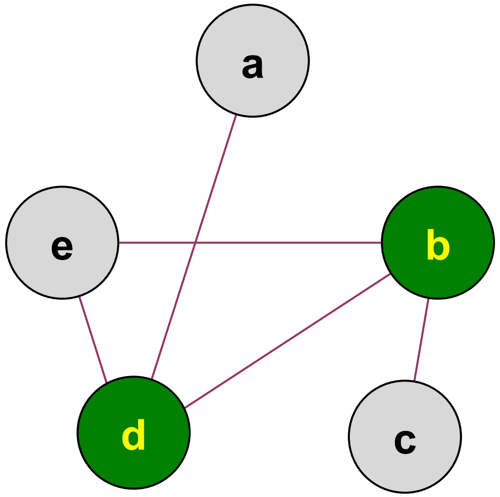
What are the positive certificates for these problem? In P?



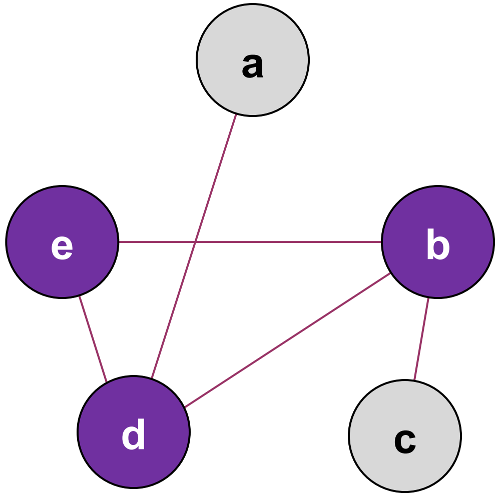
Is there a 2 independent set?



Is there a 2-vertex-cover?



Is there a 3-clique?



Definition of NPC: We say that problem A is **NP-Complete** (**NPC**) if

1. **A is NP, and**
2. **every NP problem can be *reduced* to A in polynomial time (i.e. ≤p).**

Cook's theorem (1971):
SAT is an NP-Complete Problem.

Theorem 1:
If A is NP and SAT $\leq_p$ A, then A is NPC.

Theorem 2:
If A is NP, B is NPC and B $\leq_p$ A, then A is NPC.

REDUCTIONS

- Real world examples:
  - Finding your way around the city reduces to reading a map
  - Traveling from Odisha to WB reduces to driving a car
  - Other suggestions…

POLYNOMIAL TIME REDUCTIONS

- PARTITION = { $n_1, n_2, \ldots n_k$ | we can split the integers into two sets which sum to half }
- SUBSET-SUM = { <$n_1, n_2, \ldots n_k, m$> | there exists a subset which sums to m }

1) If I can solve SUBSET-SUM, how can I use that to solve an instance of PARTITION?
2) If I can solve PARTITION, how can I use that to solve an instance of SUBSET-SUM?

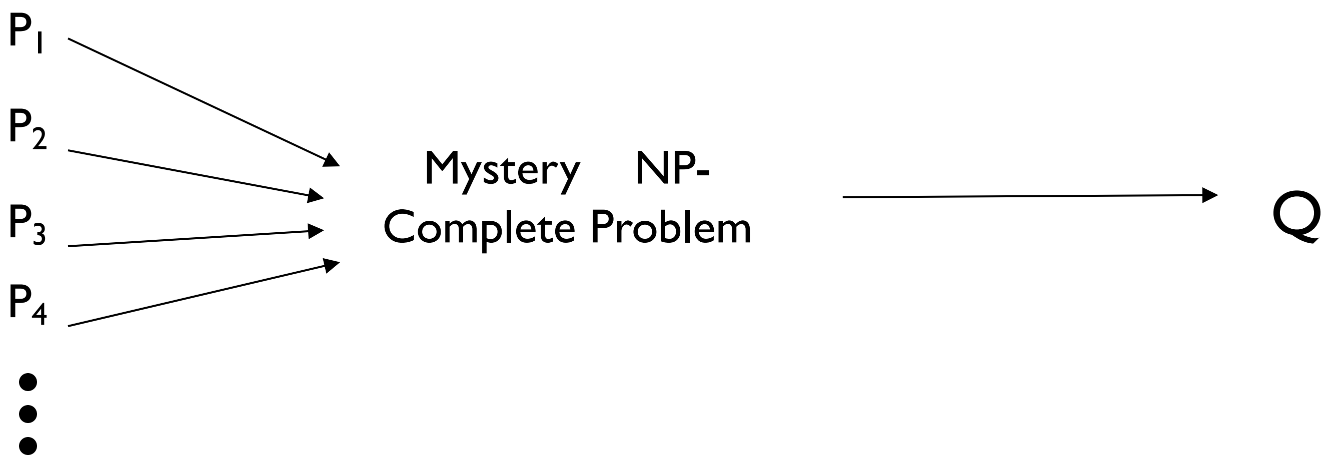
POLYNOMIAL REDUCTIONS

- 1) Partition REDUCES to Subset-Sum
  - Partition $<_P$ Subset-Sum
- 2) Subset-Sum REDUCES to Partition
  - Subset-Sum $<_P$ Partition

Therefore they are equivalently hard

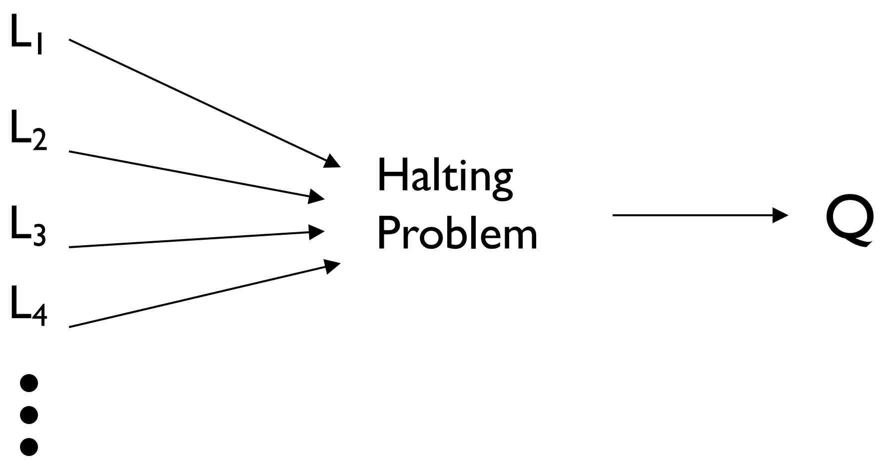
GETTING STARTED

- How do you show that EVERY NP problem reduces to Q?
- One way would be to already have an NP-Complete problem and just reduce from that



REMINDER: UNDECIDABILITY

- How do you show a language is undecidable?
- One way would be to already have an undecidable problem and just reduce from that



SAT

- SAT = { f | f is a Boolean Formula with a satisfying assignment }

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

- Is SAT in NP?

3-SAT

- 3-SAT = { f | f is in Conjunctive Normal Form, each clause has exactly 3 literals and f is satisfiable }

- CNF if it is a conjunction of one or more than one clause, where each clause is a disjunction of literals.

- It is a product of sums where ∧ symbol occurs between the clauses and the ∨ symbol occurs in the clauses.

- ❑ 3-SAT is NP-Complete
- ❑ (2-SAT is in P)

NP-COMPLETE

- To prove a problem is NP-Complete show a polynomial time reduction from 3-SAT
- Other NP-Complete Problems:
  - PARTITION
  - SUBSET-SUM
  - CLIQUE
  - HAMILTONIAN PATH (TSP)
  - GRAPH COLORING

(and many more)

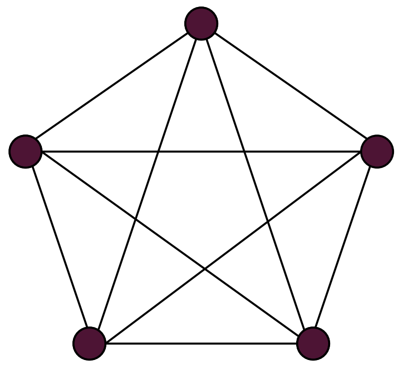
NP-COMPLETENESS PROOF METHOD

To show that Q is NP-Complete:

- 1) Show that Q is in NP
- 2) Pick an instance, R, of your favorite NP-Complete problem (ex: Φ in 3-SAT)
- 3) Show a polynomial algorithm to transform R into an instance of Q

EXAMPLE: CLIQUE

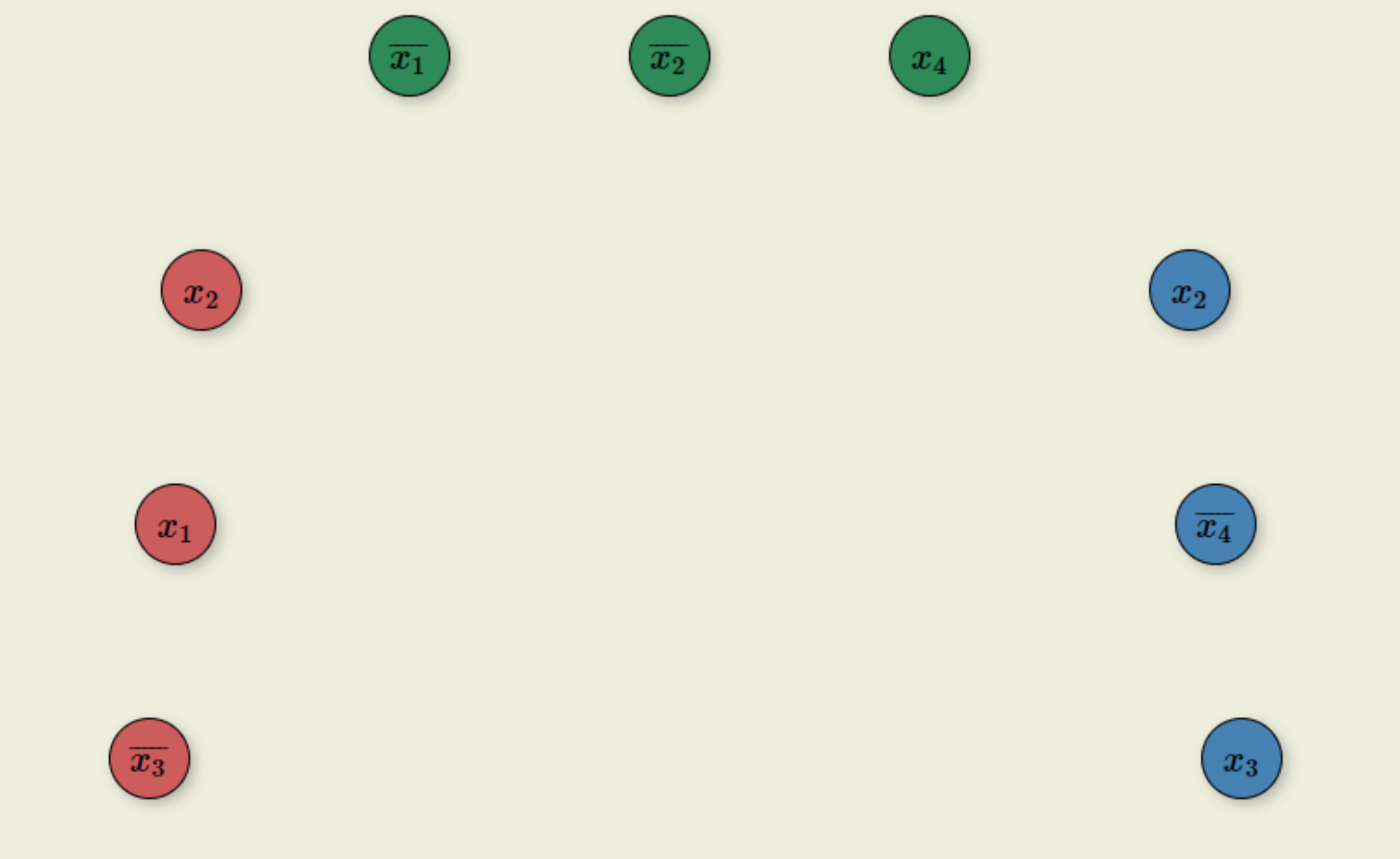
- CLIQUE = { <G,k> | G is a graph with a clique of size k }
- A clique is a subset of vertices that are all connected
- Why is CLIQUE in NP?



