Lab11-NP Reduction

CS214-Algorithm and Complexity, Xiaofeng Gao & Lei Wang, Spring 2021.

* Name: Haoyi You   Student ID: 519030910193   Email: yuri-you@sjtu.edu.cn

1. We are feeling experimental and want to create a new dish. There are various ingredients we can choose from and we'd like to use as many of them as possible, but some ingredients don't go well with others. If there are n possible ingredients (numbered 1 to n), we write down an $n \cdot n$ matrix giving the discord between any pair of ingredients. This discord is a real number between 0.0 and 1.0, where 0.0 means "they go together perfectly" and 1.0 means "they really don't go together." Here's an example matrix when there are five possible ingredients.

	1	2	3	4	5
1	0.0	0.4	0.2	0.9	1.0
2	0.4	0.0	0.1	1.0	0.2
3	0.2	0.1	0.0	0.8	0.5
4	0.9	1.0	0.8	0.0	0.2
5	1.0	0.2	0.5	0.2	0.0

In this case, ingredients 2 and 3 go together pretty well whereas 1 and 5 clash badly. Notice that this matrix is necessarily symmetric; and that the diagonal entries are always 0.0. Any set of ingredients incurs a penalty which is the sum of all discord values between pairs of ingredients. For instance, the set of ingredients $(1, 3, 5)$ incurs a penalty of $0.2+1.0+0.5 = 1.7$. We define the EXPERIMENTAL CUISINE as follows:

Given n ingredients to choose from, the $n \times n$ discord matrix and integer k and a number p, decide whether there exists a collection of at least k ingredients that has a penalty $\leqslant p$

Prove that 3-SAT $\leq_p$ EXPERIMENTAL CUISINE

Solution. (a) Firstly, we know 3-SAT $\leq_p$ INDEPENDENT-SET,so we only need to prove INDEPENDENT-SET $\leq_p$ EXPERIMENTAL CUISINE.

(b) For $\forall$ graph $G = (V, E)$,we wonder whether there exists an independent set S with $|S| \geq k$.Let $n = |V|$,then we can mark the vertices to $\{v_1, v_2, \dots v_n\}$. Then we map the vertices to ingredients one to one. And $\forall i, j$,we define the discord between i, j that

$$discord_{i,j} = \begin{cases} 2 & if\ (v_i, v_j) \in E \\ 1 & if\ (v_i, v_j) \notin E \end{cases} \tag{1}$$

Then we ask if there exists a collection of at least k ingredients that has a penalty $\leq \frac{k*(k-1)}{2}$

Here we prove this construction is correct.
If there exists an independent set S with $|S| \geq k$, we choose S' as a subset of S that $|S'| = k$. Obviously, S' is also an independent set. So we can assume $S = k$.
We assume $S = \{v_{l_1}, v_{l_2}, \dots v_{l_k}\}$, then $\forall i, j, (v_{l_i}, v_{l_j}) \notin E$, so $discordl_i, l_j = 1$. We choose $\{l_1, l_2, \dots l_k\}$ as ingredients, then the penalty $\leq \frac{k(k-1)}{2}$.

If there exist an ingredients set $L = \{l_1, l_2, \dots l_k\}$ with penalty $\leq \frac{k(k-1)}{2}$. Since $\forall x, y, discord_{x,y} \geq 1$, so $\forall i, j \in L, discord_{i,j} = 1$. So the S is an independent set. □

2. An induced subgraph $G' = (V', E')$ of a graph $G = (V, E)$ is a graph that satisfies $V' \subseteq V$ and $E' = \{(u, v) \in E | u, v \in V'\}$. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ and an integer b, we need to decide whether G_1 and G_2 have a common induced subgraph G_c with at least b nodes. This problem is called Maximum Common Subgraph (MCS). Prove that MCS is NP-complete. (Hint: reduce from INDEPENDENT-SET)

Solution.

If Independent-Set $\leq_p$ Maximum Common Subgraph, then Maximum Common Subgraph will be NP-complete because INDEPENDENT-SET is an NP-complete problem.

(b) For $\forall$ given graph $G = (V, E)$ and an integer k, we wonder whether there a subset $S \subseteq V$ with $|S| \geq k$ and no two vertices in S are adjacent(Independent Set).
Then we construct two graph $G_1 = (V, E_1), G_2 = (V, E_2)$ that G, G_1, G_2 have the same vertices set.G_1 is a complete graph, and $E_2 = E_1 - E$, which means G_2 is a complement graph of G.
Then for G_1, G_2 we only need to decide whether they have a common induced subugraph $G_e = (V', E')$ with at least k nodes.(Maximum Common Subgraph).

(c) Here we prove this change is correct.// If there exists $S \subseteq V$ satisfies Independent-Set condition. Then $\forall v_1, v_2 \in S, (v_1, v_2) \notin E$. As for G_1, G_2, obviously $(v_1, v_2) \in G_1$. Also, $(v_1, v_2) \in G_2$ because G_2 is a complement graph of G and $(v_1, v_2) \notin G$. So we get $S = V'$, then from V' and G we can construct G_e.
Similarly, if there exists G_e satisfies MCS condition, the $\forall v_1, v_2 \in V', (v_1, v_2) \notin E$, so $S = V'$.

