Algorithm Design and Analysis

Assignment 3

Deadline: Nov 25, 2024

1. (35 points) In the class, we learned Kruskal's algorithm to find a minimum spanning tree (MST). The strategy is simple and intuitive: pick the best legal edge in each step. The philosophy here is that local optimal choices will yield a global optimal. In this problem, we will try to understand to what extent this simple strategy works. To this end, we study a more abstract algorithmic problem of which MST is a special case.

   Consider a pair $M = (U, \mathcal{I})$ where U is a finite set and $\mathcal{I} \subseteq \{0,1\}^U$ is a collection of subsets of U. We say M is a *matroid* if it satisfies

   - (**hereditary property**) $\mathcal{I}$ is nonempty and for every $A \in \mathcal{I}$ and $B \subseteq A$, it holds that $B \in \mathcal{I}$.
   - (**exchange property**) For any $A, B \in \mathcal{I}$ with $|A| < |B|$, there exists some $x \in B \setminus A$ such that $A \cup \{x\} \in \mathcal{I}$.

   Each set $A \in \mathcal{I}$ is called an *independent set*.

   (a) (7 points) Let $M = (U, \mathcal{I})$ be a matroid. Prove that maximal independent sets are of the same size. (A set $A \in \mathcal{I}$ is *maximal* if there is *no* $B \in \mathcal{I}$ such that $A \subsetneq B$.)

   (b) (7 points) Let $G = (V, E)$ be a simple undirected graph. Let $M = (E, \mathcal{S})$ where $\mathcal{S} = \{F \subseteq E \mid F \text{ does not contain a cycle}\}$. Prove that M is a matroid. What are the maximal sets of this matroid?

   (c) (7 points) Let $M = (U, \mathcal{I})$ be a matroid. We associate each element $x \in U$ with a nonnegative weight $w(x)$. For every set of elements $S \subseteq U$, the weight of S is defined as $w(S) = \sum_{x \in S} w(x)$. Now we want to find a maximal independent set with maximum weight. Consider the following greedy algorithm.

Algorithm 1 Find a maximal independent set with maximum weight

Input: A matroid $M = (U, \mathcal{I})$ and a weight function $w : U \to \mathbb{R}_{\geq 0}$.

Output: A maximal independent set $S \in \mathcal{I}$ with maximum $w(S)$.

```
S ← ∅
Sort U into decreasing order by weight w
for x ∈ U in decreasing order of w:
    if S ∪ {x} ∈ I:
        S ← S ∪ {x}
    endif
endfor
return S
```

Now we consider the first element x the algorithm added to S. Prove that there must be a maximal independent set $S' \in \mathcal{I}$ with maximum weight containing x.

(d) (7 points) Prove that the greedy algorithm returns a maximal independent set with maximum weight. (Hint: Can you see that Algorithm 1 is just a generalization of Kruskal's algorithm?)

(e) (7 points) Let $U \subseteq \mathbb{R}^n$ be a finite collection of n-dimensional vectors. Assume $m = |U|$ and we associate each vector $\mathbf{x} \in U$ a positive weight $w(\mathbf{x})$. For any set of vectors $S \subseteq U$, the weight of S is defined as $w(S) = \sum_{\mathbf{x} \in S} w(\mathbf{x})$. Design an efficient algorithm to find a set of vectors $S \subseteq U$ with maximum weight and all vectors in S are linearly independent.

2. (30 points) Given a constant $k \in \mathbb{Z}^+$, we say that a vertex u in an undirected graph *covers* a vertex v if the distance between u and v is at most k. In particular, a vertex u covers all those vertices that are within distance k from u, including u itself. Given an undirected *tree* $G = (V, E)$ and the parameter k, consider the problem of finding a minimum-size subset of vertices that covers all the vertices in G. Design a polynomial time algorithm for this problem. Prove the correctness of your algorithm and analyze its running time.

   You will receive 20 points if you can solve the problem for $k = 1$.

3. (35 points) Given a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$, let $\sigma(S)$ be the number of vertices that are reachable from S. Consider the maximum reachability problem: given a graph $G = (V, E)$ and a positive integer $k \in \mathbb{Z}^+$ as inputs, find $S \subseteq V$ with $|S| \leq k$ that maximizes $\sigma(S)$.

   (a) (10 points) Design a polynomial time algorithm for this problem if the input graph G is undirected.

   (b) (15 points) Design a polynomial time $(1 - 1/e)$-approximation algorithm for this problem when the input graph can be directed. You need to prove that your algorithm indeed achieves a $(1 - 1/e)$-approximation.

   (c) (10 points) We have mentioned in the class that the max-k-coverage problem is NP-hard and we do not believe there is a polynomial time algorithm for it. Prove that the maximum reachability problem with the directed graph setting in part (b) is also NP-hard by proving that the max-k-coverage problem can be viewed as a special case of the maximum reachability problem.

4. How long does it take you to finish the assignment (including thinking and discussion)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Please write down their names here.