Lab11-NP Reduction

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

∗ If there is any problem, please contact TA Yihao Xie.

∗ Name:BeichenYu   Student ID:519030910245   Email: polarisybc@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. Given an instance Φ of 3-SAT, we construct an instance of Experimental Cuisine (G, k, p) that exists a collection of at least k ingredients that has a penalty $\leqslant p$.

In each clause of Φ, the form is like $x_1 \bigvee x_2 \bigvee x_3$. To make this clause true, we have 7 possible situations. They are: $x_1 \bigwedge x_2 \bigwedge x_3$,$\bar{x_1} \bigwedge x_2 \bigwedge x_3$,$x_1 \bigwedge \bar{x_2} \bigwedge x_3$,$x_1 \bigwedge x_2 \bigwedge \bar{x_3}$,$\bar{x_1} \bigwedge \bar{x_2} \bigwedge x_3$, $\bar{x_1} \bigwedge x_2 \bigwedge \bar{x_3}$ and $x_1 \bigwedge \bar{x_2} \bigwedge \bar{x_3}$. For each condition, we define it an ingredient. So if we have k clauses in Φ, we will have $7k$ ingredients. For the 7 ingredients from the same clause, we set the discord value between each two ingredients to 1; and set the discord value between two ingredients to 1 as well if they are contradicting. For all of other pairs, set the discord value as 0.

We set $p = 0$. If we can find at least k ingredients that has a penalty $\leqslant p = 0$, that means for each clause we can find an ingredient to make it true and all the chosen ingredients are non-contradicting. Then if Experimental Cuisine is solvable in polynomial time, then 3-SAT is solvable in polynomial time. That means 3-SAT $\leq_p$ Experimental Cuisine. □

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. We already know that INDEPENDENT-SET is NP-complete. So if we want to prove that MCS is NP-complete, we just need to prove that:

$$\text{INDEPENDENT-SET} \leq_p \text{MCS}$$

Given a graph $G = (V, E)$. Let $G_1 = (V, E)$ and $G_2 = (V, \varnothing)$. If G_1 and G_2 have a common induced subgraph $G_c = (V_c, \varnothing)$ with at least b nodes, then G has a independent set V_c which have at least b nodes. Meanwhile, if G has a independent set V_c of at least b nodes, then $G_c = (V_c, \varnothing)$ is the common induced subgraph of G_1 and G_2. So INDEPENDENT-SET $\leq_p$ MCS, then MCS is NP-complete. □

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. We already know that HAMILTONIAN-CYCLE is NP-complete. So if we want to prove that K-SPANNING-TREE is NP-complete, we just need to prove that:

   $$\text{HAMILTONIAN-CYCLE} \leq_p \text{K-SPANNING-TREE}$$

   .

   First we just prove that HAMILTONIAN-CYCLE $\leq_p$ 2-SPANNING-TREE. Given a graph $G = (V, E)$. If there exists a hamiltonian cycle in G, we can delete any edge in the hamiltonian cycle and then we get a 2-spanning tree of G. Meanwhile through adding an edge, we can change the 2-spanning tree of G into a hamiltonian cycle of G. So 2-SPANNING-TREE is NP-complete.

   Give n a graph $G(V, E)$ as an instance of 2-SPANNING-TREE, we construct another graph $G'(V', E')$.

   $$V' = V \cup \{v_1, v_2, \cdots, v_{k-2}\}$$
   $$E' = E \cup \{vv_1, vv_2, \cdots, vv_{k-2} | v \in V\}$$

   Assume T is a 2-Spanning-Tree of G. For each vertices in $\{v_1, v_2, \cdots, v_{k-2}$, we find an edge in $\{vv_1, vv_2, \cdots, vv_{k-2} | v \in V\}$ and add it on T, then we get a k-spanning tree. Obviously, if T^{prime} is a k-spanning tree of G', removing all edges of $v_i v$ it will be turned into a 2-spanning tree.So 2-SPANNING-TREE $\leq_p$ K-SPANNING-TREE. Then K-SPANNING-TREE is NP-complete.

   □

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 already know that SUBSET-SUM is NP-complete. So if we want to prove that KNAPSACK is NP-complete, we just need to prove that:

   $$\text{SUBSET-SUM} \leq_p \text{KNAPSACK}$$

.

We create such a Knapsack problem that:

$$w_i = v_i = s_i$$

$$W = k = t$$

We have:

$$\sum_{i=1}^{n} w_i \leqslant W \Leftrightarrow \sum_{i=1}^{n} s_i \leqslant t$$

$$\sum_{i=1}^{n} v_i \geqslant k \Leftrightarrow \sum_{i=1}^{n} s_i \geqslant t$$

That means:

$$\sum_{i=1}^{n} s_i = t$$

That is a Subset-Sum problem. So we proved that SUBSET-SUM $\leq_p$ KNAPSACK, that means KNAPSACK is NP-complete.

$\square$

Remark: Please include your .pdf, .tex files for uploading with standard file names.