Lab06-NP Reduction

CS2308-Algorithm and Complexity, Xiaofeng Gao, Spring 2022.

∗ If there is any problem, please contact TA Hongjie Fang.

∗ Name: Zhenran Xiao   Student ID: 520030910281   Email: xiaozhenran@sjtu.edu.cn

1. *Minimum Test Collection.*

   - **Input**. A finite set $A = \{a_1, a_n, \cdots, a_n\}$ of "possible diagnoses", a collection $\mathcal{C}$ of subsets of A representing binary "tests", and a positive integer $k \leq |\mathcal{C}|$.
   - **Output** (Decision version). Whether there is a subcollection $\mathcal{C}' \subseteq \mathcal{C}$ with $|\mathcal{C}'| \leq k$ such that, for every pair (a_i, a_j) of possible diagnoses from A, there is some test $c \in \mathcal{C}'$ for which $|\{a_i, a_j\} \cap c| = 1$, that is, a test c that "distinguishes" between a_i and a_j.

   Given a black box to solve the *Minimum Test Collection* decision problem in polynomial time, you are required to use the black box to solve the corresponding search problem in polynomial time. Notice that you can use the black box with different inputs more than once. (Hint: the search problem is finding and outputing the smallest possible set $\mathcal{C}'$)

   Solution.

   Algorithm 1: *MTC_Search*$(A, \mathcal{C})$

   Input: a finite set $A = \{a_1, a_n, \cdots, a_n\}$ of "possible diagnoses", a collection $\mathcal{C}$ of subsets of A representing binary "tests"

   Output: the smallest possible set $\mathcal{C}'$

   for $i \leftarrow$ *1 to* $\frac{n(n-1)}{2}$ **do**
       if *BlackBox*$(A, \mathcal{C}, i) =$ *yes and BlackBox*$(A, \mathcal{C}, i-1) =$ *no* **then**
           for *all subcollections* $\mathcal{C}' \subseteq \mathcal{C}$ *with* $|\mathcal{C}'| = i$ **do**
               if *BlackBox*$(A, \mathcal{C}', i) =$ *yes* **then**
                   Output $\mathcal{C}'$;

   □

   Explanation:

   - There must exist $\mathcal{C}'$ satisfying $|\mathcal{C}'| \leq \frac{n(n-1)}{2}$. $|\mathcal{C}'| = \frac{n(n-1)}{2}$ means for every pair (a_i, a_j) of possible diagnoses from A, there is a different test $c \in \mathcal{C}'$ for which $|\{a_i, a_j\} \cap c| = 1$.
   - If *BlackBox*$(A, \mathcal{C}, i) =$ *yes* and *BlackBox*$(A, \mathcal{C}, i-1) =$ *no*, we can easily know the smallest possible set $\mathcal{C}'$ satisfies $|\mathcal{C}'| = i$.

2. *Not-All-Equal Satisfiability (NAE-SAT)* problem is similar to *SAT* problem, except that the clause is satisfied if at least one literal is true and one is false. Define *NAE-Ek-SAT* problem as a special case of *NAE-SAT* problem that all clauses have length exactly k. Answer the following questions.

   (a) Write down the certificate and certifier of *NAE-Ek-SAT* problem.

   (b) Prove that *NAE-Ek-SAT* is an NP problem.

   (c) Prove that *NAE-E4-SAT* is an NP-complete problem. (Hint: consider reduction from *3-SAT* problem)

(d) Prove that *NAE-E3-SAT* is an NP-complete problem. (Hint: consider reduction from *NAE-E4-SAT* problem)

Solution.

(a) certificate: An assignment of truth values to the n boolean variables.
certifier: Check that each clause in ϕ has at least one true literal and one false literal.

(b) **Example.**

- instance: $\phi = (x_1 \vee \bar{x_2} \vee ... \vee x_k) \wedge (\bar{x_1} \vee x_2 \vee ... \vee x_k)$
- certificate: $x_1 = 1, x_2 = 1, ..., x_n = 1$

Conclusion. There exists a poly-time certifier for *NAE-Ek-SAT* problem, so *NAE-Ek-SAT* is in NP.

(c) **Claim.** *3-SAT* $\leq_p$ *NAE-E4-SAT*
Reduction. Given an instance ϕ of *3-SAT*, add one false literal to each clause of *3-SAT* to construct ϕ'. ϕ' is an instance of *NAE-E4-SAT*.

Proof.
- *NAE-Ek-SAT* is in NP. So *NAE-E4-SAT* is NP problem.
- The reduction can be done in polynomial time.
- Assume an assignment X to all the variables. If it can make ϕ true, considering the reduction, obviously it can make ϕ' true. If it can't make ϕ true, which means there exists at least one clause with all literals false, considering the reduction, obviously it can't make ϕ' true.

Therefore, *NAE-E4-SAT* is an NP-complete problem. □

(d) **Claim.** *NAE-E4-SAT* $\leq_p$ *NAE-E3-SAT*
Reduction. Given an instance ϕ of *NAE-E4-SAT*, for each clause in ϕ, if the number of true literals $\geq$ the number of false iterals, remove a true literal from it. If not, remove a false iteral. Construct ϕ' by that, ϕ' is an instance of *NAE-E3-SAT*.

Proof.
- *NAE-Ek-SAT* is in NP. So *NAE-E3-SAT* is NP problem.
- The reduction can be done in polynomial time.
- Assume an assignment X to all the variables. If it can make ϕ true, considering the reduction, obviously it can make ϕ' true. If it can't make ϕ true, which means there exists at least one clause with all literals false or all literals true, considering the reduction, obviously it can't make ϕ' true.

Therefore, *NAE-E3-SAT* is an NP-complete problem. □

□

3. Given a graph $G = (V, E)$, a distance function $d(u, v) \in \mathbb{N}$ for each pair of vertices u, v, a starting vertex $s \in V$ and two integers $k, l \in \mathbb{N}$. Are there l ($l \geq 2$) cycles that all starts from vertex s, such that for all $v \in V$ is on at least one of l cycles, and all cycles have length at most k. We call this problem *l-CYCLE* problem. Show that this problem is NP-complete.

Solution.
certificate: l ($l \geq 2$) cycles in G.
certifier: Check whether the l ($l \geq 2$) cycles all start from vertex s, such that for all $v \in V$ is on at least one of l cycles, and all cycles have length at most k.
Example.

- instance: A graph $G = (V, E)$, a distance function $d(u, v) \in \mathbb{N}$ for each pair of vertices u, v, a starting vertex $s \in V$ and two integers $k, l \in \mathbb{N}$.
- certificate: l ($l \geq 2$) cycles in G.

Conclusion. l-*CYCLE* problem is in NP.
Claim. *HAM-CYCLE* $\leq_p$ l-*CYCLE*
Reduction.
instance of *HAM-CYCLE*: ϕ is in an graph $G = (V, E)$.
instance of *l-CYCLE*: ϕ' is in an graph G' constructed by l graphs same as G. Arbitrarily choose a start vertex for each $HAM-CYCLE$ and merge these start vertices into one vertex in G'. This vertex is the start vertex of ϕ'. Let k be the length of ϕ.

Proof.
- The reduction can be done in polynomial time.
- If ϕ is true, obviously ϕ' is true. If ϕ is not true, there are vertices in G can't be passed through. So in G', they are not on at least one of l cycles. ϕ' is not true, either.

Therefore, l-*CYCLE* is an NP-complete problem. □

□

Remark: You need to include your .pdf and .tex files in your uploaded .rar or .zip file.