CS101 Algorithms and Data Structures
Fall 2022
Homework 12

Due date: 23:59, December 21th, 2022

1. Please write your solutions in English.
2. Submit your solutions to Gradescope.
3. If you want to submit a handwritten version, scan it clearly.
4. When submitting, match your solutions to the problems correctly.
5. No late submission will be accepted.
6. Violations to any of the above may result in zero credits.
7. You are recommended to finish the algorithm design part of this homework with LaTeX.
8. Please check your Account Settings for Gradescope when submitting! Set your FULL name to your Chinese name and your 10-digit STUDENT ID correctly.

1. (0 points) Proving a problem in NP-Complete

When proving problem $A \in$ NP-Complete, please clearly divide your answer into the following (1+2) sections:

1. Show that $A \in$ NP.
2. Choose a problem $B \in$ NP-Complete and show that $B \leq_p A$.
   (a) For every instance I_B of B, construct an instance I_A of problem A.
   (b) Prove that $I_B \in B \iff I_A \in A$.
3. Conclude that A is in NP-Complete.

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. 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 show that 3-COLOR can be reduced to this problem.
   (a) 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.
   (b) 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.
3. The given problem is in NP and a NP-Complete problem can be reduced to it in polynomial time, so it is also NP-Complete.

2. (6 points) Multiple Choices

Each question has **one or more** correct answer(s). Select all the correct answer(s). For each question, you will get 0 points if you select one or more wrong answers, but you will get 1 point if you select a non-empty subset of the correct answers.

Write your answers in the following table.

(a)	(b)	(c)
BD	AC	C

(a) (2') A problem in NP is NP-Complete if:

A. It can be reduced to another NP-Complete problem in polynomial time.

B. There exists a NP-Complete problem which can be reduced to it in polynomial time.

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

D. Any other NP problem can be reduced to it in polynomial time.

(b) (2') Assuming that P $\neq$ NP, which of the following problems are in NP-Complete? You may search the Internet for more information if you are unfamiliar with the problems.

A. `LONG-PATH`: (G, s, t, k) Given an undirected graph G, determine whether there exists a simple path from s to t whose length is greater or equal to k.

B. `HALTING`: (P, I) Given a compilable C++ program P and the input I for P, determine if P runs infinitely on I.

C. `4-SAT`: ϕ Given a CNF (conjunction normal form) where each clause is the disjunction of exactly 4 literals, determine whether ϕ is satisfiable.

D. `PRIME`: n Given a positive integer n, determine whether it is a prime number.

(c) (2') For two decision problems A and B, suppose that $A \leq_p B$. Which of the following statements are true? (Hint: there exists complexity classes that are strictly bigger than NP)

A. $A \in$ P $\implies$ $B \in$ P

B. $A \in$ NP-Complete $\implies$ $B \notin$ NP-Complete.

C. $B \in$ P $\implies$ $A \in$ P.

D. $B \in$ NP-Complete $\implies$ $A \in$ NP-Complete.

3. (10 points) PARTITION is NP-Complete

Given an array $A = [a_1, a_2, ..., a_n]$ of non-negative integers, consider the following problems:

1.Partition: Determine whether there is a subset $P \subseteq [n]([n] = \{1, 2, ..., n\})$ such that $\sum_{i\in P} a_i = \sum_{j\in[n]\setminus P} a_j$.
For example, given $A = [2, 4, 6, 8]$, then $P = \{1, 4\}$ is a partition of A since $a_1+a_4 = a_2+a_3 = 10$.

2.Subset Sum: Given some integer K, determine whether there is a subset $P \subseteq [n]$ such that $\sum_{i\in P} a_i = K$.
For example, given $A = [1, 3, 5, 7]$ and $K = 6$, then $P = \{1, 3\}$ gives a subset sum of $a_1+a_3 = 6$.

Suppose we have proven that Subset Sum problem is in NP-complete, prove that Partition problem is also in NP-complete.

(a) (2') Prove that the partition problem is in NP.

Solution:
for any given partition P
we can calculate $s_1 = \sum_{i\in P} a_i$, and $s_2 = \sum_{i\in[n]\setminus P} a_i$
both s_1, s_2 can be compute in polynomial time.
we just need to check whether $s_1 = s_2$ or not.
so the partition problem is in NP

(b) (7') Find a polynomial time reduction from subset sum problem to parittion problem, and prove its correctness.

Solution:
let $M = \sum_{i\in[n]} a_i$,
and let N be a number that is big enough, such as take $N > 10M$,
we can generate a new array $A' = [a_1, a_2, \cdots, a_n, N - k, N + k - M]$
and below we all talk about the new array A'.

$\Rightarrow$: if the Subset Sum problem (A, k) is a yes-instance,
then there exist a partition $P \subseteq [n]$ s.t. $\sum_{i\in P} a_i = k$
let $P' = P \cup \{n + 1\}$,
so $\sum_{i\in P'} a_i = k + (N - k) = N$
and since $\sum_{i\in[n+2]\setminus P'} a_i = (M + (N - k) + (N + k - M)) - N = N$
which means that $\sum_{i\in P'} a_i = \sum_{j\in[n+2]\setminus P'} a_j$
so the partition problem is yes-instance.

