NP-completeness proofs

Recall from last day

Definition Problem X is NP-complete if

① $X \in NP$

② $Y \leq_p X$ for all $Y \in NP$
↖ reduces in poly. time

We will prove Satisfiability is NP-complete
even the special case of 3-SAT

How to prove a problem Z is NP-complete (after 1st proof)

① show $Z \in NP$

② show $X \leq_p Z$ for some known NP-complete X.

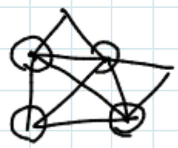
Last day: Ind. Set is NP-complete
using reduction 3-SAT $\leq_p$ Ind. Set

X

Today: more NP-complete problems.

<u>Clique</u>: Given a graph $G=(V,E)$ and $k \in \mathbb{N}$, does G have a <u>clique</u> of size $\geq k$

set of vertices, every two joined by an edge.

 $k=4$

<u>observe</u> $C \subseteq V$ is a clique in G iff C is an independent set in G^c – the <u>complement</u>

G^c – vertices V

– edge (u,v) iff $(u,v) \notin E(G)$.

<u>Thm</u> Clique is NP-complete.

<u>Proof</u> (1) Clique $\in$ NP

certificate: the vertices of the clique

verification: – check $\geq k$ vertices

– check every 2 joined by edge

(2) Ind Set $\leq_P$ Clique

Alg. for Ind. Set assuming poly. time alg. for clique

– give input G, $n-k$ to clique alg.

– return YES/NO answer.

This takes poly. time

It is correct because G has an ind. set of size

Today: more NP-complete problems.

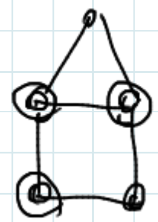
Vertex Cover

Input: Graph $G=(V,E)$, number $k \in \mathbb{N}$

Question: Does G have a vertex cover of size $\cancel{\geq}$ $\leq$ (typo) k

= a set $S \subseteq V$ s.t. every edge $(u,v) \in E$ has u or v (or both) in S

e.g.



observe: $V \setminus S$ is an independent set

Theorem Vertex Cover is NP-complete

Pf. ① Vertex Cover $\in$ NP

certificate: the set S

verification: check that every edge has endpoint in S

and check $|S| \leq k$

② Ind. Set $\leq_p$ Vertex Cover

Assume we have an alg. for Vertex Cover.

Give an alg. for Ind. Set.

We use relationship between Vertex Cover & Ind Set in G:

Claim S is a vertex cover iff $V \setminus S$ is an ind. set.

Pf. Exercise

Here's our alg. for Ind. Set

input G, k

give $G, n-k$ to Vertex Cover alg.

output YES/NO answer

Correct. Poly. time (assuming Vertex Cover alg. is poly-time)

Road Map of NP-completeness Reductions

$$\text{3-SAT} \le_p \text{IND.SET} \le_p \text{VERTEX COVER} \le_p \text{SET COVER}$$

$$\text{Circuit-SAT} \le_p \text{3-SAT} \le_p \text{HAM.CYCLE} \le_p \text{TSP}$$

$$\text{3-SAT} \le_p \text{SUBSET SUM}$$

History

proof that 3-SAT is NP-complete
due to Prof. Stephen Cook, U. of Toronto, '71

W https://en.wikipedia.org/wiki/Stephen_Cook

and independently to Levin.

Other "first" NP-completeness proofs above due to Richard Karp.

Directed Hamiltonian Cycle

Input: a directed graph $G=(V,E)$

Question: Does G have a [directed] Hamiltonian cycle, i.e. a directed cycle that visits every vertex exactly once.