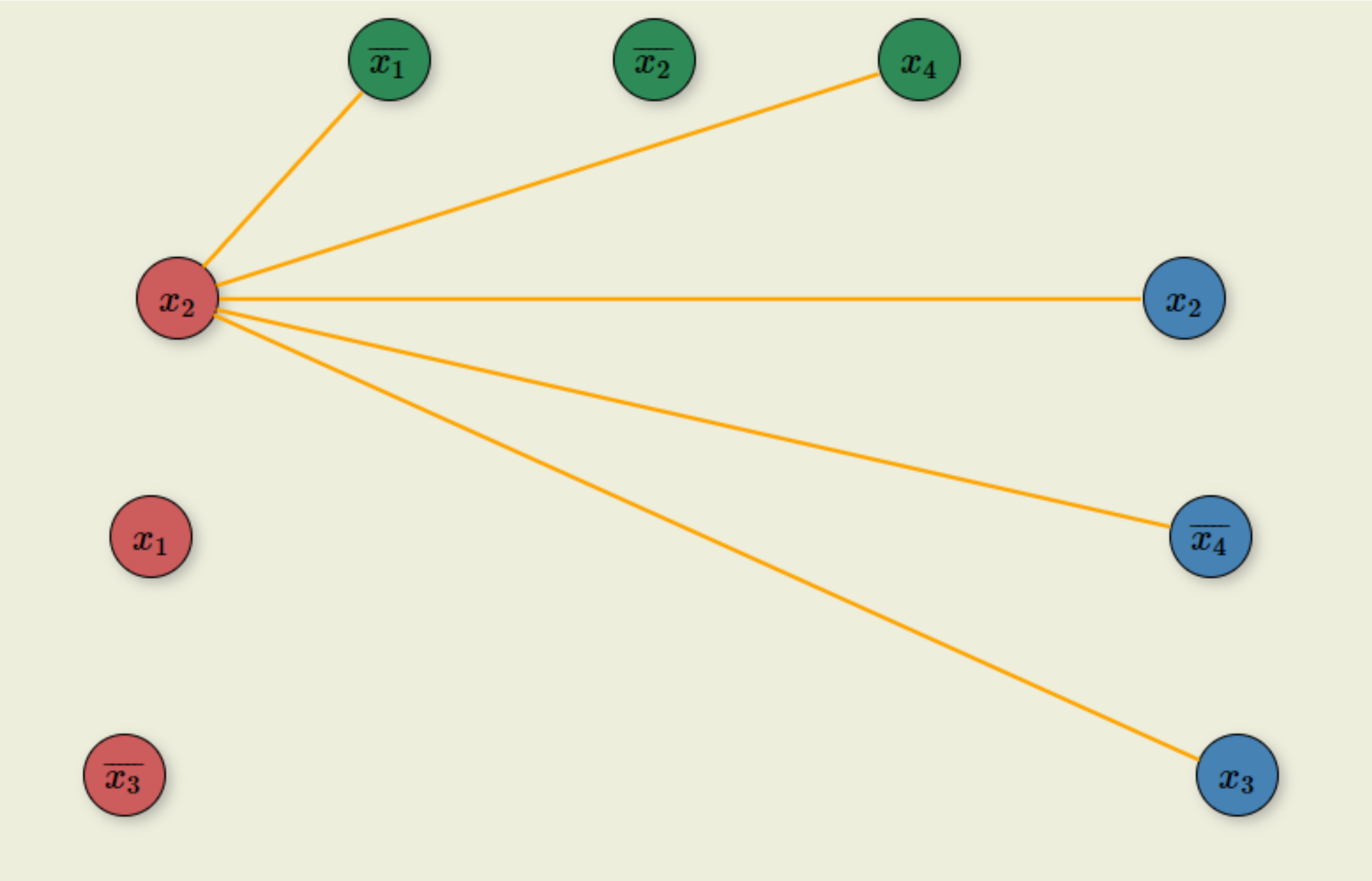
REDUCE 3-SAT TO CLIQUE

- Pick an instance of 3-SAT, Φ, with *k* clauses
- Make a vertex for each literal
- Connect each vertex to the literals in other clauses that are not the negation
- Any k-clique in this graph corresponds to a satisfying assignment

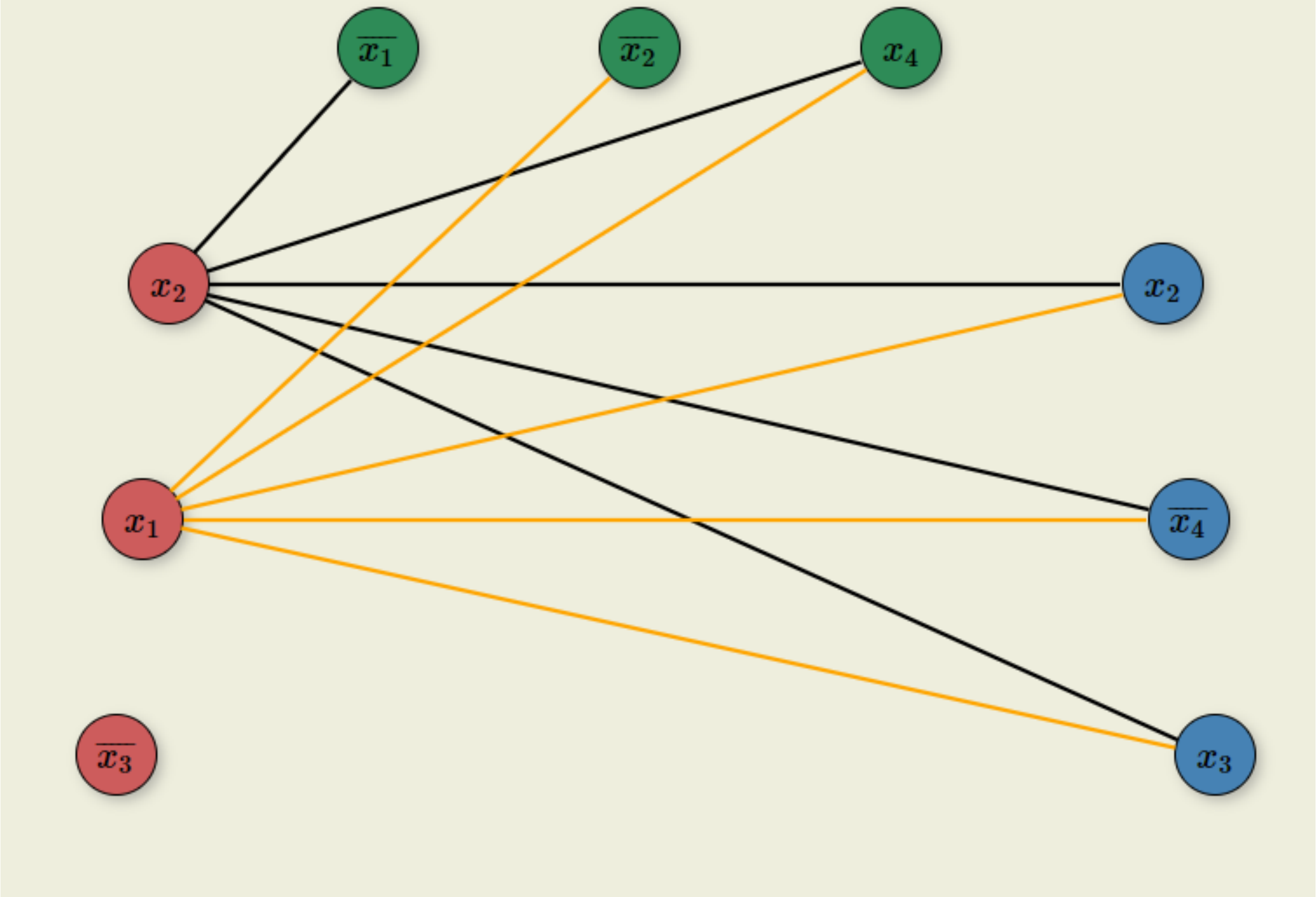
$$\Phi = (x_2 + x_1 + \overline{x_3}) \,.\, (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



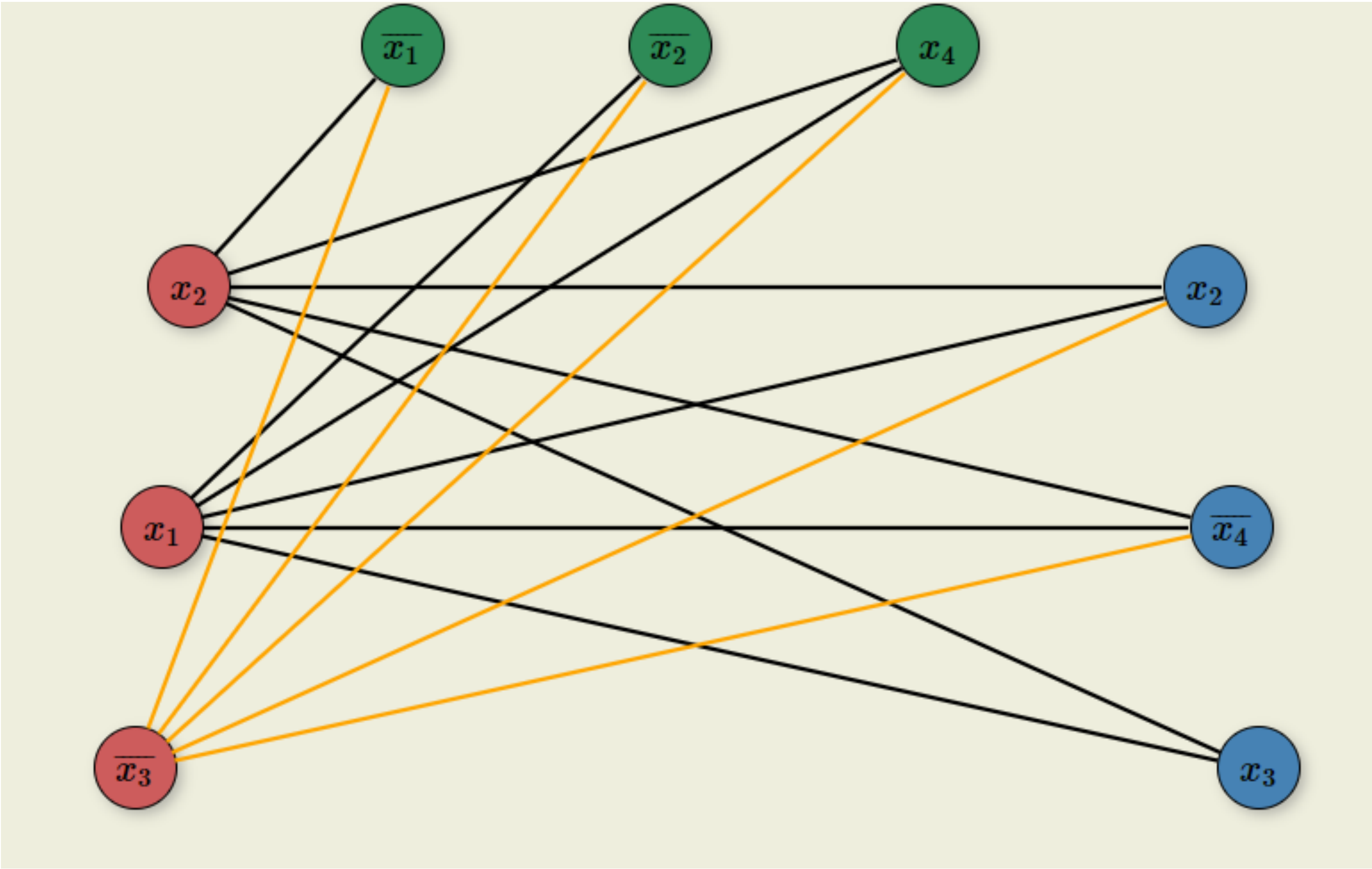
$$\Phi = (x_2 + x_1 + \overline{x_3}) \; . (\overline{x_1} + \overline{x_2} + x_4) \quad \cdot (x_2 + \overline{x_4} + x_3)$$



