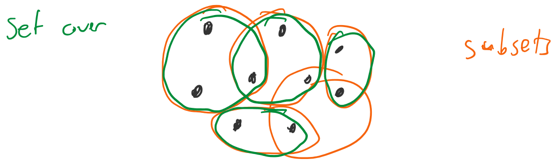
Lec28-more-NP

KT Chapter 8

Saturday, November 18, 2023 9:48 AM

Problem (Set Cover) Given a set U of elements, and a collection $S_1, \ldots, S_m$ of subsets of U, is there a collection of at most k of these subsets whose union is U?



Lemma: Set Cover $\in NP$.

proof. The certificate is a list of k sets from $S_1, \ldots, S_m$. We can take the union of these sets and check that the union is equal to U in poly-time ∎

Lemma: Vertex Cover $\leq_p$ Set Cover.

proof- Let $G=(V,E)$ and k be an instance of Vertex Cover. Create an instance of Set Cover:

- $U = E$
- Create a S_u $\forall u \in V$ where S_u contains the edges adjacent to u.

U can be covered by $\leq k$ sets iff G has a vertex cover of size $\leq k$.

Why? If k sets $S_{u_1}, \ldots, S_{u_k}$ cover U, then every edge is adj to at least one of vertices $u_1, \ldots, u_k$, giving a vertex cover.

Conversely, if $u_1, \ldots, u_k$ is a vertex cover, then sets $S_{u_1}, \ldots, S_{u_k}$ cover U. ∎

$\Rightarrow$ Set Cover is NP-hard

$\Rightarrow$ Set Cover is NP-complete.

$\Rightarrow$ Set Cover is NP-complete.

Punchline: If you can reduce
Vertex Cover
Ind. Set
or Set Cover
} known NP-complete

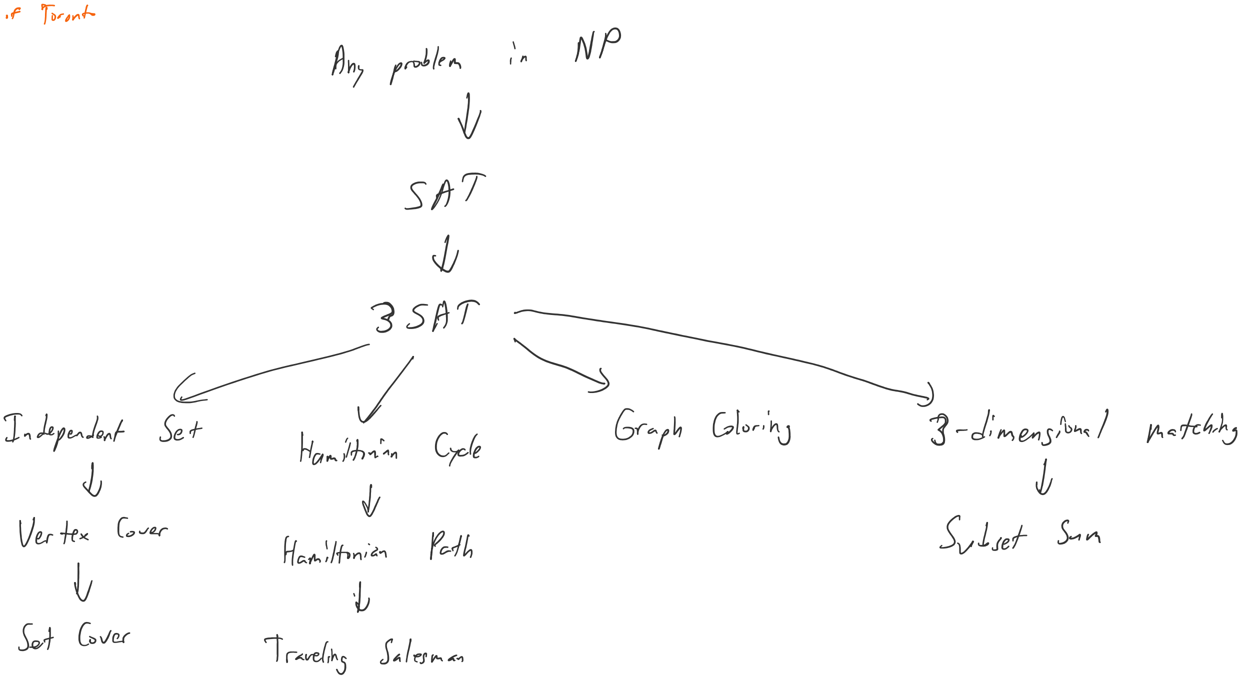
$\boxed{\text{to}}$ some problem X, then X is NP-hard,
and if additionally, $X \in NP$, then X is NP-complete.

