Algorithm Design XVI

NP Problem II

Guoqiang Li
School of Software

SHANGHAI JIAO TONG UNIVERSITY

P and NP Problems

Set Cover

Set Cover

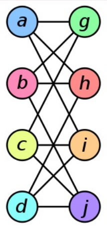
- Input: A set of elements B, sets $S_1, \ldots, S_m \subseteq B$
- Output: A selection of the S_i whose union is B.
- Cost: Number of sets picked.

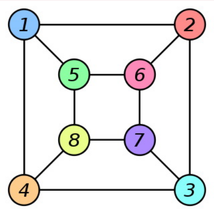
Graph Isomorphism

An isomorphism of graphs G and H is a bijection between the vertex sets of G and H

$$f : V(G) \to V(H)$$

such that any two vertices u and v of G are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in H.





Hard Problems, Easy Problems

Hard problems (NP-complete)	Easy problems (in P)
3SAT	2SAT, Horn SAT
Traveling salesman problem	Minimum spanning tree
Longest path	Shortest path
3D matching	Bipartite matching
Knapsack	Unary knapsack
Independent set	Independent set on trees
Integer linear programming	Linear programming
Rudrata path	Euler path
Balanced cut	Minimum cut

if A search problem satisfies:

1. there exists an efficient checking algorithm C, taking as input the given `instance` I, a `solution` S, and outputs `true` iff S is a solution I.
2. The running time of $C(I, S)$ is bounded by a polynomial in $|I|$.

We denote the class of all such problems by NP.

P

An algorithm that takes as input an `instance` I and has a running time polynomial in $|I|$.

- I has a solution, the algorithm returns such a solution;
- I has no solution, the algorithm correctly reports so.

The class of all search problems that can be solved in polynomial time is denoted P.

Why P and NP

P: polynomial time

NP: nondeterministic polynomial time

Complementation

A class of problems $\mathcal{C}$ is closed under complementation if for any problem in $\mathcal{C}$, its complement is also in $\mathcal{C}$.

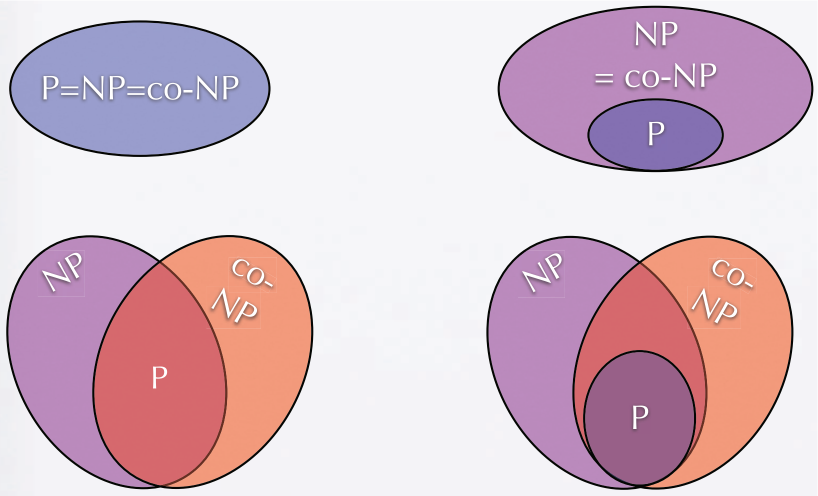
P: is closed under complementation.

NP ?

Example (Complementation of TSP)

Given n cities with their intercity distances, is it the case that there does not exist any tour length k or less?

Conjectures



P $\neq$ NP

Theorem Proving

- Input: A mathematical statement φ and n.
- Problem: Find a proof of φ of length $\leq n$ if there is one.

A formal proof of a mathematical assertion is written out in excruciating detail, it can be checked mechanically, by an efficient algorithm and is therefore in NP.

So if P $=$ NP, there would be an efficient method to prove any theorem, thus eliminating the need for mathematicians!

Solve One and All Solved

Even if we believe $\mathsf{P} \neq \mathsf{NP}$, can we find an evidence that these particular problems have no efficient algorithm?

Such evidence is provided by reductions, which translate one search problem into another.

We will show that the hard problems in previous lecture exactly the same problem, the hardest search problems in NP.

If one of them has a polynomial time algorithm, then every problem in NP has a polynomial time algorithm.