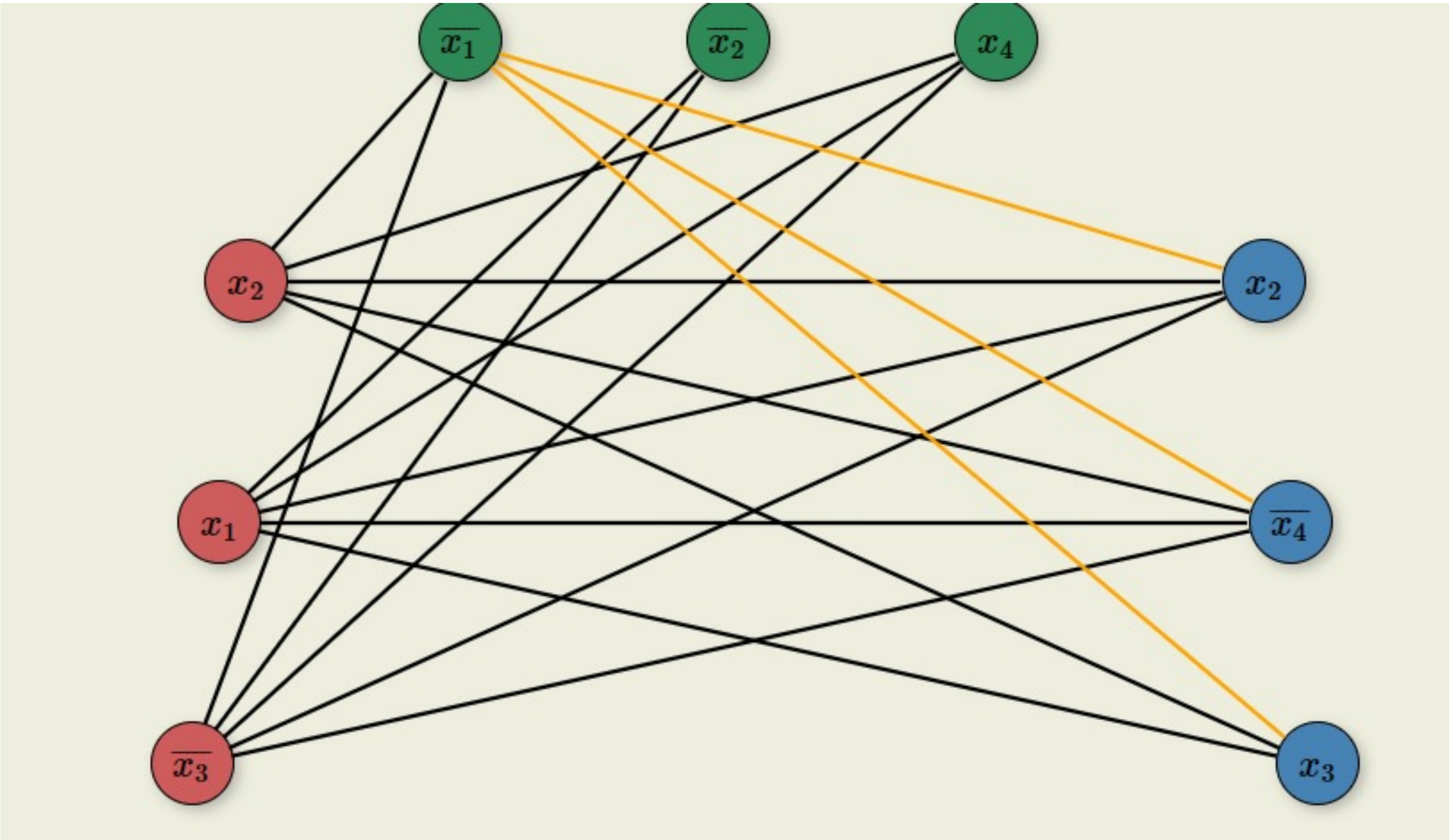
$$\Phi = (x_2 + x_1 + \overline{x_3}) \,.\, (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



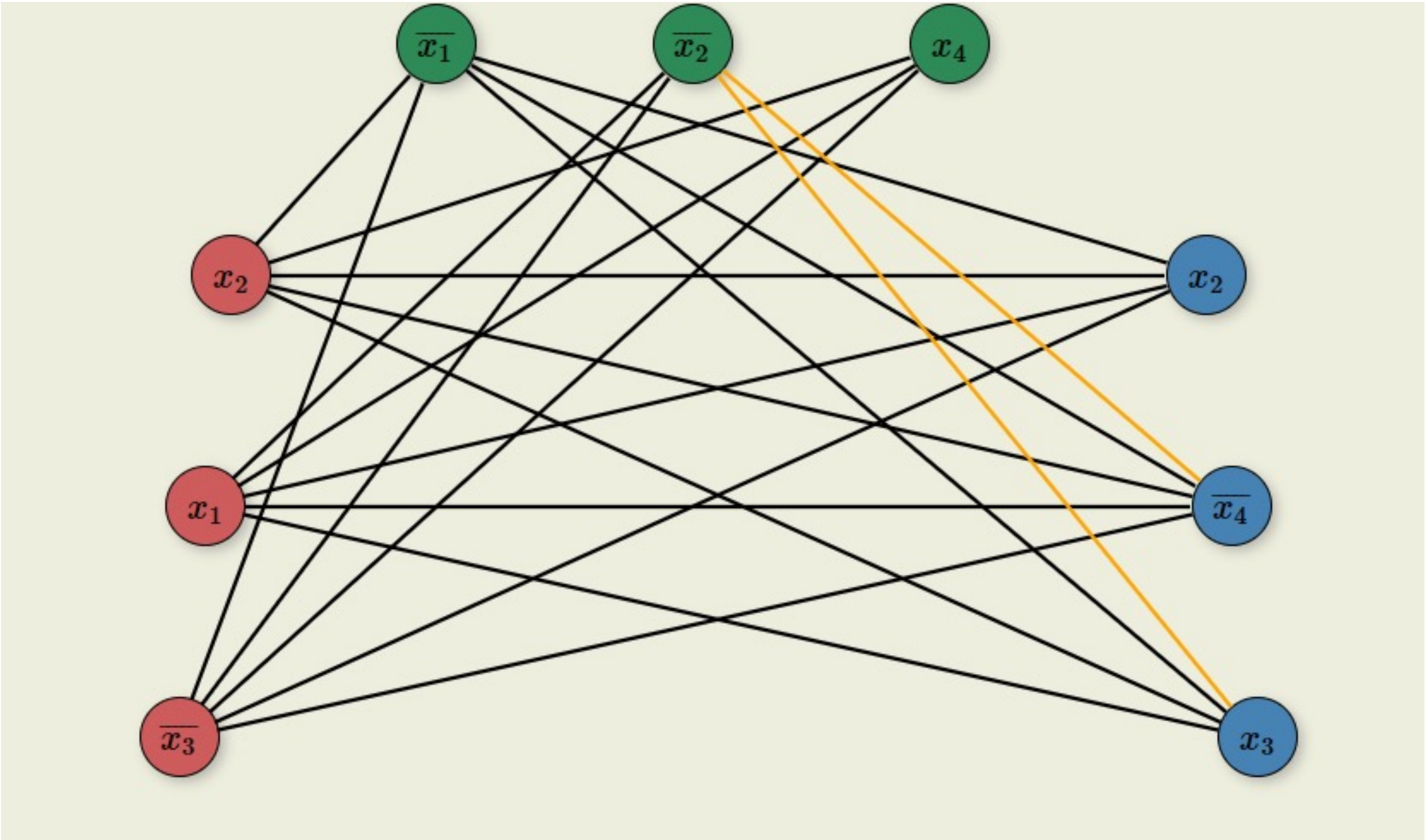
$$\Phi = (x_2 + x_1 + \overline{x_3}) \,.\, (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



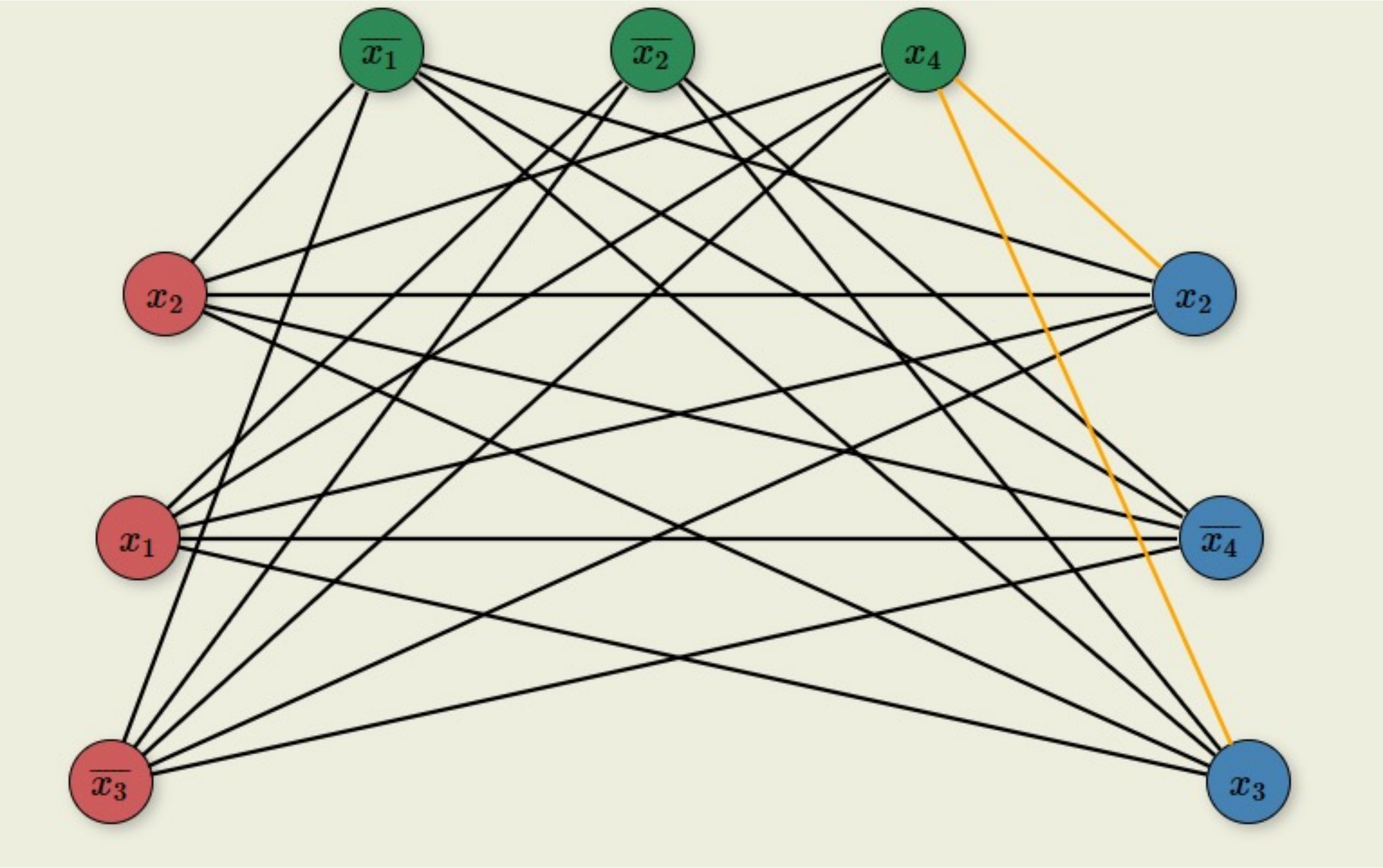
$$\Phi = (x_2 + x_1 + \overline{x_3}) \,.\, (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



$$\Phi = (x_2 + x_1 + \overline{x_3}) \,.\, (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



$$\Phi = (x_2 + x_1 + \overline{x_3}) \, . (\overline{x_1} + \overline{x_2} + x_4) \cdot (x_2 + \overline{x_4} + x_3)$$



INSIGHTS FROM THE GRAPH?

- If two nodes are connected then the corresponding literals we can simultaneously assign TRUE

- If two literals, not from the same clause can be assigned True simultaneously, the nodes corresponding to these literals in the graph are Connected.

- Construction of the graph: P time

G HAS K-CLIQUE IFF Φ IS SATISFIABLE

HOW?

- If the graph has a K-clique, then the clique has exactly one node from each cluster. Why?

No two nodes in the same cluster are connected to each other, hence they can never be part of the same clique!

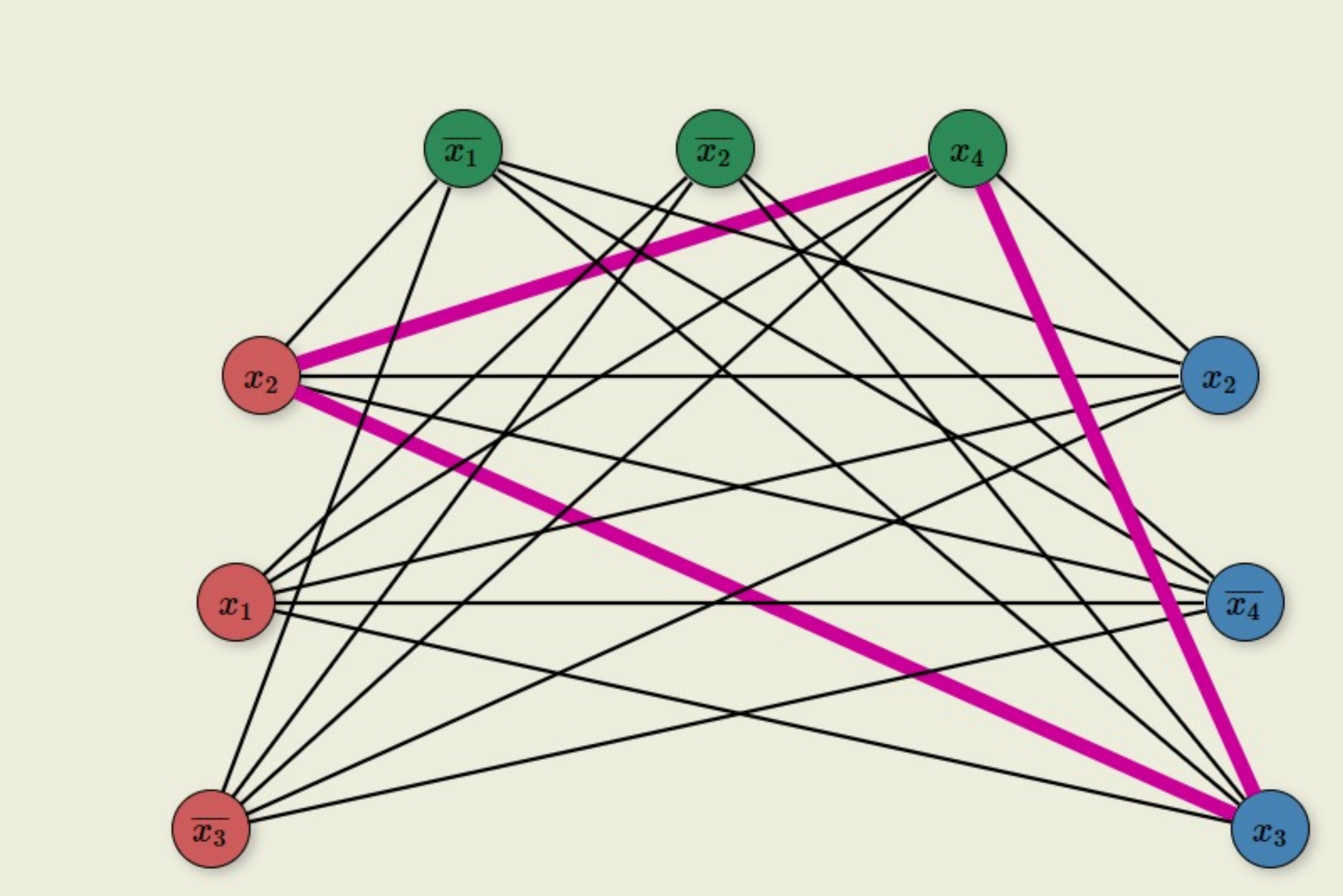
HOW?

