CSC 320 - Lecture 17

#np #membership #PATH #HAMPATH #sat #satisfiable #clique #independent-set #vertex-cover #p #np #sat #cnf #np-complete #np-hard #3cnf #reducibility #mapping-reducibility #polynomial-time-reducibility #3sat

---

Membership in NP

Question. How can we show that language A is in NP?

Answer. By giving a polynomial verifier V that checks any candidate solution or certificate c for correctness. What are certificates?

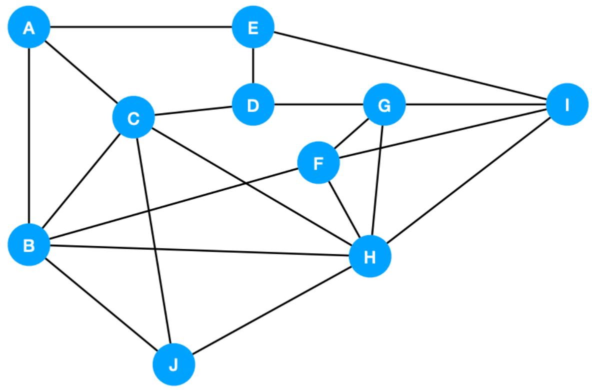
- $PATH = \{\langle G, s, t\rangle \mid G \text{ is a directed graph that has a directed path from } s \text{ to } t\}$.
  - A certificate that passes the verifier for $PATH$ consists of a list of vertices $v_1, v_2, \ldots, v_k$ that correspond to a path from s to t in G.
  - A verifier will ensure that $v_1, v_2, \ldots, v_k$ are pairwise distinct vertices in G, $v_1 = s$ and $v_k = t$. Furthermore, each (v_i, v_{i+1}) must be an arc in G.
- $HAMPATH = \{\langle G, s, t\rangle \mid G \text{ is a directed graph with a Hamiltonian path from } s \text{ to } t\}$.
  - A certificate that passes the verifier for $HAMPATH$ consists of a list of exactly all vertices in the graph that correspond to a path from s to t in G.
  - A verifier will ensure that $v_1, v_2, \ldots, v_k$ are pairwise distinct vertices in G, $v_1 = s$ and $v_k = t$. Furthermore, each (v_i, v_{i+1}) must be an arc in G.

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

- Let $G = (V, E)$ be graph, and let $C \subseteq V$
  - C is a k-clique for G if
    - (1) $|C| \geq k$ and
    - (2) for each pair $a, b \in C$: $(a, b) \in E$

Example

G_1



Example 3-Clique

$\langle G_1, 3\rangle \in$ Clique, where (B, J, H) is the certificate.

$\langle G_1, 3\rangle$, where (E, D, G) is the certificate, will not be accepted.

Example 4-Clique

$\langle G_1, 4\rangle \in$ Clique, where (F, G, H, I) is the certificate.

Note. Showing 4-clique lets us know that we have 3-clique.

Clique $\in NP$

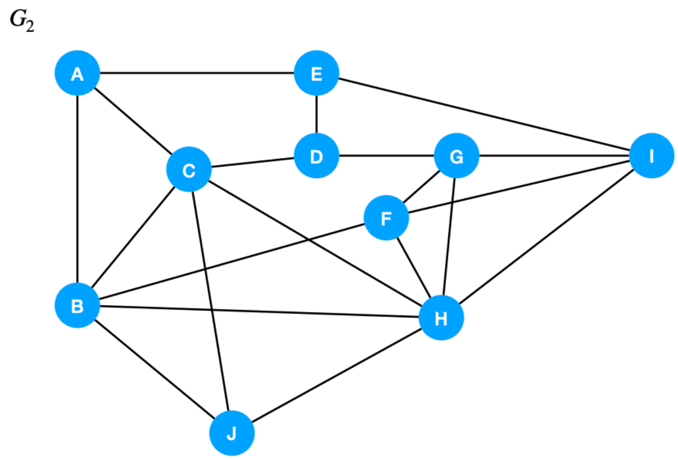
- For $\langle G, k\rangle$, what does a certificate look like?
  - A subset of vertices $C \subseteq V$ with $|C| \geq k$ for each pair $a, b \in C:\ (a, b) \in E$.
- $\textit{Polynomial}$ Verifier
  - $\text{If } |C| < k \text{ then reject}$
  - For each pair $a, b \in C$
    - If $(a, b) \notin E$ reject
  - Accept

