

NP-Completeness

(Based on [Manber 1989])

Yih-Kuen Tsay

Department of Information Management
National Taiwan University

P vs. NP

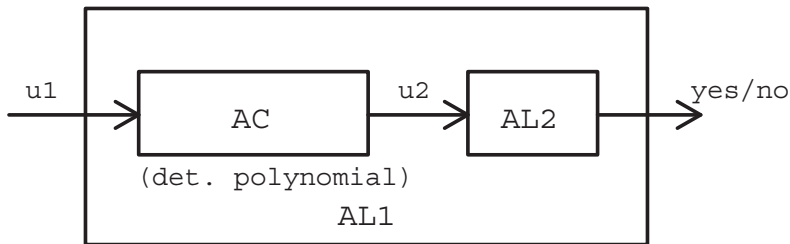
- 🌐 P denotes the class of all problems that can be solved by *deterministic* algorithms in *polynomial* time.
- 🌐 NP denotes the class of all problems that can be solved by *nondeterministic* algorithms in *polynomial* time.
- 🌐 A *nondeterministic* algorithm, when faced with a choice of several options, has the power to *guess* the right one (if there is any).
- 🌐 We will focus on *decision* problems, whose answer is either yes or no.

Decision as Language Recognition

- 🌐 A *decision* problem can be viewed as a *language-recognition* problem.
- 🌐 Let U be the set of all possible inputs to the decision problem and $L \subseteq U$ be the set of all inputs for which the answer to the problem is yes.
- 🌐 We call L the *language* corresponding to the problem.
- 🌐 The decision problem is to *recognize* whether a given input belongs to L .

Polynomial-Time Reductions

- Let L_1 and L_2 be two languages from the input spaces U_1 and U_2 .
- We say that L_1 is *polynomially reducible* to L_2 if there exists a **conversion** algorithm AC satisfying the following conditions:
 - AC runs in **polynomial time** (deterministically).
 - $u_1 \in L_1$ if and only if $AC(u_1) = u_2 \in L_2$.



Polynomial-Time Reductions (cont.)

Theorem (11.1)

If L_1 is polynomially reducible to L_2 and there is a polynomial-time algorithm for L_2 , then there is a polynomial-time algorithm for L_1 .

Theorem (11.2: transitivity)

If L_1 is polynomially reducible to L_2 and L_2 is polynomially reducible to L_3 , then L_1 is polynomially reducible to L_3 .

NP-Completeness

- 🌐 A problem X is called an **NP-hard** problem if every problem in NP is polynomially reducible to X .
- 🌐 A problem X is called an **NP-complete** problem if (1) X belongs to NP, and (2) X is NP-hard.

Lemma (11.3)

A problem X is an NP-complete problem if (1) X belongs to NP, and (2') Y is polynomially reducible to X , for some NP-complete problem Y .

- 🌐 If there exists an efficient (polynomial-time) algorithm for any NP-complete problem, then there exist efficient algorithms for all NP-complete (and hence all NP) problems.

The Satisfiability Problem (SAT)

Problem

Given a Boolean expression in conjunctive normal form, determine whether it is satisfiable.

- 🌐 A Boolean expression is in *conjunctive normal form* (CNF) if it is the product of several sums, e.g.,
 $(x + y + \bar{z}) \cdot (\bar{x} + y + z) \cdot (\bar{x} + \bar{y} + \bar{z})$.
- 🌐 A Boolean expression is said to be *satisfiable* if there exists an assignment of 0s and 1s to its variables such that the value of the expression is 1.

Theorem (Cook's Theorem)

The SAT problem is NP-complete.

- 🌐 This is our starting point for showing the NP-completeness of some other problems.
- 🌐 Their NP-hardness will be proved by **reduction directly or indirectly from SAT**.

NP-Complete Problems

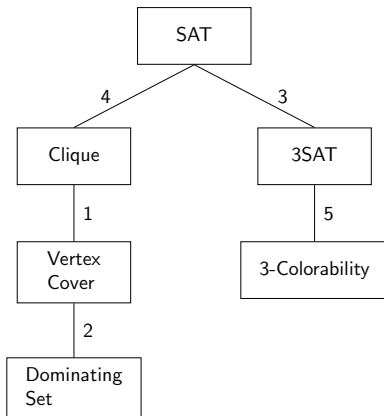


Figure: The order of NP-completeness proofs we will study.

Source: redrawn from [Manber 1989, Figure 11.1].