- If the graph has a K-clique, then the clique has exactly one node from each cluster. Why?
- All nodes in a clique are connected, hence all corresponding literals can be assigned TRUE simultaneously
- Each literal belongs to exactly one of the K clauses, hence Φ is satisfiable

IF Φ IS SATISFIABLE (Let A be a satisfying assignment)

- Select from each clause a literal that is TRUE in A to construct a set S
- $||S||=K$
- As no two literals in A are from the same clause and all of them are simultaneously TRUE, all the corresponding nodes in the graph are connected to each other, forming a K-clique
- So, the graph has a k-clique!

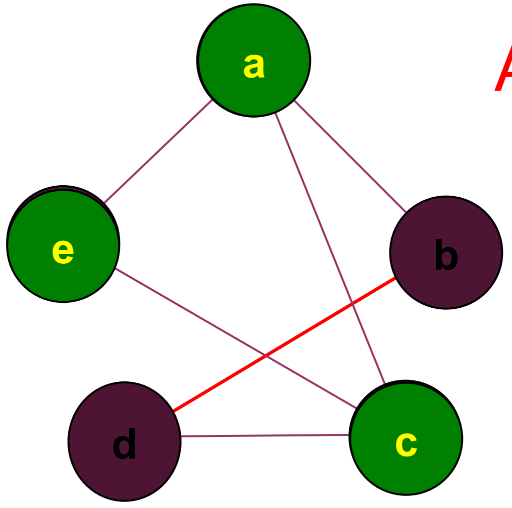
EXAMPLE OF 3-SAT TO K-CLIQUE REDUCTION



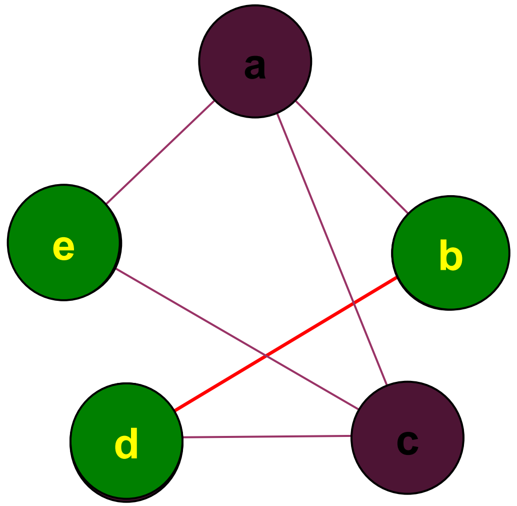
The corresponding assignment: $x_2 = True, x_3 = True, x_4 = True$

IS THERE A 3-CLIQUE?

Is there a 2-independent set?



A wrong reduction



$\equiv_p$

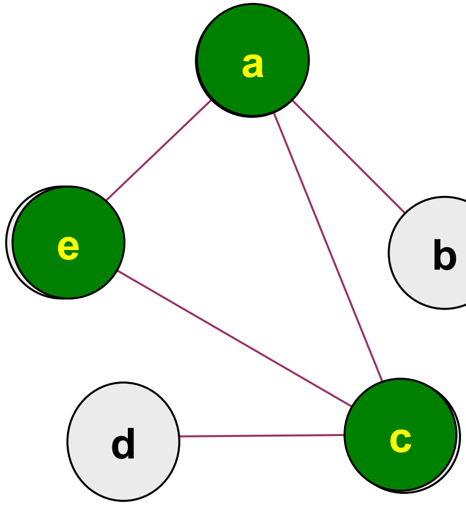
Not always working (why)

IS THERE A 3-CLIQUE?

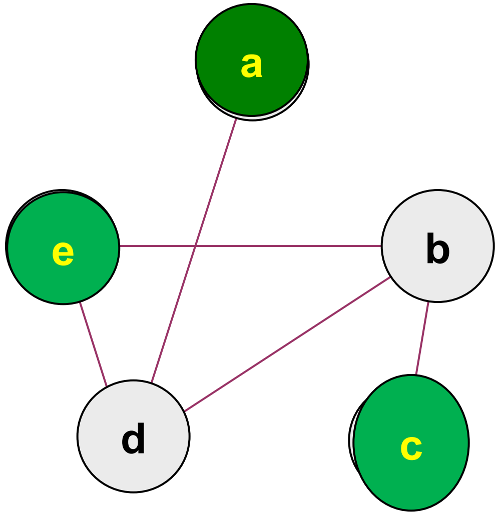
Is there a 3-independent set?

complement graph

Why so?



$\equiv_p$



← complement graph →
can be constructed in P

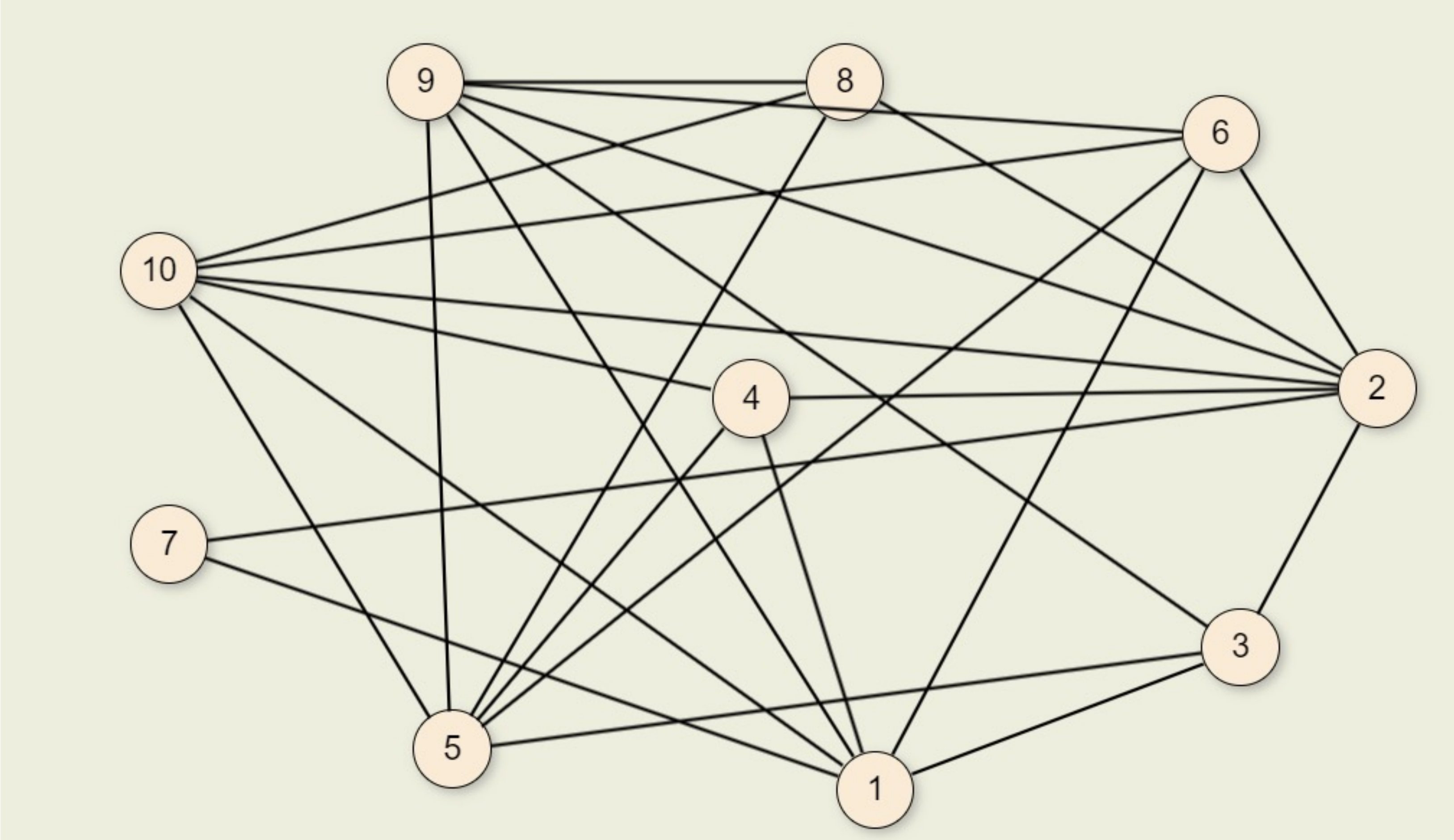
Clique problem and Independent Sets are (complexity) equivalent problem.

WHY THIS WORKS? (CLIQUE PROBLEM REDUCED TO IS)

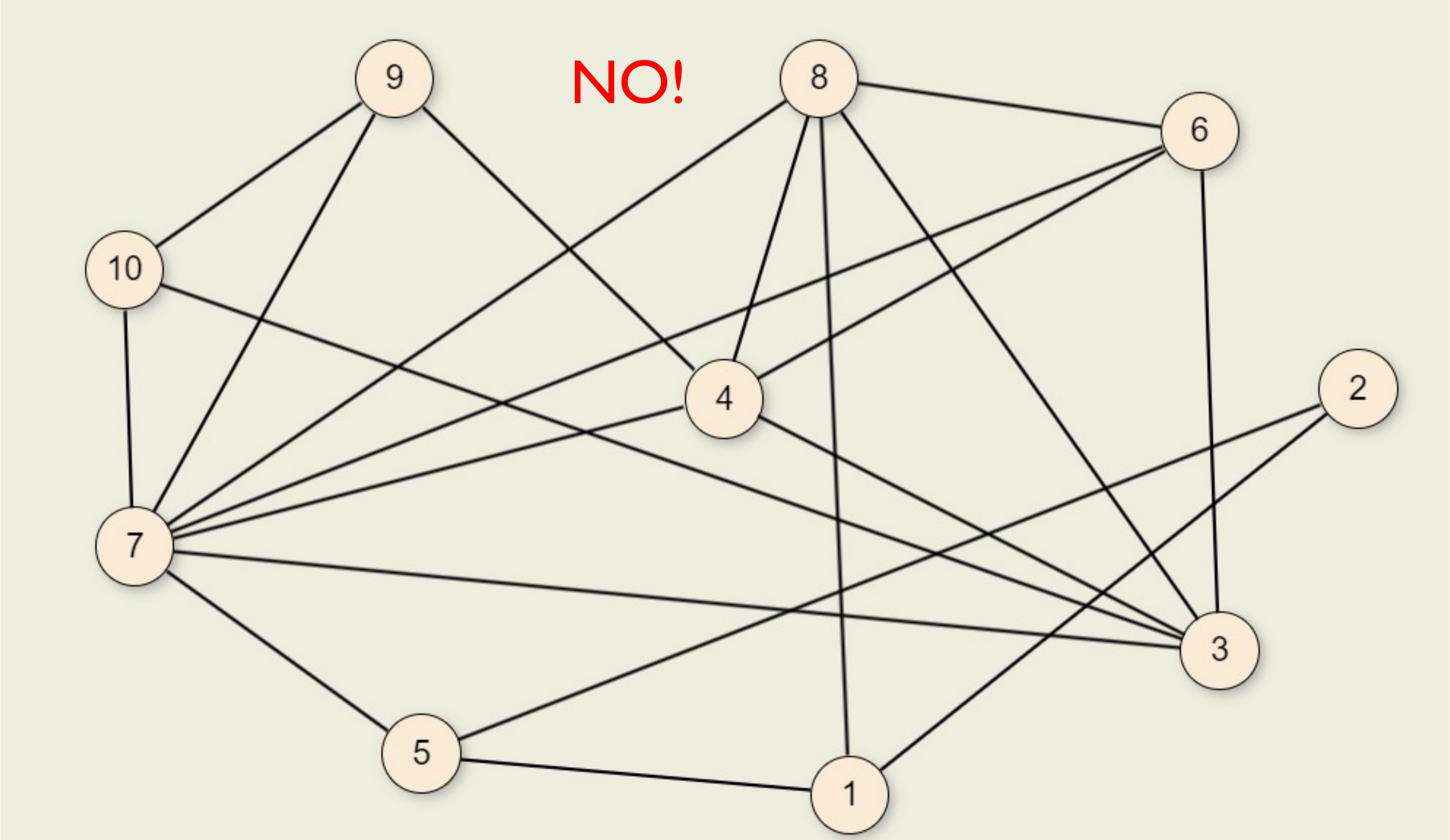
1. **If there is an independent set of size k in the complement graph** G', it implies no two vertices share an edge in G' which further implies all of those vertices share an edge with all others in G forming a clique. that is **there exists a clique of size k in** G

