Logic and Computation: I

Chapter 2 Propositional logic and computational complexity

Kazuyuki Tanaka

BIMSA

November 29, 2022

北京雁栖湖应用数学研究院
YANQI LAKE BEIJING INSTITUTE OF MATHEMATICAL SCIENCES AND APPLICATIONS

Logic and Computation I

- **Part 1. Introduction to Theory of Computation**
- **Part 2. Propositional Logic and Computational Complexity**
- **Part 3. First Order Logic and Decision Problems**

Part 2. Schedule

- Nov.17, (1) Tautologies and proofs
- Nov.22, (2) The completeness theorem of propositional logic
- Nov.24, (3) SAT and NP-complete problems
- Nov.29, (4) NP-complete problems about graphs
- Dec. 1, (5) Time-bound and space-bound complexity classes
- Dec. 6, (6) PSPACE-completeness and TQBF

NP-complete problems about graphs

Recap

- A Yes/No problem belongs to P if there exists a **deterministic** TM and a polynomial $p(x)$ s.t. for an input string of length n, it returns the correct answer within $p(n)$ steps.
- A problem belongs to NP if there is a **nondeterministic** TM and a polynomial $p(x)$ s.t. for an input string of length n, it always stops within $p(n)$ steps and answers
  - ▷ Yes, if at least one accepting computation process admits it;
  - ▷ No, if all the computation processes reject.
- Q_1 is polynomial (time) reducible to Q_2, denoted as $\mathrm{Q}_1 \leq_p \mathrm{Q}_2$, if there exists a polynomial-time algorithm A which solves a problem q_1 in Q_1 as problem $A(q_1)$ in Q_2.
- Q is NP-hard if for any NP problem Q′, $\mathrm{Q}' \leq_p \mathrm{Q}$.
- An NP-hard NP problem is said to be NP-complete.

Theorem

The Cook-Levin theorem: SAT is NP-complete.

We also showed the satifiablity problem SAT restricted to some special Boolean formulas remains NP-complete.

- A variable x and its negation $\neg x$ are called **literals**. A disjunction ($\vee$) of literals is called a **clause**. A conjunction ($\wedge$) of clauses is called a **CNF** (conjunctive normal form).
- **CNF-SAT** is the satisfiability problem for conjunctive normal forms.

Theorem

$\mathrm{CNF\text{-}SAT}$ is NP-complete.

- A CNF with exactly 3 literals in each clause is called a **3-CNF**. **3-SAT** is the satisfiability problem for 3-CNF.

Theorem

3-SAT is NP-complete.

Proof.

- To show $\mathrm{CNF\text{-}SAT} \leq_p \text{3-}\mathrm{SAT}$, let ϕ be a CNF formula.
- If ϕ has a clause $l_1 \vee \cdots \vee l_k (k \geq 4)$, replace it with the following:

  $(l_1 \vee l_2 \vee x_1) \wedge (l_3 \vee \bar{x}_1 \vee x_2) \wedge (l_4 \vee \bar{x}_2 \vee x_3) \wedge \cdots \wedge (l_{k-2} \vee \bar{x}_{k-4} \vee x_{k-3}) \wedge (l_{k-1} \vee l_k \vee \bar{x}_{k-3})$

  where $\bar{x}$ represents $\neg x$.
- For a clause with only one literal l_1, replace it with

$$(l_1 \vee x_1 \vee x_2) \wedge (l_1 \vee x_1 \vee \bar{x}_2) \wedge (l_1 \vee \bar{x}_1 \vee x_2) \wedge (l_1 \vee \bar{x}_1 \vee \bar{x}_2).$$