Vertex Cover

Problem

Given an undirected graph $G = (V, E)$ and an integer k , determine whether G has a vertex cover containing $\leq k$ vertices.

A *vertex cover* of G is a set of vertices such that every edge in G is incident to at least one of these vertices.

Theorem (11.4)

The vertex-cover problem is NP-complete.

Main idea: by reduction from the *clique* problem.

Vertex Cover (cont.)

Proof outline:

- 🌐 The vertex-cover problem is in NP, since given a graph we can guess a subset of vertices and check whether it contains $\leq k$ vertices and is indeed a vertex cover in polynomial time.
- 🌐 The clique problem, which is NP-complete, is polynomially reducible to the vertex-cover problem.
 - ☀️ Let $G(V, E)$ and k represent an arbitrary instance of the clique problem.
 - ☀️ Let $\overline{G}(V, \overline{E})$ be the complement of G ; computing the complement of a graph takes only polynomial time.
 - ☀️ Claim: G has a clique of size $\geq k$ iff \overline{G} has a vertex cover of size $\leq |V| - k$.

Dominating Set

Problem

Given an undirected graph $G = (V, E)$ and an integer k , determine whether G has a dominating set containing $\leq k$ vertices.

A *dominating set* D is a set of vertices such that every vertex of G is either in D or is adjacent to some vertex in D .

Theorem (11.5)

The dominating-set problem is NP-complete.

By reduction from the *vertex-cover* problem.

Dominating Set (cont.)

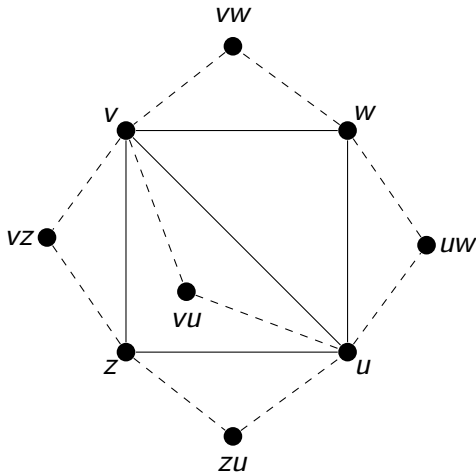


Figure: The dominating-set reduction.

Source: redrawn from [Manber 1989, Figure 11.2].

Problem

Given a Boolean expression in CNF such that each clause contains exactly three variables, determine whether it is satisfiable.

Theorem (11.6)

The 3SAT problem is NP-complete.

By reduction from the regular SAT problem.

3SAT (cont.)

From an arbitrary clause $(x_1 + x_2 + \cdots + x_k)$, where $k \neq 3$, of the SAT problem to clauses of the 3SAT problem:

When $k \geq 4$,

$$\begin{aligned} &(x_1 + x_2 + y_1) \cdot \\ &(x_3 + \bar{y}_1 + y_2) \cdot \\ &(x_4 + \bar{y}_2 + y_3) \cdot \\ &\quad \vdots \\ &(x_{k-2} + \bar{y}_{k-4} + y_{k-3}) \cdot \\ &(x_{k-1} + x_k + \bar{y}_{k-3}) \end{aligned}$$

When $k = 2$,

$$(x_1 + x_2 + w) \cdot (x_1 + x_2 + \bar{w})$$

When $k = 1$,

$$(x_1 + y + z) \cdot (x_1 + \bar{y} + z) \cdot (x_1 + y + \bar{z}) \cdot (x_1 + \bar{y} + \bar{z})$$

Clique

Problem

Given an undirected graph $G = (V, E)$ and an integer k , determine whether G contains a clique of size $\geq k$.

A *clique* C is a subgraph of G such that all vertices in C are adjacent to all other vertices in C .

Theorem (11.7)

The clique problem is NP-complete.

By reduction from the SAT problem.

Clique (cont.)

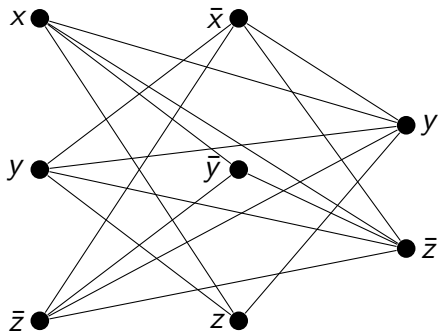


Figure: An example of the clique reduction for the expression $(x + y + \bar{z}) \cdot (\bar{x} + \bar{y} + z) \cdot (y + \bar{z})$.