Reduction

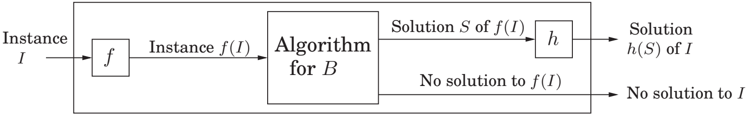
Reduction Between Search Problems

A reduction from A to B is a polynomial time algorithm f that transforms any instance I of A into an instance $f(I)$ of B

Together with another polynomial time algorithm h that maps any solution S of $f(I)$ back into a solution $h(S)$ of I.

If $f(I)$ has no solution, then neither does I.

These two translation procedures f and h imply that any algorithm for B can be converted into an algorithm for A.



The Two Ways to Use Reductions

Assume there is a reduction from a problem A to a problem B.

$$A \to B$$

- If we can solve B efficiently, then we can also solve A efficiently.
- If we know A is hard, then B must be hard too.

If $A \to B$ and $B \to C$, then $A \to C$.

NP-Completeness

NP-Completeness

Definition

A NP problem is NP-complete if all other NP problems reduce to it.

Reductions to NP-Complete

NP-complete problems are hard: all other search problems reduce to them.

For a problem to be NP-complete, it can solve every NP problem in the world.

If even one NP-complete problem is in P, then P = NP.

If a problem A is NP-complete, a new NP problem B is proved to be NP-complete, by reducing A to B.

Co-NP-Completeness

Definition

A co-NP problem is co-NP-complete if all other co-NP problems reduce to it.

A problem is NP-complete if and only if its complement is co-NP-complete.

If a problem and its complement are NP-complete then co-NP = NP.

TAUTOLOGY

SHANGHAI JIAO TONG UNIVERSITY

TAUTOLOGY

A CNF formula f is unsatisfiable if and only if its negation is a TAUTOLOGY. The negation of a CNF formula can be converted into a DNF formula. The resulting DNF formula is a TAUTOLOGY if and only if the negation of the CNF formula is a tautology.

The problem TAUTOLOGY: Given a formula f in DNF, is it a tautology?

- TAUTOLOGY is in P if and only if co-NP = P, and
- TAUTOLOGY is in NP if and only if co-NP = NP.

Factoring

SHANGHAI JIAO TONG UNIVERSITY

The difficulty of FACTORING is of a different nature than that of the other hard search problems we have just seen.

Nobody believes that FACTORING is NP-complete.

One evidence is that a number can always be factored into primes.

Another difference: FACTORING succumbs to the power of quantum computation, while SAT, TSP and the other NPC problems do not seem to.

Primarily and Composite

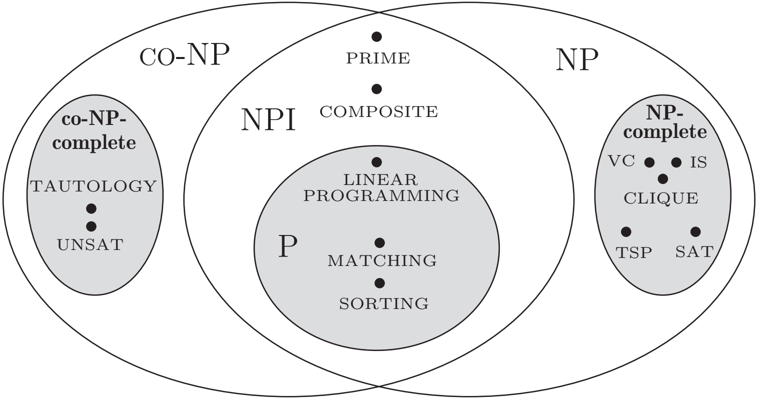
PRIMARILY

Given an integer $k \geq 2$, is k a prime number?

COMPOSITE

Given an integer $k \geq 4$, are there two integers $p, q \geq 2$ such that $k = pq$?

NPI (A Problematic Category)



NP-Intermediate

SHANGHAI JIAO TONG UNIVERSITY

Definition (NPI)

Problems that are in the complexity class NP but are neither in the class P nor NP-complete are called NP-intermediate, and the class of such problems is called NPI.

Theorem (Lander Theorem)

If $P \neq NP$, then NPI is not empty; that is, NP contains problems that are neither in P nor NP-complete.

Quiz

Prove that if NP $\neq$ Co-NP, then P $\neq$ NP.