Exercise 5-6. Let $M = (E, \mathcal{I})$ be a matroid. Let $k \in \mathbb{N}$ and define

$$\mathcal{I}_k = \{X \in \mathcal{I} : |X| \leq k\}.$$

Show that $M_k = (E, \mathcal{I}_k)$ is also a matroid. This is known as a truncated matroid.

Exercise 5-7. A family $\mathcal{F}$ of sets is said to be *laminar* if, for any two sets $A, B \in \mathcal{F}$, we have that either (i) $A \subseteq B$, or (ii) $B \subseteq A$ or (iii) $A \cap B = \emptyset$. Suppose that we have a laminar family $\mathcal{F}$ of subsets of E and an integer $k(A)$ for every set $A \in \mathcal{F}$. Show that $(E, \mathcal{I})$ defines a matroid (a *laminar* matroid) where:

$$\mathcal{I} = \{X \subseteq E : |X \cap A| \leq k(A) \text{ for all } A \in \mathcal{F}\}.$$

5.2 Matroid Optimization

Given a matroid $M = (E, \mathcal{I})$ and a cost function $c : E \to \mathbb{R}$, we are interested in finding an independent set S of M of maximum total cost $c(S) = \sum_{e \in S} c(e)$. This is a fundamental problem.

If all $c(e) \geq 0$, the problem is equivalent to finding a maximum cost *base* in the matroid. If $c(e) < 0$ for some element e then, because of (I_1), e will not be contained in any optimum solution, and thus we could eliminate such an element from the ground set. In the special case of a graphic matroid $M(G)$ defined on a connected graph G, the problem is thus equivalent to the maximum spanning tree problem which can be solved by a simple greedy algorithm. This is actually the case for any matroid and this is the topic of this section.

The greedy algorithm we describe actually returns, for every k, a set S_k which maximizes $c(S)$ over all independent sets of size k. The overall optimum can thus simply be obtained by outputting the best of these. The greedy algorithm is the following:

- ▷ Sort the elements (and renumber them) such that $c(e_1) \geq c(e_2) \geq \cdots \geq c(e_{|E|})$
- ▷ $S_0 = \emptyset$, k=0
- ▷ For $j = 1$ to $|E|$
    - ▷ if $S_k + e_j \in \mathcal{I}$ then
        - ▷ $k \leftarrow k + 1$
        - ▷ $S_k \leftarrow S_{k-1} + e_j$
        - ▷ $s_k \leftarrow e_j$
- ▷ Output $S_1, S_2, \cdots, S_k$

Theorem 5.2 *For any matroid $M = (E, \mathcal{I})$, the greedy algorithm above finds, for every k, an independent set S_k of maximum cost among all independent sets of size k.*

Proof: Suppose not. Let $S_k = \{s_1, s_2, \cdots, s_k\}$ with $c(s_1) \geq c(s_2) \geq \cdots \geq c(s_k)$, and suppose T_k has greater cost $(c(T_k) > c(S_k))$ where $T_k = \{t_1, t_2, \cdots, t_k\}$ with $c(t_1) \geq c(t_2) \geq \cdots \geq c(t_k)$. Let p be the first index such that $c(t_p) > c(s_p)$. Let $A = \{t_1, t_2, \cdots, t_p\}$ and $B = \{s_1, s_2, \cdots, s_{p-1}\}$. Since $|A| > |B|$, there exists $t_i \notin B$ such that $B + t_i \in \mathcal{I}$. Since

$c(t_i) \geq c(t_p) > c(s_p)$, t_i should have been selected when it was considered. To be more precise and detailed, when t_i was considered, the greedy algorithm checked whether t_i could be added to the current set at the time, say S. But since $S \subseteq B$, adding t_i to S should have resulted in an independent set (by (I_1)) since its addition to B results in an independent set. This gives the contradiction and completes the proof. △

Observe that, as long as $c(s_k) \geq 0$, we have that $c(S_k) \geq c(S_{k-1})$. Therefore, to find a maximum cost set over all independent sets, we can simply replace the loop

▷ For $j = 1$ to $|E|$

by

▷ For $j = 1$ to q

where q is such that $c(e_q) \geq 0 > c(e_{q+1})$, and output the last S_k.

For the maximum cost spanning tree problem, the greedy algorithm reduces to Kruskal's algorithm which considers the edges in non-increasing cost and add an edge to the previously selected edges if it does not form a cycle.