$\Leftarrow$: if the partition problem is yes-instance
then there must exist a partition $P, s.t. \sum_{i\in P} a_i = \sum_{j\in[n+2]\setminus P} a_j$ and since we took N as a very big number that $N > 10M$,
so $a_{n+1} + a_{n+2} = (N - k) + (N + k - M) = 2N - M > M$,
so we can make sure that $a_{n+1} = N - k, a_{n+2} = N + k - M$ is impossible to be in the same partition.
Without loss of generality, we suppose that $(n + 1) \in P$, then $(n + 2) \in [n + 2]\setminus P$
and since for the partition problem, each sum of the partition is half of the sum of the

whole array, i.e. the value is $\frac{\sum_{i\in[n+2]} a_i}{2} = \frac{M+(N-k)+(N+k-M)}{2} = N$,
so $\sum_{i\in P\setminus\{n+1\}} a_i = N-(N-k) = k$,
so (A,k) is a yes-instance, i.e. the Subset Sum problem is a yes-instance.
and all operations we have done are in polynomial time.

so above all, the subset sum problem can be reducted to partition problem in polynomial time.
i.e. subset sum problem$\leq_p$ partition problem

(c) (1') Conclusion:

Solution:
the partition problem is in NP
the subset sum problem can be reducted to the partition problem in polynomial time

let A be the partition problem, B be the subset sum problem,
since $B \in NP-complete, B \leq_P A, A \in NP$
so $A \in NP-complete$.

so the partition problem is also in $NP-Complete$.

4. (10 points) HALF-CLIQUE ∈ NP-Complete

In an undirected graph $G = (V, E)$, a subset of the vertices $S \subseteq V$ is said to be a **clique** if for all pairs of vertices in S are connected or formally $\forall(u, v) \in S \times S(u \neq v \rightarrow \{u, v\} \in E)$. Note that a subset of zero or one vertex is also considered as a clique.

The k-CLIQUE problem is a classic NP-Complete problem stated as follows:
Given (k, G) where k is a non-negative integer and G is an undirected graph, determine if G contains a clique of at least k vertices.

Now let's consider the HALF-CLIQUE problem which is defined as follows:
Given an undirected graph $G = (V, E)$, determine if G contains a clique of $\lfloor V/2 \rfloor$ vertices.
Show that HALF-CLIQUE is a NP-Complete problem.

Hint: reduce from k-CLIQUE, consider the cases where $k = |V|/2$, $k < |V|/2$ and $k > |V|/2$.

(a) (2') HALF-CLIQUE ∈ NP

Solution:
for a given vertex set $S \subseteq V$,
when can check whether $|S| = \lfloor \frac{|V|}{2} \rfloor$ in polynomial time.
If so, we can check whether $\forall(u, v) \in S \times S(u \neq v \rightarrow \{u, v\} \in E)$ in polynomial time.
If all these two checks are satisfied, then the half-clique is hold, we can check it in polynomial time.
so above all, HALF-CLIQUE ∈ NP

(b) (7') Polynomial time reduction: construction and correctness

Solution:

- $k = \lfloor \frac{|V|}{2} \rfloor$ the k-clique is exactly the half-clique, so it can be seen as reducted.
- $k > \lfloor \frac{|V|}{2} \rfloor$ we can generate $2k - |V|$ new vertices, and do not conntect them with any other vertices, so the new vertex set $V' = V + the\ new\ 2k - |V|\ points$ and the new graph is $G' = (V', E)$, and $|V'| = |V| + 2k - |V| = 2k$
  so $k = \frac{|V'|}{2}$
  $\Rightarrow$: if (k, G) is a yes-instance for k-clique, then (k, G') is a yes-instance for k-clique, so half-clique is a yes-instance for graph G'

  $\Leftarrow$: if G' for half-clique is a yes-instance, then (k, G') is a yes-instance for k-clique,
  and since no new edges are added, so the new vertices do not have any edge connected to other vertices,
  so (k, G) is a yes-instance for k-clique

so k-clique can be reducted to half-clique when $k > \lfloor \frac{|V|}{2} \rfloor$

- $k < \lfloor \frac{|V|}{2} \rfloor$ we can generate $|V| - 2k$ new vertices, and connect each of the new vertices with all the origin vertices,
  so we get $V' = V \cup \{$ *the* $|V| - 2k$ *new vertices*$\}$,
  so $|V'| = |V| + (|V| - 2k) = 2|V| - 2k$
  and $E' = E \cup \{$ *the new edges we said above*$\}$

  and let $G' = (V', E')$
  $\Rightarrow$: if (k, G) is a yes-instance for k-clique,
  and let S be the clique.
  let $S' = S \cup \{$ *the* $(|V| - 2k)$ *new vertices we newly generate*$\}$
  since $|S| = k$, so $|S'| = k + |V| - 2k = |V| - k$,
  and since $|V'| = 2|V| - 2k$,
  so $|S'| = \frac{|V'|}{2}$,
  so it is yes-instance for half-clique.

  $\Leftarrow$: if G' for half-clique is a yes-instance
  Suppose that the clique is S,
  so $|S| = \frac{|V'|}{2} = \frac{2|V|-2k}{2} = |V| - k$,
  and let $S' = S \backslash \{$ *the* $(|V| - 2k)$ *vertices we newly added*$\}$
  $|S'| = |S| - (|V| - 2k) = (|V| - k) - (|V| - 2k) = k$
  since the $|V| - 2k$ vertices we newly added are connect to all vertices in V,
  so if we remove all the newly added vertices, the new clique is still following
  $\forall (u, v) \in S' \times S' (u \neq v \rightarrow \{u, v\} \in E)$
  i.e. the k-clique is a yes-instance.

  so k-clique can be reducted to half-clique when $k < \lfloor \frac{|V|}{2} \rfloor$

and all operations we have done are in polynomial time.
so above all, the k-clique can be reducted into half-clique in polynomial time.
i.e. k-clique$\leq_p$ half-clique

(c) (1') Conclusion

Solution:
the half-clique is in NP
the k-clique can be reducted to half-clique in polynomial time

let A be half-clique, B be k-clique,
since $B \in NP - complete, B \leq_P A, A \in NP$
so $A \in NP - complete$.

so the half-clique is also in $NP - Complete$.