(d) As a result, $\forall$ Independent-Set problem, we can change it to an MCS problem, so Independent-Set $\leq_p$ Maximum Common Subgraph.

□

3. Let us define the k-spanning tree as a spanning tree in which each node has a degree $\leqslant k$. Given a graph $G = (V, E)$ and a positive integer k, we need to decide whether there exists a k-spanning tree in G. Prove that this problem is NP-complete. (Hint: reduce from HAMILTONIAN-CYCLE)

Solution.

There are two steps. Firstly we prove Hamiltonian-Cycle $\leq_q$ Hamiltonian-Road. Secondly we prove Hamiltonian-Road $\leq_q$ k-spanning tree. Then we get Hamiltonian-Cyc k-spanning tree. We know Hamiltonian-Cycle is an NPC problem, so the k-spanning tree is also an NPC problem.

(b) Hamiltonian-Cycle $\leq_q$ Hamiltonian-Road.

i. For $\forall$ given graph G,randomly choose an vertex $v \in V$ and splits it into two vertices v_1, v_2. Then we add two vertices v_3, v_4 into V. Then we construct a new graph $G' = (V', E')$ that $V' = (V - \{v\}) \cup \{v_1, v_2, v_3, v_4\}$. $E' = E_1 \cup E_2 \cup \{(v_1, v_3), (v_2, v_4)\}$ with $E_1 = \{(v_i, v_j) | v_i, v_j \in V - \{v\} \& (v_i, v_j) \in E\}$, $E_2 = \{(v_i, v_1), (v_i, v_2) | (v_i, v) \in E\}$.

ii. Then we can assert there exists a Hamiltonian-Cycle in $G \iff$ there exists a Hamiltonian-Road in G'.

iii. proof:
If there exists a Hamiltonian-Cycle C in G, assume C begins and ends at v that $C = (v, u_1 \ldots, u_{n-1}, v)$. So $R = (v_3, v_1, u_1, \ldots, u_{n-1}, v_2, v_4)$ is a Hamiltonian-Road in G'.
Similarly, if there exists a Hamiltonian-Road R in G'. Since $d(v_3) = d(v_4) = 1$, R must begin at v_3/v_4 and end at v_4/v_3. So assume $R = (v_3, v_1, u_1, \ldots, u_{n-1}, v_2, v_4)$, so $C = (v, u_1 \ldots, u_{n-1}, v)$ is a Hamiltonian-Cycle in G.

(c) Hamiltonian-Road $\leq_q$ k-spanning tree.

i. For $\forall$ given graph G, there exists a Hamiltonian-Road in $G \iff$ there exists a 2-spanning tree in G.
ii. Actually, a Hamiltonian-Road in G is also a 2-spanning tree. Then we prove a 2-spanning tree is a road,and from the definition of tree we can get it is a Hamiltonian-Road.
iii. Firstly, we prove only 2 of the node v_1, v_n with $d(v_1) = d(v_n) = 1$ and for other node v_i with $d(v_i) = 2$. That is because $\sum\limits_{i=1}^{n} d(v_i) = 2 * (n-1)$ and $\forall v_i, d(v_i) > 0$.
iv. Secondly, we start at a leaf v_1 of the tree. Then we can find a node v_2 as the neighbor of v_1. Then as for two neighbors of v_2, one has been visited just now, and the other must have not been visited because the two neighbors of one node must not linked if we delete this node,due to the characteristic of tree.
v. Lastly,we visit the other neighbor.Then we can recursively do this until having visited all nodes. Then we get a Hamiltonian-Road.

□

4. We define the decision problem of Knapsack Problem as follows:
Given n indivisible objects, each with a weight of $w_i > 0$ kilograms and a value $v_i > 0$, a knapsack with capacity of W kilograms and a number k, decide whether there is a collection of objects that can be put into the knapsack with a total value $V \geqslant k$.
Prove that Knapsack Problem is NP-complete.

Solution. We prove it based on another problem called Subset Sum Problem.Since we have known that Subset Sum Problem is an NP-complete problem, we only need to prove Subset Sum Problem $\leq_p$ Knapsack Problem.

(a) For $\forall$ given set A with $|A| = N$, the value of the items in A is s_i,and the target sum is S.
(b) Then as for a Knapsack Problem, there are N objects, and the i^{th} object has $\omega_i = s_i, v_i = s_i$, capacity $W = S$, and then we need to decide whether there is a collection of objects that can be put into the knapsack with a total value $V \geq S$.
(c) Here we prove the above two problems are equal. If there exists a subset satisfies the sum is S, and also choose the same objects in Knapsack Problem, obviously the value is S and the weight is also S it satisfies the knapsack property.
If there exists a set of objects that satisfies the knapsack property. Since for every objects, the weight is equal to the value, the sum of weight is equal to the sum of value. However, total value $V \geq S$ and total weight $W \leq S$, so $V = W = S$. So the sum of subset is S.

□