Lec 23. More on NP-completeness

Eunjung Kim

How we reached the first NP-complete problem

- All deterministic TM's are polynomially equivalent.

How we reached the first NP-complete problem

- All deterministic TM's are polynomially equivalent.
- Class NP: $A \in$ NP iff admits a nondeterministic TM deciding A in $O(n^c)$ for some fixed c on length n input.

How we reached the first NP-complete problem

- All deterministic TM's are polynomially equivalent.
- Class NP: $A \in$ NP iff admits a nondeterministic TM deciding A in $O(n^c)$ for some fixed c on length n input.
- Equivalently: iff admits a witness of $O(n^c)$ for some fixed c, verifiable by a deterministic TM.

How we reached the first NP-complete problem

- All deterministic TM's are polynomially equivalent.
- Class NP: $A \in \mathrm{NP}$ iff admits a nondeterministic TM deciding A in $O(n^c)$ for some fixed c on length n input.
- Equivalently: iff admits a witness of $O(n^c)$ for some fixed c, verifiable by a deterministic TM.
- Completeness for NP: every problem in NP are P-time many-one reducible to A.

How we reached the first NP-complete problem

- All deterministic TM's are polynomially equivalent.
- Class NP: $A \in$ NP iff admits a nondeterministic TM deciding A in $O(n^c)$ for some fixed c on length n input.
- Equivalently: iff admits a witness of $O(n^c)$ for some fixed c, verifiable by a deterministic TM.
- Completeness for NP: every problem in NP are P-time many-one reducible to A.

The first NP-complete problem

The very definition of NP-completeness indicates that to exhibit the first NP-complete problem (not as a concept), we need to design a many-one reduction that relies (solely) on the fact that a problem is in NP.

This is Cook-Levin Theorem, which uses the equivalent characterization of NP.

Transmitting hardness

It is cumbersome to use a similar strategy as in Cook-Levin to show that a new language is NP-complete.

Fortunately, once we get a first NP-complete problem, we can use the transitivity of P-time many-one reduction.

The following problems are NP-complete

- 3-CNF
- Independent Set
- 3-Coloring
- Directed Hamiltonian Path

Conjunctive Normal Form (CNF)

- CNF (Conjunctive Normal Form) formula: $\wedge$ ('AND') of $\vee$'s ('OR) of variables and their negations.
  $(x_1 \vee \bar{x}_3) \wedge (x_2 \vee x_4) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5)$.
- Literals are variables and its negations.
- A clause is a OR's over literals.
- CNF formula is a conjunction ('AND') of clauses.
- A formula φ is k-CNF if each clause contains at most k literals.

Problem definition: CNF-Sat

Input: a CNF formula φ

Question: is there a satisfying assignment to the variables of φ?

Reduction: CNF-SAT to 3CNF-SAT

Problem 3CNF-SAT

Input a 3-CNF formula φ

Question is there a satisfying assignment to the variables of φ?

Key step in the reduction

$C = (x_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4) \rightsquigarrow C_1 = (x_1 \vee \bar{x}_2 \vee z),\ C_2 = (x_3 \vee \bar{x}_4 \vee \bar{z})$

REDUCTION: CNF-SAT TO 3CNF-SAT

- Replace each clause C with $\ell \geq 4$ literals by two clauses C_1, C_2.
- C_1, C_2 are (simultaneously satisfied) if and only if C is satisfied .
- How to: divide the literals of C roughly into two parts so that one part has two literals (the other $\ell - 2$), and add a new variable y to toggle

KEY STEP IN THE REDUCTION

$$C = (z_1 \vee z_2 \vee z_3 \vee \cdots \vee z_\ell) \rightsquigarrow C_1 = (z_1 \vee z_2 \vee y),\ C_2 = (z_3 \vee \cdots \vee z_\ell \vee \bar{y})$$

Reduction: CNF-Sat to 3CNF-SAT

Proof. Let φ be an arbitrary formula in CNF.

1. Construction: for each clause C with at least $\ell \geq 4$ literals, we replace C by the two clauses C_1 and C_2 as described in the previous slide.
2. Polynomial-time: By each replacement the following value

$$\sum_{C \in \varphi : |C| \geq 4} |C|$$

   strictly decrease. Therefore after $||\varphi||$ rounds of replacement, we obtain a 3-CNF formula φ'.
3. (This implies that the entire construction can be completed in polynomial-time in $||\varphi||$. Why?)
4. Equivalence: That φ is satisfiable if and only if φ' is follows from that C is satisfiable if and only if C_1 and C_2 above.

Reduction: 3CNF-SAT to IND

An independent set I of a graph is a vertex subset I of G s.t. any two vertices $u, v \in I$ is non-adjacent.