2. **If there is a clique of size** k **in the graph** G, it implies all vertices share an edge with all others in G which further implies no two of these vertices share an edge in G' forming an Independent Set. that is **there exists an independent set of size k in** G'

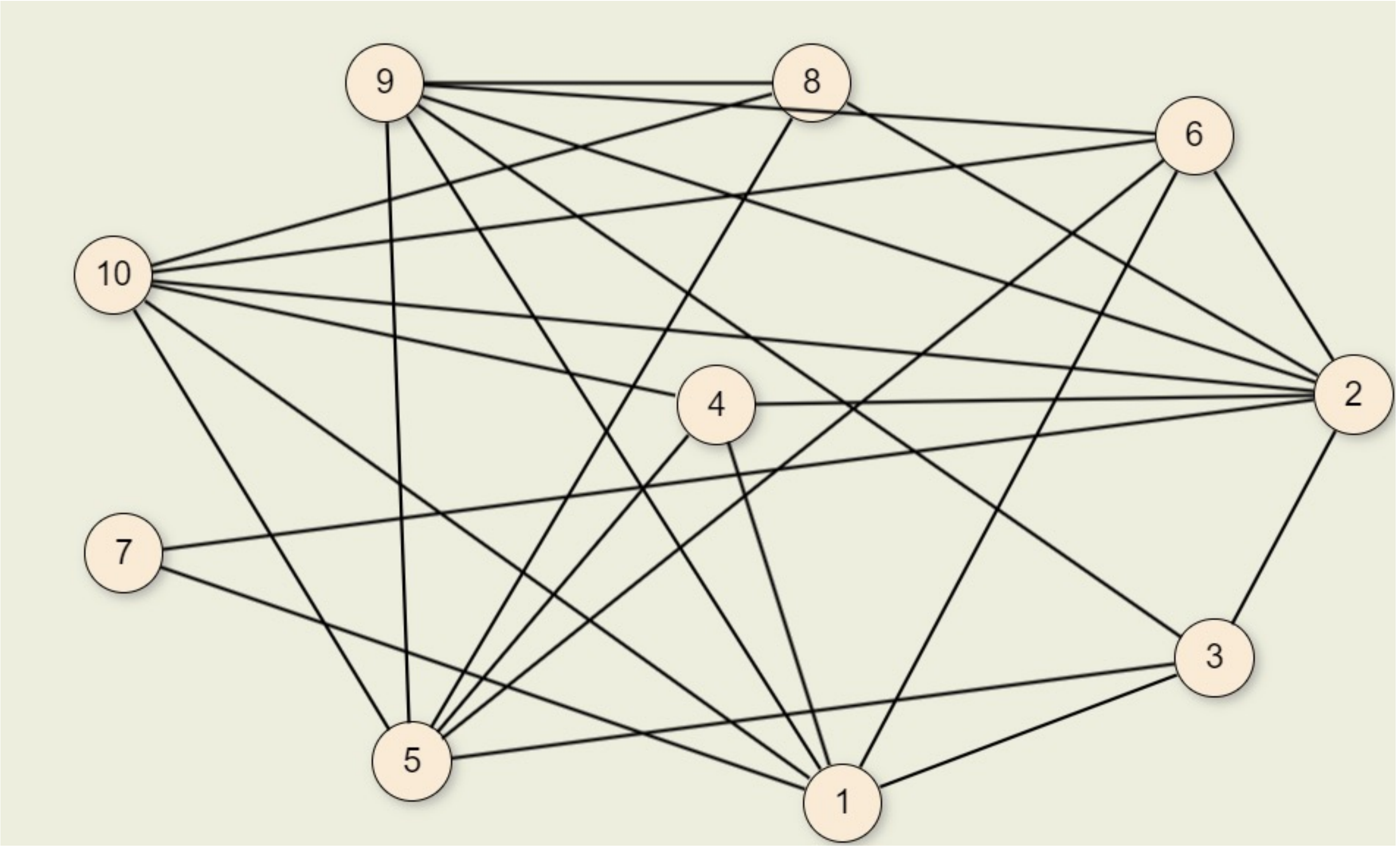
ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE GRAPH HAVE AN IS OF SIZE 8



ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE GRAPH HAVE CLIQUE OF SIZE 8

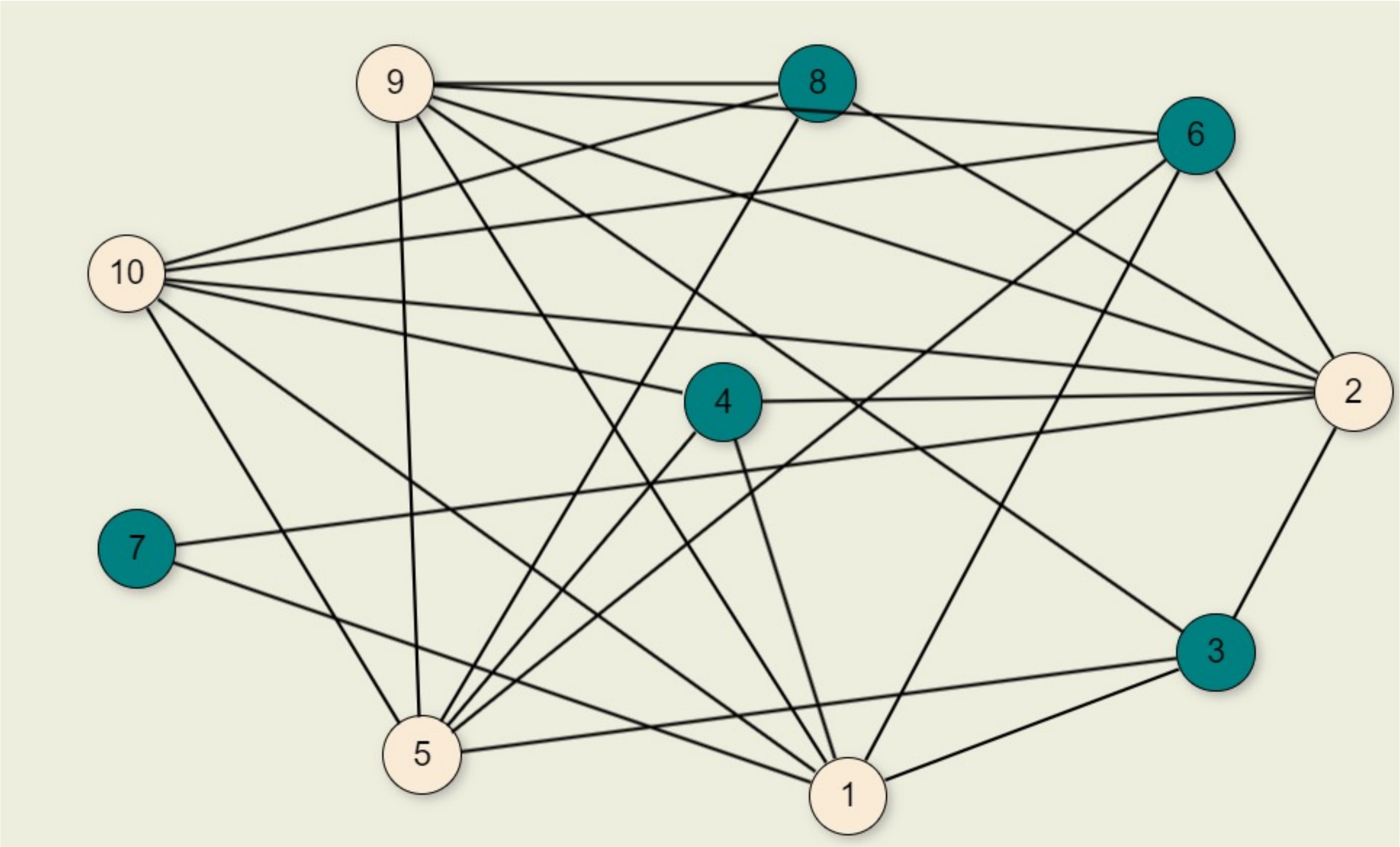


ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE GRAPH HAVE AN IS OF SIZE 5



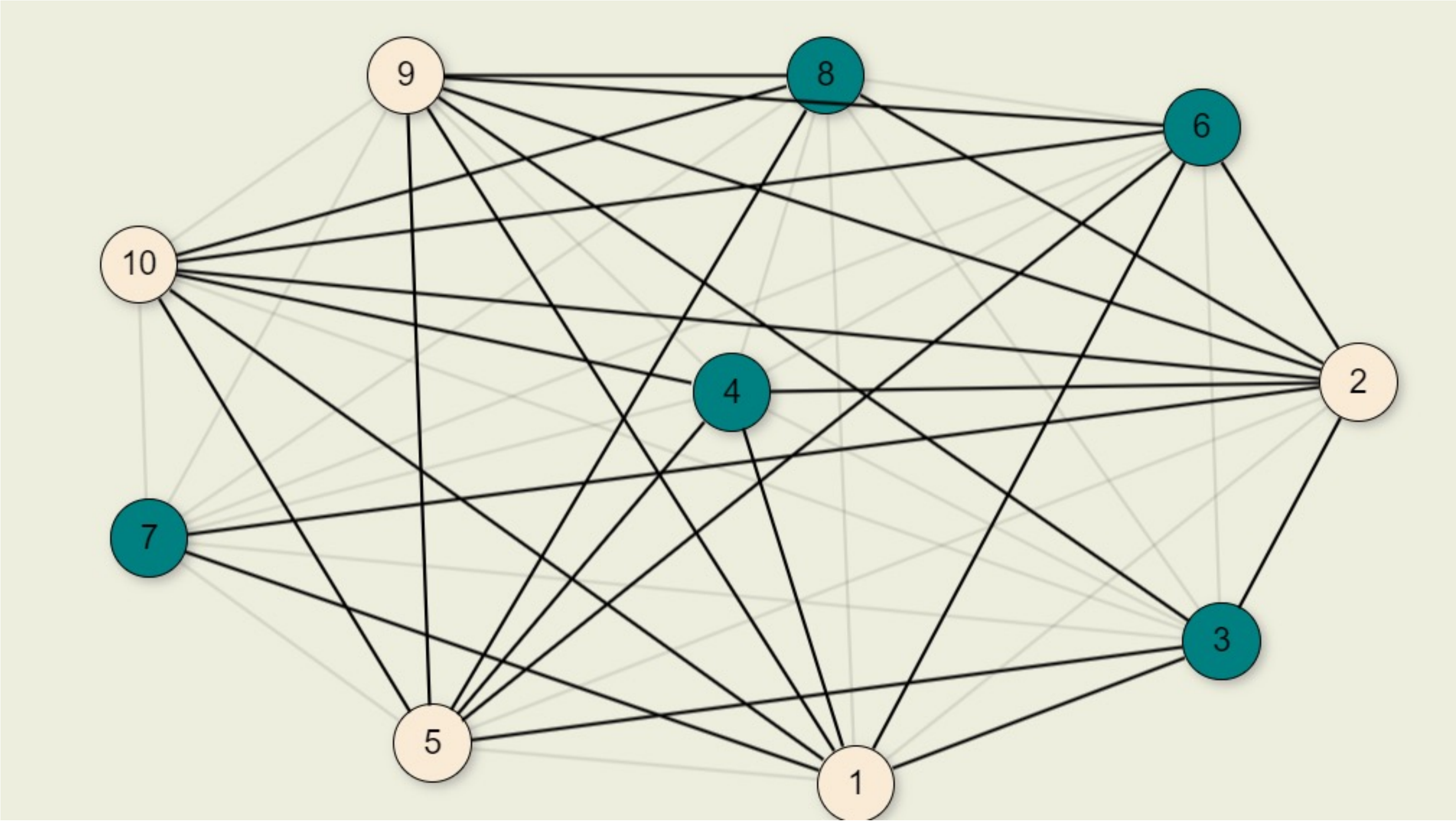
ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE GRAPH HAVE AN IS OF SIZE 5

YES!



