Combinatorial optimization for undergraduate students

Lecture note 3. NP-completeness

Lecturer : O-joung Kwon
Spring, 2018

Here are two problems :

1) Hamiltonian Cycle
Given a graph G, does G have a Hamiltonian Cycle?

2) Independent Set
Given a graph G, find a maximum size of a subset of vertices (called an independent set) that have no edges between them.

The problem of type 1) is called a *decision* problem, while the problem of type 2) is called an *optimization* problem.

For an optimization such an Independent Set, we may consider its decision version : given a graph G and an integer k, does G contain an independent set of size at most k?

Observation : if we can solve this decision version in polynomial time, then we can solve the original optimization version in polynomial time, by looking at all possible values of k (1 to $|V(G)|$).

So we can say that an optimization problem is at least as hard as the corresponding decision problem.

We define important classes of problems.

P : the class of all polynomial decision problems

NP : the class of decision problems for which a positive answer can be verified in polynomial time. This means that we require the specification of a certificate enabling us to verify the correctness of a positive answer in polynomial time.
Caution : It only requires for a positive answer, not a negative answer.

Example : in the Hamiltonian Cycle problem, if one give me some subgraph (claiming that it is a solution), then I can check whether it is indeed a hamiltonian cycle or not. First check that it is a cycle, this can be done by checking that it is connected, and every vertex has degree 2. Secondly, we check whether it contains all vertices.

Example : Consider a problem that ask whether a given graph is 'not' an Eulerian graph. Then the certificate would be any vertex of odd degree.

Co-NP : the class of decision problems for which a negative answer can be verified in polynomial time.

Observation : $P \subseteq NP \cap$ Co-NP. Why?

One of 7 millenium problems in mathematics :
Is $P = NP$?

A decision problem Q is called an *NP-complete* if

- Q is in NP,
- the polynomial time solvability of Q would imply that all other problems in NP are solvable in polynomial time as well.

Important! : NP-completeness is a very strong condition: if we could find a polynomial time algorithm for an NP-complete problem, then it implies that P = NP.

It is not obvious whether there is at least one NP-complete problem.

A celebrated result by Cook :

Theorem 1 (Cook's theorem, 71)**.** *SAT and 3-SAT is NP-complete.*

SAT (satisfiability) problem : Let $x_1, \ldots, x_n$ be Boolean variables that take values true or false. Let $C_1, C_2, \ldots, C_m$ be such that each C_i is of the form $(y_{i_1} \vee y_{i_2} \vee \cdots \vee y_{i_q})$ where $y_{i_j} = x_{i_j}$ or $\overline{x_{i_j}}$ (negation of x_{i_j}). And then we take the formula $C_1 \wedge C_2 \wedge \cdots \wedge C_m$. The problem asks whether there is a possible combination of values for x_i's saying that the formula is true. If each C_i consists of 3 literals, then we say that it is 3-SAT problem.

We skip its proof since it is not the main focus of the lecture.

A vertex subset S is called a *vertex cover* of G if $G - S$ have no edges; in other words, all edges are incident with a vertex in S.

Vertex Cover : Given a graph G and an integer k, does G contain a vertex cover of size at most k?

Theorem 2. *Vertex Cover is NP-complete.*