Computational Complexity – Example Solution

Exercise 1. Consider the following decision problem TOP3. In this problem, the input consists of a tuple (A, T, u), where $A = \{a_1, \ldots, a_m\}$ is a finite set, $T = (t_1, \ldots, t_k)$ is a sequence of subsets $t_i \subseteq A$ of size exactly 3, and $u \in \mathbb{N}$ is a natural number. Intuitively, each such tuple (A, T, u) describes a collective choice problem, where we are to select a subset of u alternatives from the ones in A, and where k individuals gave their preferences in the form of submitting a top three. The problem is to decide whether there exists a subset $C \subseteq A$ such that $|C| = u$ and such that $C \cap t_i \neq \emptyset$ for each $1 \leq i \leq k$. In other words, the answer is yes if and only if we can choose u alternatives from A such that we pick at least one alternative from everyone's top three.

Prove that the problem TOP3 is NP-complete.

Proof. Firstly, we argue that the problem TOP3 is in NP, by describing a nondeterministic polynomial-time algorithm that solves the problem. The algorithm takes as input (A, T, u) for the problem, and works as follows. It first uses nondeterminism to guess a subset $C \subseteq A$ with $|C| = u$. Because A is part of the input, selecting a subset C can be done nondeterministically within a polynomial number of steps, and checking that $|C| = u$ can also be done in polynomial time (by counting the number of elements in C). Then, the algorithm iterates over all t_i in T and verifies that $t_i \cap C \neq \emptyset$. There are a polynomial number of t_i's, and for each, this check can be done in polynomial time (by looping over both t_i and C and checking for the existence of an element in both). The algorithm returns "yes" if and only if all these checks pass—in other words, it returns "no" if at least one check fails. Then, there exists a subset $C \subseteq A$ such that $|C| = u$ and such that $C \cap t_i \neq \emptyset$ for each $1 \leq i \leq k$ if and only if the algorithm returns "yes," so the algorithm correctly decides the problem. Moreover, the algorithm runs in polynomial time, because all the subsequent subroutines run in polynomial time.

Next, we argue that TOP3 is NP-hard, by giving a polynomial-time reduction from the NP-complete problem X3C (*Exact Cover by 3-Sets*). For the problem X3C, the inputs consist of a tuple $(X, \mathcal{C})$, where X is a finite set with $|X| = 3q$ for some $q \in \mathbb{N}$, and where $\mathcal{C} = \{C_1, \ldots, C_k\}$ is a set of sets $C_i \subseteq X$ where $|C_i| = 3$ for each $1 \leq i \leq k$. The question is whether there exists some $\mathcal{C}' \subseteq \mathcal{C}$ such that $|\mathcal{C}'| = q$ and $\bigcup \mathcal{C}' = X$. In other words, whether one can select q subsets among those in $\mathcal{C}$ that cover all elements of X.

We describe our reduction from X3C to TOP3 by describing the instance of TOP3 that is mapped to for an arbitrary instance of X3C. Let $(X, \mathcal{C})$ be an arbitrary instance of X3C, where $X = \{x_1, \ldots, x_{3q}\}$ and where $\mathcal{C} = \{C_1, \ldots, C_k\}$. Without loss of generality, we may assume that each element $x_j \in X$ appears in exactly three sets in $\mathcal{C}$ [1, Theorem A.1].

We construct an instance (A, T, u) of TOP3 as follows. We let $A = \{a_1, \ldots, a_k\}$, i.e., we introduce one alternative a_i for each $C_i \in \mathcal{C}$. We let $u = q$, i.e., the number of alternatives to select is the number of sets to use in an exact cover of X. Finally, we construct the tuple $T = (t_1, \ldots, t_{3q})$ as follows. For each $1 \leq j \leq 3q$, let k_1, k_2 and k_3 be the three (distinct) indices such that $x_j \in C_{k_1}$, $x_j \in C_{k_2}$, and $x_j \in C_{k_3}$. We let $t_j = \{a_{k_1}, a_{k_2}, a_{k_3}\}$. Each element of (A, T, u) can be constructed in polynomial time from $(X, \mathcal{C})$—in other words, this reduction runs in polynomial time. Next, we will argue that this reduction is correct.

Suppose that there exists some $\mathcal{C}' \subseteq \mathcal{C}$ such that $|\mathcal{C}'| = q$ and $\bigcup \mathcal{C}' = X$. We will argue that there exists a set $C \subseteq A$ with $|C| = u$ such that $C \cap t_i \neq \emptyset$ for each $1 \leq i \leq 3q$. We let $C = \{a_j \mid C_j \in \mathcal{C}'\}$. Then $|C| = |\mathcal{C}'| = q = u$. Take an arbitrary $1 \leq i \leq 3q$. Because $\bigcup \mathcal{C}' = X$, we know that $x_i \in C_j$ for some $C_j \in \mathcal{C}'$. Then $a_j \in t_i$ and $a_j \in C$, so $C \cap t_i \neq \emptyset$. Because t_i was arbitrary, we can conclude that $C \cap t_i \neq \emptyset$ for each $1 \leq i \leq 3q$.

Conversely, suppose that there exists a set $C \subseteq A$ with $|C| = u$ such that $C \cap t_i \neq \emptyset$ for each $1 \leq i \leq 3q$. We will argue that there exists some $\mathcal{C}' \subseteq \mathcal{C}$ such that $|\mathcal{C}'| = q$ and $\bigcup \mathcal{C}' = X$. We let $\mathcal{C}' = \{C_j \mid a_j \in C\}$. Then, $|\mathcal{C}'| = |C| = u = q$. Take an arbitrary $x_i \in X$. Because $t_i \cap C \neq \emptyset$, we know that there is some $a_j \in t_i \cap C$, and thus that $C_j \in \mathcal{C}'$ and $x_i \in C_j$. Therefore, because $x_i \in X$ was arbitrary, we know that $\bigcup \mathcal{C}' = X$. □

References

[1] Teofilo F. Gonzalez. Clustering to minimise the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.