Exercise Set X, Advanced Algorithms 2022

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. Solve as many problems as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

These problems are taken from various sources at EPFL and on the Internet, too numerous to cite individually.

1 Consider a submodular function $f : 2^N \to \mathbb{R}$ over the ground set $N = \{1, 2, 3, 4\}$. What is the value of the Lovàsz extension $\hat{f}(0.75, 0.3, 0.2, 0.3)$ as a function of f?

Solution: By the definition of $\hat{f}$ (see Lecture Notes 19) we get

$$\hat{f}(0.75, 0.3, 0.2, 0.3, 0) = 0.25f(\emptyset) + 0.45f(\{1\}) + 0.1f(\{1, 2, 4\}) + 0.2f(\{1, 2, 3, 4\})\,.$$

2 **Online ad allocation.** Alice and Bob started companies selling hand sanitizer, and are now advertising their products online to potential customers $c_1, c_2, \ldots, c_n$, where $c_i \in \mathcal{C}$ for all $i = 1, \ldots, n$. When a customer c_i arrives, they can be shown advertisement for either Alice's or Bob's hand sanitizer – we say that the customer is *allocated* to either Alice or Bob in that case. If S_1 and S_2 are the sets of customers allocated to Alice and Bob respectively at the end of the sequence, Alice will pay $v_1(S_1)$ Francs to the online advertisement engine and Bob will pay $v_2(S_2)$. Here $v_1 : 2^{\mathcal{C}} \to \mathbb{R}_+$ and $v_2 : 2^{\mathcal{C}} \to \mathbb{R}_+$ are non-negative monotone submodular functions. The goal in the online ad allocation problem is to design an allocation rule that maximizes $v_1(S_1) + v_2(S_2)$. In this problem you will analyze the competitive ratio of the greedy algorithm, stated below:

Algorithm 1 Greedy algorithm for online ad allocation

```
S_1 ← ∅, S_2 ← ∅
for i = 1, ..., n do
    if v_1(c_i|S_1) ≥ v_2(c_i|S_2) then
        S_1 ← S_1 ∪ {c_i}                 ▷ Allocate i-th customer c_i to Alice
    else
        S_2 ← S_2 ∪ {c_i}                 ▷ Allocate i-th customer c_i to Bob
    end if
end for
```

You will prove that greedy achieves a competitive ratio of $1/2$ by induction on n, the number of customers. We now describe the inductive step. Suppose that the first customer is allocated to Alice (the other case is analogous). Define for $S \subseteq \mathcal{C}$ the functions $v_1'(S) = v_1(S|\{c_1\}) = v_1(S \cup \{c_1\}) - v_1(S)$ and $v_2'(S) = v_2(S)$, and let $p = v_1(\{c_1\})$. Let $\mathcal{I} = (v_1, v_2; c_1, \ldots, c_n)$ denote the input instance of the ad allocation problem, and let $\mathcal{I}' = (v_1', v_2'; c_2, \ldots, c_n)$ denote the instance $\mathcal{I}$ with the first customer removed and the functions v_1, v_2 replaced with v_1', v_2'. Let ALG denote the value achieved by greedy on $\mathcal{I}$, and let OPT denote the optimal offline solution on $\mathcal{I}$. Similarly, let ALG' denote the value achieved by greedy on $\mathcal{I}'$, and let OPT' denote the optimal offline solution on $\mathcal{I}'$.

2a Prove that $v'_j, j = 1, 2$, are non-negative monotone submodular functions.

2b Prove that $OPT \leq OPT' + 2p$.

2c Show how to complete the proof using **(b)**.

