ShanghaiTech University

CS101 Algorithms and Data Structures

Fall 2021

Homework 12

Due date: 23:59, January 3, 2022

1. Please write your solutions in English.
2. Submit your solutions to gradescope.com.
3. Set your FULL NAME to your Chinese name and your STUDENT ID correctly in Account Settings.
4. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
5. When submitting, match your solutions to the according problem numbers correctly.
6. No late submission will be accepted.
7. Violations to any of the above may result in zero grade.
8. In this homework, all the proofs need three steps. The demands and an example are given on the next page. If you do not organize your answer in the standard format, you will not get any point.

Demand of the NP-complete Proof

When proving problem A is NP-complete, please clearly divide your answer into three steps:

1. Prove that problem A is in NP.
2. Choose an NP-complete problem B and for any B instance, construct an instance of problem A.
3. Prove that the yes/no answers to the two instances are the same.

Proof Example

Suppose you are going to schedule courses for the SIST and try to make the number of conflicts no more than K. You are given 3 sets of inputs: $C = \{\cdots\}, S = \{\cdots\}, R = \{\{\cdots\}, \{\cdots\}, \cdots\}$. C is the set of distinct courses. S is the set of available time slots for all the courses. R is the set of requests from students, consisting of a number of subsets, each of which specifies the course a student wants to take. A conflict occurs when two courses are scheduled at the same slot even though a student requests both of them. Prove this schedule problem is NP-complete.

1. Firstly, for any given schedule as a certificate, we can traverse every student's requests and check whether the courses in his/her requests conflicts and count the number of conflicts, and at last check if the total number is fewer than K, which can be done in polynomial time. Thus the given problem is in NP.
2. We choose 3-coloring problem which is a NP-complete problem. For any instance of 3-coloring problem with graph G, we can construct an instance of the given problem: let every node v becomes a course, thus construct C; let every edge (u, v) becomes a student whose requests is $\{u, v\}$, thus construct R; let each color we use becomes a slot, thus construct S; at last let K equals to 0.
3. We now prove G is a yes-instance of 3-coloring problem if and only if (C, S, R, K) is a yes-instance of the given problem:
   - "$\Rightarrow$": if G is a yes-instance of 3-coloring problem, then schedule the courses according to their color. Since for each edge (u, v), u and v will be painted with different color, then for each student, his/her requests will not be scheduled to the same slot, which means the given problem is also a yes-instance.
   - "$\Leftarrow$": if (C, S, R, K) is a yes-instance of the given problem, then painting the nodes in G according to their slots. Since $K = 0$, then for every student, there is no conflict between their requests, which suggests that for every edge (u, v), u and v will not be painted with the same color. It is also a yes-instance of 3-coloring problem.

Therefore, the given problem is NP-complete.

1: (3'+3'+4') Multiple Choice

The following questions are multiple choice questions, each question may have one or more correct answers. Select all correct answers.

Note: You should write those answers ***in the box*** *below.*

Question 1	Question 2	Question 3
B	D	BC

Question 1. *A problem in NP is NP-complete if:*

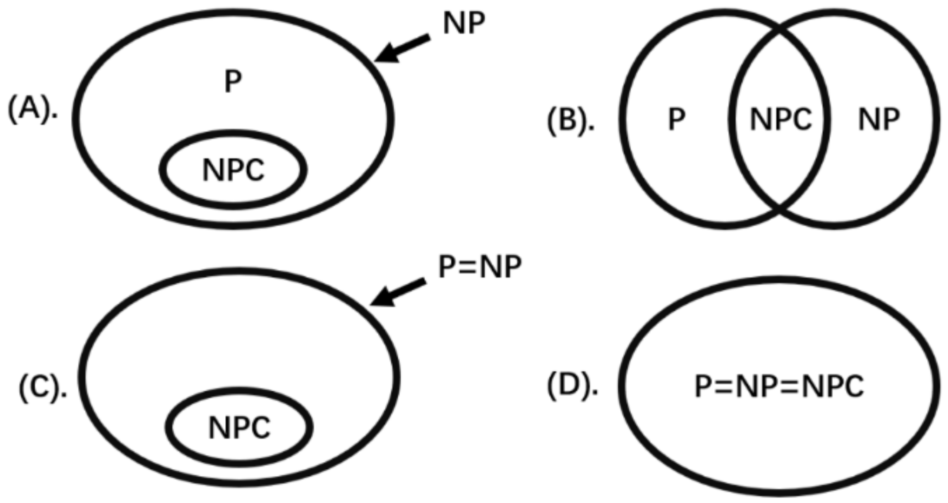
(A) It can be reduced to the 3-SAT problem in polynomial time.

(B) The 3-SAT problem can be reduced to it in polynomial time.

(C) It can be reduced to any other problem in NP in polynomial time.

(D) Some problem in NP can be reduced to it in polynomial time.