Problem Independent Set

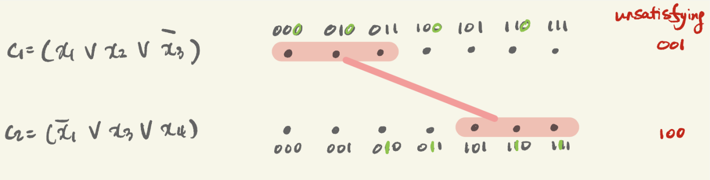
Input a graph G and an integer k

Question does G have an independent set of size (at least) k?

Reduction: 3CNF-SAT to IND

Key step in the reduction

- For a clause $C_i = (x \vee y \vee z)$ in the 3-CNF formula φ, create 7 vertices.
- Each of 7 vertices corresponds to a satisfying partial assignment. That is, an assignment on $V(C_i)$ satisfying C_i.
- Incompatible assignments between two clauses is expressed as an edge.



Reduction: 3CNF-SAT to IND

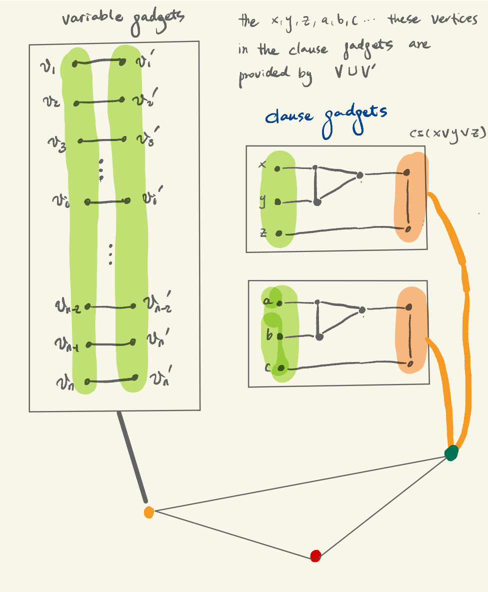
- Construction: set $k = |\varphi|$, i.e. the number of clauses in φ. The 7 vertices from $C \in \varphi$ form a clique.
- Easy to see that the construction can be done in polynomial time
- ($\Rightarrow$) If γ is a satisfying assignment to φ, then for each 3-clause C one can choose a vertex which corresponds to the assignment γ on the variables of C.
- The set of $|\varphi|$-many chosen vertices are independent.
- ($\Leftarrow$) If there is an independent set I of size $k = |\varphi|$, I has precisely one vertex out of the 7 vtxs.
- The set I encodes a consistent assignment (no edge), and they satisfy each $C \in \varphi$.

Reduction: 3CNF-SAT to 3-COL

A 3-Coloring of a graph $G = (V, E)$ is a mapping from V to $\{red, yellow, green\}$ such that each color class is an independent set.

Problem	3-Coloring
Input	a graph G.
Question	does G have 3-Coloring?

Reduction: 3CNF-SAT to 3-COL



Reduction: 3CNF-SAT to 3-COL

- The construction can be clearly done in polynomial time.
- ($\Rightarrow$) Let γ be a satisfying assignment to φ.
- Consider the following (partial) coloring c on V

$$c(v_i) = \begin{cases} green & \text{if } \gamma(x_i) = T \\ red & \text{if } \gamma(x_i) = F \end{cases}$$

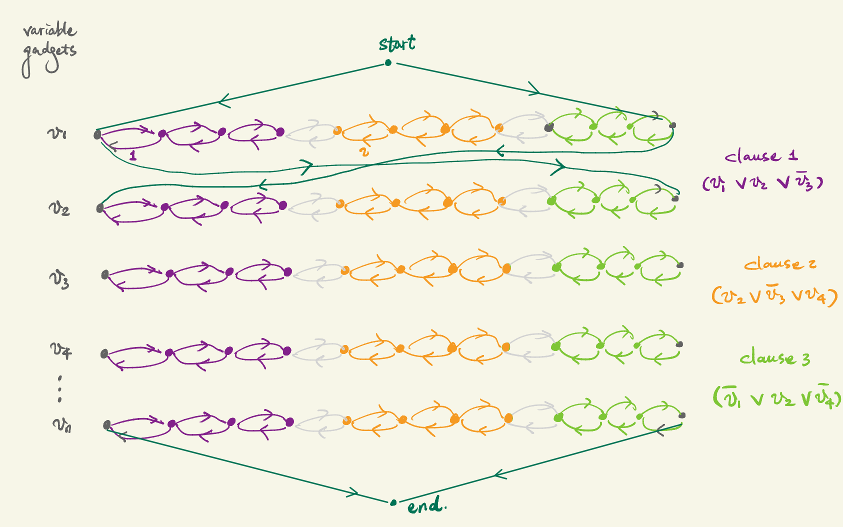
- Verify that c can be extended into a full 3-coloring.
- ($\Leftarrow$) Let $c : V(G_\varphi) \to \{green, red, yello\}$.
- Key obs: in each clause gadget, at least one literal vertex should be colored green.
- Consider the assignment $\alpha : V(\varphi) \to \{T, F\}$ s.t. $\alpha(x_i) = T$ if and only if $c(v_i) = green$.
- Due to *Key Obs*, this is a satisfying assignment.