Note. $O(n^2)$. $\text{Using an adjacency matrix, for example.}$

$IS = \{\langle G, k\rangle \mid G \text{ is an undirected graph with an independent set of size at least } k\}$

- Let $G = (V, E)$ be a graph, and let $I \subseteq V$
  - I is an independent set of size at least k for G if
    - (1) $|I| \geq k$ and
    - (2) for each pair $a, b \in I:\ (a, b) \notin E$

Example



Example Independent Set of Size At Least 4

$\langle G_2, 4\rangle \in IS$, where (A, D, J, I) is the certificate.

IS $\in NP$

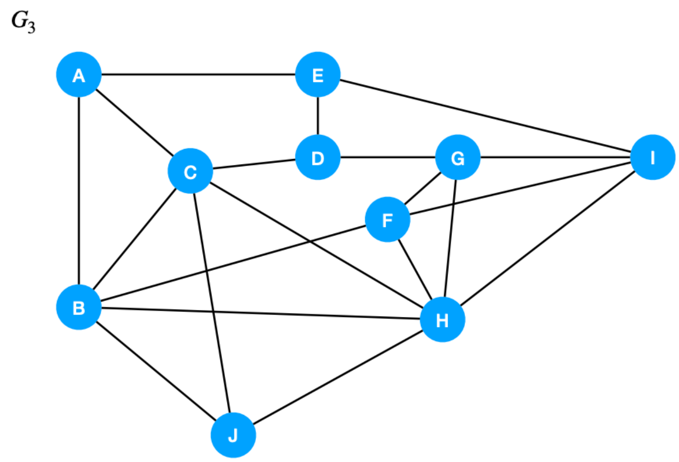
- For $\langle G, k\rangle$, what does a certificate look like?
  - A subset of vertices $I \subseteq V$ with $|I| \geq k$ and for each pair $a, b \in I:\ (a, b) \notin E$.
- $Polynomial$ Verifier
  - $\text{If } |I| < k \text{ then reject}$
  - For each pair $a, b \in I$
    - If $(a, b) \in E$ reject
  - accept

Note. $O(n^2)\text{. Using an adjacency matrix, for example.}$

$VC = \{\langle G, k\rangle \mid G \text{ is an undirected graph with a vertex cover set of size at least } k\}$

- Let $G = (V, E)$ be a graph, and let $V' \subseteq V$
  - V' is a vertex cover of size at most $k4for\$G$ if
    - (1) $|V'| \leq k$ and
    - (2) for each pair $(a, b) \in E:\ a \in V'$ or $b \in V'$

Example



Example Vertex Cover of Size At Most 6

$\langle G_3, 6 \rangle \in VC$, where (B, C, E, F, G, H) is the certificate.

The P VS NP Question (Revisited)

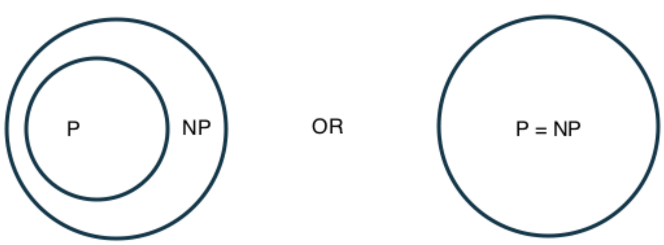
- **We Know**.
  - $P \subseteq NP$

$$\bigcup_k NTIME(n^k) = NP \subseteq \bigcup_k TIME(2^{n^k})$$

- **Want to Know**.
  - If $P = NP$...

$$\bigcup_k NTIME(n^k) = \bigcup_k TIME(2^{n^k})$$

- What about $NP \subseteq P$?