Solution:

2a Note that

$$\begin{aligned} v_1'(e \mid X) &= v_1'(\{e\} \cup X) - v_1'(X) \\ &= v_1(\{e\} \cup X \mid \{c_1\}) - v_1(X \mid \{c_1\}) \\ &= v_1(\{e, c_1\} \cup X) - v_1(\{c_1\}) - (v_1(\{c_1\} \cup X) - v_1(\{c_1\})) \\ &= v_1(\{e, c_1\} \cup X) - v_1(\{c_1\} \cup X) \\ &= v_1(e \mid \{c_1\} \cup X). \end{aligned}$$

Let X, Y be be such that $X \subseteq Y$. Note that for any element c, $X \cup \{c\} \subseteq Y \cup \{c\}$. Hence, we have

$$v_1'(e \mid X) = v_1(e \mid \{c_1\} \cup X) \geq v_1(e \mid \{c_1\} \cup Y) = v_1'(e \mid Y).$$

Thus v_1' satisfies diminishing returns property, and hence submodular. The non-negativity and monotonocity of v_1' follows from the monotonocity of v_1.

2b Let $O = (O_1, O_2)$ be the optimal allocation for I, so that $OPT = v_1(O_1) + v_2(O_2)$ and $OPT' = v_1'(O_1) + v_2'(O_2)$. Also note that $(O_1 \setminus \{c_1\}, O_2 \setminus \{c_1\})$ is a feasible solution on I' and moreover, since we assume c_1 is allocated to Alice, $v_1(\{c_1\}) \geq v_2(\{c_1\})$. Thus we have

$$\begin{aligned} OPT' &\geq v_1'(O_1 \setminus \{c_1\}) + v_2'(O_2 \setminus \{c_1\}) \\ &= v_1(O_1) - v_1(\{c_1\}) + v_2(O_2 \setminus \{c_1\}). \end{aligned}$$

Due to submodularity, we have $v_2(O_2 \setminus \{c_1\}) + v_2(\{c_1\}) \geq v_2(O_2) + v_2(\emptyset)$, and due to non-negativity, this yields $v_2(O_2 \setminus \{c_1\}) \geq v_2(O_2) - v_2(\{c_1\})$. Thus

$$OPT' \geq v_1(O_1) + v_2(O_2) - v_1(\{c_1\}) - v_2(\{c_1\}) \geq OPT - 2v_1(\{c_1\}) = OPT - 2p.$$

2c For $n = 1$, we get the optimal solution. Suppose $n > 1$. By the inductive hypothesis, $ALG' \geq OPT'/2$. Thus we have

$$ALG = p + ALG' \geq p + OPT'/2 = (2p + OPT')/2 \geq OPT/2.$$

3 Hypergraph cuts. Let $G = (V, E)$ be a hypergraph with vertex set V and hyperedge set E (every hyperedge $e \in E$ is a subset of V; see Fig. 1 for an illustration). For $S \subseteq V$ the set of hyperedges crossing the cut $(S, V \setminus S)$ is defined as

$$E(S, V \setminus S) = \{e \in E : e \cap S \neq \emptyset \text{ and } e \cap V \setminus S \neq \emptyset\},$$

and the size of the cut $(S, V \setminus S)$ as $|E(S, V \setminus S)|$.

3a Give an algorithm that finds the size of the minimum cut in a given hypergraph G, i.e. outputs

$$\min_{S \subset V, S \neq \emptyset} |E(S, V \setminus S)|.$$

For example, the size of the minimum cut in the hypergraph G in Fig. 1 is 1. There are two minimum cuts: $(\{a\}, \{b, c, d, f, g\})$ and $(\{a, b, c\}, \{d, f, g\})$.

Your algorithm should run in time polynomial in the number of vertices and hyperedges in G.