One can show that the greedy algorithm actually characterizes matroids. If M is an independence system, i.e. it satisfies (I_1), then M is a matroid if and only if the greedy algorithm finds a maximum cost set of size k for every k and every cost function.

Exercise 5-8. We are given n jobs that each take one unit of processing time. All jobs are available at time 0, and job j has a profit of c_j and a deadline d_j. The profit for job j will only be earned if the job completes by time d_j. The problem is to find an ordering of the jobs that maximizes the total profit. First, prove that if a subset of the jobs can be completed on time, then they can also be completed on time if they are scheduled in the order of their deadlines. Now, let $E(M) = \{1, 2, \cdots, n\}$ and let $\mathcal{I}(M) = \{J \subseteq E(M) : J$ can be completed on time $\}$. Prove that M is a matroid and describe how to find an optimal ordering for the jobs.

5.3 Rank Function of a Matroid

Similarly to the notion of rank for matrices, one can define a rank function for any matroid. The rank function of M, denoted by either $r(\cdot)$ or $r_M(\cdot)$, is defined by:

$$r_M : 2^E \to \mathbb{N} : r_M(X) = \max\{|Y| : Y \subseteq X, Y \in \mathcal{I}\}.$$

Here are a few specific rank functions:

- For a linear matroid, the rank of X is precisely the rank in the linear algebra sense of the matrix A_X corresponding to the columns of A in X.
- For a partition matroid $M = (E, \mathcal{I})$ where

$$\mathcal{I} = \{X \subseteq E : |X \cap E_i| \leq k_i \text{ for } i = 1, \cdots, l\}$$

(the E_i's forming a partition of E) its rank function is given by:

$$r(X) = \sum_{i=1}^{l} \min(|E_i \cap X|, k_i).$$

- For a graphic matroid $M(G)$ defined on graph $G = (V, E)$, the rank function is equal to:

$$r_{M(G)}(F) = n - \kappa(V, F),$$

where $n = |V|$ and $\kappa(V, F)$ denotes the number of connected components (including isolated vertices) of the graph with edges F.

The rank function of any matroid $M = (E, \mathcal{I})$ has the following properties:

(R_1) $0 \leq r(X) \leq |X|$ and is integer valued for all $X \subseteq E$

(R_2) $X \subseteq Y \Rightarrow r(X) \leq r(Y)$,

(R_3) $r(X) + r(Y) \geq r(X \cap Y) + r(X \cup Y)$.

The last property is called *submodularity* and is a key concept in combinatorial optimization. It is clear that, as defined, any rank function satisfies (R_1) and (R_2). Showing that the rank function satisfies submodularity needs a proof.

Lemma 5.3 *The rank function of any matroid is submodular.*

Proof: Consider any two sets $X, Y \subseteq E$. Let J be a maximal independent subset of $X \cap Y$; thus, $|J| = r(X \cap Y)$. By (I_2), J can be extended to a maximal (thus maximum) independent subset of X, call it J_X. We have that $J \subseteq J_X \subseteq X$ and $|J_X| = r(X)$. Furthermore, by maximality of J within $X \cap Y$, we know

$$J_X \setminus Y = J_X \setminus J. \tag{1}$$

Now extend J_X to a maximal independent set J_{XY} of $X \cup Y$. Thus, $|J_{XY}| = r(X \cup Y)$.

In order to be able to prove that

$$r(X) + r(Y) \geq r(X \cap Y) + r(X \cup Y)$$

or equivalently

$$|J_X| + r(Y) \geq |J| + |J_{XY}|,$$

we need to show that $r(Y) \geq |J| + |J_{XY}| - |J_X|$. Observe that $J_{XY} \cap Y$ is independent (by (I_1)) and a subset of Y, and thus $r(Y) \geq |J_{XY} \cap Y|$. Observe now that

$$J_{XY} \cap Y = J_{XY} \setminus (J_X \setminus Y) = J_{XY} \setminus (J_X \setminus J),$$

the first equality following from the fact that J_X is a maximal independent subset of X and the second equality by (1). Therefore,

$$r(Y) \geq |J_{XY} \cap Y| = |J_{XY} \setminus (J_X \setminus J)| = |J_{XY}| - |J_X| + |J|,$$

proving the lemma. $\triangle$