careful about direction of reduction

Cook-Levin Theorem (1971): The SAT problem is NP-complete
(Boolean Satisfiability)

North American
(denied tenure at Berkeley)
Emeritus at University of Toronto

USSR
(Now at BU)



Boolean Formula

Variables: $x_1, x_2, x_3, \dots \in \{T, F\}$ (Boolean)

Terms (literals): $t_1, t_2, t_3, \dots, t_\ell$, where $t_j = x_i$ or $\overline{x_i}$ (not x_i)

OR clause: $t_{K_1} \vee t_{K_2} \vee t_{K_3} \vee \cdots \vee t_{K_p}$ (OR of variables & their negations)

CNF: AND of ORs. Let $C_1, \dots, C_K$ be clauses.

CNF : AND of ORs. Let $C_1, \dots, C_k$ be clauses.

(conjunctive normal form) $C_1 \wedge C_2 \wedge \cdots \wedge C_k$ is a Boolean formula in CNF.

Ex. $(x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_2 \vee x_3)$ Ex. $x_1 = T$, $x_2 = T$, $x_3 = F$

Ex. $(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee x_1)$ Ex. $x_1 = T$, $(x_2 = F, x_3 = F)$

Def. A truth assignment is a choice of T/F for each variable.
i.e. function $v: X \to \{T, F\}$.

Def. A truth assignment is a satisfying assignment for a Boolean formula if it makes the formula true.
(For CNF, if every clause is set to True)

Problem (SAT) Given a set of OR-clauses $C_1, \dots, C_k$ over variables $X = \{x_1, \dots, x_n)$, is there a satisfying assignment.

Problem (3SAT) Given a set of OR-clauses $C_1, \dots, C_k$, each of length 3, over variables $X = \{x_1, \dots, x_n\}$, is there a satisfying assignment

naturally decision problems

Reducing SAT to 3SAT (to show 3SAT is NP-hard)

Given SAT instance $\mathcal{C} = \{C_1, \dots, C_m\}$. Suppose $|C_i| > 3$.

$$C_i = (t_1 \vee t_2 \vee t_3 \vee \cdots \vee t_k)$$ ← True if any one of variable t_i is T.

Replace clause with set of clauses with new y_i variables

$$(\underset{F}{t_1} \vee \underset{F}{t_2} \vee \overset{T}{y_1})$$

$$(\overset{T}{\bar{y}_1} \vee \underset{F}{t_3} \vee \overset{T}{y_2})$$

$$(\overset{T}{\bar{y}_2} \vee \overset{F}{\underset{\uparrow T}{t_4}} \vee y_3)$$

$$(\overset{T}{\underset{\overline{F}}{\bar{y}_3}} \vee \overset{F}{t_5} \vee \underset{T}{y_4})$$

$$\ddots$$

If $t_i = T$, can set $y_j = \begin{cases} T, & j \le i-2 \\ F, & \text{else} \end{cases}$

If all clauses are true F T .

$(y_3 \vee t_5 \vee \overset{}{\underset{T}{y_4}})$
$\overline{F}$

$\left(\overline{y_{k-2}} \vee \overset{F}{t_{k-2}} \vee \overset{T}{y_{k-3}}\right)$
$\overline{F}$

$\underbrace{\left(\overline{y_{k-3}} \vee \overset{F}{t_{k-1}} \vee \overset{F}{t_k}\right)}_{\text{not true.}}$

If all clauses are true,
then at least one $t_i = T$.
Suppose not. Then $y_1 = T$
$\Rightarrow y_2 = T \Rightarrow \dots y_{k-3} = T$,
but then last clause is F.
$\Rightarrow$ all clauses $= T$ implies original clause was T.

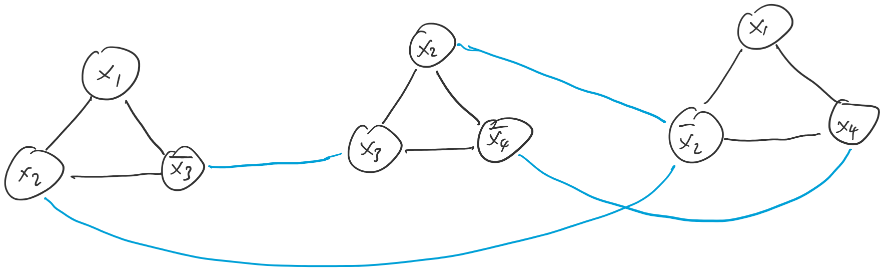
$\Rightarrow$ SAT $\leq_p$ 3SAT

Thm 3SAT $\leq_p$ Independent Set

proof. To solve 3SAT, have to choose a term from each clause to set to True but can't set both x_i + $\overline{x_i}$ to True

Strategy: Construct a graph whose choice of nodes in independent set corresponds to choice of True variables. (but not choosing doesn't imply False)

Ex. $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee x_3 \vee \overline{x_4}) \wedge (x_1 \vee \overline{x_2} \vee x_4)$



triangle gadget means that given k clauses, ind. set of at most size k.

negation gadget links x_i with all $\overline{x_i}$, so you can't ever set both x_i + $\overline{x_i}$ simultaneously to True.