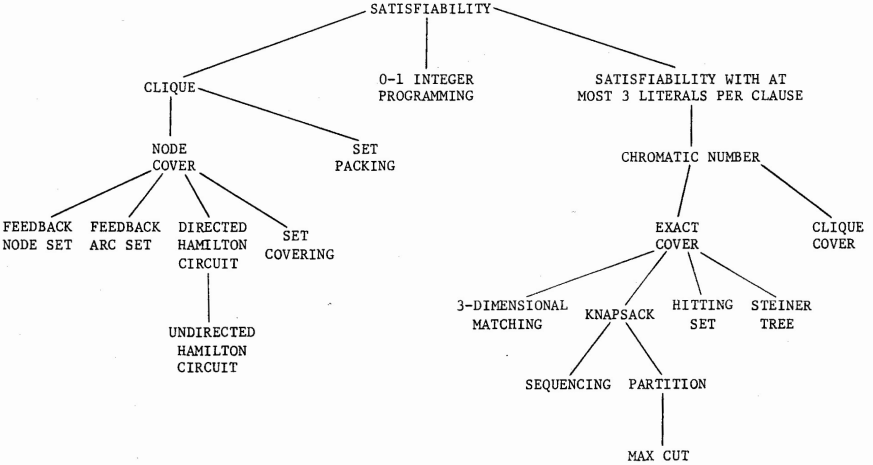
- For a clause with only two literals $l_1 \vee l_2$, replace it with

$$(l_1 \vee l_2 \vee x_1) \wedge (l_1 \vee l_2 \vee \bar{x}_1).$$

- It is easy to see that the satisfiability condition does not change by these transformations.
- Since the above transformations can be performed by polynomial-time algorithms, $\mathrm{CNF\text{-}SAT}$ is polynomial-time reducible to 3-SAT.

- Following Cook's work on SAT, in 1972 R. Karp published a list of 21 NP-complete problems. In addition to 3-SAT, there are Hamiltonian cycle problem, graph coloring problem, knapsap problem.
- Today, we focus on vertex cover and Hamiltonian cycle problems.

Richard Karp



Source: R. Karp (1972) Reducibility Among Combinatorial Problems

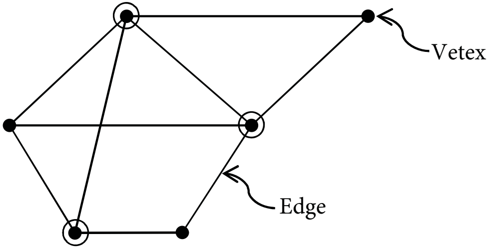
Definition

A **directed graph** $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq V \times V$. A graph such that $(u, v) \in E \Leftrightarrow (v, u) \in E$ is called an **undirected graph**.

Here we only consider finite graphs.

Definition

A **vertex cover** of an undirected graph $G = (V, E)$ is a subset $S \subseteq V$ of vertices such that for any edge $(u, v) \in E$ of G, $u \in S$ or $v \in S$.



The set of $\circledcirc$ is the vertex cover. Every edge has one endpoint in the vertex cover.

Definition

- The **vertex cover problem VC**: Given an undirected graph G and a natural number k, decide whether there exists a vertex cover S of G consisting of k vertices.
- The problem of finding the minimum vertex cover size k for an undirected graph G is called **minimum vertex cover problem**.

Theorem

The vertex cover problem for undirected graphs is NP-complete.

Consider to input $G = (V, E)$ to the TM

- If the cardinality of V is n, E can be represented by an $n \times n$ matrix with components $0, 1$, which is called **adjacency matrix**. Then graph G can be represented by a $0, 1$ sequence of length n^2.
- Thus, a polynomial size of the graph G can be regarded as a polynomial of V.

Proof.

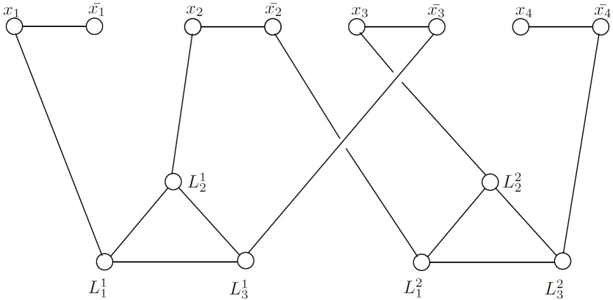
VC is an NP problem

- Choose a arbitrary set S of k vertices and check whether it is a vertex cover.
- It is easy to decide (in polynomial time) whether S is a vertex cover, since we only need to check that for each edge, one of its endpoints belongs to S.

