CSCI3160: Special Exercise Set 12

Prepared by Yufei Tao

Problem 1 (Textbook Exercise 35.3-1). Consider $\mathcal{S} = \{\texttt{arid}, \texttt{dash}, \texttt{drain}, \texttt{heard}, \texttt{lost}, \texttt{nose}, \texttt{shun}, \texttt{slate}, \texttt{snare}, \texttt{thread}\}$. Treat each word in $\mathcal{S}$ as a set of letters. Run the set-cover algorithm discussed in the lecture and describe its output.

Problem 2. Recall that our set-cover algorithm in each iteration picks a set with the largest *benefit*. Prove: if we lay out the sets in the order they are picked, their benefits are non-ascending.

Problem 3*. Give a counterexample input to show that the approximation ratio of our set-cover algorithm cannot be bounded by 2.

Problem 4. As mentioned in the lecture, the set cover problem is NP-hard. This means that it cannot be solved in polynomial time unless P = NP. Now consider the following decision version of the set cover problem. As before, let $\mathcal{S}$ be a collection of sets and define the universe $U = \bigcup_{S \in \mathcal{S}} S$. But now we are also given an integer k. The goal is to decide whether there is a set cover $\mathcal{C} \subseteq \mathcal{S}$ such that $|\mathcal{C}| = k$ and return such a $\mathcal{C}$ if the answer is yes. Show that, unless P = NP, this decision version does not admit any polynomial-time algorithm.

Problem 5. Let $\boldsymbol{M}$ be an $n \times m$ matrix where each cell is either 0 or 1. It is guaranteed that every row of $\boldsymbol{M}$ has at least one 1. A set S of columns is a *column cover* if every row of $\boldsymbol{M}$ has a 1 in at least one column of S. If OPT is the minimize size of all column covers, describe a poly(n, m)-time algorithm (i.e., polynomial in n and m) that finds a column cover of size $O(\text{OPT} \cdot \log n)$.