Lab11-NP Reduction

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

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

∗ Name:Renyang Guan Student ID:519021911058 Email: guanrenyang@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

Proof.

A different view of the original problem: If we view the discord matrix as a adjacent matrix, the n ingredients play the role of vertices of a complete graph K_n while the *discord* between ingredient u and v is the weight of the corresponding edge.

Base on the description above, the Experimental Cuisine is retailed as: *Given a weighted complete graph K_n with n node and C_n^2 edges, decide whether there exists a clique K_{sub} of at least k vertices that has total weight of it $\leq p$.*

Construction for reduction: Our reduction will be based on this second view of the 3-SAT instance:

1. Construct a complete graph $G = (V, E)$ grouped into k K_3 of which the weight of each edge is $p + 1$. That is, for $i = 1, 2, \cdots, k$, we construct k K_3^i composed of v_{i1}, v_{i2}, v_{i3} to represent each clause C_i, and v_{ij} to represent the jth term in clause C_i.
2. Then we add edges to G to transform it into K_{3k}. The rules of adding edges is defined as follows: *(1) If v and u represent opposite terms($v = X$ and $u = \bar{X}$)*, the weight of the edge $w_{uv} = p$. *(2) Otherwise, $w_{uv} = 0$.*

The* Experimental Cuisine *problem in our construction: The weight of the particular edge between vertices which belong to the same clause or who are conflict is $p + 1$, so the corresponding two vertices won't be included in the subgraph K_k with total weight $\leq p$.

Equivalent proposition of* 3-SAT $\leq_p$ Experimental Cuisine*: The original 3-SAT instance is satisfiable using polynomial times of standard operations plus polynomial times of

the black box for deciding if the graph K_{3k} we have constructed has an clique of size at least k with total weight $\leq p$.

Proof: If the 3-SAT instance is satisfiable, then each K_3^i in our graph contains at least one node whose label evaluates to 1, the set of which is S. The elements of S consists of the object subgraph K_k, whose total weight $= 0$, because each two vertices in S is neither in the same clause nor conflict. Additionally, the construction of K_{3k} takes $O(k)$ time by traversing the conjunctive paradigm.

□

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)

Proof.

First, let's show that MCS is a NP problem.

Certifier: We could encode as string s, where $|s| = 2b$. The set of s is $X = \{s | s_0 \sim s_b$ is the permutation of b elements of V_1 and $s_{b+1} \sim s_{2b}$ is the permutation of b elements of $V_2\}$.

Certificate: Compare each pair of nodes of $< s_{1+i}, s_{b+i} >$, where $1 \leq i \leq b$, to see if each node in the pair has the same set of neighbours as the other, which could determine whether the two subgraphs are identical.

Since INDEPENDENT-SET problem is a NP-complete problem, **we just need to prove that INDEPENDENT-SET$\leq_P$MCS.** The construction is shown below:

Given a graph $G = (V, E)$ and we want to get an independent set of G whose size $\geq b$. Supposing G' to be the complement graph of G, the set of vertices S is an independent set of G if and only if its complement graph G' consists a clique of size b in G'.

Based on the claim, the construction is very simple: *Let $G_1 = G'$ and $G_2 = K_b$.* Graph G has an independent set of size $\geq b$ if and only if its complement graph G' has a clique of size $\geq b$. The latter problem could be solved by the black box solving MCS, of which the input is G_1 and G_2.

Proof:

Sufficiency: If G has an independent set S with $|S| \geq b$, which means there doesn't exist an edge between arbitrary pair nodes u and v. As a result, there is an edge existing between u and v in G'. Because of the arbitrariness of u and v, elements of S consist a subgraph of K_b in G'.

Necessity: If G' has a clique of K_b, the set of whose vertices is S, the set S is an independent set of G. Because all the incoming and outgoing edges of arbitrary vertex u in S are included in K_b, there is no edge connecting u and any other nodes of S. As a result, S is an independent set of G. □

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)

Proof. First, that's show that the k-spinning tree is a NP problem:

Certifier: Permutations of nodes which start from the root of the k-spinning tree.

Certificate: Check each string to see if there is a node whose degree $> k$, to calculate the

number of edges, and to see if the graph is connected. To prove the k-spinning tree is NP-complete, we just need to prove **HAMILTONIAN-PATH** $\leq_P$ **K-SPINNING-TREE**, which means we could solve the k-spinning tree problem using the blackbox used to solve the HAMILTONIAN-CYCLE problem.
Construction: Given a graph $G = (V, E)$, we claim that a HAMILTONIAN path exists in G if and only if there is a 2-spinning tree in G.
Proof:
Sufficiency: The HAMILTONIAN path tours each vertex only once, so there is no cycle included in the path. Furthermore, each vertex except for the starting vertex and ending vertex will have exactly an in-coming edge and an out-going edge while the two exceptions have only one edge. Finally, the HAMILTONIAN path traverses all the vertices in G. Considering the two properties, the HAMILTONIAN path is a 2-spinning tree.
Necessity: The 2-spinning tree has $n - 2$ vertices with degree $= 2$ and two vertices with degree $= 1$, so it is indeed a path of G. Since the path traverses each vertex once, it is actually a HAMILTONIAN path. □

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.

   Proof. First, let's show that the KNAPSACK-PROBLEM is NP
   Certifier: A collection X of set S_i, where S_i denotes any subset of the set of objects and $|X| = 2^n$, is the collection of all possible schemes.
   Certificate: Check each set S_i in the collection X to see whether total weights exceeds the constraint and whether the total value $V \geq k$. The answer of the previous question is YES if and only if the answer of the decision problem is YES.
   Since the SUBSET-SUM problem is NP-complete, to prove the KNAPSACK-PROBLEM is NP-complete, we just need to prove that **SUBSET-SUM**$\leq_P$**KNAPSACK-PROBLEM**.
   Construction: Assume that we have a black box for solving the KNAPSACK-PROBLEM. In the KNAPSACK-PROBLEM problem given in the problem, we set $W = \infty$(or to say that W is greater than the total weights of individual objects). Then the KNAPSACK-PROBLEM with $W = +\infty$ is transformed into SUBSET-SUM PROBLEM, if we assume the n objects are the n elements in the set and we want a subset whose values adds up to exactly k. Let's claim that the answer of the decision problem SUBSET-SUM is YES if and only if the answer of the KNAPSACK-PROBLEM is YES. ***Proof:***
   Sufficiency: Assign the value of each object to the corresponding element of the set. Since there is no constraint of weight, the answer of the KNAPSACK-PROBLEM is the expected answer of SUBSET-SUM problem.
   NECESSITY: Similar to the proof of sufficiency, assign the value of each element to the value of each object. The conclusion holds, too.

   □

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

Reference