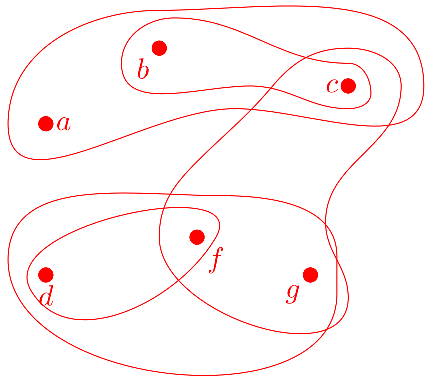

Figure 1. A hypergraph $G = (V, E)$ with $V = \{a, b, c, d, f, g\}$ and hyperedge set $E = \{e_1, e_2, e_3, e_4, e_5\}$, where $e_1 = \{a, b, c\}, e_2 = \{b, c\}, e_3 = \{d, f\}, e_4 = \{c, f, g\}$ and $e_5 = \{d, f, g\}$.

Solution: Let $f(S) = E(S, V \setminus S)$. We first observe that f is submodular. We verify the diminishing returns property. For every $S \subseteq T \subseteq V$ and $u \in V$ we have

$$\begin{aligned} f(u|S) &= f(S \cup \{u\}) - f(S) \\ &= |\{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}| \qquad (1) \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \end{aligned}$$

Since $S \subseteq T$, we have

$$\{e \in E : u \in e, e \cap T = \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) \neq \emptyset\} \subseteq \{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}$$

and

$$\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\} \subseteq \{e \in E : u \in e, e \cap T \neq \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) = \emptyset\}.$$

We thus get by (1)

$$\begin{aligned} f(u|S) &= |\{e \in E : u \in e, e \cap S = \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) \neq \emptyset\}| \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \\ &\leq |\{e \in E : u \in e, e \cap T = \emptyset \text{ and } e \cap V \setminus (T \cup \{u\}) \neq \emptyset\}| \\ &- |\{e \in E : u \in e, e \cap S \neq \emptyset \text{ and } e \cap V \setminus (S \cup \{u\}) = \emptyset\}| \\ &\geq f(u|T) \end{aligned}$$

and therefore $f(S)$ is submodular.

Now we would like to use the fact that unconstrained submodular function minimization can be done in polynomial time. However, we do have constraints $S \neq \emptyset$ and $S \neq V$. Thus, instead we pick an element $u \in V$ and then for every other $v \in V \setminus \{u\}$ define

$$g(S) = f(S \cup \{u\})$$

and find

$$\min_{S \subseteq V \setminus \{u,v\}} g(S).$$

Note that $g(S)$ is submodular, since for every $w \in V \setminus \{u, v\}$ and $S \subseteq T \subseteq V$ one has

$$g(w|S) = f(w|S \cup \{u\}) \geq f(w|T \cup \{u\}) = g(w|T)$$

by the diminishing returns property of f.

3b Give a randomized polynomial time algorithm that outputs, given a hypergraph G where every hyperedge contains three vertices, a cut $(S, V \setminus S)$ such that

$$\mathbb{E}[|E(S, V \setminus S)|] \geq (3/4)OPT, \tag{$*$}$$

where

$$OPT = \max_{S \subseteq V} |E(S, V \setminus S)|.$$

Note that unlike **3a**, here we are interested in the **maximum** cut. Your algorithm should run in time polynomial in the number of vertices and hyperedges in G, and you should prove that the expected size of the cut that it outputs satisfies $(*)$.

Solution: Let S include every vertex $u \in V$ independently with probability $1/2$. Then

$$\mathbb{E}[|E(S, V \setminus S)|] = \sum_{e \in E} \Pr[e \in E(S, V \setminus S)].$$

Let $e = \{u, v, w\}$, and suppose that $u \in S$ (this is without loss of generality, as $E(S, V \setminus S) = E(V \setminus S, S)$). Then

$$\Pr[e \in E(S, V \setminus S)] = 1 - \Pr[v, w \in S] = 3/4.$$

Thus, a random cut cuts at least $3/4$ of the hyperedges in expectation. By Markov's inequality applied to $|E| - |E(S, V \setminus S)|$ we have

$$\Pr[|E| - |E(S, V \setminus S)| > 1/4(1 + 1/n)|E|] \leq 1/(1 + 1/n) = 1 - O(1/n).$$