Another SAT Instance

$\Phi = (x_1 \wedge \bar{x_2}) \vee (\bar{x_2} \wedge x_2 \wedge x_3) \vee \bar{x_1}$, where x_2 must be false, x_3 can be true or false, and x_1 must be false.

Question. Is Φ satisfiable? YES!

Note. No other assignment satisfies formula Φ. One satisfying assignment is sufficient.

Φ	x_1	x_2	x_3
1	0	0	1
1	0	0	0

SAT

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

We can show that SAT is in NP. A certificate for SAT is a truth assignment for all variables of the given formula. One can then evaluate in polynomial time in the length of the formula whether or not the formula is satisfied.

Therefore, SAT is in NP. **BUT**. Nobody knows whether or not the problem is also in P.

Note. The certificate would be assigning values to $x_1, \ldots, x_n$.

SAT Links P and NP

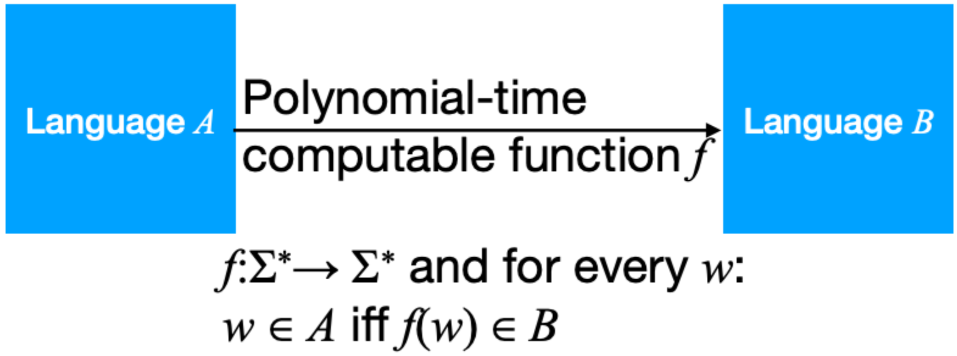
Theorem. $SAT \in P$ if and only if $P = NP$. (Cook/Levin).

Idea. Take any language L in NP, decidable by a nondeterministic TM in polynomial time, and show how to reduce L to SAT in polynomial time.

Turn a polynomial-time nondeterministic TM into a boolean formula.

Polynomial-Time Reducibility

- A function $f: \Sigma^* \longrightarrow \Sigma^*$ is a polynomial-time computable function if some polynomial-time TM M exists that halts with just $f(w)$ on its tape, when started on any input w.
- Language A is polynomial-time mapping reducible (or polynomial-time reducible) to language B, written $A \leq_p B$, if a polynomial-time computable function $f: \Sigma^* \longrightarrow \Sigma^*$ exists, where for every w, $w \in A$ if and only if $f(w) \in B$.
- Function f is called polynomial-time reduction for language A to language B.



$f : \Sigma^* \longrightarrow \Sigma^*$ computable function (by polynomial time TM) and for every $w : w \in A$ if and only if $f(w) \in B$.

Therefore, if $A \leq_p B$... "$w \in A$?" can be decided in time $f(w)$ plus the time it takes to decide whether or not $f(w) \in B$.

Thus, if B is decidable in polynomial time and $A \leq_p B$, then A is decidable in polynomial time also.

Theorem. If $A \leq_p B$ and $B \in P$, then $A \in P$.

Proof.

- Let M be polynomial-time TM / Algorithm deciding B, let f be polynomial-time reduction from A to B.
- We describe polynomial-time TM N deciding A:
  - $N =$ "On input w:
    - (1) Computer $f(w)$
    - (2) Run M on input $f(w)$ and output whatever M outputs."
- Since $w \in A$ if and only if $f(w) \in B$ (f is a reduction from A to B) and: f can be computed in polynomial time and M is polynomial-time decider for B.
  - N runs in polynomial time.

Proving Reductions

To prove that a language A is polynomial-time reducible to a language B normally involves these three steps...

