

Lecture 33: NP-Completeness

Agenda:

- Classes P , NP (recall), and $co-NP$
- The notions of hardest and complete
- Polynomial-time reduction
- NP-completeness proof steps

Reading:

- Textbook pages 979 – 995

The classes of P and NP (recall):

- Class P
 - decision problem
 - there exists some algorithm solving the problem in polynomial time
 - which of the above problems are in P :
 1. sorting
 2. longest common subsequence
 3. minimum spanning tree
 4. single-source shortest paths
 5. shortest x -to- y path
 6. determining Eulerian graphs
- Class NP
 - decision problem
 - for every **yes**-instance, there exists a proof that the answer is yes and the proof can be verified in polynomial time
 - which of the above problems are in NP :
 1. sorting
 2. longest common subsequence
 3. minimum spanning tree
 4. single-source shortest paths
 5. shortest x -to- y path
 6. determining Eulerian graphs
 7. longest x -to- y path
 8. determining Hamiltonian graphs

Some well-known problems in NP :

- k -clique:
Given a graph, does it have a size k clique? (i.e. complete subgraph on k vertices)
- k -independent set:
Given a graph, does it have a size k independent set? (i.e. k vertices with no edge between them)
- k -coloring:
Given a graph, can the vertices be colored with k colors such that adjacent vertices get different colors?
- Satisfiability (SAT):
Given a boolean expression, is there an assignment of truth values (T or F) to variables such that the expression is satisfied (evaluated to T)?
- Travel salesman problem (TSP):
Given an edge-weighted complete graph and an integer ℓ , is there a Hamiltonian cycle of length at most ℓ ?
- Hamiltonian Path (or Hamiltonian Cycle): Given a graph G does it have a Hamiltonian path? (i.e. a path that goes through every vertex exactly once)

Completeness and Reduction:

- For the class NP of problems, which one is the hardest?

Reasons to ask:

- Is every problem in NP solvable in polynomial time?
- If not, what are the characteristics of the hard problems?

- Hardest problem — solving it means you can solve every other in NP

- Informally, complete = hardest;

Formally, we need “polynomial-time reduction”:

- decision problem Π_1 is polynomial-time reducible to Π_2 , written as

$$\Pi_1 \leq_p \Pi_2,$$

if

- $\exists$ polynomial-time transformation function t which
- maps instances of Π_1 to instances of Π_2 , such that
- for every instance x of Π_1 , the answer to x is the same as the answer to $t(x)$

- Formally, $\Pi \in NP$ is **NP-complete** if every other problem Π' is polynomial-time reducible to Π : $\Pi' \leq_p \Pi$

Examples of (polynomial-time) reduction:

1. k -independent set is reducible to k -clique

Proof. From a k -independent set instance (G, k) , construct a k -clique instance to be $(\overline{G}, k)$, where $\overline{G}$ is the complement graph of G .

The construction takes $\Theta(n^2)$ time for every instance (G, k) containing n vertices and thus polynomial --- take this construction as the transformation function.

Now, X is an independent set in G iff it is a clique in $\overline{G}$.
Proof done.

2. 3-SAT is reducible to SAT

A trivial exercise.

3. SAT is reducible to 3-SAT

A non-trivial exercise.

Proof of NP-completeness:

- Suppose we want to prove Π is NP-complete, then we need to prove
 1. $\Pi \in NP$
 2. for every problem $\Pi' \in NP$, Π' is reducible to Π
- Question:
 1. How many problems in NP? — ∞ ??!! inefficient way

Proof of NP -completeness:

- Observations:
 - $\Pi \in NP$
 - Π' is NP -complete and is reducible to Π
 - $\implies \Pi$ is NP -complete !
- Why it is true:
 - for every problem $\Pi'' \in NP$
 - since Π' is NP -complete, Π'' is reducible to Π'
 - using the transitivity of reduction
 - Π'' is reducible to Π
- Switching our goal to:
 - Prove that $\Pi \in NP$
 - Look for an NP -complete problem Π' and
 - Prove that Π' is reducible to Π
- Where to start with — we need a first NP -complete problem

Cook's Theorem: SAT is NP -complete.

Lecture 33: NP-Completeness

Have you understood the lecture contents?

well	ok	not-at-all	topic
☐	☐	☐	classes P , NP , $\text{co-}NP$
☐	☐	☐	relationships among the 3 classes
☐	☐	☐	NP -complete
☐	☐	☐	polynomial time reduction
☐	☐	☐	steps of NP -completeness proof