Source: redrawn from [Manber 1989, Figure 11.3].

3-Coloring

Problem

Given an undirected graph $G = (V, E)$, determine whether G can be colored with three colors.

Theorem (11.8)

The 3-coloring problem is NP-complete.

By reduction from the 3SAT problem.

3-Coloring (cont.)

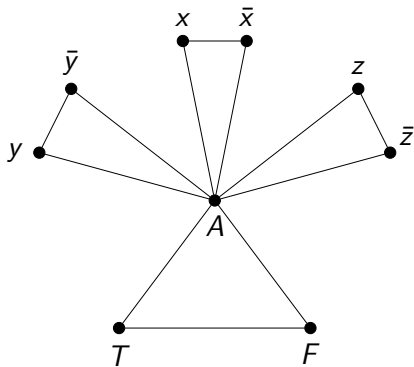


Figure: The first part of the construction in the reduction of 3SAT to 3-coloring.

Source: redrawn from [Manber 1989, Figure 11.4].

3-Coloring (cont.)

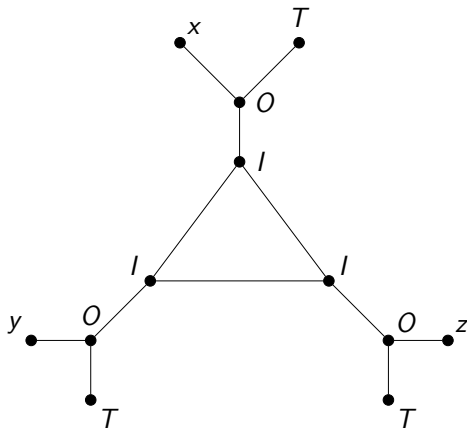


Figure: The subgraphs corresponding to the clauses in the reduction of 3SAT to 3-coloring.

Source: redrawn from [Manber 1989, Figure 11.5].

3-Coloring (cont.)

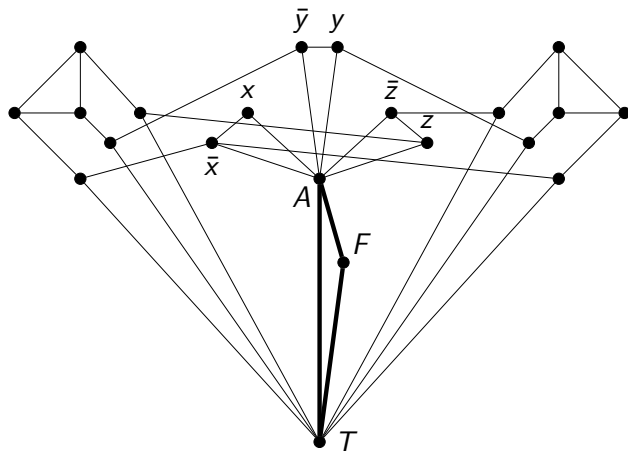


Figure: The graph corresponding to $(\bar{x} + \bar{y} + z) \cdot (\bar{x} + y + \bar{z})$.

Source: redrawn from [Manber 1989, Figure 11.6].

More NP-Complete Problems

Independent set:

An independent set in an undirected graph is a set of vertices no two of which are adjacent. The problem is to determine, given a graph G and an integer k , whether G contains an independent set with $\geq k$ vertices.

Hamiltonian cycle:

A Hamiltonian cycle in a graph is a (simple) cycle that contains each vertex exactly once. The problem is to determine whether a given graph contains a Hamiltonian cycle.

Travelling salesman:

The input includes a set of cities, the distances between all pairs of cities, and a number D . The problem is to determine whether there exists a (travelling-salesman) tour of all the cities having total length $\leq D$.

More NP-Complete Problems (cont.)

Partition:

The input is a set X where each element $x \in X$ has an associated size $s(x)$. The problem is to determine whether it is possible to partition the set into two subsets with exactly the same total size.

Knapsack:

The input is a set X , where each element $x \in X$ has an associated size $s(x)$ and value $v(x)$, and two other numbers S and V . The problem is to determine whether there is a subset $B \subseteq X$ whose total size is $\leq S$ and whose total value is $\geq V$.

Bin packing:

The input is a set of numbers $\{a_1, a_2, \dots, a_n\}$ and two other numbers b and k . The problem is to determine whether the set can be partitioned into k subsets such that the sum of numbers in each subset is $\leq b$.