If formula is satisfiable, at least one true literal in each clause.
Let S be a set of one true literal from each clause
$|S| = k$ and no two nodes in S are connected by an edge
(otherwise, would get $x_i = T = \overline{x_i}$)

If graph has independent set $|S| = k$, must have one node from each triangle gadget. Set those terms to true in original 3SAT formula, giving a solution to 3SAT. $\boxtimes$

General strategy for NP-complete proofs.

1. Show $X \in NP$ by showing that there is a (s.t.) certificate that is efficiently checked.
2. Look at known NP-complete problems.
   Choose a Y that seems "similar" to X.
3. Show that $Y \leq_p X$.

   A) Let I_Y be any instance of Y.

   B) Construct instance I_X in poly time s.t.

   - If I_Y is Yes-instance, then I_X is yes-instance.
   - If I_X is Yes-instance, then I_Y is yes-instance.

   So if you can solve X, then you can solve Y.
   (but not vice versa, since we are converting Y instance to X-instance)

Problem (Hamiltonian Cycle) Given directed graph G, is there a cycle that visits every vertex exactly once. (and then returns to start)

Known as Hamiltonian Cycle.

Theorem: Hamiltonian Cycle is NP-complete.

proof. Ham Cycle $\in NP$, because given a cycle, it is straight-forward to check all nodes visited & all edges valid.

Want to show 3SAT $\leq_p$ Ham Cycle.

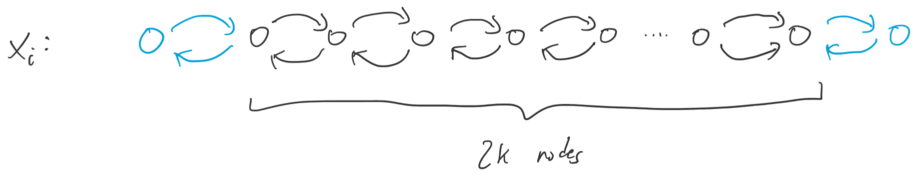
Need to encode 3SAT instance as Ham Cycle instance.

variables $x_1, \ldots, x_n$
clauses $C_1, \ldots, C_k$

} construct "gadgets" representing both & hook them up in graph.

Show this graph has Ham. cycle iff formula is satisfiable.

Variable gadget:



Note that once you choose a direction, you can only go in that direction.