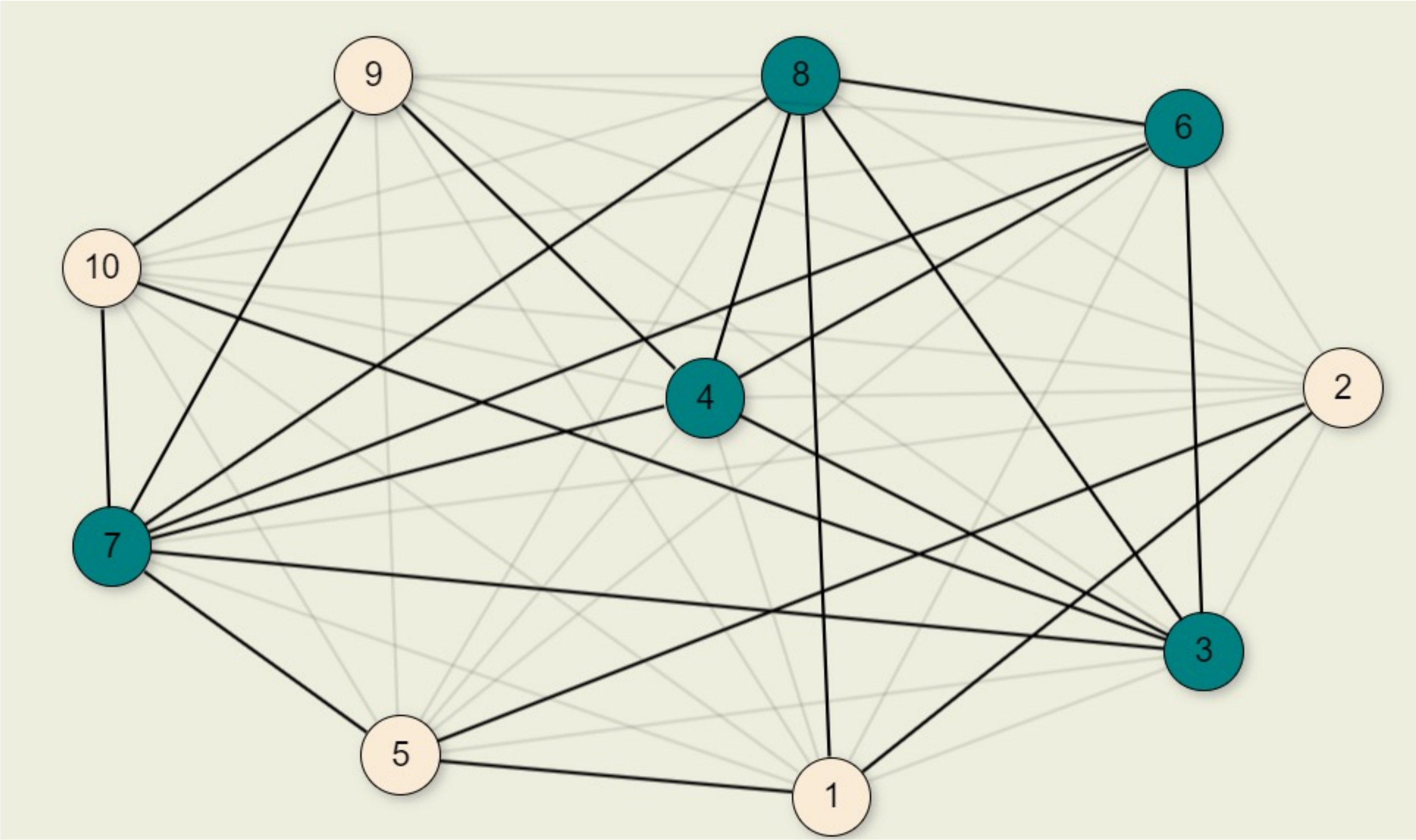
ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE COMPLEMENT GRAPH HAVE AN CLIQUE OF SIZE 5

YES!



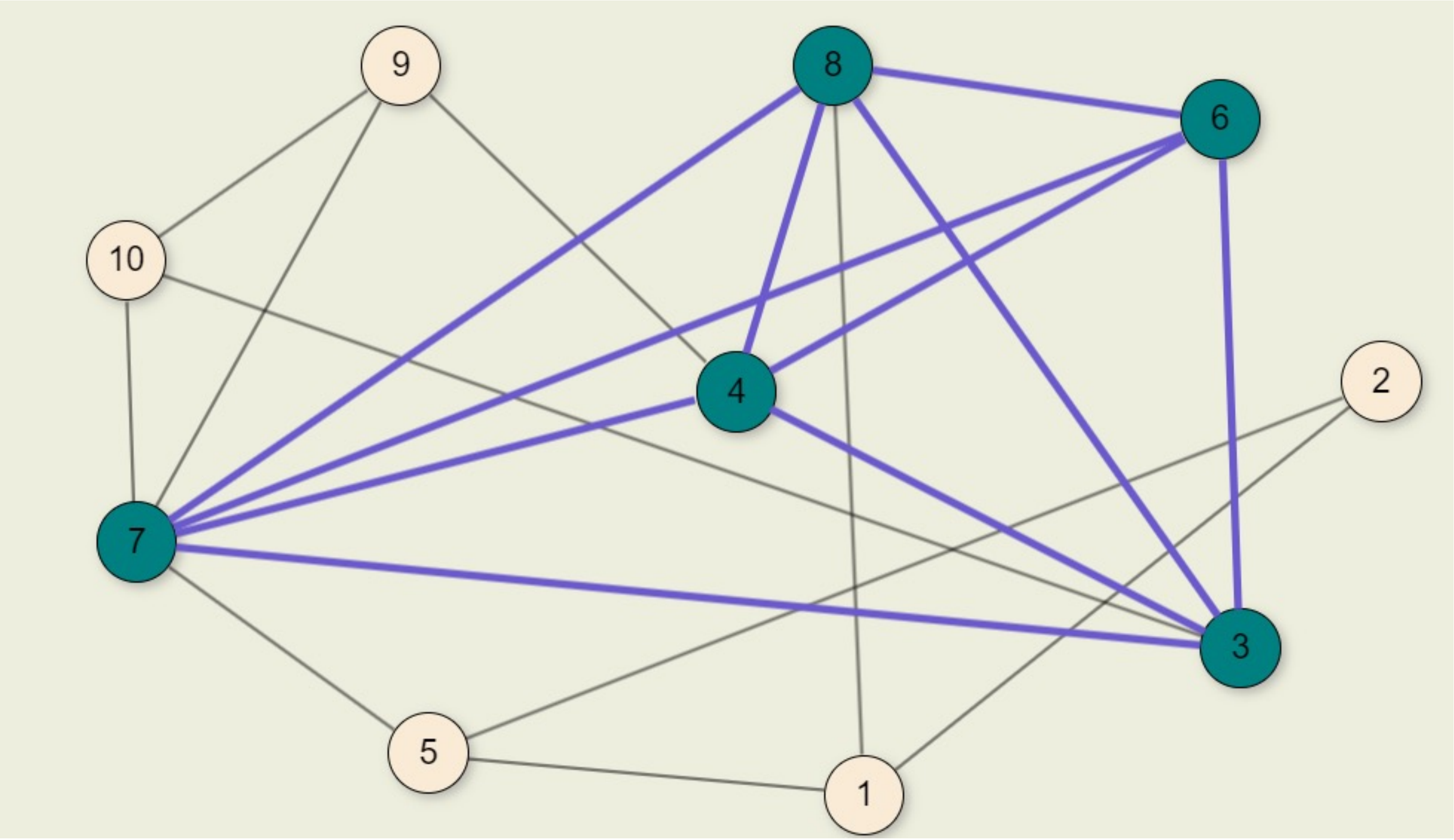
ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
DOES THE COMPLEMENT GRAPH HAVE AN CLIQUE OF SIZE 5

YES!



ANOTHER EXAMPLE (CLIQUE PROBLEM REDUCED TO IS)
IT FORMS A CLIQUE OF SIZE 5

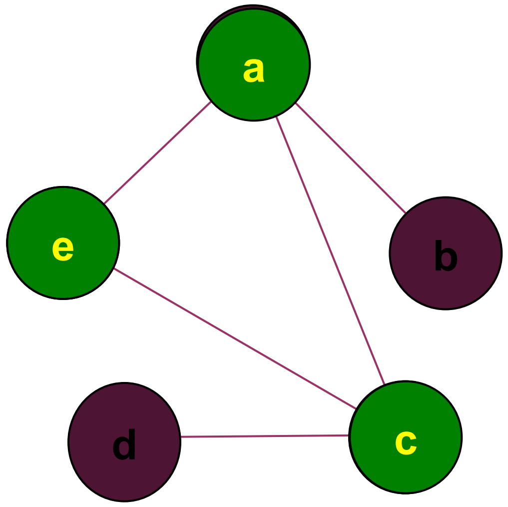
YES!



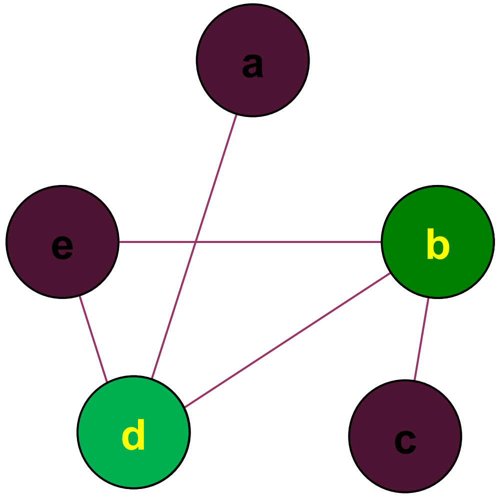
IS THERE A 3-CLIQUE? → Is there a 2-vertex-cover?

complement graph

Why so?



$\equiv_p$



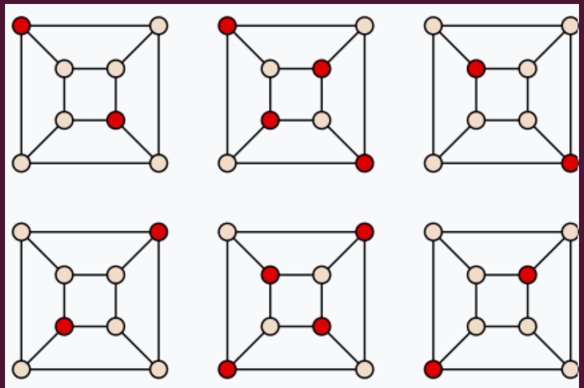
← complement graph →
can be constructed in P

Thus, Clique, Vertex Cover, and Independent Sets are (complexity) equivalent problem.

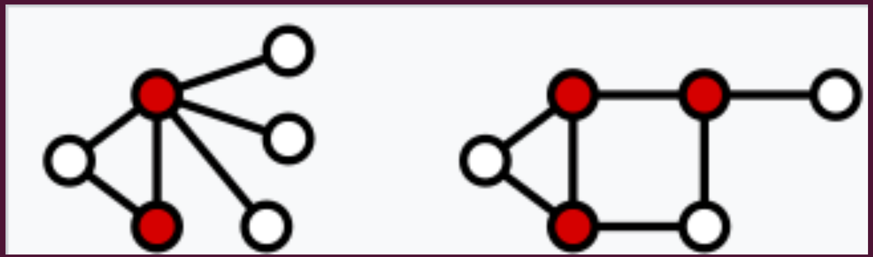
CLIQUE, VERTEX COVER, AND INDEPENDENT SETS ARE (COMPLEXITY) EQUIVALENT PROBLEM.

- If one of them is NPC, then all of them are NPC.
- If one of them can be solved in P, all of them can.
- If one of them proven can’t be solved in P, neither can the other two.