1. The description of a polynomial-time reduction / function f
2. Proving that if $w \in A$ then $f(w) \in B$ (Correctness of Reduction)
3. Proving that id $f(w) \in B$ then $w \in A$ (Correctness of Reduction)

$3SAT$

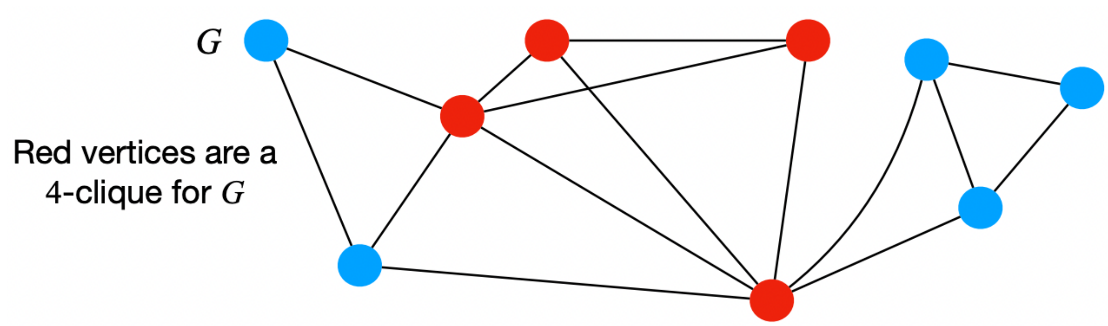
- A special case of SAT.
- Here, formulas are of a special form.
    - Conjunctive normal form (CNF) where each clause is of size 3.
- **Literal**. Boolean variable or negated Boolean variable, such as x or $\bar{x}$.
- **Clause**. Several literals connected with $\vee$'s (i.e., in clause "or" operator, no "and" operator).
    - Ex. $(x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5)$
- A Boolean formula is in conjunctive normal form, called cnf-formula, if: **it comprises several clauses connected with** $\wedge$'s ("and" operator, no "or" operator).
    - $(x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee x_4) \wedge (x_4 \vee \bar{x}_4 \vee \bar{x}_5)$
- A cnf-formula formula is a 3cnf-formula if every clause has *exactly* three literals.
    - $(a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \ldots \wedge (a_k \vee b_k \vee c_k)$
- $3SAT = \{\langle \Phi \rangle \mid \Phi \text{ is a satisfiable 3cnf-formula}\}$

Cliques and CLIQUE

Given an undirected graph $G = (V, E)$ and $V' \subseteq V$.

Reminder. V' is a **clique** for G if for each $x, y \in V' : \ (x, y) \in E$

If V' is a clique for G and $|V'| \geq k$ then V' is a **k-clique** for G.



$3SAT \leq_p CLIQUE$

- $3SAT = \{\langle \Phi \rangle \mid \Phi \text{ is a satisfiable 3cnf-formula}\}$
- $CLIQUE = \{\langle G, k \rangle \mid G \text{ is an undirected graph with } k\text{-clique}\}$

Describe polynomial-time reduction f from $3SAT$ to $CLIQUE$ that converts a given 3cnf-formula with k clauses to a graph, such that... 3cnf-formula is satisfiable if and only if graph has a k-clique.

Let $\Phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \ldots \wedge (a_k \vee b_k \vee c_k)$ be a 3cnf-formula.

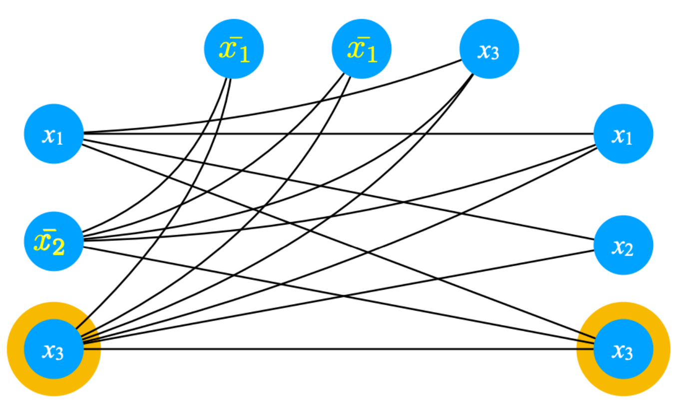
- Reduction f: generates string $\langle G, k \rangle$ where G is an undirected graph (with k positive integer).
  - Vertices in G: organized into k groups, $t_1, \ldots, t_k$, of 3 nodes each, called **triplets**... (Create 3k vertices for G, on for each literal):
    - Each triplet corresponds to one of the clauses in Φ
    - Each node in a triplet corresponds to a literal in associated clause.
  - Label each node of G with its corresponding literal in Φ.
  - Add edges in G for all but two types of pairs of nodes in G:
    - No edge is present between nodes in the same triplet,
    - No edge is present between two nodes with contradictory labels (i.e., between $x_i and \bar{x_i}$).