Thm Directed Ham. cycle is NP-complete

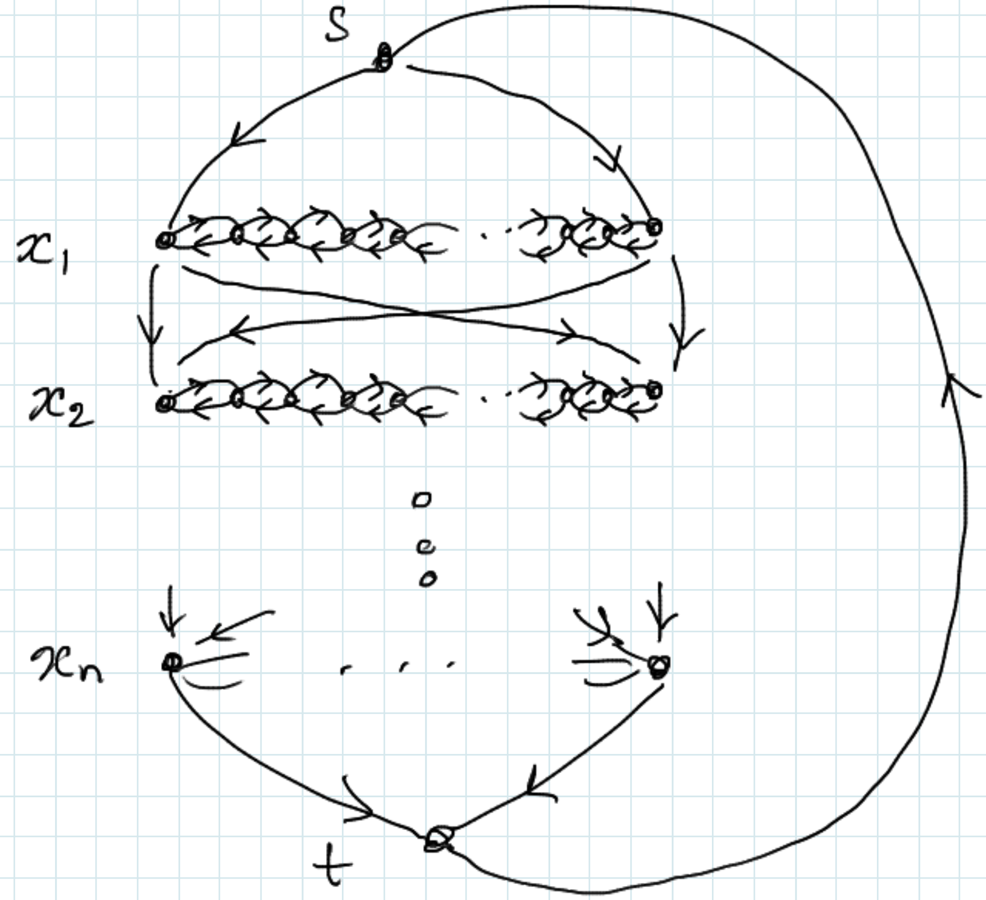
Pf. ① $\in NP$ certificate - order of visiting vertices
verification - check for directed edge between each pair of vertices, and that all vertices visited once.

② 3-SAT $\leq_p$ Directed Ham. cycle.

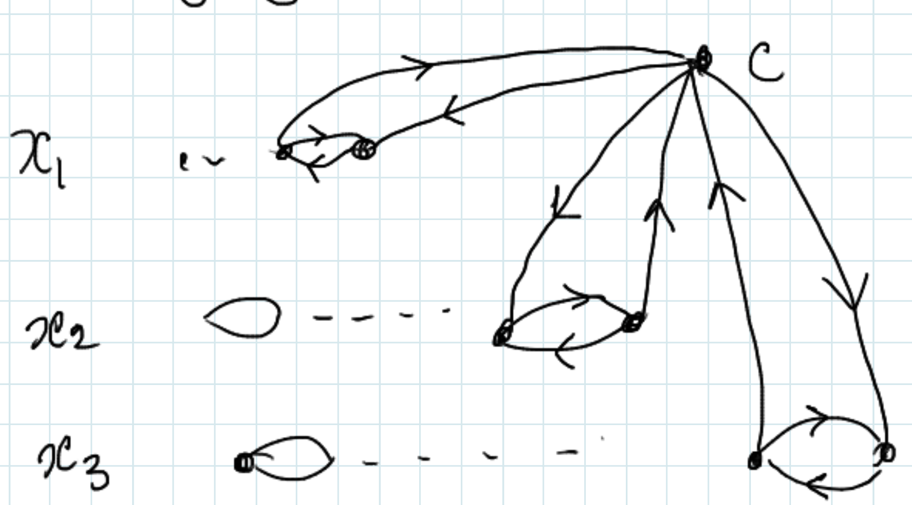
Assume we have a poly. time alg. for Directed Ham. cycle.
Design a poly. time alg. for 3-SAT.

Input: Clauses $C_1 \cdots C_m$, each clause has 3 literals
variables $x_1 \cdots x_n$.