Question 2. *Suppose we have found an algorithm which correctly solves the vertex cover problem in polynomial time. Then under this circumstance, which one of the following Venn diagrams correctly represents the relationship among the complexity classes P, NP and NP Complete (NPC)?*



Question 3. *Assume problem A reduces to problem B in polynomial time. Which of the following choice(s) is/are correct?*

(A) If A is in ***P****. Then B is in* ***P****.*

(B) If B is in ***P****. Then A is in* ***P****.*

(C) If A is ***NP****-complete. Then B is* ***NP****-complete.*

(D) If B is ***NP****-complete. Then A is* ***NP****-complete.*

2: (10') 4-COLOR

Given an undirected graph and 4 different colors, can we color the nodes so that no adjacent nodes have the same color? Show that the 4-COLOR problem is NP-complete. **(Hint: Reduce from 3-COLOR)**

Solution:

1. First show that 4-COLOR is in NP:

let certificate be a coloring plan. For any given instance, we can check all pairs of two adjacent vertices to see if they have the same color, which has time complexity $O(|V|^2)$ and is in polynomial time. So 4-COLOR is in NP.

2. Choose 3-COLOR to reduce from. For any 3-COLOR instance, we can construct a 4-COLOR instance by adding another vertex that connects all the other points.

3. Now prove a yes-instance of 3-COLOR will lead to a yes-instance of 4-COLOR by such construction, and vice versa:

- "$\Rightarrow$": For any yes-instance of 3-COLOR, in the 4-COLOR instance, then use the color plan to color the origin vertices, and color the new vertex with a new color, then there is a yes-instance for 4-COLOR.
- "$\Leftarrow$": For any yes-instance of the constructed 4-COLOR: As the new added vertex is connected to all the other vertices, then it should be a color that not the same with all the other vertices. The other vertices then actually use 3 kinds of colors, which means there is a yes-instance for the 3-COLOR.

Therefore, the given problem is NP-complete.

3: (10') Reduction from Independent Set

Given a set E and m subsets of $E, S_1, S_2, ..., S_m$, is there a way to select k of the m subsets such that the selected subsets are pairwise disjoint?
Show that this problem is NP complete.
HINT: Reduction from Independent Set.
Solution:
Let (V, E) be a graph. For every $v \in V$, let S_v be the set of edges that are incident on v. Hence, (G, k) is an instance of independent set. Let U the set of k different vertices, then it is an independent set if and only if the sets S_u and S_v are disjoint for all $u \in U$ and $v \in U$.
Obviously, this problem is in NP, and the mapping function runs in polynomial time.

4: (10') Exact 4-SAT problem

In the Exact 4-SAT problem, the input is a set of clauses, each of which is a disjunction of exactly four literals, and such that each variable occurs at most once in each clause. The goal is to find a satisfying argument, if one exists. Prove that Exact 4-SAT is NP-complete by a reduction from 3-SAT.

Solution:

We will try to solve the problem by proving EXACT-4-SAT problem as NP Complete in following steps:

1. **EXACT-4-SAT problem is NP Problem.**

Argument $\rightarrow$ For a given solution S of the problem, for each clause, it will take O(1) time to verify if any one literal of the clause is satisfied, and we will check for all m clauses. Thus in O(m) time we can verify the solution S which is in polynomial time.

2. **EXACT-4-SAT problem is as hard as the 3-SAT problem. We will apply reduction 3-SAT $\rightarrow$ EXACT-4-SAT.**

2.1 Input to 3-SAT can be converted into input to EXACT-4-SAT in polynomial time.

Argument $\rightarrow$ We can convert the input of 3-SAT problem by taking the following action:

if clause length is 3, we can replace it by new clauses with one more variable x:

$$(a \vee b \vee c) \rightarrow (a \vee b \vee c \vee x) \wedge (a \vee b \vee c \vee \neg x) ------(1)$$

Each of clause C can be converted into new clause C1 in polynomial time O(n) and we have m such clauses. Thus we can convert into O(nm), which is polynomial time.

2.2 Solution to the EXACT-4-SAT problem can be converted to solution for 3-SAT in polynomial time.

Argument $\rightarrow$ For the solution found S, for EXACT-4-SAT problem, we can convert to solution of 3-SAT by simply dropping the extra new variables from each clause that we added in step(a). This can be done in polynomial time as well as we have m clauses, and constant time to remove the extra variables added - O(m)

2.3 Solution to EXACT-4-SAT exists iff solution to 3-SAT problem.

Argument $\rightarrow$ We can prove that if there is a solution to EXACT-4-SAT solution then there is a solution to 3-SAT problem and vice versa.

If we check for clause (1) , if the EXACT-4-SAT is true, that is $(a \vee b \vee c \vee x) \wedge (a \vee b \vee c \vee \neg x)$ is true, the reduced clause for 3-SAT $(a \vee b \vee c)$ will always be true as value for new variable x can be either True or False, which satisfies either one of the clauses.

We can also prove that if $(a \vee b \vee c)$ is satisfied, then by adding a new variable x, the clause $(a \vee b \vee c \vee x) \wedge (a \vee b \vee c \vee \neg x)$ will be true as well, as value of x can be either True or False, which will satisfy either of one of the clause, thus satisfying the clause.

We can do similar exercise for $-(2)$ and $-(3)$ clauses.

Thus, Solution to EXACT-4-SAT exists iff solution to 3-SAT problem.

Thus EXACT-4-SAT is a NP Complete problem.