REDUCTION OF INDEPENDENT SET (IS) TO VERTEX COVER (VC)



Graph G = (V, E), a subset S of the vertices is independent if there are no edges between vertices in S *(Fig.: red colored)*



Graph G = (V, E), a subset S of the vertices is a vertex cover if every edge in E has at least one endpoint in S

REDUCTION OF IS TO VC

In a graph $G = \{V, E\}$,

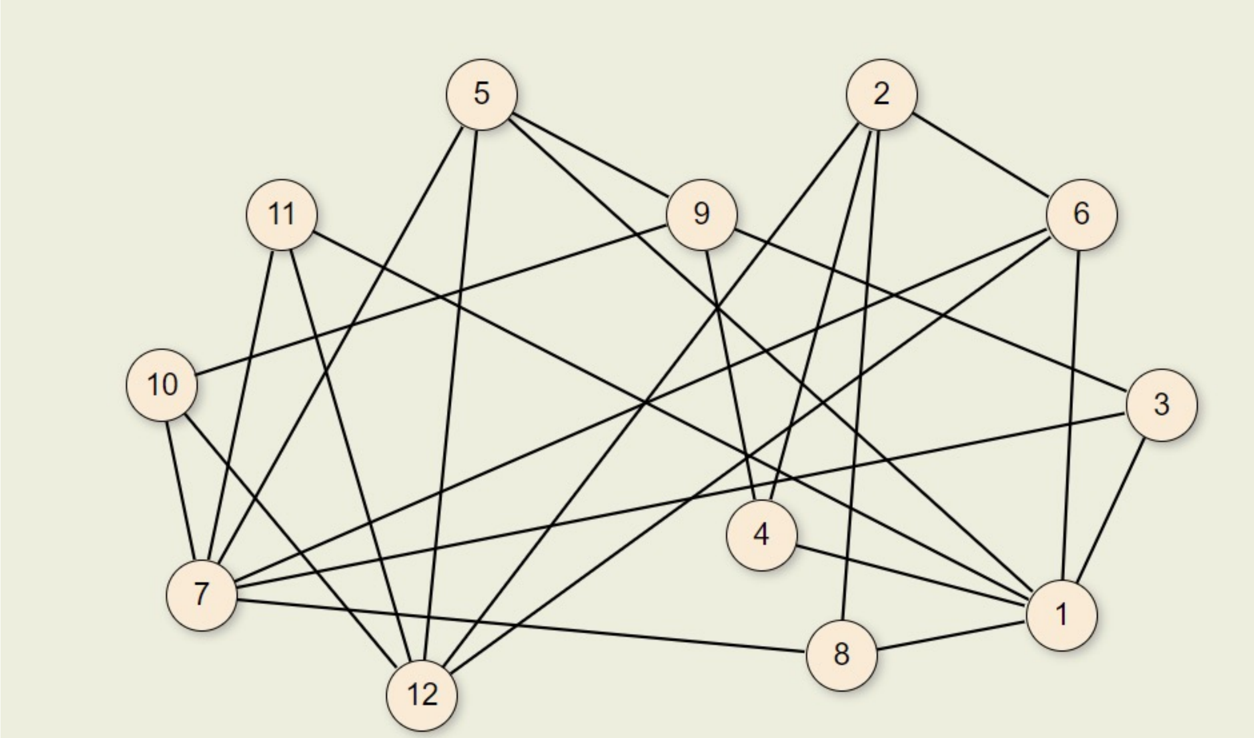
S is an **Independent Set** $\Leftrightarrow (V - S)$ **is a Vertex Cover.**

1. **If S is an Independent Set** ,there is no edge $e = (u, v)$ in G, such that both $u, v \in S$.
Hence for any edge $e = (u, v)$, atleast one of u, v must lie in $(V - S)$.
$\Rightarrow (V - S)$ **is a vertex cover in G**.

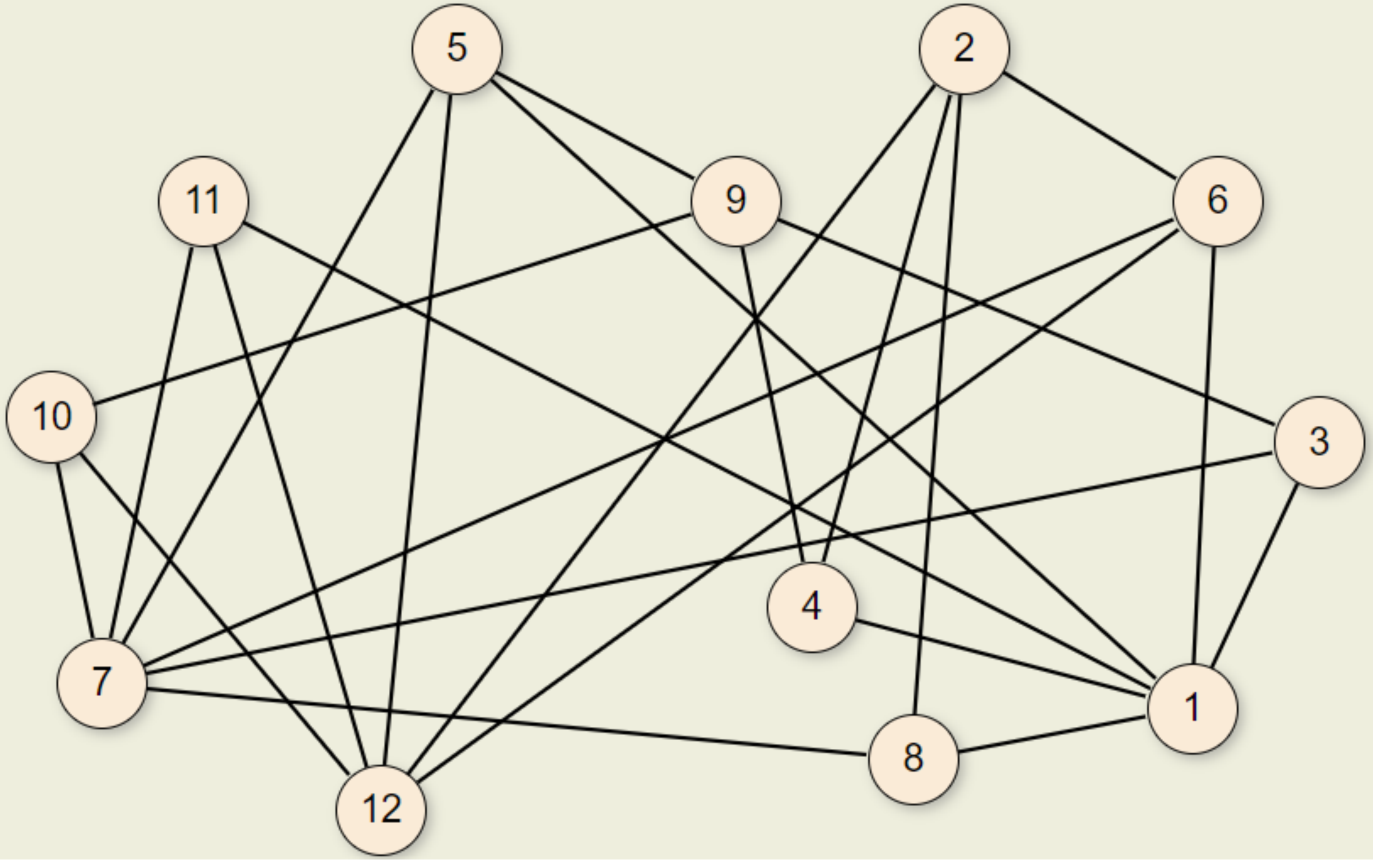
2. **If $(V - S)$ is a Vertex Cover,** between any pair of vertices $(u, v) \in S$ if there exist an edge e,
none of the endpoints of e would exist in $(V - S)$ violating the definition of vertex cover.
Hence no pair of vertices in S can be connectedby an edge.
$\Rightarrow S$ **is an Independent Set in G**.

Hence G contains an Independent Set of size k $\Leftrightarrow$ G contains a Vertex Cover of size $|V| - k$.

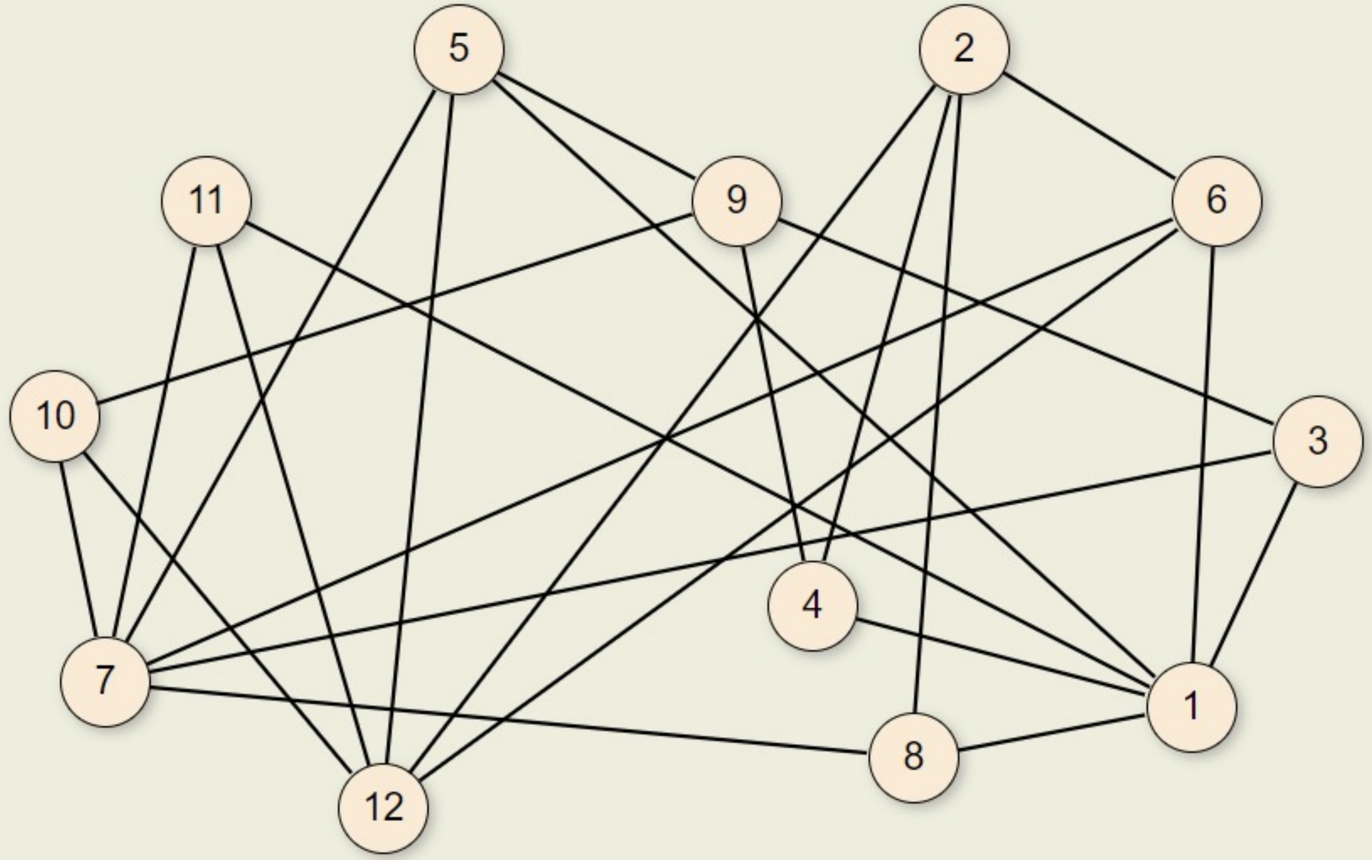
EXAMPLE GRAPH



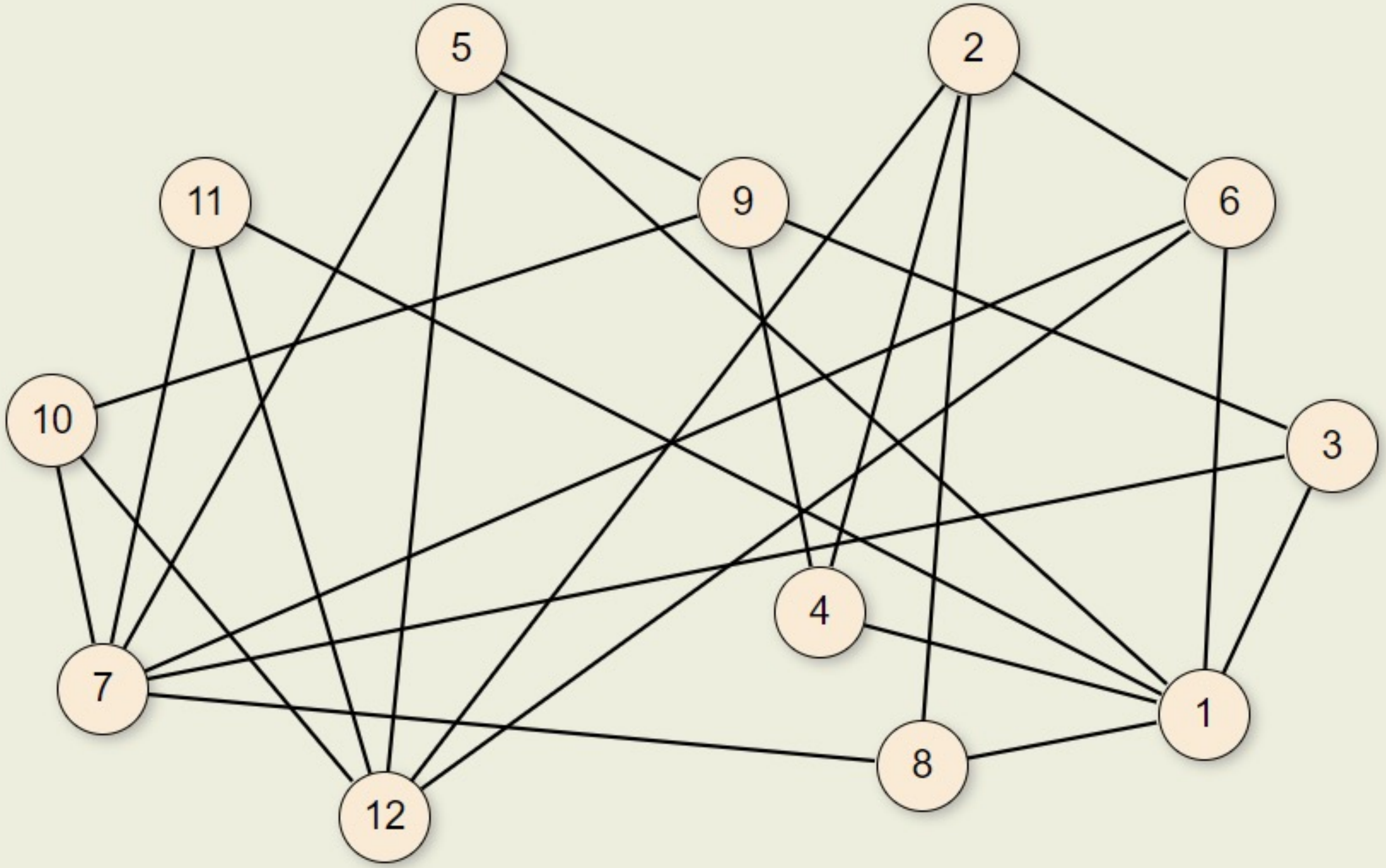
Does this 12-node graph have an Independent Set of size >= 9 ?