Reduction: 3CNF-SAT to Directed Hamiltonian Path

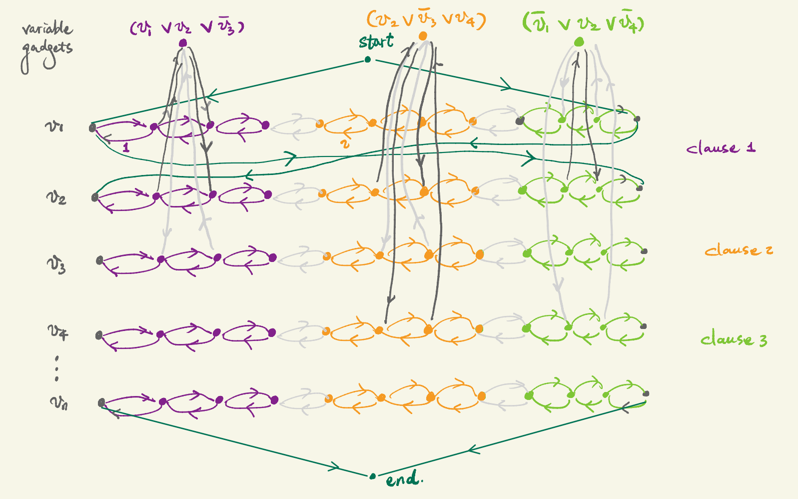
A Hamiltonian path of a directed graph $G = (V, E)$ is a directed path which visits every vertex of G precisely once.

Problem	Directed Hamiltonian Path
Input	a graph G.
Question	does G have a Hamiltonian path?

Reduction: 3CNF-SAT to Directed Hamiltonian Path



Reduction: 3CNF-SAT to Directed Hamiltonian Path



Reduction: 3CNF-SAT to Directed Hamiltonian Path

- The construction can be clearly done in polynomial time.
- ($\Rightarrow$) Let γ be a satisfying assignment to φ.
- Consider the following HAM path which zig-zags through the variable gadgets (and traverses clause vertices) in order by taking
  - a forward path if the variable is set to T,
  - a backward path if the variable is set to F.

Reduction: 3CNF-SAT to Directed Hamiltonian Path

- ($\Leftarrow$) Suppose the digraph G_φ has a HAM path P.
- Key obs: any HAM path must finish traversing the entire variable gadget before it visits another variable gadget!

Reduction: 3CNF-SAT to Directed Hamiltonian Path

- ($\Leftarrow$) Suppose the digraph G_φ has a HAM path P.
- Key obs: any HAM path must finish traversing the entire variable gadget before it visits another variable gadget!
- Consider the assignment $\alpha : V(\varphi) \to \{T, F\}$ s.t.
  - $\alpha(x_i) = T$ if P traverses the corresponding gadget forward.
  - $\alpha(x_i) = F$ if P traverses the corresponding gadget backward.
- α is well-defined due to Key obs 1. Why is α satisfying?

Reduction: 3CNF-SAT to Directed Hamiltonian Path

- ($\Leftarrow$) Suppose the digraph G_φ has a HAM path P.
- Key obs: any HAM path must finish traversing the entire variable gadget before it visits another variable gadget!
- Consider the assignment $\alpha : V(\varphi) \to \{T, F\}$ s.t.
  - $\alpha(x_i) = T$ if P traverses the corresponding gadget forward.
  - $\alpha(x_i) = F$ if P traverses the corresponding gadget backward.
- α is well-defined due to Key obs 1. Why is α satisfying?
- Key obs 2: for each clause vertex C_i, that $v(C_i)$ is traversed means that
  - if $v(C_i)$ is traversed during a 'forward' trip on gadget for variable x, then $\alpha(x) = T$ while x appears positively in C_i.
  - if $v(C_i)$ is traversed during a 'backward' trip on gadget for variable x, then $\alpha(x) = F$ while x appears negatively in C_i.
- Thanks to Key obs 2, at least one literal of each C_i is set to T.