3-SAT $\leq_p$ VC

- Consider a 3-CNF formula $\varphi = \bigwedge_{j \leq m} (l_1^j \vee l_2^j \vee l_3^j)$.
- Let $\{x_1, \ldots, x_n\}$ be the variables in φ. That is, l_1^j $(j \leq m, s \leq 3)$ is x_i or $\overline{x_i}$.
- Then construct the graph $G = (V, E)$ such that
  - $V = \{x_i, \bar{x}_i : i \leq n\} \cup \{L_1^j, L_2^j, L_3^j : j \leq m\}$,
  - $E = \{(x_i, \bar{x}_i) : i \leq n\} \cup \{(L_1^j, L_2^j), (L_2^j, L_3^j), (L_3^j, L_1^j) : j \leq m\}$
    $\cup \{(l_s^j, L_s^j) : j \leq m, s \leq 3\}$.

Proof. (continued) Consider a graph for $\varphi = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4)$:



- Check whether there is a v.c. S of size $k = n + 2m$ for the graph thus constructed ($k = 8$ for φ).
- A v.c. must contain x_i or $\bar{x}_i$ for each $i \leq n$ and at least two of L_1^j, L_2^j, L_3^j. for each $j \leq m$.
- Hence, a v.c. of size $k = n + 2m$ will contain exactly one of x_i or $\bar{x}_i$ and exactly two of L_1^j, L_2^j, L_3^j.
- Namely, one of L_1^j, L_2^j, L_3^j is not in S, and it must connect with either x_i or $\bar{x}_i$ in S.
- Now, put $V(x_i) = \mathrm{T}$ if $x_i \in S$, and $V(x_i) = \mathrm{F}$; otherwise, $V(l_1^j \vee l_2^j \vee l_3^j) = \mathrm{T}$ for all j and so $V(\varphi) = \mathrm{T}$.
- Conversely, suppose there is a V such that $V(\varphi) = \mathrm{T}$. We construct a v.c. S of size $n + 2m$. First, put x_i (or $\bar{x}_i$) into S if $V(x_i) = \mathrm{T}$ (or $V(\bar{x}_i) = \mathrm{T}$). Then at least one of L_1^j, L_2^j, L_3^j connects to x_i or $\bar{x}_i$ in S. Except one of such, put the other two in S.
- In short, φ is satisfiable $\Leftrightarrow$ the graph has a vertex cover of size $k = n + 2m$
- That is, 3-SAT is polynomial-time reducible to VC.

Hamiltonian cycles

Only finite connected graphs are considered.

Recall: Eulerian cycles in an undirected graph

- A **Eulerian path** passes through every edge exactly once. A **Eulerian cycle** is a Eulerian path whose start and end points coincide.
- An Eulerian cycle exists iff the degree of each vertex is even.

- A **Hamiltonian cycle** is a cycle passing through every vertex exactly once.
- There is no known simple criterion for the existence of Hamiltonian path.
- R. Karp showed that this problem is NP-complete and clarified it is difficult in principle to find such a criterion.

Definition

Directed Hamiltonian cycle problem (dHAMCYCLE): for a directed connected graph, decide whether there is a Hamiltonian cycle (passing every vertex exactly once, following the direction of the edges)

Theorem

dHAMCYCLE is NP-complete.

Proof. To show dHAMCYCLE $\in$ NP, choose an arbitrary path of the directed graph and check whether it is a Hamiltonian cycle.

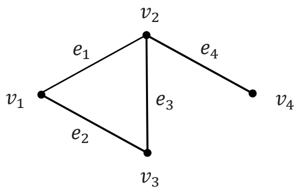
For its NP-completeness, we prove that VC is polynomial reducible to it.

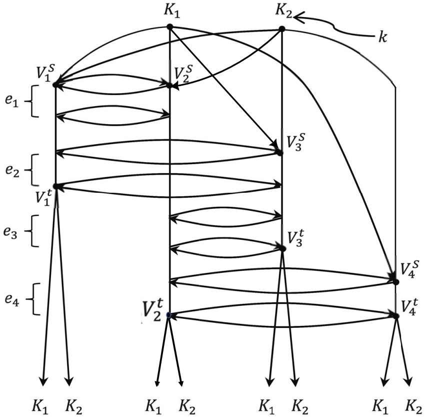
- Assume an undirected graphs $G = (V, E)$ and k. We construct a directed graph $G^* = (V^*, E^*)$.
- We show the following are equivalent:
  - ▷ G^* has a Hamiltonian cycle.
  - ▷ G has a vertex cover of size k.