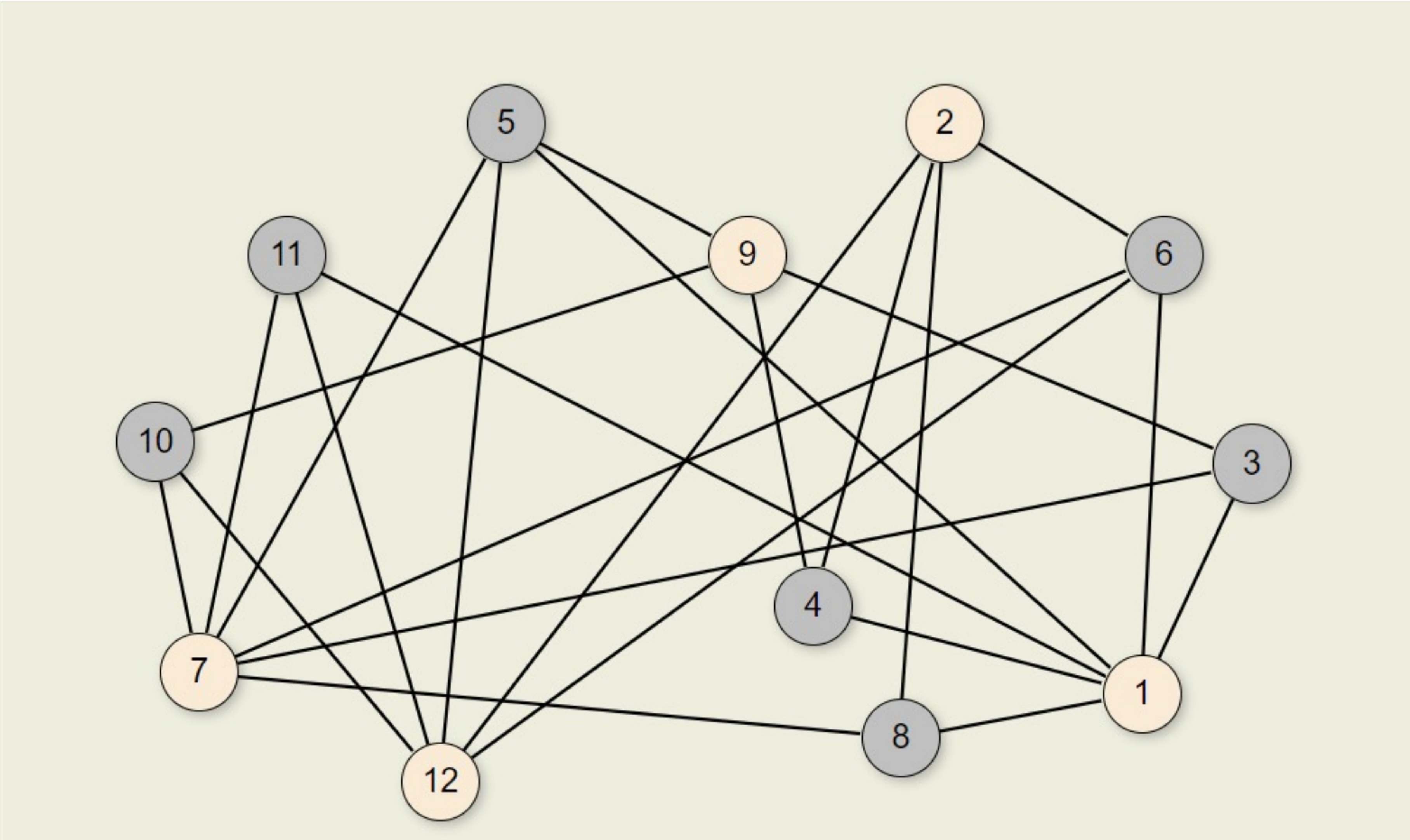


Does this 12-node graph have an Vertex Cover of size <= 3 ?

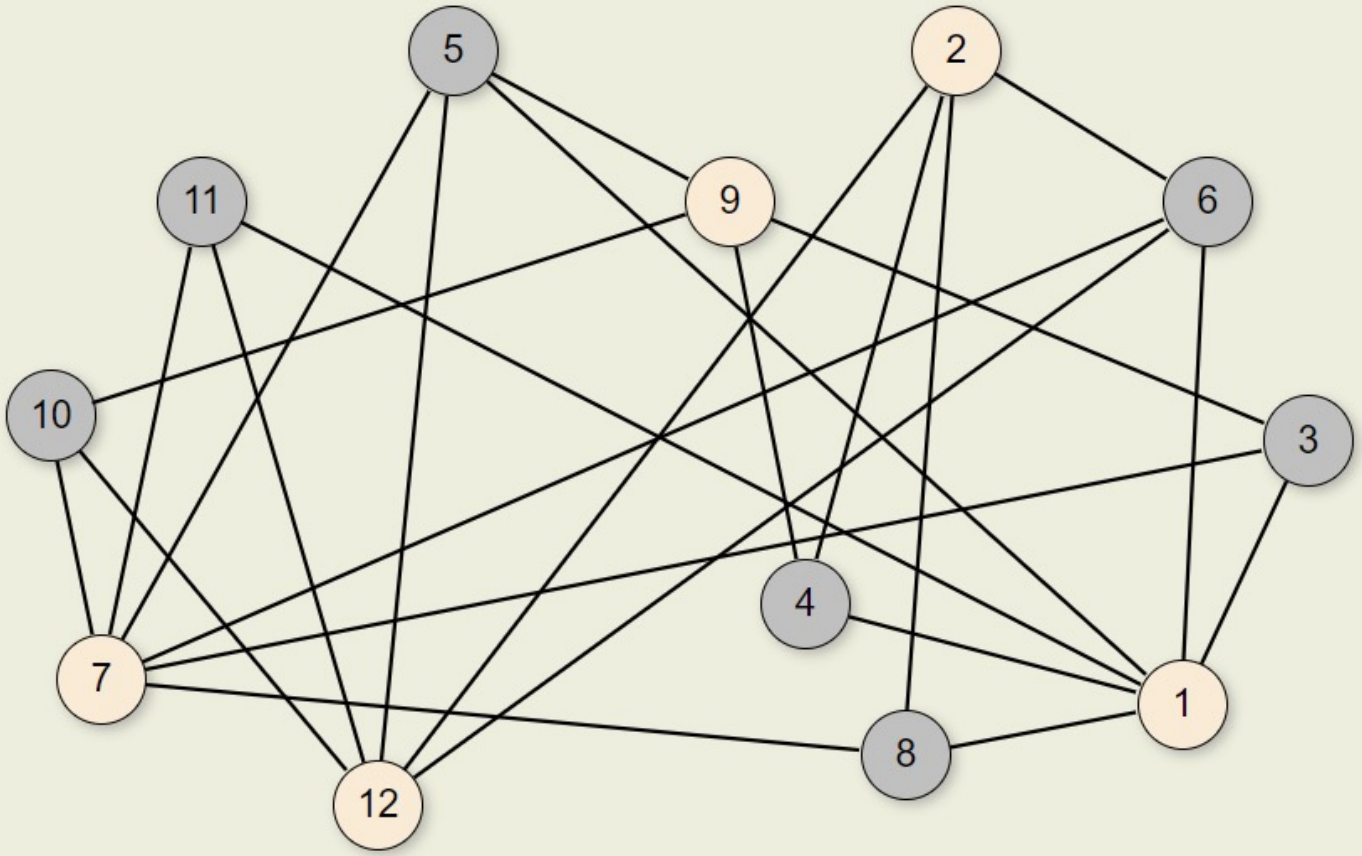


Does this 12-node graph have an Independent Set of size >= 7 ?



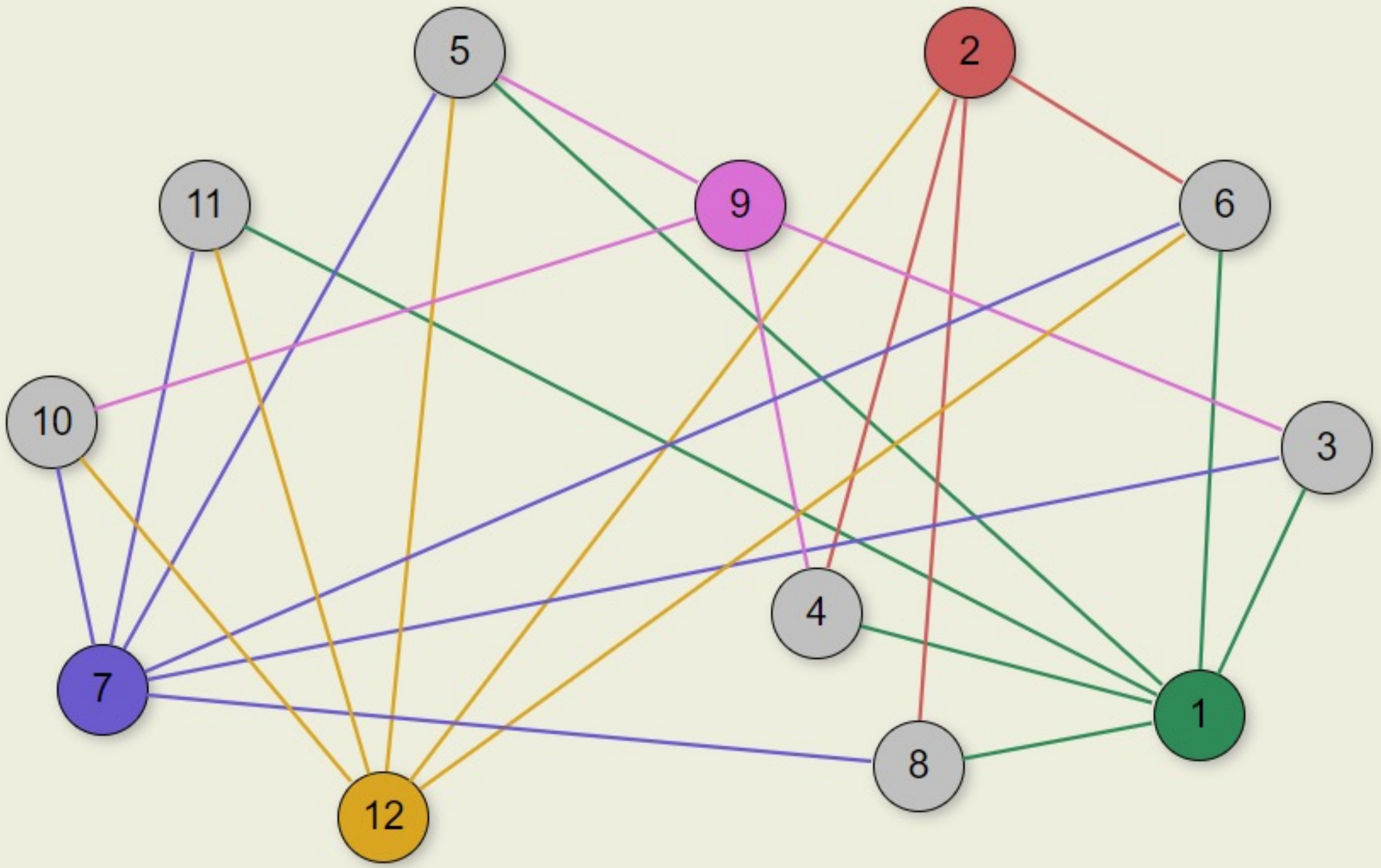

5
2
11
9
6
10
3
4
7
1
8
12

Does this 12-node graph have an Vertex Cover of size <= 5 ?



Does this 12-node graph have an Vertex Cover of size <= 5 ?

Yes



REDUCTION OF 3-SAT TO HAMILTONIAN CYCLE (HC) PROBLEM

– **a simple cycle including all the vertices of the graph (start and end vertex are same)**