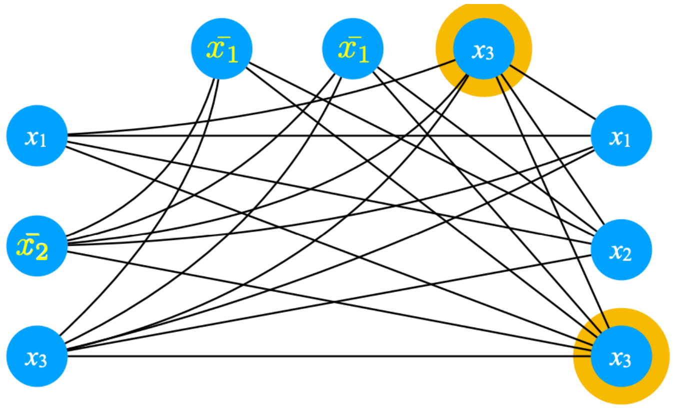
Example

$\Phi = (x_1 \vee \bar{x_2} \vee x_3) \wedge (\bar{x_1} \vee \bar{x_1} \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$

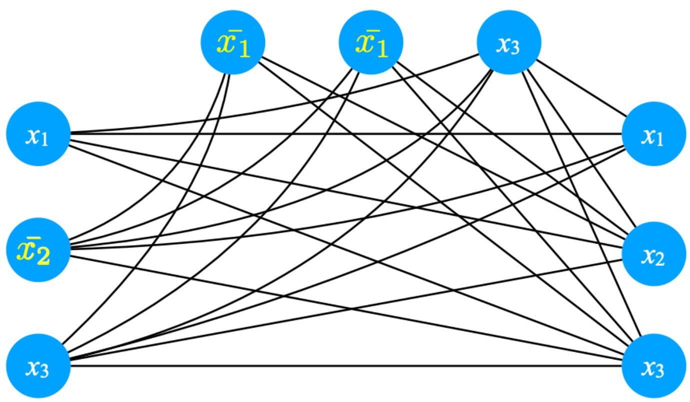
1. Build $G = (V, E)$. First for each clause we create 3 vertices.