Proof. ($\mathrm{VC}\leq_p\mathrm{dHAMCYCLE}$, continued) Consider the graph G in the right figure with $k=2$.
The construction of G^*:

- ▷ For $k=2$, two points K_1, K_2 are fixed at the top.
- ▷ Since G consists of 4 vertices $v_i (i=1,2,3,4)$, 4 downward lines $\overrightarrow{V_i^s V_i^t}$ are drawn.
- ▷ If there is an edge between v_i and v_j in G, then a pair of double bridges are built so that it can go back and forth between $\overrightarrow{V_i^s V_i^t}$ and $\overrightarrow{V_j^s V_j^t}$.
- ▷ Finally, draw a line from each K_l to each V_i^s and from each V_i^t to each K_l.

- In a Hamiltonian cycle on G^*, if a vertex V_i^s has an edge from a point K_l, then v_i is in the vertex cover of G.
- Since there are only $m=2$ upper points K_l, there are exactly m such v_i's, i.e., the size of the vertex cover corresponding to the Hamiltonian cycle is always 2.



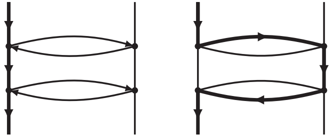


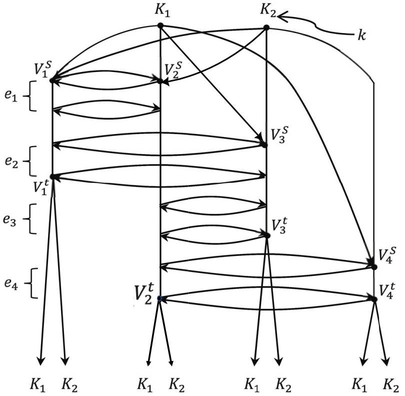
Proof. (VC$\leq_p$dHAMCYCLE, continued)

- From the selected vertex V_i^s, go down toward V_i^t in one of the two ways shown in the right figures (straight and detour).
- Consider a Hamiltonian cycle C entering V_i^s from K_1.
- If C does not include the edge from K_2 to $V_j^s (j \neq i)$ and there is a bridge between $\overline{V_i^s V_i^t}$ and $\overline{V_j^s V_j^t}$, C detours from $\overline{V_i^s V_i^t}$ to $\overline{V_j^s V_j^t}$ and returns to $\overline{V_i^s V_i^t}$. Otherwise, go down straight.
- If C makes a detour, just one end of the corresponding edge belongs to the vertex cover; otherwise, both endpoints are in the vertex cover.
- In any case, if G^* has a Hamiltonian cycle, the set of vertices v_i such that V_i^s connects with some K_l is a vertex cover of k vertices.

That is,

G has a vertex cover of size $k \Leftrightarrow G^*$ has a Hamiltonian cycle.





Proof. ($\mathrm{VC} \leq_p \mathrm{dHAMCYCLE}$, continued)

- The above argument can be generalized to any graphs. We omit this routine work.
- Although G^* looks much larger than G, it can be obtained by a polynomial-time algorithm. In fact, the number of vertices of G^* is a constant multiple of the number of edges of G.
- That is, VC is polynomial-time reducible to $\mathrm{dHAMCYCLE}$.

From this result, we can also show that decision problem on the existence of Hamiltonian cycles for undirected graphs is NP-complete.

Homework

Show that the decision problem of the existence of Hamiltonian cycles for undirected graphs (HAMCYCLE) is NP-complete.

- The **Traveling Salesman Problem** (TSP) is a variation of the Hamiltonian cycle problem.
  - A weight (distance) assigned for each edge of an undirected graph.
  - Is it possible to traverse all the points so that the sum of the weights of the passed edges does not exceed a given limit k?
  - Equivalently, does there exist a Hamiltonian cycle such that the sum of edge weights is less than or equal to k?

It can be shown that $\mathrm{TSP}$is also NP-complete.

- For TSP to be NP, choose an arbitrary path and check whether it satisfies the condition or not.
- For the reversal, the existence of a Hamiltonian cycle is the existence of a TSP solution with edge weight 1 and sufficiently large k, and so

$$\mathrm{HAMCYCLE} \leq_p \mathrm{TSP}.$$

Summary

- We have shown that the vertex cover problem VC and the directed Hamiltonian cycle problem dHAMCYCLE are NP-complete.

Further readings

M. Sipser, Introduction to the Theory of Computation, 3rd ed., Course Technology, 2012

Thank you for your attention!