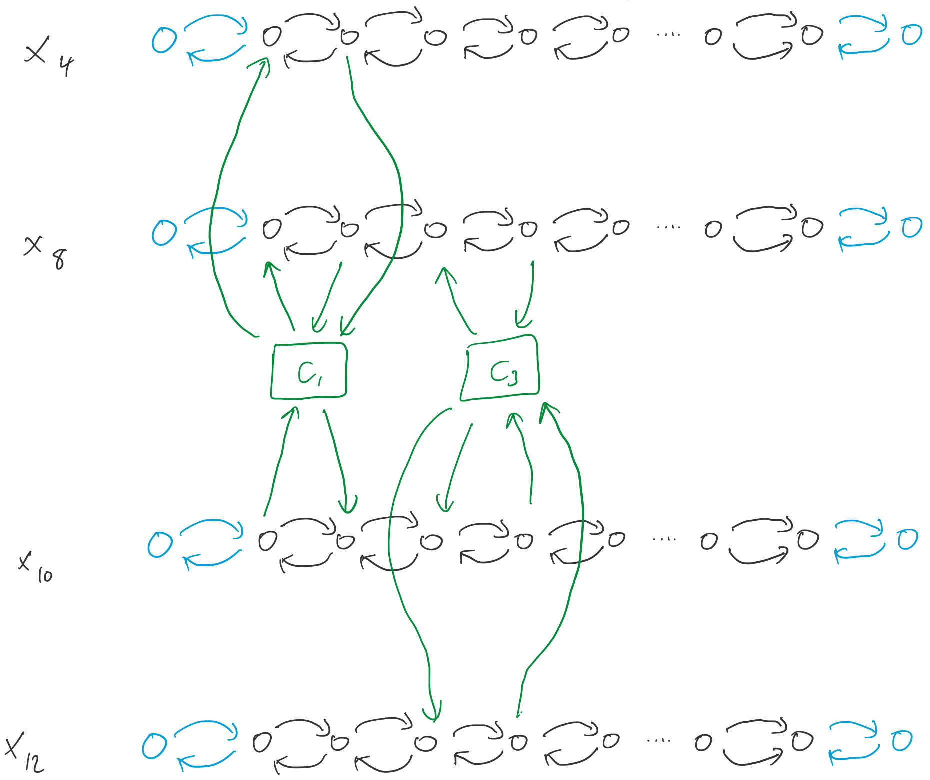
Let ⟵ correspond to setting $x_i = T$

⟶ correspond to setting $x_i = F$.

Clause gadget: Add a single node for $C_j = t_1 \vee t_2 \vee t_3$ hooked into the variables corresponding to t_1, t_2, t_3, with direction specified by if $t_\ell = x_i$ or $\overline{x_i}$, and hooked into the pos. on path corresponding to j.

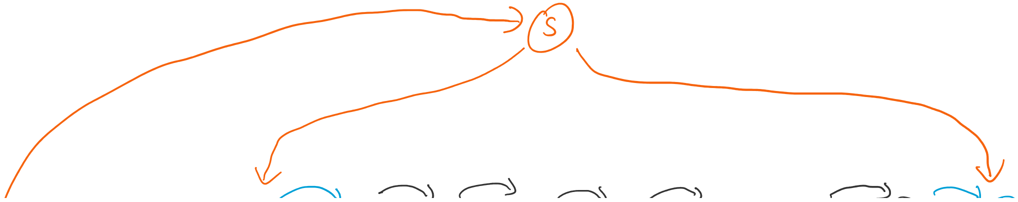
Ex. $C_1 = x_4 \vee x_8 \vee \overline{x_{10}}$