2. Then create edges for vertex pairs of different clauses without contradictory labels.





Final Result



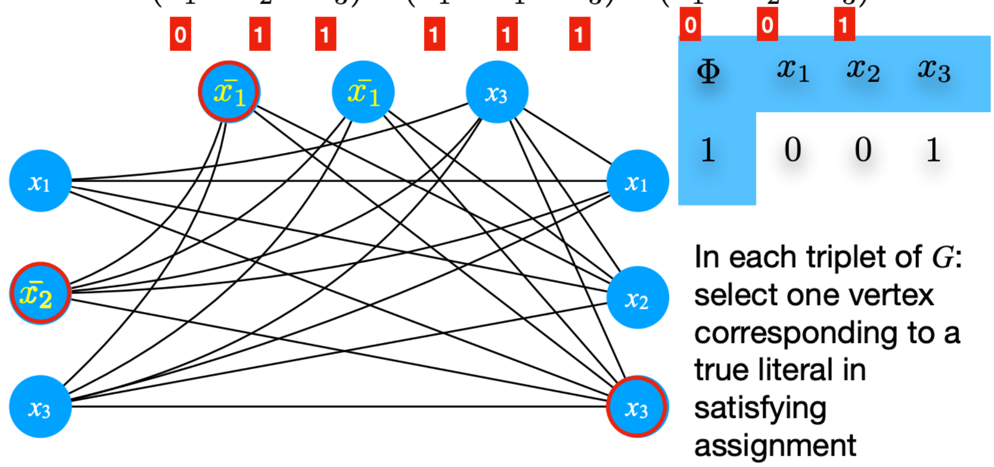
- Correctness of reduction: we show that Φ is satisfiable if and only if G has k-clique.
- $\Rightarrow$ Suppose Φ has satisfying assignment...
  - Corresponding to the satisfying assignment: at least one literal is true in every clause.
  - In each triplet in G: select one vertex corresponding to a true literal in the satisfying assignment.
  - The vertices just selected form a k-clique:
    - k vertices are selected since we chose one for each of the k triplets.
    - Each pair of selected nodes is joined by an edge because no pair stems from the same clause and no pair's labels are contradictory.
- $\Leftarrow$ Suppose G has k-clique
  - No two of the vertices in clique occur in same triplet since such pairs are not connected by any edges.
  - Thus each of the k triplets contains exactly one of the k-clique nodes.
  - We can assign truth values to the variables of Φ so that each literal labelling a clique vertex is made true.
    - Two vertices labeled in a contradictory way are not connected by an edge and hence cannot be both in the clique.

- This truth assignment satisfies Φ because each triplet contains a clique vertex and thus each clause contains a literal that is TRUE.
- Φ is satisfiable.

Build Clique From Satisfying Assignment

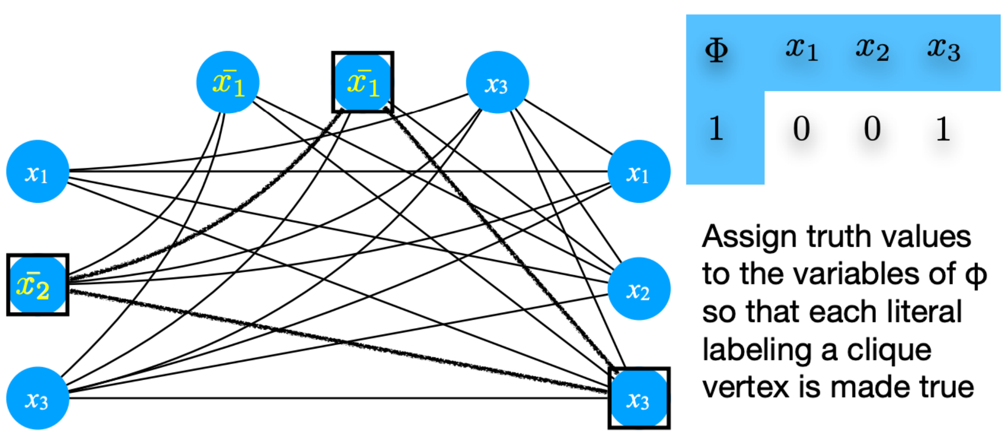
Φ	x_1	x_2	x_3
1	0	0	1

$$\Phi = (x_1 \vee \bar{x_2} \vee x_3) \wedge (\bar{x_1} \vee \bar{x_1} \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$$



G Has A k-Clique

$$\Phi = (x_1 \vee \bar{x_2} \vee x_3) \wedge (\bar{x_1} \vee \bar{x_1} \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$$



Because $3SAT \leq_p CLIQUE$... If $CLIQUE$ is decidable in polynomial time, then so is $3SAT$.

How can we decide whether a formula Φ in 3cnf is decidable?

Since $3SAT$ is NP-Complete, so is $CLIQUE$

NP-Completeness

Definition. A language B is NP**-Complete** if it satisfies the following two conditions...

1. $B \in NP$
2. B in NP-Hard

And: B is NP**-Hard** if for every $A \in NP: \ A \leq_p B$

Therefore, If B is NP-Complete and $B \in P$ then $P = NP$.

Furthermore, If B is NP-Complete and $B \leq_p C$ for $C \in NP$, then C is NP-Complete.

Previous Lecture

Lecture16

Next Lecture

Lecture18