Construct gadget to choose T/F for each variable

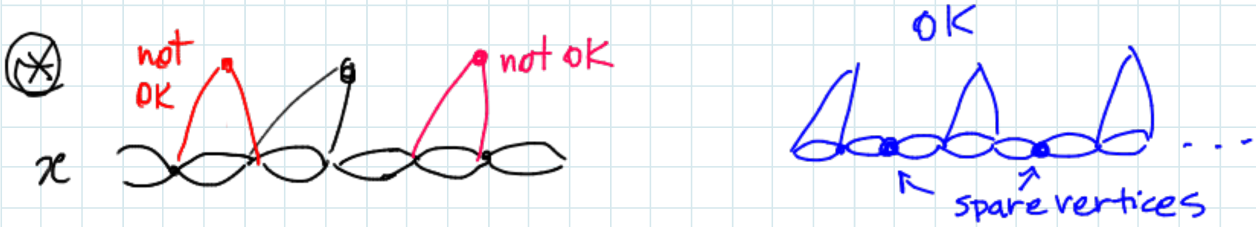


Going left → right on path for x_i corresponds to x_i = True
" right → left - - - - - x_i = False
A Ham. cycle must use one or the other path for each x_i.
Clause gadget for clause $C = (x_1 \vee \neg x_2 \vee x_3)$



Idea: visit C by detouring off x_1 T path OR x_2 F path OR x_3 T path

Note: make sure to leave a spare vertex between 2 clause detours



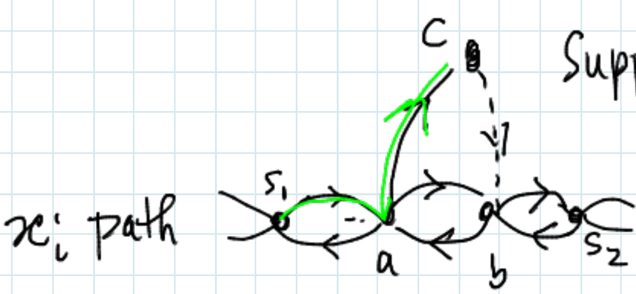
To prove this construction is correct we must prove there's no other way to visit C.

Claim G has a directed Ham. cycle iff all clauses satisfiable

Pf $\Leftarrow$ traverse the variable paths in True/False direction. For each clause C, at least one literal is set True — take a detour from that path to node C.

$\Rightarrow$ Suppose G has a Ham. cycle

Claim visiting C must happen as a detour off a path



Suppose we use (a, c) but not (c, b)

Can't use (a, s_1) so must enter s_1 from left. Must use (s_1, a).

Thus the Ham. cycle must traverse a T or F path for each variable, and must visit each clause vertex off such a path.

So this corresponds to satisfying truth value assignment.

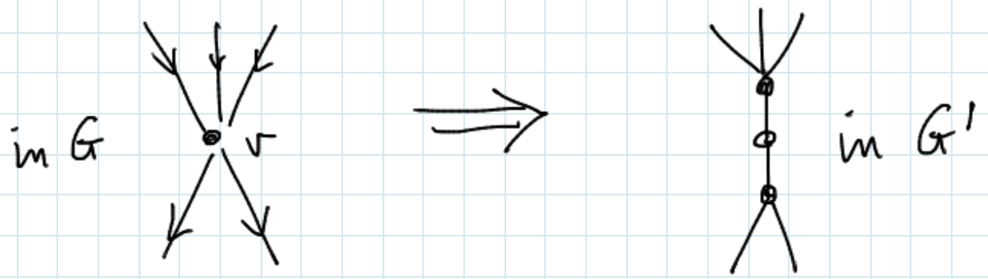
Claim This construction takes poly. time.

Thm undirected Ham. cycle is NP-complete.

Pf. ① $\in$ NP

② directed Ham. cycle $\leq_p$ undirected Ham. cycle

Given $G=(V,E)$ input for directed Ham cycle, construct G', input for undirected " " as follows



Claim 1 G has directed Ham cycle iff G' has Ham. cycle

Claim 2 poly. time.

Thm TSP (directed or undirected version) is NP-complete

Pf. EX.

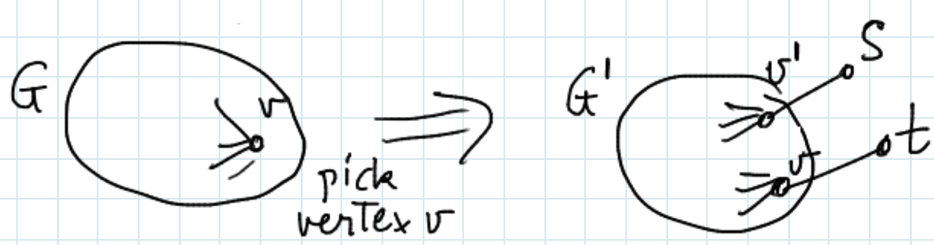
Thm Ham. path is NP-complete

Pf ① $\in$ NP

② Ham. cycle $\leq_p$ Ham. path

Added afterwards

Suppose we have a poly. time alg. for Ham. path.
Design a poly. time alg. for Ham. cycle
Given graph G, input for Ham. cycle, construct G'



v' = new copy of v
s,t = new vertices.

Claim. G has Ham. cycle iff G' has Ham. path.