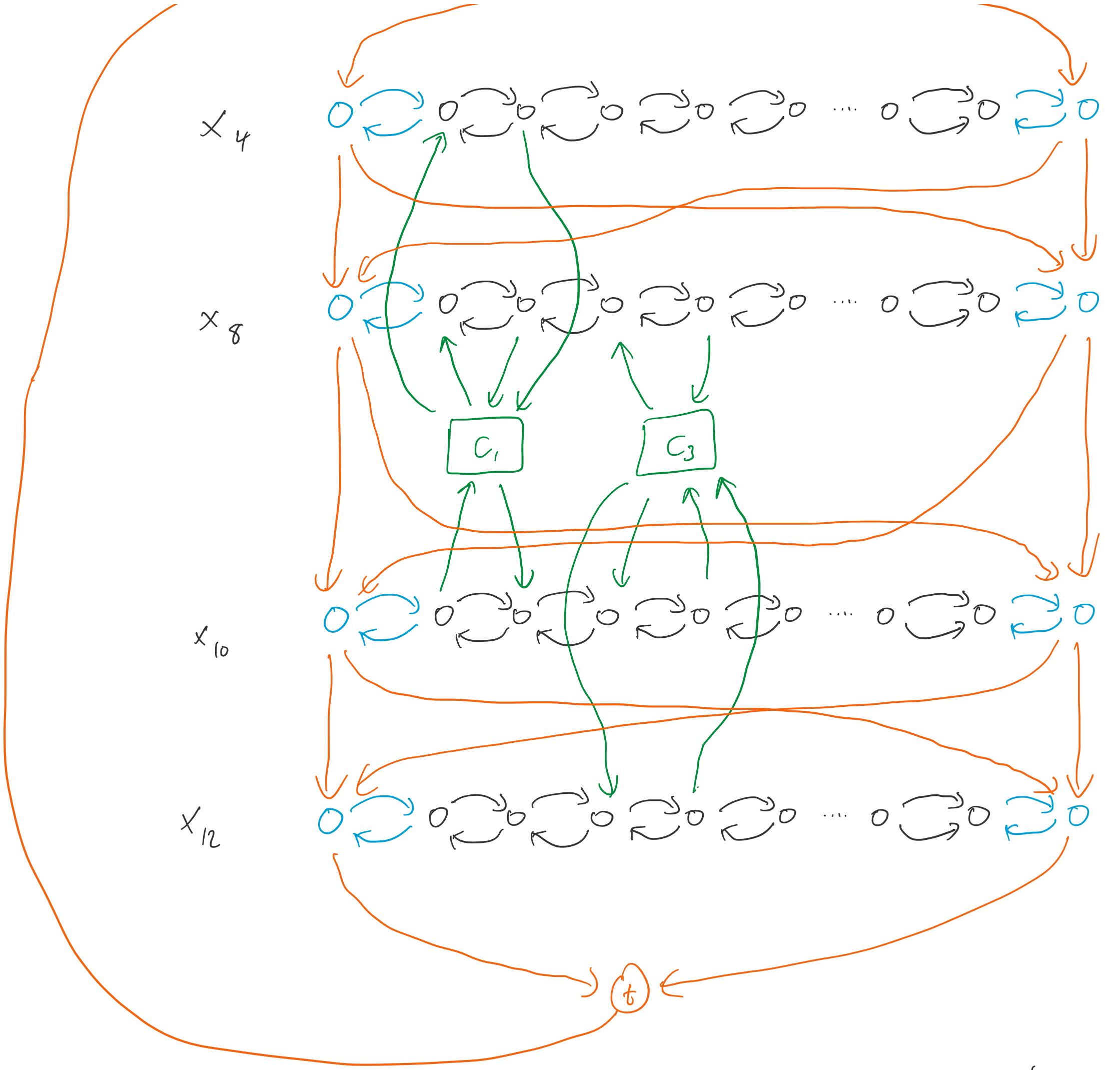
$C_3 = x_8 \vee x_{10} \vee x_{12}$



We have to visit each clause, but can only visit them while going along a variable, in the right direction.

Connecting up graph: Add start + end nodes, and directional connections from x_1 to x_2 to x_3, ...





A Hamiltonian cycle on this graph has to start at s & end at t. But it has to walk through all the variable gadgets either going left or right. For each clause node, it can only be reached if we walk along one of its literals in the right direction.

So a Hamiltonian cycle is exactly equiv to a 3SAT solution

□

Hamiltonian Path: Does G contain a path that visits every node exactly once?
(doesn't have to return to start)

We could adapt the 3SAT → HamCycle reduction.

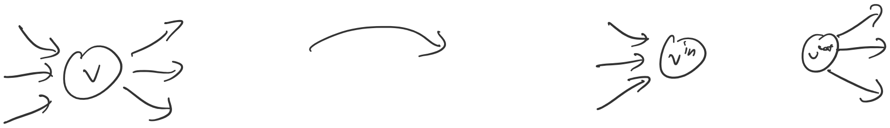
Or, we can instead show HamCycle → 3SAT.

Thm. HamPath is NP-complete.

proof HamPath $\in$ NP because easy to check path.

Let's show HamCycle $\leq_p$ HamPath.

Given HamCycle Instance G, chose arbitrary node v and split it into two nodes v^{in}, v^{out}, in graph G'.



Any Ham Path must start at v^{out} & end at v^{in}.

G' has Ham Path $\iff$ G has Ham Cycle.

because if G' has HamPath, same path after gluing v^{in} and v^{out} back together is HamCycle,

and if G has HamCycle, that provides a HamPath in G' by using the split above. ▨

Traveling Salesman Problem

Given n cities, distances $d(i,j)$ between cities, does there exist a path of length $\leq k$ that visits each city and returns home?

Note: $d(i,j) \neq d(j,i)$ in general

And: $d(i,j) \leq d(i,k) + d(k,j)$ does NOT hold.

(no $\triangle$ inequality)

Thm. TSP is NP-complete.

pf. TSP $\in$ NP because if we are given a path, we can easily check that it visits every city & compute pf. TSP $\in$ NP because if we are given a path, we can easily check that it visits every city & compute the length.

Next, we will show that HamCycle $\le_p$ TSP.

Need to reduce HamCycle to TSP instance

HamCycle:

$|V| = n$

Graph $G = (V, E)$

Want cycle that visits every city exactly once & returns to start

TSP instance D:

city c_i for every vertex v_i.

Let $d(c_i, c_j) = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 2 & \text{otherwise} \end{cases}$

Lemma: G has Hamiltonian cycle

$\iff$

D has tour of length $\le n$.

proof. If G has Ham. Cycle, then this ordering of cities gives a tour of length $\le n$ in D

(only distances of length 1 used)

Suppose D has a tour of length $\le n$.
Then the tour can't ever use a dist-2 connections, because with n stops, that would be too long, so it visits cities only via dist-1 connections, which are exactly edges in G, giving a Ham. Cycle.

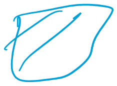
□

Thus, Ham Cycle $\le_p$ TSP.

$\Rightarrow$ TSP is NP-hard.

Since TSP $\in$ NP + is NP-hard,

TSP is NP-complete.



Even if distances are Euclidean, TSP is NP-complete.